

Essence Kernel

Kristian Sandahl

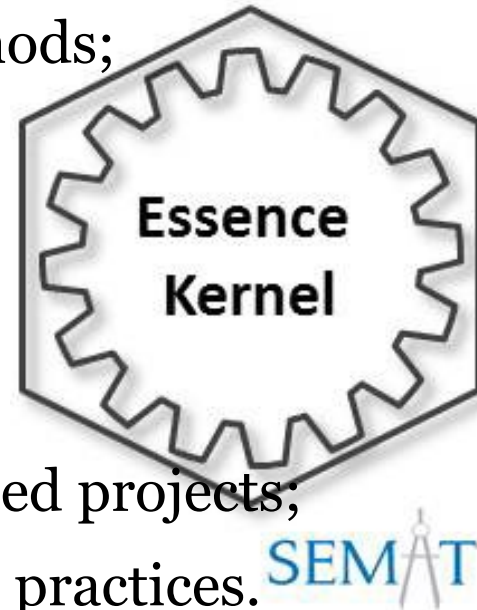
Software Engineering Method And Theory

- A common ground for software engineering
- Moving away from SE methods “fashion” industry.
- Founded in 2009 by:
 - Ivar Jacobson
 - Bertrand Meyer
 - Richard Soley
- OMG Standard under the name Essence
- The SEMAT Kernel – manifestation of the common ground



The Kernel

- comprises the central elements for all SE methods;
- provides a common language for comparing, applying, and improving methods;
- supports progress monitoring;
- works in small- and large-scale projects;
- works for well documented and less documented projects;
- comes with a language and tool for developing practices.
- Uptake in China, Russia, South Africa, Japan, Silicon Valley, Florida, Mexico, Germany



What's in it for us?

- It is highly probable that this will be used much more in the future.
- By focusing on the Essentials, the project groups have more freedom and responsibility.
- Our students will not become “methodists”.
- Taught in TDDE46 Software quality.



Areas of concern

Use and exploitation of the system

Customer

Specification and development

Solution

The team and approach of work

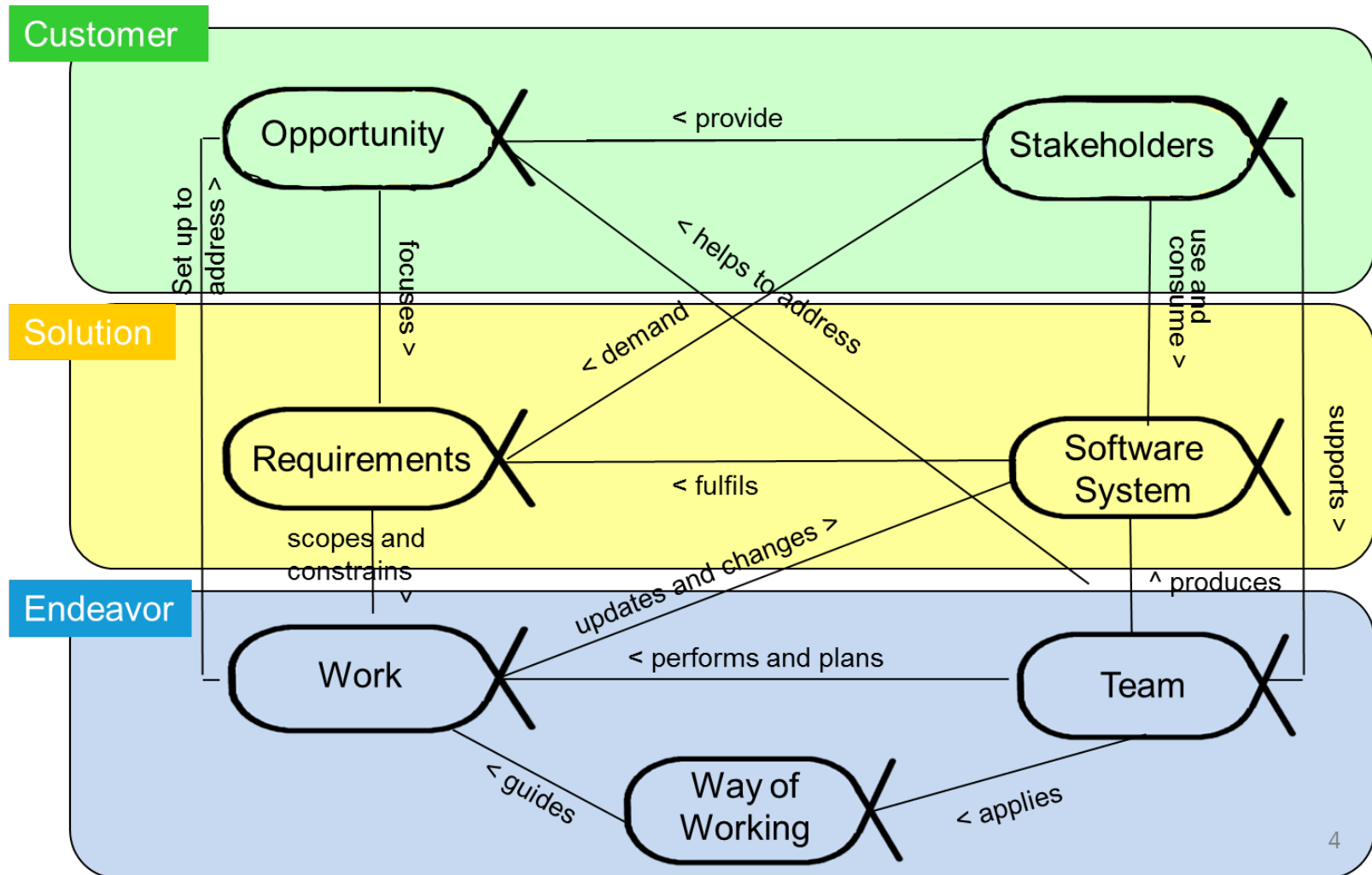
Endeavor

What is an ALPHA?

- Alpha is an acronym for an Abstract-Level Progress Health Atttribute.
- A critical indicator of things that are most important to monitor and progress.

A large, bold, black Greek letter alpha (α) is positioned on the right side of the slide. It is a cursive-style character with a thick stroke and a small loop at the bottom right.

The Kernel ALPHAs



Brief explanation

A reason for developing the system. Ex: user need

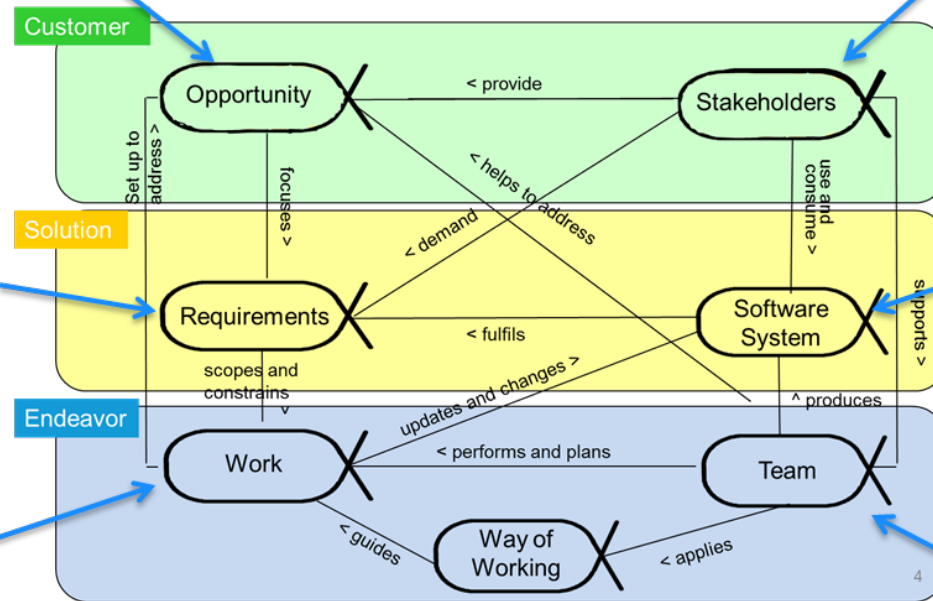
An agent affected by the system
Ex: Customer, project team

What the system must do.
Ex: Store data, be usable

The system of hardware, data, and software items. Ex: a TV

Activity performed.
Ex: Test a GUI.

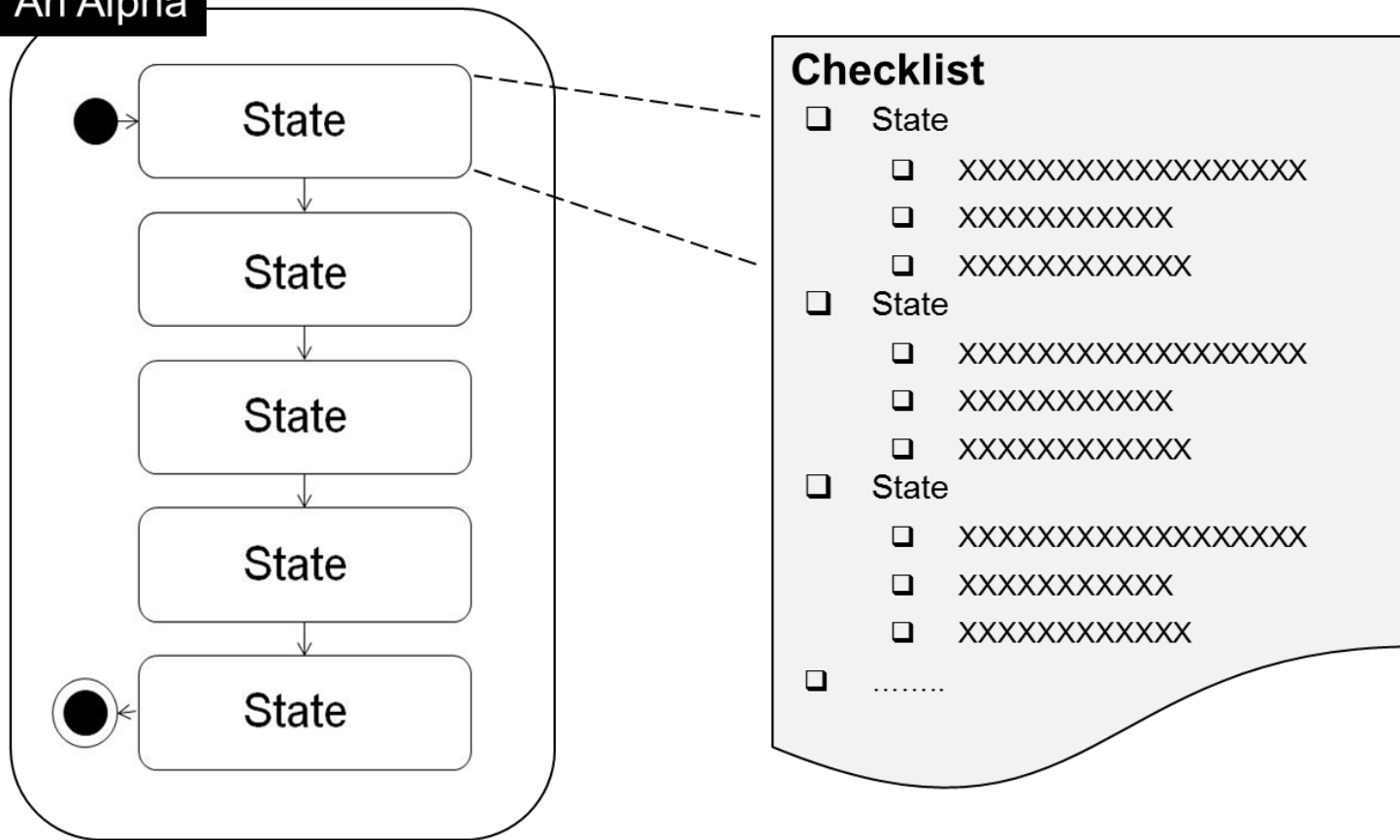
The people engaged in the project. Ex: group 4



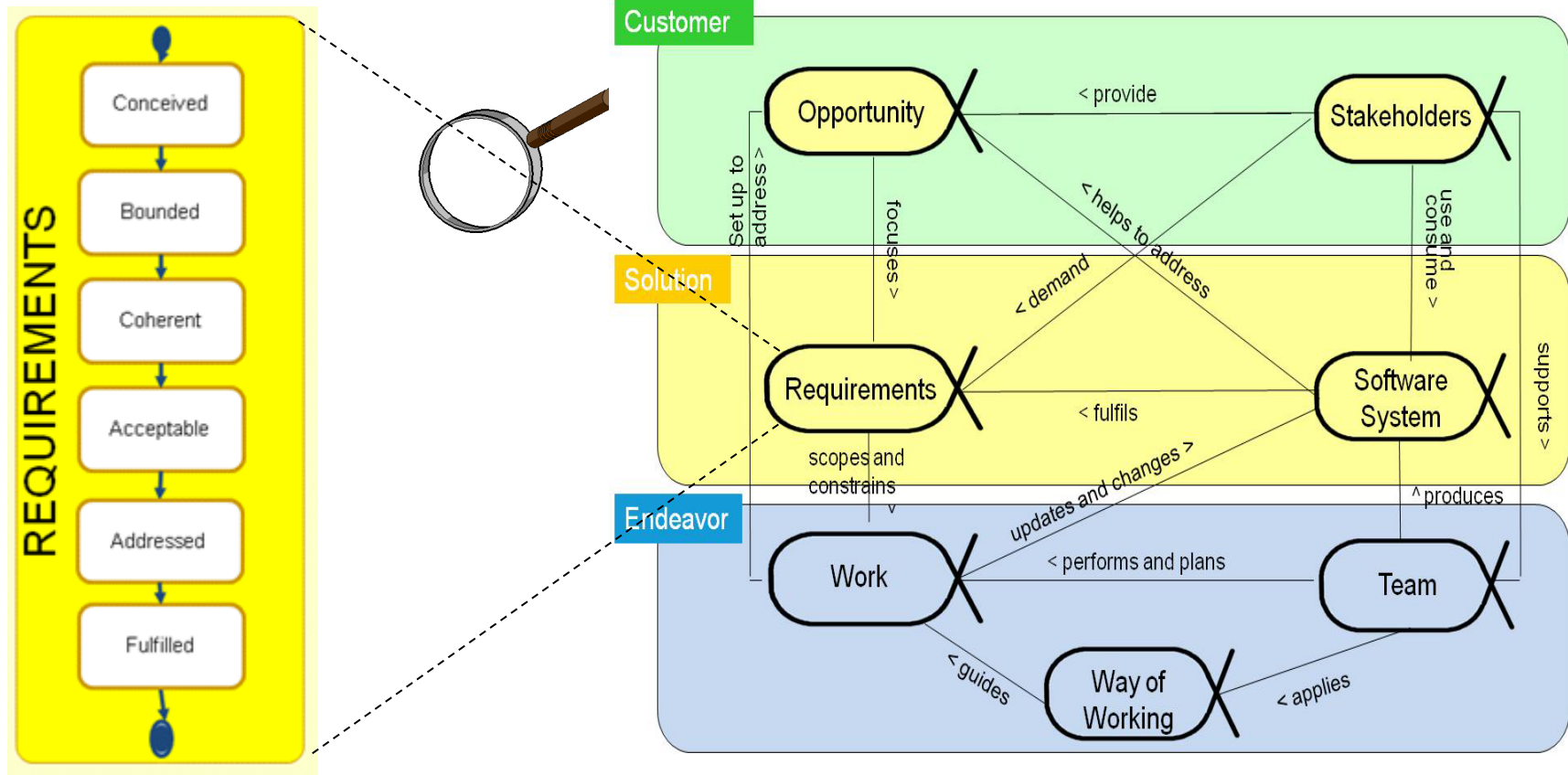
Tailored set of practices.
Ex: TDD, Kick-off meeting

The structure of an ALPHA

An Alpha

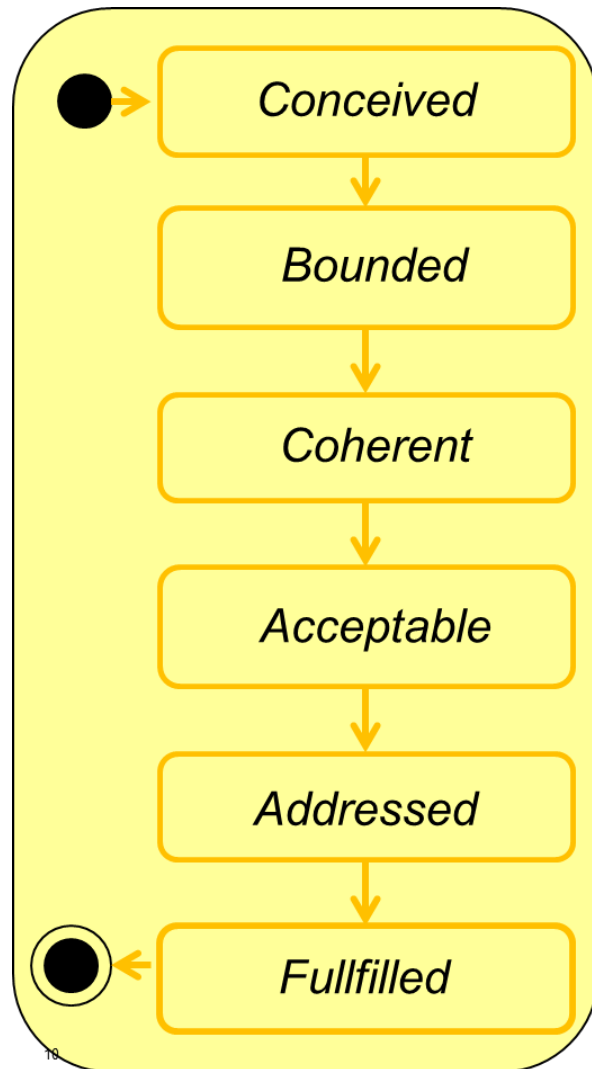


Requirements– one of the alphas



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.

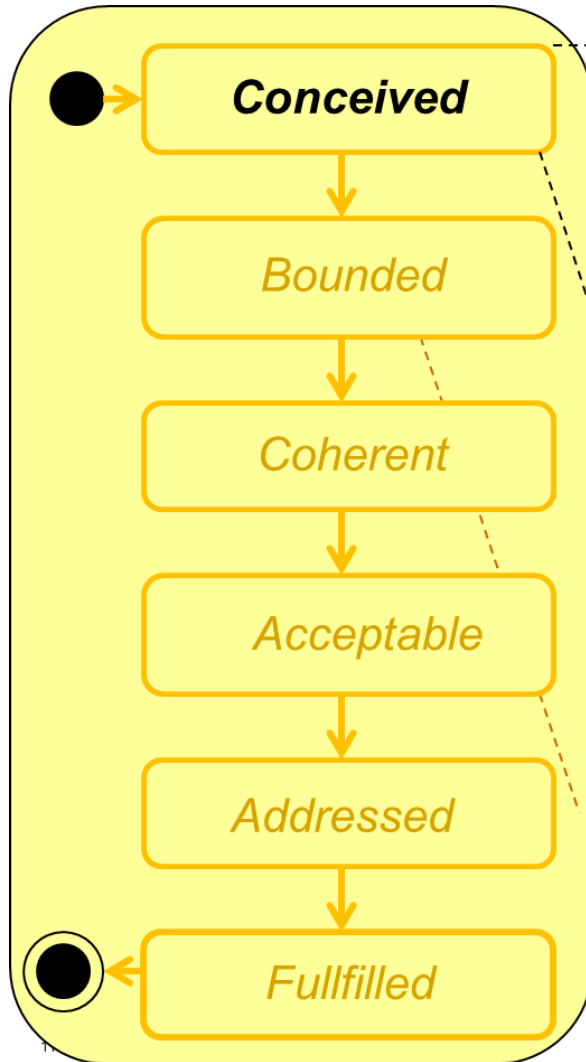
The requirements provide a coherent description of the essential characteristics of the new system.

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.

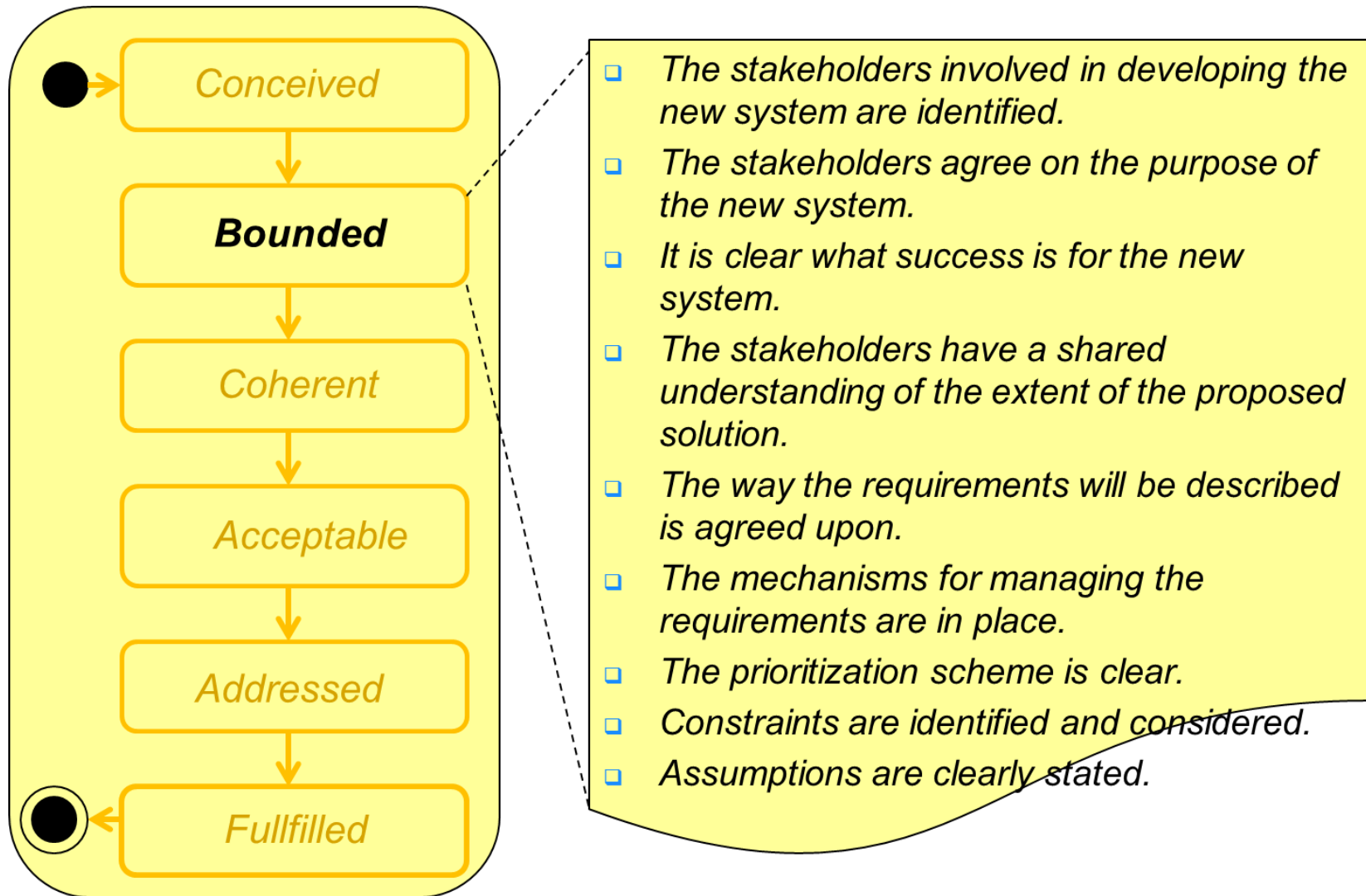
The requirements have been addressed to fully satisfy the need for a new system.

Checklist for requirements states

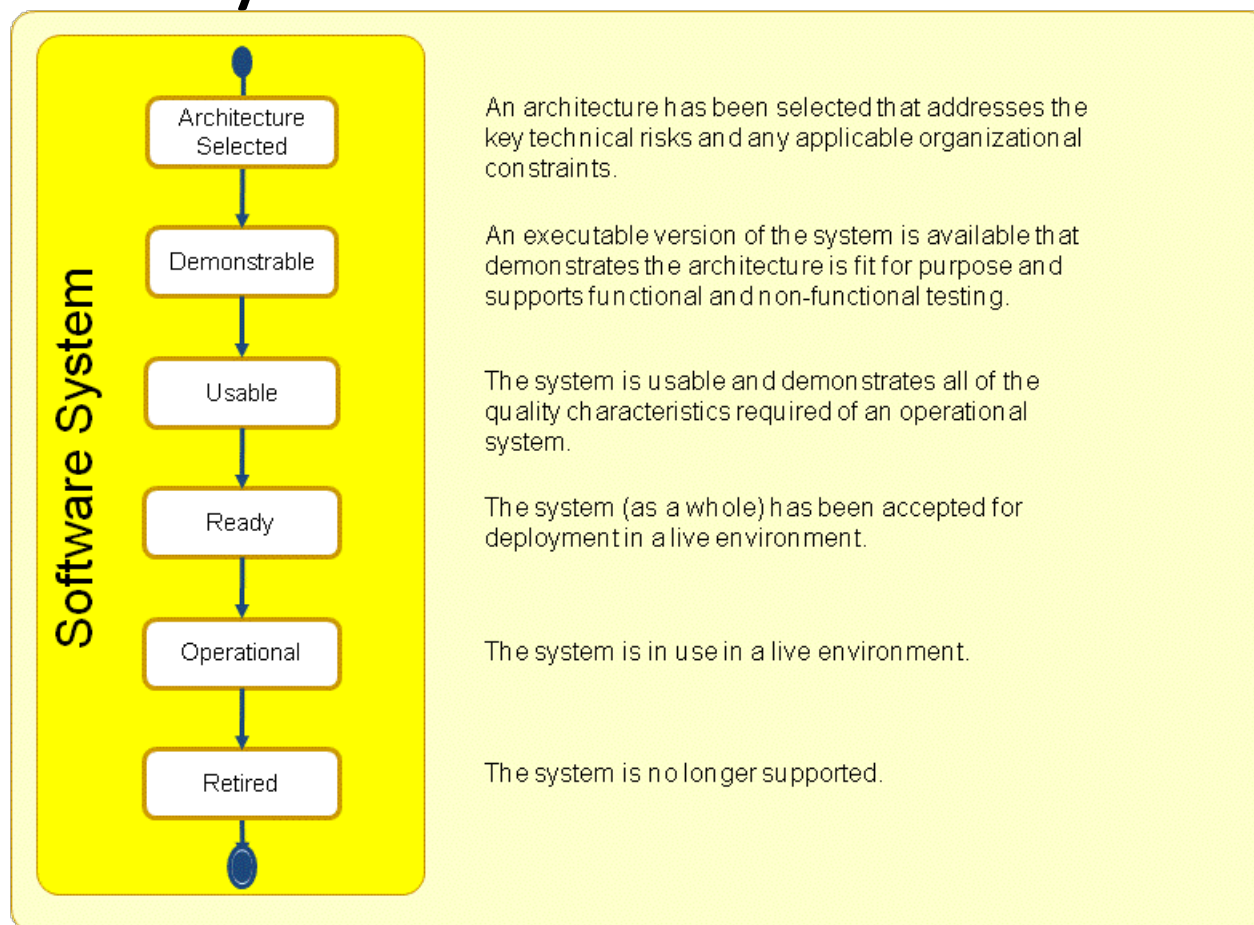


- ❑ *The initial set of stakeholders agrees that a system is to be produced.*
- ❑ *The stakeholders that will use the new system are identified.*
- ❑ *The stakeholders that will fund the initial work on the new system are identified.*
- ❑ *There is a clear opportunity for the new system to address.*

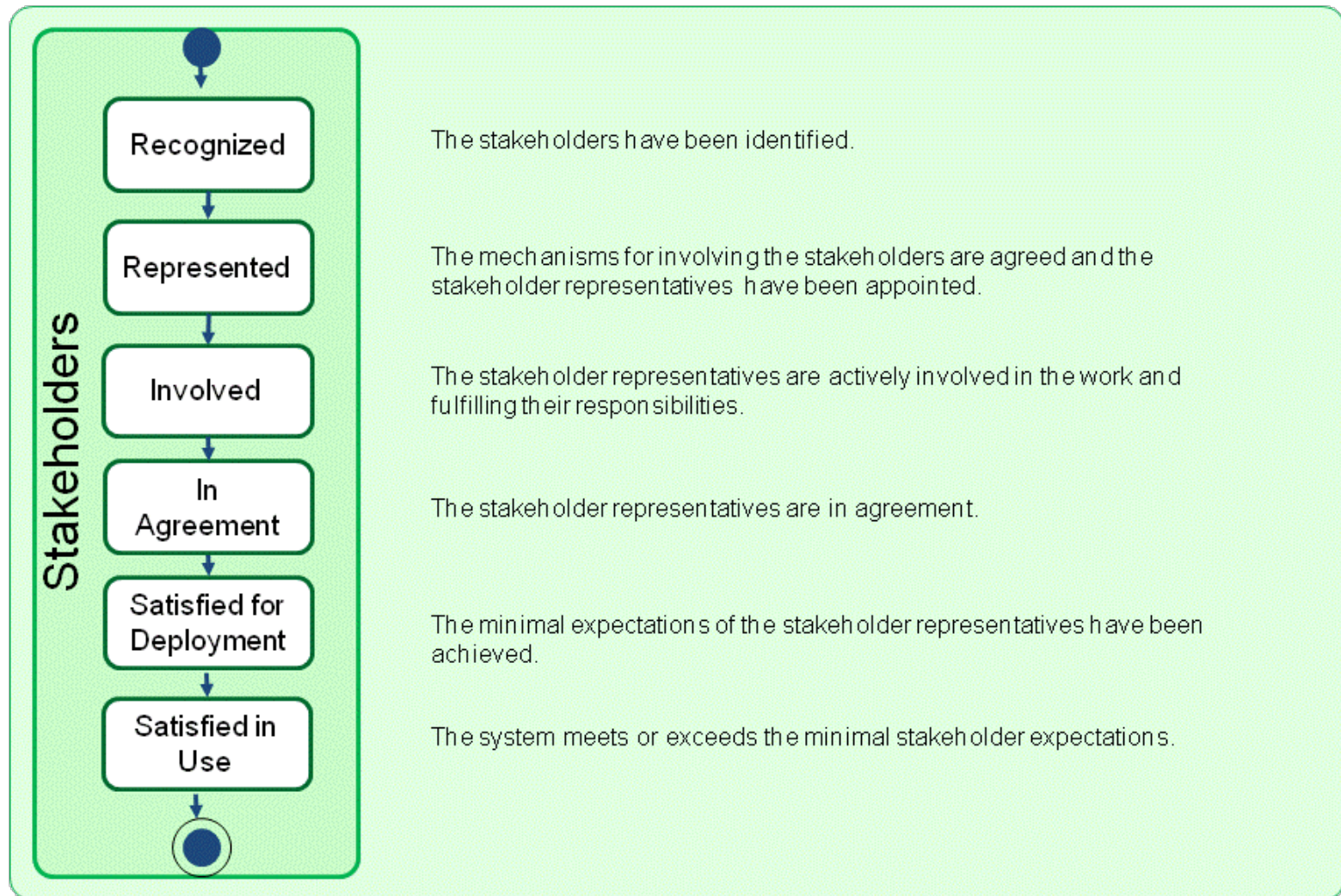
Checklist for requirements states



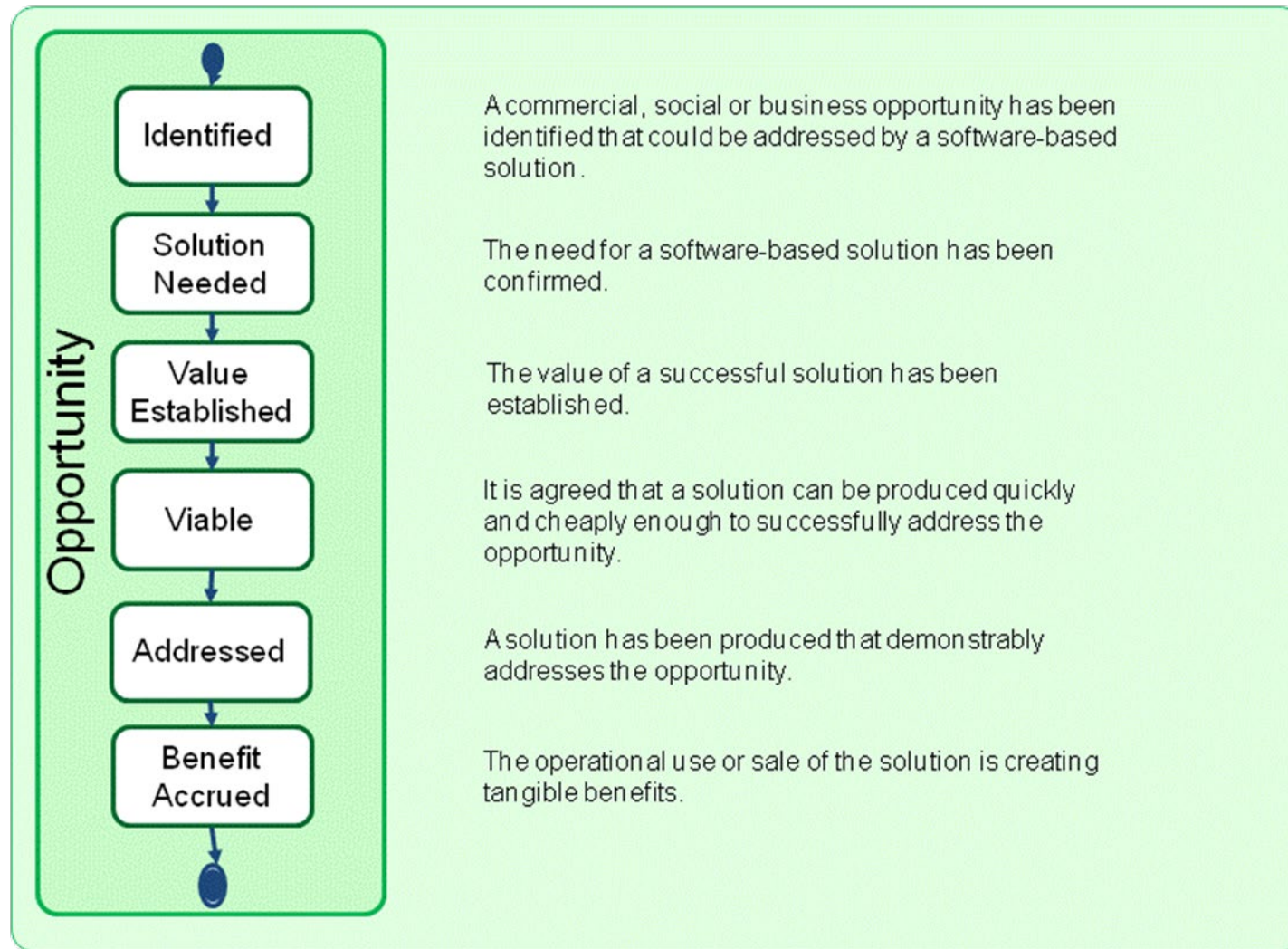
Software system



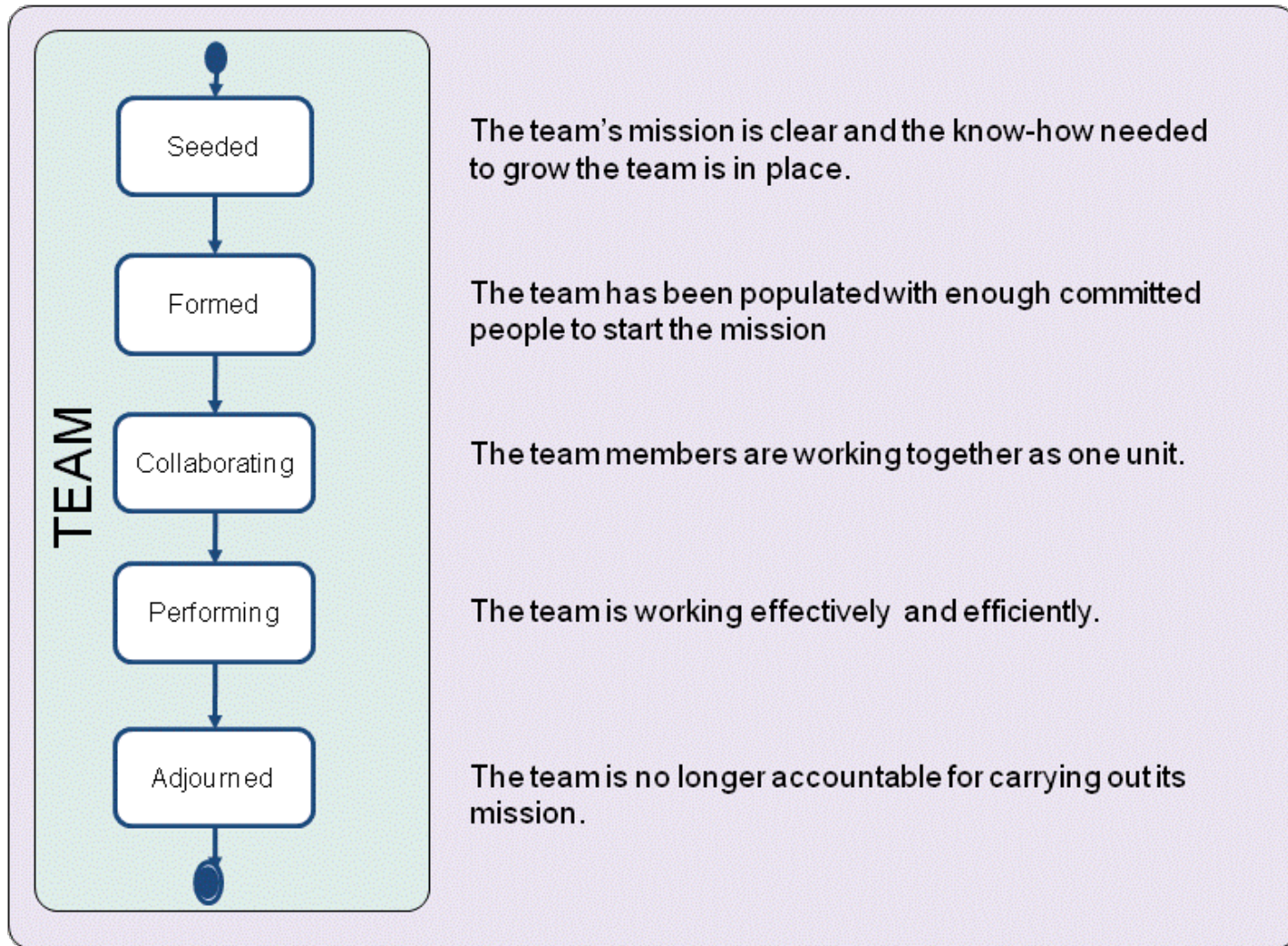
Stakeholders



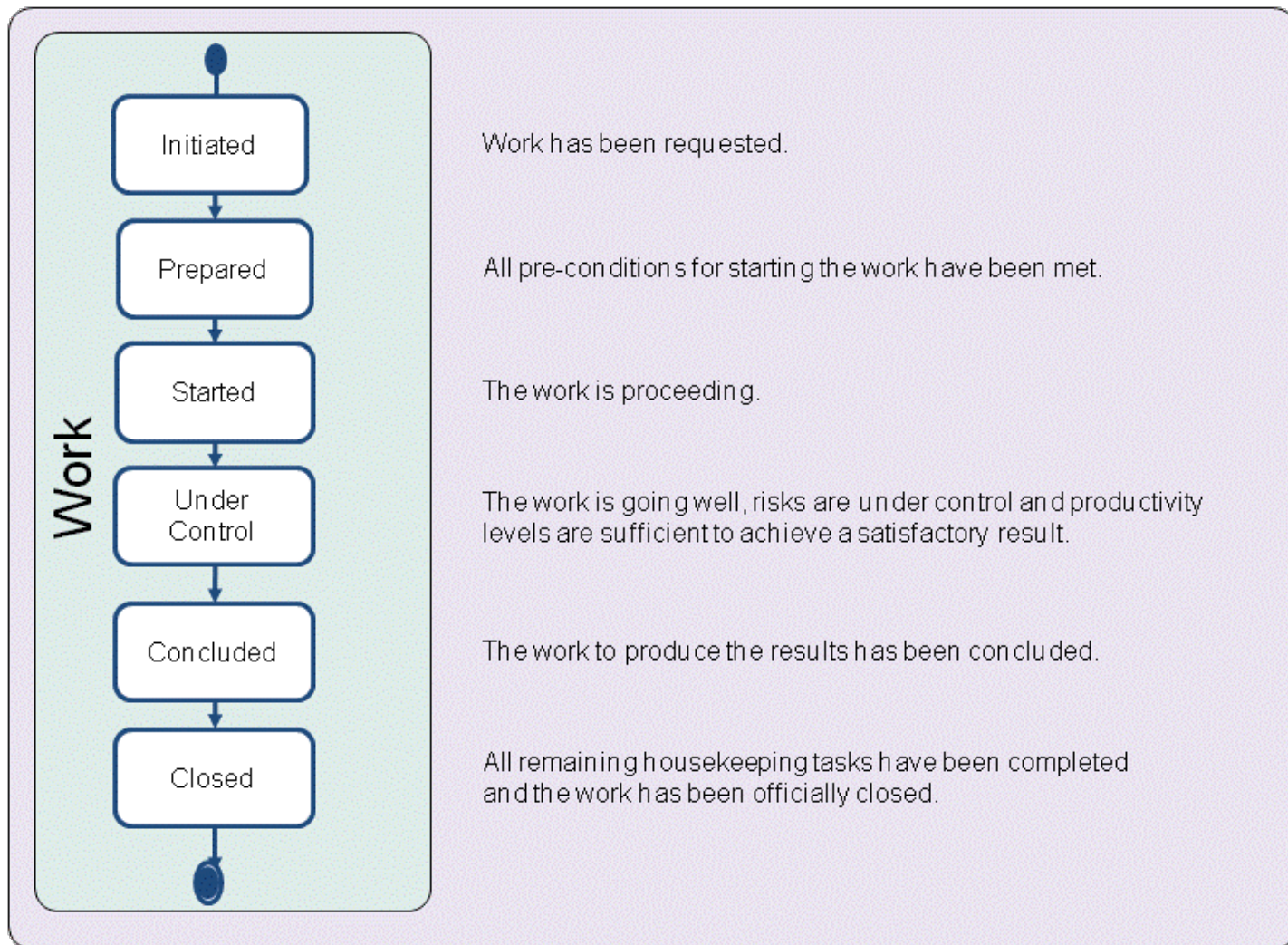
Opportunity



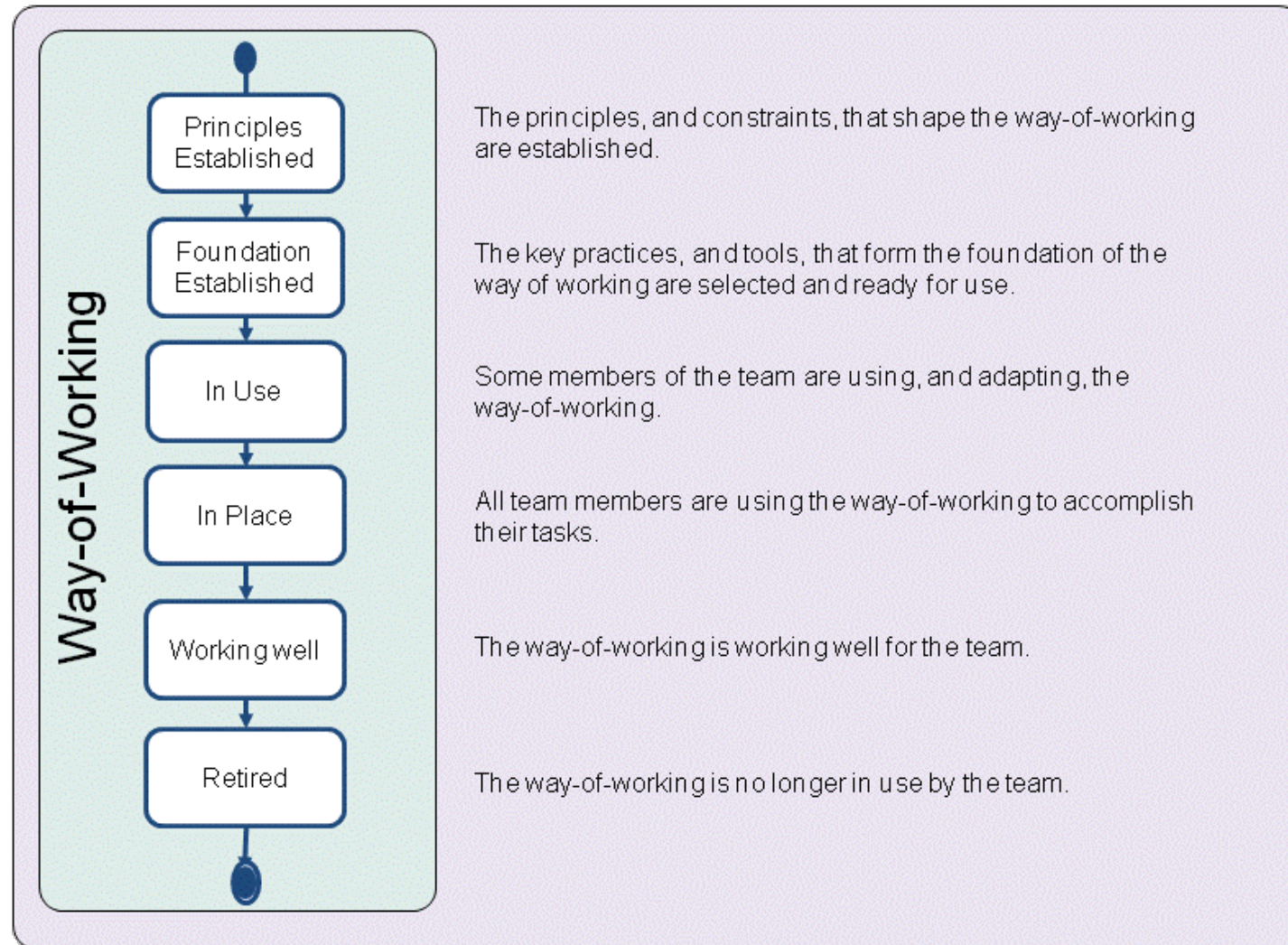
Team



Work



Way of Working



What is the real situation

Requirements

Requirements	Requirements	Requirements	Requirements	Requirements	Requirements
Conceived	Bounded	Coherent	Acceptable	Addressed	Fulfilled
<ul style="list-style-type: none"> The need for a new system is clear Users are identified Initial sponsors are identified 	<ul style="list-style-type: none"> The purpose and extent of the system are agreed Success criteria are clear Mechanisms for handling requirements are agreed Constraints and assumptions identified 	<ul style="list-style-type: none"> The big picture is clear and shared by all involved Important usage scenarios explained Priorities are clear Conflicts are addressed Impact is understood 	<ul style="list-style-type: none"> Requirements describe a solution acceptable to the stakeholders The rate of change to agreed requirements is low Value is clear 	<ul style="list-style-type: none"> Enough requirements are implemented for the system to be acceptable Stakeholders agree the system is worth making operational 	<ul style="list-style-type: none"> The system fully satisfies the requirements and the need There are no outstanding requirements items preventing completion
1 / 6	2 / 6	3 / 6	4 / 6	5 / 6	6 / 6

Software System

Software System	Software System	Software System	Software System	Software System	Software System
Architecture Selected	Usable	Demonstrable	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"> 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"> Key architecture characteristics demonstrated Relevant stakeholders agree architecture is appropriate Critical interface and system configurations exercised 	<ul style="list-style-type: none"> User documentation available Stakeholder representatives accept system Stakeholder representatives want to make system operational 	<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	3 / 6	2 / 6	4 / 6	5 / 6	6 / 6

Work

Work	Work	Work	Work	Work	Work
Initiated	Prepared	Started	Under Control	Concluded	Closed
<ul style="list-style-type: none"> Work initiator known Work constraints clear Sponsorship and funding model clear Priority of work clear 	<ul style="list-style-type: none"> Cost & effort estimated Funding and resources to start work in place Acceptance criteria understood Governance procedures agreed Risk exposure understood Dependencies clear 	<ul style="list-style-type: none"> 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 	<ul style="list-style-type: none"> Work going well, risks being managed 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

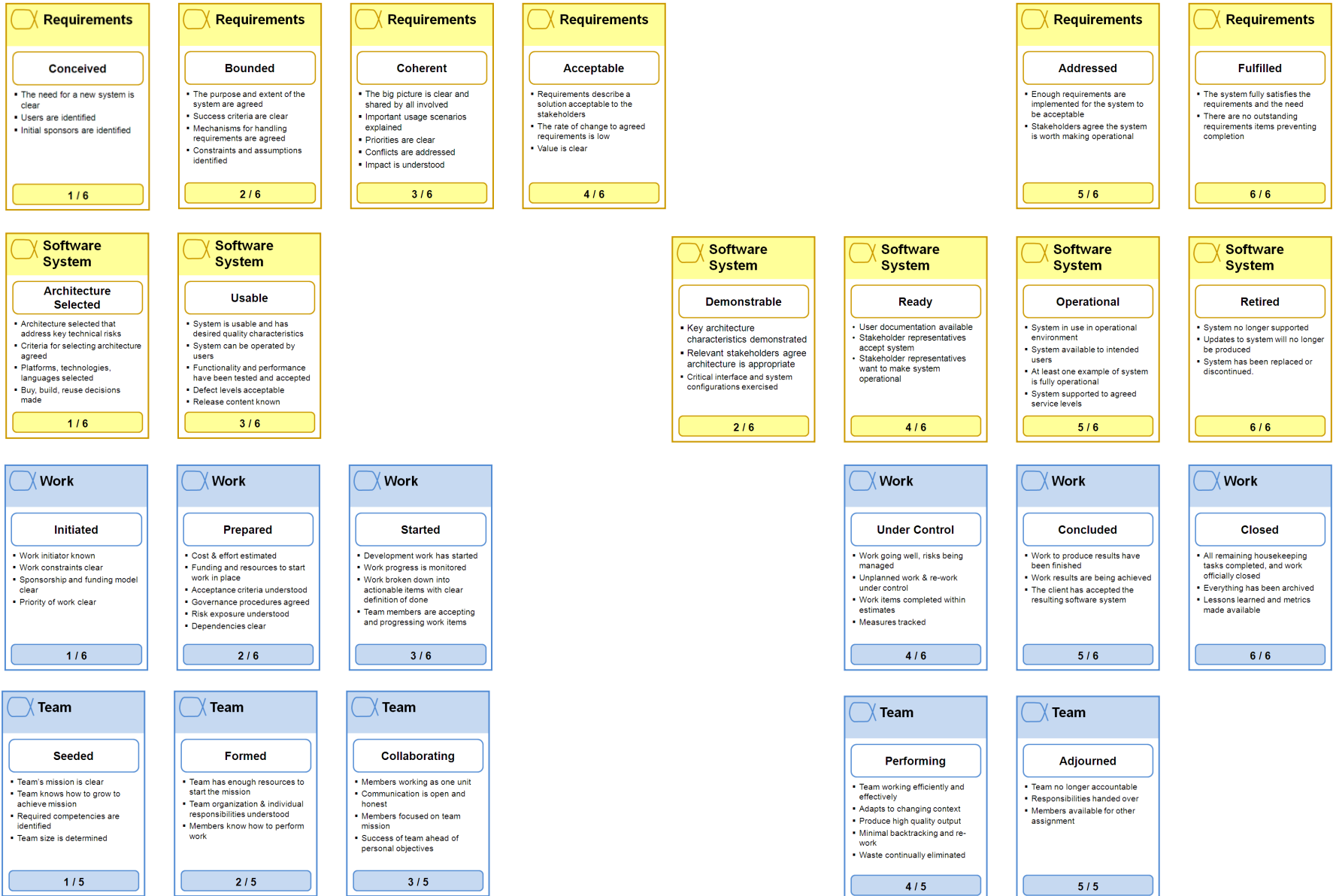
Team

Team	Team	Team	Team	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

Plan: Determine Current State



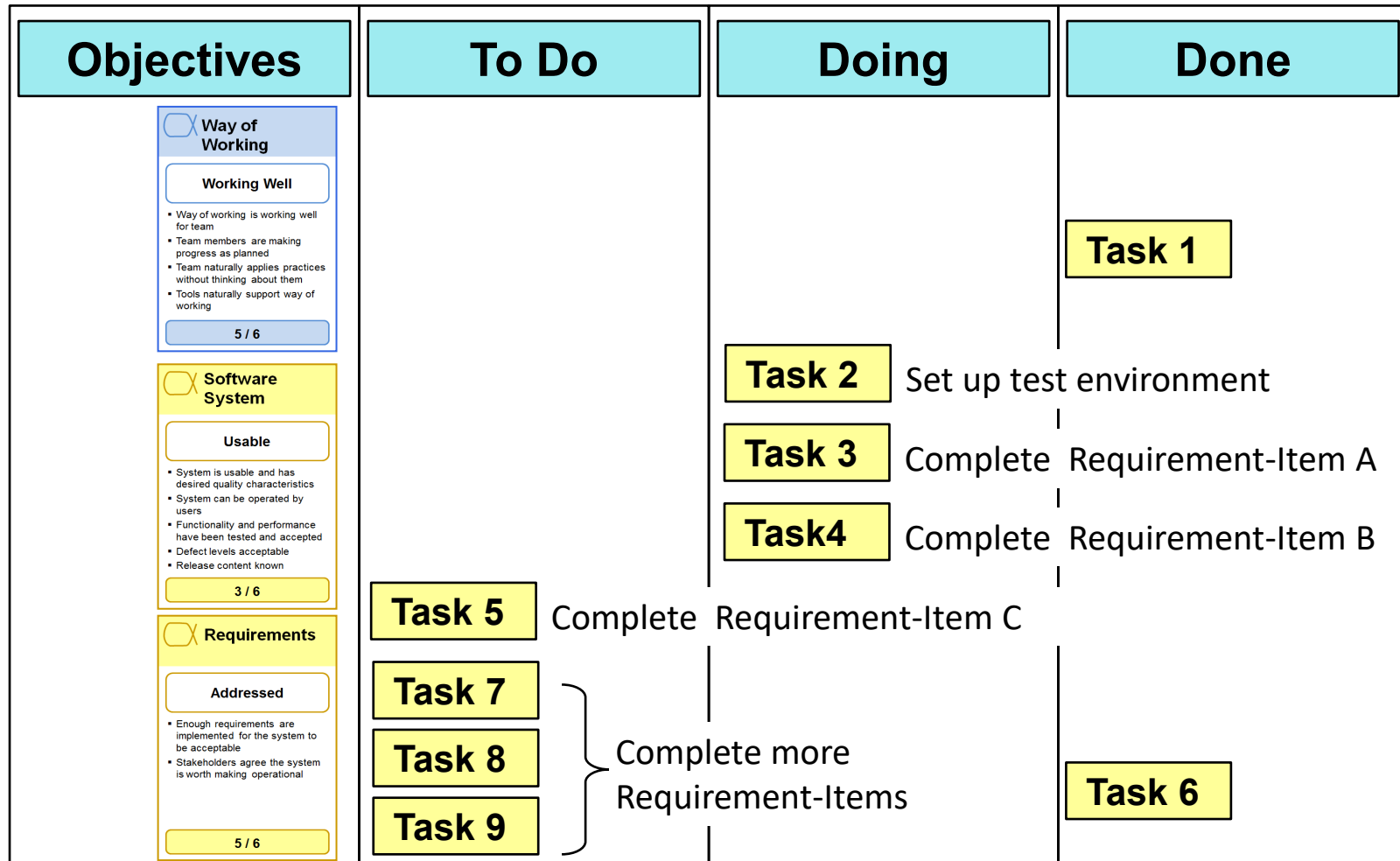
Identify States by Applying State Cards



Tasks and Sub-Alphas

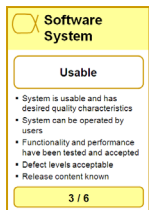
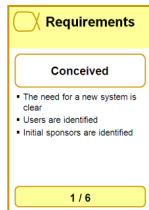
2022-02-14

23

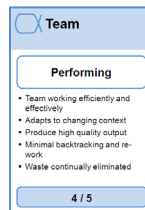
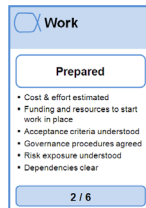
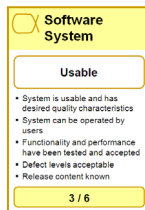
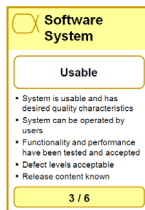
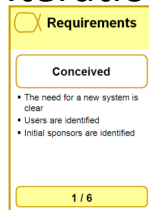


Exercise: How would you like your life-cycle?

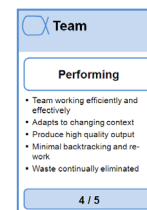
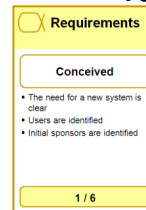
Prestudy



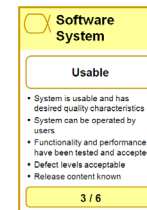
Iteration1



Iteration2

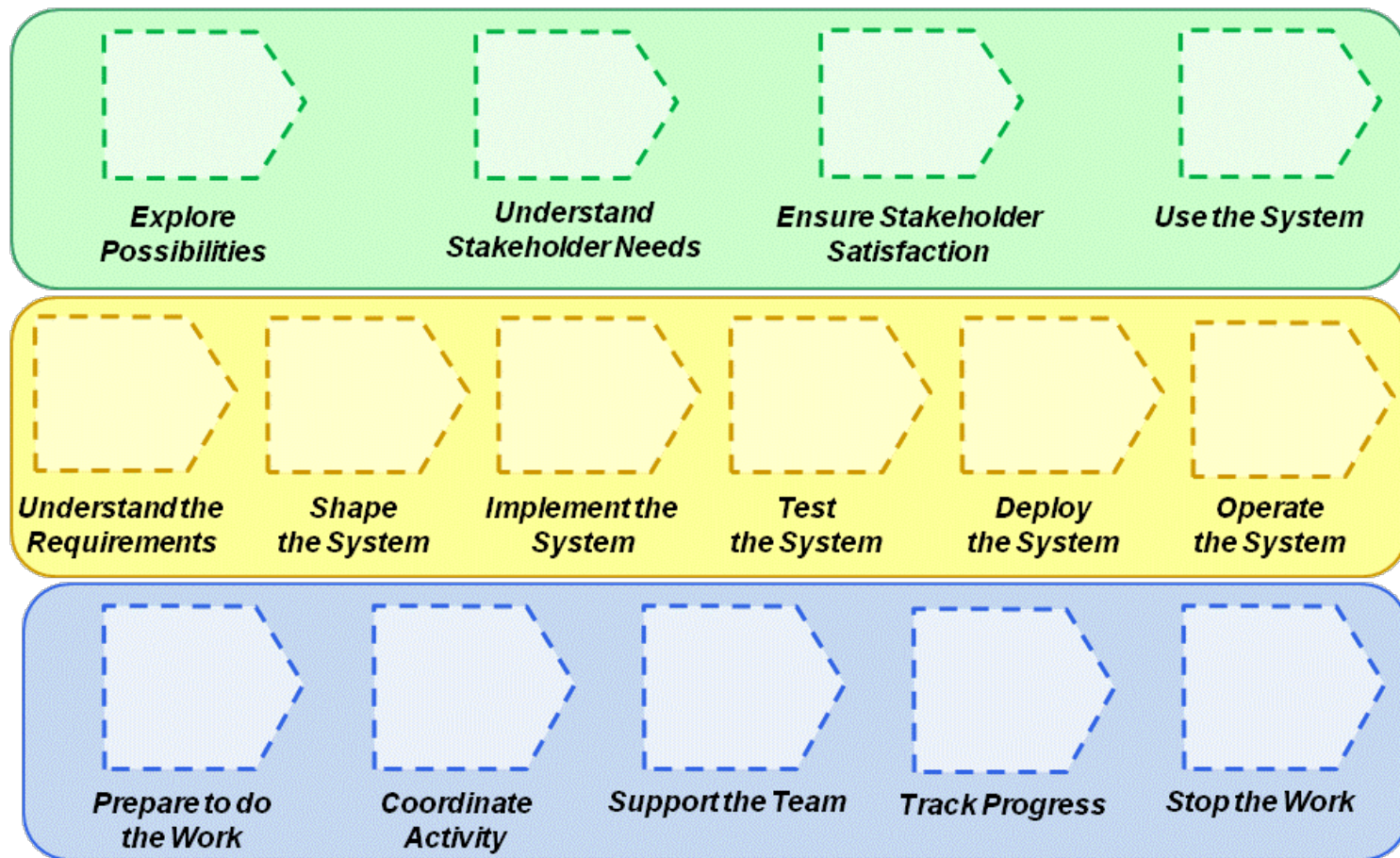


Iteration3



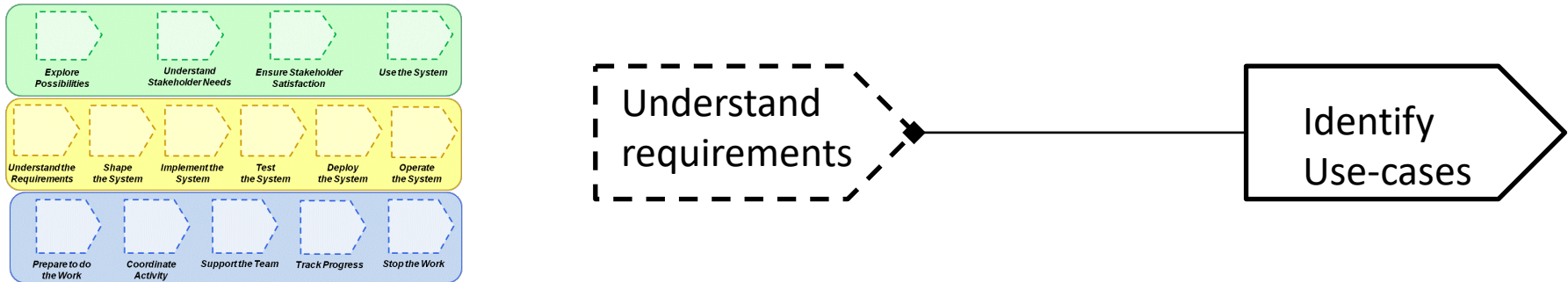
We will use
the gym system
example

Activity spaces: things to do



Classification of concrete Activities

- From earlier practice and/or theoretical studies

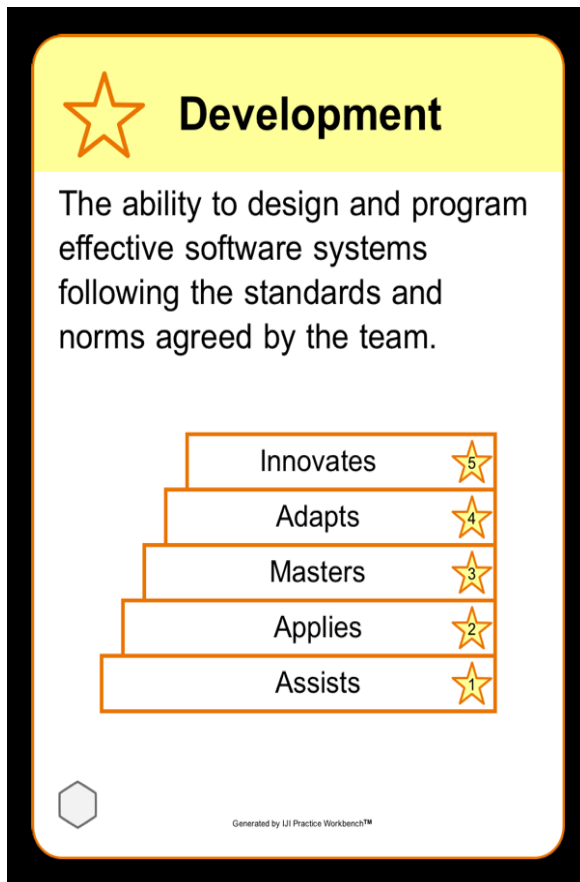


- Some are specified in a document
- Some are specified on a card
- Some are just mentioned
- Some are unspoken, common-ware

Kernel competencies



Levels of competencies



Assists Demonstrates a basic understanding of the concepts and can follow instructions.

Applies Able to apply the concepts in simple contexts by routinely applying the experience gained so far.

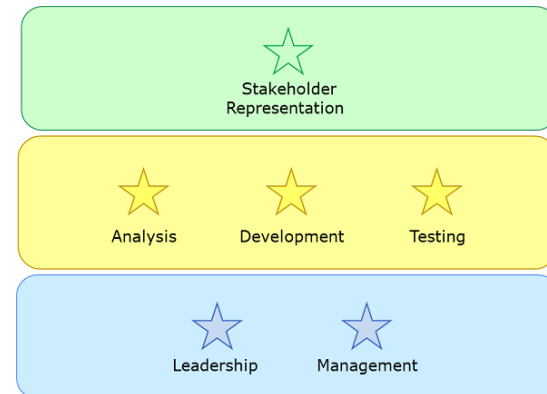
Masters Able to apply the concepts in most contexts and has the experience to work without supervision.

Adapts Able to apply judgment on when and how to apply the concepts to more complex contexts. Can enable others to apply the concepts.

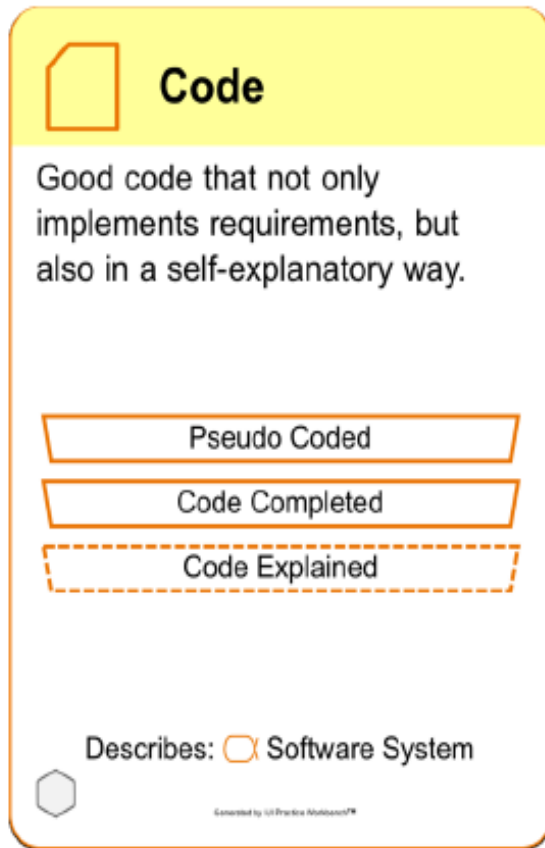
Innovates A recognized expert, able to extend the concepts to new contexts and inspire others.

Practical usage

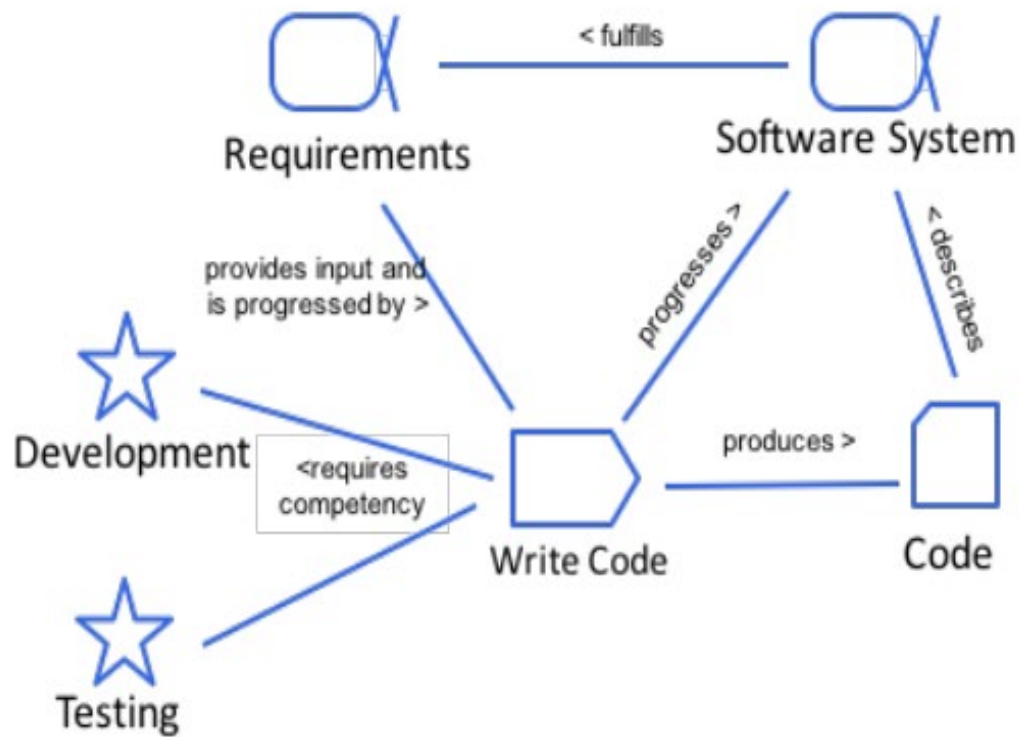
- Make a rating of competency levels needed for the roles
- Make an (honest) individual rating
- Assign the best-fit roles
- Make a gap analysis
- Develop an education plan



Work product



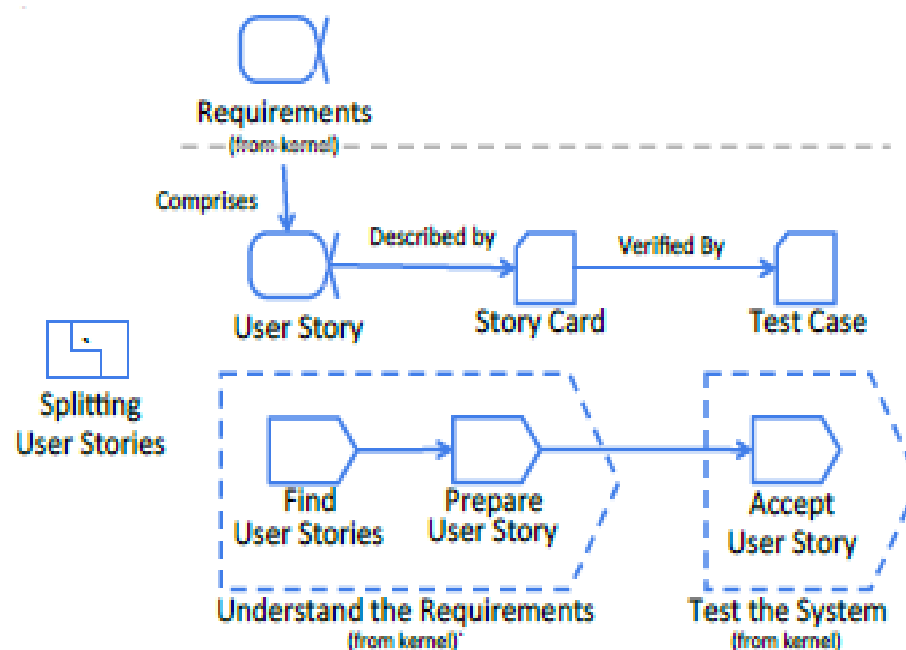
Snap-shot of relations between elements



Exercise: Essentializing a practice

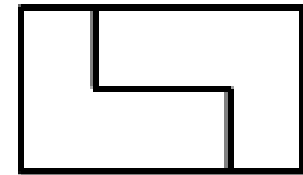
- *A repeatable approach to doing something with a specific purpose in mind*
- Identify elements
- Identify things to watch, the alphas
- Draft relationships
- Add details
- Produce cards

Example: User story



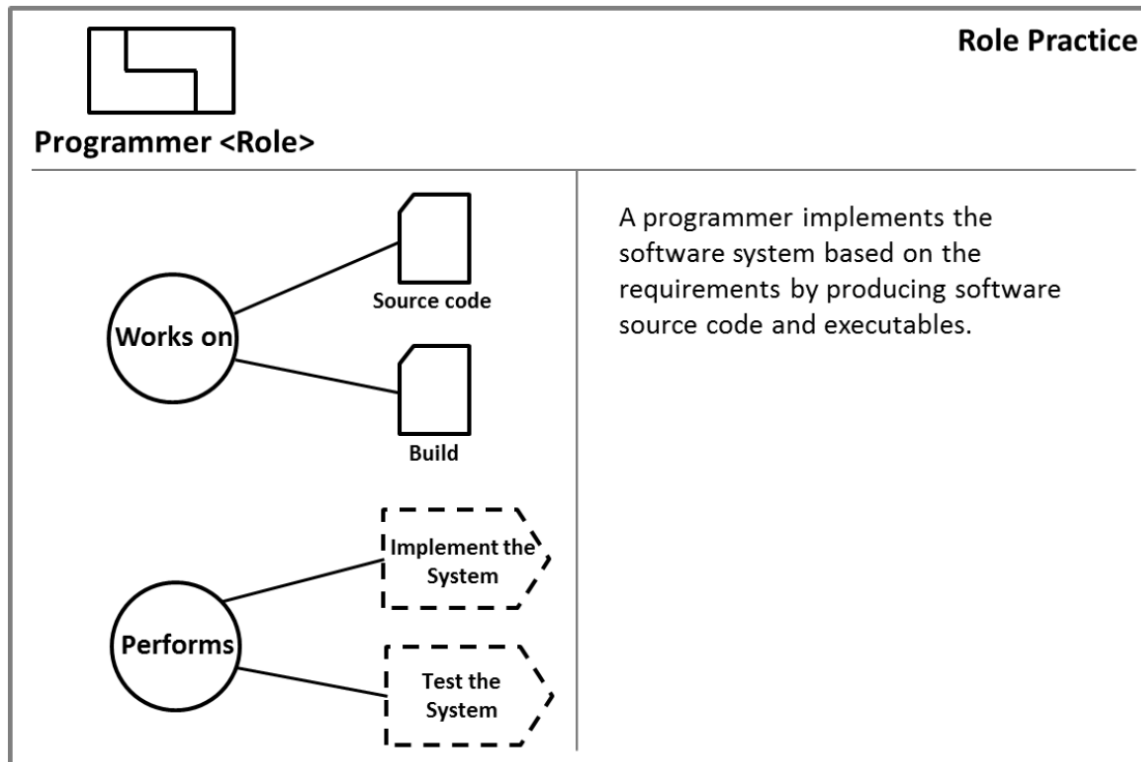
Patterns describe (complex) solutions to typical problems

- Structure, e.g. organization of working space
- Resources, e.g. tools
- Roles, e.g. programmer
- Checkpoints, e.g. a mile stone



name

Example of a role pattern card



Exercise: Describe the practice of having a kick-off meeting

Exercise: Describe the practice of automated unit testing

Good links

- Material:

<http://www.software-engineering-essentialized.com/home>

The standard:

<https://www.omg.org/spec/Essence/>

- Browse the library of Essence 365:

<https://practicelibrary.ivarjacobson.com/start>

(read-only, requires login)

Essence Kernel/Kristian Sandahl

www.liu.se