First Steps in Innovative Tool Support for Decision Making in Agile Software Development

Fearghal McHugh\textsuperscript{a} and Thomas Acton\textsuperscript{b}

\textsuperscript{a}. The Whitaker Institute, J.E. Cairnes School of Business & Economics, & LERO, the Irish Software Engineering Research Institute, National University of Ireland Galway, Ireland, Email: fearghal.mchugh@nuigalway.ie.

\textsuperscript{b}. Business Information Systems, J.E. Cairnes School of Business & Economics, & LERO, the Irish Software Engineering Research Institute, National University of Ireland Galway, Ireland, Email: thomas.acton@nuigalway.ie

Abstract

The fast, unstructured and turbulent nature of decision making in agile software development (ASD) teams at times necessitates software-based support tools (Dabrowski, Acton, Drury, Conboy, and Dabrowska, 2011). The demands on development teams are increasing in terms of delivery expectations, and therefore requires innovation in how decision support tools are used. Decisions in agile teams are intense, with teams iteratively producing customer-focused software in short bursts of time. Decision support tools may facilitate an increased innovative capacity in the team, and by extension, the software development organisation, through the provision of competitive advantage. This paper provides an initial ‘first steps’ exploration of tool support for decision making in agile software development, outlines the major aspects of agile processes requiring such support, and proposes a way forward for future work.

Keywords: innovation, decision making, software tools, decision support

Introduction

Decisions in agile teams are intense, with teams iteratively producing customer-focused software in short bursts of time. Decision making generally in Agile Software Development is demanding. Demands on agile software development teams are increasing in terms of
delivery expectations, and therefore require innovation in how decision support tools are used. Innovation is required in decision support systems to deal with the demand in ASD and for the software development industry to continue to evolve. Agile methods have been in use for some time but it is important to examine the issues and implications of using decision support systems in ASD. Decision support software tools have evolved over many decades, and their direction is towards systems that can deal with multiple data types and from multiple data sources. For these reasons we provide a brief description of the background and progress of decision support systems. Subsequently, we outline core decision categories to indicate key areas in which decision support systems can contribute, and we identify the crucial issues for decision making in ASD. A collection of current ASD decision tools are then included, lending to the identification of need for a more integrated framework to collate the various functions of the myriad of tools that are used to support innovative development of a comprehensive decision support tool framework for ASD. The four sections outlined below, to help understand the need for ASD decision tool framework, are decision support tools, issues in ASD, the concept of decision making, the decision categories in ASD and the current ASD decision support tools.

**Decision making in agile environments**

In software development, “the market demands and expects innovative, high quality software that meets its needs. Agile methods are a response to this expectation. Their strategy is to reduce the cost of change throughout a project. Extreme Programming (XP), for example, calls for the software development team to produce the first delivery in weeks, to achieve an early win and rapid feedback; invent simple solutions, so there is less to change and making those changes is easier; improve design quality continually, making the next story less costly to implement; and test constantly, for earlier, less expensive, defect detection. Agile software
development stresses quality in design. Each agile method addresses quality in certain ways. Agile approaches recommend short iterations in the two- to six-week range during which the team makes constant trade-off decisions and adjusts to new information.” (Highsmith, Cocburn, 2001). In adopting and using any software development process, it is critical to ensure that it addresses the full development cycle. “To be successful at IT you must take a multi-system, multi-life cycle stage view and the reality is that organisations have many potential projects in the planning or iterations stage, many in development, and many in production,” (Ambler, 2011).

Agile software development requires alignment of decisions on the strategic, tactical, and operational levels in order to overcome challenges. Agile development also requires a transition from specialised skills to redundancy of functions and from rational to naturalistic decision-making. This takes time; the case companies needed from one to two years to change from traditional, hierarchical decision-making to shared decision-making in software development projects. (Moe, Aurum, Dyba, 2012). Software development is ever changing, and quality decision making in turbulent and rapidly changing agile development scenarios requires support. Indeed, effective software development relies on quality decisions. There is a need to understand how these decisions are made and how decisions can be supported in agile software development environments to improve the quality and timeliness of team outputs (Drury, Conboy, Acton., 2012). There are multiple dimensions to decision quality in decision making, exacerbated in the agile context. In traditional systems development, decisions lie with the manager. In contrast, “Agile software development teams are, in theory, empowered to make decisions, while the role of manager has changed from one of command and control (i.e. to make decisions and ensure they are implemented) to one of a facilitator. Decision making in software development is not characterised by a sequence of isolated or exclusive decisions; rather, decisions are inter-related, with each decision leading to further
decisions, the chain of which often spans the entire duration of a project. Over this extended period, there are several potential factors that can negatively affect the efficacy of decision making by Agile teams. … High level of empowerment of a cohesive software development team undertaking an Agile project may be one of these negative factors, as empowered, cohesive teams can exhibit problems such as groupthink or the Abilene Paradox.” (McAvoy, Butler, 2009). McAvoy and Butler (2009) argue that the decision making process has changed, “that the role of project manager in Agile development initiatives needs to be reassessed, with project managers taking on the role of devil’s advocate in the decision-making process”.

According to Riddal & Bennet (2003), an accurate mental model of the decision-making process is a fundamental prerequisite of good decision making. Decision makers need timely and accurate knowledge of business data, which should furnish them with the means, but not necessarily the ability, to improve decision making. Using a mixed method approach in agile settings, Drury et al. (2011) investigate decisions involved in iteration planning, execution, review and retrospective, and identify six obstacles to decision-making. They connect the findings to a theory of descriptive decision-making and describe the effects of these obstacles. This work provides a basis for further exploration of how to identify good practice in agile decision making, and also how to support good decision making.

**Tool support for agile decision making**

Supporting decision making through software has evolved from the 1960s onwards, over the intervening decades ranging from analytical tools to the use of intelligence systems. The growth in business software development in organisations spawned a need for software to support and assist decision making. With reference to the 1980s, software decision support tools “included spreadsheet programs for analysing complex problems with trails that have
different sets of data, data base management programs that permit the orderly maintenance
and manipulation of vast amounts of information, and graphics programs that quickly and
easily prepare professional-looking displays of data (Briticania, 2012).

Indeed decision support systems have seen continuous innovation, which began “with
building model-driven DSS in the late 1960s, theory developments in the 1970s, and
implementation of financial planning systems, spreadsheet-based DSS and Group DSS in the
early and mid 1980s. Data warehouses, Executive Information Systems, OLAP and Business
Intelligence evolved in the late 1980s and early 1990s. Finally, the chronicle ends with
knowledge-driven DSS and the implementation of Web-based DSS beginning in the mid-
1990s. The field of computerised decision support is expanding to use new technologies and
to create new applications” (Power, 2007). Included in this are “knowledge-driven DSS
which can suggest or recommend actions to managers. The DSS having specialised problem-
solving expertise. The expertise consists of knowledge about a particular domain,
understanding of problems within that domain, and "skill" at solving some of these problems
(Power, 2002). These systems have been called, suggestion DSS (Alter, 1980), knowledge-
based DSS (Klein & Methlie, 1995), and includes Goul, Henderson, and Tonge (1992)
Artificial Intelligence (AI) contributions to DSS” (Power, 2007).

The current phase of decision support innovation is in the development of business
intelligence and data analysis based systems that can incorporate structured and unstructured
data from multiple sources. (Dabrowski, Acton, Drury, Conboy, and Dabrowska, 2011) “The
first target for intelligent systems technology should be the overwhelming flow of data,
information and knowledge produced for executives by an increasing number of sources”,
which needs to be managed (Shim, Warkentin, Courtney, Power, Sharada, Carlsson, 2002).
Crucial decisions in ASD relate to managing, planning, execution, reviews and retrospectives (Power, Drury, 2010). However, the data needed for decision-making are often spread across a range of collaboration platforms and/or information sources, for example, instant messaging, email, scanned documentation, spreadsheets, and proprietary document formats associated with various tools. Agile resources and decision tools require the ability to deal with structured and unstructured data and need to be somewhat elastic or scaleable in the context of projects as they arise (Dabrowski et al., 2011). However, some support tools currently used in ASD are low barrier entry solutions such as scrumboards: others are proprietary or open source. Existing tools include the common applications of word processors, spreadsheets, presentation software and instant messaging. Some agile specific software applications in use include Rally, Scrumdo, Jira, Greenhopper, Hansoft and Version One (Sudheer, 2012). Many of these tools are focused on the management of projects, project planning, tracking with some lending themselves as prediction tools to others, for example back log, iteration management tools, and burn-ups (Version One, 2011).

A way forward

Whilst existing support tools focus on the various phases of software development, they focus on tasks, whereas integrated decision support is the next level of evolution required to ensure shortening the development time and ultimately support more efficient decision making. The basis for the ideal agile tool is to integrate these with tracking, tracing, automated build, data sharing for collaboration and feedback in addition to scrum project management and the decision making embedded. This necessitates the incorporation of people management over distributed teams with changing customer requirements and expectations so as to make software development in agile environments more efficient (Dabrowski et al., 2011). Further, shortening software development time is the ideal (Cooper, Cerulli, Lawson, Peng, and
Rezgui, 2005). One way to achieve this would be through an embedded expert domain knowledge based system that could work through the potential development decisions that are relatively common and routine and time consuming in development projects (Pearlson, Saunders, 2009). Such decision support for ASD would take a snapshot of the customer requirements and present a snapshot prototype development project, to help focus on the higher impacting decision points rather than routine or low-impact decisions. In particular, potential positive impacts may include time to task completion, reduced bug count, reduction in task dependencies, increase in data availability, and product owner confidence (Minkiewicz, 2010). Further, such decision support aids may facilitate an increased innovative capacity in the team, and by extension, the software development organisation, through the provision of competitive advantage.

To consider the path for innovation of DSS, Shim et al recommended that DSS researchers and developers should (i) identify areas where tools are needed to transform uncertain and incomplete data, along with qualitative insights, into useful knowledge; (ii) be more prescriptive about effective decision making by using intelligent systems and methods; (iii) exploit advancing software tools to improve the productivity of working and decision making time, and (iv) assist and guide DSS practitioners in improving their core knowledge of effective decision support … to … expand the interactivity and pervasiveness of decision support technologies (Shim et al, 2002). Agile software “decision makers in … collaborative environments need flexible systems that allow for seamless integration among all members … of organisational networks without being dependent on the knowledge of the users”. This sets up a platform to appreciate the need for decision support and the identifies the need for innovation in decision systems (Shafiei, Sundaram, Piramuthu, 2012). The reason for this perspective is suggested in that “agile software development practices have increasingly been adopted to respond to the challenges of volatile business environments, where the markets
and technologies evolve rapidly and present the unexpected”. (Pikkarainen, Haikara, Salo, Abrahamsson, Still, 2008). “As business and technology environments change at an unprecedented rate, software development agility to respond to changing user requirements has become increasingly critical for software development performance. Agile software development approaches, which emphasize sense-and-respond, self-organization, cross-functional teams, and continuous adaptation, have been adopted by an increasing number of organisations to improve their software development agility” (Lee, Xia, 2010).

Information and communication is key to decision quality yet little is known about how agile practices affect communication. We need to increase our understanding of communication in the context of agile software development: internally among the developers and project leaders and in the interface between the development team and stakeholders (i.e. customers, testers, other development teams). The study by Pikkarainen et al. (2008) shows that agile practices improve communication but indicates that, “in larger development situations involving multiple external stakeholders, a mismatch of adequate communication mechanisms can sometimes even hinder the communication …The use of agile practices requires that the team and organisation use also additional plan-driven practices to ensure the efficiency of external communication between all the actors of software development. These plan driven practices have to be based around the phases steps methods and tools used in agile development. However, according to Giblin and Ryan (2010), their findings indicate that the [agile] team was not making decisions collaboratively, estimates were inaccurate when there was little experience and decisions were not tracked so decision quality was unclear. These findings help drive the development of improved decision strategies and tracking mechanisms to assist these teams to improve their outputs.
“The agile development literature is …lacking empirical evidence and theoretical foundation to support the principles and practices of agile development. Little research has empirically examined the software development agility construct in terms of its dimensions, determinants, and effects on software development performance. As a result, there is a lack of understanding about how organisations can effectively implement an agile development approach” (Lee, Xia, 2010). Agile software development teams are characterised by having flexible team structures with team members taking different roles to learn new skills. Teams in theory participate in collaborative, group decision making with customers and product owners as team members. They operate short iterations in dynamic environment with decisions made daily. The team leader act as facilitator instead of an accountable decision maker. In practice there is minimal documentation and tracking. These characteristics can impact the decision making process as flexible structures, short iterations and minimal documentation could indicate that decision making in an agile environment should be ad hoc, unstructured and without discipline. However, it is unclear how these characteristics affect the decision process (Drury, Conboy, Acton, 2012). Further, the fast, unstructured and turbulent nature of decision making in agile software development (ASD) teams at times necessitates software-based support tools (Dabrowski, Acton, Drury, Conboy, and Dabrowska, 2011). The demands on development teams are increasing in terms of delivery expectations, and therefore requires innovation in how decision support tools are used. Decisions in agile teams are intense, with teams iteratively producing customer-focused software in short bursts of time.

Decisions in ASD relate to managing, planning, execution, reviews and retrospectives (Drury, Power 2010). The planning decisions revolve around time planning and scheduling. These decisions take place in meetings at the start of each sprint or iteration, during which the team collectively defines and plans tasks for next sprint or iteration. Following on are
meetings held after each sprint/iteration, during which a team reflects on what went well, what didn’t, and then decide what could be improved in future sprints/iterations. These planning meetings give team members visibility on requirements, individual task assignments, and estimates agreed for each task, ensure that decision outcomes are communicated on time and not through others. The daily stand-up provides transparency and visibility on the day-to-day progression of tasks where decisions can be made on potential delays are immediately addressed by the team. The retrospectives also provide transparency and visibility regarding achievement of sprint goals. Team members could quickly seek clarifications from each other when delays occurred and immediately improve decision-making in the work processes to avoid recurrence. The iteration/sprint planning practice provides an opportunity for team members to determine, and agree on estimates. Estimates can be calculated by the team leader or more collaboratively if the team members have some expertise their estimates are equally valued and decisions taken collectively.

The level of accountability and collective responsibility in agile methods nurture trust by facilitating vigilance, aligning members’ perceptions realistically with individual competences and abilities, fostering a sense of benevolence through team solidarity, and enhancing perceived integrity through a demonstrated shared work ethic. The decisions following the task estimates are in deciding who does which tasks. Other management decisions include determining the scope of the sprints, what happens with changing to the scope, technical decisions, user story and task prioritisation, the schedule order of delivery. Some critical decisions include the velocity, capacity and feature discovery during iterations which will all affect execution and implementation (McHugh, 2012).

Obstacles to appropriate decision making can result from the following actions as described by Drury and Power (2010). They suggest that difficulties arise as people are pulled onto
other projects, dependencies on other teams, the scrum master acting as team lead rather than facilitator, other project / product stakeholders that can influence decision making, absent product owner / customers, a lack of detail on requirements with a multiple of competing / conflicting priorities (Drury, Power 2010). These obstacles are significant and require systems to support the decisions around them however, the data needed for decision-making are often spread across a range of collaboration platforms and/or information sources, for example, instant messaging, email, scanned documentation, spreadsheets, and proprietary document formats associated with various tools. Agile resources and decision tools require the ability to deal with structured and unstructured data and need to be somewhat elastic or scaleable in the context of projects as they arise (Dabrowski et al., 2011).

Some support tools currently used in ASD are low barrier entry solutions such as scrumboards: others are proprietary or open source. Existing tools include the common applications of word processors, spreadsheets, presentation software and instant messaging. Some agile specific software applications in use include Rally, Scrumdo, Jira, Greenhopper, Hansoft and Version One (Sudheer, 2012). Many of these tools are focused on the management of projects, project planning, tracking with some lending themselves as prediction tools to others, for example backlog, iteration management tools, and burn-ups (Version One, 2011). Whilst existing support tools focus on the various phases of software development, they focus on tasks, whereas integrated decision support is the next level of evolution required to ensure shortening the development time and ultimately support more efficient decision making.

The basis for the ideal agile tool is to integrate these with tracking, tracing, automated build, data sharing for collaboration and feedback in addition to scrum project management and the decision making embedded. This necessitates the incorporation of people management over
distributed teams with changing customer requirements and expectations so as to make software development in agile environments more efficient (Dabrowski et al., 2011). Further, shortening software development time is the ideal (Cooper, Cerulli, Lawson, Peng, and Rezgui, 2005). One way to achieve this would be through an embedded expert domain knowledge based system that could work through the potential development decisions that are relatively common and routine and time consuming in development projects (Pearlson and Saunders, 2009). An immediate implementation of such decision support for ASD might take a snapshot of the customer requirements and present a snapshot prototype development project, to help focus on the higher impacting decision points rather than routine or low-impact decisions. In particular, potential positive impacts may include time to task completion, reduced bug count, reduction in task dependencies, increase in data availability, and product owner confidence (Minkiewicz, 2010). Further, such decision support aids may facilitate an increased innovative capacity in the team, and by extension, the software development organisation, through the provision of competitive advantage.

Conclusion

Agile systems development is a highly intensive and evolving paradigm. This paper highlights a need for research into decision tools to support ASD decision making. The paper provides some insights on salient aspects of the literature on ASD, opening up an avenue for further understanding of the decisions made in ASD and the decision making process, and moving towards the development of a framework for tool support in agile decision making.

There is a need to further understand decision-making processes in ASD, and the subsequent identification of ways in which decision tool usage can be measured and assessed. Benefits of measurability include the ability to compare usage across ASD projects and identify potential areas of improvement in application of decision tools over time and over project
teams. As agile methods are adopted and embedded in software development organisations, there is a need to develop a better understanding of the implementation of decision tools at the day-to-day level. There are many challenges to overcome, driven primarily by the intensity of the ASD environment. Further research will aid in the development of a framework for innovation in decision tools to support ASD.
References


DRURY, M., CONBOY, K., ACTON, T., 2012. Understanding an agile software development teams decision making [Online].


LEE, G., XIA, W. D. 2010. TOWARD AGILE: AN INTEGRATED ANALYSIS OF QUANTITATIVE AND QUALITATIVE FIELD DATA ON SOFTWARE DEVELOPMENT AGILITY. Mis Quarterly, 34, 87-114.


This work was supported, in part, by Science Foundation Ireland grant 10/CE/I1855 to Lero - the Irish Software Engineering Research Centre (www.lero.ie).