A “Thinking Framework” to Power Software Development Team Performance

Paul E. McMahon, PEM Systems

Abstract. Essence is a new Object Management Group (OMG) software standard [1, 2, 3] developed specifically for software development practitioners and teams. This article explains specific features of Essence that could help software teams improve performance in ways previous frameworks, including the CMMI®, Lean Six Sigma, and Scrum, have fallen short. The article also provides insight into why many performance improvement efforts fail, and how Essence— or a framework with characteristics similar to Essence— could provide the help organizations need to hit performance targets more consistently.

Essence: What it Is

Essence is a “thinking framework.” Just saying this should start you thinking about performance differently from the way many view it when using frameworks such as the CMMI, Lean Six Sigma, and Scrum. This is not meant to imply everything we have been doing in the past to improve performance is wrong. The vast majority of improvement approaches have strengths, and have helped organizations get better. However, there exists significant evidence indicating most improvement projects fail to achieve their goal [4, 5, 6].

Why are we facing this problem?

It is my contention, based on my forty years of working in the software industry, that the problem is not with the frameworks, tools, or practices organizations are using, but rather with the way organizations are going about deploying practice guidance and practice improvements. This is where a framework such as Essence can help.

Essence: What it Is Not

Essence is not a new method, or a new set of practices. It is not in competition with any of the popular frameworks or methods including CMMI, Lean Six Sigma, Scrum, or Kanban. It is a framework that can help organizations achieve performance goals by helping practitioners and teams implement whatever approach they are already using more effectively and efficiently. Today Essence is being tested in University field studies [7, 8] to determine the degree to which it can help teams work more effectively than using other popular approaches alone.

What Do We Mean By A “Thinking Framework” and How Can Essence Power Other Software Development Approaches?

By thinking framework we mean a framework that can help software practitioners and teams think through the tough problems they face by helping them ask the right questions, and find the optimum solutions based on their specific situation. The reason this framework can work with whatever your team is currently doing is because it contains no practices that might conflict with your current approach, and it brings higher visibility to how well a team is implementing essentials that have been proven to exist on all software development endeavors. More importantly, when projects fail to meet performance goals the root cause can usually be traced to poor implementation of one or more of the essentials.

Why Software Development Endeavors Get in Trouble

Anyone who has been involved in the business of software development for any length of time can tell you the primary reason most software development endeavors get in trouble is because they fail to adequately address things most of us know are essential, and even when they recognize this failure they then fail to take timely corrective action.

Why—When We Understand the Problem—Can’t We Effectively Implement the Solution?

We have known for some time how to solve this problem. We know that empowering development teams is a critical part of the solution as the team is best equipped to observe issues early, and take timely action. But exactly how an organization should go about empowering its development teams is hotly debated today—largely due to fear of lost control.

Why Do We Need Another Framework to Help Empower Teams?

The CMMI [9] and Lean Six Sigma [10] are useful improvement frameworks, but they are intended to be applied by trained process professionals, not software practitioners and development teams. Therefore they do little to help empower development teams to solve the common challenges faced each day on the job.

A popular framework commonly used today by practitioners and development teams is Scrum [11]. Scrum provides an effective framework to encourage teams to raise issues, solve issues, and keep their progress visible in a timely fashion, but Scrum provides little help to teams with respect to where they should look for issues, and how to accurately assess where the team really is with respect to addressing issues and achieving its goal. Furthermore, while Scrum, and Agile methods in general, have been extremely popular, there is considerable literature available that indicates many organizations are continuing to fail when using Agile approaches alone.

Scott Ambler stated, based on a Dr. Dobbs State of the IT Union survey conducted in November, 2009, that only 11% of respondents indicated that their existing governance strategy works well with agile teams. Ambler said that this is an approach they are already using more effectively and efficiently.
indication that their organization is likely to apply traditional governance strategies (e.g. command and control) and that this strategy will not work with agile teams [12].

Another reason cited for these failures [13] is: “trying to force a strict agile approach when it is not appropriate for every environment”

These facts leads us to ask two questions:

“How can we help our software teams determine the right level of agility for their specific project?” and, “How can we help our senior management move to the right governance strategy to best support their software teams and their organizational objectives?”

How a Framework, Such as Essence, Can Help

Essence is not an alternative to Scrum, or any other method, set of practices, or improvement framework. It is not another methodology. It was intentionally developed to be agnostic to specific practices and methods. It is a framework intended to be used with whatever practices, method and lifecycle an organization chooses to use and it can help teams ask the right questions and take the right actions leading to the right balance of agility and control given their specific situation.

This framework was developed by volunteers representing a wide range of software experiences and cultures including people from industry, academia and research in countries around the world [3]. The framework is not just a theory, but is based on what has proven to be essential to effective and efficient software engineering. Most important to the issues being raised in this paper, it has the potential to help senior management and software development teams work together to implement an appropriate agile governance strategy as recommended by Ambler.

To give you a real example demonstrating how the Essence framework can help with the challenge we face, at Carnegie-Mellon West in a recent field study [8] where students were asked to try Essence, the following was reported:

“While most styles of Agile retrospectives tend to focus on known issues, Essence reflections tend to make unknown issues apparent by covering the project holistically and reminding participants of critical areas that might be overlooked. These differences make Essence reflections and Agile retrospectives complementary. This is illustrated by the following student quote: “Though the team was holding retrospectives every week already, having Essence discussions be a part of it allowed the team to touch on important aspects of the project; aspects which would otherwise be ignored.”

This finding about Essence, based on student feedback from a field study, demonstrates its potential to improve a team’s understanding of unknown issues, or risks. Identifying risks is a critical first step necessary for teams to take action early to eliminate risks. This is a similar observation that was made by a senior experienced engineer in a major US DoD organization when first exposed to the Essence framework [6].

According to the Carnegie-Mellon Software Engineering Institute 60-80% of the cost of software development is in rework [14]. Rework is often caused by issues that were unknown to the team when the work was originally done. Rework is preventable by reducing unknowns early. Therefore anything we can do to reduce or eliminate risks can potentially reduce the cost of software development by 60-80%.

Key Elements Inside the Essence Framework

Key elements inside the Essence framework that I want to focus on in this article are Alphas, Alpha States, and Alpha State Checklists. Alphas are the essential things teams work with. There are seven Alphas inside the framework including Opportunity, Stakeholders, Requirements, Software System, Work, Way of Working and Team. Refer to Figure 1.

Figure 1 Seven Alphas inside the Essence framework

What Alphas Are, and Are Not

Alphas are ultimately about helping your team more accurately assess where they are and where they need to focus their effort next for project success. Some have struggled with the Alpha notion. Alphas are not abstract work products. Alphas always exist regardless of the degree of concrete work products supporting their existence.

The reason this is important is because when we focus on concrete work products alone we can miss essentials for software endeavor success. Over-focus on work products and missing the real goal is one of the problems commonly observed in the way past improvement models have been applied causing organizations to fall short of their performance goals [6].

How Essence States Can Help Accurate Progress Assessment Regardless of Lifecycle and Practice Choices

The Essence Alphas have states and checklists that can help teams assess where they currently are and where they need to focus their effort next. This includes projects using incremental, agile or waterfall lifecycles. As an example let us look at the Work Alpha.

Two of the states within the Work Alpha are Work Prepared and Work Started. The Alpha States and Checklists can be represented on cards. An example of these two Alpha States with a subset of checklists is shown in Figure 2.
The Checklist shown for the Work Prepared State is:

- The work is broken down sufficiently for productive work to start.

The three checklists shown for the Work Started State are:

- Development work has been started.
- Work progress is monitored.
- The work is being broken down into actionable work items with clear definitions of done.

One of the reasons why projects often fail to maintain their schedule commitments regardless of lifecycle or practice choices is due to inadequate work break-down and estimation before the work begins. When the work is not adequately broken down and estimated unrealistic schedule estimates often occur leading to poor schedule performance.

If you look closely at the way these checklists are worded in these two states you can see how a team can effectively use the checklists to assess the work break down status on any project including an agile project, an incremental project or a waterfall project.

For example, if your team is using Scrum and the planning poker practice [15] they could conclude that “the work is broken down sufficiently for productive work to start” as long as backlog items [11] have been selected and the team has reached consensus on the estimates for each backlog item for the first Sprint [11] using their planning poker estimation technique.

As the project proceeds, as long as they are holding their daily standups [11] to monitor work progress, and at the start of subsequent Sprints, backlog items are selected, broken down and estimated by the team, the team could conclude they are meeting all three checklist items under work Started.

On the other hand, a team that is using a waterfall lifecycle might conclude they have not passed the work Prepared state if only part of the work has been broken down because they might deem that insufficient given their lifecycle choice. This example demonstrates how a team can use the Essence framework to more accurately assess if the work is sufficiently broken down and estimated regardless of lifecycle choice.

If the work is not adequately broken down the team can raise this issue and make it a priority to solve the problem before it leads to more serious issues later in the project. Today, on many projects, including those that use other popular frameworks and methods it is too easy to let this type of “small issue” slip by, and lead to a “big issue” downstream.

Experienced practitioners have known for a long time that small issues not handled early are often the cause of big issues downstream, but knowing this has not helped. One possible reason is because we have not given our practitioners and teams a framework that gives them solid evidence that actions need to be taken to resolve issues at an appropriate time.

**How Essence Checklists Are Different and Can Help Teams Assess Progress More Accurately and Take Action at an Appropriate Time**

Let us now look at an example demonstrating how Essence checklists are different from traditional checklists and how they can help team performance.

Traditional checklists are what I refer to as “existence checks” [6]. Examples include:

- Do you have a plan?
- Do you have a design document?
- Did you conduct a peer review?

Existence checklists are easy to use because the answer is a simple yes or no that requires little discussion. But existence checklists can lead to a checklist mentality, and they do not help with questions related to how well the team is performing or how much of a certain activity the team should be doing. Existence checklists are what many quality organizations use today and they are common among organizations that use the CMMI model. One reason for this is because existence checklists are easy for external appraisers to use, or external quality audit personnel who are not intimately familiar with the project they are auditing.

But existence checks do little to help teams assess where they are in terms of how much effort is still required to get the job done on a specific project. Following is an example demonstrating how Essence checklists go beyond existence checks helping teams improve performance by improving their progress assessment and improving their decisions on where they need to focus their attention next.

**Example: Checklist item Requirements Alpha Coherent State**

A checklist item for the Requirements Alpha Coherent state is:

- Conflicts are identified and attended to

Just verifying that a requirements document exists would not be sufficient to verify that this checklist item is met. This checklist item is asking us whether or not we have conflicting requirements and if we do are they being addressed? Often the real pain that teams face originates from conflicting requirements that they do not know how to handle, and it is these types of problem areas that often lead to latent defects and extended integration schedules.

Because Essence contains states and checklists as a stable reference it provides a degree of assurance that progress is assessed more objectively and accurately— and appropriate control is maintained. Furthermore, because the Essence states and checklists are agnostic to a team’s lifecycle and practice choices it can help power whatever approach your team chooses to use.
Not Checklists an External Auditor Can Easily Apply

Many of the Essence checklists are not checklists an external auditor can easily apply by looking at a project from the outside. You need to be intimately involved in the project to answer honestly the questions that many of the Essence checklists lead you to ask. This is one reason why this framework is for software teams and why practitioners need to be more actively involved when making key decisions that help to steer a successful high performance project.

Traditional checklists lead us to simple yes/no answers that require little discussion, and fail to provide an accurate status of the project. Many of the Essence checklists lead us to deeper questions, and deeper analysis—the kind that gets to the real issues that ultimately affect project performance.

Example Of a Team Using Essence to Assess and Solve a Difficult Situation

The SEMAT2 volunteers have been working on an Essence User Guide that will be made available from the SEMAT web site [16] to help guide teams with options they have to apply the framework. One of the scenarios from the User Guide2 provides an example of how a team could use the Essence framework to assess a difficult situation.

In this scenario the team realizes they have a problem with a resistant stakeholder. The scenario demonstrates how a team can use the alpha states and their checklists to drill down and solve a specific problem.

Since the team knows they are having trouble with a stakeholder they first assess where they are with respect to the Stakeholder Alpha. Their discussion leads to team agreement that they have achieved the first state, stakeholders Recognized, but they have not achieved the second state, stakeholders Represented. They agree their next step is to get a stakeholder representative appointed.

The next step is to get the stakeholder Involved and they do this by interviewing him trying to find out what is behind the resistance. They learn through the interview that he does not see the value of the new system, which leads to the Opportunity Alpha and the Value Established state. This in turn leads to the Software System Demonstrable state once the team recognizes they need to conduct a demonstration to help the stakeholder see the value. Refer to Figure 3.

What you learn through this scenario is how the discussion leads through a sequence of alphas helping the team figure out the next action to solve the problem. This scenario demonstrates how a team can conduct their own root cause analysis and figure out the right actions to solve a problem in a timely way.

When a group of students at Carnegie-Mellon West that used Essence in an early field study [7] were asked if Essence’s monitoring and steering approach had value to a project team, 90% of the students said that following the approach was worth their time, and 80% said they would use the approach again on their next project.

Lean Six Sigma provides a powerful tool kit to help organizations conduct root cause analysis, and take action to improve. But to use this tool kit often requires the expertise of a Lean Six Sigma Black Belt which requires hundreds of hours of study to reach the required proficiency level. Like the CMMI, Lean Six Sigma is a tool kit for process professionals, whereas the Essence framework is for development teams to help them solve their problems themselves in a timely way.

What’s Different About the Essence Approach?

I stated in the beginning of this article that the reason many organizations are falling short of their performance goals is not because of their choice of practices or tools, but rather because of the way they are going about deploying their practice guidance and practice improvements.

Fundamental to the Essence approach is—rather than making radical changes to whatever you are doing today—to make changes in small steps based on where your development teams need the most help right now.

How often do we hear our practitioners say: “My company processes do not help me with the real problems I face each day on the job.”

It is my contention that the cause of this problem is the fact that too many organizations are focusing their guidance and improvement efforts in the wrong area. We spend too much effort trying to tell our teams what to do in situations they already know how to handle, and too little effort helping them with the common but difficult situations where they need help the most. Instead of just telling our development teams what they should be doing, we should be listening more for where they need help, and then focusing our improvement efforts accordingly. This is an area where the Essence pattern notion could help. [6]
What is an Essence pattern?
A Pattern in the Essence system is defined as, “an arrangement of the other language elements (e.g. alphas, alpha states,...) into meaningful structures.”
You can think of a pattern as a simple mechanism to deploy a small slice of useful information that can help your teams with a specific challenge. Following is a simple example [6].

The Dictated Schedule Pattern
Project Lead speaking to her team: “Management has dictated we will not slip schedule so do whatever it takes to get the job done.”
A developer responds: “I will skip my design review, and much of the testing I planned to do because I do not see any options.”
A second developer replies: “We could focus the design review and testing just on the areas where we have seen problems in the past.”
What would you do, if faced with the same dilemma? Are there other options?

Using Patterns To Help Empower Your Development Teams To Make Better Decisions
When I help companies that want to move decision-making deeper into the organization I encourage them to write their processes in a way that supports practitioner decision-making. One way to do this is by providing criteria and options in their processes. Now let us look at this scenario from the Essence framework perspective.

The Dictated Schedule Pattern From the Essence Framework Perspective
The dictated schedule scenario relevant alphas are Work and Way of Working. Refer to figure 4.

Work is defined as: Activity involving mental or physical effort done in order to achieve a result. And Way of working is defined as: The tailored set of practices and tools used by a team to guide and support their work.

Now let us assume the team had assessed the project to have achieved the following states (Refer to Figure 5):
• Work Under Control
• Way of Working In Place.

Based on the checklists, these states imply that:
• the work is going well,
• risks are being managed, and
• all members of the team are using the way of working.

Figure 5 Work Under Control State and Way of Working In Place State

The answers to these questions depend on each project’s specific situation, and your team’s agreed way of working. Note, this is an example demonstrating how a team could fall back, and how the Essence framework when applied by a development team as part of their progress assessment tool-set can help a team assess progress more accurately helping them take the timely actions needed to keep their software endeavor on course.

Current Status and Next Steps
The Essence standard is still young having been adopted by the OMG in June, 2014. Some early adopters including MunichRe, the world’s leading Reinsurance Company, and Fujitsu Services which used an early version of the framework have reported encouraging results [2].

There is also effort going on in multiple Universities around the world to integrate the Essence framework into existing software engineering curriculum including at the Universidad Nacional de Colombia where Dr. Carlos Maria Zapata has developed and is currently delivering the first full course based on SEMAT and the Essence framework.
The next critical step for Essence is to gather more case study information and hard data to help us better assess the value this new standard can bring to power software development team performance.

**Disclaimer:**

CMMI® is registered in the U.S. Patent and Trademark Office by Carnegie Mellon University.

**About the Author**

Paul E. McMahon, Principal, PEM Systems has been an independent consultant since 1997. He has published more than 45 articles and multiple books including “15 Fundamentals for Higher Performance in Software Development.” Paul is a Certified Scrum Master and a Certified Lean Six Sigma Black Belt. His insights reflect 24 years of industry experience, and 17 years of consulting/coaching experience. Paul has been a leader in the SEMAT initiative since 2010.

Phone: 607-798-7740
E-mail: pemcmahon1@gmail.com

---

**References**

10. George, Mike, Rowlands, Dave, Kastle, Bill, What is Lean Six Sigma?, McGraw-Hill, 2004
16. SEMAT Web Site, <www.semat.org>

**Notes**

1. The Essence figures in this document are provided courtesy of SEMAT Incorporated.
2. SEMAT stands for Software Engineering Method and Theory. SEMAT is the worldwide group of volunteers that developed the Essence framework. For more information about SEMAT refer to <www.semat.org>.
3. I would like to thank the following SEMAT volunteers for their contributions on this scenario: Dr. Cecile Peraire, Dr. Mira Kajko-Mattsson, Winifred Menezes, Barry Myburgh, and Dr. Robert Palank.