Diagnosis of software projects based on the Viable System Model

Julio César Puche Regaliza[1,*], Alfredo Jiménez[2], Pablo Arranz Val[1]

[1]Department of Applied Economics, University of Burgos

Faculty of Economics and Business, Infanta Doña Elena Square, s/n, 09001, Burgos, Spain

email: {jcpuche, parranz}@ubu.es

[2]Department of Management, Kedge Business School

680, cours de la Libération, 33405, Talence. France

email: alfredo.jimenez@kedgebs.com

**Corresponding author* (+34 947 259 389)

Diagnosis of software projects based on the Viable System Model

Abstract

***Purpose**: This work aims to identify the shortcomings that may impair or endanger the viability of software projects, and attemps to reduce their failure rates.*

***Design/methodology/approach**: A diagnosis based on the Viable System Model of software projects is shown. The data collection processes necessary to make a diagnosis have been carried out through a questionnaire and personal interviews with 38 companies in the Information and Communications Technology (ICT) sector.*

***Findings**: The subsequent descriptive analysis allows us to detect 8 weak points in the organizational structure of the software projects. We highlight two of the 8 weak points in particular: the nondefinition of the next lower level of recursion and the need to improve the exploration mechanisms used to evaluate future scenarios.*

***Research limitations/implications**: Although the sample seems small, a great effort was expended to interview the 38 companies due to an extension of the interview time (approximately 75 minutes) and the confidentiality and complexity of the topic, which required gathering a high level of detail and in-depth information on each of the projects and companies in the survey.*

***Practical implications**: This work offers managers a detailed reference for the diagnosis or design of viable software projects.*

***Originality/value**: The novel aspect of the work focuses on the application of the Viable System Model for proposing organizational changes in the development of software projects. This holistic perspective of the organizational structure of software projects will allow an increase in their success rate.*

Keywords: Software Project, Structural Diagnosis, Cybernetics, Organizational Cybernetics, Viable System Model, Organizational Change Management.

1 Introduction

The importance of software in modern-day society is well known. The software production process is usually focused on a project-based viewpoint and generally seeks three goals: (1) budget compliance; (2) delivery time compliance; and (3) sufficient compliance with the established requirements (Jun *et al.*, 2011; Albert *et al.*, 2017; Sanchez et al., 2017). Although there is no clear consensus on the definition of a successful project (Bronte-Stewart, 2015; Pollack et al., 2018), meeting these goals generally leads to successful projects (Pressman and Maxim, 2014; Radujkovic and Sjekavica, 2017; Gandhi et al., 2018). In any case, the percentage of successful projects is too low (Standish Group, 2018), even though some authors question the validity of these percentages (Eveleens and Verhoef, 2010).

The reasons the percentage of successful projects is not sufficiently high have been widely studied by a large number of authors that identified the critical factors that affect the success of a software project (Mohapatra and Gupta, 2011; Nasir and Sahibuddin, 2011; Pankratz and Loebbecke, 2011; Müller and Jugdev, 2012; Sutar and Ghatule, 2013; Lehtinen *et al.*, 2014; Ahimbisibwe *et al.*, 2015; 2017). Although it seems clear that there is no unanimous agreement on these success factors, common factors are frequently found in these studies. By looking at these, we can group the factors in different categories, for example, process-related factors, personal factors and technical factors.

There are many models and practices to improve the process-related factors, there are many theories and techniques for managing people and improving the personnel factors, and the technology offers multiple tools to improve the technical factors. All of these models, practices, theories, techniques and technologies have had partial and relative success in the development of software projects (Pressman and Maxim, 2014). Nonetheless, the failure rate of software projects is still not falling sufficiently. These partial solutions refer mainly to the object and system level, while the current status of a complex adaptive system (as a software

project it is) speaks to the urgent need to develop a complexity science knowledge system, which requires that the complexity is addressed at three levels: the object, system and meta levels (Motloch, 2016). In this same sense, Beer (1979) points out that if the organizational structure is inadequate, none of the knowledge and practices applied to the organization will guarantee its success.

Therefore, it seems clear there is a need for meta level contributions based on a structural change for the building of software projects where complexity and uncertainty can be handled in a way that ensures the quality, reliability, simplicity and robustness of the project at a reasonable cost and delivery time. Moreover, the challenges that their own environment poses also needs consideration (Beer, 1995, Hornstein, 2015; Miterev *et al.*, 2017a; 2017b).

System Thinking offers a wide variety of methodologies, models, methods and techniques that are especially appropriate to fill this research gap (Monat and Gannon, 2015; Remington and Pollack, 2016). It seems to be appropriate to visualize a software project from another perspective, in such a way that all the elements involved in the project are linked as a whole, including its environment. From the broad set of available tools, the principles of Cybernetics (Wiener, 1948) and more specifically Organizational Cybernetics (Beer, 1959) (science of efficient organization, it applies the principles of Cybernetics to the study of organizations) seems particularly well positioned to be applied as the conceptual framework for the management of a software project. Within this framework, we can apply the Viable System Model (VSM) (Beer, 1979, 1981, 1985) as a tool to diagnose or to design the organizational structure of software projects in a scientific manner in favor of their viability. Ensuring viability is a way of contributing to the increased rate of success of these sorts of projects. The advantages of doing so arises from the systemic, comprehensive and multilevel nature of the

VSM, as well as its capability to deal with the complexity of a software project and its interaction with the environment (Beer, 1985).

In this work, we suggest a systemic contribution that facilitates the treatment of the dynamic complexity that a software project implicitly encapsulates together with its interaction with the environment; the proposed contribution also facilitates the treatment of a set of structural and behavioral changes in an organization. We propose an organizational change management formula that leads to the definition of a structure that incorporates the different roles people adopt, the units in which they participate, the resources they use, and the relationships between all these elements (Beer, 1985). More specifically, we present a descriptive analysis that allows exploration of the degree of compliance of the structural characteristics defined by the VSM on software projects.

The novel aspect of this work focuses on diagnosis of the organizational structure of a set of software projects from a holistic VSM perspective. This diagnosis will make it possible to identify the degree of compliance to the characteristics indicated by the VSM as a necessary and sufficient condition to achieve the viability of an organization; in this case, a software project. In this way, the reinforcement of these characteristics can contribute to increasing the percentage of successful software projects and as a consequence, the progress of software project management.

In addition to the initial proposal and the objective that have been presented here, we will also detail the theoretical background of the VSM; subsequently, we will develop the research methodology, and we will present the results that were obtained along with a brief discussion. Finally, we will end with a section containing the conclusions and possible future research directions.

2 Viable System Model (VSM)

Beer (1979, 1981, 1985) presented the VSM (Figure 1) as a management tool that helps us to understand the complexity of organizations, allowing us to place them in a simple, decentralized and participatory structure. The VSM indicates the necessary and sufficient conditions for an organization to be viable in a scientific manner. In other words, it ensures an organization's capabilities for an independent existence, self-regulation, learning, adaptation and evolution, which are all needed to guarantee its survival in the face of changes that may occur in its environment (even though they may not have been foreseen) over time. More specifically, he identified as essential in every organization, the presence and proper functioning of a series of functions or subsystems (System One, System Two, System Three, System Four and System Five) and a series of communication relations either between these functions or subsystems or between them and the environment in which they operate.

System One represents the productive processes that enable the organization to generate its products or services. System One can be divided into a series of independent operational elements (management units) that interact between each other. The remaining Systems, from Two to Five, have the mission of supporting System One.

System Two has the principal function of damping the oscillations that occur between the operational elements of System One and of promoting an ordered transference of information from those operational elements to System Three (also in the reverse direction). Therefore, System Two is a support system for System Three that seeks to minimize its intervention on System One, maximizing automatization of the functioning of System One through the design of appropriate coordination systems.

System Three is concerned with the internal and short-term activities of the system. Its mission is to transmit information, goals, instructions and guidelines coming from Systems

Four and Five to the operational elements that compose System One; to negotiate resources and to report on their utilization with those operational elements; and eventually (only if the coherence of the organization as a whole is endangered) to intervene in those cases in which the coordination (System Two) has been unable to resolve the conflicts between the different operational elements. In addition, it is in charge of identifying synergies between operational elements and of contributing to an integral approach between them, seeking to achieve a global optimum of the organization and its internal stability. System Three* is another supporting element of System Three, providing information directly from the operational elements in a nonroutine manner, ensuring that it is not filtered when following the usual flow of information, thereby auditing its functioning. Through System Three*, System Three obtains immediate information on what is happening in the operational elements without having to trust the information that they send.

System Four is concerned with the external and middle-large term activities of the system, monitoring what is happening in the environment and exploring different future scenarios on a continuous basis to help take decisions that increase the probability of achieving future goals. It offers possible recommendations for future actions in accordance with the changes observed in the organizational environment with the aim of keeping the organization in a constant state of preparedness for change, thereby ensuring its adaptation to those changes.

System Five controls the interaction between System Three and System Four, preserving, in these interactions, the identity of the organization and taking charge; moreover, of defining the identity, the mission, the style, the ideological and normative aspects and the general principles and objectives of the organization. It ensures the organization adapts itself to the environment while maintaining an acceptable degree of internal stability, balancing the internal and external needs of the organization, which in many cases, are contradictory.

Both the five systems and the organization's environment are connected through homeostatic regulators, which are responsible for achieving a continuous balance in the interaction of the connecting elements. This balance is achieved when the information (in quantity, content and format) is adequate, i.e., when the relation works in a desired way, dynamically and harmonically. Eight elements intervene in each homeostatic regulator. The transmitter; the "transducer", that encodes the information coming out of the transmitter to be transported; the communication channel, that must be able to drive the amount of information per unit of time required; again the "transducer", that decodes the information from the communication channel and converts it to the appropriate format so that it can be interpreted by the receiver; and the receiver itself. In the opposite direction, the receiver and the transmitter switch their roles, rediscovering the same elements but in a reverse way.

Among all homeostatic regulators, there is one of special importance to what we call the alarm channel or the algedonic channel that connects the operational elements of System One directly with System Five. This alarm channel has the function of alerting System Five of any circumstance that has occurred in System One which has not been resolved by either this system or System Two and System Three and which may cause serious risk to the viability or function of the organization.

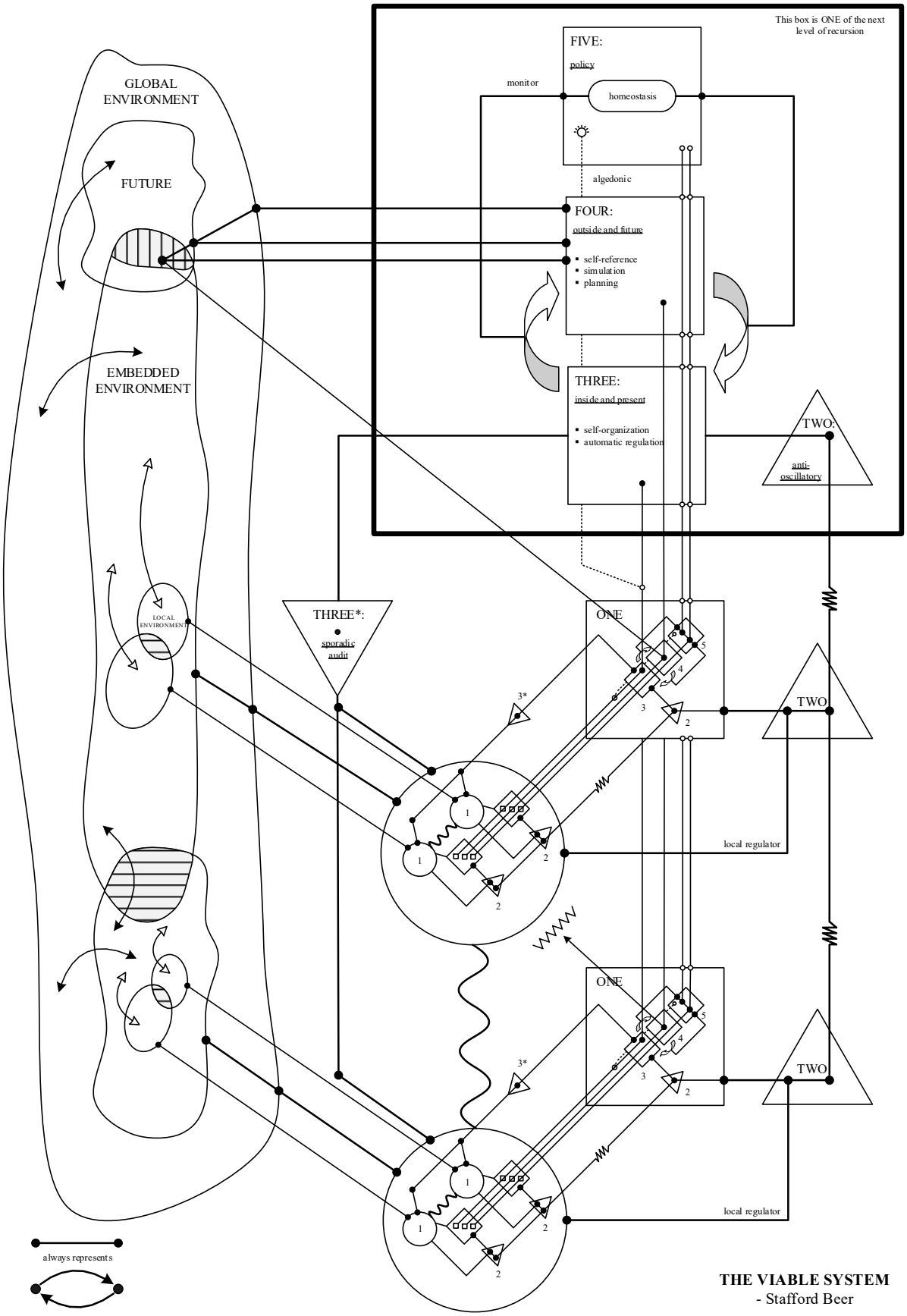

Figure 1: Viable System Model. Beer (1985) p. 136.

Another essential aspect of the VSM is its recursive character (Beer, 1984; Schwaninger, 2009). In a recursive organizational structure, each viable system contains viable systems and, in turn, forms part of systems that are also viable. The direct consequence of this recursiveness is that all the operational elements of System One replicate, in structural terms, the total in which they are contained, i.e., the viability of the system requires that all five functions exist recursively at all levels of the organization. Figure 1 shows that each operational element of System One contains an exact replica of the general structure of the system (rotated forty-five degrees). Thus, science offers a natural invariant that provides enormous savings on analysis, diagnosis and calculation, reinforcing viability, making it possible by offering a uniform description and representing an excellent solution to complexity problems for the managers.

Any insufficiency, whether due to the lack of any level of recursion or any system, the malfunctioning of existing systems or a lack or deficient homeostatic regulator design, will lead to a corresponding organizational pathology (structural pathology, functional pathology, informational pathology), which will make the organization malfunction or even disappear (Beer, 1984). Identifying a pathology is a prerequisite to prescribing any treatment for the diagnosed deficiency (Hetzler, 2008).

Finally, we would like to emphasize that, despite its mathematical and logical conception, the VSM has not been free of criticism. The principal criticisms are as follows: (1) its purely theoretical design, abstract nature, scant formalization and the inexistence of clear procedures for its application complicate its empirical confirmation (Jackson and Flood, 1988; Anderton, 1989; Van der Zouwen, 1996; Nechansky, 2013; Orengo, 2018); (2) the analogy between the human brain and other organizations is questionable (Checkland, 1986; Jackson, 2000); and (3) its hierarchical layout, authoritarian nature and lack of flexibility minimize the importance

of individuals who form part of an organization in favor of its structural design (Ulrich, 1981; Jackson, 1986; Thomas, 2006).

Despite all the criticism, authors such as Beer (1983) himself, Jackson (1992), Malik (1993), Schwaninger (2000, 2006), Yolles (2006), Tsuchiya (2007), Leonard (2009) and Espejo and Reyes (2011) have encouraged us to trust the VSM. This credibility seems to be legitimate considering the extensive application of the VSM. Different studies have illustrated the breadth of the application of the VSM (Espejo, 1979, 1989, 1990; Beer, 1981; De Raadt, 1987; Espejo and Harnden, 1989; Espejo and Schwaninger, 1993; Kawalek and Wastell, 1999; Shaw *et al.*, 2004; Al-Mutairi *et al.*, 2005; Devine, 2005; Dodis *et al.*, 2005; Midgley, 2006; Nyström, 2006; Türke, 2006; Crisan-Tran, 2008; Herrmann *et al.*, 2008; Hoverstadt, 2008; Leonard, 2008; Chronéer & Mirijamdotter, 2009; Gmür *et al.*, 2010; Pfiffner, 2010; Chan, 2011; Khosrowjerdi, 2011; Reissberg, 2011; Stephens and Haslett, 2011; Azadeh *et al.*, 2012; Burgess and Wake, 2012; Alqurashi *et al.*, 2013; Brecher, 2013; Dhillon *et al.*, 2013; Dominici and Palumbo, 2013; Espinosa and Walker, 2013; Hildbrand and Bodhanya, 2013, 2015; Preece *et al.*, 2013; Fitch *et al.*, 2014; Zadeh *et al.*, 2014; Alinaghian *et al.*, 2015; Stich and Groten, 2015; Tavella and Papadopoulos, 2015; Schwaninger and Scheef, 2016; Terra *et al.*, 2016; Hart and Paucar-Caceres, 2017; Toledo-Parra *et al.*, 2017; Toprak and Torlak, 2018). Although these are just a few examples, we can also find studies on project management in general (Britton and Parker, 1993; Morales Arroyo *et al.*, 2012) and of software project management in particular (Murad and Cavana, 2012; Kummamuru and Hussaini, 2015; Bathallath *et al.*, 2016; 2019). Therefore, we think that all these applications validate the VSM as a very useful tool for designing and diagnosing viable systems.

3 Research Methodology

The research methodology is based on the design of a structured questionnaire (available upon request) that contains direct questions. The questions were adapted to the field of software projects from similar studies that apply the VSM over other fields of study (Crisan-Tran 2008, De Raadt, 1987). Open questions were used to obtain additional information about the software projects (qualitative variables), and closed questions (Likert scale) were used to obtain information related to the success of the software project and related to the existence and degree of compliance of the main components of the VSM (quantitative variables). In this study, only the variables of the second type, the quantitative variables, were used.

The sampling method selected was convenience sampling. The sample was defined from the companies that were part of the Association of Information Technology, Communications and Electronics Companies of Castilla y Leon (AETICAL, for its Spanish acronym); specifically, the companies that had reached a maturity level of at least 3 in accordance with the Capability Maturity Model Integration (CMMI). The 40 companies that met this requirement were selected. Each of these 40 companies was sent a cover letter explaining the purpose of the study and were subsequently contacted through a telephone call. Among the 40 companies selected, only 2 declined to participate, reaching an acceptance level of 95%. We consider this acceptance rate especially deserving, taking into account the confidentiality and complexity of the topic, which required a high level of detail and in-depth information on each of the companies. Therefore, we finally gathered a total of 38 different companies.

To validate the reliability (absence of random errors) and the validity (absence of biases) of the questionnaire, a pilot test was carried out with the person responsible for R&D of one of the selected companies. Later, the questionnaire was used to gather information on the 38 software projects through personal interviews, voice to voice, collecting the answers of the respondents in writing with their respective project managers (approximate duration of 75

minutes). The last completed software project was analyzed in each of the 38 companies finally selected. Data were collected over a period of approximately 6 months. With the information obtained from these 38 software projects, a descriptive analysis was carried out based on the structural concepts of the VSM.

4 Results

In this section, we present a descriptive statistical analysis of the answers offered by the respondents. In general, this analysis shows that most software projects have the five structural components of the VSM. Next, we see in greater detail the results obtained.

4.1 Success/Viability

To analyze the success or viability of a software project, we have reviewed the adjustment with the budget compliance, the adjustment with the delivery time compliance and the adjustment with the compliance with the established requirements. The quality of the initial estimates made of these three factors and the overall success of the software projects have also been reviewed (Figure 2).

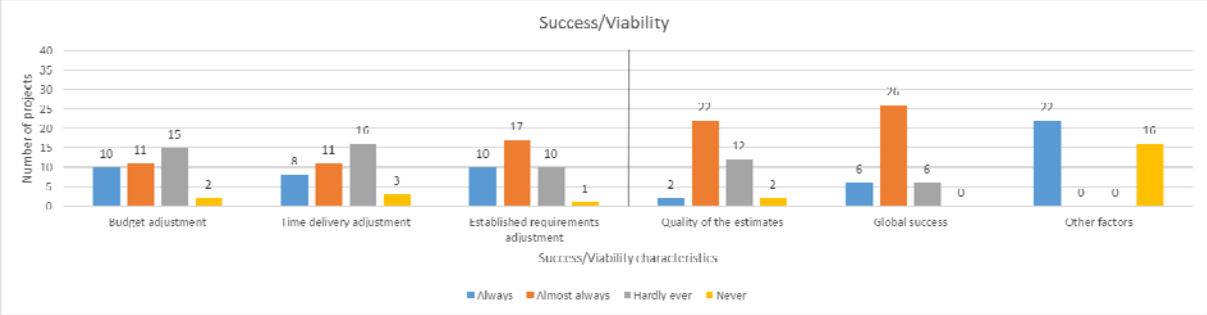

Figure 2: Diagnosis of software projects Success/Viability.

55% of the software projects surveyed consider that there is always, or almost always, an adequate adjustment between the initially budgeted cost and the final cost. 50% consider that always, or almost always, the adjustment between the initially planned delivery time and the final delivery time is adequate, and 71% consider that always, or almost always, the

adjustment between the initially established requirements and the final functionality of the software project is adequate.

We detect that 63% of the software projects consider that the quality of the initial estimates of cost, time and requirements is always or almost always adequate. 84% of the software projects surveyed consider always, or almost always, there was a global success of said software projects; and finally, we found that 58% of the software projects claim to use other factors, in addition to those indicated, to determine the success of the software project.

4.2 System One

As seen in Figure 3, 100% of the software projects have clearly differentiated the operational elements (organizational units, work environments, operating units, work teams, etc.) and the role that each one plays within the software project. Next, we examine the quality of the relationship between System One (operational elements) and System Three (project manager). In this sense, 79% of the surveyed software projects have high participation of their operational elements when each of these agrees on the objectives with System Three. In 84% of the software projects, the operational elements have clear operating guidelines, norms, policies and lines of action set by System Three; that is, the legal and corporate requirements are clearly defined, reinforcing the cohesion of the software project. Practically, in all software projects, the resource bargaining is completely clear, i.e., the resources that System Three assigns to each operational element and what activities each of them must develop are clear. 89% of the software projects present methods that allow System Three to confirm that operational elements do what they are asked (accountability). Of this 89%, 3% use these methods only a few times, the rest always, or almost always, use them.

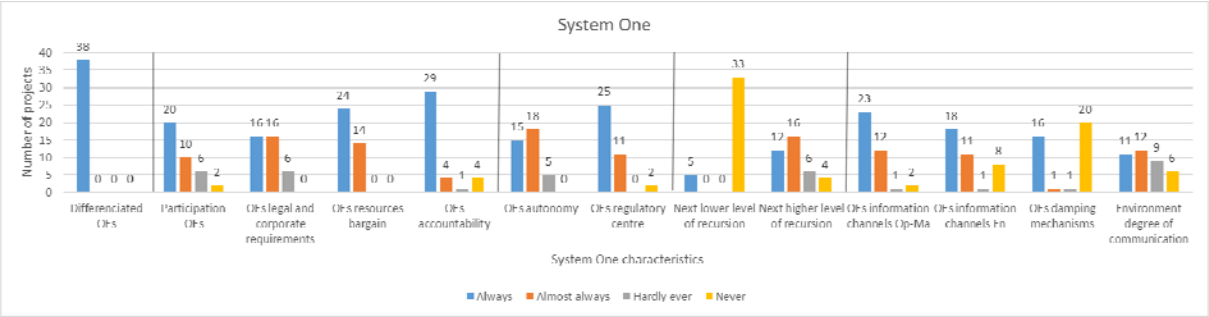

Figure 3: Diagnosis of software projects System One.

We also examine the autonomy of the operational elements (System One approach). 87% of the software projects consider their operational elements to be autonomous, because they are directed by someone and interact with a client or part of the specific client, so that each one of them is responsible for its own success or failure. In addition, in 95%, each operational element has a regulatory center in which detailed plans, programs, procedures, processes, etc. are defined. Of this 95%, 66% always use this regulatory center and 29% almost always use the regulatory center. The fact that the operational elements are self-coordinating through their regulatory centers, empowers their autonomy. Information has also been obtained on the relationship with the next lower and next higher level of recursion. Regarding the next lower level of recursion, only 13% of the software projects surveyed have their operational elements organized into smaller units with some level of autonomy. In regards to the next higher level of recursion, 74% of software projects have substantial relationship with other projects in the company, while 26% have little or no relationship. Finally, in relation to the internal functioning of System One, in 95% of the software projects, the operational elements have the appropriate information channels between their operations and their management; and of these, 92% of the software projects use them, always, or almost always. In 79%, the operational elements have the appropriate information channels to obtain and offer their opinion to their environment (client). In 76% of the cases, these channels are always, or almost always, used. In 50%, the operational elements have damping mechanisms that allow a cushion for the mismatches between them caused by work (for example, contingency plans).

For this 50%, the damping mechanisms are always, or almost always, used 44% of the time. 39% of the software projects consider that the degree of communication between the different parts of the environment related to each operational element is of little use or not adequate.

4.3 System Two

100% of the software projects surveyed present routine mechanisms aimed at facilitating the coordination of the activities carried out by the different operational elements (Figure 4). Regarding their use and internal functioning, 97% of software projects always, or almost always, use these mechanisms. 34% of the software projects indicate that the operational elements never, or hardly ever, participate in the design of the indicated coordination mechanisms together with System Three. Finally, 82% of software projects show adequate flexibility or adaptability of the coordination mechanisms that make up System Two.

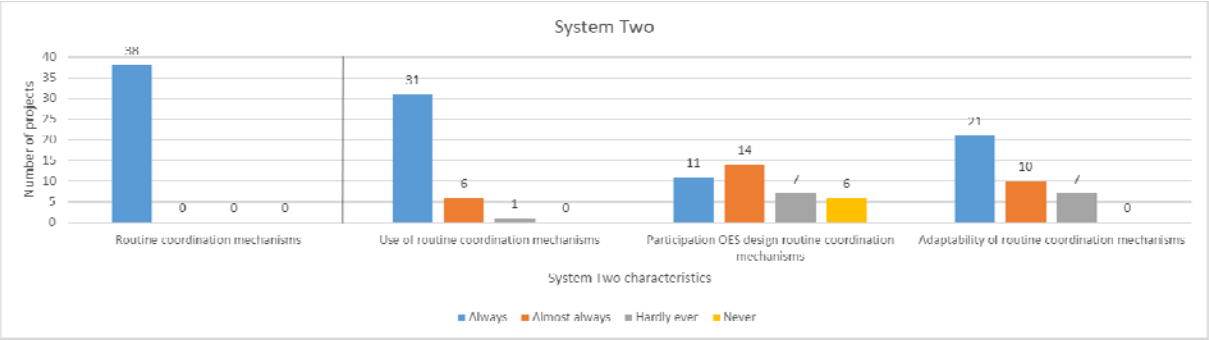

Figure 4: Diagnosis of software projects System Two.

4.4 System Three

The existence of the project manager function (System Three) reaches a value of 100% in the surveyed software projects; therefore, all the project managers are clear that they are in charge of having a global, integrated and coherent vision of all the operational elements throughout the development of the software project. In the same way, 100% of the project managers are clear that the relationship between the different operational elements is a dynamic relationship, that is to say, it varies over time (Figure 5).

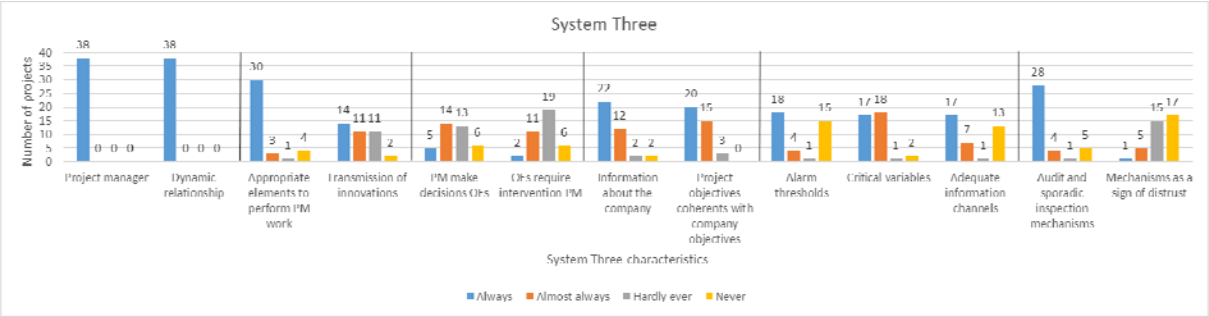

Figure 5: Diagnosis of software projects System Three.

Regarding the internal functioning of System Three, that is, with respect to whether the project manager has the appropriate elements to perform his/her work effectively in the search for synergies between the operational elements; in 90% of the software projects surveyed, the project manager disposes of these elements. Of this percentage, 87% of the time, the elements are always, or almost always, used. In addition, in 66% (always or almost always) of the software projects, the project manager transmits the technological innovations or any other type of innovation for the operational elements.

In relation to the autonomy of the operational elements (System Three approach), in 50% (always or almost always) of the software projects, the project manager makes decisions or performs activities that correspond to the operational elements, developing the functions of System One. In the same sense, in 34% of the software projects, the operational elements always, or almost always, require intervention by the project manager to function harmoniously.

The data obtained on the relationship between the software projects and the next higher level of recursion indicates that in 97% of the software projects, the project manager has sufficient general information about the company. This information is taken into account 89% (always or almost always) of the 97% total. Moreover, in 92% of software projects, their objectives are coherent (always or almost always) with the general objectives of the company.

In regards to the communication between System Three and System One that aims to control the objectives of the software project (static balanced scorecard), in 61% of the software projects surveyed, the project manager specified alarm thresholds and an information channel in such a way that he/she is alerted of possible dangers from the operational elements in case these thresholds are exceeded. In the case that these thresholds exist, 58% of the occasions are always, or almost always, used. In 92% of cases, the project manager always, or almost always, has clear critical variables that must be controlled in order to ensure compliance with the objectives of the software project. In 66% of the software projects, the project manager has an adequate information channel that indicates the behavior of these critical variables and the discrepancies with the objectives. Of these 66%, in 63% of the occasions, project managers said the information channel is used (always or almost always) and as a consequence take into account the information it provides.

Finally, this section also considers the existence and internal functioning of System Three*. 87% of the software projects present audit and sporadic inspection mechanisms about what happens in the software project and therefore, they inform the project manager about the operational element's behavior. Of the 87%, in 84% of the cases, the results of these mechanisms are taken into account (always or almost always) to modify the behavior of the software project (operational elements). Furthermore, of said 87%, in 16% of the software projects, the operational elements perceive these mechanisms as a sign of distrust (always or almost always) towards them rather than as a service in favor of the development of the software project itself.

4.5 System Four

The existence and internal functioning of System Four is revised, that is, the adaptability of a software project (Figure 6) is revised. 84% of the project managers surveyed consider that their software projects are adaptable and flexible (always or almost always) in response to

changes that occur in the environment. This allows us to detect the dissonance between what the project manager thinks and the structure actually created. 82% of software projects have defined a function responsible for monitoring the software project's environment and transmitting the corresponding information to the interior of said software project so that it can be adapted (System Four). Of this 82%, in 66% of the cases, the monitoring function is always, or almost always, used. In addition, of the same 82%, 76% of those responsible for the monitoring function consider that it is always, or almost always, necessary to explore the future evolution of the environment. Of the 82% of software projects that have monitoring functions, 22% have exploration mechanisms that allow their managers to evaluate the possible action alternatives in different future scenarios for the software project. When they do exist, 19% of the occasions are always, or almost always, used. Finally, of the 22% of the software projects that have these mechanisms, in 71% of the occasions (always or almost always, i.e., all or many variables), such mechanisms allow analyzing the evolution over time of the variables relevant to the software project.

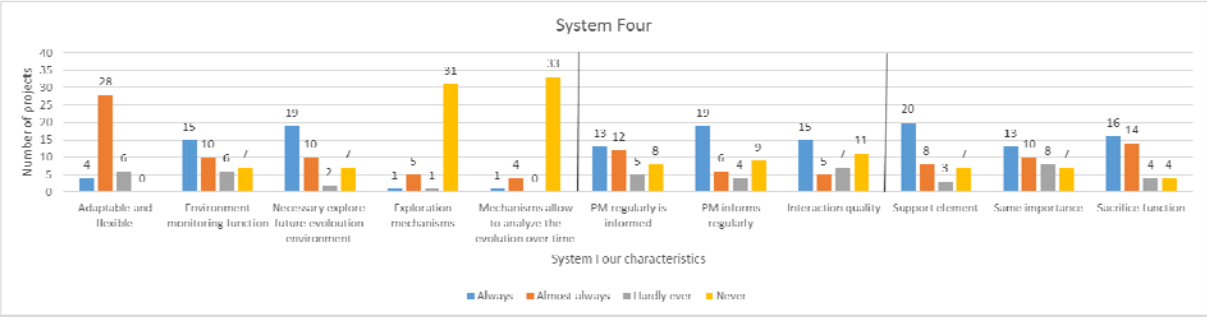

Figure 6: Diagnosis of software projects System Four.

Second, the relationship between System Three and System Four is checked. For this, we start with 82% of software projects that present System Four. Of this 82%, in 97% of the cases, the project manager is regularly informed of possible changes in the environment so that he/she is aware of the impact they may have on the software project. Of the 97%, in 81% of the occasions the project manager always, or almost always, uses this information to prepare his

software project (operational elements) for said changes, that is, he applies the results obtained in System Four. In this way, the relationship between System Four and System Three is represented. Conversely, of that 82%, in 93% of the cases, the project manager regularly informs System Four about the current status of the software project (relationship between System Three and System Four). Of the 93%, in 86% of the occasions System Four always, or almost always, uses that information to carry out the inspection of the environment and its possible evolution. To check the quality of this interaction between System Three and System Four in both directions, of the 82% of the software projects that present System Four, 87% of them have tools that effectively facilitate an interaction. Of this 87%, in 73% of the occasions, the tools are always or almost always used.

Finally, we analyze the importance given to System Four within a software project. We continue, considering 82% of the software projects that present System Four. Of these, in 90% the project manager always, or almost always, knows clearly that System Four is a support element instead of a problem, constituting the organ of adaptation of the software project, insofar as System Four is the knower of the future circumstances that will affect the software project itself (assessment made by System Three of System Four). Of the 82%, 42% of the surveyed software projects consider that System Three (project manager) always has the same importance as System Four. Of the remaining 58%, all software projects consider the importance of System Three is greater than that of System Four (32% slightly more important and 26% much more important). Of the 82% of the software projects with System Four, 74% sacrifice always, or almost always, the functions of System Four to reinforce the execution of the software project (System Three) against the demands of time, cost and compliance of requirements.

4.6 System Five

In this section, we will review the existence and internal functioning of System Five and the existing relationship between System Five of a software project with System Five of the next higher level of recursion (Figure 7). In addition, we detect the existence of necessary information in System Five from the rest of the systems, including the operational elements. In the first place, 79% of the software projects surveyed indicate that they present elements in charge of defining their identity, mission, ethical code, goals, philosophy, intention, corporate principles, etc. (System Five). The rest of the information is evaluated from this 79%.

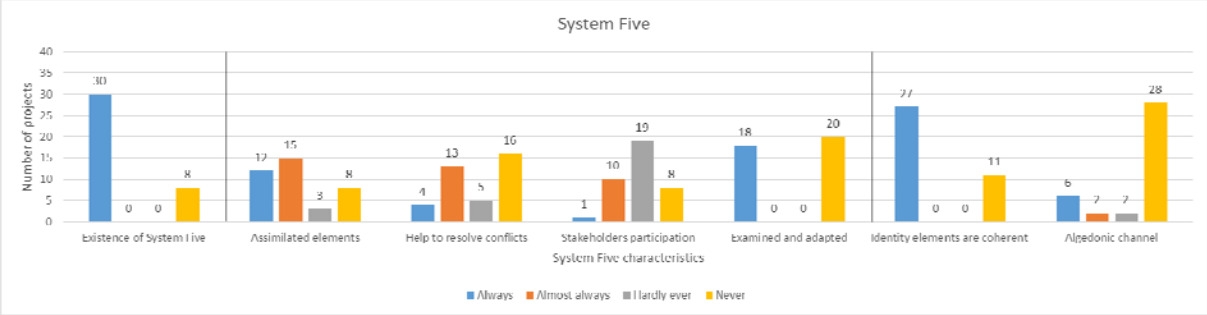

Figure 7: Diagnosis of software projects System Five.

In this sense, in 90% of cases, these elements are always, or almost always, assimilated by the members of the software project. In 57% of software projects, these elements always, or almost always, help to resolve conflicts that have not been resolved in any other way between System Three and System Four, and in 36% of software projects, always, or almost always, the people or entities related to said software projects (stakeholders) participate in the definition of System Five. Finally, in 60% of the cases, System Five is initially defined, is periodically examined and adapted if necessary (reflection on its own identity).

Regarding the existing relationship between System Five of a software project and System Five of the next higher level of recursion, of 79% of software projects with System Five, in 90% of cases it is considered that the identity elements are coherent (they are in harmony) with the identity elements of the company as a whole. Finally, of said 79%, only 33% present

an information channel, formal or informal, from the operational elements or from the rest of the systems to the persons or entities in charge of defining the identity elements (System Five) that alert them of any serious danger of failure of the software project (algedonic channel). Of the 33%, in 26% of the occasions this information channel is always, or almost always, used.

5 Discussion

Approximately, half of the software projects surveyed present a good fit between the initially estimated values of the three factors studied and the final values obtained for each of them. We highlight a slight improvement in the assessment of compliance with requirements. This may be due to the fact that the effort made to achieve compliance with the requirements causes an increase in costs and delivery times and, consequently, a minor adjustment with what was initially planned. At the same time, we would like to point out that, in approximately half of the cases, the initial estimates are deficient and that other factors are used to measure the success of a software project, and such deviations may be justified in these situations. In this sense, many authors have highlighted the importance of a good initial estimate of cost, delivery time and compliance with requirements as one of the key factors of the success of a software project. For example, Nasir and Sahibuddin (2011) indicate that the most important factor is to have clear requirements and specifications and Gheni et al. (2017) indicate that the most important factor is to have a committed and motivated team. In any case, both agree that if the estimates are good, one can expect to have a successful project with enough probability. In our particular case, and despite the percentage of bad estimates, project managers consider a high percentage of successful projects (84%) value very close to that obtained (87%) by other authors such as Mohagheghi and Jorgensen (2017) for agile development and flexible scope projects.

The operational elements are clearly defined from the beginning of the software project and maintain a high degree of autonomy, which favors their performance in dynamic

environments characterized by strong turbulence. Other works, such as the one carried out by Hoda and Murugesan (2016), seem to reinforce this trend of granting greater independence to the operational elements of a software project instead of following a traditional approach, i.e., following a command and control style of management. The justification that the operational elements do not have total autonomy is based on the fact that, in software projects, the project manager assumes, on many occasions, the development and control of the activities of the operational elements themselves (next lower level of recursion). A poor degree of communication between the different parts of the environment related to each operational element existing in some software projects may also have influence since it causes a greater dependence on the operational elements, with the project manager updating the requirements and specifications of the project; update requirements is one of the most important factors of success (Nasir and Sahibuddin, 2011). We also highlight that more than half of the software projects lack damping mechanisms that make it possible to cushion the mismatches that may appear between the different operational elements. This situation can be justified by a strong and sequential relationship between the operational elements that make up a software project, which to a large extent avoids the need to use mechanisms that cushion the mismatches that occur between them. Sequentiality forces each operational element to act after its predecessor and before its follower. A more detailed analysis of the importance of task dependency on the software project success can be seen in Mahmood et al. (2017) and Kuthyola et al. (2017). Furthermore, as mentioned above, this relationship can cause a poor degree of communication between their respective environments, so the interaction between the operational elements and their environment is carried out through System Four.

Software projects, in general, are robust, that is, there is a strong relationship between the operational elements, causing the appearance of few oscillations between them, even reaching, in some situations, a slight overlap of their environments. So a rigorous definition of

the regulatory centers in each operational element can cause System Two (predefined rules, standard behaviors, etc.) to be redundant, so the requirement of the existence of a strong coordination system or a strong System Two is not necessary. In addition, the existence of an unstable environment can cause an inflexible System Two to not absorb the oscillations of the operational elements caused by that instability. In this sense, the existence of a strong System Two is not only unnecessary but can even be counterproductive. In this same line of thought, other authors such as Molokken-Ostvold and Jorgensen (2005) point out that software projects which employ a flexible development model experience less effort overruns than do those which employ a sequential model. Alike, Yadav (2016) highlight that research and practice have accentuated the need to embrace flexibility in management practices, and propose a flexible management approach for globally distributed software projects. It would therefore be advisable to improve the participation of the operational elements when defining these mechanisms to increase their flexibility.

In respect to System Three, one of the main problems detected is the excessive intervention that the project manager makes in the operational elements when they make decisions, to develop their own activities and to make everyone work harmoniously resolving the conflicts that arise between them. In the same way that we discussed this in the previous paragraph, this situation may be caused by the unstable environment associated with software projects (Nasir and Sahibuddin, 2011). This variability in the environment means that System Two needs to provide a certain degree of flexibility and, as a consequence, obliges System Three to act on more occasions than recommended. In addition, we note that project managers, despite having clear the variables that must be controlled for the correct development of the software projects, often lack adequate information channels and alarm thresholds that facilitate their control. Hazir (2015) proposes an interesting review of different analytical models, approaches and decision support tools to monitor and control a software project. Gómez et al.

(2017) present an emplirically grounded ontology to support different strategic decision-making processes and extends the ontology to cover the context of managing quality in rapid software development projects. This ontology can serve to improve information channels and alarm thresholds that facilitate better monitoring and control of a software project. Last, the transmission of innovations from System Three to the operational elements can be improved, thus improving the relationship between the operational elements and the software project environment. Salerno et al., (2015) argue which configuration of innovation processes and resource allocations should be employed in a given project and the rationale behind the choice. In relation to software projects, Munir et al., (2015) suggest that firms assimilating knowledge (external innovation) into their internal R&D activities (internal innovation) have a higher likelihood of gaining financial advantages.

In most cases, the software projects surveyed present a function responsible for conducting a study of their environment, that is, responsible for adaptability and strategic planning. In contrast, the quality of this function can be improved since it is not equipped with adequate mechanisms to carry out this study in a systematic way. In addition, it suffers from the ability to control the relevant variables for the software project. These variables allow it to measure and react appropriately to the consequences caused by changes in the environment. The more variables that are measured, the better System Four will be, and as a consequence, the better the adaptability of the software project will be. Therefore, the adaptation of the system to changes that occur in the environment of a software project can at least be questioned. This is one of the main contributions of the study: the detection of the absence of exploration mechanisms that allow software project managers to evaluate the possible alternative actions in different future scenarios. In this sense, we consider that simulation is an adequate tool to answer questions that ask “what would happen if…?” and, as a consequence, improve the adaptability of software projects. Many authors have contributed ideas on the application of

simulations to improve the development of software projects. For example, De França and Travassos (2015) analyzed different works that relate to validity in the use of simulation-based studies in software engineering. Chaikovska (2017) used a system-dynamic to implement a discrete simulation model of the IT enterprise with different software projects. Jain et al., (2017) presented a simulation model for software maintainability prediction in favor of its adaptability. Finally, Tanir (2017) proposed the simulation as an invaluable technique for assessing complex multifaceted solution spaces early on during the planning and design phases in a cost-effective and timely manner without the need for physically deploying possible design alternatives in software projects. Despite the importance given to System Four within a software project, System Three is considered more important because of its ability to complete the software project successfully will be chosen over System Four.

Regarding System Five, it would be convenient to increase the number of stakeholders related to the software project in its definition and examine and adapt more frequently. In this way, its capacity to resolve conflicts between System Three and System Four could be increased. A certain lack is also detected when clearly differentiating the next lower level of recursion and the next higher level of recursion (already detected when discussing System One). Normally, some of the principles, policies, etc., defined in System Five of the software project are shared with those defined for the organization that develops this and other software projects. In the same way, in many cases there is no direct connection between the next lower level of recursion and System Five of the software project, preventing a clear separation between both levels. It is possible that this confusion when differentiating the levels of recursion is due to the fact that the project manager assumes, on many occasions, the development of the activities of the operational elements (next lower level of recursion) and in the same way, he develops activities of the organization (next higher level of recursion). Moreover, the functions of Systems Three, Four and Five, are often done by the same person, so the

delimitation of the associated functions with each system is not entirely clear. To deal with this situation, several authors have worked on identifying the main activities and competencies associated with a project manager. Regarding the activities, Hornstein (2015) presents an interesting literature review about the integration of project management and organizational change management, Hoda and Murugesan (2016) present a mapping between the emergent challenges and standard project management activities through a multilevel perspective and, Linares et al. (2018) show that project portfolio management present difficulties in their adoption in small contexts in the software industries. Regarding competencies, authors such as Takey and de Carvalho (2015) propose a mapping, evaluation and development process for the competence of project managers and practitioners in an organizational context and, Millhollan and Kaarst-Brown (2016) show a combination of skills that contribute to the effectiveness of a software project manager.

Finally, we would like to emphasize that the results obtained in the previous section seek to complement other studies conducted on the same dataset. In particular, in Puche Regaliza (2015) a quantitative study was carried out using Structural Equation Modeling (SEM) and tried to empirically confirm the hypothesis (H): the better the System One to System Five proposed by VSM are defined, the better they work, the better they cooperate, and the greater the success/viability of the software project. The results of the work highlight that the hypothesis can be confirmed for System One and for System Four, with the latter having the greatest influence on the success of a software project. That is, software projects need to clearly define their operational elements (organizational units, business units, working environments, working teams, etc.) and the relationships that appear between them. Additionally, in software projects, it is necessary to find the appropriate preventative actions to observe the changes that take place in the environment and thereby make decisions that allow the projects to adapt to these changes.

In Puche Regaliza *et al.* (2017) the main success factors of a software project structured upon the basis of the VSM are identified. The results of the study indicate that the most influential factors in achieving global success in a software project are the local environment, the organizational units, and the intelligent system. Moreover, the work develops a mathematical prediction model to estimate the probability of success of a software project based on values that takes into account the aforementioned influential structural factors and reaches an accuracy of 63.16% in its predictions. This accuracy rate may be considered acceptable, especially in view of the exploratory nature of the study and the number of projects that were surveyed. The same process was followed considering the degree of success with regard to cost compliance, with regard to compliance with deadlines and with regard to compliance with requirements. The project manager was the most influential factor for the first case; the project manager and the coordinating mechanisms, for the second; and the local environment and the organizational units for the third.

6 Conclusions

The development of projects that stay within the initially budgeted costs, meet the initially estimated deadline, and comply in a sufficient manner with the established requirements, is one of the principal challenges pursued in the field of project management, in general, and in the field of software projects, in particular. The successful achievement of these objectives may be conditioned by compliance with a series of factors that have been widely studied in the literature. Even so, the increase in the percentage of successful software projects is in no way sufficient. There is therefore a need to develop new contributions that allow the success rate to increase.

We believe that this study's contribution may be related to the organizational structure of the software projects, in such a way that all the elements involved in the software projects to be linked as a whole, including their environment. In the same vein, we think that system

thinking is an appropriate framework to visualize a software project from this perspective, and more particularly, we support the application of the Viable System Model as a tool to diagnose or to design the organizational structure of software projects in pursuit of their viability. Ensuring viability is a way of contributing to increasing the rate of success of these sorts of projects. In this way, structuring a software project based on the Viable System Model allows us to relate all its components in a scientific and systemic way, thus offering a notable difference with the classic structure of a software project.

To make a diagnosis of the current situation of software projects in regards to the Viable System Model, this work presents a descriptive analysis of 38 software projects of companies in the ICT sector. This analysis detects the weak points in the organizational structure of the software projects. Improving these weaknesses can contribute to improving the percentage of successful projects. Managers that apply the Viable System Model to their software projects and ensure the existence and proper functioning of the five systems will manage viable software projects and as a consequence will help to improve the number of successful projects.

The main conclusions that are drawn from the study are: (1) a next lower level of recursion is not usually defined, so the operational elements of a software project are not organized into smaller units with some level of autonomy, (2) the operational elements do not always have damping mechanisms that allow the cushioning of mismatches between them caused by work (for example, contingency plans), (3) many times, the project manager makes decisions or performs activities that correspond to the operational elements or the operational elements require intervention by the project manager to function harmoniously, (4) rarely does the project manager specify alarm thresholds and an information channel in such a way that he/she is alerted to possible dangers from the operational elements in case these thresholds are

exceeded, (5) an information channel, to the project managers, that indicates the behavior of critical variables and the discrepancies with the objectives is rarely used, and as a consequence the information it provides is not taken into account, (6) very few software projects have exploration mechanisms that allow their managers to evaluate the possible action alternatives in different future scenarios, (7) the identity elements of a software project are not periodically examined and adapted as necessary, and normally they do not help to resolve conflicts that have not been resolved in any other way between System Three and System Four and, (8) in a few occasions, the software projects present some information channel (algedonic channel), formal or informal, from the operational elements or from the rest of the systems to the persons or entities in charge of defining the identity elements that alert them of any serious danger of failure of the software project.

The detection of these 8 weaknesses of the software projects is the main contribution of this work, so that the managers of software projects that aim to increase the success or viability of the software projects they manage, should place greater emphasis on improving these weaknesses. We also emphasize, through these weaknesses, the practical application of the work, establishing the characteristics that a software project must contain in order to be successful. Thus, a manager of a software project must review the correct fulfillment of the indicators defined by the Viable System Model and avoid the possible pathologies that can produce a deficit and/or the defective behavior of its components. A detailed practical implementation of the Viable System Model in a software project can be seen in Puche Regaliza (2014).

We provide to project managers a detailed reference for the diagnosis or design of the organizational structure of a software project with defined viability characteristics, facilitating the detection of absence, or unsatisfactory development of, any component, or a negative

cooperation between them that may deteriorate or endanger the viability of the software project. Possible improvements of the detected deficiencies aim to reduce the number of failed software projects. In addition, we encourage the practical use of the Viable System Model, contributing to its better understanding and greater formalization, thus lessening its main criticisms related to its abstraction and limited applicability, and increasing its rigor and validity as a tool for the diagnosis and design of viable organizations.

In closing we must say that, although the results of this study are quite promising, we must also interpret them with caution and from an exploratory stance due to the small number of projects employed. In addition, we want to highlight, at this point, one of the main limitations of the work, since it has been a great effort to interview 38 companies due to the extension of the interview time (approximately 75 minutes) and the confidentiality and complexity of the topic, which required a high level of detail and in-depth information on each of the software projects and companies interviewed.

To reduce this level of caution, give the study a confirmatory nature, and validate and generalize the results set forth here, the pursuit of further research is recommended. In this regard, to maintain the scientific progress that leads to the evolution and improvement of this work, we present a series of possible future research directions.

We must first consider carrying out the same work completed for the software project level, for the next lower level of recursion, i.e., for each operational element that comprises System One. This will allow improvements to the detected shortcomings related to the next lower level of recursion. Second, we consider developing better simulation models that allow us to improve another detected deficiency, specifically, the exploration mechanisms of a software project used to evaluate the possible action alternatives in different future scenarios.

With these conclusions and possible future directions of research, we seek to show that the Viable System Model is an extremely useful tool for the management of software projects. We therefore suggest that its knowledge would be of incalculable value for managers wishing to manage software projects successfully and survive in such a complex and rapidly changing environment. Its application allows us to diagnose and detect critical characteristics to achieve success for software projects.

References

Ahimbisibwe, A., Cavana, R.Y. and Daellenbach, U. (2015), "A contingency fit model of critical success factors for software development projects: A comparison of agile and traditional plan-based methodologies", *Journal of Enterprise Information Management*, Vol. 28 No. 1, pp. 7-33.

Ahimbisibwe, A., Daellenbach, U. and Cavana, R.Y. (2017), "Empirical comparision of traditional plan-based and agile methodologies: Critical success factors for outsourced software development projects from vendors' perspective", *Journal of Enterprise Information Management*, Vol. 30 No. 3, pp. 400-453.

Al-Mutairi, S.G., Burns, N.D. and Backhouse, C.J. (2005), "Using a viable system model as a diagnostic tool for small-sized companies", *International Journal of Services and Operations Management*, Vol. 1 No. 3, pp. 220-238.

Albert, M., Balve, P. and Spang, K. (2017), "Evaluation of project success: a structured literature review", *International Journal of Managing Projects in Business*, Vol. 10 No. 4, pp. 796-821.

Alinaghian, R., Ramezani, M., Rahman, A.A. and Ibraim, R. (2015), "Viable diagnosis of ICT policy management in the context of higher education", *International Journal of Management & Information Technology*, Vol. 10 No. 7, pp. 2360-2371.

Alqurashi, E., Wills, G. and Gilbert, L. (2013), "A viable system model for information security governance: establishing a baseline of the current information security operations systems", in: Janczewski, L.J., Wolfe, H.B. and Shenoi, S. (Eds.), *SEC 2013, IFIP AICT*, Vol. 405, Springer, Heidelberg, pp. 245-256.

Anderton, R. (1989), "The need for formal development of the VSM", in: Espejo, R. and Harnden, R. (Eds.), *The Viable System Model. Interpretations and Aplications of Stafford Beer's VSM,* John Wiley & Sons, Chichester, pp. 39-50.

Azadeh, A., Darivandi, K. and Fathi, E. (2012), "Diagnosing, simulating and improving business process using cybernetic laws and the viable system model: the case of a purchasing process", *Systems Research and Behavioral Science*, Vol. 29 No. 1, pp. 66-86.

Bathallath, S., Smedberg, Å. and Kjellin, H. (2016), "Project interdependency management in IT/IS project portfolios: from a systems perspective", *Procedia Computer Science*, Vol. 100, pp. 928-934.

Bathallath, S., Smedberg, Å. and Kjellin, H. (2019), "The Viable System Model for Diagnosing and Handling IT-Project Interdependences in Large Portfolios", *International Journal of Information Technology Project Management*, Vol. 10 No. 1, pp. 72-87.

Beer, S. (1959), *Cybernetics and Management*, English Universities Press, London.

Beer, S. (1979), *The Heart of Enterprise*, John Wiley & Sons, Chichester.

Beer, S. (1981), *Brain of the firm, 2nd edition*, John Wiley & Sons, Chichester.

Beer, S. (1983), "A Reply to Ulrich's Critique of Pure Cybernetic Reason: The Chilean Experience with Cybernetics", *Journal of Applied Systems Analysis*, Vol. 10, pp. 115-119.

Beer, S. (1984), "The Viable Systems Model: Its Provenance, Development, Methodology and Pathology", *Journal of the Operational Research Society*, Vol. 35 No. 1, pp. 7-25.

Beer, S. (1985), *Diagnosing the System for Organizations*, John Wiley & Sons, Chichester.

Beer, S. (1995), *Platform for Change*, John Wiley & Sons, London.

Brecher, C., Müller, S., Breitbach, T. and Lohse, W. (2013), "Viable system model for manufacturing execution systems", *Procedia, CIRP*, Vol. 7, pp. 461-466.

Britton, G.A. and Parker, J. (1993), "An explication of the Viable System Model for project management", *System Practice*, Vol. 6 No. 1, pp. 21-52.

Bronte-Stewart, M. (2015). "Beyond the Iron Triangle: Evaluating Aspects of Success and Failure using a Project Status Model", *Computing and Information Systems Journal*, Vol. 19 No. 2, pp. 19-36.

Burgess, N. and Wake, N. (2012), "The applicability of the Viable Systems Model as a diagnostic for small to medium sized enterprises", *International Journal of Productivity and Performance Management*, Vol. 62 No. 1, pp. 29-46.

Chaikovska, M. (2017), "Metodological bases of IT project management with simulation modelling tools", *Scientific Journal of Polonia University*, Vol. 21 No. 2, pp. 55-66.

Chan, J.W. (2011), "Enhancing organizational resilience: application of viable system model and MCDA in a small Hong Kong company", *International Journal of Production Research*, Vol. 49 No. 18, pp. 5545-5563.

Checkland, P. (1986), "Review of Diagnosing the System for Organizations", *European Journal of Operational Research*, Vol. 23, pp. 269-270.

Chronéer, D. and Mirijamdotter, A. (2009), "Systems thinking benefits in supply change management: an illustration of the Viable Systems Model in a Supply Chain", *International Journal Intelligent Systems Technologies and Applications*, Vol. 6 No. 3/4, pp. 227-248.

Crisan-Tran, C.I. (2008), "Assessing the Viable System Model: an empirical test of the viability-hypothesis", *International Journal of Applied Systemic Studies*, Vol. 2 No. 1/2, pp. 66-81.

De França, B.B.N. and Travassos, G.H. (2015), "Simulation Based Studies in Software Engineering: A matter of validity", *CLEI Electronic Journal*, Vol. 18 No. 1.

De Raadt, J.D.R. (1987), "The implications of Beer's viable system model for organisational adaptation: a study in an insurance organization", *General Systems*, Vol. 30, pp. 9-13.

Devine, S. (2005), "The viable systems model applied to a national system of innovation to inform policy development", *Systemic Practice and Action Research*, Vol. 18 No. 5, pp. 491-517.

Dhillon, S.K., Ibrahim, R., Selamat, A. and Sani, S.I. (2013), "Diagnosis of Key Performance Indicatros delivery process using Viable System Model", *International Journal of Digital Content Technology and its applications*, Vol. 7 No. 13, pp. 64-85.

Dodis, C., Kitis, K. and Panagiotakopoulos, D. (2005), "Organizational Cybernetics For Waste Management Authorities: A Case Study", available at: http://www.bvsde.paho.org/bvsacd/iswa2005/case.pdf (accessed 30 August 2017).

Dominici, G. and Palumbo, F. (2013), "Decoding the Japanese Lean Production System according to a Viable Systems Perspective", *Systemic Practice and Action Research*, Vol. 26 No. (2), pp. 153-171.

Espejo, R. (1979), "Information and Management: The Cybernetics of a Small Company", *Journal Management Research New*, Vol. 2 No. 4, pp. 2-15.

Espejo, R. (1989), "P.M. Manufacturers: the VSM as a diagnostic tool", in: Espejo, R. and Harnden, R. (Eds.), *The Viable System Model: Interpretation and Applications of Stafford Beer's VSM*, John Wiley & Sons, Chichester, pp. 103-120.

Espejo, R. (1990), "The Viable System Model", *Systemic Practice and Action Research*, Vol. 3 No. 3, pp. 219-221.

Espejo, R. and Harnden, R. (1989), *The Viable System Model. Interpretations and Applications of Stafford Beer's VSM*, John Wiley & Sons, Chichester.

Espejo, R. and Reyes, A. (2011), *Organizational systems: Managing complexity with the viable system model*, Springer Science & Business Media, Heidelberg.

Espejo, R. and Schwaninger, M. (1993), *Organisational Fitness: Corporate Effectiveness through Management Cybernetics*, Campus, Frankfurt.

Espinosa, A. and Walker, J. (2013), "Complexity management in practice: A Viable System Model intervention in an Irish eco-community", *European Journal of Operational Research*, Vol. 225 No. 1, pp. 118-129.

Eveleens, J.L. and Verhoef, C. (2010), "The Rise and Fall of the Chaos Report Figures", *IEEE Software*, Vol. 27 No. 1, pp. 30-36.

Fitch, D., Parker-Barua, L. and Watt, J.W. (2014), "Envisioning public child welfare agencies as learning organizations: Applying Beer's Viable System Model to Title IV-E program evaluation", *Journal of Public Child Welfare*, Vol. 8 No. 2, pp. 119-142.

Gandhi, R., Jadhav, A. and Chougule, A. (2018), "Success Factors of Project Management", *International Research Journal of Engeneering and Technology*, Vol. 5 No. 9, pp. 40-44.

Gheni, A.Y., Jusoh, Y.Y., Jabar, M. A. and Ali, N.M. (2017), "The critical success factors (CSFs) for IT projects", *Journal of Telecommunication, Electronic and Computer Engineering*, Vol. 9 No. 3-3, pp. 13-17.

Gmür, B., Barlet, A. and Kissling, R. (2010), "Organization from a systemic perspective: application of the viable system model to the Swiss youth hostel association", *Kybernetes*, Vol. 39 No. 9/10, pp. 1627-1644.

Gómez, C., Ayala, C.P., Franch, X., López, L., Behutiye, W. and Martínez-Fernández, S. (2017), "Towards an ontology for strategic decision making: the case of quality in rapid software development projects", paper presented at the International workshop on Conceptual Modeling in Requirements and Bisiness Analysis, 6-9 November, Valencia, Spain, available at: http://www.essi.upc.edu/~smartinez/wp-content/papercite-data/pdf/gomez2017ontology.pdf (accessed 15 February 2019).

Hart, D. and Paucar-Caceres, A. (2017), "A utilisation focused and viable systems approach for evaluating technology supported learning", *European Journal of Operational Reseach*, Vol. 259 No. (2), pp. 626-641.

Hazir, O. (2015), "A review of analytical models, approaches and decision support tools in project monitoring and control", *International Journal of Project Management*, Vol. 33, pp. 808-815.

Hetzler, S. (2008), "Pathological systems", *International Journal of Applied Systemic Studies*, Vol. 2 No. 1-2, pp. 25-39.

Herrmann, C., Bergmann, L., Halubek, P. and Thiede, S. (2008), "Lean production system design from the perspective of the viable system model", in: Mitsuishi, M., Ueda, K. and Kimura, F. (Eds.), *Manufacturing Systems and Technologies for the New Frontier*, Springer, London, pp. 309-314.

Hildbrand, S. and Bodhanya, S. (2013), "The potential value of the Viable System Model as a managerial tool", *Management Dynamics*, Vol. 22 No. (2), pp. 2-15.

Hildbrand, S. and Bodhanya, S. (2015), "Guiadance on applying the viable system model", *Kybernetes*, Vol. 44 No. 2, pp. 186-201.

Hoda, R. and Murugesan, L. (2016), "Multi-level agile project management challenges: A self-organizing team perspective". *The Journal of Systems and Software*. Vol 117, pp. 245-257.

Hornstein, H.A. (2015), "The integration of project management and organizational change management is now a necessity", *International Journal of Project Management*, Vol. 33 No. 2, pp. 291-298.

Hoverstadt, P. (2008), *The fractal organization: creating sustainable organizations with the Viable System Model*, John Wiley & Sons, Chichester.

Jackson, M.C. (1986), "The Cybernetic Model of the Organisation: An assessment", in: Trappl, R. (Ed.), *Cybernetics and Systems*, Reidel Publishing Complany, Dordrecht, pp. 189-196.

Jackson, M.C. (1992), "The soul of the Viable System Model", *Systemic Practice and Action Research*, Vol. 5 No. 5, pp. 561-564.

Jackson, M.C. (2000), *Systems Approaches to Management*, Kluwer Academic Publishers, New York.

Jackson, M.C. and Flood, R.L. (1988), "Cybernetics and Organization theory: a critical review", *Cybernetics and Systems*, Vol. 19 No. 1, pp. 13-33.

Jain, R., Sharma, D. and Khatri, S.K. (2017), "Hybrid artificial intelligence model based on neural network simulation models for software maintainability prediction", paper presented at the International Conference on Infocom Technologies and Unmanned Systems (Trends and Future Directions) (ICTUS), 18-20 December, Dubai, pp. 705-708.

Jun, L., Qiuzhen, W. and Qingguo, M. (2011), "The effects of project uncertainty and risk management on is development project performance: a vendor perspective", *International Journal of Project Management*, Vol. 29 No. 7, pp. 923-933.

Kawalek, P. and Wastell, D.G. (1999), "A case study evaluation of the use of the viable system model in information system development", *Journal of Database Management*, Vol. 10 No. 4, pp. 24-32.

Khosrowjerdi, M. (2011), "Designing a viable scientific communication model: VSM approach", *Library Hi Tech*, Vol. 29 No. 2, pp. 359-372.

Kummamuru, A. and Hussaini, S.W. (2015), "Designing an organization structure for large and complex IT programs using the Viable System Model (VSM)", paper presented at IEEE Region 10 Conference (TENCON), 1-4 November, Macao, pp. 1-5.

Kuthyola, K.F., Liu, J.Y.C. and Klein, G. (2017), "Influence of task interdependence on teamwork quality and project performance", in: Abramowicz W. (Ed.), *Business Information Systems, BIS 2017, Lecture Notes in Business Information Processing*, Vol. 288, Springer, Cham, pp.135-148.

Lehtinen, T.O., Mantyla, M.V., Vanhanen, J., Itkonen, J. and Lassenius, C. (2014), "Perceived causes of software project failures – An analysis of their relationships", *Information and Software Technology*, Vol. 56, pp. 623-643.

Leonard, A. (2008), "Integrating sustainability practices using the viable system model", *Systems Research and Behavioral Science*, Vol. 25 No. 5, pp. 643-654.

Leonard, A. (2009), "The viable system model and its application to complex organizations", *Systemic practice and action research*, Vol. 22 No. 4, pp. 223-233.

Linares, J., Melendez, K., Flores, L. and Dávila, A. (2018), "Project Portfolio Management in Small Context in Software Industry: A Systematic Literature Review", in: Mejia, J., Muñoz, M., Rocha, Á., Quiñonez, Y. and Calvo-Manzano, J. (Eds.), *Trends and Applications in Software Engineering, CIMPS 2017, Advances in Intelligent Systems and Computing*, Vol. 688, Springer, Cham, pp. 45-60.

Mahmood, S., Anwer, S., Niazi, M., Alshayeb, M. and Richardson, I. (2017), "Key factors that influence task allocation in global software development", *Information and Software Technology*, Vol. 91, pp. 102-122.

Malik, F. (1993), "Understanding a Knowledge Organisation as a Viable System", in: Espejo, R. and Schwaninger, M. (Eds.), *Organisational Fitness: Corporate Effectiveness through Management Cybernetics*, Campus, Frankfurt, pp. 93-115.

Midgley, G. (2006), "Systemic intervention for public health", *American journal of public health*, Vol. 96 No. 3, pp. 466-472.

Millhollan, C. and Kaarst-Brown, M. (2016), "Lesson for IT Project Manager Efficacy: A Review of the Literature Associated with Project Success", *Project Management Journal*, Vol. 47 No. 5, pp. 89-106.

Miterev, M., Mancini, M. and Turner, R. (2017a), "Towards a design for the project-based organization", *International Journal of Project Management*, Vol. 35 No. 3, pp. 479-491.

Miterev, M., Turner, J.R. and Mancini, M. (2017b), "The organization design perspective on the project-based organization: a structured review", *International Journal of Managing Projects in Business*, Vol. 10 No. 3, pp. 527-549.

Mohagheghi, P. and Jorgensen, M. (2017), "What contributes to the success of IT projects? An empirical study of IT projects in the Norwegian public sector", *Journal of Software*, Vol. 12, No. 9, pp. 751-758.

Mohapatra, S., and Gupta, D.K. (2011), "Finding Factors Impacting Productivity in Software Development Project using Structured Equation Modelling", *International Journal of Information Processing and Management*, Vol. 2 No. 1, pp. 90-100.

Molokken-Ostvold, K. and Jorgensen, M. (2005), "A comparison of software project overruns-Flexible versus sequential development models", *IEEE Transactions on Software Engineering*, Vol. 31 No. 9, pp. 754-766.

Monat, J.P. and Gannon, T.F. (2015), "What is System Thinking? A review of selected literature plus recommendations", *American Journal of Systems Science*, Vol. 4 No. 1, pp. 11-26.

Morales-Arroyo, M.A., Chang, Y.K., Barragán-Ocaña, A., Jiménez, J. and Sánchez-Guerrero, G. (2012), "Coordination mechanisms illustrated with project management using the Viable System Model (VSM) as organizational framework", *Jindal Journal of Business Research*, Vol. 1 No. 2, pp. 163-176.

Motloch, J.L., (2016), "Unlocking complexity: big science project and research agenda", *International Journal of Design & Nature and Ecodynamics*, Vol. 11 No. 4, pp. 563-572.

Murad, R.S.A. and Cavana, R.Y. (2012), "Applying the viable system model to ICT project management", *International Journal Applied Systemic Studies*, Vol. 4 No. 3, pp. 186-205.

Müller, R. and Jugdev, K. (2012), "Critical success factors in projects: Pinto, Slevin, and Presscot – the elucidation of project success", *International Journal of Managing Projects in Business*, Vol. 5 No. 4, pp. 757-775.

Munir, H., Wnuk, K. and Runeson, P. (2015), "Open innovation in software engineering: a systematic mapping study", *Empirical Software Engineering*, Vol. 21 No. 2, pp. 684-723.

Nasir, M.H.N. and Sahibuddin, S. (2011), "Critical success factors for software projects: A comparative study", *Scientific Research and Essays*, Vol. 6 No. 10, pp. 2174-2186.

Nechansky, H. (2013), "Issues of organizational cybernetics and viability beyond Beer's viable systems model", *International Journal of General Systems*, Vol. 42 No. 8, pp. 838-859.

Nyström, C.A. (2006), "Design rules for intranets according to the viable system model", *Systemic Practice and Action Research*, Vol. 19 No. 6, pp. 523-535.

Orengo, M. (2018), "Theoretical notes regarding the practical application of Stafford Beer's viable system model", *Kybernetes*, Vol. 47 No. 2, pp. 262-272.

Pankratz, O. and Loebbecke, C. (2011), "Project Manager's Perception of IS Project Success Factors – A Repertory Grid Investigation", paper presented at the European Conference on Information Systems – ICT and Sustainable Service Development (ECIS2011), 9-11 June, Helsinki, Finland, available at: http://www.mtm.uni-koeln.de/team-loebbecke-publications-conf-proceedings/Conf-146-2011-ProjectManagersPerception.pdf (accessed 20 August 2017).

Pfiffner, M. (2010), "Five experiences with the viable system model", *Kybernetes*, Vol. 39 No. 9/10, pp. 1615-1626.

Pollack, J., Helm, J., and Adler, D. (2018), "What is the Irong Trinangle, and how has it changed?", *International Journal of Managing Projects in Business*, Vol. 11 No. 2, pp. 527-547.

Preece, G., Shaw, D. and Hayashi, H. (2013), "Using the Viable System Model (VSM) to structure information processing complexity in disaster response", *European Journal of Operation Research*, Vol. 224 No. 1, pp. 209-218.

Pressman, R.S. and Maxim, B.R. (2014), *Software Engineering. A practitioner's approach, eight edition*, McGraw-Hill Education, New York.

Puche Regaliza, J.C. (2014), "Extending the Viable System Model scope on ICT-sector software projects in Castilla y León", *Kybernetes*, Vol. 43 No. 2, pp. 192-209.

Puche Regaliza, J.C. (2015), "Quantitative analysis of Viable System Model on software projects in the ICT sector in Castilla y León", *Kybernetes*, Vol. 44 No. 5, pp. 806-822.

Puche Regaliza, J.C., Jiménez, A. and Arranz Val, P. (2017), "Viable System Model structuring of success factors in software projects", *International Journal of Managing Projects in Business*, Vol. 10 No. 4, pp. 897-919.

Radujkovic, M. and Sjekavica, M. (2017), "Project Management Success Factors", *Procedia Engineering*, Vol. 196, pp. 607-615.

Reissberg, A. (2011), "The advanced syntegration as the most effective and efficient tool for large-scale disaster response coordination", *Systems Research and Behavioral Science*, Vol. 28 No. 5, pp. 455-464.

Remington, K. and Pollack, J. (2016), *Tools for complex projects*, Tailor & Francis Group, London.

Salerno, M.S., Gomes, L.A.V., da Silva, D.O., Bagno, R.B. and Freitas, S.L.T.U. (2015), "Innovation processes: Which process for which project?", *Technovation*, Vol. 35, pp. 59-70.

Sanchez, O.P., Terlizzi, M.A. and de Moraes, H.R.C. (2017). "Cost and time project management success factors for information systems development projects", *International Journal of Project Management*, Vol. 35 No. 8, pp. 1608-1626.

Schwaninger, M. (2000), "Managing Complexity – The Path Toward Intelligent Organizations", *Systemic Practice and Action Research*, Vol. 13 No. (2), pp. 207-241.

Schwaninger, M. (2006), "Design for viable organizations. The diagnostic power of the Viable System Model", *Kybernetes*, Vol. 35 No. 7/8, pp. 955-966.

Schwaninger, M. (2009), *Intelligent Organizations. Powerful Models for Systemic Management, 2nd ed.*, Springer, Berlin.

Schwaninger, M. and Scheef, C. (2016), "A Test of the Viable System Model: Theoretical Claim vs. Empirical Evidence", *Cybernetics and Systems*, Vol. 47 No. 7, pp. 544-569.

Shaw, D.R., Snowdon, B., Holland, C.P., Kawalek, P. and Warboys, B. (2004), "The viable systems model applied to a smart network: the case of the UK electricity market", *Journal of Information Technology*, Vol. 19 No. 4, pp. 270-280.

Standish Group (2018), "Chaos Report", Available at: https://www.standishgroup.com/sample_research_files/DemoPRBR.pdf (accessed 10 February 2019).

Stephens, J. and Haslett, T. (2011), "A set of conventions, a model: an application of stafford beer's viable systems model to the strategic planning process", *Systemic Practice and Action Research*, Vol. 24 No. 5, pp. 429-452.

Stich, V. and Groten, M. (2015), "Design and simulation of a logistics distribution network applying the Viable System Model (VSM)", *Procedia Manufacturing*, Vol. 3, pp. 534-541.

Sutar, S. and Ghatule, A. (2013), "Evaluating Critical Success Factors in Distributive Software Projects: Implementation Efforts", *International Journal of Research in Computer Science and Information Technology*, Vol. 1 No. 1(A), pp. 114-118.

Takey, S.M. and de Carvalho, M.M. (2015), "Competency mapping in project manager: An action research study in an engineering company", *International Journal of Project Management*, Vol. 33, 784-796.

Tanir, O. (2017), "Simulation-Based Software Engineering", in: Mittal, S., Durak, U. and Ören, T. (Eds.), Guide to Simulation-Based Disciplines. Simulation Foundations, Methods and Applications, Springer, Cham, pp. 155-166.

Tavella, E. and Papadopoulos, T. (2015), "Expert and novice facilitated modelling: A case of a viable system model workshop in a local food network", *Journal of the Operational Research Society*, Vol. 66 No. (2), pp. 247-264.

Terra, L.A., Ventura, C.A., Medeiros, M.L. and Passador, J.L. (2016), "Strategies for the Distribution of Power in Brazil: A Proposal from the Perspective of the Viable System Model (VSM)", *Systems Research and Behavioral Science*, Vol. 33 No. (2), pp. 224-234.

Thomas, R. (2006), "Is the Viable System Model of Organization inimical to the concept of human freedom?", *Journal of Organisational Transformation & Social Change*, Vol. 3 No. (1), pp. 69-83.

Toledo-Parra, C.A., Gamboa-Sarmiento, S.C. and Di Fatta, D. (2017), "Studying University as Social Systems Using the Viable System Model: mApp and Semantic Web Technologies at the Industrial University of Santander", *Journal of Organisational Transformation & Social Change*, Vol. 14 No. (1), pp. 56-77.

Toprak, S. and Torlak, N. (2018), "An adaptative use of Viable System Model with knowledge system diagnositcs serving industrial democracy in a textile manufacturing company", *Systemic Practice and Action Research*, Vol. 31 No. 1, pp. 1-26.

Tsuchiya, Y. (2007), "Autopoietic Viable System Model", *Systems Research and Behavioral Science*, Vol. 24 No. 3, pp. 333-346.

Türke, R.E. (2006), "Towards productive and sustainable forms of interaction in governance", *Kybernetes*, Vol. 35 No. 1/2, pp. 164-181.

Ulrich, W. (1981), "A Critique of Pure Cybernetic Reason: The Chilean Experience with Cybernetics", *Journal of Applied Systems Analysis*, Vol. 8, pp. 33-59.

Van der Zouwen, L. (1996), "Methodological problems with the empirical testability of sociocybernetic theories", *Kybernetes*, Vol. 25 No. 7/8, pp. 100-108.

Wiener, N. (1948), *Cybernetics or the Control and Communication in the Animal and the Machine*, MIT Press, Cambridge.

Yadav, V. (2015), "A flexible management approach for globally distributed software projects", *Global Journal of Flexible Systems Management*, Vol. 17 No. 1, pp. 29-40.

Yolles, M. (2006), *Organizations as complex systems: An introduction to knowledge cybernetics, Vol. 2*, IAP, Connecticut.

Zadeh, M.E., Lewis, E, Millar, G., Yang, Y. and Thorne, C. (2014), "The use of viable system model to develop guidelines for generating enterprise architecture principles", paper presented at the IEEE International Conference on Systems, Man and Cybernetics (IEEE SMC14), 5-8 October, San Diego, USA, available at: https://ieeexplore.ieee.org/document/6974047/ (accessed 15 August 2017).