aemes TI
Asociación Española para la Gobernanza, la Gestión y la Medición de las Tecnologías de la Información

Revista de Procesos y Métricas

31 de diciembre

2013

VOLUMEN 10, NÚMERO 2, AGOSTO-DICIEMBRE 2013
ISSN 1698-2029

De las Tecnologías de la Información

Revista de Procesos y Métricas

De las Tecnologías de la Información

Volumen 10 Número 2

Revista fundada por la Asociación Española para la Gobernanza, la Gestión y la Medición de las Tecnologías de la Información (AEMES) <http://www.aemes.org>

ISSN: 1698-2029. Nº Depósito: M23879-2006

Revista de Procesos y Métricas

De las Tecnologías de la Información

Índice

Volumen 10 Número 2 Agosto-Diciembre 2013

Índice 2
Artículos de Investigación 3
Towards to a new vision of Global Software Development 3
Juegos serios para la Gestión de Ingeniería de Software 17
ISO/IEC 29110: Current overview of the standard 24
Procesos y Métricas en la WWW 41
Relación con RPM 42

RPM 10 (2) (2013) ISSN 1698-2029

Towards to a new vision of Global Software Development

José Ángel Cuadrado Mingo, Roberto Esteban Santiago
Departamento de Informática
Universidad Carlos III de Madrid
Madrid - España
{joseangel.cuadrado, roberto.esteban}@uc3m.es

***Abstract**: We live in a globalized world in which people communicate with other and interact with each. Among these interactions, we can found people that work with other people that live in the other extreme of the world. In Software Engineering, this way of working is called Global Software Development. Many professionals in the sector have talked about this, but there are little works collecting information and work in an unique vision of this technique. For this, we analyze in this paper various definitions of Global Software Development and we offer our own definition of this. Also, we compare the advantages and drawbacks of this technique and propose the collaboration between Global Software Development and other research areas.*

***Keywords**: Global Software Development, definition, advantage, drawback, e-learning, gamification, coaching, agile methodologies.*

1. Introduction

Nowadays, the world is more and more globalized. This is due to the large quantity of ways to communicate between people [1]. While a couple of decades ago, the more common way of communication was postal mail (taking several days to deliver within a country, and several weeks across continents), now, the most common way of communication is the e-mail and other Internet services, by which a person can communicate with another that is thousands of miles away in a few seconds.

As today it is possible to communicate in a few seconds with the opposite extreme of the world, this allows doing other things, such as learning and working in the distance. And, if we talk about work in the distance, we can also speak of global projects, or in the context of software engineering, of Global Software Development.

The origin of the Global Software Development dates back to the decade of 1970. In these years, there was a big demand of software applications, but the number of software engineers was insufficient to cover all the work. For this reason, the organizations dedicated to information systems started to contract external engineers that could help them [2].

Moreover, there were other problems. On the one hand, there were economic problems; on the other hand, the problem of insufficiency of engineers caused delays in the delivery of projects. This resulted in that organizations searched some solution for both economic and temporal problems. The solution found involved associating with other companies and engineers of other countries [3], such that it was possible to hire cheaper engineers and coordinate the work with other teams in different countries and different time zones.

In one of the publications of Erran Carmel [4], he shows what the influential factors for the establishment of Global Software Development were. One of these factors was the fusion between IT companies. These organizations want to complement their product lines and fuse with other companies as a way to enter in

new markets and meet this objective. Due to this, the development teams must collaborate with other teams and work like a unique global development team.

Another factor was the globalization of the company: other global organizations prefer a global provider instead of a negotiation with several small local companies. It is also important that the company has a market approach, such that it can improve its reputation, clients can know the organization, people may invest in it, and governments can provide tax benefits.

An important point for Global Software Development is the possibility of access to the best developers. If an organization wants to develop the best software products, it must hire the best developers, and these don't have to be found in a unique location.

But the two more important factors are the reduction of development costs and time to market. On the one hand, the companies located in countries with elevated costs outsource their tasks to teams located in countries with cheaper costs, like China, Brazil, India or countries of East Europe. On the other hand, the dispersion of the software developers around the world enables software to be developed 24 hours a day and reduce the development cycles (due to the increase the amount of time spent daily software development).

And why is it important to study Global Software Development? Different studies show some reasons for studying it. A research of Kwan-Sik Na [5] specified that 75 percent of the US companies have outsourced their work to other countries.

Another survey of the SIIA (Software and Information Industry Association) [6], involving the participation of114 firms, shows that Global Software Development initiatives are advancing rapidly (62 percent of these companies began their initiatives less than three years ago -on the date on which the study was conducted) and has a positive net impact on both the revenues and profitability of software companies (none of companies who current work with global development teams report a negative impact on the overall revenues, and 87 percent report a positive impact on their profits).

Also, many companies who used Global Software Development experienced process improvement. About companies with no global development operations, three in four companies not currently working offshore expect to initiate this activity in the future -the vast majority of those planning to go global intend to do so within 18 months-.

The paper is structured as follows: section 2 presents a review of the definitions proposed by different authors who have written on the subject; section 3 discusses the various advantages and disadvantages of Global Software Development; section 4 analyzes the collaboration and integration between Global Software Development and different research areas; and section 5 concludes with the observations presented in this paper.

2. Definitions

Now, there are a lot of definitions about Global Software Development, however, it is very difficult to find a common definition in the field of research. For this reason, we will make a compilation of the literature, thus being able to find all aspects of the definitions of GSD, to make our own definition correct on Global Software Development.

First, Allen, in 1984, defined *"Global Software Development (GSD) as software development that uses teams from multiple geographic locations. In some cases, these teams may be from the same organization; in other cases, there may be collaborations or outsourcing that involves different organizations. These teams could be within one country (in fact, evidence suggests that once teams are separated by more than 50 meters, further distance is immaterial) or on other sides of the world"* [7]. This definition, despite being of 1984, already had in mind that the Global Software Development teams could be located at one or more sites.

On the other hand, Carmel, in 1999, defined *"One specific case of a 'hybrid' team is a globally distributed software development team. Globally distributed projects involve two or more teams working together from different geographical locations to accomplish common project goals. In addition to geographical dispersion, globally distributed teams face time-zone and cultural differences that may include different languages, national traditions, values and norms of behavior that may greatly reduce the extent of socialization between remote counterparts"* [8]. In this definition it is noteworthy that the teams are in multiple geographic locations, but in our opinion this is not strictly necessary, because in Global Software Development teams may be located in the same geographic locations.

From 2001, we have found many definitions about GSD; we found a new definition of Carmel with Agarwal: *"Global software development means that the software development is disperse along several sites (software development centers) that could be located in different countries and even continents. A global software team executes the activities collaborating on a common software development project"* [9]. As in the previous definition, this definition also identifies that the development team must be in multiple geographic location. Additionally, it correctly identifies that the global software team executes the activities of a common project.

In that year, Audris Mockus and James Herbsleb also defined GSD: *"global software development (GSD), in which the software development activities are distributed across multiple sites"* [10]. This brief definition also takes into account that the development team must be in multiple sites.

The last definition found in 2001 is of Herbsleb and Moitra that defined the GSD as *"organizations have increasingly been adopting a geographically distributed software development paradigm, which characterizes software development as a multisite, multi-culture, and global undertaking"* [11]. This short definition identifies the main aspects about GSD: development as a multisite, multi-culture, and global undertaking.

In the paper *A Framework for Overcoming Supplier Related Threats in Global Projects* there is a definition about GSD, defined by Karolak, Sahay, et al.: *"The concept of global software development (GSD) addresses distribution of common software life cycle activities among teams separated by various boundaries, such as contextual, organizational, cultural, temporal, geographical, and political. Risks associated with these boundaries make managers struggle with pressures unique to this type of environment"* [12,13]. In this definition, the researchers identified correctly that the teams are separated by various boundaries, but these boundaries may be contextual, or geographical, and not only geographical, as other researchers identified.

Later, in 2005, Ågerfalk et al., defined as Global Software Development *"All activities of the software lifecycle where the project involves actors who are dispersed across at least two locations which are separated by country or continental borders, and typically across multiple time zones with a degree of socio-*

cultural distance amongst the actors" [14]. In this definition as in the Carmel one, it is identified that the team must be dispersed across multiples sites, with different time zones with a degree of socio-cultural distance.

Finally, the last definition found is in 2011, by Darja Šmite and Claes Wohlin: *"Many software companies are responding by shipping their development work to offshore locations, either through outsourcing projects in whole or in parts to a third party or through insourcing to an offshore site within their own organizations"* [15]. Noteworthy in this definition that may be outsourcing and insourcing, these aspects are not taken into account in the other definitions.

Other researches, refer the GSD with virtual teams, for example in 2009, Bird et al., identified the software development team as *"A software development team can geographically be distributed within a campus, city, country, or globally"* [16]. In this definition about software development teams, it is identified what must be distributed a team a project of GSD.

In 1988, Curtis, Krasner and Iscoe, defined virtual team as *"a popular structure in software development for several reasons: they provide access to lower-cost labor as well as access to a range of disciplines and technical specialties"* [17]. In this definition, the researchers identified some of the most important features in the global software development teams, which are the cost reduction and the access to a range of disciplines and technical specialties.

On the other hand, Townsend, DeMarie and Hendrickson, in 1998, defined virtual team as *"Virtual teams are composed of geographically distributed coworkers linked though information technologies to achieve an organizational task"* [18]. According to our opinion, this definition is not accurate enough, as the virtual team is not strictly required to be geographically distributed.

In the paper *Occurrence and effects of leader delegation in virtual software teams*, of Suling Zhang et al., of 2009, defined virtual team as *"The structure that is typically in place does not constitute virtuality for each and every team member but, rather, distributed teams in which some subsection of the software development team is co-located and other sections are virtual […]. The overall team is working on the same software product, but the work has typically been compartmentalized in some way so that each co-located portion of the team has specific assignments. However, the work is such that there is a need for continued communication between each of the non-co-located portions of the team to resolve integration issues"* [19]. In these definitions, the researchers identified correctly that the teams can be co-localized or not. Furthermore, so also correct, each team is assigned a project some related tasks.

With all the definitions found, we will get the consistent characteristics, from our point of view, and we will propose our own definition. It is worth noting that this definition should not be too extensive, but rather a brief one which is easy to understand. For definition, first, we must point it out that we want to develop a software project. Furthermore, the teams members may or may not be in the same geographical location and therefore can have different time zone and / or socio-cultural scopes. Then, with these features greater performance and effectiveness can be achieved for the project. In summary, GSD is the development of a software project in which the members could be in different places, time zones or even having different socio-cultural scopes, in order to achieve a higher performance and effectiveness in project implementation.

3. Advantages and drawbacks y Negocios

Actually, there are a lot of drawbacks to perform a Global Software Development Project, because the person is resistant to change and does not like the change in the organization of the company. However, there are not only drawbacks, there are many advantages about Global Software Development, which are the following:

A distributed Software Configuration Management (for managing the system component versions) reduces miscommunication because it enforces a common work process and a common view of the project [9]. With this advantage, everything will be much more documented, and then later will be more easily to solve the bugs.

Similarly the advantage of the previous, Martha Maznevski and Kathy Chudoba found that effective virtual teams had a *"deep rhythm" of regular team meetings, both face-to-face and over distance. Of course, a meeting is but communication formalism. Communication need not always occur within a formal, hierarchical configuration. When global software teams collaborate on innovative projects, informal channels of coordination —or lateral channels— are critical. They "help developers fill in the details in the work, handle exceptions, correct mistakes…"* [9].

For example, if an organization can manage daily handoffs of work between remote sites and focus attention around the clock on critical-path tasks, it is possible to take advantage of widely dispersed time zones. We could theoretically extend the productive hours of the day from the current 8- to 10- hour norm to somewhere near the limit of 24 hours [11]. In this way, gained time zone effectiveness and reduced cost in various countries. This advantage is one of the big reasons that may motivate companies for doing a project using GSD, as because it takes less time to complete the project on time effectiveness, the project cost is also lower.

In the paper *A review of awareness in distributed collaborative software engineering*, it is stated that distributed software development offers a number of theoretical benefits, including shortened time-to-market cycles and rapid response to customer needs since collaborations are independent of time and space. The mixing of developers with different cultural backgrounds can help trigger new ideas. Finally, distributed software development benefits from access to a larger qualified resource pool with the promise of reduced development cost [20]. One notable advantage of all these previous mentioned paper, is that the mixing of developers with different cultural backgrounds can help trigger new ideas, as this idea is an important idea and no other paper mentions, and the company is an aspect that may differ from your competitors.

This continues to be facilitated by the availability of well educated and technically competent software engineers in low-cost centers in Eastern Europe, Latin America, India and the Far East [9,21]. It is a commonly held belief that these savings can be coupled with the opportunity for round the clock development facilitated by the temporal difference between remote development locations. The logic underpinning this approach is that these two factors can facilitate competitive pricing and reduce time to market, thus enabling companies to compete more effectively by gaining, expanding or maintaining their market share [11,22].

According to the paper A Structured Approach to Global Software Development, 10 factors were determined which were directly relevant and needed to be specifically addressed in order to establish and facilitate the operation of globally distributed virtual teams. These factors are summarized as follows [22]:

1. Understand why, at what cost and what risk a distributed strategy is undertaken
2. Provision of effective infrastructure and documented process
3. Requirement to effectively establish the teams
4. Implement an efficient distributed team project management strategy
5. Ensure the development of common goals, objectives and rewards
6. Need for the clear definition of roles and responsibilities
7. Address issues related to culture, communication, motivation and fear
8. Ensure provision of adequate training and knowledge transfer
9. Facilitate and monitor the operation of collaborative and supportive teams
10. Document and leverage lessons learned

Following these ten factors, a GSD project would be much easier to carry out; thus, the quantity of GSD projects was increased.

On the other hand, another factor very important is that English is mandatory, and language classes are provided in most non-native speaking countries to leverage skills [23]. Furthermore English is spoken by most people in the world.

Also, it is now easier to do these projects because electronic workspaces based on groupware systems can provide a supportive infrastructure for an evaluation in the context of GSD [14].

A drawback of the Global Software Development is that some of the places where some virtual work teams are not enough qualified. Thus, in India, most of the large IT organizations provide training programs for their new employees [24]. Then, the drawback is not so serious because unskilled teams are formed.

Many companies do not realize GSD projects because there are many drawbacks, however now there are a lot of techniques to reduce these drawbacks, as these [9]:

- Intensive collaboration
- Cultural distance
    - Bridgehead
    - Internalization of Foreign Entity
    - The cultural liaison
    - Language
- Temporal distance

So far, we have seen many of the advantages of a GSD project, however there are many drawbacks. Below are shown the drawbacks found in the literature on the GSD.

Working on a globally distributed project means operating costs for planning and managing people, along with language and cultural barriers. It also creates jealousy as the more expensive engineers (who are afraid of losing their jobs) are forced to train their much cheaper counterparts [23]. These drawbacks are important, because creating jealousy between teams can disastrously affect to the project development.

The paper *Global Software Development shows many drawbacks to perform a GSD project* [11]. Based on statistical modeling of development interval and on survey results, that multisite development tasks take much longer than comparable collocated tasks and that communication and coordination play major roles in this delay. Moreover, deciding how to divide up the work across sites is difficult. Solutions are constrained by the resources available at the sites, their levels of expertise in various technologies, the infrastructure, and so on.

In the same paper, another fundamental challenge is the organization's resistance to GSD. This resistance often surfaces because of misalignment between senior and middle management on the intent and perceived benefits of GSD. Many individuals might believe their jobs are threatened, experience a loss of control, and fear the possibility of relocation and the need for extensive travel [11]. This challenge is important because if the team members work under pressure, can affect team effectiveness.

GSD requires close cooperation of individuals with different cultural backgrounds [11]. This aspect can be a drawback or an advantage, because although some people do not like this aspect, there may be other people who like it and want to learn about other cultures.

On the other hand, previous qualitative research suggests that multi-site development may increase development cycle time [25].

Gary Anthes presents a telling example of poor communication in a Global Software Development project, when a tester interpreted a spacebar instruction as a *"b-l-a-n-k,"* clearly not the intended message of the sender. Furthermore, is more difficult to share knowledge with no co-located teams, because for example if a team member has a question and another member is co-located can performing the question.

In the paper *A review of awareness in distributed collaborative software engineering* also identifies some of the drawbacks of the GSD. Developing and maintaining such awareness is more difficult in distributed software teams than co-located ones. This is because the awareness information required during software collaboration is tacit, inherent, dynamic and contextual and therefore extremely challenging to distribute automatically. It is tacit since most of what developers do in collaboration spaces builds from experience, skills, heuristics and interactions that can hardly be documented and inherent since this knowledge is deeply bound to these developers. Its dynamic nature stems from the ever changing state of software projects. Finally, the relevance of such information varies across differing project contexts: developer, task and artifact [20].

Studies have revealed the problems caused by these particular attributes of distributed teams. These include poor visibility and control of remote resources; inadequate communication, collaboration and coordination across distributed teams; diminished trust; and, lack of shared context awareness [20].

On the other hand, another researches identified that some of the difficulties encountered include such factors as the problem of understanding requirements, testing of systems and the coordination of these types of projects [8].

These difficulties are further compounded by cultural and language differences, lack of communication, geographical and temporal distance from team members and the customer, different process maturity levels, development and testing tools, standards, technical ability and experience. As a result the management of globally distributed software development projects has been recognized as a difficult and complex task [8].

Distance has been identified as a key problem and by its very nature introduces barriers and complexity into the management of globally distributed projects [8].

All drawbacks above are difficulties for project managers, and managing all these is a challenge for them. Furthermore, although some advantages reduce project development time, the drawbacks increase management time because is very difficult to coordinate all this.

With all these advantages and drawbacks, the companies could decide whether it would be appropriate to undertake a Global Software Development project or not. To make this decision, the company must consider future projections having it, because the company may want or not have customers in other countries.

Researcher	Advantage	Researcher	Drawback
[9]	Reduces miscommunication	[23]	Costs for planning and managing people
[11]	Gain time zone effectiveness	[23]	Language and cultural barriers
[11]	Reduced cost in various countries	[23]	Creates jealousy as the more expensive engineers
[20]	Shortened time-to-market cycles	[11]	Multisite development tasks take much longer than comparable co-located tasks
[20]	Rapid response to customer needs	[11]	Communication and coordination play major roles in this delay
[23]	English is mandatory	[11]	To divide up the work across sites is difficult
[11,22]	Expanding or maintaining their market share	[11]	Many individuals might believe their jobs are threatened, experience a loss of control, and fear the possibility of relocation and the need for extensive travel
[9]	There are a lot of techniques to reduce drawbacks	[20]	Poor visibility and control of remote resources
		[20]	Inadequate communication, collaboration and coordination across distributed teams
		[20]	Diminished trust; and, lack of shared context awareness
		[8]	Problem of understanding requirements, testing of systems and the coordination of these types of projects

Table 1. Advantages and Drawbacks

4. Areas research

Global Software Development is a great tool for the software development due to the several advantages that have been mentioned before. However, although this is referent to software, it can apply to many fields, inferring this technique as globalization of processes development. In this context, there are other research areas where globalization and software development can be integrated. Thanks to globalization, it is possible that many tasks can be performed, like the education or the team-working.

In reference to education, some comments will be made about e-learning. It is possible defining e-learning as *"all forms of electronic supported learning and teaching, which are procedural in character and aim to effect the construction of knowledge with reference to individual experience, practice and knowledge of the*

RPM 10 (2) (2013) ISSN 1698-2029

learner. Information and communication systems, whether networked or not, serve as specific media [...] to implement the learning process." [26]. In other words, e-learning is the ability to learn through online services and mobile devices.

This is an example of the advantages of globalization, which allow students learning without being in a physical class. Thus, it enables the people who are working or can't move learning without the necessity to go to the class physically.

In the same line, but in the opposite direction, it is necessary to teach students the concept of globalization, because in this world, more people are getting connected every day. If the focus is Software Engineering, instead of teaching the concept of globalization, then Global Software Development must be taught.

In the university, there are various courses which teach concepts about software development and Global Software Development. Also, studies mention that it is necessary that the courses are performed together by different universities so students can interact with other students from different cultures [27]. However, there are difficulties for obtain an adequate level of coordination and collaboration between the universities that allow interaction among students [28].

It is also important that students can practice the theory. Thus, there are some universities that organize practical activities with other universities in different countries in which students communicate with others through e-mail, telephone and instant messaging, and so, they can simulate what happen in the company [29].

Another possibility is to learn Global Software Development in business environment. Although this is uncommon, there are some experiences in which was given a training course for the employers of a company -the course included topics such as communicative practices, cultural differences, coordination and confidence among the participants- [30].

Because Global Software Development is a new technique, it has been shown how the concept of globalization and learning are taught in the context of software engineering. Now, it is possible to know how to integrate other research areas and Global Software Development to motivate staff and increase the benefits that brings this technique so that, ultimately, IT companies can improve its software development department.

Another research area in which is possible to integrate Global Software Development is gamification. Gamification is *"the use of game design elements in non-game contexts"* [31], like team-working or processes development, with the goal of adopt some behavior in people.

This technique may encourage people to do boring tasks. It is possible to get this by maximizing the individual competition (for example, when a person plays to a videogame, he or she returns to play for overcome his or her score), or, over all, maximizing the social competition (when a person wants to overcome the score of the rest of people), in which should get the best performances. However, this competence can produce negative effects among the employers, so it is necessary control an excessive competence.

Gamification will have great importance in the future as shown by a study of the consultant and technological research Gartner, that predicts *"over 70 percent of Global 2000 organizations will have at least one gamified application by 2014"* [32] and *"50 percent of innovations will be gamified by 2015"* [33].

But, how gamification can help to Software Engineering? The main contribution of gamification is to motivate people for improving the performance of a process in which is participant. Regarding Global Software Development, this technique can gamificate tools used for the coordination of people or teams wherever, repositories that are used by the team, process manager information used to collect the activity of team members, or tools for evaluation and testing.

Another example of using gamification in a global context is encouraging the social competence between different groups from around the world who belong to the same development teams. Thus, it motivates different development groups to be the best and improve the performance of all team.

A real example of this happened with the tool of cloud storage Dropbox. An initiative of Dropbox, called *"Space Race"* [34], made that universities from around the world compete for achieving the highest number of students enrolled in the web service. As a reward, a higher number of students of a university resulted in a higher storage capacity for those students. Therefore, each university students was motivated in order to its university could win the competition.

Another way to motivate people and employers of a company is coaching. Coaching allows to *"develop and maximize what is best in each individual, keep the individual focused and aware of new opportunities for growth and development, identify and change the thoughts or beliefs that limit the development of the individual, and reconcile private and professional life"* [35].

There are a lot of kinds of coaching: sports coaching, personal coaching, executive coaching, business coaching, etc. Coaching is based in the relation between the coach and the sportsman (coaching is born in sports context), where the coach try to improve the qualities of the sportsman. This technique was moved to other contexts, like the personal and the business. Personal coaching *"is aimed at clarifying values and visions, as well as the establishment of new objectives and actions for the individual to lead a more satisfactory life"* [35].

Otherwise, the business and executive coaching are oriented to improve the interpersonal and communication skills, leadership development, balance professional and private sphere, planning and strategy capacity development, conflict management and increase productivity.

As already stated, coaching allows motivating people. In the case of business coaching, this motivation is oriented to people are working in a company, from the highest office to the lowest office. However, this technique permits the development of general competencies and the improvement of learning and performance instead of develop technical competencies and professional careers how doing other techniques, like mentoring.

Sometimes, people confuse the coaching and mentoring concepts and use both interchangeably for referring the same one. However, the definition of mentoring can be summarized in *"the matching of a novice with a more experienced person in the same role"* [36], or in other words, some experienced person teaches to a new employee in the company how develop his -or her- career. Note is the fact that mentoring is one of the practices of one of maturity levels of People-CMM, a maturity framework that helps address the critical people issues in an organization [37].

Talking about Global Software Development, there are several studies that show the benefits of coaching in this area. In various studies of Christof Ebert [23,38], it is concluded that it is necessary a certain level of coaching in global projects for reduce cost of non-quality -the time needed to detect and correct defects-

and improve the performance of the inexperienced engineers. In another study of Maria Paasivaara [39], she investigates the effect of coaching in three global projects when these introduce agile practices in their IT organizations, and concludes that change management can succeed if experienced coaches collaborate from an early phase of the project. Mentoring is also important in Global Software Development, because it may be a way to bridge the gap between people, and solve cultural issues in the organizations [40].

One of the principal issues in Global Software Development is the different cultures of the people, as show Geert Hofstede in his publications [41] (it is important to mention his website [42], where it is possible find a comparison between several countries about the dimensions that include his study), but coaching in short and medium term, and mentoring in long term, can help to solve this and others challenges of Global Software Development.

With reference to the investigation of Maria Paasivaara mentioned before, agile practices in global projects was discussed. Thereby, it is also a good idea to talk about agile methodologies in the context of Global Software Development. The principles of agile methodologies allow teams to develop software and responding quickly to changes that may arise during the project [43].

Going into further detail, agile methodologies, through *"Agile Manifesto"* [44], reward the team interactions, the correct functionality of software, collaboration with the client and respond to the changes. Therefore, everyday it is more common the use of agile methodologies in the organizations. Moreover, there are a lot of agile methodologies that can use the organizations, but the main methodologies are Scrum, eXtreme Programming, Kanban and Crystal.

Due to rapid growth of agile methodologies, and the relation of this with the dealing of people (team-working, collaboration with the client, etc.), it is also important linking agile methodologies with globalization and, in particular, with Global Software Development. However, how can these areas be linked when agile methodologies need a face to face communication and in globalization people can be thousands of miles away?

Thanks to agile methodologies, it is possible to improve the communications in global projects [45-47]. Also, the shorts iterations of agile methodologies bring various benefits, like transparency of work progress to all partners. While the customer can monitor real progress, the distributed developers can get instant feedback on their work, which is motivating [46].

The collaboration between agile methodologies and Global Software Development may be beneficial for the latter and can help meet the challenges of this; only it is necessary a balanced level between these areas [47].

5. Conclusions

In this paper, we have analyzed different definitions of this technique by several authors. From the definitions that have been presented, we have elaborated our own definition, including various aspects like the situation and context of the team members and expected results using this process.

Also, we performed a comparison between the advantages and drawbacks of Global Software Development in which discusses all the details of this.

Finally, we explore different knowledge areas which can apply in Global Software Development for improve the performance and the use of this in the companies. In particular, we could see how this technique is integrated with e-learning, gamification, coaching and mentoring, and agile methodologies.

For future research, we have various ideas for continue this work. On the one hand, it is possible to make a study of the problems and its possible solutions of Global Software Development; too it is possible to make a systematic review. On the other hand, a future line is to deepen in the different research areas and write various papers about the collaboration between Global Software Development and each one of those.

References

[1] Azizi Yahaya, Jamaludin Ramli, Shahrin Hashim, Mohd. Ali Ibrahim, Raja Roslan Raja Abd Rahman, and Noordin Yahaya, "Discipline Problems among Secondary School Students in Johor Bahru, Malaysia," *European Journal of Social Sciences*, vol. 11, no. 4, 2009.

[2] Jae-Nam Lee, Minh Q. Huynh, Kwok Ron Chi-wai, and Shih-Ming Pi, "The Evolution of Outsourcing Research: What is the Next Issue?," *Proceedings of the 33rd Hawaii International Conference on System Sciences*, 2000.

[3] D. Šmite, C. Wohlin, T. Gorschek, and R. Feldt, "Empirical evidence in global software engineering: a systematic review," *Empir Software Eng*, vol. 15, no. 1, pp. 91–118, Feb. 2010.

[4] E. Carmel, *Global software teams: collaborating across borders and time zones*. Upper Saddle River, NJ, USA: Prentice Hall PTR, 1999.

[5] K.-S. Na, J. T. Simpson, X. Li, T. Singh, and K.-Y. Kim, "Software development risk and project performance measurement: Evidence in Korea," *Journal of Systems and Software*, vol. 80, no. 4, pp. 596–605, Abril 2007.

[6] SIIA, "SIIA Global Software Development Survey Report," Dec. 2006.

[7] R. Sangwan, N. Mullick, and M. Bass, *Global Software Development Handbook*. CRC Press, 2007.

[8] I. Oshri, J. Kotlarsky, and L. P. Willcocks, "Global software development: Exploring socialization and face-to-face meetings in distributed strategic projects," *The Journal of Strategic Information Systems*, vol. 16, no. 1, pp. 25–49, Mar. 2007.

[9] E. Carmel and R. Agarwal, "Tactical approaches for alleviating distance in global software development," *IEEE Software*, vol. 18, no. 2, pp. 22 –29, Apr. 2001.

[10] A. Mockus and J. Herbsleb, "Challenges of global software development," in *Software Metrics Symposium, 2001. METRICS 2001. Proceedings. Seventh International*, 2001, pp. 182 –184.

[11] J. D. Herbsleb and D. Moitra, "Global software development," *IEEE Software*, vol. 18, no. 2, pp. 16 –20, Apr. 2001.

[12] D. W. Karolak, *Global Software Development: Managing Virtual Teams and Environments*, 1st ed. Los Alamitos, CA, USA: IEEE Computer Society Press, 1999.

[13] S. Sahay, B. Nicholson, and S. Krishna, *Global IT Outsourcing: Software Development across Borders*. Cambridge University Press, 2003.

[14] M. Ali Babar, "A framework for groupware-supported software architecture evaluation process in global software development," *Journal of Software: Evolution and Process*, vol. 24, no. 2, pp. 207–229, 2012.

[15] D. Smite and C. Wohlin, "Strategies Facilitating Software Product Transfers," *IEEE Software*, vol. 28, no. 5, pp. 60 –66, Oct. 2011.

[16] C. Bird, N. Nagappan, P. Devanbu, H. Gall, and B. Murphy, "Does distributed development affect software quality?: an empirical case study of Windows Vista," *Commun. ACM*, vol. 52, no. 8, pp. 85–93, Agosto 2009.

[17] B. Curtis, H. Krasner, and N. Iscoe, "A field study of the software design process for large systems," *Commun. ACM*, vol. 31, no. 11, pp. 1268–1287, Nov. 1988.

[18] A. M. Townsend, S. M. DeMarie, and A. R. Hendrickson, "Virtual teams: Technology and the workplace of the future.," *ACAD MANAGE PERSPECT*, vol. 12, no. 3, pp. 17–29, Aug. 1998.

[19] S. Zhang, M. Tremaine, R. Egan, A. Milewski, P. O'Sullivan, and J. Fjermestad, "Occurrence and Effects of Leader Delegation in Virtual Software Teams," *International Journal of e-Collaboration*, vol. 5, no. 1, pp. 47–68, 2009.

[20] I. Omoronyia, J. Ferguson, M. Roper, and M. Wood, "A review of awareness in distributed collaborative software engineering," *Software: Practice and Experience*, vol. 40, no. 12, pp. 1107–1133, 2010.

[21] Galen B. Crow and Balakrishnan Muthuswamy, "International Outsourcing in the Information Technology Industry: Trends and Implications," presented at the Communications of the International Information Management Association, 2003, vol. 3.

[22] Valentine Casey and Ita Richardson, "A Structured Approach to Global Software Development," *European Systems and Software Process*, 2008.
[23] C. Ebert and P. De Neve, "Surviving global software development," *IEEE Software*, vol. 18, no. 2, pp. 62 –69, Apr. 2001.
[24] O. Gotel, V. Kulkarni, M. Say, C. Scharff, and T. Sunetnanta, "Quality indicators on global software development projects: does 'getting to know you' really matter?," *Journal of Software: Evolution and Process*, vol. 24, no. 2, pp. 169–184, 2012.
[25] J. D. Herbsleb, A. Mockus, T. A. Finholt, and R. E. Grinter, "An empirical study of global software development: distance and speed," in *Proceedings of the 23rd International Conference on Software Engineering, 2001. ICSE 2001*, 2001, pp. 81 – 90.
[26] Djamshid Tavangarian, Markus E. Leypold, Kristin Nölting, Marc Röser, and Denny Voigt, "Is e-Learning the Solution for Individual Learning?," *Electronic Journal of e-Learning*, vol. 2, no. 2, pp. 273–280, 2004.
[27] D. Petkovic, G. D. Thompson, and R. Todtenhoefer, "Assessment and comparison of local and global SW engineering practices in a classroom setting," *SIGCSE Bull.*, vol. 40, no. 3, pp. 78–82, Jun. 2008.
[28] U. Bellur, "An Academic Perspective on Globalization in the Software Industry," in *Computer Software and Applications Conference, 2006. COMPSAC'06. 30th Annual International*, 2006, vol. 1, pp. 53 –54.
[29] W. L. Honig and T. Prasad, "A classroom outsourcing experience for software engineering learning," in *Proceedings of the 12th annual SIGCSE conference on Innovation and technology in computer science education*, New York, NY, USA, 2007, pp. 181–185.
[30] R. Prikladnicki and L. Pilatti, "Improving Contextual Skills in Global Software Engineering: A Corporate Training Experience," in *IEEE International Conference on Global Software Engineering, 2008. ICGSE 2008*, 2008, pp. 239 – 243.
[31] S. Deterding, D. Dixon, R. Khaled, and L. Nacke, "From game design elements to gamefulness: defining 'gamification'," in *Proceedings of the 15th International Academic MindTrek Conference: Envisioning Future Media Environments*, New York, NY, USA, 2011, pp. 9–15.
[32] Christy Pettey, "Gartner Predicts Over 70 Percent of Global 2000 Organisations Will Have at Least One Gamified Application by 2014," *Gatner*, 09-Nov-2011. [Online]. Available: http://www.gartner.com/it/page.jsp?id=1844115.
[33] Brian Burke, "What's Next: The Gamification of Everything," *Gartner*. 27-Jan-2011.
[34] Dropbox, "Space Race," *Dropbox*, 2012. [Online]. Available: https://www.dropbox.com/spacerace.
[35] Ricardo Colomo Palacios and C. C. Lumbreras, "Mentoring & Coaching. Perspectivas en las TICs Mentoring & Coaching. IT Perspective. Mentoring & Coaching. IT Perspective.," 2006.
[36] Reiss, K., *Leadership and coaching for educators*. Thousand Oaks, California, USA: Corwin Press, 2007.
[37] Bill Curtis, Sally A. Miller, and William E. Hefley, *People Capability Maturity Model (P-CMM)*, vol. 2. Carnegie Mellon University, Pittsburgh, Pennsylvania: Software Engineering Institute, 2001.
[38] C. Ebert, C. H. Parro, R. Suttels, and H. Kolarczyk, "Improving validation activities in a global software development," in *Proceedings of the 23rd International Conference on Software Engineering*, Washington, DC, USA, 2001, pp. 545–554.
[39] M. Paasivaara, "Coaching Global Software Development Projects," in *2011 6th IEEE International Conference on Global Software Engineering (ICGSE)*, 2011, pp. 84 –93.
[40] C. Casado-Lumbreras, R. Colomo-Palacios, P. Soto-Acosta, and S. Misra, "Culture dimensions in software development industry: the effects of mentoring," Jun-2011. [Online]. Available: http://e-archivo.uc3m.es/handle/10016/14307. [Accessed: 09-Jan-2013].
[41] G. Hofstede, Culture's Consequences: International Differences in Work-Related Values. SAGE, 1980.
[42] Hofstede, Geert, "The Hofstede Centre," *The Hofstede Centre*, 2012. [Online]. Available: http://geert-hofstede.com/countries.html.
[43] José H. Canós, Patricio Letelier, and María del Carmen Penadés, "Métodologías Ágiles en el Desarrollo de Software," Universidad Politécnica de Valencia, 2003.
[44] Kent Beck, Mike Beedle, Arie van Bennekum, Alistair Cockburn, Ward Cunningham, Martin Fowler, James Grenning, Jim Highsmith, Andrew Hunt, Ron Jeffries, Jon Kern, Brian Marick, Robert C. Martin, Steve Mellor, Ken Schwaber, Jeff Sutherland, and Dave Thomas, "Manifesto for Agile Software Development," *Manifesto for Agile Software Development*, Feb-2001. [Online]. Available: http://agilemanifesto.org/.
[45] A. Avritzer, F. Bronsard, and G. Matos, "Improving Global Development Using Agile," in *Agility Across Time and Space, ISBN 978-3-642-12441-9. Springer-Verlag Berlin Heidelberg, 2010, p. 133*, 2010, p. 133.
[46] M. Paasivaara and C. Lassenius, "Could Global Software Development Benefit from Agile Methods?," in *International Conference on Global Software Engineering, 2006. ICGSE'06*, 2006, pp. 109 –113.

[47] B. Ramesh, L. Cao, K. Mohan, and P. Xu, “Can distributed software development be agile?,” *Commun. ACM*, vol. 49, no. 10, pp. 41–46, Oct. 2006.

Juegos serios para la Gestión de Ingeniería de Software

Alejandro Ruiz Fernández
Universidad Carlos III de Madrid
Madrid - España
100301324@alumnos.uc3m.es

***Abstract**: In this paper the concept of "Serious Games" is explained. Moreover, the aim is to clarify if "Serious Games" can add value to Software Engineering Management for professional practice but also from a educational perspective.*

***Resumen**: En este artículo se analiza el concepto de "Serious Games", y se identificará si los "Serious Games" aportan valor a la gestión de la ingeniería de software, tanto en la práctica profesional como en la enseñanza de la misma en entornos de educación superior.*

***Keywords**: Serious Games; Software Engineering, Software Engineering Management.*

1. Introducción

Los "Serious Games", son juegos dedicados al aprendizaje, los cuales están muy integrados en la sociedad, debido a su capacidad para llevar al jugador a un estado de diversión y a su vez a una recopilación de conocimientos, muchos autores que han tratado el tema de los "Serious Games", llegan a la conclusión de que tienen grandes impactos positivos y muchos aspectos que nos pueden beneficiar en nuestra vida cotidiana, además también mejoran el autocontrol, el reconocimiento del problema y la solución del problema, toma de decisiones, mejor memoria a corto plazo y largo plazo, y un aumentó en las habilidades sociales tales como la colaboración, la negociación y la toma de decisiones. [1,2]. Debido a esto los "Serious Games", son un gran aporte a la sociedad.

El resto del artículo se estructura de la siguiente manera. El primer apartado es el denominado "Serious Games", donde os acercaremos de manera breve al concepto de los "Serious Games", este apartado está compuesto por otros sub apartados, el primero de estos sub-apartados, es el concepto, donde mediante definiciones explicamos de manera más detallada lo que son los "Serios Games", el siguiente apartado, son las ventajas, donde se ven las aportaciones de estos tipos de juegos tanto al usuario como a la sociedad de hoy en día y por ultimo para cerrar este gran apartado, una clasificación de los "Serious Games". Los siguientes apartados uno es sobre la aportación que puede dar la Ingeniería del Software a los "Serious Games" y el otro sobre la aportación de la gestión de la ingeniería del software. Para finalizar unas conclusiones donde daré mi opinión sobre el artículo.

2. Serious Games

Como pequeña introducción a los "Serious Games", se podría decir que son juegos diseñados para un propósito principal, más que para la pura diversión, como pueden ser los juegos normales, en donde el usuario busca la diversión y pasar un buen rato durante el uso del juego.

En 1970, Clark Abt ya definió este término en su libro Serious Games, publicado por Viking Press, aunque hablaba principalmente de los juegos de mesa y de los juegos de cartas. En 2005, Mike Zyda abordó este término de una forma actualizada y lógica, donde decía que [3]:*"La mejora de la tecnología de la simulación visual y las comunidades de realidad virtual, los juegos serios proporcionar un sistema de entrega para la instrucción de la organización de los videojuegos y la capacitación".*

Para terminar los "Serious Games", aparecen en varias áreas bastante importantes, como son [4]:

- La Educación
- La Sanidad
- La Defensa
- El Arte y la cultura
- La Religión

2.1. Concepto

La definición básica o concepto básico es [5]: *"Un juego diseñado para un propósito principal, más que para la pura diversión, donde la palabra "Serious", se refiere a las industrias como la educación, la sanidad, la ingeniería, la religión etc.".*

Según Michael Zyda [1], un Serious Game, se podría definir como: "*Un concurso mental, donde juegas con un equipo de acuerdo a unas normas específicas, que utiliza el gobierno para el entretenimiento o la formación empresarial, la educación, la salud, las políticas públicas y los objetivos estratégicos de la comunicación*", además Michael y Chen [6], definen los Serious Games como *"juegos en los que su objetivo principal no es ni el entretenimiento, ni el placer, ni la diversión"*, apoyándome en estos investigadores y en una definición mas técnica, se puede decir que los Serious Games, son juegos destinados al aprendizaje o mejora de habilidades y que la diversión o el ocio no son su principal propósito como suele ser en los demás juegos.

Aparte de las definiciones de los investigadores, Serious Games en 2002, dice que la iniciativa de los Serious Games se centra en los usos para los juegos en la exploración de los problemas de gestión y liderazgo que se enfrenta el sector público [7]. Parte de su carta general es ayudar a establecer vínculos productivos entre la industria de los juegos electrónicos y proyectos relacionados con el uso de juegos en la educación, capacitación, salud y política pública [6].

	Serious Games	Entertainment Games
Tarea vs Rica experiencia	La resolución de problemas en el enfoque	Se prefiere una experiencia rica
Enfoque	Elementos importantes de aprendizaje	La diversión
Simulación	Supuestos necesarios para las simulaciones viables	Simulación de procesos simplificados
Comunicación	Debe reflejar comunicaciones naturales	A menudo la comunicación es perfecta

Tabla 1. Diferencias entre los "Games" y los "Serious Games"

2.2. Desventajas

En cuanto a las ventajas de los "Serious Games", encontramos varias, ya que los "Serious Games", son juegos donde su principal objetivo no es la diversión, sino el aprendizaje.

Una de las ventajas que plantean Kurt Squire y Henry Jenkins es que "Lo que sí sabemos es que los juegos, entornos y sistemas de simulación, etc., permiten a los estudiantes a experimentar situaciones que son imposibles en el mundo real, por razones de seguridad, costo, tiempo, etc." [8], lo cual es una gran ventaja, ya que se puede aprender sin poner en peligro la vida, como por ejemplo en un simulador de vuelo y también el coste, ya que un simulador de vuelo es más barato que pilotar uno de verdad.

En cuanto a los impactos positivos, los juegos pueden apoyar el desarrollo de una serie de diferentes habilidades, como comentan Mitchell y Savill-Smith, habilidades analíticas y espaciales, estratégicas habilidades y capacidades de comprensión, el aprendizaje y el recuerdo, habilidades psicomotoras, selectiva visual atención, etc., e incluso los juegos violentos pueden ser beneficioso, ya que proporcionan una salida para aliviar la frustración [2].

Se puede ver claro que los "Serious Games", tienen grandes impactos positivos y muchos aspectos que nos pueden beneficiar en nuestra vida cotidiana, además también mejoran el autocontrol, el reconocimiento del problema y la solución del problema, toma de decisiones, mejor memoria a corto plazo y largo plazo, y un aumentó en las habilidades sociales tales como la colaboración, la negociación y la toma de decisiones. [1,2]

Además de muchos aspectos positivos, también hay aspectos negativos, según Mitchell y Savill-Smith, discuten una serie de estas cuestiones. Los posibles impactos negativos incluyen [2]:

- **Problemas de salud** (dolores de cabeza, fatiga, estado de ánimo columpios, lesiones por esfuerzos repetitivos, etc.)
- **Problemas psicosociales** (depresión, aislamiento social, menos comportamiento positivo hacia la sociedad en general, el sustituto para el desarrollo social, etc.)
- **Los efectos de los videojuegos violentos** (conducta agresiva, negativa desarrollo de la personalidad, etc.)

2.3. Clasificación de los Serious Games

Los "Serious Games", se clasifican en distintas áreas, lo que hace que haya varios tipos, las áreas más utilizadas y donde más uso se les da, es en las áreas de defensa o en temas militares, en la educación también tiene una gran repercusión y por último la sanidad, aunque los nombrados antes sean los más utilizados, los "Serious Games" también aparecen en áreas como el arte y la cultura, la religión o en temas del gobierno.

Según el autor Zyda, el cual afirma que la tecnología de los juegos serios puede aplicarse a dominios tan diversos como la salud, la política pública, la comunicación estratégica, la defensa, la formación y la educación [3], de esta manera podemos clasificar los "Serious Games" en las áreas antes nombradas, además del auto Zyda, Michael y Chen, también clasificaron los "Serious Games", donde podemos encontrar, juegos militares del gobierno, juegos educativos, juegos corporativos, juegos dedicados a la salud, juegos de carácter político, religiosos y de arte. A pesar de estas categorizaciones, sobre todo muchos juegos pueden pertenecer a más de una categoría [6].

Una vez vista la clasificación de los "Serious Games", profundizaremos en los distintos tipos que conforman la clasificación de los "Serios Games".

2.3.1. Juegos Militares o de defensa

Como dicen Michael y Chen, Los militares tienen una larga historia de uso de los juegos para el entrenamiento. Entre los más antiguos juegos de guerra nos encontramos el tablero de juego Chaturanga de la India y la China Wei Hei, el cual tiene cuatro mil años de antigüedad. [6], Estos juegos son extremadamente complejos, ya que son usados para planificar las batallas o cualquier tema militar, los más

usados son los simuladores de tanques o helicópteros. Junto con el desarrollo hacia los simuladores más avanzados, la cantidad relativa de dinero que se gasta en los juegos militares también ha cambiado, y los equipos de simulación y juegos de guerra ocupan 4000 millones de dólares.

Históricamente hablando, Michael y Chen dicen, que las simulaciones militares han sido, y siguen siendo dominantes, pero hay una tendencia hacia el uso de componentes comerciales de software y de hardware. [6].

Aparte de los juegos producidos para la guerra, el ejército de EE.UU. mostró un gran interés en los videojuegos de entretenimiento para sus propósitos de entrenamiento. Uno de los ejemplos más famosos es el entrenador Bradley. También conocida como Military Battlezone Battlezone o el Ejército, este juego es una versión personalizada de Battlezone. El juego original pone al jugador con un tanque en un mundo 3D, y le pide derribar vehículos opuestos.

2.3.2. Juegos de Educación

Los juegos educativos no han estado en uso hasta la década de 1990 con ordenadores multimedia, a pesar de que estos juegos fueron creados y utilizados mucho antes. En ese momento, los juegos educativos y otros programas se desarrollaron en "edutainment". Sin embargo, el interés en edutainment pronto disminuyó, en parte debido a la pobre calidad de los juegos en sí, y en parte debido a un creciente interés en Internet [6].

Con el renovado interés general en los “Serious Games”, los desarrolladores de juegos han pasado de "paradigmas interactivos de aprendizaje de habilidades y de perforación hacia enfoques situacionales y construccionista" [1].

Los juegos en la educación está ganando aceptación, pero su uso no está muy extendido, y es un tema controvertido [1,6].

Uno de los más famosos antepasados de los "Serious Games" se pueden encontrar en el campo de la Educación. The Oregon Trail comenzó como un juego de sólo texto creado por tres profesores de historia: Don Rawitsch, Bill y Paul Heinemann Dillenberger.

Pero el juego original sigue siendo popular hoy en día gracias a las versiones de teléfonos móviles y una aplicación de Facebook. En última instancia, este juego demuestra claramente que un juego "educativo" o "seious" no es necesariamente lo opuesto a un "popular y de éxito comercial" del juego [4].

2.3.3. Juegos de dedicados a la salud

Las aplicaciones de “Serious Games” relacionados con la salud y la asistencia sanitaria son cada vez más comunes, y en la actualidad existe un gran número de ellos. Además, Ben Sawyer (co-fundador de la Iniciativa de Serious Games) espera que el área de la salud en los “Serious Games” crezca más en los próximos años [9].

Hay una gran variedad de tipos y áreas de aplicaciones relacionadas con la salud física o mental, como [9]:

- Aptitud física.
- La Educación en salud / cuidado auto dirigido.

- Terapia de distracción.
- Recuperación y rehabilitación.
- Formación y simulación.
- Diagnóstico y tratamiento de enfermedades mentales / condiciones mentales.
- El funcionamiento cognitivo.

3. Serious Games e Ingeniería del Software

Dentro del mundo del videojuego, se necesita que se apliquen técnicas de Ingeniería del Software, ya que tienen grandes desafíos técnicos, los cuales se pueden soluciones de manera exitosa con la Ingeniería del Software, aunque suele ser duro porque puede que haya algún fracaso a la hora de aplicar estas técnicas.

Poco a poco se esta integrando los juegos como puente al aprendizaje de la ingeniería del software, ya que es una manera más entretenida y didáctica para que los jóvenes aprendan el mundo de la ingeniería del software y además luego puedan aplicar lo aprendido en la mejora de los "Serious Games", según unos estudios, aplicar los videojuegos en el aprendizaje de la ingeniería del software, tiene un gran éxito, ya que es mas entretenido y ameno.

Para orientar las decisiones acerca de si el uso de juegos para la educación de la ingeniería de software, un equipo de investigadores de la Universidad de "do Vale do Itajaí" de Brasil llevó a cabo una revisión sistemática de la literatura para ver si había pruebas para responder a estas preguntas [5]:

- ¿Qué tipo de juegos se utilizan en la enseñanza de ingeniería de software?
- ¿Cómo son de eficaces los juegos para la educación en comparación con otros métodos de enseñanza?
- ¿Cómo pueden ser mejor diseñados los juegos para ser mas atractivos e útiles para mejorar el aprendizaje?

Las pruebas realizadas no son muy concluyentes, ya que a los usuarios que probaron los juegos, se les pidió su opinión subjetiva de si habían aprendido algo o no, pero según el siguiente cuadro podemos ver que si ha habido una mejora del conocimiento mediante el uso de Software Engineering Management [5].

En el siguiente cuadro podemos apreciar, como la ingeniería del software administrativa tiene un gran uso en el mundo de los videojuegos, la ingeniería del software en cuanto a procesos esta algo más baja, pero se puede apreciar el uso de la ingeniería del software.

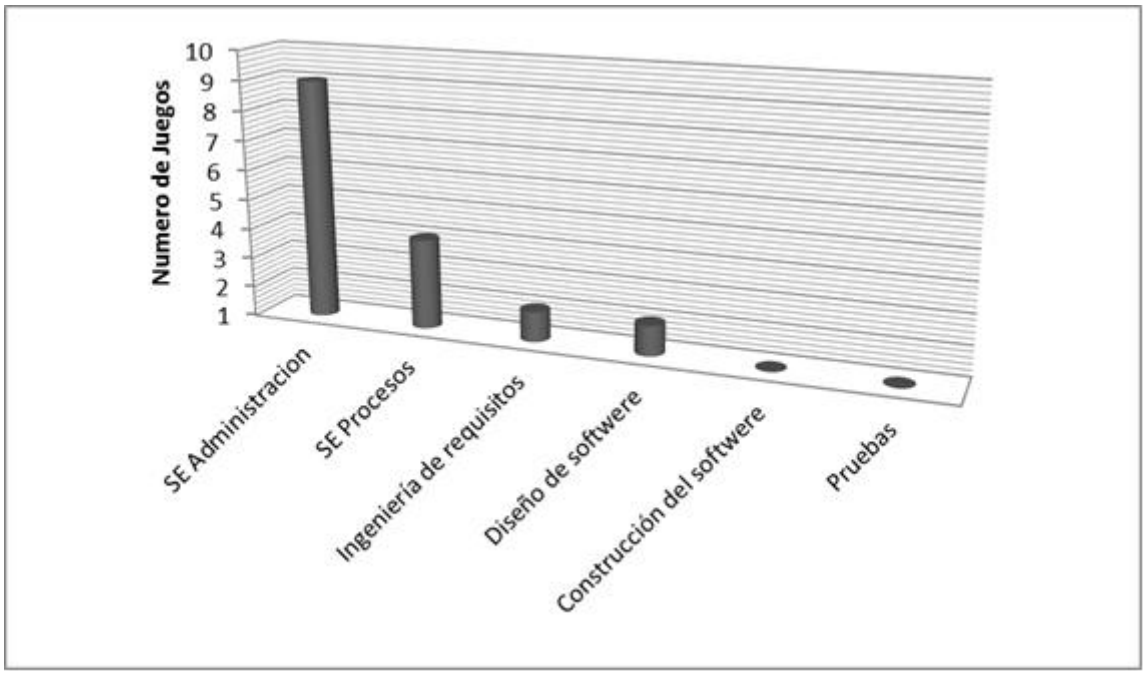

Ilustración 1. Área de conocimiento de ingeniería de software [5]

En esta otra gráfica, podemos apreciar el gran uso del conocimiento en el resultado final del aprendizaje, lo malo son tanto las habilidades como la actitud las cuales están por debajo del conocimiento.

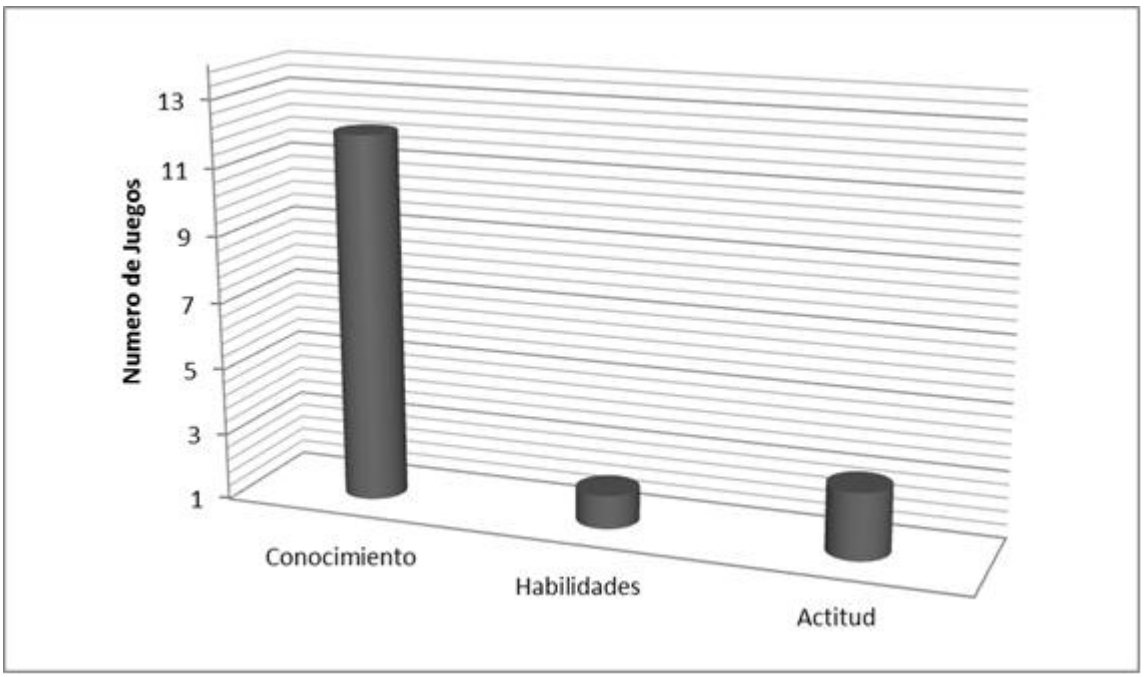

Ilustración 2. Dominio resultado de aprendizaje [5]

4. Serious Games y Software Engineering Management

El Software Engineering 2004 define el volumen de un cuerpo básico de conocimientos llamado Conocimiento Software Engineering Education (SEEK), que fue la base de recomendaciones curriculares. Software Engineering 2004 también definió siete resultados de los estudiantes, que incluyen [10]:

- *"Trabajar como individuo y como parte de un equipo para desarrollar y entregar los artefactos de software de calidad."*
- *"Conciliar los objetivos del proyecto en conflicto, la búsqueda de compromisos aceptables dentro de las limitaciones de coste, tiempo, conocimientos, sistemas y organizaciones existentes."*

Donde se opta por hacer software de calidad y que sea en coste y tiempo, se dispone de una tabla donde hay cinco unidades del conocimiento, que pertenecen al área de Software Engineering Management, las cuales aparecen a continuación.

KA/KU	Software de gestión	Horas Requeridas
MGT.con	Conceptos de gestión	2
MGT.pp	Planificación de proyectos	6
MGT.per	Personal del proyecto y de la organización	2
MGT.ctl	Control del Proyecto	4
MGT.cm	Configuración del software de gestión	5

Tabla 2. Software de gestión de unidades de conocimiento [10]

En esta tabla podemos observar que donde más horas se requieren es en la planificación de proyectos, por eso siempre en todo proyecto, ya sea de videojuegos o de otro campo, la planificación tiene que ser donde dediquemos más tiempo, ya que es el pilar del proyecto, y la ingeniería del software tiene conceptos y herramientas para planificar bien los proyectos.

5. Conclusiones

Como hemos podido comprobar a lo largo de la investigación, los "Serious Games", son usados hoy en día, y tiene una gran repercusión en el aprendizaje, tanto en las áreas de la Educación, donde ha constado más su iniciación, aunque está mejorando con la llegada de las nuevas tecnologías, la sanidad, que es el área con mayor expectativa, y del cual se cree que va a tener una gran evolución, y por último el más utilizado, el área dedicado a la defensa o el área militar, el cual dispone de varios simuladores de vuelo, ya que para el aprendizaje y a su vez para la seguridad de los alumnos y el coste que lleva pilotar un avión se convierte en una herramienta muy importante en este área, también veo un gran aportación de la Ingeniería del Software, aunque todavía le queda mucho por avanzar, para que se la valore como tiene que ser, pero mediante las gráficas podemos ver una gran evolución en este campo, en mi opinión creo que los "Serious Games", son una gran herramienta para el aprendizaje y para las nuevas generaciones, ya que es un medio didáctico y ameno a su vez.

Referencias

[1] H. Ellis, S. Heppell, J. Kirriemuir, A. Krotoski, and A. McFarlane, "Unlimited learning," *Entertainment and Leisure*, 2006.
[2] A. Mitchell and C. Savill-smith, *The use of computer and video games for learning*. .
[3] M. Zyda, "From visual simulation to virtual reality to games," *Computer*, no. September, pp. 25–32, 2005.
[4] D. Djaouti and J. Alvarez, "Origins of serious games," *Serious Games and …*, pp. 1–22, 2011.
[5] F. Shull and E. S. Engineering, "Voice of evidence 'To Game or Not to Game?'," *IEEE Computer Society*, 2009.
[6] D. R. Michael and S. L. Chen, "Serious Games: Games That Educate, Train and Inform.pdf." 2005.
[7] "'Serious Games Initiative'."2002.
[8] K. Squire and H. Jenkins, "Harnessing the power of games in education," vol. 3, pp. 5–33, 2003.
[9] T. Susi, M. Johannesson, and P. Backlund, "Serious games: An overview," 2007
[10] C. Caulfield, D. Veal, and S. P. Maj, "Teaching Software Engineering Management – Issues and Perspectives," vol. 11, no. 7, pp. 50–54, 2011.

ISO/IEC 29110: Current overview of the standard

Euclides J. Moreno Campos, Mary-Luz Sanchez-Gordón, and Ricardo Colomo-Palacios

Departamento de Informática

Universidad Carlos III de Madrid

Madrid - España

100289251@alumnos.uc3m.es, mary_sanchezg@hotmail.com, rcolomo@inf.uc3m.es

***Abstract**: The software industry recognizes the value of VSEs in contributing valuable products and services. Unfortunately current ISO/IEC standards do not completely address the needs of VSEs. Due to this, the ISO/IEC 29110 standard has been developed. While reading this paper, one can find previous attempts of approaching ISO and ISO/IEC standards such as ISO/IEC 12207, ISO/IEC 15289, ISO/IEC 15504 and ISO 9001 to VSEs, as well as models inspired on ISO and ISO/IEC standards and Maturity models like CMMI and oriented to VSEs such as MoProSoft and projects like COMPETISOFT. A summary of part 5 of the standard, making special focus on the Entry Profile, belonging to the generic profile group as well as an initial implementation of the standard in VSEs both through the creation of a customized approach and the utilization of the available deployment packages are going to be depicted. Finally a short comment about the importance of Deployment Packages and the Network Support Centers in adopting and raising awareness about the standard, as well as future work to be done on the standard, for example, the completion and creation of new profiles is going to be found.*

***Keywords**: VSEs, Very Small Entities, ISO/IEC 29110, Lifecycle profiles, Software Engineering Standards, Software Processes.*

1. Introduction

According to the OECD (Organization for Economic Co-operation and Development) SME (Small and Medium Enterprises) and Entrepreneurship Outlook report (2005) “SMEs constitute the dominant form of business organization in all countries world-wide, accounting for over 95 % and up to 99 % of the business population depending on country” [1]. Hence, a set of studies cited in [2] show the same behavior in technology-based companies. As depicted in Table 1, in Europe, 85% of the Information Technology (IT) sector's companies have between 1 and 10 employees. In Canada, the Montreal area was surveyed, and it was found that close to 80% of IT companies have fewer than 25 employees. Another study conducted by the Technology Assessment Group (CITA) of Wallonia has published similar data, which reveal that about 60% of IT companies there have fewer than 5 employees. In Brazil, small IT companies represent about 70% of the total number of companies. In Northern Ireland, a survey reports that 66% of IT organizations within companies employ fewer than 20 employees [2]. Finally, a study performed in 2004 by Industria Mexicana de Software demonstrated that 92% of the IT companies in Mexico are small and medium-sized (with less than 100 people) [3]. Thus, certain Very Small Entities (VSEs[1]) provide software components that are being assembled in larger software companies in order to generate critical and intensive software configurations [4,5], so one must conclude that software industry recognizes the value of VSEs in contributing valuable products and services [1,2].

[1] The terms "very small entity” and “very small entities" (VSE/VSEs) have been defined by the ISO/IEC JTC1/SC7 Working Group 24 (WG24) as being "an entity (enterprise, organization, department or project) having up to 25 people" and have subsequently been adopted for the use in the ISO/IEC 29110 standard. *V. Ribaud, P. Saliou, R. V. O’Connor, and C. Y. Laporte, “Software engineering support activities for very small entities,” in Systems, Software and Services Process Improvement, Grenoble (France), 2010, pp. 165–176.*

Size (employees)	IT Companies (%)	Country/Region
<= 10	85	Europe
< 25	80	Montreal (Canada)
< 5	60	Wallonia (Belgium)
(Small IT companies)	70	Brazil
< 20	66	Northern Ireland
< 100	92	Mexico

Table 1. Presence of VSEs in general level IT companies by country/region

According to [1], from studies and surveys conducted, it is clear that the majority of International Standards do not address the needs of VSEs. Conformance with these standards is difficult, if not impossible. Subsequently VSEs have no, or very limited, ways to be recognized as entities that produce quality software in their domain. Therefore, VSEs are often cut off from some economic activities. It has been found that VSEs find it difficult to relate International Standards to their business needs and to justify the application of the standards to their business practices. Most VSEs can neither afford the resources, in terms of number of employees, budget and time, nor do they see a net benefit in establishing software life cycle processes. To rectify some of these difficulties, a set of guides has been developed according to a set of VSE characteristics. The guides are based on subsets of appropriate standards elements, referred to as VSE Profiles (ISO/IEC 12207[2], ISO/IEC 15289[3], ISO/IEC 15504[4], ISO 9001[5]). The so-called guides are gathered into the standard *ISO/IEC 29110 Software engineering — Lifecycle profiles for Very Small Entities*, which describes processes for project management and software implementation [6] and pretends to facilitate access to, and utilization, ISO software engineering standards in VSEs [2].

The section two details a critical analysis of previous attempts of approaching ISO/IEC standards and other methodologies and models such as MoProSoft[6] to Very Small Entities. Section three gives an overview of the standard, making special focus on part five, highlighting the Entry Profile, belonging to the generic profile group. Section four describes an initial implementation of the standard both through the creation of a customized approach and the utilization of the available deployment packages in small technology-based firms. Conclusions are discussed in section five.

2. Previous and related work

2.1. Previous and related standards

The International Organization for Standardization (ISO) and the International Electrotechnical Commission (IEC) have been working together in order to design standards related to software engineering issues. Products of these efforts are:

[2] International Standard ISO/IEC 12207 Software Life Cycle Processes.
[3] International Standard ISO/IEC 15289 Systems and software engineering — Content of life-cycle information products (documentation).
[4] International Standard ISO/IEC 15504 Software Process Improvement Capability Determination.
[5] International Standard ISO 9001 Quality management systems — Requirements.
[6] Modelo de Procesos para la Industria de Software (Processes Model for the Software Industry).

ISO/IEC 12207 Software Life Cycle Processes

According to [7], this standard establishes a top-level architecture of the life cycle of software. The life cycle begins with an idea or a need that can be satisfied wholly or partly by software and ends with the retirement of the software. The architecture is built with a set of processes and interrelationships among these processes. The derivation of the processes is based upon two basic principles: *modularity and responsibility*.

- Modularity: The processes are modular; that is, they are maximally cohesive and minimally coupled to the practical extent feasible. An individual process is dedicated to a unique function.
- Responsibility: A process is considered to be the responsibility of a party in the software life cycle. In other words, each party has certain responsibilities. Responsibility is one of the key principles of total quality management.

ISO/IEC 15289 Systems and software engineering — Content of life-cycle information products (documentation)

The purpose of this International Standard is to provide requirements for identifying and planning the specific information items (information products) to be developed and revised during systems and software life cycles and service processes. The standard specifies the purpose and content of all identified systems and software life-cycle information items, as well as information items for information technology service management. The information item contents are defined according to generic document types and the specific purpose of the document. Information items may be combined or subdivided as needed for project or organizational purposes [8].

ISO/IEC 15504 Software Process Improvement Capability Determination

This standard defines a reference model for software process assessment, and a set of requirements on assessment models and methods [9]. It is intended to harmonize the many different approaches to software process assessment. It has nine parts. The reference model in Part 2 documents the set of universal software engineering processes that are fundamental to good software engineering and that cover best practice activities. It describes processes that an organization may perform to acquire, supply, develop, operate, evolve and support software and the process attributes that characterize the capability of those processes. The purpose of the reference model is to provide a common basis for different models and methods for software process assessment, ensuring that results of assessments can be reported in a common context [10].

ISO 9001 Quality Management Systems - Requirements

This international standard promotes the adoption of a process-based approach during the development, implementation and efficiency enhancements of a Quality Management System, in order to increase the customer satisfaction through the fulfillment of his or her requirements. ISO 9001 specifies the requirements for a Quality Management System, requirements that can be used for its application by organizations, with certification or contractual purposes. It focuses in Quality Management System efficiency to satisfy customer's requirements [11].

Unfortunately, none of these standards completely address the needs, goals and objectives of small and medium software enterprises. These standards were intended to be performed in larger organizations, in which the software development processes differ from the ones performed in VSEs. Nevertheless, there

have been some attempts to approach some of these standards to the small and medium software firms. Some of these attempts have taken place in Latin America; one of these efforts is called MoProSoft.

2.2. The MoProSoft model

According to [3], the high cost of SW-CMM[7] and CMMI[8] adoption in small enterprises and the need for a national standard were the basic reasons to develop a new software process model for the Mexican software industry. MoProSoft is a process model for small enterprises, built on the well-known practices of SW-CMM, ISO 9000:2000, PMBoK (Project Management Body of Knowledge) and others, and offers a new process structure, some new process documentation elements, a more precise process relationship, and an explicit process improvement mechanism. The model is complemented with the process assessment method EvalProSoft, which is based on the recommendations of ISO/IEC 15504 Part 2. The process model and the assessment method were applied to four small enterprises that had a typical Mexican software industry company profile. After experimentation (the application of MoProSoft and EvalProSoft in the four small software companies), authors evidenced the fulfillment of the following criteria [3]:

- C1. Proper for small and medium-sized enterprises (SME) with low maturity levels.
- C5. Defined as a set of processes based on internationally recognized practices.

Due to the results of this experiment the Mexican Secretary of Economy decided to formally make MoProSoft and EvalProSoft a Mexican standard [3]. Moreover, the Peruvian National Institute for the Defense of Competition and Protection of Intellectual Property (INDECOPI) developed a technical standard based on MoProSoft called *"Software Engineering: Software Development and Maintenance Process and Evaluation Models NTP 291.100:2009"* which was published in 2009. It is also important to highlight that MoProSoft was essential in the development of the ISO/IEC 29110 standard [14].

2.3. The COMPETISOFT project

Another Latin-American initiative is COMPETISOFT. The COMPETISOFT project, according to [15], is based on ISO/IEC 12207, ISO/IEC 15504, CMMI, MANTEMA[9], Métrica v3[10], Agile SPI[11], and mainly MoProSoft and EvalProSoft. This processes model is aimed to companies or internal areas of companies, dedicated to software maintenance and development. This model was developed and enhanced by persons that own wide knowledge in international model contents, as well as expertise on implanting such models in VSEs. This model consists of three categories, which cluster processes according to the typical organization structure:

[7] The Capability Maturity Model for Software (SW-CMM) describes the principles and practices underlying software process maturity and is intended to help software organizations improve the maturity of their software processes in terms of an evolutionary path from chaotic processes to mature, disciplined software processes [12].

[8] CMMI® (Capability Maturity Model® Integration) is a process improvement maturity model for the development of products and services. It consists of best practices that address development and maintenance activities that cover the product lifecycle from conception through delivery and maintenance [13].

[9] MANTEMA is a methodology for managing the Software Maintenance Process. This methodology defines precisely and rigorously all the activities and tasks, which must be executed during the Maintenance process, and explicitly considers the integration of the necessary activities for establishing and completing outsourcing relationships between customer and supplier organizations.

[10] Métrica is a methodological environment developed by the Spanish Ministry of Public Administration. In the latest version of Métrica v3, the object-oriented paradigm is included as a development option, and proposes the use of UML to model different aspects in the software lifecycle. Métrica v3 is the reference frame for software development in Spanish public entities [16].

[11] Agile SPI is a framework intended to support process improvement for the software industry. Its main goal is to motivate small and medium size companies towards improving and certifying their development processes [17].

- Corporate level management: Establishes the rationale of the organization, what the organization is willing to achieve and the respective strategies to make it possible.
- Tactical level management: Establishes action plans to implement strategies in relation to projects, processes and resources. It monitors Operation category and gives feedback to corporative management category.
- Operational level management: Performs maintenance and software development projects that cover customer's needs in terms of time and expected costs and report results to the tactical management category.

According to [14], the COMPETISOFT project is a Process improvement to enhance the competitiveness of small and medium organizations in Latin America. COMPETISOFT defined three objectives:

- Create a common methodological framework in Latin America.
- Spread the process culture into the researchers, academics and students communities.
- Influence in the standardization and certification entities, in order to establish a common and mutually recognized mechanism.

2.4. The 15504MPE project

Another effort in approaching ISO/IEC software engineering standards are depicted in [18], which describes some experiences gained from applying ISO/IEC 15504 for software process assessments focusing on process improvement in four small software companies in Brazil. The assessments has been performed in the context of a project called 15504MPE, which aims at the development of a customized assessment method based on the standard ISO/IEC 15504 adapted to small Brazilian software companies. After experimentation, the authors evidenced important benefits:

- Better understanding of the assessed processes based on the assessment results.
- Strengths and weaknesses of the assessed processes were identified in relation with the process assessment model.
- Suggestions for improvement with relevant impact on the software process were formulated and started to be implemented.
- Increased motivation for improvement due to a better understanding of the actual process and the identified weaknesses.
- Increased commitment to improve process quality.

To shorten the gap between *small and medium enterprises* and their compliance with ISO/IEC software engineering standards, a set of guides has been developed according to a set of VSE characteristics. The guides are based on subsets of appropriate standards elements, referred to as VSE Profiles. This set of guides are gathered in the so-called standard *ISO/IEC 29110 Software engineering — Lifecycle profiles for Very Small Entities* which is going to be overviewed during the next section.

3. ISO/IEC 29110: Overview of the standard

The *ISO/IEC 29110 Software engineering — Lifecycle profiles for Very Small Entities* standard is aimed to approach Software Engineering and Project Management good practices to VSEs. According to [1,19], the ISO/IEC 29110 standard is divided in five parts as follow:

ISO/IEC TR 29110-1 defines the business terms common to the VSE Profile Set of Documents. It introduces processes, lifecycle and standardization concepts, and the ISO/IEC 29110 series. It also introduces the characteristics and requirements of a VSE, and clarifies the rationale for VSE-specific profiles, documents, standards and guides.

ISO/IEC 29110-2 introduces the concepts for software engineering standardized profiles for VSEs, and defines the terms common to the VSE Profile Set of Documents. It establishes the logic behind the definition and application of standardized profiles. It specifies the elements common to all standardized profiles (structure, conformance, assessment) and introduces the taxonomy (catalogue) of ISO/IEC 29110 profiles.

ISO/IEC TR 29110-3 defines the process assessment guidelines and compliance requirements needed to meet the purpose of the defined VSEs Profiles. ISO/IEC TR 29110-3 also contains information that can be useful to developers of assessment methods and assessment tools. ISO/IEC TR 29110-3 is addressed to people who have direct relation with the assessment process, e.g. the assessor and the sponsor of the assessment, who need guidance on ensuring that the requirements for performing an assessment have been met.

ISO/IEC 29110-4-1 provides the specification for all the profiles of the Generic Profile Group. The Generic Profile Group is applicable to VSEs that do not develop critical software products. The profiles are based on subsets of appropriate standards elements. VSEs' Profiles apply and are targeted to authors/providers of guides and authors/providers of tools and other support material.

ISO/IEC 29110-5-1 provides an implementation management and engineering guide for both the Entry and Basic Profile of the Generic Profile Group described in ISO/IEC 29110-4-1. The Entry Profile describes software development of a single application by a single project team with no special risk or situational factors for start-up VSEs (i.e. VSEs who started their operation less than 3 years) and/or for VSEs working on small project (e.g. project size of less than 6 person-months). The Basic Profile describes software development of a single application by a single project team with no special risk or situational factors.

In the following subsections a summary of standard's part 5-1, with a special highlight in the *Project Management and Software Implementation* processes for the Entry Profile belongings to the Generic Profile Group, will be presented.

If one carefully sees, the lifecycle profiles follow the typical structure of an organization. The *advanced profile*[12] addresses the corporate level management, *Intermediate profile*[13] covers the tactical level management and *Entry and Basic profiles* address the operational level management; level that involves Project Management and Software Implementation processes. According to[14], the reason to include the Project Management process is that VSEs' core business is software development (Software Implementation process) and their financial success depends on project profits.

3.1. Project Management (PM) process – Purpose

The purpose of the Project Management process is to establish and carry out in a systematic way the tasks of the software implementation project, which allows complying with the project's objectives in the

[12] This profile targets VSEs, which want to sustain and grow as an independent competitive software development business.
[13] This profile targets VSEs developing multiple projects within the organizational context.

expected quality, time and cost. PM process uses the customer's statement of work to elaborate the project plan. The PM project assessment and control tasks compare the project progress against the project plan. The PM project closure activity delivers the software configuration, produced by the Software Implementation Process, and gets the customer's acceptance to formalize the end of the project. A project repository is established to save the work products during the project [1].

3.1.1. PM objectives

According to [1], the objectives are specific goals that ensure the accomplishment of the process purpose. The objectives are identified by the abbreviation of the process name, followed by the letter "O" and a consecutive number, for example PM.O1, SI.O2, etc. Each objective is followed by the square box, which includes a list of the chosen processes for the entry profile from ISO/IEC 12207:2008 and its outcomes related to the objective. In this paper, the chosen processes are only going to be numbered.

PM.O1. The *Project Plan* for the execution of the project is developed according to the *Statement of Work* and reviewed and accepted by the Customer. The tasks and resources necessary to complete the work are sized and estimated (ISO/IEC 12207:2008, 6.3.1, 6.3.7).

PM.O2. Progress of the project is monitored against the *Project Plan* and recorded in the *Progress Status Record*. Closure of the project is performed to get the Customer acceptance documented in the *Acceptance Record* (ISO/IEC 12207:2008, 6.3.2, 6.3.7, 6.4.8).

PM.O3. The *Changes Requests* are addressed, evaluated and tracked (ISO/IEC 12207:2008, 7.1.2).

PM.O4. Review meetings with the Work Team and the Customer are held. *Agreements* are registered and tracked (ISO/IEC 12207:2008, 7.2.6).

PM.O5. *Risks* are identified as they develop and during the conduct of the project (ISO/IEC 12207:2008, 6.3.4, 7.2.6).

PM.O6. Items of Software Configuration are identified and controlled (ISO/IEC 12207:2008, 7.2.2).

PM.O7. Software Quality Assurance is performed to provide assurance that work products and processes comply with the Project Plan and Requirements Specification (ISO/IEC 12207:2008, 7.2.3).

The authors in [1] also note that the implementation of the Software Quality Assurance process depicted in objective PM.07. is going to be achieved through the performance of the verifications, validations and review tasks performed in Project Management and Software Implementation processes.

3.1.2. Products

According to [1], artifacts of this process are classified in three groups:

Input Products – products required to perform the process and its corresponding source, which can be another process or an external entity to the project, such as the Customer. Identified by the abbreviation of the process name and showed as two column table of product names and sources (see Table 2).

Name	Source
Statement of Work	Customer
Software Configuration	Software Implementation
Change Request	Customer

Table 2. PM Input products [1]

Output Products – products generated by the process and its corresponding destination, which can be another process or an external entity to the project, such as Customer or Organizational Management. Identified by the abbreviation of the process name and showed as two column table of product names and destinations (see Table 3).

Name	Source
Project Plan	Software Implementation
Acceptance Record	Customer
Project Repository	Software Implementation
Meeting Record	Customer
Software Configuration	Customer

Table 3. PM Output products [1]

Internal Products – products generated and consumed by the process. Identified by the abbreviation of the process name and showed as one column table of the product names (see Table 4).

Name
Change Request
Meeting Record (only work team)
Progress status record

Table 4. PM Internal products [1]

3.1.3. PM roles involved

The roles are names and abbreviation of the functions to be performed by project team members. Several roles may be played by a single person and one role may be assumed by several persons. Roles are assigned to project participants based on the characteristics of the project (see Table 5) [1].

Name	Abbreviation	Competency
Customer	CUS	Knowledge of the Customer processes and ability to explain the Customer requirements. The Customer (representative) must have the authority to approve the requirements and their changes. The Customer includes user representatives in order to ensure that the operational environment is addressed. Knowledge and experience in the application domain.
Project Manager	PM	Leadership capability with experience making decisions, planning, personnel management, delegation and supervision, finances and software development.
Work Team	WT	Knowledge and experience according to their roles on the project.

Table 5. PM Roles involved [1]

3.1.4. PM activities

According to [1], an activity is a set of cohesive tasks and a task is a requirement, recommendation, or permissible action, intended to contribute to the achievement of one or more objectives of a process. A process activity is the first level of process workflow decomposition and the second one is a task. Activities are identified by process name abbreviation followed by consecutive number and the activity name.

PM.1 Project planning, (PM.O1, PM.O5, PM.O6, PM.O7)

The Project Planning activity documents the planning details needed to manage the project. The activity provides:

- Reviewed Statement of Work and the tasks needed to provide the contract deliverables and to satisfy customer requirements.
- Project quality assurance approach through verification and validation of work products/deliverables, customer reviews.
- Work team and customer roles and responsibilities.
- Project resources needs.
- Estimates of effort, cost and schedule.
- Identified project risks.
- Project repository to store, handle and deliver controlled product and document versions and baselines.

PM.2 Project plan execution (PM.O2, PM.O3, PM.O4, PM.O5, PM.O7)

The Project Plan Execution activity implements the documented plan on the project. The activity provides:

- Monitoring the project against the Project plan.
- Status of the Project Plan Execution.
- Change Request accepted by the Customer.
- Reviews and agreements with the Customer.

PM.3 Project assessment and control (PM.O2)

The Project Assessment and Control activity evaluates the performance of the plan. The activity provides:

- Evaluation of actual plan performance and progress against targets.
- Change requests tracking.
- Documented problem, corrective action defined, and tacked to closure.

PM.4 Project closure (PM.O2)

The Project Closure activity provides the project's documentation and products in accordance with contract requirements. The activity provides:

- Support of Customer product acceptance.
- Completion of the project and sign of the Acceptance Record.
- Summary and updated project repository for project closure.

In order to facilitate a better understanding of the standard's Project Management process structure, a table showing the interrelation between activities, objectives, roles and products is provided below (see Table 6).

ID	Activities	Associated objectives	Roles Involved	Products Involved
PM.1	Project planning	PM.O1. The Project Plan for the execution of the project is developed according to the Statement of Work and reviewed and accepted by the Customer. The tasks and resources necessary to complete the work are sized and estimated. PM.O5. Risks are identified as they develop and during the conduct of the project. PM.O6. Items of Software Configuration are identified and controlled. PM.O7. Software Quality Assurance is performed to provide assurance that work products and processes comply with the Project Plan and Requirements Specification.	PM, WT, CUS	Statement of Work, Project Plan
PM.2	Project plan execution	PM.O2. Progress of the project is monitored against the Project Plan and recorded in the Progress Status Record. Closure of the project is performed to get the Customer acceptance documented in the Acceptance Record. PM.O3. The Changes Requests are addressed, evaluated and tracked. PM.O4. Review meetings with the Work Team and the Customer are held. Agreements are registered and tracked. PM.O5. Risks are identified as they develop and during the conduct of the project. PM.O7. Software Quality Assurance is performed to provide assurance that work products and processes comply with the Project Plan and Requirements Specification.	PM, WT, CUS	Project Plan, Progress Status Record, Meeting Record, Change Request
PM.3	Project assessment and control	PM.O2. Progress of the project is monitored against the Project Plan and recorded in the Progress Status Record. Closure of the project is performed to get the Customer acceptance documented in the Acceptance Record.	PM, WT	Project Plan, Progress Status Record, Change Request
PM.4	Project closure	PM.O2. Progress of the project is monitored against the Project Plan and recorded in the Progress Status Record. Closure of the project is performed to get the Customer acceptance documented in the Acceptance Record.	PM, CUS	Project Repository

Table 6. Interrelation between PM activities, objectives, roles and products

3.2. Software Implementation (SI) process – Purpose

According to [1], the purpose of the Software Implementation process is the systematic performance of the analysis, software component identification, construction, integration and tests, and product delivery activities for new or modified software products according to the specified requirements. The execution of the SI process is driven by the project plan. SI process starts with an initiation activity of the project plan revision. Project plan will guide the execution of the software requirements analysis, software component identification, software construction, software integration and test, and product delivery activities.

3.2.1. SI objectives

SI.O1. *Tasks* of the activities are performed through the accomplishment of the current *Project Plan*.

SI.O2. *Software requirements* are defined, analyzed for correctness and testability, approved by the Customer, and communicated (ISO/IEC 12207:2008, 6.4.1, 7.1.2).

SI.O3. *Software components* and their *interfaces* are identified (ISO/IEC 12207:2008, 7.1.3).

SI.O4. Software components are produced. *Unit test* are performed to verify the consistency with software requirements (ISO/IEC 12207:2008, 7.1.5).

SI.O5. Software is produced. Software components are integrated and verified using *Test Cases and Test Procedures*. Results are recorded at the *Test Report*. Defects are corrected (ISO/IEC 12207:2008, 7.1.6, 7.1.7).

SI.O6. *Software configuration* is prepared for delivery (ISO/IEC 12207:2008, 6.1.2, 7.2.1).

SI.O7. *Verification and Validation* tasks of all required work products are performed to achieve consistency among output and input products in each activity. Defects are identified and corrected (ISO/IEC 12207:2008, 7.2.4, 7.2.5).

3.2.2. SI products

Artifacts of this process are classified in three groups and presented in the following tables:

Name	Source
Project Plan	Project Management
Project Repository	Project Management

Table 7. SI Input products [1]

Name	Source
Software Configuration: • *Requirements Specification* • *Software*	Project Management

Table 8. SI Output products [1]

Name
Software component identification
Test cases and test procedures
Software component
Test report

Table 9. SI Internal products [1]

3.2.3. SI roles involved

The roles involved in the Software Implementation process are the same which are involved in the Project Management process (Customer, Project Manager, Work Team).

3.2.4. SI activities

According to [1], Software Implementation activities are detailed as follow:

SI.1 Software Implementation initiation (SI.O1)

The Software Implementation Initiation activity ensures that the Project Plan, established in Project Planning activity, is committed to by the Work Team. The activity provides:

- Review of the Project Plan by the Work Team to determine task assignment.
- An implementation environment established.

SI.2 Software requirements analysis (SI.O2, SI.O6, SI.O7)

The Software Requirements Analysis activity analyzes the agreed customer requirements and establishes the validated project software requirements. The activity provides:

- Work Team review of the Project Plan to determine task assignment.
- Elicitation, analysis and specification of customer's requirements.
- Agreement on the customer requirements.
- Verification and validation of requirements.

SI.3 Software component identification (SI.O3, SI.O6, SI.O7)

The Software Component Identification activity transforms the software requirements to the architecture of system software components. The activity provides:

- Work Team review of the Project Plan to determine task assignment.
- Identify software components and associated interfaces.

SI.4 Software construction (SI.O4, SI.O6, SI.O7)

The Software Construction activity develops the software code and data from the Software Component Identification in the SI.3. The activity provides:

- Work Team review of the Project Plan to determine task assignment.
- Understand the identified Software Components.
- Test Cases and Test Procedures for unit and integration testing.
- Coded Software Components and applied unit tests.

SI.5 Software integration and tests (SI.O5, SI.O6, SI.O7)

The Software Integration and Tests activity ensures that the integrated software components satisfy the software requirements. The activity provides:

- Work Team review of the Project Plan to determine task assignment.
- Understanding of Test Cases and Procedures and the integration environment.
- Integrated Software Components, corrected defects and documented results.

SI.6 Product delivery (SI.O6, SI.O7)

The Product Delivery activity provides the integrated software product to the Project Manager and support for delivery. The activity provides:

- Work Team review of the Project Plan to determine task assignment.
- Delivery of the software product and applicable documentation in accordance with the Project Plan.

In order to facilitate a better understanding of the standard's Software Implementation process structure, a table showing the interrelation between activities, objectives, roles and products is provided below (see Table 10).

ID	Activities	Associated objectives	Roles Involved	Products Involved
SI.1	Software Implementation Initiation	SI.O1. Tasks of the activities are performed through the accomplishment of the current Project Plan.	PM, WT	Project Plan
SI.2	Software Requirements Analysis	SI.O2. Software requirements are defined, analyzed for correctness and testability, approved by the Customer, and communicated. SI.O6. Software configuration is prepared for delivery. SI.O7. Verification and Validation tasks of all required work products are performed to achieve consistency among output and input products in each activity. Defects are identified and corrected.	PM, WT	Project Plan, Requirements Specification
SI.3	Software Component Identification	SI.O3. Software components and their interfaces are identified. SI.O6. Software configuration is prepared for delivery. SI.O7. Verification and Validation tasks of all required work products are performed to achieve consistency among output and input products in each activity. Defects are identified and corrected.	PM, WT	Project Plan, Progress Status Record, Change Request, Software Component Identification
SI.4	Software Construction	SI.O4. Software components are produced. Unit test are performed to verify the consistency with software requirements. SI.O6. Software configuration is prepared for delivery. SI.O7. Verification and Validation tasks of all required work products are performed to achieve consistency among output and input products in each activity. Defects are identified and corrected.	PM, WT	Project Plan, Software Component Identification, Requirements Specification, Test Cases and Test Procedures, Software Components
SI.5	Software Integration and Tests	SI.O5. Software is produced. Software components are integrated and verified using Test Cases and Test Procedures. Results are recorded at the Test Report. Defects are corrected. SI.O6. Software configuration is prepared for delivery. SI.O7. Verification and Validation tasks of all required work products are performed to achieve consistency among output and input products in each activity. Defects are identified and corrected.	PM, WT	Project Plan, Test Cases and Test Procedures, Software Components, Software, Test Report, Requirements Specification, Software Configuration
SI.6	Product Delivery	SI.O6. Software configuration is prepared for delivery. SI.O7. Verification and Validation tasks of all required work products are performed to achieve consistency among output	PM, WT	Project Plan, Software Configuration

		and input products in each activity. Defects are identified and corrected.		

Table 10. Interrelation between SI activities, objectives, roles and products

4. ISO/IEC 29110: Implementation in VSEs

In this section, documented and published research work regarding to adoption and implementation of the standard in small organizations are going to be shown. In [20], the authors proposed a three-step approach of the standard in order to implement it in a small Thai government academic institute's IT department. The so-called approach consists of:

A Feasibility Study: This step emphasized on finding the possibility of adapting ISO/IEC 29110 standard into the existing software processes used by the development unit.

Risk Management: All risks were identified and evaluated based on the results of the feasibility study in order to manage the risk that may occur during the implementation. The risk management plan was created and the risk mitigation was defined.

The Execution: After considering the feasibility study result and the risk management report, the implementation of the ISO/IEC 29110 processes (PM and SI) was executed. All constraints defined in the previous steps were reviewed and monitored throughout the implementation plan.

With the three-step implementation approach (the authors say), the case study unit gained a better understanding in the ISO/IEC 29110 standard, subsequently was able to effectively handle and prepare documents under the standard. As a result, the unit has a clear, well-defined, step-by-step approach in the software development that leads to a better reputation of the organization. The authors' findings pointed out the significance of such standard with respect to apply their processes in in-house software units under government agencies.

In [21], the authors talk about Deployment Packages (DPs). Their main objective of DPs is to facilitate the implementation, by VSEs, of a Profile. A deployment package is a set of artifacts developed to facilitate the implementation of a set of practices, of the selected framework, in a VSE. DPs are available, at no cost, on the Internet [1]. This paper outline a pilot project initiative currently underway to evaluate these Deployment Packages and assist very small companies in understanding and exploring the potential usage of an international software process development standard like ISO/IEC 29110.

About pilot projects, they mention there is a series of projects that have been taking place in Canada, utilizing some of the deployment packages developed. For example in Canada a pilot study has been conducted with an IT department with a staff of 4: 1 analyst and 3 developers, who were involved in the translation and implemented 3 DPs: Software Requirements, Version Control and Project Management. In Belgium a VSE of 25 people started with a process assessment phase aiming to identify strengths and weaknesses in development related processes. This company is now working on improvement actions mainly based on the following Deployment Packages: Requirement Analysis, Version Control, and Project Management. Finally in Ireland a VSE of 8 people are working on improving project management and tracking and control practices using the Project Management deployment package. In [22], the authors also comment about a pilot project conducted with a 14-person VSE based in France, which successfully implemented ISO/IEC 29110 processes practices utilizing the available Deployment Packages.

The authors continue explaining that the Brazilian Standard Organization ABNT (Associação Brasileira de Normas Técnicas) has developed an ISO/IEC 29110 certification scheme. A first series of Brazilian VSEs should have obtained an ISO/IEC 29110 certificate of conformity during this year. The auditing scheme, developed by Brazil, will probably be used by other countries, such as Canada, to audit their VSEs. It is also known that the Peruvian National Institute for the Defense of Competition and Protection of Intellectual Property (INDECOPI) has adopted in 2012 the ISO/IEC 29110 international standard, as the NTP-RT-ISO/IEC TR 29110 Peruvian Technical Standard (Norma Técnica Peruana), which is making easier de implementation of the standard in small Peruvian software firms [23].

In [24], authors present results of implementing ISO/IEC 29110 standard (Basic profile part 5-1-2), using DPs, through the performance of pilot projects in very small Irish companies. VSEs were invited to participate in a training program in order to implement the standard. Seven VSEs joined the program. The adoption of the standard was performed in a 4-step-method:

- VSEs were sent a DP and other supporting material.
- VSEs implement the process and report on activities, successes and problems to the researchers.
- The researchers review the reports and return any useful comments to the companies.
- The researchers make any amendment to the process to ensure greater success with the next process module.

After a 3-month period, four of the firms stopped the implementation of the standard, another just quitted, the sixth one did not even started the program after an initial expression of interest and the last company stopped and then restarted work on the standard implementation and submitted some documentation after a while. The authors conclude saying that VSEs have too much work to do, with too little time and people to do it. This was supported by one company, who commented that they do not even know if they are going to be *"in business"* next month, so implementing a standard would be too much workload for them. One can evidence that in some cases, a standard is still viewed as an add-on task, not a way to do business. Nevertheless, despite the lack of apparent success in terms of bringing all companies successfully through this program, the researchers are optimistic about the future for this standard. The authors have detected the need of enhancing the mentoring and assessment labor with VSEs in order to adequately implement this type of programs.

Another initiatives for the dissemination and adoption of the standard are the Network Support Centers (NSC) commented by authors in [22]. The main purpose of NSC, born from an informal meeting conducted by WG24[14] delegates in order to create a network of collaborators, is to facilitate and develop collaborative activities between institutions in the field of software engineering, information technology and others to improve VSE capabilities especially in Software Engineering and Information Technology. The principal goals to achieve by the implementation of NSC are clear: Speed up both, the deployment of Standard and Guides for VSEs and the development and application of Guides and DPs (e.g. through pilot projects). Some participants of the Network Support Centers are:

- Center of Excellence in Information and Communication Technologies (CETIC) - **Belgium.**
- RIOSOFT agent for Brazilian software excellence in Rio de Janeiro - **Brazil.**

[14] The ISO/IEC JCT1/SC7 Working Group 24 was established in 2005 with a mandate to investigate the need for and propose software life cycle profiles and guidelines for use in VSEs [22].

- Superior School of Technology (ETS) - **Canada.**
- Parquesoft Foundation - **Colombia.**
- Tampere University of Technology, Pori - **Finland.**
- University of Western Brittany (UBO) - **France.**
- Quisqueya-America University Institute (INUQUA) - **Haiti.**
- Polytechnic University - **Hong Kong (China).**
- Lero, The Irish Software Engineering Research Center - **Ireland.**
- Public Research Center Henri Tudor - **Luxembourg.**
- University of Lima - **Peru.**
- Institute of Software Promotion for Industries - **Thailand.**

Further countries, such as Ecuador, Mexico, Spain and Japan are considering joining the NSC.

5. Conclusions

As far as one can see, it has come a long way up to the creation of a resilient standard to carry out processes of the software lifecycle in VSEs, from adapting previous ISO and ISO/IEC standards, to the construction of models such as MoProSoft, essence of the ISO/IEC 29110 standard. The adoption has been sometimes difficult as depicted in [24], and sometimes easier but still incipient as described in [20]. One might think that the main efforts in order to raise awareness among VSEs relative to the standard should be done on the Deployment Packages to bring a more simple way to implement it, as well as on the Network Support Centers, as a mean to accelerate its deployment in small and medium software firms.

One must agree with [21] that ISO/IEC 29110, as an emerging standard, has yet work to be done. The main remaining work item is to finalize the development of the two remaining profiles: *Intermediate - Management of more than one project* and *Advanced - Business management and portfolio management practices*. In addition, the development of further Profile Groups for other domains such as: Critical software, game industry, scientific software development, etc.

Finally, after carefully reviewing the existent literature, only the time will tell, according to the worldwide degree of standard's adoption, if it is fully focused on VESs. The history of Software Engineering shows that is necessary to harmonize actual practices with new proposals, therefore the recent creation of the Entry Profile and Deployment Packages are perhaps not enough to facilitate its implementation, but rather the creation of a initial framework in order to guide the implementation process is needed, framework that prepares VSEs to successfully adopt the standard, particularly VSEs located in developing countries.

References

[1] ISO/IEC, “ISO/IEC TR 29110-5-1-1:2012 Software engineering -- Lifecycle profiles for Very Small Entities (VSEs) -- Part 5-1-1: Management and engineering guide: Generic profile group: Entry profile.” ISO Central Secretary, 15-Sep-2012.

[2] C. Y. Laporte, A. April, and A. Renault, “Applying ISO/IEC software engineering standards in small settings: historical perspectives and initial achievements,” in *Proceedings of SPICE Conference, Luxembourg*, 2006.

[3] H. Oktaba, “3.2 MoProSoft®: A Software Process Model for Small Enterprises,” in *International Research Workshop for Process Improvement in Small Settings*, 2006, p. 93.

[4] C. Laporte, R. O'Connor, and G. Fanmuy, "International systems and software engineering standards for very small entities," *CrossTalk J. Def. Softw. Eng.*, vol. 26, no. 3, pp. 28–33, 2013.
[5] C. Y. Laporte, S. Alexandre, and R. V. O'Connor, "A software engineering lifecycle standard for very small enterprises," *Softw. Process Improv.*, pp. 129–141, 2008.
[6] T. Varkoi and T. Mäkinen, "A Process Assessment Model for Very Small Software Entities," in *In: Rout, T., Lami, G. & Fabbrini, F.(eds.). Process Improvement and Capability Determination in Software, Systems Engineering and Service Management, Proceedings of: 10th International SPICE Conference 2010, Pisa, Italy 18-20 May, 2010*, Pisa, Italy, 2010.
[7] R. Singh, "International Standard ISO/IEC 12207 software life cycle processes," *Softw. Process Improv. Pract.*, vol. 2, no. 1, pp. 35–50, 1996.
[8] ISO/IEC, "ISO/IEC 15289 Systems and software engineering — Content of life-cycle information products (documentation)." ISO Central Secretary, 11-Jan-2011.
[9] K. El-Emam and I. Garro, "ISO/IEC 15504," *Int. Organ. Stand.*, Nov. 1999.
[10] M. C. Paulk, "Analyzing the conceptual relationship between ISO/IEC 15504 (software process assessment) and the capability maturity model for software," in *1999 International Conference on Software Quality*, 1999.
[11] ISO, "Norma Internacional ISO 9001 Sistemas de gestión de la calidad — Requisitos (Traducción oficial)." ISO Central Secretary, 15-Nov-2008.
[12] R. Prikladnicki, J. Audy, and R. Evaristo, "Requirements management in global software development: Preliminary findings from a case study in SW-CMM context," in *The International Workshop on Global Software Development*, 2003, pp. 53–58.
[13] C. P. Team, "CMMI for Development, version 1.2," Aug. 2006.
[14] M. M. Trujillo, H. Oktaba, T. Ventura, and R. Torres, "From MoProSoft Level 2 to ISO/IEC 29110 Basic Profile: Bridging the Gap.," in *CIbSE*, 2012, pp. 28–41.
[15] M. Llaneza, G. N. Dapozo, C. L. Greiner, and M. G. Estayno, "Análisis comparativo de modelos de calidad orientado al desarrollo de software en pymes," in *XV Workshop de Investigadores en Ciencias de la Computación*, Paraná, Argentina, 2013.
[16] M. J. Escalona, J. J. Gutiérrez, F. Morero, C. L. Parra, J. Nieto, F. Pérez, F. Martín, and A. Llergo, "A Practical Environment to Apply Model-Driven Web Engineering," *Inf. Syst. Dev.*, pp. 249–258, 2010.
[17] J. A. H. Alegrıa and M. C. Bastarrica, "Implementing CMMI using a combination of agile methods," *CLEI Electron. J.*, vol. 9, no. 1, 2006.
[18] A. Anacleto, C. G. von Wangenheim, C. F. Salviano, and R. Savi, "Experiences gained from applying ISO/IEC 15504 to small software companies in Brazil," in *4th International SPICE Conference on Process Assessment and Improvement, Lisbon, Portugal*, 2004, pp. 33–37.
[19] ISO/IEC, "ISO/IEC TR 29110-5-1-2:2011 Software engineering — Lifecycle profiles for Very Small Entities (VSEs) — Part 5-1-2: Management and engineering guide: Generic profile group: Basic profile." ISO Central Secretary, 15-May-2011.
[20] V. Siddoo, N. Wongsai, and R. Wetprasit, "An Implementation Approach of ISO/IEC 29110 for Government Organizations."
[21] R. V. O'Connor and C. Y. Laporte, "Towards the provision of assistance for very small entities in deploying software lifecycle standards," in *Proceedings of the 11th International Conference on Product Focused Software*, 2010, pp. 4–7.
[22] R. V. O'Connor and C. Y. Laporte, "An Innovative Approach to the Development of an International Software Process Lifecycle Standard for Very Small Entities.," *Int. J. Inf. Technol. Syst. Approach*, 2013.
[23] INDECOPI, "NTP-RT-ISO/IEC TR 29110-5-1-2. Ingeniería de Software. Perfiles del ciclo de vida para las pequeñas organizaciones (PO). Parte 5-1-2: Guía de gestión e ingeniería: Grupo de perfil genérico. Perfil básico." INDECOPI, 16-May-2012.
[24] R. V. O'Connor and M. Sanders, "Lessons from a Pilot Implementation of ISO/IEC 29110 in a Group of Very Small Irish Companies," in *Software Process Improvement and Capability Determination*, 2013, vol. 349, pp. 243–246.

Procesos y Métricas en la WWW

En esta sección de la revista se presenta una lista ordenada de sitios web en los que se tratan los temas de interés de los lectores de la misma.

Sitios Web de Asociaciones Nacionales de Medición del Software

Alemania. Asociación Alemana de Medición del Software. DASMA. www.dasma.org
Dinamarca. Asociación Danesa de Métricas del Software. DANMET. www.danmet.dk
Finlandia. Asociación Finlandesa de Métricas del Software. FISMA. www.sttf.fi
Italia. Asociación Italiana de Métricas del Software. GUFPI-ISMA. www.gufpi-isma.org
Holanda. Asociación Holandesa de Métricas del Software. NESMA. www.nesma.nl
Reino Unido. Asociación de Métricas del Software del Reino Unido. UKSMA. www.uksma.co.uk

Sitios Web de Organismos Internacionales de Medición del Software

COmmon Software Measurement International Consortium. COSMIC. www.cosmicon.com
International Function Points Users Group. IFPUG. www.ifpug.com
International Software Benchmarking Standards Group. ISBSG. www.isbsg.org.au

Sitios Web de Laboratorios de Investigación en Medición del Software

Alemania. Laboratorio de Medición del Software. SMLAB. ivs.cs.uni-magdeburg.de/sw-eng/us
Canadá. Laboratorio de Investigación en Ingeniería del Software. GELOG. www.gelog.etsmtl.ca
España. Laboratorio de Medición del Software. CuBIT. www.cc.uah.es/cubit

Guía para Autores de Artículos de Divulgación

Los artículos de divulgación podrán ser publicados por cualquier persona que pertenezca a una organización miembro de AEMES. Con la pertinente autorización de su organización. Deberán versar sobre algún asunto de interés relacionado con el alcance de AEMES. Los artículos no tendrán revisión por pares pero no podrán ser artículos de información meramente comercial.

Los autores deberán enviar los artículos electrónicamente utilizando la dirección de correo electrónico rpm@aemes.org. Por favor dirigir los artículos al Editor de la Revista de Procesos y Métricas de las Tecnologías de la Información. El artículo debe ser enviado para el proceso de revisión en formato Microsoft Word.

Guía para Autores de Artículos de Investigación

Los artículos de investigación podrán ser publicados por cualquier persona que pertenezca a una organización miembro de AEMES. Deberán versar sobre algún asunto de interés relacionado con el alcance de AEMES.

Los autores deberán enviar los artículos electrónicamente utilizando la dirección de correo electrónico rpm@aemes.org. Por favor dirigir los artículos al Editor de la Revista de Procesos y Métricas de las Tecnologías de la Información. El artículo debe ser enviado para el proceso de revisión en formato Microsoft Word.

El envío de un artículo implica que el trabajo descrito no ha sido publicado previamente (excepto en el caso de una tesis académica), que no se encuentra en ningún otro proceso de revisión, que su publicación es aceptada por todos los autores y por las autoridades responsables de la institución donde se ha llevado a cabo el trabajo y que en el caso de que el artículo sea aceptado para su publicación, el artículo no será publicado en ninguna otra publicación en la misma forma, ni en Español ni en ningún otro idioma, sin el consentimiento de AEMES.

Una vez recibido un artículo se enviará al autor de contacto por correo electrónico un acuse de recibo.

Todos los artículos de investigación recibidos para ser considerados para su publicación serán sometidos a un proceso de revisión. La revisión será realizada por dos o, en su caso, tres expertos independientes. Para asegurar un proceso de revisión lo más correcto posible los nombres de los autores y los revisores permanecerán confidenciales. Una vez revisado un artículo se enviarán por correo electrónico los resultados de la revisión. En el caso de que el artículo haya sido rechazado se adjuntarán las valoraciones de los revisores. El proceso de revisión está libre de costes para los autores.

Una vez que un artículo haya sido aceptado, se solicitará a los autores que transfieran los derechos de autor del artículo a AEMES. Recibida la transferencia, se solicitará a los autores el envío de una versión del artículo lista para publicación que se deberá enviar en formato Microsoft Word.

La publicación de un artículo en la revista está libre de costes para los autores, pero todas las instituciones de origen de todos los firmantes del artículo deberán ser miembros de AEMES.

Guía para la preparación de manuscritos

El texto deberá estar escrito en un correcto castellano (Uso Español) o en Inglés (Uso Británico). Excepto el abstract que deberá estar escrito en un correcto Inglés (Uso Británico).

Abstract y Resumen. Se requiere un abstract en inglés con un máximo de 200 palabras. El abstract deberá reflejar de una forma concisa el propósito de la investigación, los principales y resultados y las conclusiones más importantes. No debe contener citaciones. Se debe presentar a continuación del abstract en inglés una traducción del mismo al castellano bajo el epígrafe Resumen.

Palabras clave. Inmediatamente después del Resumen se proporcionarán un conjunto de 5 palabras clave evitando términos en plural y compuestos, tampoco se deben usar acrónimos o abreviaturas a no ser que sean de un uso ampliamente aceptado en el campo del artículo. Estas palabras claves serán utilizadas a efectos de indexación.

Subdivisión del artículo. Después del abstract y el resumen, que no llevarán numeración, se debe dividir el artículo en secciones numeradas, comenzando en 1 y aumentando consecutivamente. Las subsecciones se numerarán 1.1 (1.1.1, 1.1.2, etc.), 1.2, etc. No se deben incluir subdivisiones por debajo del tercer nivel (1.1.1). Cada sección o subsección debe tener un título breve que aparecerá en una línea separada.

Apéndices. Si hay más de un apéndice, se deben identificar como A, B, etc. Las ecuaciones en los apéndices tendrán una numeración separada: (Eq. A.1), (Eq. A.2), etc.
Agradecimientos. Se deben situar antes de las referencias, en una sección separada.
Tablas. Se deben numerar las tablas consecutivamente de acuerdo con su orden de aparición en el texto. Se deben poner títulos a las tablas debajo de las mismas.
Figuras. Se deben numerar las figuras consecutivamente de acuerdo con su orden de aparición en el texto. Se deben poner títulos a las figuras debajo de las mismas.
Referencias. Se debe verificar que cada referencia citada en el texto se encuentra también en la lista de referencias y viceversa. Los trabajos no publicados o en proceso de revisión no pueden ser citados.

Citaciones en el texto: Un solo autor. El primer apellido del autor, seguido de una coma y la primera inicial, seguida de un punto, a continuación, tras una coma, el año de publicación. Todo entre corchetes. Dos o más autores. Los nombres de los autores, siguiendo el formato de un solo autor, separados por puntos y comas y el año de publicación. Lista. Las listas deberán ser ordenadas, primero de forma alfabética y luego, si fuera necesario, de forma cronológica. Si hay más de una referencia del mismo autor en el mismo año deben ser identificadas por las letras “a”, “b”, etc., situadas después del año de su publicación.

Formato

Los autores deberán bajar de la página web de RPM en el sitio web de AEMES el artículo de ejemplo y seguir estrictamente el mismo formato.

ASOCIACIÓN ESPAÑOLA DE MÉTRICAS DE SISTEMAS INFORMÁTICOS AEMES

Facultad de Informática de la UPM. Campus de Montegancedo. 28660-Boadilla del Monte (Madrid)
Teléfono: 91 336 66 08. Fax: 91 336 74 12. E-mail: admon@aemes.org

FORMULARIO DE INSCRIPCIÓN A AEMES

Información Personal
Apellidos: Nombre:
Teléfono: e-mail:
Dirección Personal:
Empleo:

Inscripción Institucional o Empresarial
Institución o Empresa: C.I.F.:
Dirección:

Datos Bancarios
Número de Cuenta:
Titular:

Autorizo a AEMES a cargar a la cuenta arriba indicada la cuota Anual de inscripción que asciende a 360 € + la cuota de inscripción que asciende a 150 €

Fecha y Firma