CMMI with Agile, Lean, Six Sigma, and Everything Else

NEWS AT SEI

Author

Mike Phillips

This library item is related to the following area(s) of work:

CMMI

Process Improvement

This article was originally published in News at SEI on: January 1, 2008

I repeatedly encounter those seeking the one solution that will solve the problems in their organization. Such a search is often commissioned by a boss who wants the single answer and a quick fix to the organization's problems. In this column, I try to describe how to relate some of these answers rather than trying to make any of them—even CMMI—a single solution.

The Choices

There are a wide range of improvement approaches that are often mentioned as the solution to problems confronting organizations that recognize a need to improve. For years, various standards and modelscaptured **principles** for process improvement, often called best practices. ISO 15288 and ISO 12207 are likely standards that are familiar to you if you are faced with complex, software intensive systems development. In addition to the collection of CMM and CMMI models, collections of other models such as the Project Management Institute's Organizational Project Management Maturity Model (OPM3) and Control Objectives for Information and related Technology (CoBIT) are also likely familiar to many of you.

Others interested in process improvement investigated the best practices in these and other models and standards and found ways to capture some of these principles into **methods** that accomplish a process, such as software system development. Examples of current software development methods include Agile methods, Scrum, and the SEI's Team Software Process (TSP) methodology. Those of you who are past the major

Please note that current and future CMMI research, training, and information has been transitioned to the CMMI Institute, a wholly-owned subsidiary of Carnegie Mellon University.

Related Links

News

SEI to Co-Sponsor 26th Annual IEEE Software Technology Conference

SEI Cosponsors Agile for Government Summit

Training

See more related courses »

Events

Team Software Process (TSP) Symposium 2014

Nov 3 - 6

development stage and are more concerned about maintaining information-technology capabilities are likely familiar with collections of methods for service related activities such as the Information Technology Infrastructure Library (ITIL).

To add another layer of complexity, there are also improvement **techniques** that apply in various domains and disciplines. Three current representative techniques are Lean, Six Sigma, and Theory of Constraints. One way to relate the principles, methods, and techniques discussed thus far is depicted below in Figure 1. I'll spend the remainder of this column discussing some of the relationships and synergies that makes the search for *the best one* a limiting strategy.

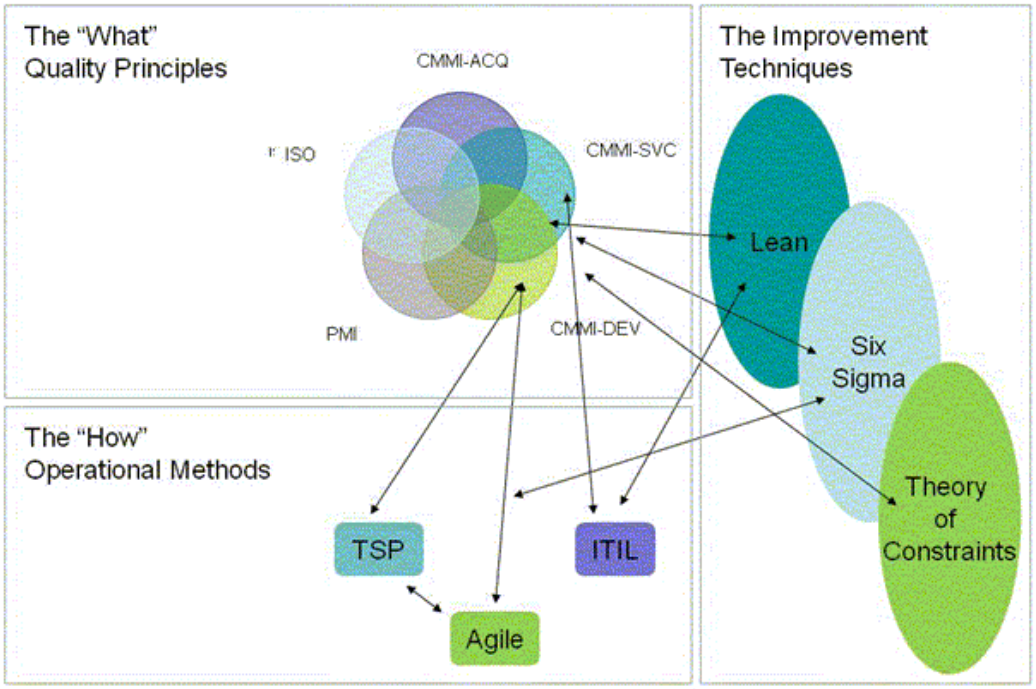

Figure 1: Principles, methods, and techniques

How do these improvement elements relate?

Figure 1 is an attempt to aid the discussion of these fundamentally different groupings of principles, methods, and techniques as interrelated improvement elements. Typically, models and standards provide high-level information about reasonable frameworks for a discipline, such as engineering or project management, or a major effort, such as development or acquisition.

If we use CMMI for Development (CMMI-DEV) as a familiar example, we suggest that many approaches to development can be taken within the broad framework represented by the content of CMMI-DEV. Methods, then, can perhaps best be thought of as ways actually to do the work—the "how" that the "what" of the models may not adequately address. The best example we have of an effective method linked to CMMI-DEV is TSP. Similarly, Agile methods also fit within a model structure such as CMMI-DEV. Like TSP, Agile methods seek to provide detailed approaches to development tasks.

The figure illustrates a vertical collection of techniques that represent a different direction or dimension of improvement elements. These techniques can investigate and solve problems in organizations. Three that have been called out by the Department of Defense for process improvement are Lean, Six Sigma, and Theory of Constraints. (Often the first two are combined into a single approach now described as Lean Six Sigma.) These techniques are depicted as vertical approaches because they allow disciplined ways both to improve how particular task elements are performed as well as to provide broad deployment approaches.

Apples and Oranges

When you are searching for a silver bullet to solve your organizational challenges, the truth is that there is no one-size-fits-all answer. The choices you need to make are actually quite different. Each example in the figure has proved its value in improving elements of various organizational challenges. However, some of these approaches fit better in specific organizational cultures or professional disciplines. I observed one large company in which one unit was actively engaged in CMMI-based process improvement. Its sister division, about two blocks away, was equally committed to its Six Sigma investment, with many black belts and green belts on staff. However, the two divisions could not seem to share their lessons learned. As many organizations have learned, Six Sigma is a powerful improvement technique within a CMMI framework. So this particular company had not yet seen the synergistic value of teaming a model and a technique.

How about choices between models and standards?

As much as I love our CMMI Product Suite, I recognize that there are other valuable views of the enterprise that also stimulate improvement and a strong commitment to quality. Many of you are familiar with the chart I use to suggest coupling ISO 9000 with CMMI [Figure 2]. And most of you are aware that the CMMI Product Development Team has encouraged mappings of CMMI with various other models and standards to facilitate using them together. Some of these mappings can be found at in the CMMI section of our site. The SEI has initiated a more extensive effort to investigate effective ways to perform multiple-model approaches to process improvement, and is seeking sponsors and researchers for this area of interest.

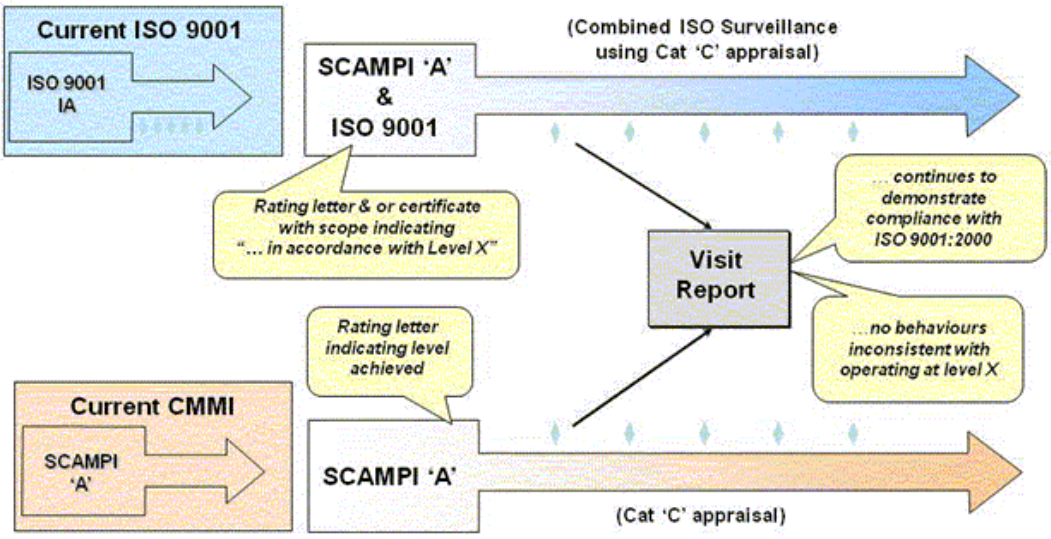

Figure 2: Coupling ISO 9000 with CMMI

Is it best to pick one model, one method, and one technique?

Most of us live in a multi-disciplinary world and we want to improve both our basic work practices and our organizational process discipline. Creating an effective mix of models and methods, with selected techniques to troubleshoot specific challenges, appears to have a high return on investment.

Summary

In the beginning, CMMI was developed as an effort to consolidate engineering best practices. Now we need to continue expanding the capabilities of our users without growing the size of the models by improving the interoperability of CMMI with its peers in the process improvement domain. While we feel that the CMMI Product Suite provides a valuable foundation for guiding process improvement choices for the enterprise, we feel that deploying effective operating methods in the domain and using improvement techniques under a CMMI umbrella can maximize the improvement of developing, deploying, and supporting our software intensive, knowledge-based systems.

About the Author

As the director of special projects at the Software Engineering Institute, Mike Phillips leads the Capability Maturity Model Integration (CMMI) project for the SEI. He was previously responsible for transition-enabling activities at the SEI. Prior to his retirement as a colonel from the Air Force, he managed the $36B development program for the B-2 in the B-2 SPO and commanded the 4950th Test Wing at Wright-Patterson AFB, Ohio. In addition to his bachelor's degree in astronautical engineering from the U.S. Air Force Academy, Phillips has master's degrees in nuclear engineering from Georgia Tech, in systems management from the University of Southern California, and in international affairs from Salve Regina College and the Naval War College.

For more information

Contact Us

info@sei.cmu.edu

412-268-5800

Resources for Getting Started with Software Product Lines

NEWS AT SEI

Author

Paul C. Clements

This library item is related to the following area(s) of work:

Software Product Lines

This article was originally published in News at SEI on: January 1, 2008

The purpose of the SEI's Product Line Practice initiative is to make software product line development and acquisition a low-risk, high-return proposition for all organizations. By *software product line*, we mean a set of software-intensive systems that share a common, managed set of features satisfying the specific needs of a particular market segment or mission and that are developed from a common set of core assets in a prescribed way [Clements & Northrop 2002]. Organizations of all sizes, government and commercial, are achieving order-of-magnitude improvements in cost, quality, and time to market using the software product line approach—improvements the scale of which the field has not seen since the advent of high-level programming languages decades ago.

The Product Line Practice initiative has a broad selection of products and services available to help organizations cross barriers and hurdles between their current state and the successful development of a software product line. However, there a number of things that an organization can do for itself, with the help of the SEI website for software product lines, www.sei.cmu.edu/productlines. On the "Getting Started" page you can find the following list of familiarization steps, each pointing directly to the most appropriate resource:

- Read case studies and experience reports of organizations pursuing software product lines.
- Get to know the SEI's Framework for Software Product Line Practice, especially the essential activities section, and the launching and institutionalizing practice

Related Links

Training

Software Product Lines - eLearning

Software Product Lines

See more related courses »

area.

- Learn the process of adopting software product line practices in your organization.
- Sign up for a course on software product lines.
- Start to build a business case for making the switch to software product lines.
- Get involved with the software product line community by participating in one of the conferences and workshops.
- Peruse the library assets in software product lines, including the SEI's publications
- Sign your organization up for an SEI Product Line Quick Look or an SEI Product Line Technical Probe to help gauge your organization's strengths and weaknesses with respect to software product line capability.
- Know about the software product line practice patterns.
- Learn the factory pattern and your organization's place in it.

Reading case studies and experience reports may be particularly helpful, and to this end there is a new resource. The SEI has recently compiled a list of 42 examples of software product lines that have been described in the open scientific literature. The catalog is available at www.sei.cmu.edu/productlines/plp_catalog.html. Each entry gives the name of the organization or organizations involved (except one case where the publication kept the name confidential), describes the product line, lists the best publications for more information about it, and summarizes the benefits experienced from the product line approach.

All 14 current member of the Software Product Line Hall of Fame are listed (www.sei.cmu.edu/productlines/plp_hof.html). The Hall of Fame is maintained by the yearly international Software Product Line Conference (SPLC) (www.splc.net) to reward excellence in software product lines and to provide exemplars for others to emulate. Hall of Fame members include examples from Hewlett Packard, Nokia, Boeing, Lucent Technologies and Bell Labs, Philips, Ericsson, General Motors, LSI Logic, and Bosch.

Also listed in the catalog are the comprehensive case studies performed by the SEI. These include in-depth examination of the successful software product lines at CelsiusTech, Salion, the U.S. National Reconnaissance Office, MarketMaker, and Cummins.

The catalog and the other getting-started steps listed on the web site are intended to help organizations become knowledgeable about software product lines and help them gain insight about where they stand relative to the organizations that have made the product-line journey successfully.

References

[Clements and Northrop 02]
Clements, P. & Northrop, L. *Software Product Lines: Practices and Patterns*. Boston, MA: Addison-Wesley, 2002.

About the Author

Paul Clements is a senior member of the technical staff at the SEI, where he has worked for 10 years leading or co-leading projects in software product line engineering and software architecture design, documentation, and analysis. Clements is the co-author of three practitioner-oriented books about software architecture: *Software Architecture in Practice* (1998; second edition, 2003), *Evaluating Software Architectures: Methods and Case Studies* (2001), and *Documenting Software Architectures: View and Beyond* (2002). He also co-wrote *Software Product Lines: Practices and Patterns* (2001), and was co-author and editor of *Constructing Superior Software* (1999). In addition, Clements has also written dozens of papers in software engineering reflecting his long-standing interest in the design and specification of challenging software systems. He received a BS in mathematical sciences in 1977 and an MS in computer science in 1980, both from the University of North Carolina at

(< 5 minute) survey.

Chapel Hill. He received a PhD in computer sciences from the University of Texas at Austin in 1994.

The views expressed in this article are the author's only and do not represent directly or imply any official position or view of the Software Engineering Institute or Carnegie Mellon University. This article is intended to stimulate further discussion about this topic.

For more information

Contact Us

info@sei.cmu.edu

412-268-5800

Predictable Security: Security Analysis Extends Use of High-Performance Chip

NEWS AT SEI

Author

John Morley

This library item is related to the following area(s) of work:

Performance and Dependability

Predictability by Construction

This article was originally published in News at SEI on: January 1, 2008

What would help a soldier crouching at the edge of a battlefield, a firefighter intently peering at the horizon, and a tornado tracker racing through the countryside? They would benefit from real-time pictures of what they cannot see provided through the high-assurance collection, processing, and dissemination of airborne imagery.

Rockwell-Collins used a technology developed by the SEI to enable the high-assurance handling of data from multiple sensors having varying levels of security, such as airborne imagery, using a powerful, fast, integrated circuit called a field programmable gate array (FPGA).

“One FPGA does the work of thousands of computers,” says Yves LaCerte, a Rockwell-Collins systems engineer in Cedar Rapids, Iowa. It is easier to develop applications on an FPGA, too, reducing the cost and time to market, according to LaCerte. And the chip can be reprogrammed at runtime—to fix bugs, for example, which can lower maintenance-engineering costs.

What High Assurance Means for Software

For software to be considered high assurance, there must be a convincing argument that the software will always perform (or not perform) key functions.

A system that controls an aircraft’s actions in flight, for instance, must be high assurance, as must one that carries out satellite communication.

Related Links

Training

Software Architecture Design and Analysis

Modeling System Architectures Using the Architecture Analysis and Design Language (AADL)

See more related courses »

(< 5 minute) survey.

"Typically, you use a high-assurance processor to securely tag variable input. Rockwell-Collins wanted to demonstrate the high-assurance potential of FPGAs," LaCerte explains. "Because FPGA behavior is more complex, architecture-level definition and analysis are needed."

Meanwhile, at the SEI in Pittsburgh, Pa., Jörgen Hansson began investigating ways to use the Architecture Analysis & Design Language (AADL) and the Open Source AADL Tool Environment (OSATE) to model system architecture and analyze it for data quality attributes, including security.

"By verifying security using an architecture model, we can validate confidentiality and integrity and also determine that sanitization is done in a controlled way," Hansson says. Sanitization is the lowering of security levels; controlled sanitization assures that lowering security occurs only within allowed boundaries. Hansson's work culminated in an OSATE plug-in for security analysis.

Using AADL and Hansson's OSATE security-analysis tool, LaCerte built a prototype system that demonstrates "the correctness of the FPGA architecture and the correctness of the system's behavior."

AADL, a Language for Collaboration

AADL is becoming a lingua franca—a common language—for sharing information on problems and solutions among investigators in commercial, research, and academic organizations. In support of that notion, Bruce Lewis, head of the Society of Automotive Engineers subcommittee guiding the standard's development, points to the many consortia employing the standard.

In particular, Lewis notes the AVSI (Aerospace Vehicle Systems Institute) and SPICES (Support for Predictable Integration of mission Critical Embedded Systems). The AVSI uses AADL to demonstrate model- based validation of a system through architecture models. SPICES, an Information Technology for European Advancement (ITEA) project, offers designers of distributed, real-time, embedded systems a modeling, analysis, generation, and integration environment based on AADL.

The SEI and Rockwell-Collins stand out among the organizations leading development and transition of AADL. From the SEI, Peter Feiler provides technical leadership, and Bruce Lewis—an SEI resident affiliate from the U.S. Army Aviation and Missile Research, Development, and Engineering Center—runs the Society of Automotive Engineers (SAE) subcommittee guiding enhancement and expansion of the standard. Rockwell-Collins participates in the development of the AADL standard, publishes papers about the standard, creates example models, and demonstrates how to incorporate AADL into the development life cycle. Because of that involvement and interest, LaCerte learned of Hansson's OSATE security analysis plug-in.

While his achievement is significant for FPGAs and their use, LaCerte sees that the work he began with AADL and the security-analysis plug-in can go further. "We need to certify FPGAs for high-assurance use according to the NSA [National Security Agency] common criteria. AADL can be used to generate the artifacts needed to obtain that certification," LaCerte says.

Hansson's work goes on, as well. "We are currently investigating how to conduct tradeoff analysis by evaluating the effects of security on performance and resource usage."

For more information about model-based engineering and architectural modeling analysis, contact us using the link in the *For More Information* box at the bottom of this page.

Find Us Here

Share This Page

For more information

Contact Us

info@sei.cmu.edu

412-268-5800

Performance Improvement: It's a Small World After All

NEWS AT SEI

Author

Bill McSteen

This library item is related to the following area(s) of work:

Process Improvement

This article was originally published in News at SEI on: January 1, 2008

"Small businesses account for 99 percent of all employer firms in the U.S. and are critical to the U.S. economy. In other countries, small business *is* the economy. Organizations and governments around the world are approaching the SEI for help," says Caroline Graettinger.

Graettinger is leading the Improving Processes in Small Settings (IPSS) project, a recently formed SEI project that focuses on small businesses, projects, and organizational units. It is part of the International Process Research Consortium (IPRC), an SEI-led collaboration of industry, government, and academia from around the world seeking to advance process research.

"It is sometimes said that 'Process improvement is only for large companies,'" says William Peterson, director of the SEI's Software Engineering Process Management Program. "IPSS will give motivation, insight, and guidance to small organizations so that they can also get the associated performance-improvement benefits, but at an affordable cost relative to their size and resources."

The first sponsors of and collaborators in the IPSS project are the University of Pittsburgh Medical Center (UPMC) and Lockheed Martin Corporation. Why would these two organizations—with employees numbering in the tens or hundreds of thousands—be interested in small settings? Suzanne Garcia of the SEI responds, "Because, like many large organizations, they are amalgams of small projects, business units, partners, and suppliers."

Related Links

News

SEI to Co-Sponsor 26th Annual IEEE Software Technology Conference

SEI Cosponsors Agile for Government Summit

Training

See more related courses »

Events

Team Software Process (TSP) Symposium 2014

Nov 3 - 6

Health care companies such as UPMC contend with almost constant change in regulations that must be reflected in their information technologies. One-day projects are common. Large DoD contractors such as Lockheed Martin regularly subcontract to small businesses and obviously benefit from contractors with effective, efficient processes.

Lynn Penn of Lockheed Martin says, “Small settings are part of our daily lives. Although everyone associates Lockheed Martin with a large company, the interfaces are often not so large. We must manage small projects as well as large ones, and the small ones can often be more challenging. Through IPSS, we hope to gain guidance on skills and competencies needed to sufficiently manage small-settings projects,” Penn says. “Understanding the requirements going in is crucial to quality coming out.”

IPSS Field Guide

The IPSS Field Guide will be an on-the-job resource to help answer questions and solve problems, independent of the process model or standard used. Caroline Graettinger describes the guide: “We intend it to help small-setting practitioners be smarter consumers of process-improvement products and services or be better at improving processes themselves.”

She continues, “Our plan for populating the Field Guide includes collecting real-world experiences from experts across the process community who can provide knowledge, examples, checklists, and other artifacts to help others succeed in small settings.

Both Penn and Chris Carmody of UPMC say that working with other IPSS sponsors and team members has been a learning opportunity. Carmody adds, “Another benefit to me and to UPMC has been assessing how we currently approach our process-improvement work in our small departments and projects. We’ve used the structure of the emerging IPSS Field Guide to reassess and modify our own Project Management Guide to improve its acceptance and usability.”

“We are early in the development of the Field Guide,” Graettinger says. “Our first prototype details a few tasks, and with input from the process community, we’ll create step-by-step instructions for various situations.”

Penn says it is significant that the IPSS Field Guide will be the first research product out of the IPRC. “Individuals put stock in first impressions. To start in small settings emphasizes their worldwide importance. Isn’t it time that the little guy got attention?”

(< 5 minute) survey.

Find Us Here

Share This Page

For more information

Contact Us

info@sei.cmu.edu

412-268-5800

Effective and Measurable Results from Combining CMMI and Six Sigma

NEWS AT SEI

This library item is related to the following area(s) of work:

CMMI

Process Improvement

This article was originally published in News at SEI on: January 1, 2008

A new book, *CMMI and Six Sigma: Partners in Process Improvement*, was released in December 2007 by Jeannine M. Siviy, deputy director for the Dynamic Systems Program at the SEI; M. Lynn Penn, director of process management at Lockheed Martin; and Robert M. Stoddard, a senior member of the SEI technical staff.

"If people combine the initiatives thoughtfully, they can achieve more effective and measurable results from their process improvement efforts, and they can get results that are better aligned to their business needs and mission," Siviy explained in a recent interview. "For instance, our research includes data from organizations that have achieved their CMMI implementation objectives significantly faster than average, and in a way that is well aligned with business objectives."

Not long after her arrival at the SEI in 2000, Siviy, who was initially hired into the Software Engineering Measurement and Analysis (SEMA) initiative, began fielding questions about CMMI and Six Sigma and how the two relate. Penn, who has worked closely with the SEI on several initiatives and is highly regarded in the process improvement community as a CMMI pioneer, had also been fielding a lot of questions regarding CMMI, Lean, and Six Sigma—particularly Lean with its focus on time, and the variant of Six Sigma that was first adopted in Penn's organization. The two began collaborating on tutorials and presentations as well as a 2005 SEI technical note. The book seemed a natural extension of their work—a way to document the voiceover from the tutorials and presentations delivered over the years and also to extend more deeply into and beyond the topics they traditionally presented.

The book explores many aspects of combining CMMI and Six Sigma, to deal with the inevitable challenges any organization faces when trying to combine the two.

"There are a lot of challenges with CMMI and Six Sigma. A significant challenge is the perception that CMMI and Six Sigma are competitive, rather than complementary—leading to either/or adoption decisions, rather than joint implementation. Once this hurdle is overcome, there are questions about how they logically and technically fit together—what are the distinguishing features and what are the overlaps? Who gets trained? How do they get trained? Who is responsible to figure out a joint

Please note that current and future CMMI research, training, and information has been transitioned to the CMMI Institute, a wholly-owned subsidiary of Carnegie Mellon University.

Related Links

News

SEI to Co-Sponsor 26th Annual IEEE Software Technology Conference

SEI Cosponsors Agile for Government Summit

Training

See more related courses »

Events

Team Software Process (TSP) Symposium 2014

Nov 3 - 6

implementation? And so on," Siviy explained. Consequently, the book includes the value proposition for joint implementation and both strategies and implementation tactics. It features two organizational case studies and also contains several illustrations describing real-life process and product improvement projects using Six Sigma (DMAIC and DFSS) and Lean methodologies in a CMMI context.

Siviy, a chemical and systems engineer by training who worked at Eastman Kodak for 12 years before coming to the SEI, said the new book also delves much more deeply into the Lockheed Martin case study than any previous materials, detailing the organization's journey over a decade to join these two initiatives. Siviy describes Lockheed's approach as "state of the art" that needs to become "state of the practice."

Penn—who at Lockheed oversees policies and process-command media, process compliance via audits, and process-improvement activities—said an important part of Lockheed's implementation was to take the time, up front, to integrate all the processes throughout the organization and to make the involvement of every employee a priority, up to and including the president.

"The key to our approach is the process architecture, and that has to be done with forethought," Penn said. "The software and systems engineers didn't have to worry about what industry standard was new or what the flavor of the month was, they just remembered that they had to be compliant to the organization process standard."

The second organizational case study complements the Lockheed case study with a focus on the commercial sector: Motorola, the birthplace of Six Sigma. Prior to his joining the SEI's SEMA initiative, Stoddard worked at Motorola University where he led the development of the first software design for Six Sigma curriculum and served as quality director for the Motorola 3G cell phone business. In the Motorola case study, Stoddard highlights mappings between the initiatives to support decisions about CMMI adoption and an integrated training approach.

CMMI and Six Sigma extends beyond just these two initiatives and offers a snapshot of current research and practices for multi-model improvement, an essential topic given the hundreds of process improvement initiatives available to organizations. The book describes an emerging reasoning framework that builds on the lessons learned from the pairing of CMMI and Six Sigma as well as work done by SEI staff member Pat Kirwan, from the Frankfurt, Germany, office, and other colleagues and partners, and begins to build bridges to research and practices developed by others in the community.

Process improvement professionals facing questions about the joint implementation of CMMI and Six Sigma will find practical information in this book, as will the managers and technical staff with whom they frequently interact.

Siviy is a Kodak-certified Six Sigma Black Belt. Penn is a Lockheed-certified Six Sigma Black Belt, and Stoddard is a Motorola-certified Six Sigma Black Belt.

(< 5 minute) survey.

For more information

Contact Us

info@sei.cmu.edu

412-268-5800

Tackling the Growing Botnet Threat

NEWS AT SEI

Author

Julia H. Allen

This article was originally published in News at SEI on: February 1, 2008

This column is based on a podcast recorded with Nick Ianelli and posted to CERT's Podcast Series: Security for Business Leaders. Nick Ianelli is a member of the CERT Coordination Center, conducting artifact analysis on malicious code. Julia Allen, who interviewed him, manages CERT's Podcast Series and conducts research in security governance and software assurance.

Part 1: The Threat

Julia Allen: Would you describe what botnets are (short for robot networks) and why they are on the rise?

Nick Ianelli: A botnet is made up of compromised hosts, which are commonly referred to as bots or zombies.Botnets are collections of compromised hosts, centrally managed, or managed from multiple points, but they're logging into a location that's easily manageable.

It's hard to say with absolute certainty why these things are ongoing and rising. The simple fact is the code is out there, it's very easy to use, and if anybody has any questions, there's free support on the Internet to assist in trying to get these things to run, operate, or exploit vulnerable machines.

Julia Allen: When you say "compromised hosts," and you talk about the code, what that means to me is that someone's taken a piece of software, one of these bot software packages, and actually installed it on a whole bunch of computers.Do I have that right?

Nick Ianelli: Yes.

Julia Allen: And then they have an ability to control that in some fashion, upon request.

Nick Ianelli: The most popular command and control method that's being used is

what's commonly referred to as IRC, Internet relay chat. It's a text-based chatting program that's been around for years. Botnet attackers can log into IRC servers, go into specific channels, and see all of the users within that channel by issuing a command in that channel. All the compromised hosts there will respond and do the action requested by the person issuing the command.

Julia Allen: So is that one of the things that makes botnets so dangerous, this command and control structure, and why attackers find them so appealing?

Nick Ianelli: Yes, it's very easy for attackers to log into a central location, issue one command, and if they have a botnet of 1000 compromised hosts, have them all do the same thing at the same time.

Julia Allen: So how would attackers find a compromised host to install one of these bot software packages on?

Nick Ianelli: They could buy compromised hosts from other attackers. There's an underground economy, where attackers have a smorgasbord of items for sale:compromised hosts, stolen accounts, malware, botnet source code—you name it, they have it for sale.Some of these guys even start with infecting their own machine, or infecting a friend's machine, and then get that particular machine to start scanning the network for other vulnerable hosts.And once they have one, they just keep trying to look for others, and then once those get compromised, they're instructed to look for others. It's an ongoing repeatable process for them to add more hosts to their botnet.

Julia Allen: It sounds like there's a fair amount of information sharing that goes on within the intruder community about many topics, but about this topic in particular.Is that correct?

Nick Ianelli: Yes, it's amazing.These people have no qualms about sharing information with other people, whether they're direct competitors, senior people, or newbies. If somebody knows the answer to the question, or if multiple people know the answer to a question, they don't hesitate to assist, provide information, provide pointers, and to even provide code to do what the person's asking to do.

Julia Allen: That's remarkable.Well, without disclosing specific cases, could you describe some of the impacts of a successful botnet attack?

Nick Ianelli: One of the biggest impacts is data theft or data exfiltration.Once a bot gets on your machine, whatever you as the user have access to, or whatever your system has access to, the bot malicious code now has access to.And the people who are running these botnets understand that. So they've built in features that permit them to access all of the resources on the infected system.They put in key loggers, they look for specific files, they look for .doc (Microsoft Word) extensions, and spreadsheet .xls (Microsoft Excel) extensions. Once they find this information, they upload it to a specific location or they send the information to a central site where they can parse through it and decide what they want to do with the information.

Julia Allen: So pretty much anything that I would be doing on my laptop, be it at home or in the office, if there is bot code installed on my computer, an attacker can see or access everything I'm doing.

Nick Ianelli: That is correct.So if you have access to network shares for example, where other people store their backup files, the bot malicious code has the capability to potentially access that network share and have access to all of those files.All of your mail contacts, your address book—botnet malware has the ability to capture all of that.In addition, bots have the ability to send email from your machine as you to the people in your address book.

Julia Allen: Some pretty scary thoughts here, wouldn't you say?

Nick Ianelli: I'll say.

Part 2: How Do I Know If There's a Botnet on My Computer?

Julia Allen: Why is the infiltration of botnet agents on our computers so hard to

control?Couldn't firewalls, anti-virus (AV), or intrusion-detection systems be used to find these agents on our computers and eradicate them?

Nick Ianelli: In most cases I would say yes, as long as everything is properly configured, and everything is patched, secured, hardened and up-to-date.But if you think about it, botnets tend to propagate in two ways.

The first way is vulnerability exploitation; attackers are looking for vulnerable hosts on the Internet.Once they find one, they try to exploit it.What we see in analyzing botnet malware is that the majority of vulnerabilities attackers attempt to exploit have had patches available for three, four, five years. Attackers are still attempting to exploit vulnerabilities that came out in 2003, 2004, 2005.If people were to patch their systems, these vulnerabilities would no longer work.

The other way we're seeing is social engineering.As you may recall, I said I could send email, as you, to somebody within your address book. Sending email as you adds an extra layer of confidence in the person receiving that message.If I send email that says, "Hey, check out this attachment," or "Check out this Word document," and you double-click on it, there's a chance that your computer is going to run a piece of code that could install bot malicious code on your machine, and now you're infected and you're owned (by the attacker).

Julia Allen: Right, because I'm assuming the email came from you, so I trust it.

Nick Ianelli: Correct.

Julia Allen: And then proceed from there without realizing that I've just done myself some serious damage.

Nick Ianelli: Right. You're probably wondering, "Why didn't my AV catch that? There was malicious code in that email." You need to consider that attackers are in an arms race with the anti-virus community.Attackers will take a piece of code and they will try to obfuscate it so that anti-virus tools don't detect it.

There are public and private websites where an attacker can upload his malicious code, get the AV results from 10, 20, 35 different AV engines.He may be only concerned with a couple of them, but as long as they pass those couple, he'll try to exploit with that piece of malicious code.If the majority of the AV tools detect the attacker's code, the attacker will attempt to obfuscate it in another manner so that those AV tools don't detect it.

Julia Allen: Right, so effectively it flies under the radar screen and no one's the wiser.

Nick Ianelli: Exactly, and that's just trying to obfuscate it.Let's say that this is a brand new piece of malware—the AV community has never seen it.First, they need to get a copy.If their existing signatures or heuristics don't detect it, they need to get a copy, analyze it, and then they need to make a determination, "Do I adjust an existing signature?Do I create a new signature? Or do I adjust one of the heuristics to try to catch this?"Once they make these changes, then they need to push the updates out to all of their end users. All of this takes time but during this window, the botnet malware is propagating.

Julia Allen: Right, and that's a reactive solution, because you're analyzing the bot code after the fact, correct?

Nick Ianelli: That's correct.

Julia Allen: How can business leaders determine if botnet agents are on their computers, or on their organization's computers and networks, and if they do locate them, how can they get rid of them?

Nick Ianelli: They need to have logging on all their critical systems, including all of their systems that may touch the Internet.A primary example is if you have a router, you want to make sure that you're logging net-flow data from that router. Then you want to see if you can correlate that data with any of your other logs, say for your mail server or your DNS (domain-name system) server.

See if you can correlate or even visualize that data. There's a good chance that if there's a botnet, an infected host, or one machine that's in a botnet, and if you can quickly look at some net flow data, you have the potential to see quickly that there's an infection on your network.Generally what we see is once a computer's infected, it will try to scan for other computers. You'll be able to pick that out right away by looking at net-flow data visually.

Julia Allen: In other words, if there's an infected computer in my network, and it's trying to scan other computers, I'm going to see a real up-tick in the number of messages or the types of messages that computer is sending out, right?

Nick Ianelli: Yes. If your company has an authoritative DNS server—if your machines are configured to ask your company's DNS server first how to resolve a DNS name or a host name—you can set up some type of logging there, where you're looking for anomalies or odd-looking domain names, and you can alert on them. You don't have to drop the DNS request, but you can alert on them and then just manually review them.

With botnets, their DNS requests—the DNS names and the host names—are quite odd and obvious.That's a pretty good indicator of a potential infection.

Julia Allen: So taking that kind of preventive monitoring action before things get out of hand?

Nick Ianelli: Yes.

Part 3: And What Can I Do About It?

Julia Allen: Let's say that I've found this kind of activity through my logging or using an authoritative DNS service. How can I get rid of these guys?

Nick Ianelli: Your best bet is going to be to try to locate all the critical files on the system, pull them off or back them up, and then scan the files that you want to keep to make sure they're clean. The best way to ensure that your computer is clean is to just wipe it and start from scratch.Rebuild the operating system and then load all of your applications. Load all of your files back onto the system after you've ensured that they're not infected.It's really the only way you're going to know that your machine is no longer infected.

Julia Allen: So basically doing a thorough house cleaning?

Nick Ianelli: Yes.

Julia Allen: Are there any actions that business leaders and other users can take to get in front of this, both to prevent and to detect further infection?You had mentioned having your patches be up to date as one step.

Nick Ianelli: If you take a defense-in-depth approach, that's a great starting point. You want to raise awareness, both with your senior management and your employees.You want to make them aware that this type of activity is occurring, and when it occurs, what the potential losses are that both the organization and the individual can suffer. Not only can the organization suffer, but if individuals are going to a website, and a key logger is turned on, and that website just happens to be their personal bank account, that information can be exfiltrated, and attackers can get their hands on it.

Provide education and training classes.They don't have to be anything in-depth — maybe a lunchtime session to continue to keep this fresh in the people's minds. If they're doing the right thing at work, they're generally going to do the right thing at home.That just makes the Internet a better place for people to surf and visit.

Make sure that all of your patches and your software are up-to-date.If you need to test before you apply a patch, just know that. Have a plan ready so you can secure the system. Lock it down as tight as possible, so that until you get it patched, your services aren't down. Actively monitor this to make sure that nobody's exploiting the unpatched vulnerability.

Julia Allen: It occurs to me, given that attackers are looking for vulnerable machines, if my machine is well-patched, up-to-date, and securely configured, they're going to go somewhere else, right?

Nick Ianelli: Sure.A botnet logs into whatever command and control mechanism is in place (IRC or HTTP for example). These mechanisms use TCP port 80, so you've got to filter both inbound and outbound requests.You also have to watch the infamous peer-to-peer malware, such as the Storm worm, which operated on what appeared to be an edonkey peer-to-peer traffic.

So, yes, you need to watch this because when bot malware logs into its command and control server, if the command is to scan for a particular vulnerability, it will continue to scan for it as long as it's programmed to, and that can mean hours, days, or weeks.So as soon as a machine is compromised, it logs into the IRC server. The next thing you know, the bot code on the compromised machine is scanning the entire Internet, starting with your network.

Julia Allen: Right, so one of the best preventive measures is to make sure that they don't stop at my computer in the first place, right?

Nick Ianelli: Right.If you're raising awareness and providing education to your employees, the chances that they're going to get infected at work are slimmer and the chances they're going to be infected at home are slimmer as well.

Julia Allen: If I do get in trouble, are there certain organizations I can contact and where are some good places to learn more about this?

Nick Ianelli: The first thing you want to make sure of is that you have contacted your upstream Internet service provider (ISP).You want to know who to call and when to call before you actually have a problem. When you have a problem, the last thing you want to do is fumble around and be saying, "Oh man, where is this number?Who do I call?What line?"The important thing is time. If a machine on your network has been instructed to perform a DDoS (distributed denial-of-service attack) against another company, you're going to want to get that shut down. If there's a DDoS attack being launched against you, you're going to want to make sure you get that shut down. Your ISP is going to be your best friend in that case.

Know where to submit malicious code.Let's say you found botnet malware on your machine.What do you do with it?Know what to do with that information. Again, it's all about time.In a time of crisis, you want to make sure you've already defined a game plan.

Julia Allen: So another good preventive measure is to have incident response contacts and procedures in place, so that you can use them when you need them.

Nick Ianelli: Exactly.Being prepared ahead of time is going to save you in the long run.

Resource for More Information

CERT Podcast Series

Ianelli, Nicholas; Kinder, Ross; Roylo, Christian. "The Use of Malware Analysis in Support of Law Enforcement." CERT Coordination Center, Carnegie Mellon University, July 11, 2007.

Ianelli, Nicholas & Hackworth, Alan. "Botnets as a Vehicle for Online Crime." CERT Coordination Center, Carnegie Mellon University, December 1, 2005.

Share This Page

For more information

Contact Us

info@sei.cmu.edu

412-268-5800

User Network for Software Architecture Technology Is Growing Steadily

NEWS AT SEI

Authors

Robert Nord

Ipek Ozkaya

Rob Wojcik

This article was originally published in News at SEI on: February 1, 2008

The SEI Architecture Technology User Network (SATURN) Workshop series was initiated in 2005 at the Software Engineering Institute. In its three-year history, SATURN has attracted software and systems engineering practitioners throughout government and industry from all over the world. Because of the workshop's emphasis on software architecture practices, participant interaction, information exchange, and addressing the needs of practitioners, attendance continues to grow each year.

This year, SATURN will offer a program that includes

- keynote talks by software architecture experts Philippe Kruchten from the University of British Columbia, Canada, and Eoin Woods from Barclays Consultants, UK
- speakers who are government or industry professionals or SEI experts
- a panel discussion on software architecture from a manager's perspective led by moderators with both managerial and software architecture experience
- numerous opportunities for networking, including interactive and working sessions and social events
- opportunities to strengthen software architecture knowledge by taking courses from the SEI Software Architecture Curriculum or attending tutorials led by industry experts

A system's software architecture manifests the earliest design decisions that an organization makes to ensure that the resulting system meets its business goals and

quality concerns. SATURN 2008 will address:

- architecture-centric system evolution
- evaluating architectures of software-intensive systems
- architecture and technical trends
- improving software architecture practices within an organization

Architecture-Centric System Evolution

Many systems are built to be in service over a long period of time. However, planning for productive use over the life span of a system is difficult, since the future is uncertain due to market, social, economic, political, and technological forces. These forces and the system's reaction to them determine the utility—or measure of satisfaction relative to the system's business goals—of a system. Therefore, changes along any of these dimensions may affect the future utility of the system. For example, technologies on which systems are built change, and new architectural design paradigms emerge. Evolving technologies put the life expectancy of existing systems at risk when they become obsolete. It is a challenge for organizations to decide when to switch technology and how to prepare the system for evolution via its architecture. One method is to look at architecture in conjunction with economics to make decisions that provide the most value [Ozkaya 2007].

SATURN 2008 will provide insights into architecture-centric system evolution through

- a presentation by Rick Kazman of the SEI on the architecture-evolution techniques being developed by SEI
- a case study presented by Pia Stoll of ABB Corporate Research titled "Reconstructing the Architecture Model for a Sustainable Architecture"
- a working session led by SEI staff who are working in architecture evolution: Mark Klein, Rick Kazman, and Robert Nord. Topics will include the architecture-evolution challenges that organizations face and possible approaches to solving them.
- a tutorial led by Philippe Kruchten that will describe how system evolution can benefit from architectural knowledge management: "Software Architectural Knowledge = Architectural Design Decisions + Architectural Design."

Evaluating Architectures of Software-Intensive Systems

An essential aspect of mature software development practices is evaluating the architecture appropriately to determine risks early and to respond to them. This evaluation can occur in the context of an organization's developing new software systems. However, architecture-based evaluation becomes even more critical when an organization must understand the state of legacy systems before evolving them or the challenge of acquiring software-intensive systems through either mergers or relationships with subcontractors.

The SEI Architecture Tradeoff Analysis Method (ATAM) has been widely adopted as an architecture evaluation technique. The ATAM is a stakeholder-centric evaluation method that bases its evaluation outcomes on how the architecture responds to the business and mission goals of the organization and quality-attribute requirements by walking through the architecture using key, high-priority quality-attribute scenarios collected from key stakeholders [Bass 2003, Clements 2005]. The ATAM also is well suited for helping to understand the impact of new technological approaches, such as service-oriented architecture (SOA) [Bianco 2007].

At SATURN 2007, a panel on adopting the ATAM presented several approaches that different organizations took to evaluating architectures. The panelists discussed the benefits of using a common approach that could be tailored to the needs of individual organizations. In addition, a working session covered the topic of applying the ATAM to systems. At SATURN 2008, we will continue to explore this topic:

- Phil Bianco and Paulo Merson of the SEI and Rick Kotermanski of Summa Technologies will discuss key aspects of architecture evaluations in an SOA

context in their tutorial titled "Evaluating a Service-Oriented Architecture."

- A special session will be dedicated to hearing from several organizations about how they adopt evaluation techniques and use the ATAM in their organizations.
- Felix Bachmann of the SEI will present trends and challenges faced in organizations in conducting stakeholder-based evaluation techniques, such as the ATAM.
- A working session led by Mike Gagliardi and Bill Wood of the SEI will focus on how evaluation techniques originally developed for software architecture also can be used in system and system-of-system contexts.

Architecture and Technical Trends

To align the architecture with the overall life cycle of the product, it is critical to select and integrate tool support and respond to emerging technologies and trends. Architecture conformance—how the architecture aligns with implementation—is an example of where emerging technical trends, such as aspect-oriented programming, can be helpful [Merson 2007].

SATURN 2008 Is April 28–May 1 in Pittsburgh

The SEI is hosting SATURN 2008 in Pittsburgh, Pa., from April 28 to May 1 at the Radisson Hotel Pittsburgh Green Tree. Online registration runs through Monday, April 4; on-site registration is available at the conference.

SATURN 2008 is the leading forum for engineers, architects, technical managers, and product managers who work with or have a stake in software architecture practices. The workshop provides a venue for practitioners to reflect on the achievements made, assess the current state of the field, and identify key challenges still facing researchers and practitioners. To learn more about SATURN 2008 or to register, visit the SATURN website.

During SATURN 2008, participants will investigate emerging and existing technical trends in software architecture:

- Scott Hissam and Rob Wojcik of the SEI will lead a working session about the challenges and successes participants have encountered in developing systems with open-source technologies.
- J. D. Baker of BAE Systems will describe the place that architecture has in a model-based end-to-end engineering methodology in his presentation titled "Software Architecture in the Integrated Engineering Methodology."
- Kyungsoo Im and John D. McGregor of Clemson University will present their work in debugging software architectures.
- Phil Bianco of the SEI will talk about the current technology investigations in software architecture.
- Doug Kimelman from the IBM Thomas J. Watson Research Center will make a presentation titled "Lessons Learned from Deployment and Production Use of Architects' Workbench: An Architectural Thinking and Modeling Tool."

Incorporating Software Architecture Practices Within an Organization

Institutionalizing proven and effective software architecture techniques in an organization is of interest to the software architecture community. As organizations become more competent, we believe that architecture practices will be influential throughout the life cycle. During SATURN 2007, Stuart Kerrigan and Richard van Schelven of Ericsson talked about how they enhance the Rational Unified Process with SEI architecture techniques [Kerrigan 2007]. SEI techniques can be integrated with RUP and other agile methodologies such as eXtreme Programming [Kazman 2004, Nord 2004]. SATURN 2008 will offer opportunities for participants to share their experiences in this area:

- Philippe Kruchten of University of British Columbia will share his experiences during his keynote titled “On Software Architecture, Agility, Cost, and Value.”
- Andre Leclerc from Unisys will present his organization’s experiences in considering quality attributes in its general-requirement-analysis methods in his talk titled “Quality Attributes and Requirements Traceability.”
- Issac Eldo from Philips Medical System will present “Architectural Empowerment: A Quality Attribute of Software Architecture Realm to Build Empowered Organizations.”

These are only some of the highlights of SATURN 2008. The program was compiled with the architecture needs of the software engineering community in mind. The workshop is organized with the goal of maximizing learning, interaction, networking, and information exchange. SATURN organizers provide an environment in which these encounters are encouraged and facilitated. Visit http://www.sei.cmu.edu/architecture/saturn/2008 to get the latest information on what we’ve planned and to register.

References

[Bass 2003]
Bass, Len; Clements, Paul; & Kazman, Rick. *Software Architecture in Practice, Second Edition*. Addison-Wesley, 2003.

[Bianco 2007]
Bianco, Phil; Kotermanski, Rick; & Merson, Paulo. *Evaluating a Service-Oriented Architecture* (CMU/SEI-2007-TR-015). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2007.

[Clements 2005]
Clements, Paul; Bergey, John; & Mason, Dave. *Using the SEI Architecture Tradeoff Analysis Method to Evaluate WIN-T: A Case Study* (CMU/SEI-2005-TN-027, ADA447001). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2005.

[Kazman 2004]
Kazman, Rick; Kruchten, Philippe, Nord, Robert L.; & Tomayko, James E. *Integrating Software-Architecture-Centric Methods into the Rational Unified Process* (CMU/SEI-2004-TR-011, ADA455624). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2004.

[Kerrigan 2007]
Kerrigan, Stuart & van Schelven, Richard. *SEI Architecture Techniques Complementary to the RUP*.

[Merson 2007]
Merson, Paulo. *Using Aspect-Oriented Programming to Enforce Architecture*(CMU/SEI-2007-TN-019). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2007.

[Nord 2004]
Nord, Robert L.; Tomayko, James E.; & Wojcik, Rob. *Integrating Software-Architecture-Centric Methods into Extreme Programming (XP)* (CMU/SEI-2004-TN-036, ADA431084). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2004.

[Ozkaya 2007]
Ozkaya, Ipek; Kazman, Rick; & Klein, Mark. *Quality-Attribute-Based Economic Valuation of Architectural Patterns* (CMU/SEI-2007-TR-003). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2007.

About the Authors

Ipek Ozkaya is a senior member of the technical staff in the Product Line Systems Program at the SEI. Her research interests include developing methods for improving software architecture practices focusing on software economics and requirements

management. Ozkaya lectures on architecture-centric approaches as an instructor in the SEI Software Architecture Curriculum. She earned a PhD in computational design from Carnegie Mellon University. She is currently working on techniques for methodological design of software architectures with value-based approaches.

Robert Wojcik has been a member of the technical staff in the Product Line Systems Program at the SEI since 2004. He performs training and consulting in software architecture technology and software architecture evaluations. Prior to his position at the SEI, he devoted more than 20 years of his career to developing software systems that use traditional as well as artificial intelligence and object-oriented technologies. He has an MS in software engineering from Carnegie Mellon University and a BS in computer science from the University of Pittsburgh.

Robert L. Nord is a senior member of the technical staff in the Product Line Systems Program at the SEI where he works to develop and communicate effective methods and practices for software architecture. Prior to joining the SEI, he was a member of the software architecture program at Siemens, where he balanced research in software architecture with work in designing and evaluating large-scale systems. He earned a PhD in computer science from Carnegie Mellon University. Nord lectures internationally on architecture-centric approaches. He is co-author of *Applied Software Architecture* (2000) and *Documenting Software Architectures: Views and Beyond* (2002).

The views expressed in this article are the author's only and do not represent directly or imply any official position or view of the Software Engineering Institute or Carnegie Mellon University. This article is intended to stimulate further discussion about this topic.

For more information

Contact Us

info@sei.cmu.edu

412-268-5800

Workshop Examines SOA Hard Problems and Potential Solutions

NEWS AT SEI

Author

John Morley

This library item is related to the following area(s) of work:

System of Systems

This article was originally published in News at SEI on: February 1, 2008

When it comes to the current perspectives on service-oriented architecture (SOA), there are gaping needs for robust tooling, reliable metrics, and education about business or mission implications, among other requirements. But there are efforts moving forward in some government, academic, and commercial organizations to define strategies for SOA adoption, study security issues, form governance models, and create measurement vehicles.

Those observations are among the more than 200 issues, challenges, and steps to fill gaps voiced by participants in the Hard Problems in SOA Workshop at the Software Engineering Institute (SEI) on January 30. The SEI co-sponsored the workshop with IBM (the Federal SOA Institute), and Carnegie Mellon University.

The observations highlighted by the 110 workshop participants illustrate a broad concern about the implications of SOA adoption. The participants work in government agencies, military service branches, health care organizations, universities, research and development centers, and commercial enterprises. Some of the more than 35 organizations represented were the U.S. Air Force, Army, and Navy; University of Pittsburgh Medical Center; Internal Revenue Service; Harris Corporation; the Office of the Secretary of Defense; Northrop Grumman; Villanova University; PNC Financial Services; the Federal Aviation Administration; and Westfield Insurance.

Participants represented the interests, viewpoints, and experiences of service consumers, service providers, system architects, acquirers, researchers, and others. They contributed to working sessions on these dimensions of SOA:

Related Links

Training

Migrating Legacy Systems to SOA Environments - eLearning

Service-Oriented Architecture: Best Practices for Successful Adoption

See more related courses »

(< 5 minute) survey.

1. SOA Governance
2. Strategy, Justification of SOA Projects, ROI, Strategic Plan for SOA Introduction
3. Security for SOA
4. SOA Design, Development, and Deployment: Methods and Tools

"These dimensions focus on problems specific to the use of SOA within federal government organizations, but they have implications for commercial organizations as well," says Grace Lewis of the SEI.

The workshop was focused on a taxonomy of major SOA research challenges that the SEI has developed. The workshop discussions followed a template that helped attendees to address areas of the taxonomy by focusing on the following questions: Which issues are most important and why?; What is known now about the issues?; Where do the gaps exist between what is known and what needs to be discovered about the issues?; and What are some ideas for addressing the gaps? Following the working sessions, each group shared its views with all workshop participants.

SOA Governance: Just Enough

From the session on "SOA Governance," participants identified as key issues the need for a governance metamodel and for consideration about "ecosystem" (beyond the scope of a single organization) governance.

A metamodel, the Governance group reported, would ensure consistency among organizations in an SOA environment, while allowing individual organizations to tailor governance policies as needed. Among the gaps noted in the current state of development are

1. the need for a business process ontology
2. consideration of service reuse and measurement
3. definition of knowledge and event rules

In the context of the ecosystem, SOA has led to a world that encompasses much broader governance than a single organization. A key challenge is to address federation while still providing only the minimum level of governance needed.

SOA Strategy: Finding Where SOA Applies and Measuring Its Effectiveness

The "SOA Strategy, Justification of SOA Projects, ROI, and Strategic Plan for SOA Introduction" working group identified these issues:

1. SOA strategy definition
2. applicability of SOA
3. operational effectiveness

An SOA strategy definition is needed, the group contended, because SOA crosses boundaries of areas of interest, so a narrow approach will not be successful. Also, SOA adoption requires an organizational transformation that can come about only incrementally. The group noted that current funding models are incompatible for an SOA paradigm and that many organizations resist relinquishing control.

Further, SOA is not a one-size-fits-all proposition. There needs to be guidance about where SOA applies and where it does not. Decision makers, the group concluded, are not as well-versed as they need to be in the business implications of SOA.

Operational effectiveness is the core motivation for any organization to consider an SOA environment. And the heart of operational effectiveness is measurement. Yet it is hard now to determine the cost of things that cross organizational boundaries, such as shared services.

SOA Security: Challenge of Federated Environment

The working session on "Security for SOA" found that the complication of composite applications, implications for user identity, and management of security across diverse

environments were the top issues.

In composite applications, or choreographed services as the working group defined them, it can be a challenge to guarantee security results when it is unknown how the system will be assembled. Specifying security policies in this environment is a particularly difficult issue.

At times, the group noted, service providers can also be service consumers, making it difficult to form a chain of identity. Also, because the systems from which the services are exposed belong to different organizations, federated schemes of security are needed. In addition, there is a lack today of large-scale tooling and consistent security metrics to aid the managing of security in those federated situations.

SOA Design, Development, and Deployment: New Methods for Greater Complexity

In the session on "SOA Design and Deployment," participants identified these issues:

1. a methodology for a system life cycle that features a high rate of change and high degree of complexity
2. the need to understand and evaluate service-oriented systems
3. assessment

The methodology discussed will facilitate better decision making and aid interoperability. The lack of a common method for IT architects and business analysts represents a gap today in meeting this need.

In terms of architecture, the group asked

1. What are attributes that are most important, and particular, in an SOA environment?
2. How can you model and measure agility, rate of change, resiliency of architecture, and the ability to evolve?
3. What are the generations of the architecture?

SOA environments demand assurance, yet they offer new and significant testing challenges. There is an impedance mismatch between the specification and implementation of services, according the group. Also, an organization may rely on services that it does not control.

Full Workshop Documents Available

Outbriefs for all four working sessions along with a presentation on the SOA research taxonomy are available on the Interoperability section of our website.

This was an exciting workshop. People came together to share their thoughts on the hard problems of SOA," said Frank Stein of IBM. "We want to continue doing this because we want to know and address what the next hard problems are as SOA adoption increases."

For additional information, contact us using the link in the *For More Information* box at the bottom of this page.

412-268-5800

Being Your Own Boss—Part V: Building Trust

NEWS AT SEI

Author

Watts S. Humphrey

This library item is related to the following area(s) of work:

Process Improvement

This article was originally published in News at SEI on: March 1, 2008

This fifth and final column on being your own boss describes how to take control of your own work. It covers the key issues you will face, the fears you must overcome, and the self-confidence and credibility you must build. While the challenge of doing all of this may seem daunting, and while it does take personal courage, it isn't really that difficult. The key is to be willing and able to take responsibility for your commitments. While you can take all of these steps by yourself, it helps to have teammates to work with you. Once you know how to manage yourself, however, you will not want to work any other way.

Starting a Job

To take charge of your own work, you must assert your position at the very beginning of every job. It is the only way that you will be able to consistently work on projects with realistic schedules and plans. Working in this way, however, requires the courage to take personal risks, a willingness to step out of the crowd, and the responsibility to own your own work.

Just about every one of us in the software business has been regularly told what to do and when to do it. This is the way our business has always been run. While there are many reasons for this state of affairs, the fundamental cause is that management does not trust us to establish our own plans or to set our own schedules. This is not surprising, however, for, with few exceptions, when software groups have been asked to make their own commitments, they have rarely met them.

So you start out with two strikes against you. In this negative environment, you must somehow convince your managers that they can trust you to do what you say. Then, of

Related Links

News

SEI to Co-Sponsor 26th Annual IEEE Software Technology Conference

SEI Cosponsors Agile for Government Summit

Training

See more related courses »

Events

Team Software Process (TSP) Symposium 2014

Nov 3 - 6

course, you must do what you promised. This is critical, because if you don't, it will be even harder to convince management the next time. To do all of this, however, you must overcome some fears and build your self confidence. Only then will you be able to convince your managers that they can trust you. The challenges are to overcome fears, to build self-confidence, and to establish trust.

Overcoming Fear

While there are many kinds of fears, the worst ones are fears of the unknown. The best way to deal with these is to consider rationally what is most likely to happen and to assess all of the realistic outcomes. For example, suppose management told you and your teammates that this new project must be done in three months. And suppose also that none of you believed there was the slightest chance that you could do the job in that time. What would happen if you told management that three months was not nearly enough time?

In response, management is likely to say one of two things. First, they could say: "Three months is critical to the business, so do your best." Since you can't refuse to try, all you can do is agree and, regardless of what you think, you will then be stuck with a three-month commitment for doing the job. The second thing management could say is: "How long do you think it will take?" Then you would probably have to answer: "While I don't know how long it will take, it will certainly be longer than three months." Then you will again get management's first answer and be stuck with the same three-month schedule.

Of course, instead of saying you don't know how long the job will take, you could make a guess and say it will take five months. At this point, any manager with half a brain would say: "Prove it." And again, unless you had a really convincing story, you would be stuck with the same three-month schedule. So this fear is really not an unknown at all. If you complain about the short schedule, there are very few possible outcomes, and they all end up exactly where you started: stuck with the three-month schedule. There is, however, another approach you could take and it concerns building your self-confidence.

Building Self-Confidence

Some people can speak with assurance about things they don't know or understand, but that skill is more common to politicians than technical people. The fundamental problem with bluffing is that the person with the most resources or power usually wins. In development work, of course, that is the manager and not the developer. This means that if you want to debate the schedule with your management, you must know what you are talking about.

This suggests that you follow a different strategy. Now, when management says that the job must be done in three months, first make sure that you understand what they want you to do and then tell management that you need to make a plan before you can discuss the schedule. At this point, management really has only two choices: They could refuse to let you make a plan or they could agree and wait for you to come back. In all of my years in development work, I have never run into a manager who refused to let the developers make a plan. While the manager may be surprised at your request, he or she will invariably agree to let you make a plan, as long as you don't take too long.

This means that you will almost certainly be faced with the need to make a plan. In doing so, however, you must try to make a plan that does the job on the requested schedule. Then, if you can't, you will know why and be able to explain the problem to management. Better yet, you could propose several alternate plans, each of which meets some but not all of management's goals. Also, of course, you must know how to make a realistic and convincing plan. I have written extensively on this subject, and if you are working by yourself, the best reference is *PSP: A Self-Improvement Process for Software Engineers* [Humphrey 05]. If, however, you are part of a development team, you should still consult this PSP reference but also look at: *TSP: Coaching Development Teams* [Humphrey 06].

Establishing Trust

Let's assume that you are to do the job by yourself and that you have now made a plan. Also let's assume that, in making this plan, you followed the methods described in the PSP book. Now your job is to convince management that you have made a realistic and aggressive plan and that you are willing to commit to a delivery schedule. Also, if history is any guide, you will have found that this job is far too big for you to complete in the desired three months, and that five months would be required. To make your plan presentation to management convincing, you must describe both how you made the plan and the plan itself. The key topics to cover in this meeting are the following:

1. the goals of the project—in this case, to finish in three months
2. the assumptions you made—presumably about the requirements and other key project characteristics
3. your concept of what the product will look like
4. your estimate of the product's size and how it compares to other previously developed products
5. your estimate for the time required to develop the product and the data you used to make this estimate
6. some alternate plans that show how adding resources or reducing product function could shorten the schedule
7. the recommended plan and schedule
8. the key project risks and recommended mitigation actions

While this is a lot of work, it is a lot easier than struggling for months with an unrealistic plan and schedule. Furthermore, with a little guidance and experience, you will find that it takes surprisingly little time to make such a plan. For your own personal work, you should be able to make such a plan in a few hours to a day or two. Even teams building large products can typically make these plans in about a week. Finally, I can guarantee that, if you take these steps and make such a presentation, management will trust your plan and negotiate an aggressive but realistic schedule with you.

Maintaining Trust

Once you have produced a complete and realistic plan, and once you have convinced management to accept that plan, you can get to work. However, as you do the work, you must continue to maintain management's trust. Trust is fleeting, and after a relatively brief period, management will start to worry and begin to suspect that you are having problems. Given a little time, they will even start thinking that you cannot be trusted to meet your commitments. To counter this natural tendency, you must keep management regularly informed about your status and progress.

Again, this is not that difficult, and the PSP and TSP books can show you how to do it. I have found that, at the project level, weekly reports to your immediate managers are required. If you waited for a month between reports, management would almost certainly worry and even two weeks is probably too long. These reports should factually describe your progress and summarize any key issues and problems together with the actions you are taking to address them. By doing this, you can keep management on your side and ensure that they will be willing and able to help you if you run into serious problems.

Since problems are common in development work, and since some of these problems are likely to impact your schedule, you will occasionally find that you cannot meet an important commitment. While preparing a sound and detailed project plan will usually enable you to meet your commitments, occasionally that won't be possible. This is when frequent management reports are most important. As soon as you know that the schedule is exposed and that you are almost certain to miss the date, go to your management for help. As long as you keep them regularly informed and give them plenty of warning of problems, they will be willing to help solve the problems. Then, even if you do have to miss the schedule, management will continue to trust you to manage your own work.

Commitments in Practice

Depending of your organization's business, use the same commitment principles but follow different implementation practices. For example, with an existing fixed-price contract, cost and schedule will be paramount and the key will be finding the alternative approaches that most closely meet the existing commitments. Since even fixed-price contracts can run late and over cost, however, make the most realistic plans that you can and face any bad news as quickly as possible. When developers merely hope things will improve, they practically never do. Then, when the bad news must be faced, there is no time to consider alternatives or prepare for the overruns. While the bearers of bad new are often shot, if you know what you are talking about and can defend your story, you will almost always come out ahead. But since there are no guarantees in this business, you must have the courage of your convictions.

Regardless of the business situation or the size and scope of the job, you will know least about the job at the beginning and learn more as development proceeds. Therefore, for any reasonably sized project, you must make progressively more detailed and better informed plans every few months. Also, since it is impossible to make detailed plans for complex development work that extend for more than three or four months, you must regularly replan anyway.

Regardless of your situation, the key is to negotiate a realistic plan and commitment at the beginning. Then, as you learn more about the work, continue refining your plan as you strive to meet that commitment. If you find that you can no longer meet the commitment, get to management right away so they can both help you meet the commitment and prepare for any possible delay.

Conclusions

This series of five articles has described how to take charge of your personal work and explained why this is the only way to have a truly satisfying and rewarding work life. The first column in the series described the different ways that managers and developers view success and it concluded that, to be truly successful, projects must be successful for both the managers and the developers. The second column pointed out that managers often seem autocratic merely because their developers don't know how to make accurate plans or consistently meet commitments. The third column points out that the only ones who can make accurate and useful plans for knowledge-working projects like software development are the knowledge workers themselves. Part four of the series then describes the five steps required for developers to take charge of their own work, and part five discusses the challenges and risks of actually doing so. Taking charge of our own work is the key to a rewarding and satisfying career. Once you know how to do it, you won't want to work in any other way.

Acknowledgments

In writing papers and columns, I make a practice of asking associates to review early drafts. For this column, I particularly appreciate the helpful comments and suggestions of Tim Chick, Julia Mullaney, and Bob Stoddard.

In Closing, an Invitation to Readers

In these columns, I discuss development issues and how they impact the work of engineers and their organizations. However, I am most interested in addressing the issues that you feel are important. So, please drop me a note with your comments, questions, or suggestions. Better yet, include a war story or brief anecdote to illustrate your ideas. I will read your notes and consider them when planning future columns.

Thanks for your attention and please stay tuned in.

Watts S. Humphrey
watts@sei.cmu.edu

References

[Humphrey 05]
Watts S. Humphrey, *PSP: A Self-Improvement Process for Software Engineers*, Reading, MA: Addison Wesley, 2005.

[Humphrey 06]
Watts S. Humphrey, *TSP: Coaching Development Teams*, Reading MA: Addison

Wesley: 2006.

For more information

Contact Us

info@sei.cmu.edu

412-268-5800

Cyber Attack Scenarios Test Responses

NEWS AT SEI

Author

Heidi Price

This article was originally published in News at SEI on: March 1, 2008

Imagine how your organization would function without the Internet. Or imagine how your day would proceed—or not—in the wake of a power outage that lasts for days on end similar to the one that blanketed the northeast in darkness in 2003.

If you can, then you've imagined a day in the life of Marty Lindner.

As a member of the Software Engineering Institute's CERT Coordination Center (CERT/CC), Lindner serves as both architect and designer of the worst-case cyber scenarios that an organization, whether commercial or governmental, could face.

Then he tries to make them happen. Well, sort of.

Lindner envisions and then creates scenarios for cyber attacks and other disruptive events and then tests organizational response as if the disruptive events are actually occurring. The type of scenario depends on the objectives of the exercise.

"You can have a technical objective where you find out if your IT guys really know how to apply a patch. You could also do it from a policy level. You want to understand that you really do have the policies in the right place to handle certain situations and anywhere in between," Lindner explains.

One of the most extensive cyber exercises is "Cyber Storm," a pseudo-cyber attack coordinated through the U.S. Department of Homeland Security's (DHS) National Cyber Security Division (NCSD). The exercise tests how senior leaders of the U.S. government would respond to a cyber incident of national significance.

Although similar exercises had been conducted previously, the first Cyber Storm exercise was conducted over five days in early 2006 and involved more than 100 public and private organizations in five different countries. A second Cyber Storm will be conducted this spring.

During the first Cyber Storm exercise, many of the people involved weren't aware that it was an exercise, according to Lindner. Those in the know followed a scripted response. Those who didn't had their calls routed to someone who did.

Lindner said the goal is to try to make the exercises as "real-world" as possible.

"There are real-world problems like software flaws. A software flaw is an underlying root cause, but it's not a problem until a bad guy takes advantage of it," Lindner says. "The whole process is controlled. At the end of the day, the intent is not to make anyone look bad. You're not trying to prove someone is better than another person. By going through the process, you are raising the bar."

For more information about CERT, visit www.cert.org/

(< 5 minute) survey.

Find Us Here

Share This Page

For more information

Contact Us

info@sei.cmu.edu

412-268-5800

SMART Ultra-Large-Scale Systems Forum: "Scale Changes Everything"

NEWS AT SEI

Author

Bill Pollak

This library item is related to the following area(s) of work:

Ultra-Large-Scale Systems

This article was originally published in News at SEI on: March 1, 2008

On March 6, 2008, the Software Engineering Institute (SEI) conducted a forum, "Scale Changes Everything," on ultra-large-scale (ULS) systems in conjunction with Strengthening the Mid-Atlantic Region for Technology (SMART), a non-profit organization dedicated to integrating regional science and technology activities within Delaware, Maryland, New Jersey, and Pennsylvania. The forum focused on the results of a recent study, *Ultra-Large-Scale Systems: The Software Challenge of the Future*, that was led by the SEI. Held on the Carnegie Mellon University campus in Pittsburgh, Pa., the event brought experts involved in the ULS systems study together with community leaders interested in the growing trend toward ULS systems.

ULS systems are systems that exceed some critical limit of today's software engineering technology. The SEI-led research study began when Claude M. Bolton, Jr., former assistant secretary of the Army (Acquisition, Logistics, and Technology), posed this question to the SEI in 2004: "Given the issues with today's software engineering, how can we build systems of the future that are likely to have billions of lines of code?" Although Mr. Bolton initially defined the challenge of ULS systems in terms of lines of code, size is only one limit. Others include unboundedness, continuous requirements evolution, and continuous operation.

In her opening keynote address, Linda M. Northrop, director of the Product Line Systems Program at the SEI and leader of the SEI ULS systems study, discussed the impact of scale and presented a summary of the key insights from the SEI study. "Software is the least well-understood and the most problematic element of our largest

systems today," Northrop said. "Government and industry need to be prepared to build the systems of the future that will be ultra-large scale in many dimensions. We cannot afford to wait or postpone this research."

Later in the morning, John Goodenough of the SEI moderated a panel discussion in which representatives from the community shared their experiences and challenges with increasing scale in their domains.

John Bloomer, CIO of Virtua Health, discussed the challenges of scale in the health-care industry. “U.S. health care remains a specialized compendium of silos and compartments with distinct care processes and protocols,” he said. He envisioned for the audience a future of “unified, preventative, holistic care processes and teams enabled by technology.” However that future, said Bloomer, depends on the availability of solutions to the challenges of scale.

Bob Kent, executive director of the System of Systems Security (SOSSEC) Integration Initiative, discussed the challenge of integrating disparate agencies and organizations charged with defending homeland security. “Unless an integrated system can be fashioned,” said Kent, “we will continue to fail in our prevention and response efforts.”

Patricia Hoffman, principal deputy assistant secretary, Office of Electricity Delivery and Energy Reliability, U.S. Department of Energy, discussed the complexity of the critical electric-power infrastructure.

Daniel J. Paulish, Siemens Corporate Research, Inc., validated the central tenets of the SEI study from the point of view of a large company with more than 30,000 software developers that spends more than 3 billion euros a year on software development. “Our software systems engineering methods and technologies,” said Paulish, “must address the increasing scale and complexity of emerging software systems.”

Mark Uland of the Boeing Company discussed his role as deputy chief architect of the System of Systems Common Operating Environment for the Army’s Future Combat Systems, a system of more than 33 million lines of code whose scale and complexity are already challenging current software engineering capabilities.

Links to these presentations are available in our library.

Thomas J. Killion, chief scientist of the United States Army, delivered the afternoon keynote address about ultra-large-scale systems in the Army. He underscored the importance of the ULS systems study and the need for research in this area.

The afternoon panel discussion, moderated by Northrop, focused on the areas identified in the SEI study as most in need of research breakthroughs: human interaction; computational emergence; design; computational engineering; adaptive system infrastructure; adaptable and predictable system quality; and policy, acquisition, and management. The panel session provided more detail about the research areas proposed by the ULS systems report, described progress that has occurred since the report’s publication, and engaged all attendees in a discussion about ongoing or needed research to meet ULS system challenges. The panelists, all members of the ULS system study team and coauthors of the report, were Richard P. Gabriel of IBM Research, Douglas C. Schmidt of Vanderbilt University, Kevin Sullivan of the University of Virginia, and Mark Klein, Kurt Wallnau, and John B. Goodenough of the SEI.

Links to these presentations are available in our library.

The event, attended by 80 government and industry leaders and researchers, was followed by an exhibit program and reception where Pennsylvania Lt. Governor Catherine Baker Knoll expressed her appreciation to the ULS system study team.

For more information about ULS systems, contact us using the link in the For More Information box at the bottom of this page or visit the SEI ULS website.

Share This Page

For more information

Contact Us

info@sei.cmu.edu

412-268-5800

Steps Toward Network-Centric Operation

NEWS AT SEI

Author

John Morley

This library item is related to the following area(s) of work:

System of Systems

This article was originally published in News at SEI on: March 1, 2008

According to Input, a market research firm, the U.S. Department of Defense (DoD) is spending heavily on information technology (IT)—more than $23 billion in 2007 alone. What's the top DoD priority for those dollars? David Wennergren, the DoD deputy Chief Information Officer and deputy assistant secretary of defense for information management and technology, says that the development of network-centric operation (NCO) is at the top of the list.

NCO refers to systems or activities that are enabled by large-scale communications networks. Thus, most modern military systems of systems are network-centric systems, and military operations are often network-centric operations. Wennergren says the purpose for NCO is to improve access to data.

NCO "calls for a shift in focus to the importance of sharing data and the awareness of data," agrees Craig Meyers of the Software Engineering Institute (SEI). "It is important because threats are continually evolving, and the pace at which we can apply technology advances influences the outcome of defending against those threats," Meyers adds.

"But today, acquisition and development are system-centric—that is reflecting a single-program, stovepipe view. Within in the program acquisition, there is sharing of data—but it is within the stovepipe, between a contractor and subcontractor, for instance. We need to move to a mission-centric, collaborative view that looks at the system-of-systems context to gain the benefits of network-centric operation," Meyers says. "We need to acquire systems so that their integration as systems-of-systems can later occur."

Related Links

Training

Migrating Legacy Systems to SOA Environments - eLearning

Service-Oriented Architecture: Best Practices for Successful Adoption

See more related courses »

(< 5 minute) survey.

Meyers, along with SEI colleagues David Fisher and Pat Place, identified six conditions necessary for network-centric operations in a recent technical report. “The conditions are characteristics of network-centric operation. But they also point to the transition needed to operate in a network-centric environment,” Meyers says.

Six Conditions for Network-Centric Operation

- **The social and cultural environment in which systems of systems are acquired, developed, used, and evolved motivates collaborative behavior critical to achieving operational effectiveness.**

  “The funding purpose directs the focus for the acquisition and development,” Meyers notes. “We need to see incentives that reward organizations for acting and interacting in cooperation and shared understanding of evolving operational needs.”

- **The legal and regulatory framework supports the acquisition of systems of systems.**

  “Consider an example of the integrated master schedule [IMS],” Meyers says. “Regulation, policy, and practice dictate that an IMS, a machine readable [commercial off-the-shelf] COTS tool, be used. But what if different program managers use different tools for their IMSs? It’s a case of ‘I can’t read what your tool reports, and you can’t read what mine reports. How can those program managers share data?”

- **Management practices are sufficiently defined and performed to enable the acquisition of systems of systems.**

  Management practices refine laws and regulations, according to Meyers. “With the IMS, for instance, management practices pertaining to the areas program managers work with—cost, schedule, risk, and performance—have to allow for the sharing of information. But the users, those in the operational environment, might point out that the acquisition and development of systems takes too long, costs too much, and doesn’t deliver what is needed to meet threats that have changed.

- **Governance is cooperative, distributed across the constituents, and applied selectively.**

  Governance involves policies for the control and coordination of IT resources, enforcement of those policies, and measurement of the outcomes. In a network-centric environment, according to the report, owners of the component systems must share governance rather than dictate it to one another. As a step toward meeting this requirement, Meyers notes that the new CMMI® for Acquisition (CMMI –ACQ) model includes a process area on agreement management that aims “to ensure that the supplier and the acquirer perform according to the terms of the supplier agreement” [1].

- **Engineering practices appropriate for evolving (including developing) systems of systems are available, widely understood, and applied.**

  “We need to address how system engineering practices must change to enable us to acquire, build, operate, and maintain systems of systems,” Meyers points out. Traditional engineering practices just won’t work to build systems that can operate effectively in network-centric environments, according to the report. New engineering practices are needed that
  - emphasize flexibility and adaptability
  - envision operational use in a dynamic, uncertain world of continuously changing needs
  - exploit the benefits of emergent effects

- **A technology base exists that is capable of realizing the network-**

centric vision.

Primarily, industry will supply the technology base for network-centric operations. In the report, Meyers points out that the DoD can spur the necessary research by industry into network-centric problems through organizations such as the Defense Advanced Research Projects Agency (DARPA) and service-branch-oriented research programs including the Army Research Office (ARO), Office of Naval Research (ONR), and the Air Force Office of Scientific Research (AFOSR). In addition, the technology base can be widened by leveraging technologies developed by coalition partners.

Meyers acknowledges that the list of conditions eventually will exceed six. But he underscores the importance of identifying and making progress toward achieving NCO.

"If we could acquire and develop the capabilities called for more closely to the time they are needed, how would that change the warfighters' world?"

References

[1] *CMMI for Acquisition, Version 1.2*

Additional Information

To learn about the SEI workshop on interoperable acquisition, visit the Interoperable Acquisition Overview workshop page.

For information about Integration of Software-Intensive Systems, visit the ISIS pages on the SEI website.

For information about NCO, contact us using the link in the *For More Information* box at the bottom of this page.

Suggested Reading

Conditions for Achieving Network-Centric Operations in Systems of Systems

An Emergent Perspective on Interoperation in Systems of Systems

System-of-Systems Governance: New Patterns of Thought

Making SIMPLE Decisions about Software Product Lines

NEWS AT SEI

Author

Paul C. Clements

This library item is related to the following area(s) of work:

Software Product Lines

This article was originally published in News at SEI on: April 1, 2008

Some of the most frequently asked questions about software product lines involve whether there will be a cost benefit to using the product-line approach. So researchers from the SEI, Siemens, the Fraunhofer Institute for Experimental Software Engineering, and Clemson University collaborated on a model that can be used to predict software product line costs and benefits under a variety of real-world situations and that can be used easily by product line decision-makers who may not be skilled in intricate economic theories.

SIMPLE is the Structured Intuitive Model of Product Line Economics, a general-purpose business model that supports the estimation of the costs and benefits in a product line development organization. SIMPLE helps in decisions such as whether to use a product line strategy in a specific situation, the specific strategy to apply, and the appropriateness of acquiring or building specific assets.

SIMPLE has several objectives:

- It must model real situations completely and correctly so that it can give high-fidelity answers to the real problems of organizations.
- It must be sufficiently intuitive for product line personnel to easily produce answers whose derivation can be shown to and understood by others.
- It should be understandable by managers and technicians alike.
- It should be flexible enough to help answer a wide range of questions. In fact, our

Related Links

Training

Software Product Lines - eLearning

Software Product Lines

See more related courses »

model is structured in parts to allow the modeler to select the appropriate levels of detail and model elements to answer a specific question.

- It should not assume any particular approach to software product line engineering beyond the basic tenets implied by the definition of a software product line: "a set of software-intensive systems sharing a common, managed set of features that satisfy the specific needs of a particular market segment or mission and that are developed from a common set of core assets in a prescribed way." [Clements 02]

SIMPLE Functions

SIMPLE approaches the question of how much a particular software product line strategy will cost an organization and how much it gains compared to alternatives. To express these quantities, SIMPLE introduces cost functions and benefit functions that describe these constituents of the overall economic question. Rather than rigorously defined mathematical functions, they should be thought of as an invitation to do a thought experiment to come up with a reasonable cost (or monetary benefit) estimate in each area. This divides otherwise-intractable questions about the costs and benefits of product line strategies into smaller questions that can each be attacked systematically. SIMPLE relies on four basic cost functions:

1. $C_{org}()$—how much it costs an organization to adopt the product line approach for its products. Such costs can include reorganization, process improvement, training, and whatever other organizational remedies are necessary.
2. $C_{cab}()$—how much it costs to develop a core asset base to satisfy a particular scope.
3. $C_{unique}()$—how much it costs to develop the unique parts (both software and non-software) of a product that are not based on assets in the core asset base.
4. $C_{reuse}()$—how much it costs to build a product reusing core assets from a core asset base.

Other cost functions can be introduced as necessary. Each cost function takes a set of parameters that identify factors needed to calculate the cost. For instance, the time period under consideration can be a parameter. This lets us reflect that the cost of a core asset base is higher during product line set-up, and less during product line sustainment.

In addition to cost functions, SIMPLE introduces benefit functions as well. You can use $B_i()$ to signify a particular benefit (such as decreased time to market or increased customer satisfaction) brought about by the approach being considered.

How to use SIMPLE

To use SIMPLE, you must carry out three steps:

1. Create a scenario that you wish to model. The scenario will likely describe two alternative strategies or options that you would like to choose between. An example of a scenario is "An organization has a set of existing stand-alone products undergoing periodic evolutionary updates. Its managers wish to know which is cheaper: (a) converting them to a product line and continuing their evolution in that form, or (b) continuing to evolve them separately and foregoing the cost of setting up the product line." The SIMPLE website has nine pre-defined scenarios that represent typical product line situations. In the future, you will be able to define your own scenarios.
2. Construct a SIMPLE formula that expresses the scenario. Usually, this formula will be of the form (benefit of option 1 - cost of option 1) - (benefit of option 2 - cost of option 2) using the cost and benefit functions to express each term. (For the pre-defined scenarios, the corresponding equations are already constructed on the website).
3. Enter data values for the cost and benefit functions and see the result.

For example, Equation 1 expresses the cost of setting up a product line.

$$C_{org}() + C_{cab}() + \sum_{i=1}^{n} (C_{unique}(product_i) + C_{reuse}(product_i))$$

Equation 1

SIMPLE was recently used by a large European telecommunications company after acquiring two other companies. The company's managers wanted to know if they should merge all three product sets into a common product line. SIMPLE helped them predict that the work required would not pay off, because the architectures of the various product suites were so diverse and incompatible.

The SEI continues to expand SIMPLE by

- pre-defining formulas for more scenarios
- handling uncertainty by, for example, taking a range of inputs rather than a single number, or by accepting a confidence level associated with input data, or by providing Monte Carlo simulation to produce the most likely outcomes given uncertain data
- working to quantify the benefit functions by, for example, appealing to business models for the cost benefit of increased market share or customer satisfaction
- taking into account qualitative relationships among the parameters such as, for example, noting that a large product portfolio usually suggests more complex core assets which in turn suggests higher core asset maintenance costs.

We continue to seek organizations with whom we can work to apply SIMPLE. For more information, contact us using the link in the *For More Information* box at the bottom of this page.

Reference

[Clements 02]
Clements, Paul & Northrop, Linda. *Software Product Lines: Practices and Patterns*. Boston, MA: Addison-Wesley, 2002.

About the Author

Paul Clements is a senior member of the technical staff at the SEI, where he has worked for 10 years leading or co-leading projects in software product line engineering and software architecture design, documentation, and analysis. Clements is the co-author of three practitioner-oriented books about software architecture: *Software Architecture in Practice* (1998; second edition, 2003), *Evaluating Software Architectures: Methods and Case Studies* (2001), and *Documenting Software Architectures: View and Beyond* (2002). He also co-wrote *Software Product Lines: Practices and Patterns* (2001), and was co-author and editor of *Constructing Superior Software* (1999). In addition, Clements has also written dozens of papers in software engineering reflecting his long-standing interest in the design and specification of challenging software systems. He received a BS in mathematical sciences in 1977 and an MS in computer science in 1980, both from the University of North Carolina at Chapel Hill. He received a PhD in computer sciences from the University of Texas at Austin in 1994.

Share This Page

For more information

Contact Us

info@sei.cmu.edu

412-268-5800

First Workshop on CMMI High Maturity Measurement Brings Organizations Together

NEWS AT SEI

Author

Robert W. Stoddard

This library item is related to the following area(s) of work:

Process Improvement

CMMI

This article was originally published in News at SEI on: April 1, 2008

Organizations are increasingly looking for guidance on what it takes to reach high maturity and how to keep improving once they get there. As high-maturity organizations work to improve their use of measurement and analysis, they often look to examples of successful implementations for direction. In response to the need for clarification and guidance on implementing measurement and analysis in the context of high-maturity processes, members of the SEI's Software Engineering Measurement and Analysis (SEMA) initiative organized a workshop at the 2008 SEPG North America conference to bring leaders in the field together at a forum focused specifically on this topic.

Organizations were invited to share information about two important aspects of their measurement and analysis practice: process performance baselining and modeling. Workshop participants were from organizations using process-performance models and baselines. Several representatives from Hill Air Logistics Center, Lockheed-Martin, Northrop Grumman, and Raytheon attended the two-day workshop. "This workshop provided us with a great opportunity to both learn from each other through the sharing of best practices and further build upon our network of fellow practitioners for future collaboration," says Neal Mackertich, a participant from Raytheon.

Workshop goals

The main goals of the workshop were to

Related Links

News

SEI to Co-Sponsor 26th Annual IEEE Software Technology Conference

SEI Cosponsors Agile for Government Summit

Training

See more related courses »

Events

Team Software Process (TSP) Symposium 2014

Nov 3 - 6

- allow CMMI high maturity organizations to share best practices and case studies
- identify ways to develop CMMI high maturity measurement and analysis practices and accelerate their adoption
- enable networking among practitioners

Organizations gave 20-minute presentations summarizing their past experiences and future plans related to the following topics:

- barriers faced by their organizations
- lessons learned in the deployment, training, adoption, and institutionalization of CMMI process performance baselines and models
- best practices and examples of valid, practical methods for implementing process performance models and baselines
- data quality and integrity issues
- plans for modeling over the next three to six months, including the nature of the performance outcomes and drivers most likely to be investigated
- suggestions for subject matter to include in future SEI state-of-the-practice surveys

A sampling of the discussion in each topic is presented in the following section.

Barriers faced

The organizations noted that they face challenges

- establishing the value of developing and using process-performance models (PPMs) and baselines (PPBs)
- convincing project managers to collect new measures to be used for their PPMs and PPBs
- retaining consistent operational definitions as the scope of their measurement and analysis activities expand

Lessons learned

Among the lessons learned, the organizations noted that

- useful PPMs and PPBs require domain and statistical knowledge. Neither alone is sufficient.
- coaching and mentoring are critical elements of the adoption strategy when developing and using the PPMs. This includes guidance on possible decisions and actions associated with results from PPMs and PPBs.

Best practices

Participants discussed what they considered to be best practices and tips for successful implementation of performance models and baselines, including

- providing education and tools to support modeling and analysis
- verifying data integrity before using the data for PPBs and PPMs
- performing product simulation and analysis in addition to process simulation and analysis

Data issues

Data quality and integrity problems that are encountered at lower maturity levels continue to threaten the potential value from PPMs and PPBs. Some examples include

- inconsistent operational definitions, which wreak havoc on modeling attempts
- manual data collection subject to human error
- problems aggregating and disaggregating data
- missing context information to go with the collected data
- consistency problems arising from decentralized databases

(< 5 minute) survey.

Modeling plans

Participating organizations plan to use the following modeling techniques in the next three to six months:

- Bayesian methods to calculate control limits during statistical management
- regression analysis to model and predict customer satisfaction
- measurement system evaluations to identify the degree of noise in data due to the measurement process
- discrete event simulation to facilitate Lean Six Sigma improvements

Subject matter for state-of-the-practice surveys

Workshop participants suggested the following topics of interest for future SEI state-of-the-measurement-practice studies:

- adoption and use of measurement and analysis related to high maturity practices, particularly the use of PPMs and PPBs
- balancing statistical and domain expertise to develop and sustain the value and use of PPMs and PPBs
- bases for choosing critical subprocesses to place under statistical control and ways in which to develop a collection of PPMs
- data archeology (i.e., creating baselines from paper records for previously unmeasured attributes)
- data quality and integrity

Future plans

The discussion during the workshop will be documented in greater detail in an SEI technical note.

The SEI plans to hold high maturity measurement workshops semi-annually to allow invited attendees to continue sharing their experiences and lessons learned in the adoption, development, and use of measurement and analysis in high maturity settings.

At the next workshop later this year, participants from the first workshop and other invited representatives from CMMI high maturity organizations will present their measurement and analysis procedures and initial results. SEI experts will offer additional guidance on high maturity topics and present pertinent results from the 2008 SEI State of the Measurement Practice survey.

Work products resulting from the next workshop will include

- thorough case studies of process performance models and their outcomes in high maturity organizations
- requirements definitions for a possible SEI course on the coaching, adoption, and institutionalization of CMMI process performance models and baselines
- plans for a coordinated empirical study of common performance outcomes and associated controllable and uncontrollable drivers of those outcomes

Subsequent workshops will be open to a larger group of CMMI high-maturity organizations. Organizations wishing to participle in future workshops must be willing to document and share their experiences with the use of measurement and analysis methods in relation to high-maturity practice. To ensure high-value workshops, the SEMA team will screen submissions prior to accepting an organization's request to participate.

Share This Page

For more information

Contact Us

info@sei.cmu.edu

412-268-5800

Process Improvement in a Multimodel Environment Builds Resilient Organizations

NEWS AT SEI

Authors

Lisa Marino

John Morley

This library item is related to the following area(s) of work:

Process Improvement

System of Systems

This article was originally published in News at SEI on: April 1, 2008

"Successful improvement in a multimodel environment is essential to building the resilient organization. And resilient organizations have the agility to achieve and maintain competitive advantage," Mike Phillips of the Software Engineering Institute (SEI)[1] told participants at a recent workshop sponsored by Lockheed Martin Corporation and the SEI.

Organizational growth and evolution test that agility and underscore the need for an integrated approach to using the multiple improvement processes, according to Ray Johnson, chief technology officer of Lockheed Martin Corporation. In his keynote address at the May 8, 2008, Hard Questions for Process Improvement in Multimodel Environments workshop, Johnson remarked that his company plans to bring in as many new employees over the next 10 years as it has today. Some new hires will represent company growth; others, replacements for those leaving the company. All will need to understand how process improvement is central to the company's success.

More than 80 managers, process improvement professionals, technical professionals, and researchers from 40 organizations participated in the workshop. The participants represented U. S. Department of Defense and civilian government agencies, consulting firms, commercial businesses, universities, and research centers. Based on the experience of those participants, three out of every five larger organizations are already

Related Links

News

SEI to Co-Sponsor 26th Annual IEEE Software Technology Conference

SEI Cosponsors Agile for Government Summit

Training

Migrating Legacy Systems to SOA Environments - eLearning

Service-Oriented Architecture: Best Practices for Successful Adoption

See more related courses »

Events

Team Software Process (TSP) Symposium 2014

Nov 3 - 6

facing the challenges of using multiple technologies to meet customer satisfaction, business profitability, market share, product and service quality, cost reduction, and other objectives. Overall, 85% of participants reported that their organizations use one or more of the Capability Maturity Model Integration (CMMI) constellations (CMMI-DEV, CMMI-ACQ, and CMMI-SVC). About half of the organizations represented use an ISO standard, Six Sigma, Lean, or ITIL. In all, the participants brought experience with 30 process improvement standards, frameworks, technologies, models, and practices.

Emphasizing the overarching importance of taking a strategic approach to multimodel process improvement, the workshop featured two rounds of process strategy sessions. In the first round, all of the participants worked in concurrent process strategy sessions discussing scope, solutions, needs, and priorities. The results were then funneled into the second round where two groups continued discussing process strategy and three groups separately discussed process architecture, technology relationships, and implementation and deployment.

From these sessions, participants recommended that organizations in multimodel environments form a strategy that measures process improvement against business objectives and create an architecture that reveals where additional technologies are needed. They called for process model developers and the process improvement community to develop guidance for combining popular technologies and reduce duplication of effort by defining the differences between single-model and multimodel environments. Participants strongly urged developers in particular to work together toward building interoperability into standards, models, and technologies to make their transition into multimodel environments easier.

“We believe our discussions here can help to catalyze conversation between process improvement model developers and their customers about designing interoperability into the models,” Jeannine Siviy of the SEI said.

Align Performance and Business Goals (Sessions on Process Strategy)

The process strategy groups identified the need to align performance strategy with business objectives as the primary focus. They suggested the need for:

- a cookbook for what is needed to address multiple models
- a common terminology
- an open architecture (Process improvement is not about modeling, but how models are implemented.)
- a management paradigm for process improvement comparable to that of the rest of the organization

The group also noted that some solutions for the challenges of developing a strategy for process improvement in a multimodel environment might come from other domains such as service-oriented architecture (SOA).

Combine Technology Elements (Session on Technology Relationships)

A way to combine technologies that delivers the best of each as demanded by the organization is important in a multimodel environment. However, the technology relationships group reported there is a paucity of solutions for combining improvement technologies.

One proposed approach is to use element classification, suggested Patrick Kirwan of the SEI. Improvement technologies can be classified according to their primary element type—good practice, improvement method, or institutionalization. According to the white paper *Improvement Technology Classification and Composition in Multimodel Environments* that was provided to workshop participants:

- Good practice elements define what or how an organization needs to improve.
- Improvement method elements drive the change and facilitate the technology transition processes in the organization.
- Institutionalization elements help an organization in sustaining achieved

improvements.

The technology relationship session called for research to provide:

- practical guidance for combining the top 10 most popular improvement technologies
- an open forum for community discussion
- ways that appraisals and audits can enable convergence of technologies
- a taxonomy for common aspects among models and increased understanding of the differentiating elements between them

Map the Delta (Session on Process Architecture)

The value of process architecture, according to the group, is that it allows an organization to "map the delta"—that is, to determine how to add needed technologies at a higher level where there is greater flexibility in making tradeoff decisions.

At the beginning of the workshop, M. Lynn Penn, director of quality systems and process management at Lockheed Martin, presented a case study attesting to the value of creating a process architecture. According to Penn, Lockheed Martin's Information Systems and Global Services (IS&GS) organization of 10 companies and 52,000 employees uses process architecture to "instantiate compliance to industry standards via a single organizational standard process."

Among the benefits that this approach has delivered for IS&GS and Lockheed Martin are a "30% cycle-time reduction, a 20% lowering of software costs" and a robustness that has made it "easy to build in new models and practices," according to Penn.

The process-architecture group suggested that more examples like Penn's experience are needed. Such case studies would help to draw the distinction between architecture and a mere collection of processes. The group also identified a need for languages, methods, and tools to define an architecture and subsequently define the processes that are ready for users.

Reduce and Minimize Duplicate Efforts (Session on Implementation Issues)

A multimodel environment challenges the process-improvement engineers and developers charged with implementation because of more complicated sharing of audit and appraisal processes and data, greater coordination in training, and potentially overlapping roles and responsibilities, among other aspects. Those kinds of challenges can duplicate effort and foster waste.

To gain understanding about how to reduce duplication and waste, the implementation working group recommended that research be conducted in several areas, including the following:

- cost-effective and transparent means to get integrated feedback
- case studies that document the cost comparison between multimodel and single-model environments
- an integration framework that shows Venn diagrams of various models and standards in public domain

Multimodel Environment Issues are Pervasive

Workshop participants emphasized that issues in process strategy, technology relationships, process architecture, and implementation and deployment arising in multimodel environments cut across organizational levels, roles, responsibilities, and practices. The Process Improvement in Multimodel Environments (PrIME) project that the SEI proposes to lead will span the range of those topics and others in establishing an approach to harmonizing process-improvement technologies in multimodel environments. "Multimodel improvement is our reality," Siviy said. "Our processes should be engineered with the same attention we bring to the engineering of our products."

For Further Reading

Visit http://www.sei.cmu.edu/prime/ for the workshop presentations—including the proposed SEI approach to model harmonization—and the white paper series that examines problems organizations encounter when operating in multimodel environments and the current process improvement approaches such organizations must consider.

PrIME Project Sponsorship

The SEI is looking for organizations now to fund the PrIME efforts for the full three years of the project. Project sponsors enjoy benefits such as influence on the order in which models and research themes are addressed, royalty-free license for the duration of the project, one-on-one implementation meetings with the SEI, early access to research outputs, and participation in selected workshops. For more information on sponsorship opportunities, contact us using the link in the *For More Information* box at the bottom of this page.

1 Along with Jeannine Siviy and Patrick Kirwan, Mike Phillips is the co-technical lead for the Process Improvement in Multimodel Environments (PrIME) project. The SEI is proposing PrIME as a three-year effort to establish an approach to harmonizing process improvement technologies in multimodel environments.

For more information

Contact Us

info@sei.cmu.edu

412-268-5800

Place of Architecture in a Crowdsourced World, The

NEWS AT SEI

Author

Rick Kazman

This article was originally published in News at SEI on: May 1, 2008

Yochai Benkler, in his book *The Wealth of Networks*, puts forth a provocative argument: that we are in the midst of a radical transformation in how we create our information environment. This change is at the heart of the open-source software (OSS) movement but OSS is only one example of how society is restructuring around new models of production and consumption of services. The aspect that is most startling "is the rise of effective, large-scale cooperative efforts—peer production of information, knowledge, and culture ... . We are beginning to see the expansion of this model not only to our core software platforms, but beyond them into every domain of information and cultural production" [Benkler 06]. The networked information environment has dramatically transformed the marketplace, creating new modes and opportunities for how we make and exchange information. "Crowdsourcing" is now used for creation in the arts, in basic research, and in retail business [Howe 06]. These changes have been society-transforming.

So what is the place of architecture in a crowdsourced world? Crowdsourced systems are created via *commons-based peer production* [Benkler 06]. A "commons" is the opposite of property; the term refers to a set of shared, accessible community resources. Peer production is production that harnesses the creative energies of many self-selecting participants without any financial compensation and lacking a formal managerial structure. The importance of this form of production is undeniable: according to Alexa.com, five of the 10 most popular websites in the world are produced this way (MySpace, YouTube, Facebook, Wikipedia, and Blogger), and with the exception of Wikipedia, all are for-profit enterprises.

There are a number of characteristics of crowdsourced systems—observed in the SEI ultra-large-scale (ULS) systems report [Northrop 06] and in our own surveys of websites and OSS projects—that challenge existing models of system development.

Software engineering has long embraced a centralized production model, where requirements are collected and negotiated, projects are managed, architectures are created, and correctness is determined in a controlled, planned process. It is hierarchical and rule-oriented, not commons-based or egalitarian. Even Agile methods are centralized, stressing the importance of face-to-face communication and the advantages of the bullpen—a single open office where workers freely interact.

Crowdsourced systems, however, are community driven and de-centralized with little overall control [Mockus 02]. Consequently we can no longer design and implement such systems using older models. If systems are constantly in a state of "perpetual beta" [O'Reilly 05], if they derive value from being constantly updated and combined in novel ways, and if their value is in their comprehensiveness and ubiquity, then the new model must reflect this. Examples of fundamental shifts in the logic for system development are:

- *Open teams*: assumptions of a closed team of developers who work from a consistent set of requirements must be abandoned. "Based on our usual assumptions about volunteer projects and decentralized production processes that have no managers, [Linux] was a model that could not succeed. But it did," Benkler observes [Benkler 06] Even in the for-profit world, the assumption of closed teams is outmoded: as Peter Drucker observed a decade ago, managers must lead and motivate knowledge workers as though they were unpaid volunteers, including them in strategic direction and governance [Drucker 98].
- *Mashability*: Enormous effort traditionally goes into making systems that are difficult to tear apart, for historical, intellectual property and security reasons. However, mashability is as a core capability of crowdsourced systems. Web browsers make it simple to view source any page's source, and it is accepted practice to use parts of existing websites in new creations. For example, Google Maps, prior to making its APIs public, was already used by others in their mashups.
- *Conflicting, unknowable requirements*: While iterative life cycles accept that requirements will change, they still operate under the assumption that, in any given iteration, a team can collect and analyze those requirements. However requirements in a peer-produced system *emerge* from its individuals, operating independently.
- *Continuous evolution*: As a consequence of having constantly changing requirements and non-centralized resources, a peer-produced system is never *done*, and hence it is never stable. The term "perpetual beta" was coined to describe this new phenomenon. One can not conceive of its functionality in terms of *releases* any more than a city has a release. Parts are being created, modified, and torn down at all times. In other words, we must accept *change as a constant*. Wikipedia entries, Facebook or Orkut applications change from day to day; OSS projects employ a continuous build process [Mockus 02].
- *Focus on operations*: Historically lifecycle models have focused on *development* and maintenance as the activities of interest. However much of the value of peer-produced systems is that they are as reliable and accessible as public utilities. Clearly Google, eBay, Amazon and other popular websites have taken this lesson seriously.
- *Sufficient correctness*: Completeness, consistency, and correctness are goals that are, to varying degrees, anathema to peer produced systems. For example, collaborative tagging—while enormously valuable for the semantic web—does not depend upon consistency among the taggers. Wikipedia never claims to be complete or even fully correct. Similarly "perpetual beta" is an admission and acceptance of ongoing incompleteness in software [O'Reilly 05].
- *Unstable resources*: Applications that are peer-produced are subject to the whims of the peers. Resources—people, computation, information, and connectivity—come and go. Mockus et al, describing OSS development, noted that such systems "are built by potentially large numbers of volunteers ... . Work is not assigned; people undertake the work they choose to undertake." [Mockus 02]. However, large numbers tend to ameliorate the whims of any individual or individual resource.

- *Emergent behaviors*: large-scale systems—computational and biological—exhibit emergent behaviors. This has been noted in traffic patterns, epidemics, computer viruses, and systems of systems [Fisher 06]. Certainly large-scale, web-based applications such as Second Life, eBay, and MySpace have seen complex behaviors emerge that were beyond the vision and intent of their creators, such as Second Life's tax revolt and the recent eBay seller boycott.

Clearly this new environment requires a new logic for development. To better understand this logic, we distinguish three realms of a crowdsourced project and some example roles within each realm: *kernel* (architects, business owners, policy makers), *periphery* (developers, producers/consumers), and *masses* (customers, end users):

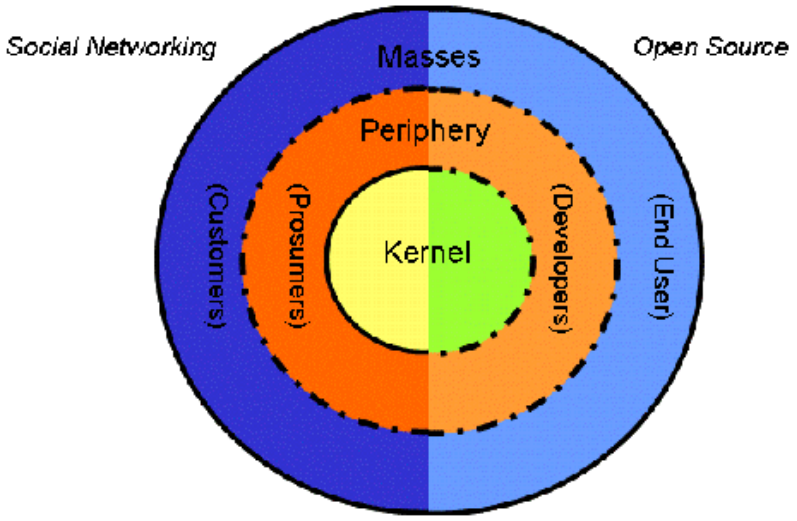

Figure 1: The realms of crowdsourced projects

As the figure indicates, there may be differences in the permeability between the realms. For example, in OSS it is possible to move from the role of an end user to a developer to a kernel architect. In social networking it is effectively impossible for a prosumer to become part of the kernel. Given this model, what is the role of architecture?

The architecture must be divided into a *kernel* infrastructure and a set of peripheral services, and these are created by different communities using different processes. Kernel services—like the kernels of Linux and Perl, the Apache Core, Wikipedia's wiki, or Facebook's application platform—are designed and implemented by a select set of highly experienced and motivated developers who are themselves intense users of the product. These kernel services provide a platform on which subsequent development is based (like Linux's kernel), a set of zoning rules (like the Internet's communication protocols), or both (like Facebook's application platform). The kernel must be highly modular; this allows a project to scale as its community grows while allowing an original visionary developer or team to retain intellectual control [Northrop 06]. The kernel provides the means to achieve and monitor quality attributes such as performance, security, and availability. The design of the periphery is enabled by and constrained by the kernel, using its services and complying with its protocols; but the periphery is otherwise unspecified. This lack of specification permits the unbridled growth and parallel creation at the periphery.

Similarly,requirements must be bifurcated into:

- kernel-service requirements that, in and of themselves, deliver little or no end-user value (Linux's kernel, Wikipedia's wiki, Facebook's open platform, BitTorrent's P2P network)
- periphery requirements that are contributed by the peer network (the prosumers) which deliver the majority of the end-user value: YouTube videos, Wikipedia

entries, Firefox add-ons, Facebook applications.

The nature of the requirements in these two categories is different: kernel service requirements are about quality attributes and their tradeoffs while periphery requirements are about end-user perceivable functions.

Finally, implementation is also bifurcated: the vast majority of implementation is crowdsourced but the crowdsourcing model applies *only* to the periphery. A distinct group needs to implement the kernel and this group will be close-knit and highly motivated. As Mockus has noted of OSS projects: "developers are working only on things for which they have a real passion" [Mockus 02]. The periphery will develop at its own pace, to its own standards, using its own tools, releasing code as it pleases.

What are the implications of this model on software development? For some projects, there are no implications. Not all projects will take advantage of crowdsourcing. Some projects will be deemed high security, or highly proprietary, or simply have too much legacy to take advantage of this model. However, there is an increasingly important class of projects for which this model applies. And we need to understand, plan for, and analyze the architectures of such systems. In those projects the kernel architecture must be built by a small, experienced, motivated team that focuses on modularity, core services, and core quality attributes to enable the parallel activities of the periphery.

References

[Benkler 06]
Benkler, Y. *The Wealth of Networks: How Social Production Transforms Markets and Freedom*. New Haven, CT: Yale University Press, 2006.

[Drucker 98]
Drucker, P. "Management's New Paradigms." *Forbes*, Oct. 5, 1998, 152-177.

[Fisher 06]
Fisher, D. *An Emergent Perspective on Interoperation in Systems of Systems* (CMU/SEI-2006-TR-003). Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2006.

[Howe 06]
Howe, J. "The Rise of Crowdsourcing." *Wired 14* (June 6, 2006).

[Mockus 02]
Mockus, A.; Fielding, R.; & Herbsleb, J. "Two Case Studies of Open Source Software Development: Apache and Mozilla." *ACM Transactions on Software Engineering and Methodology 11*, 3 (July 2002): 309-346.

[Northrop 06]
Northrop, L.; Feiler, P.; Gabriel, R.; Goodenough, J.; Linger, R.; Longstaff, T.; Kazman, R.; Klein, M.; Schmidt, D.; Sullivan, K.; & Wallnau, K. *Ultra-Large-Scale Systems: The Software Challenge of the Future*. Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University, 2006.

[O'Reilly 05]
O'Reilly, T. "What is Web 2.0: Design Patterns and Business Models for the Next Generation of Software."

About the Author

Rick Kazman is a senior member of the technical staff at the SEI, where he is a technical lead in the Architecture Tradeoff Analysis Initiative. He is also an adjunct professor at the Universities of Waterloo and Toronto. His primary research interests within software engineering are software architecture, design tools, and software visualization. He is the author of more than 50 papers and co-author of several books, including the book titled *Software Architecture in Practice*. Kazman received a BA and MMath from the University of Waterloo, an MA from York University, and a PhD from Carnegie Mellon University.

The views expressed in this article are the author's only and do not represent directly or imply any official position or view of the Software Engineering Institute or Carnegie Mellon University. This article is intended to stimulate further discussion about this topic.

For more information

Contact Us

info@sei.cmu.edu

412-268-5800

PACC Starter Kit Provides Tools for Developing Systems that Exhibit Predictable Behavior

NEWS AT SEI

This library item is related to the following area(s) of work:

Predictability by Construction

This article was originally published in News at SEI on: May 1, 2008

Many software systems have stringent quality attribute requirements. For example, industrial robots must perform tasks with strict deadlines, medical devices must comply with safety requirements, and most software must minimize security vulnerabilities. Although analysis theories and techniques to satisfy these requirements have existed for many years, they are not widely used because of the resources and expertise required to create, maintain, and evaluate analysis models. Even when such theories are used, doubts often remain about the accuracy of the results because of uncertainty of the correspondence between the models that are analyzed and the code that will be executed.

The PACC Starter Kit (PSK) is an integrated set of software development tools that demonstrates how existing technologies can be integrated to provide objective confidence in predictions of system behavior. “Our primary intent is to provide working examples and building blocks that help organizations get started with integrating such technologies in their own development environments,” says James Ivers, a researcher at the SEI and one of the developers of the PSK.

“The PSK is an Eclipse-based development environment for Windows that combines a model-driven development approach with reasoning frameworks that apply analyses to predict runtime behavior based on specifications of component behavior and are accompanied by some measure of confidence,” says Ivers.

Often, quality attribute requirements in areas such as performance, security, or safety are some of the most difficult system behaviors to get right. Effective techniques to provide early confidence in the ability to satisfy such requirements usually include some form of architectural or design analysis that exploits knowledge of the relevant quality attributes.

(< 5 minute) survey.

Unfortunately, there often remains a gap between architectural concepts and the implementations that will actually be deployed. And that gap leads to uncertainty as to whether the qualities designed into an architecture will ultimately be realized in the executing software.
The PSK shows how a collection of today's technologies can be integrated to mitigate this risk. Key features of the concepts integrated in the PSK include

- reducing the gap between architecture and code by using a component technology that mirrors the architectural idioms and enforces key constraints on software developers that ensure applicability of associated quality attribute analyses
- packaging technologies for quality attribute analysis into reasoning frameworks that can be used to automatically predict system behavior without requiring users to become theory experts
- providing measurable confidence that architectural analysis results will be consistent with executing software behavior, through either statistical confidence labels or automated proof-generation techniques

The PSK includes online tutorials that guide a user through its use on a number of examples. According to Ivers, the most interesting example is the audio-mixing application, which is built from components that include a signal generator, WAV decoder, splitter, adder, inverter, and graphical display. Ivers explains that "the audio example has some of the same quality attribute requirements as larger, more complex systems, but is a small application that is easy to understand. Problems tend to be readily observable, either audibly or by using the included graphical display."

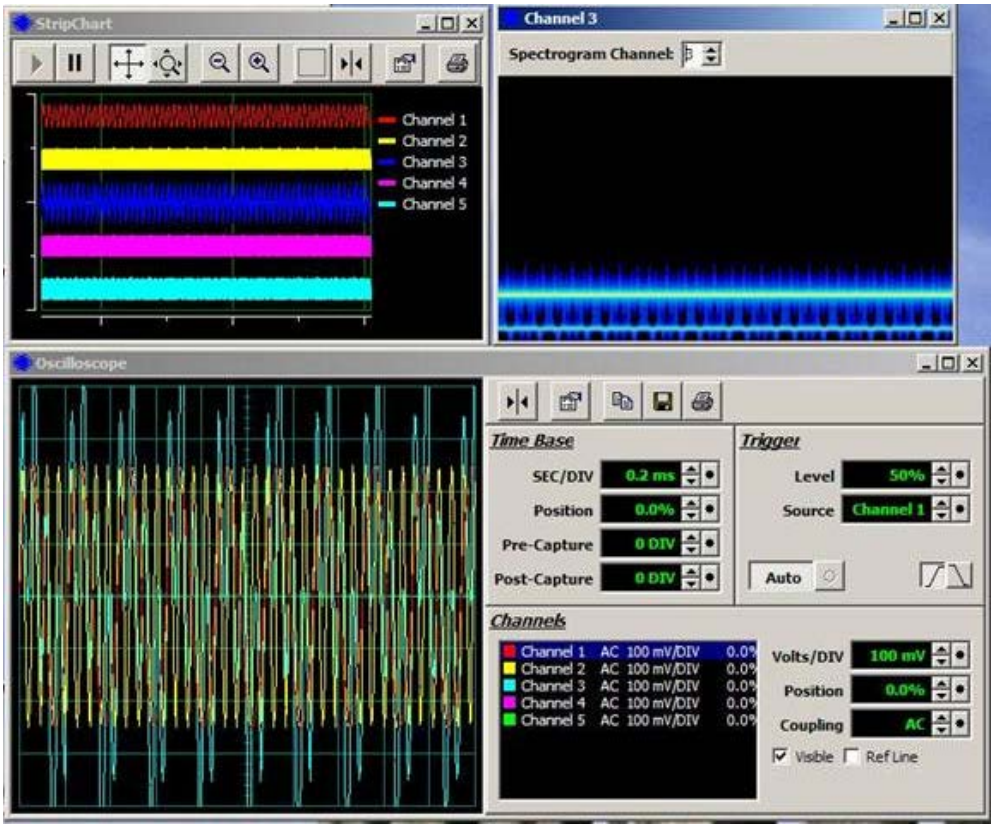

Figure 1: Graphical display of signals processed in a multiple-channel use of the audio-mixing application

The tutorial demonstrates how the behavior of individual components is specified and how the architecture of the mixer is depicted in the PSK. It shows how code for the included component technology is generated from component specifications and how that code is deployed and executed. It also shows how the included reasoning frameworks are applied to provide quality attribute analyses; two of the analyses that are demonstrated for this example are

- performance analysis: To avoid artifacts like skips in the audio output, all processing for a given frame of input data must be completed within a specific timeframe. The tutorial demonstrates how to supply the information needed to perform this analysis (e.g., assignment of thread priorities and how to measure execution time) and how to perform the analysis using one of several evaluation procedures.

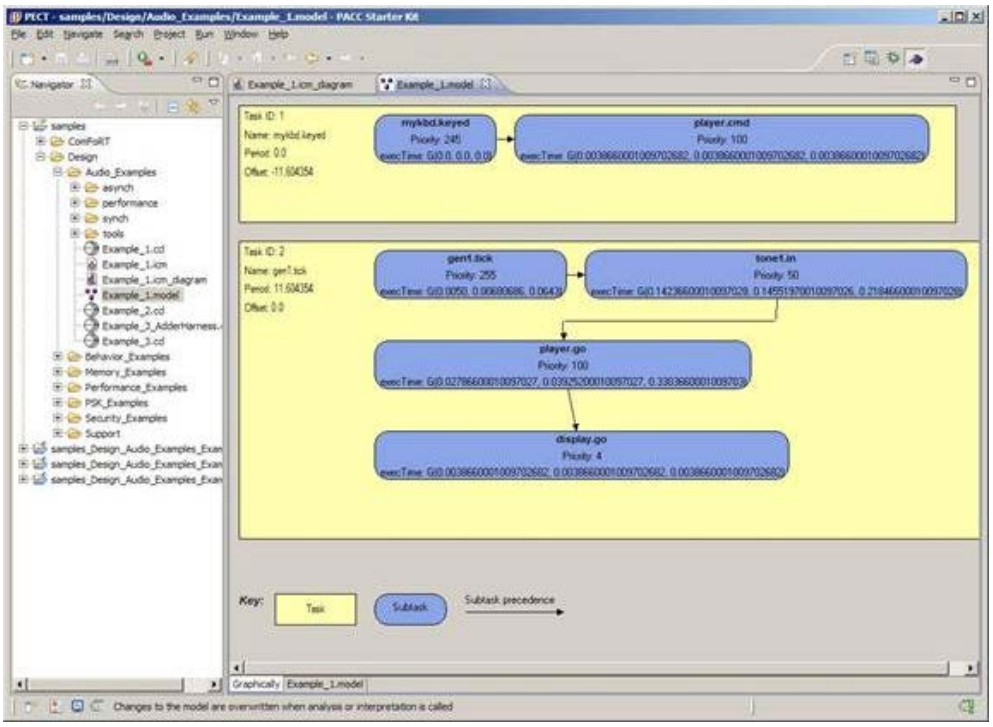

Figure 2: The performance model that is generated for one of the audio mixer application

This view is the result of information extracted from the specification of the audio application; this form is more suitable for analysis by the various performance evaluation procedures found in the PSK.

- behavior analysis: For the user to hear the right audio output, the application must follow important behavioral rules. For example, components must follow rules about their interactions with each other, such as acknowledging the reception of information in protocols that require acknowledgement, even under error conditions. The tutorial demonstrates how users can define these rules and check for any violations.

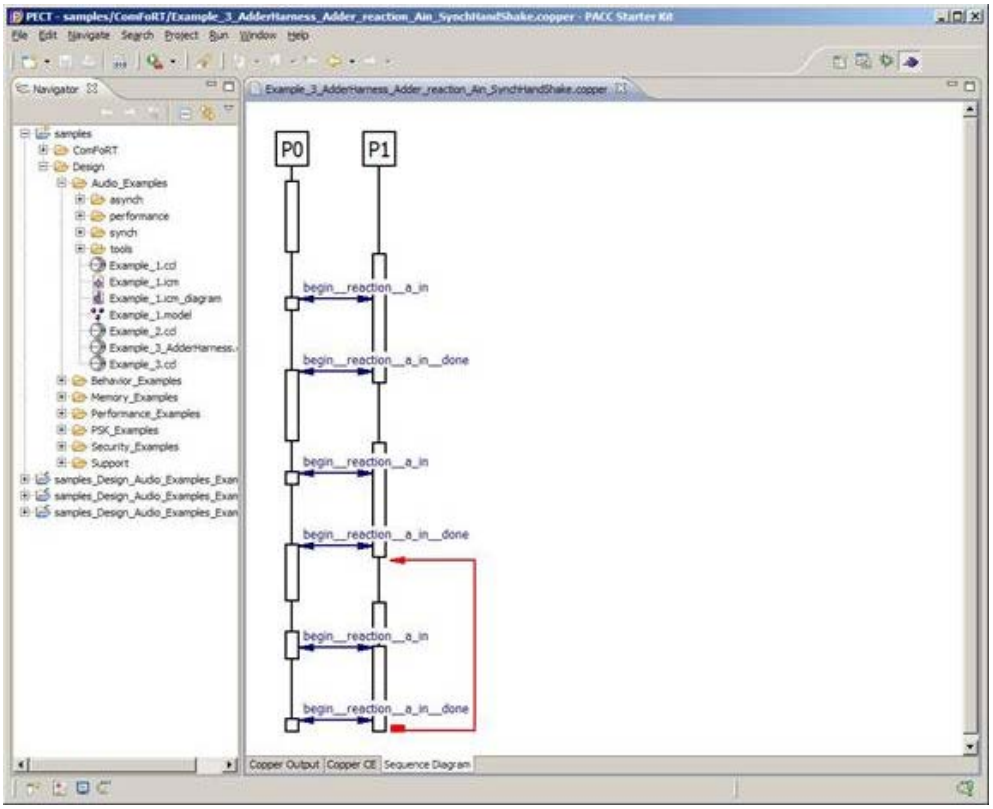

Figure 3: Sequence diagram demonstrating a specific execution in which the specification fails to follow user-supplied rules about inter-component communication

This is one example of the type of evidence supplied to give users a reason to believe analysis results.

Ivers explains how these examples are used: “You can use them to see how problems can be detected early, modify the architecture or the details of individual components to correct problems, re-analyze the results to confirm fixes, and confirm that the executing system exhibits the qualities indicated by the analyses.”

The PSK is intended as a vehicle for demonstrating how these concepts can be integrated in practice. The PSK, now available for download integrates a collection of technologies that include

- a design language based on UML statecharts for describing component behavior and descriptions of how components are assembled
- a code generator for creating software implementations ready to execute in the provided execution environment (the Pin component technology and a real-time extensions layer for Windows) that enables you to go from software designs to executing examples
- several automated quality attribute analyses
  - performance analysis for worst-case, average-case, and sporadic-server latency
  - behavior analysis for user-specified claims of software behavior (e.g., satisfaction of application-specific invariants, conformance to interaction protocols, or absence of deadlock)
  - security analysis for detecting buffer overflows in C code
  - memory analysis for anticipated resource use
- accompanying means of demonstrating objective confidence. For example, statistical measures of confidence in performance analysis are based on data

supplied by measurement tools included in the kit. In the behavior analysis, execution traces are used to demonstrate failure, and proof generation is used to demonstrate success.

To learn more, visit the PSK pages or contact us using the link in the For More Information box at the bottom of this page.

For more information

Contact Us

info@sei.cmu.edu

412-268-5800

Architecture Competence: What Is It? How Do We Measure It?

NEWS AT SEI

This article was originally published in News at SEI on: May 1, 2008

Software architecture is the single most important software artifact determining the success of a software system, agree many experts. That's why so many researchers have examined its technical aspects and offered tools and methodologies toward making it better. What they hadn't investigated were the people wielding these technologies—architects—and the human and organizational factors necessary to producing sound architecture. Such investigation, determined five SEI researchers, might help define architecture competence and enable its measurement and improvement. Paul Clements, Len Bass, Rick Kazman, Mark Klein, and John Klein, all members of the SEI Software Architecture Technology (SAT) Initiative, undertook the study. For years this team had taught the architecture-centric practices that they had defined for creating quality systems. Now they wanted to know the characteristics of the architect and organizational environment that expedite these practices. Their goals were to

- identify the measurable factors that contribute to architecture competence in individuals and organizations
- develop an instrument for evaluating and an approach for improving these factors

To achieve these goals, Clements, Bass, Kazman, Klein, and Klein first examined four models of performance competence, discussed below, that they could adapt to software architecting.

The Duties, Skills, and Knowledge (DSK) Model

To establish the DSK model, the investigators set out to identify the key duties, skills, and knowledge that a competent architect must possess. They gleaned their data from about 200 information sources targeted to the practicing architect, such as books, courses, and thousands of job descriptions. The duties, skills, and knowledge areas most frequently encountered are listed below.

- duty areas: architecting, other development-cycle phases, interacting with stakeholders, management, organization- and business-related duties, leadership, and team building

- skill areas: communication, interpersonal, work, and personal
- body of knowledge topics: basic software engineering, people, business, architecture techniques, requirements engineering, software project management, programming, platform technology, systems engineering, architecture documentation, reuse and integration, domain knowledge, and mentoring

Organization of the data according to the DSK model would provide valuable structure and material for a future assessment instrument.

The Human Performance Technology (HPT) Model

The concept underlying the HPT model is that competent individuals produce valuable results at a reasonable cost. This model evolved from the human engineering work of Thomas Gilbert, who believed that competence is usually hindered by inadequate performance support at work, rather than by an individual's lack of knowledge or skill [Gilbert 1996]. This approach expresses the worth of an individual's performance as a ratio of the value of the performance to its cost. One challenge here is to determine the value of quality architecting, some of which lies in the avoidance of costly problems over the life of the system. Isolating the various duties identified in the DSK model might be a first step towards solving this problem, reasoned the team. Another challenge is to determine the infrastructure required for calculating such worth. The team views these issues as important areas for further research.

The Organizational Coordination Model

The organizational coordination model involves the sharing of information among organizational members and teams. The researchers focused on coordination and communication activities necessitated by particular types of architecture decisions. For example, when each of several teams is developing a different module in a software system, an architecture decision that results in dependencies among modules will require increased coordination between module developers. The SEI researchers also hoped to learn the effectiveness of mechanisms for facilitating coordination, such as shared discussion boards and engagement of intermediaries. These mechanisms can be useful for measuring coordination activity (for example, by the number of discussion board posts required to solve a problem). How well the coordination requirements brought on by an architecture are met by the organization's coordination capability reflects the organization's architectural competence.

The Organizational Learning Model

Finally, the organizational learning model of competence assumes that just as individuals can learn, so can organizations. This learning is evident when change in the organization occurs as a function of experience and is observable in the organization's knowledge, practices, or performance. Competence lies in an organization's ability to convert experience into knowledge through "mindfulness"; conducting architectural reviews, lessons learned, or analyses on completed projects exemplifies such mindfulness. The organization will strive to understand which learning processes are best suited for different types of learning and how various types of experience affect the transfer of such experience into knowledge. An architecturally competent organization performing architecture-centric practices, for example, will recognize the learning opportunities contained in those practices and conduct postmortems or comparisons with previous projects to maximize learning. Organizational learning is measurable through questionnaires and surveys.

Exploring the models proved an effective strategy for this study. Between them, the four models (1) covered a continuum of observational possibilities that would apply well to individuals, teams, or whole organizations, and (2) offered principles that examined past performance as well as present activity. These are valuable characteristics for developing useful instruments and assessing a wide spectrum of competency factors. "We found that together the four models provide strong coverage across these important dimensions, giving us confidence that they are effective choices for informing an evaluation instrument." This statement is from the SEI technical report Models for Evaluating and Improving Architecture Competence that discusses the study in detail.

The team has developed survey questionnaires by engaging both a top-down and a

(< 5 minute) survey.

bottom-up approach towards assessment. To quote the report:

> *We have generated questions from a knowledge of the place of architecture in the software development and system development life cycles. For example, we know that architectures are critically influenced by quality attribute[1] requirements, so questions in the instrument must probe the extent to which the architect elicits, captures, and analyzes such requirements. In the bottom-up approach, we examine each category in the models and generate questions that address each component. This approach leads to tremendous overlap, which helps to validate and refine the questions in the instrument.*

These questions map straightforwardly to the four models and lay the foundation for future competence assessment instruments. Depending on how they are administered and used, such instruments and assessments promise to serve at least three groups:

1. Acquisition organizations should find that architecture competence assessment can help in evaluating a contractor or in choosing among competing bids. Hiring the more architecture-competent contractor typically brings fewer future problems and less reworking [Boehm 2007].
2. Service organizations should benefit from maintaining, measuring, and advertising their architecture competence to attract and retain customers. Objective assessments of their competence levels by outside organizations would strengthen their clients' trust.
3. Development organizations could assess, monitor, and then increase their levels of architecture competence, thus benefiting the advertising of their products' quality and improving their internal productivity and predictability.

The team's work has opened an area that is clearly poised to expand. In June the SEI invited interested researchers from several countries to a workshop on architecture competence that resulted in a rich exchange of ideas. These concepts are being integrated into the team's assessment work to best benefit organizations that desire evaluation and improvement. For more information contact us using the link in the For More Information box at the bototm of this page.

References

[Boehm 2007]
Boehm, B., Valerdi, R., & Honour, E. "The ROI of Systems Engineering: Some Quantitative Results." *Proceedings of the Seventeenth International Symposium of the International Council on Systems Engineering (INCOSE).* San Diego, CA, June 2007. INCOSE, 2007.

[Gilbert 1996]
Gilbert, Thomas F. *Human Competence: Engineering Worthy Performance*. Washington, DC: International Society for Performance Improvement, 1996.

[1] *Quality attributes* are qualities such as modifiability, security, and performance that must be built into the system to fulfill stakeholder requirements.

Contact Us
info@sei.cmu.edu
412-268-5800

Requisite Agility

NEWS AT SEI

Authors

Bill Anderson

Philip J. Boxer

This library item is related to the following area(s) of work:

System of Systems

This article was originally published in News at SEI on: May 1, 2008

The ability of software-intensive systems or organizations to respond rapidly to changing demand is a key determining factor in their achieving business or mission objectives [Alberts 03]. Because of the primacy of achieving objectives to all organizations—in the U.S. Department of Defense (DoD), civilian government, and industry—understanding of the agility needed to respond to changing demand is a key challenge on the research agenda of the SEI Integration of Software Intensive Systems (ISIS) initiative. The ISIS team is developing the SoS Navigator, a growing set of modeling techniques that offer insights into the relationship between systems or organizational structure and agility.

Related Links

Training

Migrating Legacy Systems to SOA Environments - eLearning

Service-Oriented Architecture: Best Practices for Successful Adoption

See more related courses »

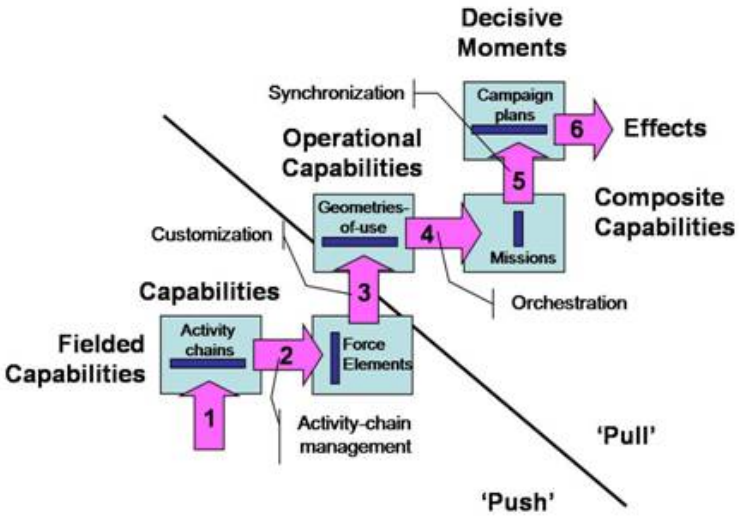

Figure 1: Modeling Requisite Agility

Figure 1 labels the outputs of the various matrix groupings according to a military context; hence fielded capabilities, operational capabilities, and so on. These labels change to fit the domain of interest (e.g., perhaps protocols, modalities, treatment plans, and the like in the medical domain).

The contrast between traditional systems engineering and system-of-systems approaches highlights the different nature of response to demand. Following traditional systems-engineering precepts and practices, a supplier organization composes fielded capabilities—products or services—in response to a requirements-driven process. This activity is represented on the "push" side of Figure 1. Many traditional systems suppliers stop with a push approach, defining their relationship to their customers in terms of supplying fielded capabilities (arrow 2). In such a supply-side model, the use of their products or services to create operational capabilities (arrow 3) and ultimately to accomplish end effects (arrow 6) is left to the customer's organization.

Unlike traditional systems engineering practices, the SoS Navigator approach makes use of distributed collaboration that requires building an explicit awareness of the demand or "pull" implications of the customer's organization. On this *pull* side of the model depicted in Figure 1, composite capabilities (arrow 4) put together by a customer are synchronized (arrow 5) to create the decisive moments required to produce the effects (arrow 6). Keeping with the military context, a decisive moment might be a sequence of events that a force is trying to effect or to prevent. In the medical domain, decisive moments might be a sequence leading to a treatment option such as medication or therapy [Boxer 08]. The composite capabilities are in turn orchestrations or arrangements of operational capabilities (arrow 3).

These orchestrated operational capabilities (arrow 4) hold a special place at the nexus of the supply- side *push* and the demand-side *pull* activities. We call these orchestrations "geometries-of-use." Geometries-of-use are the particular ways in which capabilities need to be put together to meet demand. The variability across these geometries-of-use defines the **requisite agility** of the system of systems—that is, the system of systems is required to provide these geometries-of-use (as specified in the matrix generating arrow 4) to be able to respond to the decisive moments addressed by arrow 5.

Geometries-of-use are the units of requisite agility, due to the pivotal push-pull (supply-demand) paradigm shift that is so critical to our conceptual framework [Boxer 06]. Brewer et al. define requisite agility as the capacity to innovate quickly in the face of rapid technological change [Brewer 06]. They derive it from Ashby's cybernetics work where he defined "requisite variety" as [paraphrasing] "the capacity of a biological system to regulate or adapt to an environment, [and] if the focal biological system is attempting to regulate the behaviors of others in a common environment, then the variety of moves must equal or exceed theirs" [Ashby 63]. Brewer et al. go on to assert from "Ashby's law" that "it is not enough for an agency to have a sufficient

variety of moves. It must also be able to execute these moves quickly enough to be effective" [Brewer 06].

Requisite agility is one of the SoS Navigator's fundamental concepts.[1] It allows demand-side behaviors to be arranged according to demand-in-context, instead of around the capabilities on the push side of the model (i.e., capabilities of constituent systems/organizations that are likely to be over-determining or design-time constraining). When the system of systems can support a variety of geometries-of-use on the demand-side, it benefits from an infrastructure that is more flexible and adaptive to anticipated and unanticipated demands. In other words, its infrastructure has the agility needed for its use to be determined closer to runtime.

While it might seem to be a worthy goal, it is not sufficient to have a stockpile of geometries-of-use against the possibility of facing different forms of demand. Rather, it is necessary to have the agility to respond quickly to variations in demand. This is not to say that we are proposing automation of this entire model into a real-time adaptive super machine. We are simply advocating that the socio-technical processes required to respond to changing demand can be described (modeled) and better equipped to handle change if the organization is driven from a demand-side perspective rather than a supply-side perspective of the constituent parts.

At the same time, however, we are excited about the potential for software services acting as constituent parts to automate more and more of the existing geometry-of-use space in order to create new possibilities on the demand side. These techniques or models allow us to reason about the granularity of those software services, the support structures required, reuse potential, cost of alternatives, prioritization of value, labor verses automation tradeoffs, and interoperability risks—what we call SoS Navigation.

[Alberts 03]
Alberts, David S. & Hayes, Richard E. *Power to the Edge: Command and Control in the Information Age*. CCRP, 2003.

[Ashby 63]
Ashby, W. R. *Introduction to Cybernetics*. New York: John Wiley, 1963.

[Boxer 08]
Boxer, Philip, Morris, Edwin, Anderson, William, & Cohen, Bernard. "Systems-of-Systems Engineering and the Pragmatics of Demand," 341–347. *Proceedings of the 2008 IEEE International Systems Conference*. Montréal, Québec, Canada, April 7–10, 2008. IEEE, 2008.

[Boxer 07]
Boxer, Philip, Morris, Edwin, Smith, Dennis, & Anderson, Bill. "The Double Challenge in Engineering Complex Systems of Systems." *Eye on Integration, news@sei 2007,* 5.

[Boxer 06]
Boxer, Philip. *The Three Agilities*. January 1, 2006.

[Brewer 06]
Brewer, Gene A., Neubauer, Bruce J., & Geiselhart, Karin. "Designing and Implementing E-Government Systems: Critical Implications for Public Administration and Democracy." *Administration & Society 38,* 472 (2006): 472–479.

[1] Other fundamental concepts are the SoS enterprise and the purchaser-provider boundary; supply, demand, and organizational context; the gap between supply and demand; context-of-use; the double challenge of governance and demand; and the implications of decentralized governance. All of these concepts are explored in an upcoming SEI technical note, *SoS Navigator 2.0: A Context-Based Approach to System-of-Systems Challenges* (CMU/SEI-2008-TN-001); the double challenge has also been examined in a previous column [Boxer 07].

About the Authors

Bill Anderson is a senior member of the SEI technical staff. Bill's research interests include integration and interoperability of complex software systems, COTS and reuse

management, cost estimation, and business case justification of complex systems. A former Vice President for a Fortune 500 company, Bill is broadly experienced with factory floor and business; processes, support systems, automation, and management. He has many years of experience in large system project management and has successfully led operational, financial, product line, and new product launch groups.

Phil Boxer has been a strategy consultant to organizations since the late 1970s, supporting leadership teams across many different industry sectors, both public and private. His focus is on the challenges organizations face from asymmetric forms of demand and the mitigation of risks associated with failing to develop requisite agility. He has developed a number of supporting methods and tools needed to support strategy formation and collaborative design processes, including visual PAN and its associated forms of analysis. He is a senior member of the technical staff of the Integration of Software-Intensive Systems (ISIS) team at the Software Engineering Institute.

The views expressed in this article are the author's only and do not represent directly or imply any official position or view of the Software Engineering Institute or Carnegie Mellon University. This article is intended to stimulate further discussion about this topic.

Building More Secure Software

NEWS AT SEI

Author

Julia H. Allen

This article was originally published in News at SEI on: June 1, 2008

This column is based on a podcast recorded with Julia Allen and posted to CERT's Podcast Series: Security for Business Leaders. Julia Allen is a senior researcher with CERT, working on security governance and software assurance. Bill Pollak, who interviewed her, is the manager of communications for the SEI.

Part 1: Software Security Is Just Good Business

Bill Pollak: Generally we think of security as an operational IT issue focused on defending our computers and networks from attackers and from security breaches; or we think of security as information security, concerned with protecting information in digital form.But in this conversation we're discussing software security.What is software security and how is it different from IT or information security?

Julia Allen: In much of our work at CERT and in the software and IT communities at large, we're all familiar with IT security—defending networks and defending systems from attack. With all of the compliance issues that are emerging today around the protection of personally identifiable information, we're all becoming much more sensitive to issues of information security.

Software security is just starting to make it onto the radar screen for organizations. In a nutshell, the objective of software security is to build better, defect-free software. Typically software has many defects and quite a few of these tend to be the source of security vulnerabilities that show up in our operational systems and networks.So another way to think about software security is developing software that is more able to resist attack.And in the face of an attack—a successful attack—it's better able to tolerate the attack and recover from the attack as quickly as possible.

So to net it out, secure software is less vulnerable to attack and a little bit more bulletproof. IT security tends to be more about detecting and responding to incidents, finding and patching vulnerabilities—more remedial or reactive.For software security,

the idea is to get in front of the issue by building more robust software. Software defects are identified and addressed much earlier in the software development lifecycle. It's more preventive or more proactive, if you will. The intent is to put better products in the field so we can eliminate some of the issues we're dealing with today.

Bill Pollak: Why do business leaders need to pay more attention to software security?

Julia Allen: There are probably two major trends or topics that are worthy of business leader consideration.The first one is that attackers are getting much smarter and the second one is that it's just good business. I'll say a little bit more about both of these.

What we're seeing is that the attacker community is becoming much more sophisticated in their approaches. In fact, this is actually a well-funded profession with a very robust underground economy where sensitive information including personal identities and credit card information is being bought and sold quite freely.As a community, we've gotten better at protecting our networks, our IT infrastructures, and our information. As a result, attackers are now focusing on the more vulnerable parts of our systems, which are the applications, particularly web-facing applications.They are the primary gateway to sensitive data. So this is where software security can really provide an advantage.That's the part about the attackers becoming a bit more sophisticated and kind of upping the ante. Another thing to think about here is that an attacker only requires one entry point into a system whereas software developers and IT staff need to anticipate and defend all entry points—which is a nearly impossible requirement.So that's one aspect of why business leaders need to pay attention.

In terms of it being just good business, we're seeing more and more indicators that the market place is starting to demand it, particularly from organizations that develop software. They're saying, "We want secure products and we're not going to put your product into our infrastructure unless it's secure."The total cost of ownership for software, 50 to 80 percent, depending on whose numbers you pay attention to, is in the maintenance and operations of software.So that's where a huge amount of the cost is going because of the poor quality of software that's going into production.What we've observed is that upwards of 50 percent of operational software vulnerabilities are actually design flaws that could have been discovered much earlier in the lifecycle.And in fact, you can predict anywhere from 100 to 1,000 times cost and schedule savings in some instances by identifying and correcting defects early in the lifecycle versus later.

Again, to net it out, we really can't keep up or get ahead of the curve by dealing with security as strictly an operational issue.We need to find a way to deploy better products in the field.

Bill Pollak: How did software get to be so insecure?

Julia Allen: At the Software Engineering Institute and elsewhere, we've seen many instances where software is growing in complexity; we keep adding to it and we keep changing it.Our marketplaces and our customers want new features, new functions, new services, faster.And more software means more vulnerabilities.

Complexity is compounded when you take into account how connected we are today in terms of the Internet; the fact that software development, assembly, and integration are happening globally; the fact that we're always adding new components and new pieces; and the fact that we're interfacing a very complex system with other complex systems creating systems of systems issues.

This is exacerbated by more and more third party software where we as an organization have responsibility for putting the whole system together but we're not developing most of it ourselves. We have to apply the same rigor, requirements, processes, and service level agreements on our suppliers as we do on ourselves.

When you think about all of these different factors, it's really hard even for a very talented development team to get their heads around these issues and try and figure out how to tackle all of this complexity.When you throw security into the mix, it's a pretty daunting undertaking.

Part 2: Develop Software with a Security Mind-set

Bill Pollak: How is developing software with security in mind different from normal software development?

Julia Allen: This is the crux of the message I want to get across, which is how to think with a security mind-set.There are a couple of factors to point out here. The first is to think about security from the beginning of the lifecycle.Security, at least historically, tends to be an afterthought if it's thought about at all. Engineering for more secure software needs to be tackled from the very beginning, when you're going through acquisition, when you're going through requirements specification, all the way through the lifecycle.

Using other non-functional, quality attributes like performance or reliability as examples, think about security during requirements, during design, during architecture, and all the way through implementation, test, and deployment.

Probably the toughest challenge for software project managers, architects, designers, and engineers is that they need to learn to think like an attacker.We often think about what the software should do, what the functions, features, and capabilities are. But we rarely think about what the software should not do.And I would hazard a guess that we really don't think about what it should do and not do when it's under attack.

And certainly last but not least, address security as a risk management issue throughout the development lifecycle.Constantly assess, at each lifecycle phase, what the highest areas of vulnerability and risk are and address the highest risks first. Fully recognize, as you go through the lifecycle, that software security risks are likely to change quite a bit depending on what kind of applications you're building and what kind of services the software is providing. The risks and priorities for tackling them need to be assessed on a continuous basis.

Bill Pollak: Could you provide some examples of known good practices for developing more secure software?

Julia Allen: The good news is that there is an emerging body of knowledge, experience, and practice that organizations that build and buy software are starting to identify. As I mentioned earlier, the first key practice is to integrate software security practices into your existing development lifecycle. We're not suggesting adding a whole new process or a whole new development strategy, method, or lifecycle definition.Take your existing lifecycle and add these practices to it.

During requirements engineering and architecture and design, we recommend a practice called misuse and abuse cases. These are helpful when you're adding a new requirement or a feature into your requirements. Give some serious thought to how the feature could be unintentionally misused or intentionally abused by an attacker.For example, given a requirement to interface a web server and a database server for some type of user-facing application, often a developer will assume that the connection between the web server and the database server can always be trusted.Defining and exercising a misuse or abuse case can actually challenge this assumption by describing scenarios and various ways to exercise the software under the assumption that the interface is *untrusted*.

Another example of good practice is the development of what we call attack patterns. Attack patterns describe a class of vulnerability, how that vulnerability can be exploited, and what kind of attacker skill is required.You can use attack patterns throughout the lifecycle to say, "Okay, if I were an attacker, how would I break this design or how would I infiltrate this particular piece of code?"

Some of the more mature practices are in the secure coding area. There's lots of great guidance on secure coding practices for different languages, using code analysis and other scanning techniques.Last but not least is security testing. Techniques include white box and black box testing as well as threat modeling where you can take an attack pattern and fold it into your testing suite. Tried and true penetration testing can also be very helpful.

Bill Pollak: What are some of the effective ways that project managers can get

started?

Julia Allen: Well, obviously don't try to boil the ocean on the first day.Think about taking small steps from the beginning.Probably one of the best ways to get started is to take a look at the competencies and skills of your current development team and enhance them with some security experts.The organizations that have had great success tackling secure software development have security experts co-resident with software development teams, helping guide, educate, and inform them as they go through their development.

The highest return on investment is to add security practices as early in the lifecycle as possible because that's where you get the greatest benefit.But adding practices early in the lifecycle isn't as mature of a practice, for example, during requirements elicitation and engineering or during the early stages of architecture and design.So most organizations today are starting out with secure coding practices, code analysis, peer review, and doing various types of lower-level testing to tease out some of the defects that might have gotten through to code.These are the most mature practices that are in use today, so many organizations start with these.

In all of our improvement-initiative work at the SEI and independent of any particular practice, there are several tried and true ways to get started.First you have to understand what you're trying to do. Why do we care about secure software engineering?Why are we doing this? What are we hoping to accomplish?Make sure the incentives are clear. Get good buy in from your sponsors so you can have a sustainable improvement initiative.Find some early pilots, just a small number, and make sure to demonstrate some early results.Make sure people understand this is going to take time.Like a lot of things in software, it's a never-ending journey because the threat, requirements, and risk landscapes are always changing. We always say "communicate, communicate, communicate," right? So make sure to conduct regular awareness training and ongoing education.

While putting secure software engineering practices in place is much easier on a new project where you have a clean slate to start with, we're getting some good case studies from the field that describe the harder issue of legacy systems—systems that are already deployed, perhaps where you have third-party software. Subjecting that software to some rigorous scrutiny may be more challenging but can produce the highest payoff.

Resources for More Information

Allen, Julia; Barnum, Sean; Ellison, Robert; McGraw, Gary; & Mead, Nancy. *Software Security Engineering: A Guide for Project Managers*, Boston, MA: Addison Wesley, 2008. An excerpt from the book's Preface is available on the Build Security In website.

The Department of Homeland Security Software Assurance Program's Build Security In website

Julia Allen's podcast: "Building More Secure Software."

Nancy Mead's podcast: "Identifying Software Security Requirements Early, Not After the Fact."

Gary McGraw's podcast: "How to Start a Secure Software Development Program."

Share This Page

For more information

Contact Us

info@sei.cmu.edu

412-268-5800

A Specification for Software Project Performance Measures

NEWS AT SEI

Author

Mark Kasunic

This library item is related to the following area(s) of work:

Process Improvement

This article was originally published in News at SEI on: June 1, 2008

Do you play golf, jog, go bowling, ride a bicycle, lift weights, or play basketball? If you do, then you likely keep track of your performance. Perhaps it is as simple as, "I knocked three strokes off my game today," or "I lifted 10 more pounds than I could last week." People keep score like this because most are performance or achievement driven. They want to know how well they are doing—whether their performance is improving or declining—and how their performance compares with their own personal best or with the performance of others.

In much the same way, companies, organizations, and software projects attempt to understand their overall performance, compare it to others, and find ways to become better. When performance measurement is used for comparison, the measures to be compared must be commonly defined.

This is the major obstacle that has hampered effective software project performance comparison and benchmarking. Broadly speaking, measurement definition is not standardized in software development. Not only is it difficult or impossible to compare measures between projects from different organizations, but comparison within the same organization is fraught with error because different operational definitions are used for the same measure.

The SEI's Software Engineering Process Management (SEPM) program launched a collaborative research effort to investigate ways to improve the practice of software project performance measurement. The collaboration brought together experts from some of the world's top organizations that were already working in or had a strong

Related Links

News

SEI to Co-Sponsor 26th Annual IEEE Software Technology Conference

SEI Cosponsors Agile for Government Summit

Training

See more related courses »

Events

Team Software Process (TSP) Symposium 2014

Nov 3 - 6

interest in software project performance measurement. Table 1 lists the organizations that participated in this initiative.

Table 1: Collaborators

David Consulting Group
Galorath Incorporated
ISBSG
Lockheed Martin
Motorola
Oracle
PRICE Systems
PRTM
QSM
Raytheon
SEI
SPR
4SUM Partners
University of Ottawa

Through a series of workshops and group work sessions, a set of software project performance measures and influence factors was identified and defined.

The group started with a large list of candidate measures and factors, then used multi-voting to whittle the list down to a manageable set of terms. The following questions guided the selection process:

1. What are the key measures that best characterize project performance?
2. What factors impact performance in a significant way?
3. What measures and factors would be most useful for software project performance comparison?
4. How difficult would it be to collect the factor or measure?
5. Is the factor or measure currently being collected as part of your organization's data repository?

The final list of performance measures is presented in Table 2, and the list of influence factors is presented in Table 3.

The newly published SEI technical report titled *A Data Specification for Software Project Performance Measures: Results of a Collaboration on Performance Measurement* (CMU/SEI-2008-TR-012) contains the full definitions for these terms.

Table 2: Performance Measures	*Table 3: Influence Factors*
Project effort	Size
Productivity	Artifact reuse
Project duration	Project type
Schedule predictability	Application domain
Requirements completion ratio	Average team size

Post-release defect density	Maximum team size
	Team expertise
	Process maturity
	Functional requirements stability

Performance measures focus on results, such as how long it took to run a certain distance. They answer the question, “What does success really mean?”

Influence factors are aspects of the environment that can impact the outcome of a performance measure. For example, if you want to compare how fast you run against others, you have to agree about the conditions under which the measurements will occur, such as terrain (flat vs. hilly), temperature (cold vs. hot), and distance (100 yards vs. 2 miles). In the context of running, these variables—terrain, temperature, and distance—are all influence factors. In software development, some influence factors are controllable by management, while others are not. When making comparisons between software projects, influence factors can be used to facilitate the comparison of projects that are similar to each other (with respect to one or more influence factors).

With data available from multiple projects that possess similar characteristics, a project’s performance can be compared against that of other projects to determine areas of strength and weakness. When used in this way, measurement comparison motivates process improvement.

Organizations just starting a measurement program do not have historical data on which to base their estimates, so they want to know what measures to use and what reasonable targets for their measures are. *A Data Specification for Software Project Performance Measures* is a good starting point for these organizations because it lists the key measures that every organization should collect.

Organizations that are more experienced in measurement want to compare their performance with competitors in their industry. If all organizations use the definitions in the specification document, comparison between companies will be more accurate and useful.

Finally, progressive organizations want to learn about the best practices used by industry leaders so they can adapt them for their own use through benchmarking.

In each of the above cases, the valid comparison of measurement data is an integral step in realizing these objectives. The use of common definitions for influence factors and performance measures is a prerequisite for valid comparison.

A more powerful use of performance measurement is within the context of benchmarking. Benchmarking is a process that uses performance measurement to identify best-in-class achievement (i.e., the benchmark), but goes beyond mere comparison to determine how the best-in-class achievement was attained. Once the how is understood, the enablers (e.g., methods, procedures, tools) that led to the stellar performance are adapted by an organization or project to improve and thereby achieve similar stellar performance.

The SEI is interested in feedback from and collaboration with organizations that intend to implement or are implementing the software project performance measures and influence factors specified in this document. If you would like to provide feedback or discuss collaboration, contact us using the links in the *For More Information* box at the bottom of this page.

Find Us Here

Share This Page

For more information

Contact Us

info@sei.cmu.edu

412-268-5800

New Priorities

NEWS AT SEI

Author

Watts S. Humphrey

This library item is related to the following area(s) of work:

Process Improvement

CMMI

This article was originally published in News at SEI on: July 1, 2008

This column has now run for 10 years, and I have decided that it is time to change its focus. I have enjoyed the privilege of sharing my thoughts with you but have concluded that 10 years is long enough for this particular way of discussing software issues and feel that now is the right time to present new ideas and different viewpoints. I say more about my specific ideas and plans at the end of this column.

In reviewing the 40-some columns I have produced for *SEI Interactive* and later *news@sei* in the last 10 years and the 17 monthly columns I produce for an on-line journal (*ObjectCurrents*) run by Bob Hathaway in 1996 and 1997, I have been struck by several thoughts. The first and most obvious reaction is how little has changed. My very first column in January 1996 was titled the "The Changing World of Software." It talked about the truly abysmal state of software practice and how our customers seem to tolerate this poor performance without a whimper. What disturbs me the most about this first column is that I could have written it 30 years ago or last week. While a lot has changed for software technology, and we have learned a great deal about software processes and how effective they can be if properly designed and used, our performance as an industry is still just about as bad as ever. Further, the way we teach our computer professionals has not materially changed, at least not by enough to improve industrial performance.

After my initial column on the changing world of software, the next four were anecdotes about software issues followed by an 11-column series on the PSP PROBE estimating method. At the time I had just completed my PSP research and was planning to devote my subsequent columns to describing the various PSP methods.

Please note that current and future CMMI research, training, and information has been transitioned to the CMMI Institute, a wholly-owned subsidiary of Carnegie Mellon University.

Related Links

News

SEI to Co-Sponsor 26th Annual IEEE Software Technology Conference

SEI Cosponsors Agile for Government Summit

Training

See more related courses »

Events

Team Software Process (TSP) Symposium 2014

Nov 3 - 6

(< 5 minute) survey.

Unfortunately, the *ObjectCurrents* journal shut down in May of 1997, so that ended my first series of 17 columns.

About a year later, in June 1998, the SEI asked me to write a quarterly column for a new on-line publication, *SEI Interactive*, with the column to be called "Watts New." I have now completed 10 years of these columns and have covered a lot of territory. In reading them over, I find that they are all just as pertinent today as they were when I wrote them. The only significant difference is that, during much of this time, we were just developing the Personal Software Process (PSP) and Team Software Process (TSP) and gathering data on their use in industry. While I was convinced that these methods were revolutionary, I couldn't prove it, so my columns tended to be more abstract with a primary emphasis on principles and methods.

We now face a different situation. Ten years ago, the PSP and TSP methods were new and little was known about them. While the PSP and TSP are not yet widely used, their use is growing and there is now a substantial body of literature about them [Callison 2008]. In this 10-year period, I have published many technical papers, 57 columns, and 7 books [Humphrey 2008]. There is also a growing literature by industrial and academic users who describe their experiences and findings regarding TSP use. Now that a lot has been written about the subject and there is a growing body of evidence that they are highly effective, at least when properly used, it is time to reassess the situation and decide what is needed next.

When I retired from IBM 22 years ago, I was concerned about the poor state of the software industry, and I decided to dedicate my efforts to transforming the world of software. I made what I called an outrageous commitment to fix the world's software problems. I then joined the SEI and formed the Process Program. At that time, we faced two major challenges:

1. to determine how software work should be done
2. to get the world to do software work that way

In the intervening years, we have made a great deal of progress with the first challenge and a modest amount of headway with the second. Regarding the first challenge, we have come a long way, but it would be presumptuous to say that we now have the last word on how software work should be done. However, we do know enough to enable software organizations to consistently and predictably deliver quality software products. That is an enormous advance. With CMMI, we have shown that management has a key role to play in the software process and that, when management issues are properly addressed, organizational performance improves. As a by-product of the CMMI work, however, we have also learned that changing the practices and behavior of the managers is not enough. Unless we address the way the individual software developers and their teams work, we cannot achieve our long-term objectives.

To address these personal and team issues, we next developed the PSP and TSP, and the results have been truly extraordinary. Consider the following facts.

- Microsoft has invested about $3,000,000 to introduce TSP into its IT organization. Microsoft has trained about 1,000 software developers in PSP and now has data on more 200 TSP projects. Because of improved product quality and more predictable schedules and costs, Microsoft estimates that TSP has saved them a total of $84,000,000 to date.
- Intuit has introduced TSP into its largest division and currently does about 60% of its development work with this method. Because of improved product quality, they report that the number of customer calls to their help lines has declined by 30%. This is a reduction of 800,000 calls a year, and each call costs them $25. The total saving from this source alone is thus $20 million a year.
- Vicarious Visions, a division of Activision, introduced TSP because the quality of engineering work life had declined so much that turnover reached 17% a year. They now report that their experienced engineers, once they have worked on a TSP project, refuse to work any other way.
- Softtek, the largest software company in Latin America, reports that the

engineering turnover on their TSP teams is one quarter of that on their other teams.

With results like these, you would think that everybody would be jumping onto the TSP bandwagon, but that is not the case. For example, even though Microsoft's IT organization has shown that they can save $84 million from a $3 million investment in TSP, none of Microsoft's product development groups have adopted TSP. How can this be?

Lest I sound critical of Microsoft, I assure you that they are not unique. Every major corporation that has introduced TSP has had a similar experience. Unless the introduction effort started at the top of the organization, adoption did not spread. Somehow, everybody seems blind to innovations made by other people or groups, even within the same company. What is most surprising is that this includes many of the large Department of Defense contractors and other organizations that have been rated at CMMI Maturity Level 5.

This is most surprising because one of the two principal requirements to be rated at CMMI Level 5 is the Organization Innovation and Deployment (OID) process area. This area requires that Level-5 organizations look for promising incremental and revolutionary improvement opportunities both within and outside of their organizations, and that they quantitatively evaluate these opportunities for potential use within their organizations. To quote from CMMI [Chrissis 2007]:

The purpose of the Identify and Analyze Innovations specific practice is to actively search for and locate innovative improvements. The search primarily involves looking outside the organization.

In all the years that we have been working on TSP, we have yet to have any CMMI Level-5 organization come to us for the data needed to conduct such an evaluation. This can't be because they have never heard of TSP. I routinely give talks at major conferences, including the SEPG conferences in the United States and other parts of the world, and I often ask how many people have heard of TSP. While 10 years ago, very few hands went up, today more than half of the people in the audience have heard of TSP. This includes audiences in Australia, Chile, China, Hungary, India, and all over the United States. Clearly, the major software process-improvement issue faced today is the second challenge.

Getting the world to do software work in the best known way.

Based on the data we have seen to date, that means using TSP. Depending on whom you talk to, this statement will likely get reactions like the following.

- We are only at CMMI Level 2 and need to get to Level 3 before trying TSP.
- We use Agile methods so we can't adopt TSP.
- We just started to use RUP so TSP is not appropriate for us.
- We use Function Points and TSP only uses LOC, so we can't change.

These views are all based in misinformation. TSP has been used successfully by organizations at every CMMI maturity level, and TSP is used by many groups that use Agile methods, including Scrum and XP. Similarly, RUP is completely consistent with TSP, and there is no reason not to use Function Points in estimating TSP jobs. TSP was designed to be language, method, and environment independent. It adds a family of measurement, planning, tracking, and quality-management practices that are not currently used by any of the popular software methods, so it does not conflict with any of them. It merely requires that the developers and their teams do some things that they do not now do.

Based on the results we have seen to date, it is clear that TSP use will grow but that the rate of this growth will be very slow. While we certainly have to continue doing what we have been doing, when you get to be my age, you would like to see results in less than the 10 to 20 years that our current rate of progress implies. This problem, however, is not new. W.E. Deming struggled for years to convince U.S. industry to adopt his well-demonstrated quality methods. Unfortunately for the U.S. auto industry,

the Japanese adopted Deming's methods first. Now, Toyota is within a hair of becoming the largest automaker in the world, and it is already the most profitable.

So is there any hope that we can accelerate the rate of TSP adoption? I believe that there are five possible avenues.

1. Convince the computer science and software engineering academic communities of the effectiveness and essential nature of TSP methods. While this community could be enormously helpful in convincing the world that TSP is the right way to go, I give this strategy very low odds. Based on what we have seen to date, the academic community changes even more slowly than industry. This does not mean that it could not change, but that it is not likely to happen soon.
2. Get the customers to demand that their software suppliers adopt TSP. While this would be nice, it is not likely to happen until a very large number of customers have experience with TSP, and it is obvious to everyone that it is the right way to go.
3. Convince the government to establish a software industrial improvement program based on TSP. In the United States, such a strategy is a non-starter, at least for now. Outside of the United States, however, the situation is quite different. When organizations are hungry for business, they are much more receptive to innovative new methods.
4. Get the U.S. government to mandate TSP use, at least as a prerequisite for DoD software development contracts. While this approach was very effective in getting CMMI adopted by the major DoD suppliers, it has a serious downside. The problem is that if the government mandated TSP as a prerequisite for software development contracts, that would be motivating organizations to do something they didn't believe in (adopt the TSP), in order to get something they wanted (a DoD contract). While this approach can work when you have a foolproof way to determine if someone is using TSP properly, there is an enormous motivation to cheat, which could easily lead to complications.
5. Find some better way to communicate the benefits of TSP to senior corporate executives. While I do not now see a clear way to do this, I do have ideas about possible avenues to explore.

In summary, I have concluded that the most important thing to do right now is to broaden the understanding of TSP's capabilities and benefits. The approach I have selected is to continue editing these columns but to invite members of the TSP team at the SEI to contribute the material. Also, where there are opportunities, I hope that some of the columns will be co-authored by TSP users. We have learned a great deal over the last 15-plus years, and these columns can provide a convenient and easy way to quickly communicate significant new results to the broader software community. So, in conclusion, that is why I have decided to refocus this column. Please stay tuned, and continue to drop me notes on TSP-related topics you would like to see addressed.

References

[Callison 2008]
R. Callison and M. MacDonald. *Bibliography of the Personal Software Process (PSP) and the Team Software Process (TSP)*, SEI Special Report, 2008.

[Chrissis 2007]
Mary Beth Chrissis, Mike Konrad, and Sandy Shrum. Guidelines for Process Integration and Product Improvement, 2nd Edition Reading, MA: Addison Wesley, 2007, p. 278.

[Humphrey 2008]
Watts Humphrey. *Watts New, the Collected Columns of Watts Humphrey*, SEI Special Report, 2008.

About the Author

Watts S. Humphrey founded the Software Process Program at the SEI. He is a fellow of the institute and is a research scientist on its staff. From 1959 to 1986, he was

associated with IBM Corporation, where he was director of programming quality and process. His publications include many technical papers and 11 books. His most recent books are *Winning with Software: An Executive Strategy* (2002), *PSP: A Self-Improvement Process for Software Engineers* (2005), *TSP, Leading a Development Team* (2006), and *TSP: Coaching Development Teams* (2006). He holds five U.S. patents. He is a member of the Association for Computing Machinery, a fellow of the Institute for Electrical and Electronics Engineers, and a past member of the Malcolm Baldrige National Quality Award Board of Examiners. He holds a BS in physics from the University of Chicago, an MS in physics from the Illinois Institute of Technology, and an MBA from the University of Chicago. In a White House ceremony in 2005, President George W. Bush awarded him the National Medal of Technology.

The views expressed in this article are the author's only and do not represent directly or imply any official position or view of the Software Engineering Institute or Carnegie Mellon University. This article is intended to stimulate further discussion about this topic.

Lessons Learned about Risk Mitigation in Acquisition through Architecture Evaluation

NEWS AT SEI

Author

John Morley

This library item is related to the following area(s) of work:

Software Architecture

Acquisition Support

This article was originally published in News at SEI on: July 1, 2008

U.S. Department of Defense (DoD) program managers care about how well the software-intensive system they are developing will perform for the warfighter. They are not particularly interested in software architecture, at least not *as* architecture.

But through efforts made by the SEI, they are gaining an understanding that the architecture of the software-intensive systems they are developing significantly determines whether those systems will be reliable, secure, and safe—characteristics known as nonfunctional or quality attributes.

“Architecture is not a set of PowerPoint slides,” says Michael Gagliardi of the SEI. “It’s the bridge between the business goals and the way a system operates. When bottlenecks are seen at integration time or unanticipated system behavior occurs in operation, the root causes for those problems can be found in the architecture.”

“There is benefit to inserting an architecture-centric approach in acquisition,” Gagliardi continues. “The program manager can use leverage in acquisition to require a fully developed architecture that can be evaluated for risks and give early insight into necessary tradeoffs.”

The SEI’s proactive, architecture-centric approach has been tested and proved in several DoD software acquisitions over many years. Those engagements underscore several key lessons about the architecture-centric approach in acquisition and some SEI technologies.

Related Links

News

SATURN Conference Announces Additional Keynote, Conference Scholarships

Distinguished Speakers, Strong Technical Program Set for SATURN 2014

See more related news »

Training

Big Data - Architectures and Technologies

Data Rights and DoD Acquisition

See more related courses »

(< 5 minute) survey.

1. **An architecture-centric approach relates mission goals to the achievement of system quality attributes**
   The architecture-centric approach begins before there is an architecture. Before a system is let out for bidding by contractors, the SEI can use an approach called a Quality Attribute Workshop (QAW) to capture key information. In a QAW, the SEI "brings together all of the system stakeholders across the life cycle in a facilitated session," according to John Bergey of the SEI.

   "We ask, 'What are the mission or business drivers? What is your notional architecture?'" Gagliardi says. In the QAW, scenarios or vignettes are used to explicitly capture the system's business and mission goals and quality attributes.

   "We can also prioritize the drivers and attributes," Gagliardi notes. "This gives an early indication of the kinds of tradeoffs that the system architect will have to deal with. For instance, if high availability is needed, then the system might not be able to provide as high a level of performance."
2. **An architecture-centric approach makes acquisition more effective**
   The U.S. Army acquisition strategy calls for means to "minimize the time and cost it takes, consistent with common sense and sound business practices, to satisfy identified, validated needs, and to maximize affordability throughout a program's useful life cycle" [Army 2008].

   A program manager, Gagliardi says, "focuses on capabilities and functionality, while managing risk to cost and schedule. Architecture embodies the nonfunctional attributes that can affect cost and schedule."

   The SEI uses the term "architecture risks" to denote problems with architecture that hinder a system's ability to deliver on the nonfunctional attribute requirements. "Letting architecture risks fester is costly," Gagliardi warns.

   As a first step toward finding and mitigating architecture risk, the SEI suggests that the request for proposal (RFP) include language such as the following: "As a software acquisition risk reduction measure, the contractor shall participate in and support a collaborative evaluation of the software architecture that is to be led by an evaluation team commissioned by the program office. The architecture evaluation shall be conducted prior to the preliminary design review (PDR) in accordance with the software architecture evaluation plan."

   A method the SEI employs to evaluate architecture risk is the SEI Architecture Tradeoff Analysis Method (ATAM). Using the ATAM , an impartial evaluation teams works collaboratively with stakeholders to evaluate a software architecture. When performed in the acquisition life cycle, an ATAM exercise reveals issues before they become problems to be addressed during the integration and test phase, when remediation is time consuming and costly.
3. **An architecture-centric approach delivers critical input at the right time for decision making**
   U.S. Army Acquisition policy calls for several key decision points. Specifically, the policy states, "The review associated with each decision point will typically address program progress, risk, affordability, supportability, program tradeoffs. . . " [Army 2003].

   Bergey suggests that ATAM-based evaluations be added at the scheduled decision points in the acquisition life cycle to feed information about what needs to be addressed into those reviews.

   "During an ATAM-based evaluation, the system architect presents the architecture and the architectural approaches taken for each nonfunctional attribute in each scenario," Gagliardi explains. "ATAM evaluators probe the architecture and pose questions to the architect." Gagliardi continues. "We search for risks, tradeoffs, and areas that comply with the scenarios. Risks are noted where we see inconsistencies between the approach chosen and the nonfunctional

attribute it is intended to support."

4. **An architecture-centric approach fosters improved communication**
   An architecture-centric approach gives "maintainers, sustainers, testers, architects, verifiers—all stakeholders—a view they didn't have," according to Gagliardi. Through application of the QAW and the ATAM, they "understand how the system is being put together," he says.

   Even more, they have input into the development of the system about their needs and the non-functional attributes that are important to achieving the business and mission goals, Bergey adds.

5. **An architecture-centric approach identifies risk themes**
   The ATAM evaluators take the risks identified during question-and-answer sessions on scenarios with the architects and group them into larger risk themes. "We percolate up the risk themes," Gagliardi says, "because significant risks are likely to cut across scenarios."

   Risk themes, Gagliardi explains, are not just generalizations, however. Rather they are statements of risk patterns that resonate more readily with key decision makers. "We might have a risk theme such as "the data architecture is not well thought out, or security has not been attended to. The individual, specific risks that inform the theme are also provided," he says.

   The purpose of the themes, too, is to point to systemic causes of risk that relate to the ability to achieve the business and mission goals and meet the nonfunctional attribute requirements. The program managers can use the risk themes to form overarching mitigation strategies, according to Bergey.

Continuum of architectural support

When software-intensive systems fail to perform as expected or exhibit safety or other problems, the warfighters and others who rely on them want to know why.

"Architecture is the underpinning to the system and the place where the causes of those problems can be found," Gagliardi explains.

"The SEI advises an architecture-centric approach throughout the life cycle, from the pre-RFP stage through development," he says. "It's in this focus on architecture from end to end that problems can be identified before they prove costly and time consuming to fix. Our continuum of architecture support—specification, evaluation, improvement—provides acquirers with a proactive approach that produces the timely information they need."

Also visit the Software Architecture Acquisition section of our site.

[Army 2008]
Department of the Army. *Army Acquisition Procedures (Pamphlet 70–3)*. Department of the Army, 2008. http://www.army.mil/usapa/epubs/pdf/p70_3.pdf

[Army 2003]
Department of the Army. *Army Acquisition Policy (Pamphlet 70-1)*. Department of the Army, 2003. http://www.army.mil/usapa/epubs/pdf/r70_1.pdf

info@sei.cmu.edu
412-268-5800

How Mexico is Doing It

NEWS AT SEI

Author

Watts S. Humphrey

This library item is related to the following area(s) of work:

Process Improvement

This article was originally published in News at SEI on: August 1, 2008

Editor's Note: Since June 1998, Watts Humphrey has taken readers of *news@sei* and its predecessor *SEI Interactive* on a process-improvement journey, step by step, in his column *Watts New*. The column has explored the problem of setting impossible dates for project completion ("Your Date or Mine?"), planning as a team using TSP ("Making Team Plans"), the importance of removing software defects ("Bugs or Defects?"), applying discipline to software development ("Doing Disciplined Work"), approaching managers about a process improvement effort ("Getting Management Support for Process Improvement"), and making a persuasive case for implementing it ("Making the Strategic Case for Process Improvement"). And now, after nearly 11 years, Watts is taking a well deserved retirement from writing the quarterly column. But you can still enjoy vintage Watts New columns.

—Richard Lynch
Editor
news@sei

Watts New [2008 | 8]

How Mexico is Doing It

Watts S. Humphrey & Anita Carleton

Introduction

In five years, Ivette Garcia, the director of Mexico's digital economy hopes Watts

Related Links

News

SEI to Co-Sponsor 26th Annual IEEE Software Technology Conference

SEI Cosponsors Agile for Government Summit

Training

See more related courses »

Events

Team Software Process (TSP) Symposium 2014

Nov 3 - 6

Humphrey will be asked, “How did Mexico do it?” In her keynote address at the Third Annual Team Software Process (TSP) Symposium held in Phoenix, Arizona, Garcia announced that she hoped the Mexican software industry would soon compete for a larger share of the U.S. software outsourcing market. How would Mexico accomplish this? “You need to differentiate yourself to compete. Mexico plans to differentiate itself through its largest competitive advantage—the TSP,” says Garcia [Garcia 08].

Background

Although the International Data Corporation (IDC) estimates that the IT services outsourcing global market reached $310 billion last year, only about 8% of that is done from offshore destinations, with India being the undisputed leader. McKinsey estimates that by 2010, the global IT outsourcing market will reach $1.1 trillion. The increase in the share of this market served from offshore destinations could reach 15%. That means that the offshore outsourcing market could reach $165 billion in the next four years.

Although Mexico is the United States’ second largest trade partner, the Mexican software industry does not yet compete effectively for a share of the U.S. software outsourcing market. For example, in 2007 India sold $3 billion of software services to the United States compared to $900 million for Mexico. However, as the market continues to grow, no single nation will be able to satisfy the market need. This provides an opportunity to increase Mexico’s participation in this growing market. The Mexican strategy to accomplish this is for Mexican organizations to improve quality, productivity, and delivery schedules.

Mexican Government Launches National Initiative

The Mexican government, through an initiative called PROSOFT, has launched an aggressive program to build its national reputation. This initiative integrates government, industry, and academia to develop competitive human capital, strengthen local industry, enhance process capabilities, improve quality assurance, and promote strategic alliances with foreign companies. A key to this program is the introduction of TSP. TSP provides teams and their management with precise operational guidance on how to implement CMMI high-maturity practices. The Personal Software Process (PSP) provides the development team members with the skills and practices needed to be productive and effective TSP team members [Humphrey 02, Humphrey 05].

Due to the global competition in information technology, Mexico has to differentiate its supply, offering quality in the development of IT products and services in less time and with higher added value. “We know that in order to develop consistent quality software, we need to have high maturity processes. The most popular process maturity model internationally is CMMI. But this model is complex to implement—especially in small enterprises,” says Garcia.

The strategy to increase the software industry’s maturity in Mexico has to consider not only the enterprises’ processes, but also the improvement of the basic element that supports the industry: the people. Building high performance knowledge workers is the focus of PSP and building high-performance working teams is the focus of TSP.

As a whole, the software industry needs to improve cost and schedule management, cycle-time, and product quality. Improving performance in these areas and developing the workforce capability are important PROSOFT goals. Previous reports [Davis 03] document the success of TSP in producing high quality products on time and within budget. TSP operationally implements high performing development processes. These processes are managed by trained individuals and teams.

PROSOFT has been able to make a number of significant advancements. Mexico now has 120 software development centers with certifications in several quality process models (e.g., CMM, CMMI, ISO, MoPoSoft). In 2002, only four centers had certification. Additionally, there are now 23 academic clusters and 17 industry integrators in the IT sector and 121 universities focusing on improving professional IT education.

Some Early Results

Based on some initial results from training and implementation data for projects and

individuals that have adopted TSP in Phase I of the Mexican TSP initiative, the results show that TSP teams are delivering high quality (low defect) software on schedule, while improving productivity. These data can be used for benchmarking, lessons learned, and other guidance to those currently using the TSP or considering participation in the future [Deploying TSP 09].

Some early results from pilot projects show that the pilot TSP teams delivered their products on average on or within two weeks of the committed date. This compares favorably with industry data that show over half of all software projects were more than 100% late or were cancelled. Key to schedule success was overall high product quality—several TSP projects had no defects in system or acceptance test.

The TSP is implemented by a highly motivated development staff and management. Given the opportunity to speak for themselves, developers say they prefer the work environment of a TSP team. Management likes depth of the data and the reliability of status reports. Low worker attrition, a relative strength of Mexico, is not only maintained, but enhanced. One company survey of employees found the TSP pilot team to have the highest job satisfaction in the plant.

During the initial TSP roll out phases, a number of challenges surfaced:

- the up-front cost in both time and money
- management acceptance and support of self-directed teams
- appropriate use of the detailed data

While these problems are not unique to Mexico, they do need to be addressed to roll out this program on an organizational and national scale. A particularly important issue for Mexico is the number of small and medium sized enterprises that cannot afford the initial training. In the outsourcing market, small short-term projects must be staffed and launched on short notice. New PSP training courses have been developed by the SEI to reduce the time and cost required to launch teams.

Next Steps

TSP is beginning to show some promising results and benefits for Mexican companies. Rolling out on a national level, however, is not only challenging, but unprecedented. In addition to the practical problems of the rollout, national success depends on visibility and recognition of the accomplishments. Next steps include:

- training PSP developers in the universities
- training Mexican university professors to deliver PSP and TSP classes
- developing TSP as a cost effective way to implement CMMI
- certifying and recognizing companies that effectively use TSP

“I want the world to recognize the benefits of working with Mexico. If you are going to work with someone, you need to trust them. That is why we need TSP/PSP. So people will have trust in the quality of our products and services,” stated Garcia. Mexico has launched an unprecedented and far reaching program to change an entire industry and has committed significant national, state, academic, and industrial resources to doing this on an aggressive schedule. If Mexico is successful, other countries are likely to follow its lead. We should all stay tuned to Mexico’s progress as it pursues this impressive and aggressive national strategy.

References

[Davis 03]
Davis, Noopur and Mullaney, Julia. “The Team Software Process (TSP) in Practice: A Summary of Recent Results,” CMU/SEI-2003-TR-014.

[Garcia 08]
Garcia, Ivette. “Prosoft 2.0 A National Program to Develop The IT Industry.” Keynote Address at the *Third Annual Team Software Process (TSP) Symposium* held in Phoenix, Arizona, September 2008.

[Humphrey 02]

Humphrey, Watts S. *Winning with Software: An Executive Strategy*, Reading, MA: Addison-Wesley, 2002.

[Humphrey 05]
Humphrey, Watts S. *PSP: A Self-Improvement Process for Software Engineers*, Reading, MA: Addison-Wesley, 2005.

[Deploying TSP 09]
Deploying TSP on a National Scale: An Experience Report from Pilot Projects in Mexico

About the Authors

Watts S. Humphrey founded the Software Process Program at the SEI. He is a fellow of the institute and is a research scientist on its staff. From 1959 to 1986, he was associated with IBM Corporation, where he was director of programming quality and process. His publications include many technical papers and 11 books. His most recent books are *Winning with Software: An Executive Strategy* (2002), *PSP: A Self-Improvement Process for Software Engineers* (2005), *TSP, Leading a Development Team* (2006), and *TSP: Coaching Development Teams* (2006). He holds five U.S. patents. He is a member of the Association for Computing Machinery, a fellow of the Institute for Electrical and Electronics Engineers, and a past member of the Malcolm Baldrige National Quality Award Board of Examiners. He holds a BS in physics from the University of Chicago, an MS in physics from the Illinois Institute of Technology, and an MBA from the University of Chicago. In a White House ceremony in 2005, President George W. Bush awarded him the National Medal of Technology.

Anita Carleton has worked at the Software Engineering Institute, Carnegie Mellon University for over 20 years on software process improvement, process measurement, and the Team Software Process. She is the author of *Measuring the Software Process: Statistical Process Control for Software Process Improvement*, published by Addison Wesley in June 1999. She has a degree in Applied Mathematics from Carnegie Mellon University and is a member of IEEE Computer Society and National Defense Industrial Association (NDIA).

The views expressed in this article are the author's only and do not represent directly or imply any official position or view of the Software Engineering Institute or Carnegie Mellon University. This article is intended to stimulate further discussion about this topic.

Create, Apply, and Amplify: A Story of Technology Development

NEWS AT SEI

Author

John Morley

This library item is related to the following area(s) of work:

System of Systems

This article was originally published in News at SEI on: August 1, 2008

Don't do it . . . *yet*.

In a nutshell, that was the recommendation in 2005 from the Carnegie Mellon Software Engineering Institute (SEI) to its first customer using the Service Migration and Reuse Technique* (SMART) approach for determining the feasibility of migrating one of its legacy systems to a service-oriented architecture (SOA) environment. The SMART process uncovered significant risk due to the state of the customer's target SOA environment and the scope of the migration effort. The SEI team suggested a pause in legacy system migration planning while work continued on the SOA environment and the scope of the migration was better defined. In May 2007, the customer successfully demonstrated the migration of a set of legacy components to services.

Recommending that a customer not plunge ahead into SOA reveals a strength of the SMART process and also illustrates the different way the SEI interacts with customers in the U.S. Department of Defense, civilian government agencies, and industry.

Technology vendors tout the advantages of moving to an SOA environment: Organizations, they point out, will benefit from lower development costs, enhanced business agility, leverage of legacy systems, greater access to information, and improved business processes. The SEI is realistic about the effectiveness of SOA-enabling technologies and focuses on transitioning best practices that provide the basis for sound decisions about SOA.

Related Links

Training

Migrating Legacy Systems to SOA Environments - eLearning

Service-Oriented Architecture: Best Practices for Successful Adoption

See more related courses »

Federal government and others organizations are attracted to the promises of SOA; some are even mandated to move toward an SOA environment because of them. But the adoption of the SOA approach, particularly by the DoD and federal agencies, lags far behind the attraction [1].

One significant contributing factor is a hesitance to migrate legacy systems to the new environment. For federal agencies, a large amount of information, including mission-critical information written in older programming languages, is housed in legacy systems such as mainframe computers.[1] Also, for these agencies—and all organizations with a heavy investment in legacy systems—

- It would be costly to redesign a system as large and complex as the legacy system.
- Users require nearly constant availability from the systems: to design a new system with the same level of availability would be costly.
- No one really knows how the legacy system works and there often is not full documentation available [4].

The SEI has investigated issues pertaining to software reuse and legacy system migration for several years.[2]Research in to those issues led to the development of the (SMART) approach by the SEI's Integration of Software-Intensive Systems (ISIS) Initiative about three years ago.

Using SMART, an organization can obtain a contextual, hands-on identification and analysis of issues in migrating legacy components to SOA environments. Since its initial development, the SMART process has been applied in several customer situations and refined into a state ready for transition.

The story of the SMART process is also an example of how the SEI is able to respond to need with an innovative concept, test it through engagements with customers, refine it based on those engagements, and support its adoption with training and tools.

Eileen Forrester has been involved with transition efforts at the SEI for more than a decade. Forrester says the transition of a technology or approach begins early in its development: "What is the problem we are trying to solve? That's the first question we ask." SEI transition moves, generally, through exploration, maturation, outreach, and support phases, according to Forrester.

"As we move through the transition life cycle, the questions get sharper. During the maturation phase, we will ask, for example, 'do we know that any intended users can use our solution.' In outreach, we are looking to find out what mechanisms we are developing for transition. And in support, we would ask how we will support the technology and our transition partners," Forrester explains.

Through the lens of the SEI technology transition life cycle, in this three-part series of articles we will examine:

- origin, creation, and initial application of the SMART process
- refinement of the process and of the tools it uses
- evolution of the process based on customer engagements and feedback

Part 1 – Becoming SMART

"SOA allows the reuse of legacy systems. But constructing services from existing systems to obtain the benefits of SOA is neither easy nor automatic," points out Grace Lewis, technical lead for SEI's SMART process and system-of-systems engineering research.

"Even before developing services, organizations need to ask whether it makes sense to migrate their legacy systems," Lewis says. "And that means understanding what services are and what SOA is and can provide."

In SMART, Lewis and her ISIS colleagues created is a process that guides an organization through addressing six critical questions for determining the feasibility of migrating its legacy systems making an initial migration plan:

- Does it make sense to migrate the legacy system to services?
- What services make sense to develop?
- What legacy system components can be mined to derive these services?
- What changes are needed to accomplish the migration?
- What migration strategies are most appropriate?
- What are the preliminary estimates of cost and risk?

According to Lewis, the backbone of the SMART approach is a "questionnaire, which is the codification of migration knowledge." The questionnaire, called the Service Migration Interview Guide (SMIG), guides the SMART team's discussions with stakeholders and developers in four activities.

The first activity is to establish the goals of the migration project and make note of its drivers and constraints such as schedule and budget.

"Why does the organization want to migrate its legacy system, what are the business and technical drivers and who are the stakeholders of the migration—these are the key questions we ask," Lewis says. "We also help the organization collect information on what the legacy system is about, the kinds of services they think it wants to produce, and the types of service consumers."

After collecting this information, the organization can decide whether to continue. The migration might be feasible, it might be potentially possible with more information, or it might not be feasible. If migration is feasible for the organization, the discussion turns to three other SMART activities: define possible services, describe the legacy components, and define the target SOA environment.

The organization should specify a small number of services, usually three or four. Dennis Smith, technical lead for the ISIS Initiative, notes that organizations at first believe they can turn an entire legacy system into services. But they need to try to identify a few components of the legacy system to make available as services.

The SMART team and the organization then have enough information to complete the last two SMART activities. They can analyze the gap identified between the existing and future states in order determine a preliminary estimate of the effort, cost, and risk for migrating legacy components into services. And they can form a migration strategy, the culmination of the SMART process.

Implementation

Having designed an approach, Lewis, Smith, and others in ISIS conducted a pilot of the approach to analyze the potential for migrating legacy components of a DoD command and control (C2) system. Among the lessons they learned was that architecture reconstruction might be needed to address issue of dependencies in detail when there is no architecture documentation or it is outdated. They found they can use the SEI-developed ARMIN tool for this purpose.[3]

With the approach tested, the ISIS group was ready to apply it to a customer situation. The first application of the process showed that SMART is innovative because it recognizes that technologies are not the main aspect of SOA for every customer. Indeed, after initial discussions with its customer, the U.S. Army CERDEC (Communications-Electronics Research, Development, and Engineering Center), the SMART team recommended that CERDEC not proceed with making a migration plan at that time.

"We saw there that their target SOA environment was not mature enough," Lewis explains. "We have seen that other customers are not ready for a migration; they need more background in SOA, in aligning their SOA strategy with their mission or business objectives, and in what services they can form from their legacy systems."

Other implementations followed the initial work with CERDEC. Today, the approach has been used or is being used with four large federal-government organizations. In two of those organizations, departments have entered into the process of transitioning

the SMART technology to act as centers to guide other departments in the migration of their legacy systems to the enterprise SOA environment.

During the course of these implementations, Lewis and her team proved the value of their solution. And they saw the need to modify some aspects to match the high level of complexity that exists among customer organizations that are considering migrating legacy systems to SOA environments. We will look at those refinements in the third and final part of the series.

They also proved that the central tool, the SMIG, provided the right means to gather data and information crucial to developing a plan for migration. But using the paper-based tool was labor-intensive, and it was becoming more cumbersome to deliver results to customers in a timely way.

In the next installment of this series, we will examine how the ISIS team made the SMIG tool more efficient through an innovative partnership with Carnegie Mellon University.

For Further Reading

SMART: Analyzing the Reuse Potential of Legacy Components in a Service-Oriented Architecture Environment (CMU/SEI-2008-TN-008)

The ISIS SOA Information Center

Making the Right Decision about SOA: How Can You Distinguish Fact from Fiction regarding SOA?

Notes

[1] http://www.govtech.com/gt/299300?topic=208827

[2] http://www.informationweek.com/641/41iuleg.htm

[3] http://www.developer.com/mgmt/article.php/1492531

[4] http://en.wikipedia.org/wiki/Legacy_system

1 The private sector seems to have overcome obstacles related to migrating legacy systems. The reliance on legacy systems is not less there. It is commonly believed that more than two-thirds of business information resides in mainframe databases [2]. Put another way, in an estimate by the International Data Corporation, there are 200 billion lines of legacy codes on “more than 10,000 large mainframe sites” in use [3].

2 See for example the SEI’s Framework for Product Line Practice and Modernizing Legacy Systems: Software Technologies, Engineering Processes, and Business Practices.

3 The ARMIN tool provides graphical and textual representations of an architecture as implemented. For more, go to http://www.sei.cmu.edu/architecture/start/reconstruction.cfm.

* As of June 2010, SMART is defined as the *SOA Migration, Adoption, and Reuse Technique*.

Contact Us
info@sei.cmu.edu
412-268-5800

Educating Educators in Making Better Architecture

NEWS AT SEI

This article was originally published in News at SEI on: August 1, 2008

Competent development of software architecture is such a major factor in creating an effective system—it seems it would naturally play a major role in a university's software engineering curriculum. But the principles underlying sound architecture, well known to many researchers and practitioners, aren't always standard fare in college classrooms. Members of the SEI Architecture-Centric Engineering team, who have been developing best practices in architecture for 14 years, promote their inclusion in college coursework by offering an annual workshop for educators. This year, conductors of the fifth Software Architecture Workshop for Educators noted how its influence had deepened and expanded. "What was striking about this year's workshop," remarked Larry Jones, a senior member of the SEI technical staff, "was the reported degree to which architectural topics were now woven into graduate and undergraduate programs in computer science, software engineering, information systems, and information technology."

The workshop was held August 18-20, 2008 at the SEI in Pittsburgh. The event was free of charge and attendance was limited, but it attracted participants from all over the world. This year 12 college faculty members attended, including representatives from Turkey, Thailand, and Brazil. Half the group had attended at least one of the previous workshops.

The three-day event introduces the concepts behind effective software architecture and how to incorporate these concepts into the computer science and software engineering curriculum. These principles promote architecture that ensures certain desirable system qualities, called *quality attributes*, such as performance, modifiability, and usability.

Over the years the event has clearly met an expanding demand, with the increased recognition that both graduate and undergraduate engineering and information science students must be educated in architecture. Software architecture represents the earliest design decisions that are both the most difficult to get right and the most difficult to change downstream. System qualities such as performance, security, interoperability, usability, and modifiability are largely supported or precluded once the software

architecture has been established. The architecture also influences how engineers communicate design decisions and how managers structure work breakdowns, and is important for creating software product lines. So a demand exists from computer science and software engineering instructors who want to understand the importance of software architecture in the creation of software systems and wish to learn how to incorporate software architecture concepts into their curricula. The Software Engineering Institute has developed a collection of architectural methods and practices for designing software-intensive systems that meet intended business and quality goals.

At the first workshop in 2004, Jones recalls, participants were asking "What is software architecture?" "How should it be built?" along with many other elementary questions.

"This year we saw attendees answering, in depth and detail, the questions raised at that first event. There was a dramatic difference in the degree of familiarity and sophistication regarding the significance of architecture and designing it properly."

During the first two days, the workshop offers a course from the series designed by SEI researchers that promotes architecture-centric practices. Such practices shape system development according to the architectural decisions that support the system's quality attribute goals. The workshop's final day involves brainstorming and in-depth discussion about integrating best architecture practices into the classrooms of the attending instructors.

The workshop originated as part of the SEI strategy for transitioning SEI architecture research into the community via graduate and undergraduate education. About 50 instructors have attended from 40 institutions over five years. Many of these are repeat attendees.

That first workshop offered an introductory course, *Software Architecture Principles and Practices*, which presents the basic concepts of software architecture, including its relationship to system qualities, along with architecture definition, evaluation, and documentation, and architecture as a reusable asset. This course was also offered during the next two workshops, to meet the pronounced demand for basic knowledge.

As attendees gained familiarity with the basic architecture-centric principles and relayed them to their students, they expressed the desire for more in-depth course content. In 2007 a more advanced course was offered, *Documenting Software Architecture*, which prescribes documentation practices that meet the needs of the entire architecture stakeholder community. The third day's brainstorming session focused on the inclusion of these architecture documentation concepts and methods in academic courses.

This year's workshop offered another advanced course, *Software Architecture Design and Analysis*, which provides in-depth coverage of the concepts needed to make effective design decisions and to successfully analyze a software architecture relative to desired system qualities. The third day's agenda featured the demonstration of two architecture development tools developed at the SEI. The first was ArchE (ArchitectureExpert), a design assistant tool that helps architects explore architectural designs driven by quality attributes and that is ideally suited for classroom use. The second demonstration presented the Predictable Assembly from Certifiable Code (PACC) Starter Kit (PSK), an integrated set of software development and analysis tools that demonstrates how existing technologies can be integrated to deliver objective confidence in predictions of system behavior. As in previous years, the third day included sharing of ideas on how attendees might incorporate course topics and other architecture-centric principles into their curricula.

Jones notes the third-day discussions have became increasingly sophisticated since the workshop began, involving more complex architecture issues as well as the challenges of escalating integration of SEI material into participants' classrooms. "Whereas the early workshops generally centered on the question 'How can we do this?' the 2008 discussions centered on 'Here's how we do this in my program.'"

In 2008, all participants reported the incorporation of architecture-centric concepts

into their curricula. Repeat attendees shared thoughts on how previous workshop topics had been applied in their programs. Cui Zhang, Professor at California State University Sacramento told how she had modified her course to incorporate material from the Documenting Software Architecture course and now needed to think about how to integrate the material from the Software Architecture Design and Analysis course. Bhabani Misra stated that six courses in the Graduate Programs in Software at the University of St. Thomas, where he is associate professor and Program Director, have been impacted by the software architecture workshop. He has recently incorporated software documentation principles into his teaching. Martin Barrett has seen increased inclusion of architecture concepts in software design courses at East Tennessee State University, where he is assistant chair of the Computer and Information Science Department. Ideas on architectural patterns, styles, and quality attribute-driven design have been integrated. Architecture is also addressed in a Web Services course and Distributed Systems course.

The 2009 Architecture Workshop for Educators will offer a two-day session of the *Software Product Lines* course, which builds on a foundation of software architecture concepts. Judging by the workshop's past success, principles that promote good architecture will be flowing into classrooms across the world and shaping the practices of future practitioners for a long time to come.

For more information

Contact Us

info@sei.cmu.edu

412-268-5800