SMEF 2004

SOFTWARE MEASUREMENT EUROPEAN FORUM 2004

ROME
28-29-30 January 2004

Organizers

ISTITUTO DI RICERCA INTERNAZIONALE

IIR is the world's largest knowledge and skills provider with a global network of 47 companies and 112 operating units. IIR has specialist companies providing groundbreaking strategies and proven technical expertise in both business functional areas and vertical markets – including management, marketing, sales, leadership, projects, HR, service, telecommunications, IT, pharmaceuticals, retail financial services, insurance, investment, manufacturing, energy, services, and the public sector. Founded as a newsletter company in 1973, IIR quickly built on its research and thought leadership capabilities to become the market leaders in conferences, seminars, exhibitions, publications, performance improvement and eLearning. Every year, IIR works directly with 650,000+ business executives to give them the 'know-how' they need to achieve.
Company information is available at www.iir-italy.it.

DPO

DPO (Data Processing Organization Srl) is the Italy's leading company in software measurement. Since 1967, DPO has been a qualified supplier of services in the market of work & process organization and information systems. DPO's excellence areas are Software Measurement and Estimation, Requirements Management, Project & Risk Management; in those areas, DPO offers exhaustive consulting, documentation & training services. Besides producing the SFERA suite for Standard and Early & Quick Function Point Analysis and cost & time estimation for software projects, DPO provides specialized services and products to promote the continuous evolution of production and management processes for both public and private organizations. DPO's high quality standards come from its active role in the research field at international level and the continuous confirmation of certificated expertise of its personnel (eg IFPUG CFPS). DPO's professionals collaborate with acknowledged technical national and international committees in the software measurement area, as the Italian GUFPI-ISMA Function Point Counting Practices Committee, the Italian GUFPI-ISMA Software Benchmarking Committee, and the international COSMIC Full Function Point Measurement Committee.
Company information is available at www.dpo.it/english

Event partner

GUFPI-ISMA

The GUFPI-ISMA non-profit association (Gruppo Utenti Function Point Italia – Italian Software Metrics Association) was founded in 1992 to promote and support the development of quantitative techniques for software measurement and management, including the IFPUG Function Point method. GUFPI-ISMA has, among its members, some of the more representative Italian public and private "ICT intensive" organizations (both ICT suppliers and customers). The Association promotes presentations, information exchange and research by means of its own technical committees and meetings.
Association information is available at www.gufpi.org.

TABLE OF CONTENTS

Proposal of Metrics for Software Process Models

Félix García, Francisco Ruiz, Mario Piattini

Abstract

One of the main reasons of the growing interest in software metrics has been the perception that software metrics are necessary for software process improvement. Software companies are becoming more and more concerned about software process improvement, when they are promoting the improvement of the final products. Measurement is essential for understanding, defining, managing and controlling the software development and maintenance processes and it is not possible to characterize the various aspects of development in a quantitative way without having a deep understanding of software development activities and their relationships. In this paper a representative set of metrics for software process models is presented in order to evaluate the influence of the software process models complexity in their quality. These metrics may provide the quantitative base necessary to evaluate the changes in the software processes in companies with high maturity levels. To demonstrate the practical utility of the metrics proposed at model level, an experiment and its replica have been performed which has allowed us to obtain some preliminary conclusions about the influence of the metrics proposed on three sub-characteristics of the software process models maintainability: understandability, analysability and modifiability.

1. Introduction

One of the main reasons of the growing interest in software metrics has been the perception that software metrics are necessary for software process improvement [7]. Measurement is essential for understanding, defining, managing and controlling the software development and maintenance processes and it is not possible to characterize the various aspects of development in a quantitative way without having a deep understanding of software development activities and their relationships [13].

Therefore, in order to provide the a quantitative basis for the improvement of software process it is necessary the measurement of the process, but before it is necessary to know the elements involved. For this reason the issue of process modelling has received growing attention in the software community during last years. The process model is seen as the starting point to analyse, improve and enact the process, but the need of strict coupling between process modelling and process measurement has not yet clearly emerge [11]. We have treated this issue in previous works [8;9].

In this paper a representative set of metrics for software process models is presented in order to evaluate the influence of the complexity in the software process models in their maintainability. These metrics are focused on the main elements included in a model of software processes, and may provide the quantitative base necessary to evaluate the changes in the software processes in companies with high maturity levels.

To demonstrate the practical utility of the metrics proposed at model level an experiment has been carried out which has allowed us to obtain some conclusions about the influence of the metrics proposed on three sub-characteristics of the software process models maintainability: understandability, analysability and modifiability.

Firstly, we present a set of representative metrics for the evaluation of software process models and an example of calculation. In Section 3 the empirical validation of the metrics proposed at model level is presented. Finally, some conclusions and further works are outlined.

2. Definition of Metrics for Software Process Models

Research on the software process evaluation has been focused in the collection of project data to obtain throughput, efficiency and productivity metrics and for this reason explicit metrics on software process models have not been defined.

The study of the possible influence of the complexity of software process models in their execution (enactment) could be very useful. For this reason the first step is the definition of a collection of metrics in order to characterise the software process model. The main objective to be achieved is the development of a empirical study to demonstrate the influence of the metrics proposed (which are applied on the attributes of software process models) on the maintainability of software process models. These metrics are indicators of a software process model structural complexity, and they could be every useful, taking into account that a software process model with high degree of complexity will be much more difficult to change, and this affects to their maintainability. This affirmation is based on the theoretical basis for developing quantitative models relating to structural properties and external quality attributes provided by [5] which is illustrated in figure 1. This basis could be applied to the software process in the same way they are applied to software artefacts. Software process models hardly maintainable affect to the execution of projects (more expensive in resources and schedule) and to the final quality of the software products obtained.

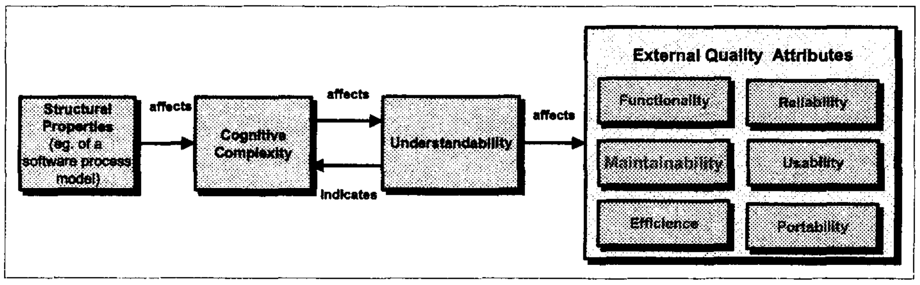

Figure. 1. Relationship between structural properties and external quality attributes

The metrics have been defined following the SPEM terminology [15], but they can be directly applied to other process modelling languages. The conceptual model of SPEM is based on the idea that a software development process consists of the collaboration between abstract and active entities, referred to as process roles, that carry out operations, called activities, on tangible entities, called work products. The basis of software processes consists of the interaction or collaboration of multiple roles through the exchange of work products and triggering the execution, or enactment of certain activities. The aim of a process is to bring a set of work products to a well-defined state.

SPEM has not a graphical notation by itself, but basic UML diagrams can be used to present different perspectives of a software process model. In particular, the following UML notations are useful: Class diagram, Package diagram, Activity diagram, Use case diagram and Sequence diagram. Figure 2 shows an example of a simplified software process model which belongs to the Rational Unified Process [10]. For the graphical representation of the model the Activity Diagram notation and the stereotypes which represent the SPEM constructors has been used:

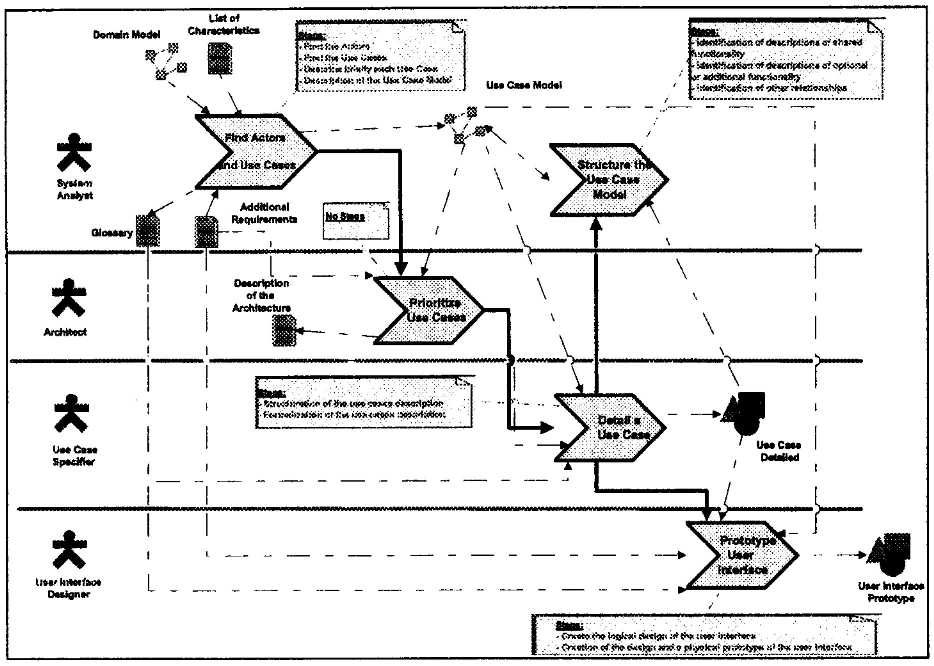
Domain Model
List of Characteristics
Steps:
- Find the Actors
- Find the Use Cases
- Describe briefly each Use Case
- Description of the Use Case Model
Use Case Model
Steps:
- Identification of descriptions of shared functionality
- Identification of descriptions of optional or additional functionality
- Identification of other relationships
System Analyst
Find Actors and Use Cases
Structure the Use Case Model
Glossary
Additional Requirements
No Steps
Architect
Description of the Architecture
Prioritize Use Cases
Steps:
- Structuration of the use cases description
- Formalization of the use cases description
Use Case Specifier
Detail a Use Case
Use Case Detailed
User Interface Designer
Prototype User Interface
User Interface Prototype
Steps:
- Create the logical design of the user interface
- Creation of the design and a physical prototype of the user interface

Figure 2. Example of a Software Process Model represented with SPEM

As we can observe in figure 2, using UML Activity diagrams it is possible to represent a view of the software process in which the different activities, their precedence relationships, the work products consumed or produced and the responsible roles are included.
The metrics have been defined examining the key software process constructors of the SPEM metamodel and they could be classified as model level metrics –they evaluate the characteristics of a software process model–, or as fundamental element (activity, process role and work product) metrics, –they describe the characteristics of a main model element–. In the following sections of this paper we focus on the definition and validation of the metrics proposed at model level.

2.1. Model Level Metrics

The model level metrics are applied in order to measure the structural complexity of the overall software process model. These metrics are represented in table 1:

Table 1. Model Level Metrics

Metric	Definition
NA(PM)	Number of **Activities** of the software process model
NWP(PM)	Number of **Work Products** of the software process model
NPR(PM)	Number of **Roles** which participate in the process
NDWPIn(PM)	Number of input dependences of the **Work Products** with the **Activities** in the process
NDWPOut(PM)	Number of output dependences of the **Work Products** with the **Activities** in the process
NDWP(PM)	Number of dependences between **Work Products** and **Activities** $NDWP(PM) = NDWPIn(MP) + NDWPOut(MP)$
NDA(PM)	Number of precedence dependences between **Activities**
NCA(PM)	Activity Coupling in the process model. $NCA(PM) = \dfrac{NA(PM)}{NDA(PM)}$
RDWPIn(PM)	Ratio between **input dependences** of Work Products with Activities and **total number of dependences** of Work Products with Activities

Metric	Definition
	$RDWPIn(PM) = \frac{NDWPIn(PM)}{NDWP(PM)}$
RDWPOut(PM)	Ratio between **output dependences** of Work Products with Activities and **total number of dependences** of Work Products with Activities $RDWPOut(PM) = \frac{NDWPOut(PM)}{NDWP(PM)}$
RWPA(PM)	Ratio of **Work Products** and **Activities.** Average of the work products and the activities of the process model. $RWPA(PM) = \frac{NWP(PM)}{NA(PM)}$
RRPA(PM)	Ratio of **Process Roles** and **Activities** $RRPA(MP) = \frac{NRP(MP)}{NA(MP)}$

In the table 2 the values of these metrics for the example of the figure 2 are shown:

Table 2. Values of Model Level Metrics

Metric	Value	Metric	Value
NA(PM)	5	NDA(PM)	4
NWP(PM)	8	NCA(PM)	5/4= 1,25
NPR(PM)	4	RDWPIn(PM)	13/18=0,722
NDWPIn(PM)	13	RDWPOut(PM)	5/18=0,278
NDWPOut(PM)	5	RWPA(PM)	8/5=1,6
NDWP(PM)	18	RRPA(PM)	4/5= 0,8

3. Empirical Validation

In order to prove the practical utility of the metrics it is necessary to do empirical studies. In this section we describe an experiment and its replica we have carried out to empirically validate the proposed measures as early maintainability indicators. We have followed some suggestions provided in [16][12] and [6] on how to perform controlled experiments and have used (with only minor changes) the format proposed in [16] to describe them.

3.1. First Experiment

3.1.1. Definition

Using the GQM template [1] for goal definition, the goal of the experiment is defined as follows:

- **Analyse** Software Process Models (SPM) structural complexity metrics
- **For the purpose of** Evaluating with respect to the capability to be used as software process model maintainability indicators
- **From the point of view of** Software Process Analysts
- **In the context of** Undergraduate Computer Science students and professors of the Software Engineering area at the Department of Computer Science at the University of Castilla-La Mancha

3.1.2. Planning

- **Context selection**. The context of the experiment is a group of undergraduate students and professors of the Software Engineering area, and hence the experiment is run off-line (not in an industrial software development environment). The subjects were ten professors and ten students enrolled in the final-year of Computer Science at the Department of

Computer Science at the University of Castilla-La Mancha in Spain. All of the professors belong to the Software Engineering area. The experiment is specific since it is focused on SPM structural complexity metrics. The ability to generalize from this specific context is further elaborated below when discussing threats to the experiment. The experiment addresses a real problem, i.e., what indicators can be used for the maintainability of SPM? With this end in view it investigates the correlation between SPM structural complexity metrics and maintainability sub-characteristics.

- **Selection of subjects**. The subjects are chosen for convenience. The subjects are undergraduate students and professors who have wide experience and knowledge in software product modelling (UML, databases, etc.), but they have not experience or knowledge in the conceptual modelling of SPM.
- **Variable selection.** The independent variable is the **SPM structural complexity**. The dependent variables are three maintainability sub-characteristics: **understandability, analysability** and **modifiability**.
- **Instrumentation**. The objects were 18 SPM. The independent variable was measured through the metrics proposed at process model level (see section 2.1). The dependent variables were measured according to the subject's ratings.
- **Hypothesis formulation.** We wish to test the following hypotheses:
  - *Null hypothesis, H0*: There is no significant correlation between the structural complexity metrics (NA, NWP, NPR, NDA, NDWP, NDWPIn, NDWPOut, NCA, RDWPIn, RDWPOut, RWPA, RRPA) and the subject's rating of three maintainability sub-characteristics, such as understandability, analysability and modifiability.
  - *Alternative hypothesis, H1*: There is a significant correlation between the structural complexity metrics (NA, NWP, NPR, NDA, NDWP, NDWPIn, NDWPOut, NCA, RDWPIn, RDWPOut, RWPA, RRPA) and the subject's rating of three maintainability sub-characteristics, such as understandability, analysability and modifiability.
- **Experiment design**. We selected a within-subject design experiment, i.e., all the tests (experimental tasks) had to be solved by each of the subjects. The tests were put in a **different order** for each subject.

3.1.3. Operation

- **Preparation.** Subjects were given an intensive training session before the experiment took place. However, the subjects were not aware of what aspects we intended to study. Neither were they aware of the actual hypothesis stated. We prepared the material we handed to the subjects, consisting of eighteen SPM. These models were related with different universes of discourse but they were general enough to be understood by the subjects. The structural complexity of each diagram is different, because as table 3 shows, the values of the metrics are different for each diagram.

Table 3. Metric values for each software process model

Mod	NA	NWP	NPR	NDWPIn	NDWPOut	NDWP	NDA	NCA	RDWPIn	RDWPOut	RWPA	RRPA
1	6	6	3	5	6	11	6	1,000	0,455	0,545	1,000	0,500
2	5	6	4	5	5	10	4	1,250	0,500	0,500	1,200	0,800
3	2	13	2	12	3	15	1	2,000	0,800	0,200	6,500	1,000
4	9	25	9	25	21	46	11	0,818	0,543	0,457	2,778	1,000
5	5	6	4	5	5	10	8	0,625	0,500	0,500	1,200	0,800
6	4	11	4	14	9	23	3	1,333	0,609	0,391	2,750	1,000
7	8	17	1	15	11	26	9	0,889	0,577	0,423	2,125	0,125
8	5	8	4	13	5	18	4	1,250	0,722	0,278	1,600	0,800
9	7	12	1	12	11	23	6	1,167	0,522	0,478	1,714	0,143
10	24	37	10	72	40	112	24	1,000	0,643	0,357	1,542	0,417
11	7	12	5	12	11	23	6	1,167	0,522	0,478	1,714	0,714
12	2	8	3	6	4	10	1	2,000	0,600	0,400	4,000	1,500

Mod	NA	NWP	NPR	NDWPIn	NDWPOut	NDWP	NDA	NCA	RDWPIn	RDWPOut	RWPA	RRPA
13	3	6	1	8	3	11	4	0,750	0,727	0,273	2,000	0,333
14	3	5	7	5	3	8	2	1,500	0,625	0,375	1,667	2,333
15	4	9	1	9	7	16	6	0,667	0,563	0,438	2,250	0,250
16	8	6	4	9	9	18	7	1,143	0,500	0,500	0,750	0,500
17	4	24	1	20	11	31	3	1,333	0,645	0,355	6,000	0,250
18	5	21	3	21	11	32	4	1,250	0,656	0,344	4,200	0,600

Each model had a test enclosed which included the description of three maintainability sub-characteristics: understandability, analysability and modifiability. Each subject had to rate each sub-characteristic using a scale consisting of seven linguistic labels. For example for understandability we proposed the following linguistic labels shown in table 4.

Table 4. Linguistic labels for understandability

Extremely difficult to understand	Very difficult to understand	A bit difficult to understand	Neither difficult nor easy to understand	Quite easy to understand	Very easy to understand	Extremely easy to understand

We chose seven linguistic labels because we considered they are enough to cover all the possible categories of our maintainability variables.

- **Execution**. The subjects were given all the materials described in the previous paragraph. We explained to them how to carry out the tests. We allowed one week to do the experiment, i.e., each subject had to carry out the test alone, and could use unlimited time to solve it. We collected all the data, including subjects' rating obtained from the responses of the experiment and the metric values of the different SPM.
- **Data validation**. We collected all the tests, checking if they were complete. As all of them were complete we consider their subjective evaluation reliable.

3.1.4. Analysis and interpretation

First we summarized the data collected. We had the metric values calculated for each SPM, and we calculated the median of the subjects' rating for each maintainability sub-characteristic. So this is the data we want to analyse to test the hypotheses stated above. We applied the Kolmogorov-Smirnov test to ascertain if the distribution of the data collected was normal or not. As the data were non-normal we decided to use a non-parametric test like Spearman's correlation coefficient, with a level of significance $\alpha = 0.05$, which means the level of confidence is 95%.

Using Spearman's correlation coefficient, each of the metrics was correlated separately to the median of the subject's rating of understandability, analysability and modifiability (see table 5).

Table 5. Spearman's correlation between the metrics and understandability, analysability and modifiability

	NA	NWP	NPR	NDWPIn	NDWPOut	NDWP	NDA	NCA	RDWPIn	RDWPOut	RWPA	RRPA
Underst	**0,629**	**0,756**	0,149	**0,841**	**0,802**	**0,888**	**0,481**	-0,201	0,220	-0,220	0,189	-0,385
Analiz	**0,612**	**0,789**	0,042	**0,830**	**0,855**	**0,892**	**0,498**	-0,254	0,131	-0,131	0,227	-0,454
Modif.	**0,675**	**0,784**	0,148	**0,871**	**0,858**	**0,931**	**0,532**	-0,243	0,145	-0,145	0,173	-0,412

For a sample size of 18 (median values for each SPM) and $\alpha = 0.05$, the Spearman cutoff for accepting H0 is 0.4684 [17]. Because the computed Spearman's correlation coefficients (table 5) are above the cut-off, the null hypothesis H0, is rejected. Analysing the table 5 we can conclude that there is a significant correlation (rejecting the null hypothesis) between the structural complexity of the SPM and the metrics (NA, NWP, NDWPIn, NDWPOut, NDWP y NDA) because the correlation coefficient is greater than 0,4684. The metric RRPA seems to be less correlated with the three maintainability sub-characteristics respect to the prior metrics, although it has a correlation value near to the cutoff. The metrics NPR, NCA, RDWPIn, RDWPOut y RWPA seem not to be correlated with the maintainability.

3.1.5. Validity evaluation

We will discuss the empirical study's various threats to validity and the way we attempted to alleviate them:

- **Threats to conclusion validity**. The only issue that could affect the statistical validity of this study is the size of the sample data (360 values, 18 models and 20 subjects), which is perhaps not enough for both parametric and non-parametric statistic tests [3] . We are aware of this, so we will consider the results of this experiment only as preliminary findings.
- **Threats to construct validity**. The dependent variables are three maintainability sub-characteristics: understandability, analysability and modifiability. We proposed subjective metrics for them (using linguistic variables), based on the judgement of the subjects. The construct validity of the metrics used for the independent variables is guaranteed by Poels and Dedene's framework [14] and Briand et al. framework [4], which we have applied to theoretically validate them.
- **Threats to Internal Validity**. The following issues have been dealt with:
  - *Differences among subjects.* Using a within-subjects design, error variance due to differences among subjects is reduced. In this experiment, professors and students had approximately the same degree of experience in modeling software products and they have a minimum knowledge about process modeling, which not influence the results because they have product modeling skills and knowledge.
  - *Knowledge of the universe of discourse among SPM.* SPM were from different universes of discourse but general and well-known enough to be familiar to the subjects. Consequently, knowledge of the domain does not affect the internal validity.
  - *Accuracy of subject responses.* Subjects assumed responsibility for rating each maintainability sub-characteristic. As they have wide experience in product modelling by mapping this experience to the process modelling, we think their responses could be considered valid. However, we are aware that not all of them have exactly the same degree of experience, and if the subjects have more experience minor inaccuracies could be introduced by subjects.
  - *Learning effects.* The subjects were given the test in a different order, to cancel out learning effects. Subjects were required to answer in the order in which the tests appeared.
  - *Fatigue effects.* On average the experiment lasted for less than one hour (this fact was corroborated summing the total time for each subject), so fatigue was not very relevant. Also, the different order in the tests helped to cancel out these effects.
  - *Persistence effects.* In order to avoid persistence effects, the experiment was run with subjects who had never done a similar experiment.
  - S*ubject motivation.* All the professors who were involved in this experiment have participated voluntarily, in order to help us in our research. We motivated students to participate in the experiment, explaining to them that similar tasks to the

experimental ones could be done in exams or practice and it could be useful in their professional career.

- *Other factors*. Plagiarism and influence among students could not really be controlled. Students were told that talking with each other was forbidden, but they did the experiment alone without any supervision, so we had to trust them as far as that was concerned. We are conscious that this aspect at some extent could can threat to the validity of the experiment, but in that moment it was impossible to join all the subjects together.

- **Threats to External Validity**. Two threats of validity have been identified which limit the possibility of applying generalization:
  - *Materials and tasks used.* In the experiment we have used SPM which are representative of real cases. Related to the tasks, the judgement of the subjects is to some extent subjective, and does not represent a real task. So more empirical studies taking "real cases" from software companies must be done.
  - *Subjects*. To solve the difficulty of obtaining professional subjects, we used professors and advanced students from software engineering courses. We are aware that more experiments with practitioners and professionals must be carried out in order to be able to generalize these results. However, in this case, the tasks to be performed do not require high levels of industrial experience, so, experiments with students could be appropriate [2].

3.2. Second Experiment

In order to confirm the results obtained in the first experiment we replicated this experiment under the same conditions. As the majority of the steps are identical to those of the first experiment we will only point out those issues which are different. The subjects were fifteen professors and ten research technicians of the Alarcos research Group of Computer Science at the Department of Computer Science at the University of Castilla-La Mancha in Spain. All of the professors belong to the Software Engineering area.

We included a new dependent variable measured by the time that the subjects spent to completely understand the SPM before answering the subjective questions. We called this time "understandability time". Understandability time comprises the time to understand the class diagram. Our assumption here is that the faster a class diagram can be understood, the easier it is to maintain.

We wish to test the following new hypotheses:

- **Null hypothesis, H0**: There is no significant correlation between structural complexity metrics (NA, NWP, NPR, NDA, NDWP, NDWPIn, NDWPOut, NCA, RDWPIn, RDWPOut, RWPA, RRPA) and understandability time.
- **Alternative hypothesis, H1**: There is a significant correlation between structural complexity metrics (NA, NWP, NPR, NDA, NDWP, NDWPIn, NDWPOut, NCA, RDWPIn, RDWPOut, RWPA, RRPA) and understandability time.

The material we gave to the subjects was the same material provided in the first experiment, consisting of a guide explaining SPEM notation and the same eighteen SPM diagrams of different application domains. We collected all the data including the **understandability** time obtained from the responses of the tests and the metrics values which were already calculated for analysis of the results of the first experiment.

Once the data was collected, we controlled if the tests were complete. All tests were complete. We calculated the mean of the maintenance time and the median of the subjects' ratings about the understandability, analysability and modifiability of the SPM. So this is the data we want to analyse to test the hypotheses stated above. We applied the Kolmogorov-Smirnov test and as the data were non-normal we decided to use a non-parametric test like Spearman's correlation coefficient, with a level of significance a = 0.05, correlating each of the metrics separately with maintenance time and subjects' ratings (see table 6).

Table 6. Spearman's correlation between the metrics and understandability, analysability and modifiability

	NA	NWP	NPR	NDWPIn	NDWPOut	NDWP	NDA	NCA	RDWPIn	RDWPOut	RWPA	RRPA
Underst	**0,684**	**0,724**	0,174	**0,775**	**0,819**	**0,852**	**0,508**	-0,181	0,075	-0,075	0,105	-0,369
Analiz	**0,602**	**0,778**	-0,012	**0,802**	**0,854**	**0,878**	**0,503**	-0,275	0,108	-0,108	0,225	**-0,506**
Modif.	**0,698**	**0,750**	0,175	**0,847**	**0,874**	**0,917**	**0,558**	-0,269	0,099	-0,099	0,128	-0,415
Underst Time	0,182	-0,007	0,189	-0,121	-0,103	-0,132	0,176	-0,085	-0,276	0,276	-0,193	0,254

The Spearman cut-off for accepting H0 is 0.4684 (the sample size is the same as the first experiment) because the computed Spearman's correlation coefficients (table 6) are above the cut-off, the null hypothesis H0 is rejected with respect to the influence of some metrics in the sub-characteristics of maintainability. Analysing table 6 we can conclude that there is a significant correlation (rejecting the null hypothesis) between the structural complexity of the SPM and the metrics (NA, NWP, NDWPIn, NDWPOut, NDWP y NDA) because the correlation coefficient is greater than 0,4684. The metric RRPA is correlated with analysability and it seems to be less correlated other two maintainability sub-characteristics respect to the prior metrics, although it has a correlation value near to the cut-off. The metrics NPR, NCA, RDWPIn, RDWPOut y RWPA do not seem to be correlated with maintainability. These results confirm the results obtained in the first experiment.

According to the correlation between the metrics proposed and the understandability time (new hypotheses) the results show that it does not exist. With these results we do not think that time is a meaningful factor to be measured in subjective experiments because it is not a real indicator. We cannot demonstrate that the subjects completely understood the diagram in the times indicated

3.3. Comparison of Results

An overall analysis of the obtained results (see tables 5 and 6) leads us to conclude that the metrics NA, NWP, NDWPIn, NDWPOut, NDWP and NDA are to some extent correlated with the three maintainability sub-characteristics we considered. The metric RRPA seems to be less correlated with the three maintainability sub-characteristics with respect to the prior metrics, although it has a correlation value near to the cut-off with respect to understandability and modifiability and seems to be correlated with analysability. The metrics NPR, NCA, RDWPIn, RDWPOut and RWPA do not seem to be correlated with maintainability, although this preliminary result may be caused by the design of the experiment.

We believe it is too early to consider these results as definitive. As previously stated, further empirical validation is needed, including internal and external replication of these experiments, and also new experiments must be carried out in which the subjects demonstrate that they have correctly understood the models and that they can modify them. Besides it is necessary to apply new experiments with practitioners who work in software development organizations. As is remarked in [2], after performing a family of experiments you can build the cumulative knowledge to extract useful measurement conclusions to be applied in real measurement projects. Moreover, data related to "real projects" is also needed for gathering real evidence that these metrics can be used as early SPM maintainability indicators.

4. Conclusions and Further Work

In this work a set of representative metrics to measure the structural complexity of the software process models has been defined and empirically validated. The aim is to evaluate the influence of these metrics in the maintainability of software process models, which is an important factor that affect to their quality. These metrics are focused on the main elements included in a model of software processes, and may provide the quantitative base necessary to evaluate the changes in the software processes in companies with high maturity levels, which are applying continuous improvement actions [13]. In order to demonstrate the practical utility of the metrics proposed, one experiment has been performed and replicated. This experiment and its replica have allowed us to draw preliminary conclusions about the influence of the metrics proposed at model level in the maintainability of the software process models through three of its sub-characteristics (understandability, analysability and modifiability). As a result of these experiments performed we could conclude that the metrics NA, NWP, NDWPIn, NDWPOut, NDWP and NDA are good maintainability indicators, however we cannot say the same about the metrics NPR, NCA, RDWPIn, RDWPOut and RWPA.

Although the results obtained in these experiments are good, we cannot consider them to be definitive results. It is necessary to elaborate new, more objective experiments in which subjects by answering questions related with the models and modifying them could enable us to make a better evaluation of their maintainability. Besides it is necessary to further develop study cases with the metrics proposed.

With these considerations in mind we propose as future research lines:

- Development of an objective experiment in order to confirm the conclusions obtained regarding the influence of the metrics in the maintainability of the software process models.
- Development of case studies in which we evaluate the metrics proposed using software process models of a specific company.
- Consideration of other views related with the modelling of software processes, like for example roles and their responsibilities in work products, in order to define and validate new possible metrics.

5. Acknowledgements

This work has been partially funded by the TAMANSI project financed by "Consejería de Ciencia y Tecnología, Junta de Comunidades de Castilla-La Mancha" of Spain (project reference PBC-02-001) and by the MAS project partially supported by "Dirección General de Investigación of the Ministerio de Ciencia y Tecnología" (TIC 2003-02737-C02-02). We also want to acknowledge the students, technicians and professors of the "Escuela Superior de Informatica" of Ciudad Real who participated in the experiment.