Sponsors of / Sponsoren der IWSM/MetriKon 2004:

BÜREN & PARTNER SOFTWARE DESIGN

T-Systems

SIGNAL IDUNA Versicherungen und Finanzen

ITERGO

Exhibitors at / Aussteller der IWSM/MetriKon 2004:

CICERO CONSULTING

bach consulting

gedas your IT partner

ISBN 3-8322-3383-0

Software Measurement – Research and Application

Abran (Eds.) et al.

Magdeburger Schriften zum Empirischen Software Engineeri

Eds.: Prof. Dr. Alain Abran, Université du Québec
Manfred Bundschuh, AXA AG, Köln, Vorsitzender der DASMA e.V.
Günter Büren, Büren & Partner Software-Design, Nürnberg
Prof. Dr.-Ing. habil. Reiner R. Dumke, Universität Magdeburg

OTTO-VON-GUERICKE-UNIVERSITÄT MAGDEBURG

Fakultät für Informatik
Institut für Verteilte Systeme
Arbeitsgruppe Softwaretechnik

OTTO-VON-GUERICKE-UNIVERSITÄT MAGDEBURG

Software Measurement – Research and Application

Proceedings of the International Workshop on Software Metrics and DASMA Software Metrik Kongress

IWSM/MetriKon 2004

2.-5. November 2004, Königs Wusterhausen, Germany

DASMA Deutschsprachige Anwendergruppe für Software-Metrik und Aufwandsschätzung

GI-Fachgruppe 2.1.10 Software Messung und Bewertung

SML Otto-von-Guericke-Universität Magdeburg Software Measurement Laboratory (SMLab)

SHAKER VERLAG

Magdeburger Schriften zum Empirischen Software Engineering

Alain Abran, Manfred Bundschuh,
Günter Büren, Reiner R. Dumke (Eds.)

Software Measurement – Research and Application

Proceedings of
the International Workshop on Software Metrics
and
DASMA Software Metrik Kongress
IWSM/MetriKon 2004

2.-5. November 2004, Königs Wusterhausen, Germany

Shaker Verlag
Aachen 2004

Bibliographic information published by Die Deutsche Bibliothek
Die Deutsche Bibliothek lists this publication in the Deutsche Nationalbibliografie; detailed bibliographic data is available in the internet at http://dnb.ddb.de.

Printed in Germany.

ISBN 3-8322-3383-0
ISSN 1618-7946

Shaker Verlag GmbH • P.O. BOX 101818 • D-52018 Aachen
Phone: 0049/2407/9596-0 • Telefax: 0049/2407/9596-9
Internet: www.shaker.de • eMail: info@shaker.de

Keynote I

Session A1

Session B1

Session A2

Session B2

Session A3

Session B3

Session A4

Session B4

Session A5

Session B5

Session A6

Session B6

Session A7

Session B7

Session A8

Session B8

Keynote II

Preface / Vorwort

Today the pressure for more efficient software development processes delivering appropriate quality is constantly increasing. But who knows how efficient his current development process actually is and whether the quality of delivered products is really appropriate? Did we substantially improve with all the improvement effort spent? How can we answer all these questions if not by measuring processes and products?

Software process evaluation and improvement require quantified methods and technologies. Issues such as the applicability of measures and metrics to software, the efficiency of measurement programs in industry and the theoretical foundations of software engineering have been researched in order to evaluate and improve modern software development approaches.

This proceedings of the joined conferences, the 14[th] International Workshop on Software Measurement (IWSM 2004) and the DASMA MetriKon 2004, try to reflect a bit of all the concepts developed and the experiences made when measuring software. They are of particular interest to software engineering researchers, as well as to practitioners, in the areas of project management and quality improvement programs, for both software maintenance and software development.

43 interesting papers, presented in English or in German, have been chosen out of more then 60 submissions. They deliver an overview of the state of the art in software metrics and estimation. We are very proud that we succeeded in getting two higly experienced practitioners as invited speakers:

- Pekka Forselius, Software Technology Transfer Finland Oy, and
- Pam Morris, Total Metrics, Australia.

Their keynote contributions are also to be found as paper in this book.

Furthermore there are several tutorials on different subjects offered, also held by very experienced speakers. For more information see the MetriKon website at **<http://www.metrikon.de>**.

At the same time there is a meeting of the ISO Working Group 12 (ISO/IEC JTCI/SC7) at the venue, providing a lot of opportunities for discussions with members of this group of metrics experts.

Softwareentwickelnde Organisationen stehen heute zunehmend unter Druck, ihre Entwicklungsprozesse effizienter zu gestalten und Ergebnisse in einer angemessenen Qualität zu liefern. Aber wer weiß schon, wie effizient sein Prozess tatsächlich ist und ob die gelieferte Qualität angemessen ist? Haben die vielen Bemühungen zur Verbesserung des Entwicklungsprozesses eine echte

References

1. Cardoso, A.I., Crespo, R.G., Kokol, P., Two different views about software complexity, 11[th] European Software Control and Metrics conference – 3[rd] Software Certification Programme in Europe Conference (ESCOM-SCOPE), Munich, Germany, 2000.
2. Codefast, 10 Complexity Drivers for Software Builds, White Paper. Online: **http://www.codefast.com/z-software-builds/sc-software-builds-tenq.html**
3. COSMIC, Full Function Point Measurement Manual 2.2, Common Software Measurement International Consortium, 2003.
4. DeMarco, T., Controlling Software Projects, Yourdon Press, New York, 1982.
5. IFPUG, Function Point Counting Practices Manual 4.2, International Function Point Users Group, 2004.
6. Ivanov, P.B., Qualitative Complexity, Troitsk Institute for Innovation and Fusion Research, Moscow, Russia, 1997. Online: **http://www.geocities.com/unihl/texts/cx/cx.htm**
7. Meli, R., The Software Measurement Role in a Complex Contractual Context, 1[st] Software Measurement European Forum (SMEF), Rome, Italy, 2004.
8. Nonaka M., Kakurai A., Bukhary E., Azuma M., A complexity-weighted functional size metric for interactive software, 12[th] International Conference on Software Quality (ICSQ), 2002.
9. Tran-Cao D., Abran A., Lévesque G., Functional complexity measurement, 11[th] International Workshop on Software Measurement (IWSM), Montréal, Canada, 2001.

Validating Metrics for UML Statechart Diagrams through a Family of Experiments

José A. Cruz-Lemus, Marcela Genero, Mario Piattini

ALARCOS Research Group
Computer Science Department, University of Castilla – La Mancha
Paseo de la Universidad 4, 13071 Ciudad Real (Spain)
{JoseAntonio.Cruz, Marcela.Genero, Mario.Piattini}@uclm.es

Abstract:

In this paper we present a family of experiments that we carried out for validating a set of metrics that were defined for measuring the size and structural complexity of UML statechart diagrams, as early understandability indicators. The family consists of a controlled experiment and two replications performed with different Computer Science students and teachers.

The obtained results reveal that the metrics Number of Activities (NA), Number of Simple States (NSS), Number of Guards (NG) and Number of Transitions (NT) are highly correlated with the understandability time of the UML statechart diagrams.

Nevertheless, further validation is needed for obtaining stronger results.

Keywords

UML Statechart Diagrams, Metrics, Understandability, Empirical Validation, Experiments

1 Introduction

In recent years, software developers have paid special attention to guarantee the quality characteristics of object oriented (OO) systems since the initial stages of their life cycle, specially focusing on the quality of conceptual models. Some paradigms, such as the Model-Driven Development [1] and the Model-Driven Architecture [16] consider that conceptual models are the backbone of the OO systems development, emphasizing the importance of building *good* conceptual models.

In the literature there exists a lot of works covering the measurement of quality aspects of UML models, but most of them focus on UML class diagrams [4][5][6][7][10][13][14] or UML use case diagrams [10][11][14]. But the existence of metrics for UML behavioural models is scarce [6][18][19][22] and has not gone beyond the definition step. For that reason in [8] a set of metrics for UML statechart diagrams was presented (see Table 1).

	Metric	Description
Size Metrics	**NEntryA** (Number of entry actions)	The total number of entry actions, i.e., the actions performed each time a state is entered
	NExitA (Number of exit actions)	The total number of exit actions, i.e., the actions performed each time a state is left
	NA (Number of activities)	The total number of activities (do/activity) in the statechart diagram
	NSS (Number of simple states)	The total number of states considering also the simple states within the composites states
	NCS (Number of composite states)	The total number of composite states
	NG (Number of guards)	The total numbers of guard conditions.
	NE (Number of events)	The total number of events.
Structural Complexity Metrics	**NT** (Number of transitions)	The total number of transitions, considering common transitions (the source and the target states are different), the initial and final transitions, self-transitions (the source and the target states are the same) and internal transitions (transitions inside a state that responds to an event but without leaving the state).
	CC (Cyclomatic Complexity)	It is defined as |NSS-NT+2|.

Table 1: Metrics for UML statechart diagrams

After being theoretically validated through Briand et al.'s property-based framework [3], some of them were characterized as size metrics (NEntryA, NExitA, NA, NSS, NCS, NG, NE) whilst the others were characterized as structural complexity metrics (NT, CC). In the theoretical validation process of these metrics, we also used a Measurement Theory-based framework, the DISTANCE framework [17], for guaranteeing that the metrics measure the attributes they try to measure, assuring thus the construct validity of the empirical studies where these metrics were used.

For empirically validating these metrics as understandability indicators, we performed a family of experiments. As Miller [15], Basili et al. [2] and Shull et al. [20] among others suggested, simple studies rarely provide definite answers. We are aware that only after performing a family of experiments, an adequate body of knowledge can be built in order to extract useful measurement conclusions regarding the use of OO design metrics to be applied to real measurement projects [2][20].

The present paper is structured as follows: in section 2, we will describe the family of experiments we have carried out. After that, in section 3 we will show the analysis and interpretation of the empirical data. Finally, in section 4 we will go over the different conclusions reached after our research process and the future work that has been planned to do.

2 A Family of Experiments

Our family of experiments consists of three different data sources corresponding to a controlled experiment and two replications of it. The original experiment (E1) took place in March of 2002, the first replication (R1) was performed in May of 2002 and finally, the second replication (R2) in May 2004. The preliminary results of E1 and R1 have been presented in [9].

The main goal of all of them is to ascertain if there is a relationship between the metrics presented in Table 1 and the understandability of UML statechart diagrams. If such relation existed, we would have found a group of indicators for UML statechart diagrams understandability, that could be used for OO systems developers when making the appropriate decisions in the early stages of a software product lifecycle.

For performing the experiments, we have taken into account some suggestions provided by Wohlin et al. [21] and Juristo and Moreno [12]. The main characteristics of these experiments are listed below:

- In the first experiment, the subjects were eight teachers of the Software Engineering area and ten students enrolled in the last (fifth) year of Computer Science at the Department of Computer Science at the University of Castilla – La Mancha. In the first replica, they were twenty-four students in their third-year of Computer Science and in the second replica, forty-nine third-year students.
- The experience of subjects in UML statechart diagrams in E1 was normal for the students, as they had already taken two complete Software Engineering courses, and it was high for the teachers, that belonged to the Software Engineering area. In the replications, R1 and R2, the experience of the students was lower, as they had only taken one Software Engineering course, and it had not been completed at that moment.
- With respect to the variable selection, in all cases the independent and dependent variables were the same. The independent variables were size and

structural complexity of UML statechart diagrams, measured through the metrics presented in Table 1. The dependent variable was the understandability of UML statechart diagrams, measured through the time that took to the subjects to do the experimental tasks, called 'understandability time'. We also measured the correctness (*correct answers vs. answered questions*) and completeness of the answers (*correct answers vs. asked questions*).

- The hypothesis we formulated were:
  - $H_{0,1}$: There is no significant correlation between the UML statechart diagrams structural complexity and size metrics and their understandability time.
  - $H_{1,1}$: $\neg H_{0,1}$
  - $H_{0,2}$: There is no significant correlation between the UML statechart diagrams structural complexity and size metrics and the correctness of the answers that show if they have been understood.
  - $H_{1,2}$: $\neg H_{0,2}$
  - $H_{0,3}$: There is no significant correlation between the UML statechart diagrams structural complexity and size metrics and the completeness of the answers that show if they have been understood.
  - $H_{1,3}$: $\neg H_{0,3}$
- The objects were twenty UML statechart diagrams. Each of them had different values for the metrics and had also enclosed a four questions test in order to check whether the subjects had understood the diagram correctly or not. The subjects wrote down the time (as hour, minutes and seconds) in which they started working with the diagram and finished answering the last question. The difference, in seconds, of these two values is what we called 'understandability time'. The tests were distributed in different order to alleviate the possible damaging effects that learning could include.
- E1 was run without supervision. The subjects were given all the described materials and were told to bring it back answered in one week. However, the replications were run in a two-hours session and there was an instructor who supervised the experiment and could solve any asked doubt.
- The external validity of an experiment is defined as the degree to which the results can be generalised to the population under study and other research settings. The greater this validity is, the more the results of an empirical study can be generalised to actual software engineering practice. In this family of experiments, we have identified two main threats to the external validity which could limit the ability for applying any generalisation:

  - Materials and tasks used. In the experiment we tried to use statechart diagrams and tasks which were representative of real cases, but more empirical studies taking 'real cases' from software companies must be done in the future.
  - Subjects. In order to solve the difficulty of obtaining professional subjects, we used teachers and 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 generalise these results. However, in this case, the tasks to be performed did not require high levels of industrial experience, so experiments with students could be appropriate, as pointed in [2].
- The conclusion validity is the degree to which conclusions can be drawn about the existence of a statistical relationship between treatments and outcomes. In all our experiments, we decided to take the whole population of the available classes as our target samples. For what concerns the quality of data collecting, we used 'pencil and paper', hence data collection could be considered critical. The supervisors did not perform any checks of the times given by the students and teachers. Nevertheless, the subjects assumed the responsibility for writing down the correct times.
- The construct validity is the degree to which the independent and the dependent variables are accurately measured by the measurement instruments used in the experiment. In our case, the construct validity of the metrics used for the measurement of the independent variable is guaranteed by the theoretical validation that was carried out with them [8].
- The internal validity is the degree of confidence in a cause-effect relationship between factors of interest and the observed results. In order to take this validity into account, the following issues have been observed: differences among subjects, knowledge of the universe of discourse among statechart diagrams, precision in the time values, learning and fatigue effects, persistence effects, subjects motivation, plagiarism influence between students...

3 Data Analysis and Interpretation

In this section, we will explain the analysis of the different data obtained in the different experiments.

Table 2 shows the descriptive statistics for all the different measures of the dependent variables.

E1 (March 2002)						
Dependent vars.	Mean	SE	Min.	Max.	Skew.	Kurtosis
Und.. Time (n=18)	132.6444	39.72498	84.61	226.72	0.525	-0.008
Correctness	0.9830	0.1942	0.93	1.00	-1.348	1.808
Completeness	0.9812	0.1976	0.93	1.00	-1.116	0.988
R1 (May 2002)						
Dependent vars.	Mean	SE	Min.	Max.	Skew.	Kurtosis
Und.. Time (n=24)	147.4460	38.23261	94.42	255.75	1.160	1.975
Correctness	0.9785	0.0653	0.71	1.00	-4.124	17.620
Completeness	0.9769	0.0611	0.73	1.00	-4.051	17.165
R2 (May 2004)						
Dependent vars.	Mean	SE	Min.	Max.	Skew.	Kurtosis
Und. Time (n=49)	129.1136	33.96969	75.08	203.33	0.410	-0.077
Correctness	0.9291	0.0725	0.70	0.99	-1.988	4.675
Completeness	0.9266	0.0738	0.70	0.99	-1.934	4.114

Table 2: Descriptive statistics of the dependent variables

When analyzing these data, we can notice that the value of the different arithmetic means for all the experiments are similar in the different measures of the dependent variables, which could mean that the difference of experience in working with UML statechart diagrams has not finally affected the results, as it was cancelled by the training session with the subjects. Nevertheless and as it was predictable, the best values for correctness and completeness correspond to the first experiment, in which teachers and last-year students were involved.

For testing the hypotheses, a correlation analysis was performed. We used the Spearman's correlation coefficient with a level of significance a = 0.05. In all cases, each of the metrics was correlated separately to the mean of the subjects' understandability time, and the correctness and completeness of the answers. We obtained the results shown in Table 3, where the first group of data corresponds to E1, as the second and third group correspond to R1 and R2, respectively.

For a sample size of 20 and a = 0.05, the Spearman cut-off for accepting the null-hypothesis is 0.44. We can appreciate how the computed Spearman's correlation coefficients for metrics NA, NSS, NG and NT, are always above this cut-off in the case of the understandability time. As the p-value < 0.05, the null hypothesis $H_{0,1}$ can be rejected.

	NEntryA	NExitA	NA	NSS	NCS	NE	NG	NT	CC
UTime	0.198	-0.213	**0.463**	**0.492**	0.342	0.400	**0.535**	**0.521**	0.406
Correct.	-0.077	0.038	-0.170	-0.376	-0.596	-0.109	-0.434	-0.207	-0.017
Complt.	0.060	0.065	-0.148	-0.266	-0.525	-0.051	-0.438	-0.125	0.029
UTime	-0.021	-0.384	**0.529**	**0.586**	0.428	0.353	**0.572**	**0.541**	0.404
Correct.	-0.239	-0.056	-0.169	0.009	-0.201	-0.132	-0.118	-0.154	-0.238
Complt.	-0.164	0.050	-0.378	0.092	-0.127	0.006	-0.203	-0.016	-0.024
UTime	-0.091	-0.391	**0.603**	**0.528**	0.344	0.358	**0.519**	**0.547**	0.436
Correct.	-0.220	-0.035	-0.116	0.005	-0.385	0.048	-0.506	0.020	0.058
Complt.	-0.206	0.019	-0.157	-0.041	-0.369	0.027	-0.519	-0.013	0.034

Table 3: Spearman's correlation coefficients

As a consequence of these findings, we can assert that the metrics NA (*Number of Activities*), NSS (*Number of Simple States*), NG (*Number of Guards*) and NT (*Number of Transitions*) have a great relationship with the time needed to understand an UML statechart diagram.

4 Conclusions

With the hypothesis that the size and the structural complexity of UML statechart diagrams could influence their understandability (and therefore in their maintainability), we defined a set of metrics for the structural complexity and size of UML statechart diagrams in a methodological way [8].

In order to gather empirical evidence that they could be good indicators of UML statechart diagrams understandability, we presented in this paper a family of experiments, composed by a controlled experiment and two replications of it. They were performed by different groups of Computer Science teachers and students.

We have finally concluded that the metrics that measure the number of activities, the number of simple states, the number of guards and the number of transitions in an UML statechart diagram are highly correlated with the time that took the subjects to understand it, and therefore could have great influence on the maintenance effort of such diagrams.

Even though the encouraging findings, we considered them as preliminaries. Further validation is needed, with practitioners and also taking data from real projects. Moreover, as future work, we have planned to build a model to predict the understandability time based on the metrics values, using multivariate regression analysis.

Acknowledgements

This research is part of the MESSENGER project (PCC-03-003-1) financed by 'Consejería de Ciencia y Tecnología de la Junta de Comunidades de Castilla-La Mancha (Spain)' and the CALIPO project supported by 'Dirección General de Investigación del Ministerio de Ciencia y Tecnología (Spain)' (TIC2003-07804-C05-03).

The authors would like to thank sincerely professor Ann Maes from University of Ghent for her valuable aid in statistics and professors Félix García and Crescencio Bravo from University of Castilla-La Mancha for allowing us to perform the experiment with their students.

References

1. Atkinson, C. and Kühne, T. Model-Driven Development: A Metamodeling Foundation. *IEEE Software*, 20(5), pp. 36-41, 2003
2. Basili, V., Shull, F. and Lanubile F. Building Knowledge through Families of Experiments. *IEEE Transactions on Software Engineering*, 25(4), pp. 435-437, 1999
3. Briand, L., Morasca, S. and Basili, V. Property-based software engineering measurement. *IEEE Transactions on Software Engineering*, 22 (1), pp. 68-85, 1996.
4. Brito e Abreu, F. and Carapuça, R. Object-Oriented Software Engineering: Measuring and controlling the development process. *4th International Conference on Software Quality*, Mc Lean, Virginia, USA, 1994.
5. Chidamber, S. and Kemerer, C. A Metrics Suite for Object Oriented Design. *IEEE Transactions on Software Engineering*, 20(6), pp. 476-493, 1994.
6. Derr, K. *Applying OMT*. SIGS Books, Prentice Hall. New York, USA, 1995.
7. Genero, M. *Defining and Validating Metrics for Conceptual Models*, Ph.D. Thesis, Department of Computer Science, University of Castilla-La Mancha, Spain, 2002.
8. Genero, M., Miranda, D. and Piattini, M. Defining and Validating Metrics for UML Statechart Diagrams. Proceedings of the *6th International ECOOP Workshop on Quantitative Approaches in Object-Oriented Software Engineering (QAOOSE 2002)*, pp. 120-136, Málaga, Spain, 2002.
9. Genero, M., Miranda, D. and Piattini, M. Defining Metrics for UML Statechart Diagrams in a Methodological Way. Proceedings of the *International Workshop on Conceptual Modeling Quality (IWCMQ 2003)*, Lecture Notes on Computer Science 2814, Springer-Verlag, pp. 118-128, Evanston, Illinois, USA, 2003.
10. Genero M., Piattini M. and Calero M. (eds.). *Metrics for Software Conceptual Models*. Imperial College Press, UK, 2004 (to appear).
11. Henderson-Sellers, B., Zowghi, D., Klemola, T. and Parasuram, S. Sizing Use Cases: How to Create a Standard Metrical Approach. Proceedings of the *8th International Conference on Object-Oriented Information Systems (OOIS 2002)*, Lecture Notes on Computer Science 2425, Springer-Verlag, pp. 409-421, Montpellier, France, 2002.
12. Juristo, N. and Moreno, A. *Basics of Software Engineering Experimentation*. Kluwer Academic Publishers, 2001.
13. Lorenz, M. and Kidd, J. *Object-Oriented Software Metrics: A Practical Guide*. Prentice Hall, Englewood Cliffs, New Jersey, USA, 1994.
14. Marchesi, M. OOA Metrics for the Unified Modeling Language. Proceedings of the *2nd Euromicro Conference on Software Maintenance and Reengineering (CSMR 98)*, pp. 67-73, Florence, Italy, 1998.
15. Miller, J. Applying Meta-Analytical Procedures to Software Engineering Experiments. *Journal of Systems and Software*, 54, pp. 29-39, 2000.
16. Object Management Group. *MDA-The OMG Model Driven Architecture*. Online available at: http://www.omg.org./mda/, August 1st, 2002
17. Poels, G. and Dedene, G. Distance-Based software measurement: necessary and sufficient properties for software measures. *Information and Software Technology*, 42(1), pp. 35-46, 2000.
18. Poels, G. and Dedene, G.: Measures for Assessing Dynamic Complexity Aspects of Object-Oriented Conceptual Schemes. Proceedings of the *19th International Conference on Conceptual Modelling (ER 2000)*, Lecture Notes on Computer Science 1920, Springer-Verlag, pp. 499-512, Salt Lake City, Utah, USA, 2000.
19. Selic, B., Gullekson, G. and Ward, P. *Real-Time Object Oriented Modelling*. John Wiley & Sons, Inc., 1994.
20. Shull, F., Carver, J., Travassos, G., Maldonado, J., Conradi, R. and Basili, V. *Replicated Studies: Building a Body of Knowledge about Software Reading Techniques*. Lecture Notes on Empirical Software Engineering, Juristo N. and Moreno A. (eds.). World Scientific, Singapore, pp. 39–84, 2003.
21. Wohlin, C., Runeson, P., Höst, M., Ohlson, M., Regnell, B. and Wesslén, A. *Experimentation in Software Engineering: An Introduction*. Kluwer Academic Publishers, 2000
22. Yacoub, S., Ammar, H. and Robinson, T. Dynamic Metrics for Object Oriented Designs. *Sixth IEEE International Symposium on Software Metrics*, Boca Ratón, Florida, USA, 1998.

A V&V Measurement Management Tool for Safety-Critical Software

Edgardo Palza, Alain Abran*, Christopher Fuhrman*, Eduardo Miranda***

**École de Technologie Supérieure – ETS*
1100 Notre-Dame Ouest, H3C 1K3 Montréal, Québec, Canada.

*** Ericsson Research Canada*

8500 Blvd. Decarie H4P 2N2, Montréal Québec, Canada

edgardo.palza-vargas.1@ens.etsmtl.ca,

aabran@ele.etsmtl.ca,

christopher.fuhrman@etsmtl.ca

eduardo.miranda@ericsson.com

Abstract:

This paper presents a V&V Measurement Management Tool (V&V MMT) to support the Management of V&V activities in the context of safety-critical software. We illustrate how V&V MMT can facilitate the quantification of the V&V processes, activities and tasks in projects recommended in the IEEE Standard for Software Verification and Validation (IEEE Std. 1012-1998) for facilitating the establishment of V&V measurement indicators: (1) Software Verification and Validation Plan (SVVP), (2) Baseline change assessment, (3) Management Review of V&V, (4) Management and Technical Review Support, (5) Interface with Organizational and Supporting Process.

Keywords

Verification and Validation, Safety Critical Software, Measurement Repository, Measurement Meta-model, Measurement Management.

1 Introduction

Today's software is becoming increasingly more complex: heterogeneous composition on a diversity of platforms, distributed execution, complexity in calculation algorithms, multiplicity of contractors with diverse development methodologies, etc. The result of such complexity is increased risk and higher costs in software projects.

The type of software that, directly or indirectly, ensures the safety of human life or significant financial investments is referred to as safety-critical software. This type of software is required to meet very high levels of safety and