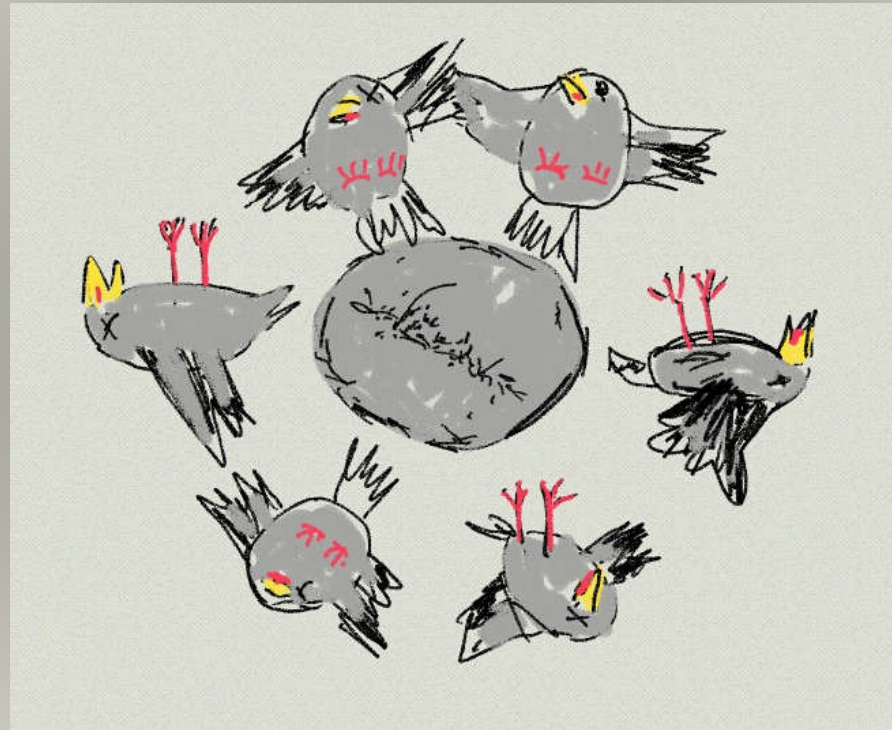


Killing Several Birds with One Stone: Using SEMAT's ESSENCE in Teaching Software Engineering



KTH Royal Institute of Technology
Stockholm, Sweden
mekm2@kth.se

Outline

- Problems within education

- The SEMAT community

- ESSENCE

- SE education at KTH

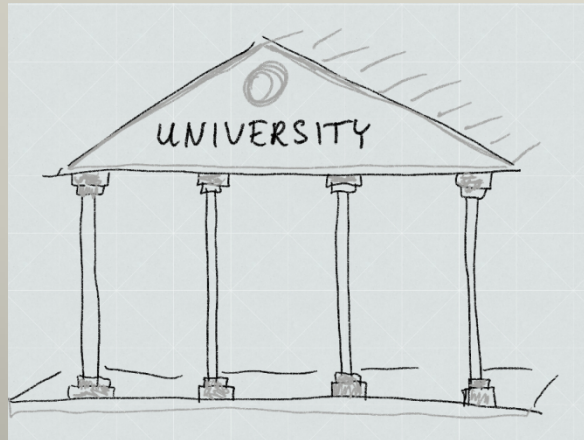
- So how many birds....

Graduated students are poorly equipped for their future careers



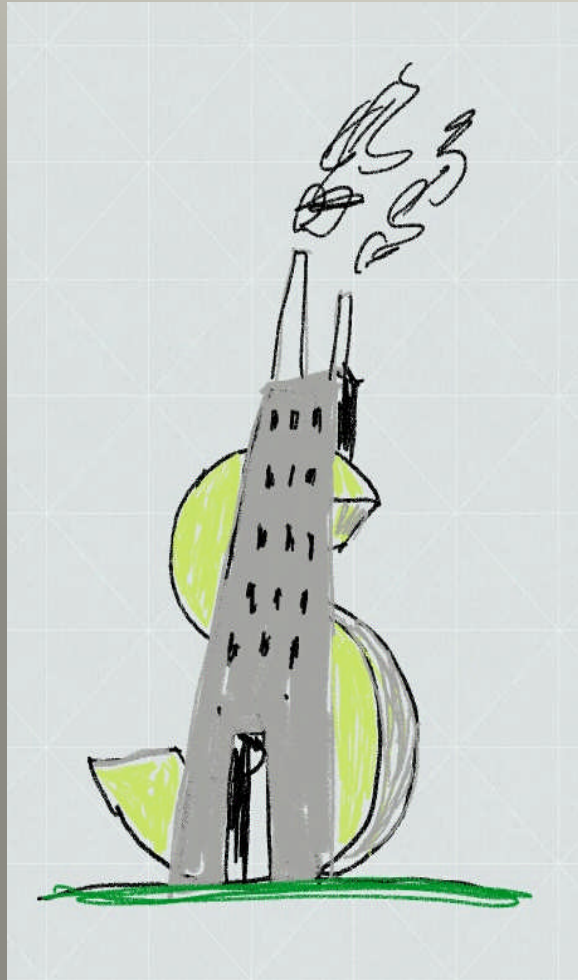
- Software engineering is difficult to learn in a classroom environment.
- Next to impossible to gain experience.
- Difficulties to assess students' progress and competence.
- Students have become nationally and internationally mobile.

Problems at universities



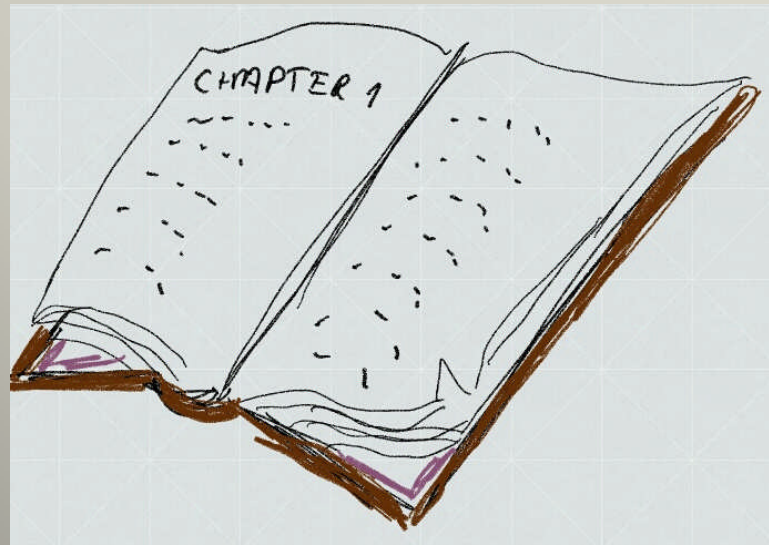
- Software engineering is often squeezed into one course.
- You cannot grasp all within one course.
- No way of assessing current educators wrt how capable they are to deliver competencies.
- Need for a global approach for evaluating the educators, not just school reputation.

Problems within industry



- Shortage of employee candidates.
- Graduate students are poorly equipped for their future careers.
- Graduate students do not possess enough knowledge about and skills within software engineering.

Problems with educational material



- No interest in writing educational books.
- Different terminology used.
- Different understanding of software engineering
- Books too difficult for students to understand.
- There are no books that provide a good high-level overview of software engineering.

Outline

- Problems within education

- The SEMAT community

- ESSENCE

- SE education at KTH

- So how many birds....

We have no widely accepted common ground



- We do not lack methods or practices.
- Everyone of us knows how to develop **our own** software, but as a community we have **no** widely accepted common ground.

Some common problems

- Software development methods are all unique in their design and use of terminology so they cannot be easily compared.
- We do not know which methods we have in a large company.
- We have no solid knowledge which we can take from job to job.
- We have no common platform on which we could base
 - development of our methods
 - improvement of our methods
 - planning
 - Project status evaluation
 - risk identification
 -

SEMAT: Software Engineering Method and Theory

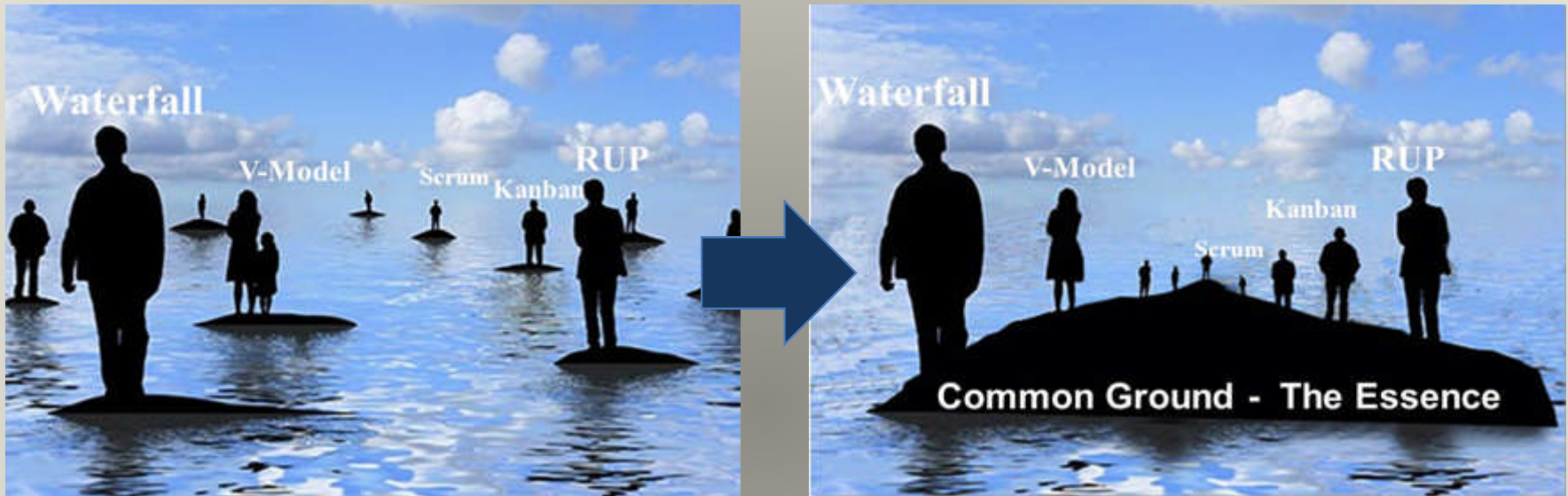


Founded by Ivar Jacobson, Bertrand Meyer, Richard Soley in 2009



Re-found software engineering as a rigorous discipline based on a general theory of software engineering and a unifying process framework

Common Ground



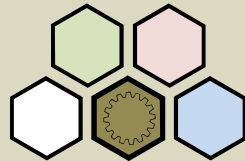
**Find a kernel (essence) of widely
agreed elements
within software engineering**

Outline

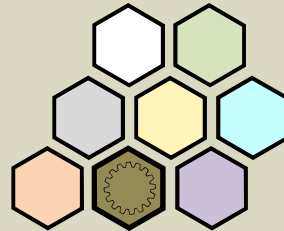
- Problems within education
- The SEMAT community
- ESSENCE
- SE education at KTH
- So how many birds....

What is Essence?

Methods

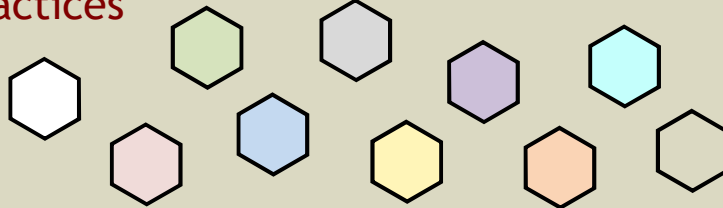


Custom Method M



Custom Method N

Practices



Kernel

Essence Kernel



Language

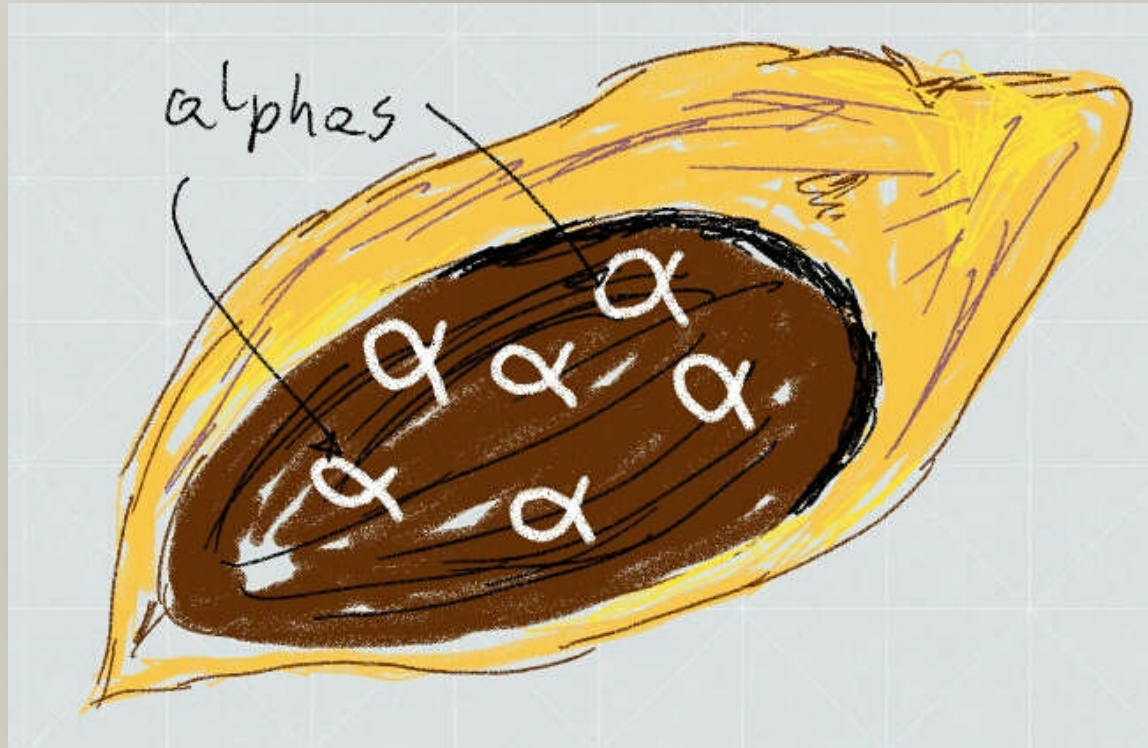
Essence Language



Essence - Kernel and Language for
Software Engineering Methods

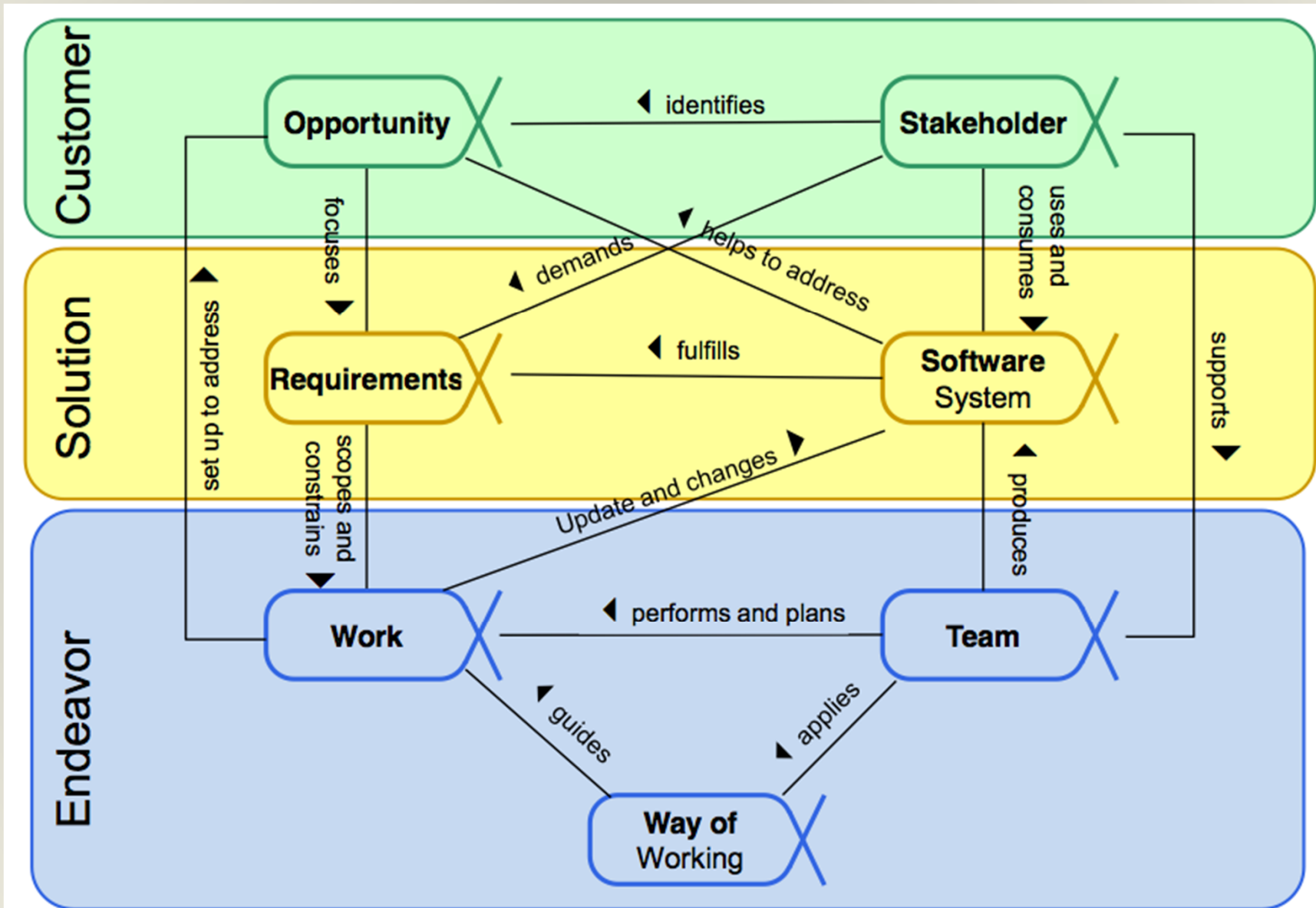
<http://www.omg.org/spec/Essence/Current>

The Kernel

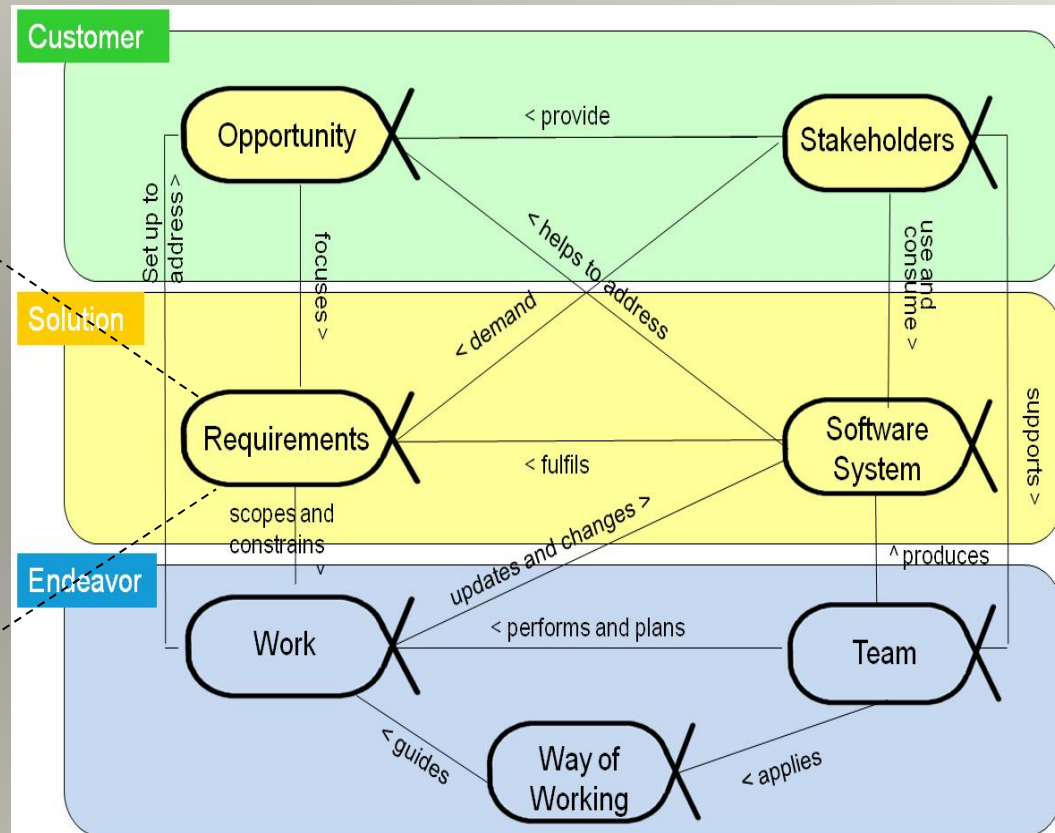
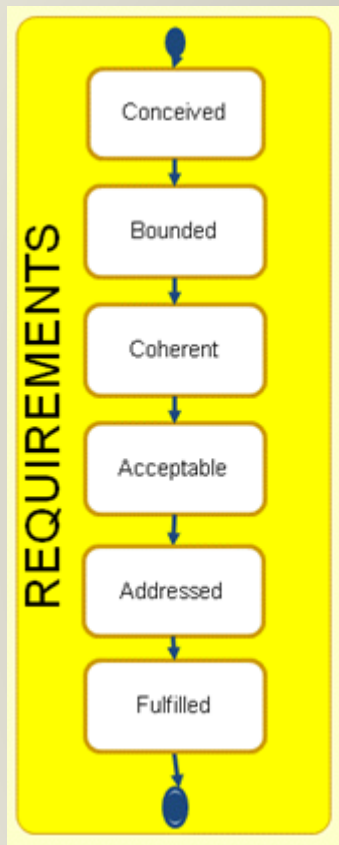


- Captures the essence of software engineering with the aid of essential properties called Alphas (Abstract-Level Progress Health Atttribute).

Essence Kernel Alphas

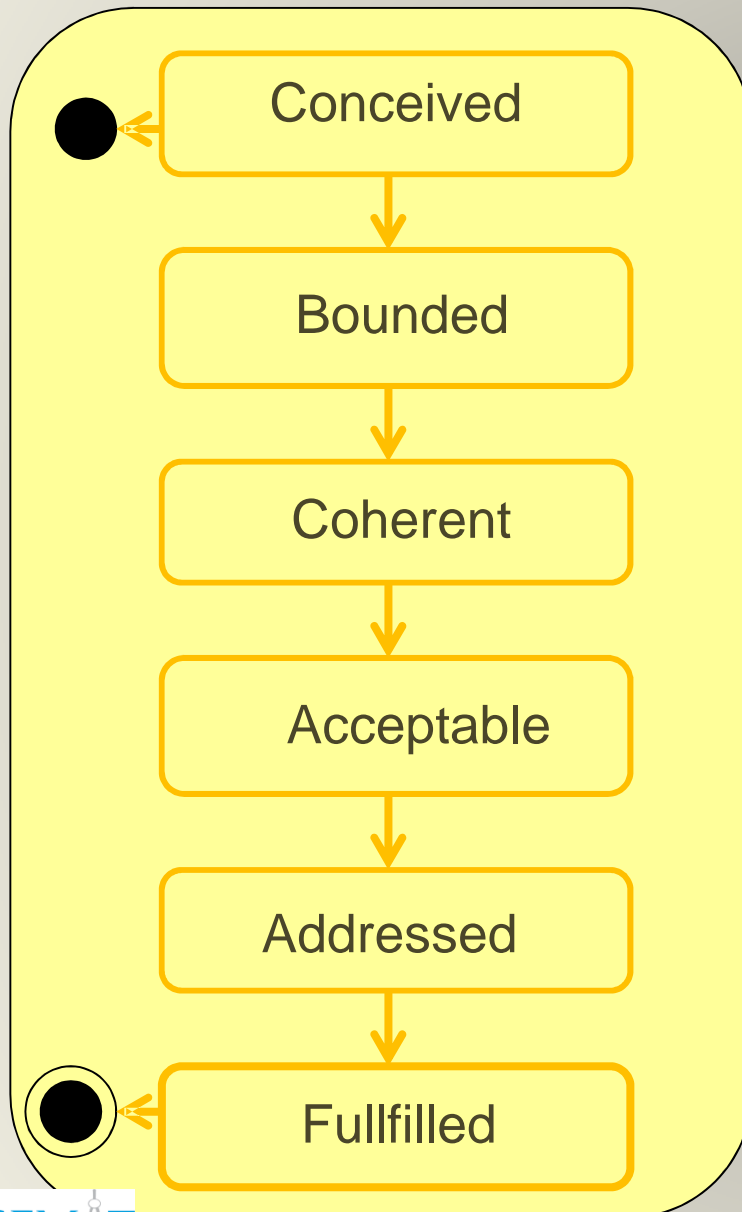


Requirements— one of the Alphas



Requirements Definition: What the software system must do to address the opportunity and satisfy the stakeholders.

The states of Requirements



The need for a new system has been agreed.

The purpose and theme of the new system are clear.

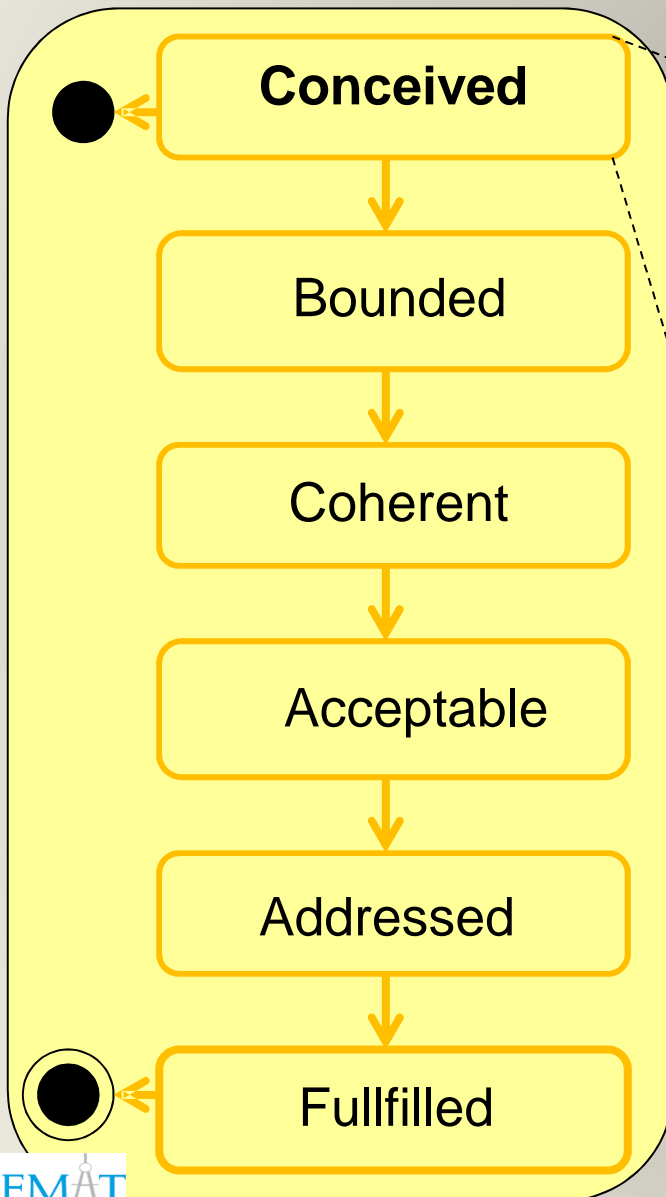
The requirements provide a coherent description of the essential characteristics of the new system.

The requirements describe a system that is acceptable to the stakeholders.

Enough of the requirements have been addressed to satisfy the need for a new system in a way that is acceptable to the stakeholders.

The requirements have been addressed to fully satisfy the need for a new system.

Checklist for achieving the Conceived state of Requirements



- The initial set of stakeholders agrees that a system is to be produced.
- The stakeholders that will use the new system are identified.
- The stakeholders that will fund the initial work on the new system are identified.
- There is a clear opportunity for the new system to address.

An example of a checklist for the Stakeholder Alpha

State	Checklist
Recognized	<p>All the different groups of stakeholders that are, or will be, affected by the development and operation of the software system are identified.</p> <p>There is agreement on the stakeholder groups to be represented. At a minimum, the stakeholders groups that fund, use, support, and maintain the system have been considered.</p> <p>The responsibilities of the stakeholder representatives have been defined.</p>
Represented	<p>The stakeholder representatives have agreed to take on their responsibilities.</p> <p>The stakeholder representatives are authorized to carry out their responsibilities.</p> <p>The collaboration approach among the stakeholder representatives has been agreed.</p> <p>The stakeholder <u>representatives</u> support and respect the team's way of working.</p>
Involved	<p>The stakeholder representatives assist the team in accordance with their responsibilities.</p> <p>The stakeholder representatives provide feedback and take part in decision making in a timely manner.</p> <p>The stakeholder representatives promptly communicate changes that are relevant for</p>

Essence Cards



- Each Alpha is materialized in a set of cards.
- One card represents one state of the Alpha.

Working with the Alphas

Requirements

<input type="checkbox"/> Requirements	<input type="checkbox"/> Requirements	<input type="checkbox"/> Requirements	<input type="checkbox"/> Requirements	<input type="checkbox"/> Requirements	<input type="checkbox"/> Requirements
Conceived	Bounded	Coherent	Sufficient	Satisfactory	Fulfilled
<ul style="list-style-type: none"> Need for system agreed by initial stakeholders Users and customers identified Expected benefit of system agreed 	<ul style="list-style-type: none"> Theme, scope, success criteria of system is clear Mechanisms for managing requirements in place Constraints and assumptions considered 	<ul style="list-style-type: none"> Described requirements provide coherent picture of the system Conflicting requirements separated Important usage scenarios explained Priority of requirements clear 	<ul style="list-style-type: none"> Requirements adequately describe solution and acceptable to stakeholders Rate of change to agreed requirements is low and under control 	<ul style="list-style-type: none"> System implementing requirements is worth making operational Enough requirements are implemented 	<ul style="list-style-type: none"> System implementing requirements is accepted as fully satisfying the need No outstanding requirement items prevent system from being accepted Stakeholders accept requirements as accurate
1 / 6	2 / 6	3 / 6	4 / 6	5 / 6	6 / 6

Software System

<input type="checkbox"/> Software System	<input type="checkbox"/> Software System	<input type="checkbox"/> Software System	<input type="checkbox"/> Software System	<input type="checkbox"/> Software System	<input type="checkbox"/> Software System
Architecture Selected	Demonstrable	Usable	Ready	Operational	Retired
<ul style="list-style-type: none"> Architecture selected that address key technical risks Criteria for selecting architecture agreed Platforms, technologies, languages selected Buy, build, reuse decisions made 	<ul style="list-style-type: none"> Executable version of system demonstrates architecture is fit for purpose Supports functional and non-functional testing Critical interface and system configurations exercised 	<ul style="list-style-type: none"> System is usable and has desired quality characteristics System can be operated by users Functionality and performance have been tested and accepted Defect levels acceptable Release content known 	<ul style="list-style-type: none"> System (as a whole) has been accepted for deployment in operational environment Sponsors, users, stakeholders accept system as fit for purpose Installation and other documents available Operational support in place 	<ul style="list-style-type: none"> System in use in operational environment System available to intended users At least one example of system is fully operational System supported to agreed service levels 	<ul style="list-style-type: none"> System no longer supported Updates to system will no longer be produced System has been replaced or discontinued.
1 / 6	2 / 6	3 / 6	4 / 6	5 / 6	6 / 6

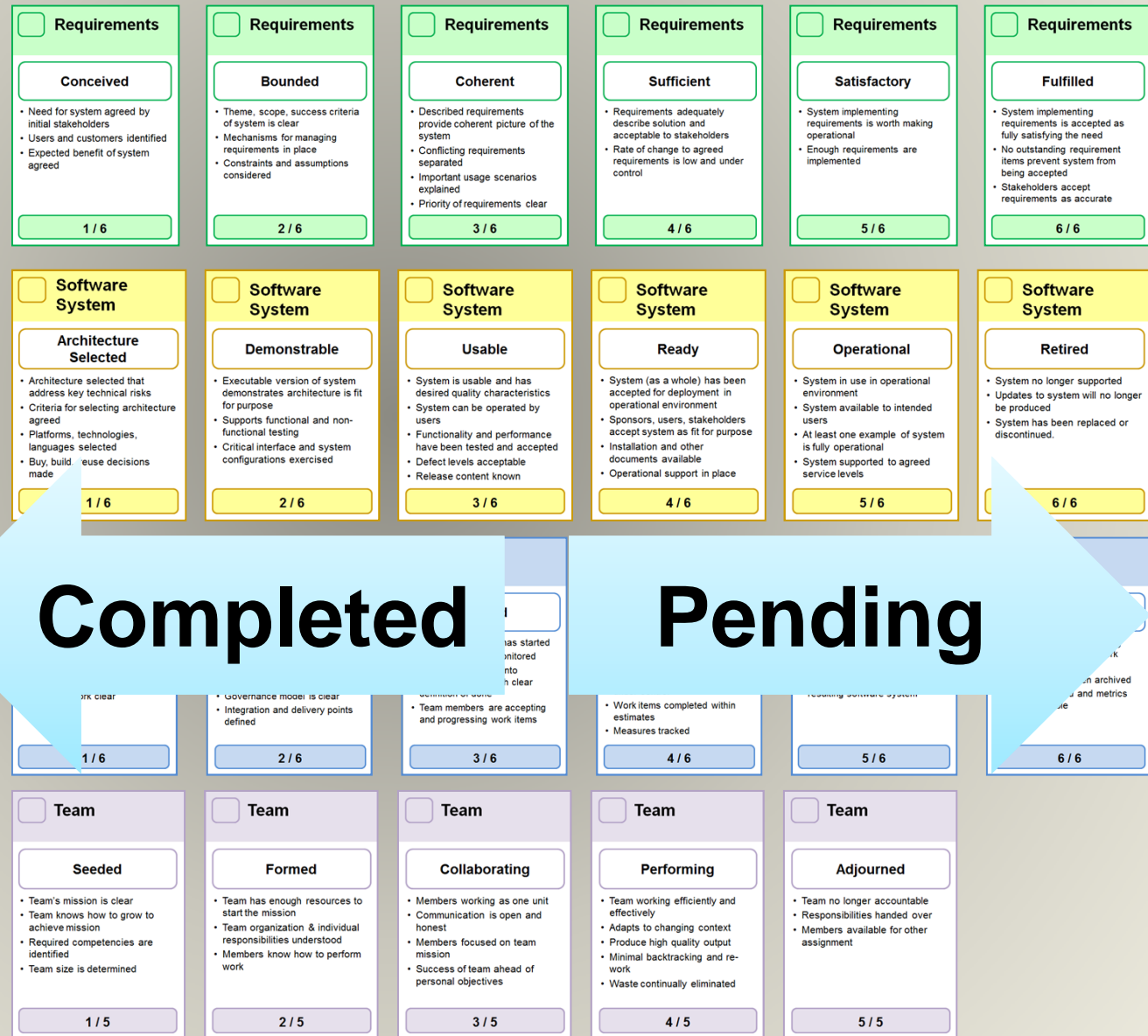
Work

<input type="checkbox"/> Work	<input type="checkbox"/> Work	<input type="checkbox"/> Work	<input type="checkbox"/> Work	<input type="checkbox"/> Work	<input type="checkbox"/> Work
Initiated	Prepared	Started	Under Control	Concluded	Closed
<ul style="list-style-type: none"> Work initiator and client known Work goal and constraints clear Sponsorship and funding model clear Priority of work clear 	<ul style="list-style-type: none"> Cost & effort understood Funding in place Resource availability and risk exposure understood Governance model is clear Integration and delivery points defined 	<ul style="list-style-type: none"> Development work has started Work progress is monitored Resource availability and risk exposure understood Work broken down into actionable items with clear definition of done Team members are accepting and progressing work items 	<ul style="list-style-type: none"> Work going well, risks being managed, productivity levels acceptable Unplanned work & re-work under control Work items completed within estimates Measures tracked 	<ul style="list-style-type: none"> Work to produce results have been finished Work results are being achieved The client has accepted the resulting software system 	<ul style="list-style-type: none"> All remaining housekeeping tasks completed, and work officially closed Everything has been archived Lessons learned and metrics made available
1 / 6	2 / 6	3 / 6	4 / 6	5 / 6	6 / 6

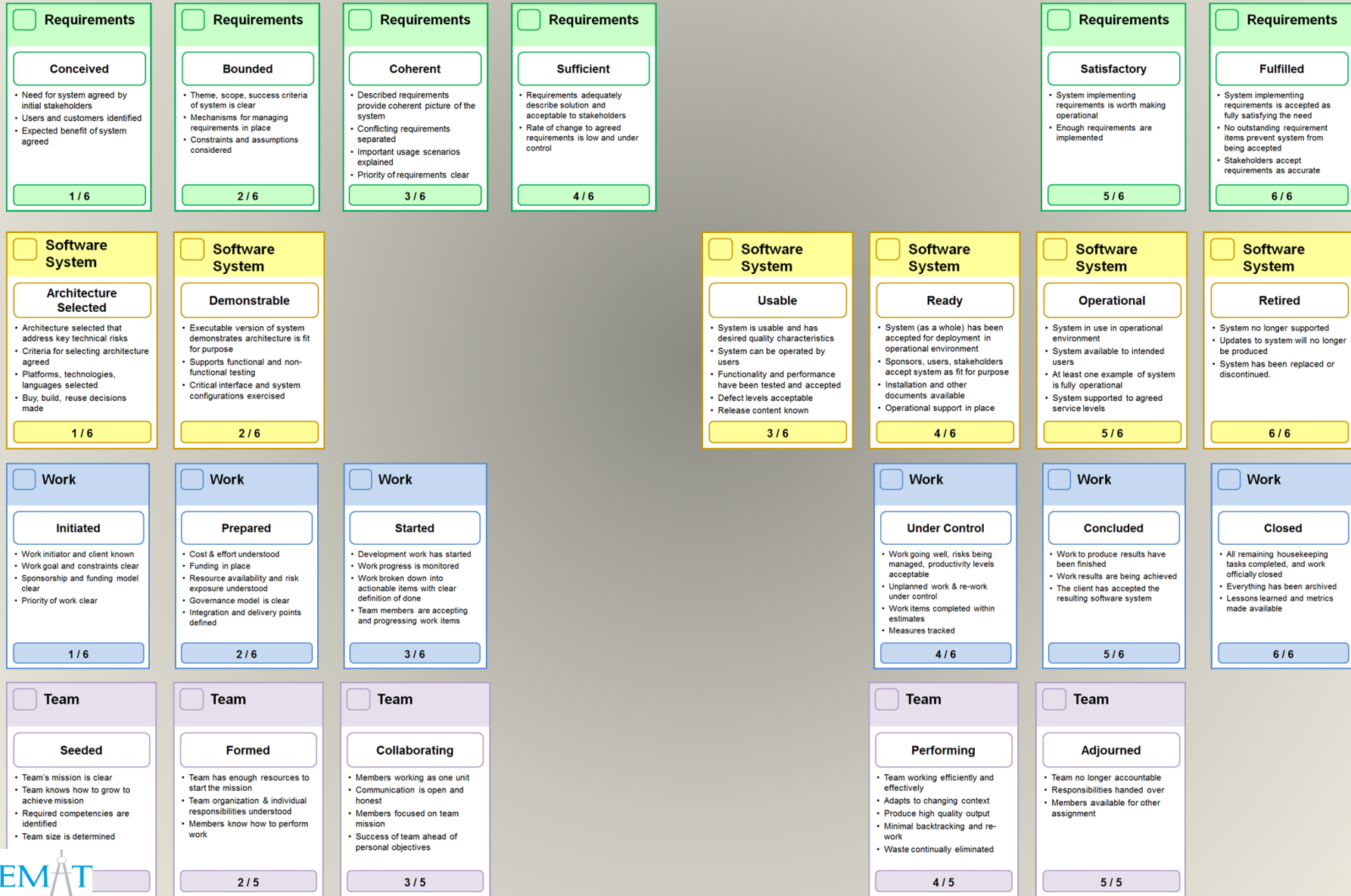
Team

<input type="checkbox"/> Team	<input type="checkbox"/> Team	<input type="checkbox"/> Team	<input type="checkbox"/> Team	<input type="checkbox"/> Team
Seeded	Formed	Collaborating	Performing	Adjourned
<ul style="list-style-type: none"> Team's mission is clear Team knows how to grow to achieve mission Required competencies are identified Team size is determined 	<ul style="list-style-type: none"> Team has enough resources to start the mission Team organization & individual responsibilities understood Members know how to perform work 	<ul style="list-style-type: none"> Members working as one unit Communication is open and honest Members focused on team mission Success of team ahead of personal objectives 	<ul style="list-style-type: none"> Team working efficiently and effectively Adapts to changing context Produce high quality output Minimal backtracking and re-work Waste continually eliminated 	<ul style="list-style-type: none"> Team no longer accountable Responsibilities handed over Members available for other assignment
1 / 5	2 / 5	3 / 5	4 / 5	5 / 5

Determine Current State



Determine Next State



Determine How to Achieve Next State

Requirements

Satisfactory

- System implementing requirements is worth making operational
- Enough requirements are implemented

5 / 6

Software System

Usable

- System is usable and has desired quality characteristics
- System can be operated by users
- Functionality and performance have been tested and accepted
- Defect levels acceptable
- Release content known

3 / 6

Work

Under Control

- Work going well, risks being managed, productivity levels acceptable
- Unplanned work & re-work under control
- Work items completed within estimates
- Measures tracked

4 / 6

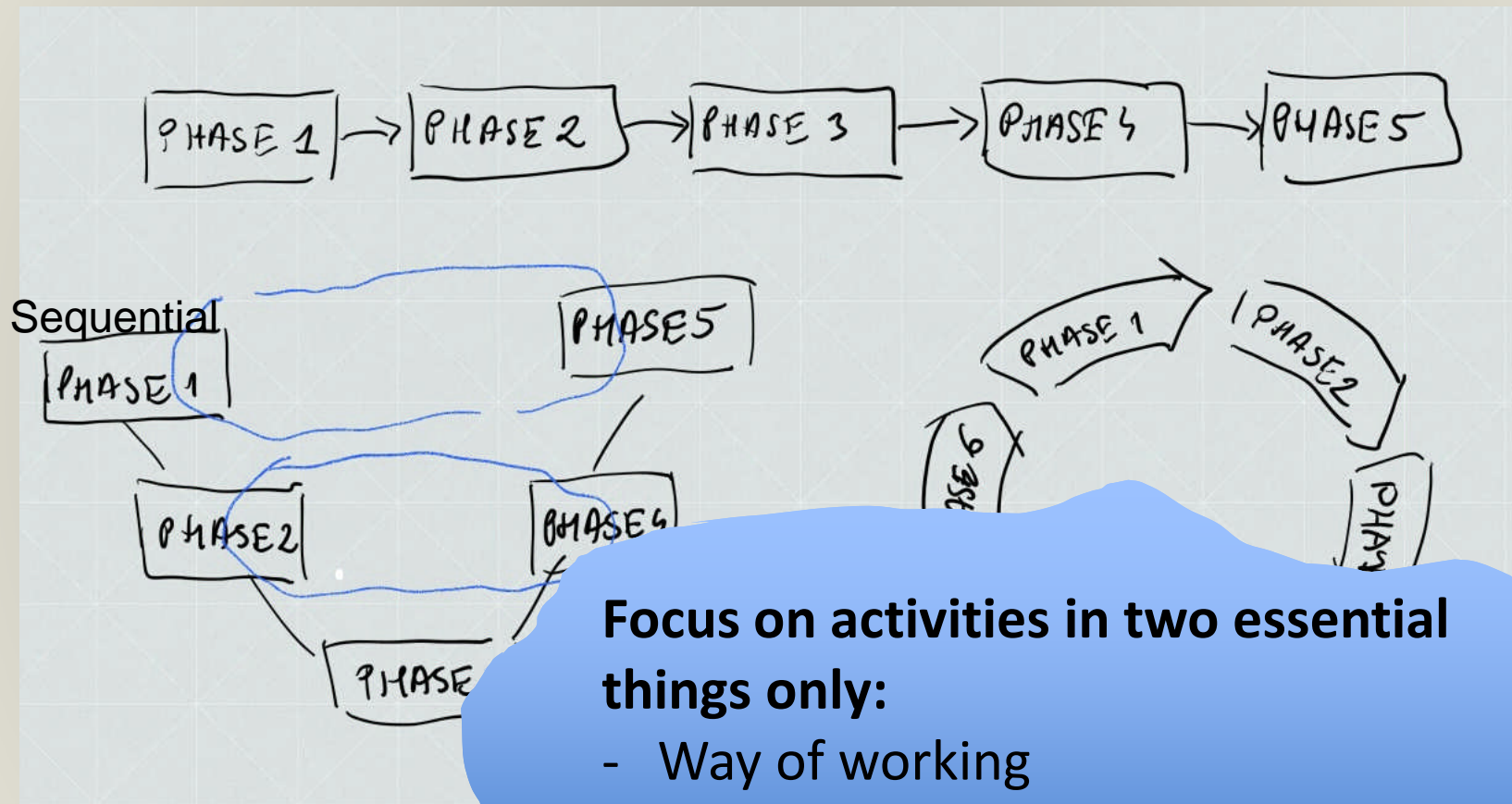
Team

Performing

- Team working efficiently and effectively
- Adapts to changing context
- Produce high quality output
- Minimal backtracking and re-work
- Waste continually eliminated

4 / 5

Some software development methods today



Focus on activities in two essential things only:

- Way of working
- Work

Essence Kernel

requirements

software system

work

team

<input type="checkbox"/> Requirements Conceived <ul style="list-style-type: none"> Need for system agreed by initial stakeholders Users and customers identified Expected benefit of system agreed <p>1 / 6</p>	<input type="checkbox"/> Requirements Bounded <ul style="list-style-type: none"> Theme, scope, success criteria of system is clear Mechanisms for managing requirements in place Constraints and assumptions considered <p>2 / 6</p>	<input type="checkbox"/> Requirements Coherent <ul style="list-style-type: none"> Described requirements provide coherent picture of the system Conflicting requirements separated Important usage scenarios explained Priority of requirements clear <p>3 / 6</p>	<input type="checkbox"/> Requirements Sufficient <ul style="list-style-type: none"> Requirements adequately describe solution and acceptable to stakeholders Rate of change to agreed requirements is low and under control <p>4 / 6</p>	<input type="checkbox"/> Requirements Satisfactory <ul style="list-style-type: none"> System implementing requirements is worth making operational Enough requirements are implemented <p>5 / 6</p>	<input type="checkbox"/> Requirements Fulfilled <ul style="list-style-type: none"> System implementing requirements is accepted as fully satisfying the need No outstanding requirement items prevent system from being accepted Stakeholders accept requirements as accurate <p>6 / 6</p>
---	--	---	---	--	--

Focus on states in seven essential things:

- Way of working
- Work
- Stakeholder
- Opportunity
- Requirements
- Software System
- Team

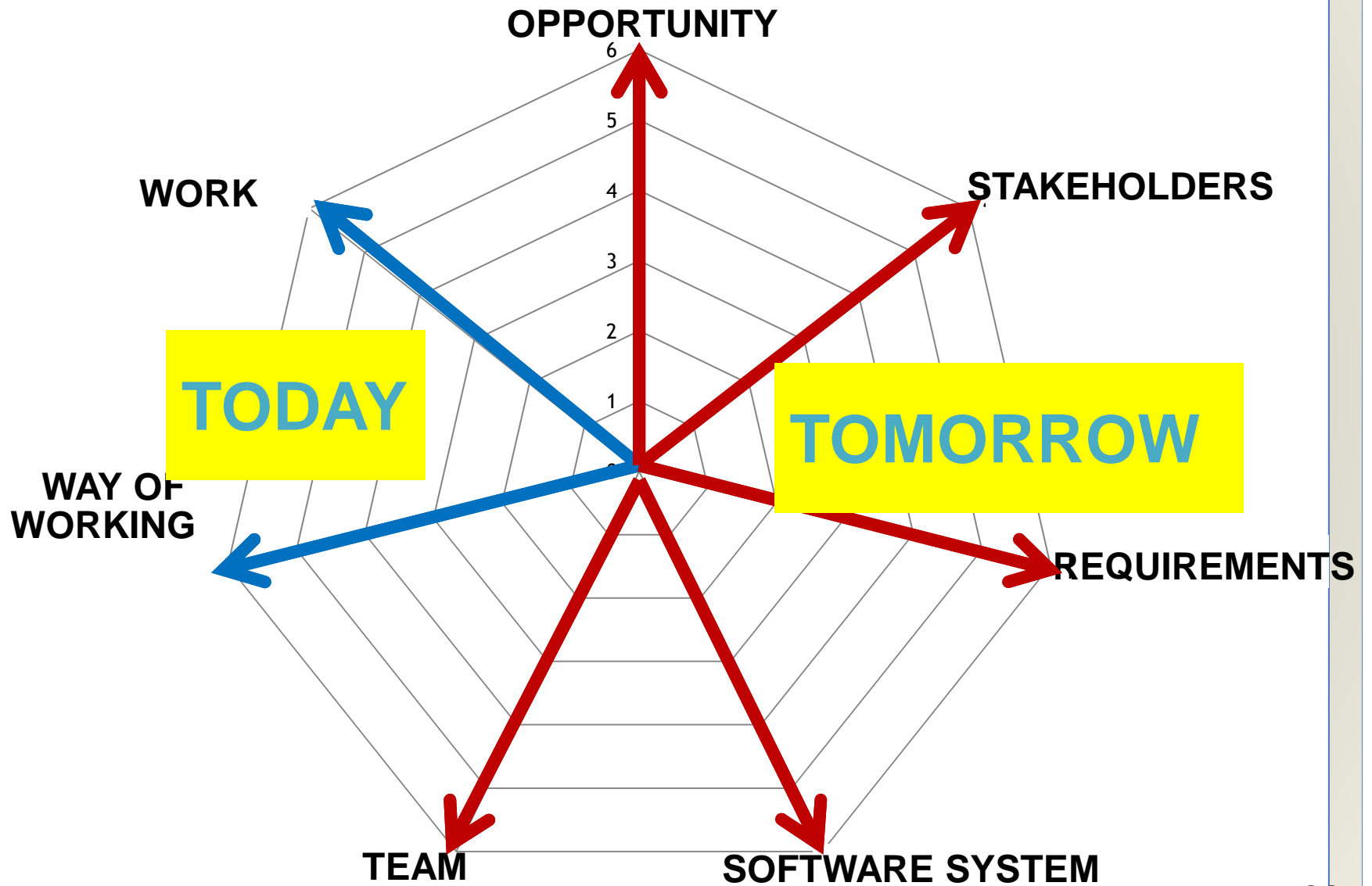
<input type="checkbox"/> Work Under Control <ul style="list-style-type: none"> Work going well, risks being managed, productivity levels acceptable Unplanned work & re-work under control Work items completed within estimates Measures tracked <p>4 / 6</p>	<input type="checkbox"/> Work Concluded <ul style="list-style-type: none"> Work to produce results have been finished Work results are being achieved The client has accepted the resulting software system <p>5 / 6</p>	<input type="checkbox"/> Work Closed <ul style="list-style-type: none"> All remaining housekeeping tasks completed, and work officially closed Everything has been archived Lessons learned and metrics made available <p>6 / 6</p>
---	--	---

<input type="checkbox"/> Team Performing <ul style="list-style-type: none"> Team working efficiently and effectively Adapts to changing context Produce high quality output Minimal backtracking and re-work Waste continually eliminated <p>4 / 5</p>	<input type="checkbox"/> Team Adjourned <ul style="list-style-type: none"> Team no longer accountable Responsibilities handed over Members available for other assignment <p>5 / 5</p>
--	--

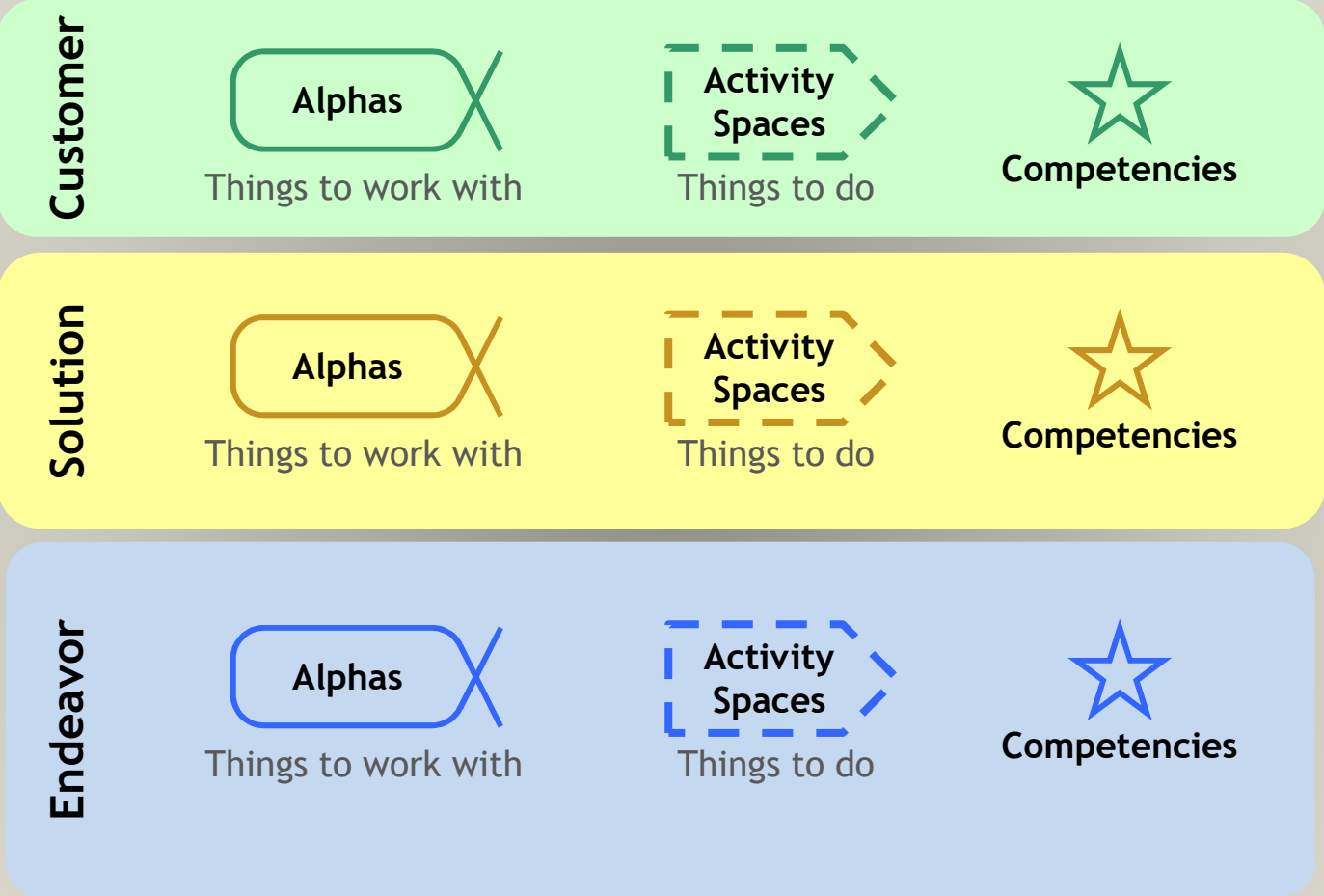
<input type="checkbox"/> Team Collaborating <ul style="list-style-type: none"> Members working as one unit Communication is open and honest Members focused on team mission Success of team ahead of personal objectives <p>3 / 5</p>	<input type="checkbox"/> Team Establishing <ul style="list-style-type: none"> Team organization & individual responsibilities understood Members know how to perform work <p>2 / 5</p>	<input type="checkbox"/> Team Forming <ul style="list-style-type: none"> Team's mission is clear Team knows how to grow to achieve mission Required competencies are identified Team size is determined <p>1 / 5</p>
--	---	---



Following essential things



Essence Kernel



Outline

- Problems within education
- The SEMAT community
- ESSENCE
- SE education at KTH
- So how many birds....

IT-Project Course at KTH



Phase 1

(IT Project, Part 1)

Theoretical part, some
practical exercises

Phase 2

(IT Project, Part 2)

Practical course

Phase 3

(IT Project, Part 1)

Theoretical part, analytical
perspective

time →

**Other teachers
have this part**

Educational material

Scenario 1

A Scenario on Kick-Starting a Project

Mira Kajko-Mattsson

Paul E. McMahon

School of
KTH Royal Institute
of Technology
SWEDEN
mekm2@kth.se

Bob Pa
St. Louis Comm
Florissant
USA
bob@stl.com

Barry My
Johannesburg Cen
Engineering
Wits University,
barrym@icse.org.za

Purpose of the Scenario
By reading this scenario, the reader will understand how to kick-start a project. The approach is advanced by identifying and solving the most critical issues. The scenario is not intended to give you a complete picture of the project, but to provide the information needed for supporting the training objectives.

Pre-conditions
To get the most out of this scenario, the reader should have a good understanding of the Essence kernel and Checklists.

When to apply
The approach presented in this scenario can be used at any time in a project. It can be used to check the current status of a project, or to plan the next steps.

Essence Scope
This scenario focuses on the Essence kernel and Checklists.

Reference Cards
The Alpha cards are not part of this scenario.

Scenario 2

A Scenario on Solving Pain Points

Cecile Peraire
Carnegie Mellon University
Silicon Valley
USA
cecile.peraire@sv.cmu.edu

Paul E. McMahon
PEM Systems
USA
pemcmahon@acm.org

Mira Kajko-Mattsson
School of ICT
KTH Royal Institute of Technology
SWEDEN
mekm2@kth.se

Barry Myburgh
Johannesburg Centre for Software
Engineering
Wits University, South Africa
barrym@icse.org.za

Purpose of the scenario
By reading this scenario, the reader will understand how to solve pain points. The approach is advanced by identifying and solving the most critical issues. The scenario is not intended to give you a complete picture of the project, but to provide the information needed for supporting the training objectives.

Pre-conditions
To get the most out of this scenario, the reader should have a good understanding of the Essence kernel and Checklists.

When to Apply
This scenario illustrates how to introduce Essence incremental meetings. This is done in the context of an existing incremental approach described in this scenario remains applicable.

Essence Scope
The focus is on leveraging the Alpha cards only. Other cards are not part of this scenario.

Reference Cards

Handout 1 for Scenario 2

Solving Pain Points with Team Alpha

Maria Augusta Vieira Nelson
Department of Software Engineering
and Information Systems
Institute of Exact Sciences and
Informatics
Pontifical Catholic University of
Minas Gerais (PUC Minas), Brazil
mavnelson@gmail.com

Barry Myburgh
Johannesburg Centre for Software
Engineering
Wits University, South Africa
barrym@icse.org.za

Cecile Peraire
Carnegie Mellon University
Silicon Valley, USA
cecile.peraire@sv.cmu.edu

Mira Kajko-Mattsson
School of ICT
KTH Royal Institute of Technology

Paul E. McMahon
PEM Systems
USA
pemcmahon@acm.org

Handout 1 for Scenario 2

Solving Pain Points with Requirements Alpha

Cecile Peraire
Carnegie Mellon University
Silicon Valley
USA
cecile.peraire@sv.cmu.edu

Maria Augusta Vieira Nelson
Department of Software Engineering
and Information Systems
Institute of Exact Sciences and
Informatics
Pontifical Catholic University of
Minas Gerais (PUC Minas), Brazil
mavnelson@gmail.com

Mira Kajko-Mattsson
School of ICT
KTH Royal Institute of Technology
SWEDEN
mekm2@kth.se

Barry Myburgh
Johannesburg Centre for Software
Engineering
Wits University, South Africa
barrym@icse.org.za

Paul E. McMahon
PEM Systems
USA
pemcmahon@acm.org

See the usage of the Essence kernel on the website where it is.

Have knowledge of the Essence kernel, particularly on Solving Pain Points.

at release.

only. Other cards, like Activity Spaces

SEMAT kernel.

IT-Project Course at KTH



Phase 1

(IT Project, Part 1)

Theoretical part, some practical exercises



Phase 2

(IT Project, Part 2)

Practical course



Phase 3

(IT Project, Part 1)

Theoretical part, analytical perspective

time

Year 2015

(1) to help students understand what the project status evaluation looked like

(2) to find out whether they had any understanding difficulties.

To exercise the handouts

LESSON 1
REF Q:s

LESSON 3
SCENARIO 1

LESSON 5
SCENARIO 2
HANDOUTS 1-2

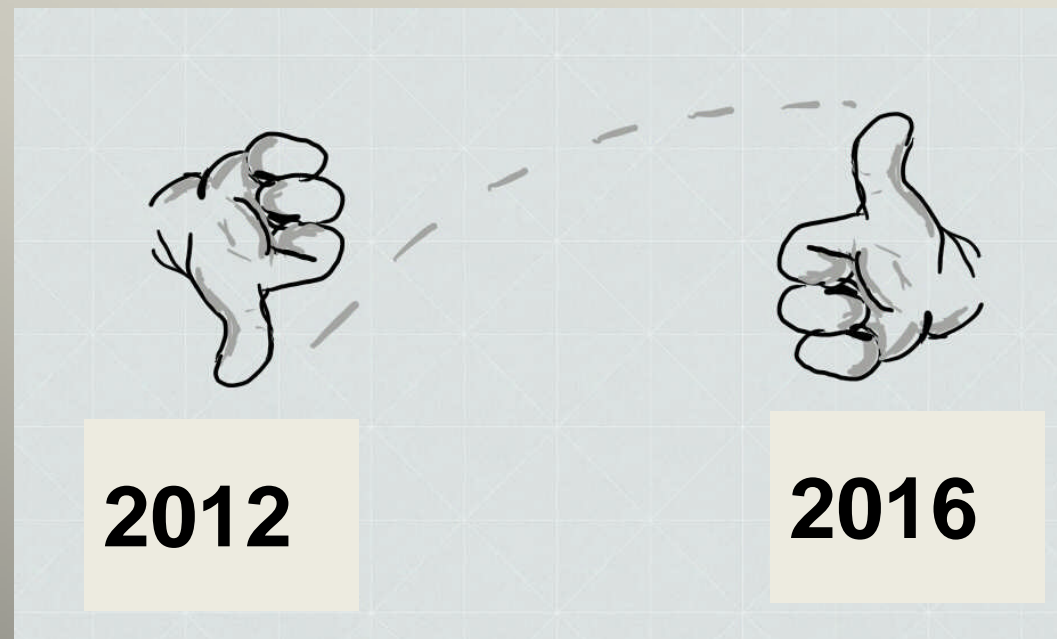
LECTURE | LECTURE | LECTURE | LECTURE | LECTURE | ...

PHASE 1

Year 2015



Students' attitude towards ESSENCE



- 60 % of the students of the year 2012 were not directly positive.
- They treated it as a burden and as an unnecessary new method.
- They expressed that they felt like guinea pigs.
- Now, about 80-90% of the students are positive.

Some opinions about ESSENCE

Intuitively understood

superior to other methods thanks to its full coverage of the essential things

provides a stable platform to stand on

enables distribution of work

assurance of project quality

time to learn ESSENCE is well invested

makes project more visible

facilitates project communication

useful in projects lacking any methods

Having a list of items to be checked off was definitely much better than having nothing and trying to figure out what to do next!

Outline

- Problems within education
 - The SEMAT community
 - ESSENCE
 - SE education at KTH
 - So how many birds....
-

So how many birds

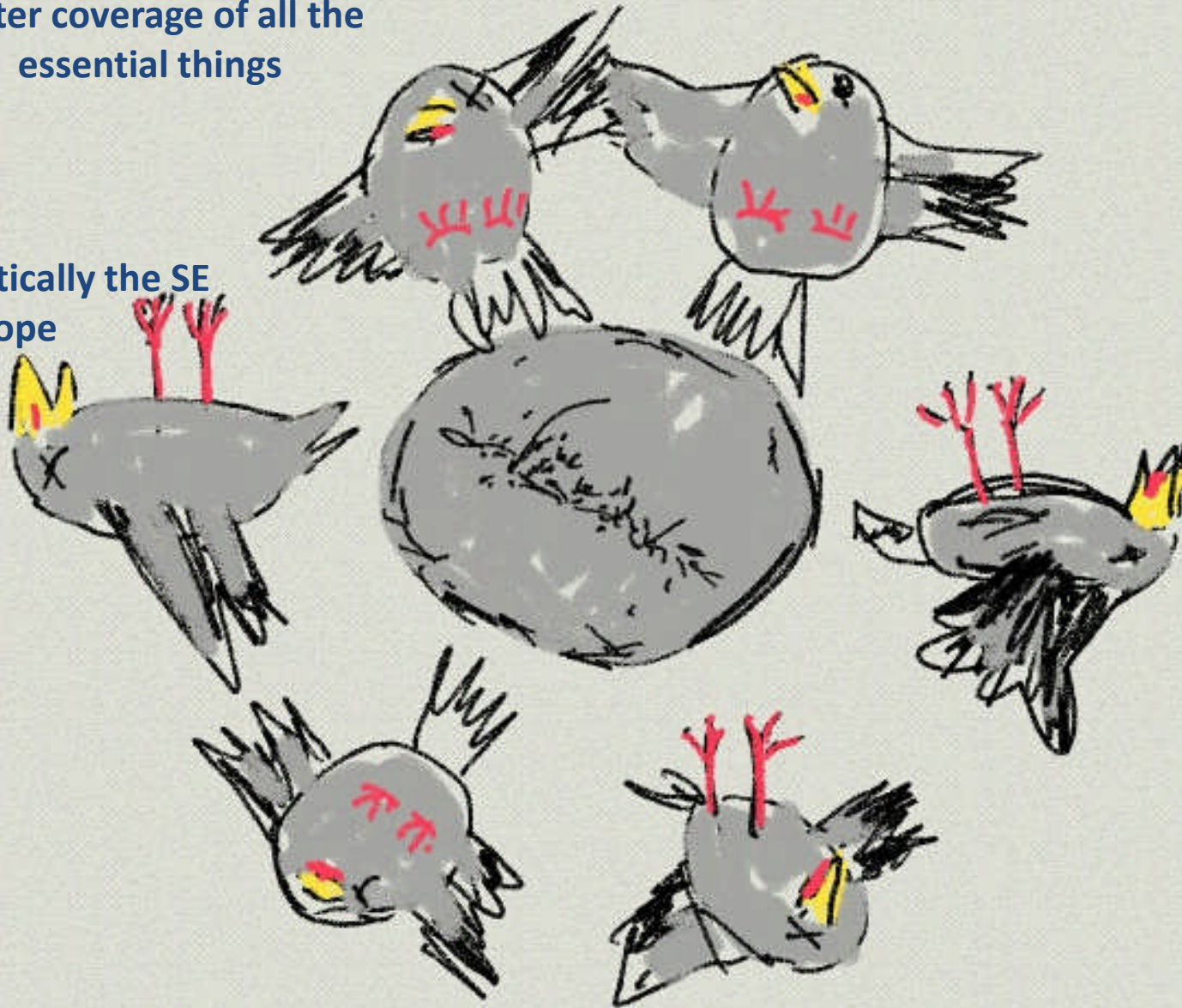
Complaints to the English
for their weird expression.

The Swedes are kinder. They only kill flies.

can we kill with one stone!

Better coverage of all the essential things

Grasp holistically the SE scope



Better coverage of all the essential things

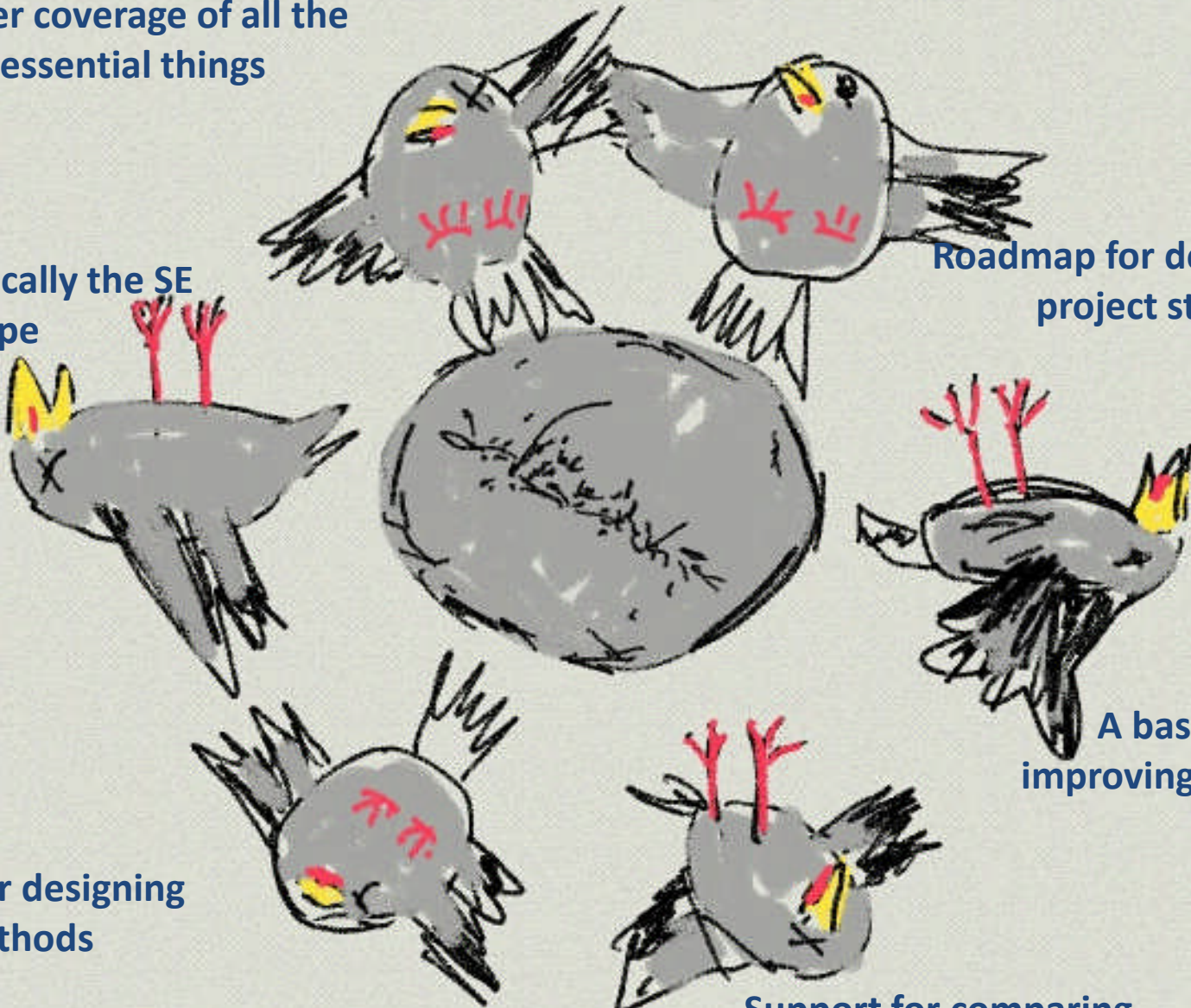
Grasp holistically the SE scope

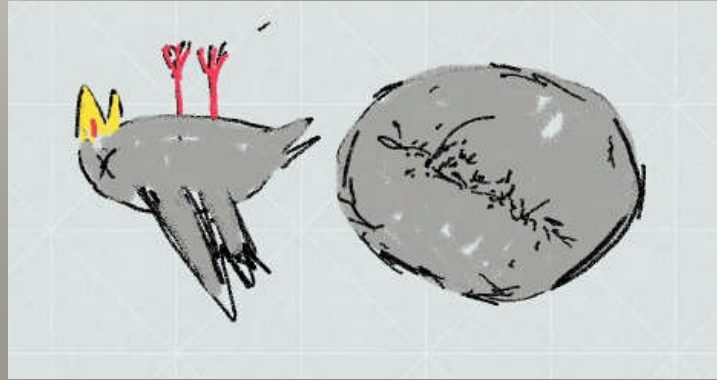
Roadmap for determining project status

A base for improving methods

A tool for designing methods

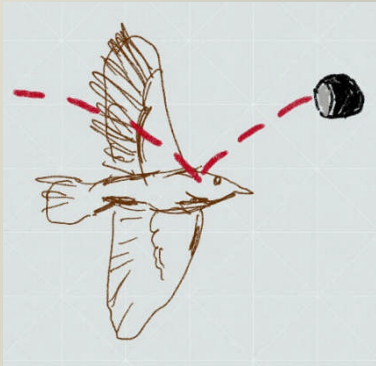
Support for comparing methods



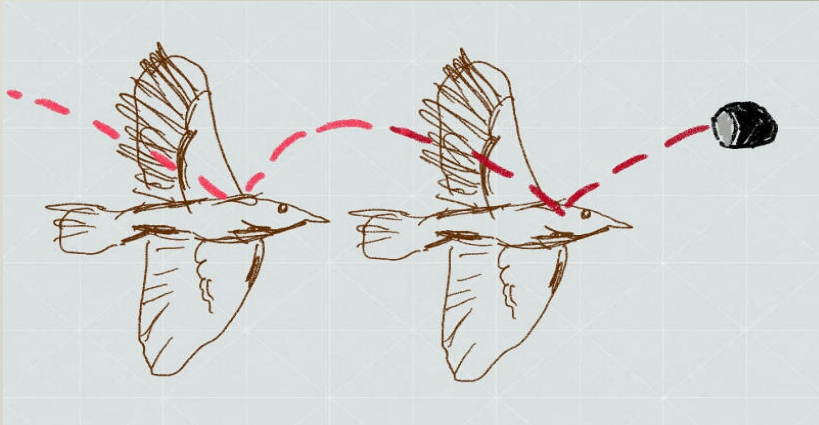


We are not ready yet!

W
e
a
r
e

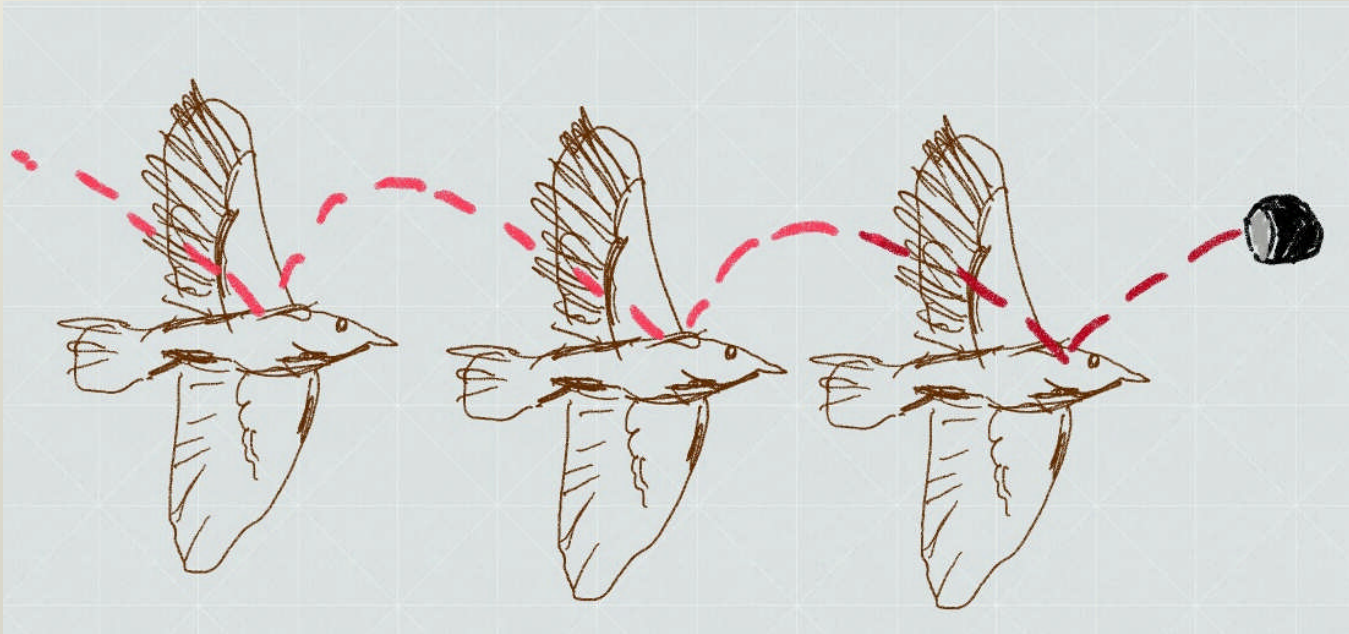


**List for planning
the work**



**List for planning
the work**

**Aid in identifying and
assessing risks**



**List for planning
the work**

Query for guidance

**Aid in identifying and
assessing risks**

- **Identify gaps in competencies**
- **Support tool building**
- **Structure for communication**
- **Template for teaching software engineering**
- **and other.**

**All this can be done in
a holistic, simple,
lightweight,
non-prescriptive and
method-agnostic
fashion**

Future

- ESSENCE is an excellent tool for squeezing software engineering education within a short period of time, even on an undergraduate level.
- Continue using ESSENCE within the education.
- Continue to develop educational material.

*Anybody interested in
cooperating with us?*



Competencies



Innovates	★5
Adapts	★4
Masters	★3
Applies	★2
Assists	★1

View of key competencies needed in software engineering