Be Smart!

or

What they don’t teach you about software at school

Ivar Jacobson

with

Ian Spence, Pan Wei Ng and Kurt Bittner

ivar@ivarjacobson.com

www.ivarblog.com

IVAR JACOBSON
INTERNATIONAL

Your goal is the same as always!

Good Software, Quickly and at Low Cost!

What it takes

Quickly

Competent & Motivated People

Low Cost

Large Scale Reuse of Components

Good Software

Useful Extensible Reliable

What it takes

What they don’t teach you about software at school

☺

One major obstacle…we are a fashion industry

Software Development is driven by fashions and fads

- Fifteen years ago it was all about OO
- Ten years ago it was about components, UML, Unified Process
- Five years ago it was about RUP and CMMI
- Two years ago it was about XP
- Today it is about Scrum

All good, but none is all you need

The software industry keeps looking for a silver bullet

One major obstacle…we are a fashion industry

Yes, Software Development is driven by fashions and fads

- Fifteen years ago it was all about OO
- Ten years ago it was about components, UML, Unified Process
- Five years ago it was about RUP a CMMI
- Two years ago it was
- Today it is abo

All good, bu ed

tware industry keeps looking for silver bullets

There is no silver bullet!

This is unsmart!

One major obstacle…we are a fashion industry

Yes, Software Development is driven by fashions and fads

- Fifteen years ago it was all about OO
- Ten years ago it was about components, UML, Unified Process
- Five years ago it was about RUP a CMMI
- Two years ago it was
- Today it is abo

All good, b ed

tware industry keeps looking for silver bullets

We have to work smarter

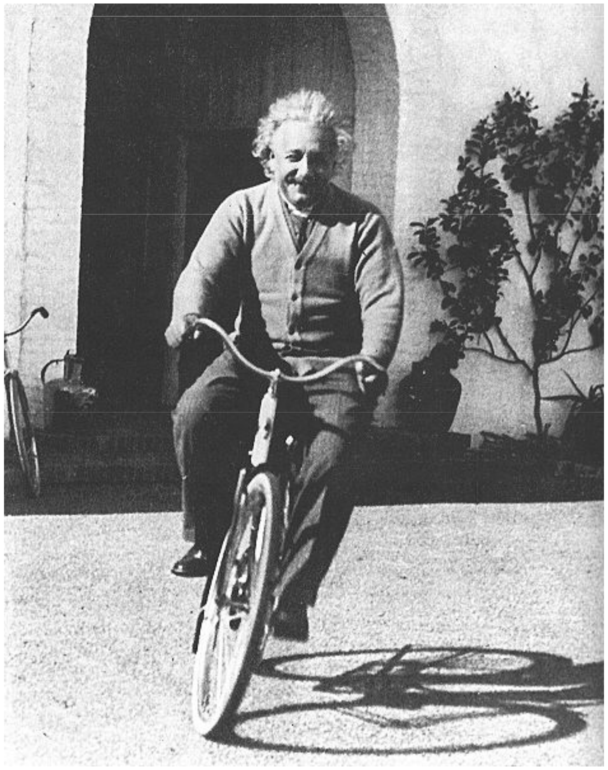
This is unsmart!

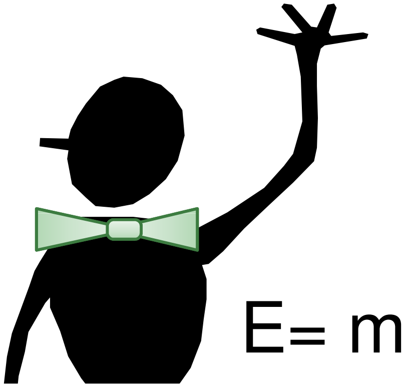
Agenda

1. What does Smart mean?
2. Smart Cases – Recognize it when you see it
3. How do you become Smart
4. What does Smart really mean?

What does *Smart* mean?

Things should be done
as simple as possible – but no simpler
- *Albert Einstein*



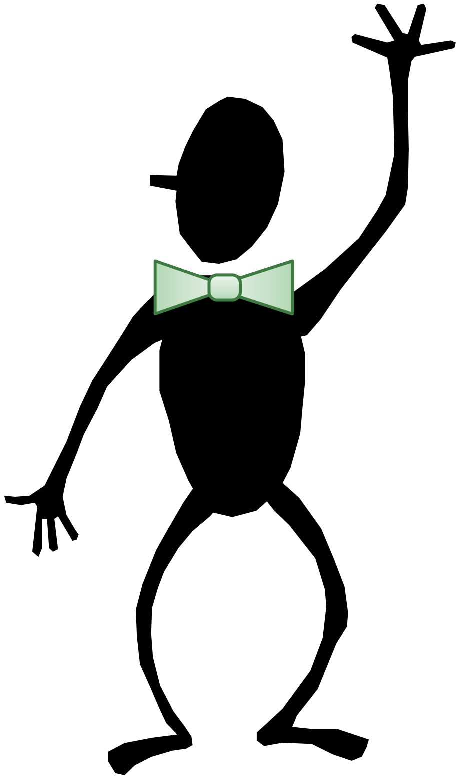


$E= mc^2$

This is smart!

Smart and Intelligent?

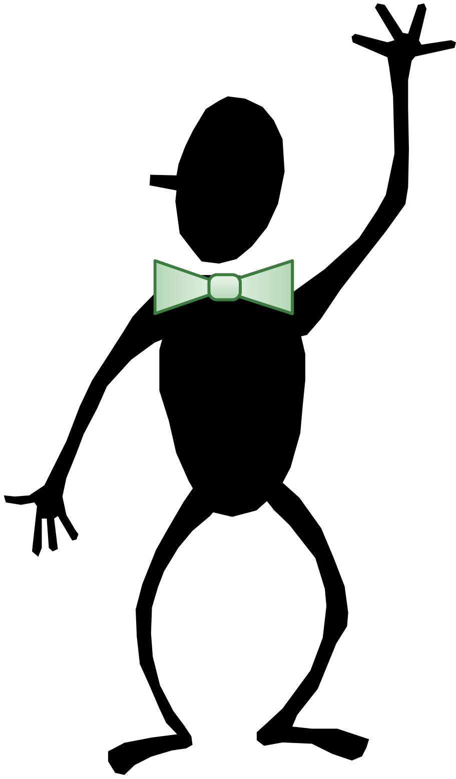
- Being Smart is not the same thing as being intelligent
  - You can be intelligent without being smart, and
  - You can be very smart without being very intelligent



Mr Smart

Smart and Agile?

- Being Smart is an *evolution* of Being Agile
  - Agile means being flexible and adaptable.
  - Agile provide simple/lightweight starting points
  - But being smart is knowing when to go beyond agile
    - Knowing when to follow the rules and when to break them
    - Knowing when to be consistent and when to change
    - Knowing when to grow and when to shrink



Mr Smart

Smart = Agile ++

Agenda

1. What does Smart mean?
2. Smart Cases – Recognize it when you see it
    1. People
    2. Teams
    3. Projects
    4. Requirements
    5. Architecture
    6. Modeling
    7. Test
    8. Documentation
    9. Process
    10. Knowledge
    11. Outsourcing
    12. Tools
3. How do you become Smart
4. What does Smart really mean?

What they don’t teach you about software at school

☺

www.ivarblog.com

Not smart with People

Some companies view process and tools as more important than people

This is unsmart!

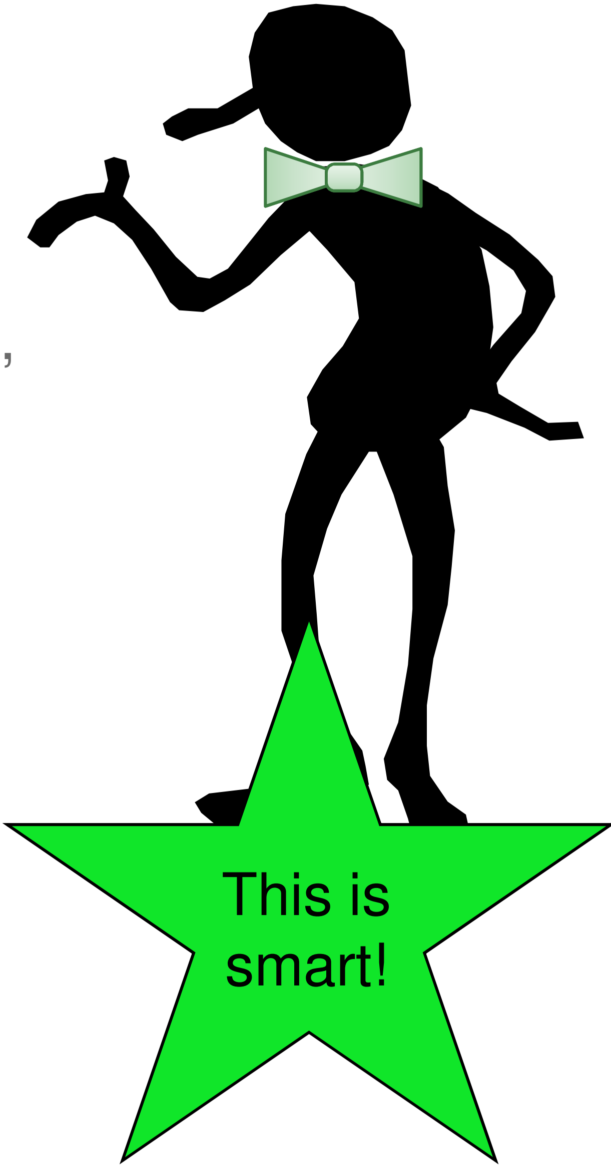
A fool with a tool is still a fool but a **dangerous** fool

Smart with People

Case study: Ericsson AXE – the largest commercial success story ever in Sweden

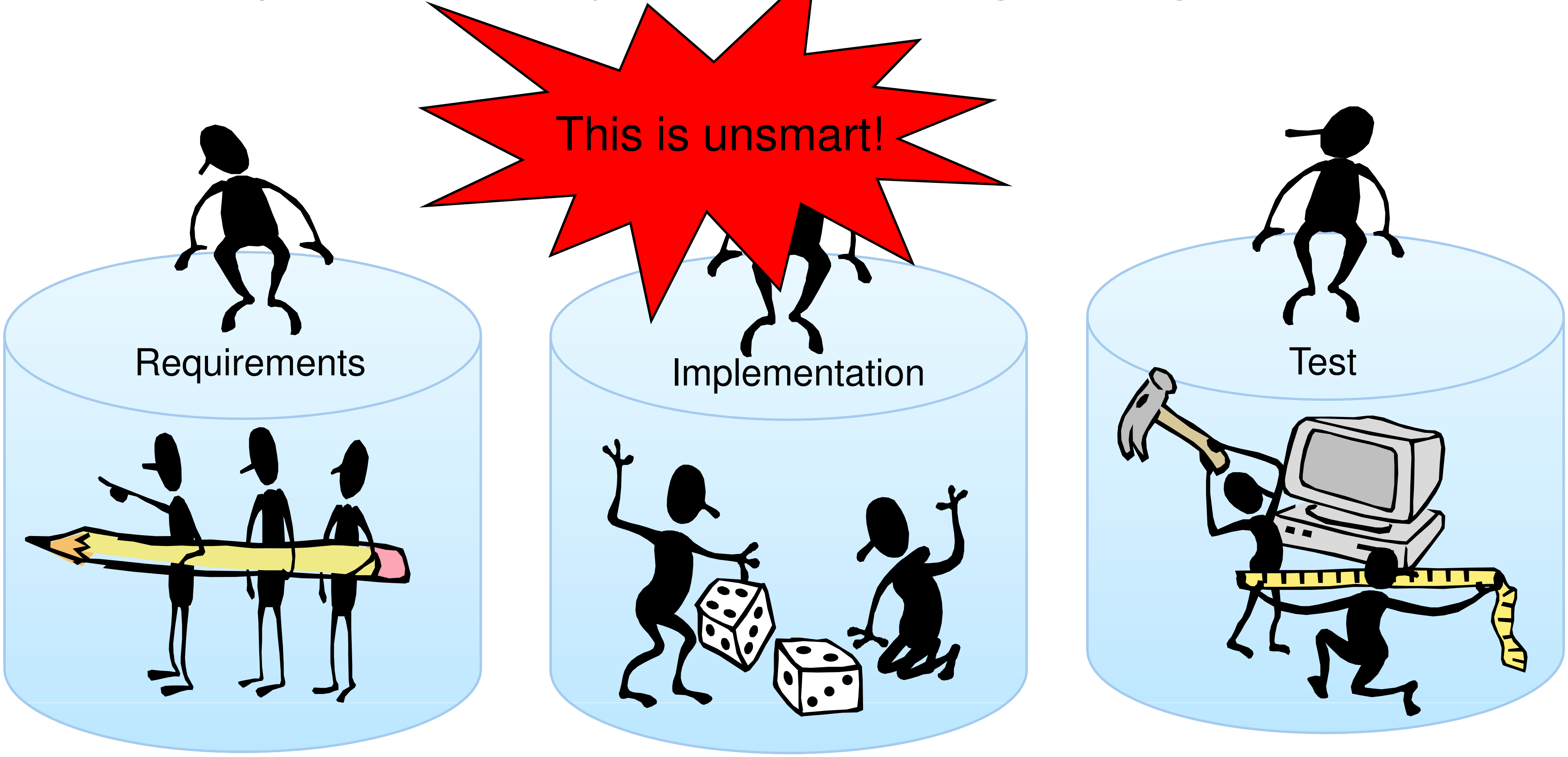
- –We had no tools and no defined process
- –Despite this, we developed components, use cases, and a modeling language now part of UML
- –This could only have been done with people – good people

Software is developed by people, not by process and tools.



Not smart with Teams

- Many software projects involve 20+ people
- Often organized into stove-pipe groups:
  - Requirements, Analysis, Design, Coding, Testing, etc.

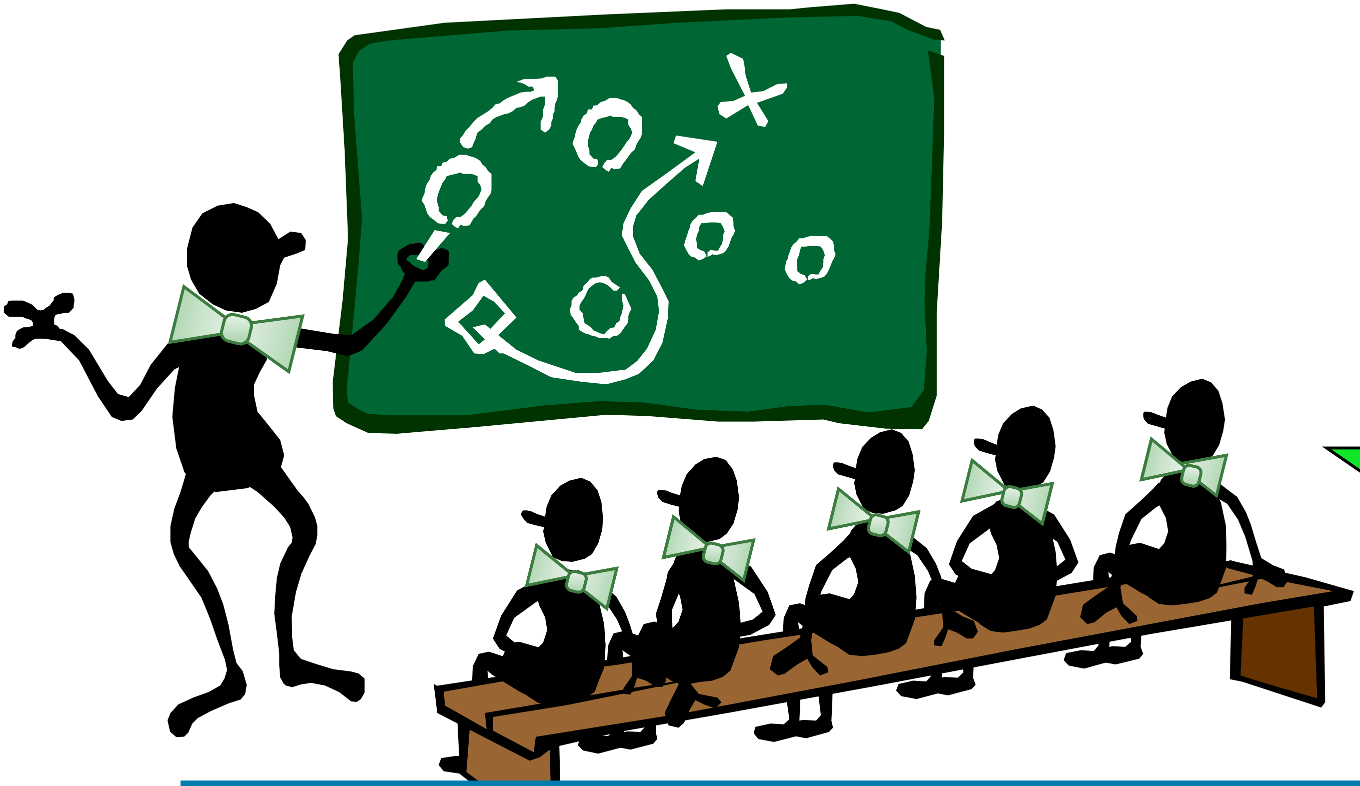


Smart with Teams

- Teams are cross-functional

  Including analysts, developers, testers etc…

- Ideal size of the team is less than 10 people

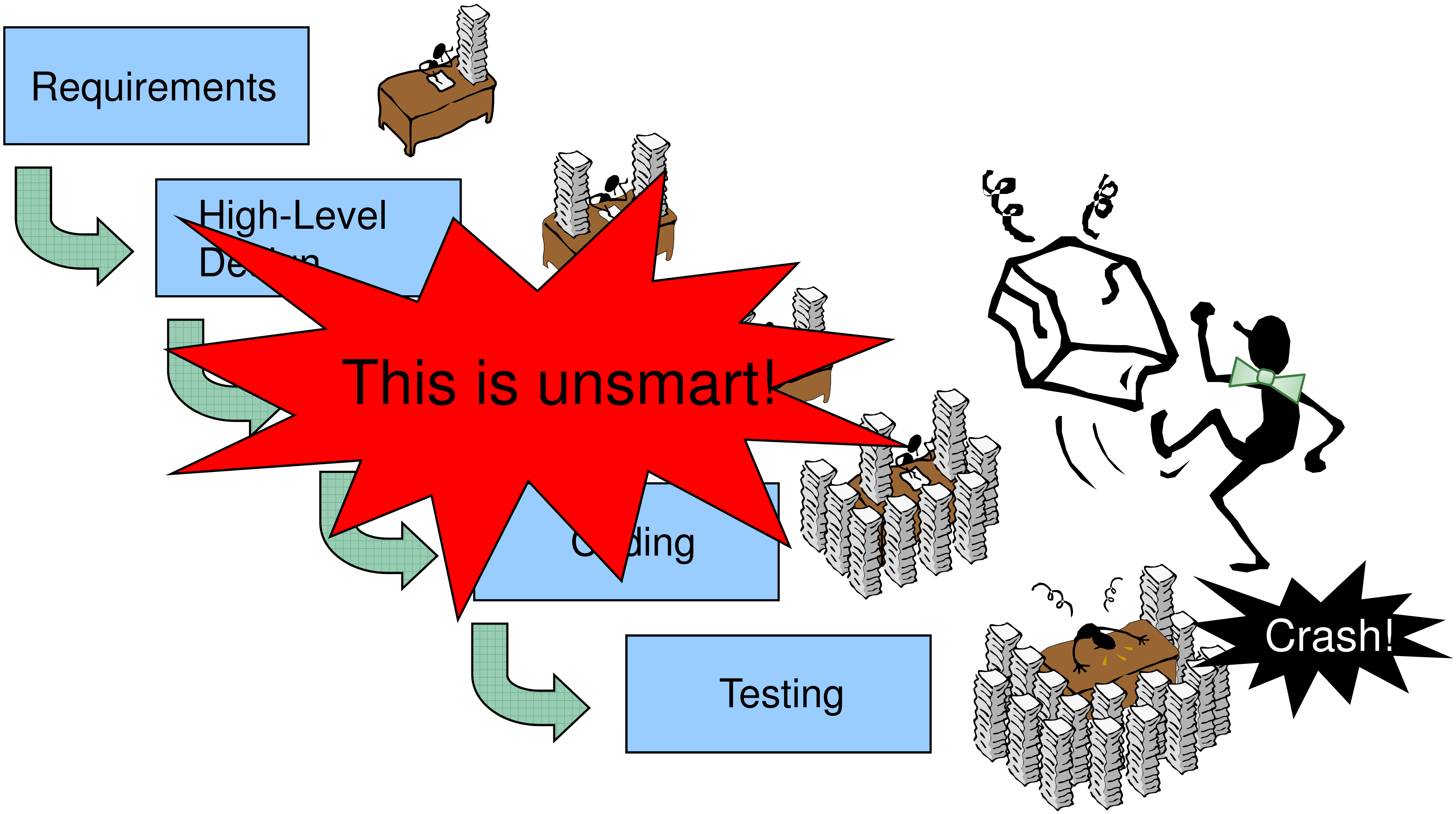


This is smart!

A software team is like a sport team with all needed competencies to win.

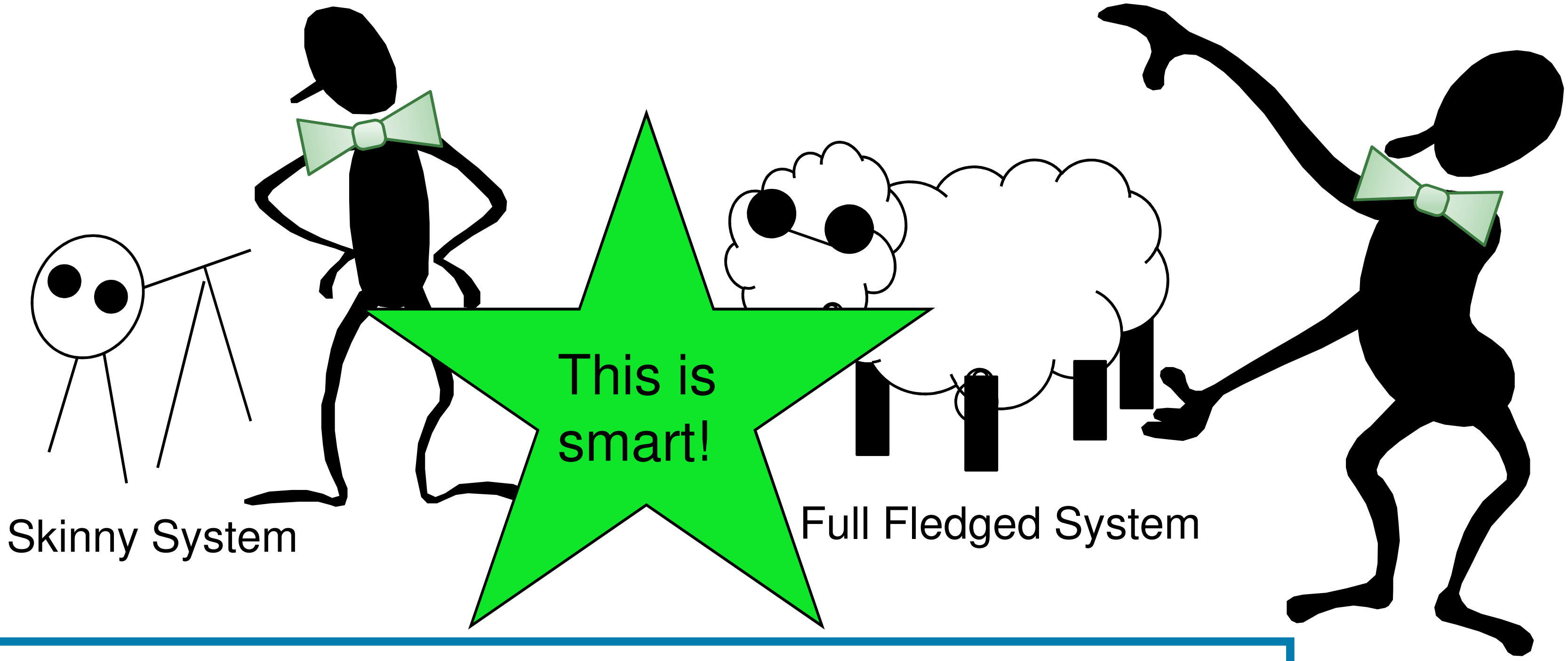
Not smart with Projects

- Most companies still follow the waterfall approach



Smart with Projects

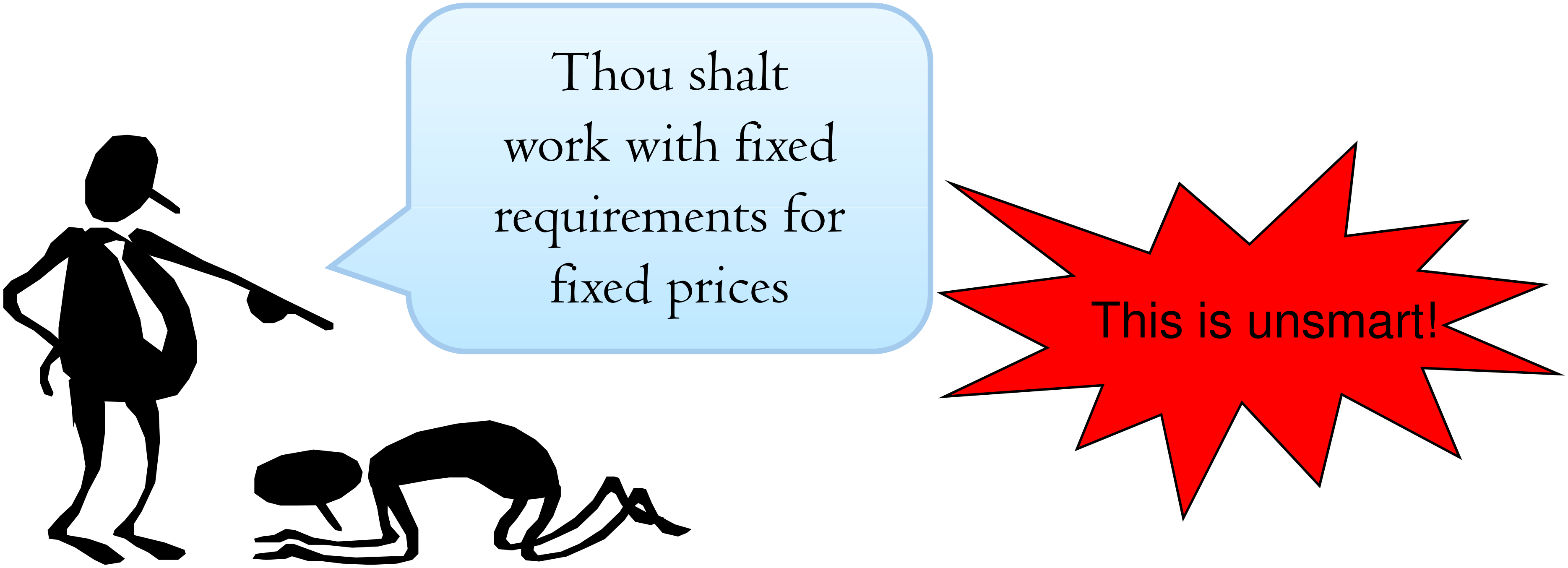
- Build a skinny system to demonstrate that you have eliminated all critical risks
- Add more capabilities on top of that skinny system



Think big, build in many steps

Not smart with Requirements

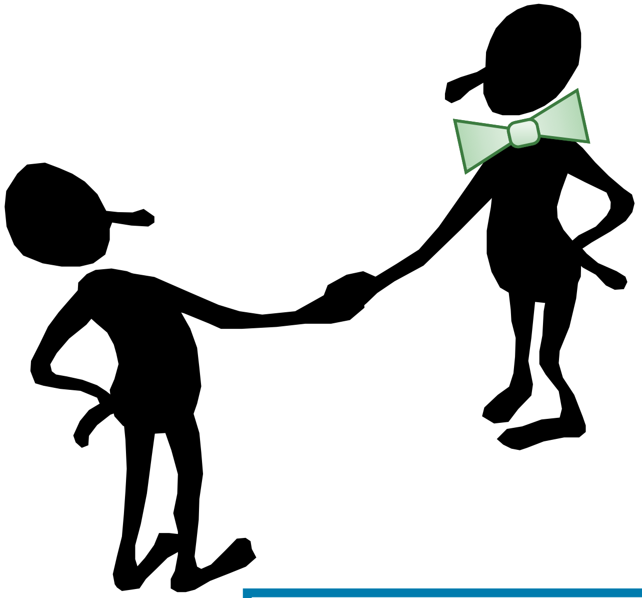
- Many managers (and customers) believe you can detail all the requirements upfront...
- ...and based on these can accurately predict the cost of the solution



A constant in software development is that requirements **always** change

Smart with Requirements

- Base early decisions on lightweight requirements and detail as and when it is needed
  - Use case outlines, feature lists or user stories
- Remember requirements are negotiable and priorities will change



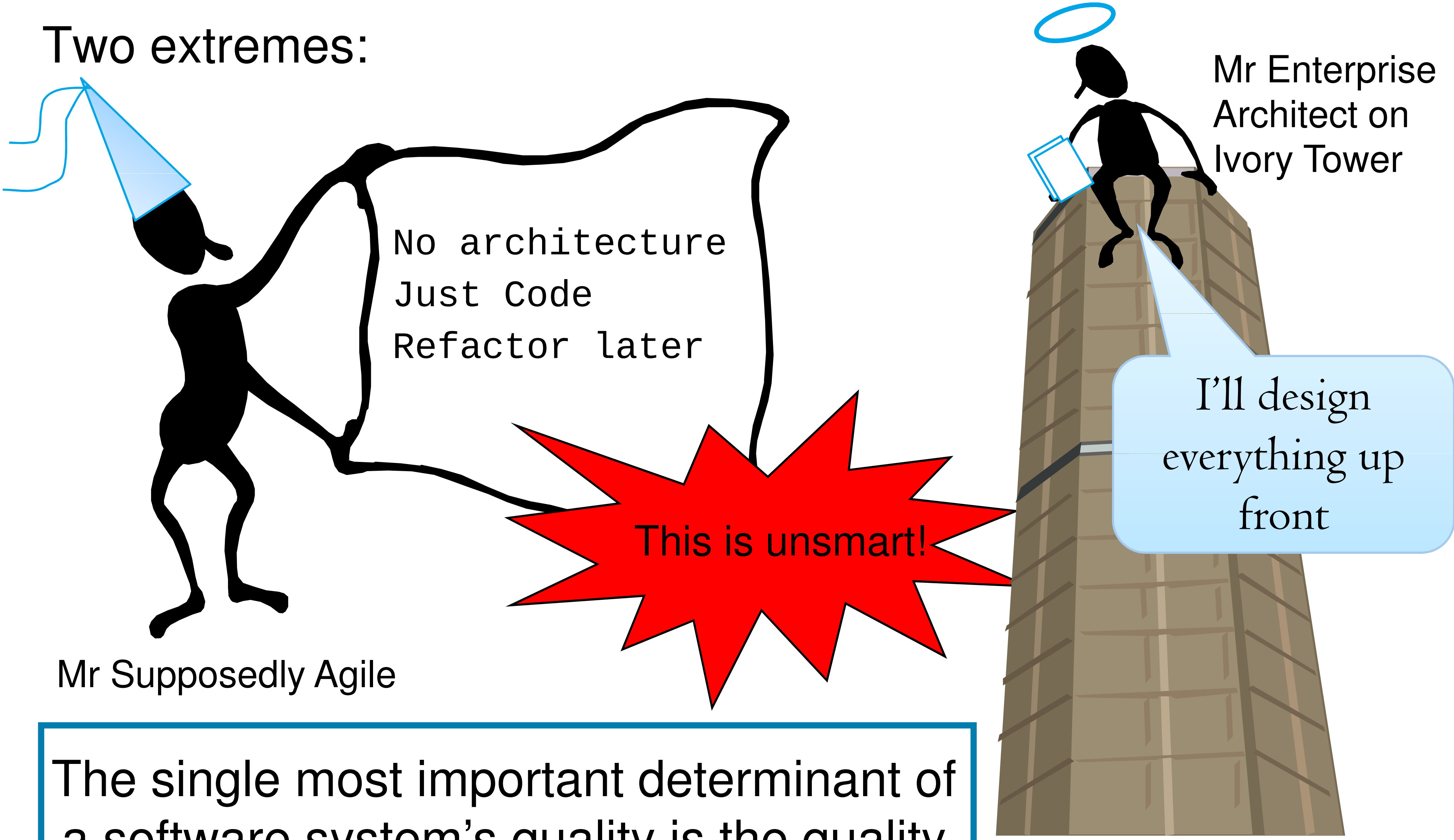
I understand your needs, let's work together to make sure we develop the right system for the right price.

This is smart!

Design your project for requirement changes

Not smart with Architecture

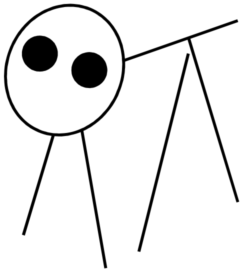
Two extremes:



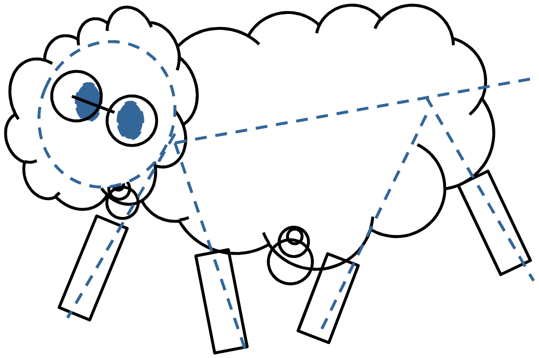
The single most important determinant of a software system's quality is the quality of its architecture

Smart with Architecture

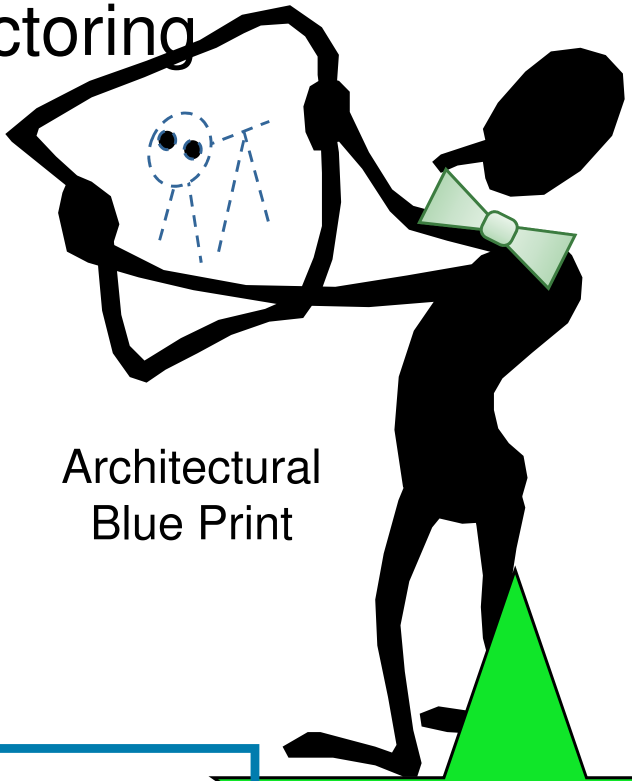
- Focus on the skinny system
- But an architecture without executable code is a hallucination
- Refactor over releases, but large refactoring is very costly



Skinny System



Full Fledged System



Architectural Blue Print

Start to build a skinny system, add muscles in later steps

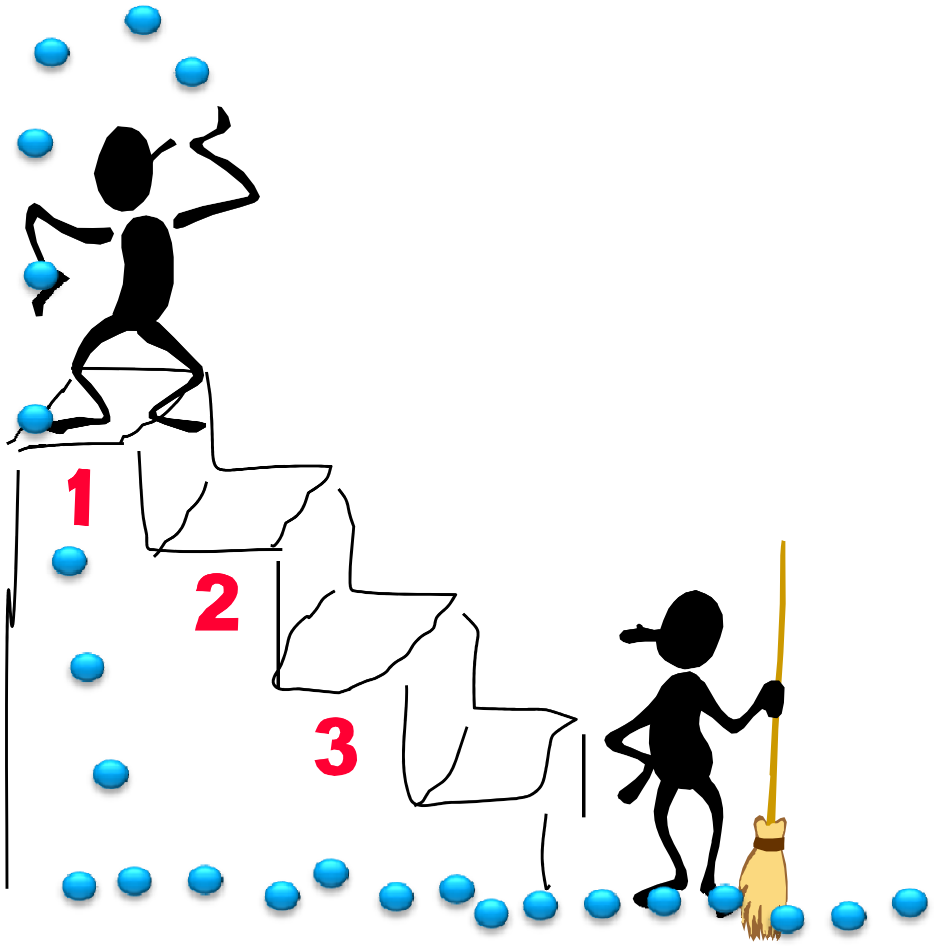
This is smart!

Not smart with Test

We have two classes of people: Developers and Testers

- *Developers are the creators…it is OK to create bugs as well*
- *Testers are the cleaners in the software world*

Testing is done as an after thought – too late and too expensive



This is unsmart!

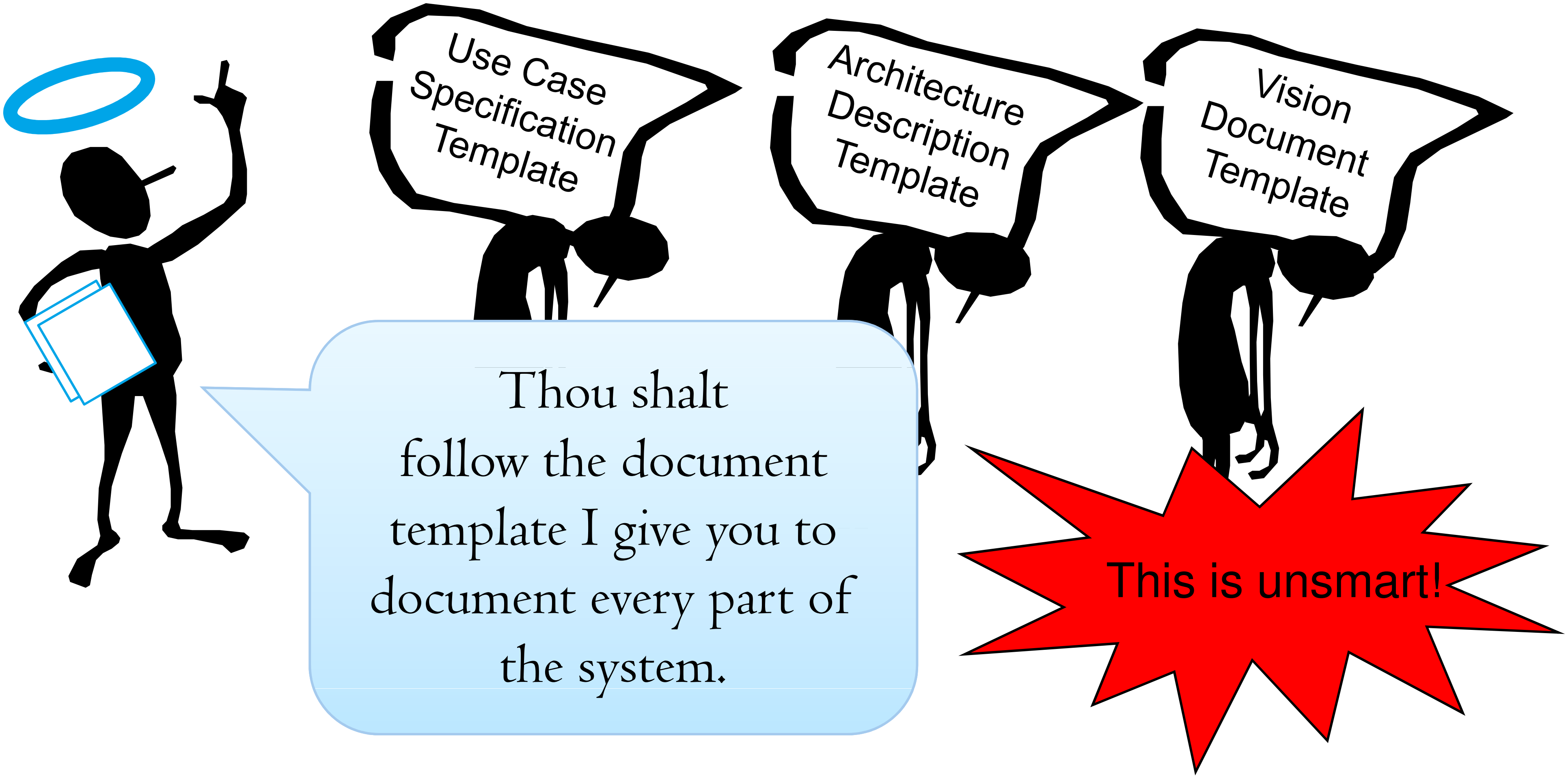
Smart with Test

This is smart!

Whatever you do **you** are not done
until **you** have verified
that **you** did what **you** wanted to do

Not smart with Documentation

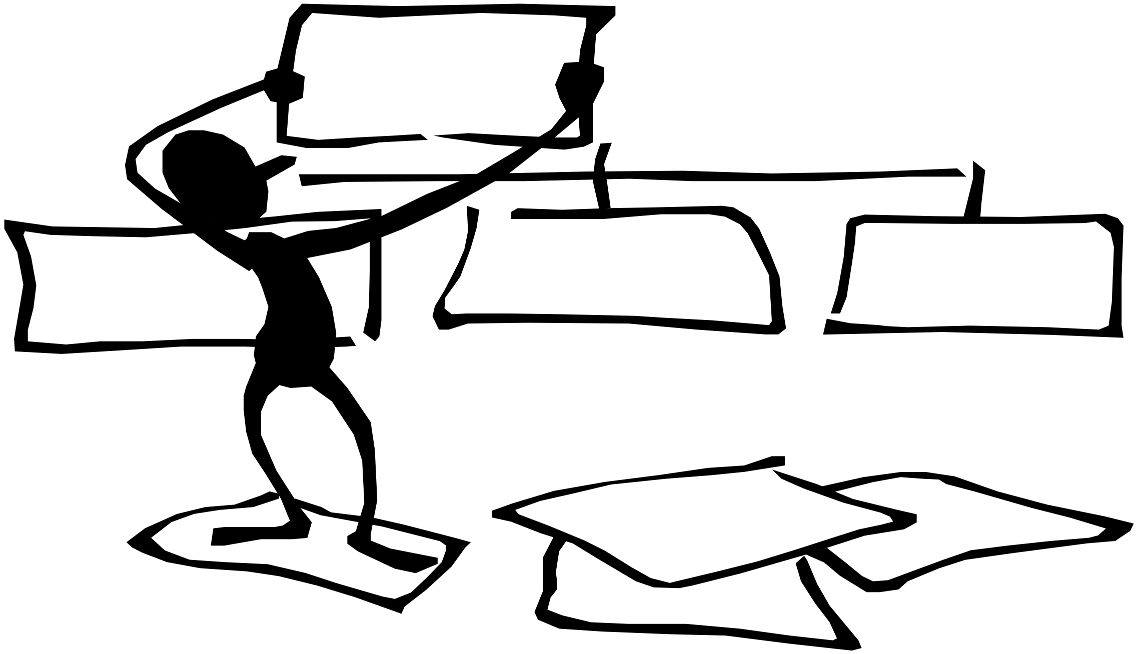
- There has been an over-emphasis on teams producing documentation


Use Case Specification Template
Architecture Description Template
Vision Document Template
Thou shalt follow the document template I give you to document every part of the system.
This is unsmart!

Smart with Documentation

Myth: The idea that you document software so people later can read what you did.

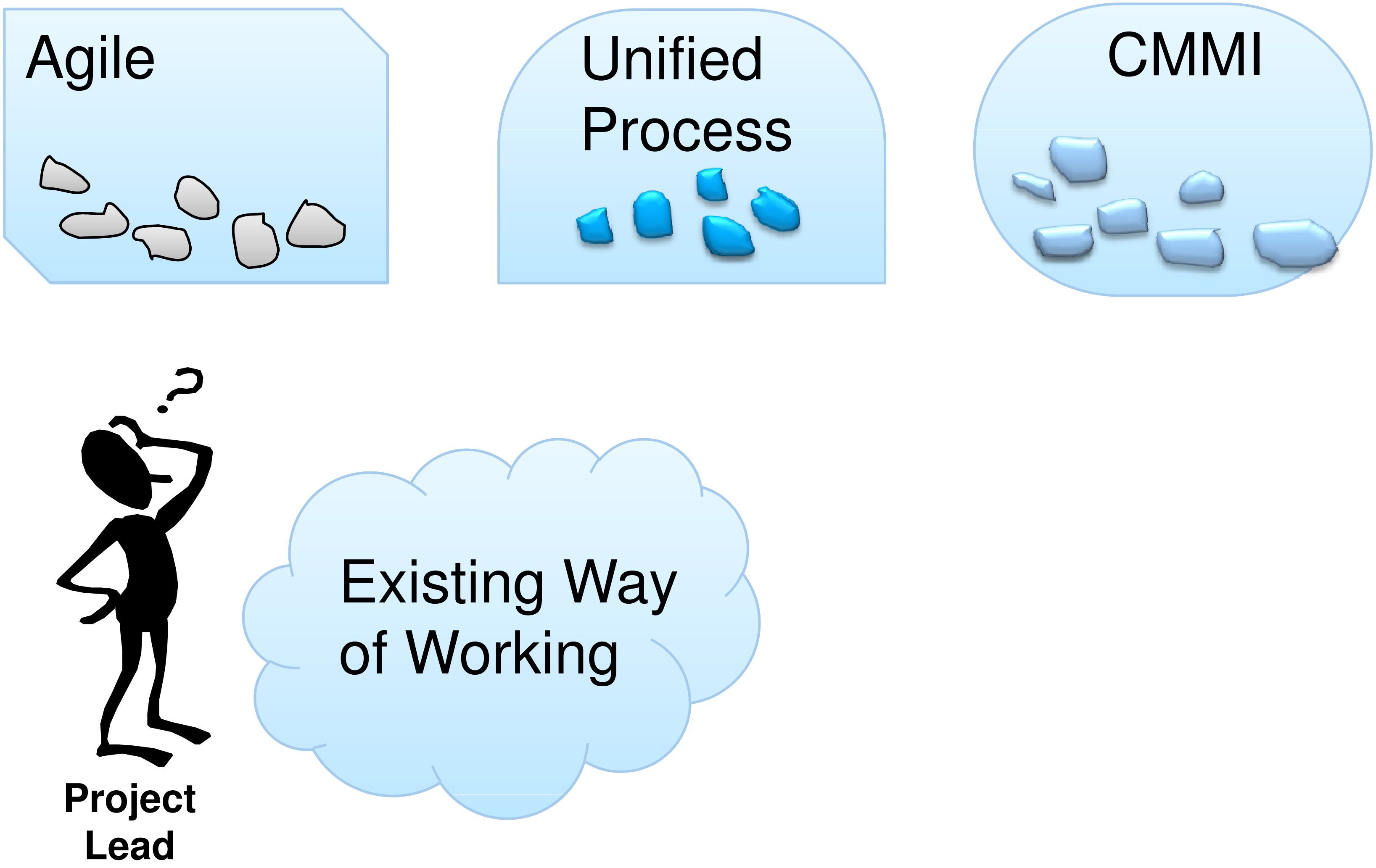
- Law of nature: People don't read documents



This is smart!

Focus on the essentials - the placeholders for conversations – people figure out the rest themselves

Not smart with Process/Methodology/Approach



Agile

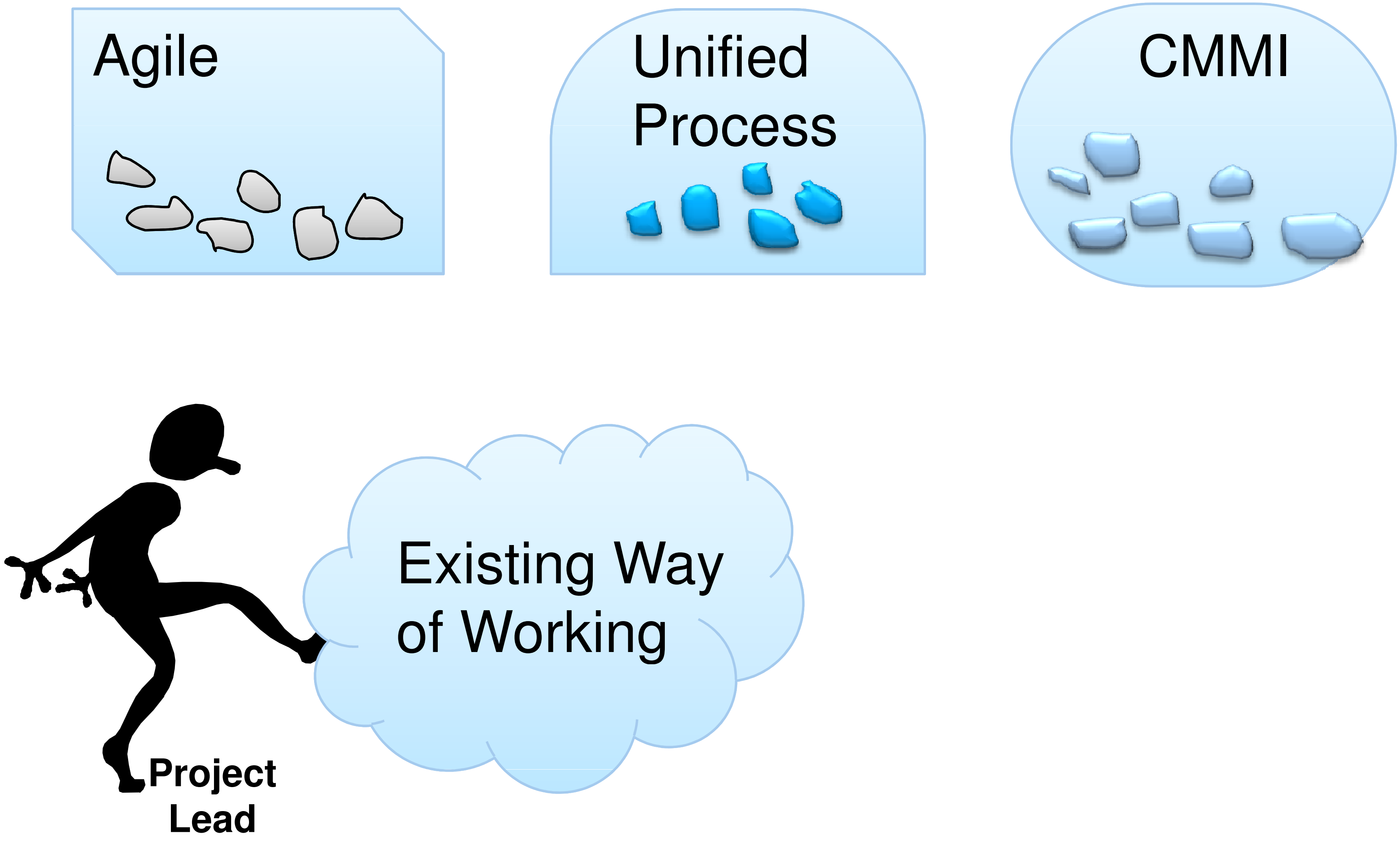
Unified Process

CMMI

Existing Way of Working

Project Lead

Not smart with Process/Methodology/Approach



Agile

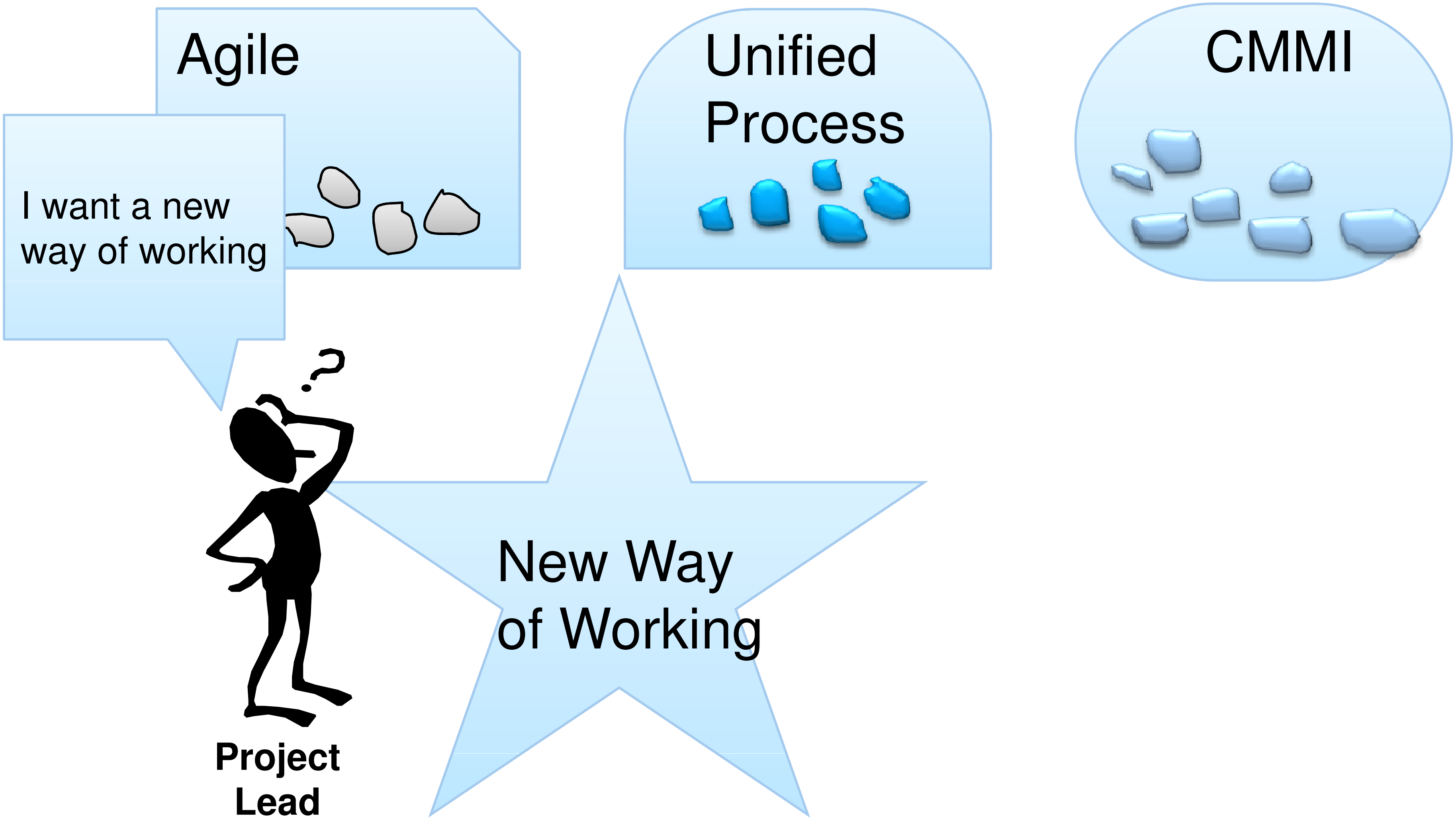
Unified Process

CMMI

Existing Way of Working

Project Lead

Not smart with Process/Methodology/Approach



Agile

Unified Process

CMMI

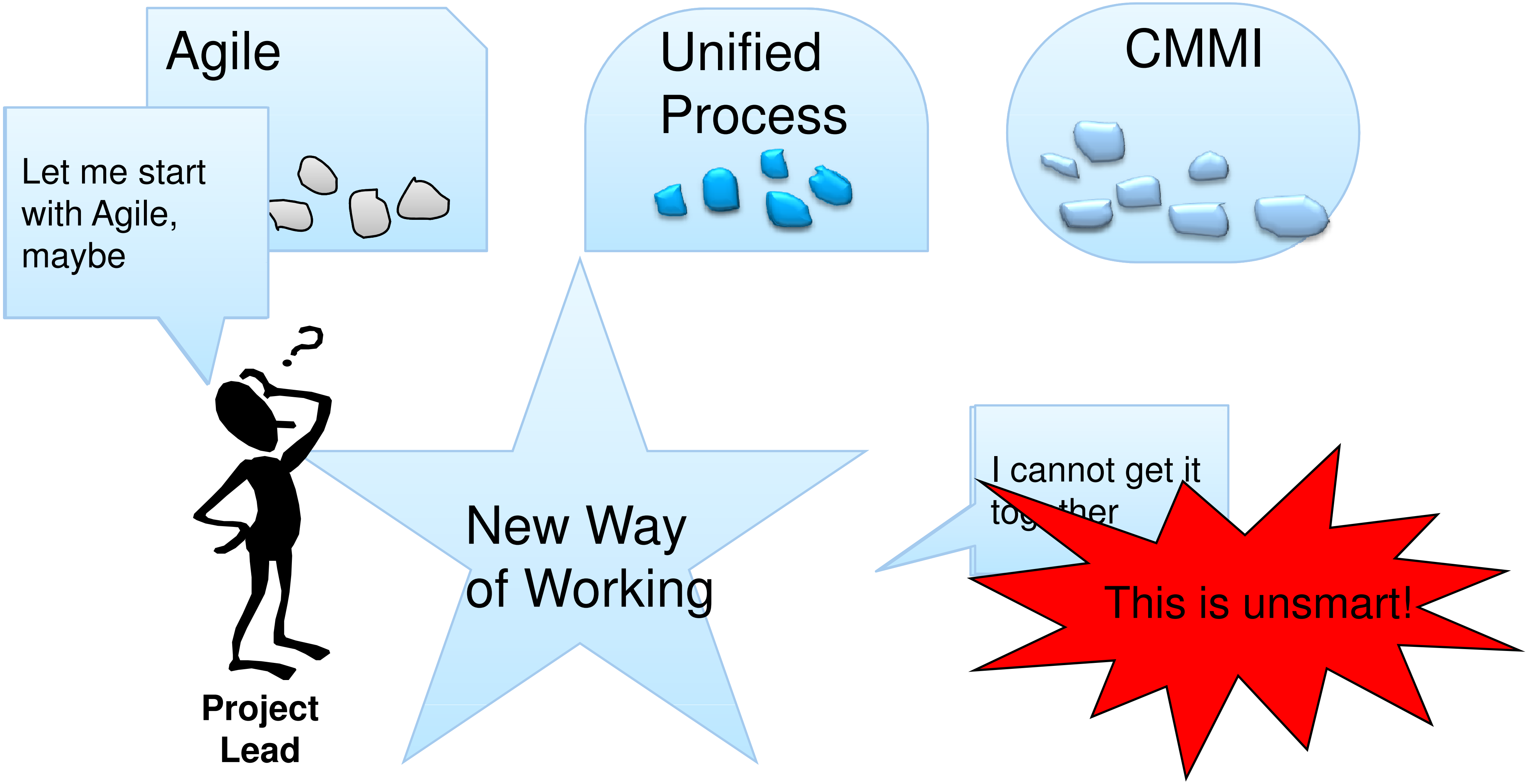
I want a new way of working

New Way of Working

Project Lead

Not smart with Process/Methodology/Approach

We have got enough of process



Agile

Let me start with Agile, maybe

Unified Process

CMMI

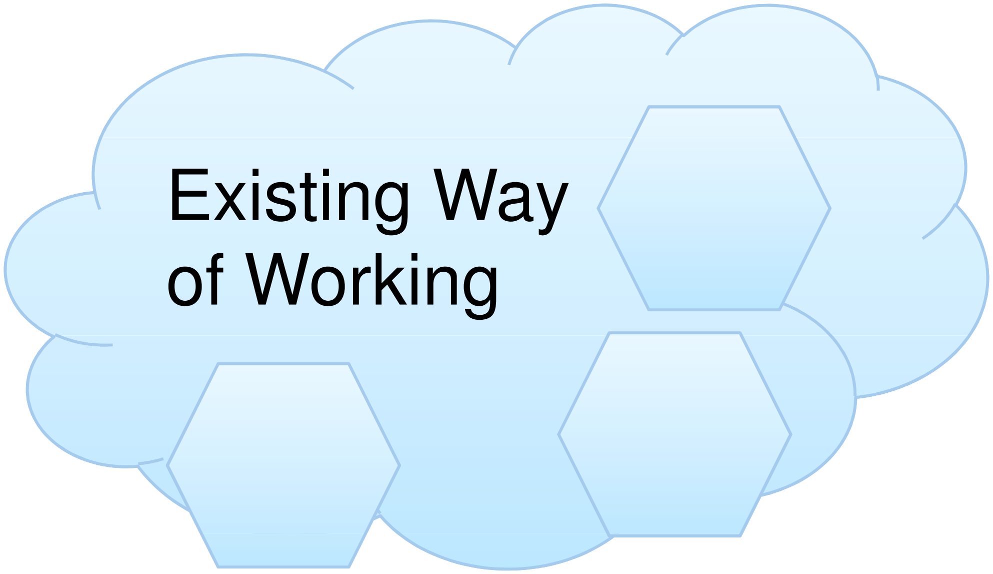
New Way of Working

Project Lead

I cannot get it together

This is unsmart!

Smart with Process/Methodology/Approach



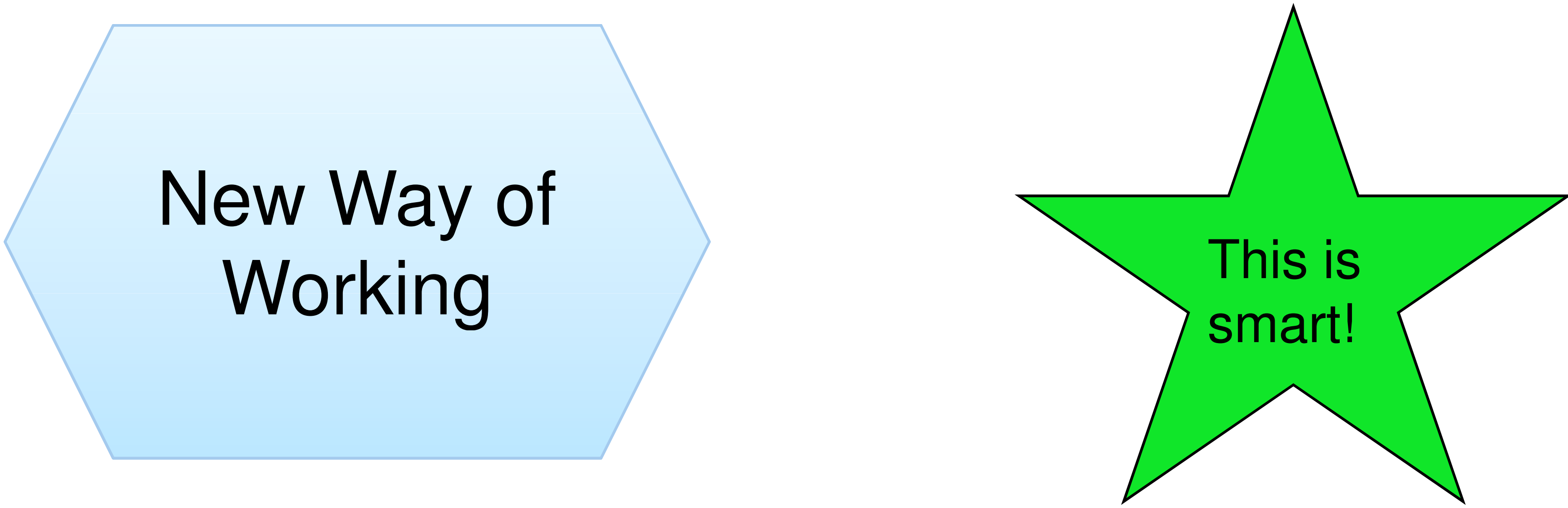
Don’t throw out your baby with the bathwater:

1. start from your existing way of working,
2. find your pain points, and
3. change one practice at the time.

Smart with Process/Methodology/Approach

And the Law of Nature: People don’t read process books

So focus on the essentials, people figure out the rest themselves



New Way of Working

This is smart!

Don’t throw away your baby with the bathwater:

1. start from your existing way of working,
2. find your pain points, and
3. change one practice at the time.

Agenda

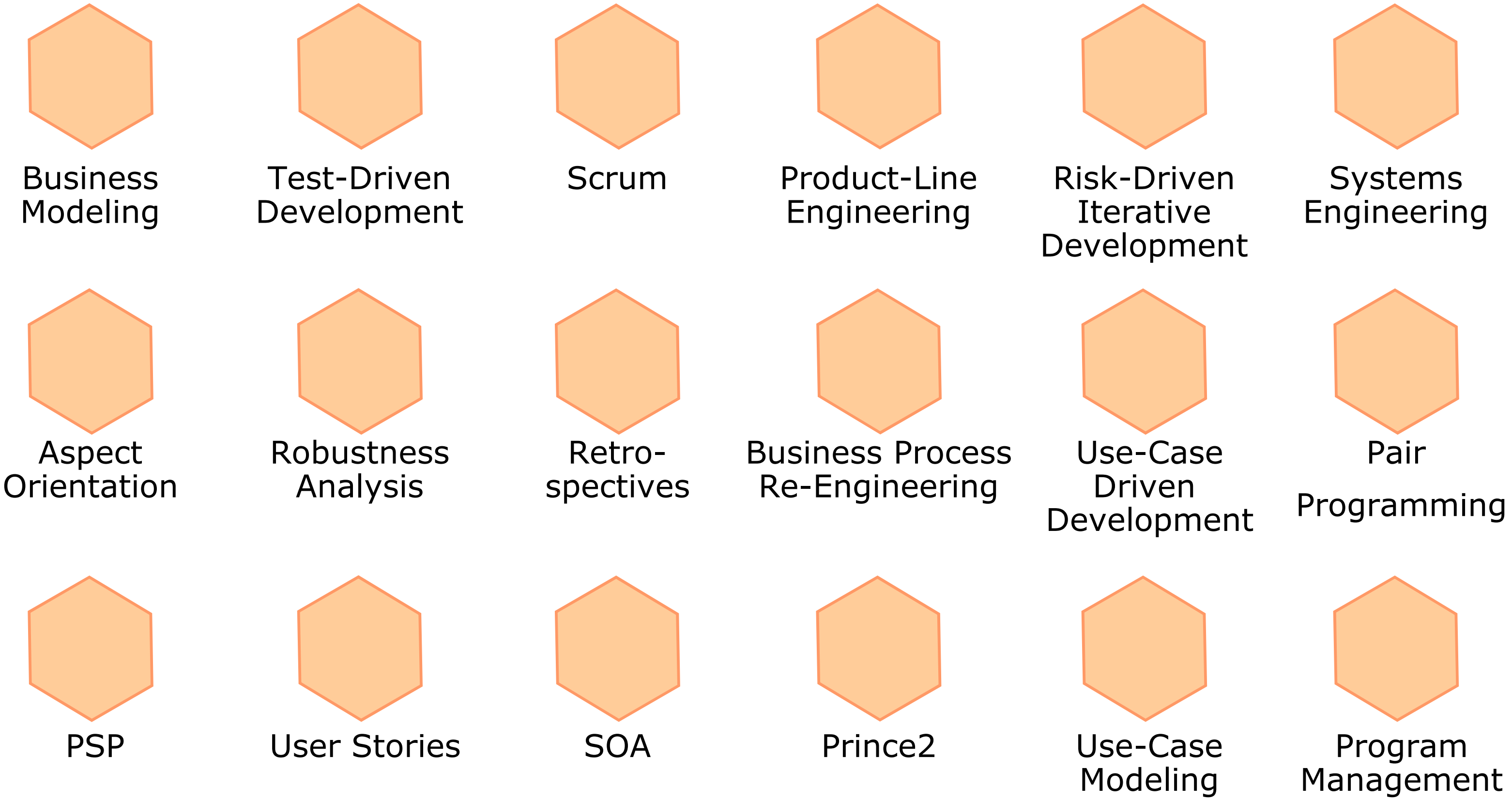
1. What does Smart mean?
2. Smart Cases – Recognize it when you see it
3. How do you become Smart
4. What does Smart really mean?

Summary: Key Elements to Smart Cases

- **People**: Software is developed by people, not by process and tools.
- **Teams**: A software team is like a sport team with all needed competencies to win.
- **Project**: Think big, build in many steps.
- **Requirements**: Design your project for requirement changes.
- **Architecture**: Start to build a skinny system, add muscles in later steps.
- **Modeling**: Don’t duplicate the effort by modeling everything
- **Testing**: Whatever you do you are not done until you have verified what you wanted to do.
- **Documentation**: Focus on the essentials - the placeholders for conversations – people figure out the rest themselves
- **Process**: Don’t throw away your baby with the bathwater:
    1. start from your existing way of working,
    2. find your pain points, and
    3. change one practice at the time.

How do you become Smart?

- You need knowledge in *good* (maybe best) practices
  - There are 100’s of practices, some of them are good



Business Modeling

Test-Driven Development

Scrum

Product-Line Engineering

Risk-Driven Iterative Development

Systems Engineering

Aspect Orientation

Robustness Analysis

Retro-spectives

Business Process Re-Engineering

Use-Case Driven Development

Pair

Programming

PSP

User Stories

SOA

Prince2

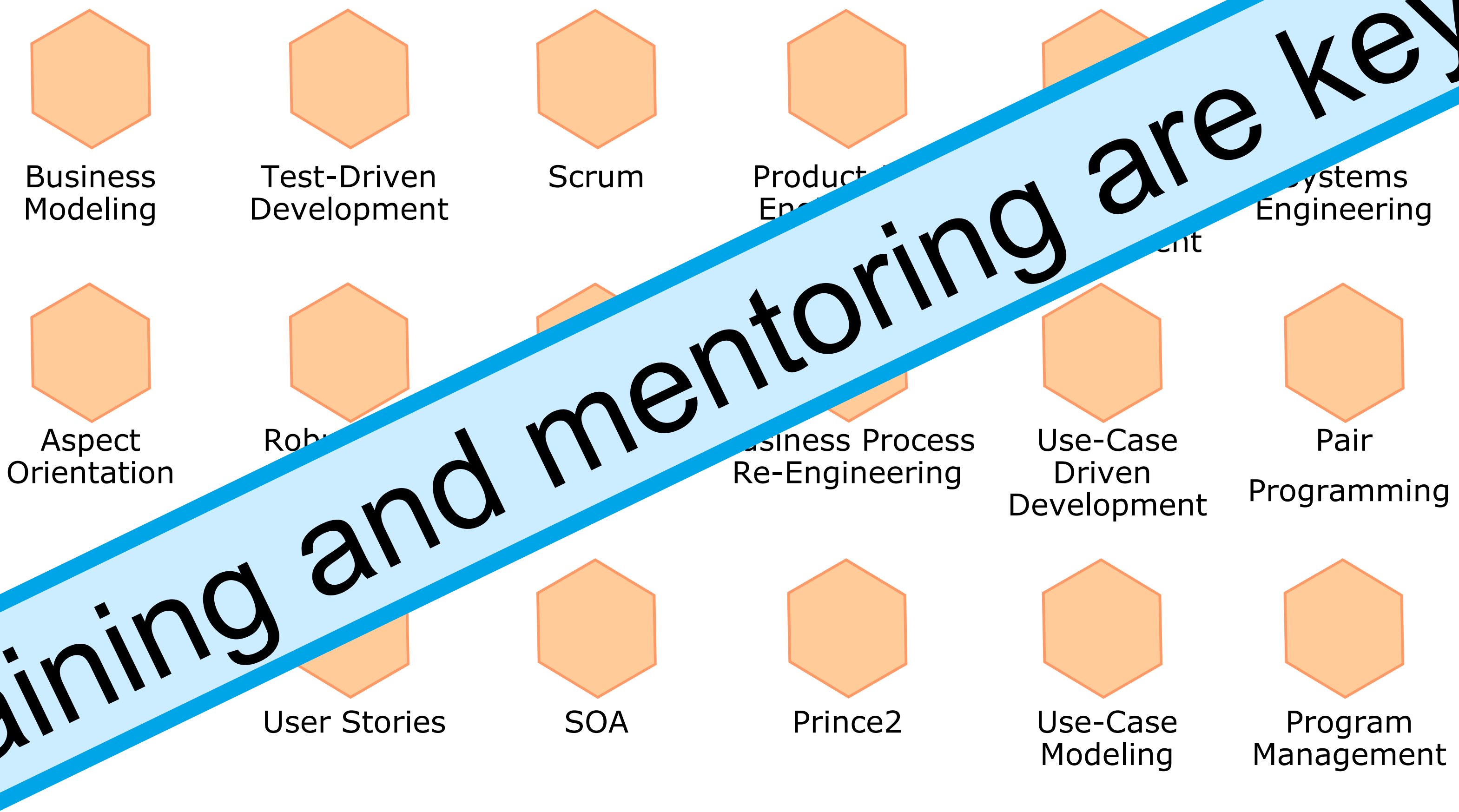
Use-Case Modeling

Program Management

- And you need experience in using them

How do you become Smart?

- You need knowledge in *good* (maybe best) practices
  - There are 100’s of practices, some of them are good



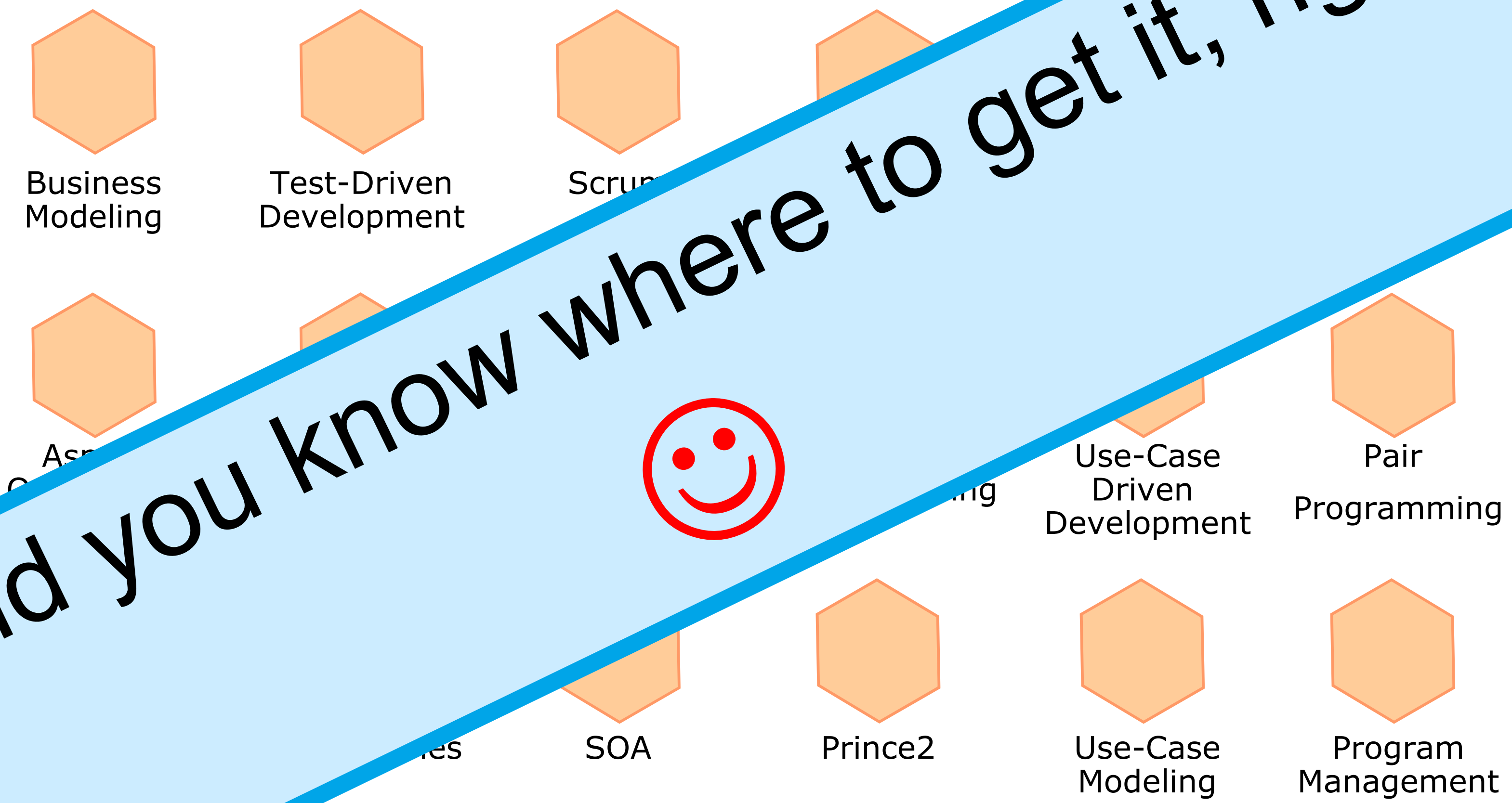
Training and mentoring are key

- And you need experience in using them

© 2008 Ivar Jacobson International

How do you become Smart?

- You need knowledge in *good* (maybe best) practices
  - There are 100’s of practices, some of them are good



… and you know where to get it, right? ☺

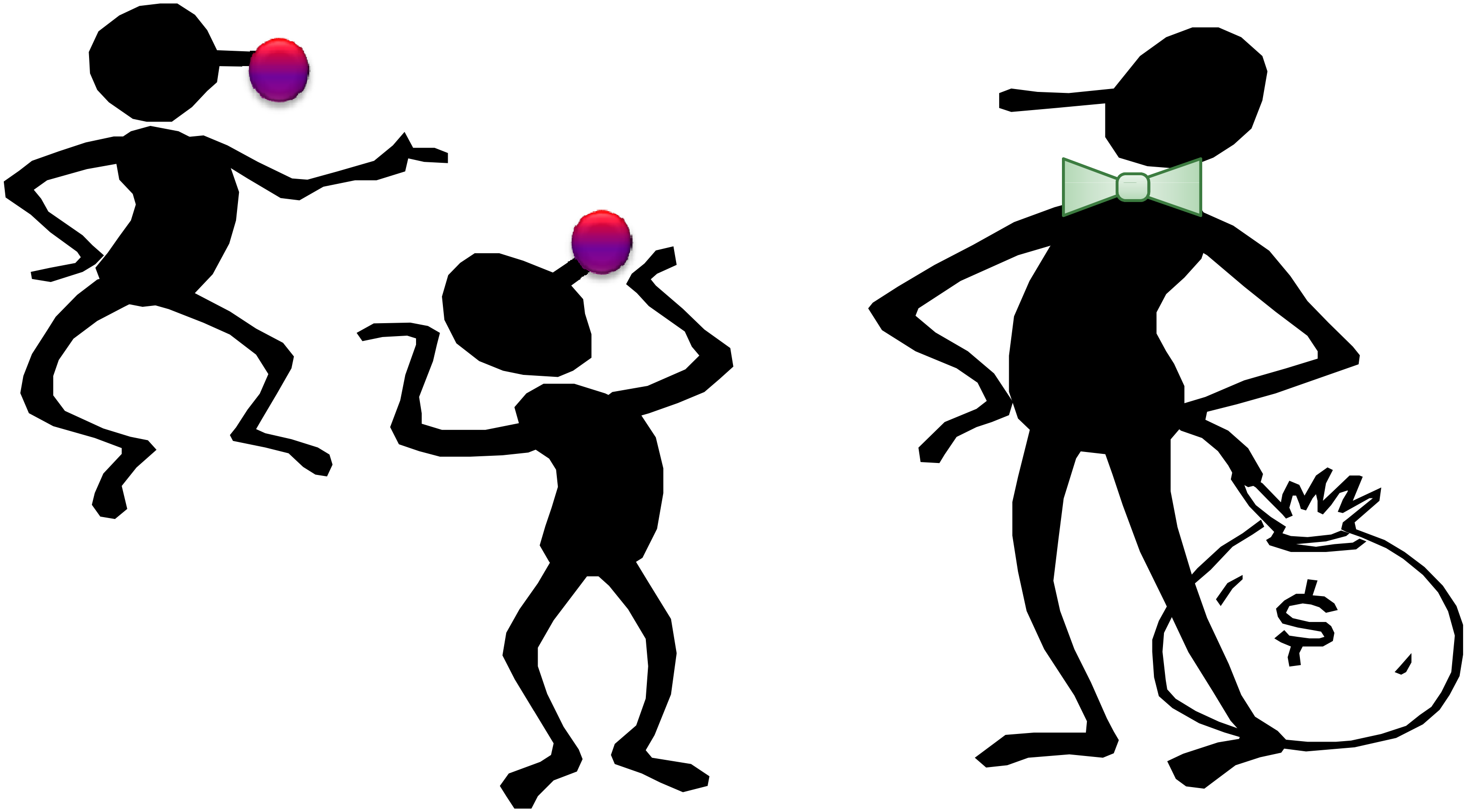
- need experience in using them

Agenda

1. What does Smart mean?
2. Smart Cases – Recognize it when you see it
3. How do you become Smart
4. What does Smart really mean?

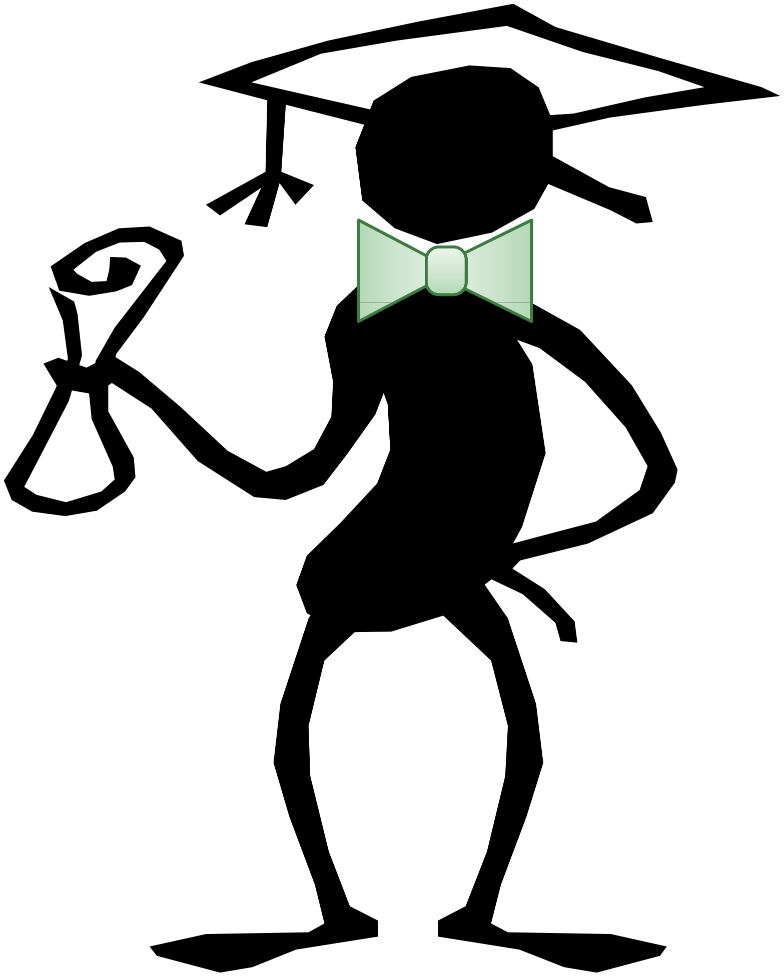
What does Smart really mean?

- If you didn’t get what smart is so far let me summarize it to you



Of course, eventually it comes back to you, but

We can all become smarter.



Thank You

Contact me at ivar@ivarjacobson.com