

What CMMI Cannot Give You: Good Software

Ivar Jacobson

ivar@ivarjacobson.com

ivar@jaczone.com

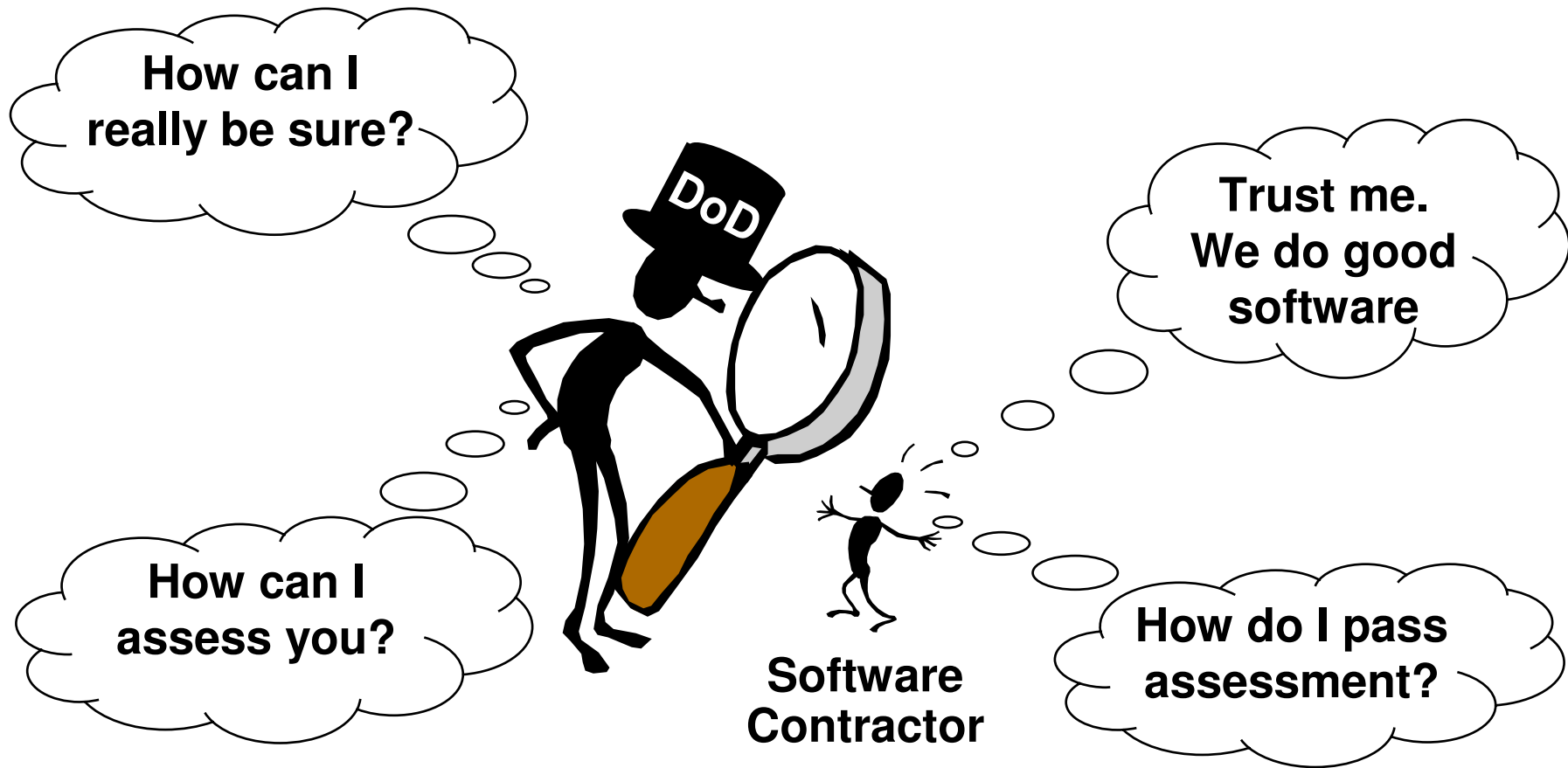
Objective

- To understand what CMM/CMMI is and what it is not
- To demonstrate how the unified process helps you achieve real software development maturity

Agenda

- What CMM/CMMI means
- What Software Development Maturity Really Means
- Developing Really Good Software
- Achieving Real Process Improvement

Purpose of CMM/CMMI



Is CMM/CMMI the answer?

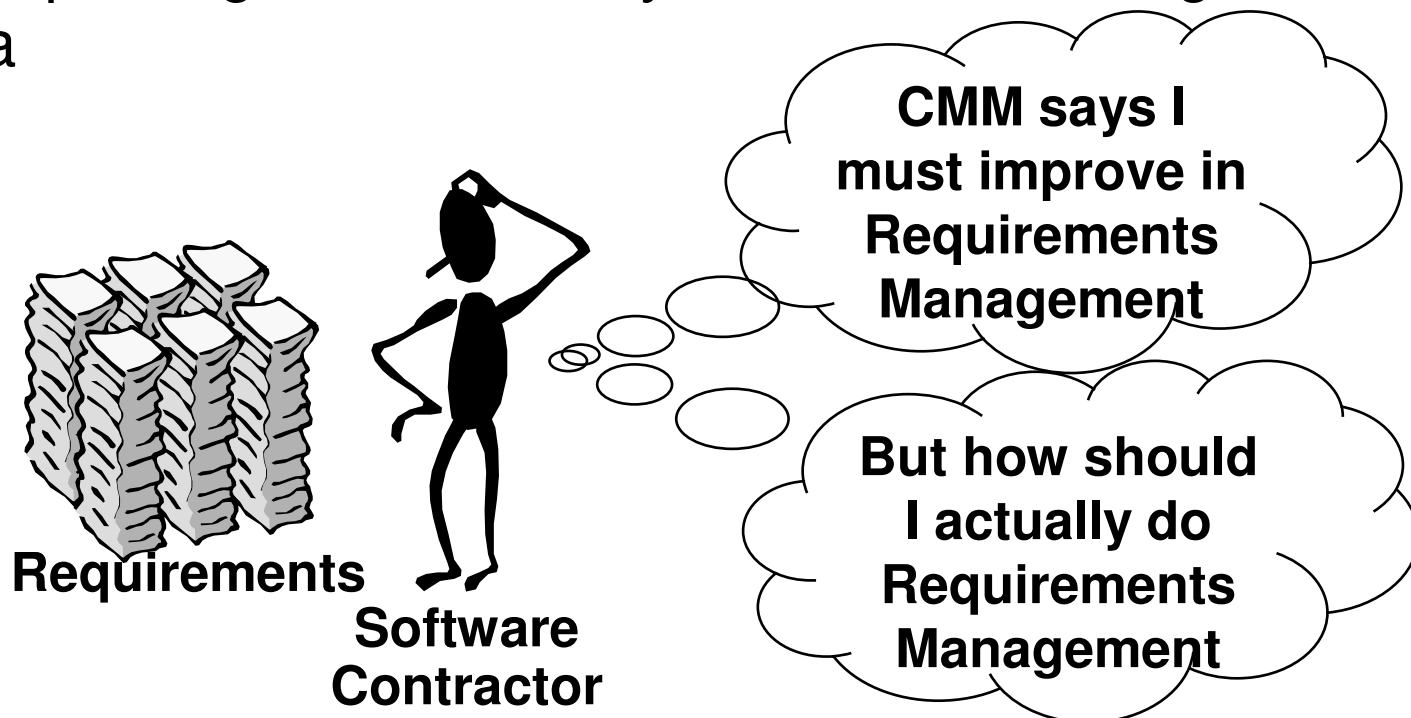
What is CMM/CMMI?

- “The CMM is a framework that describes the key elements of an effective process.
 - It provides a foundation for process improvement.
 - The CMM describes an evolutionary improvement path from an ad hoc, immature process to a mature, disciplined process.
 - It has five levels of progressive process maturity (Initial, Repeatable, Defined, Managed, and Optimizing).”
- “The CMM has become a de facto industry standard for assessing and improving processes”

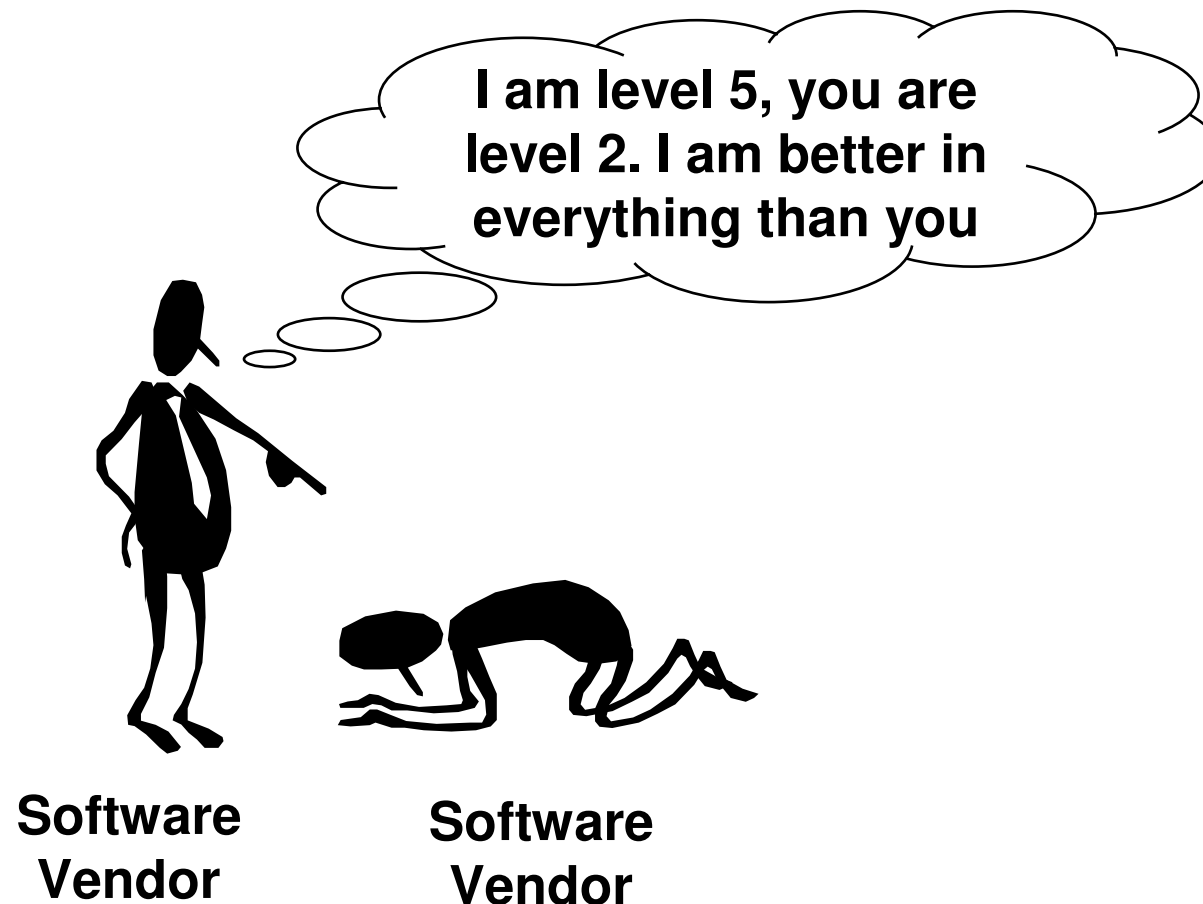
<http://www.state.gov/m/a/sdbu/pubs/9729.htm>
- CMMI is a more comprehensive than CMM, but the principles and purpose are largely the same

Truth about CMM/CMMI

- CMM contains five levels of progressive process maturity (Initial, Repeatable, Defined, Managed, and Optimizing), and indicates the Process Areas (PA) that are addressed at each level.
- It tells you which areas you (as software vendor) should be improving, but not how you should be doing in that area

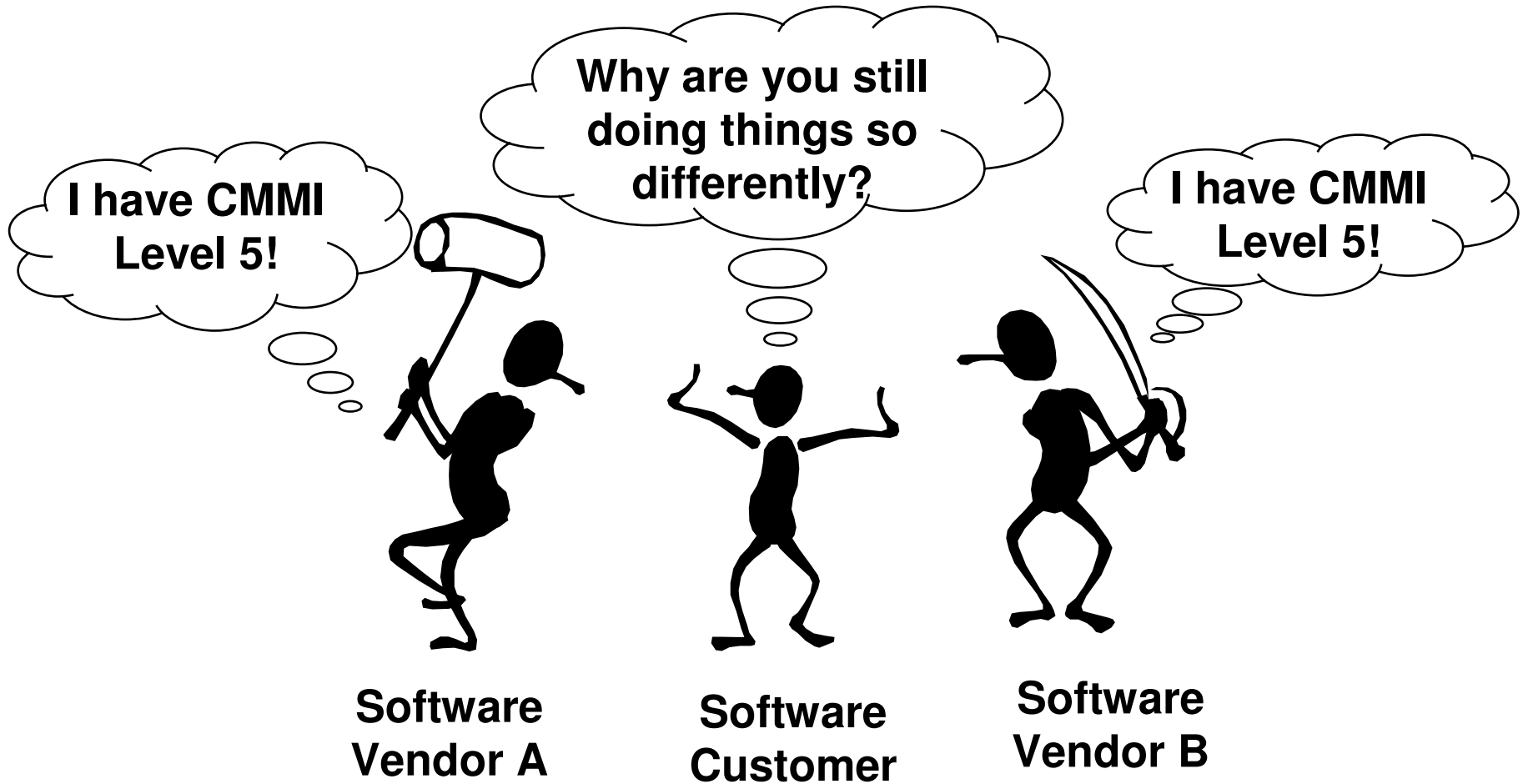


What does CMMI Level means?



Does higher CMM level ensure better capability in each area?

What does CMMI Level means?



Does same CMM level means same process?

What is a good foundation for CMMI?

Project Management

Project Planning
Project Monitoring and Control
Supplier Agreement Management
Integrated Project Management(IPPD)
Integrated Supplier Management (SS)
Integrated Teaming (IPPD)
Risk Management
Quantitative Project Management

Support

Configuration Management
Process and Product Quality Assurance
Measurement and Analysis
Causal Analysis and Resolution
Decision Analysis and Resolution
Organizational Environment for Integration (IPPD)

Engineering

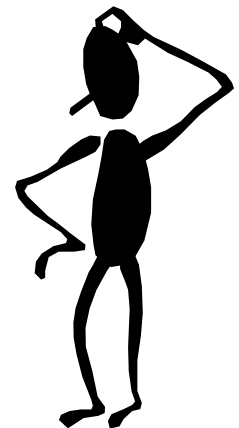
Requirements Management
Requirements Development
Technical Solution
Product Integration
Verification
Validation

Process Management

Organizational Process Focus
Organizational Process Definition
Organizational Training
Organizational Process Performance
Organizational Innovation and Deployment

CMMI Process Areas

- Many process areas – what are their relationships?
- Is there any approach that can unify all the different process areas?
- Need to get the foundation right!



Agenda

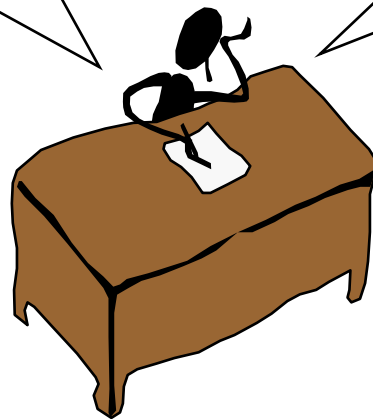
- What CMM/CMMI means
- What Software Development Maturity Really Means
- Developing Really Good Software
- Achieving Real Process Improvement

Real software maturity

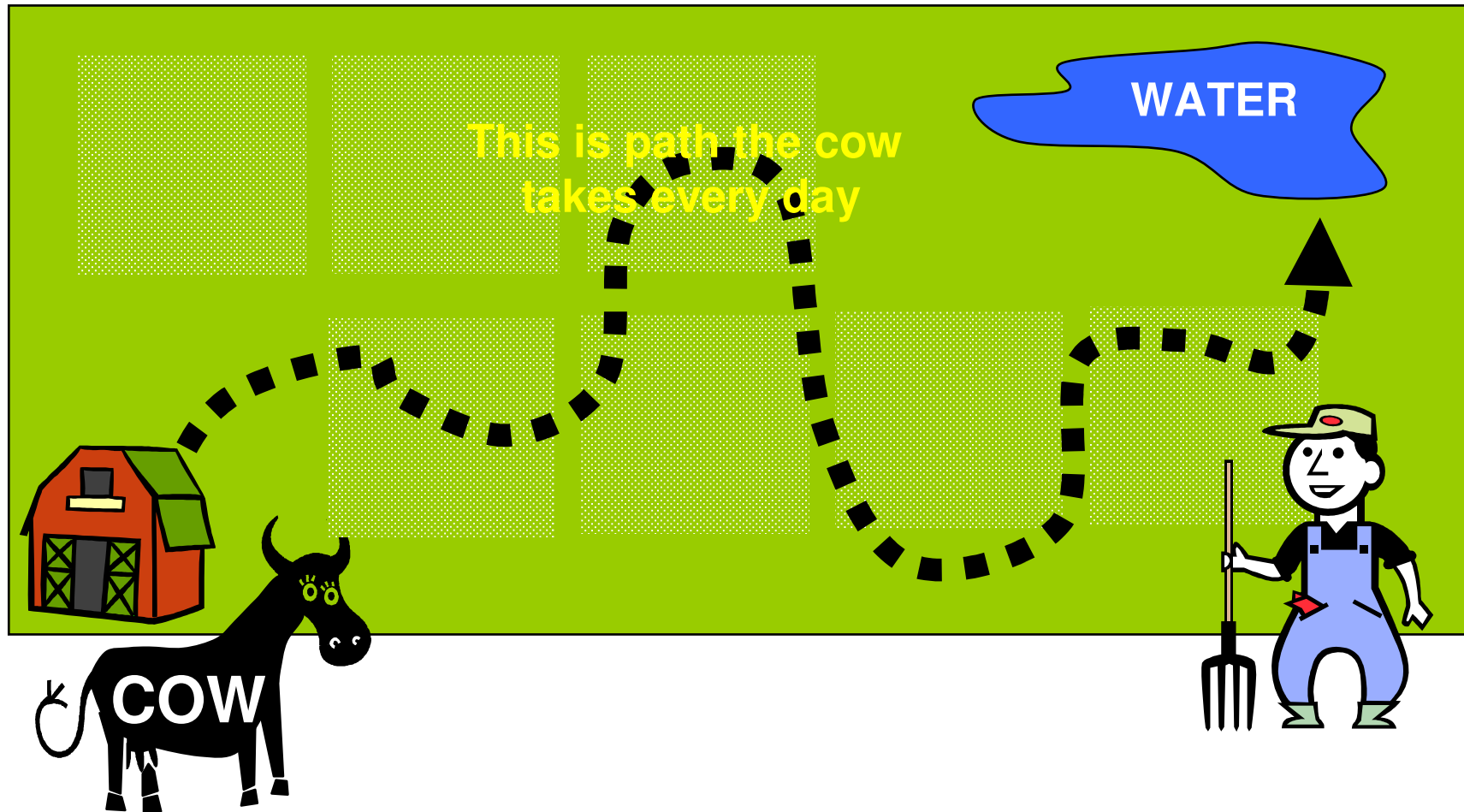
- Is software maturity about describing how you do software?
- Is software maturity about how you get good software?

How do I pass assessment?
To pass assessment, I need
to describe the current way I
am doing requirements, etc.

How do I get good requirements?
Good meaning – well-organized,
testable, understandable, etc.

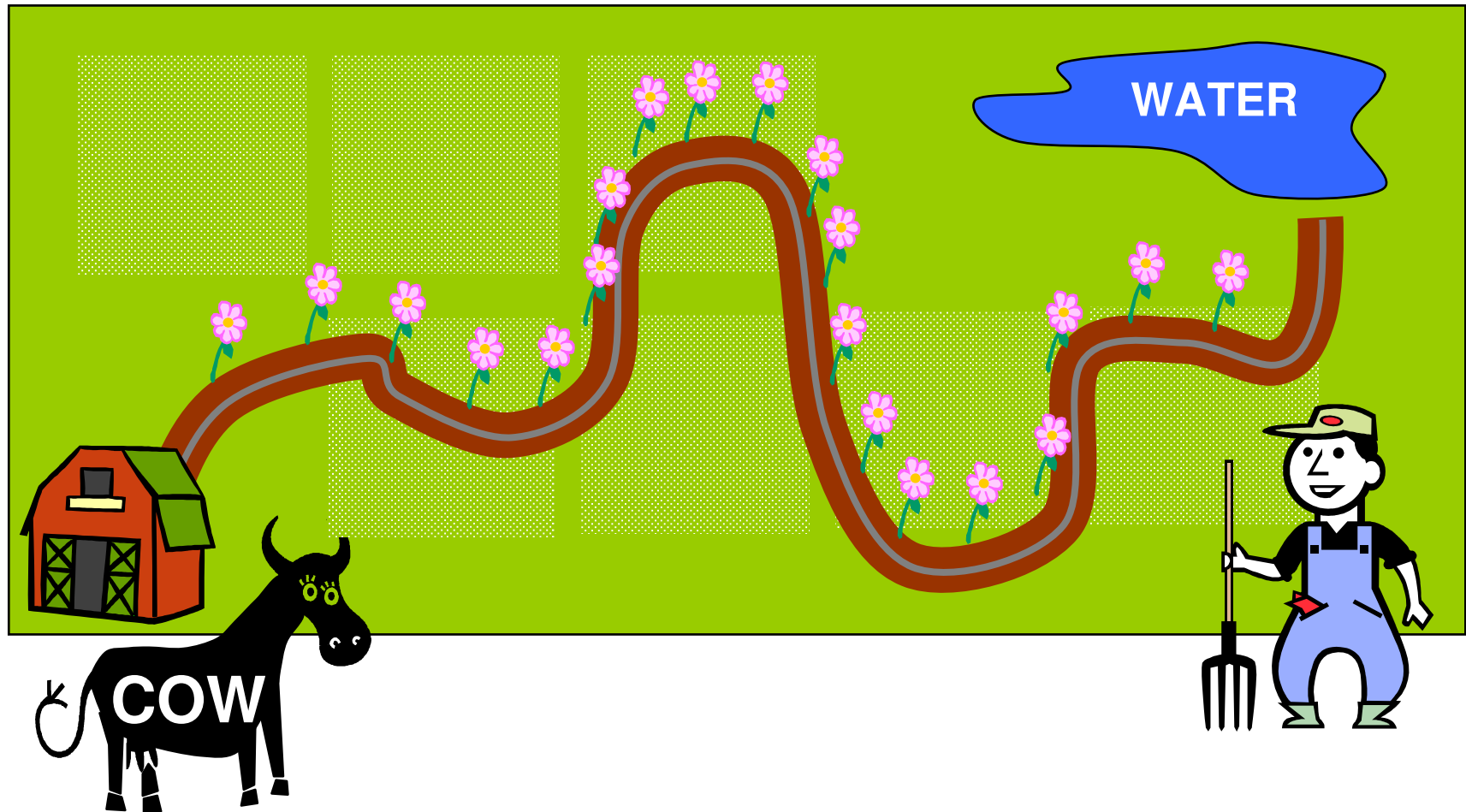


A Story about A Foolish Farmer and a Thirsty Cow



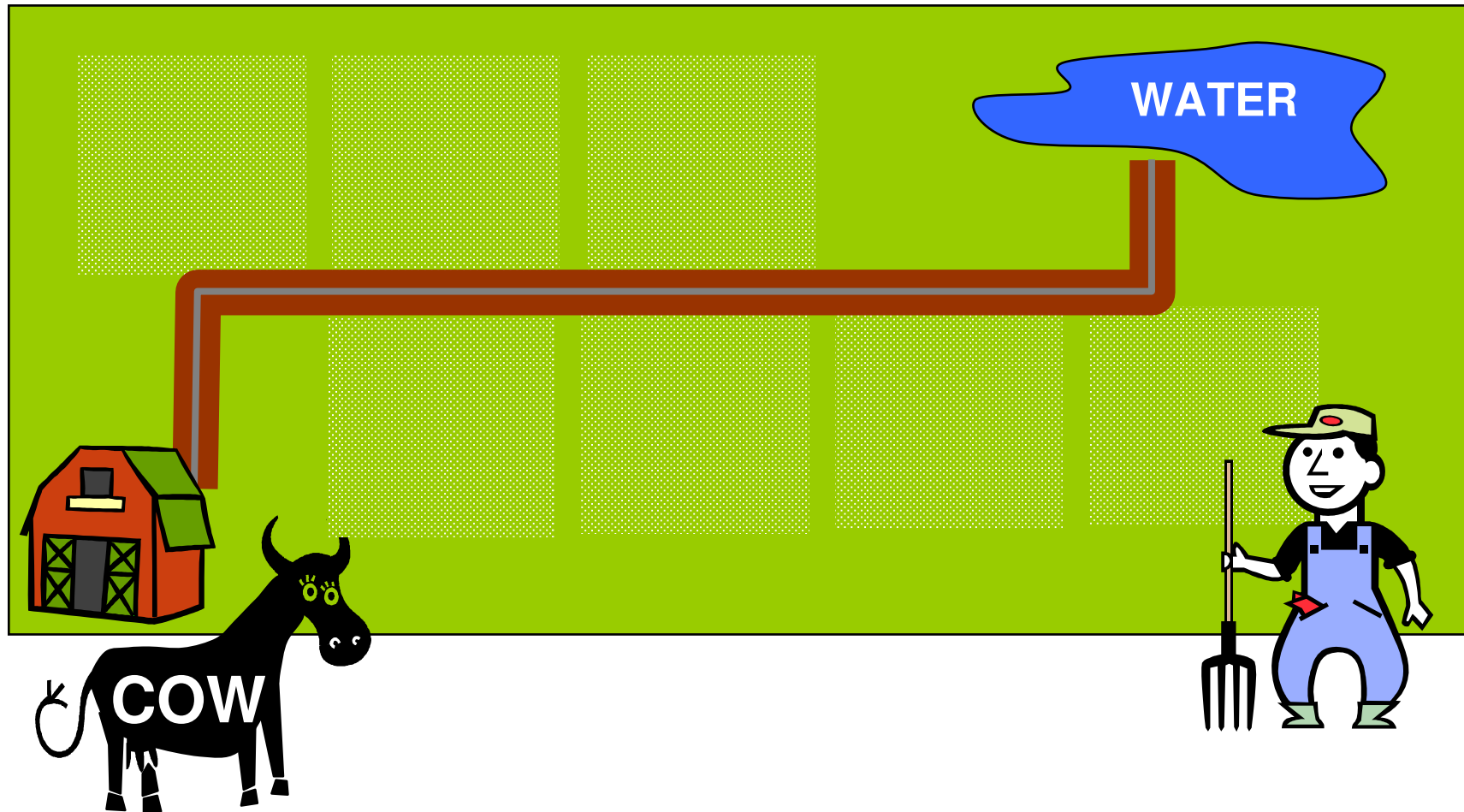
There was once a farmer who has a thirsty cow. The cow follows an unusual route to the water hole every day, and kicked some of the farmer's crops.

A Story about A Foolish Farmer and a Thirsty Cow



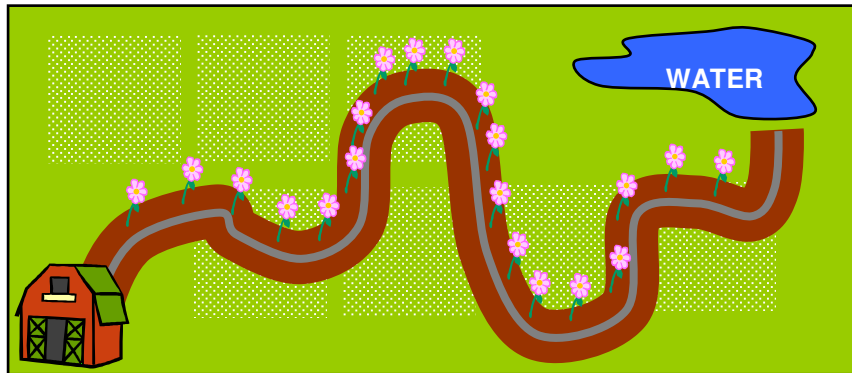
The farmer decides to build a path for the cow, so that the cow no longer damage his crops. He loves his cow, so he puts flowers on the path. Now the cow gets to the water hole faster and his crops stay healthy!

A Story about A Foolish Farmer and a Thirsty Cow

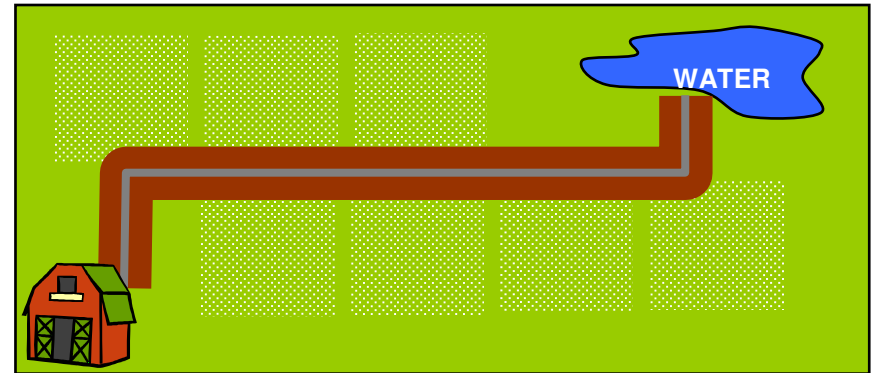


Actually, there is a better route! Much easier to build and better results!

A Story about A Foolish Farmer and a Thirsty Cow



Bad Process

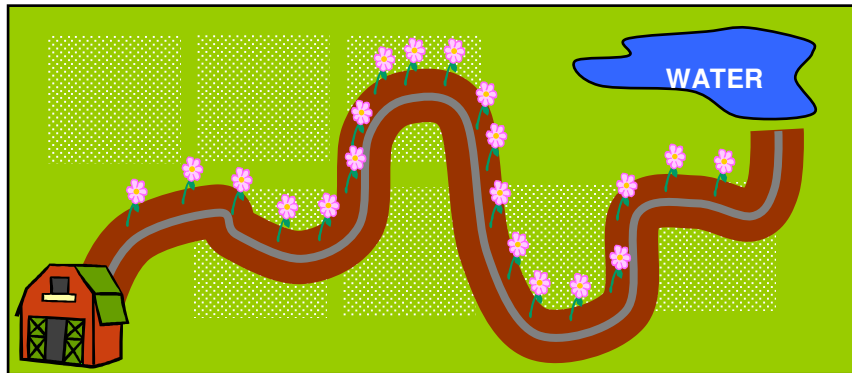


Good Process

- You can describe and even measure a bad process
 - It is easy, but is it worth it?
- But it is best to have a good process in the first place
 - It is not easy, but it is worth it!

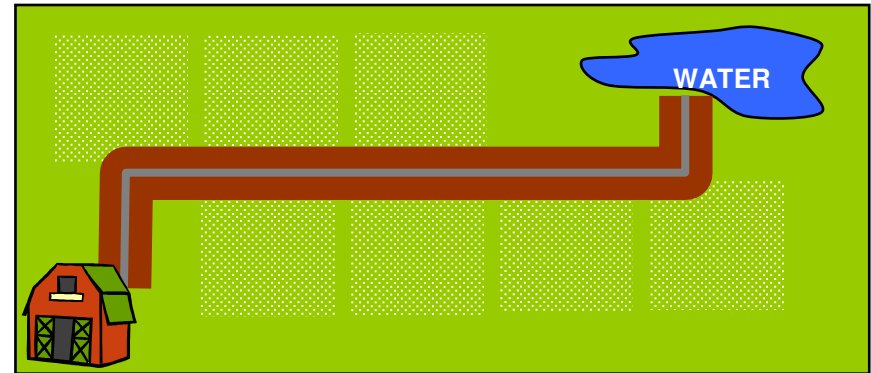
It is easy to make a good process measurable; but it is hard to make a measurable process good

A Story about A Foolish Farmer and a Thirsty Cow



Bad Process

- Focus on **documents**
- **Late** exposure and resolution of risks
- Quality verified **late** in project
- **Complex traceability** from requirement to code
- **Fragile** in face of changes
- Etc.

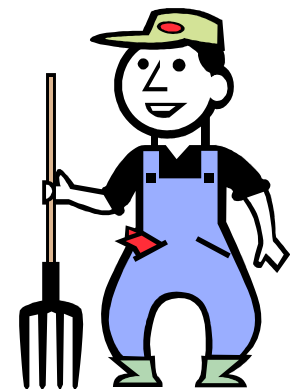
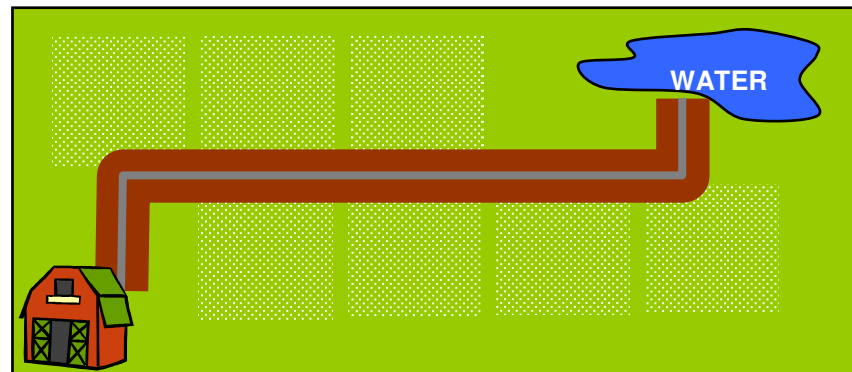


Good Process

- Focus on **working software**
- **Early** exposure and resolution of risks
- Quality **throughout** the project
- **Seamless** transition from requirements to code
- **Robust** to changes
- Etc.

Be a Smart Farmer

- Get your software processes good/right first!
 - Before you describe and publicize the process (or seek CMM certification)
- Software maturity is NOT about how well you describe and measure your software development processes
- Software maturity is about how well you develop software
 - It is about THE RESULT of your software development process



Agenda

- What CMM/CMMI means
- What Software Development Maturity Really Means
- Developing Really Good Software
- Achieving Real Process Improvement

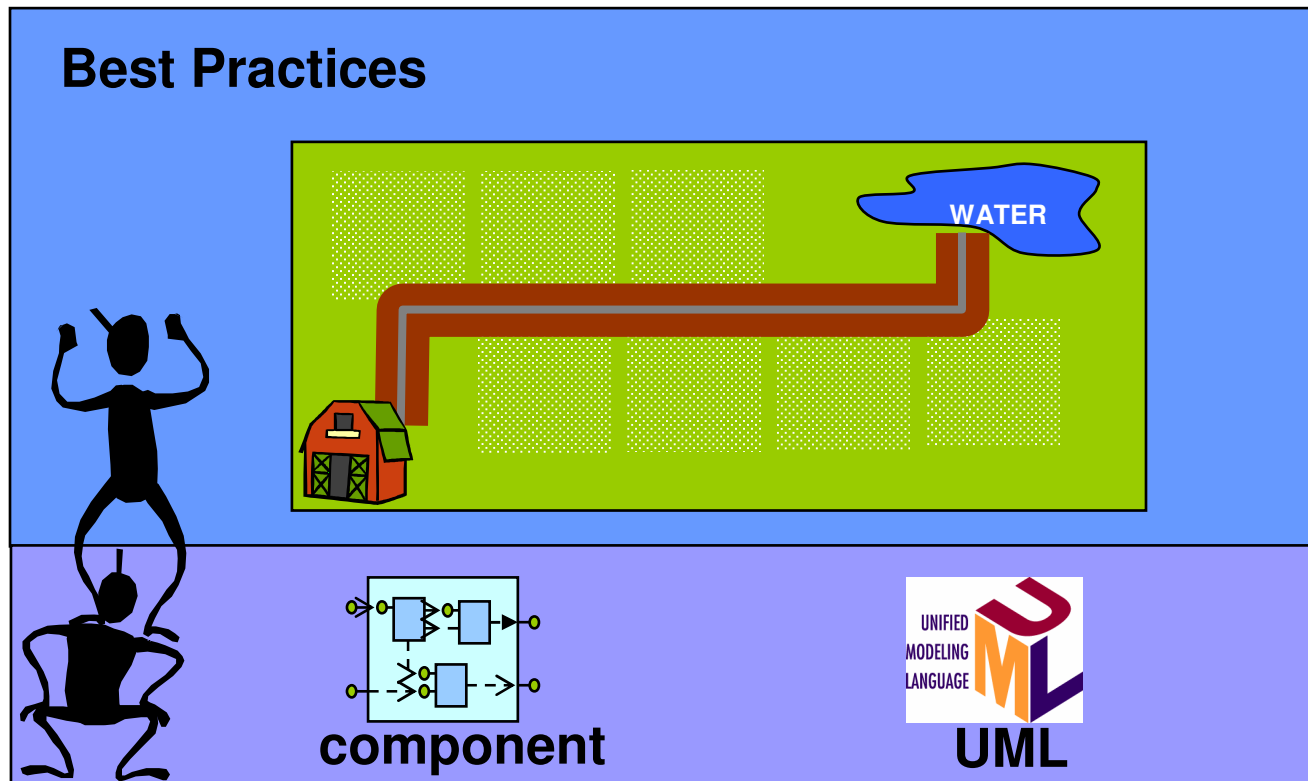
What is good software

- Good software IS component based!
- We need a way to describe software components and how they interact
 - UML



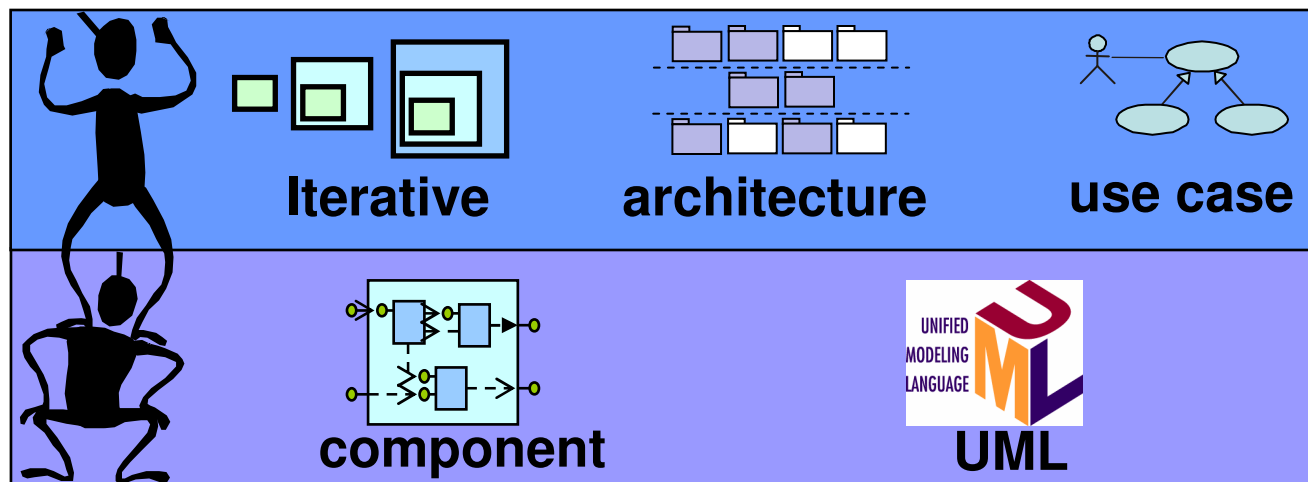
Component technology and UML is not enough!

- Components and UML can be used wrongly
- Need good practices!
 - *How to identify the right components in the first place?*
 - *How to manage the development of components?*



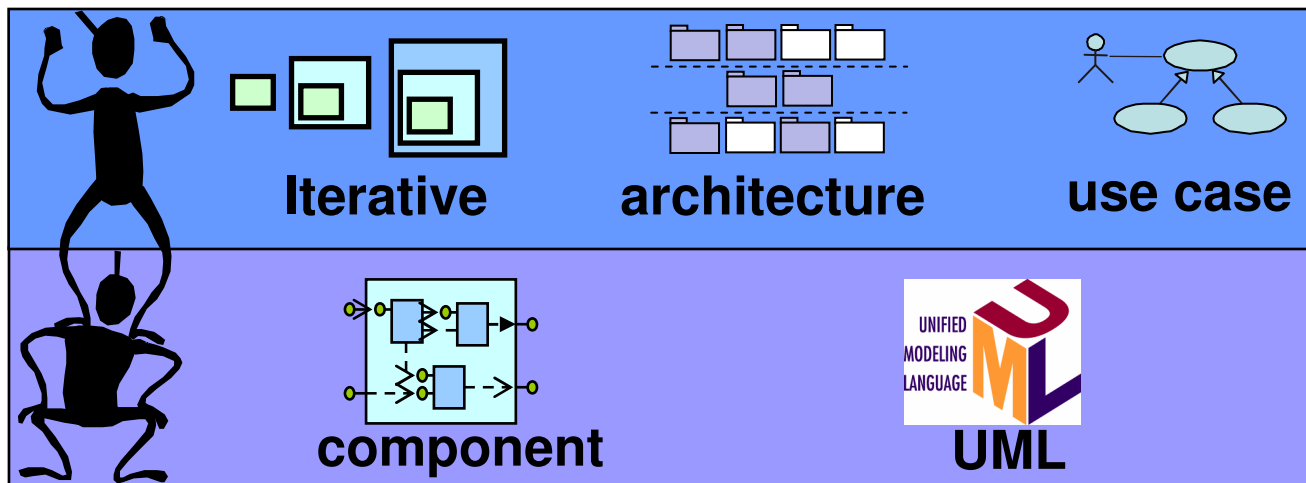
Core Best Practices to Deliver Good Software

- Focus on **working software**
- **Early** exposure and resolution of risks
- Quality **throughout** the project
- **Seamless** transition from requirements to code
- **Robust** to changes

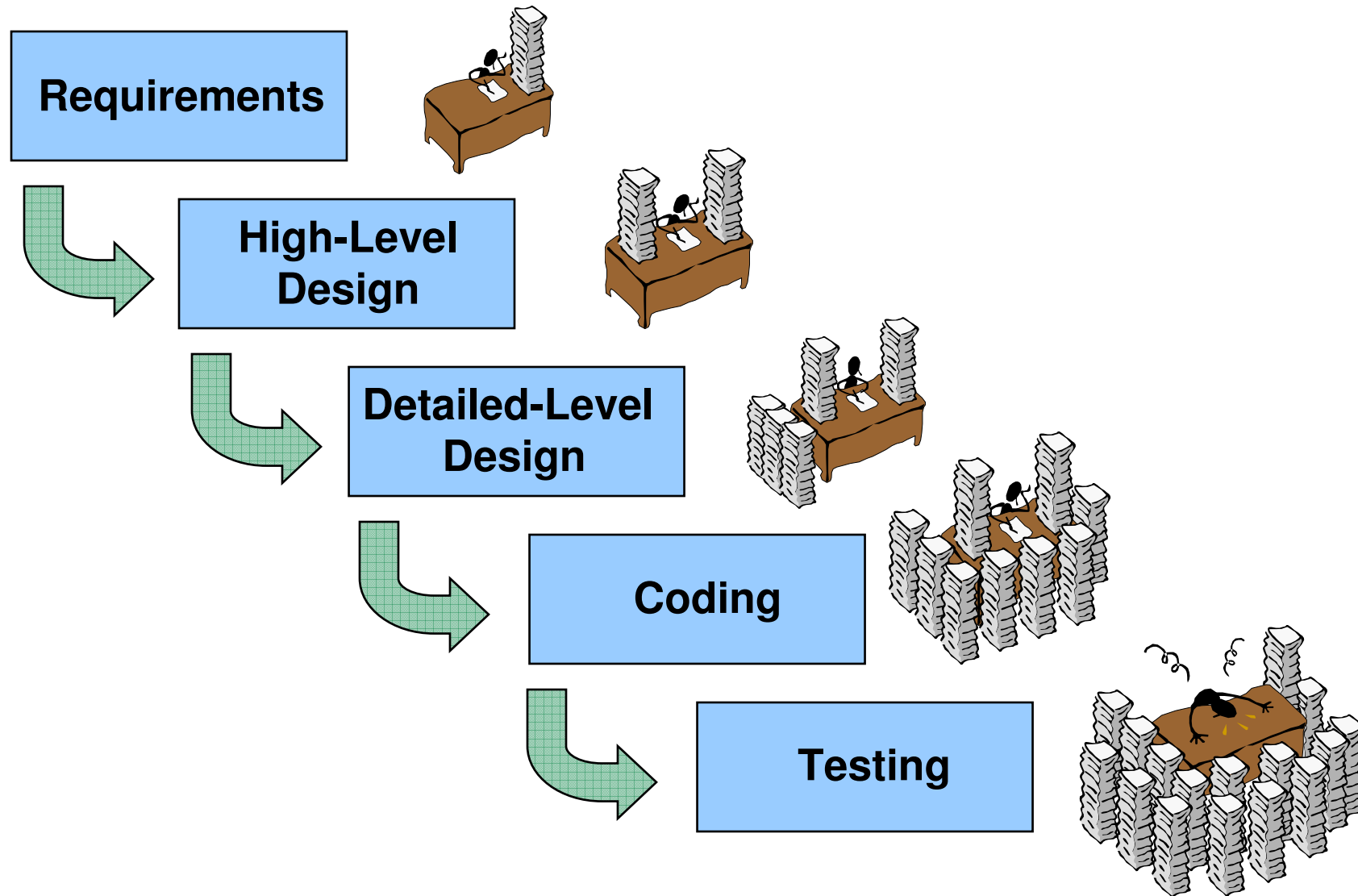


Core Best Practices

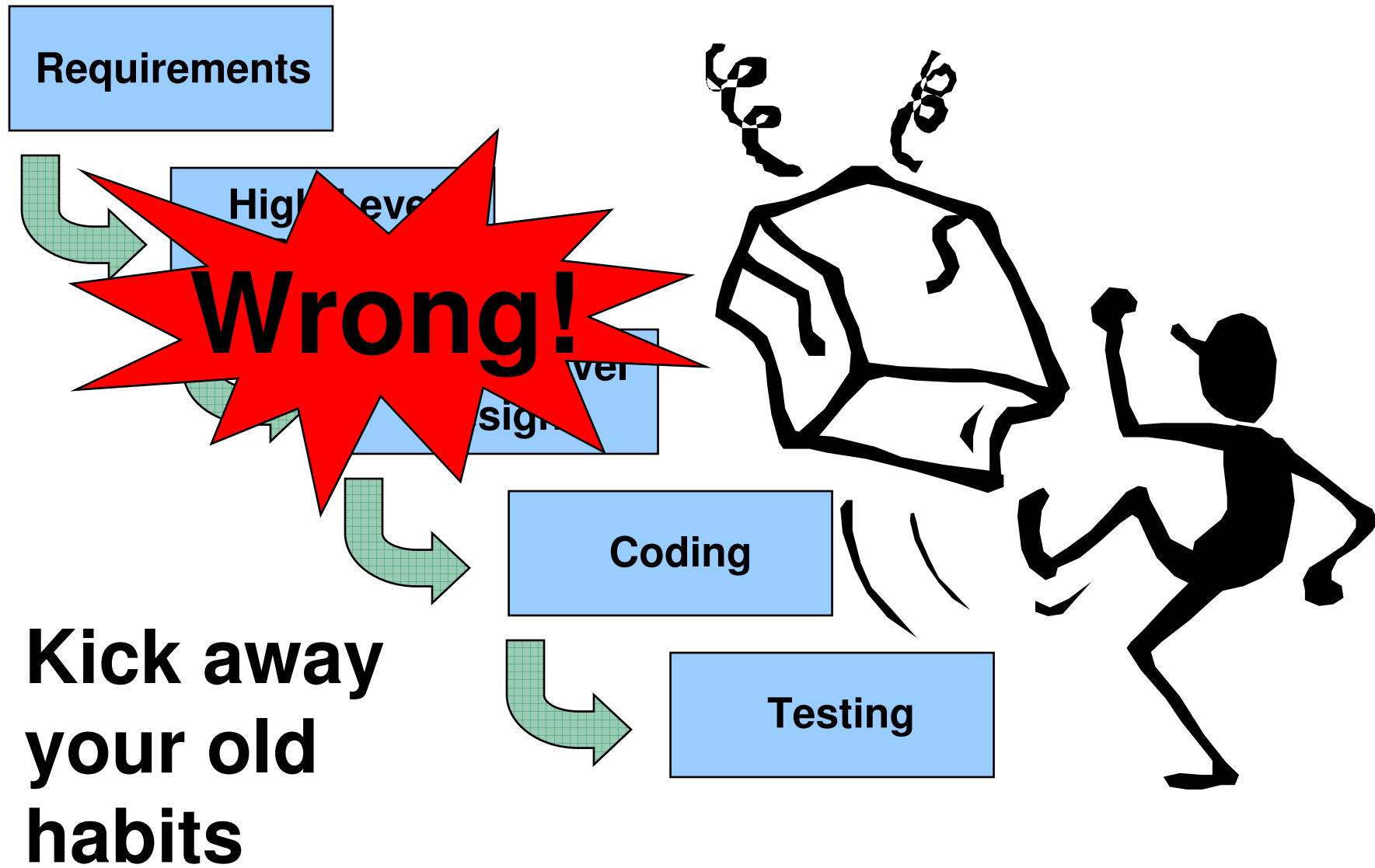
- The most important practices are
 - Iterative development
 - Use cases driven development
 - Architecture-centric



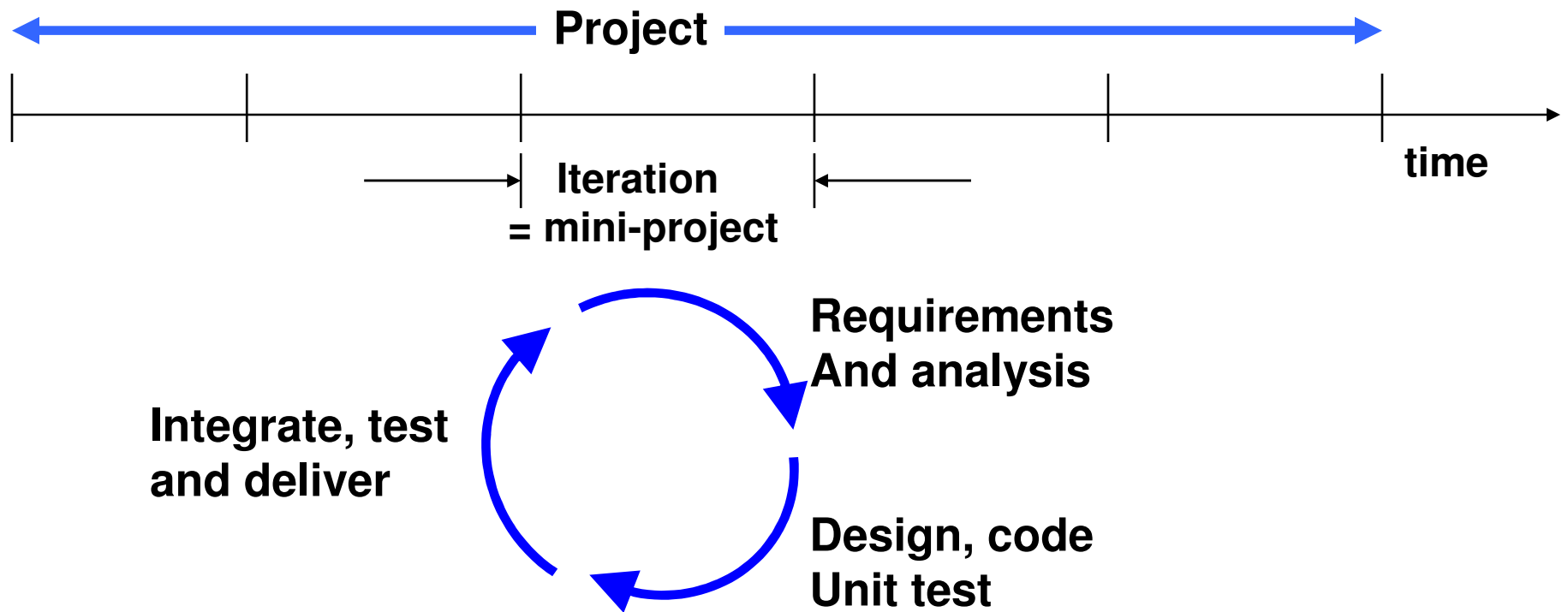
Traditional waterfall development



Traditional waterfall development and your investments

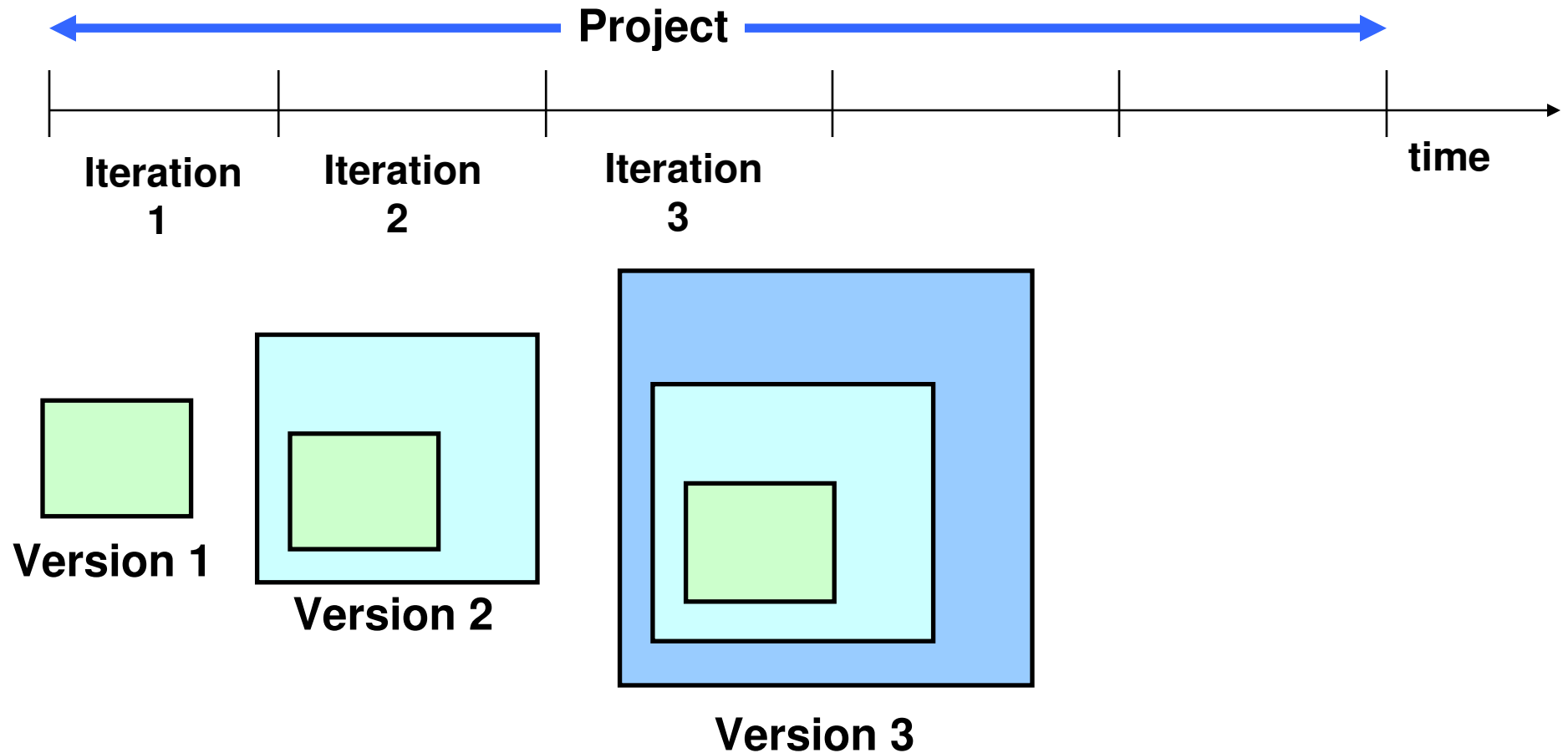


Modern iterative risk driven approach



- The whole project is divided into a number of mini-project
- Each mini-project is an iteration, i.e. iteration=mini-project
- Each mini-project is like a project
- The trick is to select what to do in each iteration

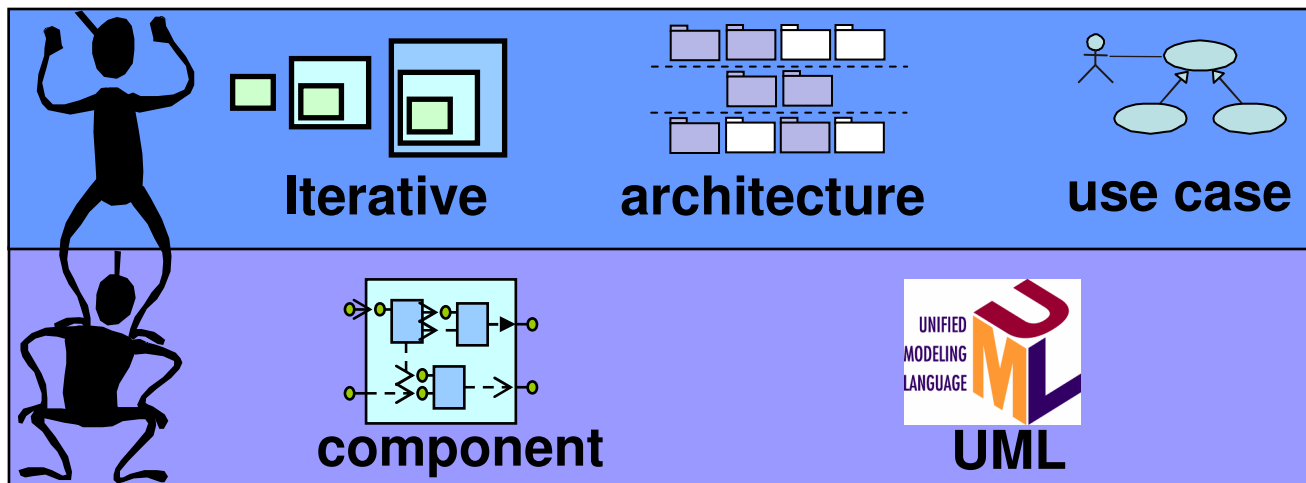
Every iteration produces new version of the system that is tested



If at any time, budget is cut or for whatever reason the project is cancelled, you still have a working system. Your investments are protected.

Core Best Practices

- The most important practices are
 - Iterative development
 - Use cases driven development
 - Architecture-centric

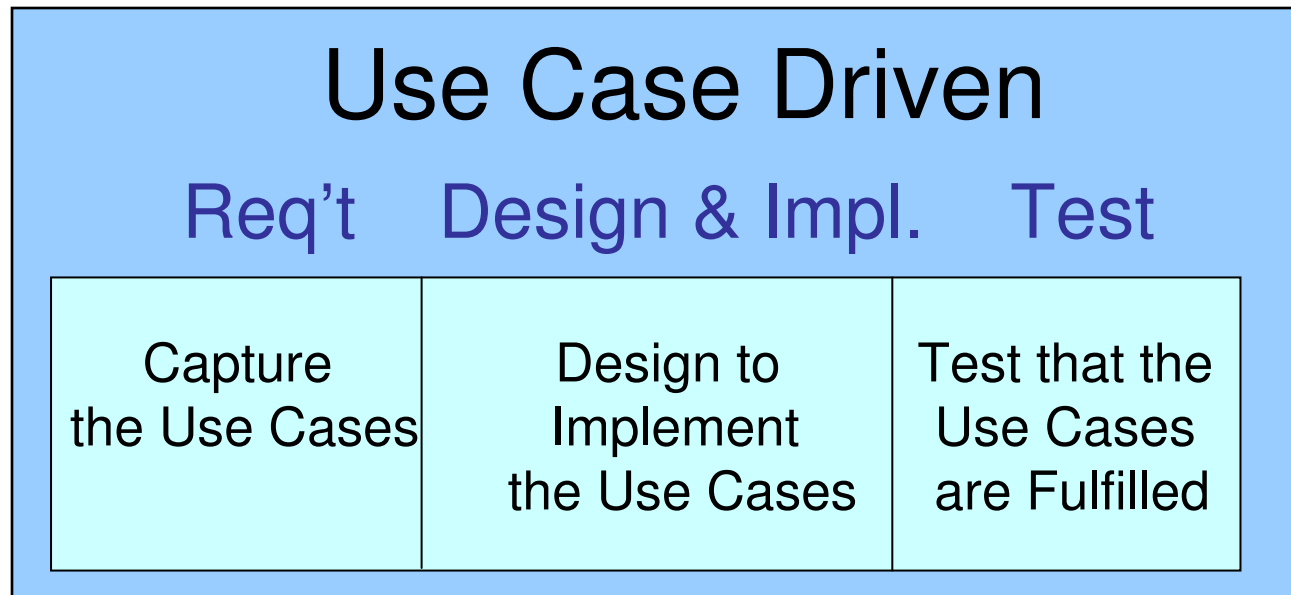


Fundamentally software development is use case driven

Any product development should follow three steps:

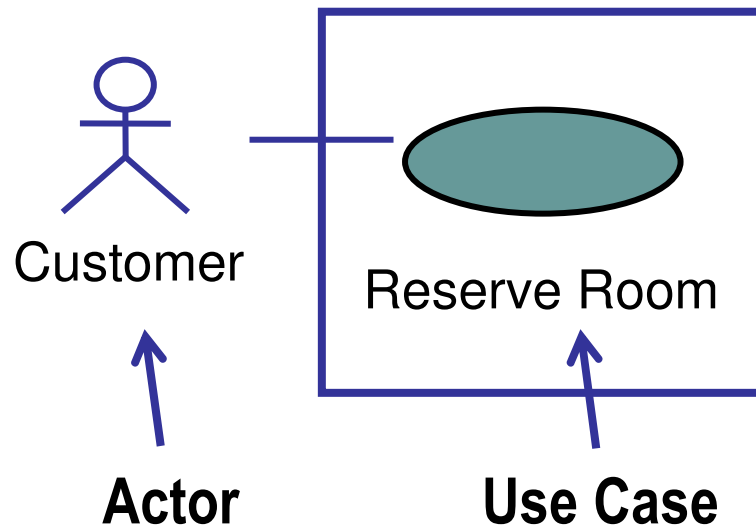
1. Capture the stakeholders' concerns
2. Design to fit those concerns
3. Test that the concerns are fulfilled

} stakeholders' concerns are modeled as use-cases



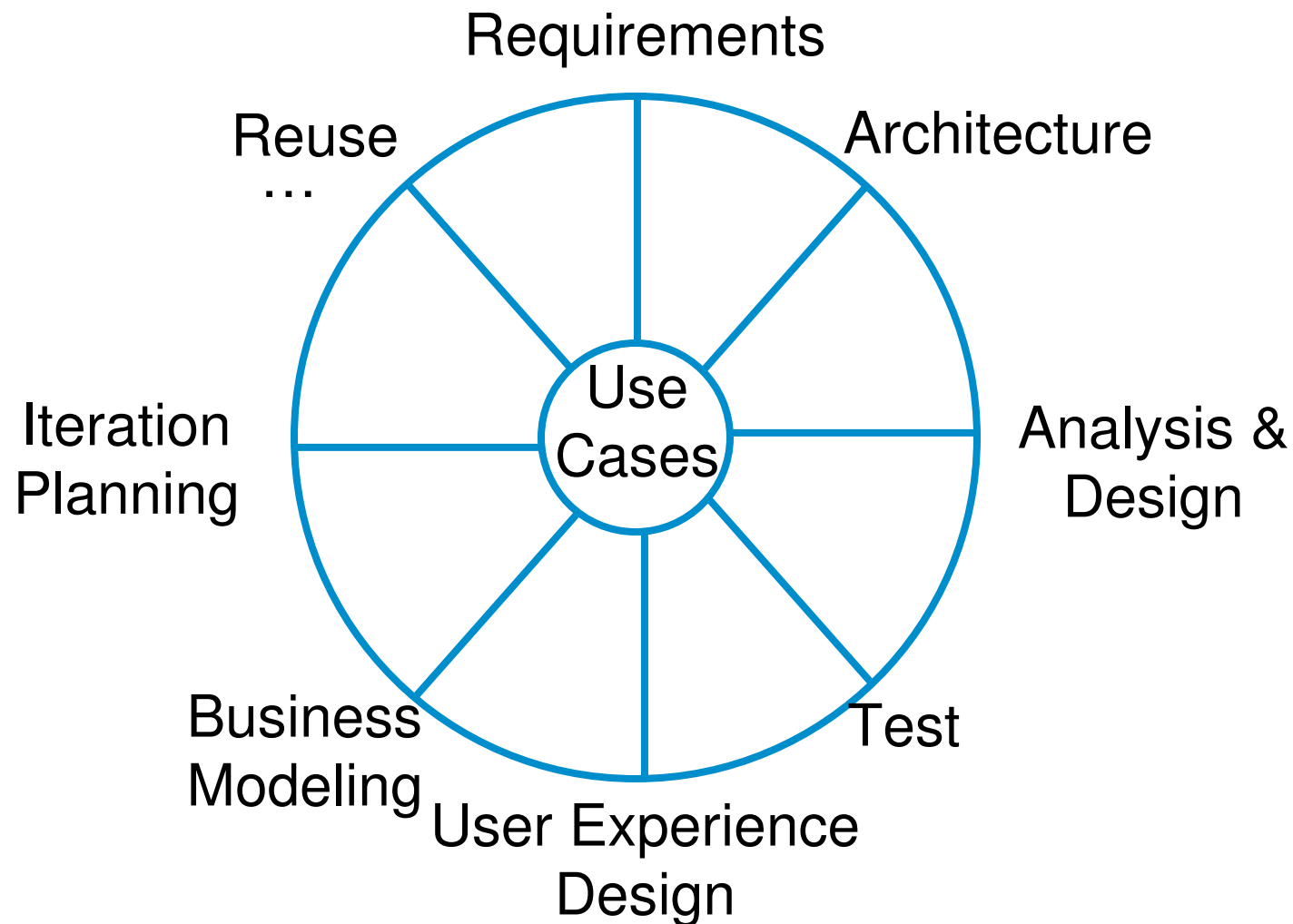
What is a use case?

- A use case is a sequence of actions a system performs that yields an observable result of value to a particular actor



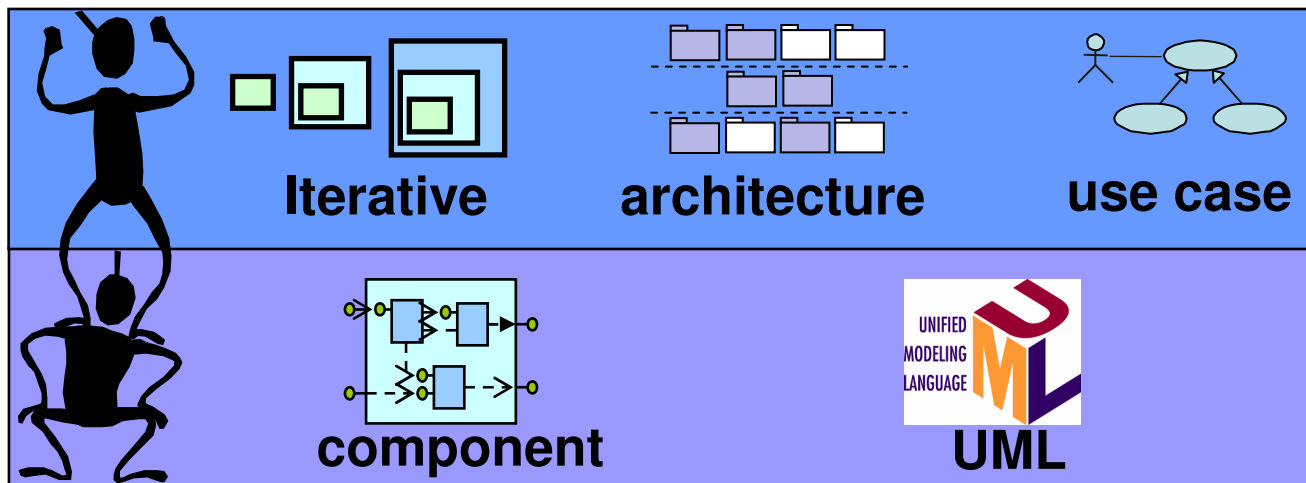
- Use case specifications versus traditional feature spec's?
 - Focus on end user concerns
 - Focus on acceptance criteria
 - Focus on incremental software development

The role of use-cases



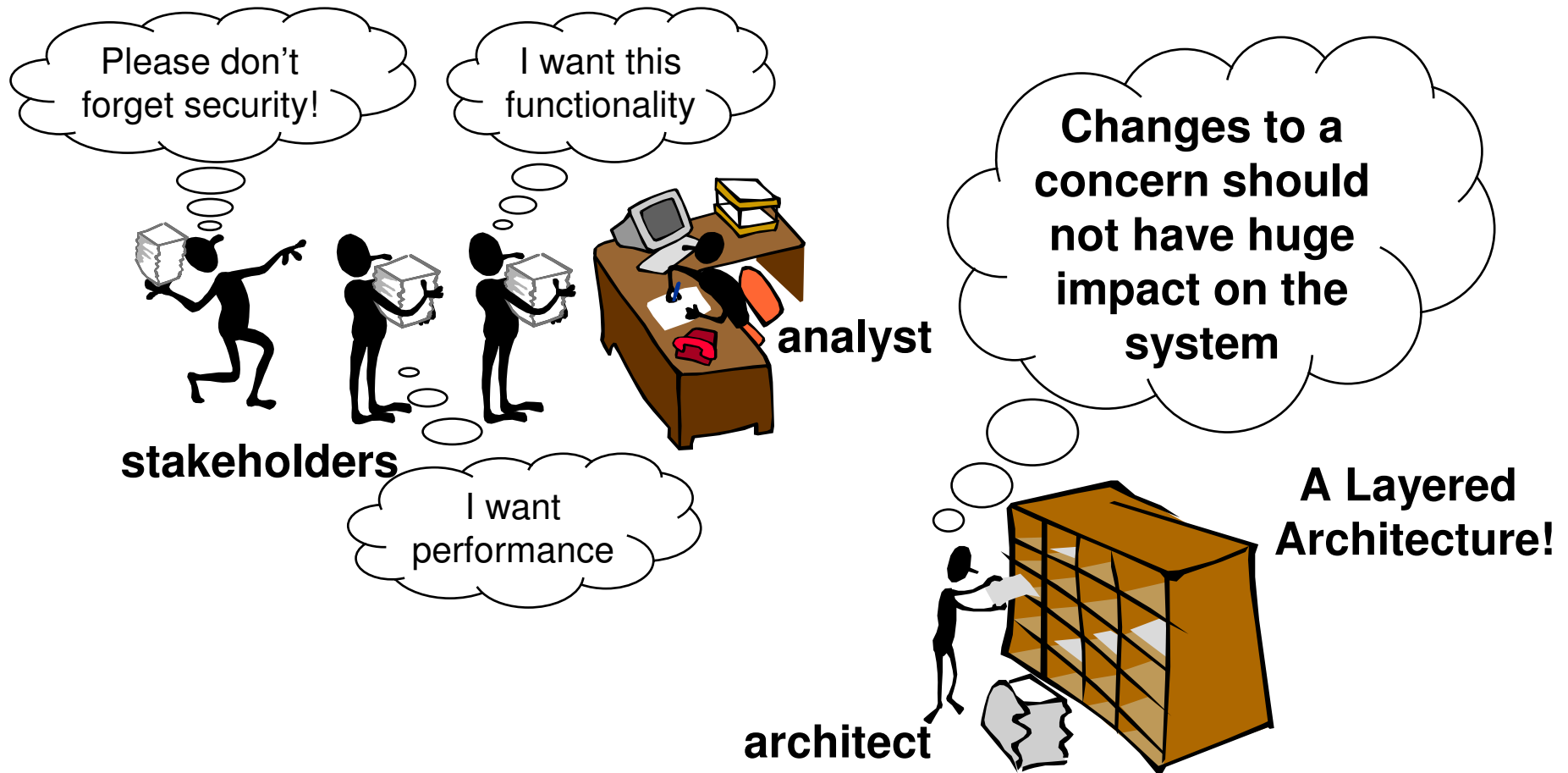
Core Best Practices

- The most important practices are
 - Iterative development
 - Use cases driven development
 - Architecture-centric



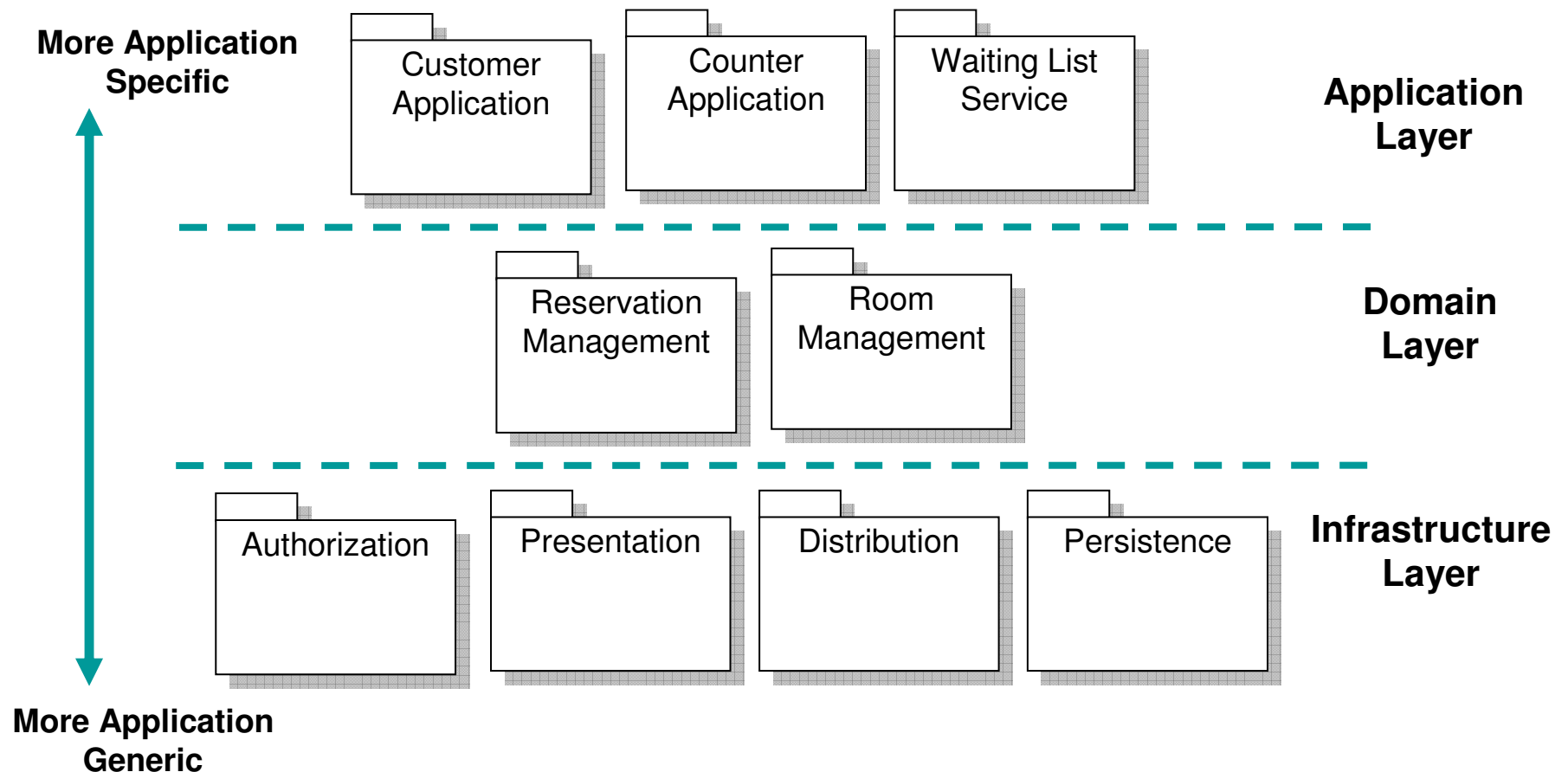
A good architecture keeps concern separate

- A system need to balance many different kinds of concerns



A layered architecture separate concerns

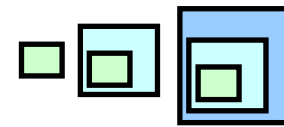
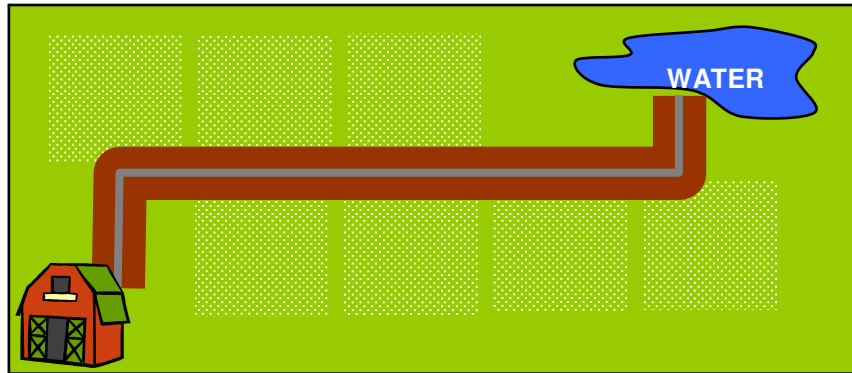
- Layers facilitate identification components
- Layers facilitate organization of resources
- Layers facilitate systematic reuse



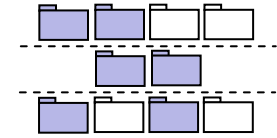
Agenda

- What CMM/CMMI means
- What Software Development Maturity Really Means
- Developing Really Good Software
- Achieving Real Process Improvement

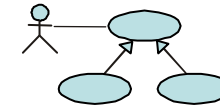
Recap: We want a better route to better software



Iterative



architecture



use case

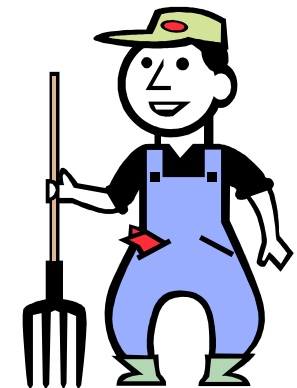
Criteria for Good Process

- Focus on **working software**
- **Early** exposure and resolution of risks
- Quality **throughout** the project
- **Seamless** transition from requirements to code
- **Robust** to changes

There is a Good Process

It is

- Iterative
- Use case driven
- Architecture centric



Build On A Strong Foundation

Project Management

- Project Planning
- Project Monitoring and Control
- Supplier Agreement Management
- Integrated Project Management(IPP)
- Integrated Supplier Management (SS)
- Integrated Teaming (IPP)
- Risk Management
- Quantitative Project Management

Support

- Configuration Management
- Process and Product Quality Assurance
- Measurement and Analysis
- Causal Analysis and Resolution
- Decision Analysis and Resolution
- Organizational Environment for Integration (IPP)

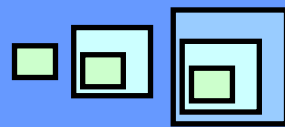
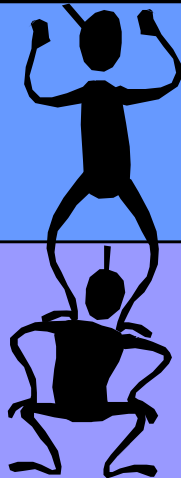
Engineering

- Requirements Management
- Requirements Development
- Technical Solution
- Product Integration
- Verification
- Validation

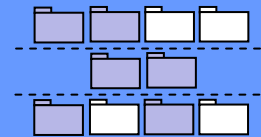
CMMI Process Areas

Process Management

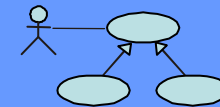
- Organizational Process Focus
- Organizational Process Definition
- Organizational Training
- Organizational Process Performance
- Organizational Innovation and Deployment



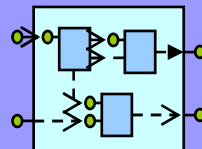
Iterative



architecture



use case



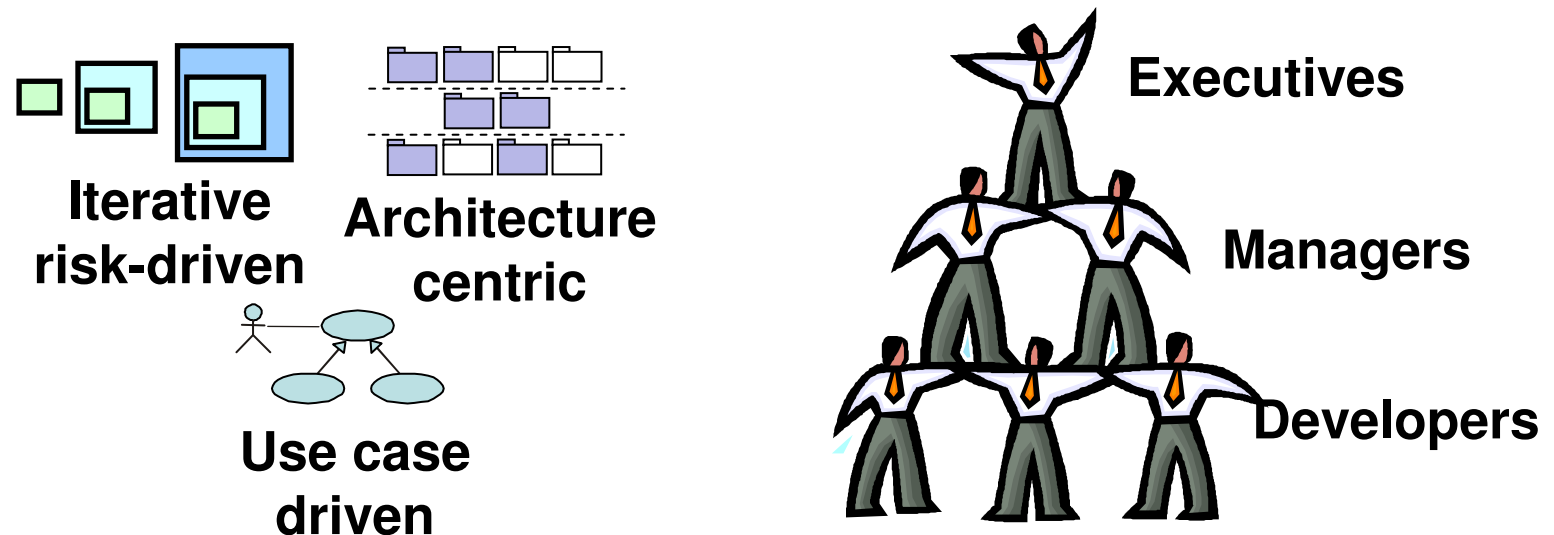
component



uml

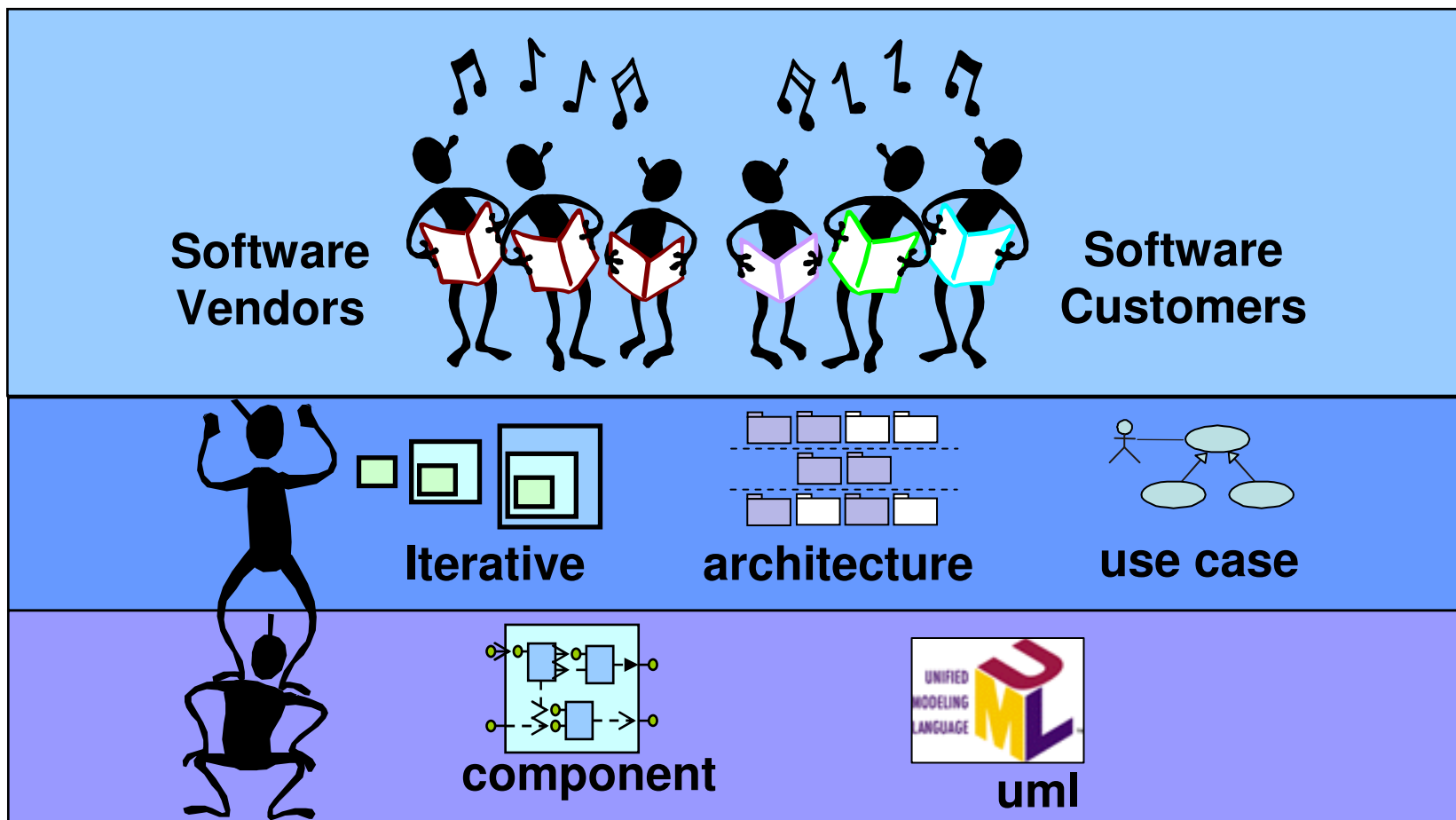
Real Software Improvement Is Hard!

- While ideas to iterative development, use case driven and architecture centric approach is simple
 - Many misinterpretation and poor formulation
- Software improvement is organizational and cultural change!
 - Affects developers, managers, executives
 - Be adequately prepared



True Software Maturity Across The Industry

- Everyone – both software vendors and software customers – use a common approach to develop good software!



Final Remarks

- We can describe and measure a bad process
- But only a good process is worth describing and measuring
- Adopt the good process
- Then you can describe and measure it

**It is easy to make a good process measurable,
but it is hard to make a measurable process good**

We will help you do both

Thank you

- Thank You

References

- The UML Books (Booch, Jacobson, Rumbaugh with Addison Wesley)
 - The UML User Guide
 - The UML Reference Manual
 - The Unified Software Development Process
- The Rational Unified Process
 - The Rational Unified Process - An Introduction
Philippe Kruchten (Addison Wesley)

References cont'd

- On Use Cases
 - Object-Oriented Software Development--A Use Case Driven Approach (Addison Wesley)
Jacobson et al, Addison Wesley Longman (1992)
- On Business Modeling
 - The Object Advantage: Business Process Reengineering with Objects (Addison Wesley)
Jacobson et al, Addison Wesley Longman (1994)

References cont'd

- On Systems of Systems
 - Software Reuse: Architecture, Process and Organization for Business Success (Addison Wesley)
Ivar Jacobson, Martin Griss & Patrik Jonsson, Addison Wesley Longman (1997)
- On Aspects
 - Aspect Oriented Software Development with Use Cases (Addison Wesley)
Ivar Jacobson, Pan-Wei Ng, Addison Wesley (2004)

Other References

- Karl E. Wiegers, Misconceptions of the Capability Maturity Model
 - <http://www.processimpact.com/articles/miscon.html>