

# SEMAT

## Software Engineering Method and Theory

[www.semat.org](http://www.semat.org)

<http://semat.org/essence-1.2>

Engenharia de Software II

3º Ano Engenharia Informática

Maria Clara Silveira

# SEMAT– Software Engineering Method and Theory



- **É uma iniciativa internacional dedicada a criar:**
  - Uma nova engenharia de software construída com base na experiência de especialistas em software, capturando o seu entendimento para educar e apoiar uma nova geração de profissionais
  - Concentra-se tanto no apoio à arte (métodos) quanto na construção de um entendimento fundamental (teoria)
  - O resultado desta iniciativa foi chamada **Essência**

# Documents Associated with Essence™ - Kernel and Language for Software Engineering Methods, Version 1.2

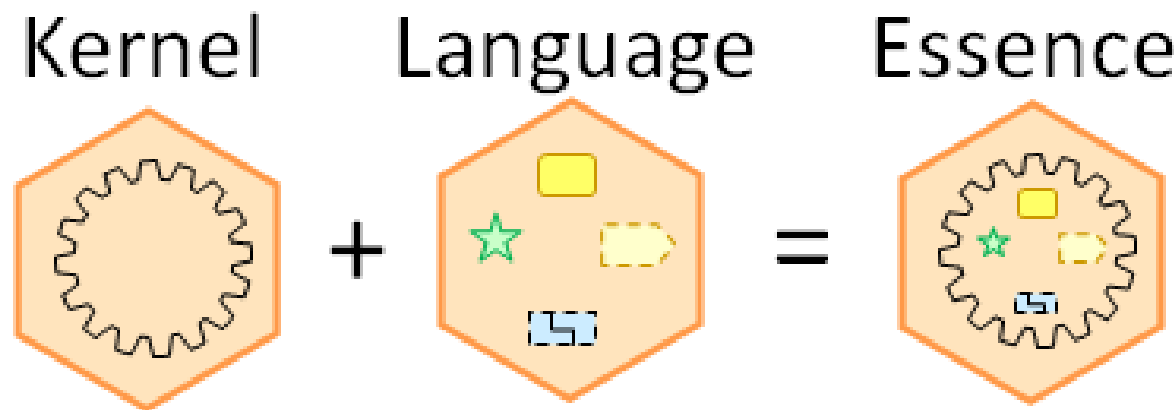
Release Date: July 2018

## Normative Documents

<b><i>OMG dokument number</i></b>	<b><i>Description</i></b>	<b><i>URL</i></b>
formal/18-10-02	Specification	<a href="http://semat.org/documents/20181/57862/formal-18-10-02.pdf/866c80c0-cdc8-488b-bcf8-0c67cb60b5d7">http://semat.org/documents/20181/57862/formal-18-10-02.pdf/866c80c0-cdc8-488b-bcf8-0c67cb60b5d7</a>

# Essence

- A essência é o **núcleo (kernel)** de uma teoria de engenharia de software, bem como
- a **linguagem** para descrever essa teoria e a abordagem para descrever métodos e práticas com base na teoria



# SEMAT - Kernel

- O primeiro resultado tangível da iniciativa SEMAT é o reconhecimento do *kernel* (**núcleo**) para engenharia de software
- Esse *kernel* pode ser considerado o conjunto mínimo de coisas que são universais para todos os projetos de desenvolvimento de software
- O *kernel* consiste em três partes:
  - Um meio para **medir** o progresso e a *saúde do projeto*
  - Uma categorização das **atividades** necessárias para avançar no progresso de um projeto
  - Um conjunto de **competências** necessárias para realizar essas atividades.

# SEMAT - linguagem

- O próprio kernel é formalmente definido como parte da especificação **Essence** que foi considerado um standard pelo Object Management Group.
- **Essence** também define a **linguagem** que pode ser usada para representar o núcleo e descrever práticas e métodos. É importante ressaltar que a linguagem é para ser usada por profissionais e *metodologistas*

## SEMAT– Método

- Um método (metodologia ou processo) é **uma descrição de uma maneira de trabalhar para realizar um esforço, como o desenvolvimento de software**
- os métodos derivam da **experiência** com o que funciona e o que não funciona
- essa **experiência** é *destilada*, primeiro em regras práticas e depois em diretrizes e, quando há consenso, eventualmente em padrões

## SEMAT– Práticas

Práticas desenvolvidas com base no Kernel permitem usar os métodos ágeis.

Uma prática pode ser expressa por:

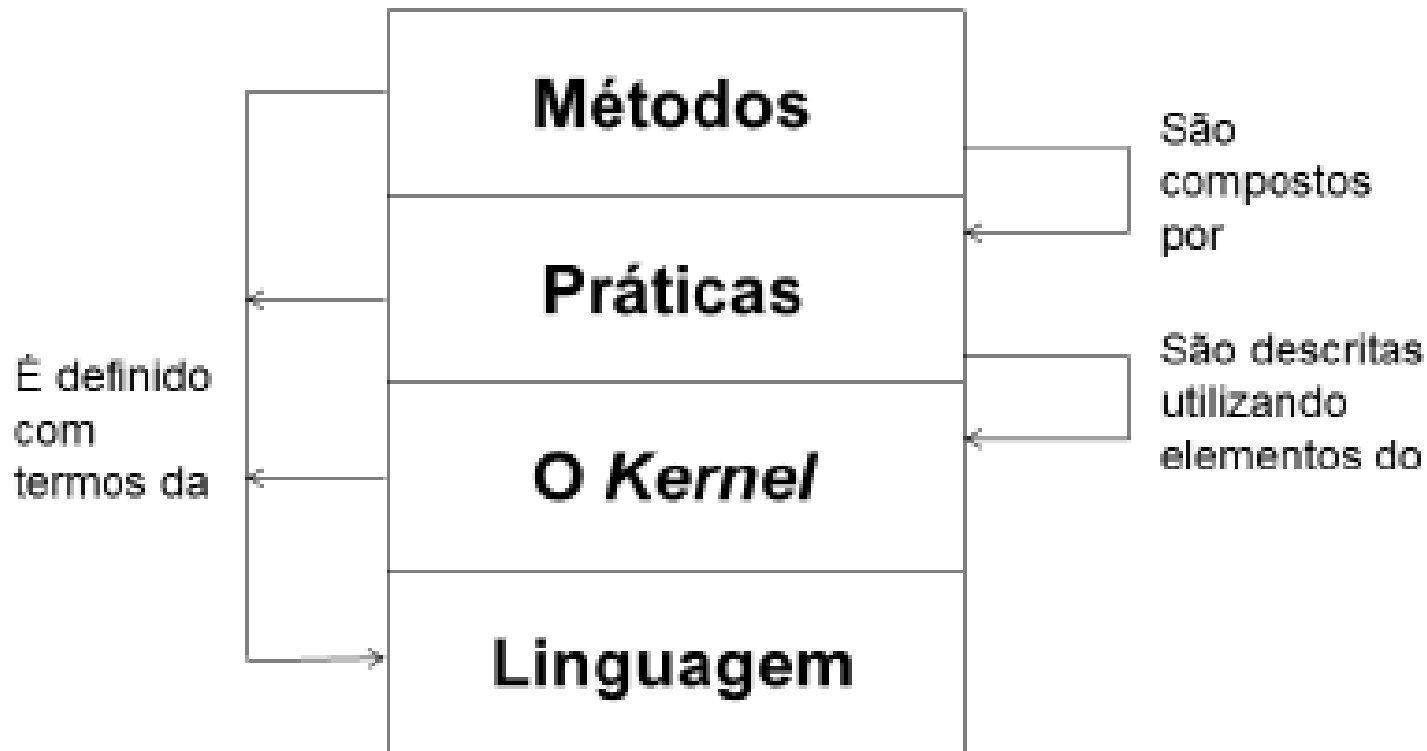
- Identificar as áreas em que avança o projeto;
- Descrever as atividades usadas para alcançar esse avanço e os produtos de trabalho produzidos;
- Descrever as competências específicas necessárias para realizar essas atividades.



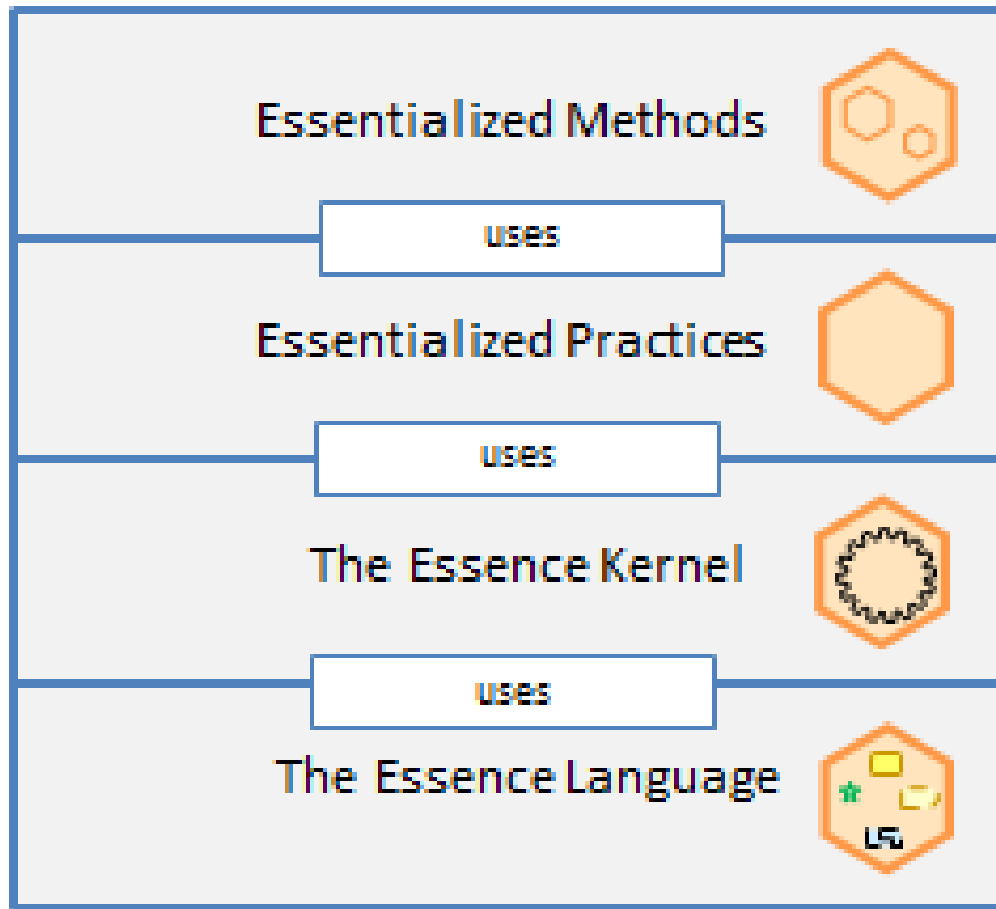
## SEMAT– Práticas

- O ponto crítico é que o kernel fornece uma estrutura comum para descrever todas as práticas e permitir que elas sejam combinadas em métodos
- A inserção de um conjunto de práticas nesse sistema permite identificar lacunas e sobreposições mais facilmente

# Arquitetura SEMAT

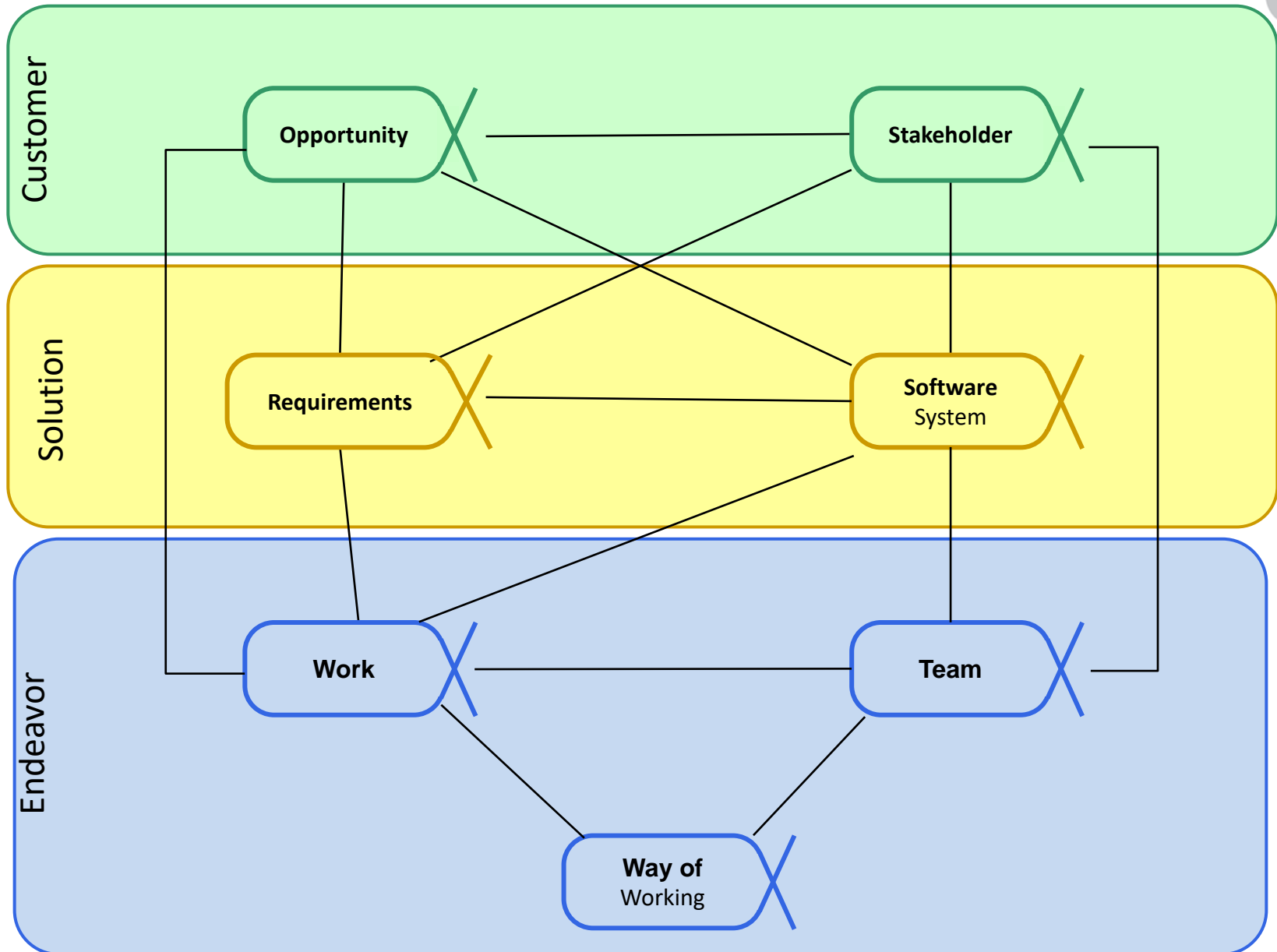


# Arquitetura SEMAT

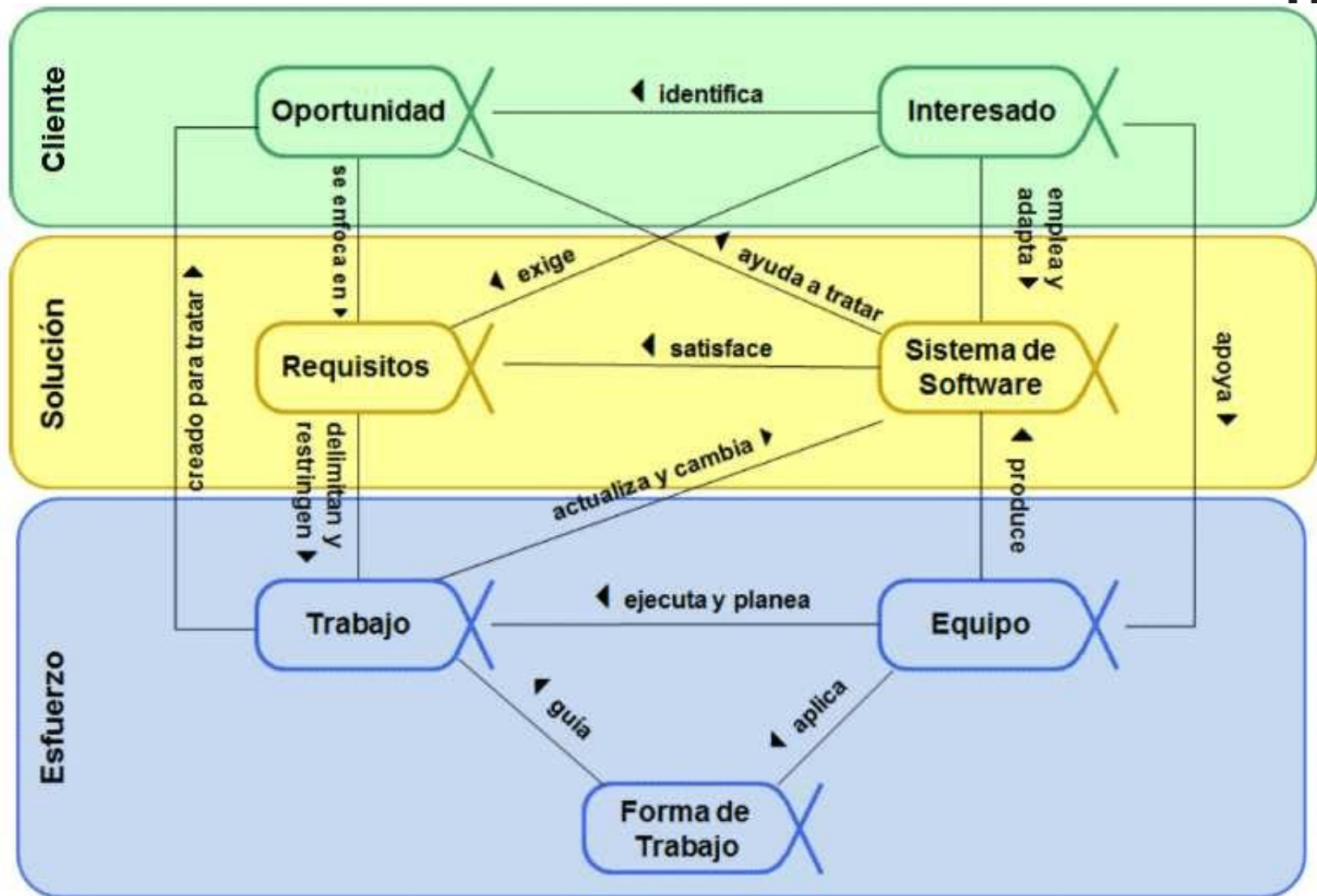


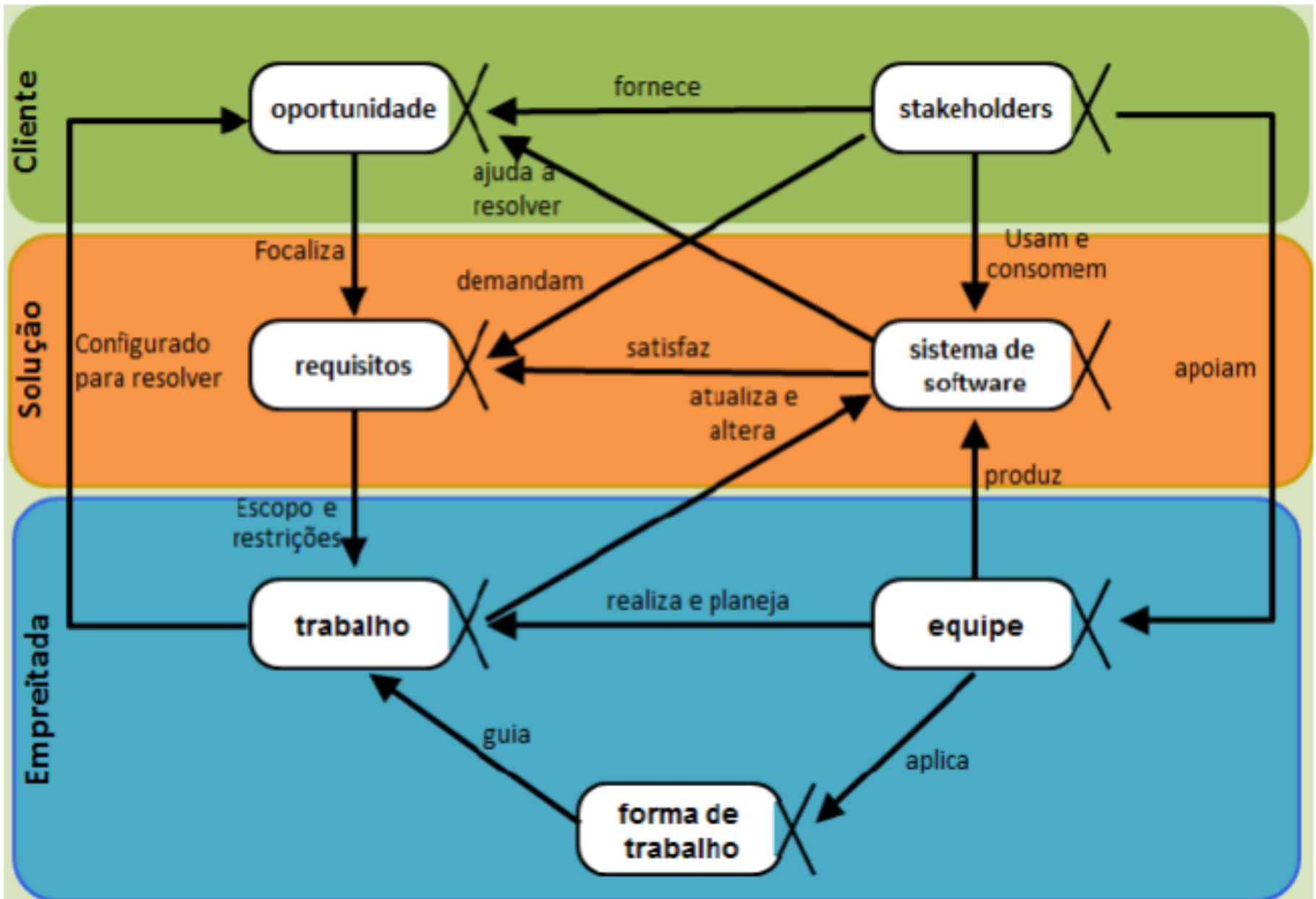
# SEMAT - Dimensões

- O núcleo SEMAT define sete **dimensões** para medir o progresso, conhecido como alfas. (o termo alfa era originalmente um acrônimo para o atributo de integridade do progresso a nível abstrato, mas agora é simplesmente usado como a palavra para uma dimensão de progresso e integridade)
- As sete dimensões são:
  - **oportunidade, stakeholders (partes interessadas), requisitos, sistema de software, trabalho, equipa e maneira de trabalhar**







# Domínios do Kernel (organização)











# Elementos chave

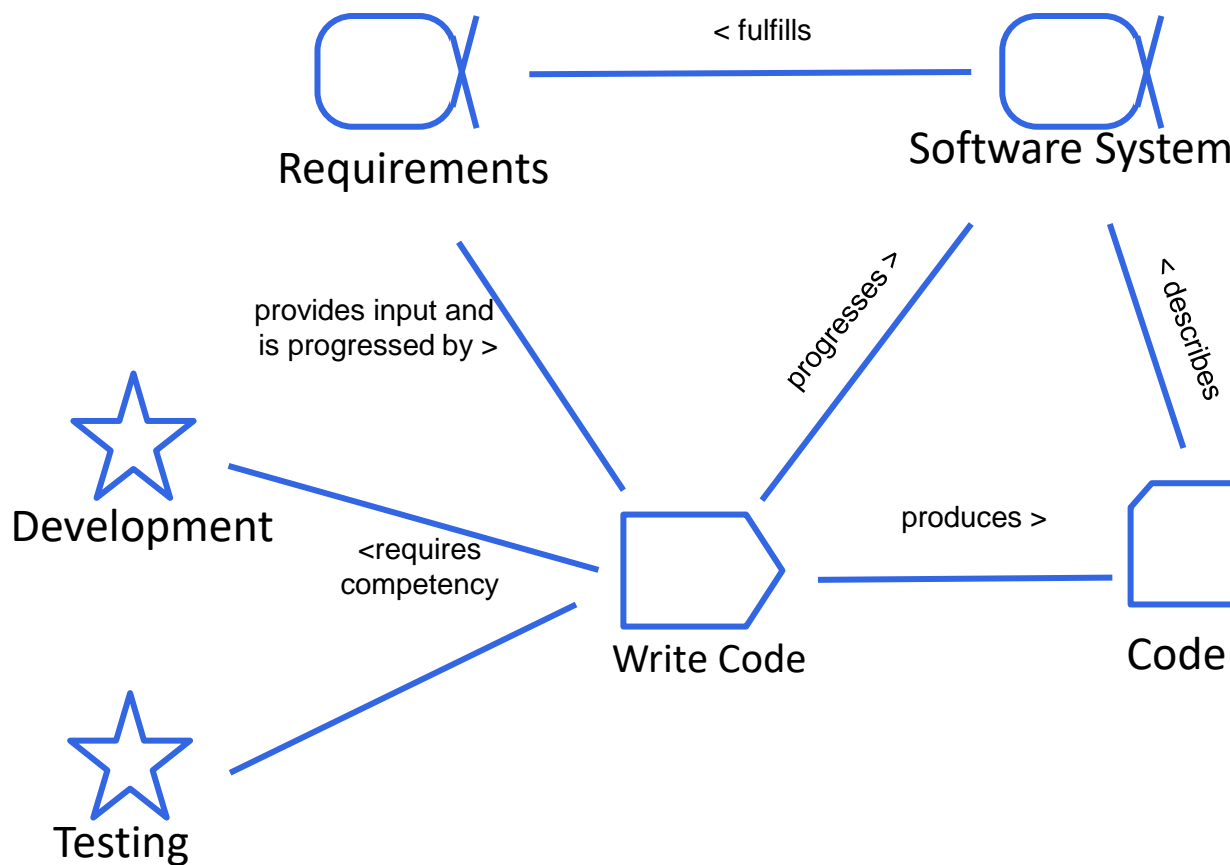
Alpha		An essential element of the software engineering endeavor that is relevant to an assessment of the progress and health of the endeavor
Work Product		The tangible things that practitioners produce when conducting software engineering activities
Activity		Things which practitioners do
Competency		Encompasses the abilities, capabilities, attainments, knowledge, and skills necessary to do a certain kind of work.





Element Type	Syntax	Meaning of element type
Activity Space		A placeholder for something to do in the software engineering endeavor. A placeholder may consist of zero to many activities.
Pattern		An arrangement of other elements represented in the language.
Alpha		An essential element of the software engineering endeavor that is relevant to an assessment of the progress and health of the endeavor
Work Product		The tangible things that practitioners produce when conducting software engineering activities
Activity		Things which practitioners do
Competency		Encompasses the abilities, capabilities, attainments, knowledge, and skills necessary to do a certain kind of work.

# Exemplo: prática da programação



# Competências



## Customer

- **Stakeholder Representation**
- Esta competência resume a capacidade de reunir, comunicar e equilibrar as necessidades dos stakeholders e representar com precisão as suas opiniões.

## Solution

- **Analysis**  
Resume a capacidade de entender as oportunidades e necessidades dos stakeholders e transformá-las num conjunto de requisitos negociado e consistente.
- **Development**  
engloba a capacidade de projetar, programar e codificar sistemas de software eficazes e eficientes, seguindo os padrões e normas acordados pela equipa.
- **Testing**  
Testar um sistema, verificar se é utilizável e se atende aos requisitos dos stakeholders

## Endeavour

- **Leadership**  
Permite que uma pessoa inspire e motive um grupo de pessoas a alcançar um resultado bem-sucedido do seu trabalho e atingir seus objetivos
- **Management**  
Esta competência resume a capacidade de coordenar, planear e acompanhar o trabalho realizado por uma equipa

# Cartão Alpha



## Requirements

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

Conceived

Bounded

Coherent

Acceptable

Addressed

Fulfilled



Generated by IJI Practice Workbench™

1.1.1

Alpha Name

Very brief  
alpha description

Alpha states  
Each alpha state has  
an alpha state card

## Alphas Made Tangible with Cards

### ✕ Opportunity

#### Identified

- opportunity identified that could be addressed by a software-based solution
- a stakeholder wishes to make an investment in better understanding potential value
- other stakeholders who share opportunity identified

1/6

### ✕ Stakeholders

#### Involved

- stakeholder representatives carry out responsibilities
- stakeholder representatives provide feedback & take part in decisions in timely way
- stakeholder representatives promptly communicate to stakeholder group

3/6

### ✕ Requirements

#### Addressed

- enough requirements are implemented for the system to be acceptable
- stakeholders agree the system is worth making operational

5/6

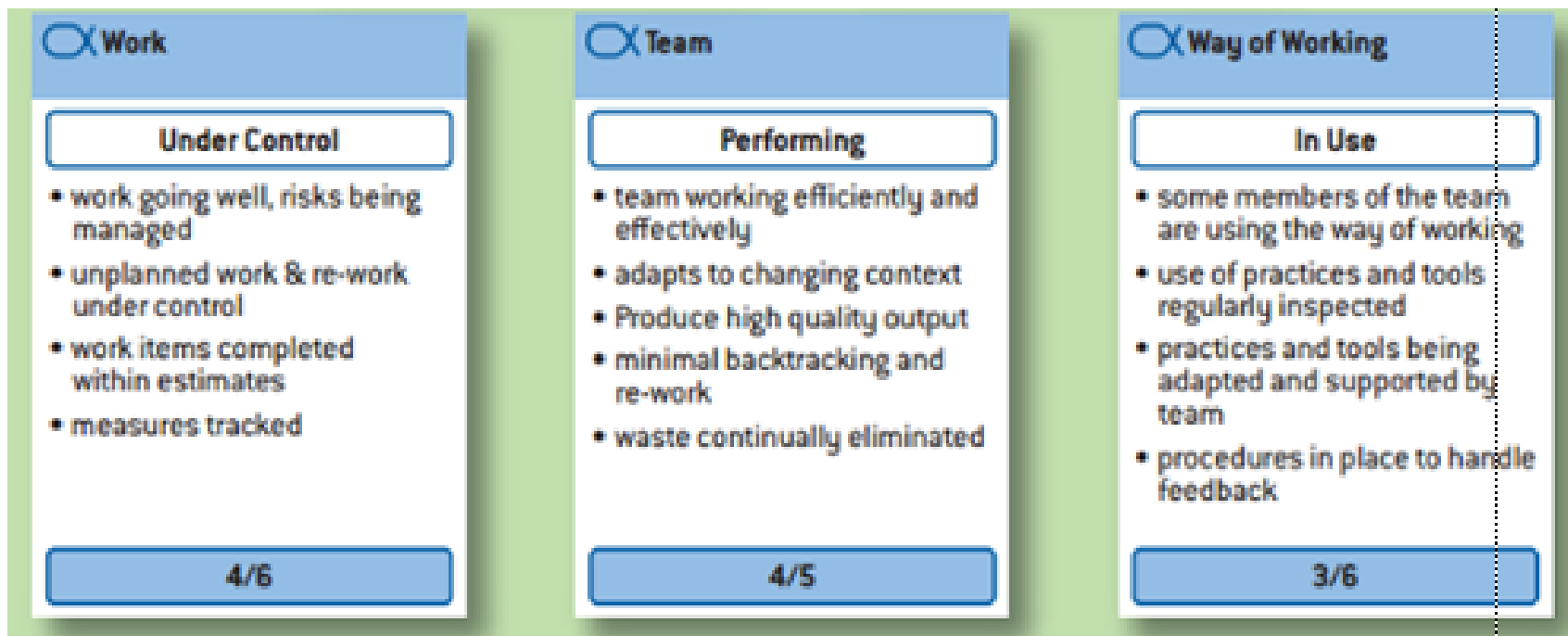
### ✕ Software System

#### Demonstrable

- key architecture characteristics demonstrated
- relevant stakeholders agree architecture is appropriate
- critical interface and system configurations exercised

2/6

# Alphas





# Cartoes essence kernel - detail cards checklist

## Stakeholders

### Recognized

- ☐ Stakeholder groups identified
- ☐ Key stakeholder groups represented
- ☐ Responsibilities defined

1 / 6



IVAR JACOBSON  
NOTES & SKETCHES  
Generated by ULP Practice Workbench™

1.1.2

## Stakeholders

### Represented

- ☐ Responsibilities agreed
- ☐ Representatives authorized
- ☐ Collaboration approach agreed
- ☐ Way of working supported & respected

2 / 6



IVAR JACOBSON  
NOTES & SKETCHES  
Generated by ULP Practice Workbench™

1.1.2

## Stakeholders

### Involved

- ☐ Representatives assist the team
- ☐ Timely feedback and decisions provided
- ☐ Changes promptly communicated

3 / 6



IVAR JACOBSON  
NOTES & SKETCHES  
Generated by ULP Practice Workbench™

1.1.2

## Stakeholders

### In Agreement

- ☐ Minimal expectations agreed
- ☐ Rep's happy with their involvement
- ☐ Rep's input valued
- ☐ Team's input valued
- ☐ Priorities clear & perspectives balanced

4 / 6



IVAR JACOBSON  
NOTES & SKETCHES  
Generated by ULP Practice Workbench™

1.1.2

## Stakeholders

### Satisfied for Deployment

- ☐ Stakeholder feedback provided
- ☐ System ready for deployment

## Stakeholders

### Satisfied in Use

- ☐ Feedback on system use available
- ☐ System meets expectations

## Opportunity

### Identified

- ☐ Idea behind opportunity identified
- ☐ At least one investing stakeholder interested
- ☐ Other stakeholders identified

## Opportunity

### Solution Needed

- ☐ Solution identified
- ☐ Stakeholders' needs established
- ☐ Problems and root causes identified
- ☐ Need for a solution confirmed
- ☐ At least one solution proposed





# Cartoes essence kernel - detail cards checklist



## Opportunity

### Value Established

- ☐ Opportunity value quantified
- ☐ Solution impact understood
- ☐ System value understood
- ☐ Success criteria clear
- ☐ Outcomes clear and quantified

3 / 6



IVAR JACOBSON  
INTERNATIONAL  
GENERATED BY LUPFABIO PONTES

1.1.2

## Opportunity

### Viable

- ☐ Solution outlined
- ☐ Solution possible within constraints
- ☐ Risks acceptable & manageable
- ☐ Solution profitable
- ☐ Reasons to develop solution understood
- ☐ Pursuit viable

4 / 6



IVAR JACOBSON  
INTERNATIONAL  
GENERATED BY LUPFABIO PONTES

1.1.2

## Opportunity

### Addressed

- ☐ Opportunity addressed
- ☐ Solution worth deploying
- ☐ Stakeholders satisfied

5 / 6



IVAR JACOBSON  
INTERNATIONAL  
GENERATED BY LUPFABIO PONTES

1.1.2

## Opportunity

### Benefit Accrued

- ☐ Solution accrues benefits
- ☐ ROI acceptable

6 / 6



IVAR JACOBSON  
INTERNATIONAL  
GENERATED BY LUPFABIO PONTES

1.1.2

## Software System

### Architecture Selected

- ☐ Architecture selection criteria agreed
- ☐ HW platforms identified
- ☐ Technologies selected
- ☐ System boundary known
- ☐ Decisions on system organization made
- ☐ Buy, build, reuse decisions made
- ☐ Key technical risks agreed to

1 / 6



IVAR JACOBSON  
INTERNATIONAL  
GENERATED BY LUPFABIO PONTES

## Software System

### Demonstrable

- ☐ Key architectural characteristics demonstrated
- ☐ System exercised & performance measured
- ☐ Critical HW configurations demonstrated
- ☐ Critical interfaces demonstrated
- ☐ Integration with environment demonstrated
- ☐ Architecture accepted as fit-for-purpose

2 / 6



IVAR JACOBSON  
INTERNATIONAL  
GENERATED BY LUPFABIO PONTES

## Software System

### Usable

- ☐ System can be operated
- ☐ System functionality tested
- ☐ System performance acceptable
- ☐ Defect levels acceptable
- ☐ System fully documented
- ☐ Release content known
- ☐ Added value clear

3 / 6



IVAR JACOBSON  
INTERNATIONAL  
GENERATED BY LUPFABIO PONTES

## Software System

### Ready

- ☐ User documentation available
- ☐ System accepted as fit-for-purpose
- ☐ Stakeholders want the system
- ☐ Operational support in place

4 / 6



IVAR JACOBSON  
INTERNATIONAL  
GENERATED BY LUPFABIO PONTES

# Alpha - requisitos

## OX Requisitos

### Compreendido

- A necessidade de um novo sistema está clara;
- Usuários estão identificados;
- Patrocinadores iniciais estão identificados.

1/6

## OX Requisitos

### Delimitado

- O propósito e os limites do sistema estão acordados;
- Os critérios de sucesso estão claros;
- Os mecanismos para lidar com os requisitos estão acordados;
- Restrições e premissas identificadas

2/6

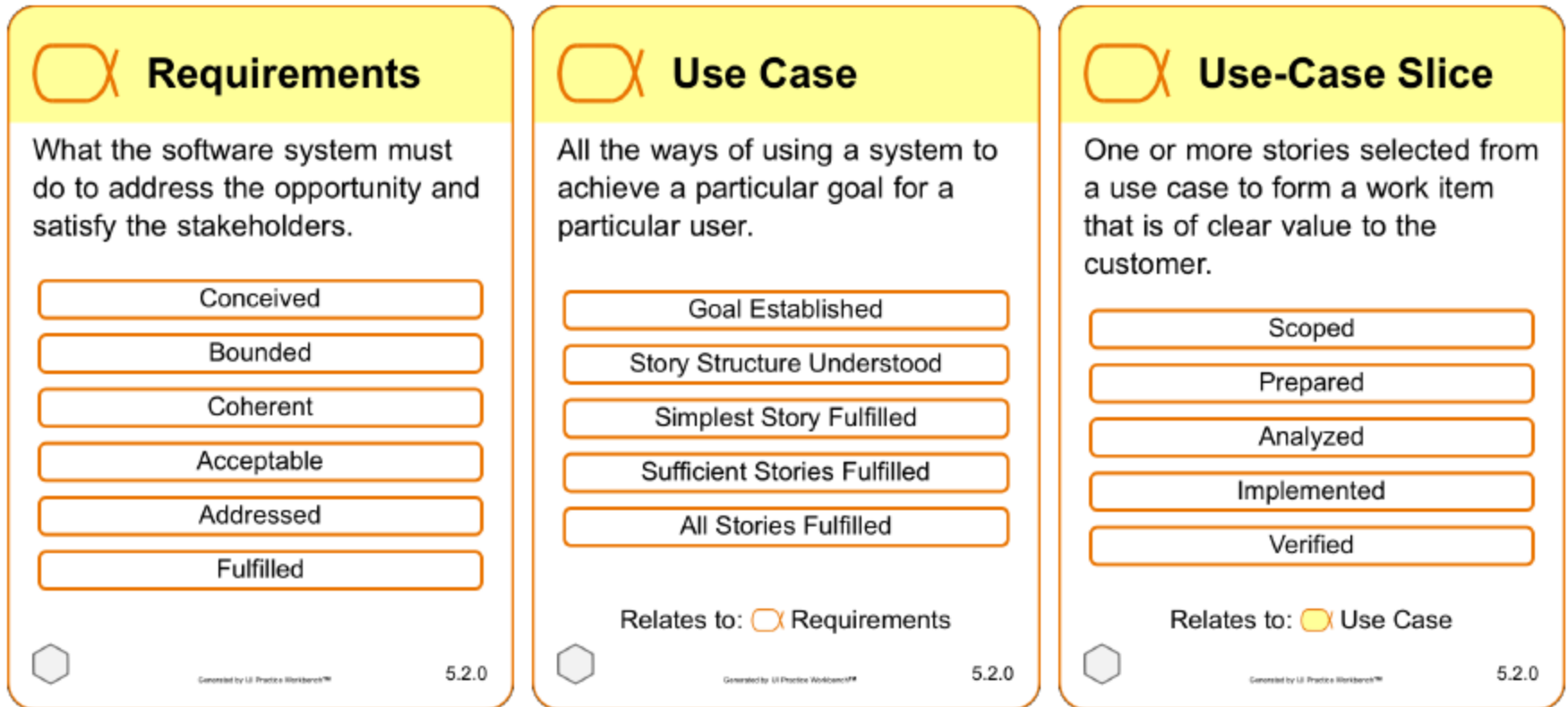
## OX Requisitos

### Coerente

- O panorama geral é claro e compartilhado por todos;
- Os cenários de uso mais importantes estão explicados;
- As prioridades estão claras;
- Os conflitos estão tratados;
- O impacto está compreendido..

3/6

# SEMAT - *Software Engineering Essentialized*



*Software Engineering Essentialized*; Jacobson et al, 2017

# User Story

As a traveller, I want to have destinations I like to be ranked higher than other destinations so that it is easier for me find these destinations

Acceptance criteria:

1. A visited destinations ranks higher than a non-visited one
2. A "liked" destination ranks higher than a "non-liked" destination



## Story Card

An index card, or equivalent, that captures the essential of a User Story. Normally expressed in "As a <role>, I want to <function>, so that <objective>" format.

Value Expressed

Acceptance Criteria Listed

Conversation Captured

Describes:  User Story



Generated by U1 Practice/Workbench/24

2016.01

Figure 103 User Story Examples

# User Story

- E.2.3.2 Work Products
  - E.2.3.2.1 User Story
- A User Story can be seen as a requirement item sub-alpha of Requirements that you want to monitor the state of. This requirement item is described by a User Story Card.

## Textual syntax

- alpha UserStory:
- "A User Story is an Independent, Negotiable, Valuable, Estimatable, Small, Testable requirement (INVEST)"

## User Story

Something that a software system could be extended to do, expressed in terms of the value that it will provide to a user of the system.

Identified

Ready for Development

In Progress

Verified

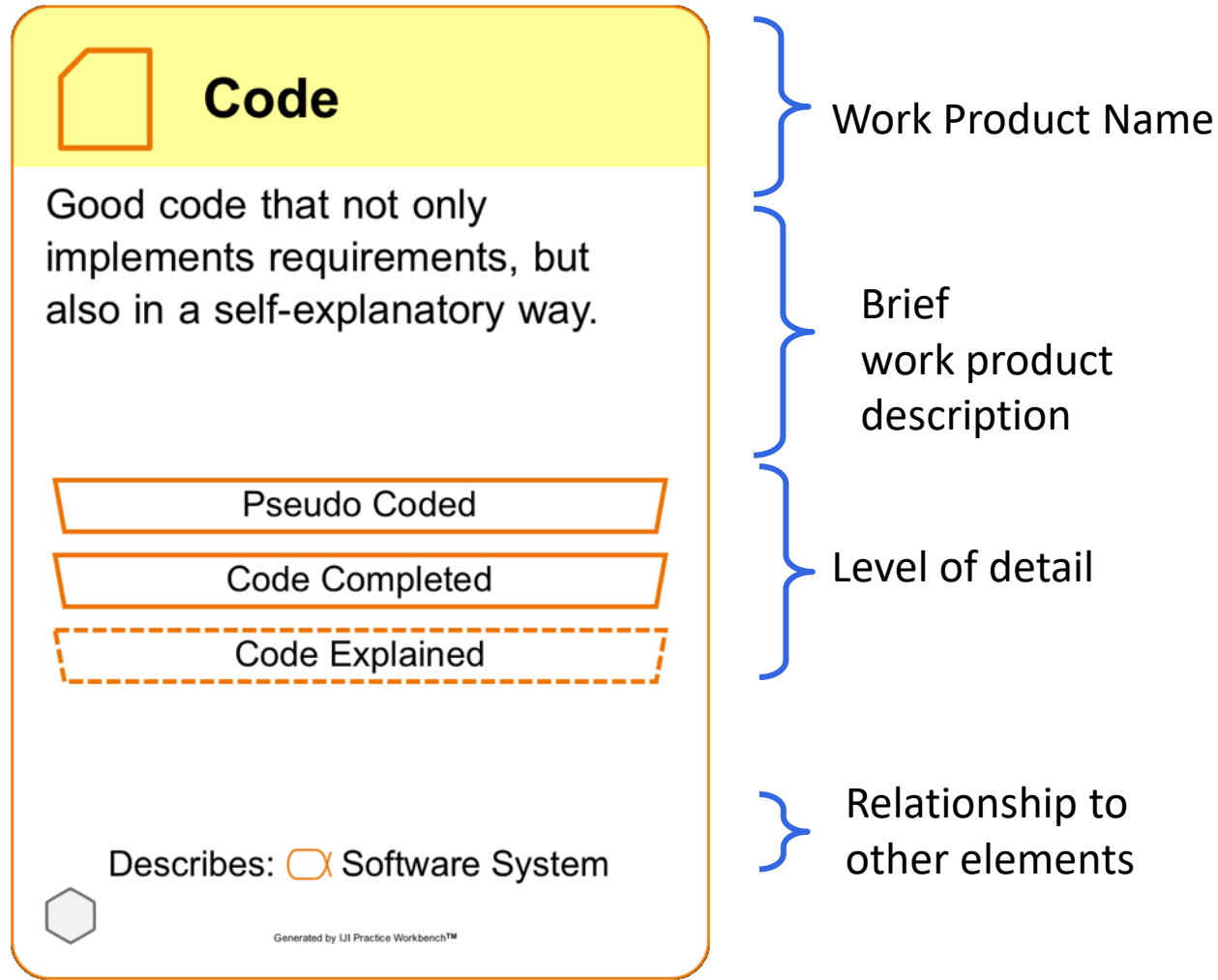
Relates to:  Requirements



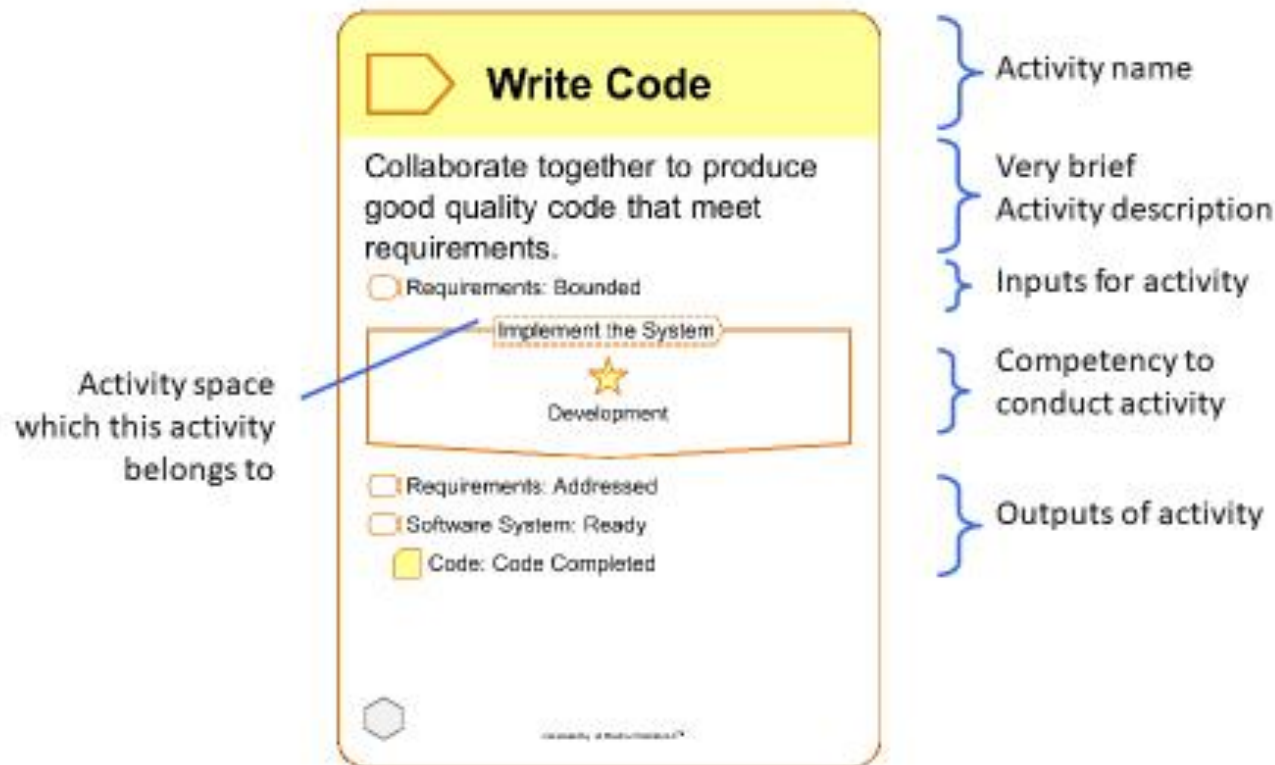
Generated by UIPractice Workbench™

2016.01

# Work Product

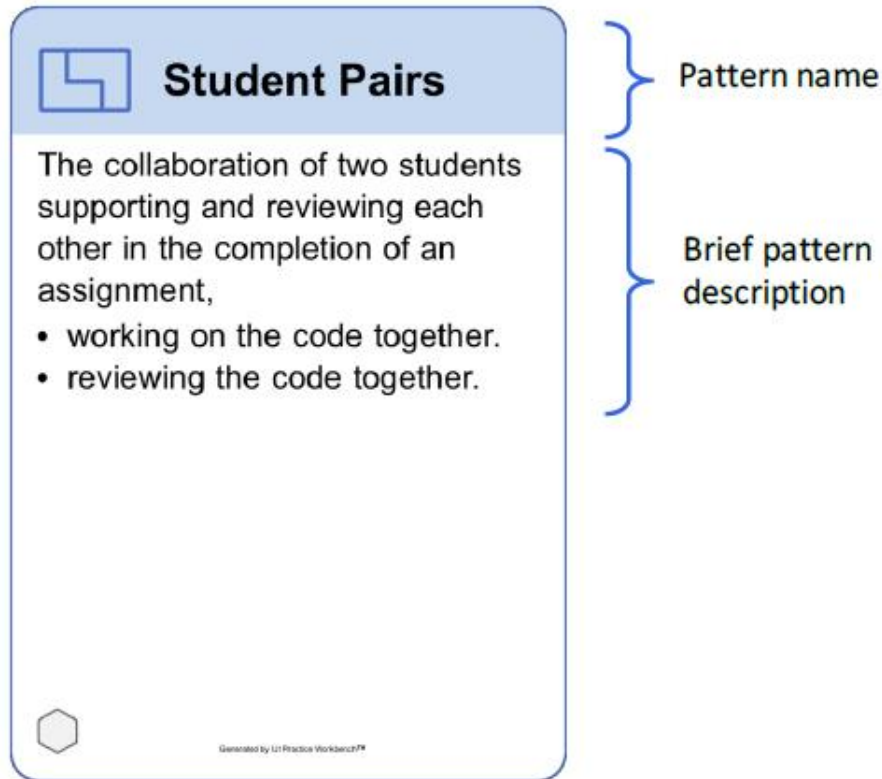


# Activity Card





# Pattern – elemento opcional



Student Pair pattern card

Jacobson et al, 2017

# Pattern – elemento opcional



## Scrum Master

The Scrum Master is responsible for ensuring that Scrum is understood and enacted. He/she is a servant leader for the Scrum Team.

Amongst other things he/she helps:

- Facilitate Scrum activities
- Remove impediments
- Team members understands Scrum
- Promote agility



Generated by LIT Politecnico da Guarda

03.2015

Practice	Description	Things to Watch (alphas)
Scrum	A practice for the iterative development of software systems working off a backlog	Sprint Product Backlog-Item
User Stories	A way to capture functionality that will be of value to a user of a software system.	User Story
Use Cases	All of the ways of using a system to achieve a particular goal for a particular user.	Use Case Use Case Slice
Microservices	A software architecture style that uses small independent processes to communicate.	Microservice

Depois de uma série de projetos piloto bem-sucedidos, o Gestor determinou que Scrum e User Stories ou Use Cases fossem empregues por todas as equipes de desenvolvimento

Após algumas discussões, a equipa também decidiu usar microsserviços para os ajudar a desenvolver o sistema de software

# Conclusão

- O termo **mudança de paradigma** pode ser um pouco exagerado atualmente; no entanto, a abordagem ***Essence*** baseada em kernel para engenharia de software pode razoavelmente ser considerada uma mudança
- Representa uma profunda mudança do ponto de vista da comunidade de engenharia de software