

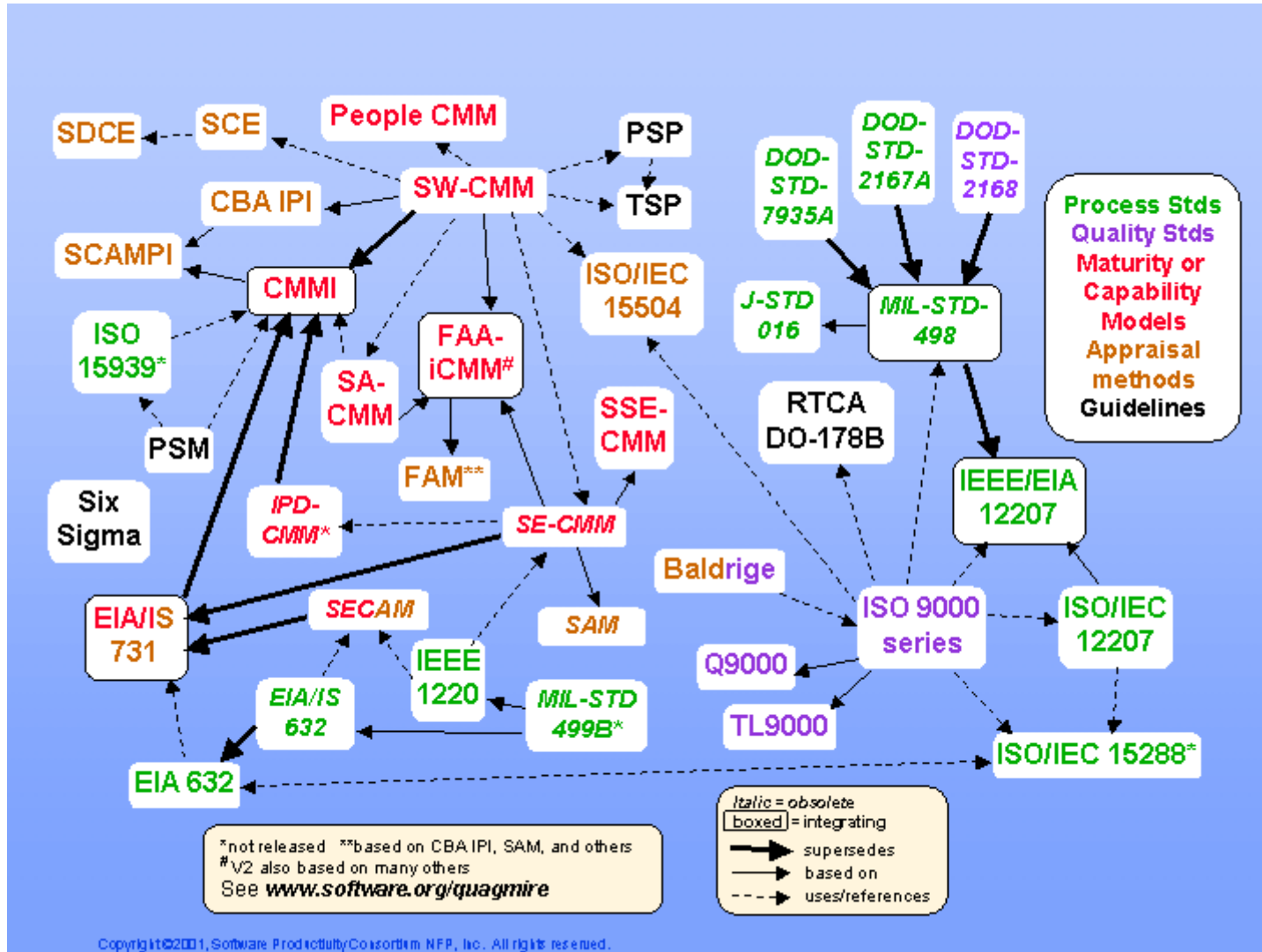


END OF AGILE

Pekka Abrahamsson
VTT TECHNICAL RESEARCH CENTRE OF FINLAND
25.11.2008, VTT, Olio-päivät '08, Tampere, Finland



PROCESS IMPROVEMENT MODELS



Source: <http://www.software.org/quagmire/>, Aug-2005

Industry
best-practice?

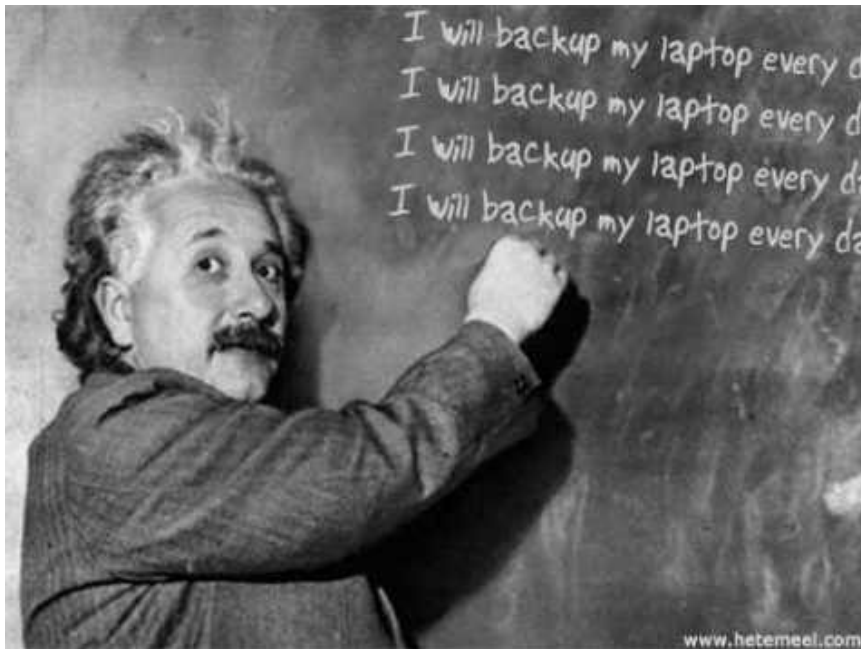
PROCESSES CANNOT BE COPIED!



We should know this?

Malouin, J. L. and M. Landry (1983). "The miracle of universal methods in systems design." *Journal of Applied Systems Analysis* 10: 47-62.

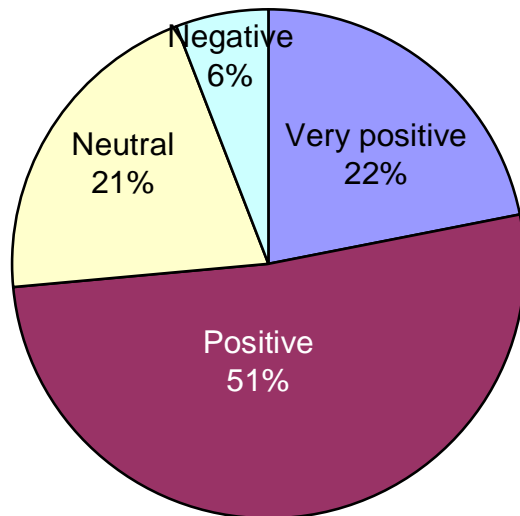
A FUNDAMENTAL RE-REALIZATION



- "The quickest way to improve productivity is to focus on people" (Boehm, 1983)
- "Everyone knows the best way to improve software productivity and quality is to focus on people." (Editor, American Programmer, 1990)
- "People issues determine project's speed" (Cockburn, 2008)

SUCCESS OF AGILE METHODS

- A proof that agile fits to embedded SW domain:
- AGILE-ITEA research project (2004-2006)
 - Domain: agile software development of embedded systems
 - 68 pilot projects executed in 2004-2006
 - 1800 engineers in 17 companies involved
 - 73% of the pilot results either positive or very positive



Source: Itea innovation report 2006

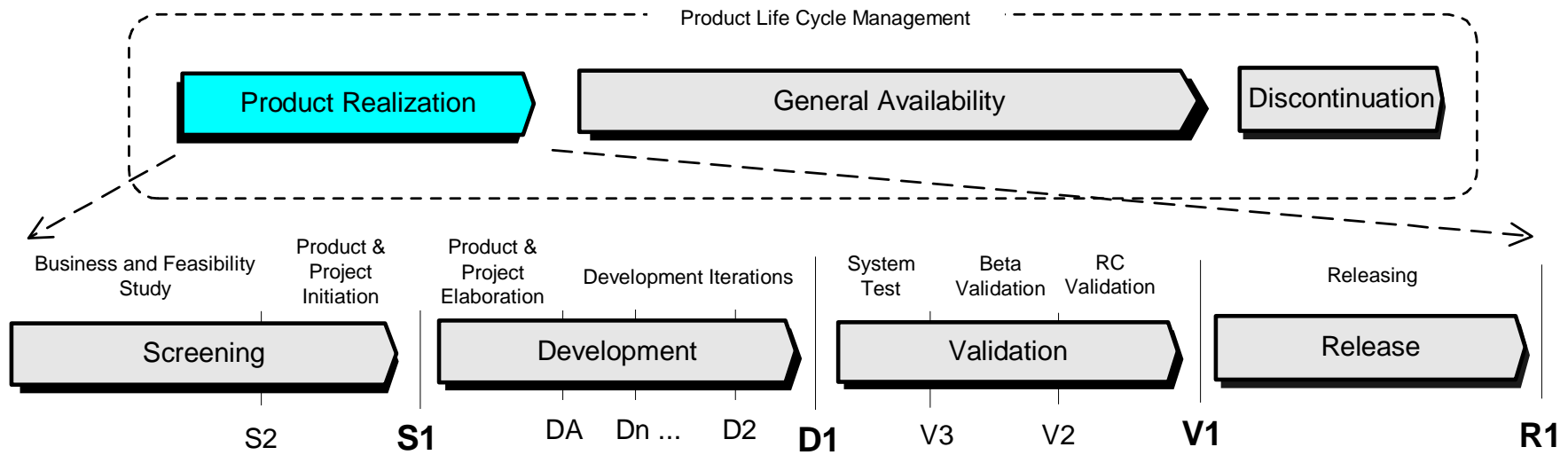
DEVELOPERS ARE ONBOARD!



Source: Nokia Networks, publicly available at:
http://www.odd-e.com/articles/2006/nokia_agile.pdf



F-Secure's starting point

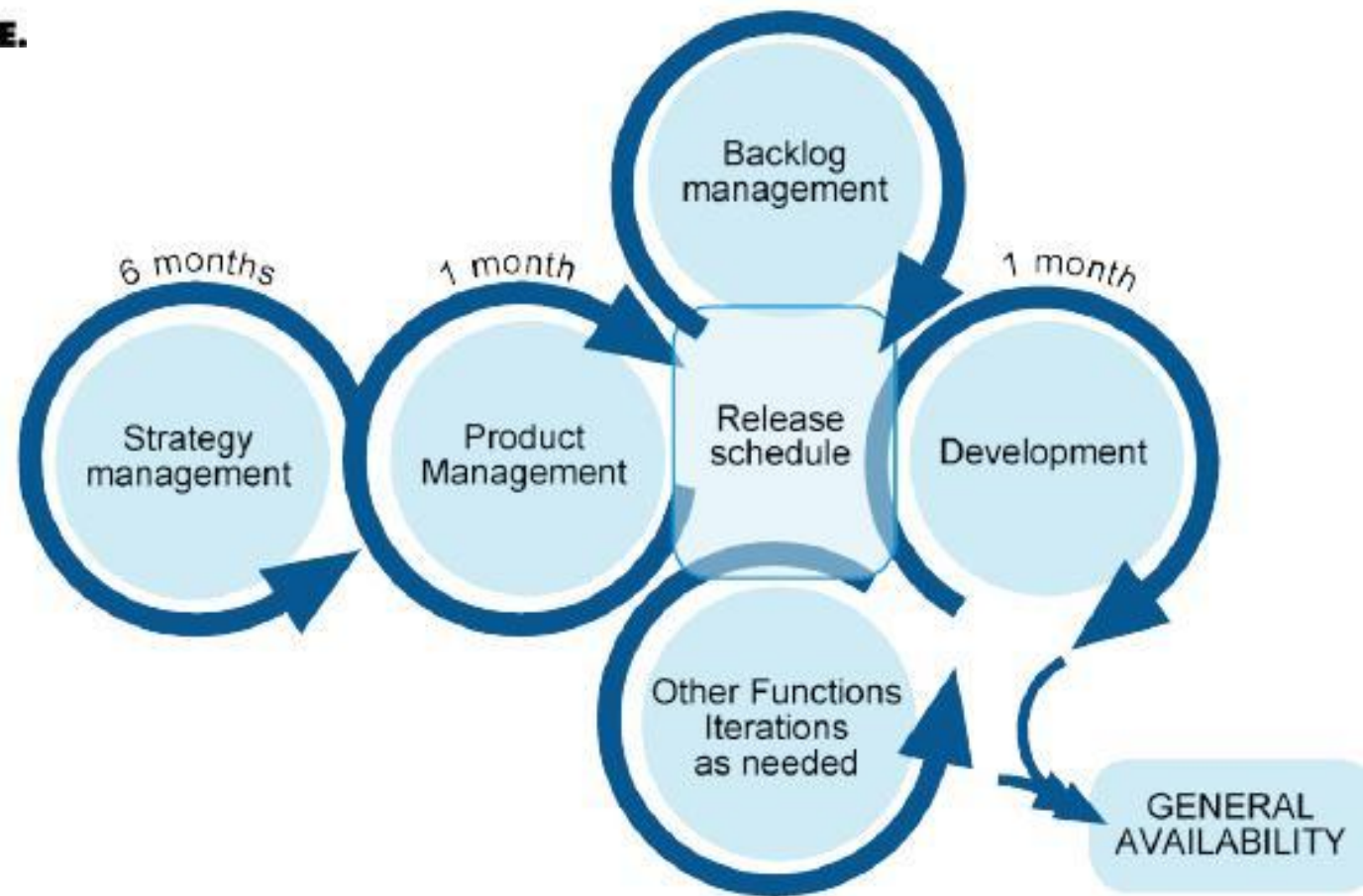


Product life-cycle and product realization cycle

Source: Agile-ITEA newsletter 1/2006
















































F-SECURE'S NEW MODEL: F-LEX



Source: Agile-ITEA newsletter 1/2006

FLEXIBLE GLOBAL PRODUCT DEVELOPMENT 2007-09

	Application partners	Technology partners
Spain	     	   
Netherlands	 	
Finland	     	   
Sweden	 	
Ireland	  	  
Norway	    	
Belgium	   	
Israel	 <p>http://www.flexi-itea2.org</p>	 

ABRAHAMSSON'S THESES 25.11.2008



- Agile, as we have come to know it, has become to its end
- Agile, as we know and read of it, fails conceptually, philosophically and empirically
- Rise of professionalism: Happy, efficient and profitable software organizations' models are emerging

CURRENT SITUATION

- In 1999, Extreme Programming emerged
- In 2001, the agile manifesto was published
- In 2005, 16% of companies used agile methods (Standish Group)
- In 2008, agile software development has crossed the Chasm! (Ambler, 2008)
- In 2009, IEEE 1648 completes its work, ISO begins the standardization work



HENCE, IT IS NOT A SURPRISE THAT

- 69% of organizations claim of having adopted agile methods (Ambler 2007)
- 70% of developers claim of using most aspects of agile methods (Rico 2007)
- And, finally: 99% of organizations claim to be using iterative development (Ambler 2007)

SUMMARIZING THE CURRENT SITUATION

- Scrum clearly method dominates agile spectrum
 - XP practices highly recommended to be used
- Other methods exist: e.g. Crystal, DSDM, FDD, ASD, Mobile-D, etc. are around but in minority
- "Agile software" search on amazon.com results in 1268 book hits! (Amazon.com, search performed 19.8.2008)
- Scientific knowledge is still scarce (33 primary studies, Dybå & Dingsoyr 2008).
 - Indeed, the science is also seriously lagging behind practitioners.
 - However, the situation is not unique within the field of software engineering

CHALLENGING AGILE



- Agile suffers severely from three perspectives. These are:
 - Conceptual confusion
 - Philosophical interpretation
 - Empirical implementation
- I will briefly address all these viewpoints

CONCEPTUAL CONFUSION

- Agility is an organizational characteristic (kruchten, 2001)
- There are 500+ factors impacting on this specific characteristic (kettunen, 2006)
- In Manufacturing industry alone there are more than 17 competing definitions of agility (Iskanius 2006). In software there is only 1 (by Conboy and Fitzgerald) or another .. Manifesto.
 - Software agility explains, perhaps, 10-15% of those 500+ factors. Agility is a business concept.
- Thus, conceptually, agile software and agility are not related, per se.
- In fact, as it is well known, term "agile" was the second choice for agile manifesto developers as well..

EMPIRICAL IMPLEMENTATION PROBLEMATIC

- One [or any] method does not fit all situations (Malouin and Landry, 1983)
 - Therefore, all agile implementations are, by definition, adaptations of concepts, ideas, techniques and practices of agile family of "stuff"
 - Strive is for a minimalistic view of software artefacts (= less is better)
 - In traditional sw development the act of *adaptation* is called "method tailoring".
 - RUP "failed" due to the fact that the proposed approach was not tailored but rather imposed or adopted as such
 - Note, RUP method includes all possibilities probably needed in SW development

EMPIRICAL IMPLEMENTATION PROBLEMATIC, cont.

- Traditionally method tailoring is (or should be) supported by a reference framework coupled with experience and feedback
- There is no reference framework whatsoever in agile software development.
 - No agreement on which practices are agile, per se
 - Debates whether such a framework can be developed as agility is a relative concept as such
 - Agile principles/methods not changed since their inception
 - Scrum knowledge proprietary knowledge, which cannot be accessed. Shared in "secret" meetings.

PHILIPS

Modena Agile practices in use

- SoftFab: Automating the build, test and reporting process
 - daily builds, static and dynamic tests, doc generation, statistics: code coverage, confidence level
 - Gives focus on the real tasks, automated work environment
- RaPiD7: used for IP generation and design documentation
- TDD: make test first, code stays smaller and more efficient
- Pair programming: on cross component level or difficult parts
- Pair review: buddy system reviewing major implementations
- Scrum: used during focus time, dead line approaching
- Daily integration + weekly release of working software
- Information Radiator: SoftFab, code size, product start-up time, PR/CR statistics, team organization
- Reflections: done once, plan to do it (bi)-monthly
- Customer collaboration: Weekly review and discussion with customer

PHILOSOPHICAL INTERPRETATION

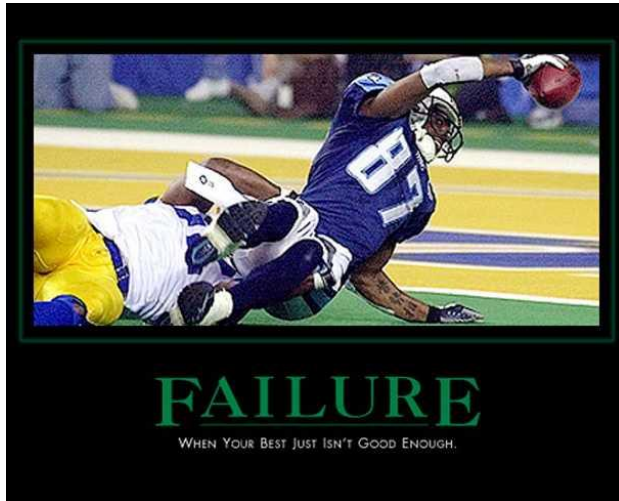
- The problem lies in the fact that due to the lack of support for composing "an agile process", "anything goes", as Fayerabend 1976 suggested, as long as it is "agile".
 - "Great scientific discoveries are only made by breaking the rules" (Fayerabend 1976)

HOW TO FAIL
WHEN
IMPLEMENTING
SCRUM?

**A
BIG
FAT
FAILURE**

*(Or more accurately, a tall,
really, really skinny one)*

HOW TO FAIL – SOME RECIPES



- Do not train people, especially leave the developers and managers without the training support
- Do not think about the impact beyond R&D settings
- Do Scrum by the book and do not change it
- Agile/Scrum is a universal solution. Do not, in any case allow cultural differences to interfere.
- Do not bother defining what agile means in your organization
- Force everyone in the organization to use Scrum
- Expect the “normal” trace to be there
- Do not plan ahead. 2-4 weeks is sufficient

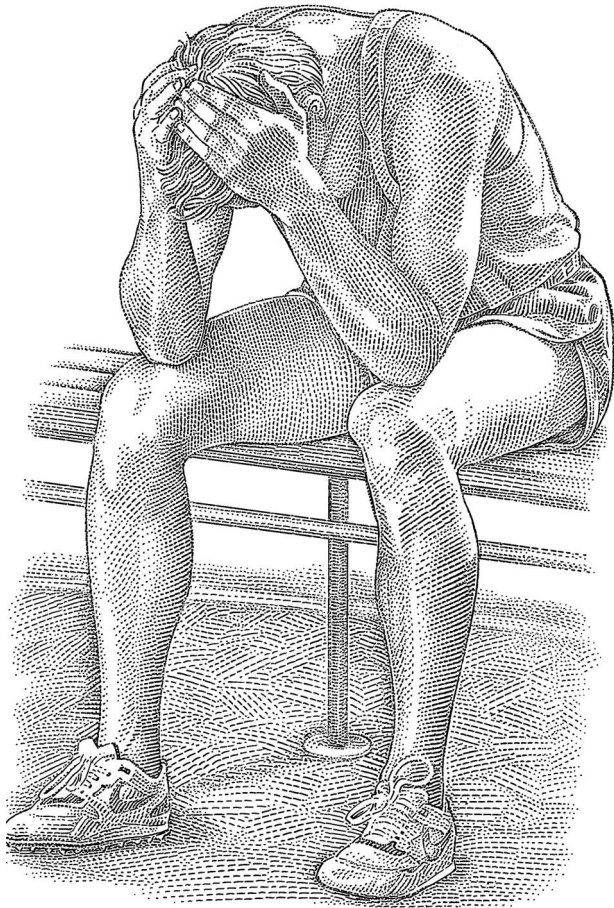
HOW TO FAIL – SOME MORE RECIPES

- Mandate the use of Test-Driven Development as company level de-facto approach. We know that it produces a lot of test cases.
- Do not develop your own development model. Insist to use only the agile approach.
- Do not produce metrics to the management (other than burn down) since they do not really need it, even if they ask for it.
- When applying Scrum, retain maximum amount of the same roles & activities as before. The change should not feel at development or management level too much.
- If a team fails to continuously to live up the expectations detailed in the sprint backlog, please do not disturb them. They will learn as they are in self-organizing mode.
- If all or most your developers are juniors, agile and Scrum will be the optimal solution for you.
- Keep rewarding individuals and champions for their performance, not the team(s)



HOW TO FAIL – THE FINAL RECIPES

DEMOTIVATION.COM



- You must forget the rest of the processes and software process improvement altogether when adopting Scrum
- Before starting, spend at least 3-4 months in careful “agile analysis”. Otherwise you start with the wrong foot.
- Let the architecture emerge while doing the work. This is true also to settings beyond Mickey Mouse environment. We can always refactor the problems out.
- Do not use any sort of external help like consultants or enablement partner. It is crucial to self-learn all the agile aspects.
- Do not collect metrics (time, size, defect), since they are not part of the agile manifesto
- Let the agile manifesto to guide you throughout the adoption process and beyond
- When you are failing, remember to keep the agile achievements secret

THE NEAR FUTURE: TO BE LAUNCHED IN 2009



AGILE POSITIONING SYSTEM™

THE EUROPEAN LEADING EDGE ON
AGILE TRANSFORMATION

A PRINCIPAL FLEXI PROJECT OUTCOME



ITEA2
INFORMATION TECHNOLOGY FOR EUROPEAN ADVANCEMENT

THANK YOU!

Contact me at:
Pekka.Abrahamsson@vtt.fi
Tel. 040-5415929

