System Dynamics for a mathematical modeling approach in Software Engineering and Information Systems

Ricardo Jaime
Grupo de Investigación en Tecnologías de la Información aplicadas a la Educación
Universitaria de Investigación y Desarrollo
Calle 9 # 23-55, 680001, Bucaramanga, Colombia
Telephone + 57 7 6352525
ricardojaime@udi.edu.co

Abstract

Supporting material to this paper comes from a research project in the use of System Dynamics (SD) modeling as mathematics for engineering education; but the author's interest is not only on mathematics for engineering education itself, but about what else to do with SD in a problematic context: first, traditional SD with simulation purposes no longer arouses the same interest as before in computing undergraduate programs in Colombia; second, those computing undergraduate programs have decreased in number of applicants, probably because of a socially constructed idea about this kind of professional are not really engineers nor really deal with major problems. Results of this projects are auspicious: students significantly improved their SD modeling competencies, and some evidence was collected showing they become interested about a possibility of using SD as the basis for the design of software and information systems, whereby such computing projects can become implementations of SD models different from assessment or consultancy, oriented to help policy makers, but also to reinforce policies with tools people can use to be aware, learn and participate, and also to get software engineering students more aware about how they can contribute in solving social problems, especially those related with sustainability.

Keywords: Engineering education, Model implementation, Software Engineering

INTRODUCTION

There is a decrease in applicants to study computing careers; universities in Colombia, like in many other countries, are looking for explanations and solutions. One of this explanations, or at least a conjecture, considered at Universitaria de Investigación y Desarrollo UDI, where this paper comes from, suggests that probably people do not want to study computing because they do not believe that its impact had been significant for the solution of the most important problems of society; may be computing in Colombia has been confined to solve important problems (accounting, inventory, staff management) that are simply internal to firms, but not social like, for example, how can society and individuals live sustainably.

A research project was the opportunity to explore how to give students at UDI a more interesting idea about computing, how to teach and learn computing in a rigorous engineering framework, and consequently how framing computing in the aim of solving social problems and not only information processing problems of companies. System Dynamics (SD) course at fifth year of a computing undergraduate program a UDI was the scenario to prove a methodology for making a new way to implement SD models through software (Jaime, 2012), whereby students could board such social problems, beyond just information ones, formulate requirements and design software oriented to give solutions to those, and to expand their professional scope. There was a formal research question for the project about how to improve students' competences in mathematical modeling; but in the teacher's own agenda as SD practitioner, one of the research questions was ¿what else to do with SD?

THE COMPUTING UNDERGRADUATE PROGRAM

The mentioned project was executed in the context of a call for research projects of the National Academic Network of Advanced Technology RENATA, a network conformed by higher education institutions in Colombia.

The experimental phase of the project was carried out with students in a SD course, corresponding to the ninth semester in curriculum of Systems Engineering at Universitaria de Investigación y Desarrollo UDI, in Bucaramanga, Colombia.

It can be noticed Systems Engineering was mentioned in last paragraph. A necessary clarification about it will be found in next section. Then some considerations will be made about math skills required in Software Engineering, their differences from the ones required in other engineering branches, and particular educational challenges derived from these. Next, it will be exposed why a SD course was selected for experimentation and how it is expected that SD modeling help to retrieve the relevance of mathematics to Software Engineering students. Later an overview of the research project, methodology, results, and conclusions will be presented.

About Systems Engineering in Colombia, it is pertinent to note that this denomination is being questioned since it was introduced by 1967, based on the curriculum of Computer Science of University of Pennsylvania (Estrella, 2010); remains a controversy about its meaning and social understanding, and there are experts that ask universities to engage the international trend which has derived five different disciplines from the general concept of computing (Gallardo, 2010). However, the name has achieved such a tradition, that some institutions have offered similar programs with more specific denominations such as Informatics Engineering, but they have not found a significant number of applications for admission (Caro, 2010).

According to Universidad de los Andes, Systems Engineering is a profession dedicated to create and build solutions that benefit society, information being the raw material of the work of Systems Engineer, that is oriented to represent, store, transform, communicate, interpret, show and operate information in safe way ensuring its quality and accuracy (Universidad de los Andes, 2013).

Universidad Nacional de Colombia defines Systems Engineering as modeling, development and implementation of complex systems through the application of mathematics and computer science, specifically mentioning within the latter the theories of information, computational complexity and programming languages, computer programming and systems theories (Universidad Nacional de Colombia, 2013).

Systems Engineering has been defined at Universidad Industrial de Santander as an eclectic mix of Computer Science, Operations Research, Control Engineering and Systems Engineering, a profession dedicated to solve problems through information systems, databases, data networks and software engineering, being its objects of study information and knowledge, and the work of engineer consisting in representation, processing, storage and transmission of information and knowledge (Universidad Industrial de Santander, 2013).

In the context of computing curriculum proposed by ACM and IEEE (Association for Computing Machinery ACM & Institute of Electric and Electronic Engineers - Computer Society IEEE-CS, 2005), from the above referenced definitions given by three of the most important universities in Colombia, it could be argued that the denomination of Systems Engineering established in Colombia is really a hybrid between Software Engineering and Information Systems. Once it was done such clarification, in the remainder of this paper only the term Software Engineering will be used to refer the computing undergraduate program where this research was developed.

MATHEMATICAL MODELLING IN ENGINEERING

Mathematical modeling is an inherent competence to engineering, defined as the discerning application of knowledge in mathematics and science, to determine the use of the materials and forces of nature for the benefit of mankind (Accreditation Board for Engineering and Technology ABET, 2011), or as a discipline rooted in mathematics, physics and other natural sciences, applied to the development of models and methods for solving problems (European Network for Accreditation of Engineering Education ENAEE, 2013).

ABET has included the following areas of mathematics in accreditation criteria applicable to 26 engineering programs: Differential Equations, Probability and Statistics, Physics based Calculus, Differential and Integral Calculus, Discrete Mathematics, Multivariate Calculus, Linear Algebra, Complex Variables. As it can be seen in Table 1, for Electrical Engineering programs, six of these areas are determined as needed; most programs require between 2 and 3, and 12 programs are not specified but referred to mathematical models related to basic sciences applicable on problems relative to each discipline. For Software Engineering is required competence in Discrete Mathematics and Probability and Statistics.

Discrete Mathematics are the basis of all computing field, including Software Engineering, as Calculus and Differential Equations are the basis to other engineering branches. While most engineering branches use mathematics to model and produce physical artifacts, Software Engineering produces intangible artifacts (Henderson, 2003). As a recommendation it is stated that Software Engineering students should learn "in reasonable depth" mathematics related to software application domain, which may be found in other engineering disciplines, natural sciences, social sciences, humanities and business, among others (Association for Computing Machinery ACM & Institute of Electric and Electronic Engineers - Computer Society IEEE-CS, 2004). In spite of Calculus, Differential Equations and Linear Algebra are often included in curriculum of Software Engineering, their purpose is not direct application to a certain type of problems, but merely to promote abstraction ability.

Table 1. Mathematics required by engineering programs

Mathematics required by engineering programs	Non specific	Differential equations	Probability and statistics	Calculus-based physics	Differential and integral calculus	Discrete mathematics	Multivariate calculus	Linear algebra	Complex variables	Total
Electrical, Computer		X	X		X	X		X	X	6
Geological		X	X	X						3
Mining		X	X	X						3
Construction			X	X	X					3
Environmental		X	X	X						3
Software			X			X				2
Biomedical		X	X							2
Mechanical		X					X			2
Civil		X		X						2
Architectural		X		X						2
Petroleum		X	X							2
Chemical, Biochemical, Biomolecular	X									1
Engineering Mechanics	X									1
Naval architecture, Marine			X							1
Aerospace	X									1
Materials, Metallurgical	X									1
Ocean	X									1
Engineering Management	X									1
Agricultural		X								1
Nuclear, Radiological	X									1
Ceramic	X									1
Manufacturing	X									1
Fire protection	X									1
Surveying	X									1
Biological		X								1
Industrial	X									1
Total	12	11	9	6	2	2	1	1	1	45

LACK OF INTEGRATION BETWEEN MATHEMATICAL MODELING AND SOFTWARE ENGINEERING

According to international recommendations, Colombian universities usually include in Software Engineering curriculum up to 3 courses in Discrete Mathematics, Probability and Statistics, and up to 6 courses in Calculus, Linear Algebra and Differential Equations. But as it occurs in other countries, there is a lack of integration between mathematical foundations and Software Engineering learning and practice: neither students find motivation to study

Discrete Mathematics in corresponding courses nor they are encouraged to apply them in Software Engineering courses (Cohoon & Knight, 2006), because the former are centered in solving problems without Software Engineering context, and the latter seems to be more related with best practices of software development, especially documentation, rather than its mathematical foundations.

Discrete Mathematics in software engineering is applied in design and optimization of algorithms and software features that are independent of its application domain. Usually for requirements determination it is assumed that these will be given by other agents to the software development team (International Council on Systems Engineering INCOSE, 2009). This leads software developers to lose interaction and to reach only a not enough understanding of the problem domain (Luna-Reyes, Black, Cresswell, & Pardo, 2008), and affects their ability for interdisciplinary work, minimizes their participation in the phase of requirements determination, and do not let them to assume responsibility about the transformation of the social order based on the use of computer systems, which is not only related with the way such systems are produced and distributed, but also how they are appropriated by society (Kling, 1991) because unlike other engineering developments that can be completely defined since requirements are determined, computer products go through different versions throughout their life cycle, they are socially constructed, and developers should study carefully the social and organizational contexts of their design and use (Kling, Rosenbaum, & Sawyer, 2005).

If in the case of Discrete Mathematics that are considered fundamental to Software Engineering is worrisome, the problem becomes more severe with continuous mathematics, the study of which typically addresses phenomena and problems in which the Software Engineering students are not directly concerned or familiar. That disjunction between theory and practice, and especially the low perceived relationship between mathematical definitions and design competence, increases the lack of metalinguistic awareness that should characterize a discursive activity such as modeling data and information (Holmboe, 2005), which could be overcome if students and teachers focus not only in solving problems, applying formulas and proving theorems, but in looking for as a desirable competence in a math course, the improvement of technical language use, with which the engineering student can express precisely in a mathematical way what may be ambiguous or confusing when expressed in everyday language (Khait, 2003), or inaccurate when represented with technical languages like Unified Modeling Language UML, because despite its usefulness in the analysis and design of software, do not represent the structure and behavior of any software application domain (Tignor, 2004).

¿ WHAT ELSE TO DO WITH SYSTEM DYNAMICS?

In Colombia, System Dynamics is usually incorporated in modeling and simulation courses in undergraduate curriculum of Software Engineering. But interest in simulation has decreased because it is not identified as a required skill according to kind of software they use to develop; only minority of software engineering students remains interested in computer based simulation, those who are engaged in control systems or scientific software development.

At UDI it was considered a possibility to use SD with other purposes like integration of knowledge for software requirements determination, by interdisciplinary teams in which software engineer could exert leadership. This proposal is relevant in addressing problems in which experts in various disciplines converge, not directly related to the knowledge of software developers.

The course joined to this project was previously the subject of methodological variations from the traditional SD modeling process, which became it in a mathematical modeling tool that serves as a bridge between the process of learning about the dynamics of a phenomenon and software design to intervene in it, contributing to increase the ability of software engineers to participate in interdisciplinary workgroups for constructing a better knowledge about problems that can be modeled with mathematical rigor, and for transforming SD models in software design models.

The alternative modeling process with System Dynamics implemented in the course consists of the following steps: 1) problem formulation; 2) qualitative modeling of the basic structure of the system through influence diagrams, including only substantives and verbs; 3) quantitative modeling consisting of differential equations for substantives and auxiliary expressions from other mathematical areas for verbs; 4) enriched modeling of system by adding new elements found during validation, especially dimensional validation, of quantitative model; 5) transformation of qualitative and quantitative system dynamics models in software design: influence diagrams transformation into class diagrams and relational diagrams (Jaime, 2012). This process takes advantage of the correspondence that can be established between substantives and verbs in the verbal approach, with variables and derivatives in mathematical representation and with attributes and methods in object oriented programming.

That way it was intended to improve integration between mathematics and Software Engineering, by demonstrating to students that organizations and processes for which information systems are supposed to be developed, can be represented with mathematical models from which designs and implementation of software can be derived. Process of building a model from conceptualization to simulation is a way to check consistency of requirements and design of software.

Even with this more favorable context for Software Engineering students to find a link between mathematical modeling and software development, they still had weakness in using mathematical skills and knowledge they supposedly should have learnt in previous levels, like arithmetic, geometry, algebra, calculus, and differential equations, all of which may be used in constructing SD models.

Modeling on sustainability

A shrinking world is clearly a problem about which, even if people do not know what to do to solve it, there is a generalized awareness: probably world is arriving to a no return point, and every discipline must act. SD has worked hard to produce models about a wide variety of topics related to sustainability. Students were encouraged to search for cases and teacher presented some others like those mentioned below, in the aim of build trust they were going to be involved in real world big problems, and they as software engineering students have many things to do about it with SD.

Wils (1998) proposes a simulation model by which determined that although the technology has gotten ever improving efficiency in the extraction of non-renewable natural resources, for the sustainability of the system is relatively more important to improve efficiency in its final use, although the optimal strategy is a combination of both.

Martínez Fernández & Esteve (2004) modeled irrigated farm land dynamics; with the model they demonstrated the adverse impact of incorporation of new land to exploitation on water availability for other uses, and also the impact of pollution of land and water with waste and nutrients carried by artificial water currents; they showed in the case the mistakes and low sustainability of focus on establishing policies for irrigation systems.

Dudley (2004) on a system of logging in Indonesia and the ban on export of raw timber (ban of log exports), concluded that the effects of the ban were not sufficiently favorable to the conservation of timberlands, due to market was set inner with more processing by the low price of wood. Jones, Seville, & Meadows (2002) modeled commodities production based on natural resources (timber natural forests also), emphasizing one of the undesirable behaviors that arise in these systems: unsustainability of resources; the two other unwanted behaviors were the price instability and social inequality along the production chain.

Arquitt, Xu, & Johnstone (2005) modeled a exploitation of farmed shrimp that after a time of booming business suffered an associated depletion of the natural environment's ability to support the exploitation fall

Taylor, Ford, Yvon-Lewis, & Lindquist (2011) they showed by modeling stratospheric ozone depletion, that potential of science, engineering and technology to mitigate the adverse effects that society has inflicted on the environment and have grown to become threats to society itself, success depends on speed with which public policy makers focus their attention on the problem, which is useful for modeling with the purpose of providing feedback information to base policy decisions; modeling with only purpose of knowledge building does not have good enough effect, if the experts do not incorporate the models and they exert some level of influence over the policy makers.

Importance and influence of models and modelers

Studying cases like those mentioned, between others, motivated students to think about SD as a modeling discipline able to tackle problems wider than the usual in their professional context. But it was necessary also to introduce students in a debate about traditional and new intervention ways with SD.

Sustainability is a useful concept to make possible to live in such a shrinking world. Its interdisciplinary approach requires SD to play a role. However, although more than fifty years of SD advances, it still remains in effect a controversy on the real influence of such an enormous production of models and simulations. Beyond so uncomfortable that controversy could appear, SD practitioners could also find ways to formulate new SD implementation practices, to give people tools to have chances to take better decisions than those that have brought the world to its current shrinking situation

Meadows & Robinson (2002) shed some lights: the dispute is between those who consider SD modelers important people and SD models effective intervention tools, and those who believe that neither the ones nor the others have been influential enough in high level decisions. Of course, some modelers claim their own success as consultants or group model building facilitators for governments or businessmen, but a question remains: is it enough influential? Following Meadows & Robinson yet, some attitudes like that suppose certain statu quo: SD clients are those who make national level decisions and policy; people are finally asked to accomplish policy issued from top; in that sense, models can be maintained as models.

To Größler (2007) , implementation as a step in SD modeling process needs investigation about low impact of SD projects; in a set of research problems, he mentioned the looking for tools and methods of organizational intervention with which system dynamics can be combined. Models may be good tools for policy makers but may be not so good for policy followers, that is to say, for people. Probably SD models, and not only simulation models, have to be transformed in a different kind of tools.

In this last sense can be situated a former work by Acharya & Saeed (1996); they made changes to the 'Limits to growth' models (World 2); original model generated a good understanding of the phenomenon of using limited natural resources, but in the improved one authors incorporated operational elements that allow its use in the development of public policies, in their own words, aimed to influence the motivations of the actors and thereby guiding their day-by-day decisions

Continuing a line of work outlined by Dana Meadows, on the incorporation of system dynamics as a tool to encourage public participation in political decisions, Stave (2002) identifies five advantages of SD (and simulation models once they have been tested) to improve public participation in political decisions: focus on the problem and not the solution (this would be important during the project to formulate software requirements not from an interview with stakeholders but from a model building process), find the causes of the problem in the structure of the system, determine policy instruments to influence system behavior, and generate feedback information for learning and policy design. In the context of the SD course in a software engineering undergraduate course, it was in the interest of teacher engage students according to an idea referenced by Stave (2002): "Dana Meadows believed that computer simulation models and systems thinking could be powerful tools for democracy, helping make social decisions and the assumptions on which they are based more transparent and open to public debate. She also believed that people should be more involved in making conscious and informed choices about their future [… SD modelers should] empower others to act on the best possible information by making all information concise, clear, compelling, and as truthful as possible. Provide feedback that makes decision-makers accountable and helps people act in ways that promote the things they value". May be software and information systems could be vehicles to implement SD models in that way.

SD modeling in the project

In an increasingly complexity sequence, not about problems themselves but about the scope of their correspondent models as the project progressed, models were built about: invasive species; fruit processing including both industrial component as the waste reuse and disposal; production of dairy products including biological agents dynamics; urban forestry management and social controversies between the conservationist viewpoint but also public policy ones to counterbalance them; solid waste management and social controversies about landfills.

During the final stage of the semester, when constructing models students became more interested in controversies, but also in how they used to be sterile, especially when given between polarized viewpoints, without systematic information about which discuss. They began from the models to propose requirements for software and information systems that make government agents able to include scientific information in policy making, and people able to understand and to participate in policy implementation. Methodology to transform influence diagrams in software design was helpful to expand students' interest in software development not only to reach requirements determined by others, but to imagine software engineering and information systems engineers participating in model building and then helping to building policies and developing software to make policy sustainable.

Initially small models (Ghaffarzadegan, Lyneis, & Richardson, 2011) were built to represent basic structures of system, like the one showed in Figure 1 about urban forestry. For the purpose of this paper, this diagram is not important because the elements included, but because some features used to facilitate comprehension about different kind of influences in the basic structure: green and red arrows represent influences from verbs that increases or decreases substantives respectively; blue continue arrows are influences from substantives to their own verbs; blue dotted arrows represents influences between subsystems; verbs are determined by the same units of measure than substantives, by relative to time.

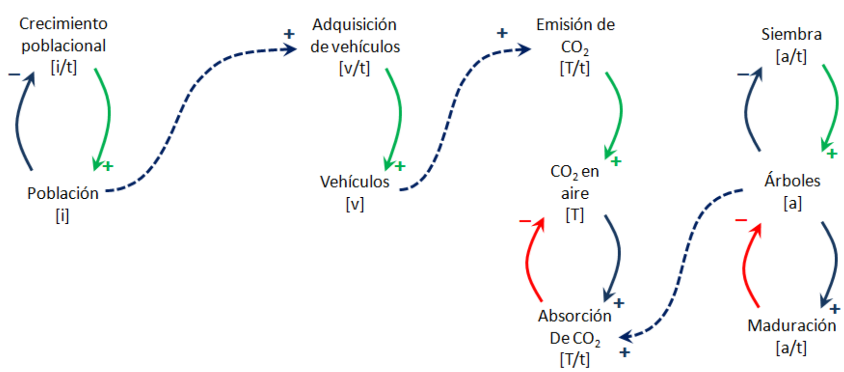

Figure 1 Influence diagram of the basic structure of a system

For each case, the first model was validated in its dimensional consistency as some of good practices compiled by Martínez-Moyano & Richardson (2013). Then a mathematical model was constructed, rigorous but with no mediation of a stock and flow diagram; that was the reason to include units of measure in influence diagram. Usually when validating equations for verbs, especially those that are influenced by other subsystems, that is, by structures whose elements have different units of measure, it becomes necessary to include new elements to get dimensional consistency of the equation. With these new elements as parameters, the influence diagram was expanded according to mathematical model as shown in Figure 2.

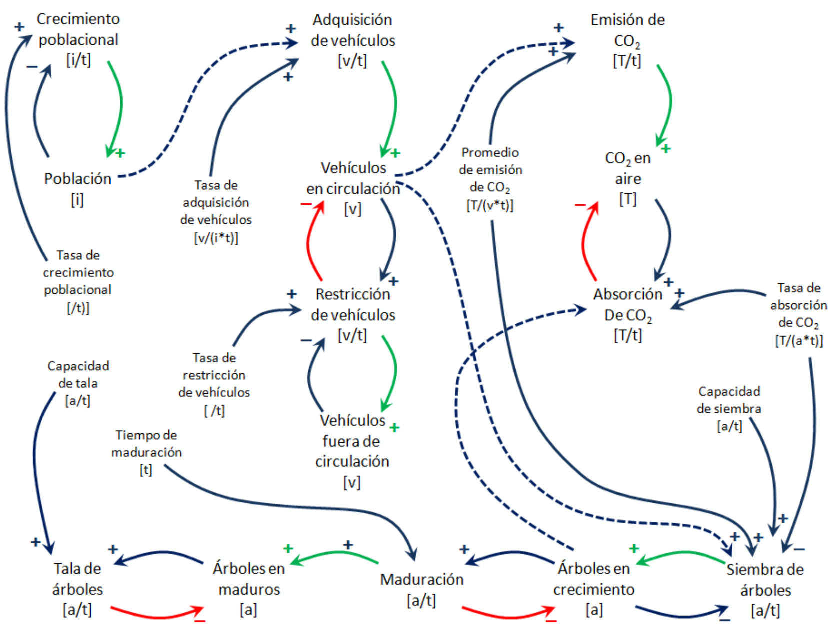

Figure 2 Enriched influence diagram of a system

Finally, mathematically improved influence diagram was transformed in a class diagram (Jaime, 2012), that is to say, in a software design: subsystems are classes, substantives are attributes, verbs are methods and parameters are also attributes.

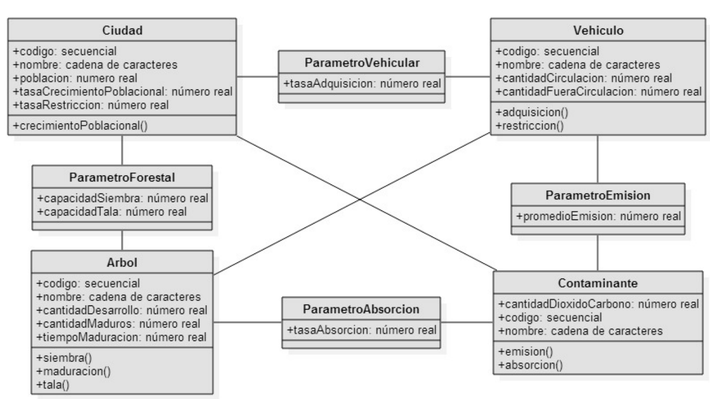

Figure 3 Class diagram obtained from influence diagram transformation

In consequence, students became aware that several things, important for the SD agenda at UDI: SD can be used for some else to simulation; software can be built as a engineering product especially regarding to mathematical rigor; and software and information systems can be another way to implementation of SD models in organizations.

DISCUSSION AND FURTHER RESEARCH

About the conclusions of the project, favorable impact of SD as mathematics for software engineering students were documented (Jaime & Lizcano, 2015); those students' mathematical modeling competencies were improved in significant amount; these results may be not in the scope of this paper but have to be mentioned to guarantee the project was developed with scientific rigor. For the author is more interesting, since his own agenda as SD practitioner, to present evidence that the transformation of SD models in software and information systems designs can become a vehicle for different implementations of SD models seeking public participation in policy and perhaps in achieving greater sustainability in the idea to think globally and act locally. A new research project in that sense is being proposed now at UDI to validate this evidence.

ACKNOWLEDGMENT

This work was made possible by research groups GIDSAW and FIELDS at Universitaria de Investigación y Desarrollo UDI in Bucaramanga, Colombia, to which authors belong. The projects mentioned also involved researchers from groups TECNICE and TECNIMAT at Fundación Universidad Central in Bogotá, Colombia, COGNITEK at Universidad Pedagógica Nacional in Bogotá, Colombia, and KISHURIM at The Hebrew University of Jerusalem, under the direction of Professor Ph.D. Luis Facundo Maldonado. The participation of the UDI and researchers in the project as well as the inclusion of System Dynamics in it was made possible by the institutional support from the Research Directorate by Eng. Armando Arevalo. Thanks to all these individuals and institutions.

REFERENCES

Accreditation Board for Engineering and Technology ABET. (2011). *Criteria for Accrediting Engineering Programs* (p. 24).

Acharya, S. R., & Saeed, K. (1996). An attempt to operationalize the recommendations of the “Limits to growth”study to sustain the future of mankind. *System Dynamics Review*, *12*(4), 281–304.

Arquitt, S., Xu, H., & Johnstone, R. (2005). A system dynamics analysis of boom and bust in the shrimp aquaculture industry. *System Dynamics Review*, *21*(4), 305–324.

Association for Computing Machinery ACM, & Institute of Electric and Electronic Engineers - Computer Society IEEE-CS. (2004). *Software Engineering 2004 Curriculum guidelines for Undergraduate Degree Programs in Software Engineering* (p. 135).

Association for Computing Machinery ACM, & Institute of Electric and Electronic Engineers - Computer Society IEEE-CS. (2005). *Computing Curricula 2005* (p. 62).

Caro, G. (2010). La imagen de la Ingeniería de Sistemas desde la perspectiva de los dirigentes de organizaciones del sector informático del país. *Sistemas*, (144), 106–123.

Cohoon, J., & Knight, J. (2006). Connecting Discrete Mathematics and Software Engineering. *Proceedings. Frontiers in Education. 36th Annual Conference*, 13–18.

Dudley, R. G. (2004). Modeling the effects of a log export ban in Indonesia. *System Dynamics Review*, *20*(2), 99–116. doi:10.1002/sdr.288

Estrella, P. (2010). Ingeniería de Sistemas y Computación: una utopía realizada en la Universidad de los Andes. *Revista de Ingeniería Universidad de Los Andes*, (32), 126–130.

European Network for Accreditation of Engineering Education ENAEE. (2013). ENAEE / IEA Glossary of Terminology. Retrieved from http://www.enaee.eu/publications/enaeeiea-glossary-of-terminology

Gallardo, S. (2010). El Ingeniero de Sistemas: “Alguien que ve mucho más allá del estrecho entorno de una sala de cómputo.” *Sistemas*, (114), 8–11.

Ghaffarzadegan, N., Lyneis, J., & Richardson, G. P. (2011). How small system dynamics models can help the public policy process. *System Dynamics Review*, *27*, 22–44.

Größler, A. (2007). System dynamics projects that failed to make an impact. *System Dynamics Review*, *23*(4), 437–452.

Henderson, P. B. (2003). The role of modeling in software engineering education. *Proceedings. Frontiers in Education. 33th Annual Conference*, 21–25.

Holmboe, C. (2005). Conceptualization and Labelling as Cognitive Challenges for Students of Data Modelling. *Computer Science Education*, *15*(2), 143–161.

International Council on Systems Engineering INCOSE. (2009). *Graduate Software Engineering 2009 (GSwE2009) Curriculum Guidelines for Graduate Degree Programs in Software Engineering* (Vol. 2009, p. 124).

Jaime, R. (2012). Modelamiento Semántico con Dinámica de Sistemas en el proceso de desarrollo de software. *RISTI Revista Ibérica de Sistemas Y Tecnologías de La Información*, (10), 19–34.

Jaime, R., & Lizcano, A. (2015). ICT mediated collaborative work in system dynamics learning. *DYNA*, *82*(189), 59–67.

Jones, A., Seville, D., & Meadows, D. (2002). Resource sustainability in commodity systems: The sawmill industry in the Northern Forest. *System Dynamics Review*, *18*(2), 171–204.

Khait, A. (2003). Goal orientation in mathematics education. *International Journal of Mathematical Education in Science and Technology*, *34*(6), 847–858.

Kling, R. (1991). Computerization and Social Transformations. *Science, Technology & Human Values*, *16*(3), 342–367.

Kling, R., Rosenbaum, H., & Sawyer, S. (2005). *Understanding and Communicating Social Informatics. A framework for studying and teaching the human contexts of Information and Communication Technologies* (1st ed.). Medford, New Jersey, USA: Information Today.

Luna-Reyes, L. F., Black, L. J., Cresswell, A. M., & Pardo, T. A. (2008). Knowledge sharing and trust in collaborative requirements analysis. *System Dynamics Review*, *24*(3), 265–297.

Martínez Fernández, J., & Esteve, M. A. (2004). The dynamics of water scarcity on irrigated landscapes: Mazarrón and Aguilas in south-eastern Spain. *System Dynamics Review*, *20*(2), 117–137.

Martínez-Moyano, I. J., & Richardson, G. P. (2013). Best practices in system dynamics modeling. *System Dynamics Review*, *29*, 102–123.

Meadows, D. H., & Robinson, J. M. (2002). The electronic oracle: computer models and social decisions. *System Dynamics Review*, *18*(2), 271–308.

Stave, K. A. (2002). Using system dynamics to improve public participation in environmental decisions. *System Dynamics Review*, *18*(2), 139–167.

Taylor, T. R. B., Ford, D. N., Yvon-Lewis, S. A., & Lindquist, E. (2011). Science, engineering, and technology in the policy process for mitigating natural-societal risk. *System Dynamics Review*, *27*(2), 173–194.

Tignor, W. (2004). System Engineering and System Dynamics models. In M. Kennedy, G. Winch, R. Langer, J. Rowe, & J. Yanni (Eds.), *Proceedings of the 22nd International Conference of the System Dynamics Society* (Vol. 34, p. 110). Oxford, Inglaterra: System Dynamics Society.

Universidad de los Andes. (2013). La Ingeniería de Sistemas y Computación en Uniandes. *Programa de Ingeniería de Sistemas y Computación - Uniandes*. Retrieved from http://sistemas.uniandes.edu.co/main/pregrado/la-ingenieria-de-sistemas-y-computacion-en-uniandes

Universidad Industrial de Santander. (2013). Introducción del programa académico de Ingeniería de Sistemas. *Programa de Ingeniería de Sistemas UIS*. Retrieved from http://www.uis.edu.co/webUIS/es/academia/facultades/fisicoMecanicas/escuelas/ingenieriaSistemas/programasAcademicos/ingenieriaSistemas/introduccion.jsp

Universidad Nacional de Colombia. (2013). Qué es la Ingeniería de Sistemas. *Pregrado en Ingeniería de Sistemas Universidad Nacional*. Retrieved from http://dis.unal.edu.co/index.php?option=com_content&view=article&id=190&Itemid=239&lang=es

Wils, A. (1998). End-use or extraction efficiency in natural resource utilization: which is better? *System Dynamics Review*, *14*(February), 163–188.