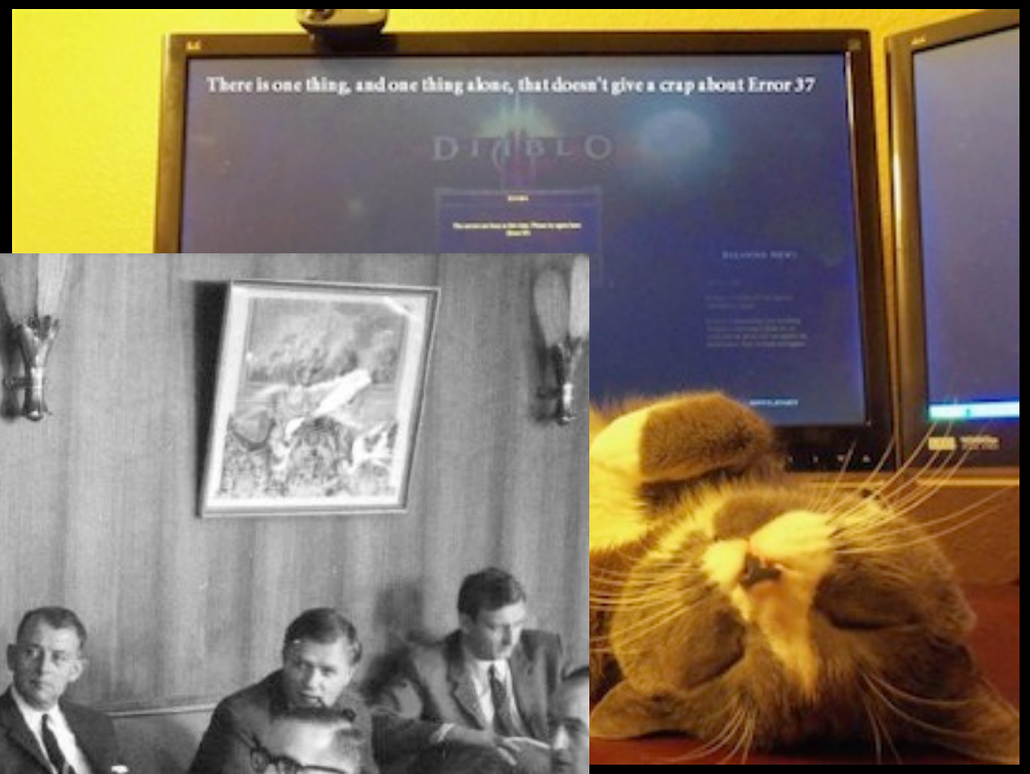
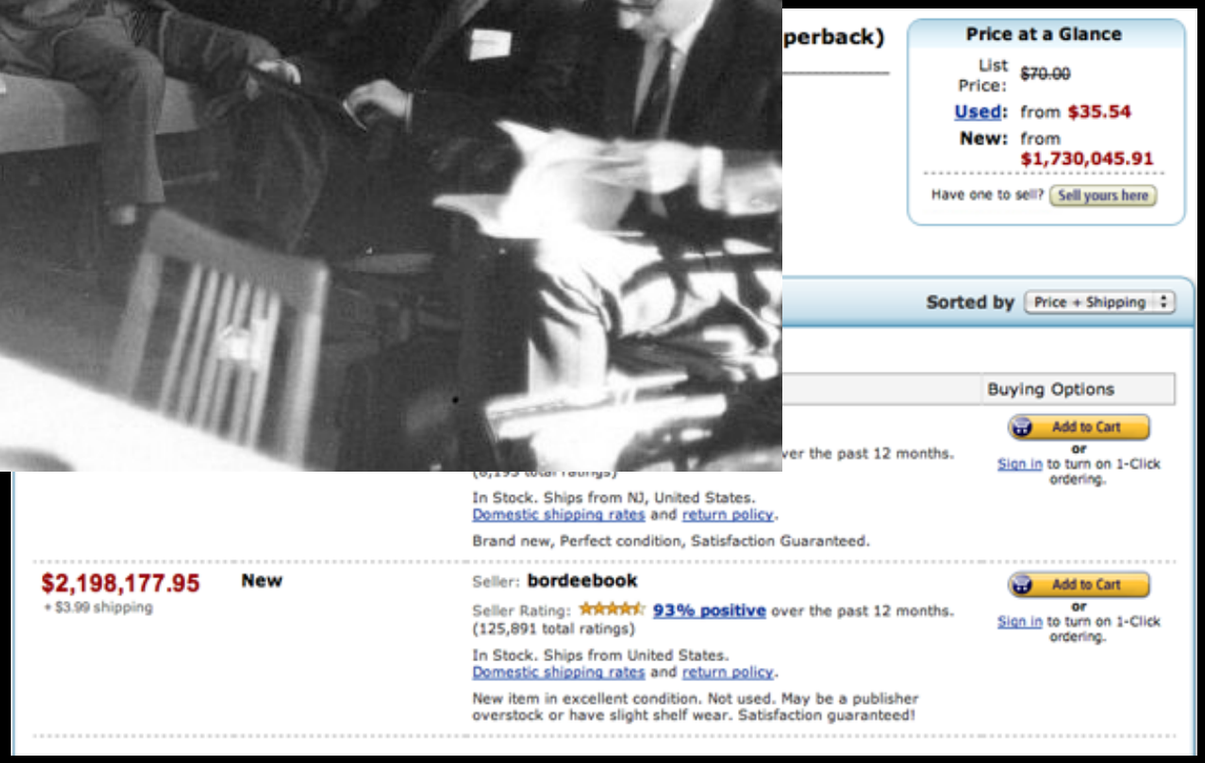
Software Engineering Project

Morgan Ericsson

morgan@cse.gu.se
@morganericsson
morganericsson
morganericsson

UNIVERSITY OF GOTHENBURG
CHALMERS





Software Engineering: Report of a conference sponsored by the NATO Science Committee, Garmisch, Germany, 7-11 Oct. 1968

Software Crisis and Engineering

- Established as a reaction to the "Software Crisis"
  - software was inefficient
  - software did not meet requirements
  - projects ran over time/budget
  - projects were unmanageable and software unmaintainable

"The major cause of the software crisis is that the machines have become several orders of magnitude more powerful! To put it quite bluntly: as long as there were no machines, programming was no problem at all; when we had a few weak computers, programming became a mild problem, and now we have gigantic computers, programming has become an equally gigantic problem."

Software Engineering

- The branch of computer science that creates practical, cost-effective solutions to computing and information processing problems
- *"Application of engineering to software"*
  - systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software
- Assure the quality of the process and the product

“Engineering Seeks Quality”

- So, the goal of software engineering is the production of quality software
- However, software is inherently complex
  - the complexity of software systems often exceeds the human intellectual capacity
- The task of the software development team is to engineer the illusion of simplicity

Three Steps Revisited

Requirements

1. What should I do?

Implementation

2. Doing it!

Quality Assurance

3. Did I actually do it?

A sequential model of software development

Five Steps

Requirements

Design

Implementation

Quality Assurance

Evolution

A sequential model of software development

The Five Steps/Phases

- Requirements
  - understanding the problem domain
  - communication between stakeholders
- Design
  - engineer a solution that addresses the requirements
  - technology, algorithms, architecture, interfaces...

The Five Steps/Phases

- Implementation
  - realizing the design
    - documentation, configuration, standards, tools, ...
- Quality Assurance
  - ensuring that the implementation meets quality standards
  - testing, analysis, reviews

The Five Steps/Phases

- Evolution
  - fixing problems
  - adding new functionality/address new requirements
  - while retaining existing functionality and code

Software Engineering

Requirements

Architecture

Coding and Detailed Design

Testing

Evolution

Change is Ubiquitous

- Manage
- change
- uncertainty
- quality

Change is Ubiquitous

Design

Requirements

Implementation

Quality Assurance

An iterative model of software development

Defined Process vs. Empirical Process

- Laying out a process that repeatedly will produce acceptable quality output is called defined process control
- When defined process control cannot be achieved because of the complexity of the intermediate activities, something called empirical process control has to be employed

Production vs. Creation

Defined Process control

- Planning and typically pre-study heavy
- Assumes (more) static environment
- Longer iterations
- Change management intensive
- Assumes good estimations
- Control over actual work (seen seen as bureaucratic)

Empirical Process control

- Change is reality
- Shorter iterations
- Problem vs. solution space (empowering the developers)
- Just enough (management, documentation, etc.)
- Self organizing teams
- Continuous “customer” interaction
- **NOT UNPLANNED, rather adaptive!!!**

In reality/practice

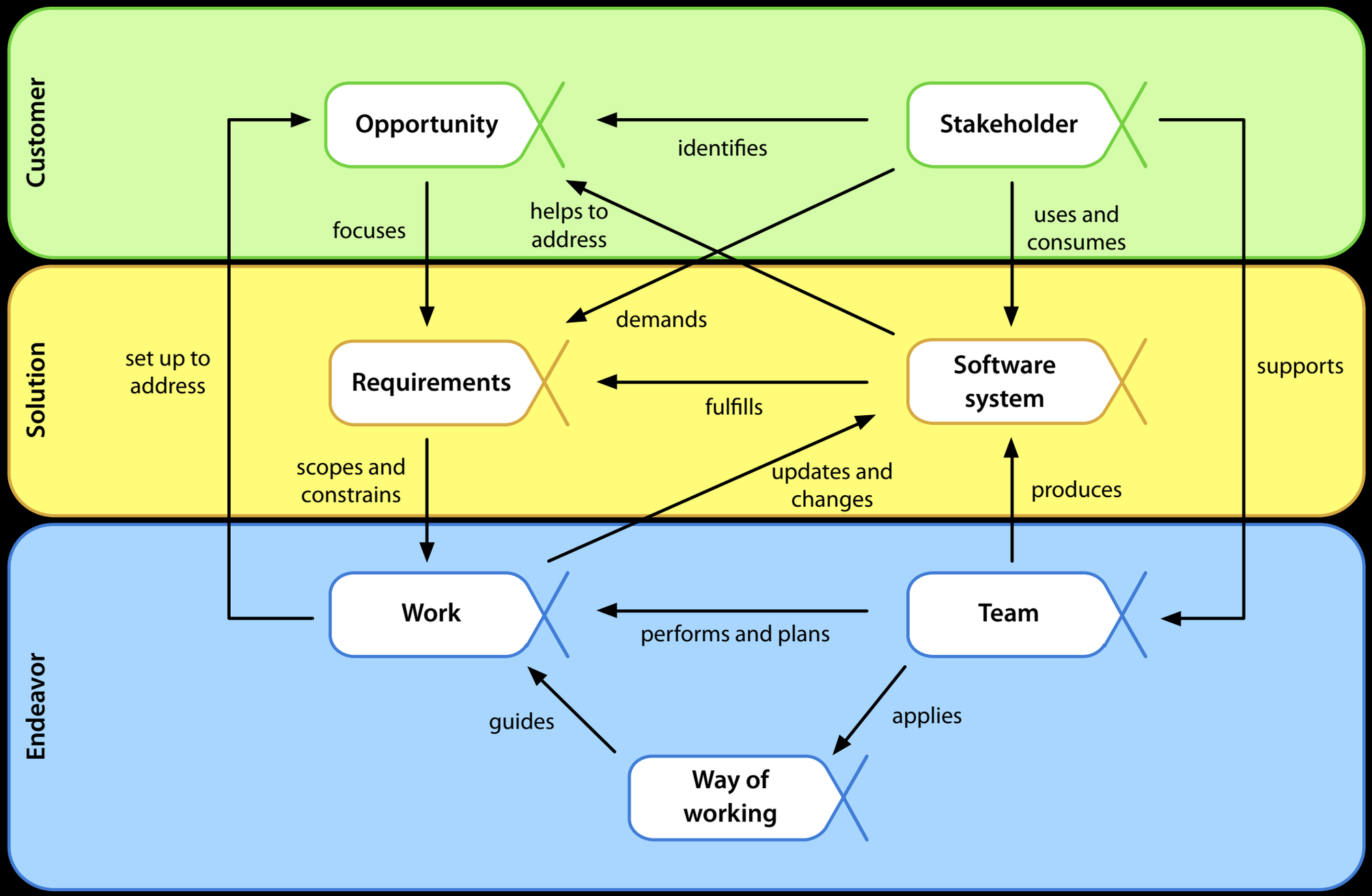
- Many different
  - processes
  - methodologies
  - practices
  - …

SEMAT

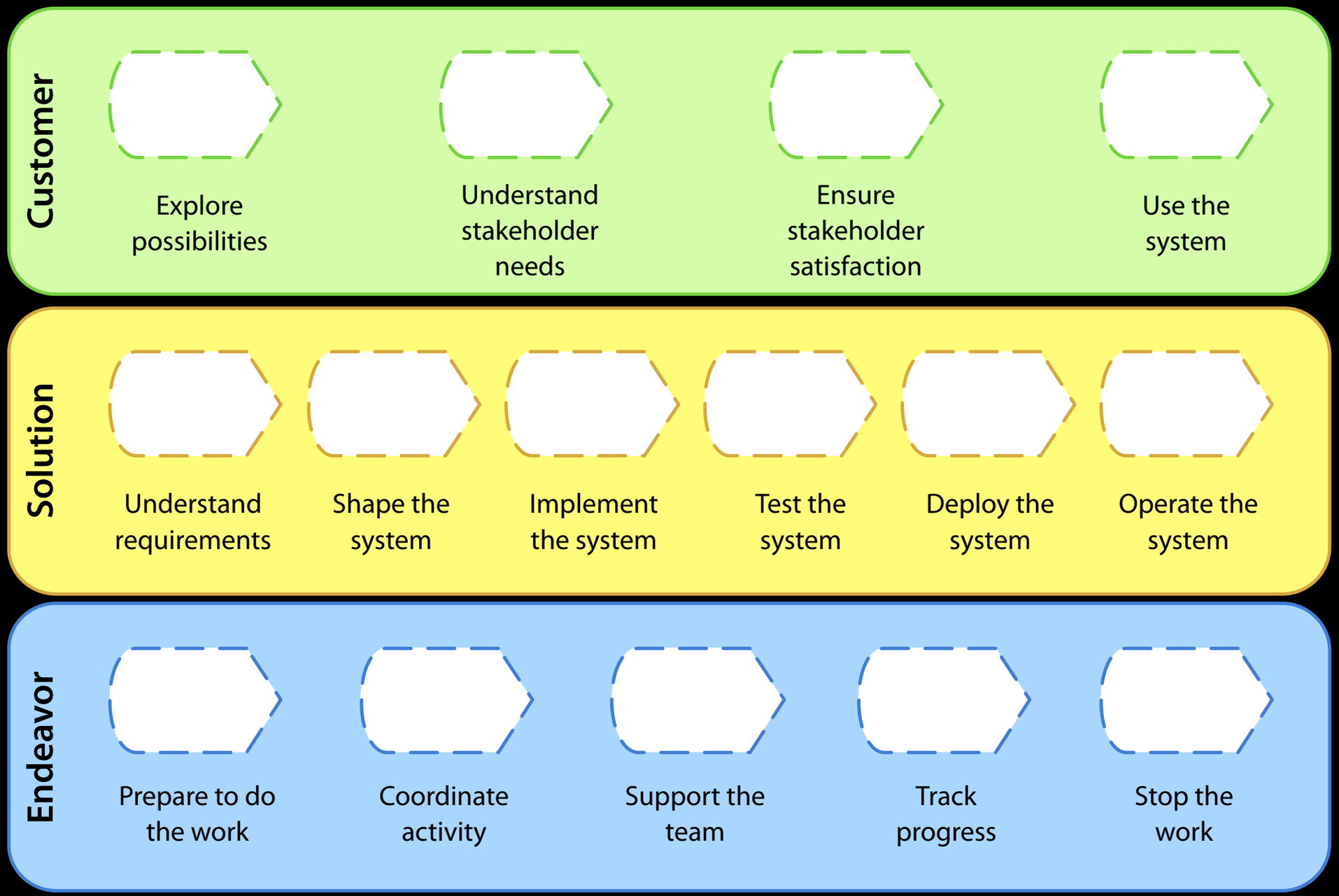
Software Engineering Method And Theory

http://semat.org

Things to work with



Things to do


Customer
Explore possibilities
Understand stakeholder needs
Ensure stakeholder satisfaction
Use the system
Solution
Understand requirements
Shape the system
Implement the system
Test the system
Deploy the system
Operate the system
Endeavor
Prepare to do the work
Coordinate activity
Support the team
Track progress
Stop the work

Fred: *"The requirements go in different directions, so our work is not converging on anything that can be released."*

Eric: *"The proposed software architecture doesn't seem to solve the right problems, but I don't know how to drive it forward."*

What can we do to help them?

Example from *"The Essence of Software Engineering: The SEMAT Kernel"*

Susan: *"Management is panicking and throwing more developers on the team, but we already have difficulty coordinating the existing team members."*

Steve: *"I have to spend a lot of time pretending I am following the mandated development process and producing documentation that isn't really useful for anyone."*

What can we do to help them?

Example from *"The Essence of Software Engineering: The SEMAT Kernel"*

Fred: *"The requirements go in different directions, so our work is not converging on anything that can be released."*

Eric: *"The proposed software architecture doesn't seem to solve the right problems, but I don't know how to drive it forward."*

What can we do to help them?

Example from *"The Essence of Software Engineering: The SEMAT Kernel"*

Susan: *"Management is panicking and throwing more developers on the team, but we already have difficulty coordinating the existing team members."*

Steve: *"I have to spend a lot of time pretending I am following the mandated development process and producing documentation that isn't really useful for anyone."*

What can we do to help them?

Example from *"The Essence of Software Engineering: The SEMAT Kernel"*

Getting to the Heart of the Problem

- The system should meet a business need or an opportunity
  - what is that need?
- The system must provide some value
  - what was that value?
- If the value is not understood, there is no point in debating detailed requirements

Getting to the Heart of the Problem (cont'd)

- Were the stakeholders engaged and thinking about requirements?
- Were the stakeholders in agreement?
- Do they have the required knowledge of the system and its users?

Getting to the Heart of the Problem (cont'd)

- Did Fred have a clear understanding of
  - scope?
  - requirements?
  - risks?
- Was he focusing on requirements in the right order?
- Was he trying to get too many requirements nailed down up front?

Getting to the Heart of the Problem (cont'd)

- Do stakeholders have a good understanding of the existing software system?
  - capabilities
  - limitations
- Is the system (and architecture) robust in the face of change?

Getting to the Heart of the Problem (cont'd)

- Do the team have the competence to get the job done?
- Is the team working together as a unit?
- Are there conflicts waiting to surface?

Getting to the Heart of the Problem (cont'd)

- Were the problems due to
  - a need to find work for all the team members?
    - the way the work had been planned?
  - Was the team working on the wrong things because of a previous planning error?

Getting to the Heart of the Problem (cont'd)

- Were the problems caused by the method?
  - requirements diverged because it insisted that all requirements were detailed before design and implementation?

Addressing the Challenge

Explore Possibilities. The opportunity is the reason for the production of the new, or changed, software system. Fred and Angela had to understand the need for the Value Added Services and make sure that for this opportunity there was clear value established.

Addressing the Challenge

- Explore Possibilities
  - opportunity is the reason for a new or changed software system.
- Understand the need for changes or features
- Make sure that value is established is for this opportunity

Addressing the Challenge (cont'd)

- Understand Requirements.
  - get the requirements bounded
    - then find the most important requirements to drive further clarification
- Walk through end-user scenarios

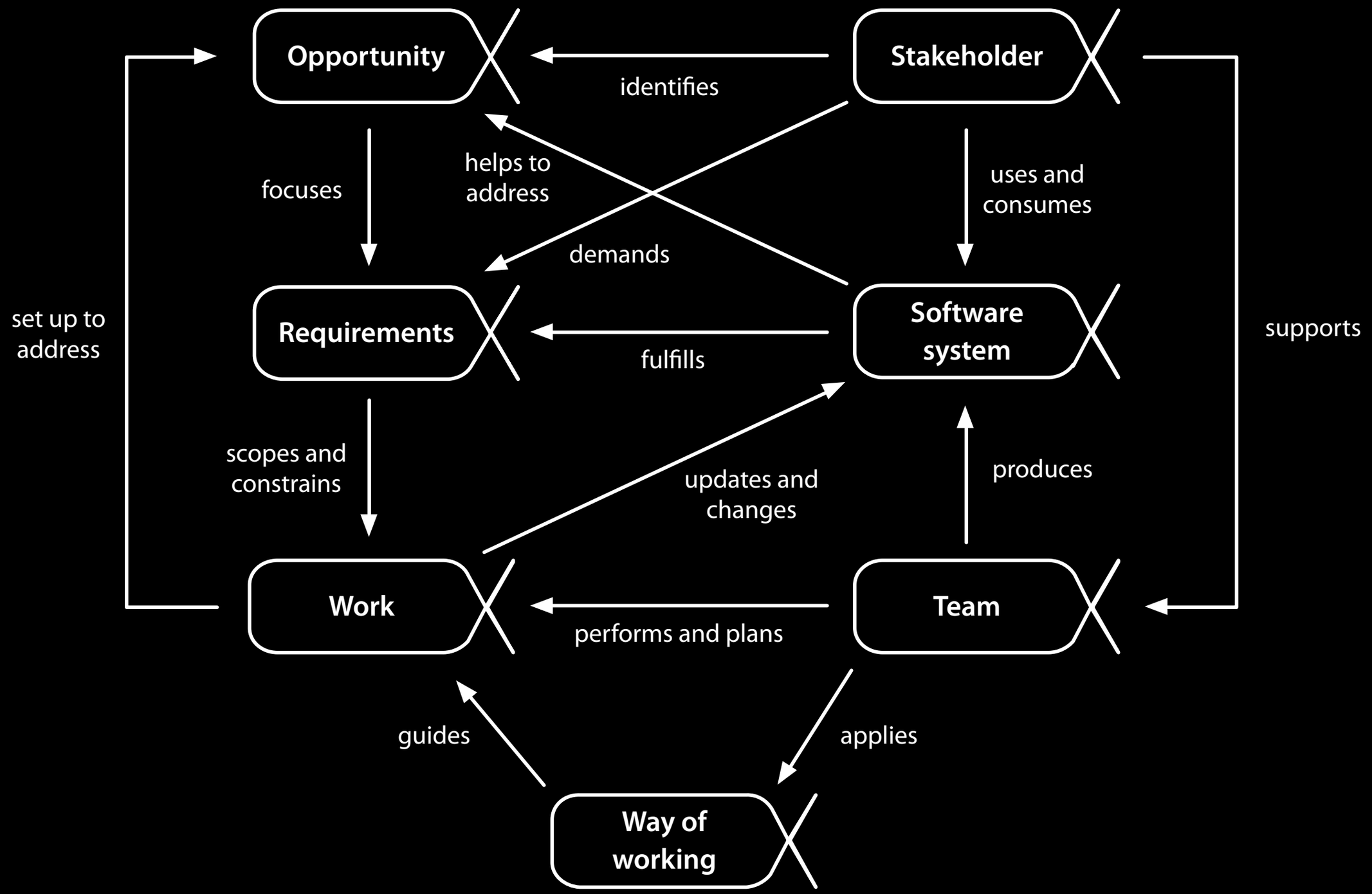
Addressing the Challenge (cont'd)

- Shape the System
  - implement the software system
  - make it demonstrable
- Confirm with stakeholders that the system is on the right track

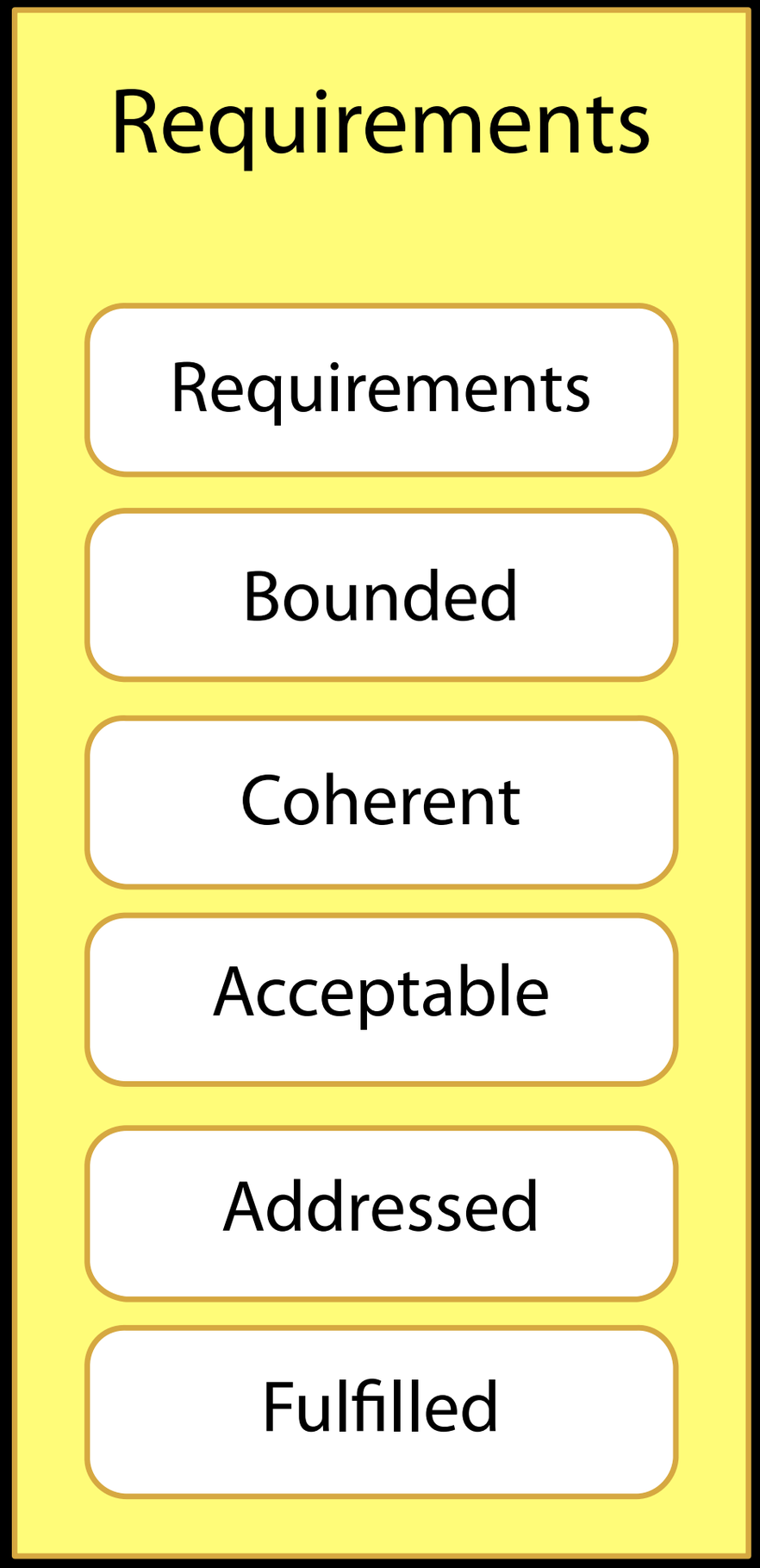
Introducing the Alphas

- Things whose evolution we want to
  - understand
  - monitor
  - direct
  - control
- Aspiration Led Progress and Health Attribute

Introducing the alphas



The Requirements alpha



Getting to the Heart of the Problem

- The system should meet a business need or an opportunity
  - what is that need?
- The system must provide some value
  - what was that value?
- If the value is not understood, there is no point in debating detailed requirements

Getting to the Heart of the Problem (cont'd)

- Were the stakeholders engaged and thinking about requirements?
- Were the stakeholders in agreement?
- Do they have the required knowledge of the system and its users?

Getting to the Heart of the Problem (cont'd)

- Did Fred have a clear understanding of
  - scope?
  - requirements?
  - risks?
- Was he focusing on requirements in the right order?
- Was he trying to get too many requirements nailed down up front?

Getting to the Heart of the Problem (cont'd)

- Do stakeholders have a good understanding of the existing software system?
  - capabilities
  - limitations
- Is the system (and architecture) robust in the face of change?

Getting to the Heart of the Problem (cont'd)

- Do the team have the competence to get the job done?
- Is the team working together as a unit?
- Are there conflicts waiting to surface?

Getting to the Heart of the Problem (cont'd)

- Were the problems due to
  - a need to find work for all the team members?
  - the way the work had been planned?
- Was the team working on the wrong things because of a previous planning error?

Getting to the Heart of the Problem (cont'd)

- Were the problems caused by the method?
  - requirements diverged because it insisted that all requirements were detailed before design and implementation?

Getting to the Heart of the Problem

- …an
- what was that value?
- If the value is not understood, there is no point in debating detailed requirements

Getting to the Heart of the Problem (cont'd)

- Do they have the required knowledge of the system and its users?

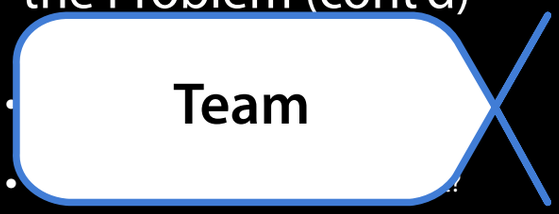
Getting to the Heart of the Problem (cont'd)

- Was he focusing on requirements in the right order?
- Was he trying to get too many requirements nailed down up front?

Getting to the Heart of the Problem (cont'd)

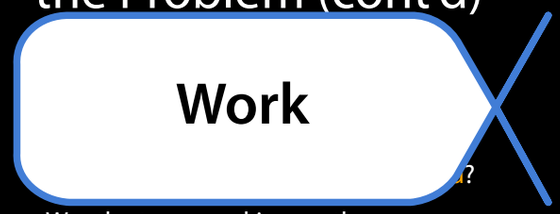
- …e
- limitations
- Is the system (and architecture) robust in the face of change?

Getting to the Heart of the Problem (cont'd)



- …?
- Are there conflicts waiting to surface?

Getting to the Heart of the Problem (cont'd)



- …?
- Was the team working on the wrong things because of a previous planning error?

Getting to the Heart of the Problem (cont'd)

- …ted
- before design and implementation?

The Kernel

- Essential things to progress and evolve
  - alphas
- Essential things to do
  - activity spaces
- Essential capabilities needed
  - competencies

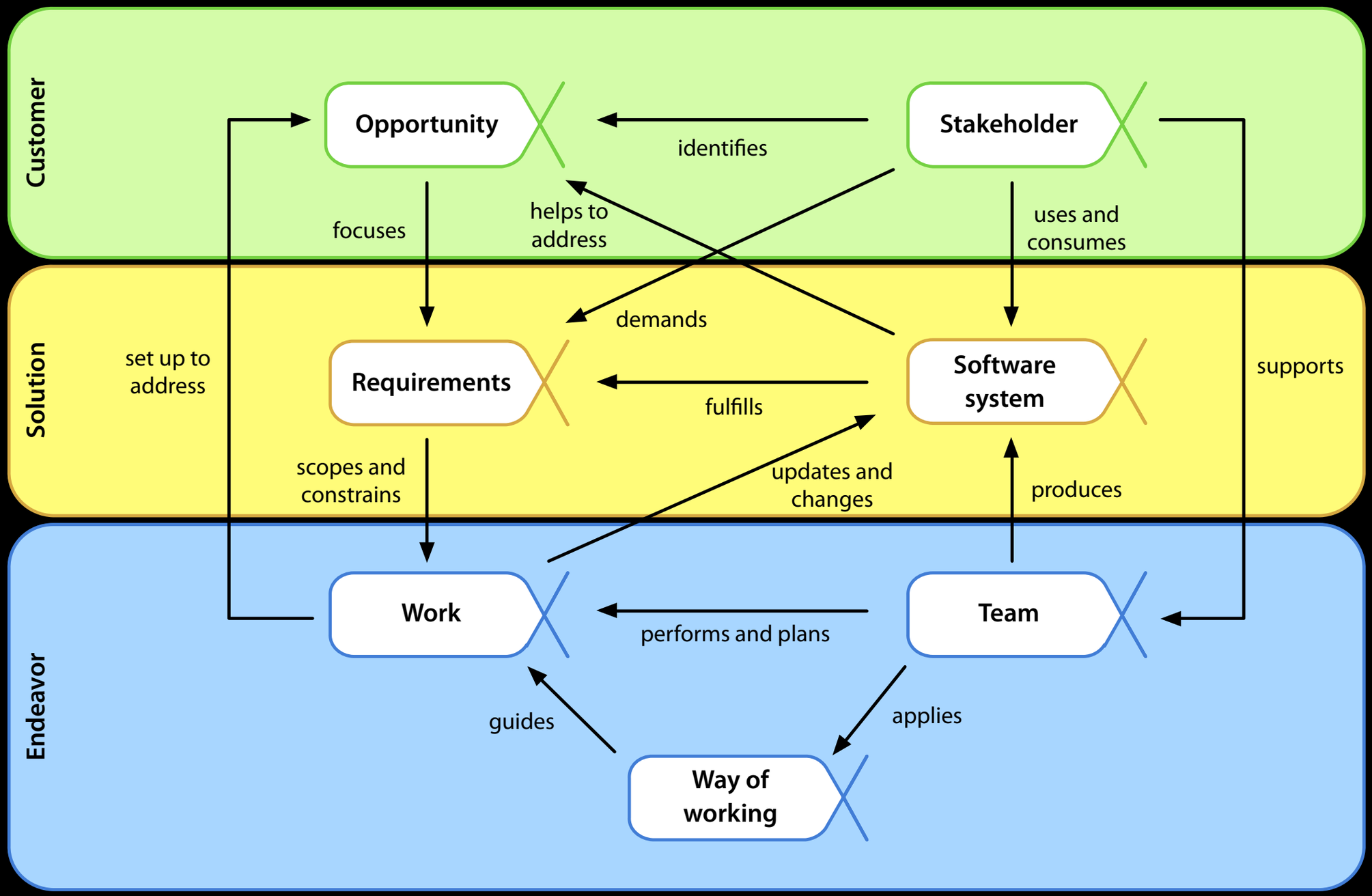
Areas of concern

Customer

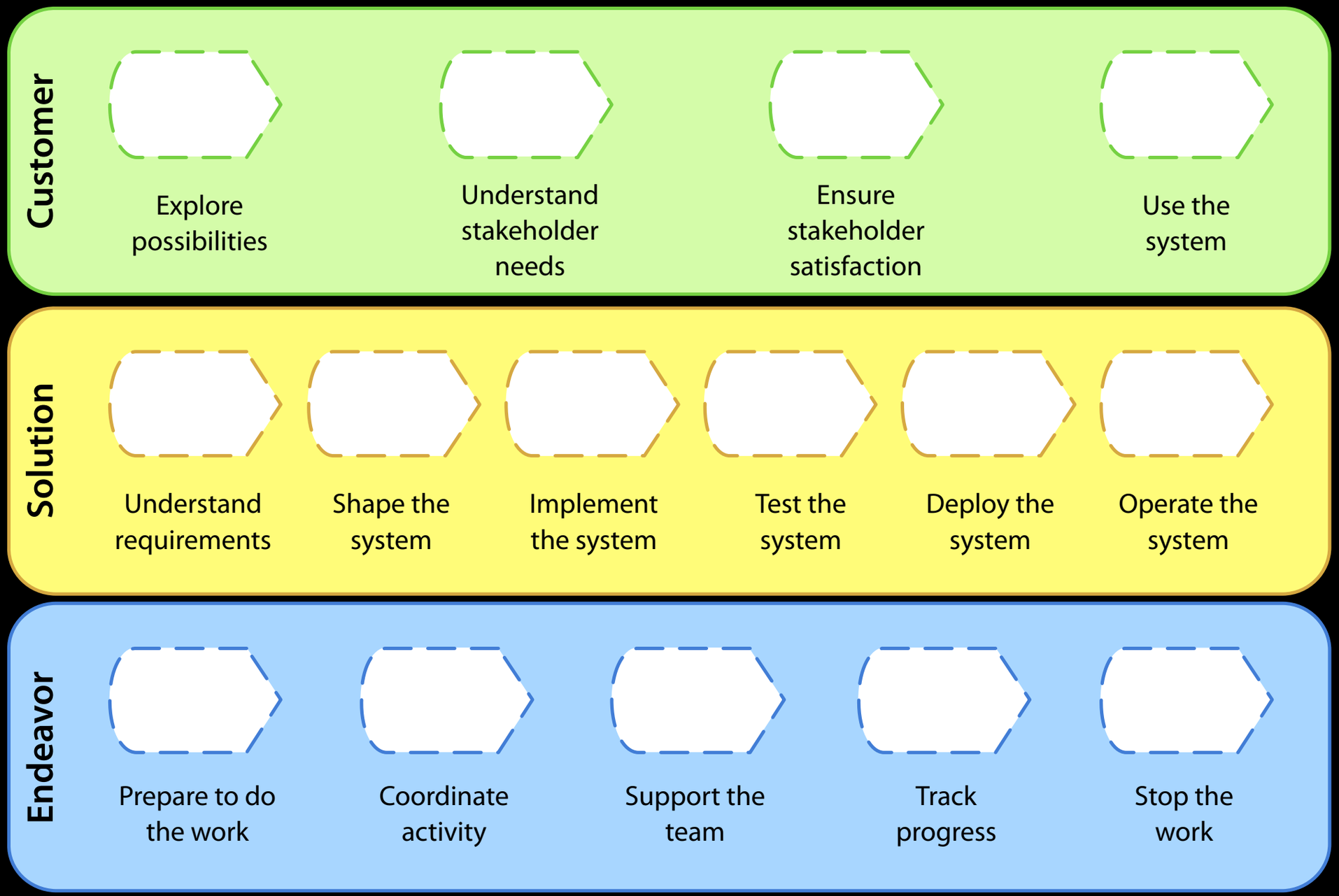
Solution

Endeavor

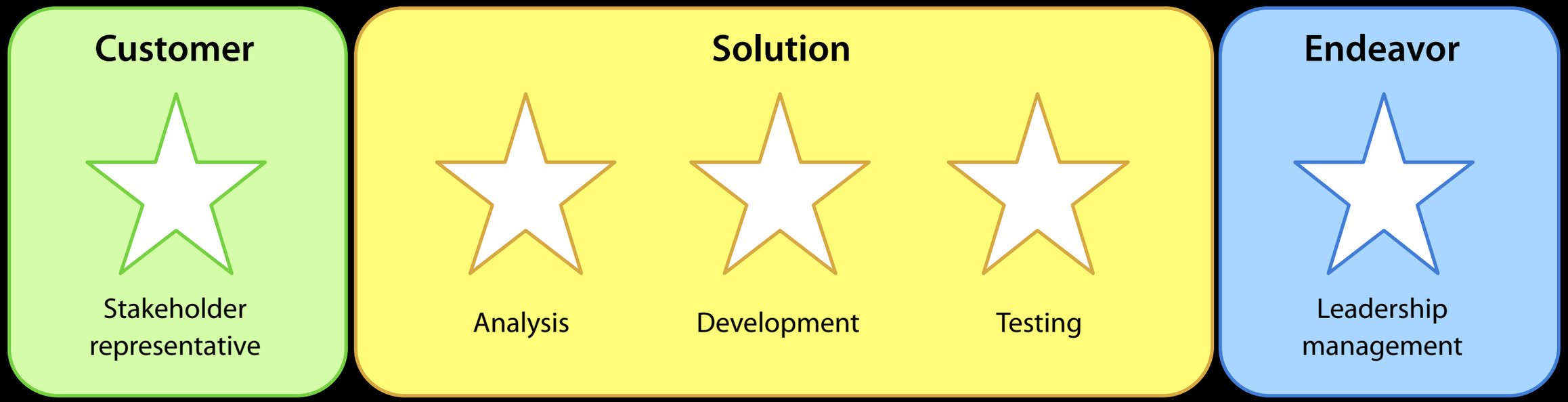
The kernel alphas



The kernel activity space



Competencies



Week 2

- **Monday**: SEMAT continued
- **Wednesday**: Pitch your app ideas
- **Friday**: RUP and XP
- **Sunday**: Submit app vision