

Chapter 4

Usability Engineering Integration as an Adoption Problem

There is evidence that the large body of knowledge detailed in the previous section often fails to be used, adopted and deployed in industry settings. As discussed in Chapter , important barriers to effective integration and adoption include their unfamiliarity with developers, their lack of maturity, and their limited support for integration. These barriers motivated new research avenues on how usability can be made more easily adopted by inserting them as extensions to commonly used software engineering methods and tools. This chapter discusses these avenues as well as the different possible approaches for the development of an adoption-centric methodology. The question that arises is how can these measurements be gathered without introducing excessive overhead to the development projects? Furthermore, how can the gathered data from different projects be integrated to yield stronger conclusions? How can the data and abstract models of the framework be presented in a form that is useful to software engineering teams? In this chapter these questions are further analyzed. Disciplines that could contribute to a solution are identified. The potential contributions of these disciplines are described.

4.1 Key Milestones in the Adoption Process

Two approaches for deploying, adopting and institutionalizing new methods are possible. The first one, expert-based institutionalization, requires resorting to third-party companies or involving external consultant experts who can help the team acquire, implement and institutionalize the new software engineering methods. The second one, evidence-driven adoption, involves adopting a measurement program for learning about the effectiveness of software engineering methods. The potential benefits of this type of approach are just beginning to be explored. Here, we will focus primarily on the second of these two adoption approaches.

Mayhew (1999) proposes three milestones for adopting software usability engineering methods at an organizational level. These steps are promotion, implementation and institutionalization. Tables 4.1–4.3 summarize these steps while highlighting the organizational and human barriers. None of these obstacles is easy to address. However, failing to recognize and address them will doom the organiza-

Table 4.1 Major obstacles in promoting usability engineer (UE) methods

Step	Mains activities	Tips and obstacles
Promotion	Identify and address organizational obstacles to change	Obstacles: – Prevalent myths, beliefs, and attitudes – Organizational incentives and practices – A high visibility of possible disaster – A commitment to address general business goals
	Identify and exploit potential motivators	Tips: – Know your audiences – Cast yourself as an ally, not an enemy – Check the credibility of motivators and their communication skills

Table 4.2 Obstacles in implementing a usability method

Step	Activities	Tips and obstacles
Implementation	– Staffing the function – Organizing the function – Defining roles – Providing career paths – Planning and budgeting	– Focus efforts on the projects of one particular section of the overall development organization – the one that seems most receptive – Expand resources, introducing techniques to other parts of the organization – Find high-visibility, high-impact projects, then expand to cover all projects

Table 4.3 Obstacles in institutionalizing the usability function

Step	Activities	Tips and obstacles
Institutionalization	– Integrate the new method with the development organization’s standard methodology – Get the new method into both the formal, documented methodology and into the actual, living, practiced methodology	– Many organizations go successfully from stage one to stage two and never get to stage three – Some organizations simply stay in phase two, with usability having an impact on certain projects but never becoming an institutionalized part of the overall development process – Most organizations fail to be strategic in phase three

tional efforts to failure and the personnel to a great deal of professional frustration. In particular, the two first steps – promotion and implementation – are often neglected and done ad hoc even though they are critical prerequisite steps for successful institutionalization.

In the first step, called promotion, the main focus is to influence people while identifying champions or motivators (Table 4.1). It is important at this step to make sure that the management staff share the same understanding about the new usability engineering technology and that they are committed to providing the necessary

resources to move into the next phase. Having an influential internal advocate at the management level virtually guarantees such a commitment.

The second step, called implementation, consists of influencing the project and product. In this step, the new method will be practiced in a specific project or product (Table 4.2). This step requires that the manager in charge of the product or the project be one of the motivators who were identified in the previous step. The motivator should be an engineer and not a visionary.

The last step, institutionalization aims to transform the new technology or method to become a standard or institutionalized part of the way the whole organization does business, to develop the whole product line and to ensure quality (Table 4.3).

4.2 On the Development of Adoption-Centric Usability Methods

A multiperspective approach is required to facilitate a faster and more systematic adoption of the existing UE techniques in the mainstream software development processes. As indicated in Fig. 4.1, a potential solution can be found at the intersection of the fields of usability engineering, empirical software engineering and organizational learning.

In the field of usability engineering, few studies exist on the relationships between project constraints, failures and the perceived benefits of specific UE techniques. In fact – similar to software engineering – many UE frameworks and methods lack any form of assessment or empirical evidence regarding the cost-effectiveness of usability. The data delivered by such studies are critical for evaluating why, when and how a method is suitable with respect to project characteristics such as software type, domain, interaction technology and the degree of impact of system defects.

Empirical evidence can be used to support decision-making in selecting a subset of UE techniques for the most prominent project settings in an organization. Data on the usability of UE techniques from the point of view of project teams can help to improve the acceptance of the UE approach and to establish UE techniques in the overall development process.

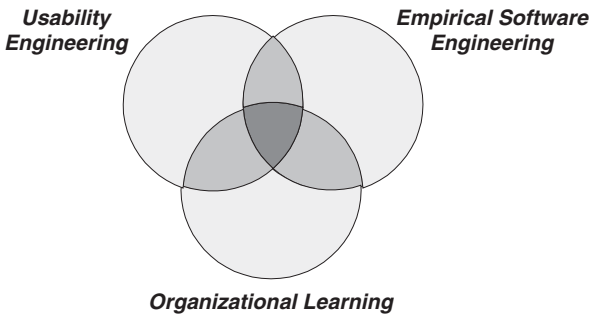


Fig. 4.1 Intersection of disciplines

From the organizational perspective, UE techniques have to be assessed as an organizational learning task. The term “organizational learning” has been discussed from the perspectives of fields as diverse as management science, psychology, sociology, and, more recently, information systems. However, a widely accepted model or even a consistent definition of organizational learning is still missing.

The classical definition by Agyris describes organizational learning as a process of “detection and correction of errors” (Agyris and Schön, 1978). According to Agyris’ definition, an organizational learning system is the “ecological system of factors that facilitate or inhibit the learning activities of individuals”.

This definition has been updated by more recent interpretations of organizational learning. Newer approaches view organizational learning as a process of knowledge acquisition, information interpretation and distribution (Huber, 1995) with the goal of having this knowledge being to modify the behavior of organizations (Garvin, 1993). The knowledge is acquired through experimentation, observation and analysis of both successes and failures (McGill et al., 1992).

This updated definition can be applied to studying the adoption of UE methods in software development organizations. Knowledge about UE techniques should be collected and disseminated in software development organizations to make the UE techniques accessible and easy to use for software engineering teams.

The emerging field of empirical software engineering also provides frameworks for conducting software engineering experiments to assess, compare and support decision-making regarding the usefulness, accuracy and adaptability of software engineering techniques (Basili et al., 1999). The most rigorous stream of this research interprets software engineering as a laboratory science.

According to this new research stream, software engineering techniques should be studied by using experimental designs and research methods similar to the controlled experiments conducted in classical disciplines such as physics, chemistry, usability engineering and HCI research.

However, such an approach to empirical software engineering suffers from several basic problems including measurement obstacles, difficulties in replicating experiments and the economical infeasibility of conducting controlled experiments in industrial settings (Juristo and Moreno, 2001). Controlled experiments are conducted almost exclusively in labs, for research purposes. The scope of the experiments is severely constrained to reduce the number of variables and allow the application of classical experimental designs and statistical tests.

Critics argue that such laboratory studies often fail to address significant problems and that “defining and executing studies that change how software development is done the greatest challenge facing empirical researchers” (Perry et al., 2000).

The framework proposed in this chapter provides means for collecting, accumulating and exploiting empirical data on UE techniques during projects. Moreover, the envisioned framework examines and exploits the relationships between the criteria for assessing usability on the one hand and the respective contexts of use on the other hand. The goal is to support evidence-based selection, deployment and evolution of UE methods in future software development projects.

4.3 Difficulties of Building an Empirical Driven Adoption Method

Several surveys on information system methodology research indicate that, in most cases, empirical studies of the effects of proposed techniques are largely missing. One consequence of this gap is that researchers kept proposing “new” engineering methodologies that, in essence, had already been built – they varied primarily in the names chosen for constructs (Smith and March, 1995). This uncertainty about the constructs proposed – mainly caused by a lack of empirical evaluation – leads to problems for both researchers and practitioners.

For researchers, the question of how to optimally integrate UE techniques into the software development lifecycle generally degenerates into “religious wars” (Paterno, 2002). A current prominent example is the dispute between the scenario-based and task-based camps (Benyon and Macaulay, 2002); (Carey, 2002); (Carrol, 2002); (Paterno, 2002). Such disputes are largely caused by a lack of empirical evidence to support a decision. Based on which data should a researcher improve or extend a UE technique if empirical data are not available on its advantages and drawbacks?

For practitioners, the situation is even worse. Based on which criteria should a project manager, usability engineer or user interface software developer select an approach, methodology, technique, best practice, pattern or guideline? If there is no empirical data to justify the selection of a technique, then why should anyone care about these techniques at all?

A look at software engineering where similar problems exist provides no ad hoc solutions to this problem. “No matter which software engineering methods are selected for a particular project, in almost all cases, there is little hard evidence backing up these decisions, and their costs and benefits are rarely understood” (Perry et al., 2000). In fact, the fundamental factors and mechanisms that drive the costs and benefits of software tools and methods are largely unknown (Basili et al., 1999). Therefore an Adoption-Centric Usability Engineering approach should provide means for incrementally building such a body of evidence. Without it, choosing a particular UE technique for a project becomes a random act.

In more mature scientific disciplines such as chemistry or physics, patterns of knowledge are built using the scientific principle of hypothesis building and hypothesis testing. These scientific principles are likewise necessary for examining UE techniques. However, it is initially unclear how to transfer these principles to examining the quality of UE techniques.

If we accept the scientific cycle of hypothesis building and hypothesis testing as a framework for building a pattern of knowledge of UE techniques, we first have to know which questions we want to ask. Each UE technique contains an implicit hypothesis that the technique is an appropriate means for solving a specific problem in the engineering of interactive systems.

Finding an appropriate experimental setting for such experiments continues to be a problem. A number of settings are possible for conducting experiments with

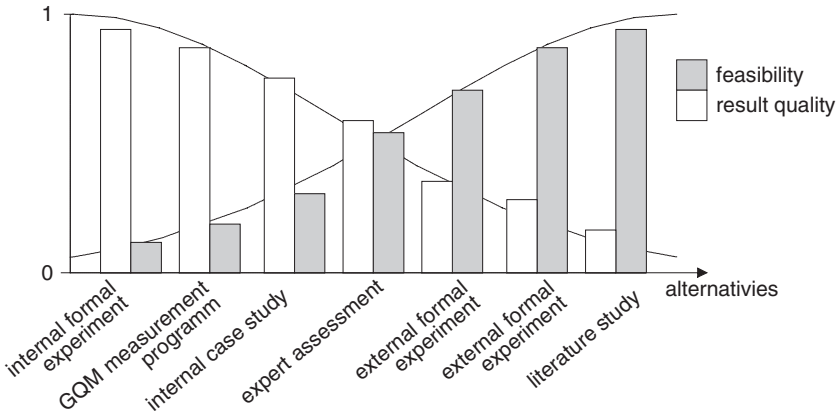


Fig. 4.2 Potential experimental settings

software engineering techniques (Basili et al., 1999); (Houdek, 1999); (Kitchenham, 1996); (Zelkowitz and Wallace, 1998). Houdek differentiates between internal formal experiments, internal case studies, expert assessments, external formal experiment, external case studies and literature studies (see Fig. 4.2).

According to this classification, “external” means that the respective study is conducted in a non-industrial environment, for example as part of a university course. Houdek ranks these alternatives according to the quality of results generated by the studies and their feasibility. According to Houdek’s classification, internal formal experiments yield an optimal quality of results, but are almost impossible to conduct in most industrial software engineering settings (Houdek, 1999). At the other extreme, external case studies are rather easy to handle for the industry partner, but it is hard to interpret them and transfer the conclusions back to an industrial setting.

According to Houdek’s classification, internal case studies and expert assessments are a good compromise between quality of results and feasibility. A project P in which a UE technique U is applied must be interpreted as an experiment to examine the quality of the technique under the conditions of project P .

But it still remains unclear how to measure the quality of UE techniques. The principles of natural science cannot be directly carried over to the study of engineering methods. Recently, it has been argued that research on the engineering of software systems needs to adopt research methods of the social sciences. This stream of software engineering research is interested in learning about the engineering of systems by observing the complex social settings existing in software engineering projects (Lofland and Lofland, 1995). Recent results of this research stream indicate that the organizational context where the interaction among humans takes place is the critical factor that determines the quality and effectiveness of the results.

A less technology-focused study of the quality of engineering methods can also be motivated from a philosophy of science point of view. Techniques for developing interactive software systems clearly belong to the domain of design science (Pinheiro da Silva and Paton, 2001); (Simon, 1981); (Smith and March, 1995).

While the aim of natural science is to understand and explain phenomena, the aim of design science is to create things that serve human purposes and develop ways to achieve human goals.

The utility principle yields important implications for measuring the quality of UE techniques. A prerequisite for the adoption of techniques is their acceptance by the people who are to use them. In the social sciences, various models exist for measuring acceptance. The technology acceptance model (TAM) represents one of the most prominent theoretical contributions to the understanding of acceptance behaviors (Davis et al., 1989); (Robey, 1996).

TAM aims to understand and predict the usage of computer systems (Davis, 1986). TAM postulates that tool acceptance can be predicted by measuring two dimensions: the perceived usefulness and the perceived ease-of-use of a software system.

TAM is based on fundamental theories of psychology such as the theory of reasoned action (TRA) (Fishbein and Ajzen, 1975). TRA examines which factors determine consciously intended behaviors. According to TRA, a person's performance of a specified behavior is determined by two central factors:

- First, by the person's attitude toward the behavior (A)
- And second, by the subjective norm (SN): the social pressure put on the person to perform the behavior

Attitude and subjective norm together form the behavioral intention (BI) of the person to perform the behavior.

TRA is a widely studied and accepted model of social psychology (see [Sheppard et al., 1988] for a review). The theory of reasoned action is depicted in Fig. 4.3.

Davis' technology acceptance model uses TRA as a theoretical basis and adapts it to the specific case of technology acceptance. Davis proposes that the attitude toward using an information system is influenced by two determinants: the perceived usefulness (PU) and the perceived ease-of-use (PEU) of a system.

The **perceived usefulness** of the system expresses the "subjective probability that using a specific application system will increase (the user's) job performance within an organizational context", i.e. it is a measure for the perceived utility of the system.

The **perceived ease-of-use** is the main factor that influences the acceptance of a system. Davis defines perceived ease-of-use as "the degree to which the user expects the target system to be free of effort", i.e. it is a measure for the usability of a system.

Together, perceived ease-of-use and perceived usefulness constitute the person's attitude toward using a system. The attitude (A) and the perceived ease-of-use (PEU) influence the behavioral intention (BI) which can be used to predict the actual use of the system. The technology acceptance model (TAM) is depicted in Fig. 4.4.

The postulated effects of perceived usefulness and perceived ease-of-use on technology acceptance have been confirmed by a large number of studies, and TAM is perceived as one of the most influential models for examining technology acceptance (Davis et al., 1989); (Robey, 1996).

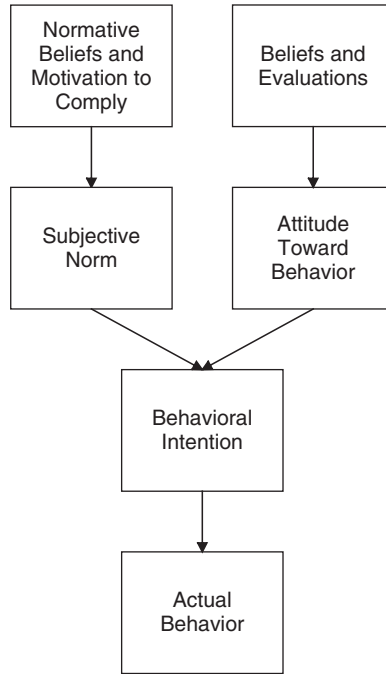


Fig. 4.3 Theory of reasoned action

There have been numerous reports on the acceptance failure of software engineering process improvement programs; we believe that the broader findings on acceptance of technology could shed some light on the acceptance of development methods (Riemenschneider et al., 2000). For example, many metrics techniques failed to become established, not because of inferior statistical models but because they were perceived by project staff as too tedious and time-consuming compared to the perceived benefits (Goldenson et al., 1999); (Niessink and Van Vliet, 1999); (Rosenberg and Hyatt, 1996). Measuring the quality of a method in terms of its acceptance by practitioners is a basis for improving the method and an opportunity for advancing the field.

An adoption-centric approach to the introduction and establishment of usability engineering within an organization needs a credible empirical base in order for practitioners to make decisions on the deployment of UE techniques. Such an empirical base can be incrementally constructed by conducting experiments in which the acceptance of methods is measured.

Software engineering projects in which UE techniques are deployed can be understood as experiments with UE techniques. The experiments consist of applying a UE technique U under the conditions S defined by the constraints of a project P . The acceptance of the UE technique can be measured in terms of perceived usefulness PU and perceived ease-of-use PEU as perceived by the project team T . After conducting such an experiment, we have some evidence to either support or

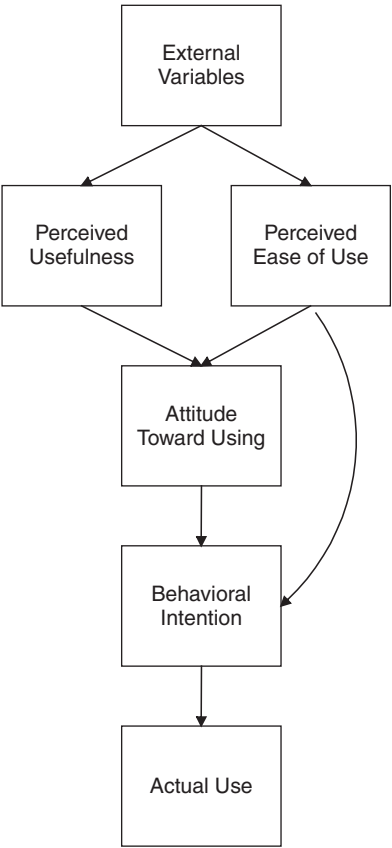


Fig. 4.4 Technology acceptance model

reject the deployment of the technique U under similar conditions S' in a future project P' .

4.4 Adoption-Centric Usability Engineering – Key Principles

In mature sciences, evidence can be strengthened by replicating the experiment. The most common type of replication is a repetition of the experiment under the same conditions S as in the initial experiment. The results of each replication are aggregated to get stronger conclusions than from the single observations. However, when we choose real industrial software development projects as a target for replication, this replication approach becomes problematic. Replication in the classical sense would mean to have the same project team with the same knowledge and experience, to develop the same system, with the same time constraints. In addi-

tion to the extreme methodical difficulty of replicating these conditions, the goal is economically infeasible in most industrial environments.

The overall goal of Adoption-Centric Usability Engineering (ACUE) is to facilitate the adoption of UE methods by software engineering practitioners and thereby improve their integration into existing software development methodologies and practices. ACUE is designed to support project teams in institutionalizing this abstract knowledge about UE methods and to help them transfer this knowledge into their development processes. UE methods are perceived as integrated into an existing software development process when they are adopted by the project team, i.e. when they are accepted and performed by the project team.

ACUE exploits empirical data collected in different projects to yield stronger evidence on how the method works in a certain context. The data form an empirical base to guide the improvement of UE methods and to facilitate the informed selection and deployment of UE methods in future projects.

Because the proposed framework is derived from more general theories of motivated human behavior, the framework can be expected to generalize to a large extent beyond the domain of tool usage to encompass method use. Building on the concept of acceptance as defined in the technology acceptance model (Davis et al., 1989), the acceptance of a UE method is defined as its perceived usefulness and ease-of-use from the perspective of project teams. The acceptance of UE methods by project teams provides a quality measure to assess and improve their adoption. This measure of acceptance needs to be qualified by additional data that characterize the constraints or context of the project.

The ACUE meta-model is depicted in Fig. 4.5. It uses qualitative and quantitative feedback on the acceptance of UE techniques provided by project teams and accumulates this feedback across project boundaries for controlling and improving the adoption of UE activities.

First, models need to be defined for measuring the acceptance of UE methods and for characterizing the projects in which the UE methods are to be deployed. Next, UE methods need to be captured to form a pool of methods for potential deployment in software development projects. For each captured method, its intended context of use needs to be defined. Sources for potential UE methods are the various existing UE methodologies.

In the next step, a subset of UE methods is selected from the method pool for deployment in a project. The appropriateness of a method for a project is assessed based on two criteria. According to the first criterion, the characteristics of the methods to be deployed in a project should match the characteristics of the project. According to the second criterion, candidate methods should have accumulated a high level of acceptance via acceptance assessments by project teams in previous projects.

In the next step, the project team is supported in mapping the selected methods onto the overall software development process and is guided in deploying the methods. After deployment of each selected method, the project team assesses the acceptance of the method.

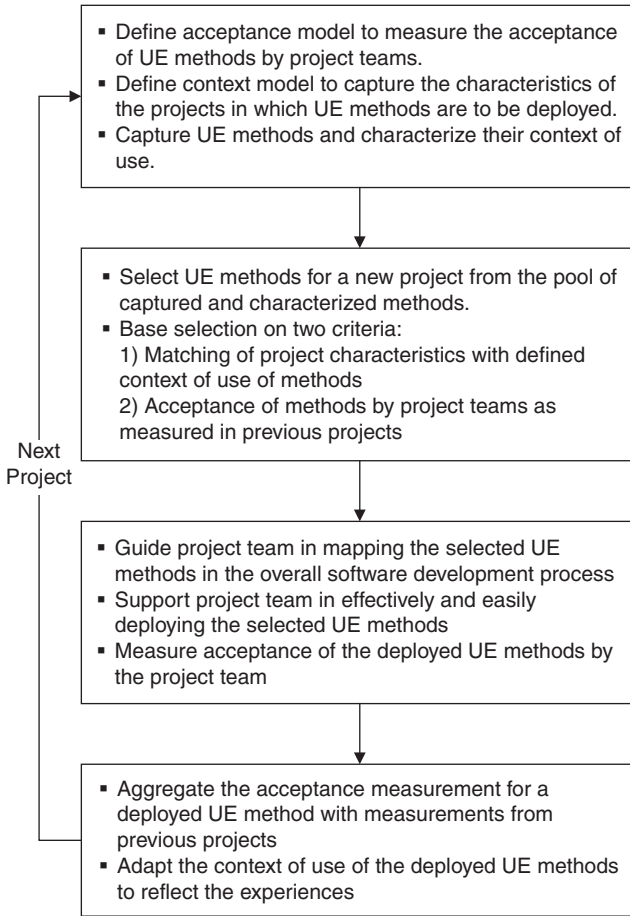


Fig. 4.5 Steps of the ACUE meta-model

Finally, the assessment results for each method are aggregated with the assessment results of previous projects. Also the context of use of each method is adapted to reflect new experiences with characteristics of projects which contribute to the acceptance or rejection of a method. This cycle is repeated for each new project, in order to accumulate empirical data about the acceptance of specific usability engineering methods and their context of use.

Adoption-centric Usability Engineering
Systematic Deployment, Assessment and Improvement
of Usability Methods in Software Engineering

Seffah, A.; Metzker, E.

2009, XIV, 222 p. 39 illus., Hardcover

ISBN: 978-1-84800-018-6