Agile Software Development from an Academic Perspective

Professor Tracy Hall
Brunel Software Engineering Lab (BSEL)
Department of Computer Science
Agenda

1. Who am I?
2. Aims of my session
3. The landscape of research on agile
4. What does this agile research tell us?
Who am I?

- Lead the Brunel Software Engineering Lab – BSEL
- Empirical software engineering research:
  - Software defect analysis
  - Human factors in software engineering
- Published many research papers.
- Have had research funding from EPSRC for many projects, including ELFF.
- Collaborate with companies, many of whom use agile
  - Sky Plc, Pitney Bowes
- Program co-chair for XP2016 conference
Aims of my session

• Agile been around since 2001 when the Agile Manifesto published
• Since then a succession of papers reporting successful agile deployment:
  • IBM, Microsoft, SAP, Adobe
• Agile now considered ‘mainstream’
• Much of the support for agile based on industrial experience and judgement
• But what does the research say about agile?

Aim: to give an overview of what the research says about agile
The landscape of research on agile

- Who does the ‘top’ research in agile?
- Where is this ‘top’ research done?
- Where is this research available?
- What is this ‘top’ research on?

Findings from [1] analysing 221 journal articles published and citations 2001-2012….
Assessment of institutions, scholars, and contributions on agile software development (2001–2012)

Sun-Wen Chuang a,⁎, Tainyi Luor a, Hsi-Peng Lub

a Graduate School of Management, National Taiwan University of Science and Technology, 43, Sec. 4, Keelung Road, Taipei 106, Taiwan
b Department of Information Management, National Taiwan University of Science and Technology, 43, Sec. 4, Keelung Road, Taipei 106, Taiwan
Who does the ‘top’ research in agile?

Most active researchers....
1. Professor Tore Dybå
2. Professor Kieran Conboy
3. Professor Sridhar Nerur

Sridhar Nerur
University of Texas at Arlington
Software development, cognition, dynamic capabilities, self-organization
Verified email at uta.edu

<table>
<thead>
<tr>
<th>Title</th>
<th>Cited by</th>
<th>Year</th>
</tr>
</thead>
<tbody>
<tr>
<td>Challenges of migrating to agile methodologies</td>
<td>584</td>
<td>2005</td>
</tr>
<tr>
<td>The intellectual structure of the strategic management field: An author co-citation analysis</td>
<td>232</td>
<td>2008</td>
</tr>
<tr>
<td>A decade of agile methodologies: Towards explaining agile software development</td>
<td>208</td>
<td>2012</td>
</tr>
<tr>
<td>Theoretical reflections on agile development methodologies</td>
<td>200</td>
<td>2007</td>
</tr>
<tr>
<td>Can agile and traditional systems development approaches coexist? An ambidextrous view</td>
<td>97</td>
<td>2008</td>
</tr>
</tbody>
</table>
Who does the ‘top’ research in agile?

Most cited researchers…
1. Professor Tore Dybå

Tore Dybå
Chief Scientist and Research Manager at SINTEF
empirical software engineering, evidence-based software engineering,
software process improvement, agile software development
Verified email at sinft.no - Homepage

<table>
<thead>
<tr>
<th>Title</th>
<th>Cited by</th>
<th>Year</th>
</tr>
</thead>
<tbody>
<tr>
<td>Empirical studies of agile software development: A systematic review</td>
<td>1143</td>
<td>2008</td>
</tr>
<tr>
<td>T Dybå, T Dingsoy</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Information and software technology 50 (9), 833-859</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Evidence-based software engineering</td>
<td>510</td>
<td>2004</td>
</tr>
<tr>
<td>BA Kitchenham, T Dybå, M Jørgensen</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Evidence-based software engineering for practitioners</td>
<td>321</td>
<td>2005</td>
</tr>
<tr>
<td>T Dybå, B Kitchenham, M Jørgensen</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Software, IEEE 22 (1), 58-65</td>
<td></td>
<td></td>
</tr>
<tr>
<td>The future of empirical methods in software engineering research</td>
<td>283</td>
<td>2007</td>
</tr>
<tr>
<td>DKJ Sjøberg, T Dybå, M Jørgensen</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Evaluating pair programming with respect to system complexity and</td>
<td>218</td>
<td>2007</td>
</tr>
<tr>
<td>programmer expertise</td>
<td></td>
<td></td>
</tr>
<tr>
<td>E Ansholm, H Gallis, T Dybå, DKJ Sjøberg</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Software Engineering, IEEE Transactions on 33 (2), 65-96</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
2. Professor Torgeir Dingsøyr

Torgeir Dingsøyr

SINTEF and Norwegian University of Science and Technology
Software engineering, software development, software process, software process improvement, agile software development
Verified email at sintef.no - Homepage

Title | 1-20 | Cited by | Year
--- | --- | --- | ---
**Empirical studies of agile software development: A systematic review**  
T Dybå, T Dingsøyr  
Information and software technology 50 (9), 833-859 | 1143 | 2008

**Knowledge management in software engineering: A systematic review of studied concepts, findings and research methods used**  
FO Bjørsen, T Dingsøyr  
Information and Software Technology 50 (11), 1055-1068 | 229 | 2008

**A decade of agile methodologies: Towards explaining agile software development**  
T Dingsøyr, S Neur, VG Bajajpally, NB Moe  
Journal of Systems and Software 85 (6), 1213-1221 | 208 | 2012

**Applying systematic reviews to diverse study types: An experience report**  
T Dyba, T Dingsøyr, GK Hanssen | 189 | 2007

Citation Indices

<table>
<thead>
<tr>
<th></th>
<th>All</th>
<th>Since 2010</th>
</tr>
</thead>
<tbody>
<tr>
<td>Citations</td>
<td>3827</td>
<td>3027</td>
</tr>
<tr>
<td>h-index</td>
<td>29</td>
<td>20</td>
</tr>
<tr>
<td>i10-index</td>
<td>48</td>
<td>39</td>
</tr>
</tbody>
</table>

Co-authors

Tore Dybå  
Nils Brede Moe  
Geir Kjetil Hanssen  
M. Ali Babar  
Hans van Vliet  
Patricia Logo  
Tor Eirland Emori
3. Professor Sridhar Nerur

Sridhar Nerur
University of Texas at Arlington
Software development, cognition, dynamic capabilities, self-organization
Verified email at uta.edu

<table>
<thead>
<tr>
<th>Title</th>
<th>Cited by</th>
<th>Year</th>
</tr>
</thead>
<tbody>
<tr>
<td>Challenges of migrating to agile methodologies</td>
<td>584</td>
<td>2005</td>
</tr>
<tr>
<td>S Nerur, RK Mahapatra, G Mangalaraj Communications of the ACM 48 (5), 72-78</td>
<td></td>
<td></td>
</tr>
<tr>
<td>The intellectual structure of the strategic management field: An author co-citation analysis</td>
<td>232</td>
<td>2008</td>
</tr>
<tr>
<td>SP Nerur, AA Rashwed, V Natarajan Strategic Management Journal 29 (3), 319-336</td>
<td></td>
<td></td>
</tr>
<tr>
<td>A decade of agile methodologies: Towards explaining agile software development</td>
<td>208</td>
<td>2012</td>
</tr>
<tr>
<td>T Dingisery, S Nerur, VG Balijepally, NB Moa Journal of Systems and Software 85 (6), 1213-1221</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Theoretical reflections on agile development methodologies</td>
<td>200</td>
<td>2007</td>
</tr>
<tr>
<td>S Nerur, VG Balijepally Communications of the ACM 50 (3), 79-83</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Can agile and traditional systems development approaches coexist? An ambidextrous view</td>
<td>97</td>
<td>2008</td>
</tr>
</tbody>
</table>

Citation indices
- All: 1902 Since 2010: 1449
- h-index: 14 Since 2010: 13
- h10-index: 16 Since 2010: 15
Where is this ‘top’ research done?

Universities (number of citations):

1. Sintef ICT, Norway,
2. The Open University, UK
3. University of Texas
4. University of Southern California

Further down the list comes:

13. University of Limerick
Where is this research available?

Journals that most research on agile available from
What is this ‘top’ research on?

- Nearly half of the articles report case studies
  - The rest are simulations, experiments, surveys etc.
- Nearly half of the articles report on agile methodology generally, with:
  - 26% on XP, 11% on TDD
- What does this research tell us?
What does this research tell us?

Look at findings from some ‘top’ papers…
1. Top cited research [2]

Empirical studies of agile software development: A systematic review

Tore Dybå *, Torgeir Dingsøyr

SINTEF ICT, S.P. Andersenstr. 15B, NO-7465 Trondheim, Norway

Received 22 October 2007; revised in revised form 22 January 2008; accepted 24 January 2008

Abstract
Aims of [2]:

1. What is currently known about the benefits and limitations of agile software development?
2. What is the strength of the evidence in support of these findings?
3. What are the implications of these findings for the software industry and the research community?

Methods:

- Systematic Literature Review of published research up until 2005
- Findings from 1,996 studies, 36 rigorous empirical studies analysed

Analysis grouped into:
- Introduction and adoption
- Human and Social Factors
- Perceptions of agile methods
- Comparative studies
Good things about agile reported in [2]

1. Agile practices
   > Can easily be adopted in a wide variety of organisations
   > Are easy to use and work well
   > Can be integrated into more ‘traditional’ project management structures

2. Benefits reported:
   > Customer collaboration, customers like the feedback and quick responses
   > Handling defects
   > Learning via pair programming
   > Thinking ahead for management
   > Estimation

3. Improved productivity reported:
   > XP increased LOC/h
   > XP increased quality

4. Increased job satisfaction
Not so good things about agile reported in [2]

1. Difficult to introduce agile into large complex organisations
2. Mixed views about pair programming
   > Exhausting, inefficient, problematic if big differences between paired developers.
3. XP works best with experienced teams
4. Lack of attention to design and architectural issues
5. The role of on-site customer is stressful and cannot be maintained for long.
6. To be successful:
   > Focus on human and social factors
   > Individual autonomy, team autonomy together with corporate responsibility.
2. Recent research from top UK agile group [3]

Agile Challenges in Practice: A Thematic Analysis

Peggy Gregory\textsuperscript{1}, Leonor Barroca\textsuperscript{2}, Katie Taylor\textsuperscript{1}, Dina Salah\textsuperscript{3}, Helen Sharp\textsuperscript{3}

\textsuperscript{1} University of Central Lancashire, Preston PR1 2HE, UK
\textsuperscript{2} The Open University, Walton Hall, Milton Keynes MK7 6AA, UK
\textsuperscript{3} agregory@uclan.ac.uk, leonor.barroca@enac.ac.uk, dina salah@open.ac.uk

Abstract. As agile is maturing and becoming more widely adopted, it is important that researchers are aware of the challenges faced by practitioners and organisations. We undertook a thematic analysis of 193 agile challenges collected at a series of agile conferences and events during 2013 and 2014. Participants were mainly practitioners and business representatives along with some academics. The challenges were thematically analysed by separate authors, synthesised, and a list of seven themes and 27 sub-themes was agreed. Themes were Organisation, Sustainability, Culture, Teams, Scale, Value and Claims and Limitations. We compare our findings against previous attempts to identify and categorise agile challenges. While most themes have persisted we found a shift of focus towards sustainability, business engagement and transformation, as well as claims and limitations. We identify areas for further research and a need for innovative methods of conveying academic research to industry and industrial problems to academia.

Keywords: Agile Methods, Challenges, Evidence-based Software Engineering
Aims of [3]:

1. To find out the current challenges associated with agile

Methods:

• Used a ‘challenge wall’ at 5 agile events during 2013-14
  • Agile business conference 2013
  • DSDM Members Day 2013
  • XP2014
  • Agile North 2014
  • Agile Business Conference 2014

• Attendees encouraged to contribute their challenges to the wall
  • 194 cards contributed (1 disregarded as inappropriate...
Challenge Wall
A challenge card

Agile is great but ...

Too few Scrum Masters address organisational change

Agile Research Network
Contact: DSDM-agile-research-network@open.ac.uk
Website: www.agileresearchnetwork.org
The most important challenges...

<table>
<thead>
<tr>
<th>Theme (number of challenge cards)</th>
<th>Description of theme</th>
<th>Example challenge card</th>
</tr>
</thead>
<tbody>
<tr>
<td>Misconceptions (n=23)</td>
<td>The multi-faceted aspects of agile are open to many different interpretations</td>
<td>&quot;Shallow Adoption&quot; where practices or processes are followed mechanically without deep insight into underlying values</td>
</tr>
<tr>
<td>Process improvement (n=15)</td>
<td>Once adopted, agile requires on-going change and commitment in order to become sustainable and embedded within teams and the organisation</td>
<td>If it is codified it becomes &quot;bureaucratic&quot; and if not it is too diverse to be taken seriously</td>
</tr>
<tr>
<td>Shortcomings (n=14)</td>
<td>Areas where information is sparse, limited or where methods are used inappropriately</td>
<td>Credibility in areas it has not traditionally used (e.g. public sector)</td>
</tr>
<tr>
<td>Organisational culture(n=13)</td>
<td>The organisation requires a philosophical belief in people over process</td>
<td>Changing from a command and control/mechanistic worldview to a future of autonomous, self-managed agents in a systemic organisation is too much if the system does not change itself – including leaders</td>
</tr>
<tr>
<td>Category</td>
<td>Description</td>
<td>Challenges</td>
</tr>
<tr>
<td>----------</td>
<td>-------------</td>
<td>------------</td>
</tr>
<tr>
<td>Business &amp; IT transformation (n=11)</td>
<td>Requires business and IT to collaborate to establish agility throughout the entire value chain</td>
<td>It’s still seen as an IT methodology. It needs the business teams to buy into it and join the party</td>
</tr>
<tr>
<td>Team practices (n=11)</td>
<td>Uncertainty and perhaps lack of training in specific practices or techniques</td>
<td>How to estimate/ better estimate the effort to support planning?</td>
</tr>
<tr>
<td>Management buy-in &amp; understanding (n=11)</td>
<td>Traditional management may see agile as just another IT method that can be implemented and structured to ‘fit’ existing organisational norms</td>
<td>Misconceptions of what ‘agile’ means. Perceived as quicker by business owners</td>
</tr>
<tr>
<td>Agile in a non-agile environment (n=10)</td>
<td>Teams successfully adopt agile but operate in an environment where wider organisational structures are more traditional</td>
<td>IT is agile but most of the business are not</td>
</tr>
<tr>
<td>Commitment/ Engagement (n=10)</td>
<td>Success can be challenged by lack of awareness or commitment from other stakeholders</td>
<td>It only works if all stakeholders get involved and support the agile process</td>
</tr>
<tr>
<td>Large projects (n=10)</td>
<td>Working at programme level where team practices need to scale across multiple teams in large complex projects</td>
<td>Agility in large projects effecting several applications, platforms, techniques</td>
</tr>
</tbody>
</table>
Another interesting study [4]

The Top 10 Burning Research Questions from Practitioners

Sallyann Freudenberg and Helen Sharp

What research do software practitioners really want?

Software practitioners frequently complain that academic research doesn’t meet their requirements or expectations—in short, that research content of the research itself and whether the research community was finding answers to “the right questions”—particularly whether the concerns being addressed by the research community matched those of most interest and value to practitioners.

At the panel discussion, Sallyann suggested we
Top ten research questions from Freudenberg and Sharp [4]

1. Agile and large projects?
2. What factors can break self-organization?
3. Do teams really need to always be collocated to collaborate effectively?
4. Architecture and agile—how much design is enough for different classes of problem?
5. Hard facts on costs of distribution (in $, £, € and so on)
6. The correlation between release length and success rate?
7. What metrics can we use with minimal side effects?
8. Distributed agile and trust—what happens around 8–12 weeks?
9. Statistics and data about how much money/time is saved by agile?
10. Sociological studies—what were the personalities in successful/failed agile teams?
Drilling down into individual factors…

- Pair programming an important topic in XP and agile…
- Authors are the top researchers in agile…
- Highly cited study…
- Published in top software engineering journal…
Evaluating Pair Programming with Respect to System Complexity and Programmer Expertise

Erik Arisholm, Member, IEEE, Hans Gallis, Tore Dybå, Member, IEEE Computer Society, and Dag I.K. Sjøberg, Member, IEEE

Abstract—A total of 295 junior, intermediate, and senior professional Java consultants (99 individuals and 98 pairs) from 29 international consultancy companies in Norway, Sweden, and the UK were hired for one day to participate in a controlled experiment on pair programming. The subjects used professional Java tools to perform several change tasks on two alternative Java systems with different degrees of complexity. The results of this experiment do not support the hypotheses that pair programming in general reduces the time required to solve the tasks correctly or increases the proportion of correct solutions. On the other hand, there is a significant 84 percent increase in effort to perform the tasks correctly. However, on the more complex system, the pair programmers had a 48 percent increase in the proportion of correct solutions but no significant differences in the time taken to solve the tasks correctly. For the simpler system, there was a 20 percent decrease in time taken but no significant differences in correctness. However, the moderating effect of system complexity depends on the programmer expertise of the subjects. The observed benefits of pair programming in terms of correctness on the complex system apply mainly to juniors, whereas the reductions in duration to perform the tasks correctly on the simple system apply mainly to intermediates and seniors. It is possible that the benefits of pair programming will exceed the results obtained in this experiment for larger, more complex tasks and if the pair programmers have a chance to work together over a longer period of time.

Index Terms—Empirical software engineering, pair programming, extreme programming, design principles, control styles, object-oriented programming, software maintainability, quasi-experiment.

1 INTRODUCTION

The concepts underlying pair programming (PP) are not new [13], [15], [17], [40], but PP itself has only recently attracted significant attention and interest within the software industry and academia. Much of the focus on PP is due to the introduction of extreme programming (XP), in being developed, and reduced duration (a measure of time to market), with only minor additional overhead in terms of total programmer hours (a measure of cost or effort) [25], [30], [32], [44], [46]. One exception is an experiment that showed no positive effects of PP with respect to time taken...
The study...

- 295 professional Java consultants participated
  - 29 consultancies from Norway, Sweden and the UK
- Hired by the study for 1 day in a controlled experiment
  - Performed change tasks on several Java systems
  - Complexity of the systems varied
- Findings:
  - Generally, pair programming does *not*
    - reduce the time taken to develop
    - Increase correct solutions
  - Overall effort increased by 84%
  - For complex systems pair programming increased correct solutions by 48%
    - This increase mainly to junior programmers
Conclusions

1. Many challenges yet to be addressed.
2. Lots of research out there.
3. Lots of research still to do.
4. Is the research accessible/useful to practitioners?
5. Gap between agile practice and agile research?
6. Participation by companies in agile research for the hard questions???
7. Funding for such research???
8. Agile changing the face of software development practice.
Consider contributing to or participating in…

XP 2016 - EDINBURGH

XP2016.org
Questions????

tracy.hall@brunel.ac.uk
References


