

The Essence Kernel

Quick Reference Guide



Version 0.3

Table of Contents

1	Introduction	3
1.1	What is Essence?.....	3
1.2	Presenting the Essence Kernel	4
2	An Overview of the Alphas	5
2.1	Stakeholders	7
2.2	Opportunity	9
2.3	Requirements.....	11
2.4	Software System.....	13
2.5	Team.....	15
2.6	Work.....	17
2.7	Way of Working.....	19
3	An Overview of the Competencies	21
3.1	Stakeholder Representation	23
3.2	Analysis.....	24
3.3	Development	25
3.4	Testing.....	26
3.5	Leadership.....	27
3.6	Management.....	28
4	An Overview of the Activity Spaces.....	29
5	Acknowledgements	30



1 Introduction

This Quick Reference Guides provides a brief introduction to the Essence Kernel. It is designed to support the use of the Essence Cards by software development teams and, in particular, the games that can be played with them. For more detailed information see the SEMAT User Guide available from www.semat.org and the full Essence Specification available from www.omg.org.

The use of the kernel and the cards has many benefits for individuals, teams and organizations. These include helping them to:

- Understand where they are
- Understand what needs to be addressed
- Track progress and health
- Keep projects in balance and avoid catastrophic failures
- Form good sprint goals and other objectives
- Form teams
- Define practice independent checkpoints, milestones and lifecycles.

The Kernel is both simple and incredibly powerful as it 1) captures the key concepts involved in software engineering, 2) allows the progress and health of any software engineering endeavor to be tracked and assessed, and 3) provides the common ground for the definition of software engineering methods and practices.

1.1 What is Essence?

Essence provides us with a thinking framework that allows teams to understand where they are and acts as a foundation for their way-of-working. At the heart of the Essence approach is the Essence Kernel – a simple state-driven model of software engineering. The Essence Kernel captures the small set of things that are universal to all software engineering endeavors; the things that a team always has to consider or work with when developing software.

The kernel contains seven key elements - *Requirements, Software System, Team, Work, Way of Working, Opportunity and Stakeholders*. Through states defined on these elements, the kernel provides an intuitive tool for practitioners to reason about the progress and health of their endeavors in a practice-independent way. To distinguish them from the many work products used to describe them these elements are called Alphas. The Alpha view is complemented with two other views 1) a competency-based view of the skill sets the team needs to be able to do them with and 2) an activity focused view of the things that teams always have to do. By populating these views with their practices, teams can quickly assemble, analyze, and share their own way-of-working.

By focusing on the essential things inherent in all software development efforts the Essence Kernel provides a simple definition of the common ground shared by all software development teams regardless of the kind of software being developed, of how the team is organized, of what practices get selected, the size of the system being produced, and the complexity of the problem being addressed.

In summary, the Essence Kernel is a stripped-down, light-weight set of definitions that captures the essence of effective, scalable software engineering in a practice independent way. It gives us a new way to look at the domain of software engineering, a new way to understand the progress and health of our development efforts, and a new way to combine practices into an effective way-of-working. It provides a common reference model all teams can use as they continuously inspect, adapt, and improve their ways of working.

1.2 Presenting the Essence Kernel

The kernel is organized into three discrete areas of concern, each focusing on a specific aspect of software engineering, and each distinguished by the use of a different color. These are:

- **Customer** – This area of concern contains everything to do with the actual use and exploitation of the software system to be produced. The area of concern and its card are colored green.
- **Solution** – This area of concern contains everything to do the specification and development of the software system. This area of concern and its cards are colored yellow.
- **Endeavor** – This area of concern contains everything to do with the team, and the way that they approach their work. This area of concern is colored blue.

Each area of concern contains a small number of:

- **Alphas** – representations of the essential things to work with. The Alphas provide descriptions of the kind of things that a team will manage, produce, and use in the process of developing, maintaining and supporting software.
- **Competencies** – representations of the key competencies required to do software engineering.
- **Activity Spaces** – representations of the essential things to do. The Activity Spaces identify and list generic challenges a team faces when developing, maintaining and supporting software systems, and the kinds of things that the team will do to meet them.

To make the Essence Kernel accessible and easy to use this guide presents the kernel elements in a number of complementary ways. The Alphas and Competencies are presented in overview form and as sets of cards with supporting definitions and checklists. The Activity Spaces are presented purely in overview form.

For example, in this guide, the Alphas are presented using Alpha overview cards, Alpha state cards and full checklists. The cards and checklists used to present the Stakeholder Alpha are shown in Figure 1 below.

Overview Card

Stakeholders

The people, groups, or organizations who affect or are affected by a software system.

The stakeholders:

- Provide the opportunity and are the source of the requirements
- Use and consume the software system
- Fund the development of the software system
- Actively represent the groups and organizations affected by the software system
- Are actively involved all the way through the endeavor
- Have representatives that collaborate with the team to reach agreement on an acceptable system

Recognized

Represented

Involved

In Agreement

Satisfied for Deployment

Satisfied in Use

State Cards

Stakeholders

In Agreement

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

Stakeholders

Represented

- Responsibilities agreed
- Representatives authorized
- Collaborative approach agreed
- Team's way of working reported & requested

Stakeholders

Involved

- Representatives actively involved in the work
- Timely feedback and decisions provided
- Changes promptly communicated

Stakeholders

Satisfied for Deployment

- Stakeholder feedback provided
- System ready for deployment

Stakeholders

Satisfied in Use

- Feedback on system use available
- System meets expectations

Full Checklist

Recognized: Stakeholders have been identified	Represented: The stakeholders are agreed and the stakeholder representatives have been appointed	Involved: The stakeholder representatives are actively involved in the work and fulfilling their responsibilities	In Agreement: The stakeholder representatives are in agreement	Satisfied for Deployment: The minimal expectations of the stakeholder representatives have been achieved	Satisfied in Use: The system has met or exceeds the minimal stakeholder expectations
<ul style="list-style-type: none"> □ Stakeholder groups identified □ Key stakeholder groups represented □ Responsibilities defined 	<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 stakeholder groups that fund, use, support, and maintain the system have been considered.</p> <p>The responsibilities of the stakeholder representatives have been defined.</p>	<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 representatives request and respect the team's way of working.</p>	<p>The stakeholder representatives have agreed upon their mutual expectations for the new deployment of the new system.</p> <p>The stakeholder representatives are happy with that involvement in the work.</p> <p>The stakeholder representatives agree that their input is valued by the team and treated with respect.</p> <p>The team members agree that their input is valued by the stakeholder representatives and treated with respect.</p> <p>The stakeholder representatives agree with how their different priorities and perspectives are being balanced to provide a clear direction for the team.</p>	<p>The stakeholder representatives provide feedback on the system from their stakeholder group perspective.</p> <p>The stakeholder representatives confirm that they agree that the system is ready for deployment.</p>	<p>The stakeholders are using the new system and providing feedback on their experiences.</p> <p>The stakeholders confirm that the new system meets their expectations.</p>

Figure 1. An Alpha overview card, and its accompanying state cards and checklist

Note: The Alpha state cards present an abbreviated view of the Alpha State checklist, useful when using the state cards in a workshop situation. The full checklists show *both the abbreviated and full views of the checklist; full checklist items that are viewed as redundant have no abbreviated version and are shown in italics.*

Sets of cards are available from www.semat.org.

2 An Overview of the Alphas

The Alphas do not stand-alone but support each other. The Alphas and their relationships are shown in Figure 2.

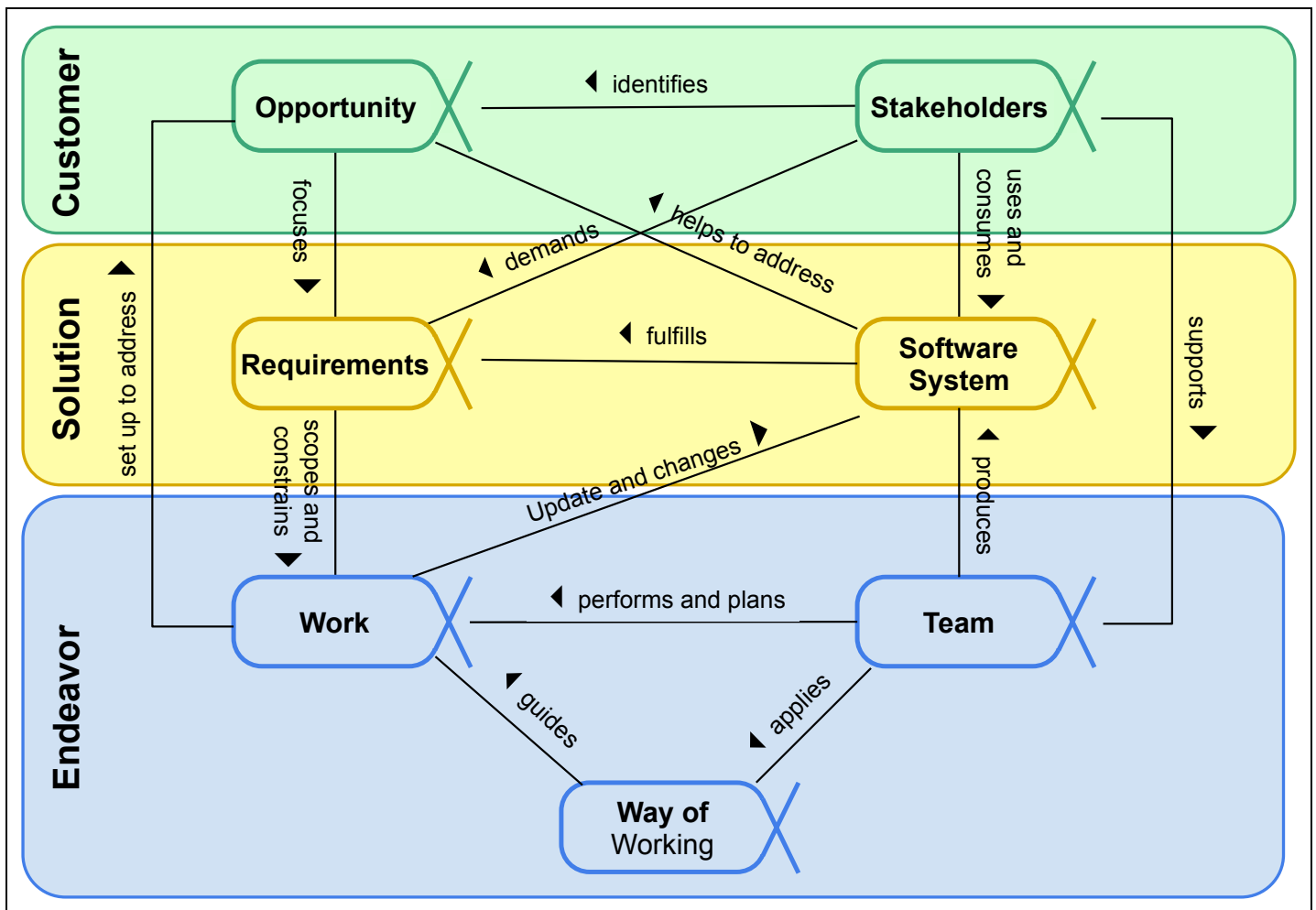


Figure 2. The Kernel Alphas

In the **customer** area of concern the team needs to understand the stakeholders and the opportunity to be addressed:

1. **Stakeholders:** The people, groups, or organizations who affect or are affected by a software system.

The stakeholders provide the opportunity and are the source of the requirements and funding for the software system. The team members are also stakeholders. The stakeholders should be involved throughout the software engineering endeavor to support the team and ensure that an acceptable software system is produced.

2. **Opportunity:** The set of circumstances that makes it appropriate to develop or change a software system.

The opportunity articulates the reason for the creation of the new, or changed, software system. It represents the team's shared understanding of the stakeholders' needs, and helps shape the requirements for the new software system by providing justification for its development.

In the **solution** area of concern the team needs to establish a shared understanding of the requirements, and implement, build, test, deploy and support a software system that fulfills them:

3. **Requirements:** What the software system must do to address the opportunity and satisfy the

stakeholders.

It is important to discover what is needed from the software system, share this understanding among the stakeholders and the team members, and use it to drive the development and testing of the new system.

4. **Software System:** A system made up of software, hardware, and data that provides its primary value by the execution of the software.

The primary product of any software engineering endeavor, a software system can be part of a larger software, hardware or business solution.

In the **endeavor** area of concern the team and its way-of-working have to be formed, and the work has to be done:

5. **Work:** Activity involving mental or physical effort done in order to achieve a result.

In the context of software engineering, work is everything that the team does to meet the goals of producing a software system matching the requirements, and addressing the opportunity, presented by the stakeholders. The work is guided by the practices that make up the team's way-of-working.

6. **Team:** A group of people actively engaged in the development, maintenance, delivery or support of a specific software system.

One or more teams plan and perform the work needed to create, update and/or change the software system.

7. **Way-of-Working:** The tailored set of practices and tools used by a team to guide and support their work.

The team evolves their way of working alongside their understanding of their mission and their working environment. As their work proceeds they continually reflect on their way of working and adapt it as necessary to their current context.

Each Alpha has a small set of states that are used when assessing progress and health. Associated with each state is a set of pre-defined checklists. The checklists are available in an abbreviated form, as shown on the cards, and in an expanded form, along with the abbreviated form as shown in tables later in this document.

It should be noted that the states are not just one-way linear progressions. Each time you reassess a state, if you do not meet all the checklist items, you can go back to a previous state. You can also iterate through the states multiple times depending on your choice of practices.

The Alphas should not be viewed as a physical partitioning of your endeavor or as just abstract work products. Rather they represent critical indicators of the things that are most important to monitor and progress. As an example, team members, while they are part of the Team Alpha, are also stakeholders, and therefore, they can also be part of the Stakeholders Alpha. In the following sections more detailed information about each of the seven alphas is provided including full state checklists and checklist abbreviations that can also be found on the alpha state cards. It is recommended that users not use the abbreviations until they have attained a solid understanding of the full checklists. To aid the use of the Essence framework some redundant checklist items have been suppressed in the abbreviated versions.

2.1 Stakeholders

Stakeholders: The people, groups, or organizations who affect or are affected by a software system.

The stakeholders provide the opportunity, and are the source of the requirements and funding for the software system. They are involved throughout the software engineering endeavor to support the team and ensure that an acceptable software system is produced.

Stakeholders are critical to the success of the software system and the work done to produce it. Their input and feedback help shape the software engineering endeavor and the resulting software system.

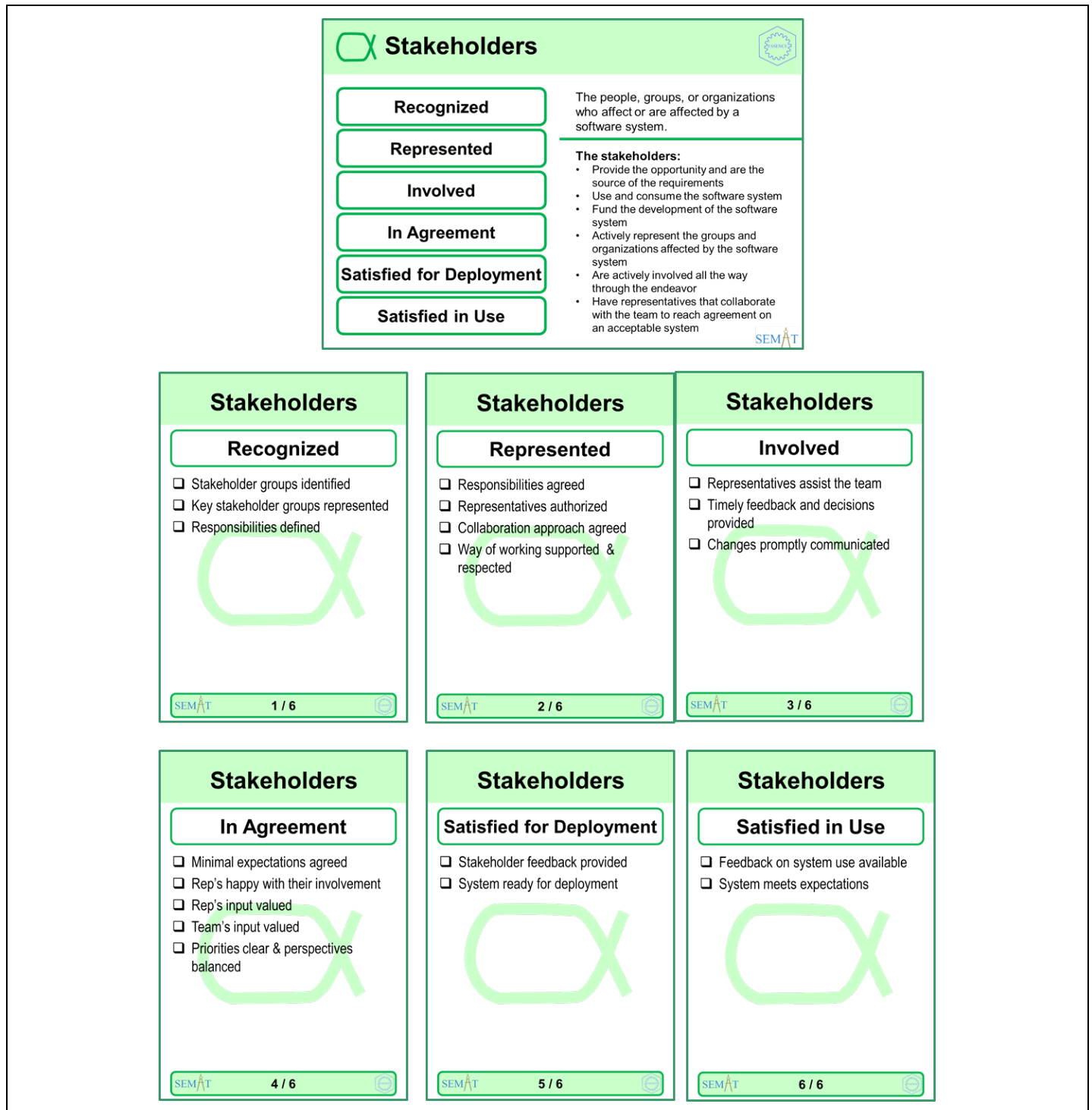


Figure 3. *Stakeholders*: Overview and alpha state cards

Table 1. Checklist for *Stakeholders*

	Recognized: Stakeholders have been identified.	
<input type="checkbox"/>	Stakeholder groups identified	All the different groups of stakeholders that are, or will be, affected by the development and operation of the software system are identified.
<input type="checkbox"/>	Key stakeholder groups represented	There is agreement on the stakeholder groups to be represented. At a minimum, the stakeholder groups that fund, use, support, and maintain the system have been considered.
<input type="checkbox"/>	Responsibilities defined	The responsibilities of the stakeholder representatives have been defined.
	Represented: The mechanisms for involving the stakeholders are agreed and the stakeholder representatives have been appointed.	
<input type="checkbox"/>	Responsibilities agreed	The stakeholder representatives have agreed to take on their responsibilities.
<input type="checkbox"/>	Representatives authorized	The stakeholder representatives are authorized to carry out their responsibilities.
<input type="checkbox"/>	Collaboration approach agreed	The collaboration approach among the stakeholder representatives has been agreed.
<input type="checkbox"/>	Way-of-working supported & respected	The stakeholder representatives support and respect the team's way of working.
	Involved: The stakeholder representatives are actively involved in the work and fulfilling their responsibilities.	
<input type="checkbox"/>	Representatives assist the team	The stakeholder representatives assist the team in accordance with their responsibilities.
<input type="checkbox"/>	Timely feedback and decisions provided	The stakeholder representatives provide feedback and take part in decision making in a timely manner.
<input type="checkbox"/>	Changes promptly communicated	The stakeholder representatives promptly communicate changes that are relevant for their stakeholder groups.
	In Agreement: The stakeholder representatives are in agreement.	
<input type="checkbox"/>	Minimal expectations agreed	The stakeholder representatives have agreed upon their minimal expectations for the next deployment of the new system.
<input type="checkbox"/>	Rep’s happy with their involvement	The stakeholder representatives are happy with their involvement in the work.
<input type="checkbox"/>	Rep’s input valued	The stakeholder representatives agree that their input is valued by the team and treated with respect.
<input type="checkbox"/>	Team’s input valued & respected	The team members agree that their input is valued by the stakeholder representatives and treated with respect.
<input type="checkbox"/>	Priorities clear & perspectives balanced	The stakeholder representatives agree with how their different priorities and perspectives are being balanced to provide a clear direction for the team.
	Satisfied for Deployment: The minimal expectations of the stakeholder representatives have been achieved.	
<input type="checkbox"/>	Stakeholder feedback provided	The stakeholder representatives provide feedback on the system from their stakeholder group perspective.
<input type="checkbox"/>	System ready for deployment	The stakeholder representatives confirm that they agree that the system is ready for deployment.
	Satisfied in Use: The system has met or exceeds the minimal stakeholder expectations.	
<input type="checkbox"/>	Feedback on system use available	Stakeholders are using the new system and providing feedback on their experiences.
<input type="checkbox"/>	System meets expectations	The stakeholders confirm that the new system meets their expectations.

2.2 Opportunity

Opportunity: The set of circumstances that makes it appropriate to develop or change a software system.

The opportunity articulates the reason for the creation of the new, or changed, software system. It represents the team’s shared understanding of the stakeholders’ needs, and helps shape the requirements for the new software system by providing justification for its development.

It is the opportunity that unites the stakeholders and provides the motivation for producing a new or updated software system. It is by understanding the opportunity that you can identify the value and the desired outcome that the stakeholders hope to realize from the use of the software system, either alone or as part of a broader business or technical solution.

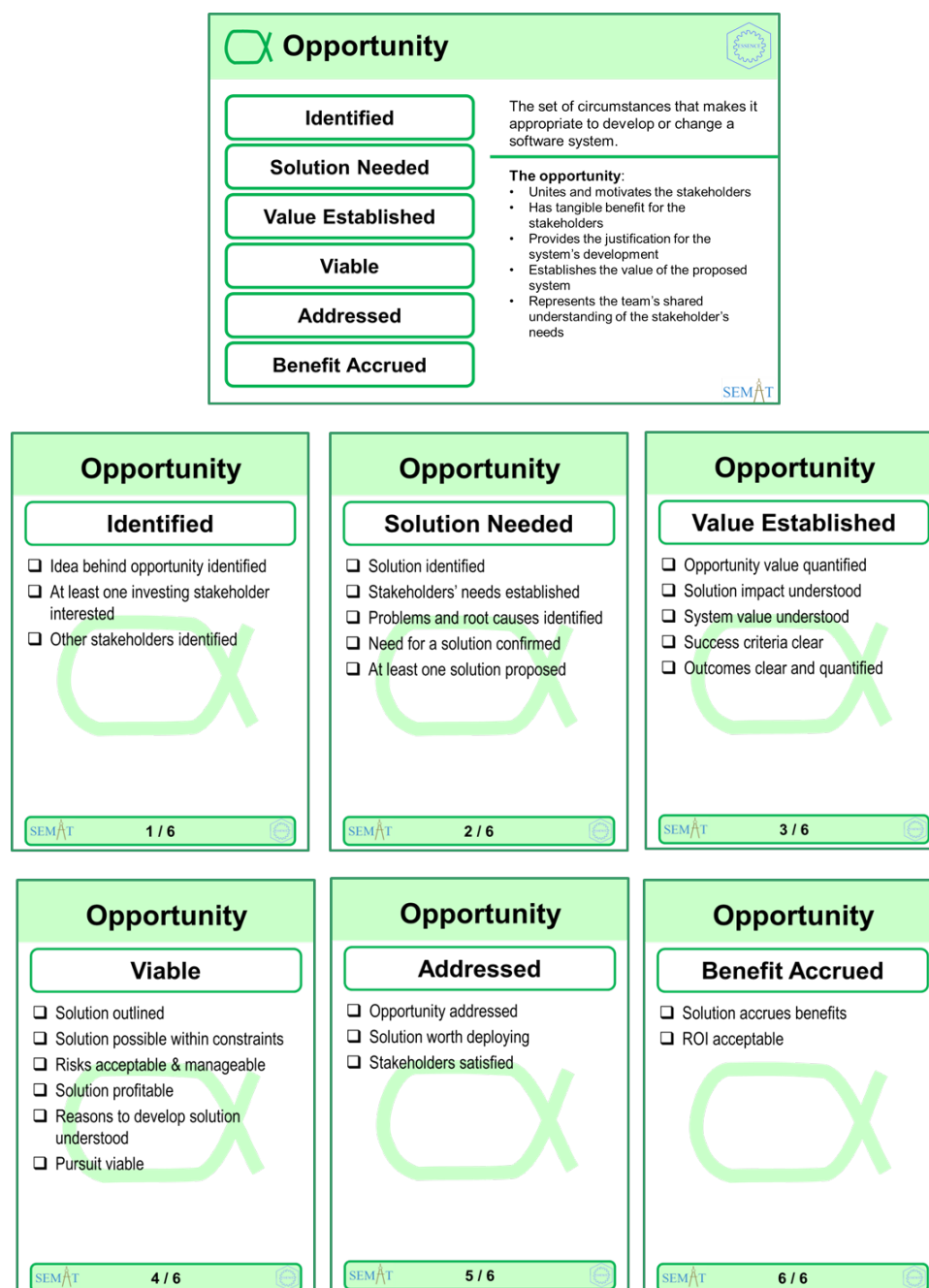


Figure 4. Opportunity: Overview and alpha state cards

Table 2. Checklist for *Opportunity*

Identified: A commercial, social or business opportunity has been identified that could be addressed by a software-based solution.		
<input type="checkbox"/>	Idea behind opportunity identified	An idea for a way of improving current ways of working, increasing market share, or applying a new or innovative software system has been identified.
<input type="checkbox"/>	At least one investing stakeholder interested	At least one of the stakeholders wishes to make an investment in better understanding the opportunity and the value associated with addressing it.
<input type="checkbox"/>	Other stakeholders identified	The other stakeholders who share the opportunity have been identified.
Solution Needed: The need for a software-based solution has been confirmed.		
<input type="checkbox"/>	Solution identified	The stakeholders in the opportunity and the proposed solution have been identified.
<input type="checkbox"/>	Stakeholders' needs established	The stakeholders' needs that generate the opportunity have been established.
<input type="checkbox"/>	Problems and root causes identified	Any underlying problems and their root causes have been identified.
<input type="checkbox"/>	Need for a solution confirmed	It has been confirmed that a software-based solution is needed.
<input type="checkbox"/>	At least one solution proposed	At least one software-based solution has been proposed.
Value Established: The value of a successful solution has been established.		
<input type="checkbox"/>	Opportunity value quantified	The value of addressing the opportunity has been quantified either in absolute terms or in returns or savings per time period (e.g., per annum).
<input type="checkbox"/>	Solution impact understood	The impact of the solution on the stakeholders is understood.
<input type="checkbox"/>	System value understood	The value that the software system offers to the stakeholders that fund and use the software system is understood.
<input type="checkbox"/>	Success criteria clear	The success criteria by which the deployment of the software system is to be judged are clear.
<input type="checkbox"/>	Outcomes clear and quantified	The desired outcomes required of the solution are clear and quantified.
Viable: It is agreed that a solution can be produced quickly and cheaply enough to successfully address the opportunity.		
<input type="checkbox"/>	Solution outlined	A solution has been outlined.
<input type="checkbox"/>	Solution possible within constraints.	The indications are that the solution can be developed and deployed within constraints.
<input type="checkbox"/>	Risks acceptable & manageable	The risks associated with the solution are acceptable and manageable.
<input type="checkbox"/>	Solution profitable	The indicative (ball-park) costs of the solution are less than the anticipated value of the opportunity.
<input type="checkbox"/>	Reasons to develop solution understood	The reasons for the development of a software-based solution are understood by all members of the team. <i>It is clear that the pursuit of the opportunity is viable.</i>
Addressed: A solution has been produced that demonstrably addresses the opportunity.		
<input type="checkbox"/>	Opportunity addressed	A usable system that demonstrably addresses the opportunity is available.
<input type="checkbox"/>	Solution worth deploying	The stakeholders agree that the available solution is worth deploying.
<input type="checkbox"/>	Stakeholders satisfied	The stakeholders are satisfied that the solution produced addresses the opportunity.
Benefit Accrued: The operational use or sale of the solution is creating tangible benefits.		
<input type="checkbox"/>	Solution accrues benefits	The solution has started to accrue benefits for the stakeholders.
<input type="checkbox"/>	ROI acceptable	The return-on-investment profile is at least as good as anticipated.

2.3 Requirements

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

It is important to discover what is needed from the software system, share this understanding among the stakeholders and the team members, and use it to drive the development and testing of the new system.

The requirements are captured as a set of requirement items. The requirement items can be communicated and recorded in various forms and at various levels of detail. It is important that the overall state of the requirements is understood as well as the state of the individual requirement items. Amongst other things this will help you tell when the system is finished, and judge whether or not an individual requirement item is in scope.

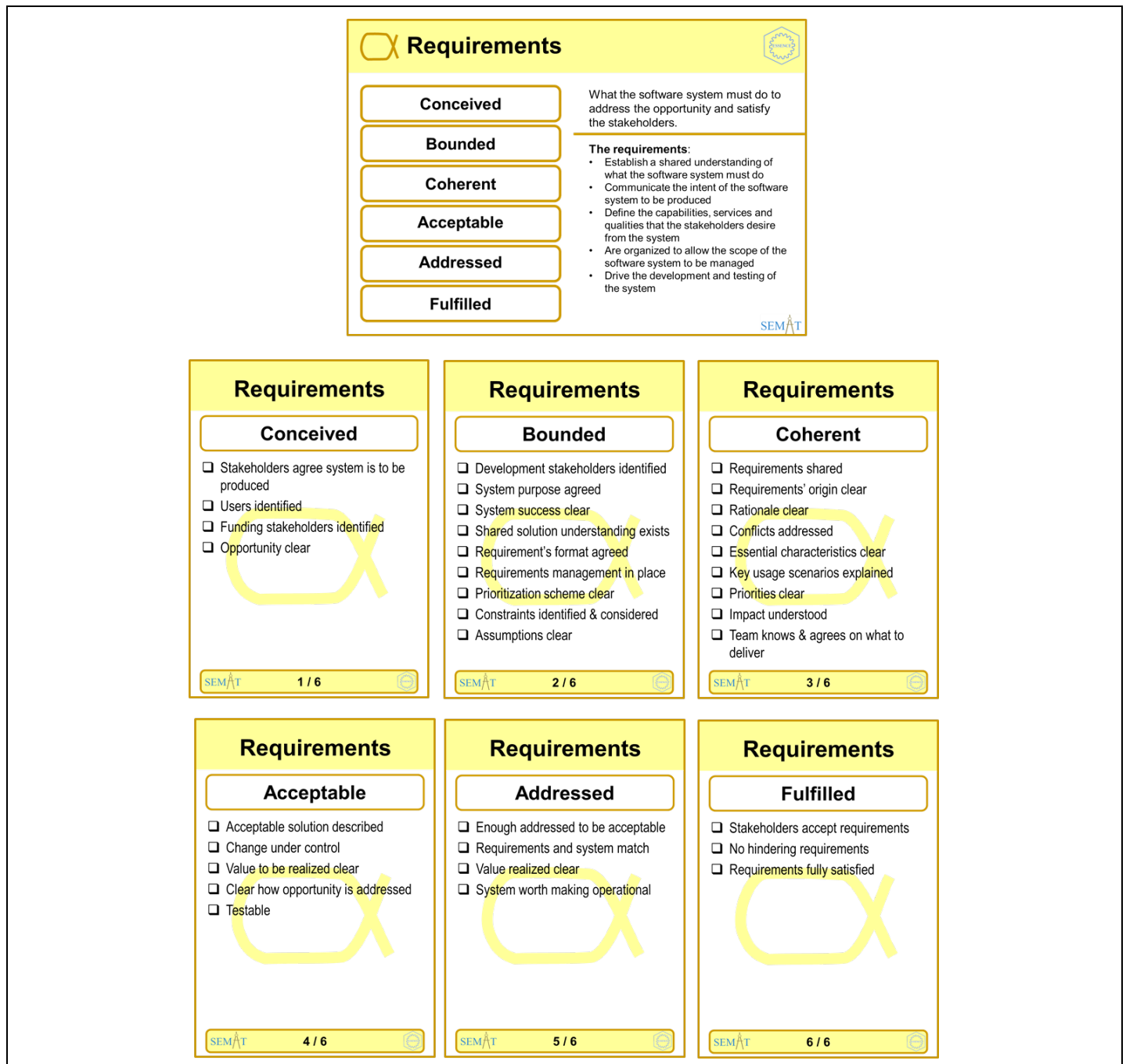


Figure 5. *Requirements*: Overview and alpha state cards

Table 3. Checklist for *Requirements*

Conceived: The need for a new system has been agreed.		
<input type="checkbox"/> Stakeholders agree system is to be produced <input type="checkbox"/> Users identified <input type="checkbox"/> Funding stakeholders identified <input type="checkbox"/> Opportunity clear		<p>The initial set of stakeholders agrees that a system is to be produced.</p> <p>The stakeholders that will use the new system are identified.</p> <p>The stakeholders that will fund the initial work on the new system are identified.</p> <p>There is a clear opportunity for the new system to address.</p>
Bounded: The purpose and theme of the new system are clear.		
<input type="checkbox"/> Development stakeholders identified <input type="checkbox"/> System purpose agreed <input type="checkbox"/> System success clear <input type="checkbox"/> Shared solution understanding exists <input type="checkbox"/> Requirements' format agreed <input type="checkbox"/> Requirements management in place <input type="checkbox"/> Prioritization scheme clear <input type="checkbox"/> Constraints identified & considered <input type="checkbox"/> Assumptions clearly		<p>The stakeholders involved in developing the new system are identified.</p> <p>The stakeholders agree on the purpose of the new system.</p> <p>It is clear what success is for the new system.</p> <p>The stakeholders have a shared understanding of the extent of the proposed solution.</p> <p>The way the requirements will be described is agreed upon.</p> <p>The mechanisms for managing the requirements are in place.</p> <p>The prioritization scheme is clear.</p> <p>Constraints are identified and considered.</p> <p>Assumptions are clearly stated.</p>
Coherent: The requirements provide a consistent description of the essential characteristics of the new system.		
<input type="checkbox"/> Requirements shared <input type="checkbox"/> Requirements' origin clear <input type="checkbox"/> Rationale clear <input type="checkbox"/> Conflicts addressed <input type="checkbox"/> Essential characteristics clear <input type="checkbox"/> Key usage scenarios explained <input type="checkbox"/> Priorities clear <input type="checkbox"/> Impact understood <input type="checkbox"/> Team knows & agrees on what to deliver		<p>The requirements are captured and shared with the team and the stakeholders.</p> <p>The origin of the requirements is clear.</p> <p>The rationale behind the requirements is clear.</p> <p>Conflicting requirements are identified and attended to.</p> <p>The requirements communicate the essential characteristics of the system to be delivered.</p> <p>The most important usage scenarios for the system can be explained.</p> <p>The priority of the requirements is clear.</p> <p>The impact of implementing the requirements is understood.</p> <p>The team understands what has to be delivered and agrees to deliver it.</p>
Acceptable: The requirements describe a system that is acceptable to the stakeholders.		
<input type="checkbox"/> Acceptable solution described <input type="checkbox"/> Change under control <input type="checkbox"/> Value to be realized clear <input type="checkbox"/> Clear how opportunity addressed <input type="checkbox"/> Testable		<p>The stakeholders accept that the requirements describe an acceptable solution.</p> <p>The rate of change to the agreed requirements is relatively low and under control.</p> <p>The value provided by implementing the requirements is clear.</p> <p>The parts of the opportunity satisfied by the requirements are clear.</p> <p>The requirements are testable.</p>
Addressed: Enough of the requirements have been addressed to satisfy the need for a new system in a way that is acceptable to the stakeholders.		
<input type="checkbox"/> Enough addressed to be acceptable <input type="checkbox"/> Requirements and system match <input type="checkbox"/> Value realized clear <input type="checkbox"/> System worth making operational		<p>Enough of the requirements are addressed for the resulting system to be acceptable to the stakeholders.</p> <p>The stakeholders accept the requirements as accurately reflecting what the system does and does not do.</p> <p>The set of requirement items implemented provide clear value to the stakeholders.</p> <p>The system implementing the requirements is accepted by the stakeholders as worth making operational.</p>
Fulfilled: The requirements that have been addressed fully satisfy the need for a new system.		
<input type="checkbox"/> Stakeholders accept requirements <input type="checkbox"/> No hindering requirements <input type="checkbox"/> Requirements fully satisfied		<p>The stakeholders accept the requirements as accurately capturing what they require to fully satisfy the need for a new system.</p> <p>There are no outstanding requirement items preventing the system from being accepted as fully satisfying the requirements.</p> <p>The system is accepted by the stakeholders as fully satisfying the requirements.</p>

2.4 Software System

Software System: A system made up of software, hardware, and data that provides its primary value by the execution of the software.

A software system can be part of a larger software, hardware, business or social solution.

We use the term software system rather than software because software engineering results in more than just a piece of software. Whilst the value may well come from the software, a working software system depends on the combination of software, hardware and data to fulfill the requirements.

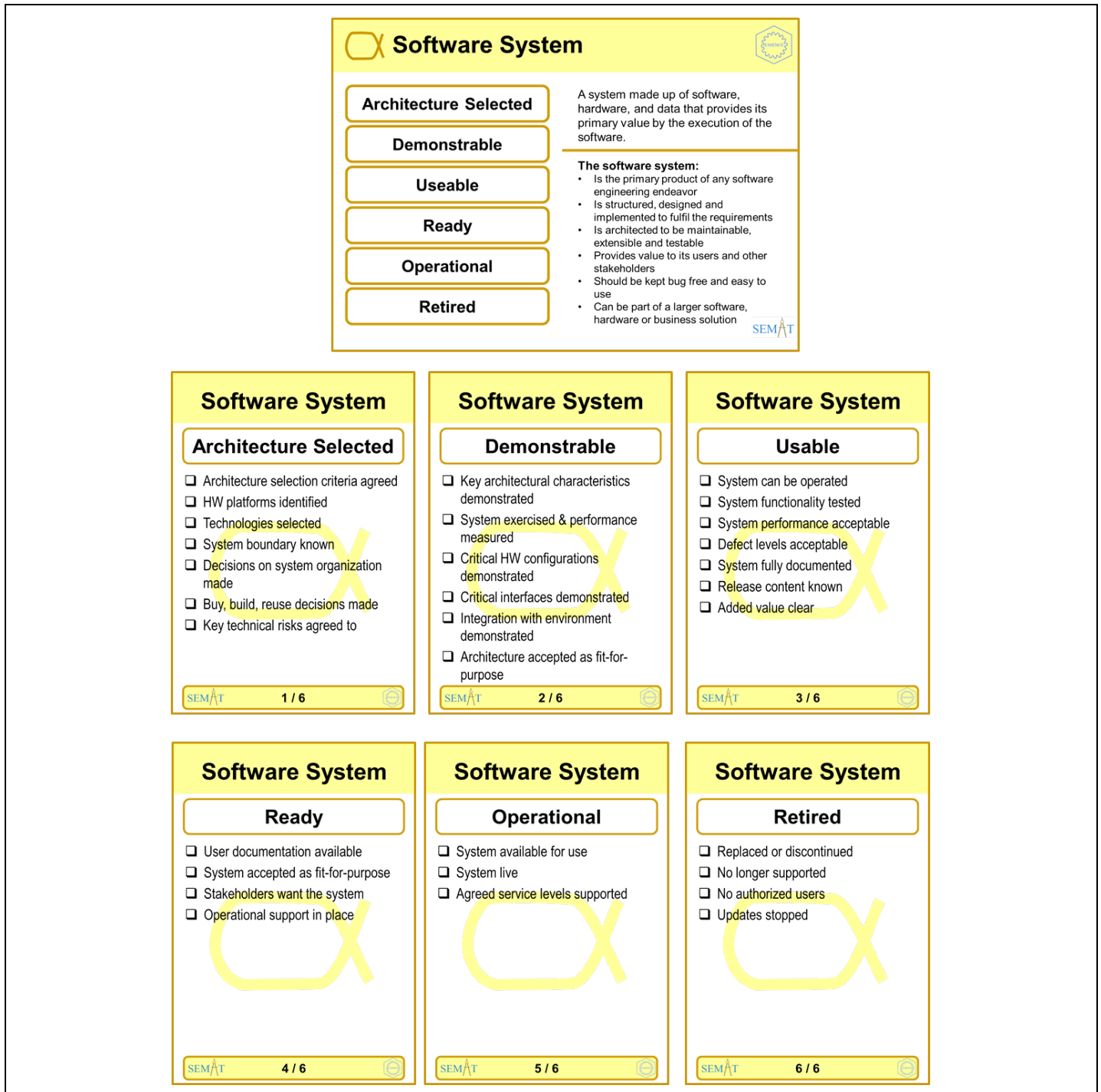


Figure 6. *Software System*: Overview and alpha state cards

Table 4. Checklist for *Software System*

Architecture Selected: An architecture has been selected that addresses the key technical risks and any applicable organizational constraints.		
<input type="checkbox"/> Architecture selection criteria agreed <input type="checkbox"/> HW platforms identified <input type="checkbox"/> Technologies selected <input type="checkbox"/> System boundary known <input type="checkbox"/> Decisions on system organization made <input type="checkbox"/> Buy, build, reuse decisions made <input type="checkbox"/> Key technical risks agreed to		<p>The criteria to be used when selecting the architecture have been agreed on.</p> <p>Hardware platforms have been identified.</p> <p>Programming languages and technologies to be used have been selected.</p> <p>System boundary is known.</p> <p>Significant decisions about the organization of the system have been made.</p> <p>Buy, build, and reuse decisions have been made.</p> <p>Key technical risks agreed to.</p>
Demonstrable: An executable version of the system is available that demonstrates the architecture is fit for purpose and supports testing.		
<input type="checkbox"/> Key architectural characteristics demonstrated <input type="checkbox"/> System exercised & performance measured <input type="checkbox"/> Critical HW configurations demonstrated <input type="checkbox"/> Critical interfaces demonstrated <input type="checkbox"/> Integration with environment demonstrated <input type="checkbox"/> Architecture accepted as fit-for-purpose		<p>Key architectural characteristics have been demonstrated.</p> <p>The system can be exercised and its performance can be measured.</p> <p>Critical hardware configurations have been demonstrated.</p> <p>Critical interfaces have been demonstrated.</p> <p>The integration with other existing systems has been demonstrated.</p> <p>The relevant stakeholders agree that the demonstrated architecture is appropriate.</p>
Usable: The system is usable and demonstrates all of the quality characteristics of an operational system.		
<input type="checkbox"/> System can be operated <input type="checkbox"/> System functionality tested <input type="checkbox"/> System performance acceptable <input type="checkbox"/> Defect levels acceptable <input type="checkbox"/> System fully documented <input type="checkbox"/> Release content known <input type="checkbox"/> Added value clear		<p>The system can be operated by stakeholders who use it.</p> <p>The functionality provided by the system has been tested.</p> <p>The performance of the system is acceptable to the stakeholders.</p> <p>Defect levels are acceptable to the stakeholders.</p> <p>The system is fully documented.</p> <p>Release content is known.</p> <p>The added value provided by the system is clear.</p>
Ready: The system (as a whole) has been accepted for deployment in a live environment.		
<input type="checkbox"/> User documentation available <input type="checkbox"/> System accepted as fit-for-purpose <input type="checkbox"/> Stakeholders want the system <input type="checkbox"/> Operational support in place		<p>Installation and other user documentation are available.</p> <p>The stakeholder representatives accept the system as fit-for-purpose.</p> <p>The stakeholder representatives want to make the system operational.</p> <p>Operational support is in place.</p>
Operational: The system is in use in an operational environment.		
<input type="checkbox"/> System available for use <input type="checkbox"/> System live <input type="checkbox"/> SLAs supported		<p>The system has been made available to the stakeholders intended to use it.</p> <p>At least one example of the system is fully operational.</p> <p>The system is fully supported to the agreed service levels.</p>
Retired: The system is no longer supported.		
<input type="checkbox"/> Replaced or discontinued <input type="checkbox"/> No longer supported <input type="checkbox"/> No authorized users <input type="checkbox"/> Updates stopped		<p>The system has been replaced or discontinued.</p> <p>The system is no longer supported.</p> <p>There are no “official” stakeholders who still use the system.</p> <p>Updates to the system will no longer be produced.</p>

2.5 Team

Team: A group of people actively engaged in the development, maintenance, delivery or support of a specific software system.

One or more teams plan and perform the work needed to create, update and/or change the software system.

Software engineering is a team sport involving the collaborative application of many different competencies and skills. To achieve high performance, team members should reflect on how well they work together, and relate this to their potential and effectiveness in achieving their mission.

Normally a team consists of several people. Occasionally, however, work may be undertaken by an individual creating software purely for their own use and entertainment. A team requires at least two people, but most of the guidance provided by the Team Alpha can also be used to help single individuals when creating software.

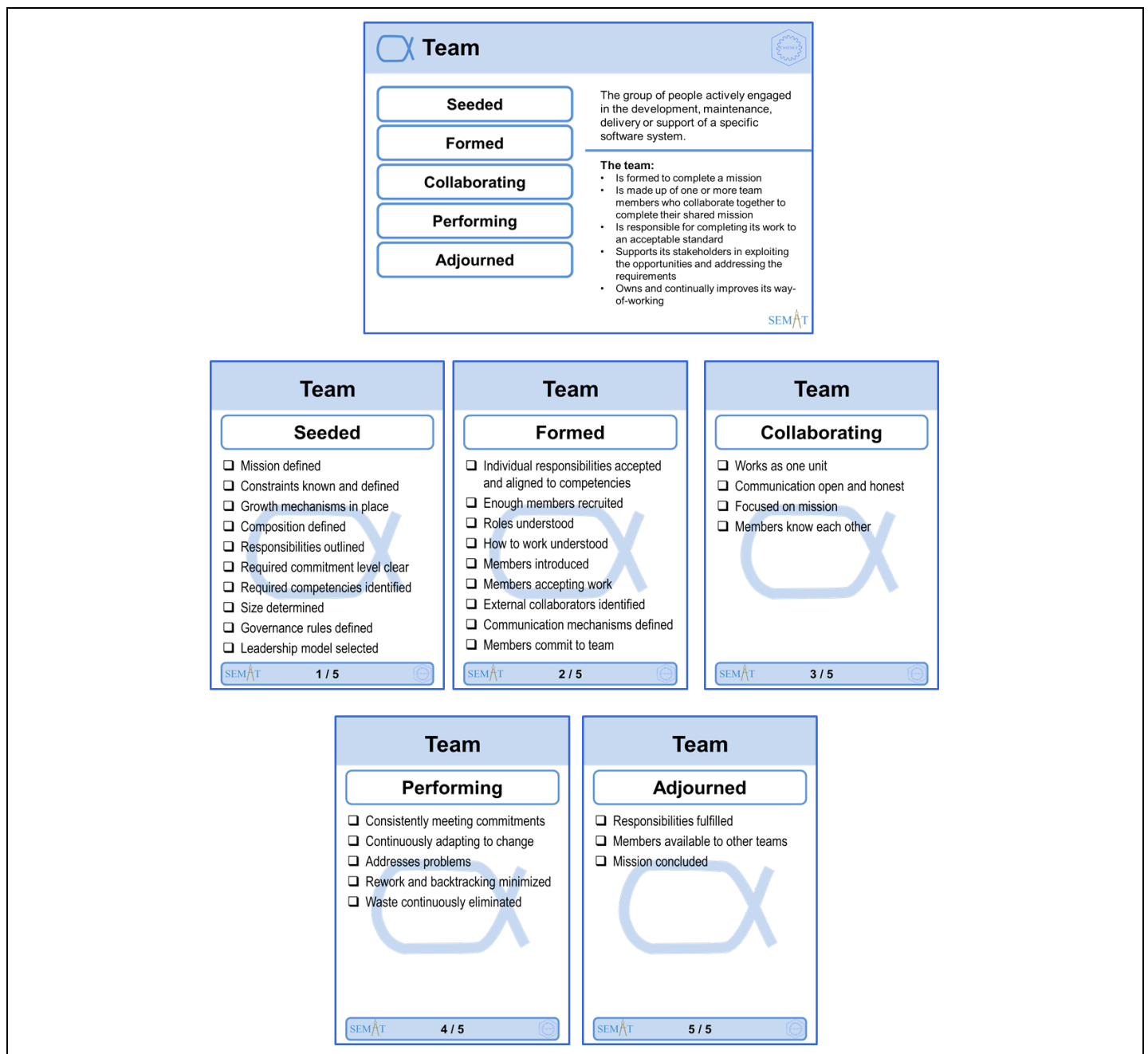


Figure 7. *Team*: Overview and alpha state cards

Table 5. Checklist for *Team*

Seeded: The team's mission is clear and the know-how needed to grow the team is in place.		
<input type="checkbox"/> Mission defined <input type="checkbox"/> Constraints known and defined <input type="checkbox"/> Growth mechanisms in place <input type="checkbox"/> Composition defined <input type="checkbox"/> Responsibilities outlined <input type="checkbox"/> Required commitment level clear <input type="checkbox"/> Required competencies identified <input type="checkbox"/> Size determined <input type="checkbox"/> Governance rules defined <input type="checkbox"/> Leadership model selected		<p>The team mission has been defined in terms of the opportunities and outcomes. Constraints on the team's operation are known. Mechanisms to grow the team are in place. The composition of the team is defined. <i>Any constraints on where and how the work is carried out are defined.</i></p> <p>The team's responsibilities are outlined. The level of team commitment is clear. Required competencies are identified. The team size is determined. Governance rules are defined. Leadership model is selected.</p>
Formed: The team has been populated with enough committed people to start the mission.		
<input type="checkbox"/> Enough members recruited <input type="checkbox"/> Roles understood <input type="checkbox"/> How to work understood <input type="checkbox"/> Members introduced <input type="checkbox"/> Individual responsibilities understood and aligned to competencies <input type="checkbox"/> Members accepting work <input type="checkbox"/> External collaborators identified <input type="checkbox"/> Communication mechanisms defined <input type="checkbox"/> Members commit to team		<p><i>Individual responsibilities are understood.</i> Enough team members have been recruited to enable the work to progress. Every team member understands how the team is organized and what their individual role is. All team members understand how to perform their work. The team members have met (perhaps virtually) and are beginning to get to know each other. The team members understand their responsibilities and how they align with their competencies. Team members are accepting work. Any external collaborators (organizations, teams and individuals) are identified. Team communication mechanisms have been defined. Each team member commits to working on the team as defined.</p>
Collaborating: The team members are working together as one unit.		
<input type="checkbox"/> Works as one unit <input type="checkbox"/> Communication open and honest <input type="checkbox"/> Focused on mission <input type="checkbox"/> Members know each other		<p>The team is working as one cohesive unit. Communication within the team is open and honest. The team is focused on achieving the team mission. The team members know each other.</p>
Performing: The team is working effectively and efficiently.		
<input type="checkbox"/> Consistently meeting commitments <input type="checkbox"/> Continuously adapting to change <input type="checkbox"/> Addresses problems <input type="checkbox"/> Rework and backtracking minimized <input type="checkbox"/> Waste continuously eliminated		<p>The team consistently meets its commitments. The team continuously adapts to the changing context. The team identifies and addresses problems without outside help. Effective progress is being achieved with minimal avoidable backtracking and reworking. Wasted work, and the potential for wasted work are continuously eliminated.</p>
Adjourned: The team is no longer accountable for carrying out its mission.		
<input type="checkbox"/> Responsibilities fulfilled <input type="checkbox"/> Members available to other teams <input type="checkbox"/> Mission concluded		<p>The team responsibilities have been handed over or fulfilled. The team members are available for assignment to other teams. No further effort is being put in by the team to complete the mission.</p>

2.6 Work

Work: Activity involving mental or physical effort done in order to achieve a result.

In the context of software engineering, work is everything that the team does to meet the goals of producing a software system matching the requirements and addressing the opportunity presented by the stakeholders. The work is guided by the practices that make up the team's way-of-working.

The ability of team members to co-ordinate, organize, estimate, complete, and share their work has a profound effect on meeting their commitments and delivering value to their stakeholders. Team members need to understand how to carry out their work, and how to recognize when the work is going well.

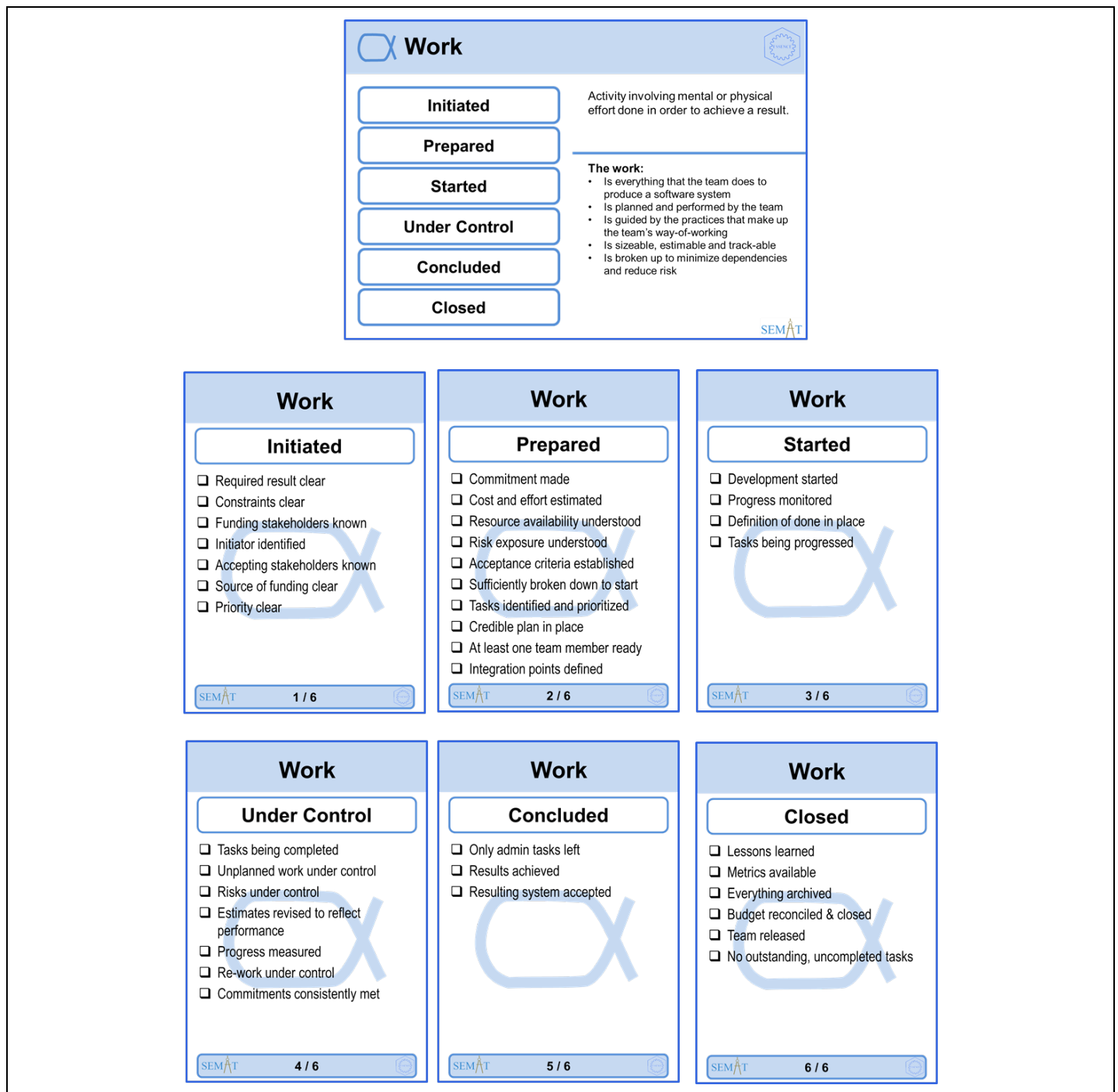


Figure 8. *Work*: Overview and alpha state cards

Table 6. Checklist for *Work*

Initiated: The work has been requested.		
<input type="checkbox"/> Required result clear		The result required of the work being initiated is clear.
<input type="checkbox"/> Constraints clear		Any constraints on the work's performance are clearly identified.
<input type="checkbox"/> Funding stakeholders known		The stakeholders that will fund the work are known.
<input type="checkbox"/> Initiator identified		The initiator of the work is clearly identified.
<input type="checkbox"/> Accepting stakeholders known		The stakeholders that will accept the results are known.
<input type="checkbox"/> Source of funding clear		The source of funding is clear.
<input type="checkbox"/> Priority clear		The priority of the work is clear.
Prepared: All pre-conditions for starting the work have been met.		
<input type="checkbox"/> Commitment made		Commitment is made.
<input type="checkbox"/> Cost and effort estimated		Cost and effort of the work are estimated.
<input type="checkbox"/> Resource availability understood		Resource availability is understood.
		<i>Governance policies and procedures are clear.</i>
<input type="checkbox"/> Risk exposure understood		Risk exposure is understood.
<input type="checkbox"/> Acceptance criteria established		Acceptance criteria are defined and agreed with client.
<input type="checkbox"/> Sufficiently broken down to start		The work is broken down sufficiently for productive work to start.
<input type="checkbox"/> Tasks identified and prioritized		Tasks have been identified and prioritized by the team and stakeholders.
<input type="checkbox"/> Credible plan in place		A credible plan is in place.
<input type="checkbox"/> Funding in place		Funding to start the work is in place.
<input type="checkbox"/> At least one team member ready to start		The team or at least some of the team members are ready to start the work.
<input type="checkbox"/> Integration points defined		Integration and delivery points are defined.
Started: The work is proceeding.		
<input type="checkbox"/> Development started		Development work has been started.
<input type="checkbox"/> Progress monitored		Work progress is monitored.
<input type="checkbox"/> Definition of done in place		The work is being broken down into actionable work items with clear definitions of done.
<input type="checkbox"/> Tasks being progressed		Team members are accepting and progressing tasks.
Under Control: The work is going well, risks are under control, and productivity levels are sufficient to achieve a satisfactory result.		
<input type="checkbox"/> Tasks being completed		Tasks are being completed.
<input type="checkbox"/> Unplanned work under control		Unplanned work is under control.
<input type="checkbox"/> Risks under control		Risks are under control as the impact if they occur and the likelihood of them occurring have been reduced to acceptable levels.
<input type="checkbox"/> Estimates revised to reflect performance		Estimates are revised to reflect the team's performance.
<input type="checkbox"/> Progress measured		Measures are available to show progress and velocity.
<input type="checkbox"/> Re-work under control		Re-work is under control.
<input type="checkbox"/> Commitments consistently met		Tasks are consistently completed on time and within their estimates.
Concluded: The work to produce the results has been concluded.		
<input type="checkbox"/> Only admin tasks left		All outstanding tasks are administrative housekeeping or related to preparing the next piece of work.
<input type="checkbox"/> Results achieved		Work results have been achieved.
<input type="checkbox"/> Resulting system accepted		The stakeholder(s) has accepted the resulting software system.
Closed: All remaining housekeeping tasks have been completed and the work has been officially closed.		
<input type="checkbox"/> Lessons learned		Lessons learned have been itemized, recorded and discussed.
<input type="checkbox"/> Metrics available		Metrics have been made available.
<input type="checkbox"/> Everything archived		Everything has been archived.
<input type="checkbox"/> Budget reconciled & closed		The budget has been reconciled and closed.
<input type="checkbox"/> Team released		The team has been released.
<input type="checkbox"/> No outstanding, uncompleted tasks		There are no outstanding, uncompleted tasks.

2.7 Way of Working

Way-of-Working: The tailored set of practices and tools used by a team to guide and support their work.

The team evolves their way of working alongside their understanding of their mission and their working environment. As their work proceeds they continually reflect on their way of working and adapt it to their current context.

Software engineering is a team sport, one that requires the whole team to collaborate effectively regardless of how the team is organized. They need to agree on a way of working that will support collaboration and guide them throughout the software engineering endeavor.

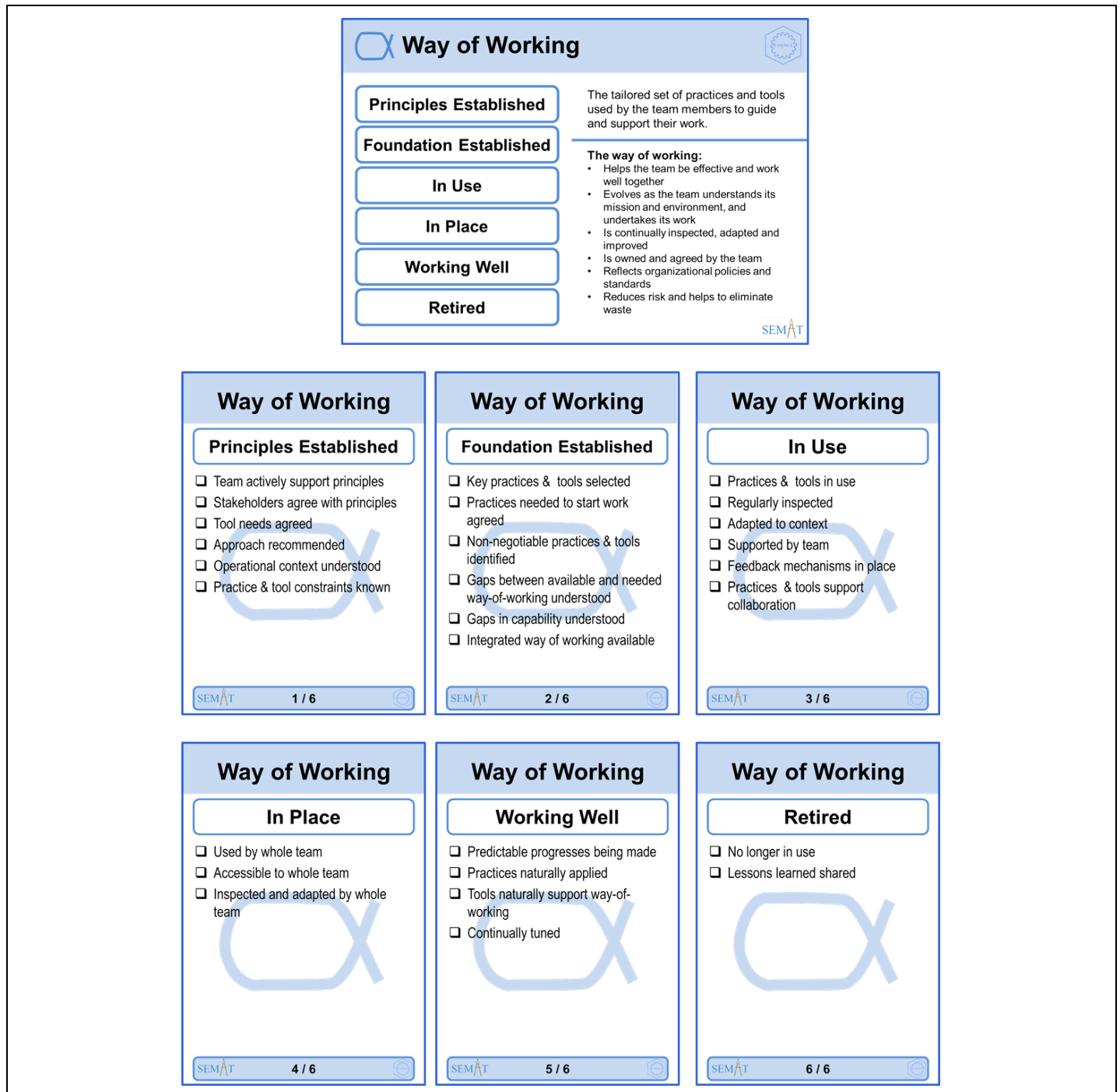


Figure 9. *Work*: Overview and alpha state cards

Table 7. Checklist for *Way-of-Working*

Principles Established: The principles, and constraints, that shape the way-of-working are established.		
<input type="checkbox"/> Team actively support principles		Principles and constraints are committed to by the team.
<input type="checkbox"/> Stakeholders agree with principles		Principles and constraints are agreed to by the stakeholders.
<input type="checkbox"/> Tool needs agreed		The tool needs of the work and its stakeholders are agreed.
<input type="checkbox"/> Approach recommended		A recommendation for the approach to be taken is available.
<input type="checkbox"/> Operational context understood		The context within which the team will operate is understood.
<input type="checkbox"/> Practice & tool constraints known		The constraints that apply to the selection, acquisition and use of practices and tools are known.
Foundation Established: The key practices, and tools, that form the foundation of the way of working are selected and ready for use.		
<input type="checkbox"/> Key practices & tools selected		The key practices and tools that form the foundation of the way-of-working are selected.
<input type="checkbox"/> Practices needed to start work agreed		Enough practices for work to start are agreed to by the team.
<input type="checkbox"/> Non-negotiable practices & tools identified		All non-negotiable practices and tools have been identified.
<input type="checkbox"/> Gaps between available and needed way-of-working understood		The gaps that exist between the practices and tools that are needed and the practices and tools that are available have been analyzed and understood.
<input type="checkbox"/> Gaps in capability understood		The capability gaps that exist between what is needed to execute the desired way of working and the capability levels of the team have been analyzed and understood.
<input type="checkbox"/> Integrated way-of-working available		The selected practices and tools have been integrated to form a usable way-of-working.
In Use: Some members of the team are using, and adapting, the way-of-working.		
<input type="checkbox"/> Practices & tools in use		The practices and tools are being used to do real work.
<input type="checkbox"/> Regularly inspected		The use of the practices and tools selected are regularly inspected.
<input type="checkbox"/> Adapted to context		The practices and tools are being adapted to the team's context.
<input type="checkbox"/> Supported by team		The use of the practices and tools is supported by the team.
<input type="checkbox"/> Feedback mechanisms in place		Procedures are in place to handle feedback on the team's way of working.
<input type="checkbox"/> Practices & tools support collaboration		The practices and tools support team communication and collaboration.
In Place: All team members are using the way of working to accomplish their work.		
<input type="checkbox"/> Used by whole team		The practices and tools are being used by the whole team to perform their work.
<input type="checkbox"/> Accessible to whole team		All team members have access to the practices and tools required to do their work.
<input type="checkbox"/> Inspected and adapted by whole team		The whole team is involved in the inspection and adaptation of the way-of-working.
Working well: The team's way of working is working well for the team.		
<input type="checkbox"/> Predictable progress being made		Team members are making progress as planned by using and adapting the way-of-working to suit their current context.
<input type="checkbox"/> Practices naturally applied		The team naturally applies the practices without thinking about them
<input type="checkbox"/> Tools naturally support way-of-working		The tools naturally support the way that the team works.
<input type="checkbox"/> Continually tuned		The team continually tunes their use of the practices and tools.
Retired: The way of working is no longer in use by the team.		
<input type="checkbox"/> No longer in use		The team's way of working is no longer being used.
<input type="checkbox"/> Lessons learned shared		Lessons learned are shared for future use.

3 An Overview of the Competencies

The kernel also provides a set of competencies that provide an overview of the key skills needed to do software engineering. These are shown in **Fehler! Verweisquelle konnte nicht gefunden werden..**

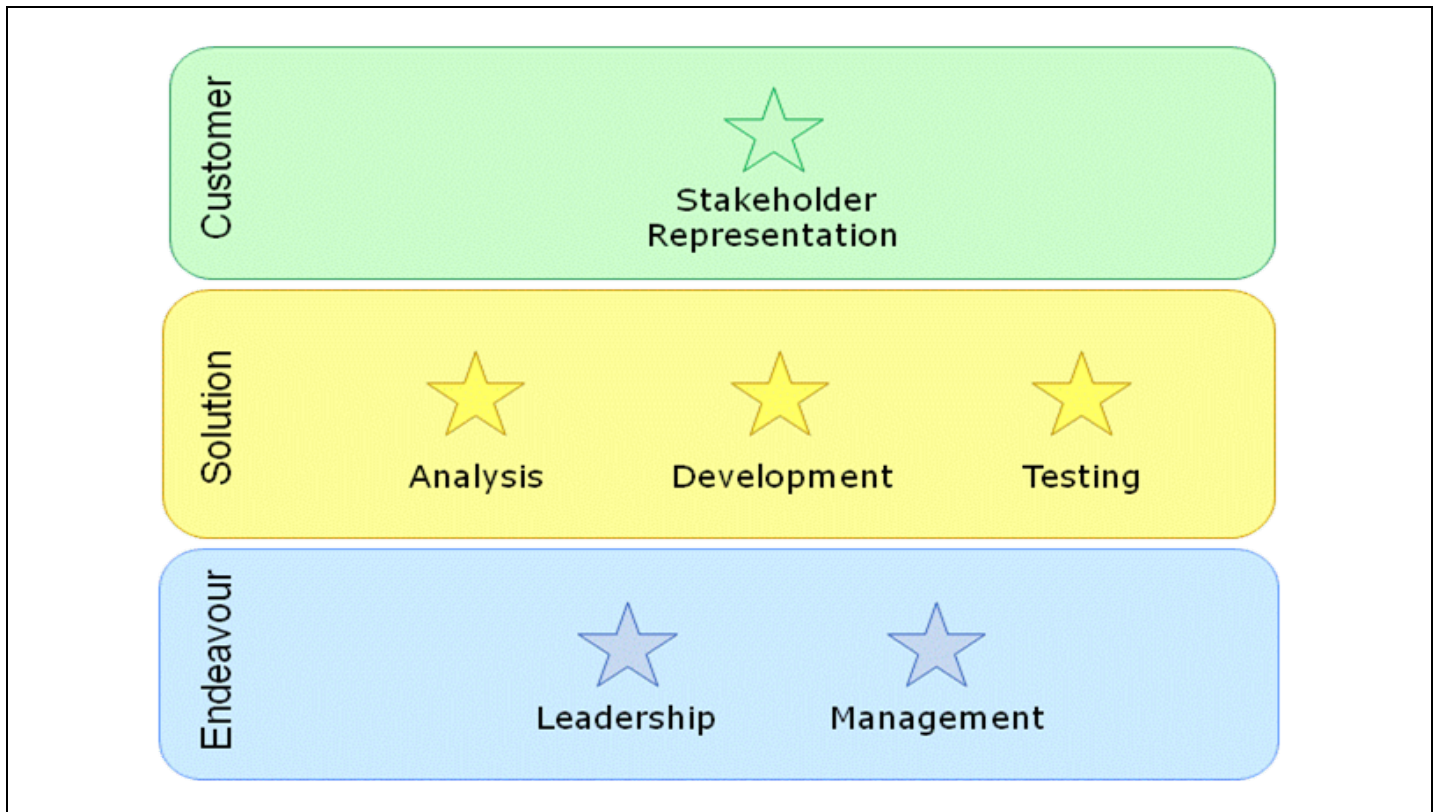


Figure 10. The Kernel Competencies

In the customer area of concern the team has to be able to demonstrate a clear understanding of the business and technical aspects of their domain and have the ability to accurately communicate the views of their stakeholders. This requires the following competencies to be available to the team:

- **Stakeholder Representation:** This competency encapsulates the ability to gather, communicate, and balance the needs of other stakeholders, and accurately represent their views.

In the **solution** area of concern the team has to be able to capture and analyze the requirements, and build and operate a software system that fulfills them. This requires the following competencies to be available to the team:

- **Analysis:** This competency encapsulates the ability to understand opportunities and their related stakeholder needs, and transform them into an agreed and consistent set of requirements.
- **Development:** This competency encapsulates the ability to design and program effective software systems following the standards and norms agreed by the team.
- **Testing:** This competency encapsulates the ability to test a system, verifying that it is usable and that it meets the requirements.

In the **endeavor** area of concern the team has to be able to organize itself and manage its workload. This requires the following competencies to be available to the team:

- **Leadership:** This competency enables a person to inspire and motivate a group of people to achieve a successful conclusion to their work and to meet their objectives.
- **Management:** This competency encapsulates the ability to coordinate, plan and track work.

Each competency has five levels of achievement. These are standard across all of the kernel competencies and shown in **Fehler! Verweisquelle konnte nicht gefunden werden.**. The table reads from left to right with the lowest level of competency shown in the first column and the highest in the last. This checklist is also available as a set of competency level cards from www.semat.org. In this case the cards and checklists have exactly the same content.

Table 8. The Generic Competency Levels

Level 1 Assists	Level 2 Applies	Level 3 Masters	Level 4 Adapts	Level 5 Innovates
<input type="checkbox"/> Has a basic understanding of the concepts <input type="checkbox"/> Is able to act in a professional manner <input type="checkbox"/> Is able to correctly respond to basic questions within his/her domain <input type="checkbox"/> Is able to perform most basic functions within the domain <input type="checkbox"/> Is able to follow instructions and complete basic tasks <input type="checkbox"/> Is able to perform tasks under supervision	<input type="checkbox"/> Is able to collaborate within the team <input type="checkbox"/> Is able to satisfy routine demands and simple work requirements <input type="checkbox"/> Can handle simple challenges with confidence <input type="checkbox"/> Is able to perform tasks under minimal supervision <input type="checkbox"/> Can handle simple work requirements but needs guidance in handling any complications or difficulties <input type="checkbox"/> Is able to reason about the context and draw sensible conclusions	<input type="checkbox"/> Is able to satisfy most demands and work requirements <input type="checkbox"/> Is able to speak the domain language with ease and accuracy <input type="checkbox"/> Is able to communicate and explain his/her work <input type="checkbox"/> Is able to give and receive constructive feedback <input type="checkbox"/> Knows the limits of his/her capability and when to call on more expert advice. <input type="checkbox"/> Works at a professional level with little or no guidance.	<input type="checkbox"/> Is able to satisfy complex demands and work requirements <input type="checkbox"/> Is able to communicate with others working outside the domain <input type="checkbox"/> Can direct and help others working within the domain <input type="checkbox"/> Is able to adapt his/her way of working to work well with others, both inside and outside their domain	<input type="checkbox"/> Has many years of experience and is currently up to date in what is happening within the domain <input type="checkbox"/> Is recognized as an expert by peers <input type="checkbox"/> Supports others in working on a complex professional level <input type="checkbox"/> Knows when to innovate or do something different and when to follow normal procedure <input type="checkbox"/> Develops innovative and effective solutions to the current challenges within the domain

The higher competency levels build upon the lower ones. An individual at Level 2 has all the traits of an individual at Level 1 as well as the additional traits required at Level 2. An individual at Level 3 has all the traits required at levels 1, 2 and 3, and so on. Individuals at levels 1 and 2 have an awareness or basic understanding of the knowledge, skills, and abilities associated with the competency. However, they do not possess the knowledge, skills, and abilities to perform the competency in difficult or complex situations and typically can only perform simple routine tasks without direction or other guidance.

Individuals at Level 3 and above have mastered this aspect of their profession. They can be trusted to integrate into and deliver the results required by the team. There are many factors that drive up the level of competency required, including:

- The size and complexity of the work.
- The size and distribution of the team.
- The size, complexity and diversity of the stakeholder community.
- The novelty of the software system being produced.
- The technical complexity of the software system
- The levels of risk facing the team.

Note that the essential skills described for each competency lists the skills needed for all competency levels. Additional skills are essential for higher competency levels, but are not described here.

3.1 Stakeholder Representation

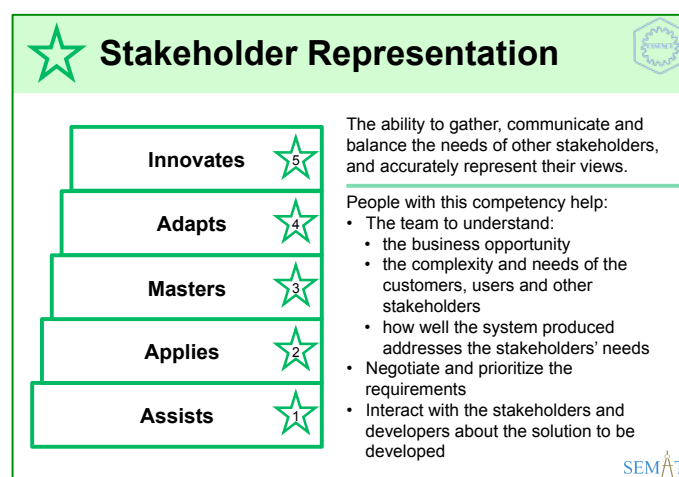


Figure 11. *Stakeholder Representation: Overview Card*

This competency encapsulates the ability to gather, communicate and balance the needs of other stakeholders, and accurately represent their views. The stakeholder representation competency is the empathic ability to stand in for and accurately reflect the opinions, rights and obligations of other stakeholders. It can be provided by an on-site customer, a product manager or a group of people from the business organization.

Table 9. *Stakeholder Representation* goals and skills

<p>People with this competency help the team to:</p> <ul style="list-style-type: none"> • Understand the business opportunity • Understand the complexity and needs of the customers, users and other stakeholders • Negotiate and prioritize the requirements • Interact with the stakeholders and developers about the solution to be developed • Understand how well the system produced addresses the stakeholders' needs 	<p>Essential skills include:</p> <ul style="list-style-type: none"> • Negotiation • Facilitation • Networking • Good written and verbal communication skills • Empathy
--	---

Competency Levels

Level 1 – Assists	Demonstrates a basic understanding of the concepts and can follow instructions.
Level 2 – Applies	Able to apply the concepts in simple contexts by routinely applying the experience gained so far.
Level 3 – Masters	Able to apply the concepts in most contexts and has the experience to work without supervision.
Level 4 – Adapts	Able to apply judgment on when and how to apply the concepts to more complex contexts. Can help others in applying the concepts.
Level 5 – Innovates	A recognized expert able to extend the concepts to new contexts and inspire others.

See 8 for the competency level checklists.

3.2 Analysis

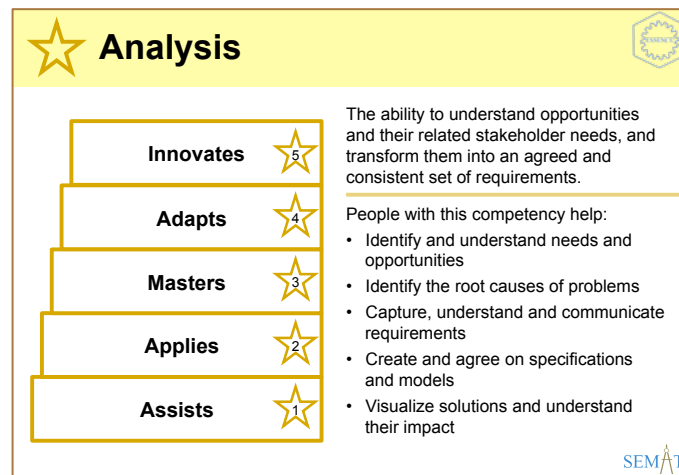


Figure 12. *Analysis*: Overview Card

This competency encapsulates the ability to understand opportunities and their related stakeholder needs, and transform them into an agreed and consistent set of requirements.

The analysis competency is the deductive ability to understand the situation, context, concepts and problems, identify appropriate high-level solutions, and evaluate and draw conclusions by applying logical thinking. It can be provided by the customer representatives, product owners, business analysts, requirement specialists or developers on the team.

Table 10. *Analysis* goals and skills

<p>People with the analytical competency help the team to:</p> <ul style="list-style-type: none"> • Identify and understand needs and opportunities. • Get to know the root causes of the problems • Capture, understand and communicate requirements. • Create and agree on specifications and models. • Visualize solutions and understand their impact 	<p>Essential skills include:</p> <ul style="list-style-type: none"> • Verbal and written communication • Ability to observe, understand, and record details • Agreement facilitation • Requirements capture • Ability to separate the whole into its component parts • Ability to see the whole by looking at what is required
--	--

Competency Levels

Level 1 – Assists	Demonstrates a basic understanding of the concepts and can follow instructions.
Level 2 – Applies	Able to apply the concepts in simple contexts by routinely applying the experience gained so far.
Level 3 – Masters	Able to apply the concepts in most contexts and has the experience to work without supervision.
Level 4 – Adapts	Able to apply judgment on when and how to apply the concepts to more complex contexts. Can help others in applying the concepts.
Level 5 – Innovates	A recognized expert able to extend the concepts to new contexts and inspire others.

See Table 8 for the competency level checklist

3.3 Development

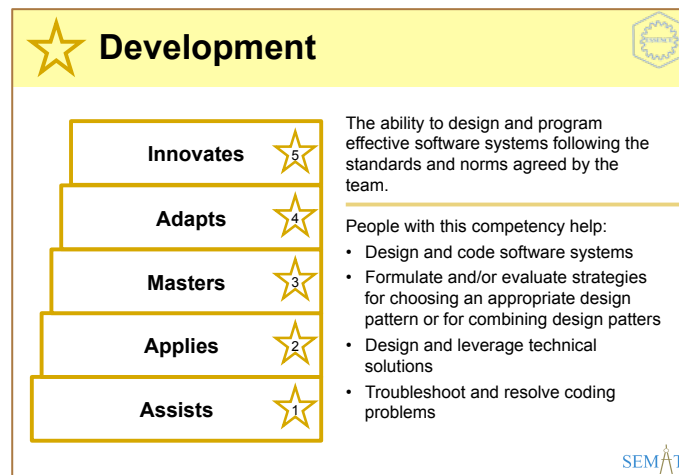


Figure 13. *Development*: Overview Card

This competency encapsulates the ability to design and program effective software systems following the standards and norms agreed by the team. It is the mental ability to conceive and produce a software system, or one of its elements, for a specific function or end. It enables a team to produce software systems that meet the requirements and can be provided by the programmers, coders, designers or architects on the team.

Table 11. *Development* goals and skills

<p>People with this competency help:</p> <ul style="list-style-type: none"> • Design and code software systems • Formulate and/or evaluate strategies for choosing an appropriate design pattern or for combining design patterns • Design and leverage technical solutions • Troubleshoot and resolve coding problems 	<p>Essential skills include:</p> <ul style="list-style-type: none"> • Knowledge of technology • Programming • Knowledge of programming languages (Ability to use the chosen programming language*) • Ability to effectively use the development environment and tools* • Critical thinking • Re-factoring • Design • Troubleshooting and problem solving skills*
--	--

*These skills are not in the OMG spec, but are considered essential for developers by SEMAT Inc.

Competency Levels

Level 1 – Assists	Demonstrates a basic understanding of the concepts and can follow instructions.
Level 2 – Applies	Able to apply the concepts in simple contexts by routinely applying the experience gained so far.
Level 3 – Masters	Able to apply the concepts in most contexts and has the experience to work without supervision.
Level 4 – Adapts	Able to apply judgment on when and how to apply the concepts to more complex contexts. Can help others in applying the concepts.
Level 5 – Innovates	A recognized expert able to extend the concepts to new contexts and inspire others.

See Table 8 for the competency level checklist

3.4 Testing

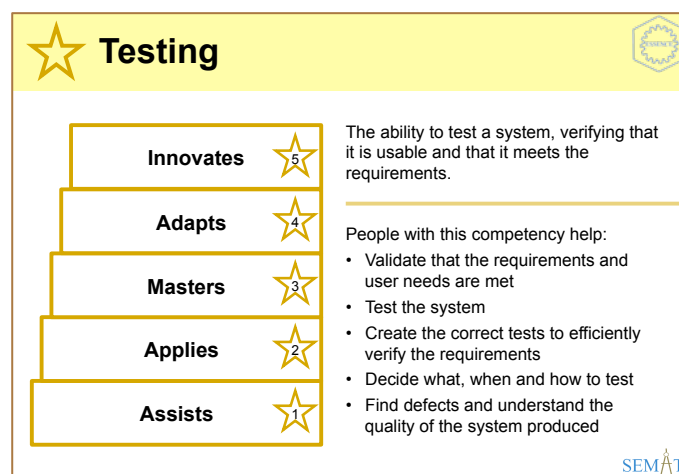


Figure 14. *Testing*: Overview Card

This competency encapsulates the ability to test a system, verifying that it is usable and that it meets the requirements. It is an observational, comparative, detective and destructive ability that enables the system to be tested. This competency can be provided by the testers, programmers, coders, designers, architects or anyone else on the team.

Table 12. *Testing* goals and skills

<p>People with this competency help:</p> <ul style="list-style-type: none"> • Validate that the requirements and user needs are met • Test the system • Create the correct tests to efficiently verify the requirements • Decide what, when and how to test • Evaluate whether the system meets the requirements • Find defects and understand the quality of the system produced 	<p>Essential skills include</p> <ul style="list-style-type: none"> • Keen observation • Exploratory and destructive thinking • Inquisitive mind • Attention to detail
---	---

Competency Levels

Level 1 – Assists	Demonstrates a basic understanding of the concepts and can follow instructions.
Level 2 – Applies	Able to apply the concepts in simple contexts by routinely applying the experience gained so far.
Level 3 – Masters	Able to apply the concepts in most contexts and has the experience to work without supervision.
Level 4 – Adapts	Able to apply judgment on when and how to apply the concepts to more complex contexts. Can help others in applying the concepts.
Level 5 – Innovates	A recognized expert able to extend the concepts to new contexts and inspire others.

See Table 8 for the competency level checklist

3.5 Leadership



Figure 15. *Leadership*: Overview Card

This competency enables a person to inspire and motivate a group of people to achieve a successful conclusion to their work and to meet their objectives. It can be provided by a Scrum Master, an appointed team leader, the more experienced members of the team, or a dedicated project manager.

Table 13. *Leadership* goals and skills

<p>People with this competency help the team to:</p> <ul style="list-style-type: none"> • Inspire people to do their work • Make sure that all team members are effective in their assignments • Make and meet their commitments • Resolve any impediments or issues holding up the team's work • Interact with stakeholders to shape priorities, report progress and respond to challenges. 	<p>Essential skills include:</p> <ul style="list-style-type: none"> • Inspiration • Motivation • Negotiation • Communication • Decision making
---	---

Competency Levels

Level 1 – Assists	Demonstrates a basic understanding of the concepts and can follow instructions.
Level 2 – Applies	Able to apply the concepts in simple contexts by routinely applying the experience gained so far.
Level 3 – Masters	Able to apply the concepts in most contexts and has the experience to work without supervision.
Level 4 – Adapts	Able to apply judgment on when and how to apply the concepts to more complex contexts. Can help others in applying the concepts.
Level 5 – Innovates	A recognized expert able to extend the concepts to new contexts and inspire others.

See Table 8 for the competency level checklist

3.6 Management

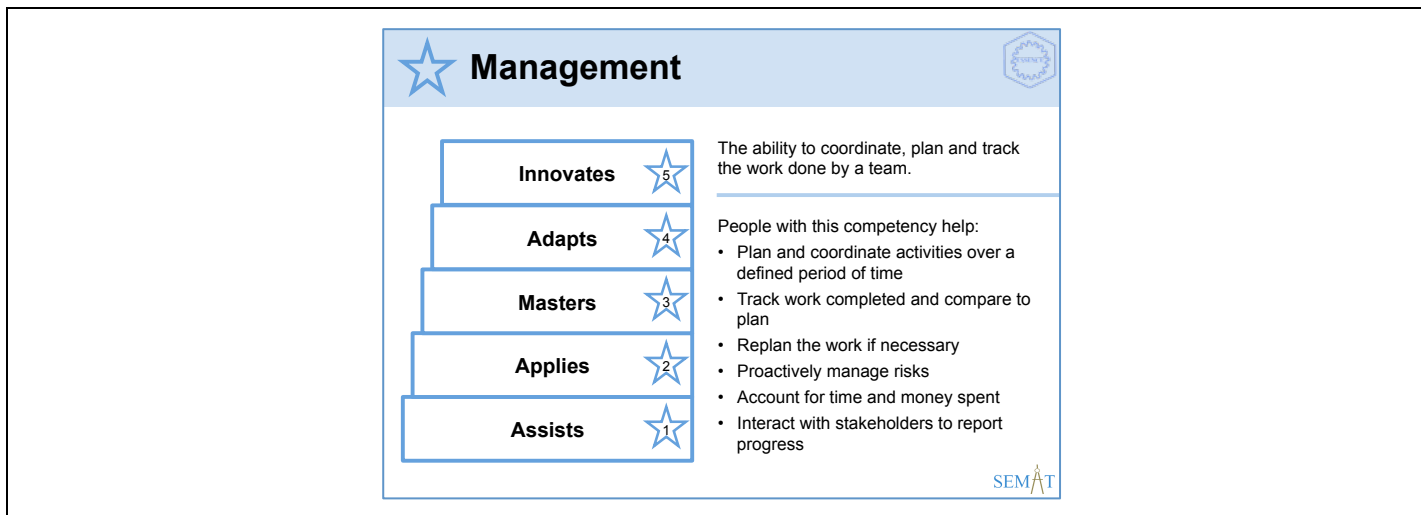


Figure 16. *Management*: Overview Card

This competency encapsulates the ability to coordinate, plan and track the work done by a team.

The management competency is the administrative and organizational ability that enables the right things to be done at the right time to maximize a team's chances of success. It can be provided by the team members themselves, a team leader, a lead developer, a project management office or a professional project manager.

Table 14. *Management* goals and skills

<p>Management helps the team to:</p> <ul style="list-style-type: none"> • Proactively manage risks • Account for time and money spent • Interact with stakeholders to report progress • Coordinate and plan activities 	<p>Essential skills include:</p> <ul style="list-style-type: none"> • Communication • Administration • Organization • Resource planning • Financial reporting
--	--

Competency Levels

Level 1 – Assists	Demonstrates a basic understanding of the concepts and can follow instructions.
Level 2 – Applies	Able to apply the concepts in simple contexts by routinely applying the experience gained so far.
Level 3 – Masters	Able to apply the concepts in most contexts and has the experience to work without supervision.
Level 4 – Adapts	Able to apply judgment on when and how to apply the concepts to more complex contexts. Can help others in applying the concepts.
Level 5 – Innovates	A recognized expert able to extend the concepts to new contexts and inspire others.

See Table 8 for the competency level checklist

4 An Overview of the Activity Spaces

The kernel also provides a set of activity spaces that complement the Alphas to provide an activity-based view of software engineering. The kernel activity spaces are shown in Figure 2.

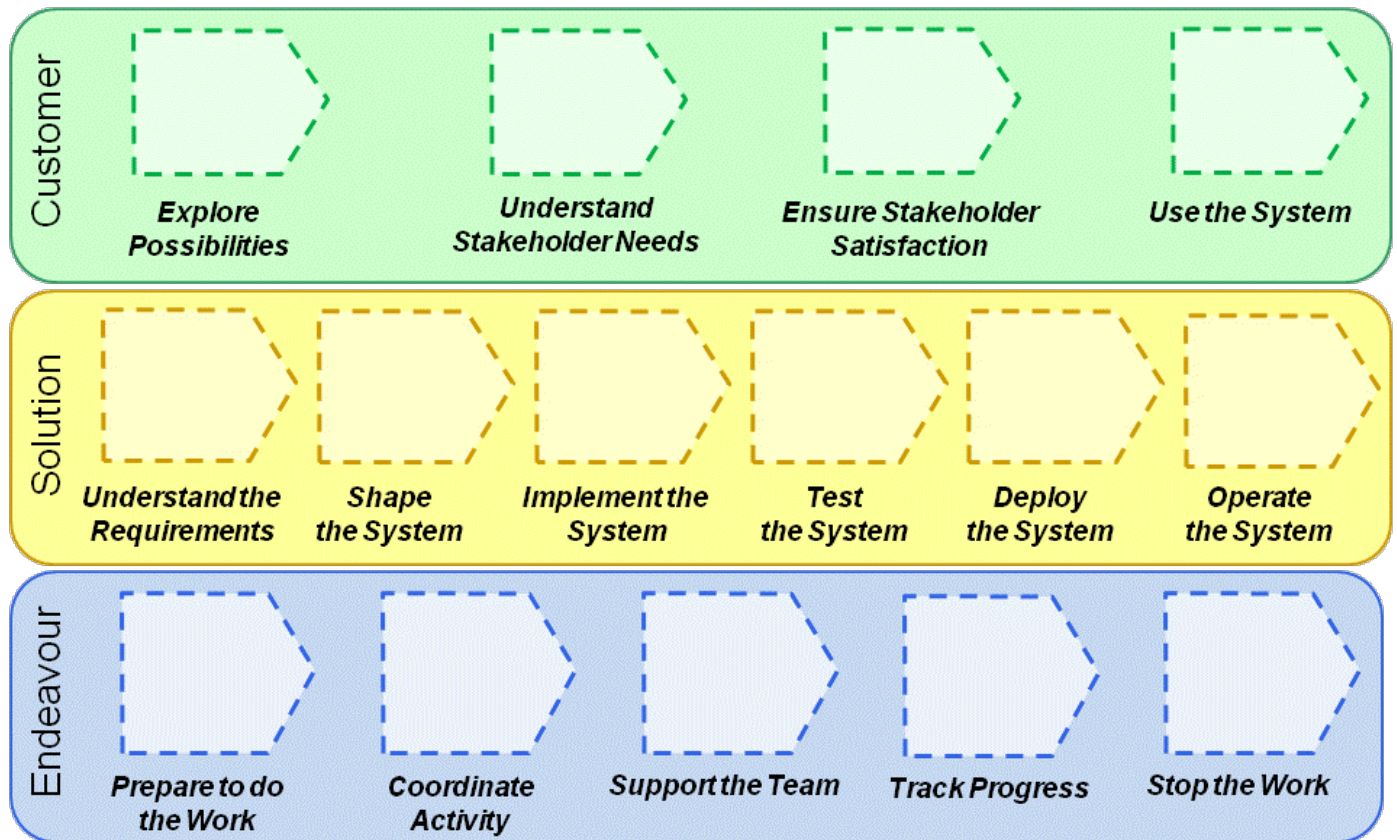


Figure 2. The Kernel Activity Spaces

In the **customer** area of concern the team has to understand the opportunity, and involve the stakeholders:

- **Explore Possibilities:** Explore the possibilities presented by the creation of a new or improved software system. This includes the analysis of the opportunity to be addressed and the identification of the stakeholders.
- **Understand Stakeholder Needs:** 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.
- **Ensure Stakeholder Satisfaction:** Share the results of the development work with the stakeholders to gain their acceptance of the system produced and verify that the opportunity has been successfully addressed.
- **Use the System:** Observe the use of the system in a live environment and how it benefits the stakeholders.

In the **solution** area of concern the team has to develop an appropriate solution to exploit the opportunity and satisfy the stakeholders:

- **Understand the Requirements:** Establish a shared understanding of what the system to be produced must do.
- **Shape the system:** Shape the system so that it is easy to develop, change and maintain, and can cope with current and expected future demands. This includes the overall design and architecting of the system to be produced.
- **Implement the System:** Build a system by implementing, testing and integrating one or more system elements. This includes bug fixing and unit testing
- **Test the System:** Verify that the system produced meets the stakeholders' requirements.
- **Deploy the System:** Take the tested system and make it available for use outside the development team.
- **Operate the System:** Support the use of the software system in the live environment.

In the **endeavor** area of concern the team has to be formed and progress the work in-line with the agreed (who agrees is dependent on team's constraints and governance rules) way-of-working:

- **Prepare to do the Work:** Set up the team and its working environment. Understand and commit to completing the work.
- **Coordinate Activity:** Co-ordinate and direct the team's work. This includes all on-going planning and re-planning of the work, and re-shaping of the team.
- **Support the Team:** Help the team members to help themselves, collaborate and improve their way of working.
- **Track Progress:** Measure and assess the progress made by the team.
- **Stop the Work:** Shut-down the software engineering endeavor and handover of the team's responsibilities.

5 Acknowledgements

Produced by SEMAT Inc

Authors and Contributors

Authors: Ian Spence, Cecile Peraire, Maria Augusta Vieira Nelson, Bob Palank, Mira Mirosława Kajko-Mattson, Winifred Menezes, Barry Myburgh, Andrey Bayda, Paul E. McMahon

Editors: Paul E. McMahon and Mira Kajko-Mattson

For more information or to join the SEMAT Community go to www.semat.org