How Do Real Options Concepts Fit in Agile Requirements Engineering?

Conference Paper · January 2010

DOI: 10.1109/SERA.2010.37 · Source: DBLP

2010 Eighth ACIS International Conference on Software Engineering Research, Management and Applications

How Do Real Options Concepts Fit in Agile Requirements Engineering?

Zornitza Racheva, Maya Daneva
Information Systems Group
University of Twente
Enschede, the Netherlands
z.racheva@utwente.nl, m.daneva@utwente.nl

Abstract—**Agile requirements engineering is driven by creating business value for the client and heavily involves the client in decision-making under uncertainty. Real option thinking seems to be suitable in supporting the client's decision making process at inter-iteration time. This paper investigates the fit between real option thinking and agile requirements engineering. We first look into previously published experiences in the agile software engineering literature to identify (i) 'experience clusters' suggesting the ways in which real option concepts fit into the agile requirements process and (ii) 'experience gaps' and under-researched agile requirements decision-making topics which require further empirical studies. Furthermore, we conducted a cross-case study in eight agile development organizations and interviewed 11 practitioners about their decision-making process. The results suggest that options are almost always identified, reasoned about and acted upon. They are not expressed in quantitative terms, however, they are instead explicitly or implicitly taken into account during the decision-making process at inter-iteration time.**

Keywords-Requirements prioritization, agile software development, decision-making process, business value, Real Options, case study

I. INTRODUCTION

Adoption of agile requirements engineering (ARE) has increased in the last few years which, in turn, led to an increased number of organizations that experienced the various elements of the agile processes, reflected on their experiences and challenges, and shared them at conferences. In the agile community, among the key lessons learnt from experiences, which draw a heightening attention [1], are those referring to the optimization of the value creation for the clients in agile projects. Indeed, the IEEE AGILE conference, a premium agile software engineering event, has now 'Business Value Creation' as a permanent track on its program. Essentially, in agile software projects, the development process is a value creation process [2]. Key to value creation [3] is the decision-making that takes place at inter-iteration time, when requirements are reprioritized in the face of project uncertainties.

As agile techniques are recognized as most suitable for projects under uncertainty, one decision-making concept that seems to fit well with agile requirements prioritization is real-option analysis [4]. In previously published studies [5] it was demonstrated that the application of real options thinking was useful to decision-makers in the RE stage of other types of IT projects (e.g. ERP). Drawing on these earlier results, we set out to investigate the fit between real options thinking and the ARE context. More specifically, our objective is to understand how the real options thinking fits into ARE from the perspective of the client.

This papers first looks into the experiences published previously in the agile software literature, to collect and analyze examples of how developers and customers reason together on those requirements which create the most business value. We, then, followed up on this analysis by collecting and comparing the experiences from agile practitioners in eight companies.

This paper is set out to answer four research questions (RQ):

RQ1: *What is the level of agile software organizations' awareness of using options thinking in support of agile requirements reprioritization?*

RQ2: *In which way does options-thinking add value?*

RQ3: *What kinds of options have been most discussed in the published experiences?*

RQ4: *Which aspects of using options thinking in agile RE can be recognized as topics for future research?*

We make the note that answering RQ1 and RQ2 brings insights into RQ3 and RQ4. The approach we deployed to answer the research questions complementarily uses a scoping review [6], the CHAPL framework [7], and a case study [8]. In what follows, Section II provides background on ARE and on real options concepts. Section III presents our research approach. Section IV evaluates the fit of real-options thinking to agile projects from clients' point of view. Section V discusses the limitations of this study and Section VI concludes with implications and pointers for future research.

II. RELATED WORK

A. Requirements Engineering in Agile Context

This section is an introduction for readers who are less familiar with agile software project contexts and agile software development and management approaches. Agile approaches to software project delivery and to software

978-0-7695-4075-7/10 $26.00 © 2010 IEEE
DOI 10.1109/SERA.2010.37

product development can be considered a paradigm, a project management philosophy, a culture, an attitude, and a state of mind. All these rest on the 'minimalist' principle of organizing work in the software development process, meaning a conscious choice in carrying out those tasks which directly create value for clients and leaving out anything that is deemed "waste" [9]. The latter refers to all work and work products not directly contributing to the development of software. The 'minimalist' principle is fundamental to the ability of the agile approaches to cope with project uncertainties. In that sense, this principle can be seen as a reaction to the 'plan-based' paradigm which assumes that problems are fully specifiable and that predictable solutions exist for every problem [9]. Agile approaches, such as Extreme Programming (XP), SCRUM or CRYSTAL, for example advocate requirements engineering (RE) through the software product development cycle in small and informal stages. That is, instead of engineering the requirements upfront, one lets requirements emerge during development. Agile software process practitioners deem this approach particularly valuable for software producers in a context that includes highly uncertain requirements, experimentation with new development technology, and clients willing to explore the ways in which an evolving product can help their business goals.

The RE process in agile projects differs from those in a 'traditional' development both in its philosophy and implementation. While the main RE activities (e.g. requirements elicitation, documentation, and negotiation/prioritization) are present in both development paradigms, the purpose of these activities and the way they are performed are fundamentally different. For example, a comparative study [10] on ARE and traditional RE highlights the unique aspects of running an ARE process, despite that the RE activities might be seemingly identical to those ones used in traditional projects. Furthermore, Cao et al [11] have investigated the topic of agile RE and have identified that the agile context carries out requirements activities by implementing extensive face-to-face communication, iterative and extreme prioritization, constant planning, prototyping, test-driven development, and reviews and tests. In [12], Ambler presents a list of the following practices (setup in italic here), which implement the RE activities: *Stakeholders actively participate; Adopt inclusive models; Take a breadth-first approach; Model storm details just in time*; *Treat requirements like a prioritized stack; Prefer executable requirements over static documentation; Your goal is to implement requirements, not document them; Recognize that you have a wide range of stakeholders; Create platform independent requirements to a point; Smaller is better; Question traceability*. Clearly, all these practices [10,11,12] may appear in traditional projects (that follow waterfall life cycle models) but when pertaining to the agile context, they do support the value offering of this paradigm, which is its ability to provide easy response to changes and faster value creation for the clients. The actual value, delivered to the client, is represented by the features of the product, at each state of the project. Thus, the selection of features for each release and each iteration determines the compound value created. We note explicitly, that value creation is a RE responsibility, and the activities, that contribute to this bottom line are: (1) requirements change management, (2) prioritization and (3) release and iteration planning. As we will see in the next sections, one important advantage in an agile project is the possibility to look at the project as a set of options at each step of its development. This not only represents a means to manage the changes, but is also an opportunity to optimize the value creation. The different 'options' a team has in respect to these activities form the core of this paper and are discussed in Section IV.

B. Options-based thinking of IT Projects

The Real Option Analysis (ROA) [4] 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 [13]. 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.

When discussing the 'options' in this paper, we explicitly take a specific viewpoint into account, namely the perspective of either the developer or the client. This is, however, not about applying a new class of mathematical models. Instead, we look at it as a way of re-framing the discussion about client's spending and investment decisions in terms of options. Clearly, the first step in re-orienting our way of looking at agile projects is to identify the options that exist in agile software project management decisions. Only then it will be possible for practitioners to incorporate options thinking into their decision-making processes.

III. RESEARCH APPROACH

To answer the questions posed in the Introduction, we applied a research approach that is explorative and qualitative in nature. It is presented in Fig.1 and it includes three stages and combines scoping review techniques [6], the CHAPL framework [7], and qualitative case study research techniques [8] for exploring the relationship between option thinking and agile RE processes.

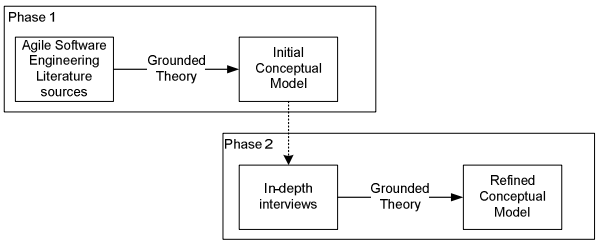

Figure 1. The two-phase research process.

Scoping review means searching literature to determine the sorts of studies addressing the research questions. Once the relevant literature sources are identified and screened for inclusion in the study, we applied the CHAPL analysis techniques (which formed the second stage of our research process). CHAPL stands for applying five analytical techniques: **C**ontextual analysis, **H**istorical analysis, **A**nalysis by analogy, **P**henomenological analysis, and **L**inguistic analysis. We make the note that these techniques can be used complementarily but it does not mean that all of them must be used simultaneously in the exploration of a specific aspect of the relationship between real-options thinking and agile RE. The techniques are described in much detail in [7]. We note that our choice of using this framework was motivated by the positive experience other researchers did with it in studying agile phenomena [7]. The five techniques have also been proven useful when explaining systems engineering phenomena and when discussing the philosophy of technology [14].

Last, after completing the analysis by using the CHAPL framework, we carried out a case study [8] with the objective to extend the exploration of options thinking in agile projects. The case study took the form of a series of consultations and in-depth interviews with practitioners based on the results of the first two stages of our research. We collected experiences from projects in eight companies and compared these experiences to what we found through the analysis in the first two stages of this research. The consultations with the practitioners were an opportunity to take stock of the potential scope of the 'options-thinking' evidence base and to decide if further work was feasible and useful for both practitioners and researchers.

We applied the three-stage research approach (in Fig. 1) as follows: First we searched and identified relevant publications in four digital libraries: Scopus, IEEE, ACM and Science Direct. The search was done in the period of August 18 and August 22, 2009. We used this search strong of key words:

(((agile software development) OR (agile project management) OR (agile software project) OR (agile requirements)) AND (prioritization OR reprioritization OR decision-making, OR options OR real options)).

Our search yielded 67 papers. As an input to the CHAPL framework, we selected a subset of those publications which met the following three criteria: (i) the publication describes a requirements prioritization decision-making experience in a real-life project setting in an organization; (ii) the publication explicitly presents the type of uncertainties accounted for in the decision-making process, (iii) the publication provides enough detail so that the reader can unambiguously understand how the experience relates to his/her own practice. We applied the inclusion criteria to each paper, based on our review of its title, abstract, case description and conclusion. 21 out of the 67 papers met our three inclusion criteria. We make the note that the list of the 21 papers can be received from the authors. Next, these papers were subjected to exploration by using CHAPL. The results of this process are presented in Sections IV.A, IV.B, and IV.C. In Section IV.D we report on the last step of our research process, namely the case study, and discuss its results.

IV. Options and Agile Requirements Engineering

This section summarizes our findings regarding each research question and discusses their implications.

A. The level of organizations' awareness of using options to make requirements decisions

We found ten publications [15,16,17,18,19,20,21,22,23] which explicitly focus on experiences in the application of options thinking to agile context. Three authors explicitly state that there is awareness of the use of ROA in the agile software development [16,18,19]. In [16,19], options-thinking has been applied to XP and the authors 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' [19]. Furthermore, [18] describes the options associated with delaying decisions and illustrate their points by using the Microsoft strategy (circa 1988) and [17] has put option thinking at work in XP and Scrum contexts. In [15,21], the author demonstrated how options-thinking was used to help organizations handle their context uncertainties and create flexibility and how this supported the creation of economic value in software projects. In [23], the author frames the discussion of option thinking from the perspective of agile risk management. He presents three techniques that deploy option thinking to manage risk in agile projects.

B. Perspectives on options

Our study found that real-options thinking can be applied two-fold in agile context: from the client's perspective as well as from the developer's one. We, however, found that an important aspect in the analyses in the papers [16,17,22] is their focus on the viewpoint of the developers. It seems that the client's role and decision-making process has received only scamp attention. We found only one publication [18], which considers options thinking from client's perspective. From this point of view, real-options thinking can be deployed to prioritize the requirements at the start of each iteration so that the delivery of business value is optimized. Suppose, the business value (BV) 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 BV (which in our case is the sum of the BVs 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 BV. The client can wait to the last responsible moment (as it is called in [18]) 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-options thinking can support the implementation prioritization process. For example, the authors of [17] report on a practice of XP and Scrum developers who defer the decision about which story to develop until just before the coding task 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 [17] where any idea for functionality can be recorded without requiring an immediate commitment to build it.

C. *Formulating client's options*

Despite the fact that we found only one publication explicitly discussing how options thinking can be a useful decision making vehicle from client's perspective, we should note that the agile literature that we searched does provide examples of organizations who think of requirements in terms of options. (Some examples are presented later in this section). We catalogued the clusters of organization-specific experiences and used the CHAPL analytical techniques to discern kinds of options which seemed to be considered and used in the requirements reprioritization processes discussed in the publications which we analyzed. The result of this analytical process is presented in Table I, which formulates the options we found to exist from client's perspective.

TABLE I. 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.*
Growth	*Decisions that will make future value growth possible.*

When regarding the agile development method as a sequence of decisions to be made, 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.

Furthermore, we looked at the papers included in this study to find examples of the types of options as presented in Table I. Some of the results of this effort are described below:

1.The option of Postponing: the project of ThoughtWorks (an agile coaching firm) and a major US insurance company, working on re-writing a large Java application used in support of core business processes, is a case in point [24]. In this project, the clients structured their business requirements in so-called 'epics' which are a compound story framing a software feature in context of a business scenario. At inter-iteration time, the client went through the release plan's epic list and marked each one as 'Must have', 'Should have', Could have', and 'Won't have (this time around)'. Epics which were not a part of the initial release plan and deemed lower in priority had been deferred until a future release.

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. The experience of a Control System Manufacturer [25] 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 [26] indicates, operations and support people, architects, regulatory compliance auditors, senior management, all may change their requirements.

4.The option of Scoping-down: Yahoo!'s Mixd project [27] illustrates the use of this option. In their social mobile product experiment, the Yahoo! Advanced Product team cut features and learnt how removing complexity and assumptions in the product allowed people to use it in unintended way. The team prioritized the features by asking if the feature was absolutely necessary to help the users achieve their goal. They brutally cut on those features that diluted the key focus of the product. Dropping those features that they had specified earlier was a major conceptual shift, but turned out to be an easy shift to make, as it eliminated development complexity.

5.The option of Switching: because agile applications are 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 agile principles [19].

6.The Growth option represents the opportunity to make decisions that will, in the future, lead to a value growth. This aligns with Ambler's observation in [28] that one should not optimize "too locally". From client's perspective – one presumes that the client pays for the software because he/she wants to achieve certain benefits in the future. Whether such growth will be realized will depend on many factors, and one of them is the software product. Thus, the growth for the client's organization will depend on the selection of the features-to-be-included in the product. The growth from the developer's perspective is related to the process of software solution development and considers those agile practices that will help the growth of the developers' organization in the future. It's worth noting, that while the growth option on the first glance contradicts with the 'just enough' agile philosophy, we found examples that companies actually consider this option.

Last, we make the note that no study in the literature sources we reviewed has discussed options thinking by using quantitative terms, for example amount of money assigned as 'value' to each option, or amount of potential savings expected to be realized by choosing an option.

D. The follow-up case study

We conducted a multiple-case study [8] to explicate the decision-making process during a project in context of agile projects and changing requirements. It is a first step in discovering the way in which the agile requirements mid-course decision process contributes to the client's value creation. We studied how requirements prioritization and decision-making on priorities happen and what are the factors that play a role in it. As part of studying how companies prioritize requirements in agile projects, we observed examples of options-thinking as practiced by agile teams. The case study consisted of semi-structured open-end in depth interviews with practitioners that work in organizations that develop software by using agile approaches. We included 11 practitioners who described a total of 11 projects and who were working for eight different companies. The application domains for which software solutions were developed included banking, ERP for small businesses, health care management, automotive, content management system, online municipality services. Each in-depth interview included questions on the following topics: *(i) who decides about priorities? (ii) what criteria are applied? (iii) what were the reasons for changes in the backlog? (iv) has value been explicitly considered? (v) how the agile process creates value for the client? (vi) does the developer consider other factors in making decisions, a part from the value for the client?*

After compiling this information and mapping pieces of practitioners' evidence against the options we presented in Table I, we identified points of convergence and divergence between what we found in the first two stages of our research (see Fig. 1.) and the third stage (namely, the case study). In the remainder of this section, we provide a discussion on our observations.

1. The case-study observations suggest that options are almost always considered. However, we must note that not all types of options in Table I were equally considered by all practitioners that we interviewed. This means, first that not all types of options are relevant for all projects, for example, we didn't observe even a single case of a terminated project (i.e. *abandon* option), while the *switch* option was present in all projects we investigated; and second, there seems to be a relation between the concrete context of the project and the options that come for consideration. For example, in two projects with hard-deadline we observed that the primary goal of both developers and clients was to implement the absolutely necessary functionality that will support the client's needs, and to stay within the deadline. In these projects, the options that were considered were *scope down* and *switch*, as the decision-makers were driven by the goal "how to implement the main functionality while remaining within resources". As those were relatively short projects, the *postponing* option didn't seem to be relevant because no additional information about concrete features was expected to arrive during the project. In addition, the *scope up* option was not feasible because of the limited resources. In one of these projects the client faced the following situation: during the project more information became available about the functionality of the open source product that the developer intended to use. It turned out that this product didn't support an important function. Based on this new information it became clear that this function had to be implemented as a part of the project, and because of that the initial backlog can not be implemented within the deadline for the project. The client then considered the following two options: (i) to downsize the initial project backlog in order to free up resources for implementing the important feature, or (ii) to find a detour way for implementing this feature and exercise the switch option. After estimating the effort needed in the both cases and the impact on the final product, the client choose the first option.

We make the note that although in this case the client and the developer explicitly discussed these options as possible ways to proceed in the situation, the discussion was not led in terms of options. This means that no quantitative comparison between those options was made.

2. We consistently observed that the options were not stated in quantitative terms, but are, instead, explicitly or implicitly taken into account during the decision-making process. This agrees with the findings from phase 1 of this research.

3. The evidence from the study shows that, in contrast to the agile literature, in most of the cases the developers are those who made inter-iteration decision making and are concerned with options and options thinking. Practitioners agreed on that most often than not the involvement of the clients consisted mainly to approve the plan and give comments. Only in few cases practitioners were able to

provide evidence that the client is really capable/interested/aware of the agile way of defining priorities, and thus able to navigate the functionality by the mid-course decision-making process. The practitioners went further to explain why this happens in practice. In their view, the developers' company is the one to make sure that the project delivery process runs in a way that is profitable for the company. If developers accommodate all wishes which clients might come up with at inter-iteration time, the company may find it not sustainable in the long run. Hence, while an agile software company lets its clients prioritize the requirements, this decision-making process can take place only when the client's sense of flexibility is balanced against the company's sense of profitability.

4. Throughout the conversations, it became explicit that there is a connection between the project's settings (e.g. size of the company) and the way the decisions are made (including the kinds of options considered). In all projects where the client's company was a small company, the decision making was deliberately delegated to the developer. It could be a product owner, a project manager or another representative of the developing team, that was responsible for the communication with the client.

5. We observed that the developers' decisions are often driven by options thinking, where the clients' perspective was taken seriously into account. Below, we present examples for switching, growth and scaling up options, considered from clients' perspective:

(i) One case study participant reported on a project aimed at delivering a mortgage management application for a bank. In this project, the developers decided to implement first those pieces of functionality, that can support the current and future workflow of the client, and then to scale up. Specifically, the developers first created the code that ensured working functionalities that let bank's staff compose and tailor mortgage offers to their clients' needs. Building upon this functional software system, they added later on mortgage approval functionality, and further – they implemented those software features that support the mortgage handling and service activities. This is an example of a Scale-up option.

(ii) Another company that implemented a management system for health-care practitioners decided to move the sub-system for financial operations earlier in the implementation schedule. This change was driven by the desire of the developers to support as early as possible the needs of the clients (in this case, the clients were the company's Taxation Operations analysts who badly needed the functionality as soon as possible because the approaching end of the quarter. Certainly, the team could have left this functionality for later stage of the project, as was initially scheduled. The switching, however, saved the end client a lot of efforts.

(iii) In a small agile team, we observed the option thinking being applied to fit into the resources – at each iteration, the team developed those features, for which minimal use of resources was estimated. This is as well a growth option, as the project can be extended on the current grounds, whenever more resources are available.

E. Open Questions Regarding the Fit of Options Thinking and ARE

Clearly, the delivery of business value for the client is an accepted position in the agile community [9,15,29,30,31,32]. Our study indicates that in order to determine the business value of a feature, both clients and developers are building on experts' knowledge in the domain of analyzing business value of IT by using options-thinking. Methods based on options-thinking let organizations account for multiple aspects of the IT adoption, as operational, strategic, and managerial benefits from financial, customers' or process perspectives. In the sources we reviewed, we could find examples of such benefits: improved process efficiency, reduced cost, increased productivity and better satisfaction of customers' needs. These findings indicated to us that it is worthwhile exploring the use of real options concepts as a decision-making vehicle because:

1. Agile project management embraces change and accepts uncertainty of project context as inherent to the project. So, real-option thinking will reflect this.

2. Options thinking 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.

While real-options-thinking fits well with agile context, a number of challenges lie ahead in further developing this approach: (1) customizing ROA for agile decisions requires estimating the costs and benefits associated with the current features and analyzing their interaction; (2) the current mathematical models used in options valuation require collection and analysis of statistical data about agile decisions. Such data may not be easily available for researchers to carry out case studies; (3) agile prioritization has a cultural aspect, so company's culture, or country's culture might favor or limit the application of certain options; so, we would need to understand the role of context when considering viable options; (4) the software architecture for a project needs to be built such that optional components are indeed really optional: it is neither necessary nor impossible to add them at a later moment in time. This means that we need an engineering approach that takes adaptability very seriously; (5) the real-options-based approach is effective only if a company is genuinely prepared to cancel projects after their initial investment. And anybody who has spent any time in the corporate world knows that projects tend to acquire invisible momentum that is hard to stop.

Furthermore, we must note that, to the best of our knowledge of agile literature, we could find no published approach experienced by an organization who used options-thinking to deliver, or maximize business value in a systematic way for the client (see Section IV.A and IV.B).

This finding converges with a finding from previously published systematic review on how agile practices deliver business value [3]. In that earlier study, we found that despite the proponents of agile software development and project management suggest a strong focus on value delivery, there is no single published study on specific instance of value an organization realized based on the use of specific agile practices.

V. LIMITATIONS

There are three main validity concerns pertinent to this study: (i) our selection of publications to be included, (ii) our analysis of experiences, and (iii) potential bias by each of the authors when applying the inclusion criteria.

The search step of our scoping review was executed separately by the first and the second authors. The first author searched the ACM, IEEE and the second – Scopus and ScienceDirect. Each co-author individually screened titles, abstracts, case study descriptions, and conclusions and discarded the hits returned in the respective databases. The authors worked in isolation from each other in two locations and met only after this step was completed.

Furthermore, approximately half of the selected papers were reviewed by both researchers. For these papers, we consistently observed a consensus. Whenever there was disagreement, the points of disagreements were discussed until both researchers arrived at a consensus.

We believe that the threat to validity due to our own bias is minimal, because no one of the authors (i) has published a study which is included in the scoping review or (ii) is in a close research-collaboration relationship with the authors of included studies.

VI. CONCLUSIONS

This paper presents a study that explored the fit between real options thinking and ARE. It yielded the following findings:

RQ1: *What is the level of agile software organizations' awareness of using options thinking in support of agile requirements reprioritization?* Both the literature sources and the case study showed that there is awareness in the organizations, and that they apply option thinking for making mid-course project decisions, both from clients and developers perspective. Although the agile companies propagate development process driven only by value creation for the client, we observed that in practice option thinking is intrinsic for the developers as well. They consider trade offs between quality and schedule, possibilities for reuse; available resources and concurrent projects. In terms of options these are: possibility to scale up, growth option and postpone option.

RQ2: *In which way does options-thinking add value?* In the searched literature, we could not find a case where options are explicitly documented and compared in terms of value or in other quantitative way. Nevertheless, the findings from the case study showed that: (i) the options thinking helps to increase the value delivery for the client by flexibly handling the changes and client's wishes, rapidly answering a change in the project situation, or in a newly gained insight, e.g. about technical feasibility; (ii) it allows for better use of resources and can help a project stay within resources; (iii) options thinking is a means to capitalize on the learning experience; (iv) the approach makes possible to make such decisions that can lead to the best possible system for the available resources (schedule or budget); (v) it can contribute to the value creation both for developer's and client's organizations; (vi) the developers remain flexible, serve the client well and create trustful relationship, and in the same time can maximize the gain for their own organization.

RQ3: *What kinds of options have been most discussed in the published experiences?* We found six types of options being applicable to agile projects. The answer to this question is extensively discussed in sections IV.B and IV.C.

RQ4: *Which aspects of using options thinking in agile RE can be recognized as topics for future research?* Based on the discussion in this paper, we derived challenges for research and practice. (This is explicitly discussed in Section IV.D.) We summarize them below:

1. Experiences shared in the literature on agile software engineering and those shared by the practitioners in our case study indicate that agile clients' and developer's organizations are aware of the potential of options thinking to support agile requirements re-prioritization at inter-iteration time. However, we found that the options thinking is mostly described in terms of how it works for developers' organizations. The perspective of the clients' organizations seems under-researched. This means that more research efforts in this area could possibly bring new insights and help us get a much deeper understanding of the options that are important to clients in a variety of agile contexts.

2. Six types of options seem to be relevant from client's perspective. However, research is needed to demonstrate in which way each option-thinking type adds value in a project.

3. In both the literature review and the case study, we found that options are not expressed in quantitative terms. This finding makes us think that it may not be realistic at all to expect agile teams to reason about options quantitatively. Whether this is the case or not is a line for future research.

4. We must note that in the literature, we found instances of using options-thinking which represent anecdotic experiences of either agile consultants or agile-practice-adopting organizations. We were really surprised that we couldn't find a more substantive evidence that could be used to answer our research questions. This raises the question about what represents the existing body of knowledge on the subject and also, which research approaches to use to uncover this knowledge. Furthermore, the matter that our scoping literature review and our case study could not find a satisfactory answer to RQ2, may also mean that this question needs to be investigated by using alternative research approaches, for example, by means of industry surveys, or action research. We think that it might be the case that time is right for ARE researchers and practitioners to look more closely at the phenomenon of value-creation through options at requirements reprioritization time. This gives us the incentive to do further empirical research on how clients make decisions in agile projects based on their own understanding of their options at inter-iteration time. For this

purpose we will apply another empirical method, following the recommendation in [31]. At the time of writing this paper, we are in the process of executing a case study research at midsized agile software companies in the Netherlands.

ACKNOWLEDGMENT

This research has been financially supported by NWO-the Netherland's Organization for Scientific Research under the QUADREAD project. The authors thank the practitioners who contributed their time to this research project.

REFERENCES

[1] Abrahamsson P., Salo O., Ronkainen J. & Warsta J., Agile Software Development Methods. Review and Analysis, VTT Publication # 478, 2002.

[2] Principles behind the Agile Manifesto, 2001, URL: http://agilemanifesto.org/principles.html

[3] Racheva, Z., M. Daneva, K. Sikkel, Value Creation by Agile Projects: Methodology or Mystery? In: Proc. of the Conf. on PROFES, Springer, LNCS, pp.141-155

[4] 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.

[5] Daneva, M., Applying Real Options Thinking to Information Security in Networked Organizations 2006, Internal Report, University Twente

[6] Arskey H, O'Malley L: Scoping studies: Towards a methodological framework. Int J Soc Res Methodol 2005, 8(1):19-32

[7] Jiang L., Eberline A., Towards a framework for understanding the relationships between classical software engineering and agile methodologies ICSE 2008 Workshop on Scrutinizing agile practices or shoot-out at the agile corral table of contents, pp. 9-14.

[8] Yin, R.K. (1984) Case Study Research: Design and Methods.,

[9] Dyba, T., T. Dingsoyr, Empirical Studies of Agile Software Development: a Systematic Review, Journal of Information and Software Technology, 50, 2008, pp 833-859.

[10] Paetsch, F. Eberlein, A. Maurer, F., Requirements engineering and agile software development, 12th IEEE Int. Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, 2003.

[11] Cao, L, Ramesh B., Agile Requirements Engineering Practices: An Empirical Study, IEEE Software, Jan/Feb, 2008 pp. 60-67.

[12] Ambler, S., Agile Requirements Best Practices, http://www.agilemodeling.com/essays/agileRequirementsBestPractices.htm

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

[14] Dahlbom B. Mathiassen L. 1992. Systems Development Philosophy, ACM SIGCAS, Computers and Society, Vol. 22, Issue 1-4, Oct. 1992, pp: 12 – 23

[15] Erdogmus, H. The Economic Impact of Learning and Flexibility on Process Decisions, IEEE Software, Nov/Dec 2005, pp.76-83

[16] 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, URL: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.24.1786

[17] Matts, C., Maassen, O., "Real Options" Underlie Agile Practices, June 8 2007, URL: http://www.infoq.com/articles/real-options-enhance-agility

[18] Poppendiek, M., T. Poppendiek, Implementing Lean Software Development: from Concept to Cash, Addison-Wesley, 2006,

[19] Williams, M., Jay Packlick, Rajeev Bellubbi, Scott Coburn, How We Made Onsite Customer Work - An Extreme Success Story, Proc. of the Agile Conf. 2007 (Agile 2007), Washington, DC, USA. IEEE CS, pp. 334-338

[20] Rico, D., H.H. Sayani, A. Sone, The Business Value of Agile Software Methods: Maximizing ROI with Just-in-Time Processes and Documentation. Ross Publishing, 2009

[21] Maasen, O. C.Matts, Realistic about Risk: Software development with Real Options, QCon, London 2008.

[22] Anderson, D., New Approaches to Risk management, Proc. Of the Annual Software Development Conference, QCon, San Francisco, 2009.

[23] Hartmann Preus, D., Lean + Real Options = Reduced Complexity, InfoQ, Jan 21, 2010, http://www.infoq.com/news/2010/01/lean-real-options

[24] Tengshe, A., Establishing the Agile PMO: Managing Variability across Projects and Portfolios, Proc. of the Agile Conf. 2007 (Agile 2007), Washington, DC, USA. IEEE CS, pp. 188-193

[25] Mahanti, A., Challenges in Enterprise Adoption of Agile Methods – a Survey, J of Computer & Information Technology, 2006, 14(3), 196-207, URL: http://cit.zesoi.fer.hr/downloadPaper.php?paper=752

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

[27] Gatz S. A., G. Benefield, Less, Never More: Launching a Product with Critical Features and Nothing More, In: Proc of AGILE Conf, 2007, IEEE CS, pp. 324-327.

[28] Ambler, S., The Manifesto for Software Craftsmanship, Dr Dobbs's Journal, March 2009; http://www.drdobbs.com/architecture-and-design/216200102

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

[30] 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

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

[32] 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