COST ESTIMATION IN AGILE DEVELOPMENT PROJECTS

Siobhan Keaveney, National University of Ireland, Galway, Ireland
Kieran Conboy, National University of Ireland, Galway, Ireland

Abstract

One of the key measures of the resilience of a project is its ability to reach completion on time and on budget, regardless of the turbulent and uncertain environment it may operate within. Cost estimation and tracking are therefore paramount when developing a system. Cost estimation has long been a difficult task in systems development, and although much research has focused on traditional methods, little is known about estimation in the agile method arena. This is ironic given that the reduction of cost and development time is the driving force behind the emergence of the agile method paradigm. This study investigates the applicability of current estimation techniques to more agile development approaches by focusing on four case studies of agile method use across different organisations. The study revealed that estimation inaccuracy was a less frequent occurrence for these companies. The frequency with which estimates are required on agile projects, typically at the beginning of each iteration, meant that the companies found estimation easier than when traditional approaches were used. The main estimation techniques used were expert knowledge and analogy to past projects. A number of recommendations can be drawn from the research: estimation models are not a necessary component of the process; fixed price budgets can prove beneficial for both developers and customers; and experience and past project data should be documented and used to aid the estimation of subsequent projects.

Keywords: *agile methods, project management, cost estimation, software development, systems development*

1 INTRODUCTION

The formation of the Agile Alliance in 2001 and the publication of the Agile Manifesto (Fowler and Highsmith, 2001) formally introduced agility to the field of Information Systems Development (ISD). Those involved sought to "restore credibility to the word *method*" (Fowler and Highsmith, 2001). The Agile Manifesto presented an industry-led vision for a profound shift in the ISD paradigm, through 12 principles. The Manifesto and its principles represent quite a popular initiative which actually complements the critique of formalised ISD methods over the past decade or so (Baskerville et al., 1992, Fitzgerald, 1994, Fitzgerald, 1996), and have been well received by practitioners and academics.

According to the Agile Manifesto (Fowler and Highsmith, 2001), agile methods stress values such as individuals and interactions over processes and tools; working software over comprehensive documentation; customer collaboration over contract negotiation; and responding to change over following a plan. The main methods that fall under the agile umbrella include XP, Scrum, Crystal, DSDM, FDD and ASD. The emergence of agile methods has been phenomenal during the past few years and is not showing any signs of ceasing (Abrahamsson et al., 2003). However, managing these agile projects has presented difficulties for many project managers who have been trained in the use of traditional development approaches (Highsmith, 2003).

The IS development process, regardless of the methodology adopted, requires effective management and planning. A large part of this planning is the creation of estimates at the beginning of a project so that resources can be appropriately allocated. Estimation techniques and models are available to simplify this activity but the prevalence of cost and schedule overruns on IS development projects indicates that accurate estimation remains somewhat elusive. The primary focus of this research is on agile methods and how the development process can be appropriately managed and planned in terms of estimating the resources for the project when agile methods are used.

One of the main principles of agile methods is to "welcome changing requirements" (Beck et al., 2001), however changing requirements are a major cause of software cost estimating problems (Jones, 2003). The objective of this research is therefore two-fold:

- To determine which of the traditional cost estimation techniques are used on agile projects.
- To examine how mangers of agile development projects adhere to the traditional critical success factors cited by the traditional cost estimation literature.

2 COST ESTIMATION IN TRADITIONAL SYSTEM DEVELOPMENT

Estimating the cost of an IS development project is one of the most crucial tasks for project managers (Keung et al., 2004) but despite this it continues to be a weak link in the IS development field (Agarwal et al., 2001). IS development projects have a long history of being delivered over time, over budget and failing to satisfy requirements (The Standish Group, 2001, Sassone, 1988, The Standish Group, 1995). The main factors that are typically estimated at the beginning of an IS development project are: cost, size, schedule, people resources, quality, effort, resources, maintenance costs, and complexity. Estimates are produced and used for a variety of purposes and a study by Lederer and

Prasad (1995) revealed the most common uses. These are: to schedule projects, to select proposed projects for implementation, to quote the charges to users for projects, to staff projects, to audit project success, to control or monitor project implementation, to evaluate project estimators, and to evaluate project developers.

2.1 Cost Estimation Techniques

Cost estimation tools, or model-based estimation techniques use data collected from past projects combined with mathematical formulae to estimate project cost. They usually require factors such as the system size as inputs into the model. The main model-based techniques include COCOMO, SLIM, RCA PRICE-S, SEER-SEM, and ESTIMACS. The major software cost and schedule estimation techniques can be grouped and classified as regression-based models, learning-oriented models, expert based approaches and finally composite-bayesian methods.

Most of the software estimation models that are available are based on some form of *regression technique* (Matson et al., 1994). Regression models have a mathematical foundation and are constructed by collecting data on completed projects and developing regression equations that characterise the relationships among the different variables (Fairley, 1992). Estimates are made by substituting the new project parameters into the mathematical model with the use of a large data set. Statistical regression models estimate software development effort as the dependent variable. Regression models however can be difficult to use in some cases, in particular if they do not satisfy a number of conditions that can either enhance or halt successful use (Finnie et al., 1997). These 4 conditions are discussed by Boehm and Sullivan (1999), and are based on experience from the use of regression-based models. They are: availability of a large dataset, no missing data items, no outliers, and the predictor variables are not correlated.The collection of approaches that fall under the heading of regression-models include ordinary least-squares regression (OLS), classification and regression trees (CART), stepwise analysis of variance for unbalanced data sets (stepwise ANOVA), combinations of CART with OLS regression and analogy, multiple linear regression, and stepwise regression.

Learning-oriented models attempt to automate the estimation process by building computerised models that can learn from previous estimation experience (Boehm et al., 2000). These models do not rely on assumptions and are capable of learning incrementally as new data are provided over time (Lee et al., 1998). Learning-oriented models cover a wide area and include techniques such as artificial intelligence approaches, artificial neural networks, case-based reasoning (Mukhopadhyay and Kekre, 1992), decision-tree learning, machine learning models, knowledge acquisition, fuzzy logic models and rule induction (Burgess and Lefley, 2001).cost estimation tools such as COCOMO, learning oriented and regression-based models; expert knowledge; and composite-Bayesian methods (Ruhe et al., 2003). The main model-based techniques include COCOMO, SLIM, RCA PRICE-S, SEER-SEM, and ESTIMACS. These estimation models produce an estimate of the cost, effort or duration of a project based on factors such as the size and desired functionality of the system.

The most widely used estimation approach was found by Briand et al. (1998) to be “comparison to similar, past projects based on personal memory”. These *expertise based approaches* are useful when no quantified, empirical data is available (Boehm et al., 2000), and they provide a practical, low-cost and highly useful process (Johnson et al., 2000). Estimation by analogy in its most basic form involves examining past projects and using the information retrieved as a guide estimate for the proposed project (Angelis et al., 2001, Jørgensen et al., 2003). The Checkpoint method is an example of an analogy-based approach to software estimation (Fairley, 1992). Rules of thumb can be derived from actual project data or a formalisation of expert opinion, either way they must make use of some form

of project data or information. Rules of thumb can be used to estimate productivity, quality or size (Hihn and Habib-agahi, 1991, Fairley, 1992). Expert judgement relies on the accumulated experiences of teams of experts in order to come up with project estimates (Peters and Pedrycz, 1999, Stamelos and Angelis, 2001). This technique is used where the estimation process is primarily based on "non-explicit, non-recoverable reasoning processes", or perception and intuition (Jørgensen, 2004b). Mukhopadhyay et al. (1992) highlight the weaknesses of using any human memory-based techniques, because past projects can be forgotten, details confused and important factors accidentally ignored. The very nature of expert judgement means that deriving an estimate is not a repeatable process (Mendes et al., 2002), however reports have proven it to be the dominant strategy in software development estimation (Jørgensen, 2004a, Höst and Wohlin, 1997, Moløkken and Jørgensen, 2003, Moløkken-Østvold et al., 2004).The Delphi technique and work breakdown structure (WBS) fall under the heading of expert judgement techniques. These help to reduce the likelihood of errors occurring in the estimation process (Boehm and Sullivan, 1999, Boehm et al., 2000). Other techniques in the expertise-based category include top-down and bottom-up estimation (Tausworthe, 1980), reasoning by analogy, formal reasoning by analogy, informal reasoning by analogy, and rules of thumb (Jones, 1996).A study by Briand et al. (1998) revealed that the most widely used estimation approach was "comparison to similar, past projects based on personal memory". Expertise based approaches come under much criticism for their reliance on human memory and the lack of repeatability of such memory-based approaches (Mukhopadhyay et al., 1992, Mendes et al., 2002), however reports have proven it to be the dominant strategy in IS development estimation (Jørgensen, 2004a, Höst and Wohlin, 1997, Moløkken and Jørgensen, 2003, Moløkken-Østvold et al., 2004).

The *Bayesian approach* is a semi-formal estimation process that combines the strengths of expertise-based methods and regression-based methods (Ferens, 1988). Bayesian analysis allows for the fact that the data required for use in most estimation techniques is typically of poor quality or incomplete. Expert judgement is incorporated in this approach to handle the missing data and provide a more robust estimation process (Boehm and Sullivan, 1999). Bayesian analysis has been used in many scientific disciplines and was used in the development of the COCOMO II model (Chulani et al., 1999, Boehm et al., 2000). Cost Estimation, Benchmarking and Risk Analysis (COBRA) is an example of a composite estimation model. It requires expert knowledge and a relatively small amount of quantitative data gathered from past projects in order to produce estimates of a project's cost and development effort, and also the quantitative risks associated with the project (Ruhe et al., 2003). COBRA has been shown by Briand et al. (1998) to be an inexpensive technique for producing cost estimation and risk models without the requiring an extensive dataset.

2.2 Causes of Inaccurate Estimates in Systems Development

There is a natural erroneous tendency associated with any form of estimation primarily because "an estimate is a probabilistic assessment of a future condition" and accuracy can therefore rarely be expected in the estimation process (Stamelos and Angelis, 2001). The causes of inaccurate estimates in IS development projects were grouped into 4 categories by Lederer and Prasad (1995), namely methodology, politics, user communication and management control.

Methodology refers to the estimation process adopted and includes the steps undertaken to produce the estimate and also the means of examining and reviewing the estimates relating to past projects. Over reliance on intuition and personal memory is a concern for project members trying to increase estimation accuracy (Lederer and Prasad, 1995). Estimation inaccuracy can also be caused from a lack of procedures and policies on how to deal with failures and avoid repeating mistakes by learning from past experiences (Ewusi-Mensah and Przasnyski, 1995).

Political forces at work within a project or company can often drive estimation inaccuracy. This is usually in the form of managerial pressure to stay within or meet the estimate (Block, 1983). The estimation process can be impacted negatively by these pressures resulting in time or cost constraints (Hihn and Habib-agahi, 1991). When estimates are produced simply in order to satisfy managers or customers it will inevitably lead to inaccuracy (Lederer and Prasad, 1991). According to Jørgensen and Moløkken (2003), it should be recognised that estimation is typically fraught with "tug of wars" and "political games", and thus high estimation accuracy is usually not the only goal of the actors involved. Chapman and Ward (2002) refer to a "conspiracy of optimism" whereby political pressures from within the organisation can lead to unrealistic estimates or reluctance to report the actual outcome. Moløkken and Jørgensen (2003) suggest that software managers may over-report causes of inaccuracy that lie outside their responsibility, such as customer-related causes. Project managers therefore have to be aware of the implications that political factors can have on IS development estimation (Winklhofer, 2002).

User communication refers to the factors relating to the customers and their changing requirements throughout a system's life-cycle. This is usually the most prominent factor in causing project estimates to be inaccurate (Jørgensen, 2003). Incomplete or unclear requirements specification at the beginning of an IS development project is typically due to the fact that customers have to determine what their requirements are and they are usually unaware what the current state of the art is, or what the competition is doing (Orr, 2004, Stamelos and Angelis, 2001). This leads to difficulty in producing a complete set of requirements and thus estimation inaccuracy is inevitable (Harker et al., 1993, Thayer, 1987).

Problems caused by *Management control* include management reviews, and comparison between estimates and actuals. When management fails to participate in the preparation of the estimate, and does not monitor the accuracy of the estimate, this is believed to contribute to the estimate being inaccurate. Inaccuracy also occurs when management does not refer to the estimate when conducting performance reviews of estimators and other project personnel (Lederer and Prasad, 1995).

2.3 Cost Estimation Critical Success Factors

There are a number of critical success factors (CSFs) that can help to ease the estimation process. A review of the literature resulted in the compilation of a list of important guidelines which should help to improve project managers' estimation success rates. These are:

Involve developers, customers, managers in estimation: In order for an estimate to be accepted and adhered to, it must consider and include all members of the development team and in particular the project manager (Agarwal et al., 2001). It also must be communicated clearly to the project team before the development begins. Research has shown that if the estimator is somebody who will be involved in the development, the estimation accuracy is likely to be higher than if an estimate is produced by a senior executive or a staff member from a different department (Jurison, 1999, Lederer and Prasad, 1992).

Use estimate to evaluate project personnel. Management can use the estimate to review project personnel, either during the project or upon project completion. In Lederer and Prasad's (1992) management guidelines for better cost estimating, they claim that using the estimate to evaluate project personnel is an important factor for completing a project within its estimate. Completing the project within the estimate usually results in rewards for those involved, such as pay increases, bonuses, and promotions.

Finalise requirements before estimation. In order for the project estimate to attain any degree of accuracy, it is important that the system requirements are defined and documented prior to the production of an estimate (Lanza, 2002, Lederer and Prasad, 1992). This will usually require the use of a structured development methodology (Golden et al., 1981), although updating the estimates to acknowledge the evolving nature of IS is also an option (Fairley, 1992).

Make the early estimating effort simpler instead of more complex: Overly complex techniques for estimation at an early stage of a development project are failing to acknowledge the inherent volatility of IS projects and are therefore highly likely to lead to inaccurate cost and effort estimates (Maxwell et al., 1999). Briand et al. (2000) found that combining modelling techniques failed to produce increased accuracy in project predictions. Simple formulae with management reserve built in for unanticipated problems can help to improve project estimation success rates (Murphy, 2001, Jurison, 1999). Strike, et al. (2001) claim that it is fundamental that the estimated cost of a particular software project is ascertained as early in the development cycle as possible as it enables project managers to make critical business decisions in a timely manner. Knowing the cost estimate when the project is almost completed, or even halfway through the development, is of diminishing benefit because as the project progresses, more and more is invested, and the harder it becomes to abandon the runaway project (Bossavit, 2003, Mahaney and Lederer, 1999).

3 COST ESTIMATION IN AGILE SOFTWARE DEVELOPMENT

In terms of the agile development, the estimation process is an iterative one whereby the user stories in XP represent pieces of functionality to be estimated and this is done every 2 weeks. An overall expected time for each of these stories is estimated by the developers, and the customers then prioritise the stories based on these initial estimates and on the business value of each one (Lovaasen, 2001). According to Highsmith (2003), the nature of agile methods often results in fixed budgets and a fixed schedule, and it is the scope of the project that remains flexible throughout. Ceschi, et al. (2005) on the other hand report that companies using agile methods usually lean towards "flexible contracts instead of fixed ones that predefine functionalities, price, and time".

Although IS projects are typically characterised by changes in scope and requirements, the impact of these changes can vary phenomenally depending on the time at which the change is introduced (Ibbs et al., 2001, Pressman, 1997). Agile methods aim to reduce the cost of changes throughout the development of a system, but not necessarily to reduce the occurrence of changes (Highsmith and Cockburn, 2001). The cost of change rises phenomenally throughout traditional development while in XP projects as time goes on, the impact of change levels off (Neill, 2003).

The techniques used to estimate agile development projects have typically been expertise-based, where the developers look to past projects or iterations, and draw on their own experiences to produce estimates for the stories (Ceschi et al., 2005). A study by (Ceschi et al., 2005) claimed that none of the companies had used COCOMO and that 40% used function points estimation on their agile projects. These results however are based on only 10 companies and so do not represent generalisable data, although from the available literature there does seem to be an inclination toward the reliance on expertise-based estimation approaches (Lippert et al., 2003, Elssamadisy and Schalliol, 2002, Grossman et al., 2004).

Reports of inaccurate estimates have not been as widespread in the literature on agile projects. This may be due to the feeling that detailed project management only needs to look at the following

iteration and as such, more reliable estimates can be produced (Williams, 2003, Taber and Fowler, 2000). Ceschi, et al. (2005) see the reliance on expertise-based approaches to estimation as a problem because of the particular uniqueness of IS projects that use agile methods. This coupled with new concepts such as pair-programming and test-driven development makes estimation based on past projects extremely difficult (Lippert et al., 2003). The assignment of control to the developers in estimating their own tasks can cause inaccuracies if the developer is pressured into underestimating their tasks in order to indulge managers or customers. This can also lead to reluctance in exposing what may appear to be poor estimating skills or even poor development capabilities (Elssamadisy and Schalliol, 2002).

User communication factors seem to pose less of a threat to estimation because of the extent of the customer's involvement, particularly when the developers are producing the estimates (Williams, 2003). Paulk (2002) however claims that customers pose a serious threat to successful agile development if they are unwilling to maintain a close relationship with the development team. Estimation inaccuracy would therefore increase if customers were not available to clarify and elaborate on confusing stories.

In terms of management control, each developer takes responsibility and ownership for the stories that they estimate and so management involvement is less of an issue as in traditional development (Schalliol, 2001).

Estimation of the user stories in agile methods is performed by the developers who are then responsible for working on the particular tasks that they have estimated (Beck, 1999). Customers are involved in the estimation process to the extent that if the developer has difficulty in estimating a user story they can discuss it with the customer and try to break the story down further (Williams, 2003). Management involvement in agile projects tends to be less intrusive than on traditional projects and their involvement is at a higher level and this enables them to oversee the estimation process from one iteration to the next (Abrahamsson, 2003). Evaluation of team members based on their ability to meet the estimates is slightly less appropriate for agile projects because it is the developers themselves who estimate their own tasks (Schalliol, 2001).

Agile methods "welcome changing requirements, even late in development" (Beck et al., 2001), however in terms of estimation, the requirements are finalised to a certain extent at the start of each iteration and so developers can estimate safe in the knowledge that the scope for the iteration has been agreed (Taber and Fowler, 2000). Simplifying the early estimation effort is done during the initial release planning sessions where the estimates produced at this early stage for the entire project are typically at a high level (Lindstrom and Jeffries, 2004). The frequency with which estimation is performed, typically at the beginning of every iteration, leads to progressively more accurate estimation by the developers as they become more and more skilled at estimating the tasks (Abrahamsson, 2003, Levy, 2003). Cockburn (2002) recommends that cost-sensitive projects should use serial development where possible, while projects sensitive to shifting requirements benefit more from concurrent development. He then goes on to state that agile project teams almost always use concurrent development, which implies that agile methods are not wholly suitable for projects that are cost-sensitive, although realistically it is rare to find a project that is not.

4 RESEARCH METHOD

Estimation in agile methods is a concept that combines an important and commonly researched project management issue with the very recent topic of agile development methods, where little prior research exists. The research method that has been chosen for this study is that of a qualitative case study. Having examined in detail the alternative approaches, the case study approach emerged as the most suitable means of conducting an investigation into the practice of estimation in agile methods.

Like most aspects of IS however, it is not always easy to locate a research approach that is relevant and applicable in all situations (Galliers, 1992). Yin (2003) defines a case study as "an empirical enquiry that investigates a contemporary phenomenon within its real-life context, especially when the boundaries between phenomenon and context are not clearly evident". Case studies examine phenomena in their natural settings in order to gain an in-depth understanding of the dynamics of both the phenomenon and the context in which it is situated (Eisenhardt, 1989, Pinsonneault and Kraemer, 1993).

Four case studies were conducted which presented an in-depth investigation into estimation practices followed by organisations using agile methods. These case studies involved semi-structured interviews which addressed the main aims of the research. The primary research produced a valuable quantity of data. The four companies, JourneyTechnology, Shinesoft, Mountain, and Software Labs are described below and discussed in terms of their agile estimation processes (Note that pseudonyms are used to protect the identity of the organisations involved). A comparative profile of the companies is shown in Table 1.

	Year of company set up	**No of employees**	**No of concurrent projects**	**Typical project length**	**Team size**
JourneyTechnology	1999	70	15-20	2-3 years	2-5
Shinesoft	1995	12-15	4-5	4-6 months	12-15
Mountain Ltd.	2002	13	1-5	1-2 months	1-10
Software Labs	1971	500	2-3	4-8 months	6-7

Table 1: Case Study Company Profiles

5 DISCUSSION

5.1 Use of Traditional Cost Estimation Techniques

JourneyTechnology would appear to be the most formal in terms of their estimation process whereby they have an estimating template in Microsoft Excel that is used for their 2-point estimation. This is subject to management approval and what they feel to be an appropriate contingency factor for risk. Rules of thumb are not used because they feel that the uniqueness of each of their projects would render this a futile exercise.

Shinesoft have a formal estimation process in place that had been designed initially for use on traditional development projects. The projects that are developed using agile approaches still use this

process but a more informal version of it. Their relative-size table tracks developer productivity and this is used to produce estimates for future projects.

Mountain rely solely on the experiences and expertise of their developers for estimating projects. They base their estimations to some extent on past projects but this is not documented and so relies on the memory of the developers. They also use simple rules of thumb but nothing that is documented to any formal degree. This may be due to the fact that they are a young company with a small and very closely knit team and formal techniques may be less appropriate or necessary.

Software Labs use expert judgement to guide their project estimates. The knowledge and experience that the development team have enables them to produce relatively accurate estimates without the use of models or formulae. In this respect their estimation process is quite informal with no data collected or stored, and because they use an agile approach the estimation is performed on a fortnightly basis.

5.2 Causes of Inaccurate Estimates

The causes of inaccurate estimates in IS development projects as discussed by Lederer and Prasad (1995) revealed that each factor: methodology; politics; user communication; and management control; were of great concern to both project estimators and implementers. These results do not correlate exactly with the cases in this study whereby JourneyTechnology were the only company to acknowledge the potential of political factors where pressures from customers or managers may result in lower estimates than would be realistically expected. Management control factors were not a cause of inaccuracies in any of the estimates produced by the companies. All of the companies did however experience user communication difficulties at some stage or another and certainly recognised this as a big potential threat to accurate estimates.

JourneyTechnology find that when inaccuracies do occur, it is typically due to some lack of understanding between the customers and developers regarding the requirements. It can also be due to a lack of technical expertise in a particular area which would prevent the accurate estimation of certain tasks. Shinesoft seem to be the most confident in their estimation abilities. Typically the estimates produced are relatively on target and if not, the discrepancy is usually negligible. They have found the major potential threats to accurate estimates to have been the introduction of new people, new technologies and too much feedback from their customers. Mountain find that change requests from customers and lack of estimation expertise can cause problems on some projects, particularly if a new blend of the development language is being used. Software Labs have found that their estimates are typically accurate to within 10% of the actual figures, however they feel that their inaccuracies may be due to their lack of formality in the estimation process or even poor tracking of actual project data which would make the estimate appear to be inaccurate.

5.3 Presence of Cost Estimation CSFs

Ensuring accurate, reliable and realistic estimates provides an obligatory challenge for the companies in this study. The estimates act as a focal point and encourage the team members to take responsibility for their tasks. This paper previously provided a set of guidelines or CSFs, compiled from the literature on estimation in traditional development. This set of guidelines or CSFs include involving developers, managers and customers in the estimation process, using the estimate to evaluate project personnel, finalising requirements before estimation takes place, and keeping the early estimation effort as simple as possible.

Software Labs was the only project that did not involve managers in the estimation process. This may be due to the large size of the company in comparison to the others in the study and their leaning towards a highly informal estimation approach. Project managers and developers are involved by all of the companies, although JourneyTechnology tend to involve individual developers only when the team leader cannot accurately estimate a specific task. Customers are involved to a certain extent, but this typically depends on the situation at hand. JourneyTechnology, Mountain and Software Labs involve their customers in the requirements stage where they are estimating for the iteration and need clarification or elaboration on the requirements. Shinesoft also involve the customer at the early stages to determine the requirements but they have found that their main contribution is to impose a deadline for delivery. Software Labs seem to have the most actively involved customer and this is probably because it is an internal customer who was able to locate on-site with relative ease.

JourneyTechnology and Shinesoft both store data relating to projects so that they can use this to inform future estimation efforts. They try to keep the early estimates as high-level as they can and they build in a contingency factor for risk and other unknowns. Mountain and Software Labs also keep their early estimates simple and they continue in this vein throughout the development. Mountain place a heavy emphasis on the experience and expertise of the developers and when they encounter an unknown in terms of the estimation of certain tasks they will learn from this and use the experience to their advantage in the future. Software Labs have rarely encountered unknowns but on the occasions that they have, they did a spike to enable them to estimate the task accurately. Requirements finalisation is a practice that does not fit entirely into the agile philosophy (Beck et al., 2001) and so this was not something that the companies did. What they tended to do was finalise the requirements at the beginning of each iteration which loosely locked the requirements while also acknowledging the inevitability of changes and in some cases catering for these.

Evaluation of project personnel as recommended by Lederer and Prasad (1992), was not performed to any great degree by the companies. While both JourneyTechnology and Shinesoft record productivity metrics and skill levels of developers, neither would actually rate individuals based on their ability to meet the estimates. Software Labs acknowledge the skill-set of the individuals who will be performing the tasks but do not evaluate them based on their ability to meet the estimate. Mountain do not formally evaluate team members to any extent but if an individual failed to meet the estimate they would informally investigate and discuss the possible reasons for this. Lederer and Prasad (1992) claim that the use of personal memory in the estimation process correlates to projects overrunning their estimates. Both JourneyTechnology and Shinesoft seem to acknowledge the potential for human error and opt to record and store project data; however Mountain and Software Labs rely almost entirely on the memory of the team members and project managers.

6 CONCLUSIONS

The emergence of agile methods in the IS field has presented many opportunities and challenges for researchers and practitioners alike. In particular, project management serves as one area that yearns attention in an agile context (Cao, 2004). A major project management activity involves planning and estimating, and on IS projects developed using traditional approaches, the success of estimation has proved controversial to say the least (The Standish Group, 2001). Reports of inaccurate estimates have typically cited customer change requests and unclear requirements as the main causes (Jørgensen, 2003), despite supposed requirements finalisation in the early stages. The agile philosophy of welcoming changing requirements seems on paper to be catastrophic to the estimation endeavours of project managers.

The main estimation techniques used across the four projects were analogy and expert knowledge with varying degrees of formality and structure between the companies. In some cases project data was

stored and in others it was simply assigned to the developers own memories. Estimation models, despite their popularity in the literature were not used by the companies and for the most part were not even recognised.

The estimates produced for agile projects were found by the companies to be easier to construct due to the frequency with which estimates are required. The iterations typically lasted 2 weeks and estimates were produced at the beginning of each iteration. This not only helped to keep a high degree of accuracy but also honed the estimation skills of the team members and developers involved. It is worth noting that estimation inaccuracy was not a great problem for any of the companies and where it was, they typically saw it as an opportunity to learn and inform their future estimation activities.

Fixed price projects where a budget is agreed at the beginning seemed to be the most common project type. In some cases the schedule was movable and in others it was the functionality that could be revised. Typically when the cost is determined, a number of developers can be assigned and the delivery date calculated from this. On the other hand, if the schedule is set by the customer then the cost can be calculated from the number of people available to work on the project. Either way this enables the project to be run in a manner that delivers increasingly more features as time progresses until the scheduled delivery date has been reached.

In conclusion, research on estimation has been conducted for decades with immense quantities of models and tools produced. This study has looked at the estimation process in the emerging field of agile development, and examined the causes of inaccurate estimates and steps to improve the process. From the four case studies, a number of recommendations can be summarised as follows: estimation models are not a necessary component of the process; fixed price budgets may be the best option for both developers and customers; and finally the most critical success factors for agile cost estimation is that experience and past project data should be documented and used to aid the estimation of subsequent projects.

References

ABRAHAMSSON, P. (2003) Extreme Programming: First Results from a Controlled Case Study. *Proceedings of the 29th Euromicro Conference, 2003.*

ABRAHAMSSON, P., WARSTA, J., SIPONEN, M. T. & RONKAINEN, J. (2003) New Directions on Agile Methods: A Comparative Analysis. *IEEE***,** 244-254.

AGARWAL, R., KUMAR, M., YOGESH, MALLICK, S., BHARADWAJ, R. M. & ANANTWAR, D. (2001) Estimating Software Projects. *ACM SIGSOFT Software Engineering Notes,* 26**,** 60-67.

ANGELIS, L., STAMELOS, I. & MORISIO, M. (2001) Building a Software Cost Estimation Model Based on Categorical Data. *Proceedings of the 7th International Software Metrics Symposium.*

BASKERVILLE, R., TRAVIS, J. & TRUEX, D. (1992) Systems without method: the impact of new technologies on information systems development projects. IN KENDALL, K., DEGROSS, J. & LYYTINEN, K. (Eds.) *The Impact of Computer Supported Technologies on Information Systems Development.* North Holland, Elsevier Science Publishers.

BECK, K. (1999) Embracing Change with Extreme Programming. *Computer,* 32**,** 70-77.

BECK, K., BEEDLE, M., VAN BENNEKUM, A., COCKBURN, A., CUNNINGHAM, W., FOWLER, M., HIGHSMITH, J., HUNT, A., GRENNING, J., MELLOR, S., JEFFRIES, R., KERN, J., MARICK, B., MARTIN, R. C., SCHWABER, K., SUTHERLAND, J. & THOMAS, D. (2001) The Agile Manifesto.

BLOCK, R. (1983) *The Politics of Projects,* Englewood Cliffs, New Jersey, Prentice Hall.

BOEHM, B. W., ABTS, C. & CHULANI, S. (2000) Software Development Cost Estimation Approaches: A Survey. USC-CSE.

BOEHM, B. W. & SULLIVAN, K. J. (1999) Software Economics: Status and Prospects. *Information and Software Technology,* 41**,** 937-946.
BOSSAVIT, L. (2003) Project Management, The Movie. *Cutter IT Journal,* 16**,** 18-23.
BRIAND, L. C., EL EMAM, K. & BOMARIUS, F. (1998) COBRA: A Hybrid Method for Software Cost Estimation, Benchmarking, and Risk Assessment. *Proceedings of the 20th International Conference on Software Engineering.* Kyoto, Japan.
BRIAND, L. C., LANGLEY, T. & WIECZOREK, I. (2000) A Replicated Assessment and Comparison of Common Software Cost Modeling Techniques. *Proceedings of the 22nd International Conference on Software Engineering.* Limerick, Ireland.
BURGESS, C. J. & LEFLEY, M. (2001) Can Genetic Programming Improve Software Effort Estimation? A Comparative Evaluation. *Information and Software Technology,* 43**,** 863-873.
CAO, L. (2004) Modeling Dynamics of Agile Software Development. *Companion to the 19th Annual ACM SIGPLAN Conference on Object-Oriented Programming Systems, Languages, and Applications.* Vancouver, Canada.
CESCHI, M., SILLITTI, A., SUCCI, G. & DE PANFILIS, S. (2005) Project Management in Plan-Based and Agile Companies. *IEEE Software,* 22**,** 21-25.
CHAPMAN, C. & WARD, S. (2002) *Managing Project Risk and Uncertainty: A Constructively Simple Approach to Decision Making,* Chichester, UK, John Wiley & Sons.
CHULANI, S., BOEHM, B. W. & STEECE, B. M. (1999) Bayesian Analysis of Empirical Software Engineering Cost Models. *IEEE Transactions on Software Engineering,* 25**,** 573-583.
COCKBURN, A. (2002) Learning From Agile Software Development – Part One. *CrossTalk, The Journal of Defense Software Engineering***,** 10-14.
EISENHARDT, K. M. (1989) Building Theories from Case Study Research. *Academy of Management Review,* 14**,** 532-550.
ELSSAMADISY, A. & SCHALLIOL, G. (2002) Recognizing and Responding to "Bad Smells" in Extreme Programming. *Proceedings of the 24th International Conference on Software Engineering, 2002.* Orlando, Florida.
EWUSI-MENSAH, K. & PRZASNYSKI, Z. H. (1995) Learning from Abandoned Information Systems Development Projects. *Journal of Information Technology,* 10**,** 3-14.
FAIRLEY, R. E. (1992) Recent Advances in Software Estimation Techniques. *Proceedings of the 14th International Conference on Software Engineering.* Melbourne, Australia.
FERENS, D. V. (1988) Software Size Estimation Techniques. *Proceedings of the IEEE 1988 National Aerospace and Electronics Conference.*
FINNIE, G. R., WITTIG, G. E. & DESHARNAIS, J.-M. (1997) A Comparison of Software Effort Estimation Techniques: Using Function Points with Neural Networks, Case-Based Reasoning and Regression Models. *Journal of Systems and Software,* 39**,** 281-289.
FITZGERALD, B. (1994) The systems development dilemma: whether to adopt formalised systems development methodologies or not? IN BAETS, W. (Ed.) *Proceedings of the Second European Conference on Information Systems.* Holland, Nijenrode University Press.
FITZGERALD, B. (1996) Formalised systems development methodologies: a critical perspective. *Information Systems Journal,* 6**,** 3-23.
FOWLER, M. & HIGHSMITH, J. (2001) The Agile Manifesto. *Software Development,* August.
GALLIERS, R. D. (1992) Choosing Information Systems Research Approaches. IN GALLIERS, R. D. (Ed.) *Information Systems Research: Issues, Methods and Practical Guidelines.* Oxford, England, Blackwell Scientific Publications.
GOLDEN, J. R., MUELLER, J. R. & ANSELM, B. (1981) Software Cost Estimating: Craft or Witchcraft. *ACM SIGMIS Database,* 12**,** 12-14.
GROSSMAN, F., BERGIN, J., LEIP, D., MERRITT, S. M. & GOTEL, O. (2004) One XP Experience: Introducing Agile (XP) Software Development into a Culture that is Willing but not Ready. *Proceedings of the 2004 Conference of the Centre for Advanced Studies on Collaborative Research.*

HARKER, S. D. P., EASON, K. D. & DOBSON, J. E. (1993) The Change and Evolution of Requirements as a Challenge to the Practice of Software Engineering. *Proceedings of IEEE International Symposium on Requirements Engineering*.
HIGHSMITH, J. (2003) Agile Project Management: Principles and Tools. *Cutter Consortium,* 4**,** 1-37.
HIGHSMITH, J. & COCKBURN, A. (2001) Agile Software Development: The Business of Innovation. *Computer,* 34**,** 120-122.
HIHN, J. & HABIB-AGAHI, H. (1991) Cost Estimation of Software Intensive Projects: A Survey of Current Practices. *Proceedings of the 13th International Conference on Software Engineering*. Austin, Texas.
HÖST, M. & WOHLIN, C. (1997) A Subjective Effort Estimation Experiment. *Information and Software Technology,* 39**,** 755-762.
IBBS, C. W., WONG, C. K. & KWAK, Y. H. (2001) Project Change Management System. *Journal of Management in Engineering,* 17**,** 159-165.
JOHNSON, P. M., MOORE, C. A., DANE, J. A. & BREWER, R. S. (2000) Empirically Guided Software Effort Guesstimation. *IEEE Software,* 17**,** 51-56.
JONES, C. (1996) By Popular Demand: Software Estimating Rules of Thumb. *Computer,* 29**,** 116-118.
JONES, C. (2003) Why Flawed Software Projects are Not Cancelled in Time. *Cutter IT Journal,* 16**,** 12-17.
JØRGENSEN, M. (2003) How Much Does a Vacation Cost? or What is a Software Cost Estimate? *ACM SIGSOFT Software Engineering Notes,* 28**,** 1-4.
JØRGENSEN, M. (2004a) A Review of Studies on Expert Estimation of Software Development Effort. *Journal of Systems and Software,* 70**,** 37-60.
JØRGENSEN, M. (2004b) Top-Down and Bottom-Up Expert Estimation of Software Development Effort. *Information and Software Technology,* 46**,** 3-16.
JØRGENSEN, M., INDAHL, U. & SJØBERG, D. (2003) Software Effort Estimation by Analogy and "Regression Toward the Mean". *Journal of Systems and Software,* 68**,** 253-262.
JØRGENSEN, M. & MOLØKKEN, K. (2003) A Preliminary Checklist for Software Cost Management. *Proceedings of the 3rd International Conference on Quality Software.*
JURISON, J. (1999) Software Project Management: The Manager's View. *Communications of the AIS,* 2**,** 1-50.
KEUNG, J., JEFFERY, R. & KITCHENHAM, B. (2004) The Challenge of Introducing a New Software Cost Estimation Technology into a Small Software Organisation. *Proceedings of the 2004 Australian Software Engineering Conference.* Australia.
LANZA, R. B. (2002) Is Your Software Bigger than a Breadbox? The Hows and Whys of Software Estimation Tools. *Information Strategy: The Executive's Journal,* 18**,** 17-25.
LEDERER, A. L. & PRASAD, J. (1991) The Validation of a Political Model of Information Systems Development Cost Estimating. *Proceedings of the 1991 conference on SIGCPR.*
LEDERER, A. L. & PRASAD, J. (1992) Nine Management Guidelines for Better Cost Estimating. *Communications of the ACM,* 35**,** 51-59.
LEDERER, A. L. & PRASAD, J. (1995) Perceptual Congruence and Information Systems Cost Estimating. *1995 ACM SIGCPR Conference on Supporting Teams, Groups, and Learning Inside and Outside the IS Function Reinventing IS.* Nashville, Tennessee.
LEE, A., HUNG CHENG, C. & BALAKRISHNAN, J. (1998) Software Development Cost Estimation: Integrating Neural Network with Cluster Analysis. *Information & Management,* 34**,** 1-9.
LEVY, J. V. (2003) If Extreme Programming is Good Management, What Were We Doing Before? *EDN,* 48**,** 81-82, 84.
LINDSTROM, L. & JEFFRIES, R. (2004) Extreme Programming and Agile Software Development Methodologies. *Information Systems Management,* 21**,** 41-52.
LIPPERT, M., BECKER-PECHAU, P., BREITLING, H., KOCH, J., KORNSTÄDT, A., ROOCK, S., SCHMOLITZKY, A., WOLF, H. & ZÜLLIGHOVEN, H. (2003) Developing Complex Projects using XP with Extensions. *Computer,* 36**,** 67-73.

LOVAASEN, G. (2001) Brokering with eXtreme Programming. *XP Universe 2001*. Raleigh, North Carolina.
MAHANEY, R. C. & LEDERER, A. L. (1999) Runaway Information Systems Projects and Escalating Commitment. *Proceedings of the 1999 ACM SIGCPR Conference on Computer Personnel Research.* New Orleans, Louisiana.
MATSON, J. E., BARRETT, B. E. & MELLICHAMP, J. M. (1994) Software Development Cost Estimation Using Function Points. *IEEE Transactions on Software Engineering,* 20**,** 275-287.
MAXWELL, K. D., VAN WASSENHOVE, L. & DUTTA, S. (1999) Performance Evaluation of General and Company Specific Models in Software Development Effort Estimation. *Management Science,* 45**,** 787-803.
MENDES, E., WATSON, I., TRIGGS, C., MOSLEY, N. & COUNSELL, S. (2002) A Comparison of Development Effort Estimation Techniques for Web Hypermedia Applications. *Proceedings of the 8th IEEE Symposium on Software Metrics.*
MOLØKKEN-ØSTVOLD, K., JØRGENSEN, M., TANILKAN, S. S., GALLIS, H., LIEN, A. C. & HOVE, S. E. (2004) A Survey on Software Estimation in the Norwegian Industry. *Proceedings of the 10th International Symposium on Software Metrics.*
MOLØKKEN, K. & JØRGENSEN, M. (2003) A Review of Software Surveys on Software Effort Estimation. *Proceedings of the 2003 International Symposium on Empirical Software Engineering.*
MUKHOPADHYAY, T. & KEKRE, S. (1992) Software Effort Models for Early Estimation of Process Control Applications. *IEEE Transactions on Software Engineering,* 18**,** 915-924.
MUKHOPADHYAY, T., VICINANZA, S. S. & PRIETULA, M. J. (1992) Examining the Feasibility of a Case-Based Reasoning Model for Software Effort Estimation. *MIS Quarterly,* 16**,** 155-171.
MURPHY, L. (2001) Using Software Project "Should-Cost" Models. *Transactions of AACE International***,** 4.1-4.3.
NEILL, C. J. (2003) The Extreme Programming Bandwagon: Revolution or Just Revolting? *IT Professional,* 5**,** 62-64.
ORR, K. (2004) Agile Requirements: Opportunity or Oxymoron? *IEEE Software,* 21**,** 71-73.
PAULK, M. C. (2002) Agile Methodologies and Process Discipline. *CrossTalk, The Journal of Defense Software Engineering***,** 15-18.
PETERS, J. F. & PEDRYCZ, W. (1999) *Software Engineering: An Engineering Approach*, John Wiley & Sons, Inc.
PINSONNEAULT, A. & KRAEMER, K. L. (1993) Survey Research Methodology in Management Information Systems: An Assessment. *Journal of Management Information Systems,* 10**,** 75-105.
PRESSMAN, R. S. (1997) *Software Engineering: A Practitioner's Approach*, McGraw-Hill.
RUHE, M., JEFFERY, R. & WIECZOREK, I. (2003) Cost Estimating for Web Applications. *Proceedings of the 25th International Conference on Software Engineering.* Portland, Oregon.
SASSONE, P. G. (1988) Cost Benefit Analysis of Information Systems: A Survey of Methodologies. *ACM SIGOIS Bulletin,* 9**,** 126-133.
SCHALLIOL, G. (2001) Challenges for Analysts on a Large XP Project. *XP Universe 2001*. Raleigh, North Carolina.
STAMELOS, I. & ANGELIS, L. (2001) Managing Uncertainty in Project Portfolio Cost Estimation. *Information and Software Technology,* 43**,** 759-768.
STRIKE, K., EL EMAM, K. & MADHAVJI, N. (2001) Software Cost Estimation with Incomplete Data. *IEEE Transactions on Software Engineering,* 27**,** 890-908.
TABER, C. & FOWLER, M. (2000) An Iteration in the Life of an XP Project. *Cutter IT Journal,* 13**,** 13-21.
TAUSWORTHE, R. C. (1980) The Work Breakdown Structure in Software Project Management. *The Journal of Systems and Software,* 1**,** 181-186.
THAYER, R. H. (1987) Software Engineering Project Management: A Top-Down View. IN THAYER, R. H. (Ed.) *Software Engineering Project Management*. Los Alamitos, IEEE Computer Society Press.
THE STANDISH GROUP (1995) The CHAOS Report (1995). *The Standish Group International Inc.* http://www.standishgroup.com/chaos.html**,** 1-9.

THE STANDISH GROUP (2001) Extreme Chaos. *The Standish Group International Inc.* http://www.standishgroup.com/chaos.html**,** 1-12.
WILLIAMS, L. (2003) The XP Programmer: The Few-Minutes Programmer. *IEEE Software,* 20**,** 16-20.
WINKLHOFER, H. (2002) Information Systems Project Management during Organizational Change. *Engineering Management Journal,* 14**,** 33-37.
YIN, R. K. (2003) *Case Study Research: Design and Methods,* Thousand Oaks, California, Sage Publications, Inc.