Using Measurements to Support Real-Option Thinking in Agile Software Development

Zornitza Racheva
University of Twente
Drienerlolaan 5, Enschede 7500,
The Netherlands
++31 53 489 4344
z.racheva@utwente.nl

Maya Daneva
University of Twente
Drienerlolaan 5, Enschede 7500,
The Netherlands
++31 53 489 2889
m.daneva@utwente.nl

ABSTRACT
This position paper applies real-option-theory perspective to agile software development. We complement real-option thinking with the use of measurements to support midcourse decision-making from the viewpoint of the client. Our position is motivated by using empirical data gathered from secondary sources.

Categories and Subject Descriptors
D.2.8 [**Software Engineering**]: Management – *cost estimation*, D.2.8 [**Software Engineering**]: Requirements/Specifications – *methodologies (e.g. agile).*

General Terms
Measurement, Economics.

Keywords
Agile Software Development, Real Options, Measurements, Decision Making.

1. INTRODUCTION
Software metrics exist to provide project decision makers, as early as the stage of requirements, with key information to understand or control aspects of the software product or process. In practice, however, most of the metrics were not designed with the agile project context in mind and therefore only partially address this context. Moreover, the unique role the clients have in agile software processes is not reflected or supported by the existing metrics. A fundamental characteristic of agile development is that the client is responsible for prioritizing the requirements and thus in charge for the outcome of the project. In this paper, we discuss the joint use of real-options-thinking and measurements in agile development of software. This effort is part of a larger research initiative with the objective to provide midcourse decision support vehicles for the client, for example, when prioritizing requirements. Our position is that viewing decisions as options helps decision-makers to become aware of alternatives, to compare them and, consequently, to make the decision that insures best fit between the business goals and the current software functionality. We propose to use metrics of the product and of the process for this purpose. In what follows, Section 2 presents related work on measurement in agile software development. Section 3 summarizes the real-option perspective, Section 4 discusses its application in agile projects from clients' standpoint, and Section 5 describes how measurements and option thinking can be jointly used. Section 6 presents future work and Section 7 concludes the paper.

APSO'08, May 10, 2008, Leipzig, Germany.

2. METRICS IN AGILE DEVELOPMENT
Recent studies indicate an increased attention by the software engineering research community in the application of metrics in an agile context [16], [18]. Existing measurement techniques have been modified [6], [8] or new ones have been created [15], [16], [24] to serve the agile development approach. Based on our review of literature sources (referenced in this section), we can classify the existing measurement approaches in three classes: The first class includes measurements at code level aiming at quality improvement (also known as internal metrics). Examples are the System Design Instability (SDI) metric [1], the Running Tested Features [14], Knoernschild's code quality and design metrics [15] and Leffingwell's Iteration and Release Retrospectives [17]. Some internal metrics are bundled as quality management tools, for example, Kunz et al. [16] describe distinct metrics and their implementation into a measurement tool for quality management, to support refactoring. Ambler [2] recommends Quality Counts.

The second class refers to productivity/effort metrics which are applied in each iteration and serve project management purposes. Examples are the Burn-down charts [2], the Project Size Unit [8], the outcome measures in the XP Evaluation Framework [27] and the COCOMO-style effort model in [6].

The third class includes economic metrics which consider the outcome of the development process in terms of the added value that the product generates. Examples are the results of Rawsthorne [24] who suggests the new metric of Earned Business Value (EBV), and of Elssamadisy [10] who defines rhythms related to delivering value to the client. Metrics as return on investment or EBV may help the client to make midcourse decisions about the future of the project. Another commonly known metric is the Break/Even Point [2] which can be used for this purpose as well.

The first class of metrics can give valuable information about the quality of the code, but has the inherent drawback that it is not informative from client's point of view and offers almost no help in making decisions about project development. The economic and productivity metrics can support decision-making process. Nevertheless, they also represent some problems: In many cases these metrics use - as a source for the estimation, the design documents or the work breakdown structure, (as in the Project-Size-Unit model [8] and the EBV metric [24]) and might consider these documents as rigid and representative for the duration of the whole project. In turn, this would not reflect the dynamic and iterative nature of the agile development as these documents are not fixed at the start and are not known a priori for the whole project. Moreover, these measurement processes take as input client's estimations or judgements, which rests on the assumption that the client has in advance a clear idea about (i) the business value of each feature that is developed in the project, and (ii) the future business environment and project development. In fact, in an agile project the requirements and their priorities change during the project, thus reflecting current business situation.

3. OPTIONS THINKING IN IT PROJECTS

The Real Options Analysis (ROA) [9] is first known as a decision support technique in the area of capital investments. The concept of 'real' means adapting mathematical models used to evaluate financial options to more-tangible investments. Since 1999, this concept has found its way into the area of appraising IT investments [4]. The core of the ROA for IT assets consists of: (i) the identification and the assessment of optional components in a project, and (ii) the selection and the application of a mathematical model for valuing financial options that serves to quantify the current value of choosing these components for inclusion at a later time. Optional components are project parts that can either be pushed ahead or pulled out at a later point in time when new information becomes available to the decision-makers. The option, therefore, is the right but not the obligation to spend a budget or put resources on a project. For example, it is often possible to first implement a data mart, and then later decide to implement a data warehouse.

4. OPTIONS AND AGILE PROJECTS

Erdogmus [12] demonstrate that the agile method is especially appropriate for projects with high uncertainty. This, in turn, implies applying different decision-making approaches like ROA, decision theory [23], or robustness analysis [25]. This paper is solely focused on the use of real-options thinking. We think it is worthwhile exploring the use of ROA-concepts as a decision-making vehicle because:

1. Unlike traditional techniques, it comprehends uncertainty and it responds to the dynamics inherent in agile project context.

2. It supports the clients of agile projects in the context of a spectrum of possibilities rather than in the context of a single or three (the best, likely or worst case) discrete set-ups, and it facilitates reprioritization as client's realities unfold over time.

3. It allows incremental expenditures while focusing on the critical pieces of software functionality essential to accomplish the project mission.

4. It rests on the understanding that not all requirements and architecture design options are of equal value.

There exists awareness of the use of ROA in the agile software development. Erdogmus and Favaro [11] have already applied options-thinking to XP and put forward two XP value propositions, namely, that (i) 'delaying the implementation of a fuzzy feature creates more value than implementing the feature now' and that (ii) 'small investments and frequent releases create more value than large investments and mega-releases' [28, p.13]. Matts and Maassen [20] have put option thinking at work in XP and Scrum contexts. An important aspect in the analyses by these authors [11],[20] is their focus on the viewpoint of the developers. The client's role and decision-making process has received only scamp attention. This motivated us to focus on the clients' perspective. In this sense, our work is complementary to their approach. To the best of our knowledge, applying option thinking in support of clients' midcourse-decision-making processes has not been addressed so far by the research community. In the community of software metrics practitioners, Longstreet [18] found that studying the client happens very rarely, too.

This position paper considers real-options thinking to be applied two-fold in agile context: First, from client's perspective, real-option thinking can be deployed to prioritize the requirements in the start of each iteration so that the delivery of business value is optimized. Suppose, the EBV for each individual requirement is known to the client, s/he can re-arrange the requirements in sets that form options. Clearly, an option will be worth having when the cost of setting it up is less than its EBV (which in our case is the sum of the EBVs of all requirements that form the option). The client can, then, compare the advantages of each option and select the one that has the optimal EBV. The client can wait to the last responsible moment to make his decision on the set of requirements to be implemented and this allows her/him the chance to incorporate late breaking information and consider alternative sets of requirements. The term 'responsible' means that the client needs to understand the last point of time to make a decision without affecting the delivery of the project. If bad information comes in it costs the client nothing whereas if good information comes in the client gains value by having the option.

Second, from developers' perspective, the real-option thinking can support the implementation prioritization process. For example, the authors of [20] report on a practice of XP and Scrum developers who defer the decision about which story to develop until just before the coding starts. This allows them to incorporate information that arrives at the last moment, such as a new client request. In fact, the Scrum Backlog provides a forum where any idea for functionality can be recorded without requiring an immediate commitment to build it.

Taking an options approach to the decision-making process in agile software development is not about applying a new class of mathematical models. Instead, we look at it as a way of re-framing the discussion about spending and investment decisions in terms of options. The first step in re-orienting our way of looking at agile projects is to identify the options that exist in software decisions. Then, we will describe how practitioners can incorporate options thinking into their decision-making processes.

When regarding the agile development method as a sequence of decisions to be made (that is, as compound option model), we treat it as a series of options before or after each iteration. We call '**option**' the set of user requirements to be implemented in each iteration. Here we don't make a difference between functionality, quality of the product or documentation requirements. Each peace

of work that the client requires from the developers has an impact on the resources spent (e.g. budget, time), and thus on the outcome of the project. What remains important is to consider a dynamic decision-making process, typically taking part in the beginning of each iteration. The following options could be considered from client's perspective (Table 1):

Table 1: Description of Options.

Option	Description
Postpone	Wait to determine whether to implement certain requirements without imperiling the potential benefits.
Abandon	Abandon the project (terminate at the current stage).
Scope up	Add new functionality or quality features, not scheduled previously.
Scope down	Remove already implemented or negotiated features.
Switch	Change or re-arranging the stack with requirements.

Note that we don't consider the growth option, as it contradicts in its essence with the 'just enough' agile philosophy. '(However, we think, it is worth investigating the state of the practice in this regard.) Furthermore, we looked at literature in agile software engineering to find examples of the types of options as presented in Table 1. Some of the results of this effort are described below:

1. The option of Postponing: the experience of Poppendiek LLC [22] highlights the fact that the agile software process permits the clients to decide late when they have the best understanding it is going to get to achieve the best fitness for use it can within the constraints of the project.

2. The option of Abandoning: In many agile projects, the client has the right to cancel at the end of any phase, receiving the working, tested software from all phases completed so far. A published experience of a Control System Manufacturer [19] indicates how clients can cancel a project early if they find it is not going as expected and thus loose minimal investment; for example, a project review found that only 20% of the projected business value had been achieved, which was used by the clients to conclude that the project should no longer be pursued.

3. The option of Scoping-up: This is an inherent part of any agile process and the varieties of features or functionality pieces that might be added in any iteration, all depend on the types of stakeholders on the client's side involved. As [3] indicates, operations and support people, architects, regulatory compliance auditors, senior management, all may change their requirements. An interesting example of scoping up for large agile projects is provided in the Canadian Pacific case study [21]. Their agile philosophy was to "not close any doors" and accounted that the option of adding new features comes at the price of some rework which was "inevitable, acceptable and manageable".

4. The option of Scoping-down: It refers to settings in which the projects advance slower than the client expects. For example, Optimation, a software firm in New Zealand, reports of a project in which the client scoped down the functionality when it was found that the project was 25% complete and that the amount of work required would significantly exceed the initial estimates [26]. The client cut, then, the project scope to fit the budget.

5. The option of Switching: because agile applications are being developed in vertical slices instead of horizontal ones, the client never receives 100% of one tier completed before moving to the next one. This lets him/her switch some features and hook them together differently from the original set up. For example, at Sabre Airlines Solutions, clients compared alternative sets of features and switched to 'simplified functionality' at the beginning of each iteration they deemed an alternative set of requirements be at odds with their principle of 'make it run, make it right, make it fast' [28].

5. INCORPORATING MEASUREMENTS INTO THE DECISION-MAKING PROCESS

Clearly, the delivery of business value is the ultimate goal of an agile project and this is a common opinion throughout the agile community [2],[7],[12],[13],[24]. However, there is no published approach of how to achieve this in a systematic way. One way to go about it is to provide clients at inter-iteration time with accurate and easily available project information that is translatable in business terms, and hence, could support the analysis of the clients' options. We draw on a recommendation by Bowers [7], who after walking a mile in client's shoes, stresses the importance of keeping focused on business value. She indicates that the client is supposed to be in control of the product, but has no power over the process. Yet, his role is maintaining at all times the vision about business value. One of Bowers's recommendations is to give the updates & high quality information for decision making. This is exactly where we want to provide help by offering a systematic decision-making procedure.

When regarding a project from a client's perspective, what we see is a sequence of iterations. Each one can be considered as a mini-project, representing a relatively independent and closed entity. At the beginning, it is a prioritized list with requirements and, at the end, it is working software, with certain features and qualities. We suppose (take for granted) that the goal of a project is to maximize the business value. That is why the measure of EBV [24] is central for our further considerations. As the EBV is measured based on the WBS, we propose the following procedure (represented on Fig. 1) for the decision-making process:

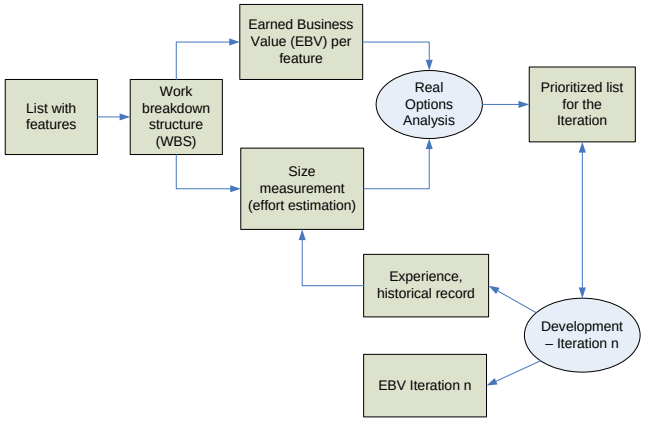

Figure 1. The iteration from client's perspective.

Assume preliminary sets of requirements for the project is defined. Based on that, the EBV per each user story (feature) can be estimated. The client, then, prioritizes the list based on max(EBV). Simultaneously, the WBS serves as a basis for the

developing team to estimate the effort, for example, by using PSU [8] as a sizing technique. Based on these two estimates, the scope of the iteration 1 can be defined. The remainder of the project can be considered as a start of a new mini-project – in which, again, the client has to prioritize the user stories, can decide to add new features, change the order of requirements to be implemented, or change quality characteristics. The decision for prioritization is taken based on options-thinking, as described in section 4. As an input at the start of this mini-project (iteration) the developer has to provide the client with information for his decision-making – namely size and effort estimation per feature or per set of features. This process is incremental and dynamic and any changes in the context can be taken into consideration.

6. FUTURE RESEARCH

Our analysis on how to re-think the decision-making processes in agile software project context points out that it is possible to find arguments which, when assembled, call for a new quantitative approach to costs and benefits to be considered when determining when and how much to invest in software functionality. Although the agile community does emphasize the importance of investigating cost/benefits relationships in agile projects, their published works [20], [24], cite isolated "islands" of solutions.

We propose the topic of options-based decision support be approached in a disciplined way as suggested by authors who applied ROA to other IT areas [5]. We plan to carry out this research, complementing it with four activities: (i) carrying out a systematic review on the topic of agile measurement, including metrics of the process (i.e. effort estimation, project size), of the product, as well as business metrics (ROI, EBV), (ii) identifying information necessary for the client's decision making, (iii) creating a dynamic measurement procedure reflecting the iterative and incremental nature of the agile development process [6], and (iv) exploring the influence of parameters of the project's context on the existing options.

7. CONCLUSIONS

We propose the idea to use the real options approach to help the client part in an agile project make decisions. We state that the prioritization of requirements and other decisions concerning project evolution can be regarded as a set of options. Metrics could be used to support this process. The client's role as a decision-maker has been under-researched and we strongly believe that this direction is worth investigating.

8. REFERENCES

[1] Alshayeb, M., W. Li, An Empirical Study of System Design Instability Metric and Design Evolution in an Agile Software Process, J of Systems and Software, 74(3), 2005, pp. 269–274

[2] Ambler, S., Scaling On-Site Customers, Dr Dobb's Journal, Dec, 2007.

[3] Ambler, S.W., Measure Me Wisely, Dr Dobbs's Journal, July 2005, http://www.ddj.com/architect/184415360

[4] Amram, M., Kulatilaka, N.: Real Options: Managing Strategic Investment in an Uncertain World. HBS Press, Cambridge, 1999.

[5] Benaroch, M, Managing Information Technology Investment Risk: A Real Options Perspective. J of Management Info Systems, 19(2), Fall 2002, pp. 43-84.

[6] Benediktsson, O., D. Dalcher, K. Reed, M. Woodman, COCOMO-Based Effort Estimation for Iterative and Incremental Software Development, Software Quality Journal, 11(4), 2003, pp. 265-281.

[7] Bowers, A., Leading From A Position Of No Power: A Customer's Perspective of an Agile Team, http://www.infoq.com/presentations/alexia-bowers-agile-leadership

[8] Buglione, L., Project Size Unit, http://www.geocities.com/lbu_measure/psu/psu.htm

[9] Childs, P. D.; Ott, SH.; Triantis, A.J. Capital Budgeting for Interrelated Projects: A Real Options Approach, J of Financial & Quantitative Analysis, 33(3), 1998, pp 305-335.

[10] Elssamadisy, A., A.Johnson, Rhythms As Agile Diagnostics, Agile Journal, June 2006, http://www.agilejournal.com/articles/articles/rhythms-as-agile-diagnostics.html

[11] Erdogmus, H., Favaro, J. Keep Your Options Open: Extreme Programming and Economics of Flexibility. In Extreme Programming Perspectives, M. Marchesi et al (eds). Addison-Wesley, 2002.

[12] Erdogmus, H. The Impact of Learning and Flexibility on the Economics of Iterative Development. IEEE Software, Nov/Dec 2005.

[13] Hartmann, D.; Dymond, R., Appropriate Agile Measurement: Using Metrics and Diagnostics to Deliver Business Value, AGILE 2006.

[14] Jeffries, R., A Metric Leading to Agility, in XProgramming, June 2004 http://www.xprogramming.com/xpmag/jatRtsMetric.htm

[15] Knoernschild, K., Using Metrics To Help Drive Agile Software, in Agile Journal, June 2006, http://www.agilejournal.com/articles/the-agile-developer/using-metrics-to-help-drive-agile-software.html

[16] Kunz, M.; Schmietendorf, A.; Dumke, R.: How to measure Agile Software Development, Int'l Conf. on Software Process and Product Measurement, Palma, Spain, 2007, pp. 319-325

[17] Leffingwell, D., Iteration and Release Retrospectives: The Natural Rhythm for Agile Measurement, Agile Journal, June 2006, http://www.agilejournal.com/articles/articles/iteration-and-release-retrospectives:-the-natural-rhythm-for-agile-measurement.html

[18] Longstreet, D., Agile Methods and Other Fairy Tales, 2007, http://www.softwaremetrics.com/Agile/Agile%20Paper.pdf

[19] Mahanti, A., Challenges in Enterprise Adoption of Agile Methods – a Survey, J of Computer & Information Technology, 14(3), 196-207

[20] Matts, C., Maassen, O., „Real Options" Underlie Agile Practices, http://www.infoq.com/articles/real-options-enhance-agility

[21] Meszaros G., J.Aston, Agile ERP: "You don't know what you've got 'till it's gone!", AGILE 2007.

[22] Poppendiek, http://www.poppendieck.com/

[23] Pratt, J.W., H. Raiffa, R. Schlaifer, Introduction to Statistical Decision Theory, MIT Press, Cambridge, MA, 1995.

[24] Rawsthorne, D., Calculating Earned Business Value For An Agile Project, Agile Journal, June 2006, http://www.agilejournal.com/articles/articles/calculating-earned-business-value-for-an-agile-project.html

[25] Rosenhead, J., Robustness Analysis: Keeping Your Options Open, In J. Rosenhead et al (eds.) Rational Analysis for a Problematic World Revisited: problem structuring methods for complexity, uncertainty and conflict, Wiley, Chichester, 2001, pp. 181-207

[26] Rusk, J., Cutting Scope in not the (Whole) Answer, 2004, http://www.agilekiwi.com/cutting_scope.htm

[27] Williams, L., W. Krebs, L. Layman, A. Anton, P. Abrahamsson, Toward a Framework for Evaluating Extreme Programming, Int'l Conf. on Empirical Assessment in Software Eng. (EASE) 2004.

[28] Williams, M., Jay Packlick, Rajeev Bellubbi, Scott Coburn, How We Made Onsite Customer Work - An Extreme Success Story, AGILE 2007, pp. 334-338