Essence – Kernel and Language

for Software Engineering Methods

------ Mira Kajko-Mattsson , Görkem Giray

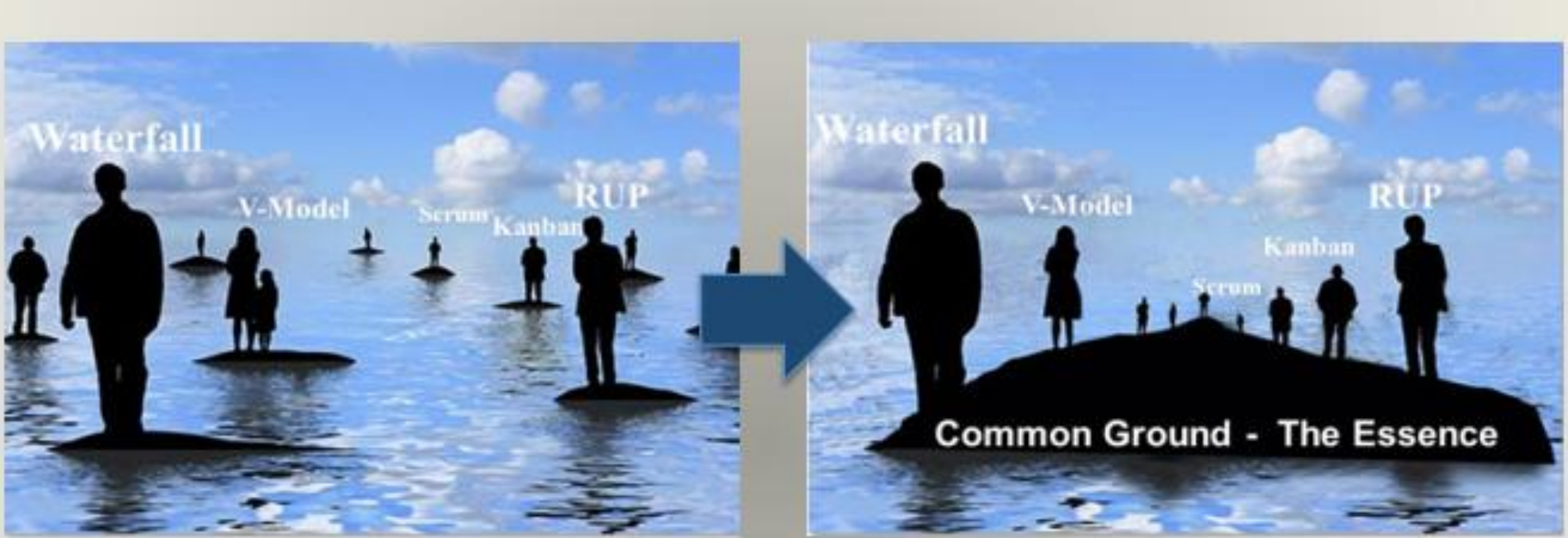
Content

1. What is Essence?
2. Essence Kernel
   - ① Alphas
   - ② Activity Space
   - ③ Competency
3. Essence Language
4. How does the Essence Kernel Work?

OMG®

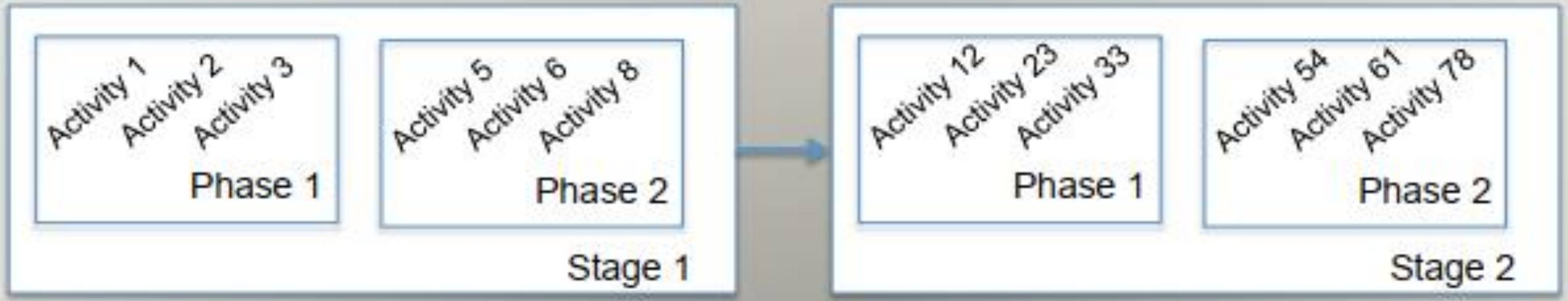
OBJECT MANAGEMENT GROUP®

Common Ground

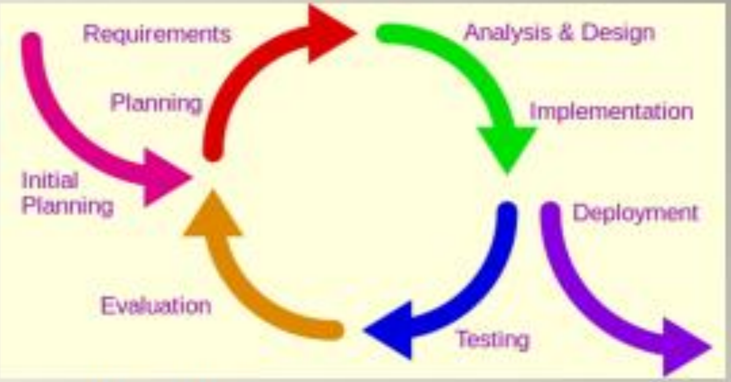


Everyone of us knows how to develop **our own** software, but as a community we have **no** widely accepted common ground

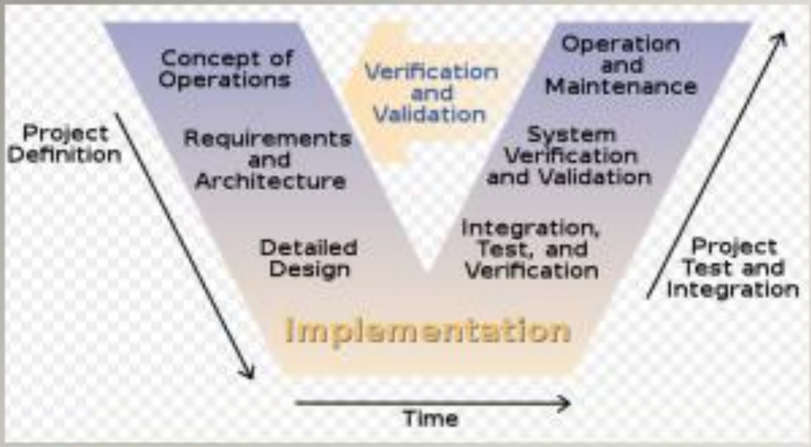
Software development methods today



Sequential



Iterative


Concept of Operations
Verification and Validation
Operation and Maintenance
Project Definition
Requirements and Architecture
System Verification and Validation
Detailed Design
Integration, Test, and Verification
Project Test and Integration
Implementation
Time

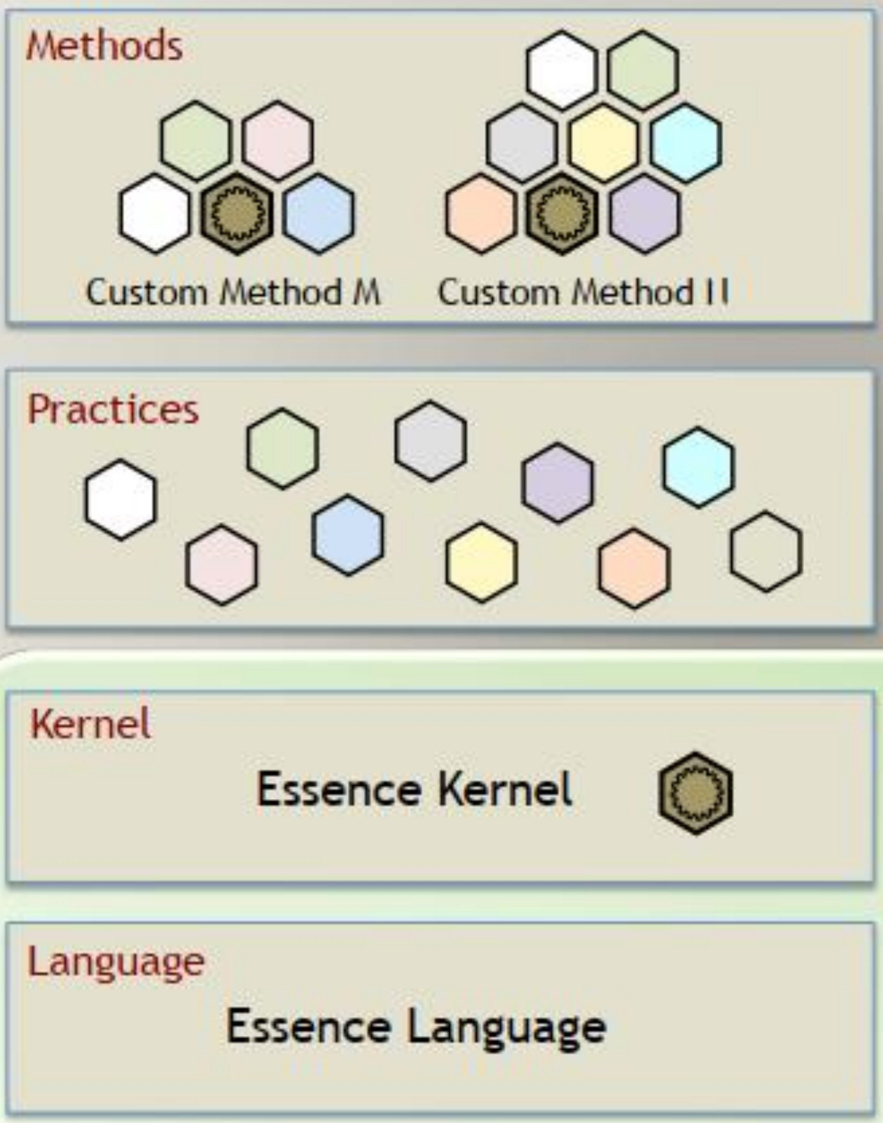
V-Model

Focus on activities in two essential things:
- Way of working
- Work

Measures

Find a kernel of widely agreed elements

What is Essence?



Methods

Custom Method M

Custom Method II

Practices

Kernel

Essence Kernel

Language

Essence Language

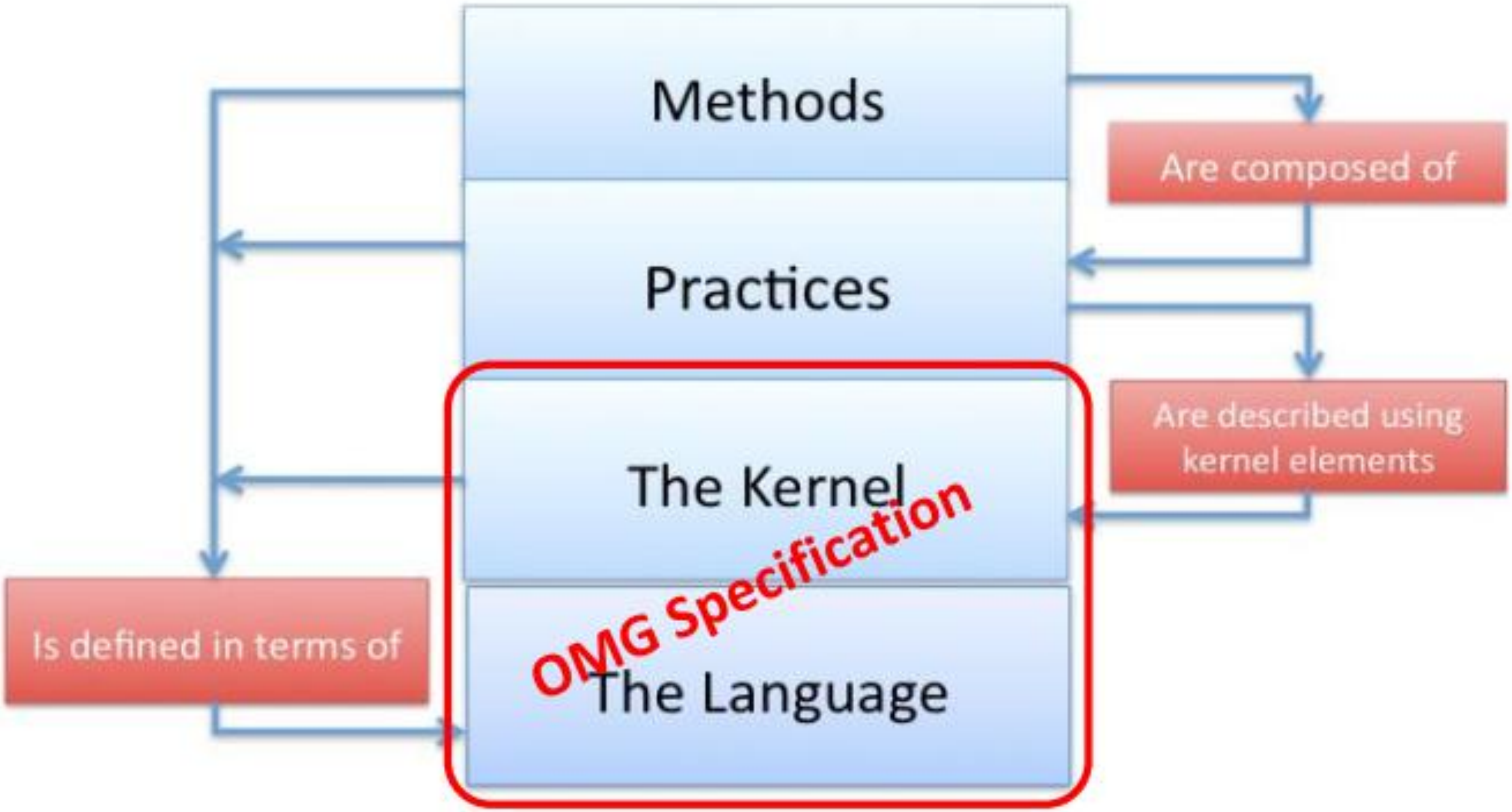
Method Architecture

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

OBJECT MANAGEMENT GROUP

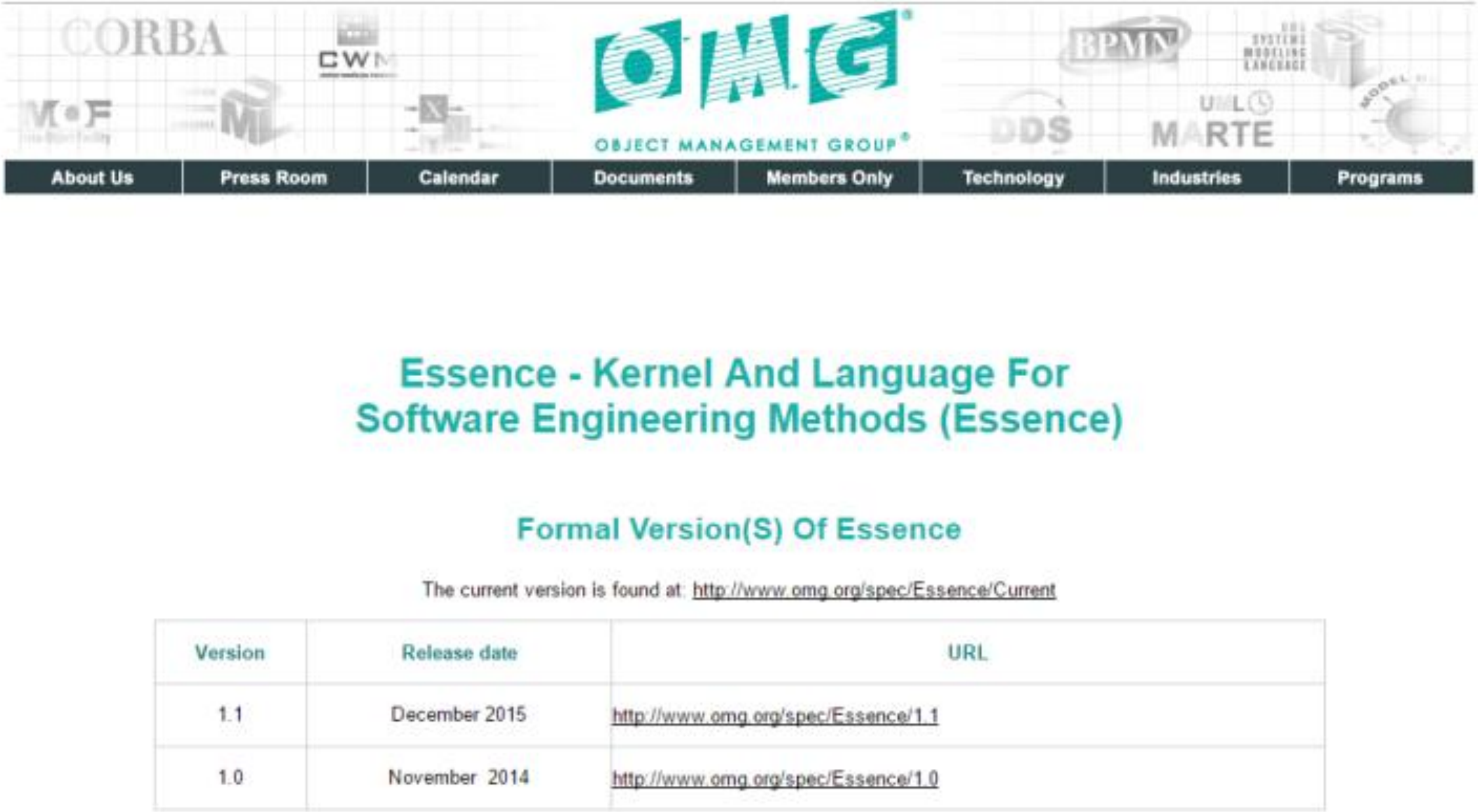
Essence - Kernel and Language for Software Engineering Methods
http://www.omg.org/spec/Essence/Current

Method Architecture



Ref: Object Management Group, "Essence - Kernel and Language for Software Engineering Methods," *OMG*, no. Version 1.1, 2015.

Essence OMG Specification



Essence - Kernel And Language For Software Engineering Methods (Essence)

Formal Version(S) Of Essence

The current version is found at: http://www.omg.org/spec/Essence/Current

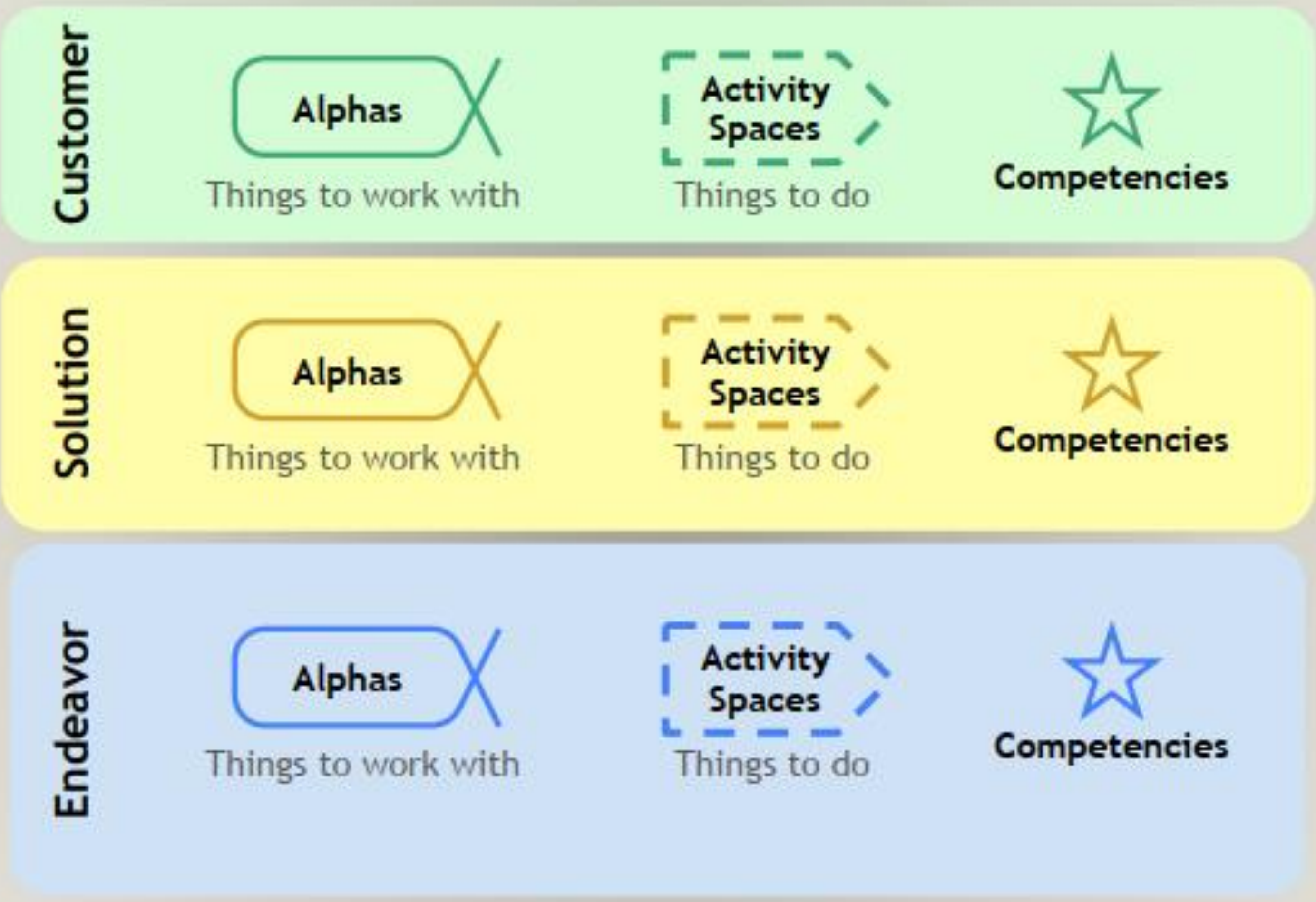
Version	Release date	URL
1.1	December 2015	http://www.omg.org/spec/Essence/1.1
1.0	November 2014	http://www.omg.org/spec/Essence/1.0

Ref: http://www.omg.org/spec/Essence/

Content

1. What is Essence?
2. Essence Kernel
   ① Alphas
   ② Activity Space
   ③ Competency
3. Essence Language
4. How does the Essence Kernel Work?

Essence Kernel



Customer

Alphas

Things to work with

Activity Spaces

Things to do

Competencies

Solution

Alphas

Things to work with

Activity Spaces

Things to do

Competencies

Endeavor

Alphas

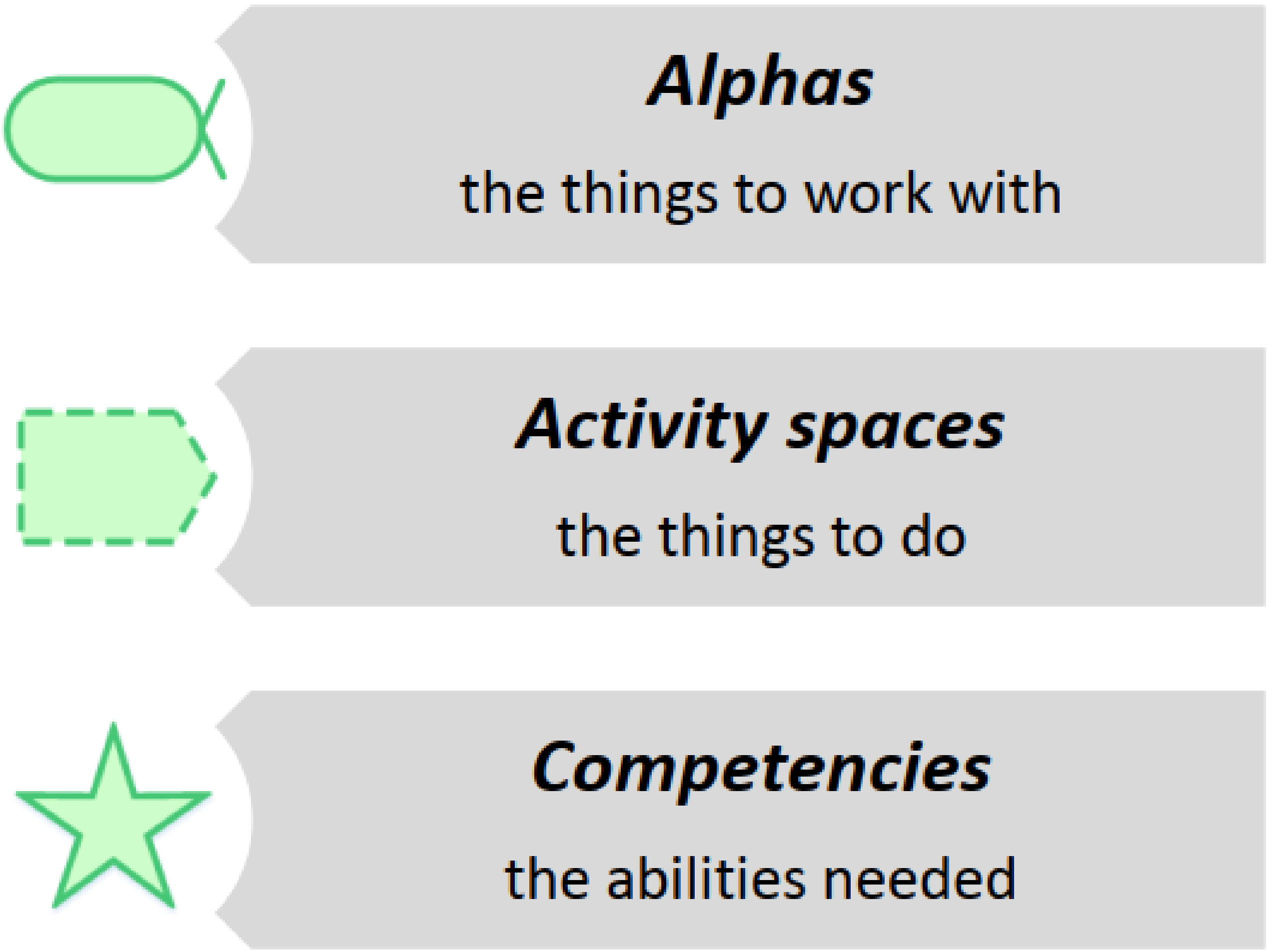
Things to work with

Activity Spaces

Things to do

Competencies

Essence Kernel



Alphas

the things to work with

Activity spaces

the things to do

Competencies

the abilities needed

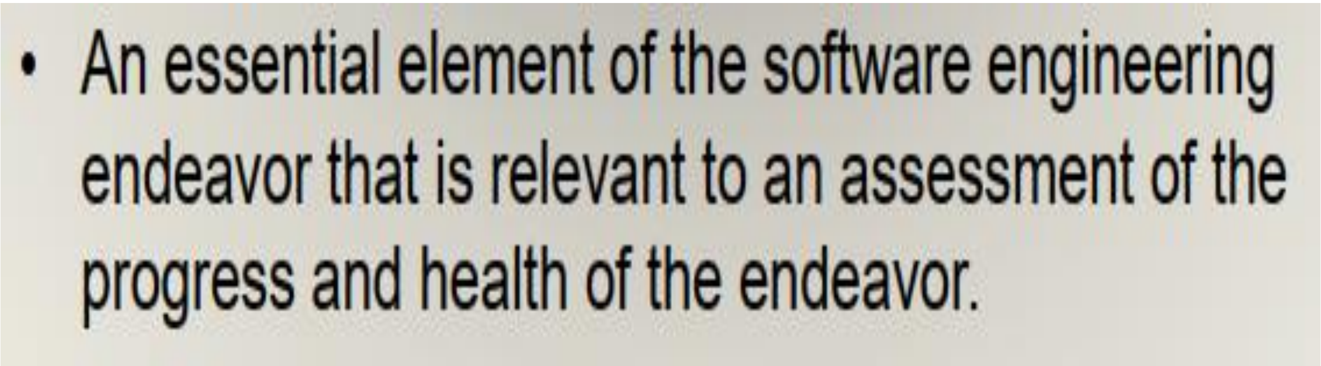
Content

1. What is Essence?
2. Essence Kernel
   - ① Alphas
   - ② Activity Space
   - ③ Competency
3. Essence Language
4. How does the Essence Kernel Work?

OBJECT MANAGEMENT GROUP®

Alpha

- is the acronym of **A**bstract-**L**evel **P**rogress **H**ealth **A**ttribute
- are subjects whose evolution we want to understand, monitor, direct, and control
- shows a progression towards achieving the objectives through its states

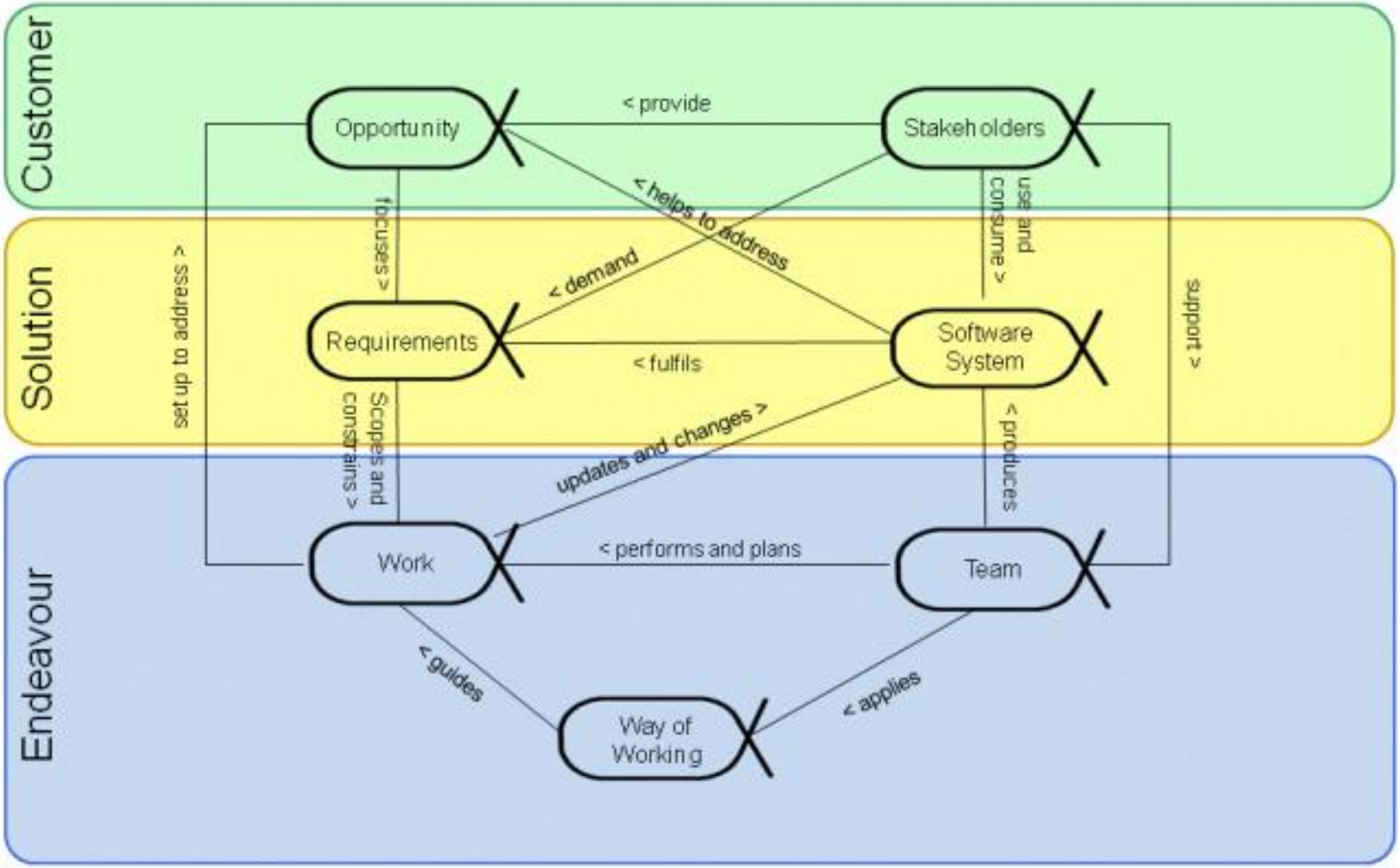


- An essential element of the software engineering endeavor that is relevant to an assessment of the progress and health of the endeavor.

α

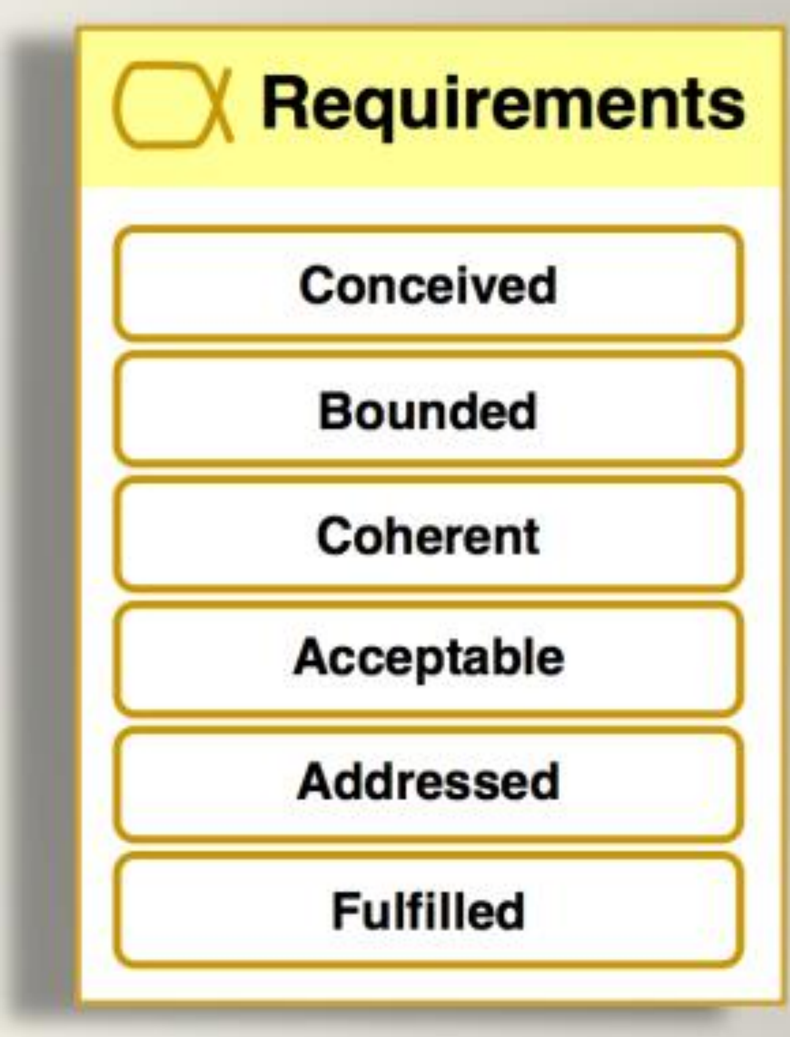
Alphas

The Essential Things to Work With



Ref: Object Management Group, "Essence - Kernel and Language for Software Engineering Methods," *OMG*, no. Version 1.1, 2015.

Peeking into the Alphas



Requirements

Conceived

Bounded

Coherent

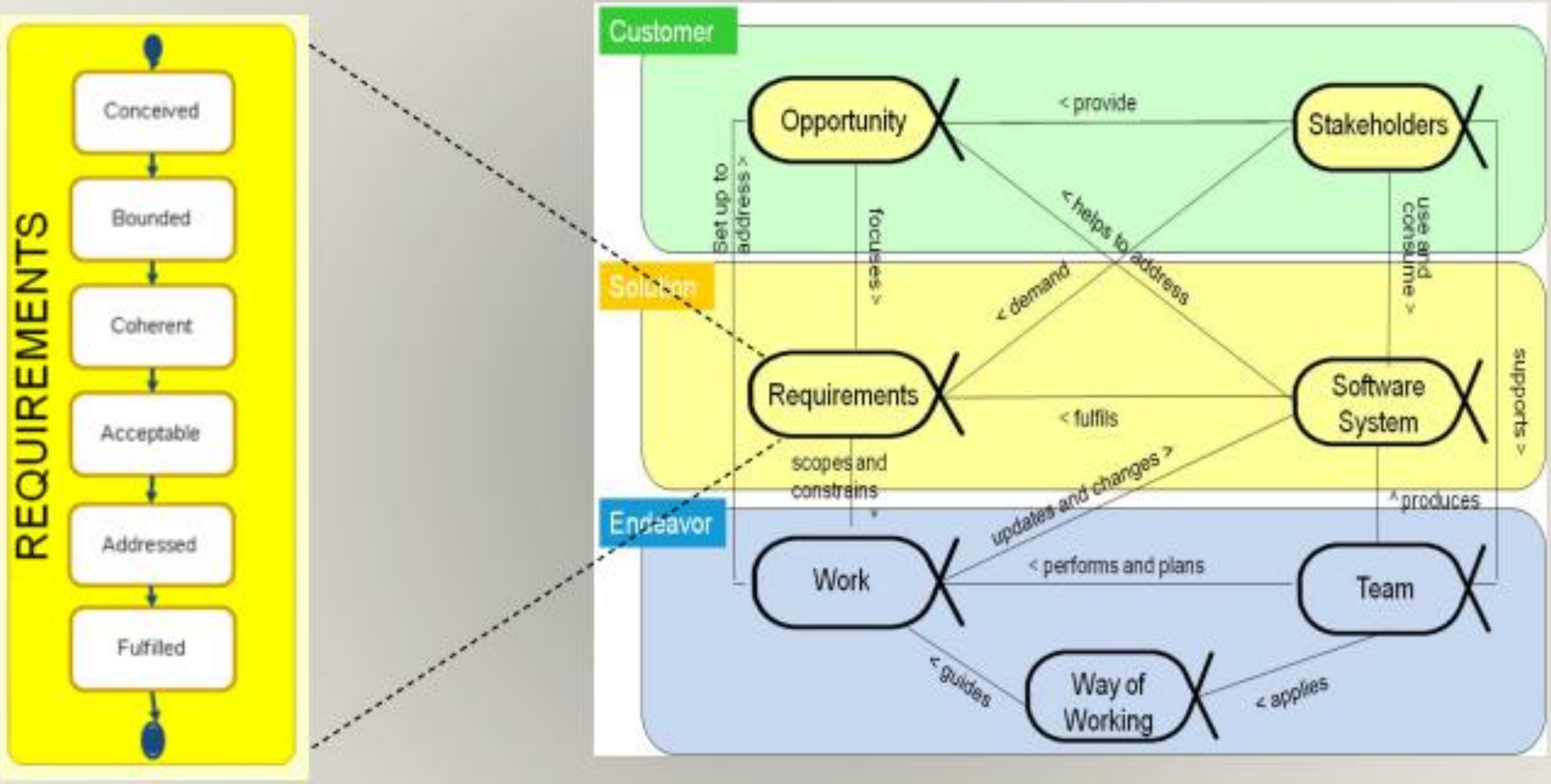
Acceptable

Addressed

Fulfilled

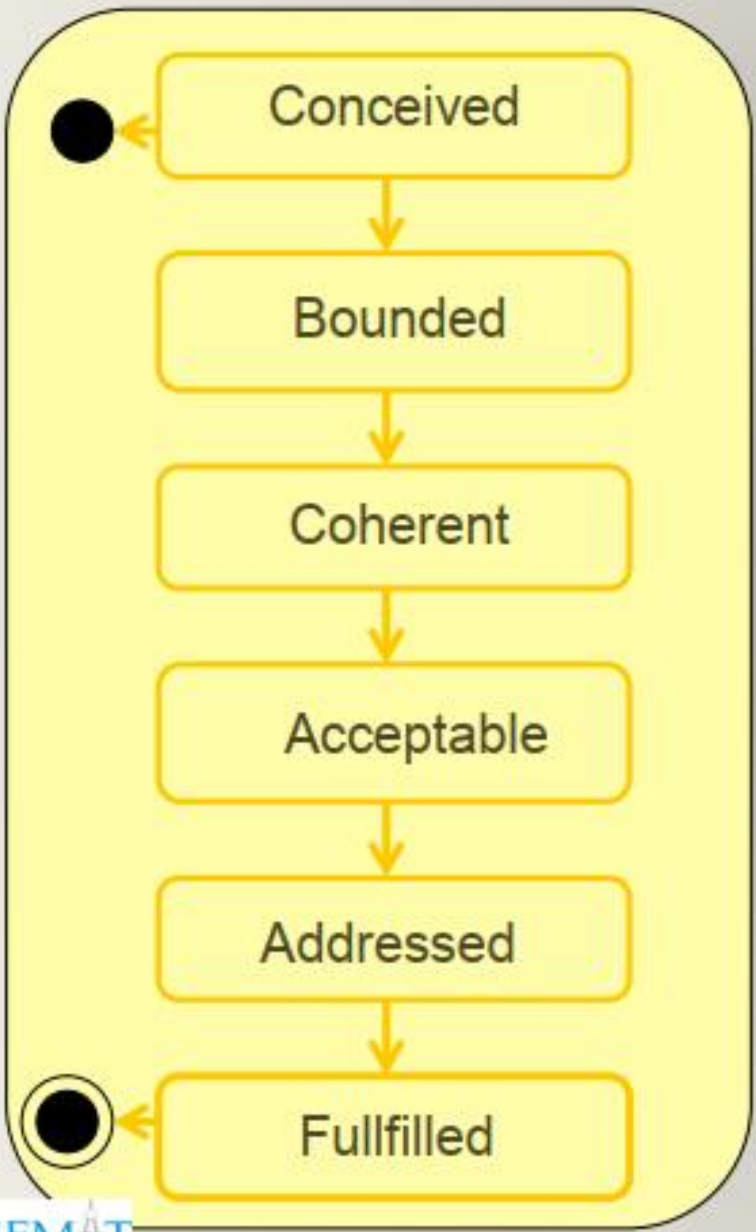
- There are several cards for each Alpha. What does each cards stand for?
- What is included in each card?

Requirements- one of the Alphas



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

Requirements states



The need for a new system has been agreed.

The purpose and theme of the new system are clear.

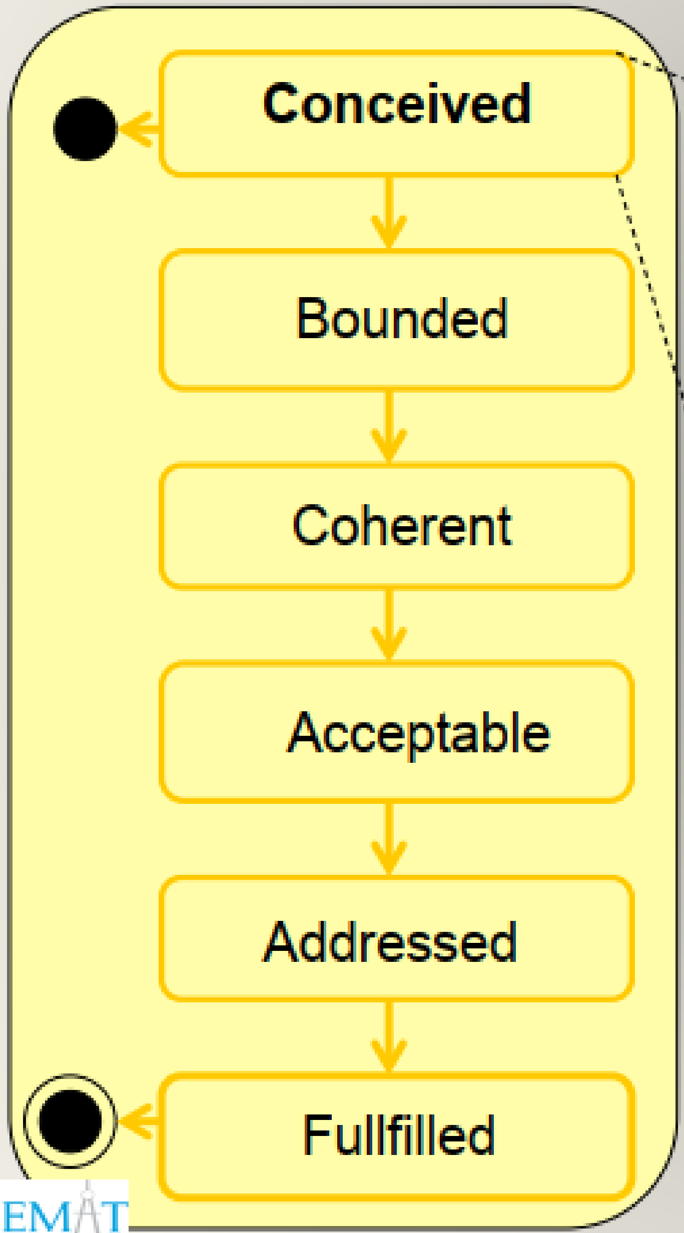
The requirements provide a coherent description of the essential characteristics of the new system.

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 requirements states



Conceived

Bounded

Coherent

Acceptable

Addressed

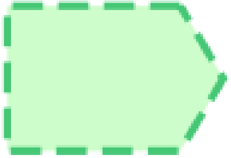
Fullfilled

- ❑ 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.

Content

1. What is Essence?
2. Essence Kernel
   ① Alphas
   ② Activity Space
   ③ Competency
3. Essence Language
4. How does the Essence Kernel Work?

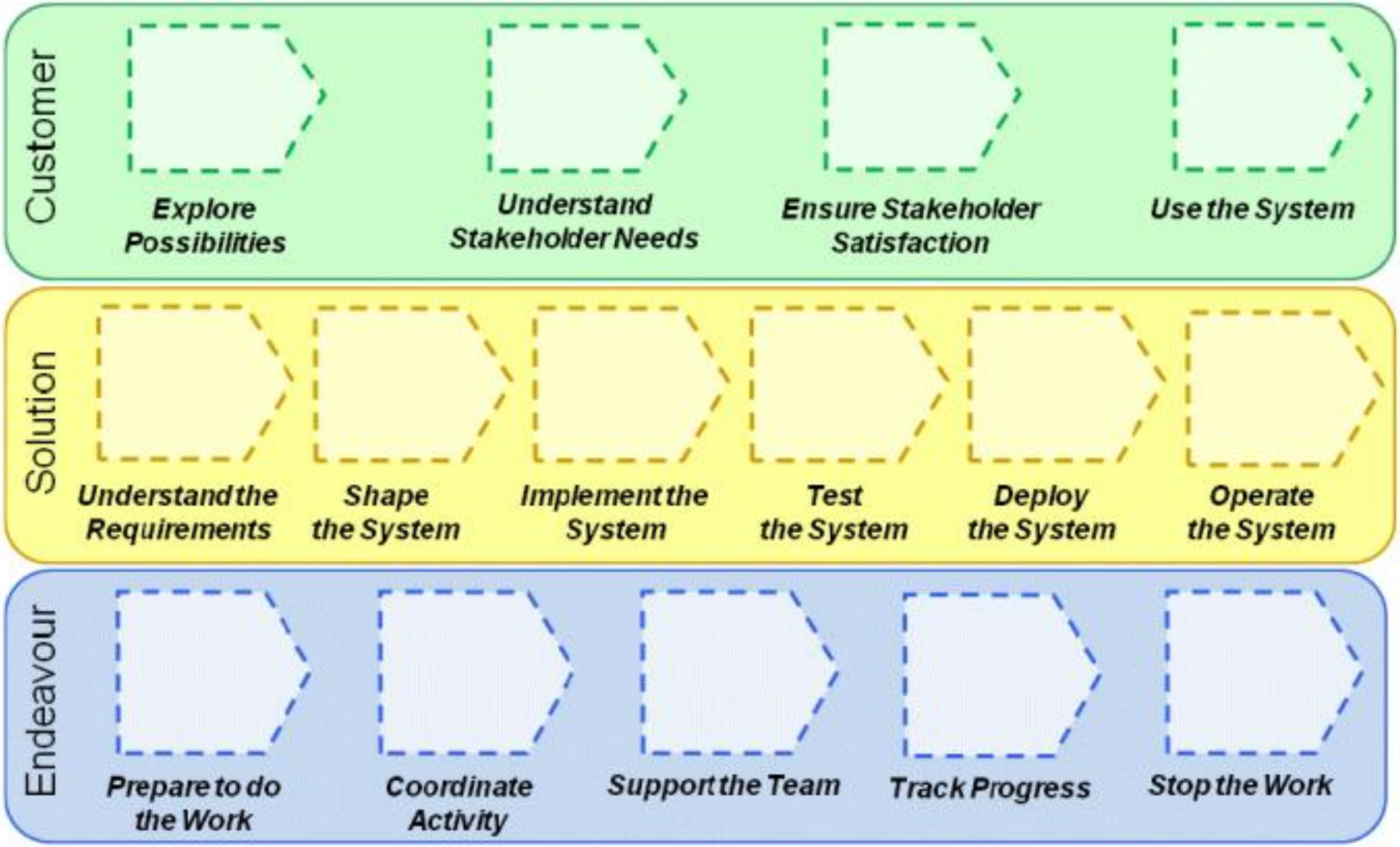
Activity Space



- is a high-level abstraction representing "something to be done"
- are containers for activities
- has completion criteria expressed in terms of states the output alphas should have reached

Activity Spaces

The Essential Things to Do



Ref: Object Management Group, "Essence - Kernel and Language for Software Engineering Methods," *OMG*, no. Version 1.1, 2015.

An Example Activity Space

Understand Stakeholder Needs

Description

Engage with the stakeholders to understand their needs and ensure that the right results are produced. This includes identifying and working with the stakeholder representatives to progress the opportunity.

Understand stakeholder needs to:

- Ensure the right solution is created.
- Align expectations.
- Collect feedback and generate input.
- Ensure that the solution produced provides benefit to the stakeholders.

Entry Criteria	Stakeholders::Recognized, Opportunity::Value Established	**Completion Criteria**	Stakeholders::In Agreement, Opportunity::Viable

Ref: Object Management Group, "Essence - Kernel and Language for Software Engineering Methods," *OMG*, no. Version 1.1, 2015.

Content

1. What is Essence?
2. Essence Kernel

    ① Alphas

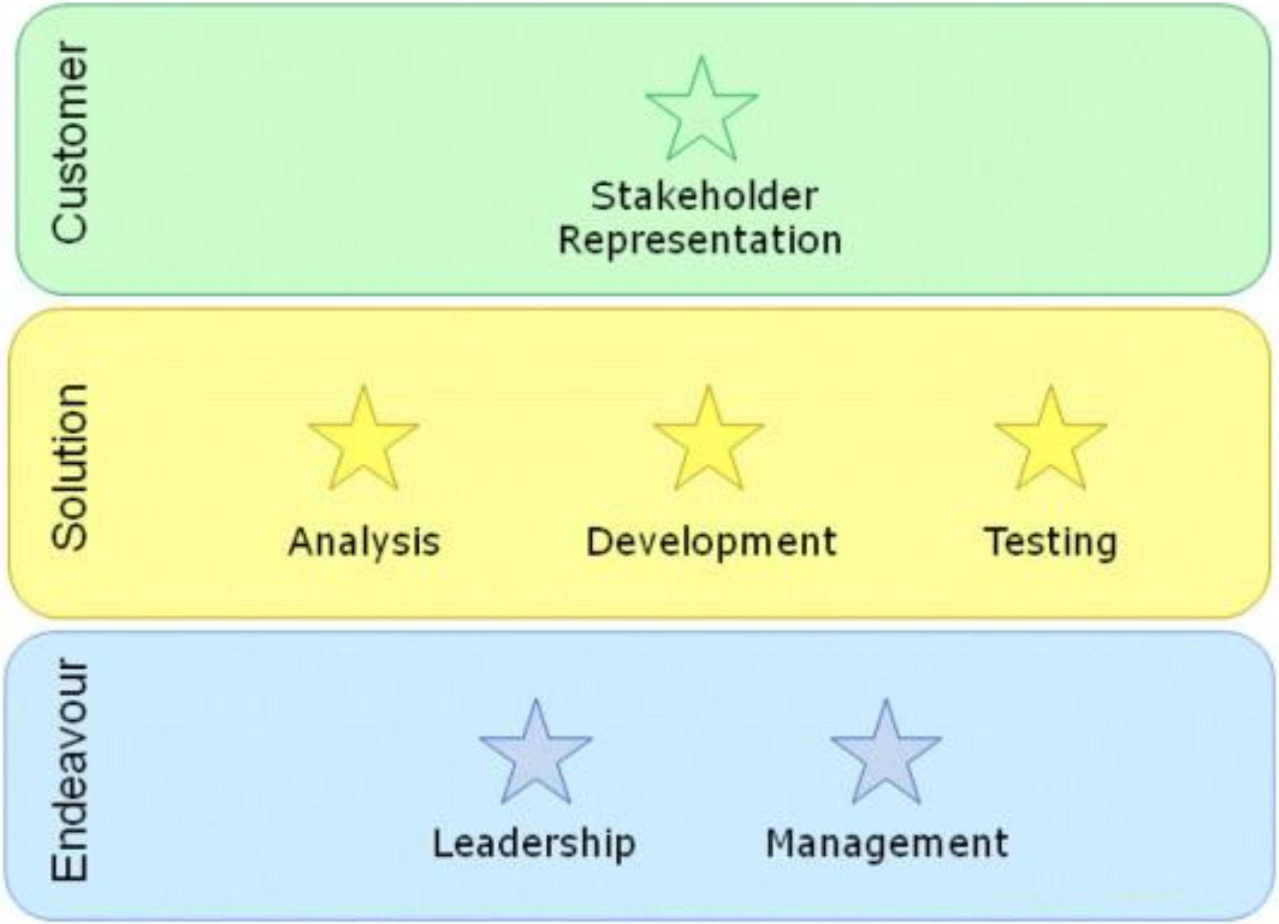
    ② Activity Space

    ③ Competency

3. Essence Language
4. How does the Essence Kernel Work?

Competency

The Abilities Needed



Ref: Object Management Group, "Essence - Kernel and Language for Software Engineering Methods," *OMG*, no. Version 1.1, 2015.

Content

1. What is Essence?
2. Essence Kernel

   ① Alphas

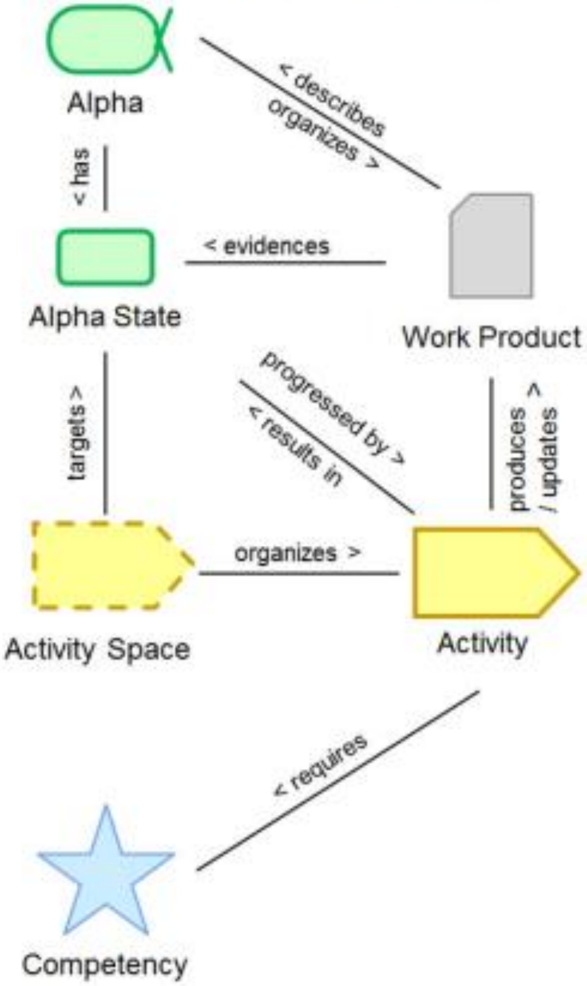
   ② Activity Space

   ③ Competency

3. Essence Language
4. How does the Essence Kernel Work?

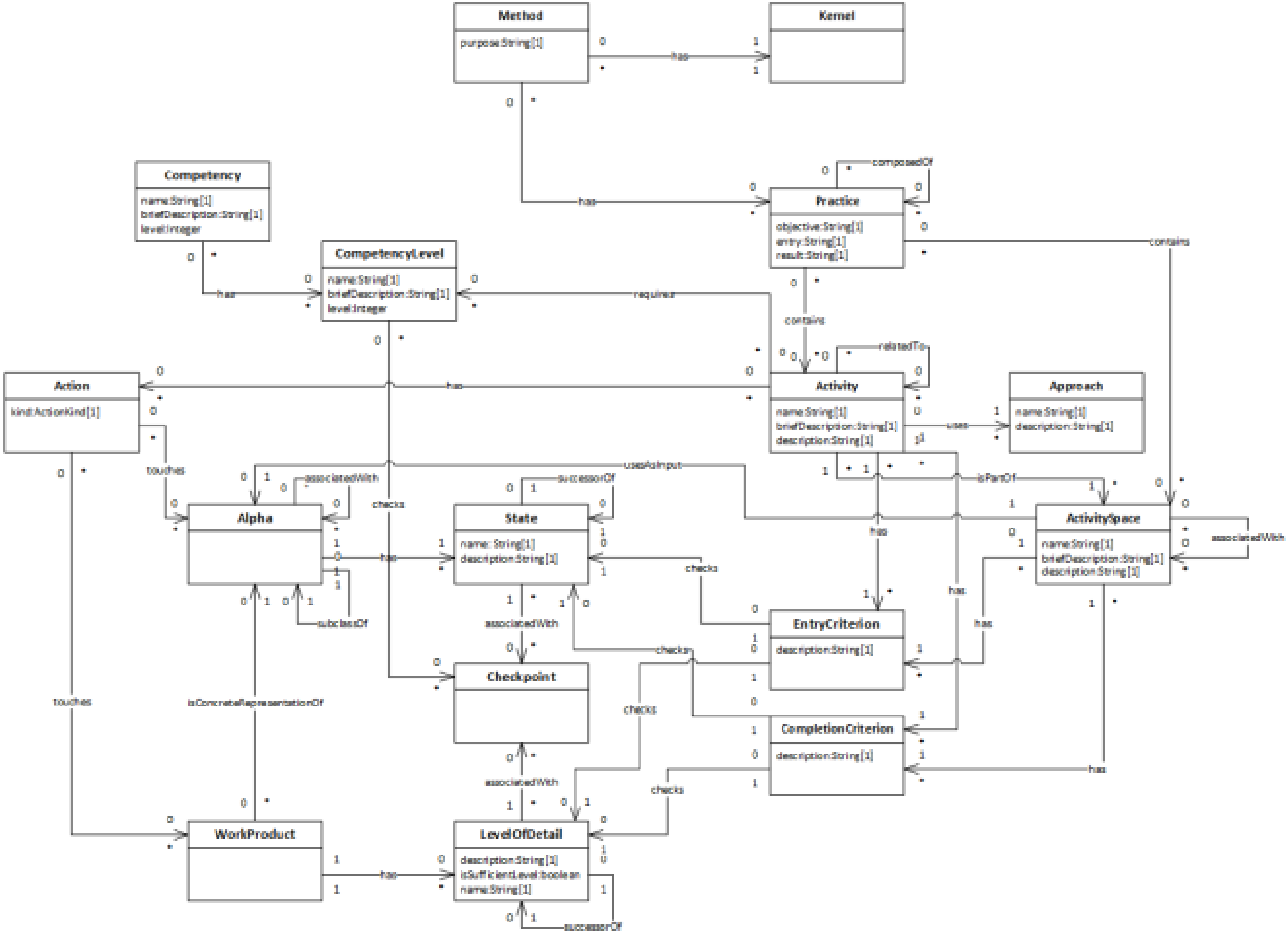
OBJECT MANAGEMENT GROUP®

Conceptual Overview of Essence Language



Ref: Object Management Group, "Essence - Kernel and Language for Software Engineering Methods," *OMG*, no. Version 1.1, 2015.

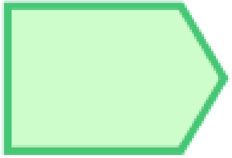
Essence Language



Work Product

- enables to represent Alphas concretely
- can be of many different types such as models, documents, specifications, code, tests, executable, spreadsheets, as well as other types of artifacts

Activity



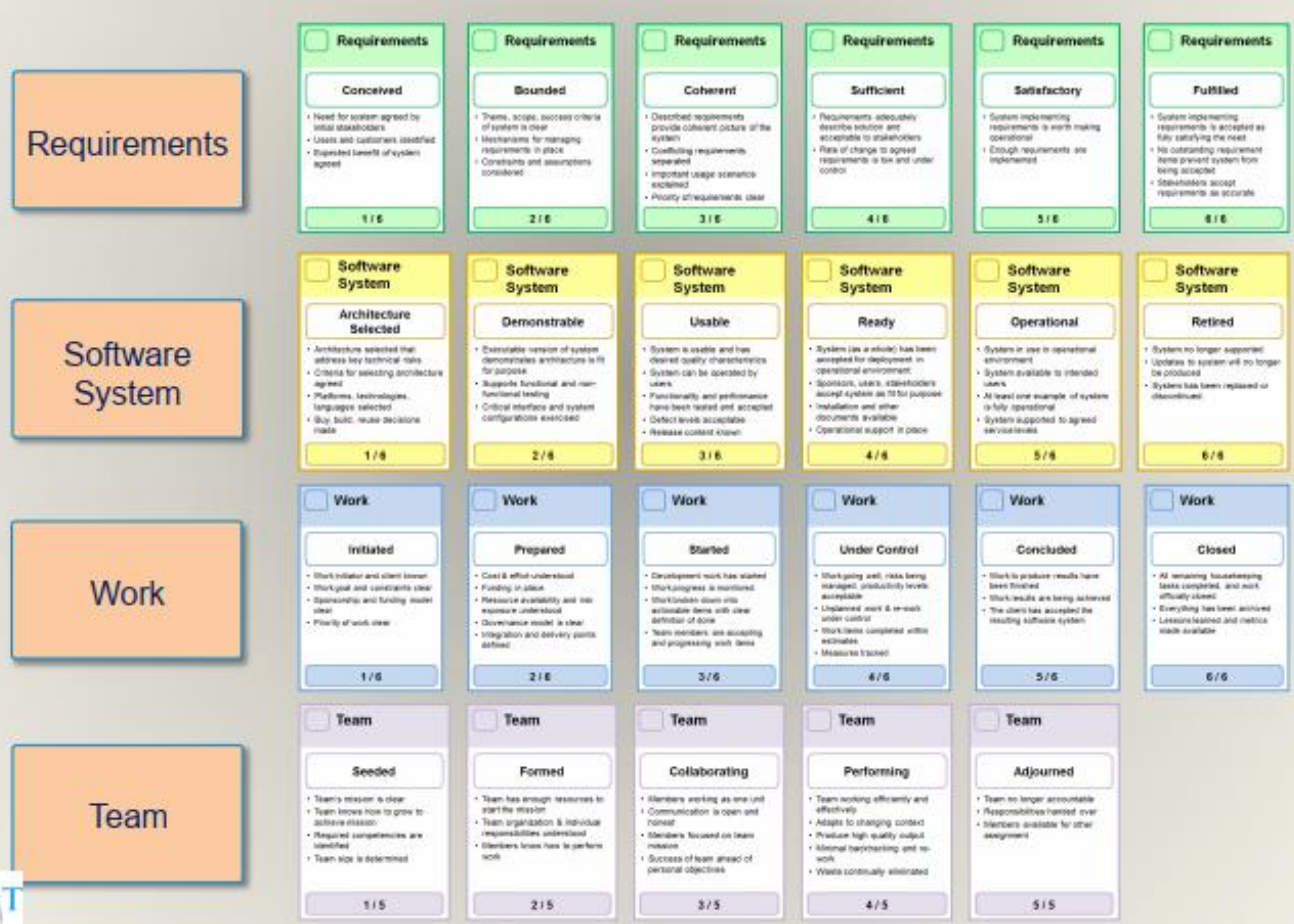
- describes some work to be performed
- can recommend to perform actions on alphas and/or work products
- can belong to one or more Activity Space(s)

Content

1. What is Essence?
2. Essence Kernel
   ① Alphas
   ② Activity Space
   ③ Competency
3. Essence Language
4. How does the Essence Kernel Work?

OBJECT MANAGEMENT GROUP®

Four of the seven essential things



Requirements

□ Requirements

Conceived

- Need for system agreed by initial stakeholders
- Users and customers identified
- Expected benefit of system agreed

1 / 6

□ Requirements

Bounded

- Theme, scope, success criteria of system is clear
- Mechanisms for managing requirements in place
- Constraints and assumptions considered

2 / 6

□ Requirements

Coherent

- Described requirements provide coherent picture of the system
- Conflicting requirements separated
- Important usage scenarios explained
- Priority of requirements clear

3 / 6

□ Requirements

Sufficient

- Requirements adequately describe solution and acceptable to stakeholders
- Rate of change to agreed requirements is low and under control

4 / 6

□ Requirements

Satisfactory

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

5 / 6

□ Requirements

Fulfilled

- System implementing requirements is accepted as fully satisfying the need
- No outstanding requirement items prevent system from being accepted
- Stakeholders accept requirements as accurate

6 / 6

Software System

□ Software System

Architecture Selected

- Architecture selected that address key technical risks
- Criteria for selecting architecture agreed
- Platforms, technologies, languages selected
- Buy, build, reuse decisions made

1 / 6

□ Software System

Demonstrable

- Executable version of system demonstrates architecture is fit for purpose
- Supports functional and non-functional testing
- Critical interface and system configurations exercised

2 / 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

□ Software System

Ready

- 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

4 / 6

□ Software System

Operational

- 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

5 / 6

□ Software System

Retired

- System no longer supported
- Updates to system will no longer be produced
- System has been replaced or discontinued

6 / 6

Work

□ Work

Initiated

- Work initiator and client known
- Work goal and constraints clear
- Sponsorship and funding model clear
- Priority of work clear

1 / 6

□ Work

Prepared

- Cost & effort understood
- Funding in place
- Resource availability and risk exposure understood
- Governance model is clear
- Integration and delivery points defined

2 / 6

□ Work

Started

- Development work has started
- Work progress is monitored
- Work broken down into actionable items with clear definition of done
- Team members are accepting and progressing work items

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

□ Work

Concluded

- Work to produce results have been finished
- Work results are being achieved
- The client has accepted the resulting software system

5 / 6

□ Work

Closed

- All remaining housekeeping tasks completed, and work officially closed
- Everything has been archived
- Lessons learned and metrics made available

6 / 6

Team

□ Team

Seeded

- Team's mission is clear
- Team knows how to grow to achieve mission
- Required competencies are identified
- Team size is determined

1 / 5

□ Team

Formed

- Team has enough resources to start the mission
- Team organization & individual responsibilities understood
- Members know how to perform work

2 / 5

□ Team

Collaborating

- Members working as one unit
- Communication is open and honest
- Members focused on team mission
- Success of team ahead of personal objectives

3 / 5

□ 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

□ Team

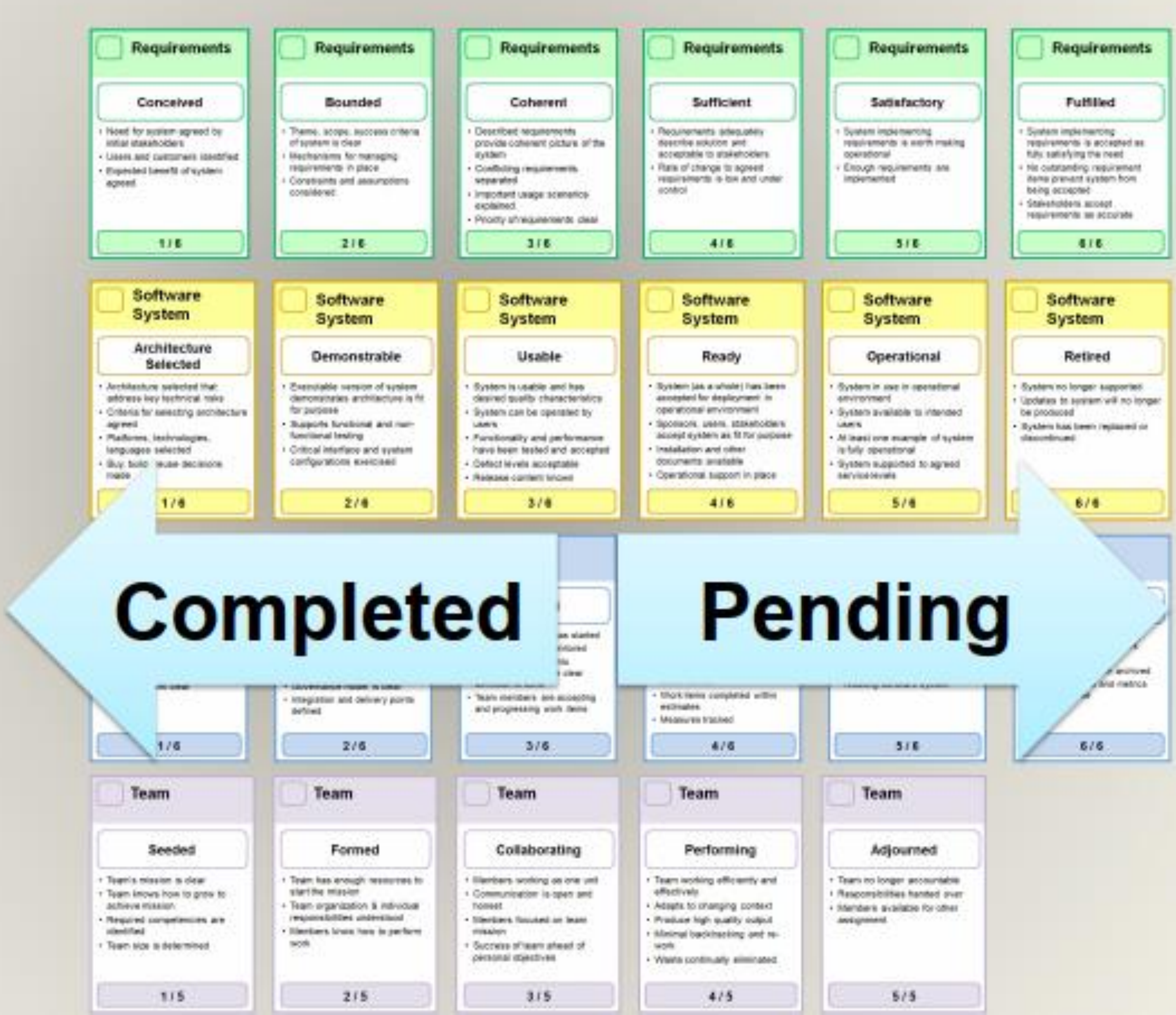
Adjourned

- Team no longer accountable
- Responsibilities handed over
- Members available for other assignment

5 / 5

SEMAT

Plan: Determine Current State



Completed

Pending

Requirements — **Conceived**
- Need for system agreed by initial stakeholders
- Users and customers identified
- Expected benefit of system agreed

1 / 6

Requirements — **Bounded**
- Theme, scope, success criteria of system is clear
- Mechanisms for managing requirements in place
- Constraints and assumptions considered

2 / 6

Requirements — **Coherent**
- Described requirements provide coherent picture of the system
- Conflicting requirements separated
- Important usage scenarios explained
- Priority of requirements clear

3 / 6

Requirements — **Sufficient**
- Requirements adequately describe solution and acceptable to stakeholders
- Rate of change to agreed requirements is low and under control

4 / 6

Requirements — **Satisfactory**
- System implementing requirements is worth making operational
- Enough requirements are implemented

5 / 6

Requirements — **Fulfilled**
- System implementing requirements is accepted as fully satisfying the need
- No outstanding requirement items prevent system from being accepted
- Stakeholders accept requirements as accurate

6 / 6

Software System — **Architecture Selected**
- Architecture selected that address key technical risks
- Criteria for selecting architecture agreed
- Platforms, technologies, languages selected
- Buy, build, reuse decisions made

1 / 6

Software System — **Demonstrable**
- Executable version of system demonstrates architecture is fit for purpose
- Supports functional and non-functional testing
- Critical interface and system configurations exercised

2 / 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

Software System — **Ready**
- 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

4 / 6

Software System — **Operational**
- 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

5 / 6

Software System — **Retired**
- System no longer supported
- Updates to system will no longer be produced
- System has been replaced or discontinued

6 / 6

1 / 6

- Governance model is clear
- Integration and delivery points defined

2 / 6

- Team members are accepting and progressing work items

3 / 6

- Work items completed within estimates
- Measures tracked

4 / 6

5 / 6

6 / 6

Team — **Seeded**
- Team's mission is clear
- Team knows how to grow to achieve mission
- Required competencies are identified
- Team size is determined

1 / 5

Team — **Formed**
- Team has enough resources to start the mission
- Team organization & individual responsibilities understood
- Members know how to perform work

2 / 5

Team — **Collaborating**
- Members working as one unit
- Communication is open and honest
- Members focused on team mission
- Success of team ahead of personal objectives

3 / 5

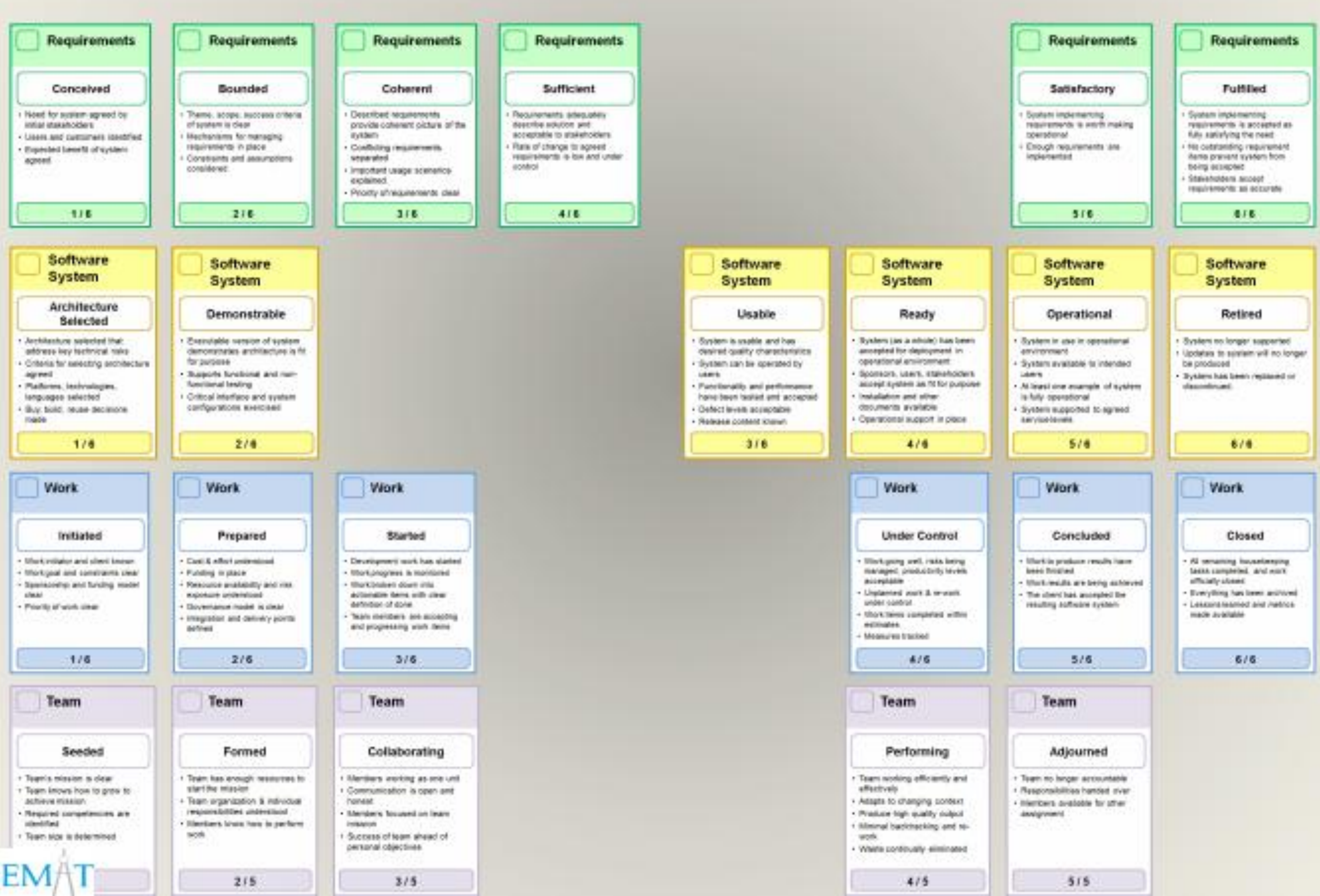
Team — **Performing**
- Team working efficiently and effectively
- Adapts to changing context
- Produce high quality output
- Minimal backtracking and re-work
- Waste continually eliminated

4 / 5

Team — **Adjourned**
- Team no longer accountable
- Responsibilities handed over
- Members available for other assignment

5 / 5

Plan: Determine Next State



□ Requirements

Conceived

- Need for system agreed by initial stakeholders
- Users and customers identified
- Expected benefit of system agreed

1 / 6

□ Requirements

Bounded

- Theme, scope, success criteria of system is clear
- Mechanisms for managing requirements in place
- Constraints and assumptions considered

2 / 6

□ Requirements

Coherent

- Described requirements provide coherent picture of the system
- Conflicting requirements separated
- Important usage scenarios explained
- Priority of requirements clear

3 / 6

□ Requirements

Sufficient

- Requirements adequately describe solution and acceptable to stakeholders
- Rate of change to agreed requirements is low and under control

4 / 6

□ Requirements

Satisfactory

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

5 / 6

□ Requirements

Fulfilled

- System implementing requirements is accepted as fully satisfying the need
- No outstanding requirement items prevent system from being accepted
- Stakeholders accept requirements as accurate

6 / 6

□ Software System

Architecture Selected

- Architecture selected that address key technical risks
- Criteria for selecting architecture agreed
- Platforms, technologies, languages selected
- Buy, build, reuse decisions made

1 / 6

□ Software System

Demonstrable

- Executable version of system demonstrates architecture is fit for purpose
- Supports functional and non-functional testing
- Critical interface and system configurations exercised

2 / 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

□ Software System

Ready

- 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

4 / 6

□ Software System

Operational

- 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

5 / 6

□ Software System

Retired

- System no longer supported
- Updates to system will no longer be produced
- System has been replaced or discontinued

6 / 6

□ Work

Initiated

- Work initiator and client known
- Work goal and constraints clear
- Sponsorship and funding model clear
- Priority of work clear

1 / 6

□ Work

Prepared

- Cost & effort understood
- Funding in place
- Resource availability and risk exposure understood
- Governance model is clear
- Integration and delivery points defined

2 / 6

□ Work

Started

- Development work has started
- Work progress is monitored
- Work broken down into actionable items with clear definition of done
- Team members are accepting and progressing work items

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

□ Work

Concluded

- Work to produce results have been finished
- Work results are being achieved
- The client has accepted the resulting software system

5 / 6

□ Work

Closed

- All remaining housekeeping tasks completed, and work officially closed
- Everything has been archived
- Lessons learned and metrics made available

6 / 6

□ Team

Seeded

- Team's mission is clear
- Team knows how to grow to achieve mission
- Required competencies are identified
- Team size is determined

□ Team

Formed

- Team has enough resources to start the mission
- Team organization & individual responsibilities understood
- Members know how to perform work

2 / 5

□ Team

Collaborating

- Members working as one unit
- Communication is open and honest
- Members focused on team mission
- Success of team ahead of personal objectives

3 / 5

□ 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

□ Team

Adjourned

- Team no longer accountable
- Responsibilities handed over
- Members available for other assignment

5 / 5

Plan: 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

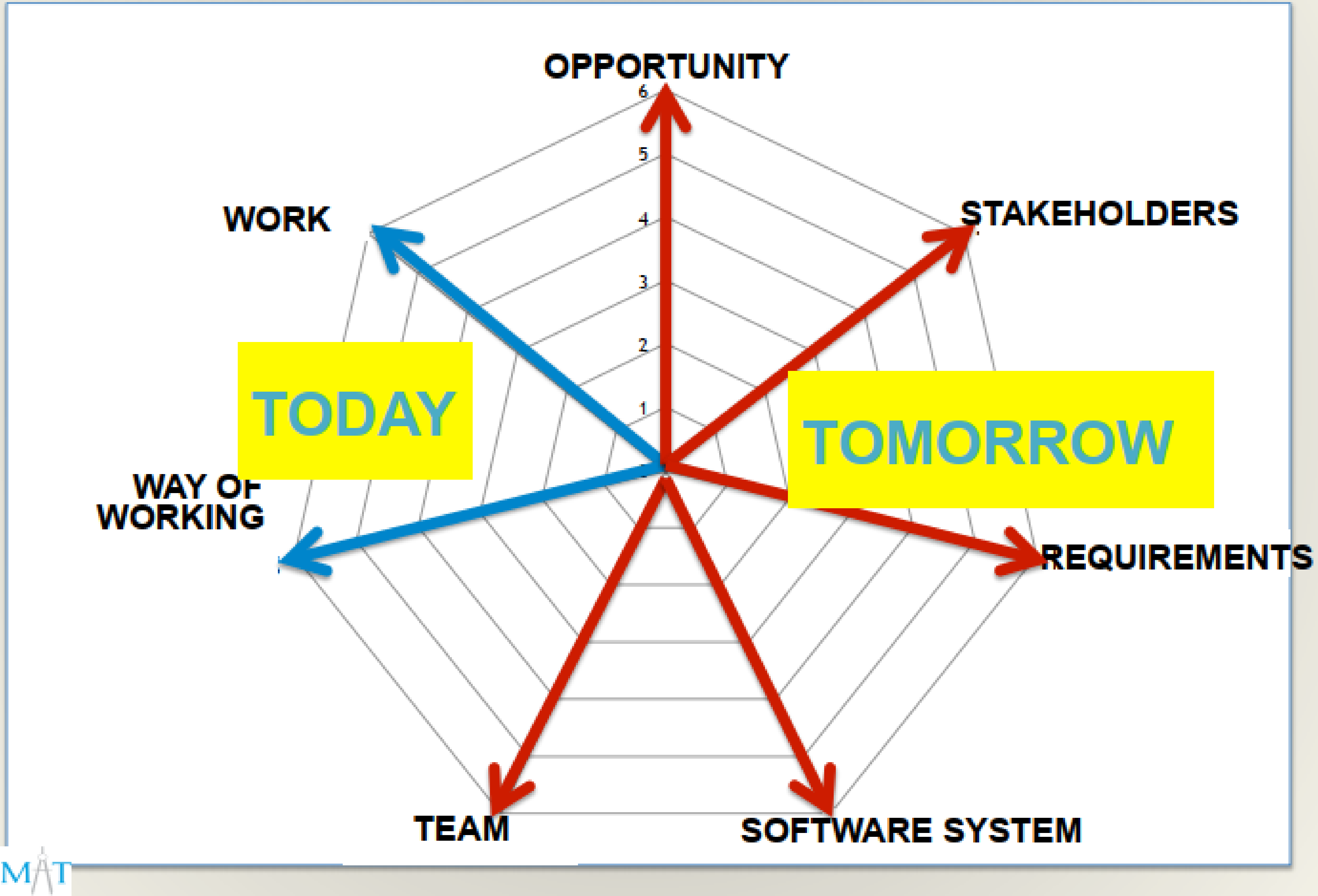
□ **Team**

Performing

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

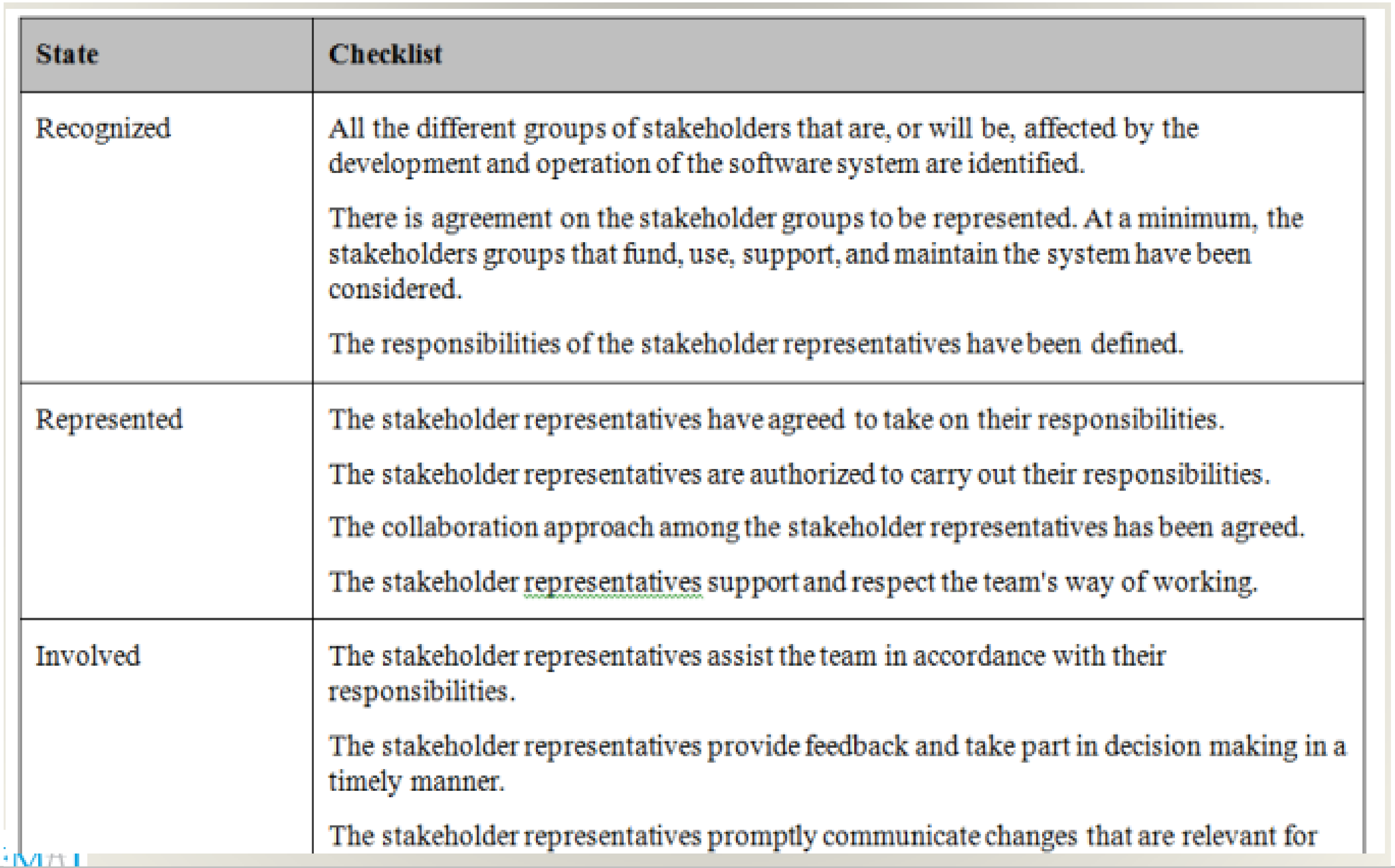
4 / 5

Following essential things



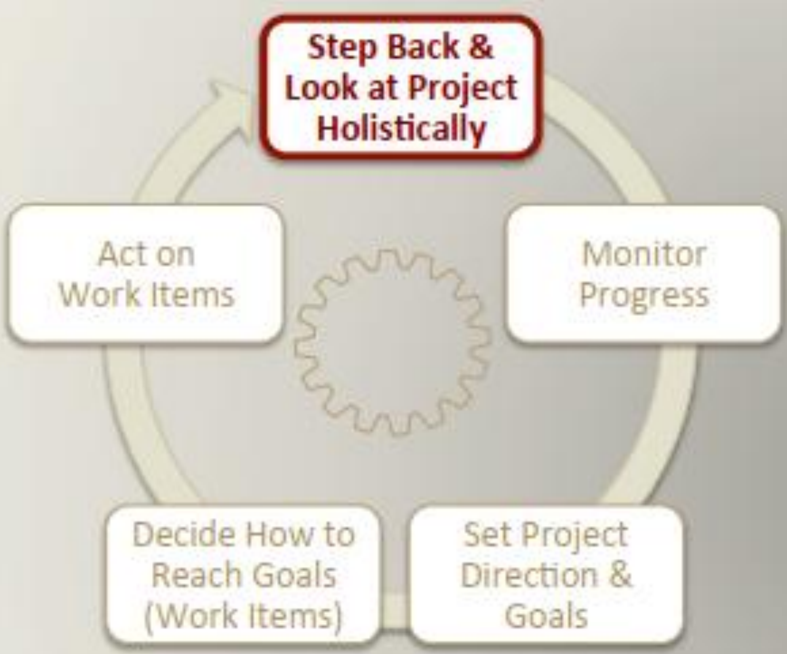
SEMAT

Use Check Lists or Use cards

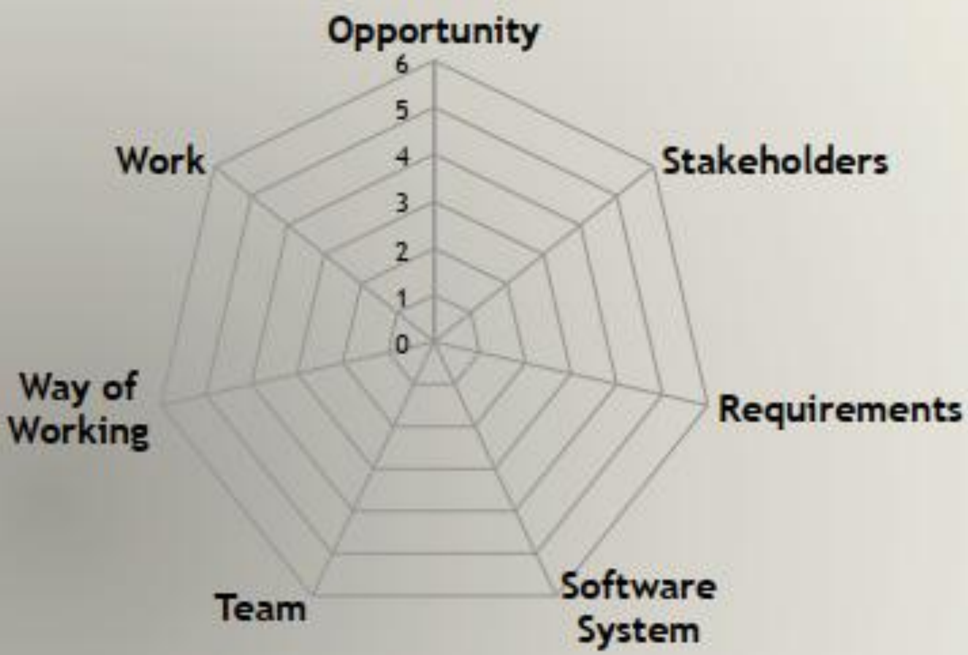


State	Checklist
Recognized	All the different groups of stakeholders that are, or will be, affected by the development and operation of the software system are identified. There is agreement on the stakeholder groups to be represented. At a minimum, the stakeholders groups that fund, use, support, and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

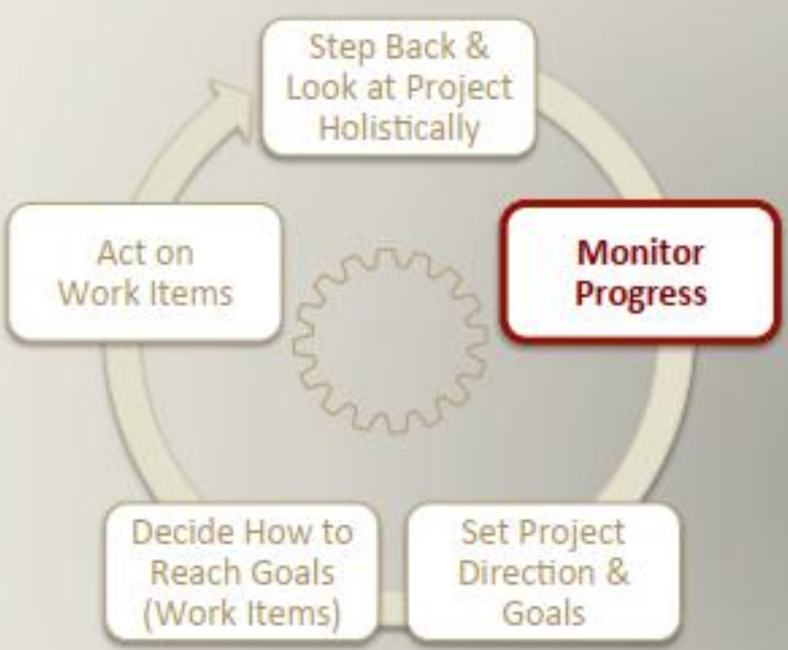
How does the Essence Kernel Work?



Step Back & Look at Project Holistically
Monitor Progress
Set Project Direction & Goals
Decide How to Reach Goals (Work Items)
Act on Work Items



How does the Essence Kernel Work?



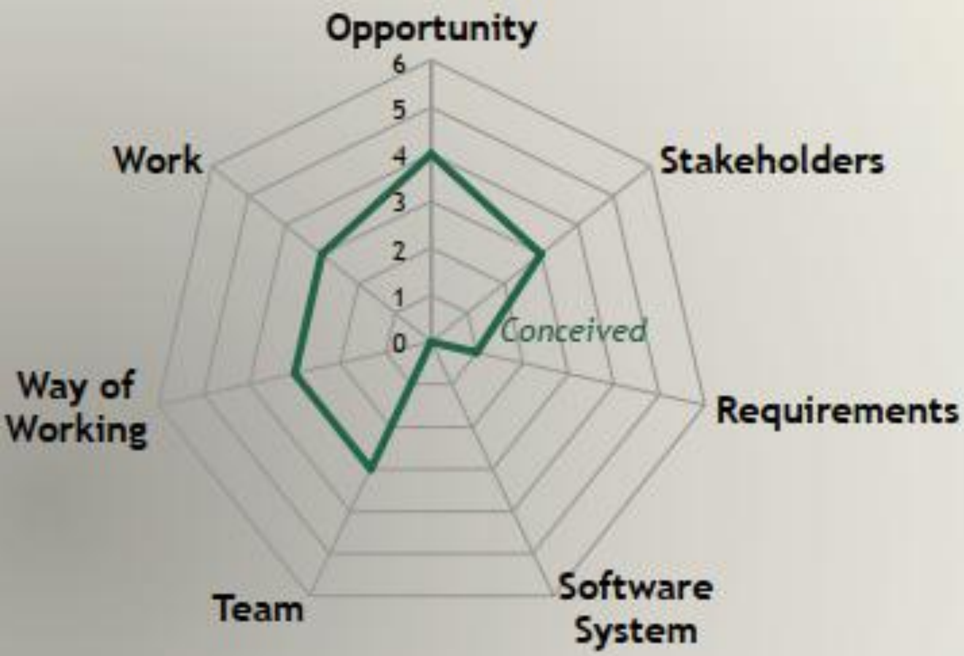
Step Back & Look at Project Holistically

Monitor Progress

Set Project Direction & Goals

Decide How to Reach Goals (Work Items)

Act on Work Items



Current State

Requirements

Conceived

- The need for a new system is clear
- Users are identified
- Initial sponsors are identified

1 / 6

Requirements

Bounded

- The purpose and extent of the system are agreed
- Success criteria are clear
- Mechanisms for handling requirements are agreed
- Constraints and assumptions identified

2 / 6

Requirements

Coherent

- The big picture is clear and shared by all involved
- Important usage scenarios explained
- Priorities are clear
- Conflicts are addressed
- Impact is understood

3 / 6

Requirements

Acceptable

- Requirements describe a solution acceptable to the stakeholders
- The rate of change to agreed requirements is low
- Value is clear

4 / 6

Requirements

Addressed

- Enough requirements are implemented for the system to be acceptable
- Stakeholders agree the system is worth making operational

5 / 6

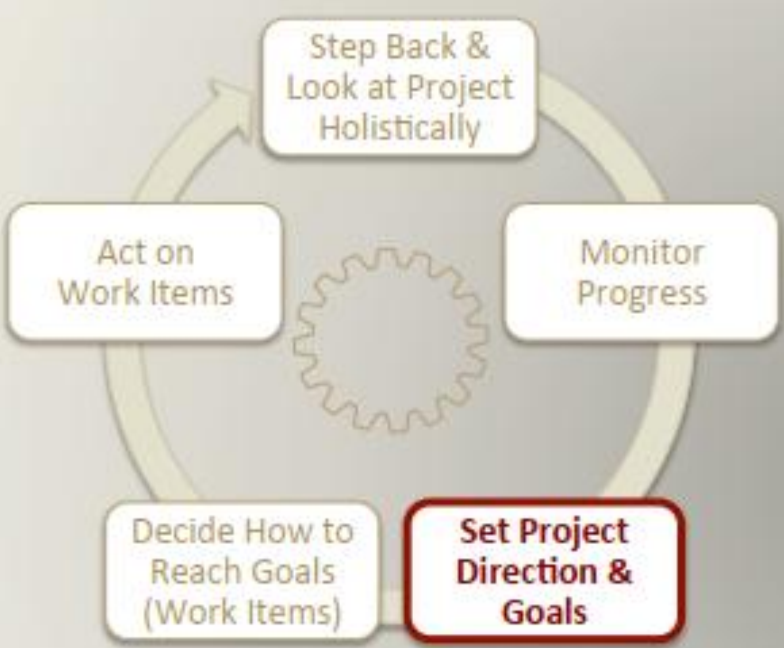
Requirements

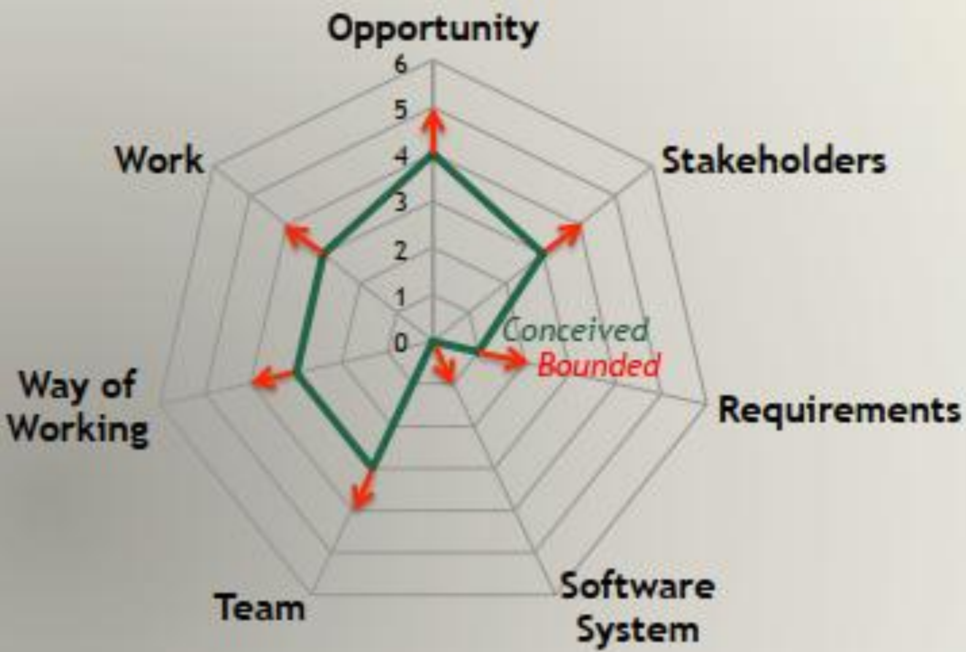
Fulfilled

- The system fully satisfies the requirements and the need
- There are no outstanding requirements items preventing completion

6 / 6

How does the Essence Kernel Work?


Step Back & Look at Project Holistically
Monitor Progress
Set Project Direction & Goals
Decide How to Reach Goals (Work Items)
Act on Work Items



Current State

Requirements

Conceived

- The need for a new system is clear
- Users are identified
- Initial sponsors are identified

1 / 6

Target State

Requirements

Bounded

- The purpose and extent of the system are agreed
- Success criteria are clear
- Mechanisms for handling requirements are agreed
- Constraints and assumptions identified

Goals

2 / 6

Requirements

Coherent

- The big picture is clear and shared by all involved
- Important usage scenarios explained
- Priorities are clear
- Conflicts are addressed
- Impact is understood

3 / 6

Requirements

Acceptable

- Requirements describe a solution acceptable to the stakeholders
- The rate of change to agreed requirements is low
- Value is clear

4 / 6

Requirements

Addressed

- Enough requirements are implemented for the system to be acceptable
- Stakeholders agree the system is worth making operational

5 / 6

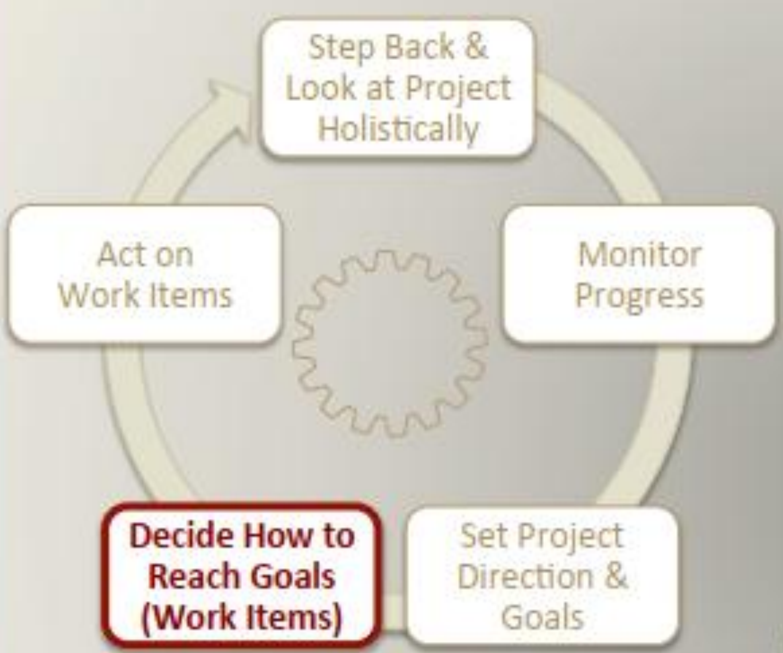
Requirements

Fulfilled

- The system fully satisfies the requirements and the need
- There are no outstanding requirements items preventing completion

6 / 6

How does the Essence Kernel Work?



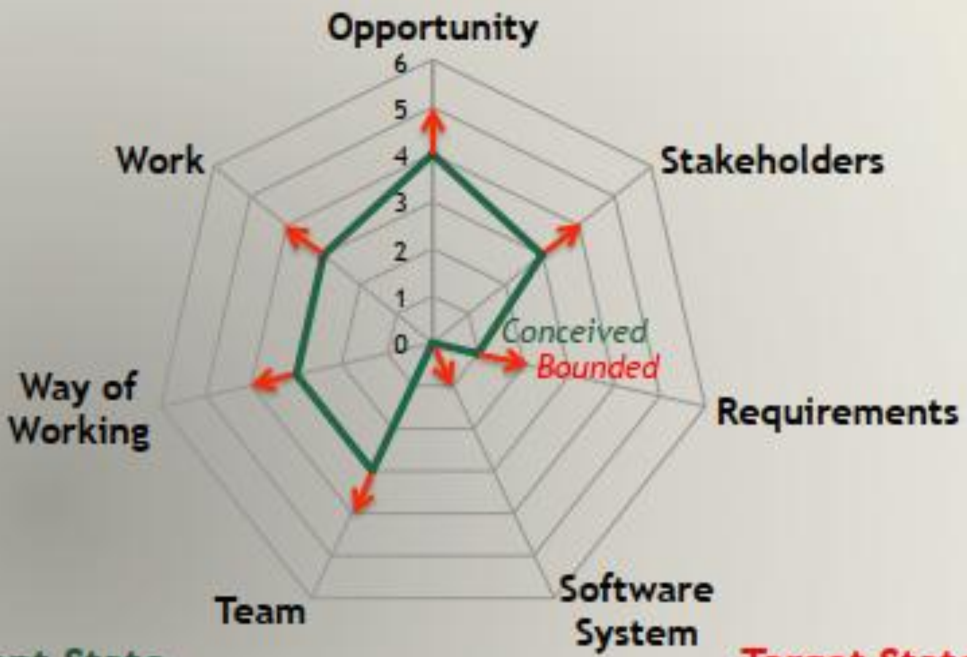
Step Back & Look at Project Holistically

Monitor Progress

Set Project Direction & Goals

Decide How to Reach Goals (Work Items)

Act on Work Items



Current State

Requirements

Conceived

- The need for a new system is clear
- Users are identified
- Initial sponsors are identified

1 / 6

Work Items:

❑ Define Project Scope

❑ Clarify Success Criteria

Target State

Requirements

Bounded

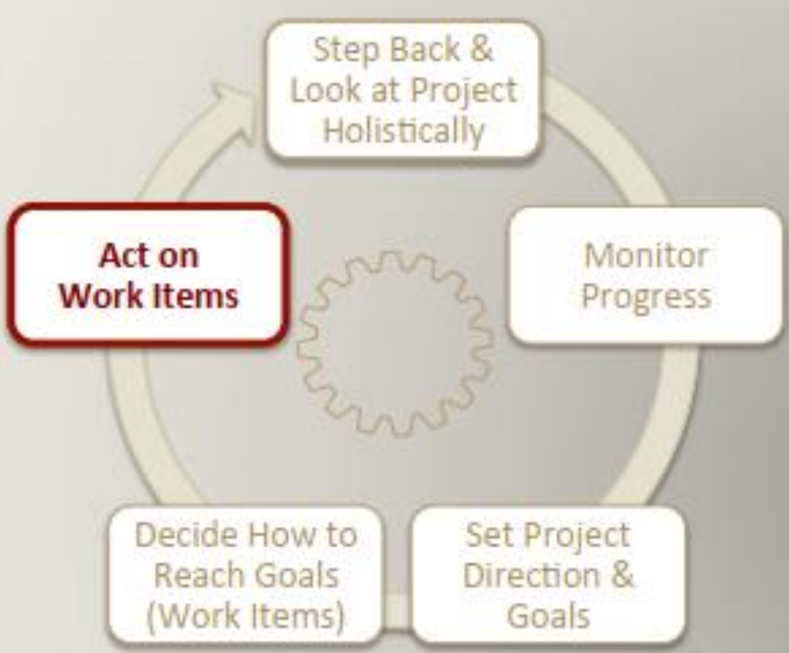
- The purpose and extent of the system are agreed
- Success criteria are clear

Goals

- Mechanisms for handling requirements are agreed
- Constraints and assumptions identified

2 / 6

How does the Essence Kernel Work?


Step Back & Look at Project Holistically
Monitor Progress
Set Project Direction & Goals
Decide How to Reach Goals (Work Items)
Act on Work Items

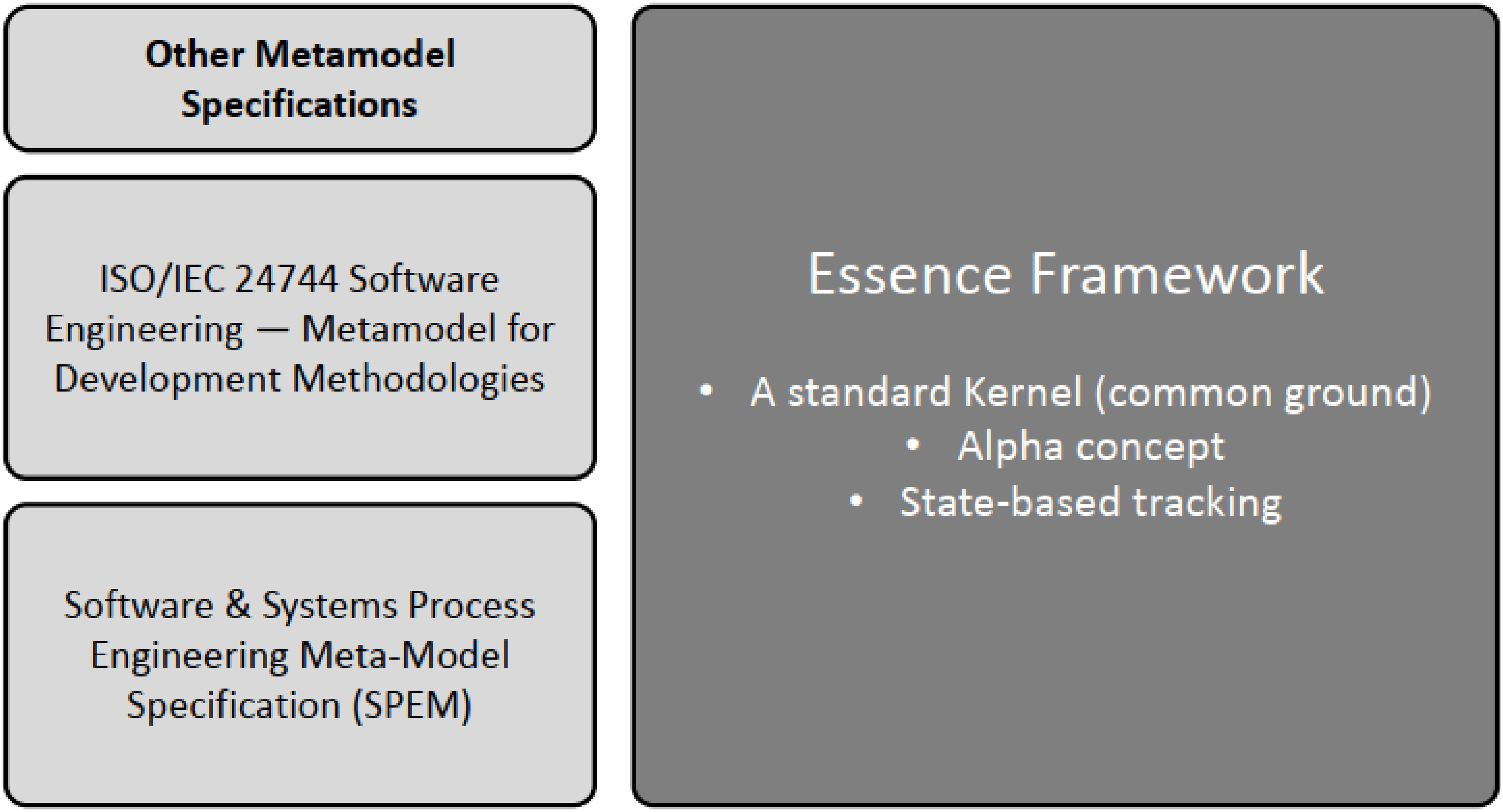
Work Items

- ❑ Define Project Scope
- ❑ Clarify Success Criteria
- ❑ …
- ❑ …
- ❑ …
- ❑ …

SEMAT

小结

What is different?



Other Metamodel Specifications

ISO/IEC 24744 Software Engineering — Metamodel for Development Methodologies

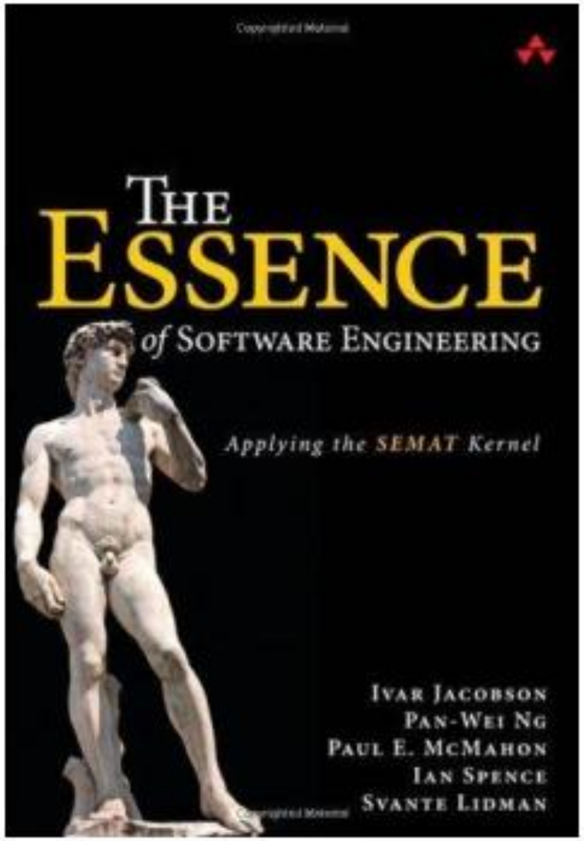
Software & Systems Process Engineering Meta-Model Specification (SPEM)

Essence Framework

- A standard Kernel (common ground)
- Alpha concept
- State-based tracking

Ref: Brian Elvesæter, Gorka Benguria, and Sylvia Ilieva. 2013. A comparison of the Essence 1.0 and SPEM 2.0 specifications for software engineering methods. In *Proceedings of the Third Workshop on Process-Based Approaches for Model-Driven Engineering* (PMDE '13). ACM, New York, NY, USA

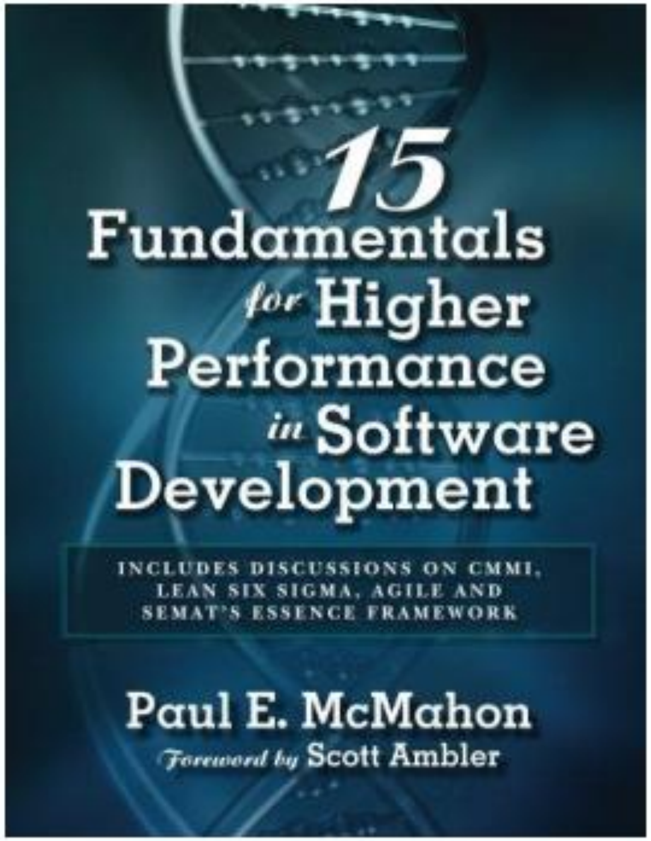
Sources of Information



THE ESSENCE of SOFTWARE ENGINEERING

Applying the SEMAT Kernel

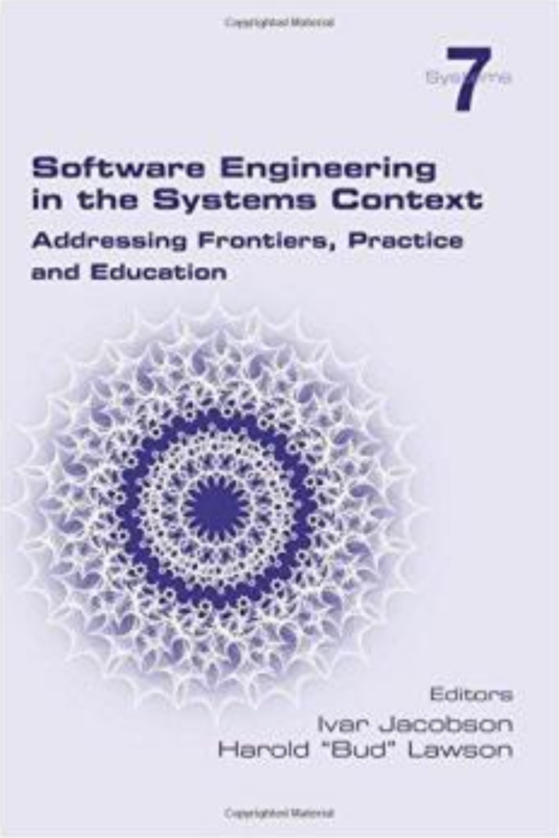
IVAR JACOBSON
PAN-WEI NG
PAUL E. MCMAHON
IAN SPENCE
SVANTE LIDMAN



15 Fundamentals *for* Higher Performance *in* Software Development

INCLUDES DISCUSSIONS ON CMMI, LEAN SIX SIGMA, AGILE AND SEMAT'S ESSENCE FRAMEWORK

Paul E. McMahon
Foreword by Scott Ambler



Systems 7

Software Engineering in the Systems Context

Addressing Frontiers, Practice and Education

Editors
Ivar Jacobson
Harold "Bud" Lawson

Sources of Information

www.semat.org

Software Engineering Method and Theory