Technology and Investment, **2013, 4, 261-268**
Published Online November 2013 (http://www.scirp.org/journal/ti)
http://dx.doi.org/10.4236/ti.2013.44031

Using the ISO 19761 COSMIC Measurement Standard to Reduce "Information Asymmetry" in Software Development Contracts and Enable Greater Competitiveness

Francisco Valdés Souto, Alain Abran
Software Engineering Research Laboratory (GÉLOG), École de Technologie Supérieure, University of Québec, Montreal, Canada
Email: francisco.valdes.1@ens.etsmtl.ca, Alain.Abran@etsmtl.ca

Received June 26, 2013; revised July 26, 2013; accepted August 4, 2013

ABSTRACT

In most economic transactions involving software development projects, differences in the amount and quality of information possessed by economic agents (*i.e*. producers and customers) can lead to significant market inefficiencies. This paper describes the way the information derived from the functional size of software measured using the ISO 19761 COSMIC measurement standard that can help reduce information asymmetry in software development transactions, and lead to increased competitiveness in the software industry.

Keywords: Software Projects; COSMIC; Information Asymmetry; ISO 19761; Software Industry Competitiveness; Software Characterization; Software Facts

1. Introduction

In the Information Technology (IT) sector, there is frequently a serious imbalance in the contracts concluded between software providers and software customers. This imbalance is generated by the greater, and superior, technical knowledge of the providers relative to that of the customers.

Software development is the practice of automating information processing tasks. The output of this practice is the software itself, which is not a tangible product, but rather a service embedded within a computing infrastructure. Negotiating a software development contract is usually based on a high-level description of the customer's needs on the one hand, in terms of the capacity to process and store information, and, on the other, the claim of the software provider that can meet those needs, and, implicitly, perhaps more.

Tellez [1] defines a computer contract as "the voluntary agreement of two or more parties in order to create bonds of obligations [that] seek to create, regulate, modify or terminate a legal equity relationship, and the provision of which must be related to all or part of the computer: hardware, software, computer servicing, the data provided by computers, or multiple or complex IT services". The "software development contract" is a type "software contract" that is a classification of computer contract.

"Unfortunately, the providers' business policy consists of offering innovative technology-related concepts to clients and not necessarily the best products, with an undefined, and often hidden, disparity between what is required and what is offered" [1].

In software development contract negotiation, we often observe what is known as "asymmetric information". Neoclassical economic theory takes for granted the existence of an ideal marketplace, where both the provider and the customer have all the information they need, and the value of the product is reflected in an agreed price. However, in reality, in most commercial transactions, there are differences in the amount and quality of information possessed by the economic agents (*i.e*. providers and customers) which are serious enough to lead to significant market inefficiencies.

This paper describes an approach using the ISO 19761 standard for software functional size, also known as the

COSMIC method, to reduce the amount of information asymmetry in commercial transactions involving software development projects, and to promote greater competitiveness in the software industry.

The paper is organized as follows: Section 2 describes some types of software contract; Section 3 describes the impact of industry regulation on the economy; Section 4 describes the international standard for functional size measurement (FSM) ISO 19761, also known as the COSMIC measurement method; Section 5 presents the approach proposed to reduce information asymmetry and to promote greater competitiveness in the software industry through the use of the ISO 19761—COSMIC standard; Finally, Section 6 presents the conclusions.

2. Computer Contracts

Computer and software technologies are advancing very rapidly, and their impact on the social environment has led to increased marketing of goods and services derived from these technologies. This market is typically regulated by computer contracts.

Tellez [1] defines a computer contract as "the voluntary agreement of two or more parties in order to create bonds of obligations [that] seek to create, regulate, modify or terminate a legal equity relationship, and the provision of which must be related to all or part of the computer: hardware, software, computer servicing, the data provided by computers, or multiple or complex IT services."

Once computers had penetrated the military and scientific domains, they were introduced into the business field, and it was here that computer contracts began to proliferate.

Since those early days, "computer contracts have evolved with technological advancement, but not on a par with the law" [1].

R. M. A. Davara [2] gives the following definition of computing goods: "All those components that make up the system (*i.e*., the computer) as related to the hardware, from the Central Processing Unit (CPU) to its peripherals and those components directly related to the computer's functionality, and which taken as a whole constitute the material support for the computer's operation; to be considered therewith are also those virtual elements responsible for giving commands, directions, data, and procedures to automatically process information, and which taken together embody the system's logical support". As for computing services, they comprise "all those meant to support and supplement computing actions in a relationship of direct affinity with them".

At the start, agreements for both goods and services were included in a single contract. This resulted in ambiguity and favored monopolistic trade practices, which enabled large suppliers to bundle them in comprehensive suites.

Following an antitrust action taken against IBM, different markets were generated, and goods and services were procured separately. This in turn led to the emergence of several companies specializing in particular aspects of software.

According to V. J. Tellez [1], "Among the main issues arising with computer contracts is the inequality of knowledge of the elements (mainly technical) being negotiated." This issue is usually referred to as adverse selection in the economics field.

Adverse selection is "the phenomenon whereby there is a hidden problem with a product and people on the informed side of the market select in a way that is harmful to the uninformed side of the market" [3]. The way this situation generally plays out is that the customer is forced to accept the contractual terms imposed by the provider.

Tellez [1] also suggests that this situation can be avoided if the consumer seeks the opinion of experts, in order to become aware of the possible implications of a contract. However, one of the drawbacks of this practice is that, in fact, the experts providing the technical assistance are seldom free from commercial bias, since they are in a position to influence what services are offered by the provider.

2.1. Classification of Computer Contracts

A classification of computer contracts needs to take into account their legal nature. Tellez [1] describes them as follows:

- These kinds of contracts are complex by nature, deriving as they do from a range of legal ties found in a variety of contracts.
- They are atypical, as they are not specifically ruled and there are no regulations explicitly intended for them.
- They are principal contracts, not being subordinated to any other contract.
- They are costly, because one of the parties makes an expenditure in order to obtain certain benefits.
- They are consensual.

Tellez [1] created this classification of computer contracts considering the focus (**Figure 1**):

- Hardware,

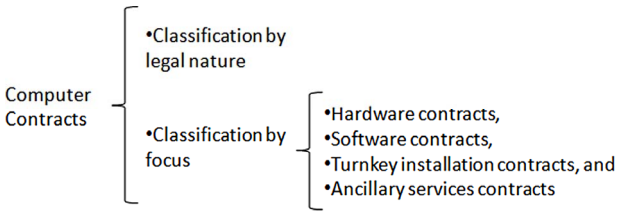

Figure 1. Computer contracts classification.

- Software,
- Turnkey installation, and
- Ancillary services.

2.2. Software Contracts

Software development is the practice of automating information processing tasks. The output of this practice is the software itself, which is not a tangible product, but rather a service embedded within a computing infrastructure.

The classification of computer contracts described by Tellez [1], includes a label for the software contracts. This type of contracts usually is used to negotiating software development. In this paper, "computer contracts" are referred to as "software development contracts", considering the classification in [1] (**Figure 1**).

3. Standards and the Economy

Information Asymmetry

The concept of information asymmetry and its implications have been studied by Akerlof, Spence, and Stiglitz, who were awarded the Nobel Prize in 2001: their work is central to current microeconomic theory [4-7].

Usually, economic studies are conducted under the assumption that buyers and sellers have the same information available to them. However, this assumption is not always correct: V. H. Ronald [8] reports that "if gathering information about quality is onerous, then this assumption is no longer true."

B. D. Ordoñez [9] defines information asymmetry as "the situation in which the buyer and seller have different information about a transaction to take place."

Suppose that M is a generic market where, for simplicity, there are two types of goods or services: "m" (poor quality) and "b" (good quality); in this case, the supply curve ("O") and the demand curve ("D") correspond to each of the two types of goods or services (O_b, O_m, D_b, D_m)—see **Figure 2**.

This figure shows that the O_b curve is above the O_m curve, which means that the suppliers of good quality goods or services expect to receive a higher price for them. The demand curves, where D_b is greater than D_m, can be interpreted as showing that buyers are willing to pay more for good quality.

Buyers who are not certain as to the quality of what they are buying will not be willing to pay the higher cost. They would rather pay a moderate amount, mitigating the risk to them inherent in paying a higher cost for lower quality.

In doing so, buyers will be paying for lower-quality items or services, thus confirming their assumptions. Consequently, they will try to further reduce the suggested price paid, repeating their lower-quality purchase in subsequent deals (t + 1). As consumers realize that most services or products sold are lacking in quality, there will be a shift in the demand, due to the perception that all products and services are low-quality merchandise. This may be observed in the downward movement of the demand line (D) in steps D_1 and D_2.

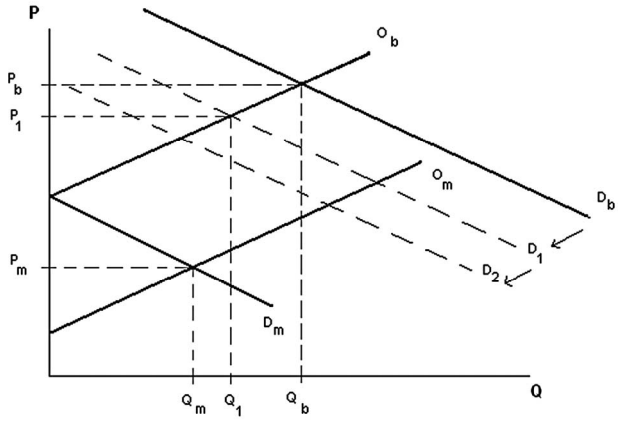

Figure 2. Displacement of supply/demand curves in a generic market characterized by information asymmetry [9].

In a market characterized by information asymmetry, fewer good quality products will be sold, even though there may be buyers willing to pay for better quality ones [9].

In this situation, market agents may draw upon different resources (*i.e*. market signals) to point out the liability of a particular service or product, usually offering a warranty [9].

Given the lack of information, buyers determined to make well-founded deals and decisions are willing to pay for it. Information is hardly accessible, if at all, for many market agents. Organisms that would generate and transmit this information to potential buyers are clearly needed in this particular industry.

Governments, for example, could perform this function for the general public, saving consumers from having to obtain and analyze information themselves on the goods or services they wish to purchase.

One mechanism for reducing the cost of acquiring and processing information is the establishment of standards covering aspects of the quality, features, and description of products [9]. If such standards were widely available, it would be easier for buyers to distinguish good quality products from poor quality ones, leading to better decision making and increased competitiveness.

"A market becomes competitive when emerging institutions put limits on the patterns of behavior of economic agents" [9].

In the particular case of software products, the phenomenon of information asymmetry that occurs in commercial transactions can be readily observed. One way to address this issue would be to develop a software char-

acterization in the form of a statement of software facts, to provide software buyers with objective, standards-based information on a software product, analogous to the nutrition facts that are provided in the context of food labeling—see **Figure 3**.

Software is classified in categories in ISO 12182:1998 Information technology—Categorization of software—which proposes a functional part classifications and a non-functional part, which accomplish the IEEE 830 Software Requirements Specifications. For the former, there are a few international functional sizing standards, such as ISO 19761. For the latter, there is recent work linking measurement with ISO 19761 to the non functional requirements described in the European ECSS set of standards [10].

4. Functional Size Measurement Standard ISO 19761

4.1. Sizing Software

There are basically two ways to measure software size:

1) By the amount of functionality the software is designed to provide;

2) By the size of the physical elements the software contains, or their number, once it has been created.

Physical measurements of size may include lines of code (SLOC), modules, classes, or lines of documentation. Although these elements can be measured automatically and accurately, they have distinct disadvantages. For instance, size is known precisely only after the software has been completely built, and all these types of measurement are technologically dependent, which means that projects cannot be compared across technologies.

According to the ISO, software functional size is defined as "a software size measure derived from functional user requirements" [11]. Furthermore, the ISO specifies that functional size measurements must be independent of technology. In comparison with physical software measurements, functional size measurements show better correlation with the cost of implementation and better applicability in the advanced stages of a project [12].

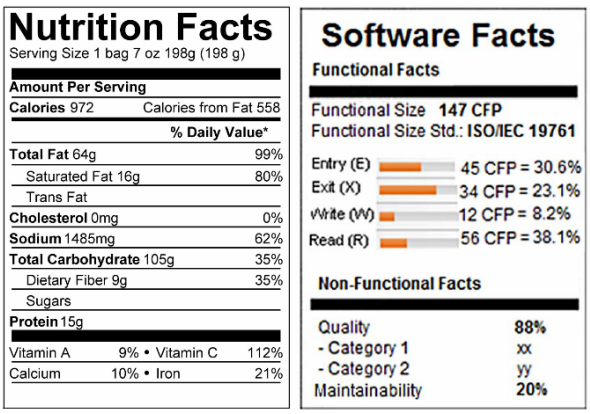

Figure 3. Software labeling: characterization in the form of software facts.

4.2. History of Functional Size Measurement

The idea of measuring software based on functionality and not on physical elements, like SLOC, was first proposed in 1979 by Allan Albrecht [13], and called Function Point Analysis. An updated version of his method followed in 1984.

Several minor improvements to Albrecht's proposal have been made over the years [14,15], but the original structure of his method has never been modified. The improved versions are referred to as 1st generation methods of functional size measurement (FSM)—see **Figure 4**.

In 1994, the ISO established a working group to develop a set of rules to which FSM methods must conform to be accepted as ISO standards.

In the late 1990s, a number of organizations in the USA, Canada, and Japan recognized that a good solution would not be found by luck alone, and they decided to get together and put up the financial resources required to set up a research project with dedicated staff and a project schedule, including tests carried out in industry [16].

Later, other organizations in Europe and Australia provided more funding to finance a second round of tests in industrial settings.

The FFP method was first presented at the IFPUG Fall Conference in Phoenix (Arizona) in 1997, at which time it was put into the public domain. It was hoped that the method would be integrated as an improvement to the original FPA method.

Subsequently, a group of international experts on functional size measurement, the Common Software Measurement International Consortium—COSMIC—was set up to make the initial design more robust from a measurement perspective, and to broaden its consensual basis to bring it up to the level of an international standard, and

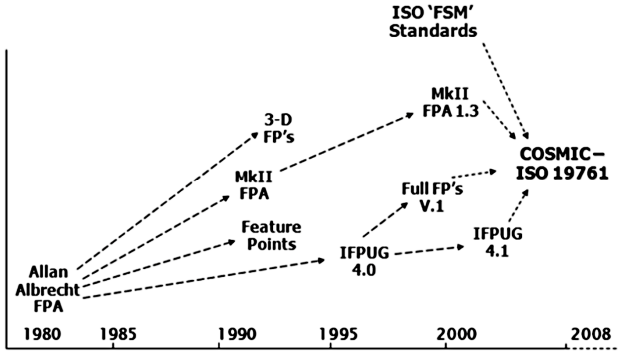

Figure 4. Evolution of functional size measurement methods [16].

to see it adopted as such by the ISO, which could be considered the ultimate reward for a measure in terms of recognition.

In September 2007, version 3 was published, submitted to the ISO, and adopted as the ISO 17971—COSMIC standard in 2009 [17]—see **Figure 4**.

The size of software measured based on ISO 19761 is defined by the number of data movements it contains: (Entries (E), Exits (X), Reads (R), and Writes (W))—see **Figure 5**. In ISO 19761, the standard measurement unit is 1 CFP (*i.e.* one COSMIC Function Point, which corresponds to the movement of a single data group of any type (Entry (E), Exit (X), Read (R), or Write (W)).

4.3. The Importance of Functional Size Measurement

In the literature, software development is sometimes compared to building a house. The plans for the house are analogous to the models used in software development.

What would happen if house plans did not have measures? Almost certainly, the result would be a badly constructed house. Agreeing on contracts for the construction of a house based on plans without precise measurements would be extremely complicated, not to mention risky.

It is easy to see that a good quality house cannot be built without knowing the size of all its elements. Moreover, without precise quantitative data, it would not be possible:

- To reasonably estimate the duration and costs of the construction work,
- To monitor the construction process based on the dimensions of the house,
- To make comparisons with similar houses,
- etc.

Can software be developed without measuring its size? Unfortunately, the answer is too often yes. In addition,

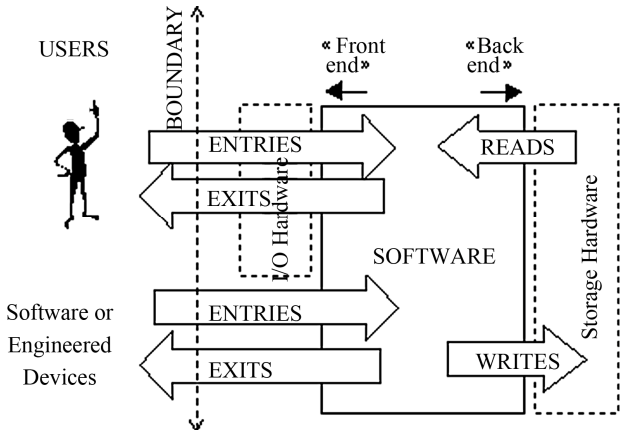

Figura 5. Data movements in ISO/IEC 19761 [17].

these projects may not be well managed. In fact, development without measurements encourages software to be developed as an art, instead of as an engineering process. This frequently leads to unpredictable results [18,19]. Most of the time, the software is not available on time, or with the desired cost and quality [20]. The situation is worse when combined with the problems identified by Tellez [1] in commercial software development relationships.

Many of those involved in the software industry have misconceptions about software measurement. They consider that software can't be measured because it is intangible and very complex. This, of course, is a simplistic view, since even the measurement of physical phenomena can be quite complex, at times requiring models; for example, the speed of light, which has to be measured with appropriate models (such as the wave model of light) and cannot be directly measured physically.

The only ISO standards that currently address the measurement of software features are those related to the size of their functional requirements [11,17].

In particular, measurements with international standards can be used for, among other things:

- Software project estimation;
- Software performance measurement;
- Control of software project scope;
- Control of software contracts;
- Determination of the productivity of software projects.

5. ISO 19761 as an Enabler of Competitiveness in the Software Industry

5.1. Information Asymmetry: The Issue and Its Solution

Information asymmetry indicates a lack of information about a product, and also the fact that individuals are willing to pay for information in an attempt to make better decisions.

The existence of information asymmetry in a market means that the goods or services that are sold are usually of lower quality. Consequently, inferior quality products displace higher quality products.

One way to deal with this issue is for governments to set rules for the generation and dissemination of the information needed by the general public through the development and adoption of standards which make clear to buyers the differences between goods and services of lower and those of higher quality, whether or not they are similar in size, and whether or not they have particular characteristics (based on the software characterization or software facts).

Considering the above, we present a proposal here that takes into consideration a scheme in which the adoption

by the software industry of the international standard ISO 19761 could improve competitiveness by reducing information asymmetry.

5.2. Software Industry Context

In the software development process, the information known at the beginning of the conceptualization phase is insufficient [21], because the user has a vision of the software product only at a fairly high level of abstraction. In addition, the price of software is usually established before the project begins, which typically leads to a number of renegotiations (usually for cost increases, rather than decreases) over its development life cycle.

Negotiations in these very early phases correspond, on the one hand, to the commercial approach taken by providers (overestimation) and, on the other, to reserve or caution on the part of poorly informed buyers, resulting in information asymmetry. According to the theory of information asymmetry, this usually leads buyers to purchase cheaper software developments or services than when the providers send confident signals, in the form of brand names or certifications, which give assurance that more expensive goods or services also offer superior quality.

This phenomenon can also be seen in bidding schemes, where the lowest bid is accepted, or in contracts where the providers are big software corporations with specific certifications.

The reality is that in neither case is it certain that a satisfactory result will be achieved (*i.e.* one that meets the functionality requirements expected in the software product). To date, most software providers do not use any international standard to measure software functionality. The main reasons are the following:

- Lack of knowledge on the part of providers and customers;
- The lack of requirements for software-related contracts at the government level as a transparency mechanism.

5.3. Proposal

"*Since information asymmetry is the absence of information to differentiate software products, and since there are international standards for the measurement of the functional size of software, we propose the use of the ISO* 19761 *standard as a basis for determining this key characteristic of the software and therefore establish a basis on which sales transactions can be conducted transparently.*"

The focus will be on measuring functional size in software clusters defined nationally. Usually, these clusters supply software development services to a government, which can be a major IT consumer in a country. This will initially require a national measurement effort to implement the international standard. The next step will be to use these measurement results to analyze the productivity of the projects developed for the government.

A current problem in the software industry is to analyze supply and demand curves, like those in **Figure 2**; *i.e.* unless adequate international measurement standards are applied, the size of software that is built is almost never known, either by those who develop it or by those who purchase it (information asymmetry). Therefore, the estimated prices are set up rather arbitrarily, which means that one set of software features (functional size) may be priced differently by different providers, and there may be very wide variations in those prices.

At the start, the mandatory use of standards of functional size measurement in software contracts will require the potential buyer of the software to determine in greater detail the functionality required, and the provider of the software to establish an appropriate and competitive cost structure.

When this approach is used on a continuous basis, the cost per unit of functional size will tend to be influenced naturally by the market.

Using this approach and assuming that ISO 19761 [17] is applied, the behavior of the supply and demand curves in **Figure 2** would be modified, because the quantity (Q) would have a standardized unit (CFP)—see **Figure 6**.

An analysis of **Figure 6** reveals that software buyers would be willing to pay more for more functional size and pay less for less software functionality.

By using ISO 19761 [17] as a mechanism for formal software characterization (software facts), and clearly indicating the functional size of specific software, the provider and the customer would both be able to correlate quantity and cost, as in any industry where a larger amount of product costs more (in this case, the more functionality software has, the higher its cost). This would enable

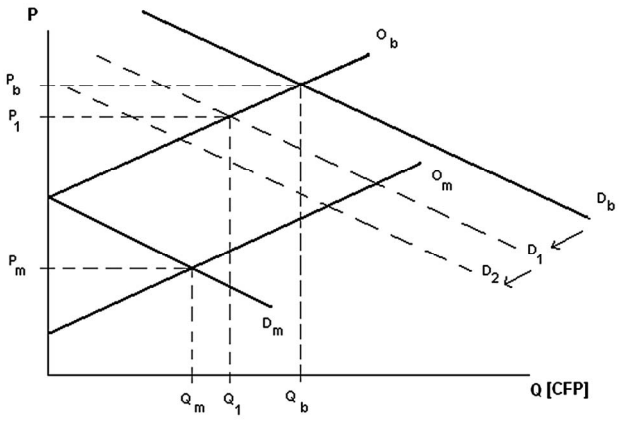

Figure 6. Supply/demand curve displacement for the software market, modified from [9].

customers to make more informed decisions, and result in the best products being brought to market, and not necessarily the cheapest.

Governments should promote greater competitiveness in the software industry by supplying information to providers and customers, giving average functional size unit cost references. For example, they should conduct studies to determine average functional size unit costs by type of application, by technology, by geographic area, etc., all based on international standards for the measurement of software size.

Of course, there are other factors—software facts—that could improve the supply/demand curve for software, in addition to functional size. Many of these are related to non functional requirements, such as quality, maintainability, usability, etc. However, mature solutions and international standards for these do not yet exist. In the interim, we must start with the existing FSM standards, as they are the only quantitative standards available to implement the software characterization mechanism in the context of software contracts, and to ensure that transactions are information symmetrical and enable competitiveness.

6. Conclusions

This paper has described the issue of information asymmetry that currently arises in software development contracts, as illustrated by the unequal level of information that the agents in a software purchase transaction have, which often give the advantage to the providers.

In any industry where a product is fabricated, there is a correlation between what is built and the cost to build it. In the case of software, without measurement standards, it has not been easy to see this correlation at work, and the result has been different prices for equivalent software functionality.

The proposal in this paper is to implement a mechanism to obtain a software characterization or "software facts" using quantitative standards, the only ones available at the present time being the ISO standards on functional size measurement. We recommend use of the COSMIC method (*i.e.* ISO 19761).

Governments could then act as society's guarantors by funding studies to document how prices are established (for technologies, geographic areas, etc.), with a view to generating greater competitiveness in the software industry.

A functional size standard is only the beginning in the characterization of software to develop the basic information required by the agents involved in this type of transaction. Of course, there are further costs associated with software development projects related to other factors; however, while there are descriptive standards for software, such as maintainability, quality, etc., there is not yet a consensus on how to describe it quantitatively. Consequently, there is a need to develop additional models that can be used to provide a quantitative index for measuring software, initially based on a qualitative approach, but later on a quantitative approach.

REFERENCES

[1] V. J. Tellez, "Derecho Informático," 4th Edition, McGraw Hill, México, 2009.

[2] R. M. A. Davara, "Manual de Derecho Informático," Décima Edición, Thomson Aranzadi, 2008.

[3] M. L. Katz and H. S. Rosen, "Microeconomics," Irwin McGraw-Hill, Boston, 1998.

[4] K. Lofgren, T. Prsson and J. W. Weibull, "Markets with Asymmetric Information: The Contributions of George Akerlof, Michael Spence and Joseph Stiglitz," *The Scandinavian Journal of Economics*, Vol. 104, No. 2, 2002, pp. 195-211. http://dx.doi.org/10.1111/1467-9442.00280

[5] G. A. Akerlof, "The Market for 'Lemons': Quality Uncertainty and the Market Mechanism," *The Quarterly Journal of Economics*, Vol. 84, No. 3, 1970, pp. 488-500. http://dx.doi.org/10.2307/1879431

[6] M. Rothschield and J. Stiglitz, "Equilibrium in Competitive Insurance Markets: An Essay on the Economics of Imperfect Information," *The Quarterly Journal of Economics*, Vol. 90, No. 4, 1976, pp. 629-649. http://dx.doi.org/10.2307/1885326

[7] M. Spence, "Job Market Signaling," *The Quarterly Journal of Economics*, Vol. 87, No. 3, 1973, pp. 355-374. http://dx.doi.org/10.2307/1882010

[8] V. H. Ronald, "Microeconomía Intermedia," 4th Edition, Un Enfoque Actual, Spain, 2007.

[9] B. D. Ordoñez, "La Economía y las Normas," NYCE, 2007.

[10] A. Abran, K. T. Al-Sarayreh and J. J. Cuadrado-Gallego, "A Standards-Based Reference Framework for System Portability Requirements," *Computer Standards and Interfaces Journal*, Vol. 35, No. 4, 2013, pp. 380-395.

[11] ISO, "Information Technology—Software Measurement—Functional Size Measurement," ISO/IEC 14143-1, 1997.

[12] L. Santillo and C. Grande, "Breve Storia della Misurazione del Software," GUFPI-ISMA, *Newsletter*, Vol. 3, No. 1, 2006.

[13] A. J. Albrecht, "Measuring Application Development Productivity," IBM Application Development Symposium, GUIDE Int and Share Inc., IBM Corp., Monterrey, 14-17 October 1979, p. 83.

[14] T. C. Jones, "The SPR Feature Point Method," Software Productivity Research Inc., 1986.

[15] C. R. Symons, "Function Point Analysis: Difficulties and Improvements," *IEEE Transactions on Software Engineering*, Vol. 14, No. 1, 1988, pp. 2-11. http://dx.doi.org/10.1109/32.4618

[16] A. Abran, "Software Metrics and Software Metrology," John Wiley & Sons Interscience and IEEE-CS Press, New Jersey, 2010, p. 328.

[17] COSMIC Core Team Authors, “Measurement Manual, the COSMIC Implementation Guide for ISO/IEC 19761: 2003,” The Common Software Measurement International Consortium (COSMIC), May 2009.

[18] Standish Group International, CHAOS Summary 2009, Coll. Research Reports, The Standish Group International, Inc.

[19] K. Emam and A. G. Koru, “A Replicated Survey of IT Software Project Failures,” *IEEE Software*, Vol. 25, No. 5, 2008, pp. 84-90. http://dx.doi.org/10.1109/MS.2008.107

[20] F. V. Souto, “Midiendo la Calidad del Software,” Software Gurú, No. 40, 2013, pp. 38-39.

[21] F. Valdés, “Design of a Fuzzy Logic Software Estimation Process,” Ph.D. Thesis, École de Technologie Supérieure, Université Du Québec, Montreal, 2011.