An Architect's Point of View on TSP

TSP Symposium 2011

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

Felix Bachmann
09/2011

Software Engineering Institute | Carnegie Mellon

An Architect's Point of View on TSP

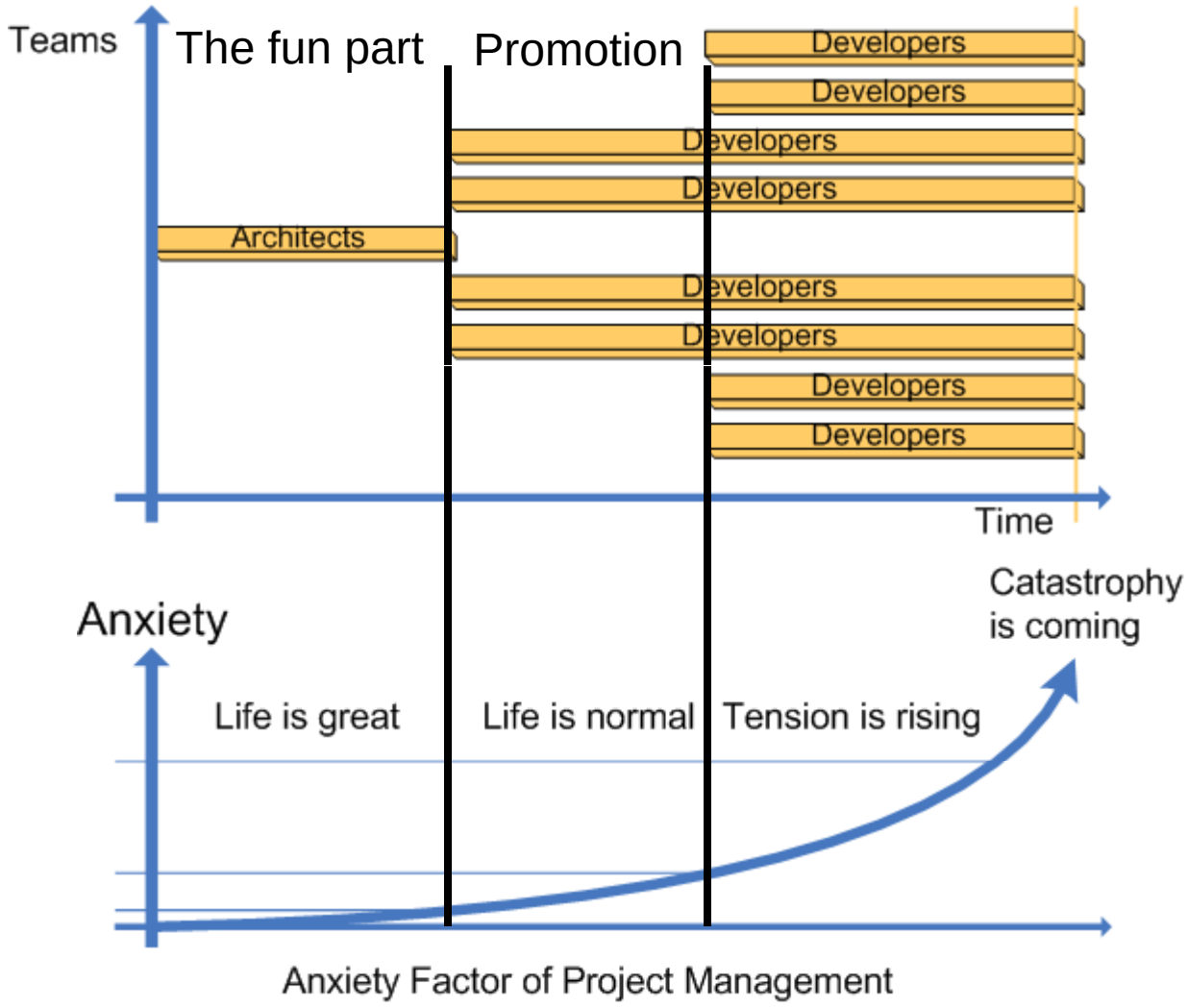
The Good Old Times

Felix Bachmann

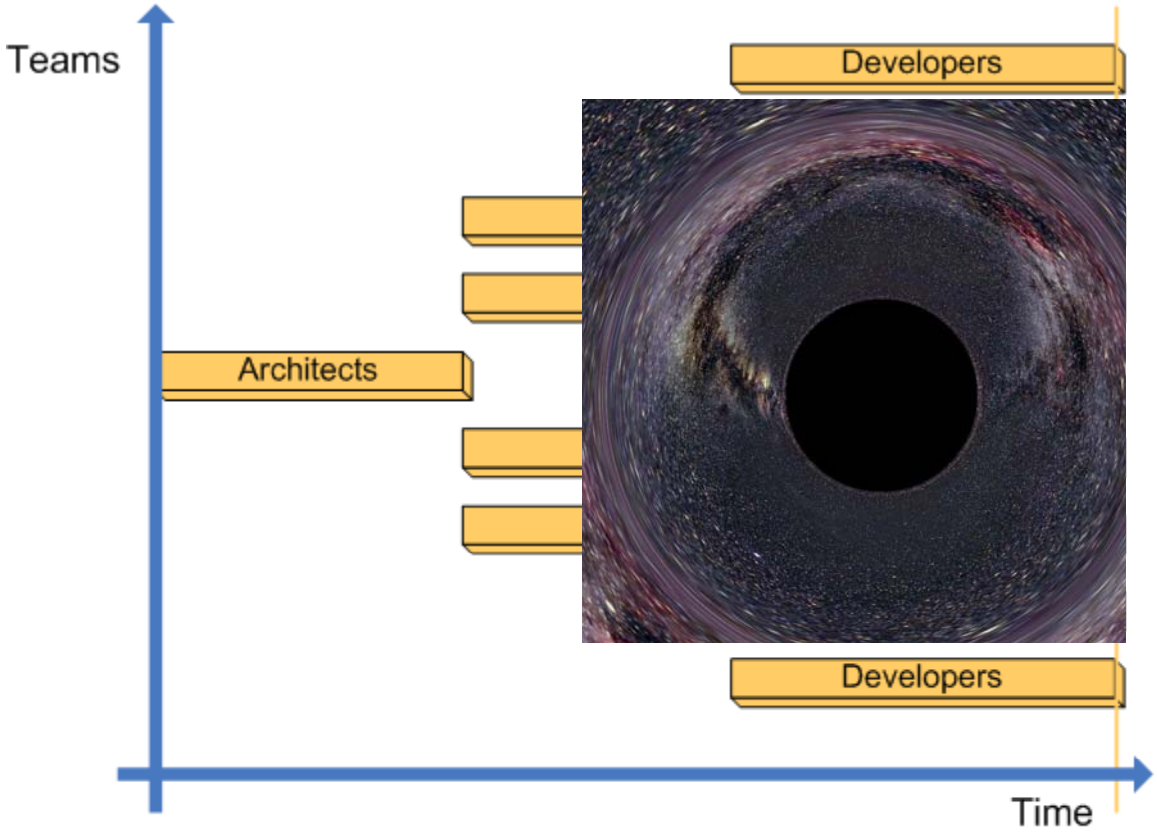
Software Engineering Institute

September 2011

… from the Architect's Perspective



… from the Architect's Perspective

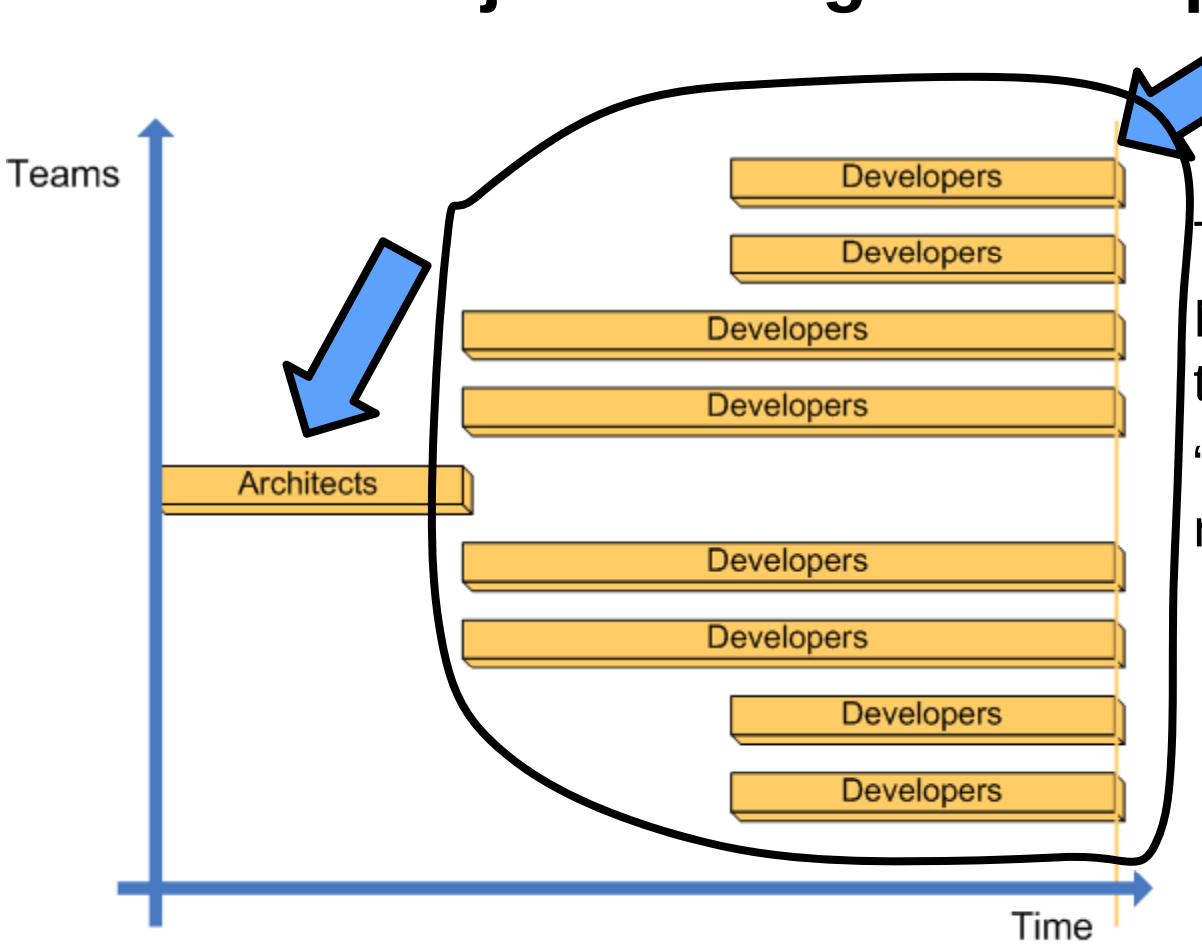


… I described everything necessary.

Why should I care!

If **they** don't listen, their fault!

… from the Project Manager's Perspective

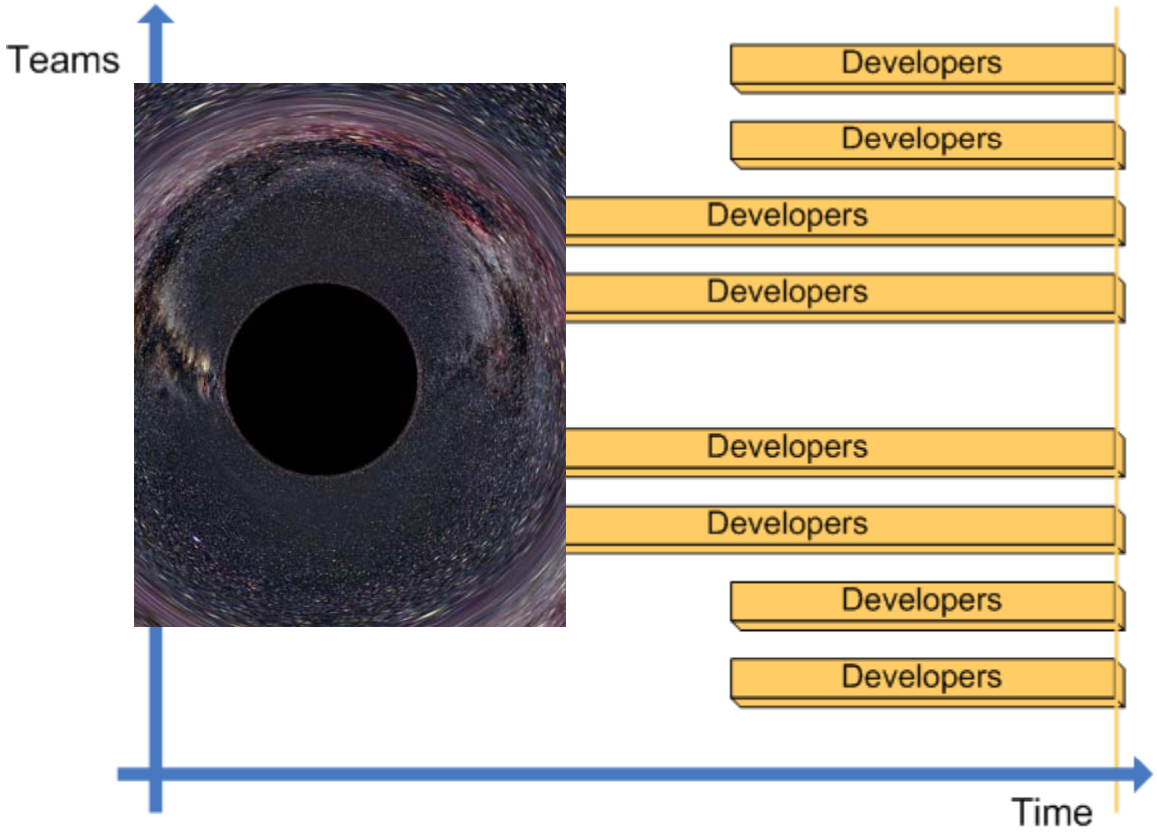


This is my deadline

Need to organize this

"They" told me I need this

… from the Project Manager's Perspective



I don't know what **they** do.

They better do some good work!

How does TSP Help?

Actually it does not.

TSP focuses on the development part – very little guidance for “design”

The team is responsible to deal with the "Black hole"

Therefore architecture work and development work stay separate

Everyone is happy!

… and the tourists can visit the Wall

Product Lifecycle

So, if everyone is happy, why would we change anything?

Developing a product means doing all the work, from requirement elicitation to deployment and keeping the system alive.

Most organizations look for guidelines for all the activities necessary during a product's lifecycle.

What they find are bits and pieces.

They find advice on how to do architecture and development, but they have to put it together and make it work!

The Message from the SEI

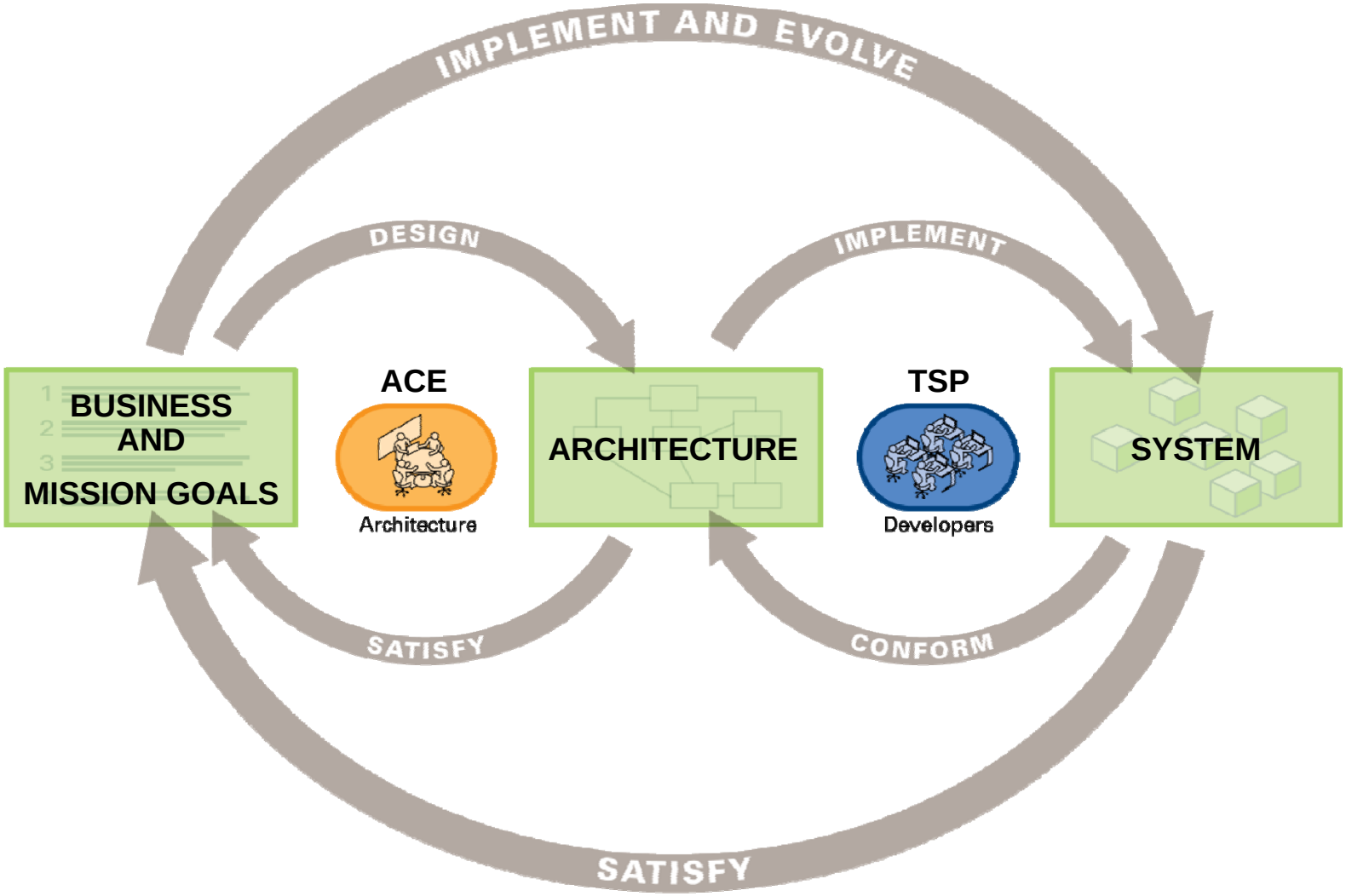
The *team software process* (TSP) has great success in enabling development teams to produce high quality software on time and within budget.

There is *architecture centric engineering* (ACE) to enable organizations to produce high quality system / software designs that fulfill the business needs.

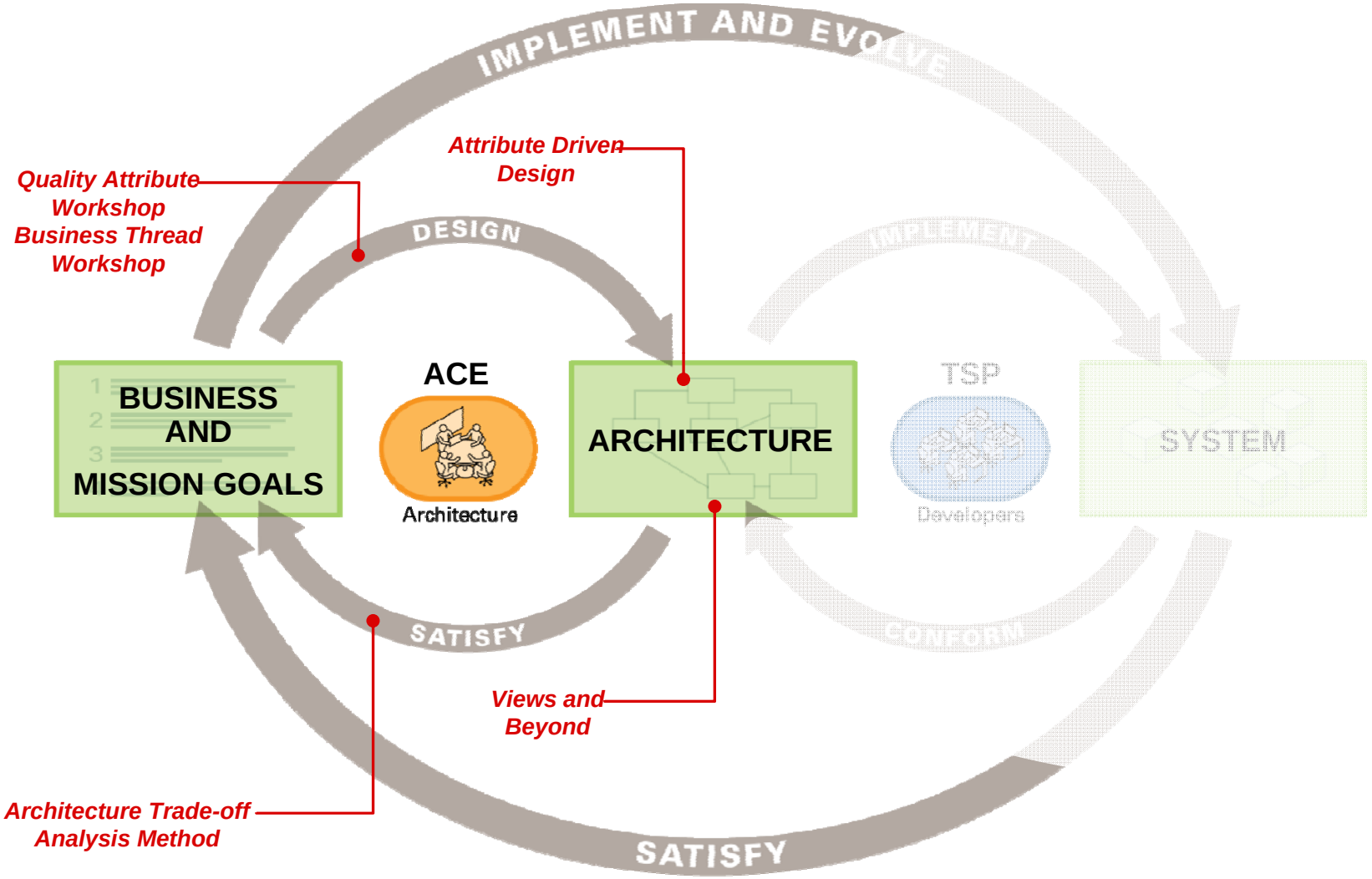
Combining those methods and techniques would address the needs of most organizations, but …

… does this provide enough benefits for the architects and the developers so that they are willing to embrace it?

Engineered for Success



The Design Cycle with ACE



TSP Contribution to ACE

Measurement Framework

Schedule

Effort

Size

Quality

The TSP measurement framework is made for implementation, not design.

Size measure is typically something like LOC or some “point” method like function points

Quality is typically measured in bugs.

This does not apply to design work!

ACE – Size Measure

Architecture design is driven by quality attribute requirements.

Those quality attribute requirements need to be specified as *quality attribute scenarios* (QAS)

This becomes the unit for the size measure

Apply the “Rule of Five” to estimate the size of architecting tasks

ACE – Size Measure / Rule of Five

Select the five most important quality attribute scenarios

Everyone of those will be refined into five more specific scenarios

Everyone of the specific scenarios will be described with five architecture diagrams

Assign one to five points to each diagram dependent on the system complexity

ACE – Effort Measure

Benchmark:

An average architecture team of four can create a review ready design of one QAS in a known domain within a week

The rework of a scenario after the first review can be done by the same team in ½ week

In a known domain the design of the system with average complexity can be done by a team of four educated architects in about 7-8 weeks.

Your effort will vary. So start collecting data.

ACE – Quality Measure

The quality of an architecture is determined in how well this current design will satisfy the stakeholder needs.

The stakeholder needs are specified as *quality attribute scenarios* (QAS)

The “distance” of the current design to the QAS is measured in Risks

Use scenario-based peer-reviews to uncover the risks.

In these type of reviews the defined QAS are the test cases to be used.

Running the test case means to convince a quality attribute expert that the design is satisfactory.

ACE – Schedule Measure

Creating a schedule for architecting is not much different from creating a schedule for any other work.

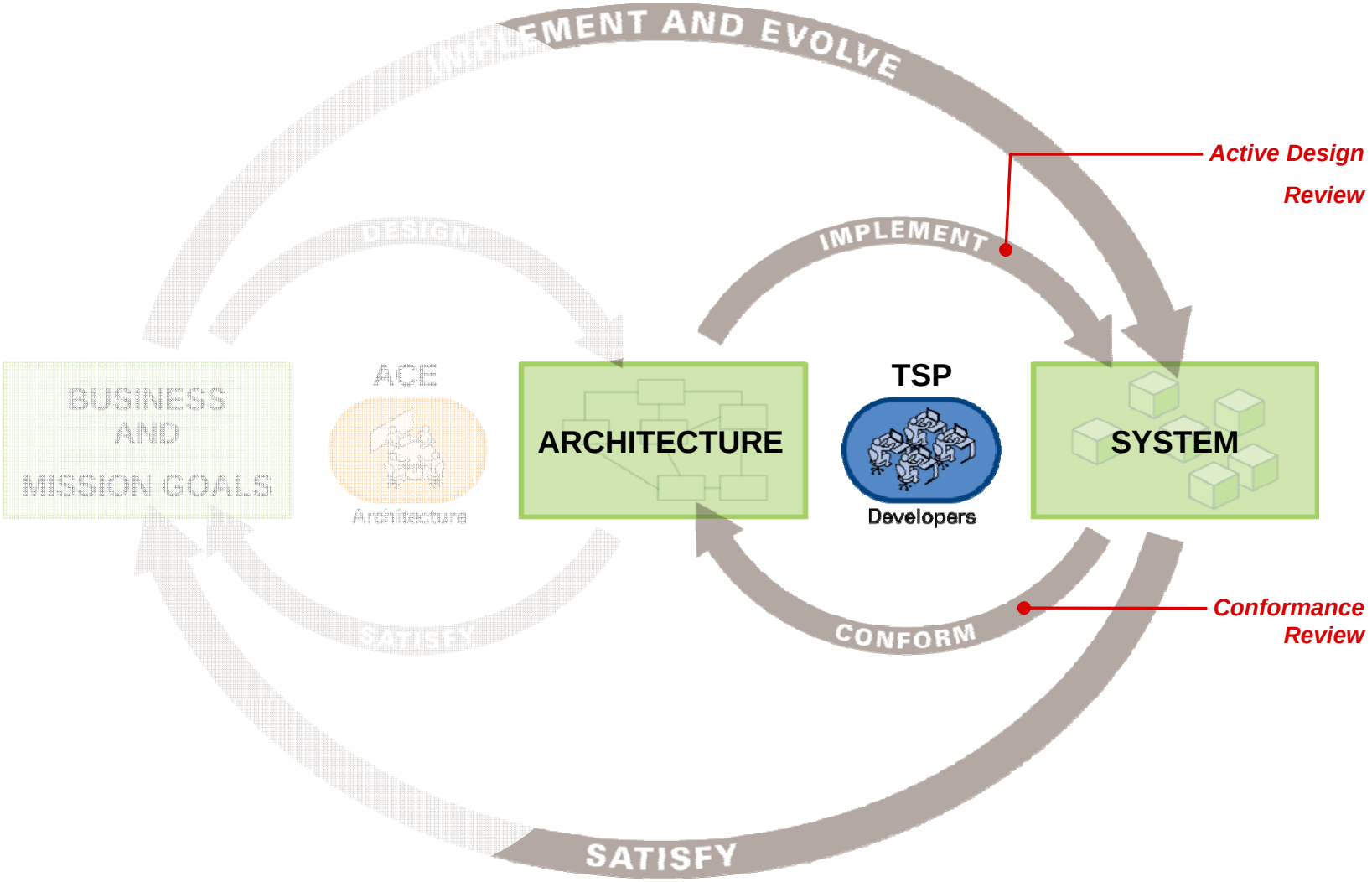
Most architecture tasks must be done as a team together. Splitting up the work for single team members does not work in most cases.

Benefits for Architect?

Why would an architect care?

- The method provides focus, which avoids many unproductive discussions.
- For the first time the project manager and other stakeholders understand and value the work of an architect
- Avoids frustration because developers understand what needs to be done

The Implementation Cycle



Design – Implementation Interface

Architecture design may or may not be refined into a detailed design

A detailed design is the blueprint for constructing the system

The detailed design can be done by:

- The architects
- The developers
- Both together (preferred method)

If done by the architects then an Active Design Review is necessary

If done by the developers a review that checks the detailed design against the architecture design is necessary

Detailed Design – Size Measure

Detailed design is driven by module specifications.

Use cases will be described to specify the interface and responsibilities of a module

A module is the unit for the size measure

Rule of five:

The detailed design of a module consists of five diagrams / descriptions:

- One module diagram
- Four sequence diagrams describing how use cases use this module

Detailed Design – Effort Measure

Benchmark:

A detailed design for an average module can be done by one person in a day.

Some coordination / learning effort for understanding the architecture is required.

After that, a detailed design of a module can be done by one person.

Detailed Design – Quality Measure

The quality of a detailed design is determined by its conformance to the architecture design.

Violations of the architecture are “bugs” that need to be fixed

Peer-reviews with architects are used to uncover mismatches.

Benefits for Developer?

Why would a developer care?

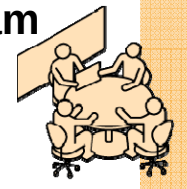
- Many tough and unpleasant decisions are already made.
- The risk for annoying rework is minimized.
- Impact analysis when requirements change is easier and more precise.
- The overall productivity is increased, which makes the development team shine!

TSP-ACE – Parallel Development

Stakeholders

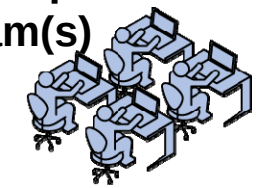
Requirements

Architecture Team



Requirements do not have to be complete for the architects to design the system

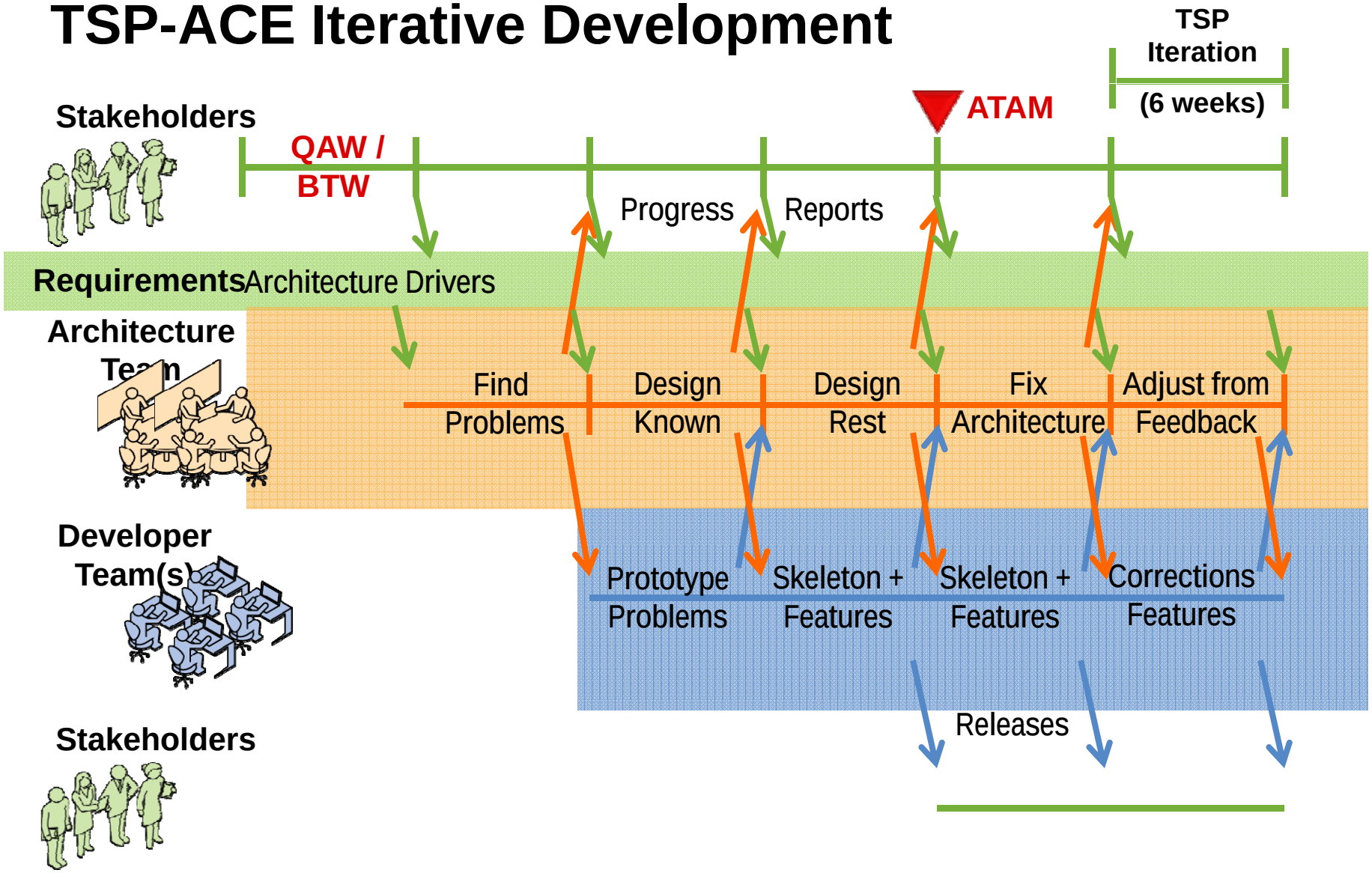
Developer Team(s)



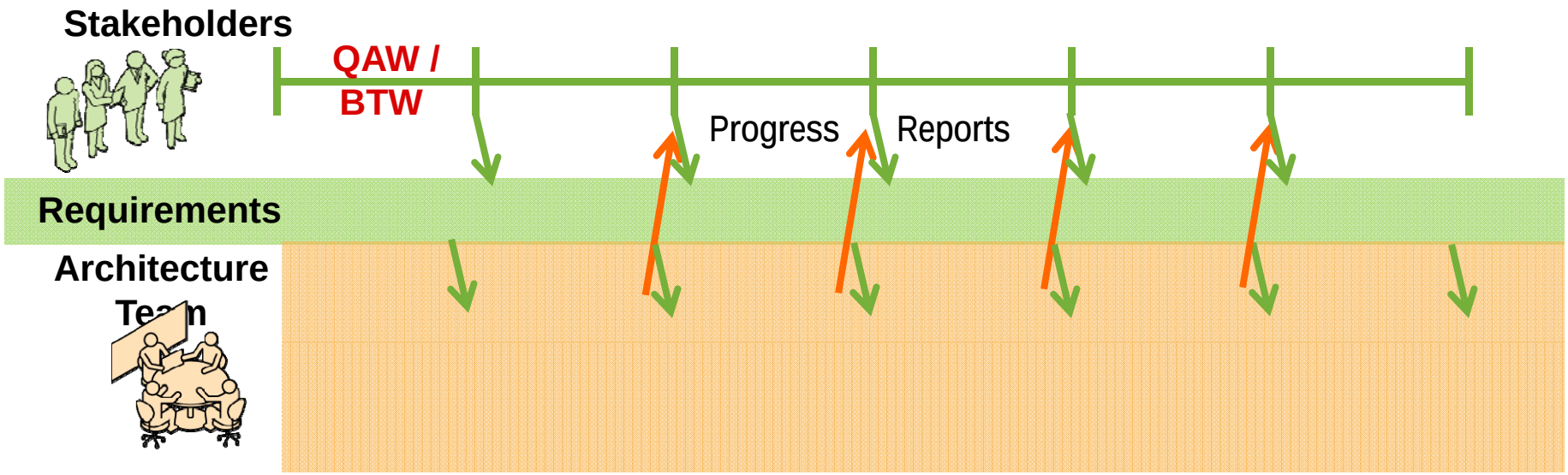
Architecture does not have to be complete for the developers to start implementing

Time

TSP-ACE Iterative Development



Stakeholder Information Exchange



New or changed requirements

Progress report

Stakeholder Information Exchange – Design

New or changed requirements can be provided to the development team at any time

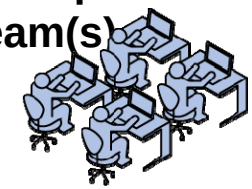
- Only the architecture driver must be know at the start of the development
- Quality attribute requirements must be specified as quality attribute scenarios

Progress report to the stakeholders via results of peer reviews for quality attribute scenarios

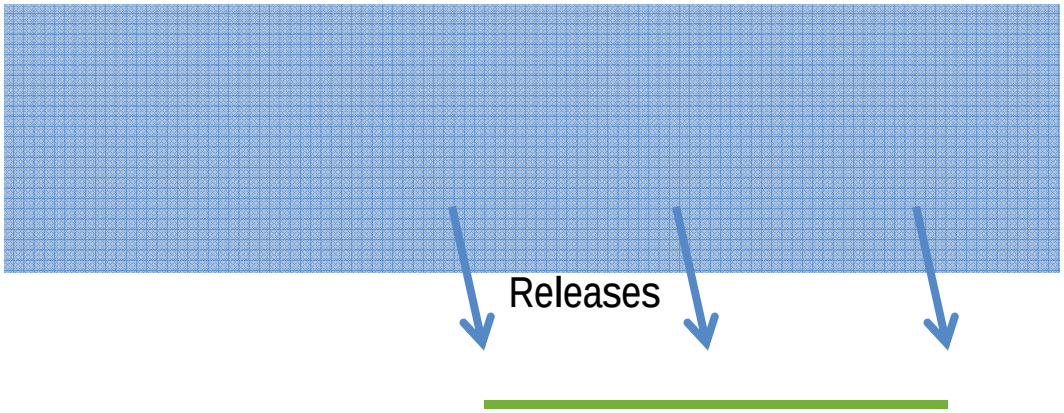
- Peer review result contain risks and action items assigned to a quality attribute scenario
  - Red: Need to repeat the review
  - Yellow: Risks are mitigated but open action items
  - Green: Risks are mitigated and no action items

TSP-ACE Iterative Development

Developer Team(s)



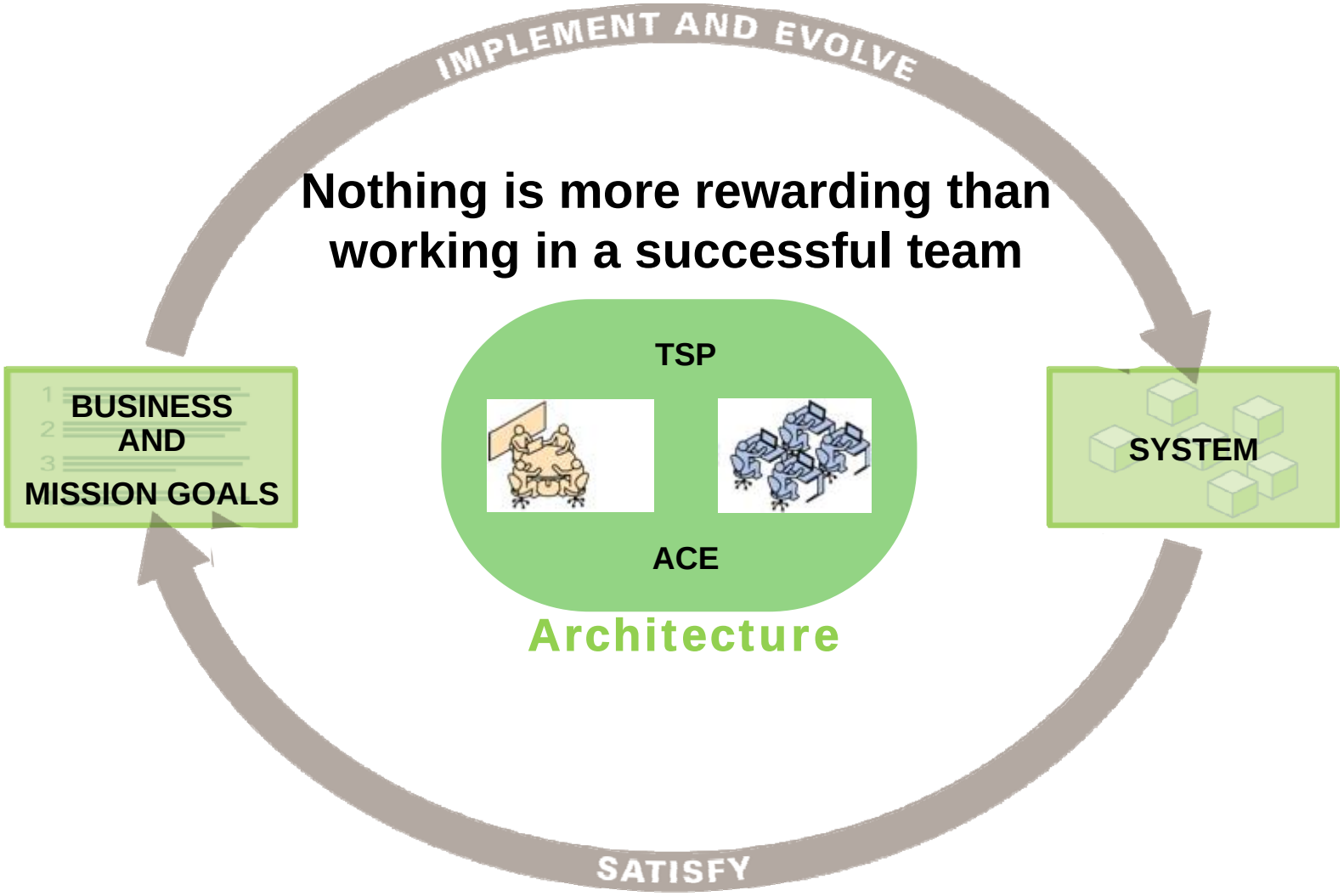
Stakeholders



Every iteration some running code can be shown to the stakeholders.

- The first release is a “skeleton system” with one little function.
- Next releases add functions to the skeleton system.

ACE and TSP – Engineered for Success



Questions?