Essence, the Standard Common Ground, in a Nutshell

The software industry has changed our world dramatically. The world runs on software, and software is key to any product and virtually all organizations! Thus of course you may say that software has been fabulously successful. What is much less known is that the way the industry operates and develops and maintains its software is worryingly immature. The mission of Essence is to facilitate change to the way the software world operates.

In this paper/email, I will lay out briefly how Essence seeks to do this and the impact you can expect from using Essence. I will cover:

- What is Essence?
- Who uses Essence?
- Why use Essence?
- The status and adoption of Essence today

I think you will agree that Essence brings a paradigm shift to the world of software and the potential for its adopters to reap huge benefits in productivity and performance.

There is one problem though … the length of the article!

I wish I could have written a shorter but still meaningful paper at this level. If the task had been to present a new method, even one as big as RUP or SAFe, it would have been much easier because they had or have a much more limited role. Essence represents a dramatic change to the industry and it means rethinking what people already know. This may sound scary, but the good news is that all the methods or practices people already have are still valid, Essence just makes them fundamentally better. It gives values that the industry never has seen before, we call them “killer” use cases. Moreover, Essence will have huge impact on everything we do, not just software, not just development, not just agile, but this won’t happen overnight, but in steps each motivated by additional value creation.

When writing I also had to balance the space talking about “what this new thing Essence is” and “why should I care”. I had to talk about the “killer” use cases to a deep enough level to make them understandable without being overly salesy. I am not sure that I have succeeded. My explanations may not be enough to be understandable, maybe they are too abstract. The use cases can easily be downplayed. For instance when I talk about ‘Creating a systematic learning organization’ it is easy to dismiss it and say “we already have one”. However, I hope it works well to create questions and stimulate good discussions.

1.0 What is Essence?

“Essence doesn’t attempt to be an alternative to any existing method. Instead, it is designed to just make them better.”

Essence is many things. It is like the elephant in the parable of the Blind Men and the Elephant.

The parable is the story of a group of blind men who come across an elephant – an animal they had never encountered before. Being blind, they learn about the elephant by touching it, but each man is feeling a different part of the elephant’s body, and each comes up with a description that is completely correct from the evidence they each have – but each conclusion is completely different.

For example, the man who felt its trunk described says is like a giant snake, the one feeling the legs said, ‘this is just a large cow’, whereas the one at the tail described it as nothing more than a piece of rope.

Essence describes the common ground of software development – we call this the kernel. Essence also helps people learn about software and its practices. It is a language to describe practices and methods. It is a thinking framework. It is a set of cards and games. And many more things. It offers hundreds of use cases to its many users. While all these things are true, much like the elephant, when you step back and look at all of them together, there is far greater value than just looking from one perspective.

When we step back, we can see that Essence is a fundamentally different way of thinking when it comes to developing software. In fact, not just software but systems and anything related to engineering.

Essence has been evolving over many years. Within Ivar Jacobson International (IJI), the work that eventually led to Essence started already in 2005. Four years later Ivar Jacobson together with Professor Bertrand Meyer (computer scientist) and Dr Richard Soley (chairman of OMG) created a community called SEMAT (Software Engineering Method and Theory), which attracted hundreds of experts around the world with the mission “to redefine software engineering based on a solid theory, proven principles, and best practices that: include a kernel of widely agreed-upon elements, …”. The work of the community resulted in the international standard Essence adopted by OMG 2014.

Thus, on the one hand today in 2023 Essence is not new. On the other hand, creating a substantial number of use cases of Essence has taken many years because we wanted to get wide adoption by the technical leaders of the world. Thus, we wanted creators of successful methods to use Essence as a platform to describe their own work, which obviously is an extremely difficult task. They need to be convinced of the value Essence brings to their work. For many reasons we abandoned what otherwise is normal, namely, to just take other creator’s work and without their involvement make your own interpretation of their work.

However, today, Scrum, Scrum at Scale, The Kanban Method, User Stories, The Spotify Model, Innovation framework and our own Use Cases have been essentialized with the support of the creators. About a dozen other well-known methods or practices are in the process of being essentialized. Essentializing means to use Essence and its language to describe the practice. This opens up many opportunities to make better use of the practice, as we shall see later.

You can compare Essence with an operating system. In the same way as an operating system without applications built on top of it is not very useful to its users, Essence is not very useful to its users without practices built on top of it.

Many people see Essence and its use cases as the next big paradigm, for instance Joseph Pelrine, an experienced Psychologist and Agilist, previously CTO of eBay: “I now consider Essence to potentially be the biggest step forwards in the methods space since the Agile Manifesto”.

Best of all, Essence doesn’t attempt to be an alternative to any existing method. Instead, it is designed to just make them better.

In fact:

At its simplest, Essence is a method-agnostic description of software development. It is an international OMG standard developed, reviewed, improved and tested by experts from all over the world. Although developed as a software standard it is far more universal and can be applied to almost any type of solution development.

It has done this by distilling the important elements of software engineering into a universal common ground. Any software endeavor can be described in Essence terms, no matter how it is being developed. It also means that any development method or practice can itself be described in Essence terms, using the Essence Language - a simple, intuitive, visual language.

This universal common ground consists of 7 things, called the 7 alphas, that all software endeavors need to take account of. Each can be in a number of states from initial conception to retirement. Those seven things are: • Opportunity • Stakeholders • Requirements • Software System • Work • Team • Way of Working

To successfully create solutions, all these alphas must be addressed, moving them from state to state as the endeavor progresses. This is important because once we have defined this common ground, we can use it to monitor, understand and improve our endeavors, independent of the practices and methods used by the teams. This allows teams to have autonomy over how they work and for organizations to have consistency, even where some teams are practicing waterfall and others practicing Scrum, SAFe or something else.

We can use Essence to describe any practice or method (not only those related to software) and because they are all connected to the same common ground, we can mix and match them to optimize how we work. By standardizing on Essence, organizations don’t need every team to adopt a common method or approach. All they need to do is relate how they work to Essence and it doesn’t matter that they use different approaches on top of it.

For individual teams, Essence allows them to find the best practices for their particular needs without having to understand or pick a comprehensive method. Because practices are built on top of Essence, it is easy to see where there are gaps and for the team to ensure they are addressing all the 7 critical alphas.

There are other aspects to Essence and many more ways in which it can improve delivery. But its core value is that it describes a universal common ground that is true for all types of software development.

2.0 Who uses Essence?

The primary or target users of Essence are teams, teams of teams, organizations and their leaders.

However, there are many more users that Essence addresses, such as the practitioner, the coach, the method champion, the development lead, and the executive. In the academic world there are another set of users for example the teacher and the researcher. All of them have their own use cases.

As of the end of 2022, our focus has been to support teams, in particular software development teams, and their organizations and leadership. For a large organization these are the most critical users – the number of teams are large, and the organization needs to have a long-term sustainable governance while also providing individual teams with some ability to tailor their way of working to suit their unique needs.

3.0 Why use Essence?

More than 50 years ago, in 1968, Ericsson was the first company to adopt component[2]-based development for large scale product development. To go for components was the most controversial decision on technical direction that I experienced during my 33 years at Ericsson. No one predicted that that decision would result in the AXE system becoming the greatest commercial success story ever in the history of Sweden, and maybe it still is. No one realized that it would dramatically change the whole organization; change how Ericsson was selling, developing, testing, installing, upgrading, etc. its products.

In hindsight Ericsson was at the least a decade ahead of the competition in telecommunication and its AXE system was prepared to take a lead in the next big thing in telecommunication – the mobile communication. No one saw it coming, and when it happened no one knew that it happened thanks to a highly controversial decision taken in June 1968 with 13 participating technical managers at three levels. The leader of the meeting presented Ivar Jacobson’s proposal asking for opinions. The most positive participant said: “I don’t know!”. With no support from the team the leader, a third level manager, decided that components were the future direction for Ericsson.

Essence puts any company in a similar situation today. It is hard to predict the outcome of adopting Essence as a platform for the company’s processes, but we know that if successful the impact will be far greater than just introducing components. It will have deep impact on essentially all processes. If successful, they will be better and faster, and most likely cheaper.

We have some very positive case studies, but more important are all the concrete examples of how organizations are using Essence and its use cases today. These will be reported in a separate article soon. However, we will do things differently with the intention to significantly improve status quo, and the reasoning behind these changes will make it highly believable that dramatic improvements may occur. The good news is that many of the things we will do differently can be experimented in small pilot projects before scaling up to whole organizations.

Let us be more concrete and give examples of what can be expected if adopting Essence. We will focus on the most critical users and their most valuable use cases. These use cases have never been offered before when using traditional ways of working with methods. They are by far not the only ones, but they illustrate what can be achieved.

3.1 “Killer” use cases for organizations

Essence allows us to build eco-systems of practices on both a global and a local level. IJI has started to build such a global eco-system with currently more than 40 practices published and another 60 in our own repository. These eco-systems are treated as any other product under configuration management and version control. (If adopting Essence Ericsson will most likely have their own eco-system but harvest practices from a global eco-system.) For the first time ever, companies can easily get access to updated versions of existing practices. This is a real problem. As an example, use cases were first widely published in 1992. Most organizations applying use cases adopted this version and were unaware of the many updates that happened afterwards. In 2005 we published use case 2.0 which essentially included user stories, but very few users of the original version have become aware of the later version. In fact, no improvements have become publicly available. This is the same situation over many practices. It is an enormous loss of experience both locally and globally.

Two of the most important use cases for organizations are:

1. Creating a systematic learning organization

The Ericsson eco-system will become an asset of great value to the company. It is easy to keep alive because its focus on what is essential is defined by an international standard. In theory it will be living forever, but we know nothing lives forever. However, with very high probability we will not see similar enormous wastes as we have seen in working with methods over that latest 50 years. Every paradigm shifts in the past, for instance from structured methods in the 1970 to object-oriented methods in the 1990 and from these to agile in the 2000 was regarded by the industry as a shock; a painful but necessary transition that needed to be managed. Had their practices been organized in an evolving eco-system, the changes could have happened more organically and iteratively.

2. Moving from software development primarily being a craft to primarily being an engineering discipline

Contrary to most popular methods or practices, which usually have been designed by one software engineering expert (I was one many years ago), Essence has been designed by an expert team of not just software engineering but also computer science professionals. Therefore, Essence is mathematically defined (using denotational semantics) which allows us to develop advanced tools to make software development more of an engineering discipline.

The potential for new kinds of efficient tools is great, which already is proven through IJI's WorkBench and TeamSpace tools. Opportunities for support through AI is enormous as demonstrated by our Waypointer tool.

3.2 “Killer” use cases for teams

Since software generally are built by teams, providing use cases to teams that dramatically improve their outcome will have big impact. Both anecdotal and Standish Group data report that teams do not generally apply practices well, and often very poorly. Scrum founder, Jeff Sutherland quotes a Standish Group report that states that 58% of Scrum teams fail to deliver on time, on budget and with happy customers. He puts this largely down to poor implementation of Scrum. Thus, improving the effectiveness of team's development practices offers significant potential. Essence provides several use cases with this outcome. Here are some:

1. *Selecting their own method*. Teams have unique problems, preferences, experiences and context. Applying the same method to all teams will almost always result in a poor fit, resulting in an unhappy team and poorer outcome. For example, the best method for a team working with severe real time requirements is quite different from that needed by a team working on big data. With Essence the team selects their practices from the company-specific ecosystem of practices and compose them into a functioning method. That some practices, for instance architecture practices, must be shared between many teams doesn't invalidate this feature. Similarly, Essence can be used to define common, but lightweight governance practices providing consistency and alignment while allowing team flexibility and autonomy.

2. *Bridging the gap between learning and delivery* comes for free with essentialized practices. Based on the practice definition and the states of the alphas a team can be instructed to perform activities, modify alphas or work products, identify proper work items to put in the backlog and deliver. This gap has been a big hurdle which we

have had with methods since their beginning. Methodologists only give support for the learning cycle, not for the delivery cycle. Thus, when a developer has learnt a practice, she or he has usually severe difficulties in applying it. The only weapon has been to recruit consultants or internal experts, but few are good and those that are good have to do routine work which is not efficient.

3. *Activating the user experience of the methods*. by using games, cards, etc. Existing methods have passive user experiences usually based on slides, handbooks, web sites, videos, etc. They were usually designed many years ago when the method was originally created. With Essence the teams get active user experiences based on games, cards, alpha state transitions, etc.
4. *Gamifying software engineering* thanks to that Essence provides a simple language with deep meaning. Gaming is not new in software engineering. Games motivate, stimulate, and engage the teams. However, non-Essence based games rely on intuition only and not on intuition supported by a well-defined language like the Essence language. Consequently, there are just say 10 practical games. Essence based serious gaming allows hundreds of practical games.

4.0 Essence status

The Essence journey started almost 20 years ago as a private initiative to experimentally address severe shortcomings in software engineering with the mission to dramatically improve our discipline. 20 years is a long time for normal projects, but for a groundbreaking and revolutionizing project this is normal. It took about 20 years from Ericsson adopted components and until Microsoft popularized them. More recently both Dave Snowden (Cynefin, SenseMaker) and David Anderson (Kanban and Kanban Maturity Model) speak of similar time frames for their ideas to form, be tested and evolve to the state today where they are really starting to have wider impact.

We have developed the beginning of a new theory in software engineering. We have applied this theory in practice and learnt how to improve it in several iterations. We can give significant values to organizations and teams when building software and now also other engineering disciplines.

The status of Essence can very well be described by quoting Winston Churchill: “This is not the end. This is not even the beginning of the end. It may be the end of the beginning.” We have just scratched on the surface of Essence, and we believe we can already now provide significant value to a company like Ericsson. However, the future potential is enormous. By treating methods as formal as we treat products many hidden values come in the day. Almost everyone in the organization will benefit with new use cases. My expectation is that we will see a significant improvement in quality (better products), faster and at a lower cost. If of interest, we would be happy to in a call present our current backlog of use cases and features. It is large.

5.0 Final Words

The longer we have been on our Essence journey, the more convinced we are that it will dramatically change the software industry and engineering in general. It has always kept me flabbergasted that big companies with thousands of developers have accepted that engineering work such as building telecommunication products is carried out as a craft instead of as of engineering. We will always need craftsmen, but they should apply engineering techniques and not work as if most of what they do requires a handcrafted solution. The reasons are obvious.

To our knowledge there is no other global initiative than Essence which addresses this issue with the highest mission.

[1] For many reasons we abandoned what otherwise is normal, namely, to just take other creator's work and without their involvement make your own interpretation of their work.

[2] At Ericsson components were named blocks or functional blocks.

[3] The decision was taken in June 1968 with 13 participating technical managers at three levels. Lars-Olof Norén was leading the meeting and presented my proposal asking for opinions. The most positive participant said: "I don't know!". □ With no support from the team Norén decided that components were the future direction for Ericsson.

[4] Scrum founder, Jeff Sutherland quotes a Standish Group report that states that 58% of Scrum teams fail to deliver on time, on budget and with happy customers. He puts this largely down to poor implementation of Scrum.