AGILE SOFTWARE DEVELOPMENT: INTRODUCTION, CURRENT STATUS & FUTURE

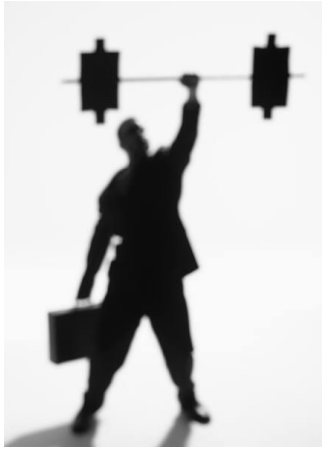
Pekka Abrahamsson

23.11.2005 Jyväskylä

EXAMPLE OF AN INNOVATIVE LEAP - 1

F-SECURE®
BE SURE.

- Due to the adoption of an agile production frame, F-Secure's mobile division in a new product development achieved
  - § 3x reduction in lead-time,
  - § 50x better quality, and
  - § 5x cheaper!
- Innovative leap due to the acceptance of radical variation in product development & management processes



Fact corner:
- SME of 250 developers
- Mobile & desktop sw
- Products sold globally

"From the management point of view, the results were amazing."
Jari Still, Director, F-Secure, Finland
AGILE-ITEA Newsletter #2, 2005

EXAMPLE OF AN INNOVATIVE LEAP - 2

- The use of diverse agile solutions lead Philips to:
  - Productivity: 8x Faster than industry average
  - Quality: 3.5x Better than industry average
  - Customer satisfaction: 4.9 in 5 point scale
- Softfab, Rapid7, reflection, sprints, … from the agile toolbox

Fact corner:
- 400+ Kloc
- 17.5 person years/1year

INNOVATIVE LEAP EXPLAINED

- "The actual processes adopted were neither novel nor particularly inventive. Rather they had the virtues of being easy to explain and relatively easy to comply with, with goals easily describable as having been met or not."

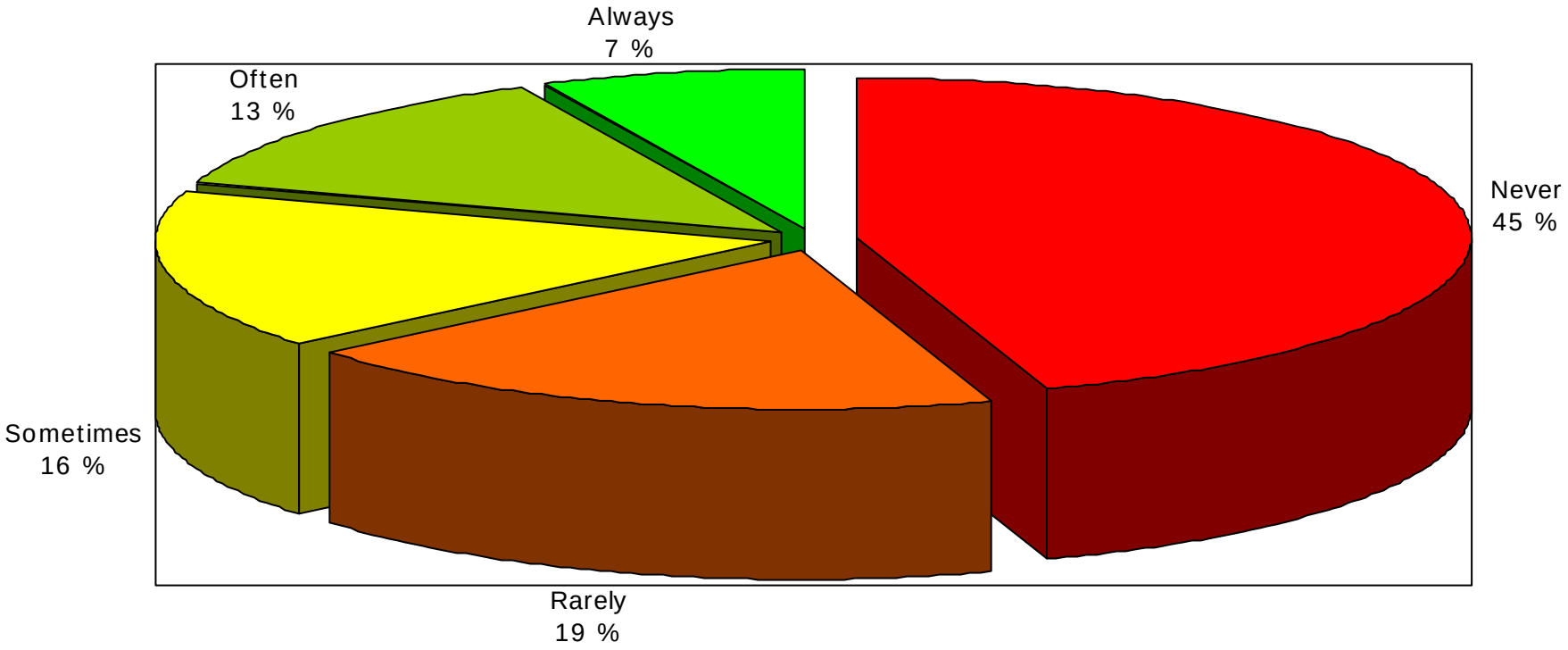
Ward et al. (2001)

Part I: Agile philosophy & rationale

HISTORICAL PERSPECTIVE

- Software Crisis (1960's)
  - Software intensive systems delivered late, over budget and do not meet the quality requirements
- Solution attempt #1: **Structured Methods** (in 1980's)
- Solution attempt #2: **Object-oriented methodologies**
- Chronic Software Crisis (1990's)
  - Software intensive systems still delivered late, over budget and do not meet the quality requirements
- Solution attempt #3: **Software process improvement**
- Solution attempt #4: **Agile development methodologies**

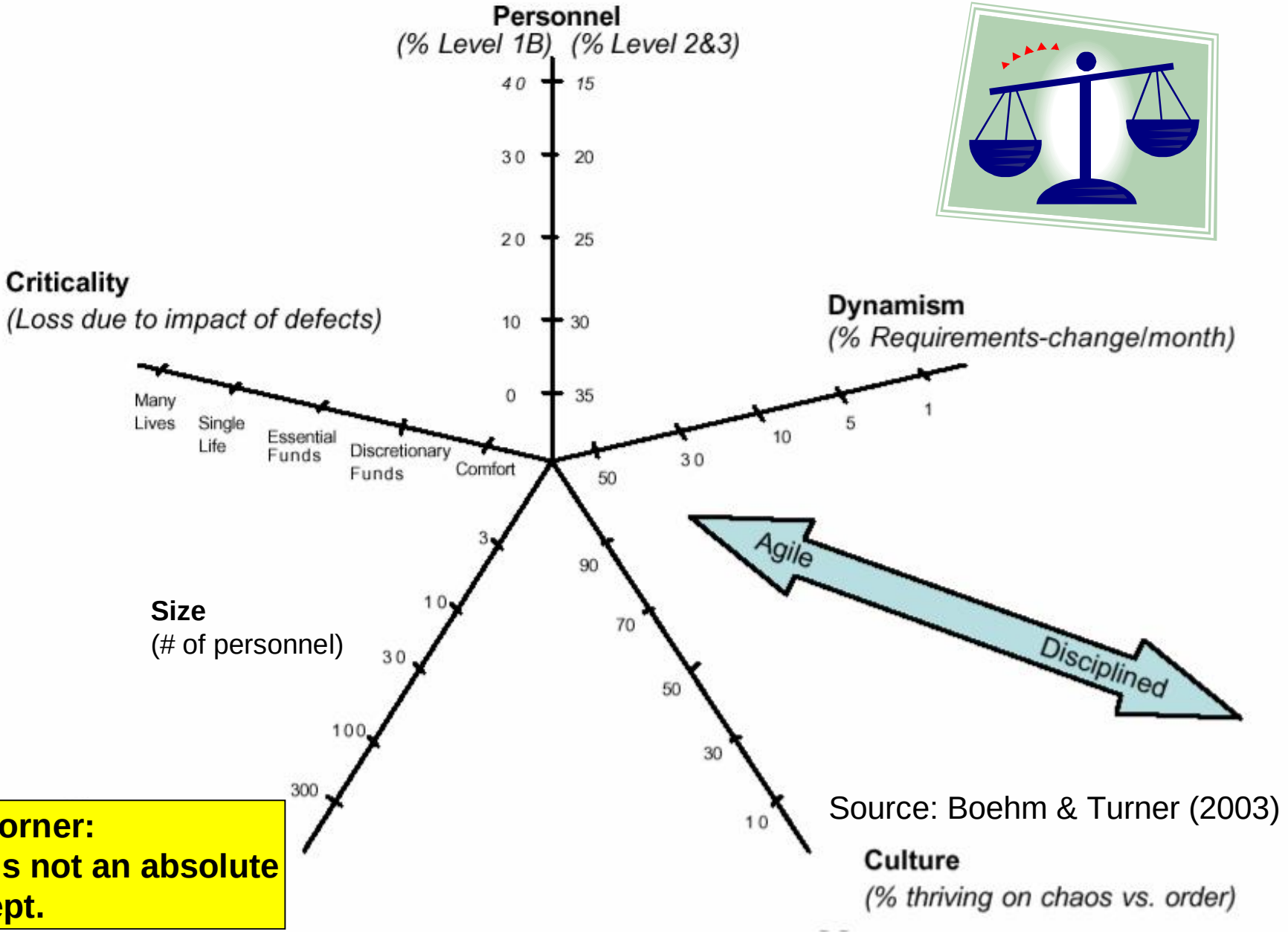
FEATURE USAGE RATE



Fact corner:
> 60% features never or rarely used!

Source: Jim Johnson of the Standish Group, Keynote Speech XP 2002

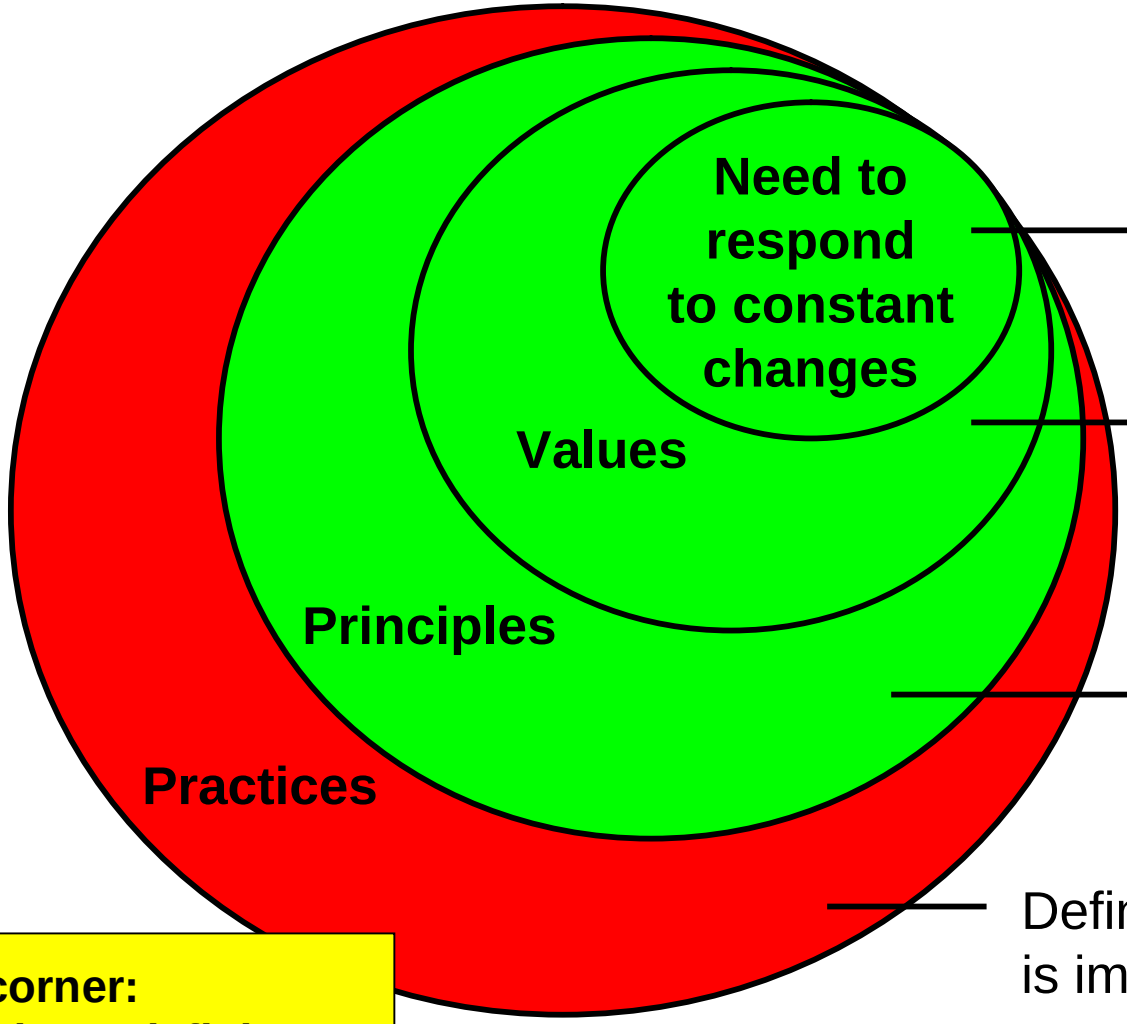
HOW AGILE CAN YOU BE?



Source: Boehm & Turner (2003)

Fact corner:
Agile is not an absolute Concept.

AGILE THINKING EXPLAINED



The fundamental reason for a “new” paradigm

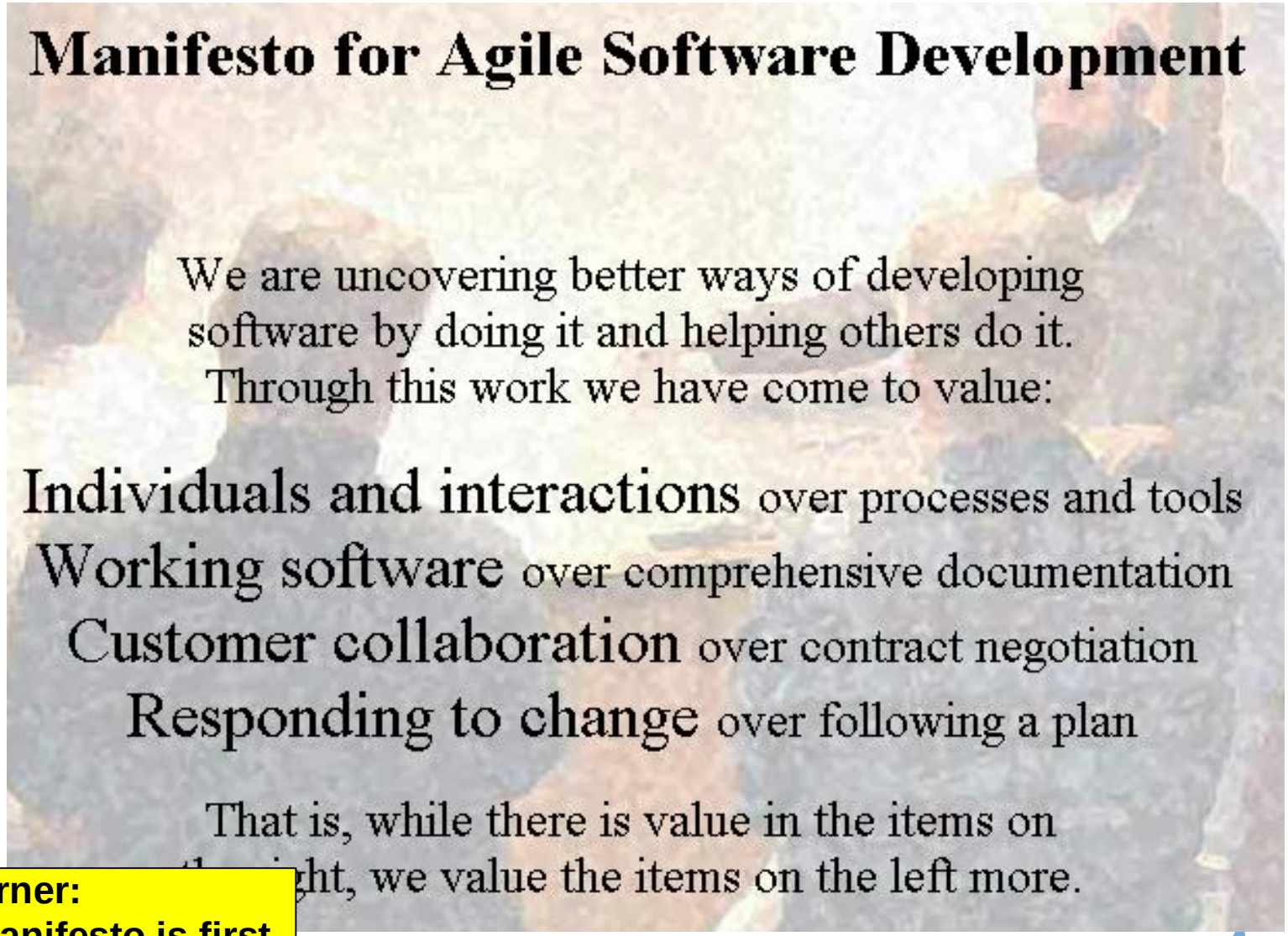
Defines the set of most important beliefs of what is truly important

Defines a set ways to meet the values

Defines in detail how this is implemented in practice

Fact corner:
There is no definite set of agile practices.

AGILE VALUES…



Fact corner:
Agile manifesto is first of its kind in software engineering field

Source: www.agilemanifesto.org

THE 12 AGILE PRINCIPLES (1/3)

DESCRIPTION

1. Our highest priority is to satisfy the customer through early and continuous delivery of valuable software

2. Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.

3. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to a shorter timescale.

4. Business people and developers must work together daily throughout the project

Fact corner:
See principals as "thinking-tools"

SUMMARY

1. Satisfy customer through early and frequent delivery.

2. Welcome changing requirements even late in the project.

3. Keep delivery cycles short (e.g., every couple of weeks).

4. Business people and developers work together daily throughout the project.

THE 12 AGILE PRINCIPLES (2/3)

5. Build project around motivated individuals. Give them the environment and support they need, and trust them to get the job done.

5. Build projects around motivated individuals.

6. The most efficient and effective method of conveying information to and within development team is face-to-face conversation.

6. Place emphasis on face-to-face Communication.

7. Working software is the primary measure for progress.

7. Working software is the primary measure of progress.

8. Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.

8. Promote sustainable development pace.

THE 12 AGILE PRINCIPLES (3/3)

9. Continuous attention to technical excellence and good design enhances agility.

10. Simplicity – the art of maximizing the amount of work not done – is essential.

11. The best architectures, requirements, and designs emerge from self-organizing teams.

12. At regular intervals, the team reflect on how to become more effective, then tunes and adjusts its behavior accordingly.

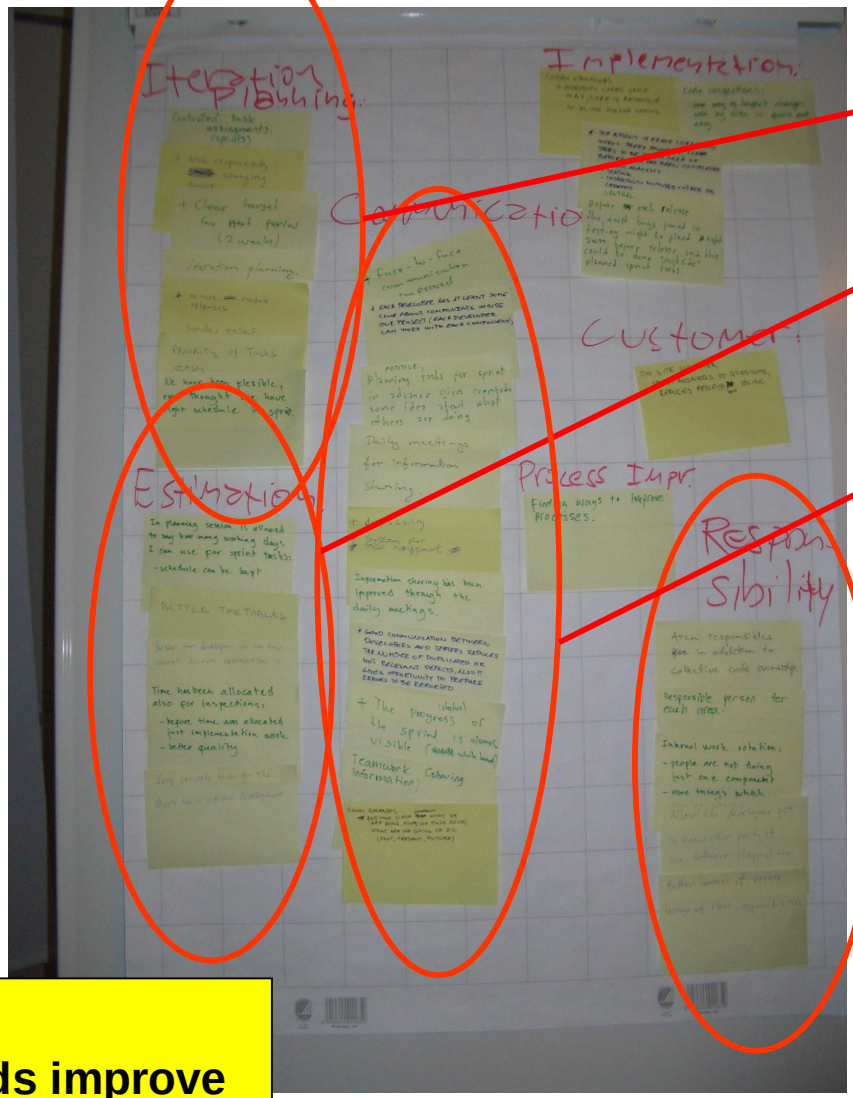
9. Continuous attention to technical excellence and good design.

10. Simplicity is Essential.

11. The best results emerge from self-organizing teams.

12. Team reflects regularly where and how to improve.

What has improved? (Asked from the industrial developers)



Fact corner:
Agile methods improve communication!

Part II: Existing agile software development approaches

EXISTING AGILE METHODS

- Methods for agile software development:
  - Agile software process model [Ayoama, 1998]
  - Adaptive Software Development [Highsmith, 2000]
  - Crystal Family of Methodologies [Cockburn, 2000]
  - Dynamic Systems Development Method [Stapleton, 1997]
  - Extreme Programming [Beck, 1999]
  - Feature-Driven Development [Palmer & Felsing, 2002]
  - Lean software development [Poppendieck x 2, 2003]
  - Scrum [Schwaber, 1995; 2002]
  - … the list is growing every year…
- Combination of approaches:
  - Agile Modeling [Ambler, 2002]
  - Internet-Speed Development [Cusumano & Yoffie, 1999; Baskerville et al., 2001; Truex et al., 1999]
  - Pragmatic Programming [Hunt & Thomas, 2000]

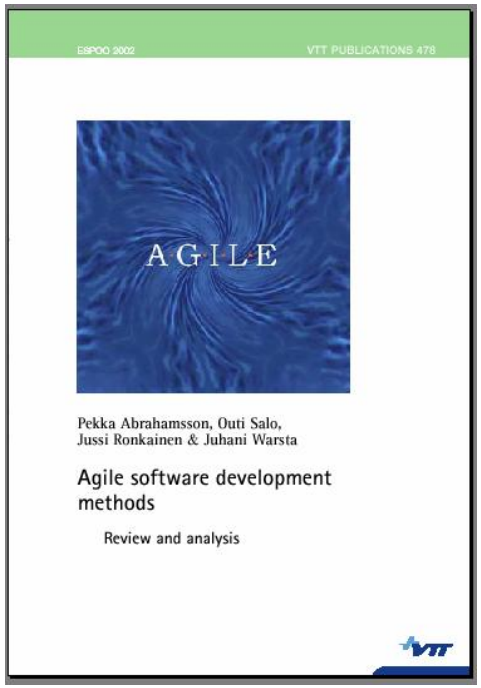
Fact corner: Only 3 out of 9 Methods has Empirical evidence

AGILE PRACTICES: THE TOOLBOX

	Agile practices							
	XP	**Scrum**	**Crystal**	**FDD**	**DSDM**	**ASD**	**Lean**	
1	Planning game	Product backlog	Staging	Domain object modelling	Active user involvement	Iterative development	Seeing waste	Perceiced integrity
2	Small/short releases	Effort estimation	Revision and review	Developing by feature	Teams must be empowered..	Feature-based planning	Value stream mapping	Conceptual integrity
3	Metaphor	Sprint	Monitoring	Individual class (code) ownership	...frequent delivery	Customer focus-group reviews	Feedback	Refactoring
4	Simple design	Sprint planning meeting	Holistic diversity	Feature teams	Fitness for business purpose...		Iterations	Testing
5	Testing	Sprint backlog	Methodology-tuning technique	Inspection	Iterative and incremental development		Syncronization	Measurements
6	Refactoring	Daily scrum meeting	User viewings	Regular builds	All changes are deversible		Set-Based Development	Contracts
7	Pair programming	Sprint review meeting	Reflection workshops	Configuration management	Requirements are baselined...		Options thinking	
8	Collective ownership			Progress reporting	Testing is integrated ...		The last responsible moment	
9	Continuous integration				A collaborative and cooperative approach shared by all stakeholders		Making decisions	
10	40-hour week						Pull systems	
11	On-site customer						Queuing theory	
12	Coding standards						Cost of delay	
13	Open workspace						Self determination	
14	Just rules						Motivation	
15							Leadership	
16							Expertise	

SOURCE FOR AN OVERVIEW OF AGILE METHODS

- Titled “Agile software development methods: Review and analysis”, 2002
- A VTT series publication freely available from http://agile.vtt.fi
- For each method the following aspects are described:
  - Purpose & motivation
  - Roles & responsibilities
  - Development process
  - Development practices
  - Current status & known limitations

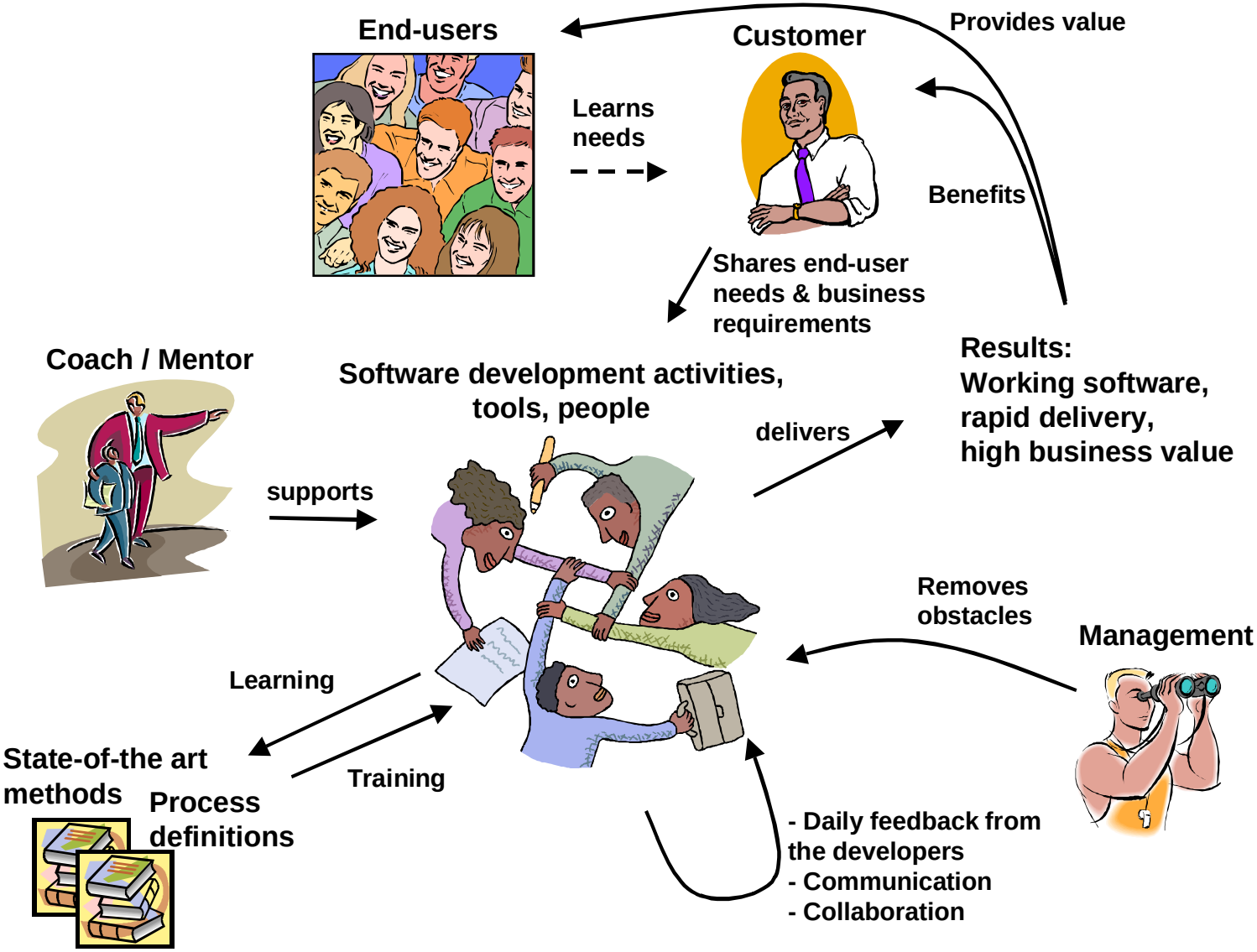


Part III: The Mobile-D™ for mobile software

FORMATION OF A BASELINE: REQUIREMENTS FOR A GOOD SOFTWARE DEVELOPMENT PROCESS

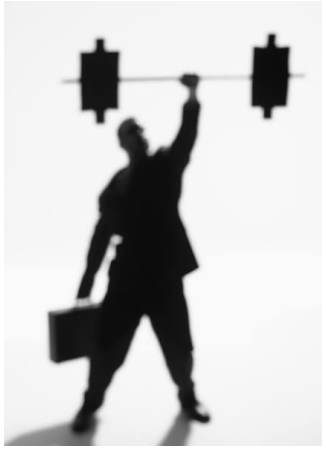
- The development process should
  - provide systematic support for high quality software development (Production frame)
  - be designed for small (and medium) sized development teams (Most teams are small)
  - produce visible results early (Early validation)
  - be easy to learn, transparent, straightforward and adjustable (= makes sense)
  - provide a fit to company's strategic planning (= aligned with the business)
  - meet the needs of standard quality requirements (= be convincing to us & customers)

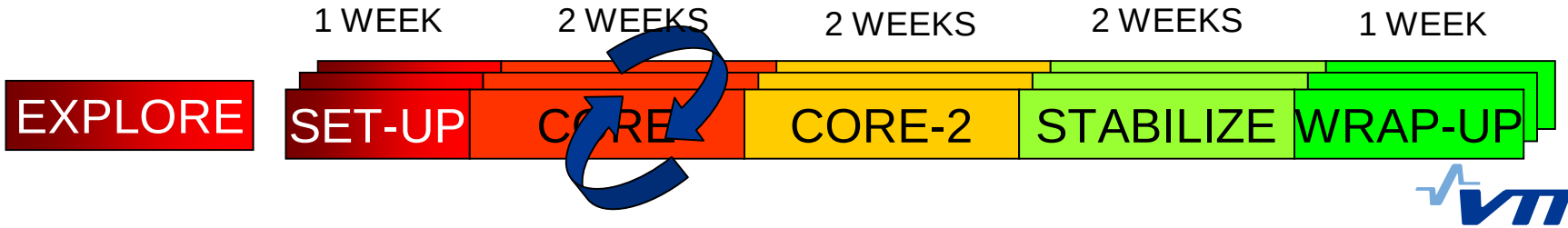
NEW PROGRAMMING ENVIRONMENT



MOBILE-D™ FOR MOBILE SOFTWARE

- Concept: An 8-week agile development rhythm
- Mobile-D™ is based on Extreme Programming (practices), Crystal methodologies (scalability) and Rational Unified Process (coverage)
- Designed to meet the specific characteristics of mobile software development & industry quality standards
- Designed for < 10 developers working in (close to) co-located office space
- Pattern-based version: http://agile.vtt.fi/mobile-d/mobiled.htm



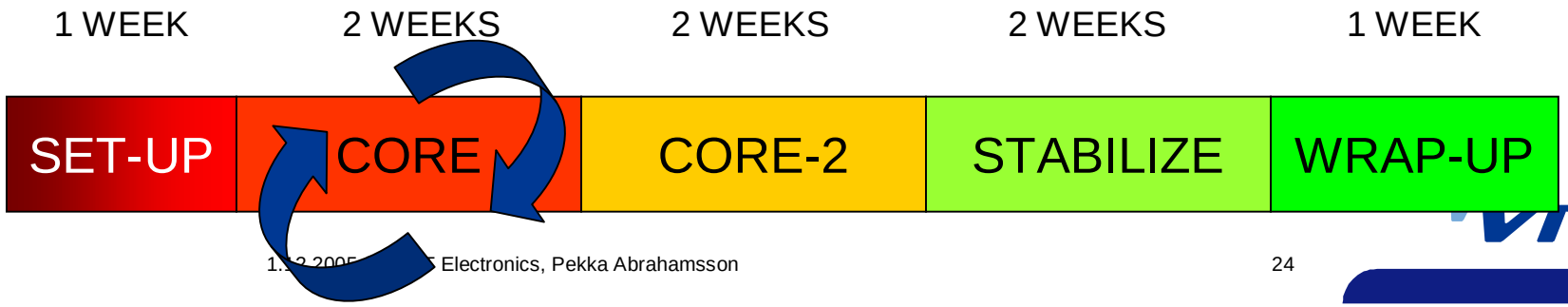


AGILE PROGRAMMING ENVIRONMENT: THE WARROOM APPROACH

Fact corner:
Software development
Is not solo business

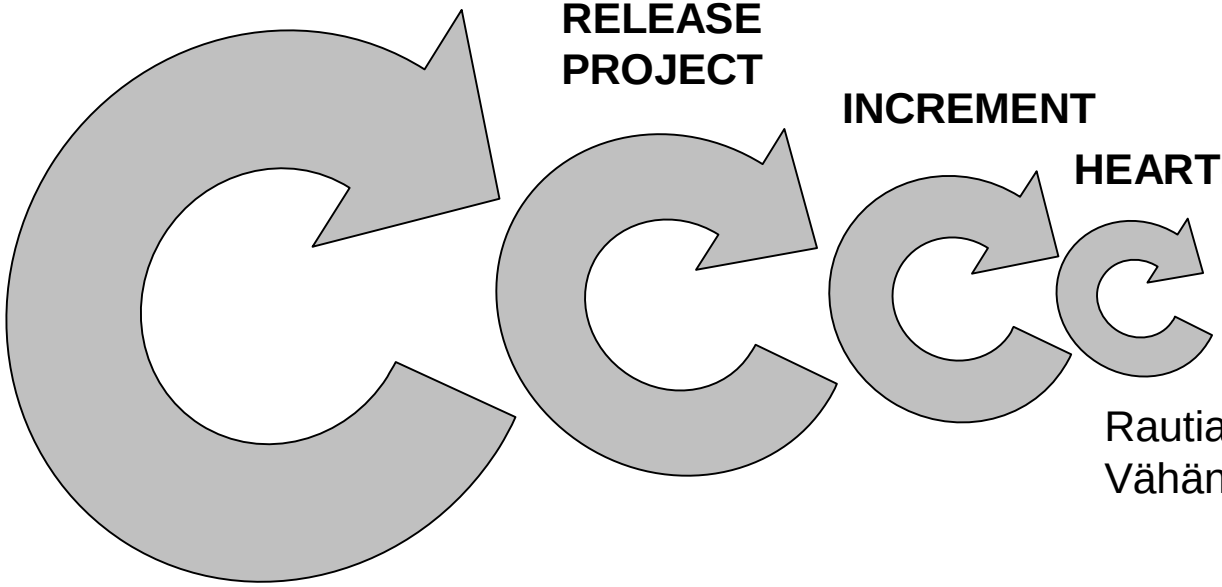
THE PRINCIPAL ELEMENTS OF MOBILE-D

- Requirements: Off-Site Customer
- Planning: Phasing and pacing in Planning Day
- Modeling: Agile modeling
- Architecture: Architecture Line
- Metrics: Time, size and defect
- Documentation: RaPiD7-method
- Improvement: Agile Software Process Improvement
- End-users: User-Centred Focus
- Testing: Mobile Test-First development



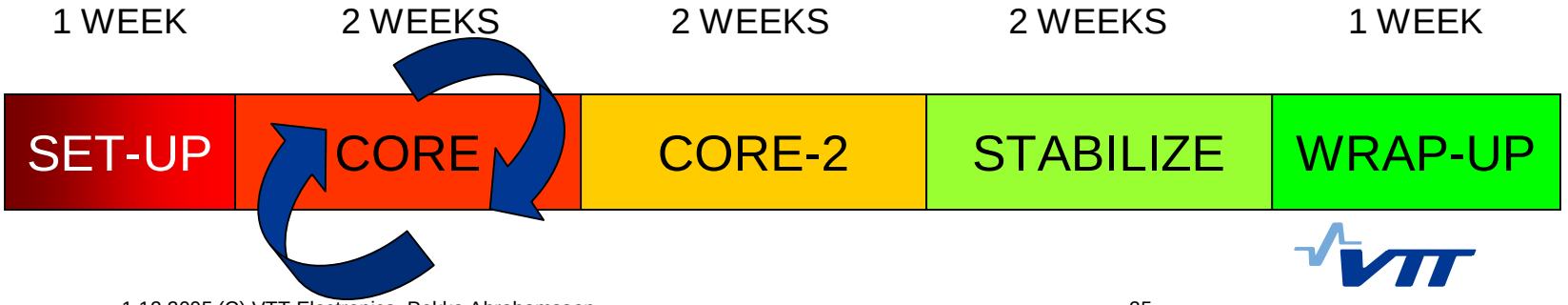
MobileD: FIT TO STRATEGIC PLANNING

4 CYCLES OF CONTROL

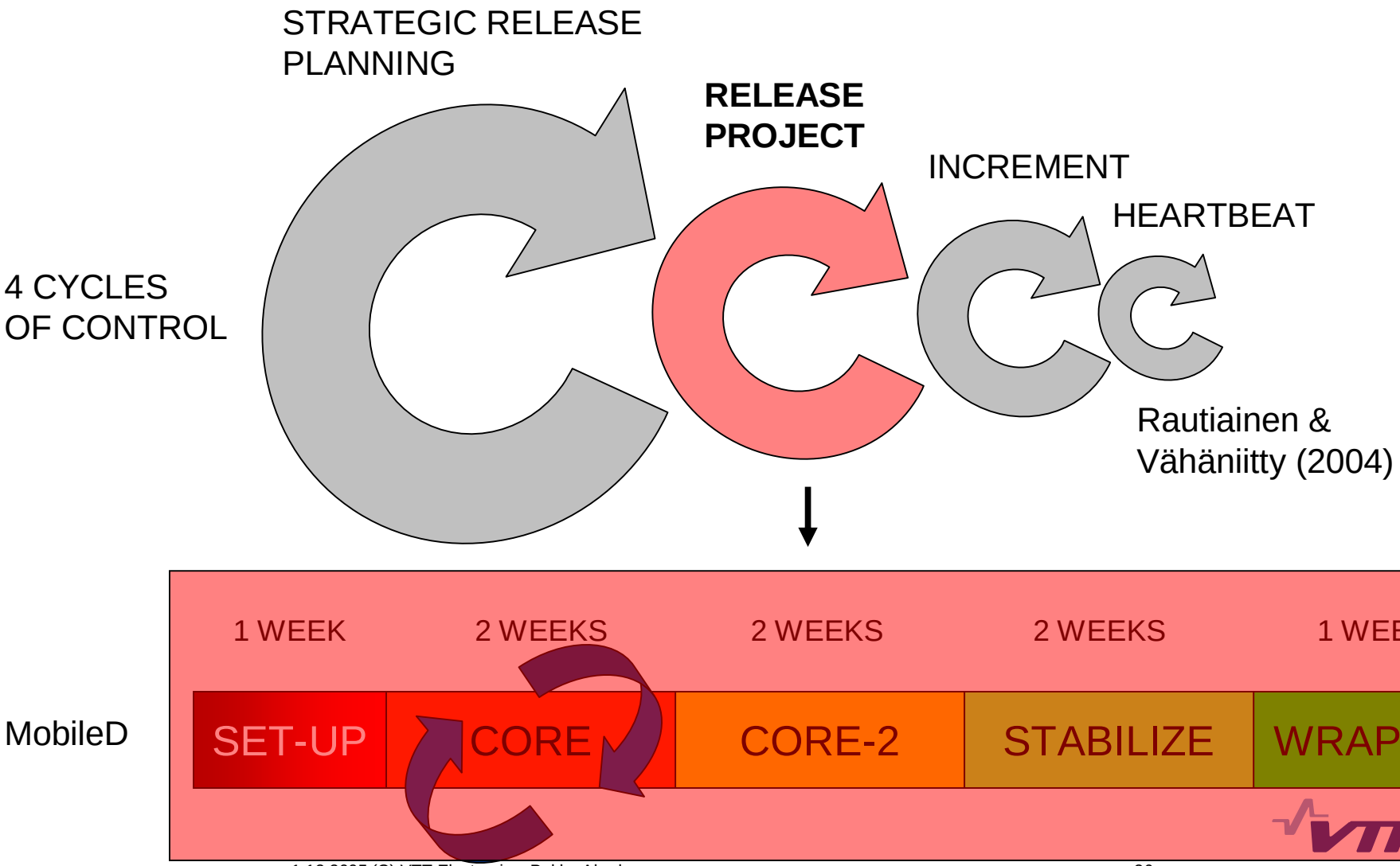


Rautiainen & Vähäniitty (2004)

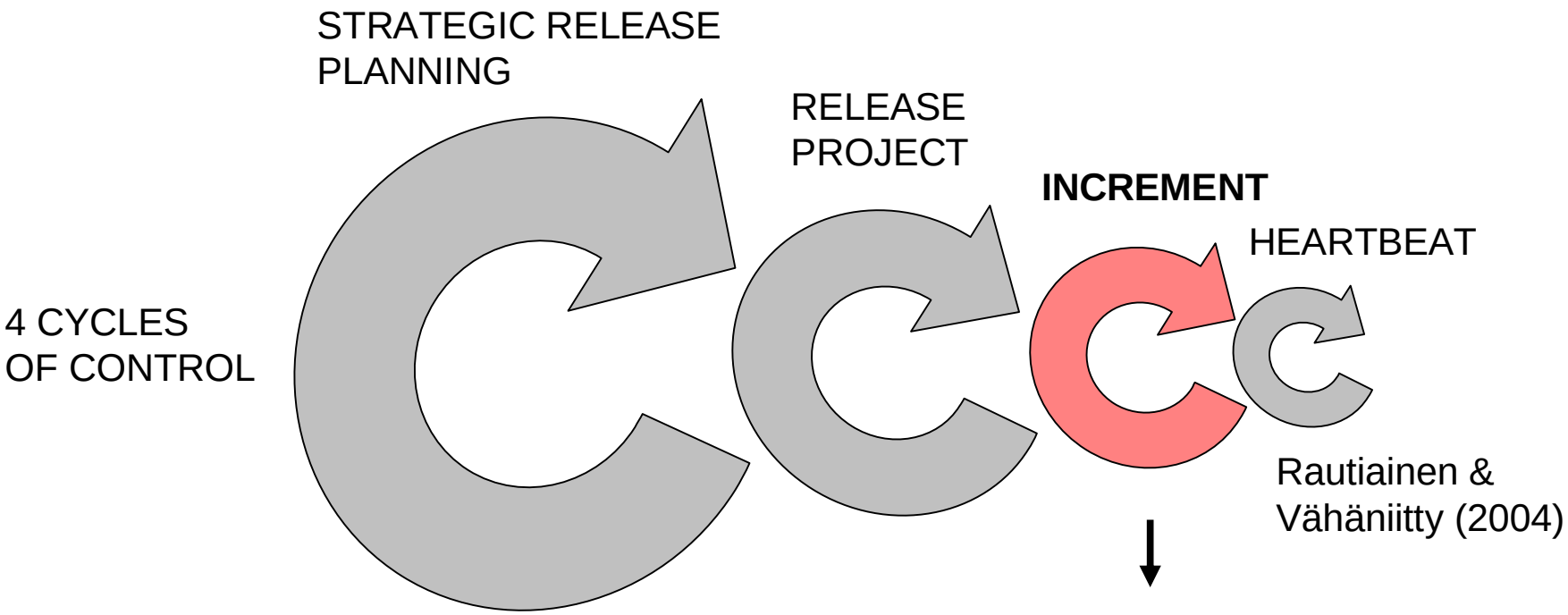
MobileD

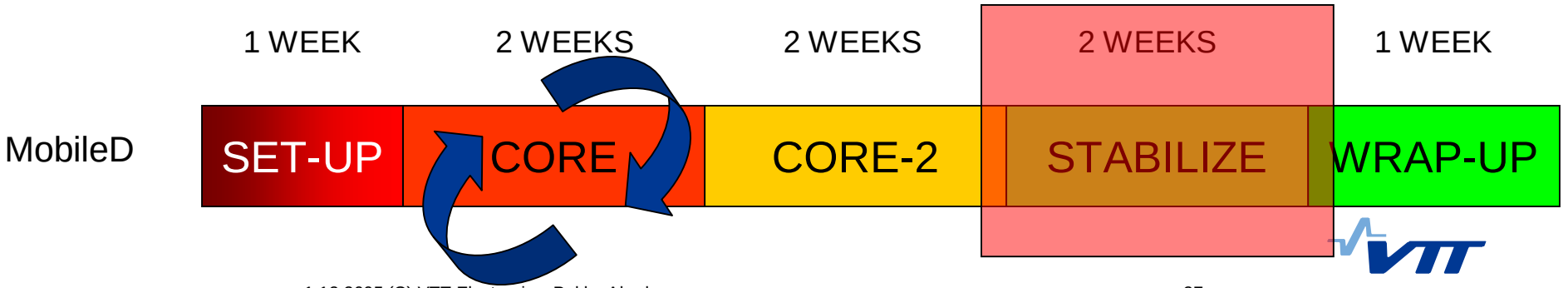


MobileD: FIT TO STRATEGIC PLANNING

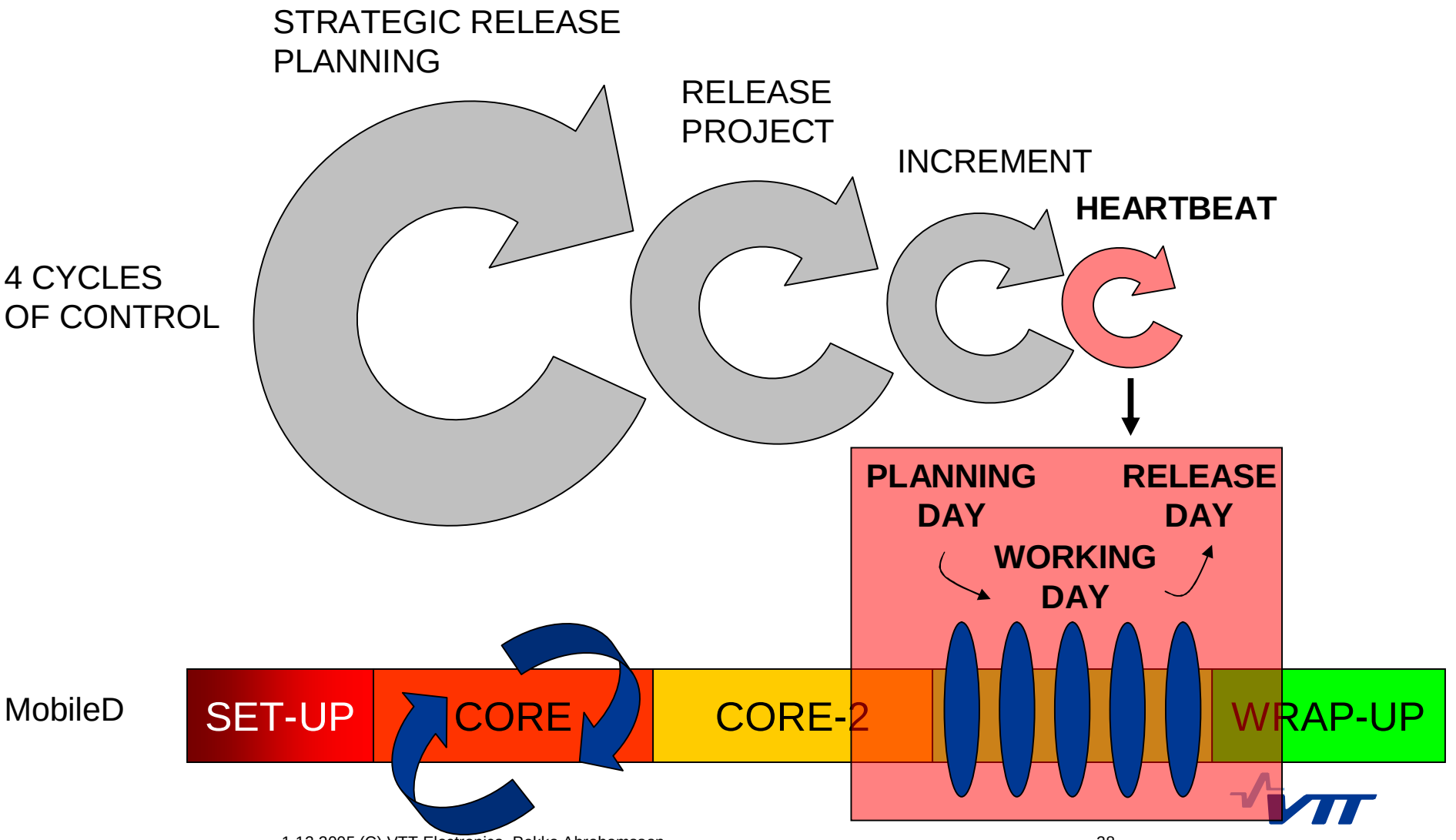


MobileD: FIT TO STRATEGIC PLANNING

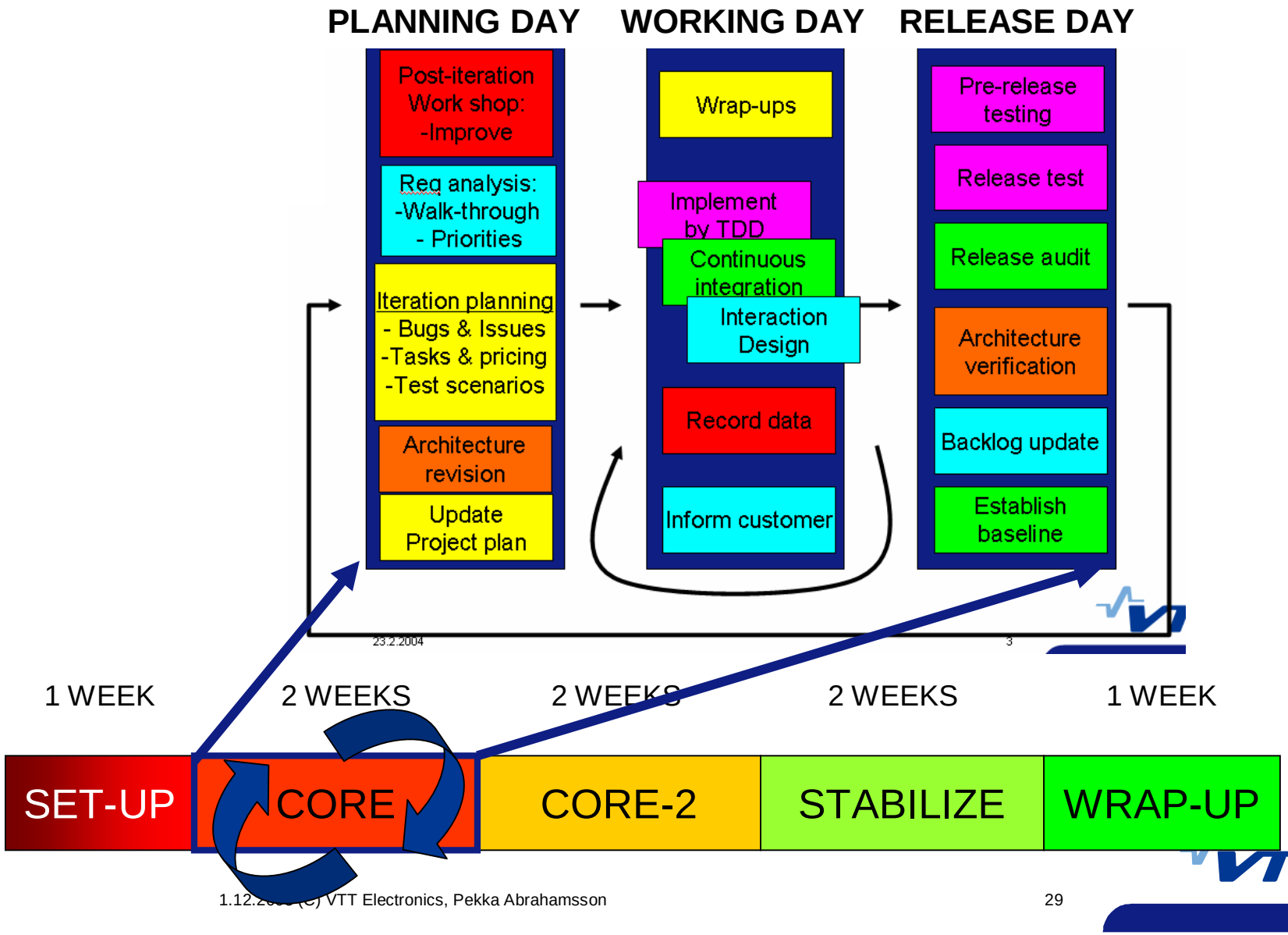




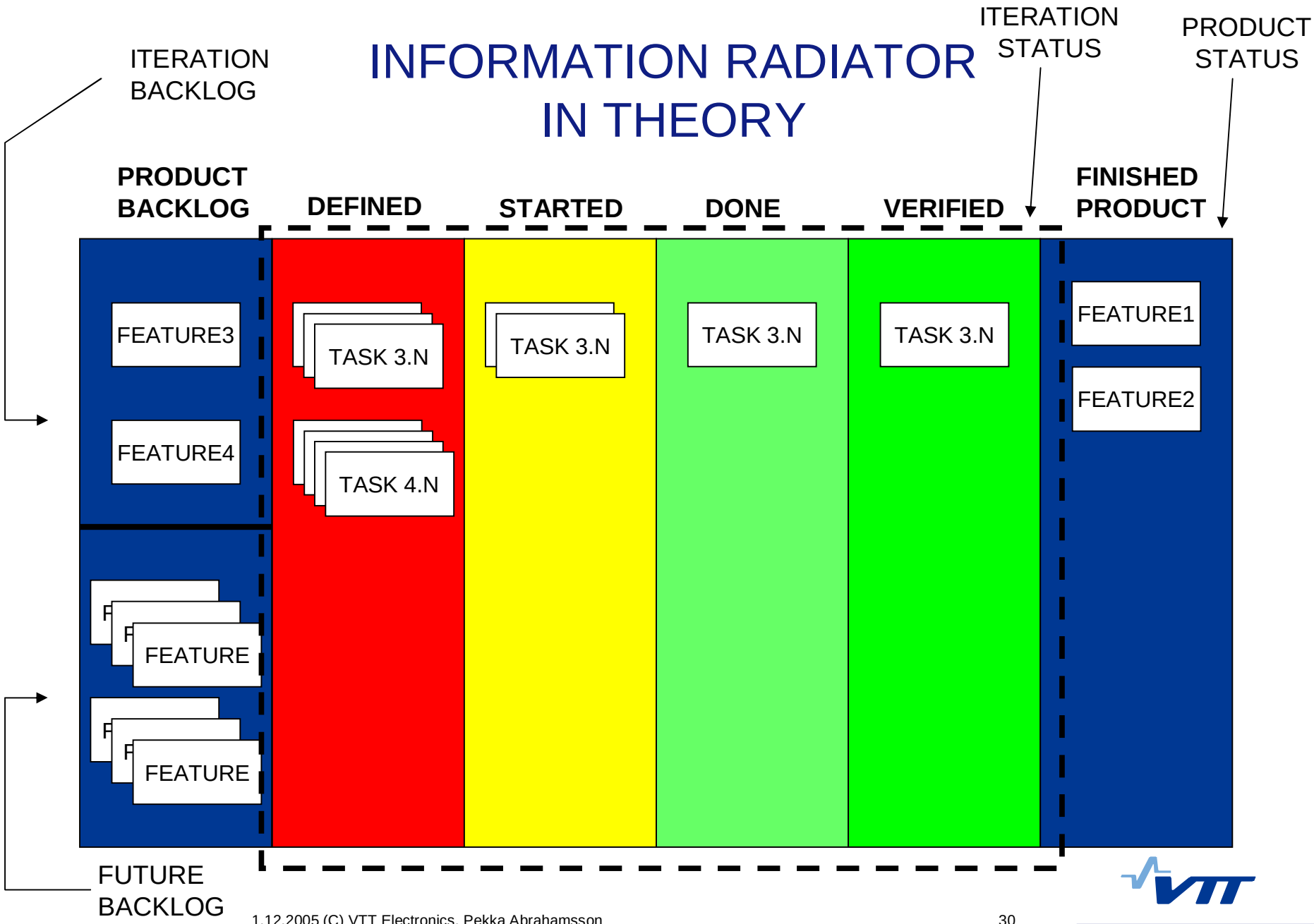
MobileD: FIT TO STRATEGIC PLANNING



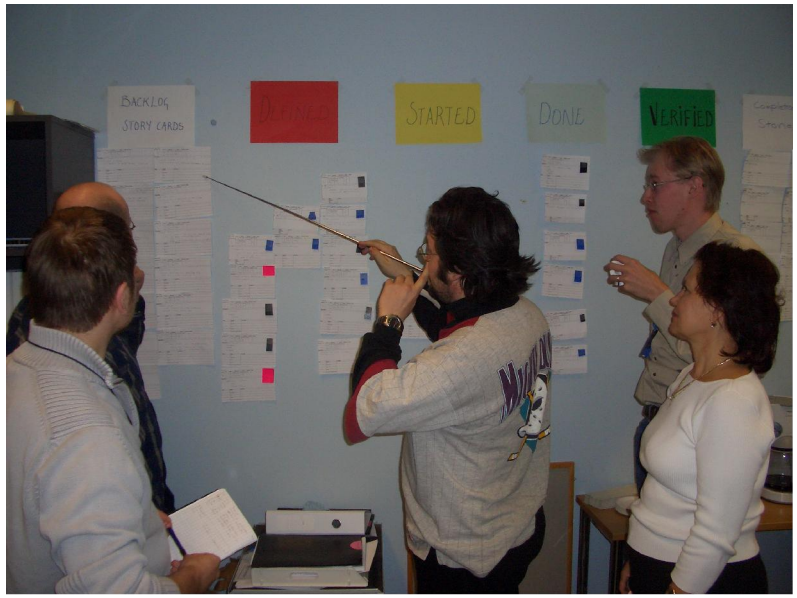
Mobile-D: The daily heartbeat



INFORMATION RADIATOR IN THEORY

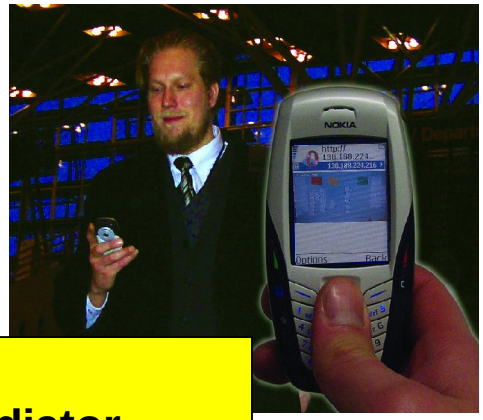


INFORMATION RADIATOR IN PRACTICE



Project team in a daily stand-up meeting

Customers and project managers working together



“Big boss” at the airport

Fact corner:
Information Radiator
Is one of the most spread agile practices

Part IV: Empirical cases

Some empirical evidence

Mobile service: Active investor

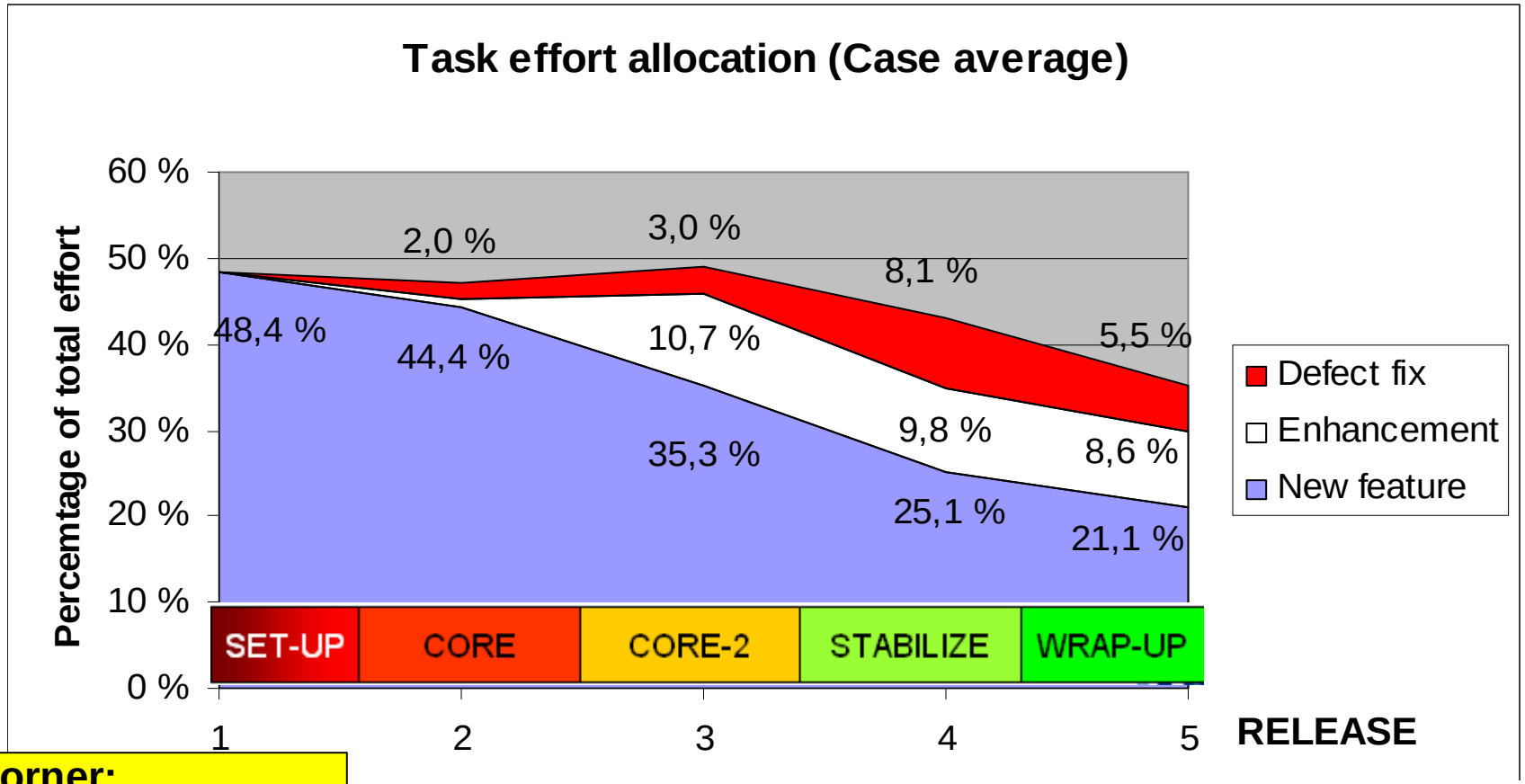
Mobile added value: Sales person

Mobile added value: Stockbroker

Fact corner:
9 projects completed
1 projects underway

3 CASE PROJECTS

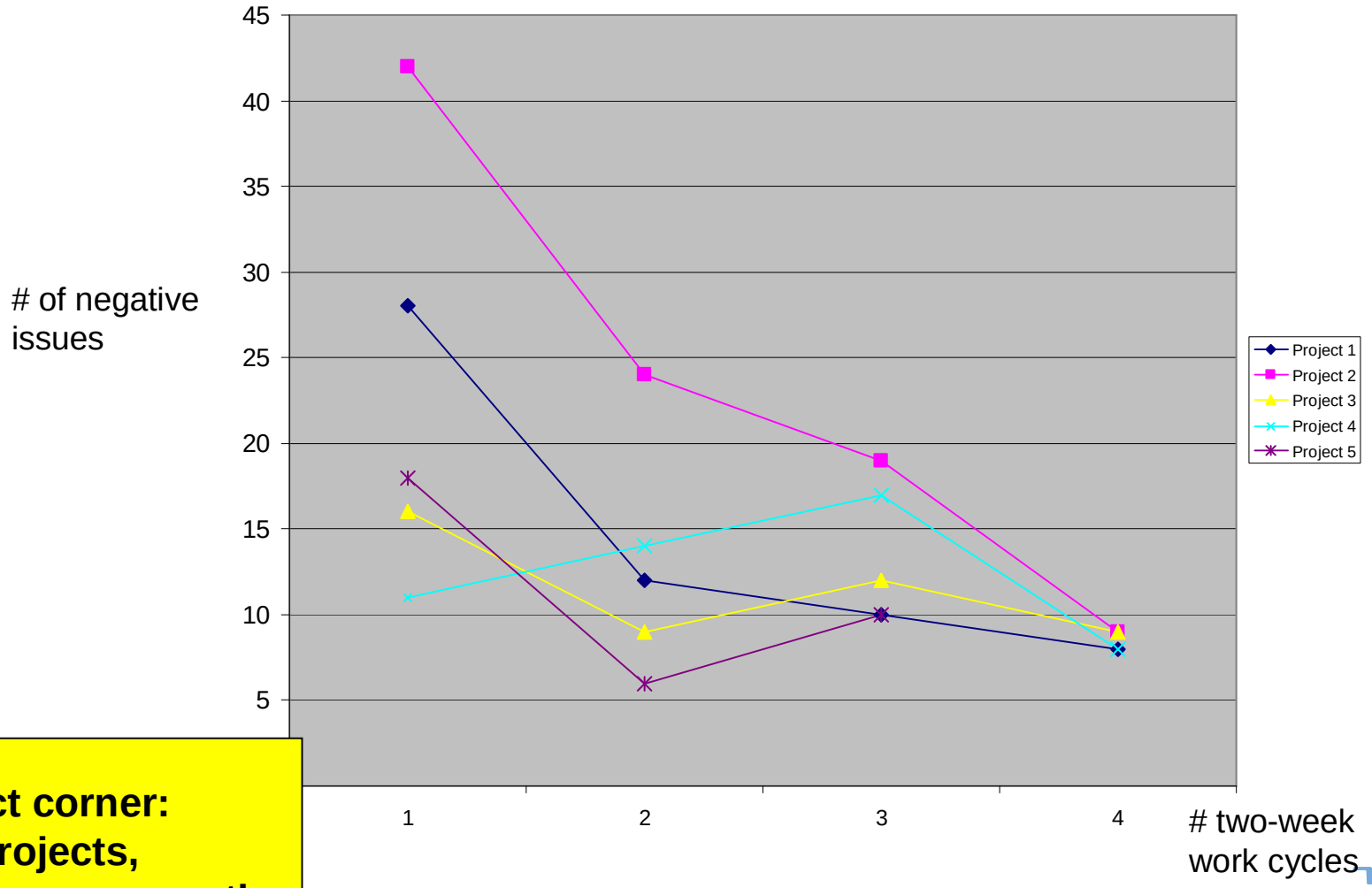
Velocity in terms of implementation (design+code+test)



Fact corner:
Explicit quantified process control, i.e. CMMI 5 level issues

What is **not** working in the process?

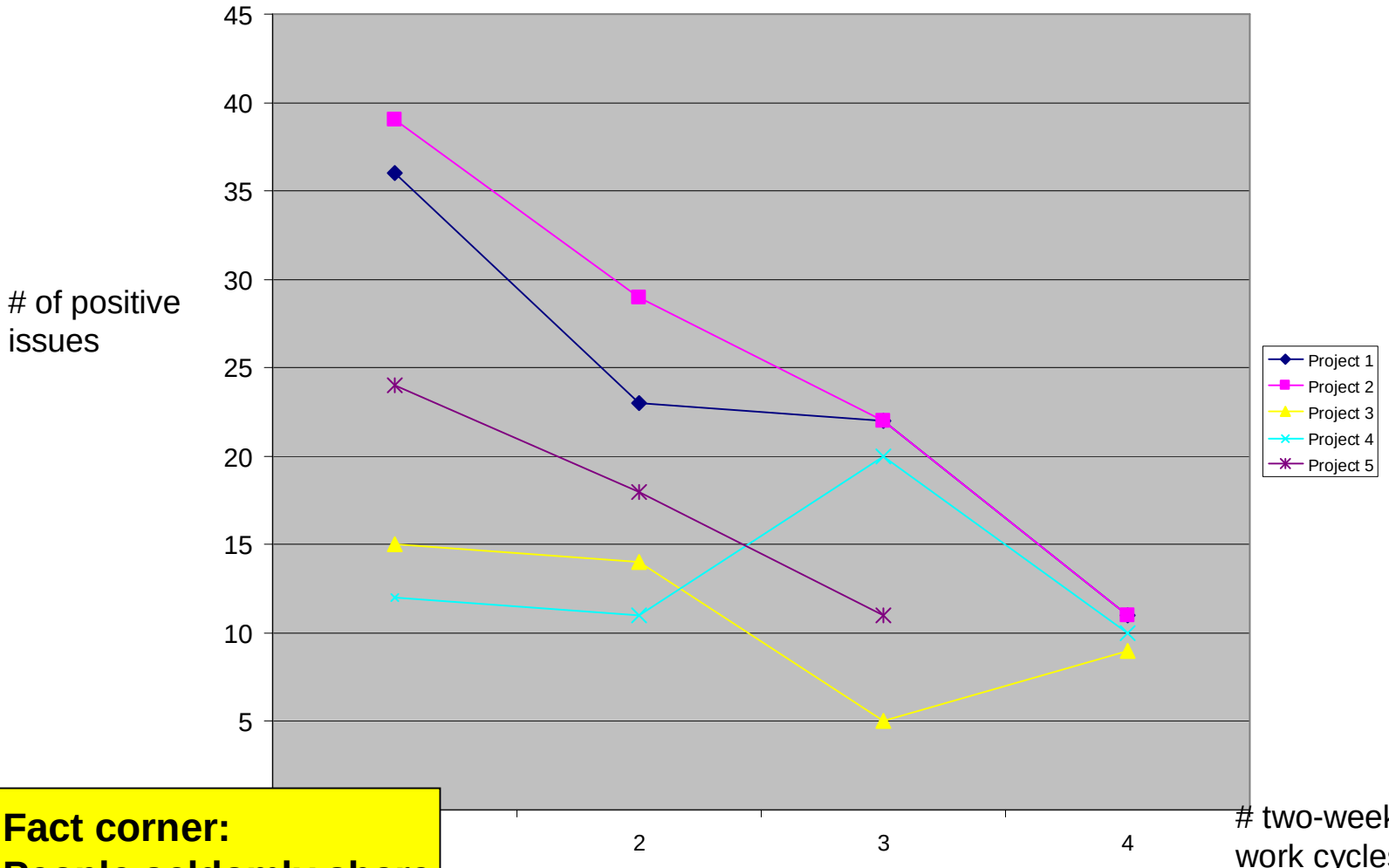
282 problems in the development process…



Fact corner:
5 projects,
50 person months

What is working in the process?

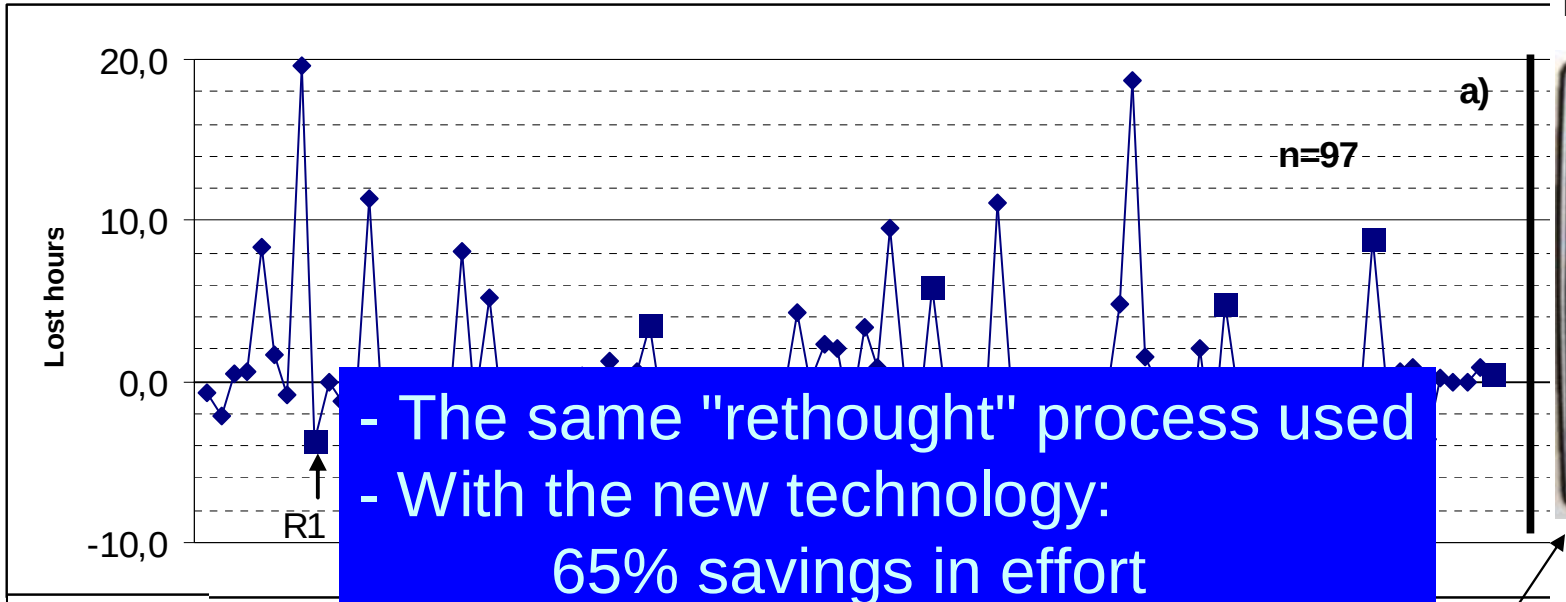
342 positive remarks in the process..



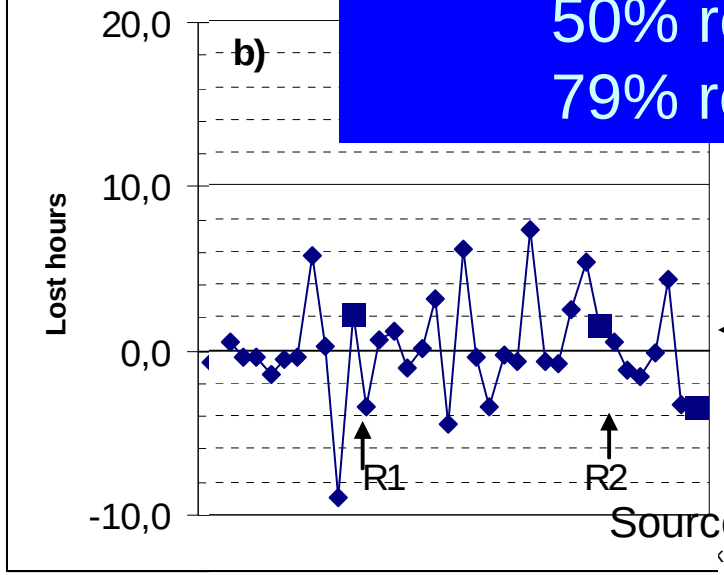
Fact corner:
People seldomly share what is going well!

BUT, BE AWARE OF THE NEW DEVELOPMENT TECHNOLOGIES

TRADITIONAL UI



- The same "rethought" process used
- With the new technology:
  - 65% savings in effort
  - 50% reduction in the lead-time
  - 79% reduction in lines-of-code



Naked Objects framework
= effort 380 hours, 4 weeks,
79% less app. code

NAKED OBJECTS

Source: Keränen & Abrahamsson, Euromicro 2005

Part IV: Future & conclusions

FUTURE

- Agile hype is getting to be over (good news)
  - XBreed, Freedom and other interesting agile methods will still keep emerging
- Yet, how to become more agile is the question now
  - Before it was more concentrated on extreme programming
  - Empirical evidence is quickly building up
- Agile becomes part of standardization work as well: Working group for IEEE 1648 (recommended practices)

THANK YOU!

Questions and comments?

Contact me at:
Pekka.Abrahamsson@vtt.fi
http://agile.vtt.fi

XP2006 will be organized in Oulu, Finland 17.6.-22.6.2006 featuring Kent Beck, Barry Boehm and others. Go check out www.xp2006.org