Reference:
Münch, Jürgen. Empirical Software Engineering Issues. Critical Assessment and Future Directions, chapter Effective Data Interpretation. Springer Verlag, 2007.
DOI: 10.1007/978-3-540-71301-2_24
URL: http://link.springer.com/chapter/10.1007%2F978-3-540-71301-2_24

Effective Data Interpretation

Jürgen Münch

Abstract. Data interpretation is an essential element of mature software project management and empirical software engineering. As far as project management is concerned, data interpretation can support the assessment of the current project status and the achievement of project goals and requirements. As far as empirical studies are concerned, data interpretation can help to draw conclusions from collected data, support decision making, and contribute to better process, product, and quality models. With the increasing availability and usage of data from projects and empirical studies, effective data interpretation is gaining more importance. Essential tasks such as the data-based identification of project risks, the drawing of valid and usable conclusions from individual empirical studies, or the combination of evidence from multiple studies require sound and effective data interpretation mechanisms. This article sketches the progress made in the last years with respect to data interpretation and states needs and challenges for advanced data interpretation. In addition, selected examples for innovative data interpretation mechanisms are discussed.

1 Introduction

Software practitioners and researchers increasingly face the challenging task of effectively interpreting data for project control and decision making, and gaining knowledge on the effects of software engineering technologies in different environments. This is caused, for instance, by the increasing necessity to use quantitative approaches in practice in order to climb up maturity ladders or the need for justifying software-related costs in the context of business strategies and business value. In the area of empirical research, there is a need to come up with sufficiently general, yet significant context-oriented evidence on the effects of software technologies based on data from individual or multiple studies.

This article focuses on three areas where data interpretation is relevant: (1) Interpretation of data for project control. Here, the focus is on project execution. Factors such as the increasing distribution of development activities, the need for monitoring risks, or regulatory constraints have accelerated the introduction of data-based project management techniques into practice. However, making valuable use of collected data is challenging and requires effective mechanisms for data interpretation. (2) Interpretation of data for individual empirical studies. Due to the specifics of software engineering studies, the data gained from such studies typically does not allow for the application of statistical analysis and interpretation techniques that are successfully applied in other fields (e.g., methods that require a significant amount of normally distributed data). Methods for data analysis and interpretation are needed that can cope with typical specifics of software engineering data. (3) Combination of evidence. Here, the focus is on aggregating evidence from multiple individual studies. Data or

V. Basili et al. (Eds.): Empirical Software Engineering Issues, LNCS 4336, pp. 83 – 90, 2007.
© Springer-Verlag Berlin Heidelberg 2007

results from different individual studies are typically heterogeneous and stem from different contexts. Interpreting data in order to gain aggregated combined evidence requires strategies and techniques to cope with these difficulties.

For these three areas, the article sketches the progress made in the last years with respect to data interpretation and states needs and challenges for advanced data interpretation. In addition, selected examples for innovative data interpretation mechanisms are discussed.

2 Data Interpretation for Project Control

Measurement is an important means for managing software development and maintenance projects in a predictable and controllable way. This requires particularly accurate and precise monitoring of process and product attributes. Systematic support for detecting and reacting on critical project states in order to achieve planned goals is needed. Single points of control are required to monitor, coordinate, and synchronize distributed development activities.

Progress

During the last years, we have observed several trends that are relevant for data interpretation for project control, especially:

- Increased industry awareness for data-driven project management and quantitive approaches. This is partially motivated by the application of maturity models, but there are also other reasons such as distributed development and globalization.
- Dashboards are currently being widely installed in industry. One of the reasons is that regulatory constraints often require higher process transparency.
- Measurement has begun to enter the acquisition process. There is, for instance, a trend for OEMs in the automotive industry to enforce measurement-based assessment of supplier software.
- Software is increasingly entering domains (such as transportation systems or medical engineering) that demand quantitative assurance of critical processes and product properties.

Selected Needs

We see the following selected needs as being important with respect to data interpretation for project control:

- Establish quantitative project control mechanisms.
- Obtain single point of control.
- Define process interfaces quantitatively.
- Integrate business and engineering processes. Currently, project controlling on the engineering level and on the higher management level are widely separated. Linking business goals to goals of the software organization of a company and to measurement goals is necessary for integrating these two levels of control.

Challenges

We see the following essential research challenges with respect to data interpretation for project control:

- How to interpret data in the context of software goals and business goals?
- How to visualize data in a purpose-, role-, and context-oriented manner?
- How to tailor and combine analysis, interpretation, and visualization techniques?
- How to integrate heterogeneous data from different sources?

Example: Software Project Control Centers (SPCC)

One means to institutionalize measurement on the basis of explicit models (i.e., process models, product models, resource models, and quality models) is the development and establishment of so-called software project control centers (SPCC) for systematic quality assurance and management support [7,9,10]. An SPCC can be defined as a means for process-accompanying interpretation and visualization of measurement data: It consists of (1) underlying techniques and methods to control software development projects and additional rules to select and combine them, (2) a logical architecture that defines logical interfaces to its environment, and (3) supporting tool(s) that implements (parts of) the logical architecture. Its input information includes, but is not limited to, information about project goals and characteristics, project plan information (e.g., target values per development phase), measurement data of the current project, and empirical data from previous projects. Its output information includes a context-, purpose-, and role-oriented visualization of collected and interpreted measurement data. That is, the visualization depends upon the context of the project, the purpose of the usage (e.g., monitoring), and the role of the user project manager. Its tasks include collecting, interpreting, and visualizing measurement data in order to provide context-, purpose-, and role-oriented information for all involved stakeholders (e.g., project managers, quality assurer, developers) during the execution of a software development project. This includes, for instance, monitoring profiles, detecting abnormal effort deviations, cost estimation, and cause analysis of plan deviations.

3 Data Interpretation for Individual Studies

Understanding the effects of software engineering techniques and processes under varying conditions can be seen as a major prerequisite towards predictable project planning and guaranteeing software (or system) quality. Evidence regarding the effects of techniques and processes for specific contexts can be gained by individual studies. Due to the fact that software development is a human-based and non-deterministic activity, the data gained in such studies typically has several limitations (e.g., limited validity and completeness) and is context-dependent. Effective data interpretation has to cope with this and support the derivation of results that are sufficiently general on an acceptable significance level.

Progress

During the last years, we have observed several trends that are relevant for data interpretation for individual studies, especially:

- New or enhanced analysis techniques and tools, e.g.,
  - for analysis of little and/or imperfect data sets
  - for combining quantitative data and expert opinion
  - for data mining
- Tools with new or enhanced capabilities, e.g.,
  - visualization tools (isolated cases)
  - tailorable product (and process) measurement tools

Selected Needs

We see the following selected needs as being important with respect to data interpretation for individual studies:

- Effectively interpret results for different stakeholders.
- Effectively develop or calibrate quantitative models for different purposes (e.g., reliability prediction).
- Preprocess imperfect data sets appropriately as prerequisite for applying data analysis techniques.

Challenges

We see the following essential research challenges with respect to data interpretation for individual studies:

- How to combine different analysis/interpretation techniques (e.g., statistical analysis and visualization)?
- How to guarantee data validity in industrial settings?
- How to preprocess imperfect data sets for analysis and interpretation?
- How to select quantitative models based on goals and available data?
  - What kind of data is needed?
  - Gap analysis: What data is missing for building the models?
- How to visualize data appropriately?
  - What are appropriate metaphors? Can they be standardized to a certain extent?

Example: Visualization

One approach to data interpretation is to use the visual capabilities of people. Visualization mechanisms support the understanding of the data and aspects under consideration, the abstract and compact representation of information, and the creation of a mental model of the data. Special visual environments have interactive capabilities and allow, for instance, easy navigation through the data by flexibly changing perspectives and abstraction levels (see, for instance, [14]).

4 Combination of Evidence from Individual Studies

One of the challenges of empirical research is to overcome the typically narrow scope of validity of the results. From the viewpoint of a practitioner, an important question is whether the results are valid for his own development context. A promising way to broaden the scope of empirical evidence is to summarize and organize evidence through integration and aggregation [5]. Integration means accumulating different kinds of evidence — ranging from quantitative results to qualitative practical experiences and human judgements. Aggregation means accumulating evidence from different contexts. Both, integration and aggregation, require effective interpretation mechanisms.

Gaining more evidence about processes, products or qualities should be packaged in explicit models. This requires a process for systematically evolving such models and creating variations of the models, if necessary. Fig. 1 illustrates such a process for evolving models. The process has been proposed by Rombach [6] and can be seen as a basis for packaging models in an experience base [3].

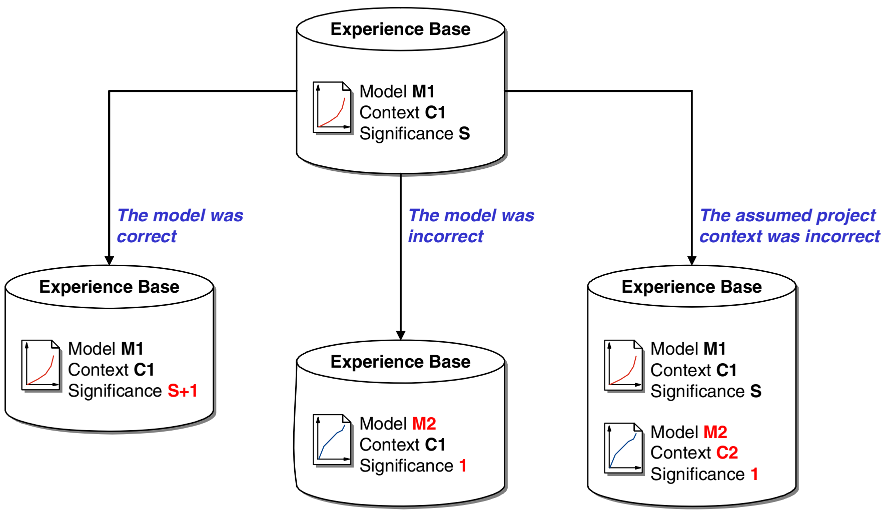

Fig. 1. A Process for Evidence-based Model Evolution [6]

Progress

During the last years, we have observed several trends that are relevant for combining evidence:

- Collections of empirical evidence (handbook, repositories) have been created.
- Many (company-specific) data repositories are available.
- The concept of virtual laboratories was developed.
- Variability concepts for products have been established.

Selected Needs

We see the following selected needs as being important with respect to combining evidence:

- Effectively derive, maintain, and present aggregated trustable evidence and statements (bottom-up).
- Verify aggregated evidence and identify lacks of evidence (top-down).
- Select and customize processes, techniques, tools, and products based on evidence.

Challenges

We see the following essential research challenges with respect to combining evidence:

- How to define appropriate operators for aggregating empirical evidence by taking the project context into account?
- How to present/visualize combined evidence for different stakeholders?
- How to identify lacks of evidence?
- How to reengineer GQM plans [2,13] from data repositories?
- How to describe process variability?
- How to represent available evidence and lacks of evidence for specific context variations?
- How to integrate evidence into process, product, and quality models as well as into tools? Evidence-based decision models for product lines and variant-rich processes are needed.
- How to evolve process, product, and quality models?

Example: Virtual Laboratory

Combining process simulation [1] and real experiments is a promising way to fill the areas of missing evidence between individual studies (e.g., combinations of impact factors that are not covered by real studies). This is addressed by the concept of Virtual Software Engineering Laboratories (VSEL), which was introduced at first in [12] and refined in [11]. One major motivation for such a virtual software engineering laboratory is cost reduction by simulating human behavior and the process environment of the method to be examined. Additionally, such a laboratory allows for better demonstrating the effects of a method in an understandable way. In particular, a multitude of variations of an experiment that is often necessary to cover different impact factors can be performed, and costs can be reduced enormously. Consequently, learning cycles can be shortened. In particular, empirical studies and process simulation can be combined in such a way that 1) empirical knowledge is used for the development and calibration of simulation models, 2) results from process simulation are used for supporting real experiments, 3) real experiments and process simulation are performed in parallel (i.e., online simulation).

5 Conclusions

Effective data interpretation plays an important role in software project management and for gaining evidence from empirical studies. Appropriate data interpretation, presentation, and dissemination of results can be seen as a major acceptance and success factor for quantitative project management and empirical studies. Concluding, we recommend the following when considering effective data interpretation:

- Make sure that the study is relevant and important before conducting the study ("test first"). For industry, the following questions might be relevant: Is there a need for the evidence? By whom? How will it be used? How does it relate to business goals? What is the cost/benefit relation of gaining the evidence? Is there a dissemination and exploitation strategy? For research, the following questions might be relevant: Is there a lack of evidence? How could the results be combined with other evidence?
- The combination of different analysis and interpretation techniques promises to broaden the scope of the evidence and provide new insights. Example techniques are visualization, simulation, qualitative analysis, quantitative analysis, and meta analysis.
- Consider data interpretation mechanisms early on during the establishment of project controlling mechanisms or the design of empirical studies.

Acknowledgements

I would like to thank Sonnhild Namingha from Fraunhofer IESE for preparing the English editing of this paper. This work was supported in part by the German Federal Ministry of Education and Research (Soft-Pit Project, No. 01ISE07A).

References

1. Armbrust, O., Berlage, T., Hanne, T., Lang, P., Münch, J., Neu, H., Nickel, S., Rus, I., Sarishvili, A., Stockum, S.v., Wirsen, A., "Simulation-based Software Process Modeling and Evaluation", In: Handbook of Software Engineering and Knowledge Engineering, Vol. 3: Recent Advances", (S. K. Chang, ed.), World Scientific Publishing Company, pp. 333-364, August 2005.
2. Basili, V.R., Weiss, D.M., "A Methodology for Collecting Valid Software Engineering Data". IEEE Transactions on Software Engineering 10(6), 728-738, 1984.
3. Basili, V.R., Caldiera, G., Rombach, D., "Experience Factory"; in: Marciniak J.J. (ed.), Encyclopedia of Software Engineering, Vol. 1, John Wiley & Sons, 2001, pp. 511-519.
4. Briand, L.C., Differding, C., Rombach, D., 1996. "Practical Guidelines for Measurement-Based Process Improvement". Software Process: Improvement and Practice 2(4), 253-280.
5. Ciolkowski, M., Münch, J., "Accumulation and Presentation of Empirical Evidence: Problems and Challenges", Proceedings of the 2005 workshop on Realising evidence-based software engineering (REBSE 2005), St. Louis, Missouri, pp. 1-3, USA, May 17, 2005.

6. Heidrich, J., Münch, J., Riddle, W.E., Rombach, D., "People-oriented Capture, Display, and Use of Process Information", In: New Trends in Software Process Modeling, (S. T. Acuña and M. I. Sánchez-Segura, eds.), Series on Software Engineering and Knowledge Engineering, Vol. 18, World Scientific Publishing Company, pp. 121-179, 2006.
7. Heidrich, J., Münch, J., Wickenkamp, A., "Usage-Scenarios for Measurement-based Project Control", Proceedings of the 3rd Software Measurement European Forum (SMEF 2006), (Ton Dekkers, Ed.), pp. 47-60,Rome, Italy, May 10-12, 2006.
8. Kellner, M.I., Madachy, R.J., Raffo, D.M. 1999. "Software process simulation modeling: why? what? how?", Journal of Systems and Software 46(2/3): 91–105.
9. Münch, J., Heidrich, J., "Software Project Control Centers: Concepts and Approaches", International Journal of Systems and Software, vol. 70, issues 1-2, pp. 3-19, February 2004.
10. Münch, J., Heidrich, J., "Tool-based Software Project Controlling", In: Handbook of Software Engineering and Knowledge Engineering, Vol. 3: Recent Advances", (S. K. Chang, ed.), World Scientific Publishing Company, pp. 477-512, August 2005.
11. Münch, J., Pfahl, D., Rus, I., "Virtual Software Engineering Laboratories in Support of Trade-off Analyses", International Software Quality Journal, Special Issue on "Trade-off Analysis of Software Quality Attributes", vol. 13, no. 4, pp. 407-428, December 2005.
12. Münch, J., Rombach, D., Rus, I., "Creating an Advanced Software Engineering Laboratory by Combining Empirical Studies with Process Simulation", Proceedings of the 4th International Workshop on Software Process Simulation and Modeling (ProSim 2003), Portland, Oregon, USA, May 3-4, 2003.
13. Rombach, D., 1991. "Practical benefits of goal-oriented measurement". Software Reliability and Metrics, 217-235.
14. Schäfer, T., Mezini, M., "Towards More Flexibility in Software Visualization Tools", in Proc. VISSOFT'05, IEEE CS Press, 2005.