Is Agility a Passing Fad?

Philippe Kruchten

Department of Electrical and Computer Engineering, University of British Columbia

Agile development methods such as eXtreme Programming (XP), Scrum, Crystal or Lean are conquering the software development world. They do bring major benefits to software development, with a renewed emphasis on the people, and they are showing great successes. However, they are no panacea; there is no "one size fits all" in software development methods. Agile methods work well out of the box in a rather limited "sweet spot". But it is very unfortunate that too many advocates of agile development practice are preaching good practices, but completely removed from the context in which they were proven to be successful. So if your projects are not typical, should you stay away from agile methods? In this seminar I will introduce a framework to analyze the context of a given software project, around the eight concepts of: size, team distribution, governance, criticality, age of the system, business model, stable architecture, and rate of change; and then use this framework to select and adapt agile practices suited to the specific context of a project. I will also show that in some circumstances, the notions of software architecture and agile methods are quite complementary and support well each other. The dilemma is not agile or waterfall, agile or discipline, agile or plan driven. The dilemma is rather between adaptation and anticipation: how much uncertainty do we have to resolve? We are entering a post-agilist era, where practitioners are looking at agile methods with a more critical look, and will complement the practices adequately with practices from the "old ways", wherever appropriate. Do not throw away your PMP certification yet. But do not think agility will pass; it is here to stay, as a matter of fact.

Biography

Philippe Kruchten is professor of software engineering in the department of electrical and computer engineering of the University of British Columbia (UBC), in Vancouver, Canada. He holds an NSERC chair in design engineering. He joined UBC in 2004 after a 30+ year career in industry, where he worked mostly in with large software-intensive systems design, in the domains of telecommunication, defense, aerospace and transportation. Some of his experience is embodied in the Rational Unified Process (RUP) whose development he directed from 1995 till 2003, when Rational Software was bought by IBM. RUP includes an architectural design method, known as "RUP 4+1 views". His current research interests still reside mostly with software architecture, and in particular architectural decisions and the decision process, as well as software engineering processes, and the application of agile processes in large and globally distributed teams. He is a senior member of IEEE Computer Society, member of ACM and INCOSE, the founder of Agile Vancouver, and a Professional Engineer in British Columbia. He has a diploma in mechanical engineering from Ecole Centrale de Lyon, and a doctorate degree in informatics from Ecole Nationale Supérieure des Télécommunications in Paris.
Writing specifications (contracts) as part of software development has always been a dream. But, notoriously, contracts written as mere comments can easily contain errors or become stale, and are therefore unreliable. With a suitable format for writing contracts and tools that do something with the contracts, contracts can become meaningful. In this talk, I present some experimental contract notations for .NET and other code, along with some verification tools we have built for use with the contract, and report on some experience with these.

Biography

Rustan Leino is a Principal Researcher in the Research in Software Engineering (RiSE) group at Microsoft Research. He is known for his work on programming methods and program verification tools. At Microsoft Research, he has led the Spec# project, which brings enforced pre- and post-conditions to the .NET platform, and is the architect of the Boogie program verification framework, which underlies several program verifiers for Spec#, C, and other languages. Previously, Leino led the ESC/Java project at Compaq SRC, and worked on specifications on the pioneering ESC/Modula-3 project at DEC SRC. Before getting his PhD (Caltech, 1995), Leino wrote and designed object-oriented software as a technical lead in the Windows NT group at Microsoft. In his spare time, he plays music and teaches cardio exercise classes.
ASWEC 2010 Keynote Presentation 3

Hacking in FORTRAN in the 21st Century

Gareth Cronin

Kiwiplan NZ

FORTRAN was first released in 1957 and the VT52 was developed in the late 1970s. As a business, Kiwiplan New Zealand relies on the revenue from applications written in FORTRAN with 80 x 24 character-mode user interfaces, despite the company having spent the last 12 years developing graphical user interfaces using object-oriented languages in sophisticated modern development environments. Modern software engineering processes, practices, technologies and tools have brought some benefits, but not to the degree that Kiwiplan's owners would have liked. After 30 years in the software industry, supplying a sometimes technologically-backwards niche industry, Kiwiplan finds itself with three technology platforms, four million lines of code, a home grown software process, a dozen products, and a staff of over 45 developers.

Biography

Gareth Cronin is the Development Manager at Kiwiplan NZ, a software house that grew out of a corrugated packaging company in South Auckland 30 years ago. Kiwiplan's products are found in over half of the corrugated packaging business in North America and are spread all over the globe in over 500 manufacturing plants. Gareth has a BSc(hons) in Computer Science and an MBA from the University of Auckland, which he gained after an aborted career as piano teacher and rock musician. Having worked as a Java developer and software architect for a number of years, Gareth now looks after Kiwiplan's 45 developers and sets the technical and development process direction for the company.