Building a Case for a Dynamic Requirements Process Improvement Model

Aminah Zawedde[*]
School of Computing and Informatics Technology, Makerere University

ABSTRACT

Software requirements management is a volatile and dynamic process that requires continuous requirements process improvement in order to arrive at adequate requirements. However, there is interdependence amongst the processes themselves that leads to requirements managers' inability to achieve satisficed requirements for the intended software system. As a result, several software projects have either a cost, schedule and quality creep or a mixture of them due to poor requirements process improvement (RPI) management. Currently, no attempt has been made to model the impact of the relationships amongst requirements processes to help gain problem understanding while using highly visualized tools. This has been cited by several authors as the reason why the attainment of a satisficed state for the requirement engineering processes cannot be achieved. This research is an attempt to develop a generic model for RPI by triangulating System Dynamics (SD) and Statistical Process Control (SPC) that exploits SD's ability to analyze dynamic complex systems and SPC's highly visualizable time series analyses to achieve satisficed system.

Keywords: Interdependence, Requirements process improvement, system dynamics, satisfication

IJCIR Reference Format:

Zawedde, Aminah. Building a Care for a Dynamic Requirements Process Improvement Model. International Journal of Computing and ICT Research, Vol. 5, Special Issue, pp. 25-33
http://ijcir.org/specialissue2011/article4.pdf

1. INTRODUCTION

In businesses across the world, the need for RPI has never been more critical to ensuring project success. To bring this into perspective, Conetta [2010] argued that the cost of replacing marine one helicopters rising from 6.1 to 11.2 billion dollars was a result of poor communication of the aircraft requirements. Interestingly, a surprising number of companies suffer from ineffective requirements process improvement. Requirements process improvement (RPI) is the process through which changes are made to improve a requirements specification in terms of having a balance amongst cost, quality and schedule variables. Without effective requirements process improvement, business users lack access to timely and accurate information, hindering communication and contributing to delays, errors and excessive rework.

In order to ensure that adequate requirements are arrived at, software requirements managers are always carrying out RPI. However, there is interdependence amongst the processes themselves that leads to requirements managers' inability to achieve satisficed requirements for the intended software system. A

[*] Aminah Zawedde. School of Computing and Informatics Technology, Makerere University. P. O. Box 7062, Kampala. Email: sazawedde@cit.mak.ac.ug

International Journal of Computing and ICT Research, ISSN 1818-1139 (Print), ISSN 1996-1065 (Online), Vol.5, Special Issue, pp. 25-33. December, 2011.

satisficed system is one that meets the criteria for an adequate system but does not identify an optimal solution. For such a system, a few alternatives are compared and the best course of action is chosen from this limited range of options [Simon, 1956]. As a result, several software projects have a cost, schedule and quality (quality triangle) creep or a mixture of them due to inefficient RPI management [Ferreira et al., 2009; Wiegers, 1999].

Previous research attributes this to the fact that the interdependence amongst the requirements processes causes complexity of analysis of these processes. Even with this problem, there is lack of tools that model impact of the relationships amongst requirements processes to help gain problem understanding while using highly visualized tools to prioritize key processes for system development [Williams, 2003; Ferreira et al., 2009]. Despite the attempt by [Ferreira et al., 2009] to capture complex dynamics for requirements resulting from their volatility in a software development project, they did not look into analysis of the impact of interacting dynamic requirements processes and the need to understand the impact of these processes on the software project management. Understanding of these interactions is important because it helps generate policies to achieve statisficed systems.

This research is an attempt to use a triangulation of SD that is known for its ability to handle complex dynamic problems [Forrester, 1991; Williams, 2003] and SPC that uses time series analysis to monitor process performance and identify the process that re- quire improvement and verify the effectiveness of the improvement made [Baldassare et al., 2004]. The triangulation of the two methodologies will help RPI achieve the required satisficed system. The rest of the paper is structured as follows. Section 2 describes the problem area and motivation of this research, while Sections 3 and 4 are a discussion about the state of art and practice of process improvement models and methodologies, Section 5 discusses the proposed solution to the identified problem, and Section 6 gives a summary of the paper and proposes the future work.

2. PROBLEM DESCRITPTION AND MOTIVATION

Ruhe et al. [2002] argue that the hardest part of building a software system is deciding precisely what to build since requirements keep on changing throughout the development process. This creates difficulty in selecting and visualizing the appropriate requirements that fulfill the critical stakeholders' preferences as well as maximization of the quality of the end product [Pfhal and Ruhe, 2003; Ruhe et al., 2002].

Sommerville and Ransom [2005] support this position by arguing that the interaction among requirements engineering processes is dynamic and reveals a non-linear relationship. To analyze this interrelationship amongst requirements engineering (RE) processes in order to achieve process improvements, current RPI tools use work breakdown structure methodologies which are unable to capture and visualize the feedback between or amongst processes [Ruhe et al., 2002; Duggan, 2006]. In reality processes are interlinked and dynamic, and thus should be analyzed in a holistic manner to enable one capture the interdependence and feedback amongst the processes with an aim of visualizing and achieving satisfication for the quality triangle [Beecham et al., 2005; Williams, 2003]. Current methods used do not capture the feedback relationships amongst requirements processes when trying to achieve a satisficing level of the quality triangle. This often results into some processes dominating requirement determination while impacting on other processes [Clempner, 2010; Ferreira et al., 2009; Beecham et al., 2005; Williams, 2003]. In the long run this may yield amplification or decay of the intended system performance hence software project failure or escalated post implementation costs. This is supported by Bendor and Kumar [2009] who advocate for a holistic approach to RPI analysis models.

The use of a standard approach to deal with real-world complexity where models of reality are built and analyzed through visualization has been recommended [Clempner, 2010; Duggan, 2006; Williams, 2003; Pfhal and Ruhe, 2003]. The two candidate methodologies are hierarchical decision process petri-nets [Clempner, 2010] and system dynamics [Forrester, 1991]. Although Petri nets have been used in dynamic modeling of complex systems, such models can best be built using a System Dynamics based methodology, which is a very comprehensive and powerful modeling paradigm for its ability to capture both static and dynamic aspects of reality. Furthermore the methodology should also have the ability to visualize the effect of the changes made in the requirements process to enable the requirements process improvement team have a common understanding of the process. This will result in the team making informed decisions about the appropriate process improvement that should be done. It is on these premises that in this paper we attempt to propose the use of a system dynamics based approach as a process improvement methodology for complex requirements process analysis.

3. STATE OF PRACTICE: CURRENT SOFTWARE DEVELOPMENT PROCESS IMPROVEMENT MODELS

Several software development process improvement models exist namely Requirements Capability Maturity Model (R-CMM) [Beecham et al., 2005], Software Capability Maturity Model (CMM) [Paulk, 1996], Requirements Engineering Process Improvement Model (REPSIM) [Williams, 2003], Bootstrap [Kuvaja, 1999], Trillium [April and Coallier, 1995], and Software Process Improvement and Capability Determination (SPICE) [Dorling, 1993]. However apart from R-CMM and REPSIM, all the models that have been mentioned above are software process improvement models. Review of literature has revealed that the SPI models have both workable and economic potential for software process improvement but they do not adequately address requirements process improvement [Sawyer et al., 1997]. We will therefore discuss in this literature review only R-CMM and REPSIM which are relevant for RPI.

With the approach that R-CMM follows one is unable to exploit the advantage of developing generic processes that can work across all levels of maturity to achieve an optimal system. This is given the fact that in practice, real systems are complex and their RE processes are also complex and dynamic, therefore they interact amongst each other depicting feedback relationships that cut across all levels of organizational maturity with respect to process improvement [Williams, 2003]. In addition R-CMM is heavily dependent on organizational learning and therefore organizations that are at a primary stage of learning may not be able to attain a satisficed state of the requirements processes and thus calls for a methodology that is able to improve its analytical strength as users familiarize themselves with it, irrespective of the organizational level of maturity. Coupled with the above drawback, Ojala [2004] asserts that capability based improvement only, is not enough to start process improvement work. To be effective, software process improvement (SPI) and development methods also need to take into account costs created in processes and thus the need to measure capability and value of processes.

REPSIM is a more promising model for enhancing into a generic tool for RPI for satisficed systems than R-CMM because of its ability to model the RE process in a holistic manner and cutting across all levels of the organizational maturity, but unfortunately it has not been applied in industry. Holistic refers to modeling the process as a whole but not in parts. Williams [2003] holistically modeled the RE process and was able to capture the inter relationships between processes using a tool he developed called REPSIM. REPSIM was used to predict cost, understand the cause and effect between processes and explain the impact of the resulting inter relationship, however, [Williams, 2003] like [Beecham et al., 2005] did not look into a dynamic prescriptive model. A dynamic prescriptive model brings to visibility all the processes interrelating with each other resulting into capture of the best fit which then is able to ensure system satisafication i.e. cost, schedule and quality are satisfied.

Given the strengths and weaknesses of the current RPI models, to successfully achieve a satisficed system, an appropriate RPI model should have the ability to capture feedback and time delay resulting from interaction amongst the RE processes. This will prevent a separation of the model structure from system behavior with an aim of attaining a satisficed state of the system under development.

Table 1 summarizes the characteristics of current tools in use and highlights their deficiencies to meet the requirements of a good tool for a satisficed system.

Table1: Comparison of existing RPI tools with a proposed tool depicting a summary of key essential attributes

RPI Models	Support of interaction with RE Actors	Relationship among attributes	Evaluation and Synthesis	Satisfication	Understand-ability	Explanation	Holism	Generic Tool Development
R-CMM	P	N	P	N	N	N	N	N
REPSIM	F	P	P	N	F	F	P	N
Bootstrap	P	N	P	N	N	N	N	N
Trillium	P	N	P	N	N	N	N	N
SPICE	P	N	P	N	N	N	N	N
Proposed Tool	F	F	F	F	F	F	F	F

KEY: F= Fully Supported; P= Partially Supported; N= Not supported

To address the highlighted gaps of the current models in Table 1, the authors have carried out further literature synthesis in addition to [Williams, 2003; Duggan, 2006; Clempner, 2010]. The literature reveals that only methods that can carry out dynamic modeling for complex systems are appropriate to bridge the gap in RPI modeling for attainment of satisficed systems. Two candidates methodologies have been identified to meet this criteria; System Dynamics as used in REPSIM [Williams, 2003] and Hierarchical Decision Process Petri-nets (HDPPN) [Clempner, 2010]. In the next section we discuss the strengths and weaknesses of both methods and recommend the most appropriate methodology proposed for use in capturing the dynamic complexity of RPI as discussed in the next section.

4. STATE OF THE ART: PROCESS MODELING METHODOLOGIES

System dynamics (SD) and Hierarchical Decision Process Petri-nets (HDPPN) have been commonly used for modeling processes of dynamic complex systems. Although some analysts opt for methods that they are more familiar with, there are authors who have come up with guidance on which approach should be used in a given circumstance [Buchholz, 1994; Duggan, 2006]. The main advantage of both SD and HDPPN is that they are used in analysis of real world systems and can generate insights which are essential in theory building [Duggan, 2006].

Petri-nets and system dynamics are used to model work flows and state transitions but the divergence in their approach and effectiveness lies in the difference in their epistemological underpinnings. While Petri-nets are best utilized when simulating workflow systems are based on discrete event simulation and state- transition, System dynamics are best utilized when modeling workflows that are aggregated and the state transitions are modeled continuously through sets of integral equations for natural systems [Duggan, 2006; Forrester, 1991]. However, to adapt Petri-nets to simulation of complex dynamic continuous systems, hierarchical petri-nets have been developed but still suffer from the drawback of time delay due to transitions occurring at discrete and uneven time intervals [Duggan, 2006]. Petri-nets have also been viewed to be inadequate in analysis of complex systems since they give a single view description on a complex system [Buchholz, 1994] as compared to system dynamics that has the ability to give a three dimensional view of the same system. The improved versions of petri-nets including colored and high level petri-nets that have tried to model complexity become more complicated to analyze as the model gets bigger [Clempner, 201]. Table 2 gives a comprehensive comparison between SD and Petri-nets and highlights the importance of each attribute.

From the analysis in **Table 2**, it is clear that SD is the most appropriate method to address the gap for RPI in the development of satisficed systems.

5. PROPOSED SOLUTION

From the comparison of the existing RPI methods and Process modeling methodologies, SD qualifies as the most appropriate method for enhancement for attaining a satisficed state for software systems under development. SD helps to explain how the feedback-loop relationships amongst the process improvement variables determine the behaviour of the model [Georgantzas et al., 1995]. This will be done using stock and flow diagrams to depict the feedback loop relationships amongst the process improvement variables and how they influence the model behaviour.

In order to augment SD's ability to identify the most influential processes for requirements engineering during system development, SD will be triangulated with statistical

Table 2: Comparison of commonly used process modelling methods for complex dynamic systems

Attribute	Petri nets	System Dynamics	Importance of Attribute	Reference
1. Level of analysis	Only Hierarchical petri nets analyze complex systems.	SD was designed for complex systems	Natural systems are large and complex	Clempner, 2010 Forrester,1961
2. Ease of Model creation	Extra coding after model creation may have to be done to superimpose some model heuristics.	Easy to create all decision rules as equations in a straight forward manner	Use by non experts	Duggan, 2006
3. Analysis of large models	Analysis becomes complex with large models	Techniques exist to handle large models holistically	Natural systems are large and complex	Taylor et al., 2010
4. Model scalability	Difficult to add extra structures to an existing model	Flexibility to add and remove variables and links	Identify variables responsible for system behaviour	Clempner, 2010 Duggan, 2006
5. Update of tables after simulation interval	Special purpose transition is written to record the state of places after each simulation	Automatically recorded by the tool	Ease of use by non experts	Duggan, 2006
6. Simulation runs	Takes long to run; time lags becomes more significant for optimization or satisfication	Runs instantly	Analysis for optimization or satisfication	Duggan, 2006
7. Complex system analysis	Hierarchical petri nets analysis of complex nets performed on reachability set of the net is too large	SD tools have the ability to identify independent loop sets within models	Natural systems are large and complex	Clempner, 2010 Forrester,1961, Buchholz, 1994
8. Link structure to system behaviour	Separates the structure from the behavior	Holistic analysis of model structure to understand system	Attribution of model structures to particular system	Buchholz, 1994 Forrester, 1983 Taylor et al., 2010
9. Modelling complex dynamic systems	Hierarchical petri nets carry out continuous dynamic modeling but suffers a time delay due to transitions occurring at discrete and uneven time intervals	Continuous dynamic analysis, holistic and dynamic complexity	To capture every instant of system behavior	Duggan, 2006

process control (SPC) [Georgantzas et al., 1995]. SPC is a statistical based approach that is well known for its ability to determine the stability of a process by discriminating between the presence of common cause and

assignable cause variations [Baldassare et al., 2004]. Triangulation of SD and SPC has been done for manufacturing processes but not yet in the requirements and software process context. Due to the primary differences between manufacturing and software or requirements processes as highlighted in [Baldassare et al., 2004], the combination of SD and SPC cannot be used as is but must be tailored to fit the requirements process. Combining SD and SPC will not only help the RE stakeholders to gain insights about the RPI feedback structure but also carry out assessment of the potential effects of the high leverage points on the stability of the interacting requirements processes directly from the control charts leading to attainment of a satisficed state for a software system under development. Furthermore, the RE stakeholders will have ownership of the RPI model together with its results, and the urge to implement the changes recommended by the model [Georgantzas et al., 1995]. This is a result of understanding the causes of the process variations and being able to deal with these variations most appropriately [Georgantzas et al., 1995]. SPC will help in enhancing visualization while identifying the most influential processes for its ability to determine satiscification limits dynamically, hence ease of use.

This will be done by examining the system dynamic's model outputs and describing the evolution of exogenous impacts on model behavior over the course of a simulation. To be able to quantify this, control limits will be set using SPC and process performance will be tracked over a time period on a control chart. If there are any processes that fall outside the control limits, then these will be the points to be considered for RPI. "Control charts either suggest that corrective action is needed to improve quality or give assurance that a process is producing satisfactory quality" [Georgantzas et al., 1995]. The results will be used to link the high-leverage parameters to specific model structures that can then be used to improve the understanding of how the model structure impacts on system behavior. Furthermore, the model will be used as a training tool and a requirements process improvement guide for the concerned RPI stakeholders. The model will also foster improved understanding of the requirements process improvement field.

A review of the RE process variables has been done and variables suited for RPI have been identified. Table 3 highlights the key variables for achieving a satisficed state.

Table 3: Process Improvement variables for the research dynamic hypothesis

PI Model Sectors	Process Improvement Variables	Process Improvement Model Outputs	Reference Mode	Process Measure
1. Project scope	Process Improvement Capability	Change in process improvement	Paulk, 1996	{scale 0-1}
2. Project Finance	Process Costs	Change in process costs	Loncosole, 2001	{currency}
3. Process technology	Process effectiveness	Change in process effectiveness	Williams, 2003	{scale 0-1}
4. Development process	Process Capability Level	Change in maturity level	Winton, 1999	{scale 0-1}
	Process productivity of requirements engineers	Change in productivit y of	Berenbach, 2004	Number of requirements per person per
5. Support environment	Process Rigor	Change in rigor	Williams, 2003	{scale 0-1}
6. Process schedule	Perceived effectiveness	Change in perceived effectiveness	Williams, 2003	{scale 0-1}
7. Quality Assurance	Errors Observed	Time to detect errors	Winton, 1999	{Number of errors}
8. Procurement and Contract	Customer Satisfaction	Change in customer satisfaction	Winton, 1999; Starz,2005;	{scale 0-1}

To depict the interrelationship between the process attributes mentioned above, a dynamic hypothesis has been developed as shown in Figure 1. The research dynamic hypothesis that shows the interrelationships amongst the requirements process improvement variables and their impact on the quality triangle in trying to attain a satisficed state for the system under development has been formulated from reviewed literature. The dynamic hypothesis is the qualitative representation of the proposed system dynamics based requirements process improvement model and it will be tested using field studies that are the next steps to the work done so far.

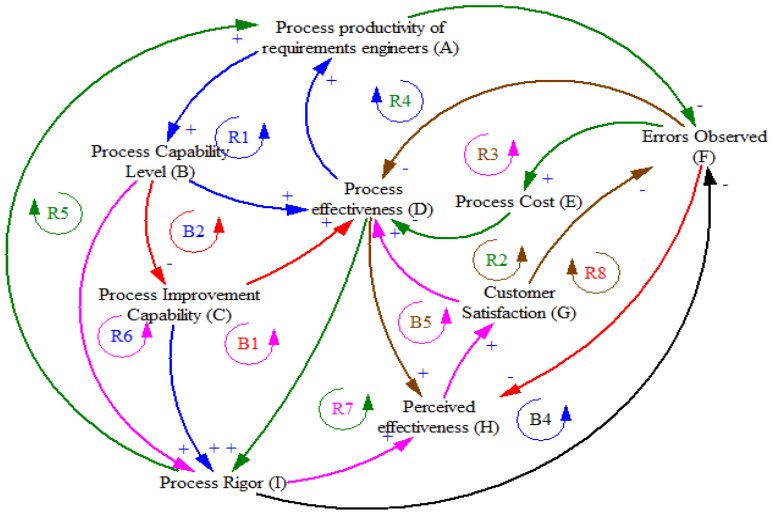

Fig. 1: Dynamic Hypothesis for the Requirements Process Improvement Model.

The results of the field studies will confirm or reject the hypothesis and will help modify the dynamic hypothesis into a detailed causal loop diagram. The dynamic hypothesis in Figure 1 above has 4 balancing loops and 8 reinforcing loops. A reinforcing loop (R) represents growth or declining actions while a balancing loop (B) is a goal seeking loop that seeks stability or return to control [Williams, 2003]. An example of one of these loops is described as follows. B1 (going through B-I-H- G-D-A-B) is a balancing loop where increasing the process capability level will cause a reduction in process improvement capability with an immediate increase in process rigor and thus almost immediate gain in perceived effectiveness that will increase customer satisfaction and process effectiveness. This feeds back into increasing process productivity of requirements engineers.

For continuous process improvement in the reinforcing loops, they have to be balanced by the balancing loops to avoid causing a system burnout or chaos. The dynamic hypothesis is expected to evolve overtime as more relationships are explored and understood. This research will study the various loop structures using system dynamics modelling to identify the combination of loop sets that help achieve a satisfication state to ensure continuous process improvement.

6. CONCLUSION AND FUTURE WORK

The contribution of this paper has been a literature survey that helped to identify the gap that exists with current requirements process improvement tools in capturing the feedback interaction amongst software engineering requirement processes. In addition this paper also proposes how this gap can be closed by proposing a unique solution of reaching a satisficed state to attain a balanced system as opposed to optimization that may not address certain requirements. This is work in progress and as immediate future work, the dynamic hypothesis will be tested in the field to confirm and or improve it based on what is in practice. Simulation models will then be built to explicitly quantify the relationships amongst the variables of the dynamic hypothesis. Quantification of the simulation models is done using mathematical equations to help capture the behaviour of the variables of requirements process improvement.

REFERENCES

APRIL, A. A. and COALLIER, F.: Trillium V3.0: A Model for the Assessment of Telecom Software System Development Capability. In: 2nd International SPICE Symposium ASQRI, Griffith University, Australia, 79-88, (1995)

BALDASSARE, T., BAFFOLI, N., CAIVANO, D., VISAGGIO, G.: *Managing Software Process Improvement (SPI) through Statistical Process Control* (SPC). In: Bomarius, F. and Iida, H. (Eds.): PROFES, LNCS 3009, 30-34, (2004)

BEECHAM, S., HALL, T., RAINER, A.: Defining a Requirements Process Improvement Model. Software Quality, 13, 247-279 (2005)

BENDOR, J.B., KUMAR, S.: Satisficing: A Pretty Good Heuristic. The B.E. J. Theoretical Economics, 9(1), Article 9 (2009)

BERENBACH, B. A.: Comparison of UML and Text based Requirements Engineering. In: *Proceedings of OOPSLA'04, Vancouver, British Columbia, Canada,* 24 – 28 October 2004, (2004).

BUCHHOLZ, P.: Hierarchical High Level Petri Nets for Complex System Analysis. In: *Proceedings of App and Theory of Petri Nets, 119 – 138, (1994).*

CLEMPNER, J.: A Hierarchical Decomposition of Decision Process Petri nets for Modeling Complex Systems. Int. J. Applied Mathematics and Computer Science, 20(2), 349-366 (2010)

CONETTA, C.: An Undisciplined Defense: Understanding the $2 Trillion Surgein US Defense Spending. Project on Defense Alternatives, Commonwealth Institute, Cambridge, 20, (2010)

DORLING, A.: SPICE: Software Process Improvement and Capability Determination. *Software Quality Journal, 2(4), 209-224, (1993)*

DUGGAN, J.: A Comparison of Petri Net and System Dynamics Approaches for Modelling Dynamic Feedback Systems. In: *24th International Conference of the System Dynamics Society, Nijmegen, The Netherlands, (2006*)

FERREIRA, S., COLLOFELLO, J., SHUNK, D., MACKULAK, G.: Understanding the effects of requirements volatility in software engineering by using analytical modeling and software process simulation. Systems and Software, 82(10), 1568-1577 (2009)

FORRESTER, J.W.: System Dynamics and a lesson for 35 years. Sloan School of Management M.I.T, Cambridge, MA, USA (1991)

GEORGANTZAS, N., MOJTAHEDZADEH, M.: Rendezvous Dynamics. 6th European Conference of Science Systems (2005)

GEORGANTZAS, N., FRASER, J., TUGSUZ, E.: Bipartisan process improvement in polymer coating: Combining system dynamics with statistical process control (SPC). System Dynamics, 2, 502-511 (1995)

KUVAJA, P.: BOOTSTRAP 3.0-A SPICE1 Conformant Software Process Assessment Methodology. Software Quality Journal, 8: 7-19, (1999)

LOCONSOLE, A.: Measuring the Requirements Management Key Process Area. In: *Proceedings of ESCOM, London, April, (2001)*

OJALA, P.: Combining Capability Assessment and Value Engineering: A BOOTSTRAP Example. In: *5th International Conference of Profes, Kansai City, Japan, (2004)*

PAULK, M.C.: Effective CMM-Based Process Improvement. In: 6th *International Conference on Software Quality, Ottawa, 119 – 138, (1996)*

PEDERSON, J.: Assessing the Value of Process Improvement. Software Process Improvement Network, (2006)

PFHAL, D. and RUHE, G.: IMMoS: A Methodology for Integrated Measurement, Modelling, and Simulation. Int. J. Software Process Improvement and Practice, 7, 189-210 (2003)

PFAHL, D, LEBSANFT, K.: Using Simulation to Analyse the Impact of Software Requirement Volatility on Project Performance. Information and Software Technology, 1001-1008, (2000)

RUHE, G., EBERLEIN, A., PFHAL, D.: Quantitative WinWin- A New Method for Decision Support in Requirements Negotiation. In: *SEKE, Italy, July, (2002)*

SAWYER,P., SOMMERVILLE, I., VILLER, S.: Requirements Process Improvement Through the Phased Introduction of Good Practice. In: *Software Process - Improvement and Practice, 31–44, (1997)*

SIMON, H. A.: Rational choice and the structure of the environment. Psychological Review, 63, 129-138, (1956).

SOMMERVILLE, I., RANSOM, J.: An Empirical Study of Industrial Requirements Engineering Process Assessment and Improvement. ACM Transaction on Software Engineering and Methodology, 14(1), 85-117, (2005)

STATZ, J.: Measure of Process Improvement. Technical Report on Practical Software and Systems Measurement, (2005)

WIEGERS, K.: Automating Requirements Management. Software Development, 7(7), 1-5, (1999)

WILLIAMS, D.: Challenges of System Dynamics to Deliver Requirements Engineering Projects: faster,

better and cheaper. In: *21st Int. Conf. of the System Dynamics Society, (2003)*