

ULRR

The combination of agile and lean in software development: an experience report analysis

Item Type	Meetings and Proceedings
Authors	Wang, Xiaofeng
Citation	Agile 2011 Conference;08/2011
Download date	2026-03-14 23:23:06
Item License	https://creativecommons.org/licenses/by-nc-sa/1.0/
Link to Item	https://hdl.handle.net/10344/1705

The Combination of Agile and Lean in Software Development: An Experience Report Analysis

Xiaofeng Wang

Lero – the Irish Software Engineering Research Centre
Limerick, Ireland
e-mail: xiaofeng.wang@lero.ie

Abstract—There has been a noticeable focus shift from agile methods such as eXtreme Programming (XP) and Scrum to lean software development in the last several years, which is indicated as “from agile to lean”. However, the reality may not be as simple or linear as the term implies. To provide a better understanding of the combined use of agile and lean approaches in software development, a set of experience reports were analysed. These reports were published in the past conferences dedicated to agile software development and report experiences of using both agile and lean. The results of the analysis show that agile and lean can be combined in different manners for different purposes in software development. Lean is often applied as guiding principles for agile development. When combined at practice level, flow-based lean processes tend to substitute time-boxed agile processes.

Keywords—*agile methods; agile software development, lean thinking; lean software development; XP; Scrum; experience report*

I. INTRODUCTION

The practice and research of agile methods has become highly prevalent since the Agile Manifesto¹ was put forth almost ten years ago. In recent years a shift of focus from agile methods such as eXtreme Programming (XP) and Scrum to lean approaches in software development has been noticed and advocated. Claimed to be “the next wave of software process”², the adoption of lean approaches is believed to be a necessary progression for organisations planning to scale up agile, and can resolve issues agile methods fail to address [3]. More and more people advocate “from agile to lean”, and the combined use of agile and lean in software development. Other terms like “Lean Startup” or “Agile 2.0” have been coined to indicate this trend too [15, 19]. However, the phenomenon may not be as simple or linear as indicated by these terms. And our understanding of the combined use of agile and lean in software development is still limited.

The purpose of this study, consequently, is to investigate *how agile and lean approaches have been combined in software development*. To this end, an analysis of a set of experience reports, in which real world experiences of combining agile and lean are reported, was conducted. These

experience reports were published in the past Agile and XP conference series where they have been peer reviewed. Patterns of combining agile and lean that recurred in these experience reports were identified. The results are intended to be useful for both research and practice.

The remaining part of the paper is organized as follows. Section II reviews the relevant literature on the combination of agile and lean. This is followed by a description of the research approach employed. Then the findings are reported in the next section, and further reflected upon in the discussion section. The paper ends with concluding remarks.

II. COMBINING AGILE AND LEAN IN SOFTWARE DEVELOPMENT

Are agile and lean just two different names for the same thing? This is one view reflected in some literature. In the study reported in [8], the two are not distinguished. They have conducted a literature review of agile practices used in global software engineering. The search string “agile and lean” is used to denote agile practices, which indicates that they do not distinguish between the two. If this is the stance taken, then the combined use of agile and lean in software development, if happens, is generally a non-purposeful act of the adopting organisation.

Most literature however does consider the differences between agile and lean approaches, thus the potential to combine the two to make the most of both. One view of the difference between agile and lean is that lean and agile are at different levels. Lean is a philosophy or a set of principles, whereas agile is more at practice level [4, 6, 11, 14]. References [13] and [14] claim that lean thinking are principles that guide ideas and insights about software development discipline. Principles are underlying truths that do not change over time or space, while practices are the application of principles to a particular situation and should differ from one environment to the next and change as a situation evolves. They believe that lean development further expands the theoretical foundations of agile software development by applying well-known and accepted lean principles to software development.

Based on this view of agile and lean, references [13] and [14] suggest that one way of combining them is to use lean thinking as guiding principles to develop and adapt agile practices. They propose seven lean principles and translate them into agile practices tailored to individual software development domains. Reference [11] also sees that agile

¹ <http://www.agilealliance.org>

² <http://atlanta2010.leanssc.org/>

project management has roots in the lean thinking which provides strength and credibility to the concept and practice of agile project management. Reference [4] uses lean thinking as an analytical lens and provides a fresh look at the existing agile methods such as Scrum. Reference [4] claims that many organizations that have modified their software development system based on Scrum consider their work a lean implementation. Reference [4] argues that even in its simplest state, Scrum uses a lean 'pull' technique to smooth the flow through the system and prevent overloading. Scrum also implements a process for eliminating muda, or waste.

Another view of the difference between agile and lean does not regard lean and agile at different levels, but consider them having different scopes and focuses [1, 5, 6, 7, 12, 15, 16]. Reference [6] believes that agile methods are mostly concerned with the specific practice of developing software and the project management that surrounds that software development. They do not generally concern themselves with the surrounding business context in which the software development is taking place. Instead, lean principles can be applied to any scope, from the specific practice of developing software to the entire enterprise where software development is just one small part. The larger the scope, the larger the potential benefits. In addition, the primary focus of agile methods is on close customer collaboration and the rapid delivery of working software as early as possible whereas the primary focus of lean software development is on the elimination of waste in the context of what the customer values.

Based on this view of the difference between agile and lean, there are several different types of combination of the two suggested in the literature. Firstly, it is suggested that top-down implementation of lean is needed to create an environment where bottom-up agile adoption can thrive. Reference [7] believes that most agile methods are tactical in nature. The major changes required to become agile must be initiated from the top of the organisation. Organisational strategy becomes the context within which agile processes can operate effectively. Without this strategic piece, agile development is "shunted aside by the organisational forces that seek equilibrium" [7]. Major improvements require radical rethinking of the entire problem space. Reference [5] also suggests that agile and lean address different audience. They believe that XP describes a set of practices primarily thought for developers. It wants to ease the tension often existing between developer and customer because of conflicting aims. Instead, lean management is addressed to the upper management of a company and sees his task in the optimisation of the whole company. Therefore lean management is a top-down approach. It is considered not a slight optimization but a radical reorganisation of the whole company.

Another potential agile and lean combination is that lean can help agile to scale up because lean can be applied to the areas that agile is not suitable to. Reference [16] claims that "experience gathered during large scale implementation of agile concepts in software development projects teaches us that the currently popular agile software development methods (like Scrum) do not scale to program, product and

organisation level without change. The fundamentals for changes to these methods are found in lean principles, or: the future of agile methods is found in its origins." This claim is echoed by industry practitioners (reported in [15]) who believe that "lean is both the precursor and future of agile". Reference [16] presents a planning framework based on three core lean concepts - muri, mura and muda, which has been used successfully in large scale agile projects. Reference [1] claims that agile projects can and should be governed. However governance is not something that is commonly associated with agile software development projects, even though there are two aspects of agile software development that promote superior levels of governance: greater stakeholder visibility into the project and more opportunities for stakeholders to steer the projects, potentially making them accountable for the scope, budget and even schedule of agile project teams. A governance framework which is based on the seven principles of lean software development is built in [2]. At the core of the framework is that good governance should motivate and then enable IT professionals to do what an organisation believes to be the correct behaviours. The framework also identifies 18 practices which support the strategy. Reference [1] believes that lean governance enables agility at scale. Lean governance practices such as aligning the team structure with the architecture, risk-based milestones, and staged program delivery address complexities inherent in large or distributed teams. Other practices such as continuous project monitoring, integrated lifecycle environment, and embedded compliance help to address the additional complexity of regulated environments.

Yet another potential way to combine agile and lean is that agile software processes can be improved using lean practices. "Scrumban" is a term coined by [9, 10] and a good real-world example of the combination of agile and lean practices. It is a software production model based on Scrum and kanban. It is believed to be especially suited for maintenance projects or projects with frequent and unexpected user stories or programming errors. "In such cases the time-limited sprints of the Scrum model are of no appreciable use, but Scrum's daily meetings and other practices can be applied, depending on the team and the situation at hand" [9]. Using these methods, the team's workflow is directed in a way that allows for minimum completion time for each user story or programming error, but on the other hand ensures each team member is constantly employed. Along the same line, the study of [12] attempts to identify possible improvable parts in agile software development processes, and explores how lean practices could be used to improve them. A hypothesis "agile software process's development can be improved using lean practice techniques" is proposed in their study. They conducted a controlled experiment with university student projects to test the hypothesis and obtained positive evidences to support it. They conclude that applying lean techniques help stabilise the agile development phase especially in later stages of the phase.

Last but not least, reference [6] suggests that existing agile methods can be used as valid supporting practices of

lean software development. They claim that, when implementing Lean software development, “it is quite common to pick a light weight agile methodology as a starting point and begin applying other lean tools (such as value stream mapping) from there”. They recommend using Scrum since it provides all the essentials and has a very low learning curve.

Table 1 below is a summary of the contrast of agile and lean as well as the combination types suggested in the reviewed literature.

TABLE I. DIFFERENT COMBINATIONS OF AGILE AND LEAN IN SOFTWARE DEVELOPMENT

Perception of the Difference between Agile and Lean	Combination Type
No perceived difference between agile and lean	The combination of both is non-purposeful
Agile and lean are at the different levels. Lean is thinking tool, agile is prescriptive practice	Use lean principles to guide the development and adaptation of agile practices
Agile and lean are at the similar levels, but have different scopes and focus	Top-down implementation of lean to create environment where bottom-up agile can thrive
	Use lean to help agile scale up
	Use lean techniques to improve agile software development processes
	Use agile practices to support lean software development processes

III. RESEARCH APPROACH

To answer the research question *how agile and lean approaches have been combined in software development*, secondary data analysis was used as the main research method in this study to analyse real world cases that have combined the two approaches. Secondary data analysis is the analysis of data that was either collected by individuals other than the researchers that conduct the study, or for some other purpose than the one currently being considered, or often a combination of the two. The sources of secondary data include newspapers, census data, maps, etc. The advantage of using secondary data is that the data collection process can be unobtrusive, fast and inexpensive, even though it needs to be cautioned that there are issues related to data quality control, accuracy of data, etc., which need the attention of the researchers. Secondary data analysis is frequently used in social science research. If it is undertaken with care and diligence, it can provide a cost-effective way of gaining a broad understanding of research questions. It is also often considered a starting point for other research methods, helpful in designing subsequent primary research and can provide a baseline with which to compare the primary data analysis results [17]. Given the exploratory nature and early stage of the study, secondary data analysis is considered a

feasible way to build initial understanding of the phenomenon under the study.

To obtain the secondary data needed for this study, the experience reports, which have been published in the agile related conferences since 2000 and are publically available, were collected. As suggested in the wikipedia page of agile software development³, agile development has been the subject of several conferences, and some of these conferences have published peer-reviewed experience reports which share industry experiences of agile software development. The experience reports that were published in the XP conference series, Agile conference series and XP/Agile Universe were scanned. Those experience reports that contain explicit evidences of using both agile and lean approaches were included in the analysis. Totally there are 23 experience reports that satisfy the selection criteria and are included in the secondary data analysis (see Appendix for the list of these experience reports). Table II shows the distribution of these experience reports in the past conferences. These experience reports were imported and coded in NVivo, using Table I as the starting coding schema.

TABLE II. DISTRIBUTION OF THE SELECTED EXPERIENCE REPORTS

Conference	Number of experience reports selected
Agile 2004	2
Agile 2006	4
Agile 2007	3
Agile 2008	7
Agile 2009	4
XP2010	3

Agile 2003, Agile 2005, XP 2003 – 2009, XP/Agile Universe 2002 – 2004: No experience reports containing agile and lean combination; Agile 2010: Experience reports not available online; XP 2000 – 2002, XP Universe 2001: Proceedings not available online.

IV. FINDINGS

The 23 experience reports identified report on different aspects of software development in 22 companies (two experience reports regard the same company). The size of these companies ranges from small and medium to large and multi-national. They operate in diverse business domains, such as telecommunication, finance, healthcare and governmental bodies. Both agile and lean approaches have been used in these companies, in various manners and to various degrees of combination. In some companies, the combined use of agile and lean is limited to a single, or several development teams or projects. In others instead, it is a company wide endeavor. The types of agile and lean combination in these companies were classified using both the predefined categories in Table I and the emergent categories through the data analysis. In this section how agile and lean are combined in each company is described under these categories.

³ http://en.wikipedia.org/wiki/Agile_software_development

A. Non-purposeful Combination

Non-purposeful combination of agile and lean is evidenced in 3 experience reports analysed [ER1, ER2, ER3]. In these reports lean and agile are generally mentioned in parallel and no specific distinction between the two is made.

Healthwise is a non-profit organization that licenses its consumer-facing decision support content and software solutions to healthcare organizations. In Healthwise Scrum and lean principles are used to manage the multi-disciplinary product development teams [ER1]. Scrum and lean are referenced together in [ER1] without further elaboration on what Scrum or lean principles are respectively. In addition, it is claimed that the success of Scrum “*allowed the practice to evolve into a process resembling lean manufacturing*” (automatically). This claim is an interesting contrast to the experience in the other companies [ER20, ER21, ER22] where the transition from Scrum to flow-based lean processes is intentional and effort-taking.

Similarly, in the account of an internet site company BabyCenter’s journey from chaos to a relatively mature agile organization, [ER2] claims that “*through the adoption of agile and lean practices, and Scrum in particular, BabyCenter was able to build a production environment with 99.99% uptime and increase shared knowledge while fostering a collaborative environment*”. Again agile and lean practices are listed in parallel without further elaboration, even though the benefit of agile and lean combination seems impressive.

The reporter of [ER3] presents the agile practices that he has practiced in a project where he worked as a technical lead. The practices in his armory include Keep/Problem/Try, Estimate Retrospectives, Positive Strokes, Iteration Planning, Darts, Task Kanban, and Overtime Tickets. Evidently he is practicing a combined approach of agile and lean, even though he regards them as agile practices in general. It is interesting to mention that the Japanese background of the development team may explain the natural and implicit adoption of lean practices.

B. Lean principles guiding agile practices

Lean principles are used to make sense and guide the development and adaptation of agile practices in several organizations [ER4, ER7, ER8, ER9, ER10].

The incremental adoption of agile practices in the Government Workflow Project, a project initiated by the government of a major California county to automate the workflow of key business processes in the criminal justice system, is described in [ER4]. The project team ended up performing more up-front analysis and using small batch size for estimations as a response to frustrating velocity fluctuations and inconsistent completion of features. Initially the team had doubt that performing additional up-front analysis was against agile principles. However, “*in retrospect, the team realized this practice implements the ‘smaller batch size’ principle of Lean Software Development, and in fact increased their agility*”. Notice that lean can be used as a sense making tool retrospectively after a practice is adapted.

One of the lean principles, eliminating waste, is a recurrent theme in [ER7, ER8, ER9, ER10]. In [ER7], Sabre Airline Solutions (A.S.), a company developing products for the airline industry, encountered agile plateau effect. In the effort of overcoming the plateau, the company chose to apply lean concepts, such as seeing waste and value chain mapping, to brainstorm the ways to improve quality, and develop quality improvement goals and action plans accordingly. [ER8] examines the use and adaptation of the “product owner team” practice in BT, British Telecom. Product owner team is used to manage the details of what should be built in a project implementing an up to 24Mbps service over the 21CN network. The lean principle of eliminating waste and the seven types of waste help the company to “*break the silo mentality and simplify the delivery by reducing the work in progress and developing collaborative teams focused on customers’ prioritized needs*”. It is believed that the project was successful due to “*the collaborative efforts of the core team to eliminate waste – applying one of the core principles of lean*”. Similarly, in a mission-critical commercial-off-the-shelf upgrade project described in [ER9], lean thinking, especially eliminating waste, is used as the guidance to the improvement of the manual testing process which is crucial to the project.

More lean principles have been applied in a distributed team of 35 developers spanning 5 continents in Canonical [ER10]. The lean principles that play an important role in the experience of this highly distributed agile team include end-to-end view of the process, removing bottlenecks in the process, and Kaizen where process improvement experiments are encouraged.

C. Lean principles facilitating agile adoption

Lean principles and thinking tools have been utilized to facilitate agile adoption in several organizations [ER5, ER6, ER11, ER17, ER18].

[ER5] describes an interesting experience of the bottom-up, self-funded agile adoption in an engineering consultancy and software company that is involved in the automotive industry “*where analogous concepts under the umbrella term ‘lean’ are part of the landscape*” and some of the companies they do business with were the originators of lean. Therefore it is natural that the driving force behind a lot of what they did came from an awareness of lean. According to the report, a self funding transformation is one of the key concepts they “stole” from lean. Another concept that they borrow from lean is that of cost accounting, viewing your software team as a fixed cost overhead. In addition, what inspires them to bootstrap agile practices themselves is the “inspect and adapt” cycle. Therefore, the success of this self-funded agile adoption can be attributed partially to the guidance of these lean concepts.

In contrast to the bottom-up, self-funded agile adoption described in [ER5], Salesforce.com took a completely different approach [ER6]. The company has completed an agile transformation of a two hundred person team within a three-month timeframe. During this large and fast “big-bang” agile rollout, lean principles, such as empowered

teams and delivering customer value early, were used as key communication tools to communicate the value of changing current behavior. If teams were feeling that something was not working the way it should be, they could refer back to the values and reject anything they thought did not correlate with their core values. Lean principles help agile behaviors to stick.

Capital One, a large Fortune 500 financial services company used the lens of lean to evaluate the current delivery process and streamline the business values prior to agile transition [ER11]. Lean principle “Eliminate Waste” and tools value stream mapping, root-cause analysis (5-whys) were applied to analyse the processes of a product or service type before Scrum was chosen as the adopted agile method. According to the experience of the company, lean tools and principles can help achieving a smoother transition and will increase the likelihood of a successful agile pilot with tangible business metrics.

This claim is supported by the experience of Systematic, an independent software systems company, which is described in two experience reports [ER17, ER18]. In [ER17] Systematic made a strategic decision to use lean as the dominant paradigm for future improvements after achieving CMMI level 5 and identified Lean Software Development as the lean dialect most relevant to Systematic. The analysis of systematic improvement opportunities and lean causal dependencies led to the decision to seek improvements based on the Lean Software Development principles of Build Integrity In, Amplify Learning and Deliver Fast. These lean thinking tools gave inspiration to consider Scrum and early testing. The result of the pilots confirmed the general idea of using lean mindset as source for identification of new improvements. Actually that is what happened in Systematic later on, which is the main subject of the second experience report [ER18].

Till this section lean is seen mainly to be applied as sense making principles in the organizations. Instead, the following sections will show that not only lean principles are applied as a thinking tool in agile software development, but concrete lean techniques are used alongside agile practices in quite several organizations.

D. Lean facilitating agile to scale up

4 experience reports have demonstrated how lean can help extending agile development to better collaborate with different stakeholders of software development, including neighboring business units and customer, and extending agile approach to non-development, organizational level activities.

In the Cardiac Rhythm Disease Management of Medtronic Inc., when the software group adopted agile development, they quickly realized that not only they were learning something new, but they also need to learn how to communicate it with other business units [ER12]. Lean concepts and principles become a convenient choice as a communication tool with the production development organisation since the product development organisation has already implemented a lean initiative. Therefore when communicating with production development organisation,

the software group emphasized the principles of Eliminating Waste and Inspect and Adapt, meanwhile implemented practices like the Customer Role, Stories, Sprint Planning and Release Planning to put focus on value-added activities.

The experience of Ericsson R&D in the Netherlands suggests that agile software development should be implemented as a broader “lean” initiative [ER14], which will create involvement of neighbouring units, like e.g. service and delivery units, product management, market units and customers. This foundation will be an incentive for neighbouring units to cooperate and optimize as a whole, and the resistance to collaborate with agile development will be reduced effectively. Their experience also suggests that lean can help align management quickly:

“In a bottom-up approach line and project management have limited involvement. However, line and project managers are a key in making the change stick, helping people resolve impediments and conflicts, building a learning organization and, above all, showing what is meant with Agile and lean ways of working. Speaking the same language and agreeing on principles like ‘build quality in’ and many more is highly needed for successful cross unit cooperation.” [ER14]

[ER13] contains an interesting case of a company applying lean thinking and systems thinking on the customer business process in order to understand what features the developers should develop that really deliver business values. Cyrus Innovation is a consulting group in New York City specializing in agile software development, usability design and operational consulting. One of their customers – a restaurant chain - needed to improve aspects of their operations that were breaking due to their rapid growth. Their main objective was reducing operating expenses by cutting down on the food product wasted each day. Cyrus Innovation uses XP for development and strongly believes in the power of agile development; however they see that agile alone is insufficient to make every software project successful:

“XP customer will tend to drive development without regard to how features impact the business system from a throughput perspective.... If business managers do not adopt analytic techniques that are more synergistic with XP, the developers themselves simply become a local optimization of the software development process. As a result, XP by itself cannot drive overall system improvement and thus by itself cannot make a company sustainable.” [ER13]

Projects must be coupled with a complimentary approach to strategy in order to achieve the overall business goals. Rather than accepting whatever the customer demand to deliver, the project team used lean thinking and techniques, including eliminating waste, value stream mapping, flow, and theory of constraints, to better understand what really deliver business value to the customer. Then the development team use agile development to build quality software that effectively contributes to overall business success.

In DTE Energy, a Fortune 300 company, agile software development and project management encountered the legacy mindset and culture of portfolio management [ER15], which means that the annual budgeting cycle drove a mindset that scope, budget, and schedule must be established up front, often many months before project work begins. Success meant delivering on that scope within the budget and schedule commitments. As the IT teams successfully applied agile methods at the project level, they began to address their approach for managing portfolios of projects to increase the amount of value they delivered with their business partners. Lean training was organised for the leadership team, and potential applicable lean techniques identified. They also introduced lean terminology and concepts to help better understand the constraints and how they could reorganize the way they prioritized their commitments and funded their work. As the way forward, one overarching strategy in their IT was to leverage an existing corporate system which is a combination of lean and Six Sigma thinking, seeing, and doing tools and techniques based upon the Plan-Do-Check-Act cycle.

E. Combining agile and lean at the principle level

The case of [ER16] indicates that agile and lean can be combined at the principle level. In Wireless Data Services Global, a service provider to wireless companies and mobile phone manufacturers, the development teams “*have experienced tremendous positive effects from utilizing Extreme Programming practices on development teams*”. However, they “*have yet to find the agile path to regularly providing positive business value*”. Over the time, they have found a family of four agile practices that merged the XP principles of implementing the “highest value features first”, and “don’t do anything extra”, with lean principles such as “eliminate waste” to address the highlighted issues. The four practices - Value-based Investment Decisions, High Confidence Stories First, Incremental Story Delivery, and Story Ownership - embody both agile and lean principles and are believed to be most effective when applied together.

F. Combining agile and lean at the practice level

Agile and lean can be combined at the practice level, which is evidenced in several organizations [ER19, ER20, ER21, ER22, ER23].

One way of combining the two is to use lean techniques to improve agile software development processes, or it can be termed “lean within agile”. [ER19] reports the projects in the IT department of ASR Insurance, one of the top 3 insurance companies in the Netherlands. While most projects that used Scrum were successful, other Scrum teams were having some difficulties. The major reason was that these projects most of the time were involved in operations work or small maintenance. The work was hard to distribute properly over sprints and often needed to be changed more frequently than the 2-week sprints allowed. What the client wanted was more flexibility and more control over the immediate results. The company wanted to keep the agile mindset and at the same time do something more appropriate for maintenance and operations so that they too can

cooperate with the rest of the IT departments and projects. Kanban technique was adopted, together with the underlying principles - make work visible, limit work in progress and help work to flow. As a result, much better understanding and cooperation between developers from different technologies as well as with the testers was observed, even though the team encountered different team and organisational challenges.

The story of SEP is a bit different from that of ASR Insurance. SEP is a privately held software engineering company with more than 70 employees. It offers full lifecycle software solutions to clients in the medical, aerospace, healthcare and national defense markets. In the process of adopting one of the agile methods, Feature Driven Development, SEP found that it failed to have the desired lasting impact across the entire organization. However, things changed when a kanban system was implemented later on alongside the agile practices. Kanban provided a more effective vehicle to introduce agile practices and principles in the company. The culture on project teams began to change as they learned the system. And more importantly, the attitudes of team members changed. The implementation of kanban helped the agile mindsets to stick in the company.

If the experience of ASR Insurance and SEP is the example of “lean within agile”, the next three cases [ER20, ER21, ER22] show that, starting with the application of lean techniques to improve agile processes, the organizations ended up in a situation that can be termed as “agile within lean” in which lean processes become dominant and agile practices play the supporting role.

Inkubook is an emerging player in the online photobook industry. The journey of the Inkubook team to improve their software development process is documented in [ER20]. The team initially adopted Scrum “by the book”. However, they entered into a chaotic no process stage when the mandate arrived that a product would be delivered in sixty days. When the schedule slowed down again, in order to avoid burnout and staff turnover, the team moved back to Scrum, but this time it was really a “*flow-based, iterationless version*” of Scrum. After three months of being in the same “sprint”, the team recognized that flow was working well for getting things done and therefore they abandoned estimates, implicitly organizing around a WIP (Work In Progress) limit of two MMFs (Minimal Marketable Feature Sets). Finally, after several more months passed, the team accepted that they were using a work-limited, pull-based kanban approach and updated the usage and terminology to reflect that fact. According to the team, “*the use of a kanban implementation survived a round of layoffs, an extreme change in team and management composition, and is still being used today*”.

A similar case is reported in [ER21]. The title “*From Chaos to Kanban, via Scrum*” illustrates the evolutionary path of the software development team in Codeweavers. The company is a UK business of approximately 20 people, delivering motor finance and insurance web services. The team comprises 8-10 co-located developers. Using simple inspect-and-adapt cycle to adopt one practice a time, the team adopted Scrum practices first to tackle the chaos, then

used value stream mapping and adopted kanban board to have a better visibility of upstream and downstream activities other than work in progress on the story board. Step by step following the kanban adoption, the team introduced limit to WIP, fine tuned it while the team began batching tasks into MMFs. Along the way, the team also adopted automated regression test and adapted stand-up meeting to hold it twice per day to ensure tasks were worked on as things developed. At the end, the development process assembles more a flow-like lean process rather than a time-boxed Scrum process. Meanwhile, as the focus on flow and throughput became more deeply ingrained in the development team, the developers' view of the value stream gradually increased, from focusing only on the "in development" column to downstream to ensure the code developed was accepted and deployed to customers. Later still they looked further downstream, helping customers to adopt the new services, and upstream, helping the business to decide what was needed and how to prioritise it. The focus was expanding even further into the company's sales and marketing functions.

If the transition of the Codeweavers team from agile to lean is an incremental, unplanned process, the experience documented in [ER22] is a more systemic move. The software development team (not named in the report) had run a Scrum-based development process for several years. The practices included continuous integration, automated, nightly build and deployment to QA servers, and suites of automated unit and integration tests. In spite of the processes and practices in place, the team was still challenged by the issues such as frequent mid-iteration changes and non correlated work items. That is the reason why the team decided to embark on a lean journey. After a careful research, the team arrived at a set of core concepts for their flow-based development: schedule individual, value-adding work items, define a workflow, limit work in process, same-size work items, establish holistic key performance indicators, visualize all the work and the entire workflow, and improve relentlessly. The team found that *"it was fully possible to run agile software development without time-boxes by using a continuously updated work item priority queue instead"*. The team's experience shows that the WIP limit is a good control variable compared to controlling capacity and scope of work under high variability, and a single WIP limit (CONWIP) works well for the team.

V. DISCUSSION

The cases described in the findings section show that there can be different ways to combine agile and lean in software development, either non-purposefully, or with specific objectives or approaches. Some companies have combined agile and lean as described in Table I, others illustrate different and new types of combination. One type of combination in Table I, top-down implementation of lean can create environment where bottom-up agile initiatives can thrive, has no corresponding cases in the experience reports analysed.

Built on Table I, Table III provides an overview of the different types of agile and lean combination identified in this study.

TABLE III. DIFFERENT COMBINATIONS OF AGILE AND LEAN IN SOFTWARE DEVELOPMENT IDENTIFIED IN THE STUDY

Perception of the Difference between Agile and Lean	Combination Type
No perceived difference between agile and lean	The combination of both is non-purposeful
Agile and lean are at the different levels. Lean is thinking tool, agile is prescriptive practice	Use lean principles to guide development and adaptation of agile practices
	Use lean principles to facilitate agile adoption
Agile and lean are at the similar levels, but have different scopes and focus	Top-down implementation of lean to create environment where bottom-up agile can thrive
	Use lean to help agile scale up
	Both can be combined at the principle level to develop and adapt practices
	Both can be combined at the practice level
	Use lean techniques to improve agile software development processes ("lean within agile")
	↓
	Use agile practices to support lean software development processes ("agile within lean")

As described in the findings section, 10 out of 23 experience reports contain the evidence of using lean as guiding principles to guide the practice of agile software development. It is evident from the analysis of these experience reports that lean principles are not only guiding the development and adaptation of agile practices (as described in Table I), but also facilitate agile adoption initiatives and can be used as a communication tool within agile teams as well as among different stakeholders of the organization. Table III adds the additional type of combination to reflect what has been learnt from the experience reports analysis.

If agile and lean are regarded at the similar levels, then the two can be combined at either the principle level, or the practice level. More cases of combining the two at the practice level were found. When combined at the practice level, depending on the starting point of the organization, there can be two different ways of combining agile and lean practices. Lean techniques can be applied to improve agile software development processes ("lean within agile"). Or vice versa, existing agile methods can be used as valid supporting practices of lean software development ("agile within lean"). What is found from the experience reports

analysis is that, when an agile adopting organisation is mature enough in using agile and especially lean practices, they have tendency to move away from time-boxed agile processes and stick to flow-based lean processes, as several experience reports have shown ([ER20, ER21, ER22]). That is the meaning of the arrow between the two types. It is interesting to note that these experience reports were the most recently published ones in the collection. This might indicate that moving from time-boxed agile processes to flow-based lean processes is the most recent tendency in agile software development.

Even though there is no built-in time dimension in Table III to denote the timeline for different types of agile and lean combination, a discernable pattern across the experience reports is that, in the early days, lean was used more as a thinking tool, and in a less conscious manner; in more recent years instead, more and more concrete lean practices, especially kanban, are adopted in software development processes, as a conscious choice of the adopting organizations.

VI. CONCLUSION

The recent focus shift from agile methods such as XP and Scrum to lean approaches in software development has been noticeable and evokes the interests of both research and practice alike. The objective of this study was to investigate how agile and lean approaches have been combined in software development. To explore this phenomenon, a secondary data analysis of 23 experience reports containing real-world experiences of combining agile and lean in software development was conducted. The patterns of combining agile and lean in these experience reports were identified and categorized in a more systemic way.

The findings of the study would enrich our understanding of how agile and lean can be combined in software development. Since the research on the broad topic of lean software development is considered a nascent area [18], this study can be an important addition to this branch of research in general, and on the topic of combining agile to lean in specific. The combination types identified in the study can serve as a thematic map for the researchers who intend to conduct more in-depth study of the phenomenon of agile and lean combination. The practical implication of the study is that it reveals different ways to combine agile and lean. There is no one-type-fits-all solution. Each organization should reflect on its own situation and needs before embarking on the journey of combining agile and lean approaches. Then the potential combinations summarised in this study could provide them with some promising directions to explore. However, how to effectively tailor the combination types to suit the specific situation and needs of the organisation is a challenge yet to be addressed satisfyingly and worth further studying.

One main limitation of the study is related to the secondary data analysis method applied. Since the collected experience reports represented secondary data, the researcher had no control on the quality of data and especially the level of details desired. The 23 experience reports included in the analysis therefore were not equally informative and

illuminative. This is a potential threat to the validity of the findings. Therefore, one potential venue for future research would be to conduct primary case studies to collect more specific and in-depth data on different types of agile and lean combination, to explore the issues, challenges and opportunities associated with the combined use of agile and lean in software development. This will help achieve a better definition of the combination types presented in the paper, and clarification of the distinctions between them. A more solid conceptualization of agile and lean will greatly benefit the investigation.

Another interesting study is to bring the analysis presented in the paper one step deeper and analyse specific agile and lean practices that are possibly disjoint, to reveal how and why these practices are related and enacted in different organizational contexts.

APPENDIX: THE LIST OF EXPERIENCE REPORTS INCLUDED IN THE SECONDARY DATA ANALYSIS

- [ER1] K. Long and D. Starr, "Agile Supports Improved Culture and Quality for Healthwise," *Proceedings of Agile 2008. AGILE '08. Conference*, Toronto, ON: IEEE Computer Society, 2008, pp. 160-165.
- [ER2] K. Nottoson and K. DeLong, "Crawl, Walk, Run: 4 Years of Agile Adoption at BabyCenter.com," *Proceedings of Agile 2008. AGILE '08. Conference*, Toronto, ON: IEEE Computer Society, 2008, pp. 116-120.
- [ER3] F. Kinoshita, "Practices of an Agile Team," *Proceedings of Agile 2008. AGILE '08. Conference*, Toronto, ON: IEEE Computer Society, 2008, pp. 373-377.
- [ER4] P. Hodgetts, "Refactoring the Development Process: Experiences with the Incremental Adoption of Agile Practices," *Proceedings of the Agile Development Conference (ADC'04)*, IEEE, 2004, pp. 106-113.
- [ER5] D. Poon, "A Self Funding Agile Transformation," *Proceedings of Agile Conference 2006*, Minneapolis, MN: IEEE Computer Society, 2006, pp. 342-350.
- [ER6] C. Fry and S. Greene, "Large Scale Agile Transformation in an On-Demand World," *Proceedings of Agile 2007*, Washington, DC: IEEE Computer Society, 2007, pp. 136-142.
- [ER7] J. Packlick, "The Agile Maturity Map A Goal Oriented Approach to Agile Improvement," *Proceedings of Agile 2007*, Washington, DC: IEEE Computer Society, 2007, pp. 266-271.
- [ER8] A. De Ste Croix and A. Easton, "The Product Owner Team," *Proceedings of Agile 2008. AGILE '08. Conference*, Toronto, ON: IEEE Computer Society, 2008, pp. 274-279.
- [ER9] A. Geras, "Leading Manual Test Efforts with Agile Methods," *Proceedings of Agile 2008. AGILE '08. Conference*, Toronto, ON: IEEE Computer Society, 2008, pp. 245-251.
- [ER10] F.J. Lacoste, "Killing the Gatekeeper: Introducing a Continuous Integration System," *Proceedings of Agile Conference 2009. AGILE '09.*, Chicago, IL: IEEE Computer Society, 2009, pp. 387-392.
- [ER11] E. Parnell-Klabo, "Introducing Lean Principles with Agile Practices at a Fortune 500 Company," *Proceedings of Agile Conference 2006*, Minneapolis, MN: IEEE Computer Society, 2006, pp. 232-242.

[ER12] K. Weyrauch, "What Are We Arguing About? A Framework for Defining Agile in our Organization," *Proceedings of Agile Conference 2006*, Minneapolis, MN: IEEE Computer Society, 2006, pp. 213-220.

[ER13] C. Rand and B. Eckfeldt, "Aligning Strategic Planning with Agile Development: Extending Agile Thinking to Business Improvement," *Proceedings of the Agile Development Conference (ADC'04)*, Salt Lake City, UT: IEEE Computer Society, 2004, pp. 78-82.

[ER14] J. Goos and A. Melisse, "An Ericsson Example of Enterprise Class Agility," *Proceedings of Agile 2008. AGILE '08. Conference*, Toronto, ON: IEEE Computer Society, 2008, pp. 154-159.

[ER15] J.C. Thomas and S.W. Baker, "Establishing an Agile Portfolio to Align IT Investments with Business Needs," *Proceedings of Agile 2008. AGILE '08. Conference*, Toronto, ON: IEEE Computer Society, 2008, pp. 252-258.

[ER16] M. Yap, "Value Based Extreme Programming," *Proceedings of Agile Conference 2006*, Minneapolis, MN: IEEE Computer Society, 2006, pp. 175-184.

[ER17] J. Sutherland, C.R. Jakobsen, and K. Johnson, "Scrum and CMMI Level 5: The Magic Potion for Code Warriors," *Proceedings of Agile 2007*, Washington, DC: IEEE Computer Society, 2007, pp. 272-278.

[ER18] C.R. Jakobsen and J. Sutherland, "Scrum and CMMI Going from Good to Great," *Proceedings of Agile Conference 2009. AGILE '09.*, Chicago, IL: IEEE Computer Society, 2009, pp. 333-337.

[ER19] O. Maassen and J. Sonneveld, "Kanban at an Insurance Company (Are You Sure ?)," *Proceedings of the 11th International Conference on Agile Software Development, XP2010*, Trondheim, Norway: Springer Verlag, 2010, pp. 297-306.

[ER20] E.R. Willeke, "The Inkubook Experience: A Tale of Five Processes," *Proceedings of Agile Conference 2009. AGILE '09.*, Chicago, IL: IEEE, 2009, pp. 156-161.

[ER21] K. Rutherford, P. Shannon, C. Judson, and N. Kidd, "From Chaos to Kanban, via Scrum," *Proceedings of the 11th International Conference on Agile Software Development, XP2010*, Trondheim, Norway: Springer Verlag, 2010, pp. 344-352.

[ER22] J.O. Birkeland, "From a Timebox Tangle to a More Flexible Flow," *Proceedings of the 11th International Conference on Agile Software Development, XP2010*, Trondheim, Norway: Springer Verlag, 2010, pp. 325-334.

[ER23] C.M. Shinkle, "Applying the Dreyfus Model of Skill Acquisition to the Adoption of Kanban Systems at Software Engineering Professionals (SEP)," *Proceedings of Agile Conference 2009. AGILE '09.*, Chicago, IL: 2009, pp. 186-191.

ACKNOWLEDGMENT

This work is supported, in part, by Science Foundation Ireland grant 03/CE2/1303_1. Thanks to Kieran Conboy for his valuable suggestions for the revision of the paper.

REFERENCES

- [1] S.W. Ambler, "Scaling agile software development through lean governance," *Proceedings of Software Development Governance SDG'09 - ICSE'09 Workshop*, Vancouver, Canada: IEEE Computer Society, 2009.
- [2] S.W. Ambler and P. Kroll, "Applying agile and lean principles to the governance of software and systems development," *IBM Software Rational*, 2009.
- [3] D. Anderson, "Business Drivers for Kanban Adoption," *Proceedings of Lean Software & Systems Conference*, Atlanta: 2010, pp. 7-14.
- [4] B. Barton, "All-Out Organizational Scrum as an Innovation Value Chain," *the 42nd Hawaii International Conference on System Sciences - 2009*, Waikoloa..
- [5] M. Dall'Agnol, A. Janes, G. Succi, and E. Zaninotto, "Lean Management - A Metaphor for Extreme Programming?," *Proceedings of the 4th International Conference XP2003*, Genova, Italy: Springer Verlag, 2003, pp. 26-32.
- [6] C. Hibbs, S. Jewett, and M. Sullivan, *The Art of Lean Software Development: A Practical and Incremental Approach*, O'Reilly Media, Inc., 2009.
- [7] J. Highsmith, *Agile software development ecosystems*, Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 2002.
- [8] S. Jalali and C. Wohlin, "Agile Practices in Global Software Engineering - A Systematic Map," *Proceedings of the 5th International Conference on Global Software Engineering (ICGSE 2010)*, IEEE Computer Society, 2010, pp. 45-54.
- [9] C. Ladas, "Scrum-ban," *Lean Software Engineering-Essays on the Continuous Delivery of High Quality Information Systems*, 2008.
- [10] C. Ladas, *Scrumban - Essays on Kanban Systems for Lean Software Development*, Modus Cooperandi Press, 2009.
- [11] R. Morien, "Agile management and the toyota way for software project management," *Proceedings of the 3rd IEEE International Conference on Industrial Informatics, (INDIN '05)*, Perth, Western Australia: IEEE Computer Society, 2005, pp. 516-522.
- [12] G.I.U.S. Perera and M.S.D. Fernando, "Enhanced agile software development - hybrid paradigm with LEAN practice," *Proceedings of 2nd International Conference on Industrial and Information Systems (ICIIS 2007)*, Peradeniya, Sri Lanka: IEEE Computer Society, 2007, pp. 239-244.
- [13] M. Poppendieck and T. Poppendieck, *Implementing Lean Software Development From Concept to Cash*, Addison Wesley Professional, 2006.
- [14] M. Poppendieck and T. Poppendieck, *Lean software development: an agile toolkit*, Addison Wesley Professional, 2003.
- [15] K. Serignese, "A sprinkle of agile, a dash of lean," *SPTechWeb*, 2011.
- [16] H. Smits, "The Impact of Scaling on Planning Activities in an Agile Software Development Center," *Proceedings of the 40th Hawaii International Conference on System Sciences (HICSS'07)*, Waikoloa..
- [17] S. Boslaugh, "An Introduction to Secondary Data Analysis," *Secondary Data Sources for Public Health: A Practical Guide*, Cambridge University Press, .
- [18] T. Dingsøy, T. Dybå, and P. Abrahamsson, "A Preliminary Roadmap for Empirical Research on Agile Software Development," *Proceedings of Agile 2008 Conference*, Toronto, 2008, pp. 83-94.
- [19] D. Norton, "Agile 2.0 - The Rise of Enterprise Agile," *Proceedings of SOA & Application Development and Integration Summit*, London, 2010.