Software Productivity: Harmonization in ISO/IEEE Software Engineering Standards

Laila Cheikhi
ÉNSIAS, Université Mohammed V- Souissi, Rabat, Morocco
Email: cheikhi@ensias.ma

Rafa E. Al-Qutaish
Al Ain University of Science and Technology - Abu Dhabi Campus, Abu Dhabi, UAE
Email: rafa.alqutaish@aau.ac.ae

Ali Idri
ÉNSIAS, Université Mohammed V- Souissi, Rabat, Morocco
Email: idri@ensias.ma

***Abstract*— The software productivity is an important key of software quality factors. The productivity measure has become a tool for managers since it is used to compare the performance between different companies (benchmarking) and to compare the efficiency of different developers in the same company. Therefore, it allows doing strategic planning and decision making based on such measurement. A variety of international standardization bodies such as IEEE and ISO as well as software engineering researchers have proposed a set of factors which influence the software productivity attribute, and also a set of measures to evaluate it. However, there is no unique model that integrates all the software productivity best practices. The aim of this paper is to survey the available international standards and research work on software productivity and figure out the key differences in order to propose a standards-based model. Such model will include the set of quality attributes that could be used to reflect the software productivity, and a set of measures that allows evaluating the software developer's productivity.**

***Index Terms*— Software Developers Productivity, Quality Models, ISO 9126, IEEE Std. 1045, Productivity Drivers, Quality Attributes, Measurements**

I. INTRODUCTION

Productivity is defined from the economic view point as "*the rate of output per unit of input used especially in measuring capital growth, and assessing the effective use of labor, materials and equipment*" [1].This definition includes two main points, that is, the measure of the productivity (output ÷ input), and the effective use of resources.

The defacto measure of productivity is the ratio of size (as an output) per effort expended (as an input). In one hand, the size can be expressed in any measurement of size. In fact, De Aquino and De Lemos Meira [2], in their survey of productivity measurements, have identified four categories of output, which are; Physical Size (e.g., SLOC - Source Lines of Code), Design Size (e.g., number of modules), Functional Size (e.g., FP - Function Point), and value based metrics or use multidimensional models that assess different aspects of what is produced in a software project. An example of the use of multidimensional approach to assess the productivity is proposed in [3], which is called QEST (Quality factor + Economic, Social and Technical dimensions). On the other hand, the effort is measured by Person-Day (PD) or Person-Month (PM).

As an example, suppose that we have measured the productivity of two programmers P1 and P2 - as in Table 1 - who have to achieve the same task using the produced LOC (Lines of Code) as a software size (output) and the PM as the expended effort (input).

TABLE I
PROGRAMMER PRODUCTIVITY

Programmer	LOC	Effort	Reuse	Productivity
P1	5000	100 PM	No	50 LOC-PM
P2	5000	100 PM	Yes	50 LOC-PM

However, it can be easily noted from Table 1 that the two programmers have the same productivity value (50 LOC-PM), but their productivity is not comparable since P2 have reused code in completing his task. Therefore, using this measure for productivity is not reliable, in particular, when comparing the software engineers' productivity without taking into account the effect of the factors which influence the software development process such as reuse.

In the literature, research work about productivity is directed towards two main topics; that is, identification of factors influencing the productivity and proposition of productivity measures. According to Fenton [4],

Manuscript received April 14, 2011; accepted June 27, 2011.

doi:10.4304/jsw.7.2.462-470

"*Software engineers define productivity in terms of a measure, rather than considering carefully what attribute is being captured*". The purpose of this paper is to identify the attributes which should be captured in order to achieve the productivity, and propose measurements method that can be used to measure these attributes based on the ISO 9126-4 and IEEE Std. 1045 international standards. Since the factors (such as software development processes, technology, and team work) influencing the productivity are important when comparing the projects, this topic is briefly addressed in this paper.

Generally, in software productivity literature as well as in the international standards, there are different viewpoints on the related productivity measures which have been developed over the last years by practitioners and researchers from various organizations. Each of them (practitioners or researchers), unfortunately, have built their model without any input from the others. In such situation, different terms have been used to express the same concept, or similar concept has been expressed in different terms. Therefore, bringing a convergence and consensus on productivity measures and their factors would facilitate both the benchmarking studies, and the repeatability and reproducibility of productivity measurements.

The rest of this paper is structured as follows: Section 2 provides an overview of the productivity term in international standards. Section 3 presents a discussion about the productivity in ISO 9126-4 and IEEE Std.1045. Section 4 introduces the enhanced software productivity model and the enhanced software productivity metrics. Finally, Section 5 concludes the paper and provides some suggestions for the improvement of the current ISO 9126-4 and IEEE Std.1045 standards.

II. Representation of Productivity in International Standards

A. Productivity in ISO 9126

The International Organization for Standardization (ISO) has published, from 2001 to 2004, four documents related to the ISO 9126 on software product quality. The first document, ISO 9126-1 [5], is an international standard that establishes a two-parts quality model, that is, internal and external quality model, and quality in-use model. Each model proposes a set of quality characteristics and sub-characteristics. The other three documents, ISO 9126-2, 3, and 4 [6, 7, 8], are technical reports which propose a set of measures. However, this ISO standard has gained an international consensus among the ISO countries and expert.

The quality in-use is defined in ISO 9126 as "*the capability of the software product to enable specified users to achieve specified goals with effectiveness, productivity, safety and satisfaction in specified contexts of use*" [8]. Its corresponding quality model contains four characteristics (see Figure 1). This type of quality is not related to the intrinsic properties of the software product, but represents the user's view of the quality measured from the use of the software in the specified environment. Moreover, the ISO 9126 states that the quality in-use is the combined effect of the external and internal quality. In particular, for the productivity quality characteristic, it may depend on others quality characteristics that belong to the software product before its delivery to the end user.

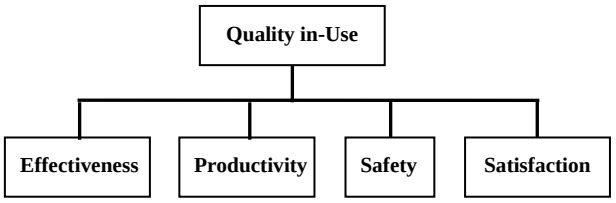

Figure 1. ISO 9126 quality in-use model.

The productivity in ISO 9126 is a quality factor of the software product and it is defined as "*the capability of the software product to enable users to expend appropriate amounts of resources in relation to the effectiveness achieved in a specified context of use*" [5]. The productivity measures assess the resources that users consume in relation to the effectiveness achieved in a specified context of use [8]. Therefore, the productivity of the software product focuses on four main points, which are, user's ability, expended resources, effectiveness achieved, and context of use. These points are defined in ISO 9126-4 [8] as the following:

- The context of use includes the users, tasks, equipment (hardware, software and materials), and the physical and social environments in which a product is used.
- The user is an individual that uses the software product to perform a specific function.
- The resource includes time to complete the task, although other relevant resources could include the user's effort, materials or the financial cost of usage.
- The effectiveness is the capability of the software product to enable users to achieve specified goals with accuracy and completeness in a specified context of use". The goal is "an intended outcome" and the task is "the activities required to achieve a goal".

Table 2 below presents the productivity and effectiveness quality characteristics with their corresponding derived and base measures, since the productivity measures are based on those of the effectiveness.

Moreover, the informative Annex E of the ISO 9126-4 presents three others productivity measures related to the three types of resources, defined as follow:

- Human productivity = effectiveness ÷ human effort.
- Temporal productivity = effectiveness ÷ time.
- Economic productivity = effectiveness ÷ cost.

For example, if the goal is to solve online users' problems, the human productivity could be measured by the ratio of the number of resolved problems and the effort expended to achieve this goal. The temporal productivity could be measured by the ratio of the number of resolved problems and the time spent to achieve this goal. The economic productivity could be

measured by the ratio of the number of resolved problems and the cost spent to achieve this goal.

TABLE II
THE ISO 9126 PRODUCTIVITY AND EFFECTIVENESS MEASURES

Quality characteristic	Derived Measures and purposes	Base Measures
Productivity	- Task time (How long does it take to complete a task?)	Task time
	- Task efficiency (How efficient are the users?)	Task effectiveness/Task time Task completion/ Task time
	- Economic productivity (How cost-effective is the user?)	Task effectiveness/Total cost of the task
	- Productive proportion (What proportion of the time is the user performing productive actions?)	Productive time (Task time - help time - error time - search time) / Task time
	- Relative user efficiency (How efficient is a user compared to an expert?)	Ordinary user's task efficiency/expert user's task efficiency
Effectiveness	- Task effectiveness (What proportion of the goals of the task is achieved correctly?)	
	- Task Completion (What proportion of the tasks is completed?)	Number of tasks completed/number of tasks attempted
	- Error frequency (What is the frequency of errors?)	Number of errors made by the user/time or number of task

From the above definitions, we can say that the productivity in ISO 9126 is also measured by the ratio of output and input in the high level, for example, effectiveness ÷ resources, goal to achieve ÷ resources, task to achieve ÷ resources in the low level. Therefore, the ISO 9126 definition of productivity is generic and can be applicable to any types of software product artefacts depending on the goal fixed to achieve and the resource expended in a specified context of use. Moreover, the productivity in ISO 9126:

- is a quality factor which can be measured during the use of the software in a specified context, which takes into account users, tasks, equipment and environment of the software product;
- focuses on all types of intermediate products and products intended for users and is related to not only to the set of programs, but also to procedures, documentation and data that will be delivered to users, such as developers, maintainer, etc; and
- can be measured in the output, in terms of goals to achieve with accuracy and completeness, where the goal can be achieved through a set of tasks, and in the input, in terms of time, human effort, or cost expended.

However, there are still some weaknesses in ISO 9126-4, which have not yet been completely tackled, such as:

- Although some recommendations are presented in Annex C of this ISO standard including different ways to measure time, effort, cost, and size measures, these are only informative, that is, there is no any consensus on them and their units of measures.
- The productivity can be measured during the whole software lifecycle, but in the ISO 9126-4 reference to ISO 12207 lifecycle, it corresponds to only validation, qualification testing and operation.
- For comparison purpose, ISO 9126-4 only mentioned that the productivity output and input measured should be made for the same purpose and consequently have a comparable scope, without detailing what is included in the scope.

In summary, the productivity in ISO 9126-4 is a quality factor focuses on the end user and the use of the software product in a specified context of use. However, we refer to it by dynamic productivity. Furthermore, context of use (user, tasks, equipment, and environment) can be sufficient to determine productivity as a product quality characteristic, that is, a change in any relevant aspect of context of use can change productivity measure and its interpretation. For example, the user efficiency in a user interface can be improved by training, then, the qualified and novice users can achieve the same task in different laps of times. Therefore, the controlled productivity drivers (such as process factors) and those not (such as product factors) can influence the productivity [9].

B. IEEE Std. 1045 Productivity Metrics

The Institute of Electrical and Electronics Engineers (IEEE) has published in 1992 a standard for software productivity metrics - IEEE Std. 1045 [10]. The objective of this standard is to standardize the way to measure the software productivity output products and input effort. The productivity is expressed in terms of output / input, and provides a set of units to measure the output and the input allowing therefore better comparison of developer's productivity.

Moreover, this standard provides a non exhaustive list of characteristics related to software development factors that can have an impact on productivity (see the Annex of this standard). However, these characteristics are subdivided into three categories [10]:

- Project characteristics include factors that the developer can manage, and are related to personnel and software development environment, etc.
- Management characteristics focus on factors related to how the project is managed. Such factors are recorded

after project completion and are related to user participation, stability of requirements, etc.

- Product characteristics include factors imposed upon the product itself, and are related to quality, criticality, etc.

To facilitate the collection of these characteristics, an example of data collection form is given in Annex B of this standard.

The IEEE Std. 1045 software productivity model consists of outputs and inputs for defining and identifying the productivity. Each of these outputs and inputs consists of a set of primitives, and each primitive has a set of attributes. Each attribute could be reflected by one or more metrics depending on the purpose of the productivity measure (see Figure 2).

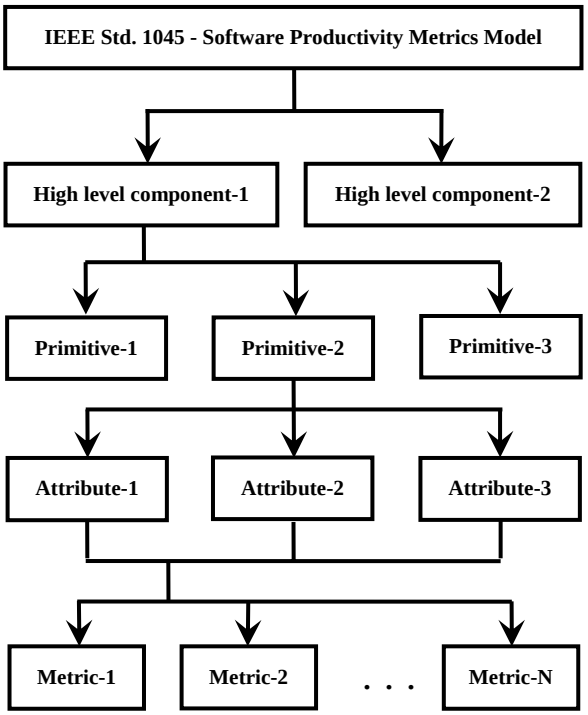

Figure 2. Structure of IEEE Std. 1045 software productivity metrics model.

In this Model, the first level represents the main components of the productivity measures, which are output and input, and describe the productivity at the high level and in more generic context. The second level presents the primitive, that is, the component from which the data is collected for the productivity measure [10]. The output primitives are source statements, documentation, and function points, which is optional in this standard. The input primitive measures the effort of the developers of the software product expressed in staff-hour. The third level is related to the attribute (the measurable characteristic of a primitive) [10], from which a set of measures can be derived depending on the purpose of the measurement.

Table 3 shows the contents of IEEE Std. 1045 model, that is, the primitives and attributes, which are defined as the following [10]:

- Source statement (SS) primitive: it represents the encoded logic of the software product, each SS primitive is defined by three attributes:
  - Type: it classifies each source statement as executable, data declaration, compiler directive, or comment.
  - Origin: it classifies software as either developed (created new, or modified from existing software) or non-developed (reused or deleted from the existing software).
  - Usage: it identifies whether the software was delivered, or not delivered to the user.
- Documentation primitive: it corresponds to all documents that consume a nontrivial amount of project resources, each document primitive is defined by three attributes:
  - Type: it is reported with two values; the name of the document and its purpose.
  - Origin: it is related to the origin of the document expressed in terms of original tokens, modified or added.
  - Usage: it shall be identified as delivered or undelivered.
- Function point primitive: it corresponds to Albrecht's definition of function point or similar methods of measuring functionality. This FP primitive has only the 'Type' as attribute, which describes the function types used in the function point method.
- Staff-hour primitive: is related to the effort expended to produce an output. The attribute is related to the nature of the effort and it consists of direct staff-hour (delivered or undelivered) and support staff-hour. Effort expended away from the goal of the productivity measure is excluded for the accuracy of the results.

TABLE III
CONTENT OF IEEE STD. 1045 SOFTWARE PRODUCTIVITY METRICS MODEL

High level components	Primitives	Attributes
Output	Source statements	Type Origin Usage
	Documentation	Type Origin Usage
	Function points	Type
Input	Staff-hour	Nature

However, based on above data when collected, the productivity can be computed easily depending on the output, that is, the goal or task to achieve (in ISO 9126-4 terminology). Examples of productivity measures are presented in Table 4.

Therefore, the IEEE Std.1045 model provides practitioners with well-defined output and inputs at three levels of details, from which a set of measures can be

composed based on the base measures with their corresponding units of measures. The IEEE Std. 1045 can be used as a reference for measuring software product productivity, especially when the output is related to the source statements, documents, and function points primitives. In addition, the IEEE Std. 1045:

- Provides a set of guidelines on software project characteristics that could have an impact on productivity measure.
- Provides guidelines and measurements that allow better understanding of software productivity.
- Allows computing productivity during the different lifecycle phases:
  - At the completion of the software product, identified by final productivity, which can be measured when the product is delivered.
  - During the development process, identified by incremental productivity, which can be measured at intervals to assess the progress of the work.

TABLE IV
IEEE STD. 1045 PRODUCTIVITY MEASURES

Productivity ratio types	Output metrics	Input metrics
Direct delivered Source Statements (SS)	Delivered new SS Delivered reused SS	Direct delivered staff-hour
Direct undelivered Source statements	Undelivered new SS Undelivered reused SS Undelivered deleted SS	Direct undelivered staff-hour
Function points	Number of function points	Direct delivered staff-hour Direct undelivered staff-hour Support staff-hour
Document	Document page count Screen count Number of words Number of ideograms Number of graphics	Direct delivered staff-hour Direct undelivered staff-hour Support staff-hour

On the other hand, the IEEE Std. 1045 has the following weaknesses:

- Focuses only on the output attributes related to LOC and documentation, while the function point as output primitive is optional.
- Allows comparison of source statement productivity of only software products developed using the same programming language, since the focus is on LOC for the output. In fact, the LOC measure is different from one programming language to another.
- Does not claim to improve productivity or to measure the quality of software and does not specify a standard life cycle or at least imply the existence of a standard software life cycle [10].

In summary, the productivity in IEEE Std. 1045 is specific for software product artifacts produced as output, and not to the use of the software product; we refer to it by static productivity. Moreover, the project, management, and product characteristics are useful and have to be documented when collecting data related to productivity measure (as stated for the context of use in ISO 9126-4) in order to make reliable comparison.

III. DISCUSSION

As stated in the introduction, the comparison between projects requires the similarity of the characteristics in IEEE Std. 1045 terminology, that is, the factors related to the context and the environment of the software product in ISO 9126-4 terminology. Not only the data shall be collected and recorded but the time of its availability is also important. For example, the LOC size measure is only available at the end of the software development, while the FP is available early (and during all the software lifecycle phases). Moreover, for the FP, the functionalities documented in the design phase, will not be similar to those at the end of the project, if requirements have been changed (instability of requirements).

Trendowicz *et al.* [11] stated that there are different types of the proposed approaches in the literature for identifying candidate factors that could influence software development productivity:

- Expert-based approach: software experts decide about a factor's importance based on their own experience and judgment [10, 12, 13, 14, 15].
- Data-based approach: available data (such as in ISBSG data repository) or collected data about a set of factors that could influence the productivity are analyzed to identify their relevance and to understand the interaction among these factors regarding a certain criterion [16, 17, 18, 19, 20, 21, 22].
- Integrated approach: combines the expert and the data based approaches [11].

In fact, from the big number of volumes that have been written by individual researchers and international standards on the topic of determining the key factors that influence productivity, there are four factors that are recurring themes, that is, product characteristics, people (stakeholder, manager, developer, etc.), software development processes, and technology, methods and tools.

A comparison of the ISO 9126-4 and IEEE Std. 1045 is conducted in order to figure out how these two standards address the productivity and how they can be used together to provide the researchers with standardized aspects to measure productivity. From this comparison the followings points have been figured out:

1. Productivity definition: Both standards use the same definition of the productivity expressed in terms of the ratio of output and input at the high level. At the low level, the outputs in IEEE Std. 1045 are related to source statements, documentation and function points

primitives, which allow to measure the productivity in a static way, i.e., without executing the software. In the ISO 9126-4 the output is the goal, which can be achieved by the task, and allows to measure the productivity in a dynamic way, i.e., during the execution of the software in a specified context of use.

2. Primitives and attributes: The ISO 9126-4 refers also to the three IEEE Std. 1045 primitives and their related attributes, but in an informative way. In more details, the IEEE Std. 1045 input primitive (effort) is expressed only in staff-hour while the ISO 9126-4 propose three types of inputs, that is, time, cost and human effort, but did not give a unit of measure for each of them. Moreover, the function point primitive is optional although this method has proved its usefulness with the improved versions produced over the years.
3. Productivity drivers: Both standards address the importance of the factors that could influence the productivity measure. The IEEE Std. 1045 provides three categories of factors (which are project, management, and product) with their corresponding sub-factors, while the ISO 9126-4 gives general aspects of context of use (which are user, tasks, equipment, and environment) without identifying the content of each aspect of context.
4. Productivity measures: Both standards provide a non exhaustive list of measures and allow collecting productivity data during the different phases of the software development process. However, while the ISO 9126-4 give a standardized software lifecycle processes (in ISO 12207) where the measure is applicable, the IEEE Std. 1045 does not provide one, and the users have to identify the activities related to each phase of their own lifecycle.

In summary, the analysis of the ISO 9126-4 definition of productivity indicates that this standard has a broader perspective about the productivity, while the IEEE Std. 1045 focuses on a limited perspective of productivity. The ISO 9126-4 focuses on the output as a goal and task to be achieved, while the IEEE Std. 1045 focuses on a subset of this goal, that is, the source statements and the documentation. Moreover, the ISO 9126-4 has addressed productivity as a quality attribute of the software product, but the IEEE Std. 1045 does not claim to measure the quality of software. However, these two perspectives are complementary; for example, if the purpose is to measure the productivity of the product or the documentation (that is a specified task in ISO 9126-4 terminology), the IEEE Std. 1045 proposed measures can be used in the specified context of use. The advantage of the IEEE Std. 1045 is to provide a specified definition of each attribute to measure with its corresponding unit of measure. This feature suppresses the ambiguity and misunderstanding of the measured attribute, and allows the benchmarking and the same interpretation of the results. Indeed, depending on how and which indicators are measured the conclusions about the productivity can be completely different [23].

The next section presents some suggestions for improving:

- ISO 9126-4 productivity quality characteristic to include more detailed level that allows capturing the measurable attributes, and
- IEEE Std. 1045 productivity model to take into accounts the function points and staff-cost as a basic primitives.

IV. Productivity Improvements

A. Enhanced Productivity Model

As stated before, the productivity in ISO 9126 is expressed in terms of goals to achieve the output with accuracy and completeness. The goal can be achieved through a set of activities called tasks. According to De Aquino and De Lemos Meira [2], an ideal metric of productivity should be able to measure how effectively and efficiently to turn ideas into software products. Therefore, the productivity as a quality characteristic can be subdivided in three sub-characteristics, that is, accuracy, completeness, and efficiency, which we define in the context of this paper as follow:

- Accuracy: the capability of the software product to enable user to achieve the task rightly or within the agreed results with the needed degree of precision in a specified context of use.
- Completeness: the capability of the software product to enable the user to achieve completely the task without requiring adjustment or refinement in a specified context of use.
- Efficiency: the capability of the software product to enable user to accomplish the task relative to the amount of resources (human, time, or cost) used, in a specified context of use.

The enhanced productivity model is presented in Figure 3, in high-level concepts. The relevant candidate measures of ISO 9126-4 are analyzed and are used to improve the third level of the productivity model structure (see Figure 4).

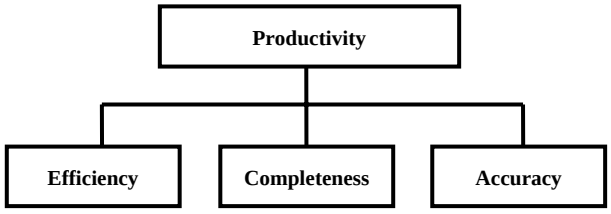

Figure 3. High-level enhanced productivity model.

B. Enhanced Productivity Metrics

Since the productivity measurement tracks how well the worker applies talents and skills - using materials and equipment - to produce products and services within a specified time period [24], the two standards are complementary. In particular, the IEEE Std. 1045 model can be used as a part of the enhanced productivity model (Figures 3 and 4) when the task in ISO 9126 (output) is about the source statements, the documents, and the function points primitives, and also when the effort (input) is measured by staff-hour.

In one hand, the use of size measures (as output), such as LOC, Halstead's measure and Albrecht FP, have been criticized in [16]. Fortunately, the standardized measurement method COSMIC (Common Software Measurement International Consortium) [25] has addressed the disadvantages, inconsistencies, and shortness of these previous size measures. Therefore, we propose to harmonize the optional FP section in the IEEE Std. 1045 standard with the origin and usage attributes (Figure 5), which to be captured based on the well known and proved COSMIC measure.

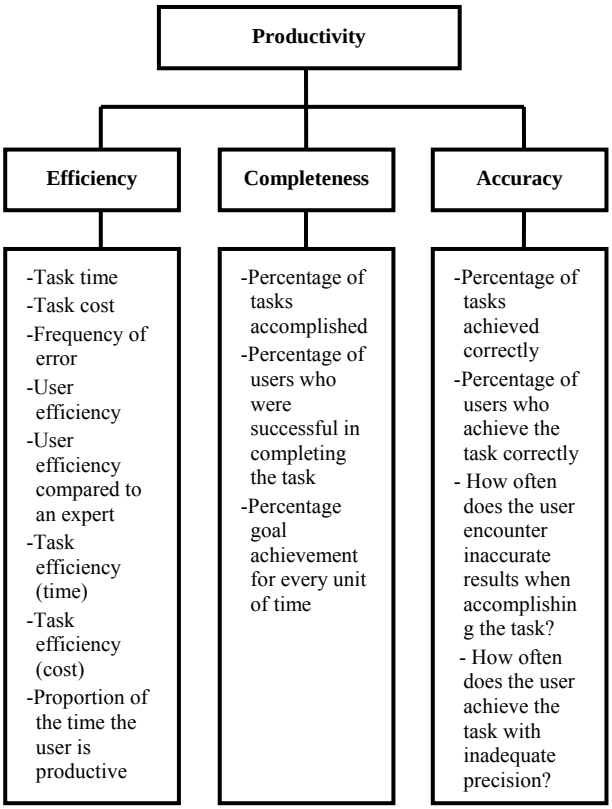

Figure 4. Consolidated productivity model.

The COSMIC measurement method (ISO 19761) is based on the functional user requirements, which are the functionalities that should be implemented in order to satisfy the user's needs. This international standard (ISO 19761) is based on the function points, which is introduced in 1979 by Allan Albrecht to help in measuring the productivity of software development [26].

However, when the function points (measured using the COSMIC measurement method) is used as an output primitive, the 'type', 'origin', and 'usage' attributes have to be considered (see Figure 5), but using the COSMIC terminology (for more details see Dumke and Abran [27]). These three attributes are defined in the context of this paper as the following:

- Type: it identifies for each functionality user requirements measured the number of Entries, Exits, Reads, and Writes.
- Origin: it classifies software as either new developed, enhanced or redeveloped in COSMIC.
- Usage: it identifies whether the software was delivered, or undelivered to the user: intermediates products can be measured and undelivered.

On the other hand, according to ISO 9126-4, the expended effort can be measured (as input) not only by time (as in IEEE Std.1045), but also by the cost. Therefore, the 'Staff-cost' primitive should be included in the IEEE model. Furthermore, the 'nature' attribute is also applicable for this primitive (see Figure 5 above).

V. Conclusion

Productivity concept is common in almost all areas and is related to the ratio of what is produced and what is consumed. In software development literature, productivity is a complex concept that needs to be tackled depending on the software project factors. The productivity of a software product can be seen as a process having an input and output and which can be influenced by different factors, as the following:

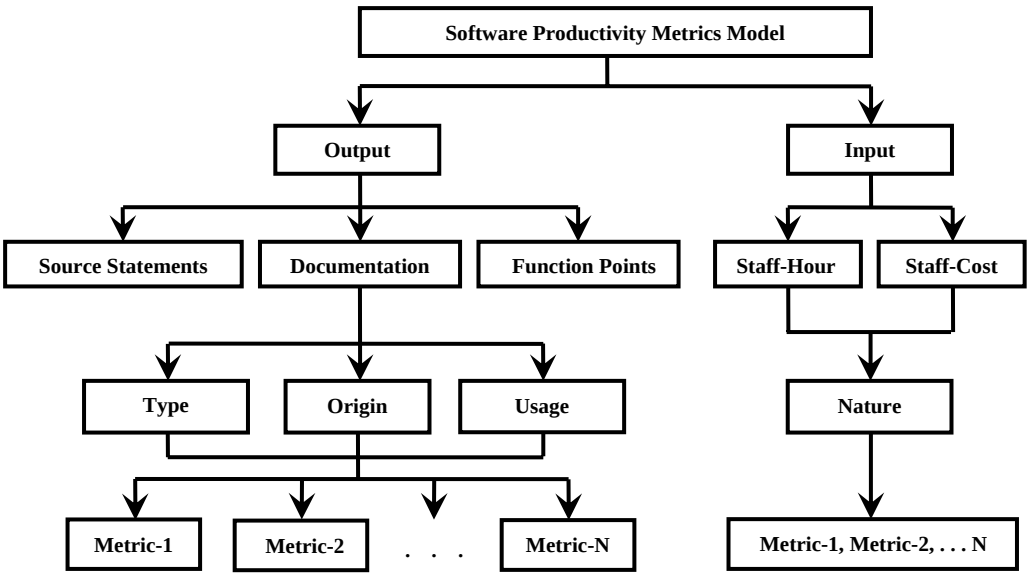

Figure 5. Enhanced software productivity metrics model.

- The input represents the entry of the process, which can be the effort, time, and cost expended to produce the output.
- The output represents the outcome of the process, which can be the product artifacts, the documentation, or the value of the outcome.
- The factors acts on the production process and influence positively or negatively the output. Such factors are organizational, technical, and personal factors included in the software development process.

This paper has addressed the software productivity in two international standards ISO 9126-4 and IEEE Std. 1045 to figure out how they define the productivity and how they can be used together to provide the researchers and practitioners with standardized aspects to measure productivity:

- The productivity in ISO 9126-4 is a quality characteristic related to the amount of resources used to enable the users to achieve a specified goal (a set of tasks) in a specified context of use. This definition is more generic and allow to measure productivity during the use of the software product (dynamic aspect of productivity). In addition, this ISO standard provides a set of measure to allow measuring the productivity.
- In IEEE Std. 1045 there is a productivity metrics model which is specific to the source statements, documentation, and function points outputs (static aspect of productivity). However, such model allows measuring the productivity of produced software product artifacts. Furthermore, this model is well-structured with three levels that allow the collection of data for productivity metric with the corresponding unit of measures. This standard also includes a set of productivity factors grouped in three categories, that is, project, management, and product.

Based on the detailed discussion conducted in section 3, the following improvements issues are recommended in both international standards:

- The productivity in ISO 9126-4 can be expressed in the second level by three sub-characteristics, which are, completeness, accuracy, and efficiency with their corresponding measures in the third level.
- The IEEE Std. 1045 software productivity metrics model can be used as a part of ISO 9126-4 productivity model, especially when the output is related to source statements, documentation, and/or function points primitives..
- The IEEE Std. 1045 software productivity metrics model have to take into account the function points output as a non-optional primitive with its three attributes using the COSMIC standardized measurement method. The staff-cost primitive can also be added as a measure of the input.

On the other hand, from the big number of volumes which have been written (by individual researchers and international standards) to determine the key factors that influence productivity, the product characteristics, people (stakeholder, manager, developer, etc.), software development processes, and technology, methods and tools factors are recurring themes. Each work group has discussed the productivity factors in different way depending on its point of view, that is, without any input from other works or existing standards. This situation does not allow building productivity model with a consensus, and thus there are various terms for the same factor, or different terms for the similar theme. This later point have to be tackled in more details in future work in order to propose, from the existing empirical, experimentation, and systematic literature review studies a standard-based model with the key drivers that influence the productivity of the software product during the project lifecycle.

REFERENCES

[1] Oxford English Dictionary, 2nd edition, Oxford, UK: Oxford University Press, 2010.

[2] G. S. De Aquino and S. R. De Lemos Meira, "Towards Effective Productivity Measurement in Software Projects", *Proceedings of the 4th International Conference on Software Engineering Advances - (ACSEA'09)*, Porto, Prazil, September 20-25, 2009, pp. 241-249.

[3] L. Buglione and A. Abran, "Multidimensional Software Performance Measurement Models: A Tetrahedron-Based Design", In R. Dumke and A. Abran (Editors), *Software Measurement: Current Trends in Research and Practice*, Germany: Springer Verlag GmbH, 1999, pp 93–107.

[4] N. E. Fenton and S. L. Pfleeger, *Software Metrics: A Rigorous and Practical Approach*, 2nd ed. Boston, MA, USA: PWS Publishing Company, 1997.

[5] ISO, *ISO/IEC 9126-1: Information Technology - Product Quality - Part 1: Quality Model*, Geneva, Switzerland: International Organization for Standardization, 2001.

[6] ISO, *ISO/IEC TR 9126-2: Information Technology - Product Quality - Part 2: External Metrics*, Geneva, Switzerland: International Organization for Standardization, 2003.

[7] ISO, *ISO/IEC TR 9126-3: Information Technology - Product Quality - Part 3: Internal Metrics*, Geneva, Switzerland: International Organization for Standardization, 2003.

[8] ISO, *ISO/IEC TR 9126-4: Information Technology - Product Quality - Part 4: Quality in Use Metrics*, Geneva, Switzerland: International Organization for Standardization, 2004.

[9] J. Vosburg, B. Curtis, R. Wolverton, B. Albert, H. Malec, S. Hoben, and Y. Liu, "Productivity Factors and Programming Environments", *Proceedings of the 7th. IEEE International Conference on. Software Engineering - (ICSE'84)*, Orlando, Florida, USA, March 26-29, 1984, pp. 143-152.

[10] IEEE, *Std 1045: IEEE Standard for Software Productivity Metrics*, New York, USA: Institute of Electrical and Electronics Engineers, 1992.

[11] A. Trendowicz, M. Ochs, A. Wickenkamp, J. Münch, Y. Ishigai, and T. Kawaguchi, "An Integrated Approach for Identifying Relevant Factors Influencing Software Development Productivity", *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, Vol. 5082/2008, 2008, pp. 223-237.

[12] S. Wagner and M. Ruhe, "A Systematic Review of Productivity Factors in Software Development", *Proceedings of the 2nd International Workshop on Software Productivity Analysis and Cost Estimation,*

(SPACE 2008), Technical Report ISCAS-SKLCS-08-08, State Key Laboratory of Computer Science, Institute of Software, Chinese Academy of Sciences, 2008.

[13] T. Sterling, "Productivity Metrics and Models for High Performance Computing", *International Journal of High Performance Computing Applications*, Vol. 18, No. 4, 2004, pp. 433-440.

[14] K. S. White, "Software Engineering Management for Productivity and Quality", *Proceedings of the 7th International Conference on Accelerator and Large Experimental Physics Control Systems - (ICALEPCS'99)*, Trieste, Italy, October 4-8, 1999, pp. 317-317.

[15] E. López-Ortega and R. Saloma-Velazquez, "A Worker Productivity Model", *Proceedings of the 20th International Conference on System Dynamics*, Palermo, Italy, July 28 - August 1, 2002.

[16] K. D. Maxwell, L. Van Wassenhove, and S. Dutta, "Software Development Productivity of European Space: Military and Industrial Applications", *IEEE Transactions on Software Engineering*, Vol. 22, No. 10, 1996, pp. 706-718.

[17] K. D. Maxwell and P. Forselius, "Benchmarking Software Development Productivity", *IEEE Software*, Vol. 17, No. 1, 2000, pp. 80-88.

[18] M. He, M. Li, Q. Wang, Y. Yang, and K. Ye, "An Investigation of Software Development Productivity in China", *Proceedings of the International Conference on Software Process: Making Globally Distributed Software Development a Success Story – (ICSP'08)*, Leipzig, Germany, May 10-11, 2008, pp. 381-394.

[19] J. D. Blackburn, G. D. Scudder, and L. N. Van Wassenhove, "Improving Speed and Productivity of Software Development: A Global Survey of Software Developers", *IEEE Transactions on Software Engineering*, Vol. 22, No. 2, 1996, pp. 875-885.

[20] A. S. Duncan, "Software Development Productivity Tools and Metrics", *Proceedings of the 10th International Conference on Software Engineering - (ICSE'88)*, Singapore, April 11-15, 1988, pp. 41-48.

[21] H. Wang, H. Wang, and H. Zhang, "Software Productivity Analysis with CSBSG Data Set", *Proceedings of the IEEE International Conference on Computer Science and Software Engineering - (ICCSSE'08)*, Wuhan, Hubei, China, December 12-14, 2008, pp. 587-593.

[22] M. S. Krishnan, C. H. Kriebel, S. Kekre, and T. Mukhopadhyay, "An Empirical Analysis of Productivity and Quality in Software Products", *Management Science*, Vol. 46, No. 6, 2000, pp. 745-759.

[23] W. Scacchi, "Understanding Software Productivity: a Comparative Empirical Review", *Proceedings of the 22nd Annual Hawaii International Conference on System Sciences*, Kailua-Kona, HI, USA, January 3-6, 1989, pp. 969-977.

[24] Y. Jun, "Towards Adaptive Project Tracking Using Individual Productivity Metrics", *Proceedings of the 6th International Conference on Machine Learning and Cybernetics*, Hong Kong, August 19-22, 2007, pp. 19-22.

[25] ISO, *ISO/IEC 19761: Software Engineering - COSMIC-FFP - A Functional Size Measurement Method*, Geneva, Switzerland: International Organization for Standardization, 2003.

[26] A. Abran, *Analyse du Processus de Mesure des Points de Fonction*, PhD Thesis, Ecole Polytechnique de Montréal, Canada, 1994.

[27] R. Dumke and A. Abran, *COSMIC Function Points: Theory and Advanced Practices*, Germany: CRC Press, Taylor and Francis Group, Auerbach Publications, 2011.

Laila Cheikhi is a Professor at Computer Science and Systems Analysis School (ENSIAS, Rabat, Morocco). She received a M.Sc. (2004) from University of Montréal and Ph.D. (2008) from ETS, University of Quebec at Montreal, and Both in software engineering. She has over eight years of experience in computer engineering at the Ministry of Finance of Morocco. Her research interests include software quality models, software metrics, software engineering ISO standards, software product and process quality, software engineering principles and data analysis.

Rafa E. Al-Qutaish received the B.Sc. in Computer Science and M.Sc. in Software Engineering degrees in 1993 and 1998, respectively. Also, he received the Ph.D. degree in Software Engineering from the School of Higher Technology (ÉTS), University of Québec, Canada in 2007. Currently, he is a Deputy Dean and an Assistant Professor of Software Engineering at Al Ain University of Science and Technology in Abu Dhabi, UAE. His research interests are in Software Measurement, Software Product Quality, Software Engineering Standardization, Reverse Engineering, Software Comprehension and Maintenance, and Compiler Construction. So far, he has published more than 20 papers in international peer-reviewed journals, 15 papers in international peer-reviewed conferences & workshops, and 3 chapters. Furthermore, Dr. Al-Qutaish is a senior member of the IEEE & IEEE-CS, and also a senior member of the IACSIT. http://www.rafa-elayyan.ca

Ali Idri is a Professor at Computer Science and Systems Analysis School (ENSIAS, Rabat, Morocco). He received DEA (Master) (1994) and Doctorate of 3rd Cycle (1997) degrees in Computer Science, both from the University Mohamed V of Rabat. He has received his Ph.D. (2003) in Cognitive Computer Sciences from ETS, University of Quebec at Montreal. His research interests include software cost estimation, software metrics, fuzzy logic, neural networks, genetic algorithms and information sciences