Call for Software Tenders: Features and Research Problems

Jorge Hochstetter, Carlos Cares
Departamento de Ingeniería de Sistemas
Universidad de La Frontera
Temuco, Chile
jhoch@ufro.cl, carlos.cares@ceisufro.cl

Abstract— **A relevant part of software industry deals with ad hoc software solutions, which are externalized by software consumers. The process of acquirment follows a public procedure of requesting proposals. This procedure is based on a call-for-tenders document that contains, at least, a software specification and project constraints, such as budget and time. The general assumption is that the requirements stage happens prior to the call-for-tender process. However, public documents of software tendering processes support the contrary assumption. The aim of this paper is to sustain that call-for-tender processes require additional study from an empirical point of view in order to solve problems derived from current industry practices. We have analyzed call for tenders in relation to requirements engineering proposals and also under a procedural approach. In order to sustain the inclusion of call for tenders in the scope of software engineering a set of different open problems is identified.**

Keywords: call for tenders; software tenders; requirements engineering; tendering process; software projects.

I. INTRODUCTION

Currently, organizations are increasingly becoming acquirers of needed capabilities by obtaining products and services from suppliers and the development of these capabilities are diminishing in-house [1]. This scenario also holds true for public institutions. In particular, organizations in the public sector who are normally required by law to make public bids for their purchases in order to achieve a transparent process. In several countries, the government is the main buyer of goods and services [2]. For this reason, these countries have invested a great deal of money and effort in defining purchasing policies [4],[5]. The Software industry is a particular case where this has happened.

Therefore, organizations stimulate the competition in the software industry by way of performing calls for tenders (C4T). A call for tenders is the process where a company invites the providers to satisfy particular needs, goods or services. In this paper, we are considering the case of software tendering processes that imply software development, i.e., a software project.

The bidding process associated with a software product is constituted by a set of common practices related to Software Engineering (SE), which have been explored mainly from the Requirements Engineering (RE) perspective. Indeed, within the various problems and challenges of the Requirements Engineering, there is improvement of the quality associated with the specification process, as part of the software process [6].

The literature about calls for tenders is contradictory to the practice. On the one hand, Requirements Engineering promotes a complete and consistent software requirements specification (SRS) before starting the software process. Thus, the theoretical suggestions and corresponding assumptions indicate that the call to public notice phase occurs after the ER stages. Therefore, an SRS becomes part of the call-for-tenders document [6],[7],[8],[9]. On the other hand, on the practice side, we have found that it is very difficult to find a SRS as part of the call-for-tenders document. What we find is that just some of these are included as part of the bid, i.e., the need of including a requirements elicitation and specification phase [10]. Other sources confirm that call-for- tenders documents just include a first approximation of Requirements Engineering's stages and products [9],[11],[12].

With this information, we can conclude that software industry generates its offers for software development projects in a scenario with uncertain and incomplete information. Moreover, if we are aware that this phase is the seed of a software project, then, it is obvious that uncertain and incomplete information at the level of a call-for-tenders document can imply a high frequency of two deviations: (a) under the given budget, time and project conditions, the software can be well developed, which probably also means that projects conditions are relaxed, hence the customer is losing efficiency, or (b) under the given scenario, the project cannot be developed, which implies that the provider will not accomplish the project goals, or should spend extra money to achieve them. The worst case would reach a state where the project results are aborted. These cases have been illustrated by both theoretical simulations [13],[14] and empirical findings [15],[16].

As a position paper, the goal of this essay is to sustain that call-for-tenders processes require additional study and research with respect to what is happens in industry today. To cover this problem, we analyze the call-for-tenders process stages comparing them to Requirements Engineering stages, recognizing different research problems and proposing research approaches to these sets of problems.

Furthermore, in Section II, we present the basics for differentiating call-for-tenders documents and processes from SRS documents and RE stages. In Section III, we present a generic call-for-tenders process from both customers' and software providers´ perspectives, and we argue about research problems associated to call-for-tenders activities. To conclude, we summarize our point of view of differentiating but, at the same time, integrating

Copyright (c) IARIA, 2012. ISBN: 978-1-61208-230-1

Requirements Engineering body of knowledge to call-for-tenders processes, as a first step to covering call-for- tenders problems.

II. DIFFERENTIATING REQUIREMENTS ENGINEERING AND CALL FOR TENDERS

To our knowledge, public tender processes for software products have been scarcely investigated in technical literature. Normally, the problem is put in the context of RE activities.

However, in Software Engineering, the acquisition problem has already been recognized as an independent process having its own stages and deliverable products. A clear example is CMMI for acquisitions [1]. In this case, the process view point is the customer´s; therefore, it includes engaging and managing suppliers. Although it is not common, the problem of Call for Tenders has been separately analyzed. For example, Costa et al. [12] address the problem of systematically performing the bidding phase of the public tender process, and propose a multicriteria socio-technical approach to increase the transparency and efficiency of the process.

TABLE I. CALLS FOR TENDERS AND REQUIREMENTS ENGINEERING DIFFERENCES

Requirements Engineering Stage	Call-for-tenders Process
The product is a software requirements specification.	The product is a call-for-tenders document.
The software requirements specification includes software requirements.	The call-for-tenders document includes managerial, economic, budget and software requirements.
It is assumed that the list of software requirements in the software requirements specification is complete.	It is assumed that the list of software requirements in the call-for-tenders document may be incomplete.
Normally it occurs after a contract.	Normally it occurs before a contract.
The software developer may be selected.	The software developer should be selected.
Most of the time a software requirements specification constitutes a specific technical solution.	Most of the time a call-for-tenders document looks for a specific technical solution.
Most of the time it´s focuses are software requirements.	Most of the time it´s focuses are business goals.
There are a set of modelling languages for representing Requirements Engineering deliverables.	There are not modelling languages for representing call for tenders's deliverables.
The involved actors are requirements engineers and stakeholders.	The involved actors are customers and providers.

On the other hand, when practice and theory are confronted, there is evidence that call-for-tenders processes are not following literature recommendations [10],[17],[18]. Moreover, by analyzing different call-for-tenders documents from public distributions [19],[20],[21] we can see that many call-for-tenders documents include, as part of the project, the requisite of formulating a Requirements Engineering stage and a software requirements specification as a deliverable product of the outsourced project.

We sustain that, from the perspective of Software Engineering, i.e., from software production point of view, the generation of a Call for Tenders is a completely different stage from requirements engineering. For example, we have detected a high dispersion of extensions, level of detail, technical specifications, and business goals descriptions among other differences. Moreover, a software requirements specification is only part of the call-for-tenders document, if at all, and there is a low and controlled interaction between suppliers and customers. The detected differences are presented in Table I.

III. CALL FOR TENDERS' RELATED WORK

In spite of the already recognized differences, it is worth mentioning that RE appears to be similar to the call-for-tenders process and, most of the time, they even appear as one phase in the software process. In order to describe current proposals regarding Requirements Engineering and Call for Tenders we summarize them.

Renault et al. [9] present the case of the elicitation and selection of software components. These processes are driven by public tender processes and the use of a Requirement Patterns Catalogue is proposed to save time and reduce errors. Carvallo & Franch [7] present a similar case study where the activities undertaken to obtain, analyze and structure the requirements to be included in a public tender process for an ERP are analyzed.

Lauesen [17] provides a set of guidelines for the creation of the call-for-tenders documents, i.e. the point of view is from the customer, although it is also recognized that these guidelines could also support the suppliers. In a practical sense he affirms that customers do not normally apply these guidelines.

Paech et al. [18] report a set of experiences of a supplier company analyzing call-for-tenders documents for generating technical proposals. They found that existing challenges can be confronted by using requirements engineering techniques.

Additionally, we have started a research line to analyze call-for-tenders documents from the perspective of requirements engineering metrics and it´s comparison to software providers' decision variables. Thus, on one hand [10], we present an approach for analyzing call-for-tenders documents adapting an ontology of speech from goal-oriented requirements engineering to generate metrics. On the other, [3] we describe the application of a focus group technique that provides some clues to understand providers' analyses for applying, or not, for a call-for-tenders process. An initial set of variables are identified in order to describe critical decision points.

From this review, our opinion is that the topic of Call for Tenders is almost unmentioned in Software Engineering literature, but, at the same time, is an underlying topic, i.e., it seems to be present, however, it is not referred to by it´s name (or similar names). This way, several results from Software Engineering research could be applied to different stages of a call-for-tenders process.

Copyright (c) IARIA, 2012. ISBN: 978-1-61208-230-1

In order to make these relationship explicit, i.e., between call-for-tenders process and existing research approaches, we offer an analysis which is based on call-for-tenders stages. We consider the stages starting from the internal requirements collection, to the software development stage made by an external provider. In each stage we have looked for existing software engineering approaches which support it. In addition, we also identify specific problems at each stage to which we have not found any approaches. Therefore, we call them open problems.

IV. CALLS FOR TENDERS SUPPORT AND OPEN PROBLEMS

In this section, we present the general stages of a call-for-tenders process. In the left part of Table 1 we have summarized these stages from both customer´s and provider´s perspectives. Further on, we describe each stage and we analyze how existing software engineering approaches support the technical task at each stage. However, it is necessary to consider that different referred to authors, most of the time, do not distinguish if they are assuming the existence of a call-for-tenders process.

1) Gather organizational needs. We assume that an organization can have a set of planned systems to develop. Requirements Engineering's proposals adopt a top-down focus, i.e. they can conceive software systems from Business strategies [22],[23] or from Business goals [6],[24]. These systems should accomplish specific functionalities and quality attributes in order to deliver business goals. However, there are two processes to which we have not found approaches. Firstly, computational units collect requirements as part of a day-to-day routine. Therefore, the process is not exclusively top-down. Thus, here is an open problem: given a set of already gathered requirements, how could the parameters of future computational systems be established in order to get convenient bids? Although the "natural" answer seems to be hybrid approaches (top-down and bottom-up together) we have found neither hybrid approaches considering bottom-up requirements, nor the way of combining requirements to conceive a set of future software systems.

2) Estimate budget and time for conceiving systems. To trigger a system development process it is necessary to have a good estimation of both the business value of the system and the cost of the system (including software and organizational adoption). Regarding the business value there are several approaches [13],[25]. In the case of estimating the software development effort, in spite of it being a familiar topic in software engineering [26],[30] the methods require a previous estimation of software size (measured by lines of code, function points or use case points), most of these methods consider particular features of software teams. Therefore, from the perspective of a call-for-tenders process, an early estimation method is required for a generic team which considers the current context. Thus, these methods consider special requirements such as interoperability of existing systems and the reuse of existing libraries or software components, but overall, an incomplete requirements specification. Thus, an open problem here is how budget can be better estimated to incentivize providers to apply and, at the same time, to pay the "fair" price.

3) Specify a call-for-tenders document. At this stage, the customer should produce the document which will be published. Although components of a call for tenders could be well-established, [1] it is necessary to reach an equllibrium between collecting enough information for providers, in order that they formulate good solutions to the problem, and the cost of collecting this information. A related problem is how much detail should the specified solution have (if any) because, on one hand, an open perspective should enable creative and unexpected technical solutions, and at the same time present new organizational challenges to adopt them. On the other hand, a closed perspective allows for the presentation of tailored solutions but, at the same time, reduces the amount of providers, hurting the competitiveness of call-for-tenders process.

4) Search and select requests for proposals. Now, from the perspective of providers, several calls could be available in a specific time period. The providers must decide from a set of call-for-tenders documents which of them they will apply for. Normally, just to study one call-for-tenders document could take too much time. We have not found approaches to make the decision, about how to select calls to evaluate nor how to evaluate them. Derived problems are: how to promise proper software functionality if this has been (normally) poorly specified. A particular problem detected by Hochstetter et al. [3] is how to match the specified problem to existing software assets in order to apply with a competitive offer.

5) Questions and answers. Normally, a public call-for-tenders process considers a public questions and answers process where providers can resolve doubts emerging from the interpretation of call-for-tenders documents. Questions and answers are published, thus, new variables are added to the process. For example, the number of questions should be an indicator of providers' interest in applying and hence an indicator of process` competitiveness which could imply that more than one potential provider may abandon the process. At the same time a resolved doubt, also resolves any doubt regarding competition. Therefore, the open problem for providers is to achieve an equilibrium between resolving uncertainness to the public and resolving uncertainness over competitiveness. From the perspective of a public-sector customer, answers should show an efficient use of public resources and a transparent way of spending them.

6) Prepare a proposal. Once a provider has decided to apply, the next problem is how to build a "good" proposal. Some work has been done about how to present requirements in a clear and structured way [9],[26] However, the problems seem wider. We have detected that multiple applications from the same organization is normal practise. Therefore, we can even consider the problem about how many proposals can be made and then answer how to make them. Therefore, "good" proposals should consider not only technical content, but available time, assets and team among internal resources and existing systems, standards, assets and stakeholders availability among the external ones. Also, it is necessary to

Copyright (c) IARIA, 2012. ISBN: 978-1-61208-230-1

achieve an equilibrium between the cost of preparing the offer and the probability of losing it. We formulate the problem of how to build an offer considering socio-technical constraints from the customer and provider.

7) Select an offer (provider). Following providers' applications, a selection process is initiated. Normally, this process has an administrative stage where legal constraints are verified. Only then is a technical evaluation begun. It seems to be a stage where different approaches from Software Engineering have been proposed: quality models including non-technical factors [7], framework for selecting COTS [28], particular suggestions from acquisition standards [1], and other contemporary approaches such as trust relationships [29]. In order to acknowledge this stage as an already addressed topic, we do not add open problems here.

8) Negotiate the contract. The main goal of this stage is to sign the final contract. In the case of call for tenders from the private sector, it could imply an opportunity to detail some "risky sentences" from both, call for tenders and the bid. However, in the case of government calls, bureaucracy should seriously delay the starting of the process, which would mean changes of socio-technical conditions under which the offer was formulated. To start, or not to start, the project without a contract could be a relevant decision to make. Therefore, trust is a variable key here. There are no studies showing a complete set of variables and how to deal with them. Thus, intermediate payment, engineering deliverables, intellectual property and exploitation of it, stakeholders' availability, post-end availability from the provider, and associated penalties are variables to consider and balance in order to reach a final agreement.

9) Develop Software. At this stage, the traditional software life cycle is activated. As we have sustained, most of the time it involves requirements elicitation and specification. Therefore, traditional approaches can be applied regarding additional aspects, such as imposed project milestones, external monitoring, programming practices and technology imposed by the contract and other considerations derived from a supervised project.

TABLE II. CALL FOR SOFTWARE TENDERS, EXISTING APPROACHES AND OPEN PROBLEMS

Customer	Supplier	Approaches	Open Problems
Gather organizational needs		Top-down proposals from Requirements Engineering e.g. goal-oriented requirements engineering [23],[31] and strategic IT alignment [32] produce systems to-be.	How to combine incomplete requirements to conceive systems which could be, moreover, separately developed even by different external teams.
Estimate budget and time		Traditional software engineering economics considers software size metrics and team features as part of estimation models [25].	How to calculate early software projects estimations under incomplete and uncertain information.
Specify a call-for-tenders document		There are some proposed standards for software acquisition processes including topics to include in call-for-tenders documents [1]. Additionally, other Requirements Engineering approaches results are useful for call-for-tenders specifications [9],[10],[11].	How to aid the writing of call-for-tenders documents balancing detailed information and enough space for creative solutions.
	Search and select call for tenders	No accessible approaches	How to select calls for tenders and how to efficiently evaluate them in order to build competitive bids with low effort.
	Ask for doubts in tender process	No accessible approaches	How to select what doubts to address or what kind of questions or answers could lose other competitors.
Answer questions		No accessible approaches	How to show, in answers, high consistency to call-for-tenders document, a transparent process and efficient use of public resources (if it corresponds).
	Prepare a proposal	The challenges can be confronted by using requirements engineering techniques for generating technical proposals [14],[18].	How to generate a bid regarding the cost of prepare it, the probability of losing it and the involved risks coming from incomplete information.
Select an offer (provider)		There are studies for applying techniques to select software components and providers [12], [26], [27].	.
Negotiate the contract		No accessible approaches	How to handle and balance variables such as changing requirements, stakeholders' availability, penalties, post-sale service at contract time.
	Develop Software	It corresponds to the classical scope of Software Engineering; therefore, its results are applicable to this stage.	How to develop software under a scenario of external supervising, monitoring and even, sometimes, under different developing conditions.

Copyright (c) IARIA, 2012. ISBN: 978-1-61208-230-1

V. CONCLUSIONS AND FURTHER LINES OF RESEARCH

In this paper, we have presented the problem of producing and applying to a public call for software tenders as a Software Engineering problem which requires additional approaches. Because it appears to be closely related to Requirements Engineering Stages, we have presented a set of differences among them. Moreover, we have sustained that the common theoretical assumption regarding to call for tenders happening after the requirements stage is not necessary true, therefore, a set of new problems are derived from actual practices. In order to analyze this situation we have followed call-for-tenders phases under the perspective of a customer-supplier interaction. In each phase, we have identified current Software Engineering approaches but we have also obtained a list of open problems corresponding to the different call-for-tenders phases.

Therefore, we have provided enough arguments to sustain that the call-for-tenders process is a different Software Engineering stage and deserves additional research attention, especially in the way that it takes place in industry. Particularly, we propose to focus on the set of problems partially treated which seem to have a high impact on software projects. After all, the call-for-tenders process is the seed of software process.

ACKNOWLEDGMENTS

This work has been sponsored by the Investigation Direction, University of La Frontera, Temuco, Chile, DIUFRO DI11-0015 Project.

REFERENCES

[1] C. Team, "CMMI for acquisition, version 1.3," Technical Report, Carnegie Mellon University, Pitts burgh, 2010.

[2] T. P. Hill, "On goods and services," The Review of Income and Wealth, vol. 23, no. 4, pp. 314-339, 1977.

[3] J.Hochstetter, C. Cachero, C. Cares, and S. Sepúlveda, "Call for Tender Challenges in Practice: a Field Study," in Proc. XV Congreso Iberoamericano en Software Engineering, Buenos Aires, Argentina, 2012.

[4] C. C. Pimenta, "Gestión de compras y contrataciones gubernamentales," RAE-electrónica, vol. 1, no. 1, pp. 1-12, 2002.

[5] S. Lauesen and J. P. Vium, "Communication gaps in a tender process," vol. 10, no. 4, pp. 247-261, Nov. 2005.

[6] A. v. Lamsweerde, "Requirements engineering: from craft to discipline," in Proc. of the 16th ACM SIGSOFT Int. Symposium on Foundations of software engineering, Atlanta, Georgia, 2008, pp. 238-249.

[7] J. P. Carvallo and X. Franch, "On the Use of Requirements for Driving Call-for-Tender Processes for Procuring Coarse-grained OTS Components," in Proc. Requirements Engineering Conference, 2009. RE '09. 17th IEEE International, pp. 287 - 292, 2009.

[8] S. Biffl, D. Winkler, R. Höhn, and H. Wetzel, "Software process improvement in Europe: potential of the new V-modell XT and research issues," Software Process Improvement Practice, Wiley 2006, vol. 11, no. 3, pp. 229–238.

[9] S. Renault, Ó. Ménez-Bonilla, X. Franch, and C. Quer, "A Pattern-based Method for building Requirements Documents in Call-for-tender Processes," International Journal of Computer Science and Applications, vol. 6, no.5 pp. 175 -202, 2009.

[10] J. Hochstetter, C. Díaz, and C. Cares, "Licitaciones Públicas de Software: Métricas Basadas en Actos de Habla," in Proc. Information Systems and Technologies (CISTI), 2012 7th Iberian Conference on, Madrid, España, 2012, pp. 451-456.

[11] J. Brender and P. McNair, "User requirements specifications: a hierarchical structure covering strategical, tactical and operational requirements," International Journal of Medical Informatics, vol. 64, pp. 83-98, 2001.

[12] C. B. e. Costa, E. Corrêa, J.-M. D. Corted, and J.-C. Vansnickd, " Facilitating bid evaluation in public call for tenders: A socio-technical approach," Omega, vol. 30, no.30, pp. 227-242, 2002.

[13] B. A. Aubert, S. Rivard, and M. Patry, "A transaction cost approach to outsourcing behavior: some empirical evidence," Information and Management, vol. 30, no. 2, pp. 51-64, 1996.

[14] S. Whang, "Contracting for software development," Management Science, vol. 38, no.3, pp. 307-324, 1992.

[15] D. Gefen, S. Wyss, and Y. Lichtenstein, "Business familiarity as risk mitigation in software development outsourcing contracts," MIS Quarterly, vol.32, no.3, pp. 531-551, 2008.

[16] R. T. Nakatsu and C. L. Iacovou, "A comparative study of important risk factors involved in offshore and domestic outsourcing of software development projects: A two-panel Delphi study," Information & Management, vol. 46, no. 1, pp. 57-68.

[17] S. Lauesen, "COTS tenders and integration requirements," In Proc. of the IEEE Int. Req. Eng. Conf. (RE), 2004, pp. 166-175.

[18] B. Paech, R. Heinrich, G. Zorn-Pauli, A. Jung, S. Tadjiky, B. Regnell, and D. Damian, "Answering a Request for Proposal – Challenges and Proposed Solutions Requirements Engineering: Foundation for Software Quality," in Proc. *REFSQ*, Essen, Germany, 2012, pp.16-29.

[19] ChileCompras, "Website. http://www.mercadopublico.cl (04 2012)."

[20] Comprasnet, "Website. http://www.comprasnet.gov.br (04 2012)."

[21] Compranet, "Website. http://www.compranet.gob.mx (04 2012)."

[22] E. J. Braude and M. E. Bernstein, "Software engineering: Modern Approaches," J. Wiley & Sons. Second Edition, 2011. ISBN 978-0-471-69208-9.

[23] E. Yu, "Modelling Strategic Relationships for Process Reengineering," Ph.D. thesis, also Tech. Report DKBS-TR-94-6, Dept. of Computer Science, University of Toronto, 1995.

[24] P. Keil, "Principal agent theory and its application to analyze outsourcing of software development," in ACM SIGSOFT Software Engineering Notes, vol. 30, pp. 1-5, 2005.

[25] B. W. Boehm, "Software Engineering Economics," Software Engineering, IEEE Transactions on, vol. 10, no. 1, pp. 4-21, 1984.

[26] N. Aissaoui, M. Haouari, and E. Hassini, "Supplier selection and order lot sizing modeling: A review," Computers \& operations research, vol. 34, no. 34, pp. 3516-3540, 2007.

[27] D. Lowe, "A framework for defining acceptance criteria for web development projects," Web Engineering, pp. 279-294, 2001.

[28] H. C. Esfahani, E. Yu, and M. C. Annosi, "Strategically balanced process adoption," in Proc. of the 2011 International Conference on Software and Systems Process, Hawaii, USA, pp. 169-178, 2011.

[29] A. Heiskanen, M. Newman, and M. Eklin, "Control, trust, power, and the dynamics of information system outsourcing relationships: A process study of contractual software development," The Journal of Strategic Information Systems, vol. 17, no. 4, pp. 268-286, 2008.

[30] H. Erdogmus, B. W. Boehm, W. Harrison, D. J. Reifer, and K. J. Sullivan, "Software engineering economics: background, current practices, and future directions," in Proc. of the 24th International Conference on Software Engineering, 2002, pp. 683-684.

[31] A. v. Lamsweerde, "Goal-oriented requirements engineering: A guided tour," in Requirements Engineering, 2001. Proceedings. Fifth IEEE International Symposium on: IEEE, 2001, pp. 249-262.

[32] S. J. Bleistein, K. Cox, J. Verner, and K. T. Phalp, "B-SCP: A requirements analysis framework for validating strategic alignment of organizational IT based on strategy, context, and process," Information and Software Technology, vol. 48, no. 9, pp. 846-868, 2006.

Copyright (c) IARIA, 2012. ISBN: 978-1-61208-230-1