Agile Methods

Morgan Ericsson
<morgan.ericsson@chalmers.se>

My First Project

- A three year development project ...
- ... to create an online platform to support student mobility in Europe
  - manage programs, courses, students, grades, contracts, etc.
- I joined in 1996 (about 6-8 months into the project)

Vision

- Distributed system, where each university could either join a shared node or set up their own node
- Completely web based
- Should replace existing processes (papers) and interact with existing, local systems

Resources

- Budget of approximately 1M euro (not including in kind)
- 3 years development time
- Unclear how many persons
  - 30-50 persons in total
  - about 6 developers
- 4 teams in different parts of Europe

Result

- Huge waste of money
  - there was no final product ...
    - ... only a set of prototypes that demo'd various parts of the intended functionality
    - no actual value was produced
- Ruined and lost lives (yes, actually!)

Why?

- In my opinion, for main factors
  - unrealistic vision/goal
  - unsuitable organization
  - poor leadership
  - the wrong technology

Leadership and Organization

- The project was initiated with few, if any, developers available
  - senior architects, that focused on customers, requirements, architecture, etc.
  - several months spent making decisions that would last the entire project, with insufficient knowledge

Leadership and Organization

Leadership and Organization

- Project managers, not developers, received technical training on the underlying technology ...
- ... at luxurious locations, with fancy dinners
- but at least the developers got a (photo)copy of the material

Leadership and Organization

- Strict hierarchy
  - the different developer teams were not supposed to communicate directly
  - developers were not included in or consulted for major decisions ...
  - ... and were only told what to do

Technology

- The main application platform was unproven ...
  - not many proven platforms existed,
    - but the selected one was not even close to proven
- ... and unfamiliar
  - objects rather than relations
  - 4GL rather than traditional languages

Technology

```
class Person
       type tuple(  name: tuple ( last_name:   string,
                                  first_name:  string),
                    photo: Bitmap,
                    age: integer,
                    read spouse: Person,
                    read children: list (Person),
                    public dossier_no: real  )

   method   public is_adult: boolean,
            private add_child(child:Person),
            private salary_bracket:real
end;
```

```
select max(select c.age
           from c in p.children)
from p in Persons,
where p.name = "Paul"
```

Unrealistic Vision

- Creating and deploying the system would be a challenge today
  - even with mature and accepted web technologies, and
  - expert architects and developers

Unrealistic Vision

- Immature domain and technology
- 6 months requirements and design
- 30 months development
- bad, bad, bad idea

Bad Idea?

- During ’95-’98
  - HTML 2.0-4.0, CSS
  - IE 1, 2, and 3 released
  - Java and Javascript released
  - PHP, MySQL, Apache...

All Bad? No!

- Small team, one person to report to
- Fussy requirements and wrong design focus, so often no “real” direction
  - read: room to slack/play
- Technology that few understood
  - playground
- Great learning opportunity

Agile Manifesto

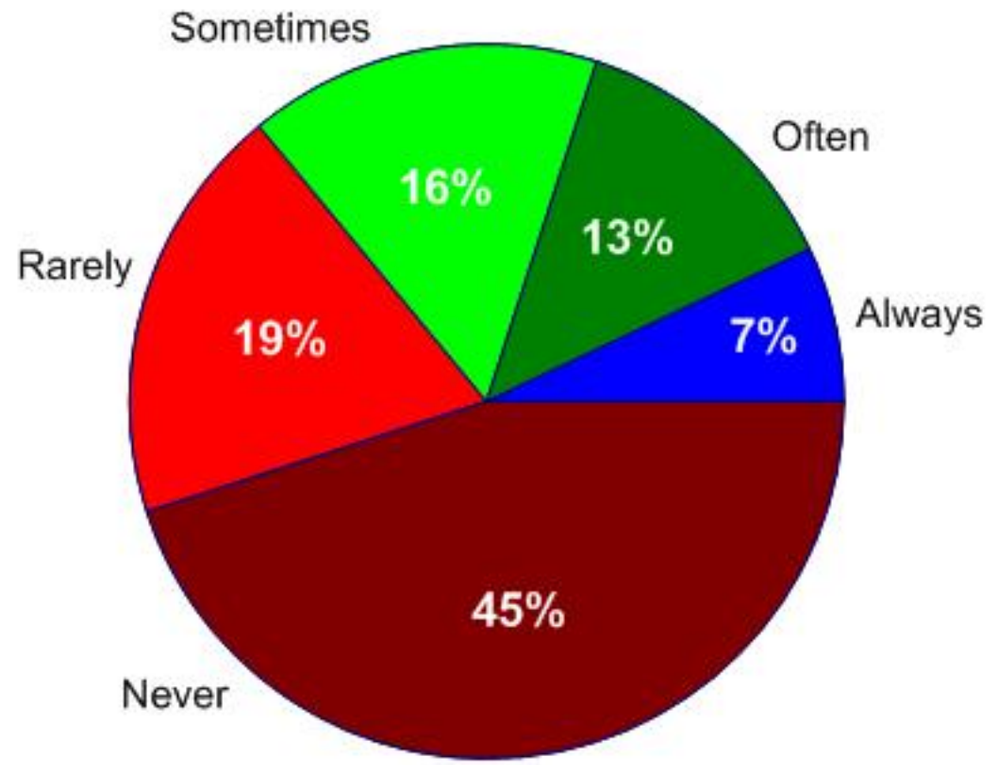
- Individuals and interactions over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan

Principles behind the Agile Manifesto

- Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage
- Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.

Functionality

Average percentage of delivered functionality actually used when a serial approach to requirements elicitation and documentation is taken on a "successful" information technology project.

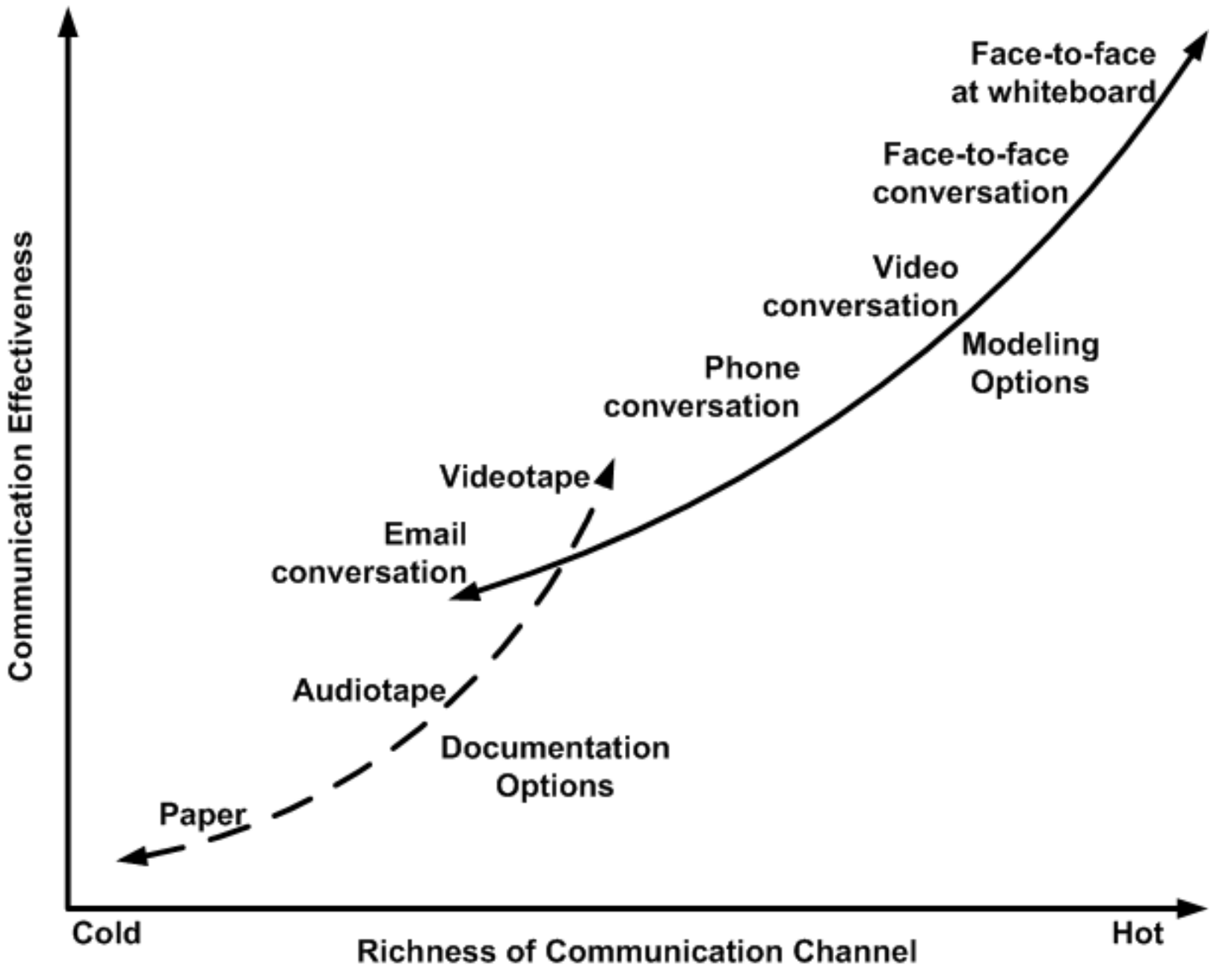


Source: Chaos Report v3, Standish Group.

Principles behind the Agile Manifesto

- Business people and developers must work together daily throughout the project.
- The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
- Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.

Communication Efficiency



Principles behind the Agile Manifesto

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

If I had to Start Over?

- Embrace what worked, no need to change for the sake of change
  - small teams
  - pair programming (came out of necessity, workstations ~ 10k euro each)
  - basically technical parts of XP (very close to how we approached assignments)

If I had to Start Over?

- Improve the things that did not work
  - "user education"
  - design and architecture
  - trust
  - early decisions, uninformed choices

User Education

- About 70,000 domains in ’95 and approximately 10-15m Internet users
  - most intended users probably never used a web browser
  - so how could they know what they wanted the system to do?
  - existing paper-based system was probably translated without too much thinking...

Design and Architecture

- The system was both over and under designed
  - vision with “epic” scale
  - design not particularly epic (e.g., UI)
  - problem with architects and understanding...

Trust

- Difficult issue, different levels
  - massive trust within team and at local site
  - yet, not enough trust to contribute at design decisions
  - reason communicated via managers, often “Chinese whispers / Telephone”

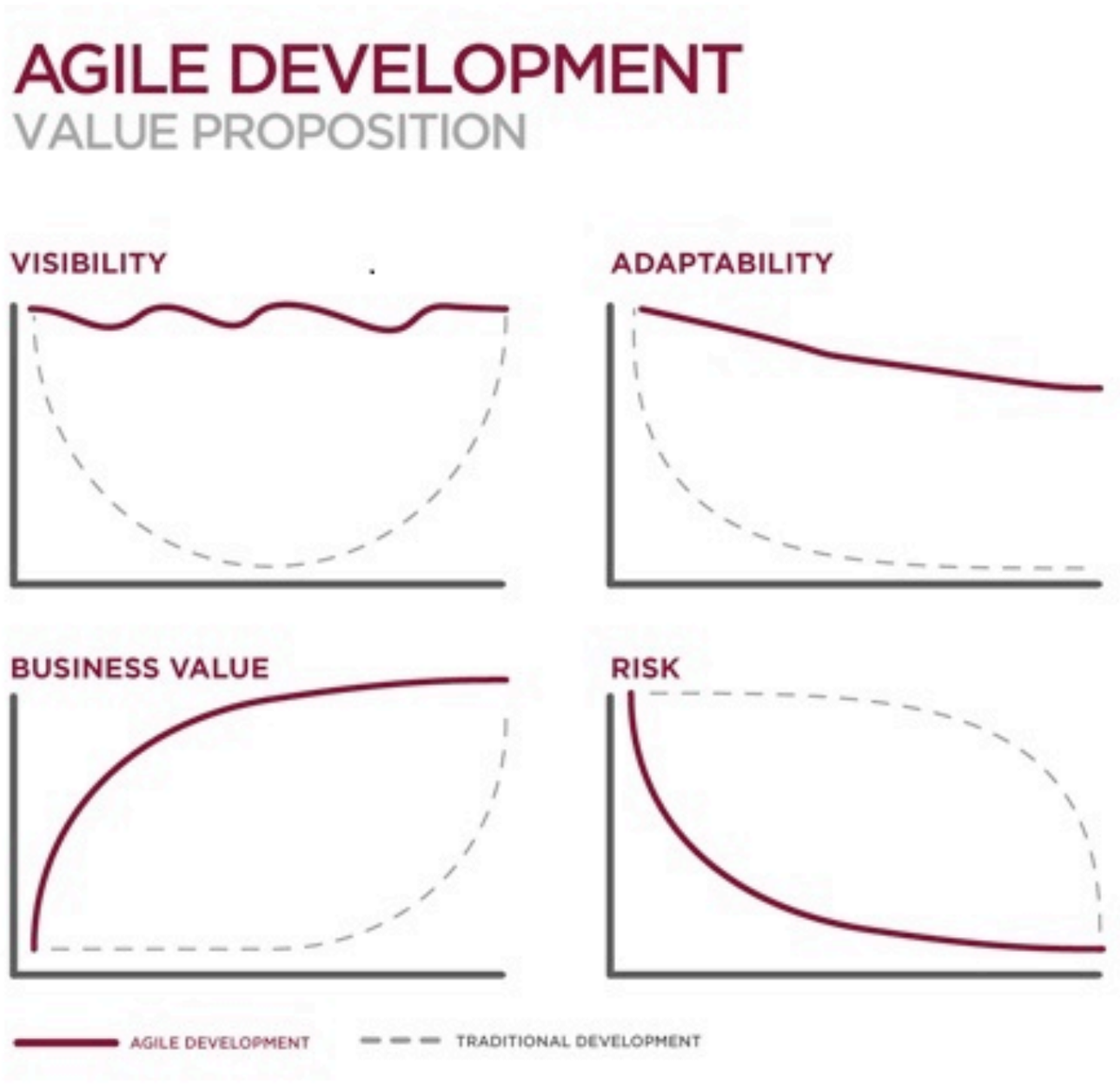
Early Decisions

- Early, uninformed choices most likely would have "doomed" the system
  - platform provider died '97-'99 and there was considerable lock-in
  - single platform could not evolve with technology
  - even if it was a brilliant choice on paper
- "Nobody" knew how to build it (in practice)

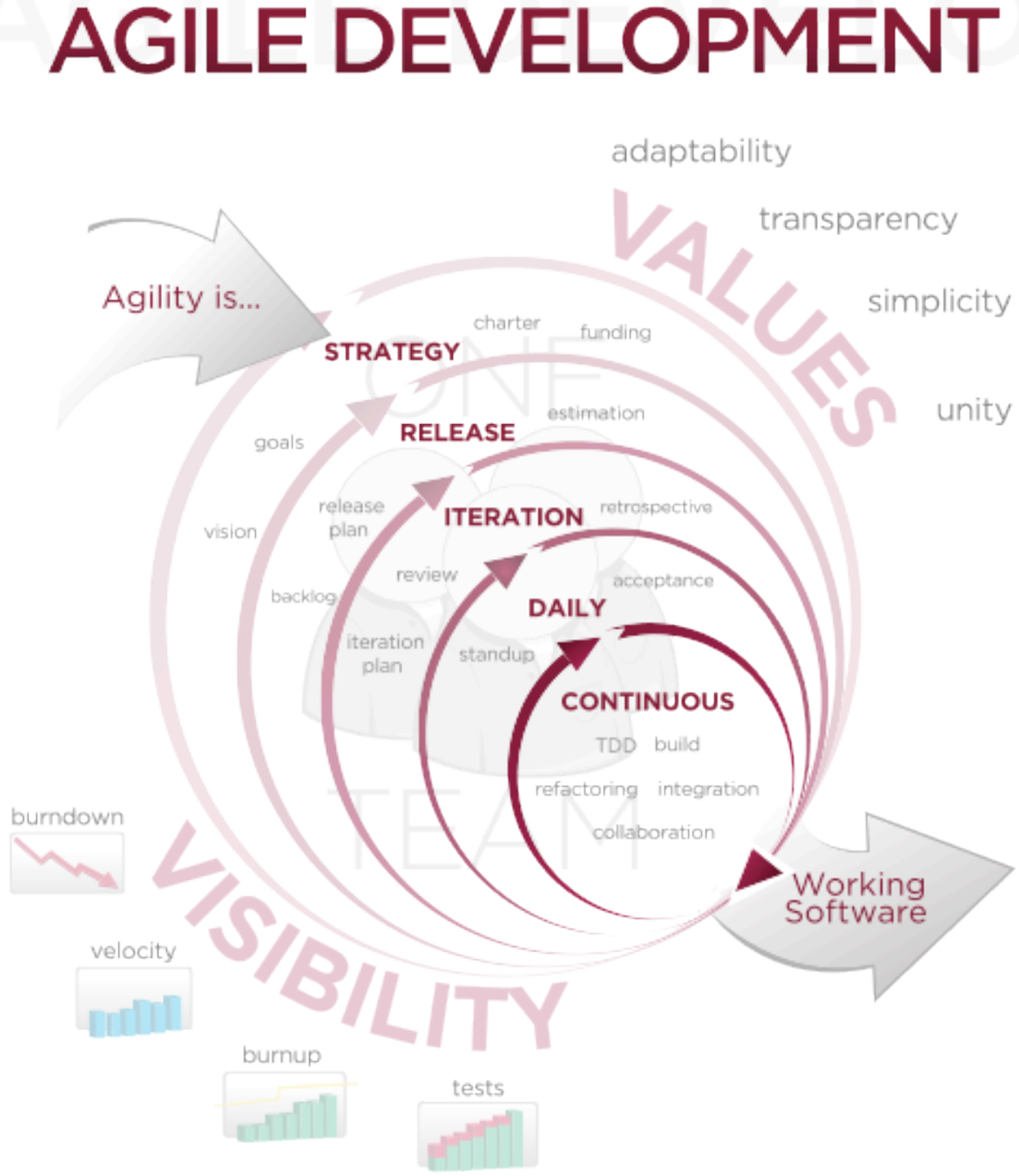
Agile?

- Would an Agile approach solved many of the problems we faced?
- it would at least have exposed many of them at an early stage...

Agile?


AGILE DEVELOPMENT
VALUE PROPOSITION
VISIBILITY
ADAPTABILITY
BUSINESS VALUE
RISK
AGILE DEVELOPMENT
TRADITIONAL DEVELOPMENT

Agile?



ACCELERATE DELIVERY

Beliefs

- XP is good, adopting what we did not do would have provided a lot
  - many questions regarding initial RE, choice of architecture, refactoring
  - refactoring is good, "how do you eat an elephant?"
  - running code exposes problems...

Beliefs

- XP needs management, but one size does not fit all
  - we basically did Kanban within the team,
  - which was all we needed
    - flow, feedback, and not in our faces
  - but others may prefer Scrum (or whatever...)

Beliefs

- Add, don't remove
  - this is not (R)UP,
  - so add support functions you need, modify practices,
  - but swap rather than remove
  - (unless you know what you are doing)

Beliefs

- When things scale,
  - consider structure in multiple dimensions
  - that could be flexible over time

Current Practice

- Remember, mainly research-orientation
  - but still 1+ KLOC per week / person
- XP with some small-group changes
- Release (delivery) orientation
  - but complex organization
  - and lacking / difficult management
- Kanban would probably be a good inspiration
- (but remember, only project manager in academia)