Proceedings of the Federated Conference on Computer Science and Information Systems pp. 1531–1538
DOI: 10.15439/2016F430
ACSIS, Vol. 8. ISSN 2300-5963

Using ESSENCE ALPHAs in a CMMI level 5 software development organization

Miguel Ehécatl Morales-Trujillo
KUALI-KAANS Research Group and Engineering Faculty of the National Autonomous University of Mexico, Mexico City, Mexico
migmor@ciencias.unam.mx

Hanna Oktaba
KUALI-KAANS Research Group and Science Faculty of the National Autonomous University of Mexico, Mexico City, Mexico
hanna.oktaba@ciencias.unam.mx

María Julia Orozco
Ultrasist, Av. Revolución 1181 8th floor, Merced Gómez, Mexico City, Mexico
mjorozcom@ultrasist.com.mx

Abstract — **Managing a software development project is a challenging task; time and effort is required to monitor the project's health and progress. In this context, organizations look for proposals that would assist them in this task. Recently a new and light alternative was introduced: ALPHAs, which are central elements of ESSENCE – Kernel and Language for Software Engineering Methods OMG standard. This paper presents the experience of a Mexican organization that uses ALPHAs to enhance its processes. The paper summarizes the actual use of ALPHAs in the organization, their advantages and disadvantages, and outlines some advice for organizations wishing to adopt ALPHAs. We conclude that ALPHAs are useful for monitoring and controlling software endeavors. Moreover, their harmonization with the organization's current process was a beneficial factor in renewing the CMMI-DEV and CMMI-SVC level 5 appraisals.**

Keywords: ALPHA, ESSENCE, CMMI, quality, software process.

I. Introduction

SOFTWARE development is a challenging task that involves soft and hard skills, such as technical knowledge to create software products of quality and social abilities to make the participants of a project work together in order to achieve a goal.

The characteristics of a software development project, in particular, asset specificity and uncertainty, affect the choice of governance structure of a project [1]. For that reason, it is important to define a governance structure for monitoring and controlling the project according to its particular context.

According to [2], this governance structure can follow a top-down or a bottom-up approach. The top-down governance approach corresponds to process-centered methodologies and bottom-up governance is similar to agile methodologies.

On the one hand, process-centered methodologies are based on process reference models, standards or body of knowledges. Examples of these are CMMI [3], ISO/IEC 12207 [4] or PMBOK [5]. To follow such a plan driven process is essential since it is the backbone of the endeavor and reduces time and cost deviations; yet, to follow it tightly requires a great effort and does not assure a high quality product.

Altogether, it is important to take into account that executing project management activities alone does not mean *managing* a software development project, and communication is vital in order to determine its health and progress.

On the other hand, agile approaches, like SCRUM [6], KANBAN [7] or ESSENCE [8], are an effective strategy for communicating with work teams in a timely and accurate fashion; their primary focus is on the people involved in the project and on delivering a product that fully satisfies the client's needs. It is important to bear in mind that the use of agile methods does not guarantee the appearance of the mentioned benefits in each project, or their contribution to a higher efficacy of the whole organization [9].

Agile or traditional, all these activities aim at figuring out how the project is progressing and allowing team members to make competent decisions. As a whole, no matter what approach is chosen, project management activities consume a large part of the project's time and effort.

In this context, organizations constantly explore alternatives to improve their processes and product quality and incorporate best practices from different sources, no matter agile or traditional. Some early works provide interesting proposals that show a growing interest in recent years on the part of the software engineering community regarding process improvement environments where multiple models are involved. [10].

In particular, this paper describes two models relevant for this study: ESSENCE and CMMI. ESSENCE is an Object Management Group (OMG) standard of a great value. It provides a domain model for organizing different factors that influence the success of a software engineering endeavor [11]. One of the values added by ESSENCE is to surface unknown issues, support the evaluation of the team, generating reflective team discussions through a thinking framework that is holistic, state-based, goal-driven, and method-agnostic [12].

As for CMMI, it is a well-known set of practices divided into three models: for Development (CMMI-DEV) [13], for

Acquisition (CMMI-ACQ) [14] and for Services (CMMI-SVC) [15]. Its last version 1.3 was published in 2010.

This paper presents the experience of a Mexican organization, evaluated CMMI level 5 that introduced ESSENCE into its processes, and uses it as an agile mechanism to evaluate the progress of its projects and its work products quality.

The structure of this paper is as follows: in Section II a general background of the ESSENCE standard, its ALPHAs and the description of the organization are presented. Section III describes how the ALPHAs were used within a CMMI based organizational process. Section IV concentrates the results and lessons learned. Finally, Section V contains conclusions and future work.

II. Background

This section presents an overview of the ESSENCE standard and its ALPHAs. Later, the context of the organization under discussion is detailed.

A. ESSENCE

ESSENCE – Kernel and Language for Software Engineering Methods was published in 2014. Its origins date back to the SEMAT initiative that supported re-foundation of Software Engineering discipline through the identification of its "common ground" [16] as a set of elements essential for software engineering endeavors. This initiative was launched in 2009 and was endorsed by the OMG in 2010.

ESSENCE consists of two parts: the Kernel and the Language. The Kernel contains a small number of "things we always work with" and "things we always do" when developing software systems [16], while the Language is used to describe methods and practices. The Kernel consists of a set of concepts called ALPHAs that provide an object-oriented state-based model of a software engineering endeavor [11].

According to [11], ESSENCE main benefits are the following:

- It provides a comprehensive model for a large-scale process improvement endeavor;
- It is a context aware model, making visible to practitioners both theory and practice;
- It is an evolvable and participatory model, it can be used in any time and by anybody of the work team.

Another value of ESSENCE comes by providing a structure for analyzing and organizing the context and factors of software engineering endeavors from different dimensions [11].

It is worth mentioning that the initial objective of the standard was to refound Software Engineering, placing it a solid theory base and giving professionals the means to define their own practices and methods. However, this study shows that the main usage addresses project management issues.

B. ALPHAs

ALPHAs are the top-level concepts, which refer to the essential elements of software engineering endeavors, relevant to an assessment of progress and health [17]. In fact, ALPHA originated as an acronym for Abstract Level Progress and Health Attribute, and its main purpose is to determine fast and at any time how the project is doing. ALPHAs provide consistent language and measurable objectives with which to assess the current state, or articulate next steps and goals [18].

An ALPHA's components are the following:

- A representative and unique name.
- A set of states through which an ALPHA passes during its lifecycle.
- A checklist for each state used to determine if the state is reached or not.

This structure allows practitioners to evaluate the project based on the states of each ALPHA. The checklist for an ALPHA state contains a number of checkpoints that can be referenced to determine whether and to what degree the project has reached that state [19]. Therefore, it is possible to establish the state of a software engineering endeavor through the ALPHAs states [11].

There are seven ALPHAs divided into three groups, which are called Customer, Solution and Endeavor areas of concern. Figure 1 displays all the ALPHAs within their areas of concern:

- Customer (green)
  1. Opportunity (6 states)
  2. Stakeholder (6)
- Solution (yellow)
  3. Requirements (6)
  4. Software System (6)
- Endeavor (blue)
  5. Work (6)
  6. Team (5)
  7. Way of Working (6)

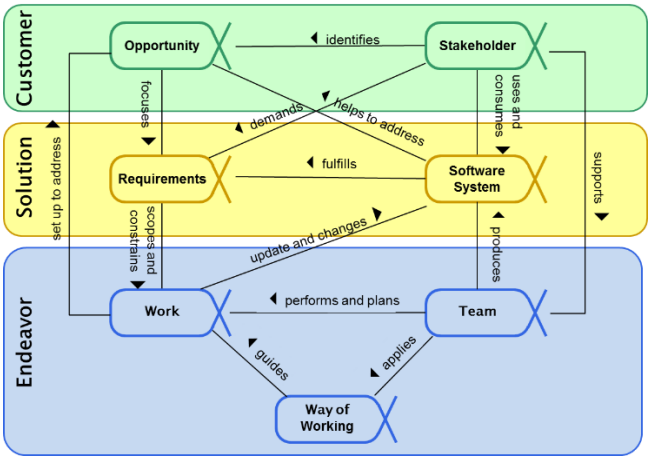

Fig. 1 Areas of concern and their ALPHAs [8]

For an easy and practical use, ALPHAs were represented as cards. Each card corresponds to one state of an ALPHA, and the color code indicates to which area of concern it belongs. As an example, Figure 2 shows the state *Addressed* of the ALPHA Opportunity.

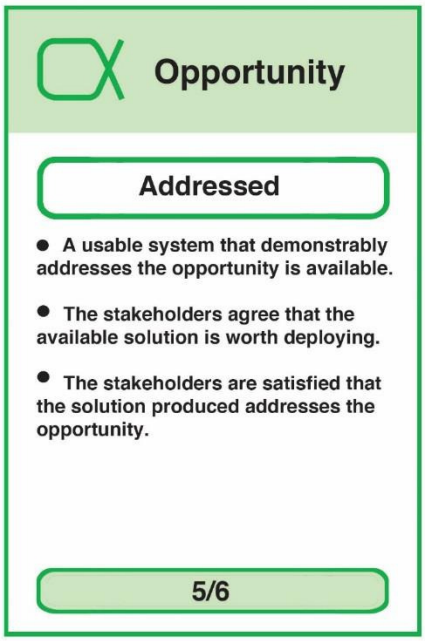

Fig. 2 An ALPHA card, based on [8]

For an opportunity to be *addressed*, the three elements of the checklist should be achieved. Notice that it is the fifth of the six states.

C. Organization

Ultrasist is a Mexican organization founded in 1994. It started as a software development organization, evolving during the last years into a service-oriented enterprise focused on Business Analysis, Enterprise Architecture, Security and Software Quality Assurance. In 2015, they renewed the CMMI-DEV appraisal and obtained CMMI-SVC, both at level 5.

The organization is constantly looking for process optimization and improvement; they carry out a weekly workshop where they discuss the ongoing work and analyze new proposals from the IT community. During such a workshop session, Ultrasist got to know ESSENCE and its ALPHAs through advice from a coauthor of this paper.

In October 2014, they started using ALPHAs for the sake of innovation. At that moment, ESSENCE was close to become an OMG standard, which happened one month later (November 2014).

In January 2015, the ALPHAs were already part of the organization's way of working. The first use of ALPHAs was to verify punctual quality attributes of work products and later on, to develop software quality assurance reviews.

In May 2015, the organization presented their use of ALPHAs as a part of software processes improvement when renewing CMMI level 5 appraisal.

The organization has a hierarchical structure (see Figure 3). The areas colored in green are those directly involved in the use of ALPHAs (the "mid-layer" of the organization):

- Internal SQA
- Sales, Marketing & Clients
- Business Analysis
- Software Construction
- SQA Specialized Services
- Project Management Office

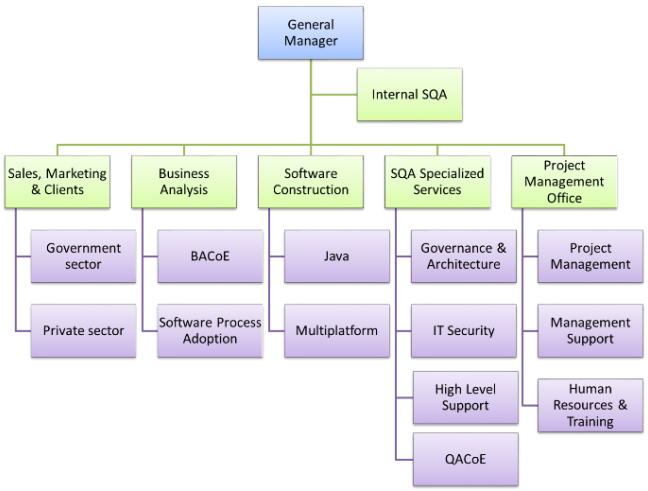

Fig. 3 Ultrasist org chart

One of improvement opportunities recognized by the organization was, for example, the quality of its work products in the Business Analysis area. Particularly, they were interested in a finer verification and validation process of Software Requirements Specification (SRS) work products.

In addition, the Internal SQA wanted to promote a better team integration and communication.

Another and even bigger concern arose after getting the CMMI appraisal, due to which the organizational processes suffered changes. Therefore, the General Manager needed to transmit the new processes to employees, especially new ones, and to generate a pocket guide for technical leaders and project managers.

The first step towards achieving this goal was to carry out a gap analysis versus their process and the actual way of working; next, the organization institutionalized the ALPHAs integrating them into its processes and in the working routine of the people.

Each ALPHA state was analyzed in order to associate organizational processes and the checklists items. When the association was established, the role in charge of the process became responsible for following the ALPHA states. Figure 4 shows the mapping between the organization's processes and the Team ALPHA.

TEAM	State	Responsible	Processes involved
	Seeded	SQA	• APS/Mgm
	Formed	SQA	• APS/Mgm • HR/Training • SQA
	Collaborating	SQA	• APS/Mgm
	Performing	SQA	• APS/Mgm
	Adjourned	SQA	• APS/Mgm

Fig. 4 Mapping between Team ALPHA and organization's process

The next section describes the actual usage of ALPHAs within the organization.

D. Methodology

The data were collected through direct interviews with the people involved in the initiative. Seven persons were interviewed individually in a face-to-face dynamic. Six interviewees lead their respective internal areas, while the seventh is the general manager of the organization.

The interviews were conducted in the following manner: (i) A set of questions on the topic of interest were designed; (ii) The questionnaire was sent to the interviewees; (iii) The interviews took place individually in the organization's facilities; (iv) Each interview was recorded and lasted in average for 20 minutes.

After the data were collected, the recorded interviews were analyzed and the fragments that were relevant for the purposes of the paper were transcribed. Later, the transcriptions were synthesized to obtain the observations listed in the discussion. This work was done by the first and second authors of this paper; this is to enrich the analysis and to moderate the threats to validity in this research.

In addition, the work products arising from the integration of ESSENCE and CMMI were analyzed in order to describe how both proposals, agile and traditional, coexist.

III. ALPHAS USAGE

First, the ALPHAs were translated from English into Spanish; some terms were modified according to the organization's customs and habits. Then, technical leaders got the ALPHA cards printed. They had to define which ALPHA state corresponds to which process or area.

The next step was to start using ALPHAs in a pilot group exclusively for verification and validation activities. Later, the rest of team members started to use the ALPHAs as a means of self-evaluation. Currently, the ALPHAs are being used in other activities and by other roles within the organization; however, this paper is based on the data obtained from the mid-layer roles.

The following paragraphs describe the actual uses of ALPHAs by different areas of the organization.

A. Internal SQA

Internal SQA processes are involved in almost all activities developed in the organization, so the SQA leader makes the most of the ALPHAs. He especially exploits the Requirements ALPHA that helps to trace requirements and verify their level of specification. During his activities, the SQA leader uses ALPHAs for:

- Checking the sufficiency of the work to be done.
- Determining if a work product is finished.
- Integrating agile practices into processes.
- Checkpoints during any time of a project.
- Tracing the requirements.

The Work ALPHA was helpful to determine if the people worked according to the organizational process, and they knew the organizational processes well.

The frequency of ALPHAs use varies depending on the process to monitor. If a SQA process is developed completely within the organization, the ALPHAs are used only during review phases, which happen at least every two weeks. However, if a SQA service is provided to clients, the ALPHAs are used almost daily.

B. Sales, marketing and clients

Sales area uses the Opportunity ALPHA for identifying clients' needs and as a support in creating requests for proposals.

Here, the ALPHAs resulted useful for reporting at what exact point the working team is. The mid-layer has found an adequate way of delivering valuable information of the project to higher levels of the organization who do not need to know all the details, but still need to know the exact progress of the project. Besides, they are especially useful when reporting progress to clients.

Moreover, using ALPHAs with clients helped to get a better understanding of the consequences of not providing a certain feature; the organization found a polite way of showing its clients cumulative effects of product absence as well as presenting missing quality attributes in terms of consequences and negative effects. So, it was possible not only to establish the existence/inexistence of a work product, but also what possible consequences it brought.

The ALPHAs became a favorable alternative for representing a state and a light way to report progress. For example, Figure 5 shows a radial graph that reports in a simple and accurate manner the state of each ALPHA in a particular moment. The ALPHA Opportunity is at state 4

(meaning that the opportunity is *Viable*), Stakeholder at state 3 (*Involved*), Requirements at state 3 (*Coherent*), Software system at state 1 (*Architecture selected*), Team at state 4 (*Performing*), Work at state 3 (*Started*) and, finally, Way of working at state 4 (*In place*).

Fig. 5 ALPHAs radial graph

The graph makes visible a general state of the project and allows work teams to decide which next state of an ALPHA to pursue.

C. Software construction

The JAVA leader guides Daily Scrums and evaluates the team progress using ALPHAs as checklists. The Software System ALPHA, in particular the *Architecture established* state, was took advantage of. He and his team also used other ALPHAs: Stakeholder, Requirements and Team.

D. Business analysis

The SRS Leader used the Stakeholder and Requirements ALPHAs, which were introduced gradually in their daily routine. First, the ALPHAs names were introduced to the work team. Later, when the ALPHAs became more familiar, the states of some ALPHAs were introduced. Thus, the complexity of adopting a new terminology was avoided. Now the SRS leader uses the states names as part of the everyday language when communicating with the team.

On the other hand, the ALPHAs were used to create a SQA Requirements guide (checkpoints for software requirements specification), which aims at improving the work products quality by making a more precise specification of requirements.

The ALPHAs states were grouped into levels of granularity corresponding to different target groups: the ALPHAs are related to the top management level, their states – to the project management, and checklists of work products – to developers. This partition helped the work team to understand ALPHAs and simplified their adoption.

An interesting fact is that the SRS leader actually uses the printed cards and prefers them in the original English version.

E. SQA specialized services

The IT Security technical leader uses the Team ALPHA as a checklist when he forms a team. He also uses the Stakeholder and Requirements ALPHAs.

Similar to the other areas, here the ALPHAs assist in controlling and improving the product quality and the work team's adherence to the process. He reported that he resorts to the ALPHAs, mostly at the moments of hesitation.

Besides that, the Governance Director uses the Opportunity ALPHA to discover client's needs and to understand the project's objective.

Another use of ALPHAs is to identify risks and to classify defects of a process or product. This, in turn, helps to evaluate the problem and to find a solution.

The Team and Software System ALPHAs contributed to know whether the people are working as a team and the communication is healthy; both are crucial factors for a successful project development.

F. How ALPHAs transformed some work products

The organization developed a series of new artifacts influenced by the ALPHAs. The first the SQA checklist that supports the SQA service provided by the organization to their clients. The SQA service usually consists in a reviewing and monitoring a particular business process of the client.

Initially this service used a checklist that consisted of quality attributes identified by the client, the organization, and more importantly, those defined in CMMI-ACQ. However, upon incorporating the ALPHAs internally, the organization observed that the ALPHAs states and checklists made the process lighter and more productive.

Derived from the observed benefits, Ultrasist decided to incorporate the ALPHAs checklist and states into the original document. In the first place, the quality attributes were classified under the scope of each ALPHA and were defined as "quality rules" in the language of the organization. These quality rules are based on the specific goals (SG) and specific practices (SP) of CMMI-ACQ process areas (PA).

For example, the Acquisition Requirements Development PA has SG3 Analyze and Validate Requirements – SP 3.2 Analyze requirements to balance stakeholder needs and constraints. It is mapped to the Requirements ALPHA, specifically to the *Bounded* state, see Figure 6, first row.

Using this rationale, when the team needs to verify the SP 3.2, it runs the checklist of the particular state, instead of reviewing all the subpractices established by CMMI-ACQ or consulting a work product. Importantly, the use of ALPHAs does not replace generation of the CMMI-ACQ related work products, but facilitates assessing the progress and health of a particular concern. Finally, this approach to ALPHAs use is independent from the reference model.

SQA service	CMMI-ACQ			Quality rule	Questions	Category (ALPHA)	Sub-category (ALPHA states)
Functional requirements	ARD	SG3	SP 3.2	The definition of the functionality and required quality attributes is established	• What is the definition of the architecture functionality and quality attributes?	3. Requirements	3.2. Bounded
Techincal Docs	TS	SG3	SP 3.2	The documentation to install, operate and maintain the system is developed	• What kind of supporting documentation is developed? • Are any standards used to generate the supporting docs? • How do you guarantee that the installation, operation and maintenance specifications will work out according to the plan? • What is the exact moment of generating those docs? • How are the generated docs checked?	4. Software system	4.2. Demonstrable
Project Management	RSKM	SG3	SP 3.2	Implement risk mitigation plans	• How often are the risks monitored? • In what way are the mitigation plans implemented?	6. Work	6.4. Under control

Fig. 6 SQA Checklist fragment

IV. Results

This section collects advantages and disadvantages of using ALPHAs. In addition, some advice from the participants involved in this experience, is listed.

A. Advantages and disadvantages

During the interviews, the practitioners expressed the following advantages of the use of ALPHAs:

- ALPHAs are easy to understand and apply, because *"they represent general concepts and put them in black and white"*.
- ALPHAs provide a common language, *"anybody in the work team can understand what you are talking about"*.
- ALPHAs are compatible with any process or lifecycle.
- ALPHAs colors help to associate them quickly to areas of concern, and the state-machine style makes the states flow clear.
- ALPHAs are useful for maintaining discussions focused and collecting clear arguments to make decisions, "*you can ask how the Opportunity is going and get clear answers*".
- ALPHAs help to accelerate convergence during meetings and discussions.
- ALPHAs work as communication facilitators between members of the organization and with the client.

Several disadvantages were also pointed out:

- ALPHAs have too many states; sometimes the cards sets are not handy.
- Some terminology needs to be adapted to the particular context of the organization, for example the *in-place* state or the *Endeavor* area of concern.
- Some states and checklists may have a different interpretation between team members, mainly between juniors and seniors. *"Experience in software development is needed in order to uniform interpretations"*.
- Compared to the mid-layer, the operational layer of the organization required more time to understand ALPHAs.
- ALPHA is not a *universal* term neither a common software engineering word.
- ALPHAs resulted of little help in bigger projects; their simplicity became an obstacle. For example, how to manage many stakeholders or how to evaluate big quantities of requirements with ALPHAs?

B. Threats to validity

The following threats to validity to this particular research were detected.

The internal validity:

- The trustworthiness of the survey responses can be considered a major issue since one of the authors is the general manager of the organization. It is possible that some negative issues were not completely expressed by the interviewees, which may explain the fact the very few disadvantages of the ALPHAs use were detected.
- The coverage of roles in the organization may be considered a threat as well. Only members of the mid-layer of the organization were interviewed, which could affect the perspective of the benefits and drawbacks of the ALPHAs as compared to the point of view of the rest of the organization.
- On the other hand, the number of interviewees that constitutes the sample is representative of the target group who used ALPHAs.

The external validity:

- The organization was not intentionally selected; they took the initiative to start to use ALPHAs and, then, to share their experience.

As for reliability, it should be mentioned that the second and the third authors of the paper has been involved in the organization for a long time (years) and are fully capable to understand the needs of the organization.

Also, the first author, who conducted the interviews, had worked in the organization and developed a trusting relationship with the participants. This fact minimizes the surveys' trustworthiness threat.

After considering the above mentioned threats, we conclude that the results of this research were not affected; however, the data collection methodology can be improved.

Finally, in order to discuss the results and findings from the interviews, peer debriefing took place, which is an extra point to the validity of the research.

C. Some advice

This experience left some useful lessons learned that are summarized in the following paragraphs.

Avoid implementing all the ALPHAs at the same time. Use the ALPHA you need and then add them one by one depending on what you need in a particular moment. Besides, let the team decide which ALPHAs they want or need to use. *"Take the best (that fits for you) of each world and apply it in your context"*. As [20] establishes, an organization has to understand that the organization itself cannot be agile, but its employees can be.

Actually, in the organization's case, the Way-of-Working ALPHA was used occasionally or almost never because they have a well-defined and mature process; in the words of the general manager: *"Everybody knows what to do and how to do it"*.

Do not be afraid to modify ALPHAs, you won't break it down. Some ALPHAs were adapted to become more familiar for the work teams, for example, the Way-of-Working ALPHA was renamed as Working-Methodology. In fact, the ALPHAs should be adapted to the organization's language in order to fit in its process, however minimal these adaptations are.

Do not see ALPHAs as a sequential set of steps, they are not a process. In many cases they were confused with a sequential method. The ALPHAs were created as a tool to be applied at any moment during a project.

It is not necessary to read the whole standard to understand ALPHAs. Actually, only one practitioner read the whole OMG document and the rest consulted section 8.1 Overview to be able to get ALPHAs going. To be more specific, apart from section 8.1 of the standard, in which ALPHAs are presented, it is advisable to read sections 8.2 through 8.4 that provide detailed descriptions of each area of concern, their related ALPHAs, states and checklists.

Try ALPHAs in the presentation you feel comfortable with. Using the printed cards or the electronic version turned out to be a discussion topic. Some participants consider having to carry all the cards a disadvantage; others believe the opposite and prefer the printed version because *"cards are like a cheat sheet to keep quality monitored"*.

V. Conclusions

ALPHAs brought benefits and improvement even for a mature and solid organization that already had its ways of working at a high level. They represented an innovation factor in order to renew the level 5 in CMMI-DEV and CMMI-SVC. Importantly, the quality of the process and the product was systematically improved with the ALPHAs states guide. ALPHAs are not technical-oriented; they are focused on serving as control points and checklists for the teams.

It was shown that ALPHAs can be integrated with other standards, and their independence is a plus when an organization decides to integrate them to the actual way of working and harmonize the whole process. According to [10] harmonizing processes allow organizations to improve, mature, acquire and institutionalize best practices and management systems from multiple approaches.

On the other hand, there are inconsistencies related to the initial objective of ESSENCE: provide practitioners with the means of describing their methods and practices [11]. This issue was addressed by creating activity spaces "descriptions of the challenges a team faces when developing, maintaining, and supporting software systems" [8]. However, in case that the organization does not execute a formal process, the activity spaces make little sense; on the contrary, if an organization possesses a well-defined process, the activity spaces are not necessary, and in the case of Ultrasist, were not used. Instead of supporting the definition of methods and practices, ALPHAs were used to guide the team's way of working. This issue affects small organizations since the simplicity and flexibility of ALPHAs require a well-predefined way-of-working.

Nowadays, the ESSENCE standard is being widely applied in many countries and with diverse objectives. However, the Latin American context features far less impact of this standard in the industry. We believe that the experience described in this paper will motivate Latin American software development industry to work with ALPHAs.

As future work, three lines are identified: (i) to establish patterns, like *usage scenarios*, of when, how and what particular problem ALPHA(s) could solve; and (ii) to determine how the ALPHAs overlap with other standards in order to define a harmonized multi-model process and not implement each one separately. Lastly, (iii) the organization under discussion started to explore how the ALPHAs could be integrated into enterprise architecture services.

Acknowledgment

The authors thank Ultrasist work team leaders: Ernesto Hernández Uribe, Ricardo Rodríguez López, Félix de la O Díaz, Alejandro Ramírez Ramos and Gibrán Granados Paredes.

This work has been developed under the Postdoctoral Fellowships Program of the General Directorate of the

Academic Staff (DGAPA) of the National Autonomous University of Mexico (UNAM).

REFERENCES

[1] B. Erbas and C. Erbas, "On a theory of software engineering: A proposal based on transaction cost economics". In SEMAT Workshop on General Theory of Software Engineering (GTSE'13), San Francisco, CA, USA. pp. 15–18, DOI: 10.1109/GTSE.2013.6613864 (2013)

[2] A. Kocatas and C. Erbas, "Extending Essence Kernel to Enact Practices at the Level of Software Modules". In SEMAT Workshop on General Theory of Software Engineering (GTSE'14), Hyderabad, India. pp. 32–35, DOI: 10.1145/2593752.2593758 (2014)

[3] Capability Maturity Model Integration (CMMI). Software Engineering Institute, Pittsburgh, PA, USA (2010)

[4]ISO/IEC 12207: Systems and software engineering – Software life cycle processes. ISO/IEC, Geneva, Switzerland (2008)

[5] A Guide to the Project Management Body of Knowledge (PMBOK Guides). Project Management Institute, Piscataway, NJ, USA (2013)

[6] K. Schwaber and J. Sutherland, "The scrum guide – the definitive guide to scrum: The rules of the game". http://www.scrumguides.org/ (Accessed 08/05/2016)

[7] S. Shingo, "A study of the Toyota production system: From an Industrial Engineering Viewpoint". Productivity Press (1989)

[8] ESSENCE – Kernel and language for software engineering methods. Object Management Group, Needham, MA, USA (2014)

[9] A. Kaczorowska, "Traditional and Agile Project Management in Public Sector and ICT". In proceedings of the 2015 Federated Conference on Computer Science and Information Systems pp. 1521–1531, DOI: 10.15439/2015F279 (2015)

[10] C. Pardo, F. García, M. Piattini, F. Pino and T. Baldassarre, "A 360-degree process improvement approach based on multiple models". Revista Facultad de Ingeniería, Universidad de Antioquia, No. 77, pp. 95–104, DOI: 10.17533 /udea.redin.n77a12 (2015)

[11] P.-W. Ng, "Theory Based Software Engineering with the SEMAT Kernel: Preliminary Investigation and Experiences". In SEMAT Workshop on General Theory of Software Engineering (GTSE'14), Hyderabad, India. pp. 13–20, DOI: 10.1145/2593752.2593756 (2014)

[12] C. Preraire and T. Sedano, "Essence Reflection Meetings: Field Study". In International Conference on Evaluation and Assessment in Software Engineering (EASE '14), London, England, United Kingdom DOI:10.1145/2601248.2601296 (2014)

[13] CMMI for Development, Version 1.3 (CMMI-DEV). Software Engineering Institute, Pittsburgh, PA, USA (2010)

[14] CMMI for Acquisition, Version 1.3 (CMMI-ACQ). Software Engineering Institute, Pittsburgh, PA, USA (2010)

[15] CMMI for Services, Version 1.3 (CMMI-SVC). Software Engineering Institute, Pittsburgh, PA, USA (2010)

[16] I. Jacobson, P.-W. Ng, P. McMahon, I. Spence and S. Lidman, "The Essence of Software Engineering: The SEMAT Kernel". ACM queue, Vol 10, No. 10, DOI: 10.1145/2380656.2380670 (2012)

[17] I. Jacobson, P.-W. Ng, P. McMahon, I. Spence, and S. Lidman, "The Essence of Software Engineering". Addison Wesley (2013)

[18] J. Park, P. McMahon and B. Myburgh, "Scrum Powered by Essence". ACM SIGSOFT Software Engineering Notes. Vol. 41, No. 1, DOI: 10.1145/2853073.2853088 (2016)

[19] J. Park, "Essence-Based, Goal-Driven Adaptive Software Engineering". In SEMAT Workshop on General Theory of Software Engineering (GTSE'15), Austin, TX, USA. pp. 33–38, DOI: 10.1109/GTSE.2015.12 (2015)

[20] R. Wendler, "Development of the Organizational Agility Maturity Model". In proceedings of the 2014 Federated Conference on Computer Science and Information Systems pp. 1197–1206, DOI: 10.15439/2014F79 (2014).