# Problem drift: a risk model for complex socio-technical projects

Silvana Costantini, Jon G. Hall, and Lucia Rapanotti

The Open University, UK

**Abstract.** Increased globalisation and market volatility put pressure on organisations to become more flexible and explore new technologies and operational domains, so that projects are becoming more complex, with increasing uncertainty and risk. Complex project management is challenging for both traditional and agile approaches: traditional techniques may be left behind by unrecognised environmental change in a volatile context, while in a stable environment, agile may require too expensive interaction with the client. One area of growing interest is how traditional and agile may combine to increase the chance of project success. Yet how to strike a balance between the two remains poorly understood. In this position paper, we propose a model for complex socio-technical projects related to risk arising from volatility, that tries to explain the balance of agile and traditional. The model introduces the concept of drift rate as a measure of volatility, with the impact of drift related to loss of development resource, risk accumulation as a function of resource expenditure and drift rate, and validation the means to manage that risk.

## 1 Introduction

Organisations must adapt to their changing business environment to remain profitable, competitive and able to meet their strategic goals. In large organisations, projects are the instrument of choice for such adaptations, each project addressing specific organisational socio-technical problems. Increased globalisation and market volatility are putting pressure on organisations to become more flexible and explore new technologies and operational domains, so that projects are becoming more complex, with increasing uncertainty and risk.

There are many sources of complexity and uncertainty. From a technical perspective, they stem from the combination, interactions and inter-dependencies of technologies and the pace of technical change [5]; from a social perspective, from "the number and diversity of stakeholders in the social network around a project" [5], their culture and power relations; and, from a knowledge perspective, the embedded, distributed and tacit nature of knowledge [1] combined with inherent 'wicked' [12] features prevents complex problems from being entirely specifiable upfront. Moreover, the volatility of the external environment, a key driver for organisational change, accelerates and magnifies complexity and uncertainty [1]. All combine to increase risk and failure associated with projects [4].

At a fundamental level, all project management approaches, whether traditional ('plan-driven') or 'agile,' seek to control uncertainty and manage risk, while making best use of resources. Complex projects are problematic for both approaches: environment volatility is a challenge for plan-driven ones, technical complexity for agile ones, and knowledge complexity for both. While often considered as antithetic, plan-driven and agile practices are increasingly combined in practice, particularly in software development, in an *ad hoc* manner to reap their benefits, with various degrees of success [10]: the reality is that in complex projects deviations and failure rates remain high [3].

[1] advocate that complex project management should be seen as a form of complex problem solving. This is also our position, therefore we apply a complex problem solving framework to explicate and model features of traditional and agile approaches. The framework is particularly strong at process modelling [8], and this allows us to consider their potentially constructive combination, indexed by organisational and problem characteristics.

The ultimate aim of the work is to equip organisations with the tools to understand better the nature of their problems and tailor their project management practices to their needs and culture. This position paper is a step towards this aim: it seeks to explicate risk characteristics of project management approaches when interpreted as problem solving processes, through the lens of a design theoretic framework for complex problem solving, called Problem Oriented Engineering [9]. Specifically, we propose a model to explicate how different approaches contribute to risk accumulation and mitigation.

## 2 Risk and uncertainty

The Project Management Institute's PMBOK defines risk as "an uncertain event or condition that, if it occurs, has a positive or negative effect on one or more project objectives such as scope, schedule, cost, or quality" [11], and many risk management approaches exist, aimed at identifing, quantifying and mitigating risk. The notion of 'uncertain event' is an indication that uncertainty and risk are closely related concepts, although there is growing recognition of their differences. For instance, [4] takes a knowledge-centric view to distinguish between different kinds of uncertainty, summarised in the four quadrant model reproduced in Figure 1.

In Cleden's view [4], all projects start with inherent uncertainty, some of which is susceptible to risk analysis, which transforms some uncertain events into risk, but still leaves latent uncertainly, which requires management approaches distinct from traditional risk management.

One of our objectives is to explicate how traditional and agile project management deal with risk and uncertainty in complex projects.

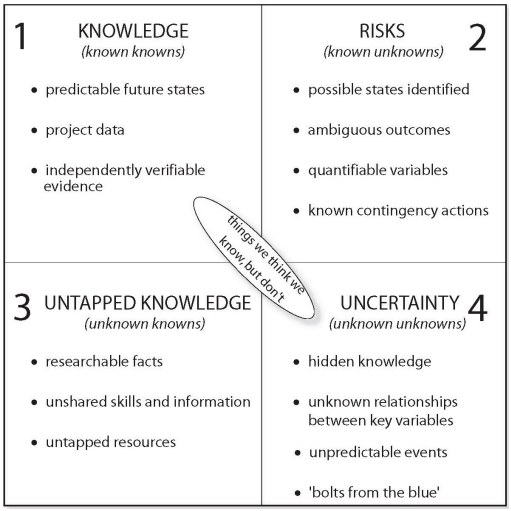


**Fig. 1.** Four quadrant model (reproduced from [4]).

## 3 Complex project management as complex problem solving

Ahern et al. [1] argue that complex project management should be seen as a form of complex problem solving. They highlight two fundamental aspects: learning, which allows the creation, organisation and coordination of emergent knowledge that cannot be specified at the outset; and fostering a common will of mutual interest among project stakeholders. The latter is similar to Conklin's coherence [5] — stakeholders having shared understanding and shared commitment to project objectives and success. These two aspects relate directly to knowledge and social complexity.

In line with this thinking, in our research, we look at complex project management through the lens of Problem Oriented Engineering (POE, see [9] for a comprehensive introduction), a design theoretic framework for complex problem solving. POE has been applied in real-world case studies as a systematic approach to address a wide range of engineering and organisational problems (for instance, [6, 7]).

POE is concerned both with different types of knowledge, spanning problem and solution spaces, and with stakeholders with a role in the problem solving process. In essence, a POE problem is a recognised (real-world) need in context, whose solution satisfies it. Arriving at a solution is a process in which need, context and solution are progressively 'explored' and 'validated.' During exploration, learning takes place so that stakeholders come together to create, organise and exchange knowledge, which reduces knowledge complexity, while validation records stakeholders' satisfaction on the outcome of exploration, reducing uncertainty and contributing to coherence.

Explorations take time and effort, so that expenditure and project risk is accumulated during such activities. Through validation, explorers transfer risk to validating stakeholders. When validation fails, backtracking to a previous state, including project start, may be the outcome.

In its idealised form, such a process is captured by the pattern in Figure 2. The pattern relates fundamental activities and actors and is not a prescriptive model — a process may be linear, iterative, spiral, fractal, etc.

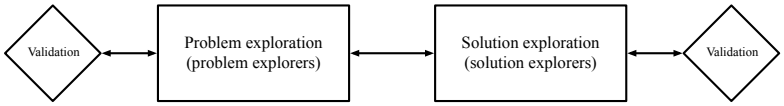


**Fig. 2.** POE Process Pattern (PPP): stakeholders interacting in problem solving activities — problem exploration and solution exploration are interleaved with validation (adapted from [8]).

In summary, as caputured by the PPP, complex project management is a problem solving endeavour to design solutions to organisational problems (need in context), and in which knowledge and social complexity are tackled through problem and solution explorations and validations, reducing uncertainty and mitigating risk.

Our conjecture is that PPP and its underlying concepts provide a fruitful theoretical framework in which to analyse risk characteristics of traditional and agile project management approaches when interpreted as problem solving processes. In the next section we put this idea to the test by proposing a model for project risk arising from problem (need in context) volatility.

## 4 Problem drift

Traditional project management approaches expect projects to be completely planned in advance, with their execution merely a process of implementation. As a consequence, resources are committed to a chosen solution early in the project lifecycle. Even if we were to discount knowledge complexity — which prevents both problem and solution from being completely knowable at the onset — such a commitment is particularly risky in the presence of problem volatility, due, for example, to rapid market or technological change (context volatility) or stakeholder conflicts (need volatility). If one were to take the same approach to archery, one would identify a target and aim, close ones eyes, pull back the string and let an arrow go. In the case that the target is static, such an approach might be appropriate. But what in the case of a moving target? The arrow will land where the original target was, not where it has moved to.

Thus, in traditional approaches, fixing a solution choice early incurs the risk that the problem will have moved as a product of the volatility of the organisation's problems (need in context): the more volatile the problem, the higher

the likelihood that a delivered solution will 'miss' its target, i.e., the delivered solution does not satisfy the need in context. Instead, we must track the target as it 'moves': this is where validation has a role to play.

To understand 'problem drift', we propose the 'problem drift model' illustrated in Figure 3:

- the vertical axis represents 'problem drift', which is a measure of divergence from the problem the project was set up to solve and that which is in the real world, as a result of problem volatility;
- the horizontal axis represents resource use (equivalently, the passage of time) from most recent validation (initially project front-end);
- we assume an 'acceptable drift range' (the shaded area), where the divergence is not sufficient to invalidate development of the solution;
- the 'drift rates' lines indicate the degree of problem volatility: from high (V1), i.e., the environment changes very quickly, to low (V3), where it changes more slowly.

Marked on the horizontal axis is a 'validation point,' which is the point where validation against the problem is sought on some validation artefact from stakeholders. Three situations are considered: i) in a situation of low drift (V3), drift has occurred but the validation artefact can still be validated; ii) when medium drift (V2), drift leads to a marginal call on the boundary between acceptable and unacceptable drift; iii) when high drift (V1), the validation artefact cannot be validated and some or all development resource is lost: based on the knowledge gained, a previous project state may be revisited or, in the extreme, the project is abandoned.

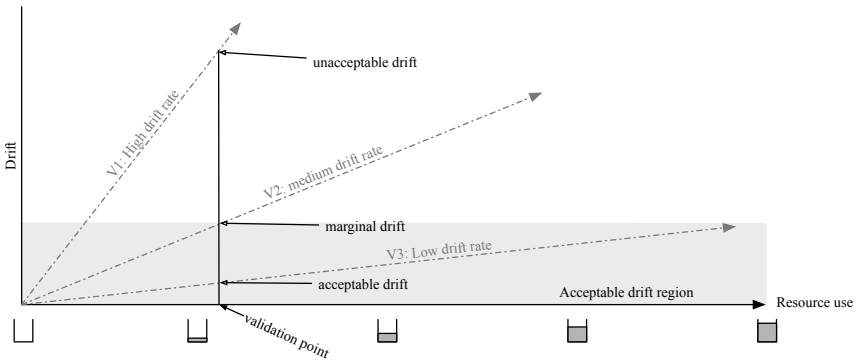


**Fig. 3.** Problem drift model

Under this model, the impact of problem drift relates to lost development resource, and risk accumulation is then a function of resource expenditure and drift rate, with validation the means to manage risk. This is consistent with

empirical evidence [14] in software projects which suggests that more frequent developer/customer communication (a form of validation) lessens the impact of requirements volatility.

Seen this way, we may interpret the role of validation in various project management approaches. At the extremes:

- in extreme plan-driven methods, such as Royce's posited Waterfall model [13], validation would occur only at project end. Unless the drift rate is zero or extremely low, any delivered solution to the original problem will be unacceptable. This corresponds to traditional approaches being appropriate only in situations of low drift (with the caveat that knowledge complexity would still pose a challenge);
- in extreme agile methods, validation occurs early and often so that, even in situations of medium to high drift, unacceptable drift is never experienced. The related risk of failure is thus reduced.

One might ask, if agile copes with all drift rates, why not eschew plan-driven methods altogether? Indeed, this view might be the source of the popular view that *all* processes should be agile. Our model also explains why this is not the case, however: validation requires time and effort, and so consumes resources, diverting them from development. As a result, agile processes 'pay' for their management of risk through potentially expensive validation.

There is therefore a balance to be struck between agile and plan-driven: the model suggests that the optimal balance is to validate at, or just before, the point of marginal drift (V2). To do this a number of things are required: i) the measurement of problem volatility; ii) the notion of acceptable drift; and iii) ways of combining traditional and agile methods to support the optimal balance.

## 5 Discussion and Future Work

Complex project management is challenging for both traditional and agile approaches: traditional techniques may be left behind by unrecognised environmental change in a volatile context, while in a stable environment, agile may require too expensive interaction with the client. One area of growing interest is how traditional and agile may combine to increase the chance of project success. Yet how to strike a balance between the two remains poorly understood.

In this position paper, we have proposed a model for complex socio-technical projects related to risk arising from problem volatility, that tries to explain the balance of agile and traditional. The model is located within the Problem Oriented Engineering (POE) framework of the second and third authors [9] and, in particular, derives from the POE Process Pattern (PPP). The model introduces the concept of drift rate as a measure of problem volatility, with the impact of problem drift related to loss of development resource, risk accumulation as a function of resource expenditure and drift rate, and validation the means to manage risk.

A different characterisation of tradition versus agile is given in [2], which takes inspiration from the SECI model for knowledge transformation in organisations [**?** ] to explicate and compare how tacit, explicit and embedded knowledge are transferred during traditional vs. agile software development. The main insight is that traditional methods rely on explicit forms (e.g., requirements specifications, design models) to communicate with a variety of specialist stakeholders, while in agile methods, tacit knowledge is shared through 'Socialization' (i.e., direct communication between problem owners and development team) supported by 'Embedment', in which knowledge is embedded in the software system through coding. With reference to the PPP, these forms of knowledge transfer fall within exploration, but the different types of transfer are not differentiated by the pattern. Instead, PPP is concerned with the accumulation of risk during exploration and its mitigation via validation. This suggests that the two approaches can be combined and may thus deliver orthogonal benefits.

The drift model proposed presupposes that drift can be measured and predicted. The objective of future research will be to explore if these measurement and prediction are feasible and how to implement them. For instance, it is plausible that measuring how problem descriptions change over time is one possible proxy measure for context volatility. However, more sophisticated notions should also be considered.

Supposing that our drift model works, the question arises as to the extent we can optimally combine agile with traditional processes. This would require some hybrid process architecture to be developed. The second and third authors have already taken preliminary steps in the analysis of the agile/plan-driven mix in the context of software projects in [8], but much remains to be done.

Finally, in our model we have focused on problem volatility and process activity, irrespective of organisational culture, which is an acknowledged key success factor for agility [2]. Any cultural implication of our model will be considered in future research.

## Bibliography


[1] Terence Ahern, Brian Leavy, and PJ Byrne. Complex project management as complex problem solving: A distributed knowledge management perspective. *International Journal of Project Management*, 32(8):1371–1381, 2014.

[2] Ilia Bider. Analysis of agile software development from the knowledge transformation perspective. In *International Conference on Business Informatics Research*, pages 143–157. Springer, 2014.

[3] Fritz Böhle, Eckhard Heidling, and Yvonne Schoper. A new orientation to deal with uncertainty in projects. *International Journal of Project Management*, 2015. ISSN 0263-7863.

[4] Mr David Cleden. *Managing project uncertainty*. Gower Publishing, Ltd., 2012.

[5] Jeffrey Conklin. *Dialogue mapping: Building shared understanding of wicked problems*. Wiley, 2006.

[6] J. G. Hall, L. Rapanotti, and M. A. Jackson. Problem oriented software engineering: Solving the package router control problem. *IEEE Transactions on Software Engineering*, 34(2):226–241, 2008. ISSN 0098-5589. doi: 10.1109/TSE.2007.

70769. URL http://ieeexplore.ieee.org.libezproxy.open.ac.uk/ielx5/32/4476751/04384506.pdf?tp=&arnumber=4384506&isnumber=4476751.

[7] Jon G. Hall and Lucia Rapanotti. Assurance-driven design in problem oriented engineering. *International Journal on Advances in Systems and Measurements*, 2 (1):119–130, 2009.

[8] Jon G. Hall and Lucia Rapanotti. Towards a design-theoretic characterisation of software development process models. In *Proceedings of the Fourth SEMAT Workshop on General Theory of Software Engineering*, pages 3–14, 2015.

[9] Jon G Hall and Lucia Rapanotti. A design theory for software engineering. *Information and Software Technology*, 87:46–61, 2017.

[10] Kati Kuusinen, Peggy Gregory, Helen Sharp, and Leonor Barroca. Strategies for doing agile in a non-agile environment. In *Proceedings of the 10th ACM/IEEE International Symposium on Empirical Software Engineering and Measurement*, page 5. ACM, 2016.

[11] Project Management Institute. *A Guide to the Project Management Body of Knowledge (PMBOK®Guide)*. Project Management Institute, Incorporated, 2013.

[12] Horst WJ Rittel and Melvin M Webber. Dilemmas in a general theory of planning. *Policy sciences*, 4(2):155–169, 1973.

[13] Winston W Royce et al. Managing the development of large software systems. In *proceedings of IEEE WESCON*, volume 26, pages 1–9. Los Angeles, 1970.

[14] Didar Zowghi and Nur Nurmuliani. A study of the impact of requirements volatility on software project performance. In *Software Engineering Conference, 2002. Ninth Asia-Pacific*, pages 3–11. IEEE, 2002.