Transactions on Information and Communications Technologies vol 11, © 1995 WIT Press, www.witpress.com, ISSN 1743-3517

Achieving quality improvement through understanding and evaluating information systems development methodologies

E. Georgiadou, C. Sadler
School of Computing, University of North London,
2-16 Eden Grove, London, N7 8EA, UK

Abstract

Since the 70s literally hundreds of different methods and tools have appeared each claiming to ease the life of the developer and the user by achieving improved productivity without compromising the quality of the software artefact. These methodologies range from integrated collections of procedures to single technique, notations, 4GLs and tools for supporting the process at the various stages of the systems lifecycle [1, 2, 3, 4].

This paper discusses how an organisation wishing to improve their development practices embarks onto the time-consuming and expensive .process of evaluation of methods and tools. The underlying complexity and application domain will themselves be decisive in the choice of methodology. The improvement process starts with the *understanding phase.* Here, we need to identify the important features of a methodology such as usability, portability, adaptability and functionality, and the nature of the problem(s) the methodology will apply to. We draw the distinction between problem, methodology (procedures, techniques) and tools and discuss their interrelationships [4, 7]. The *evaluation phase* starts with the specification of acceptance criteria and it involves the study of the features identified during the understanding phase against these criteria. Evaluations can be qualitative and quantitative.

Finally we report our experiences with using modules from the DESMET[1] methodology [3, 5, 6, 8, 9] for both qualitative and quantitative evaluations and we suggest an alternative taxonomy of methodologies in the light of our experience. Our results show that despite the plethora of methods and tools there is no panacea but 'most appropriate' choices according to the underlying problem under consideration [6, 7, 9].

A historical perspective

Today, it is hard to think of information systems without computers although organisations have dealt with information and information processing long before the proliferation of information technology. The information systems development function has tended to mirror the evolution of technology.

At the beginning systems development concentrated solely on what we know now as programming: assembly language followed by FORTRAN and COBOL using the mainframe and service bureau technology. In the 60s and 70s there was awareness of the analysis and design function. The hardware was cheaper and more accessible and the technology moved into alternative languages (ALGOL, PASCAL). The traditional lifecycle (NCC) identifies distinct stages and phases of systems development.

The need for methods was recognised in the late 60s at the same time as the emergence of the term Software Engineering. This came as a result of the whole of human activity (transport, manufacturing, service industry, social services such as health, education, entertainment) being underpinned by the use of computers and computer systems which tended to be late, over-budget and unreliable.

The development of methods expressed a constant search for solutions to these problems particularly cost saving and product quality improvement. The majority of methods concentrated on technical aspects ignoring the organisational structure and culture and often ignoring the user The latest fever comes in the form of Object-Oriented methods which are believed to be the answer to all ills in information technology. [4, 7, 13].

However, the purely technology-oriented approaches have not always been effective. The whole family of Structured Methods, Information Engineering, JSD and Formal Methods tend to concentrate mainly on technical issues often ignoring 'people problems'. The Soft Systems Method [11] established the principle of the owner's viewpoint emphasising the importance of the human involvement. The real exception

---

[1]DESMET is a DTI/SERC funded project with the following partners:
NCC, GEC Marconi Systems, BNR and the University of North London

was provided by the ETHICS methodology which has a socio-technical emphasis recognising the importance of knowledge and psychological fit from the developers and most importantly from the user's point of view [12]. Neither of these methodologies covered the whole of the development process needing to 'borrow' techniques mainly from the structured methods for the implementation of solutions.

Most recently, the development of methods has been taking place in parallel with the development of automated tools particularly CASE and I-CASE providing the environment for correctness, consistency and completeness of products through the use of central repositories, data dictionaries and encyclopaedias. Adherence to notational standards and structured procedures enabled the improved allocation of specialised, inexperienced and experienced staff to appropriate tasks and facilitated team work. Many of these developments reflected similar developments in the manufacturing industry although in systems development the focus was primarily placed on product improvement rather than process improvement which came as a by-product [4].

The understanding phase

The importance of the understanding phase can be seen in the following statement taken from a European Directive for research in the European Union : *"There is a pressing need to understand, develop and promote the technologies which will enable Europe to produce reliable, correct, efficient and user friendly software"* [10]. In order to understand the structure and function of systems we need to identify their boundaries, their environments, their components and the structure of these components. Most importantly we need to describe the purpose, behaviour and properties of the system. A system in this context can be a solution or a methodology (system for finding solutions).

Methods are applied to problems and therefore there is a requirement to analyse and understand the problem prior to matching a methodology to its solution. Furthermore, we need to understand the technologies which provide the environment for the development and operation of the solutions to the problems. It is fundamental to understand both the organisation and business requirements as well as politics within and outside the organisation in order to deal with management effectively and the technical concepts, assumptions and objectives of the particular system.

All methods use the systemic approach in order to model problems and their solutions. The degree of rigour, the techniques employed and the range of lifecycle coverage define the nature of the method. If we consider methodologies as systems we can identify their components and their inter-relationships.

Figure 1a depicts the fundamental components of a systems development process and illustrates the role of the components involved. A problem may be solved in many different ways and in fact, several solutions are possible. Two of the poles show the developers and the users pulling in their own direction. The strength of 'pull' provides the weighting that will result in a composite route sometimes veering towards the technical approach and sometimes towards the people/user oriented approach (shown in Fig. 1b and 1c respectively). The methodology underlies the route in each case.

Figure 1d shows the development route in prototyping: constant feedback from the users and team, collaborative development tending to achieve a near-perfect solution, trading off technical effectiveness to usability as shown on the solution space. The whole system of transformation from problem to solution can be likened to a system of musicians playing their instruments and making music. The musicians are the systems developers, the instruments are the methods and the music the solutions produced.

Using the techniques and tools of a methodology we can understand a problem situation by using abstraction, generalisation, classification and specialisation to model the system. The models elicitate the functionality of the system and introduce a degree of structure and rigour enabling us to produce solutions satisfying the identified requirements.

The role of the methodology as an instrument of understanding is provided by the its underlying lifecycle namely the organised phases, steps, tasks and checklists which steer the developer towards identifying and specifying the components, sub-systems and their interactions. Therefore in trying to choose and apply a methodology valuable insights are gained into the original problem and the existing procedures, highlighting problem areas and additional requirements.

The unstructured problem comprises the users and their perception of the situation, the environment, the current worries and problem, their expertise in the application domain (business) and the additional requirements of the application. The methodology provides the techniques and the control and management structures whilst the developer provides the technical expertise and knowhow. Several models of the problem and the required solution are produced using the techniques and tools. The solution represents the final model in terms of a functioning system satisfying the identified requirements to a greater or lesser degree.

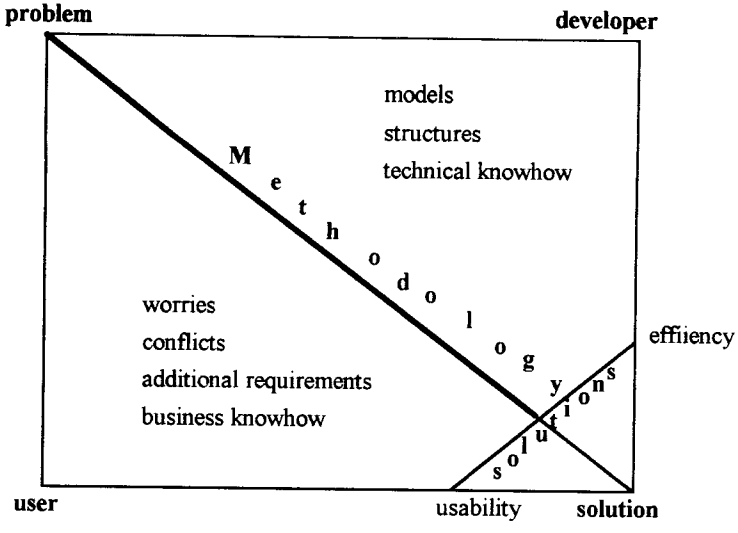

Fig. 1a - The components in the development process The 'ideal' situation

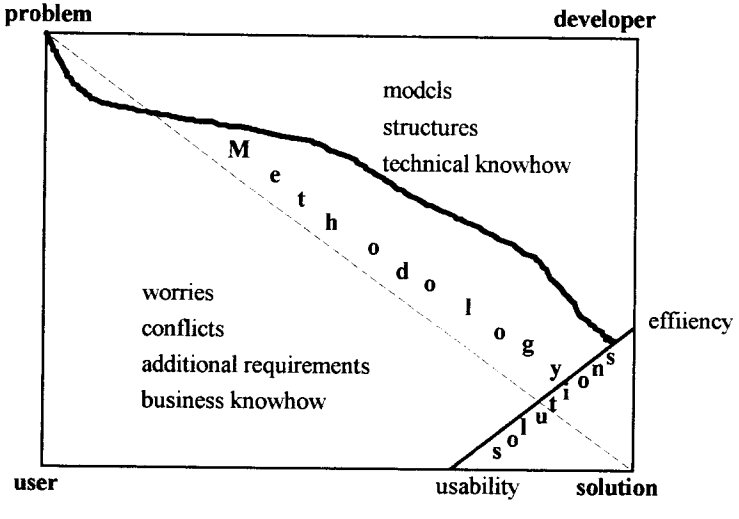

Fig. 1b - The components in the development process Technical bias -the developer taking over

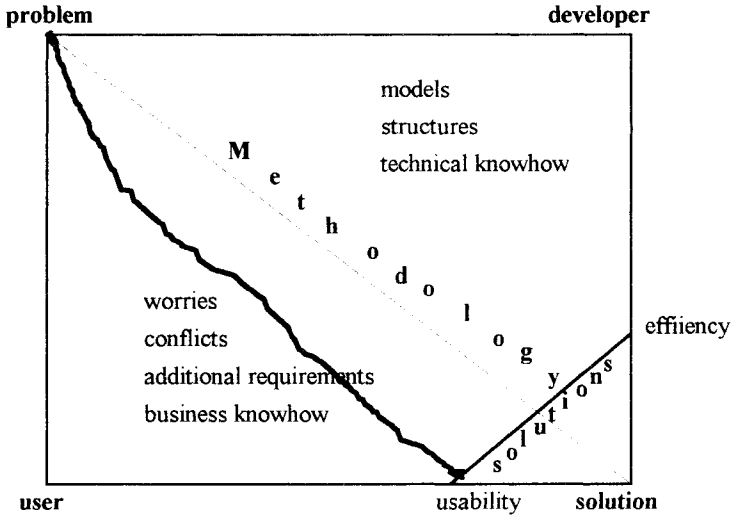

Fig. 1c - The components in the development process
The user imposing pace and style of development

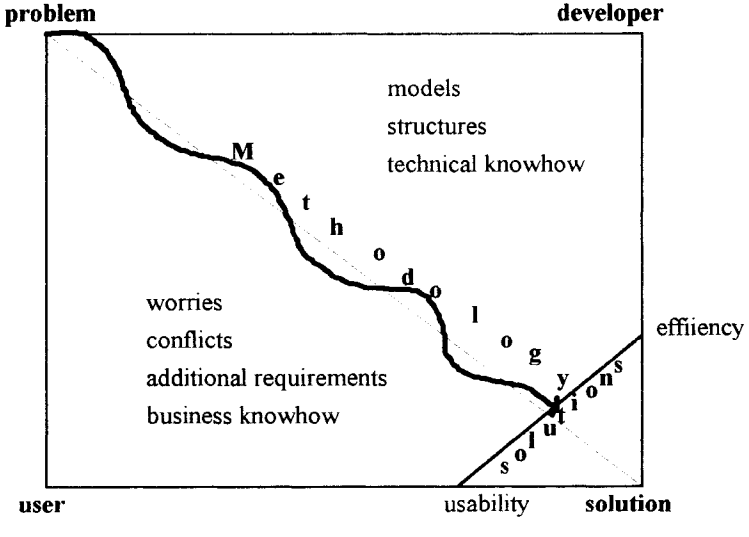

Fig. 1d - The components in the development process
The 'balanced' interaction - prototyping

The level of user participation is inherent in each methodology. Participative methodologies place a strong emphasis on the managerial and social issues and, usually, devote extra time and effort during the early stages of the systems lifecycle. This moulds the development process and the type of solution(s) achieved. User involvement is mainly through discussions with the developers and participation in walkthroughs, reviews and inspections.

Selecting candidate methodologies

During the understanding phase the problem under investigation can be classified in terms of size, reliability requirements, application domain, business priority etc. There are many different methodologies (approximately 2000 brand names!) developed by different people or organisations used in different countries [4]. It is possible that a particular methodology is more appropriate for solving a class of problem e.g. Ward-Mellor for Real-Time systems.

Information systems practitioner are only interested in selecting a methodology that is appropriate for their particular environment, capable to address their problems and finally enable to enhance their productivity.

In order to narrow down the immense list of available methodologies it is necessary to identify attributes or features, a set of properties and qualities of a methodology. Features of interest include usability, portability, functionality etc. Feature Analysis (FEA) [5] lends itself to situations where there is a plethora of choices, no prior data, no committed budget and even no prior experience. Having studied the problem the methodology requirements can be identified and classified. This way methodologies can be discarded (e.g. obvious absence of mandatory features renders the methodology unusable). Similarly the presence of required features suggests possible future choice.

Thus a short-list of candidate methodologies is generated. The degree of the required rigour, tolerance levels and scoring systems will play decisive role in the final choice of methodology. which will take place after the evaluation stage.

The evaluation phase

"Making the wrong choice of methods and tools can be a big risk to an organisation (and to those who made the choice), because the acquisition and installation costs can be very high and the consequential costs of project disruption and delay can also be very high" [9]. Most industrial evaluations of methods and tools were based largely on subjective choice of product features.

Towards the end of the 80s and in the early 90s support for research into evaluation issues came in the form of funding from the Department of Industry for three related and complementary projects namely DESMET (Determining an Evaluation Methodology for Software Methods and Tools), SMARTIE (Standards and Methods Assessment using Rigorous Techniques in Industrial Environments) and META (Measurement-based Experiment for Techniques Assessment).

The knowledge and expertise reported is mainly derived from the authors' involvement with the DESMET project throughout its development and trials. The DESMET project aimed to address the problem of objective determination of the effects and effectiveness of methods and tools for the development and maintenance of software-based systems. It was established that evaluations can be qualitative or quantitative. Several modules providing structured guidelines and techniques have been proposed. The qualitative approach was divided into Feature Analysis (FEA) and Qualitative Effects Analysis (QEA) whilst the quantitative approach was divided into three categories namely formal experiments, survey analysis and case study [8].

In order to introduce more discipline into qualitative studies it is possible to use one of the methods (experiments, surveys or case studies) usually associated with quantitative approaches. The scoring on the chosen scale (e.g. 0 - 5) can be depicted in a Kiviat diagram showing the deviations from the tolerance level.

An example

The results (averages) of a survey which attempted to compare SSADM and ETHICS are shown in the table below.. The scoring was carried out using a scale of 0 to 5 where 0 denotes bad or unacceptable and 5 excellent. The Kiviat diagram (fig. 2) assuming equal weighting for all features shows the strengths and weaknesses of the methodologies against the ideal and worst measures.

□	**SSADM**	**ETHICS**
Lifecycle Coverage	3	2
Usability	3	5
Portability	2	5
Adaptability	3	4
Availability	4	4

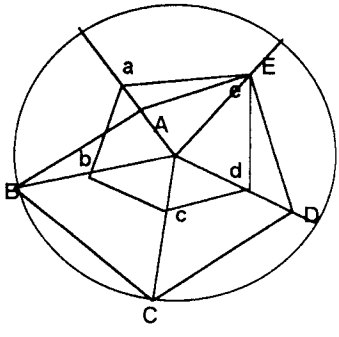

Fig. 2 - Kiviat diagram comparing two methods

The area of each polygon shows the deviation from the desired level. The larger the area the better the fit. In this study the results show that the ETHICS methodology is preferable to SSADM according to the specified criteria and adopted scoring since the area of ABCDE is larger than the area of abcde. .It is important to stress that there is a large degree of subjectivity in the choice of the attributes and the criteria. However if the technique is applied to two methodologies we can carry out comparisons.

In this study it was assumed that all chosen attributes carry equal weighting hence the circle was divided into five equal sectors. The introduction of weightings changes the shape of the polygon and shows the emphasis and importance of specific attributes such as lifecycle coverage.

Following the evaluation of the candidate methodologies and provided that there is a unified class of problem under consideration it is possible to select one methodology which best satisfies the criteria. For large organisations with a variety of applications it may not be possible to select just one methodology.

Towards a Classification of Methods

Although there are thousands of methodologies (brand names) they tend to form clusters or families with general characteristics dictated by their philosophy and type of problem they attempt to solve. Thus they have been divided into data-driven or process driven, hard and soft, large and small depending on the viewpoint of the researchers studying the whole area of methodologies.

After the completion of the case studies and experiments carried out at the University of North London we propose a composite model which incorporates the qualitative and quantitative aspects of user involvement, application domain and size of application. The qualitative aspects require the choice of attributes (using for example FEA) and specification of criteria such as presence of an attribute (mandatory) and degree/level of tolerance such as the least 3 (on a scale of 0-5) for usability. The quantitative aspects require statistical and mathematical measurement for additional rigour, repeatability and ultimate automation.

Thus we enhance the existing taxonomy of Hard versus Soft by the addition of scoring and weighting specific to a company, application or even chosen attribute. Hybrid methodologies or use of complementary techniques from a variety of methodologies is another route. A good example of this approach is Multiview (Avison & Fitzgerld) and more recently the Eurometh project which attempts to harmonise the procurement and adoption procedures throughout the European Union. Fig.3 shows a classification tree with the traditional sub-classes of methodologies.

Michael Jackson [7] argues that apart from the general classifiaction of problems there is the need for specialisation in order to deal with *'a relatively small class of problem'*. He mentions the case of compiler writing as one such specialisation .... He continues *'only specialisation, and the problem-oriented understanding that must guide it, can advance software engineering from an amateur to a professional activity".*

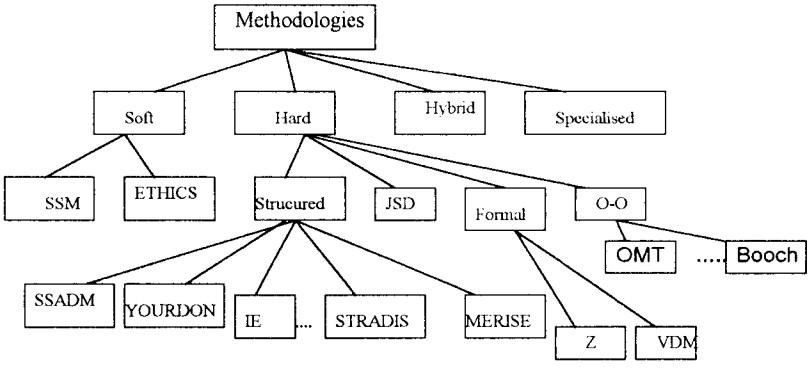

Fig. 3 - Methodology Classification Tree

Conclusion and Process Improvement

It is important to establish in a scientific way whether the introduction of a specific method or tool will have a beneficial impact on the software development process. Obviously, the ultimate aim is to improve the quality of the resulting products.

The phases of understanding and evaluation and their interaction provide the mechanism for increasing probability of improvement. Understanding the problem, shows the weaknesses and indicates the most profitable leverage points. The evaluation provides feedback and quantitative results introducing rigour, repeatability and thus possibilities for automation of the process.

It can be concluded that the attributes and criteria for selection introduce a large degree of subjectivity. And finally, although it may not be possible to select a methodology or tool for a particular problem, the fact that the stages of understanding and evaluation have taken place help in the improvement of the process which was the main aim of the exercise.

Future Work

A series of evaluations, surveys and case studies are under way using the various modules of DESMET within the University and with industrial partners in the UK and in Europe.

Acknowledgements

The authors would like to thank all the colleagues for their suggestions and encouragement throughout the life of the project. Particular thanks to Glafkos Christofides, Maria Papadopoulou, Gordon Frank and Gergina Kyprianou for making it possible to complete and send this paper to the technical committee in good time.

References

[1] Law, D.T. & Naeem, T. "DESMET: Determining an Evaluation Methodology for Software Methods and Tools", Proceedings of BCS Conference on CASE - Current Practice, Future Prospects, Cambridge, England, March 1992

[2] Glib, T. "Software Metrics", Cambridge, Mass., USA, Winthorp Publishers, 1977

[3] "DESMET, Determining an Evaluation Methodology for Software Methods and Tools", State of the Art Report, DTI, 1991

[4] Jayaratna, N. "Understnding and Evaluating Methodologies", McGraw-Hill, 1994

[5] Jones, L. "DESMET Qualitative Evaluation Procedure using Feature Analysis", NCC, February 1993

[6] Georgiadou, E., Mohamed, W-E., G., Sadler, C. " Evaluating the Evaluation Methods: :The Data Collection and Storage System using the DESMET Feature Analysis", 10th International Conference on Quality, Jerusalem, Israel, November 1994

[7] Jackson, M. "Problems, methods and specialisation" , Software Engineering Journal, Vol.9, No.6 , November 1994

[8] Kitchenham, B. 'DESMET - Guidelines for Evaluation Method Selection", DES/WP2.2.7, October 1993

[9] Kitchenham, B.A., Linkman, S.G., Law, D.T. "Critical Review of Quantitative Assessment", Software Engineering Journal, Vol.9, No.2, March 1994

[10] Working Document of the Commission Concerning the S&T Content of the Specific Technological Development (1994-1998) and the Framework Programme for Community Research and Training for the European Atomic Energy Community (1994-1998), Brussels, October 1993

[11] Checkland, P. B. "Information Systems and Systems Thinking: Time to Unite ?", International Journal of Information Management, Vol. 8, no. 4

[12] Mumford, E. "Participative Systems Design: Structure and Method", Systems, Objectives, Solutions, North- Holland, 1981

[13] Coad, P. & Yourdon, E. "Object-Oriented Analysis, IEEE Transactions, New York, 1989