An Annotated Bibliography on Software Process Improvement

Alfonso Fuggetta †‡ Gian Pietro Picco†

‡Politecnico di Milano - Dipartimento di Elettronica e Informazione
Piazza L. da Vinci 32, 20124 Milano, Italy.
Tel. +39-2-23993540, Fax. +39-2-23993411
e-mail: fuggetta@elet.polimi.it

†CEFRIEL
Via Emanueli 15, 20126 Milano, Italy.
Tel. +39-2-66100083, Fax. +39-2-66100448
e-mail: picco@mailer.cefriel.it

June 2, 1994

Software process improvement is a critical problem that is currently addressed both at the academic and industrial level. The field is relatively young, and there is no "ultimate approach" to the problem. The related scientific literature is rapidly growing, and is more and more difficult to make our way in the plethora of different perspectives, approaches, and case studies. This paper provides a guide to the process improvement literature: we think that both newbies and experts of this field can take advantage from it.

1 Motivations and perspectives on software process improvement

1.1 Universally cited references

[1] Frederick P. Brooks. *The mythical man-month–Essays on Software Engineering.* Addison-Wesley, Reading, MA, 1972.

A famous collection of fifteen essays on software engineering, focused on the problems arising in managing and developing a very large software project. Brooks cites frequently his own experience of manager of the IBM Operating System/360 operating system. A classical and yet actual book on the fundamentals issues in software engineering.

[2] Frederick P. Brooks. No silver bullet: essence and accidents of Software Engineering. *IEEE Computer*, 20(4):10–19, April 1987.

A classical paper analyzing why building software is so difficult, and why the ultimate solution to this problem does not exists.

[3] Barry W. Boehm. *Software Engineering Economics.* Prentice-Hall, Englewood Cliffs, NJ, 1981.

A classical book on software process management, with a deep focus on software project cost estimation: includes a detailed explanation of COCOMO.

[4] Philip B. Crosby. *Quality is Free.* McGraw-Hill, New York, NY, 1979.

This book, and the following by Deming and Juran, are the three best-seller books about software quality in particular, and quality in general. Highly recommended for understanding the fundamentals of quality management. Notably, in the Crosby's book a maturity grid is explained, which has been extended and tailored in the SEI maturity framework.

[5] W. Edwards Deming. *Out of the crisis.* MIT Center for Advanced Engineering Study, Cambridge, 1986.

[6] J. M. Juran. *Juran on Planning for Quality.* MacMillan, New York, NY, 1988.

[7] Imai Masaaki. *Kaizen: The Key to Japan's Competitive Success.* McGraw-Hill, New York, NY, 1986.

A classical book on the Japanese way towards quality in industrial products and processes. Many of the ideas exposed here are present in other approaches: the word "Kai-Zen" means "continuous improvement".

[8] Watts S. Humphrey. *Managing the Software Process.* SEI Series in Software Engineering. Addison-Wesley, Reading, MA, 1989.

A fundamental book on the topic. Centered around the maturity framework, the skeleton of the Capability Maturity Model, it anticipates many issues of the CMM. Nonetheless, the book ought not to be read as the "CMM view on software process management", instead it can be easily read as a general reference on the topic.

[9] David Lorge Parnas. Software Aspects of Strategic Defense Systems. *Communications of the ACM*, 28(12):1326–1335, December 1985.

A famous paper explaining why the "star wars" effort cannot achieve its stated goals, and why Parnas decided to resign from the SDIO Panel on Computing Support of Battle Management. A 360-degree wide discussion on the applicability and reliability of computer science methods and techniques. An old-dated contribution, but for many aspects also a to-do list not yet exhausted.

1.2 Other interesting references

[1] Bill Curtis. Techies as non-technological factors in software engineering? In *Proceedings of the 13th International Conference on Software Engineering–ICSE 13*, pages 147–148. IEEE Computer Society, 1991.

In this paper men's abilities are highlighted as the training factor in achieving better software processes, while technological improvement are considered as a minor one.

[2] M. Cusumano. The Software Factory: A Historical Interpretation. *IEEE Software*, 6(2):23–30, March 1989.

Is there sufficient base to apply factory concepts to software? Is this application even appropriate? What is a software factory? What degree of integration or standardization among tools, methods, controls, and skills is distinctive of a software factory? What can we learn about software engineering from Japanese software factories? These are the question Cusumano tries to answer in this paper.

[3] Alan M. Davis. Why Industry Often Says 'No Thanks' to Research. *IEEE Software*, 9(6):97–99, November 1992.

The author describes his viewpoint shift about the relation existing between research and industry. Having been formerly a researcher, and then a manager, in the same software organization, he realized how he developed outstanding research tools that were completely out of the needs for his organization. He proposes a model of the software organization needs expressed in term of the Maslow's hierarchy of needs.

[4] Norman Fenton. Software Measurement: A Necessary Scientific Basis. *IEEE Transactions on Software Engineering*, 20(3):199–206, March 1994.

A sharp paper claiming for a discipline in software measurement practice. According to the author, the basic measurement principles, which are well established in traditional engineering discipline, are systematically disregarded in common software practices. As an example, it is shown that the search for a magic number representing software complexity, which is a major issue in software metrics research, is doomed to failure from a theoretical point of view.

[5] Robert B. Grady. Practical Results from Measuring Software Quality. *Communications of the ACM*, 36(11):62–68, November 1993.

How metrics can help in driving process improvement, and in foreseeing the problems during software development.

[6] Robert Lai. The Move to Mature Processes. *IEEE Software*, 10(4):14–17, July 1993.

A short talk on the general problems involved in software process improvement.

[7] Bill Curtis and Mark C. Paulk. Creating a software process improvement program. *Information and software technology*, 35(6/7):381–386, June 1993.

After introducing briefly the SEI CMM basics, some advices on building process improvement programs are provided, based on the extensive experience of the authors. The focus is mainly on two key issues: management commitment on the initiative, which must be reflected in the organizational structure of SEPG, the group dedicated to software process improvement, and process definition, which is needed in order to know where improvement must be carried out.

[8] Howard A. Rubin. Software Process Maturity: Measuring its Impact on Productivity and Quality. In *15th International Conference on Software Engineering*, May 1993.

In order to properly guide software process improvement, measurement must be directed not only towards technical indicators, probing the areas where weaknesses are known, but also towards organization-wide indicators, reporting on the overall productivity and quality variations. This paper provides a set of data on U.S. and U.K. organizations, together with a "list" of suggested metrics for measuring software organizations.

2 Standards

[1] ESA Software Engineering Standards Issue 2, ESA PSS-05-0. Issued by the European Space Agency, Paris, France, February 1991.

Describes the software engineering standards to be applied for all deliverable software implemented for the European Space Agency (ESA). A waterfall-like software life-cycle is defined, in which are defined what practices are mandatory and which are recommended. A great emphasis is posed on the documentation to be provided at the completion of every single step.

[2] ISO 9000-3, Quality Management and Quality Assurance standards - Part 3: Guidelines for the Application of ISO 9001 to the Development, Supply and Maintenance of Software. Issued by the International Standard Organization, 1991.

Specifies the key organizational and technical activities and documentation a software organization has to provide in order to be ISO 9000-compliant. The emphasis is on "what" has to be done instead on "how".

[3] ISO/IEC 9126, Information Technology - Software Product Evaluation - Quality Characteristics and Guidelines for their Use. Issued by the International Organization for Standardization and the International Electrotechnical Commission, 1991.

This standard defines six quality attributes (functionality, reliability, usability, efficiency, maintanability, portability) which are intended to describe a software product with the minimum overlap. These attributes are further decomposed in lower-level attributes which are only informative and not officially belonging to the standard. Guidelines for the usage of the six standard

attributes are provided, including a definition of the different views of software quality (user's, developer's, and manager's), and a process model for the evaluation of a software product based on these attributes.

[4] DOD-STD-2167A, Defense System Software Development. Published by U.S. Defense Dept., Washington, D.C., February 29 1988.

One of the most important software military standards issued by the U.S. Department of Defense.

[5] AQAP-13, NATO Software Control System Requirements. Allied Quality Assurance Publication, August 1987.

Specifies the requirements for the software development environment of a NATO contractor, in order to guarantee high-quality software production.

[6] IEEE Software Engineering Standards, Fourth Edition, 1991.

These standards reflect the experiences and the work of a large community of practitioners and researchers over the last decades.

[7] A. Dorling. SPICE: Software Process Improvement and Capability dEtermination. *Information and software technology*, 35(6/7):404, June 1993.

SPICE is the name of a project aiming at becoming an ISO standard for software process improvement and capability determination, incorporating all the best features of the existing approaches, like SEI's CMM or BOOTSTRAP. This short paper gives a general and yet concise description of the project goals, together with an overview of the expected final products.

3 Capability Maturity Model

3.1 Basic references

[1] E. J. Joyce. SEI. The Software Battleground. *Datamation*, 32(18):109–116, September 1986.

SEI was created as a Federal Funded Research and Development Center, with the sponsorship of the U.S. Department of Defense, and in times when Reagan was sponsoring the Strategic Defense Ininitiative. Many critics saw a relation between the two events. This paper is an overview of the issues above, together with a description of how SEI is organized, its missions, and the visions that guide its activities it.

[2] Watts S. Humphrey. Characterizing the Software Process: A Maturity Framework. *IEEE Software*, 5(2):73–79, March 1988. Also available as technical report CMU/SEI-87-TR-11, ADA182895.

The first description of the maturity framework, which is the basis of CMM.

[3] Watts S. Humphrey and W. L. Sweet. A Method for Assessing the Software Engineering Capability of Contractors. Technical Report CMU/SEI-87-TR-23, Software Engineering Institute, Carnegie Mellon University, September 1987.

The official release of the preliminary maturity questionnaire –at the time of writing, the only publicly available questionnaire.

[4] Timothy G. Olson, Watts S. Humphrey, and David H. Kitson. Conducting SEI-Assisted Software Process Assessments. Technical Report CMU/SEI-89-TR-7, DTIC Number ADA219065, Software Engineering Institute, Carnegie Mellon University, February 1989.

The "user manual" for SEI assessment teams. It contains a concise description of the assessment methodology.

[5] Marc C. Paulk, Watts S. Humphrey, and George J. Pandelios. Software Process Assessments: Issues and Lessons Learned. In *Proceedings of ISQE92*, pages 4B/41–4B58. Juran Institute, March 1992.

Reports "lessons learned" in performing assessments of over 60 US software organizations. The focus is on organizational issues emerged from interviews with managers, which are often hostile, or misunderstand the goal of this activity. In addition, a comprehensive overview of both CMM and assessment method is given.

[6] Mark C. Paulk. U.S. Quality Advances: the SEI's Capability Maturity Model. In *3rd European Conference on Quality*, 1992.

A history of the SEI's CMM, from the origins until to the version 1.0, published in 1991, highlighting changes and different alternatives.

[7] Mark C. Paulk, Bill Curtis, Mary Beth Chrissis, and Charles Weber. Capability Maturity Model, Version 1.1. *IEEE Software*, 10(4):18–27, July 1993.

A condensed version of the SEI technical report describing CMM version 1.1. Very useful, since it contains a quick look to changes from version 1.0 to 1.1, for those who already know CMM, and a comprehensive overview of the model, for those who do not know it.

[8] Mark C. Paulk, Bill Curtis, Mary Beth Chrissis, et al. Capability Maturity Model for Software, Version 1.1. Technical Report CMU/SEI-93-TR-24, DTIC Number ADA263403, Software Engineering Institute, February 1993.

The official description of CMM version 1.1.

[9] Charles V. Weber, Mark C. Paulk, Cynthia J. Wise, and James V. Withey. Key Practices of the Capability Maturity Model, Version 1.1. Technical Report CMU/SEI-93-TR-25, Software Engineering Institute, 1993.

This report may be regarded as the "user manual" for process improvement. After the SEI assessment has identified the weak areas of the software organization, this report helps in pursuing the "key practices" that allow improvement in the identified "key areas".

[10] Priscilla Fowler and Stan Rifkin. Software Engineering Process Group Guide. Technical Report CMU/SEI-90-TR-24, ADA235784, Software Engineering Institute, 1990.

A detailed manual for establishing a Software Engineering Process Group which, according to the SEI suggested improvement practices, is in charge of actually supporting process improvement, establishing and enforcing procedures and methods, creating and maintaining process definitions, keeping in contact with management, and so on.

3.2 Criticisms and enhancements

[1] Terry B. Bollinger and Clement McGowan. A Critical Look at Software Capability Evaluations. *IEEE Software*, 8(4):25–41, July 1991.

Software Capability Evaluations are used by US government agencies to select contractors. They are quite different from SEI assessment, even if they share the underlying CMM model and the assessment questionnaire. This article reviews and criticizes SCEs, highlighting the pro and the cons of this approach to capability evaluation and, indirectly, on the whole SEI CMM approach.

[2] Watts S. Humphrey and Bill Curtis. Comments on 'A Critical Look'. *IEEE Software*, 8(4):42–46, July 1991.

Humphrey's and Curtis' reply to 'A Critical Look' of C. Mc Gowan and T. Bollinger.

[3] Judith G. Brodman and Donna L. Johnson. What Small Business and Small Organizations Say About the CMM. In *16th International Conference on Software Engineering*, May 1994.

Reports a study aimed at identifying the problems that small organizations and small businesses have to face with in order to successfully implement an improvement plan based on SEI CMM.

[4] Marilyn W. Bush. Process Assessment in NASA. In *Proceedings of the 13th International Conference on Software Engineering*, pages 299–304. IEEE Computer Society, 1991.

The history of a NASA project aiming at extending the SEI assessment questionnaire with issues about Software Quality Assurance and contract management.

[5] David Card. Understanding Process Improvement. *IEEE Software*, 8(4):102–103, July 1991.

Distinguishes between "analytic" and "benchmarking" approaches to software process improvement. It then discusses the problems of the SEI approach, which is classified as a benchmarking one. The evidenced problems are both intrinsic to the SEI model and determined by an incorrect or superficial application by the user.

[6] Bill Curtis. Maturity from the User's Point of View. *IEEE Software*, 10(4):89–90, July 1993.

Analyzes the differences and the issues about process quality (up to the software organization) vs. perceived quality (up to the software user). The latter poses intereresting questions on the way user interfaces are currently built, forcing the software organization to modify its process maturity to cope with user needs,

[7] Anthony Finkelstein. A Software Process Immaturity Model. *Software Engineering Notes*, 17(4):22–23, October 1992.

Depicts a wasted software industry in a joking way, by proposing the extension of the five levels of SEI's CMM with three more levels, level 0 (foolish), level -1 (stupid), and level -2 (lunatic).

[8] Shari Lawrence Pfleeger and Clement McGowan. Software Metrics in the Process Maturity Framework. *The Journal of Systems and Software*, 12:255–261, December 1990.

"We can evaluate the purpose and utility of each metric only in the light of the needs and desires of the development organization." In this context, the authors used the SEI maturity framework as a tool helping in the definition of the metrics best suited for the capability of a software organization. For each maturity level a set of suggested metrics is reported and some guidelines for its use are provided. Notably, while SEI recommends metrics collection to start at level 3, the authors suggest that it has to start from level 1.

3.3 SEI-based case studies

[1] Watts S. Humphrey, David H. Kitson, and T. Kasse. The State of Software Engineering Practice: A Preliminary Report. In *Proceedings of the 11th International Conference on Software Engineering*, pages 277–288. Association for Computing Machinery, February 1989. Also available as technical report CMU/SEI-89-TR-1, ADA206573.

Reports the preliminary results of the application of the early version of the maturity questionnaire and the first assessment methodology to a set of U.S. organizations.

[2] Watts S. Humphrey, David H. Kitson, and Julia Gale. A Comparison of U.S. and Japanese Software Process Maturity. In *Proceedings of the 13th International Conference on Software Engineering*, pages 38–49, Austin, TX, May 1991.

In the period from 1987 to 1989 the SEI has assessed a number of Japanese software organizations. This paper compares the results of this assessments with analogous data for US software organizations.

[3] David H. Kitson and Stephen M. Masters. An Analysis of SEI Software Process Assessment Results: 1987-1991. In *15th International Conference on Software Engineering*, May 1993.

Analyzes and presents the results of the assessments of 59 government and industry software organizations. Despite of the high number of graphs and numbers, the paper is highly informal, and some point in the development of the study appears obscure.

[4] David Rugg. Using a Capability Evaluation to Select a Contractor. *IEEE Software*, 10(4):36–45, July 1993.

Reports an experience in evaluating the software capability of the organizations which are candidate for a contract involving software. The method employed

is tailored on SEI's Software Capability Evaluations, based on Capability Maturity Model. The followed method is described, together with lessons learned.

[5] Watts S. Humphrey, Terry R. Snyder, and Ronald R. Willis. Software Process Improvement at Hughes Aircraft. *IEEE Software*, 8(4):11–23, July 1991.

Reports how Hughes' Software Engineering Division has jumped in two years from CMM level 2 to level 3. It reports the assessment procedures, improvement suggestions, actions taken, together with cost estimates and global lessons learned.

[6] Harvey Wohlwend and Susan Rosenbaum. Software Improvement in an International Company. In *Proceedings of the 15th International Conference on Software Engineering*. IEEE Computer Society Press, May 1993.

Reports the Schlumberger experience in improving its own software process. The approach is a SEI-based one, but with some major customization and tailoring. Some interesting issues arise from being a case study in an international worldwide organization.

4 Software Engineering Laboratory

[1] Victor R. Basili. Tailoring SQA to fit your own life cycle. *IEEE Software*, 5(2):87–88, March 1988.

A short and informal presentation about introducing a process improvement initiative in a software organization, following a quantitative measurement approach.

[2] Victor R. Basili, Gianluigi Caldiera, Frank McGarry, Rose Pajerski, Gerald Page, and Sharon Waligora. The Software Engineering Laboratory–an operational software experience factory. In *Proceedings of the 14th International Conference on Software Engineering–ICSE 14*, pages 370–381. ACM, May 1992.

A concise and yet comprehensive presentation of the methods, techniques, and concepts realized in the Software Engineering Laboratory based in NASA/Goddard Space Flight Center. The concepts underlying Quality Improvement Paradigm for process evolution, Goal/Question/Metric approach to measurement, and Experience Factory for overall organization are described, real case studies are analyzed and discussed, and conclusions are supported by numerical evidence.

[3] Victor R. Basili and H. Dieter Rombach. The TAME Project: Towards Improvement-Oriented Software Environments. *IEEE Transactions on Software Engineering*, 14(6):758–773, June 1988.

TAME is both the name of a software engineering process model and of the Integrated Software Engineering Environment that support it. In this paper both are discussed, starting with the enunciation of the "Software Engineering Principles" and "Software Measurement Principles" underlying them.

5 European projects

5.1 BOOTSTRAP

[1] AA.VV. BOOTSTRAP: Europe's Assessment Method. *IEEE Software*, 10(2):93–95, May 1993.

This brief article describes BOOTSTRAP, a project carried out under the auspices of the European Strategic Programme for Research in Information Technology (ESPRIT). It aims to integrate SEI CMM, ISO 9000, and European Space Agency's standards in an assessment/improvement framework.

[2] Günter R. Koch. Process assessment: the 'BOOTSTRAP' approach. *Information and software technology*, 35(6/7):387–403, June 1993.

An in-depth overview of the BOOTSTRAP project. The general philosophy behind the project is explained, together with the assumptions, the methods, and the techniques employed. BOOTSTRAP is highly based on the SEI work on process assessment and improvement, consequently some analogies are provided and analyzed, also in terms of comparison of assessment results. Yet, the set of criticisms and reported benefits is provided.

5.2 ami

[1] Annie Kuntzmann-Combelles. Quantitative Approach to Software Management: the ami Method. In *Proceedings of ESEC'93: 4th European Software Engineering Conference*, September 1993.

This paper describes the ami approach, developed within an EEC ESPRIT project, which is deeply indebted with the Quality Improvement Paradigm and Goal/Question/Metric approach. Here the twelve steps of the method are explained and some lessons learned are explained.

[2] C. Debou, J. Liptak, and L. Pescoller. Managing Software Projects by Applying ami. *Annual review in automatic programming*, 16(2):103–108, 1992. Also in Proceedings of the 4th IFAC/IFIP Workshop.

This paper reports in more depth what is the aim of the ami approach, describing its role in the state-of-art of software metrics and process improvement.

6 Other related initiatives

[1] Inderpal Bhandari, Michael Halliday, Eric Tarver, David Brown, Jarir Chaar, and Ram Chillarege. A Case Study of Software Process Improvement During Development. *IEEE Transactions on Software Engineering*, 19(12):1157–1170, December 1993.

Addresses the problem of reducing defects in the software product while the process is being executed. The method proposed is partly human-executed and partly machine-executed. Project team members classify software defects, which are then ordered accordingly to a "interestingness" function and presented through a "filtering" function, both automatically computed.

The data so presented are then interpreted by project team members. Numerical data are presented.

[2] James O. Coplien. Borland Software Craftsmanship: A New Look at Process, Quality and Productivity. In *Proceedings of the 5th Annual Borland International Conference*, Orlando, Florida, June 1994.

A researcher from AT&T Bell Labs reports on the outstanding productivity achieved by Borland in building its *QuattroPro*© for Windows. Coding was done by no more than eight people at a time, generating 1 million lines of code in 31 months. Moreover, as the author says "[...] Borland is not subject to the ISO 9000 series process standards, has no concept of its SEI CMM rating [...]".

[3] Raymond Dion. Process Improvement and the Corporate Balance Sheet. *IEEE Software*, 10(4):28–35, July 1993.

Raytheon's Equipment Division has improved its software process, jumping from the level 1 of 1988 to level 3 of 1992. The improvement paradigm (based on the concepts expressed by Deming and Juran) is explained, together with the key success factors and a deep analysis of results, from both an economical and technical perspective.

[4] Raymond Dion. Elements of a Process-Improvement Program. *IEEE Software*, pages 83–85, July 1992.

Describes the Raytheon's process improvement program from an organizational/managerial perspective.

[5] Michiel van Genuchten. Why is Software Late? An Empirical Study of Reasons For Delay in Software Development. *IEEE Transactions on Software Engineering*, 17(6):582–590, June 1991.

Describes an approach to the identification of the causes that lead a software project on effort and/or schedule overrun. Data collected by project members are used to identify reasons for delay (which can fall into predefined classes), which are then further jointly discussed by the researcher, the project leaders and the managers in order to determine actions to be followed.

[6] Joel Henry and Bob Blasewitz. Process Definition: Theory and Reality. *IEEE Software*, 9(6):103–105, November 1992.

A General Electric experience in defining a software process, based on the deep belief that process definition is a strong prerequisite for software process improvement.

[7] Albert Johnson. Software Process Improvement Experience in the DP/MIS Function. In *Proceedings of the 16th International Conference on Software Engineering*, Sorrento, May 1994. IEEE Computer Press.

Corning's experience in improving its own software process. The experience started with a SEI assessment, and generated soon relevant outcomes and lessons learned.

[8] Kouichi Kishida, Masanori Teramoto, Koji Torii, and Yoshiyori Urano. Quality-Assurance Technology in Japan. *IEEE Software*, pages 11–18, September 1987.

An overview on Japanese software production practices, evidencing analogies and differences with respect to U.S. and Europe.

[9] Clement L. McGowan and Shawn A. Bohner. Model Based Process Assessment. In *15th International Conference on Software Engineering*, May 1993.

Describes an original approach to software process assessment. Starting from the consideration that "To model is to understand", the approach starts with determining the current situation of the assessed software process by modeling it using semiformal graphical notations based on SADT. Problems are identified on the model by review and discussion with process actors, and improvement are directly shown modifying the original model. The assessment group keep working with the assessed organization to guide the implementation of suggested improvements.

[10] Shari Lawrence Pfleeger. Lessons Learned in Building a Corporate Metrics Program. *IEEE Software*, 10(3):67–74, May 1993.

Metrics are not for free, so they have to be carefully managed like any other factor in software projects management. Moreover, metrics have to be tailored not only to the organization software process maturity, but on the specific software project, too. Starting from these considerations, the author reports a case study in establishing a corporate-wide metrics initiative in a large telecom organization, reporting both a detailed explanation of the activities involved and a list of key lessons learned during this initiative.

[11] R.A. Radice, J.T. Harding, P.E. Munnis, and R.W. Phillips. A Programming Process Study. *IBM Systems Journal*, 24(2):91–101, 1985.

Reports a study in assessing and improving some software processes in IBM. The supervisor of this study was Watts S. Humphrey, and in fact many of the ideas behind this study can be found in the SEI Capability Maturity Model and related assessment methodology. A good reading to understand where CMM comes from.

Note on electronical availability

The report [2] in Section 6 is electronically available via anonymous `FTP` at `netcom.com` in directory `/pub/jbach`.

The report [7] in Section 3.2 is electronically available via anonymous `FTP` at `src.doc.ic.ac.uk` in directory `/dse/modelling`.

The CMU/SEI reports [2], [3], [4], [5], [6], [8], [9], [10] in Section 3.1, and the report [1] in Section 3.2 are electronically available via anonymous `FTP` at `ftp.sei.cmu.edu` in directory `/public/documents/cmm`.