Towards a validation process for measuring efficiency: integrating axiomatic and empirical approaches.

Miguel Lopez[1], Valérie Paulus[1], Naji Habra[2]

[1]CETIC asbl, Rue Clement Ader, 8
6041 Gosselies Belgium
{malm,vp}@cetic.be
http://www.cetic.be

[2] University of Namur, Rue Grandgagnage, 21
5000 Namur Belgium
nha@info.fundp.ac.be
http://www.info.fundp.ac.be/~software-quality

Abstract

The validity of the measures used in software engineering is a critical matter about which no consensus has yet emerged, although it has prompted hard discussion. There is a need for unambiguous definitions of the mathematical properties that characterize the major measurement concepts. Such a mathematical framework could help to generate consensus among the software engineering community.

The goal of this paper is to provide a formal validation process for software measurement. It presents a global measurement framework that integrates theoretical and empirical validation processes based on measurement theory. The concept underlying the framework is to formalize some properties of the measure to be analyzed, and then to verify the conformity of these properties to the measure by means of formal experimentation.

This validation process determines a contextual validity (scope) defined by the set of factors or validity conditions that impact the validity of the measure. The paper develops a case study which, under specified conditions, validates the response time as a measure of efficiency, as defined by ISO/IEC standard 9126.

Keywords: *Axiomatic validation approach, empirical validation, formal validation, measurement theory, web applications, web metrics, contextual validity, efficiency, response time, ISO/IEC standard 9126*

1. Introduction

The aim of this paper is to give an outline of how to validate a measure on the basis of the principles of measurement theory. A measure validation is defined as "the process which controls that a measure represents correctly the attribute it has to measure" [FENT96].

The validity of the measurements used in software engineering is a critical matter about which no consensus has yet emerged, although it has prompted hard discussion. There is a need for unambiguous definitions of the mathematical properties that characterize the major measurement concepts. Such a mathematical framework could help in reaching consensus among the software engineering community.

The goal of this paper is to provide a formal validation process for software measurement. It presents a global measurement framework that integrates theoretical and empirical validation processes based on measurement theory. The concept underlying the framework is to formalize some properties of the measure to be analyzed, and then to verify the conformity of these properties to the measure by means of formal experimentation.

This validation process determines a contextual validity (scope) defined by the set of factors or validity conditions that impact the validity of the measure. The paper develops a case study which, under specified conditions, validates the response time as a measure of efficiency, as defined by ISO/IEC standard 9126.

This paper is organized as follows. First, in Section 2 some definitions of measurement validity are explained and a new definition proposed. Next, in Section 3, a validation procedure based on the validity definition proposed in Section 2 is explained. This validation procedure is a twofold approach: first the specifications of the empirical and mathematical system are given; then these two systems are validated theoretically and experimentally. This is followed in Section 4 by the experimental validation of the preservation of the empirical order in the mathematical system. Lastly, we draw conclusions and discuss avenues for future research.

2. Measure validity

2.1. The lack of consensus

Bieman's definition

[BIEM92] gives the following definition of a valid measure: "a software measure is only valid if it can be shown to be an accurate predictor of some software attribute". This definition highlights the need to know what the measure really measures before it can be validated.

Representational condition

Another approach for validating a measure is based on the representational condition. *A measure is valid if it satisfies the representation condition: if it captures in the mathematical world the behavior we perceive in the empirical world* [FENT96]. If Mr A is taller than Mr B and if μ is a measure of the human size, then μ(A) should be greater than μ(B). The order perceived in the empirical world (Mr A is taller than Mr B) must be kept in the mathematical world (μ(A) is greater than μ(B)). This is a necessary condition but insufficient for validating the measure. The measure must satisfy other kinds of properties. For example, the measure of size must be positive. Thus a validation based only on the representational condition would not correctly validate the measures. This kind of validation does not take into account an understanding (model) of the attribute to be measured. In the above example, the measure of human size must be positive. The model of human size assumes that this measure is positive. In this sense, the representational condition is insufficient.

IEEE Definition

In [IEEE93], a valid measure is defined as a measure *whose values have been statistically associated with corresponding quality factor values*. This definition gives a necessary condition which is insufficient. In [ZUSE99], H. Zuse affirms that *Using only statistics without knowing the models* (understanding the attribute) *behind valid measures and prediction models does not lead to solid results.* The software engineer needs attribute models based on environment hypothesis to validate his/her measures. The understanding of software attributes is not accepted by the scientific community. For example, the behavior of a program with respect to the operation of concatenation can be controversial. Is program P, as a concatenation of programs P_1 and P_2, more efficient than the individual programs P_1 and P_2? The answer is not clear. There is thus a lack of consensus concerning software attribute models.

Need for consensus

This set of three definitions is not exhaustive but indicates the diversity of meanings attributed to the term validity. It is important that the scientific community, together with the industry, agree about the meaning of validation. Software engineers will use the measures if and only if they can work with a serious and robust definition of validity. A serious and robust validation process can be based on measurement theory. Measurement theory is a mathematical framework that would facilitate consensus among the experts (scientists and the industry).

2.2. Hypothesis regarding the environment

In [HEND96], Henderson-Sellers affirms that *(...) many metrics are validated against only one data set. This does not, in itself, render the validation process invalid but cannot be used to justify anything other than a very restricted and careful use of the metric.* The validation process is carried out under specific conditions that can impact on measure validity. These conditions can have such an impact on validity that changes in any one of them can invalidate the measure. It is important to give a precise description of the conditions of validation and to urge the user of the measure to verify the conditions of his/her environment.

The conditions of validity are in fact the hypothesis of the software attribute model. These assumptions capture the whole understanding of the attribute. In the example of software efficiency, the current understanding assumes that an empty program (without any statement) must have a null response time. This assertion is a hypothesis of the software efficiency model. A complete model needs to be established for the state of the art.

2.3. Expert knowledge

Experts build the model of the software attribute to be measured. This model captures their understanding of the attribute for which a consensus is reached. This model includes validity conditions, their impacts on validity and the properties of the measure (empirical order preservation, positivity, etc.). The model is an evolving one, and requires frequent revision.

2.4. Proposed definition

A measure is valid if it satisfies a set of mathematical properties or axioms that model the attribute to be measured. This implies that a group of experts must establish a set of axioms by consensus. The representational condition is a mandatory property for all measures. In fact, this is not true for all properties, e.g. the non-negativity example above. The model must capture the experts' knowledge concerning the software attribute by means of mathematical concepts defined in measurement theory [ROB79]. This theory provides a rigorous framework for the software engineer and would facilitate the achievement of consensus among the experts.

3. Proposed Validation Procedure

3.1. Aims of measurement

The aim of the measure to be validated as an example of this framework is to allow the comparison of pairs of programs in terms of software efficiency. A valid measure should provide the basis for a judgment on whether program A is faster, i.e. more efficient, than program B. For this we refer to the efficiency definition given in ISO/IEC standard 9126 [ISO9126]: *"The capability of the software product to provide appropriate performance, relative to the amount of resources used, under stated conditions."*

3.2. Specifications of the empirical relational system

Characterization of the attribute to be measured

In this study, we are particularly interested in the time behavior of the software product. In the quality model of ISO/IEC standard 9126, time behavior is a sub-characteristic of efficiency and is defined as follows: *"The capability of the software product to provide appropriate response and processing times and throughput rates when performing its function, under stated conditions."*

Measurement theory specifies the conditions under which the empirical and numeric worlds can be combined. This theory translates empirical properties into mathematical properties [ZUSE97].

According to measurement theory, in our case, we consider:

– The set ***A***: which is composed of the set of all n-tiers Internet architecture programs in PHP.
– A binary relationship R in ***A***: a relationship between programs is needed to express the comparison.

The set of pairs R includes all pairs of programs related by 'behave more efficiently than'. In the case of set ***A***, we assume that this relationship is equal to 'is faster than'.

With sets ***A*** and R we have an empirical relational system, *U*=(***A***,R). We claim that this system is empirical because it groups real world entities.

To refine our definitions, let us propose that:

– ***A*** = the set of n-tiers Internet architecture programs in PHP
– R = the relationship 'is faster than', denoted by "$\succ$"

Set A

Set ***A*** could not be described exhaustively and a full description appears unnecessary. We define ***A*** with paradigms, i.e. ***A*** includes all web-based applications with real case, for example:

– An empty PHP script: with no line of code
– A PHP script which executes a set of instructions without communication to a third party application
– A PHP script which communicates to a third party application

It is not practicable to consider all the web-based applications of a domain. Instead we propose to choose a representative sample of the theoretical set ***A*** on the basis of the sub-characteristics we want to measure. This representative set shall be denoted by set A. In other words, as the object of validation is the measure of time behavior, it is evident that the chosen set A will include scripts which suitably represent the efficiency problem.

Set A is a set of real world objects, but these objects are not practical ones. They represent various typical situations encountered in practice. For this reason, experts must specify the content of set A. Here we shall assume that the set of paradigms is representative of the problem of n-tiers Internet architecture. This should be considered as a working hypothesis for this paper, whose aim is to present a validation process and to validate internally a measure of efficiency.

Relationship

From experience, we know that if a PHP script A_1 contains three instructions and a PHP script A_2 contains A_1 and a bloc of instructions executing an SQL request, then we could say that:

$$A_1 \succ A_2 \quad (1)$$

So now we have:

– A: a set of n-tiers Internet architecture application paradigms for a given domain
– R: the relationship "is faster than", "$\succ$" to A
– *U*: (A, $\succ$): a relational empirical system

Specifications of system U

Set A has a finite number of elements. For pairs of programs to be compared, set A must be an ordered set, in other words a weak order. To have a weak order the following axioms must be satisfied:

– Strongly complete: $\forall a,b \in A : a \succ b \vee b \succ a$ (2)
– Transitive: $\forall\ a, b, c \in A : a \succ b \wedge b \succ c \Rightarrow a \succ c$ (3)

In other words, axiom (2) means that for any pair of programs (a,b) in set A, either a is faster than b or b is faster than a. Axiom (3) can be read: if a is faster than b and b is faster than c then a is faster than c. The satisfaction of these two axioms allows comparison between elements of set A.

Let us consider A to be the following finite set of PHP web-based paradigms described at length:

$$A = \{A_1, A_2, A_3, A_4, A_5, A_6, A_7, A_8\}$$

A_1 = PHP 4.2.1 script without any instructions and a blank

A_2 = PHP 4.2.1 script with 4 instructions: a test 'if' on two integers

A_3 = PHP 4.2.1 script with 2 instructions that declares a session variable and assigns the value of the environment variable HTTP_USER_LANGUAGE

$A_4 = A_2 + A_3$: PHP 4.2.1 script containing A_2 and A_3

A_5: writes one line (a real number) in a text file

$A_6 = A_3 + A_5$: writes in a text file and declares a session variable

$A_7 = A_3 + A_5$: the same script as A_6 but with a loop that writes a line in text file1,000 times

A_8: the same script as A_7 but instead of 1,000 times, it writes the same line 5,000 times

The elements of set A are chosen in order to avoid the verification procedure being affected by the problem of the sensitivity of the measurement tool.

The relationship *R* could be written as follows:

$$A_1 \succ A_2 \succ A_3 \succ A_4 \succ A_5 \succ A_6 \succ A_7 \succ A_8 \qquad (4)$$

However, this definition of system *U* must be given by experts, who will establish the sequence expressed in (4) by consensus. We prefer another notation for the set *R*:

$$R = \{(A_1, A_2), (A_2, A_3), (A_3, A_4),(A_1, A_3),(A_1, A_4), (A_2, A_4), \dots , (A_6, A_7),(A_6, A_8), (A_7, A_8)\} \quad (4a)$$

The expert who made this classification is faced with scripts easy to classify by response time. It is assumed that the distinction between the two scripts, which differ by a single instruction, is more a problem of measure instrument sensibility than a problem of judgment by the experts. New elements can be added to A; the set is not exhaustive.

3.3. The group of experts

The aim of this section is to introduce the problem of the choice of experts.

Role of the experts

Determining the empirical system, i.e. the specification of sets A and R, is a task for experts in the field of PHP web-based applications (in this example). The job of the experts is to express their understanding of time behavior by means of mathematical properties. Clearly, these mathematical properties are the axioms quoted above. It is assumed that these axioms represent the current state of knowledge in matters of time behaviour in PHP web-based applications.

Choice of experts

The choice of experts raises the issue of mutual recognition. We can choose a group of experts in our quality software laboratory. The problem is whether their expertise will be recognized by other experts and by users of the measure in the validation process. Since we claim that the list of axioms is exhaustive (in the current state of knowledge), the choice of experts will be critical for the validity of measures. The question is whether the measures should be validated privately or within the wider industrial and scientific community.

Suppose that software development company C_1 want to validate some measure in their own context (i.e. in their own validity conditions). C_1 must select some experts to determine the axioms that have to be validated. Then suppose that company C_2 wants to use the measures validated by C_1. The measure's context of use is the same in C_2 as in C_1. If the management of C_2 does not know the group of experts selected by C_1, how can they trust the latter's validation findings?

In general terms, how can we trust the completeness and relevance of axioms determined by experts we don't know? The case study below describes a validation carried out privately. If the aim is to reach a consensus within the industrial and scientific community, it is evident that private validation is purely anecdotal. We strongly believe that to obtain a global consensus, the specification of axioms must be led at international level by a group of experts. In our view, ISO is the right organisational forum for this to happen.

3.4. Specifications of the numeric relational system

Mathematical assignment rules

The aim of this section is to define the measure of time behavior. ISO/IEC standard 9126 gives us as time behavior measure: the *response time*.

On the basis of measurement theory, we could consider:

– B = the set of the reals, Re
– R = a binary relationship in B

So we have *B* (B,R) a numeric relational system. In this case we could replace B by Re and R by the relationship "is smaller than" or "<".

In measurement theory, a measure is defined as a function from *A* to *B* that preserves the relationships from system *U* in system *B.* This function is called homomorphism. It is a mapping between the empirical relational system and the numeric relational system. If μ is a measure, then homomorphism may be expressed as follows:

$\mu : A \rightarrow B : \forall\, a, b \in A : a \succ b \Leftrightarrow \mu(a) < \mu(b)$ (5)

If we consider that μ, in expression (5), represents response time, we could state that this expression is an axiom. Axiom (5) implies that if, empirically, the experts note that program a is faster than program b, then the measure of response time of a is smaller than the measure of response time of b and vice versa. If the measure of response time of a is smaller than the measure of response time of b, then a is faster than b. This axiom ensures that the relationship in the empirical world is preserved in the numerical world.

Specifications of system B

System *B* is a numerical relational system. The measure has to satisfy axioms (2) to (6) because μ must be a homomorphism.

$\mu:A\rightarrow B : \forall\, a,b \in A : a\succ b \Leftrightarrow \mu(a)<\mu(b)$ (5)
Strongly complete: $\forall a,b \in A : a \succ b \ \mathbf{v}\ b\succ a$ (2)
Transitive: $\forall\, a,b,c \in A : a\succ b \wedge b\succ c \Rightarrow (a\succ c)$(3)

Table 1: Axioms

Axioms (2) and (3), by homomorphism, may be written as follows:

- Strongly complete: $\forall\ \mu(a),\mu(b)\in B : \mu(a) > \mu(b)$ v $\mu(b)>\mu(a)$ (2a)
- Transitive: $\forall\ \mu(a),\mu(b),\mu(c) \in B: \mu(a)>\mu(b) \wedge \mu(b)>\mu(c) \Rightarrow (\mu(a)>\mu(c))$((3a)

$\mu:A\rightarrow B : \forall\, a,b \in A : a\succ b \Leftrightarrow \mu(a)<\mu(b)$ (5)
Strongly complete: $\forall\ \mu(a),\mu(b)\in B : \mu(a) > \mu(b)$ v $\mu(b)>\mu(a)$ (2a)
Transitive: $\forall\ \mu(a),\mu(b),\mu(c) \in B : \mu(a)>\mu(b) \wedge \mu(b)>\mu(c))\Rightarrow(\mu(a)>\mu(c))$ (3a)

Table 2: Revised axioms

3.5. Specifications of the instrument of measure

Null value

Null value: $\forall\ a \in A : a = \emptyset \Leftrightarrow \mu(a) = 0$ (6)

Axiom (6) is not necessary to compare pairs of programs. To do this, only a measure of time behavior which allows us to compare programs (i.e. to confirm that program a is faster than program b) is needed. But to obtain a valid measure, we must take this axiom into consideration, so μ has to satisfy axiom (6)

Non-negativity

Non-negativity: $\forall\ a \in A : \mu(a) > 0$ (7)

The same holds for axiom (7). The mapping μ must satisfy axiom (7), such that a negative measure will indicate a mistake in the measurement process (for example a defective instrument of measure.

3.6. Validation of system *U*

Strongly complete

Strongly complete: $\forall\ a,b \in A : a\succ b\ \mathbf{v}\ b\succ a$

The definition of R given by (4a) satisfies axiom (2a). All possible pairs are represented in set R and defined on the basis of the relationship "≻". Thus set A on the basis of the relationship "≻" is, by definition, strongly complete.

Transitive

Transitive: $\forall\ a,b,c \in A : a\succ b \wedge b\succ c \Rightarrow a\succ c$ (3)

Let set A be as defined in (4) and the set of relationships R as defined in (4a). Axiom (3) states that for all scripts a, b and c from A, if a is faster than b and b is faster than c then a is faster than c.

Proof: Take each pair of A defined according to the relationship "≻" and verify that transitivity is satisfied.

$$A_1\succ A_2,\ A_2\succ A_3 \Rightarrow A_1\succ A_3 \qquad (8)$$

On the basis of set R as defined in (4a), if A_1 is faster than A_2 and A_2 is faster than A_3, then A_1 is faster than A_3. The same comparison may be made for all elements in set A. The results of each comparison confirm the axiom of transitivity.

Set A based on the relation "≻" is thus transitive. Transitivity is satisfied by all elements in set A.

3.7. Validation of the instrument of measurement

Null value

If script a is empty, then measure μ(a) is null. If, during the measurement phase, a case occurs where script a is empty but measure μ(a) is not null, then it may be assumed either that measure μ is not valid or that an error is occurring during the measurement phase, for example a wrongly calibrated measure instrument. This axiom is not subject to verification but ensures the validity of the measure during measurement.

Non-negativity

Measure μ is positive. Should a negative measure be obtained, we affirm that either there is an error at the measurement state or measure μ is not valid. Once again, non-negativity is an axiom which will not be verified but which assures the user of a degree of validity in the collection of sample measures.

3.8. Validation of system B

Strong completeness

System *B* satisfies the axiom (5a) because *B* is the set of reals that is strongly complete. We could affirm that if μ(a) and μ(b) are real then either a is strictly smaller than b or b is strictly smaller than a.

Transitivity

If μ(a), μ(b) and μ(c) are real and μ(a) is smaller than μ(b) and μ(b) is smaller than μ(c), then μ(a) is smaller than μ(c). Transitivity is fully satisfied by the set of reals.

Relationships

Axiom (2) requires the relationships between the measures and the attributes of the object observed to be maintained. If script a is faster than script b, then the measure must reflect this relation, i.e. μ(a) should be smaller than μ(b). Verification of axiom (2) is based on a formal experiment. Once results have been collected, they should be subjected to a Spearman correlation test.

The rank correlation coefficient (Spearman coefficient) measures the relationship between a rank of observations of two characters X and Y. This coefficient detects the existence of monotone relations (increasing or decreasing).

To determine whether a relationship is significant, a hypothesis test must carried out as follows:

- Null hypothesis, H_0: there is no relationship between attributes X and Y.
- Set a significance level for rejecting the null hypothesis (e.g. $\alpha = 5\%$).
- Calculate the value of the Spearman coefficient r(X,Y).
- Calculate the theoretical value of the Spearman coefficient $r(\alpha,n)$ with the statistical tables; n is the degree of freedom.
- Test H_0 true if $r(\alpha,n) > | r(X,Y)|$.
- Accept or reject H_0.

The experiment and the results are described in detail in section 4. The structure of the experiment is based on [WOHL00].

3.9. Conditions of validity

Proposed definition

The conditions of validity may be considered as all the factors that influence the validity of the measure. These are the precise conditions under which the axiom validation experiments have been carried out and which could invalidate the measure. An invalid measure is a measure which does not satisfy one of the axioms in table 2. For example, in the case of response time, it is evident that processor speed and RAM quantity are major factors which influence the validity of the measure.

List of validity conditions

The definition of response time specified in ISO/IEC standard 9126 comprises two parts: execution time and time of command entry. In this example, the time of command entry can be disregarded, i.e. considered null, because it is a FOR loop that triggers the execution of the script. The formula for response time is then reduced to the execution time.

It is assumed that each factor of table 3 has an influence on the validity of the measure of response time.

1	Processor: frequency, type (Intel, PPC, …)
2	RAM: frequency, type, quantity
3	Hard disk: type, capacity (if disk access)
4	Operating system (version)
5	Network connectivity: web, LAN, …
6	Interpreter PHP version
7	Apache version
8	Browser: version, type
9	Developer maturity
10	Active process (CPU load)
11	Number of connected clients
12	Number of requests

Table 3: Conditions of validity

Discussion of conditions of validity

Each condition of validity must be specified as accurately as possible to make it easier to reproduce the axiom's experimental validation. Any change in one of the conditions of validity must systematically imply a repetition of the formal experiment described in Section 3.7. Where all the conditions of table 3 are satisfied, the response time as a measure of time behaviour is valid. Where the opposite is true, i.e. if there is at least one condition which is not satisfied, the measure of response time is not valid. Therefore, before any use is made of a measure, external validation is required. By external validation we mean a validation that enlarges the validity field of a measure. That means that at least one condition of validity has been modified and that experimental validation has been carried out in the new context.

The axioms to satisfy are the preservation of relationships (5), null value (6) and non-negativity (7). However, the axioms of weak order (strongly complete and transitive) only concern set A and the set of reals. The validity condition does not impact on the respect of the axiom of weak order of the set of reals. If conditions are changed, the set of reals *Re* will preserve the property of weak order and transitivity. The same can be affirmed concerning set A, which contains the paradigms of n-tiers Internet architecture. However, the other axioms (5,6,7) may not be satisfied if the validity conditions are modified.

The number of active processes and the CPU load are among the validity conditions that have a strong impact on the validity of the response time. One experiment to verify this impact on axioms (5),(6) and (7) could be the verification of a suitable enhancement of the CPU load. A suitable enhancement means that the CPU load is enhanced by using a process often active on a server. The launch of an Office application would not apply, for example, because this kind of program is not often active on a server; the launch of a database server would be more appropriate.

4. Experimental validation of the homomorphism

4.1. Definition

Object of the study: the response time as a measure of time behavior in terms of efficiency

Purpose: The purpose is to validate the response time as a measure of time behavior in terms of efficiency.

Quality focus: The quality focus is the validity of the response time for measuring time behavior in terms of efficiency.

Perspective: The perspective taken is the researcher's.

Context: The experiment is run on a single computer (PPC G3 300 Mhz, 160 Ram) using a software tool to measure the response time of 8 programs. Table 4 gives the measurement conditions.

Processor	PPC 300 MHz G3 L2 Cache 512K
RAM	160 MB SDRAM
Hard disk	Toshiba 3,2 Go
OS	Mac OS 10.2.3
Network connectivity	None
PHP Version	4.1.2
Apache Version	1.3.26
Browser	Lynx Version 2.8.4pre.2
Developer maturity	5 years of experience
Active process (CPU load)	See Appendix A
Number of connected clients	1
Request number	1

Table 4: Validity conditions of the experiment

4.2. Planning

Context selection: The context of the experiment is the software quality lab of the university of Namur, and hence the experiment is run off-line (non-industrial environment). The experiment is specific since it focuses on the validity of the response time of a PHP application under Internet n-tiers architecture.

It addresses a real problem, i.e. the validity of the response time measure.

4.3. Hypothesis formulation

Null hypothesis, H_0: the response time (defined in SO/IEC standard 9216) is not a valid measure (does not satisfy axiom (5) of the empirical order preservation) of time behavior in terms of efficiency for an Internet n-tiers architecture PHP application. There is no correlation (measured as rho, the Spearman correlation coefficient) between the empirical order of the 8 programs and the response time of these programs. This idea can be expressed as:

H_0: rho = 0 with a level of significance $\alpha = 5\%$

Alternative hypothesis, H_1: rho > $r(\alpha, n)$ where $r(\alpha, n)$ is the theoretical value (see Spearman statistical table)

4.4. Variable selection

The independent variables are the validity conditions of table 4 and the dependent variable is the response time (measured in microseconds).

4.5. Experiment design

Blocking: The number of active process (CPU Load > 0%), the number of connected users, the number of requests to the web server, the operating system and all the validity conditions expressed in table 4 can affect the response time in a way that is not relevant for the scope of the current experiment. It is therefore appropriate to block these factors.

Balancing: The experiment uses a balanced design, implying that there is the same number of data (1,000 response times) for each program.

Standard design type: The experiment design is a paired comparison design of the type "one factor, two treatments". The factor is the order of the programs on the basis of efficiency and the two treatments are the empirical order and the numerical order.

4.6. Instrumentation

The response time is measured using a software timer. The timer distinguishes between two timestamps: the first for the time before the execution of the program and the second for the time after execution. The timer is precise to approximately one microsecond. Documentation of the timer is provided through the online documentation of the PHP native function microtime() [PHPDOC].

4.7. Validity evaluation

Internal validity is focused on *the relationship (...) observed between the treatment and the outcome, we must be sure (...) that it is not a result of a factor of which we have no control* [WOHL00]. The result cannot be generalized outside the scope of this study. There is a large number of tests (equal to the number of measures per program), which ensures good internal validity.

Concerning *construct validity*, the experiment is intended to prove that response time is a good measure of time behavior in terms of efficiency. The study is concerned with the relationship between theory and observation. Construct validity is therefore not considered critical.

The *conclusion validity* of the experiment is not a problem. This validity is concerned with the relationship between the treatment and the outcome. This is what the study aims to prove: the correlation between the empirical order and the numerical order.

4.8. Operation

Preparation

The subjects (experts) are informed that the aim of the experiment is to validate the response time. The number of subjects is 2. The subjects must be experts in Internet development.

The 8 programs are ready before the experiment is executed. The subjects can read the programs or execute them (without measuring) to estimate time behavior. They must fill in a single form, sorting the 8 programs in ascending order by time behavior. The programs are run on the same computer (see table 4 for the computer specifications) as the measurement.

The measurement tool (timer) is plugged into the programs measured. The subjects do not receive the programs with the timer.

Execution

The experiment is executed just once. The subjects must estimate the time behavior by consensus. The response time is then measured using the timer. Each program is measured 1,000 times in just one go.

Data validation

It is assumed that the distribution of the variable response time is a normal distribution. The test of the normality hypothesis has been done graphically. The average of the response time is a good estimation of the mathematical expectation owing to the large sample size (n= 1,000).

The distribution of the response time of the empty program (A_{11}) is bimodal (see figure 1). The CPU load of the Lynx text-based browser is less than the CPU load of MS Internet Explorer. So using Lynx instead of MIE produces a normal distribution (see figure 2)

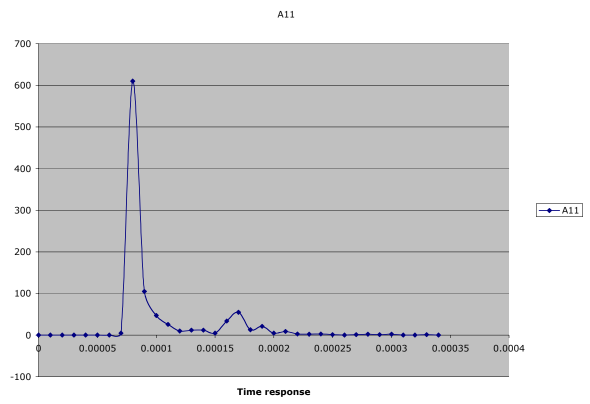

Figure 1: Script A1 with MS IE

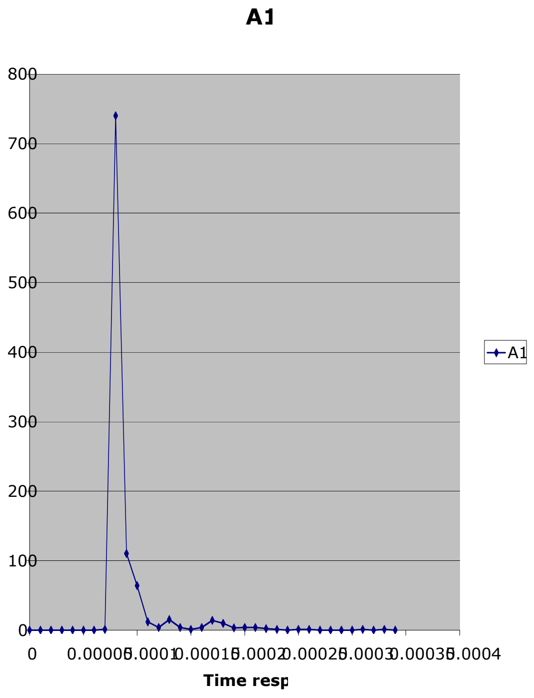

Figure 2: Script A1 with Lynx browser

The average response time of script A_{11} is not null (see table 6) but close to zero. The experimental zero is 1.3233E-04 seconds and this is a standard. The standard zero is used for calibrating the measurement tool, i.e. the timer. Using the standard zero, the axiom of nullity is satisfied (6).

Script	Response Time(s)
A11	8.33E-05
A2	1.35E-04
A3	1.07E-04
A4	1.52E-04

A5	9.42E-03
A6	9.45E-03
A7	1.19E-01
A8	5.59E-01

Table 6: Average response time

The non-negativity axiom is satisfied. None of the values collected is less than zero. The hypothesis of normality distribution and the average as a good estimation of the mathematical expectation point to the assumption that a negative value of the response time is not possible in this context.

Programs A2, A3, A4, A5, A6 present a normal distribution (see figure 3).

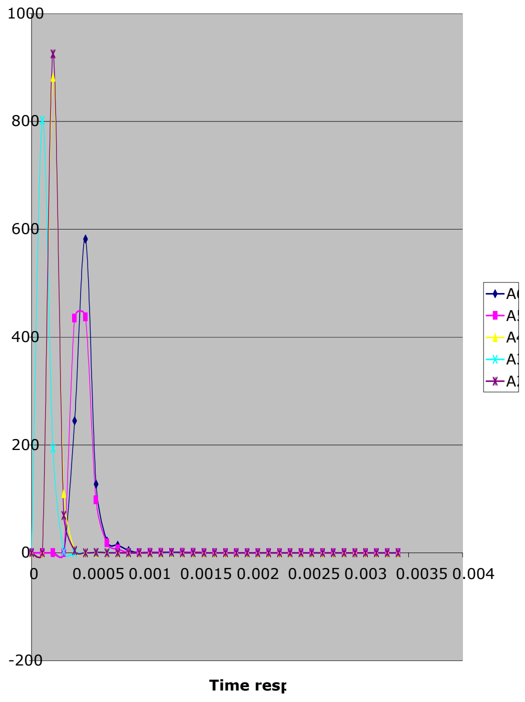

Figure 3: Distribution of response time

The distributions of programs A7, A8 are normal distributions (see figure 4).

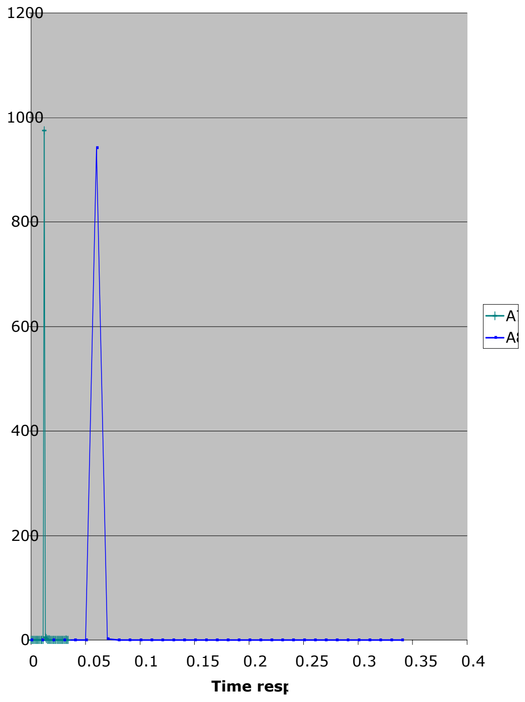

Figure 4: Distribution of response time

4.9. Analysis and interpretation

Descriptive statistics

Table 7 shows the estimation order as given by the experts (X) and the ranking of the 8 programs based on the response time average (Y). Column d_i is a term of the Spearman coefficient (rho) and represents the difference between X_i and Y_i (i is the rank).

	X	Y	d_i	$d_i * d_i$
A_{11}	1	1	0	0
A_2	2	3	-1	1
A_3	3	2	1	1
A_4	4	4	0	0
A_5	5	5	0	0
A_6	6	6	0	0
A_7	7	7	0	0
A_8	8	8	0	0

Table 7: Ranking

The Spearman coefficient equals 0.943 and the theoretical value at a level of significance $\alpha = 5\%$ equals 0.829. Thus the null hypothesis is rejected at a level of significance $\alpha = 5\%$.

The theoretical value at $\alpha = 2.5\%$ equals 0.886. Thus the null hypothesis is also rejected at $\alpha = 2.5\%$.

However, the theoretical value at $\alpha = 1\%$ equals 0.943. The null hypothesis must be accepted at a level of significance of $\alpha = 1\%$.

A correlation exists between the experts' estimation of time behavior and the response time measured. The probability that this correlation would occur by chance equals 2.5%.

5. Conclusion

The validation process presented here is composed of 7 steps:

1. Definition of the objectives of the measure
2. Selection of experts
3. Specification of an empirical relational system
4. Specification of a numerical relational system
5. Mathematical expression of the homomorphism
6. Validation of the empirical system
7. Validation of the numerical system

Steps 6 and 7 can be carried out on an experimental or theoretical basis. The validation of the weak order axiom for the set of programs is a theoretical validation, but the homomorphism axiom must be validated by formal experiment.

The objective of measurement determines the relational systems (empirical and numerical). It specifies the type of the empirical objects, the relationships between them and the operations. In the present paper, the specified relationship is "faster than". The objective is to compare different programs in terms of efficiency. In this case, it is not necessary to define operations for the programs. However, if the aim of measurement is to compare the efficiency of programs formed by the concatenation of other programs, the concatenation operation should be defined in the relational systems (empirical and numerical) with supplementary axioms for proof. The kind of question here is: "Is program $P_3 = P_1 + P_2$ faster than P_1 or P_2 ?" Such problems must be solved by experts, who must formally specify these properties by consensus.

The problem of the experts is one of confidence: how to establish whether the experts' knowledge is relevant and exhaustive.

The paradigms are also a critical point in the validation process. The choice of paradigms is made during the specification of the relational empirical system (step 3). The paradigms must reflect representative situations for the domain, the technology (client-server) and the point of measurement (software attribute, relations, operations). The problem of efficiency is a critical point in n-tiers architecture. In this architecture, the connections to a data source, session handling and other patterns must be taken into account when specifying the paradigms. This is true for any measure to be validated. Experts must elaborate patterns for each domain, technology or attribute. The paradigms become an instance of the previous patterns. This approach ensures the relevance and completeness of the paradigms.

It is assumed that the validity conditions impact on measure validity. The correlation between the validity conditions and measure validity must be verified through experimentation. It is also important to know how each condition separately influences measure validity. This experimental validation can often be hard to do. For example, how should the correlation between the measure validity and MS W2K be tested? For that, a change of processor is needed, and in this case two of the validity conditions are modified at the same time. It is therefore impossible to measure the OS impact on validity without modifying the processor. No correlation test was carried out here, as being outside the scope of this paper.

A measure is valid if it satisfies the properties (axioms) specified by experts. What is not expressed in a mathematical way (measurement theory) lies outside the scope of the validation process. Measurement theory is a framework which reduces the ambiguity of the expression *measure validity* and enhances the operability of the validation process.

This paper validates response time as measure of time behavior indicating software efficiency. Software practitioners can use the response time as defined above if and only if all the validity conditions are satisfied.

6. References

[BIEM92] Bieman, J.M.; Schultz, J.
"An Empirical Evaluation (and Specification) of the all-du-paths." Testing Criterion, *Software Engineering Journal*, Volume 7, No. 1, pp. 43-51, January 1992

[FENT96] Norman E. Fenton, Shari L. Pfleeger
Software Metrics: A Rigorous Approach, Chapman & Hall, 1996

[HEND96] Henderson-Sellers
Object-Oriented Metrics – Measures of Complexity, Prentice Hall, 1996

[IEEE93] IEEE Computer Society
IEEE Standard for a Software Quality Metrics Methodology, IEEE Standard 1061

[ISO9126] Software Engineering-Product Quality
"Part 1: Quality Model", ISO/IEC TR 9126-1, 1999
"Part 2: External Metrics", ISO/IEC TR 9126-1, 1999

[JACQ99] Jean-Philippe Jacquet, Alain Abran, Robert Dupuis
Une analyse structurée des méthodes de validation de métriques, 1999

[PHPDOC]
http://www.php.net/manual/en/function.microtime.php

[ROBE79] Fred S. Roberts

“Measurement Theory with Applications to Decisionmaking, Utility, and the Social Sciences”, *Encyclopedia of Mathematics and its Applications*, Addison Wesley Publishing Company, 1979

[WOHL00] Claes Wohlin *et al.*
Experimentation in Software Engineering: An Introduction, Kluwer Academic Publisher,
2000

[ZUSE99] Horst Zuse
"Validation of measures and prediction models", International Workshop on Software Measurement, 1999