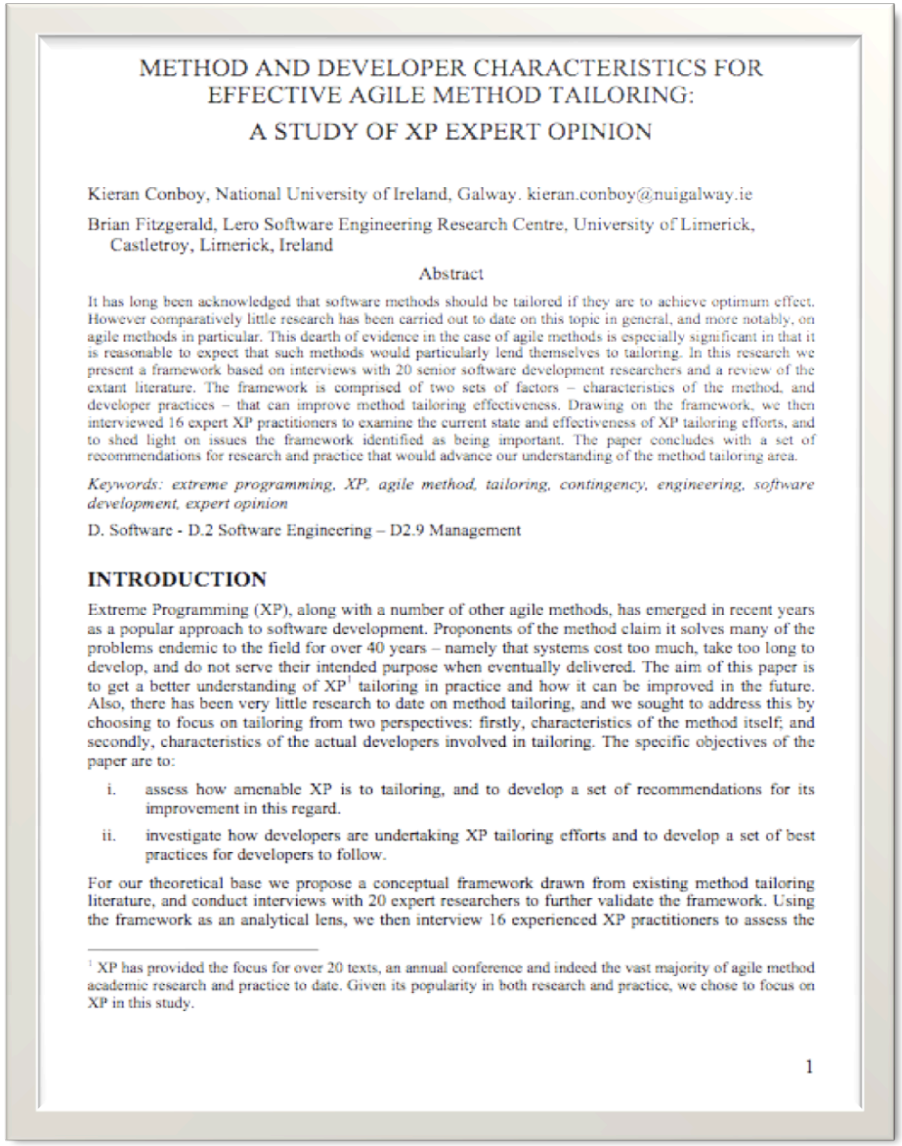


METHOD AND DEVELOPER CHARACTERISTICS FOR EFFECTIVE AGILE METHOD TAILORING:
A STUDY OF XP EXPERT OPINION

Kieran Conboy, National University of Ireland, Galway. kieran.conboy@nuigalway.ie

Brian Fitzgerald, Lero Software Engineering Research Centre, University of Limerick, Castletroy, Limerick, Ireland

Abstract

It has long been acknowledged that software methods should be tailored if they are to achieve optimum effect. However comparatively little research has been carried out to date on this topic in general, and more notably, on agile methods in particular. This dearth of evidence in the case of agile methods is especially significant in that it is reasonable to expect that such methods would particularly lend themselves to tailoring. In this research we present a framework based on interviews with 20 senior software development researchers and a review of the extant literature. The framework is comprised of two sets of factors – characteristics of the method, and developer practices – that can improve method tailoring effectiveness. Drawing on the framework, we then interviewed 16 expert XP practitioners to examine the current state and effectiveness of XP tailoring efforts, and to shed light on issues the framework identified as being important. The paper concludes with a set of recommendations for research and practice that would advance our understanding of the method tailoring area.

Keywords: extreme programming, XP, agile method, tailoring, contingency, engineering, software development, expert opinion

D. Software - D.2 Software Engineering – D2.9 Management

INTRODUCTION

Extreme Programming (XP), along with a number of other agile methods, has emerged in recent years as a popular approach to software development. Proponents of the method claim it solves many of the problems endemic to the field for over 40 years – namely that systems cost too much, take too long to develop, and do not serve their intended purpose when eventually delivered. The aim of this paper is to get a better understanding of XP[1] tailoring in practice and how it can be improved in the future. Also, there has been very little research to date on method tailoring, and we sought to address this by choosing to focus on tailoring from two perspectives: firstly, characteristics of the method itself; and secondly, characteristics of the actual developers involved in tailoring. The specific objectives of the paper are to:

i. assess how amenable XP is to tailoring, and to develop a set of recommendations for its improvement in this regard.

ii. investigate how developers are undertaking XP tailoring efforts and to develop a set of best practices for developers to follow.

For our theoretical base we propose a conceptual framework drawn from existing method tailoring literature, and conduct interviews with 20 expert researchers to further validate the framework. Using the framework as an analytical lens, we then interview 16 experienced XP practitioners to assess the

[1] XP has provided the focus for over 20 texts, an annual conference and indeed the vast majority of agile method academic research and practice to date. Given its popularity in both research and practice, we chose to focus on XP in this study.

1

METHOD AND DEVELOPER CHARACTERISTICS FOR EFFECTIVE AGILE METHOD TAILORING: A STUDY OF XP EXPERT OPINION

Kieran Conboy

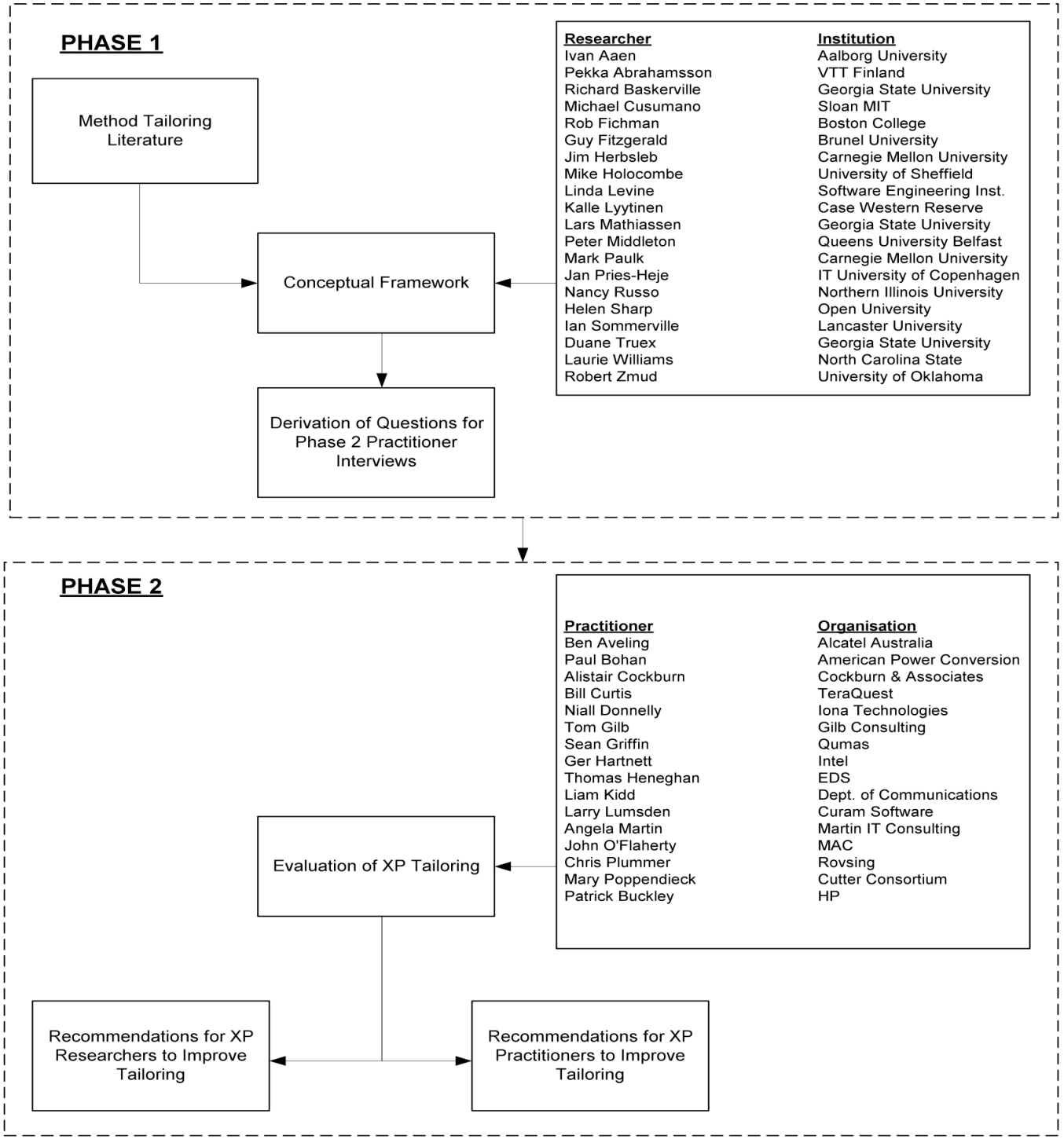
Draft in 3rd round of revision in Transaction on Software Engineering Methodology (TOSEM)

Objective

- assess how amenable XP is to tailoring, and to develop a set of recommendations for its improvement in this regard.
- investigate how developers are undertaking XP tailoring efforts and to develop a set of best practices for developers to follow.

Motivation

- Claims that agile methods are ”the silver bullet” but dissemination means that agile methods must be tailored to suit many contexts.
- Agile methods turn up the dial on social interaction – tailoring needs to be sensitive to softer issues.
- Lack of ”cohesive” empirical agile method tailoring research.
- Anything labelled as ”agile” should be amenable to tailoring.


PHASE 1
Method Tailoring Literature
Conceptual Framework
Derivation of Questions for Phase 2 Practitioner Interviews
Researcher
Ivan Aaen
Pekka Abrahamsson
Richard Baskerville
Michael Cusumano
Rob Fichman
Guy Fitzgerald
Jim Herbsleb
Mike Holocombe
Linda Levine
Kalle Lyytinen
Lars Mathiassen
Peter Middleton
Mark Paulk
Jan Pries-Heje
Nancy Russo
Helen Sharp
Ian Sommerville
Duane Truex
Laurie Williams
Robert Zmud
Institution
Aalborg University
VTT Finland
Georgia State University
Sloan MIT
Boston College
Brunel University
Carnegie Mellon University
University of Sheffield
Software Engineering Inst.
Case Western Reserve
Georgia State University
Queens University Belfast
Carnegie Mellon University
IT University of Copenhagen
Northern Illinois University
Open University
Lancaster University
Georgia State University
North Carolina State
University of Oklahoma
PHASE 2
Evaluation of XP Tailoring
Recommendations for XP Researchers to Improve Tailoring
Recommendations for XP Practitioners to Improve Tailoring
Practitioner
Ben Aveling
Paul Bohan
Alistair Cockburn
Bill Curtis
Niall Donnelly
Tom Gilb
Sean Griffin
Ger Hartnett
Thomas Heneghan
Liam Kidd
Larry Lumsden
Angela Martin
John O'Flaherty
Chris Plummer
Mary Poppendieck
Patrick Buckley
Organisation
Alcatel Australia
American Power Conversion
Cockburn & Associates
TeraQuest
Iona Technologies
Gilb Consulting
Qumas
Intel
EDS
Dept. of Communications
Curam Software
Martin IT Consulting
MAC
Rovsing
Cutter Consortium
HP

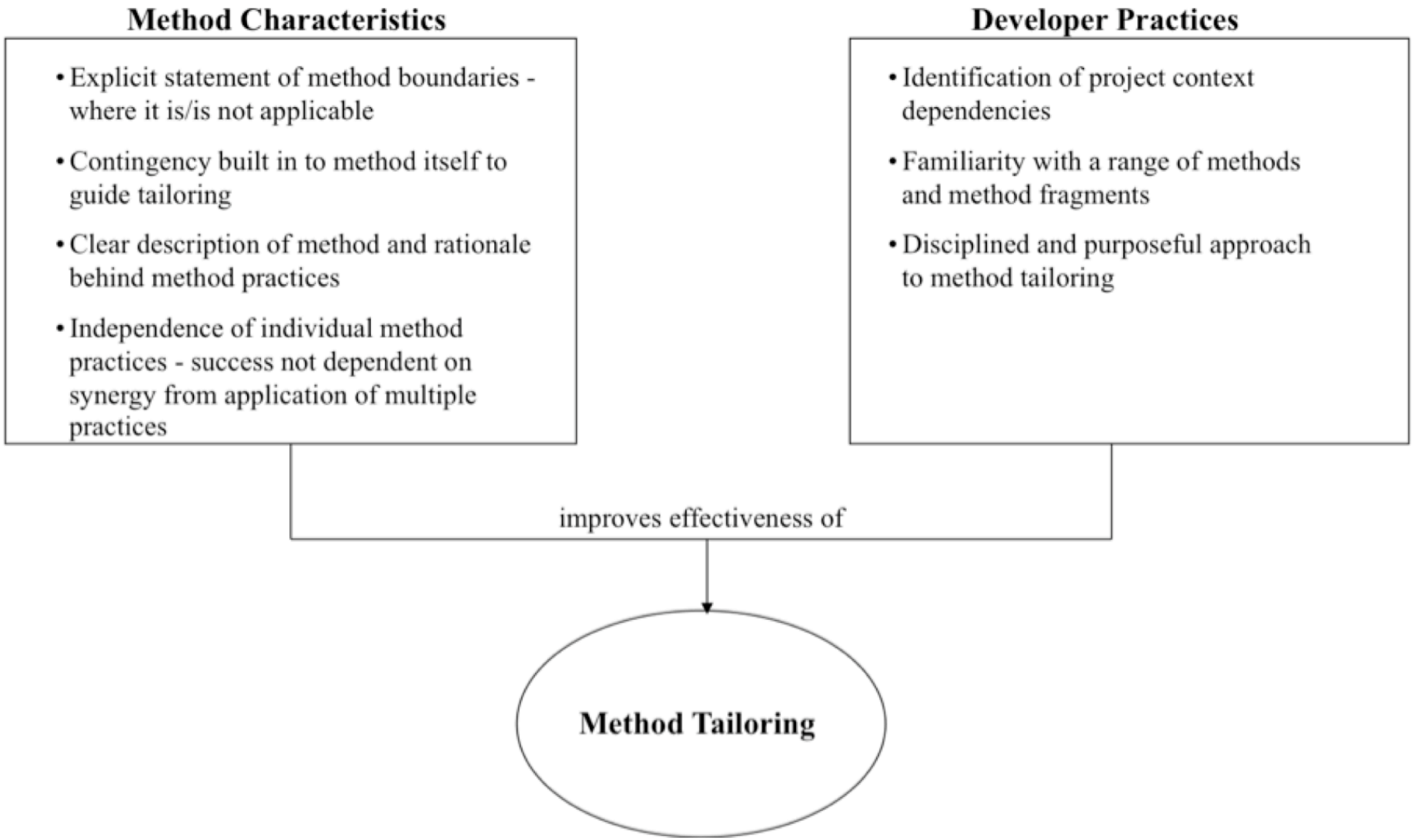


Figure 1: Method and Developer Characteristics Contributing to Method Tailoring

Insights into XP tailoring across organisations

"He [the consultant] would not commit when we asked him to advise us on which XP practices would and would not work. With us he insisted on a 'try first' approach where all practices are tried and only dropped if not working. But after six months and his refusal to accept any arguments against the method, I'd say his philosophy was 'try first, and if it doesn't work then just try harder.'" (P7)

"We changed a lot of things about XP. It took a long time to perfect, given we were flying in the dark, on a trial and error basis, but we got there. And I think we are more agile. I just wish the option to use these alternatives could have been part of the method. It would have saved a lot of time, effort and uncertainty." (P1)

"XP is not like other methods. I can get the team to carry out technical procedures; but the social side of XP that goes with pairing, stand-ups and constant collaboration- if they don't want to do it, I can't make them." (P3)

"I could have forced them to use practices, but isn't developer empowerment the whole idea behind XP? Anyway, if they don't think its worth doing, then I'm not going to tell them otherwise" (P11)

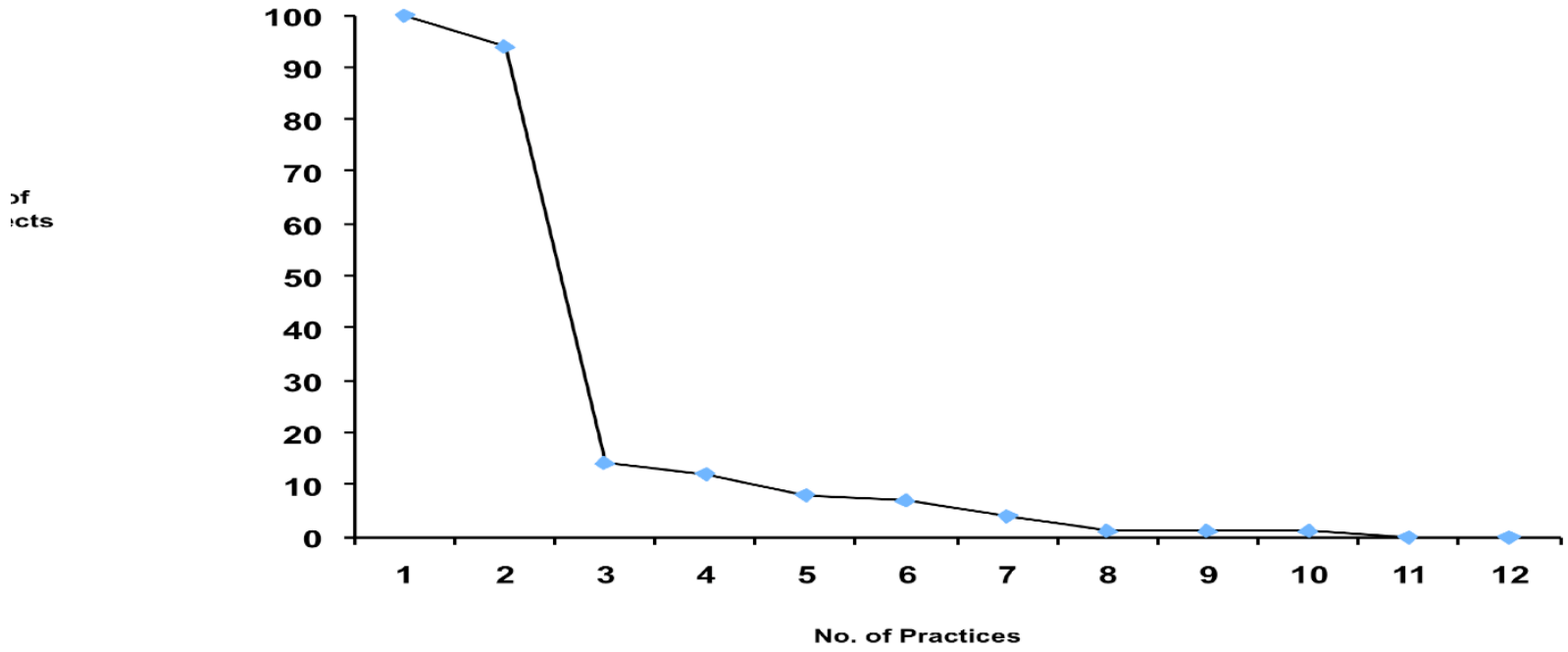


Figure 1: Sporadic Adoption of XP Practices

Construct	Finding	Recommendations for Software Development Teams
Identification of project context dependencies	Decision to adopt and tailor XP rarely involved a formal analysis of situational dependencies (1 of 16 projects studied – P5). The subsequent mismatch directly caused failure and future abandonment of the method in some cases.	#1: *Conduct a formal analysis of method's suitability to the project environment* e.g. Boehn & Turner's (2003) analysis model was used very effectively by P5 for this purpose.
	Decision to adopt XP, as well as subsequent tailoring and implementation decisions, were often driven by one single 'champion' without input from any other team members or stakeholders (8 of 16 projects studied - P1, P2, P3, P4, P6, P8, P10, P16). In some cases this resulted in a biased, uninformed decision without adequate consideration of negative consequences of introducing XP.	#2: *Involve all developers and stakeholders in (i) decision to adopt or not adopt XP, (ii) tailoring of XP, and (iii) implementation of XP* e.g. P15 held three semi-structured, open invitation workshops to identify and resolve various developer issues.
	Rather than tailoring XP to the environment, the organisation or team was tailored, sometimes substantially, to suit the method (10 of 16 projects studied - P1, P2, P3, P4, P6, P8, P10, P11, P12, P16). However, this caused significant problems in some cases (e.g. P16).	#3: *Identify any organisational or project 'breaking points' (the maximum tolerable change), and ensure these points are not crossed, regardless of what XP requires.* e.g. frequency of iterations, degree of colocation, average weekly working hours.
Familiarity with a range of methods and method fragments	Most developers using XP had not even got sufficient knowledge of all XP practices. As a result, the teams studied tended to implement easier practices and ignore more challenging ones. These practices were either omitted completely or implemented poorly as a result – only 25% of projects studied implemented more than 50% of the practices.	#4: Train developers on all XP practices e.g. 1 day in-house tutorial, web research.
	Developers' experiences of many XP practices was often second-hand or textbook-based, and insufficient for informed tailoring.	#5: *Include practical, hands-on components in XP training* e.g. work shadowing, games, role playing (http://www.xp.be/xpgame.html), mentoring.
	Few developers had experience with even one alternative method with which to substitute or extend XP practices. Experience of alternative agile methods (e.g. LSD, Crystal) was particularly lacking (3 of 16).	#7: *Encourage developers to learn and gain experience of other methods* e.g. all 8 developers on P5's project team were tasked with learning and evaluating one agile method each (XP, XP Lite, Scrum, DSDM, Crystal, LSD, ASD, FDD).
	What experience did exist amongst the team was rarely elicited and used when deciding how to tailor/extend XP.	#9: *Elicit developer knowledge and experiences of other methods and practices and incorporate into the tailoring/ extension of XP.*
Disciplined and purposeful approach to method tailoring	There was often little monitoring or control of adherence to XP practices following the initial implementation. In some cases this was beneficial as the actual users of the method decided how it should be used (e.g. P1). However, in some cases non-adherence was due to laziness or negligence and led to gradual abandonment of the method (P3, P6, P12, P15).	#10: *Frequently monitor adherence to XP practices, to ensure non-adherence is not simply due to laziness or negligence* e.g. at every retrospective meeting, P15's team reviewed the use of each practice, its pros and cons, and whether it should be retained.
	Conflict occasionally arose due to inconsistent adoption of practices across different members of the team (P3, P6, P12, P15).	#11: *Communicate post-implementation tailoring efforts across the team* (e.g. at stand-up or retrospectives). #12: *If a tailoring decision is taken by an individual developer, its impact on the other team members should be assessed and discussed* (e.g. at stand-up or retrospectives).

Construct	Finding	Recommendations for Software Development Researchers
Explicit statement of method boundaries	14 of 16 teams studied had difficulty identifying where XP should and should not be applied. XP conference attendance and help from external consultants did not provide any substantial assistance with this issue. Opinions suggested that this was a problem throughout the XP user community.	#1: *Determine levels of project success across different potentially problematic software development environments.* e.g. distributed development, large teams, critical systems, inexperienced developers.
Contingency built-in to method itself to guide tailoring	No practitioner thought that XP guided the tailoring process in any meaningful way, despite the fact that some developers needed and actively sought such guidance. All tailoring efforts were based on team members' own opinions and preferences.	#2: *Identify alternatives for each XP practice, which achieve the same or similar goals and objectives.* e.g. instant messaging, screen sharing, video conferencing, and common file storage can help replace XP's practice of co-location in a distributed development environment.
Clear description of method rationale behind method practices	There was mixed opinion regarding how clearly XP texts explain the rationale and execution of its underlying practices. Most were unclear as to the exact advantages and disadvantages of each practice, and were concerned that many accounts of benefits are often anecdotal, or too subtle to be clearly identified.	#3: *Quantitative research to determine advantages/disadvnatages of XP practices.* #4: *In-depth qualitative research to uncover more subtle, softer advantages/disadvantages of XP practices.*
Independence of individual method practices	Problems or concerns regarding splitting of XP practices occurred in 10 of the 16 projects studied (P1, P4, P6, P7, P8, P9, P11, P12, P15, P16). The consensus was that the social and softer nature of XP practices make it very difficult to identify co-dependencies and knock-on effects between practices. This had a negative impact in some cases. For example, P4, P5, P8, P11 and P12 wanted to remove non-value-adding or problematic practices but decided not to for fear of such unknown co-dependencies.	#5: *Quantitative research to determine co-relations between use of individual practices and (i) effectiveness of other practices and (ii) project success.* #6: *In-depth qualitative research to uncover more subtle, softer effects of use/ non-use of individual practices practice on (i) effectiveness of other practices and (ii) project success.*
	Existing literature suggests that there are 'clusters' of practices that are co-dependent, as opposed to simply pairs of practices. None of the project teams studied had managed to identify any such clusters.	#7: *Quantitative research to determine co-relations between use of groups or 'clusters' of practices and (i) effectiveness of other practices within that cluster and (ii) project success.*

Conboy, K. (2009) Agility From First Principles: Reconstructing The Concept of Agility in Information Systems Development, *Information Systems Research,* 20(3), pp. X-X

Motivation

- Limited applicability
- Lack of Clarity re 'agile'
  - 'fragmented adhocracy'
  - in terms of abstraction, enactment, operationality and philosophy
  - Subtlety of agile makes clarity very difficult
- Lack of 'Theoretical Glue'
- Lack of Cumulative Tradition
  - few comparisons to traditional approaches.
  - Few links to agile in other disciplines (manufacturing, management)
  - Little 'traction'

Implications for Practice

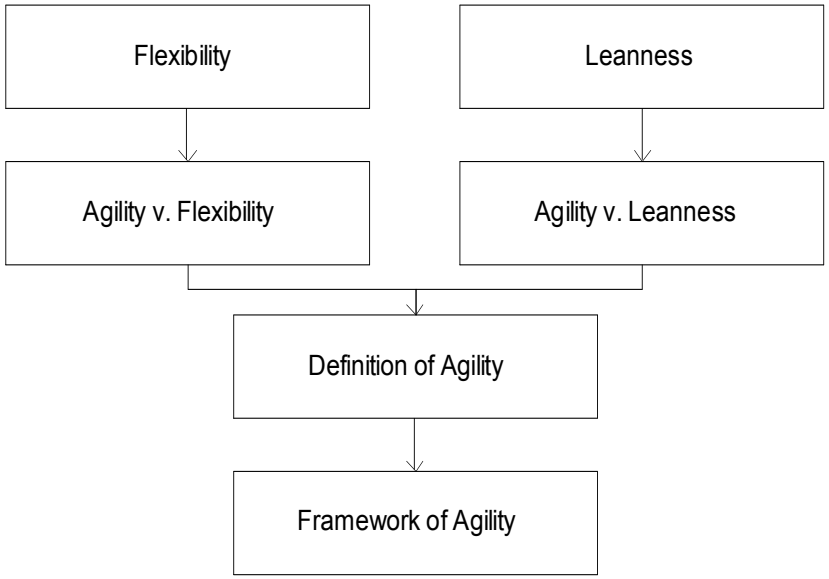
- Encouragement of method improvement
- Method comparison
- Assessment of traditional or in-house methods
- Assessment of environments unsuitable for commercial agile methods
- No spectrum- facilitation of a staged conversion to agility

Agility Framework: Implications for Practice

- Encouragement of method improvement
- Method comparison
- Assessment of traditional or in-house methods
- Assessment of environments unsuitable for commercial agile methods
- No spectrum- facilitation of a staged conversion to agility

Framework Development Strategy

- Research over 5 year period
- Review of conceptual studies
  - Agility, flexibility, leanness
  - Multi-disciplinary



Principles of Agility Framework

1: To be agile, an ISD method component must contribute to one or more of the following:
(i) creation of change
(ii) proaction in advance of change
(iii) reaction to change
(iv) learning from change

2: To be agile, an ISD method component must contribute to one or more of the following, and must not detract from any:
(i) perceived economy
(ii) perceived quality
(iii) perceived simplicity

3: To be agile, an ISD method component must be continually ready i.e. minimal time and cost to prepare the component for use.

Appendix A: Summarised Focus Group Results

Project Name	Practice	Creation	Proaction	Reaction	Learning	Perceived Customer Value			Continual Readiness	Contributor to Agility
						Economy	Quality	Simplicity		
TaxSys	Sprints	Excellent	Excellent	Excellent	Excellent	Good	Excellent	Good	Yes	Yes
	Prioritisation	No perceived effect	No perceived effect	Excellent	No perceived effect	Good	No perceived effect	Good	Yes	Yes
	On-Site Customer	Poor	Poor	Very Poor	Very Poor	Very Poor	Poor	Mediocre	Yes	No
	Stand-Up Meeting	Poor	No perceived effect	Poor	Poor	Very Poor	No perceived effect	No perceived effect	Yes	No
	Pair Programming	Mediocre	No perceived effect	No perceived effect	Good	Poor	No perceived effect	No perceived effect	Yes	No
	Automated Acceptance Testing	No perceived effect	Excellent	Excellent	No perceived effect	Excellent	Excellent	Excellent	Yes	Yes
	Collective Code Ownership	Mediocre	No perceived effect	No perceived effect	No perceived effect	No perceived effect	No perceived effect	Poor	Yes	No
	Retrospectives	No perceived effect	No perceived effect	No perceived effect	Excellent	Good	Good	Excellent	Yes	Yes
	Staged Commitment Strategy	No perceived effect	Excellent	Excellent	No perceived effect	Excellent	No perceived effect	Excellent	Yes	Yes
AccountSys	Sprints	No perceived effect	Excellent	Excellent	Excellent	Good	Excellent	Good	Yes	Yes
	Prioritisation	No perceived effect	No perceived effect	Excellent	No perceived effect	Excellent	No perceived effect	Excellent	Yes	Yes
	On-Site Customer	Excellent	Good	Excellent	Excellent	Excellent	Excellent	No perceived effect	Yes	Yes
	Stand-Up Meeting	Good	Good	Excellent	Excellent	Excellent	Excellent	Good	Yes	Yes
	Pair Programming	Excellent	No perceived effect	No perceived effect	Excellent	Good	Excellent	No perceived effect	Yes	Yes
	Automated Acceptance Testing	No perceived effect	Excellent	Excellent	No perceived effect	Poor	Excellent	Poor	No	No
	Collective Code Ownership	No perceived effect	No perceived effect	No perceived effect	No perceived effect	No perceived effect	No perceived effect	Poor	Yes	No
	Retrospectives	Good	Good	Excellent	Excellent	Good	Excellent	Good	Yes	Yes
	Multi-Site Progress Dashboard	No perceived effect	Excellent	Excellent	Excellent	Excellent	Excellent	Excellent	Yes	Yes
	Cultural Ambassadors & Cross-Pollination	Excellent	Excellent	Excellent	Excellent	Good	Good	Good	Yes	Yes

On-site Customer Practice

Appendix B: The On-Site Customer (OSC) Practice

Project Name	Creation	Proaction	Reaction	Learning	Perceived Customer Value			Continual Readiness	Contributor to Agility
					Economy	Quality	Simplicity		
TaxSys	**Poor** Limited OSC knowledge of wider business needs stifled creation of new ideas. OSC was 'highly passive'. No OSC involvement in 'spikes' (developer experimentation sessions).	**Poor** OSC was 'highly passive', and "showed little sign of proaction" (Project Manager). Many changes in client needs, processes and structure not relayed by OSC to ISD team until discovered by them.	**Very Poor** Average of 4.3 business days for OSC to give feedback on new user stories.	**Very Poor** Limited OSC knowledge of wider business needs and issues stifled team learning. OSC only attended 27 of 113 stand-up meetings, and 6 of 14 retrospectives. OSC did not learn from many mistakes, repeating certain mistakes continuously throughout the project e.g. using wrong version of organisation process documents.	**Very Poor** OSC paid more than twice highest paid developer. The OSC, 1 person of 18 accounted for a large part (14%) of project budget. Incopmpatibility between OSC and developers' working hours increased real cost per hour rate. Only 2 hours overlap on many occasions.	**Poor** Only OSC role was user story development and validation.	**Mediocre** OSC did help to simplify user stories and remove duplications and inconsistencies. However, OSC demanded user stories to be documented in formats different from developer or client organisation, adding significant complexity and effort.	**Yes**	**No**
AccountSys	**Excellent** Monthly creative brainstorming sessions organised and led by OSC. Sessions involved multiple custoemr employees, across spectrum of stakeholder groups (R&D, marketing, accounts, manufacturing).	**Good** OSC presented to developer team on multiple occasions. Presentations were tailored to cater for the developers' lack of business and domain knowledge.	**Excellent** Real-time involvement in user story validation, with instant feedback. OSC continuously issued 'live' reprioritised lists within 24 hours, allowing high priority items to be introduced mid-iteration.	**Excellent** Broad customer knowledge of wider busienss needs and issues. Attended 43 of 45 stand-up meetings.	**Excellent** Relatively low cost OSC. OSC available full-time. Full overlap between OSC and developers' working hours.	**Excellent** Different customer employees were 'revolved' in and out of the OSC role during the project, matching relevant experience to the part of the system being developed. Quality of product increased by intermittent access to other stakeholders via OSC.	**No perceived effect**	**Yes**	**Yes**

Stand-Up Meetings

Appendix C: Stand-Up Meeting (SUM) Practice

Project Name	Creation	Proaction	Reaction	Learning	Perceived Customer Value			Continual Readiness	Contributor to Agility
					Economy	Quality	Simplicity		
TaxSys	**Poor** SUMs highly formal, often critical, and so many team members were too shy or fearful to suggest new ideas.	**No perceived effect**	**Poor** Despite raising issues, potential problems and required changes, many team members felt that on many occasions no responsive action was taken. Certain team member updates were often vague and of limited value in operationally reacting to change.	**Poor** Standard format, where each team member stated what went well, what did not go well and what they were going to do next facilitated basic learning across the team. However, most team members said they paid little or no attention and just 'ran through the motions'. SUMs often dominated by project manager (often as much as 75% of meeting time), and bore closer resemblance to a 'sermon' than exchange of knowledge across team. SUMs held away from development area, with no access to storyboards or other artefacts to refer to during meeting.	**Very Poor** Time wasted due to late arrivals at SUMs. SUMs took average of approx 50 mins, ranging from 0.5 to 1.5 hours. Many SUMs dominated by over-elaborate discussion of a single issue. SUMs attended by all regardless of perceived of a lack of relevance or usefulness.	**No perceived effect**	**No perceived effect**	**Yes**	**No**
AccountSys	**Good** SUMs were highly informal with all encourgaed to speak their mind and suggest ideas regardless of how atypical they may be. At every SUM, one developer had to introduce a 'breakthrough idea', and received feedback on its usefulness and viability. SUMs took place at 2pm each day, allowing a partner team based in the U.S. to observe proceedings and contribute new ideas.	**Good** A 'Barriers' section was introduced in the SUM where each team member proactively listed (i) potential issues that may inhibit their work and (ii) team members who had the expertise to help.	**Excellent** All obstacles or problems documented by the project manager, and a list of response actions circulated to the team. Updates had to be specified clearly at the task level, and accompanied by the potential impact on user stories and initial estimates.	**Excellent** Highly interactive sessions facilitated learning. Team members often distributed excerpts of code, diagrams or other artefacts to support their presentation. SUMs held in development area to allow speakers to refer to artefacts around the room.	**Excellent** SUMs always started on time regardless of who was on time or late. SUMs limited to 15 minute maximum. 'Facilitator' appointed to keep every team member to time. Unresolved issues taken 'offline' afterwards and only discussed be relevant team members.	**Excellent** Persistent problems and defects presented at SUM for group to take away and solve. SUM ensured any expert was aware of a problem they could assist with.	**Good** Each team member left the SUM with a clear idea of (i) their tasks for the day, (ii) who they would work with that day, and (iii) which team members they had to help or advise.	**Yes**	**Yes**

Identification of New Agile Practices

Appendix D: Identification of New Agile Practices

Practice (Project)	Description of Practice	Creation	Proaction	Reaction	Learning	Perceived Customer Value			Continual Readiness	Contributor to Agility
						Economy	Quality	Simplicity		
Staged Commitment Strategy (TaxSys)	Due to a delay in confirmation of project budget, the team created three sets of requirements, based on three possible budget totals (£1.3m, £3m, £4.7m). These were factored into every prioritisation, retrospective and iteration.	**No perceived effect**	**Excellent** Staging proactively protected against any decision to reduce project funding.	**Excellent** No effort required to downsize system scope or quality in the event of funding reduction.	**No perceived effect**	**Excellent** Ensured no wasted effort on parts of the system that may be discontinued.	**No perceived effect**	**Excellent**	**Yes**	**Yes**
Multi-Site Progress Dashboard (AccountSys)	Online dashboard which represents progress of multiple teams distributed across many locations. Real-time recording of each user story, the developer(s) assigned to that story, their location(s), the status of the story, the no. of unit tests under each story and the no. of those successfully tested or not.	**No perceived effect**	**Excellent** Transparency removed much potential uncertainty often caused by distributed development. Allowed early identification of potential issues and conflicts between teams.	**Excellent** All developers in all locations instantly informed of changes in user story status, underlying unit tests, etc. Ensured complete response to change, as no task 'fell between the teams' and remained accidently unassigned.	**Excellent** High transparency and visibility taught each team about the other's abilities, progress, preferences, and cultural traits. A team member could drill-down to view and learn about technical specifications, action items, code and tests.	**Excellent** No financial cost or additional documentation or processes required. Seamless, real-time updates from local monitoring systems.	**Excellent** Large reduction in defects. No possibility for requirements to 'fall between' the teams. Transparency of progress introduced a 'competitive but helathy element' between teams.	**Excellent** Dashboard made division of work between distributed teams very simple. Dashboard provided a single point of reference simplifying communication between teams.	**Yes**	**Yes**
Cultural Ambassador & Cross-Pollination (AccountSys)	Member of offshore team is relocated to onshore team for six weeks duration. The person performing this role was changed every six weeks to allow all offshore members to spend time with the onshore team.	**Excellent** Ambassador referred to potential ideas and collaborative experimentation with offshore team.	**Excellent** Ambassador removed much potential uncertainty often caused by distributed development. Allowed early identification of potential issues and conflicts between teams.	**Excellent** Ambassador assisted with conflict resolution caused by sudden, unforeseen changes e.g. problems caused by decision to shift an extra 20% of work from onshore to offshore team.	**Excellent** Ambassador constantly disseminated information to the onshore team members regarding the offshore team's abilities, progress, preferences, and cultural traits. Ambassador held a fortnightly seminar on the above topics.	**Good** Financial cost of relocating ambassador "insignificant" relative to total project cost.	**Good** Customers were very pleased to see and be able to communicate face-to-face with members of the offshore team they were paying for.	**Good** Ambassador simplified communication between teams. Ambassador explained any anomalies or issues with any of the artefacts developed by the offshore team.	**Yes**	**Yes**

Future Research

- Large scale research
- 150 organisations +
- Large repository of experiences with agile
- Allow comparison across practices, across teams, across organisations
- Identification of new practices