

# Hybrid Software Development Approaches in Practice

## A European Perspective

Marco Kuhrmann, Clausthal University of Technology

Philipp Diebold, Bagilstein GmbH

Jürgen Münch, Reutlingen University

Paolo Tell, IT University of Copenhagen

Kitija Trektore and Fergal McCaffery, Dundalk Institute of Technology

Vahid Garousi, Wageningen University

Michael Felderer, University of Innsbruck

Oliver Linssen, FOM University of Applied Sciences for Economics and Management

Eckhart Hanser, Baden-Wuerttemberg Cooperative State University

Christian R. Prause, German Aerospace Center

// The surveyed companies applied hybrid development approaches to specific projects even when company-wide policies for process usage existed. These approaches emerged from the evolution of different work practices and were consistently used regardless of company size or industry sector. //

**AS THERE IS** no one-size-fits-all software development approach, teams and organizations use different approaches to address the manifold challenges of software development projects. They rarely follow the pure approach by implementing a process by the book. For instance, in 2011, West et al. claimed that water-Scrum-fall had become reality for software systems development.<sup>1</sup> Previous studies (see “Related Studies”) as well as experience suggest that this claim is true. However, many questions concerning so-called hybrid development approaches (see “Hybrid Software Development Approach: A Definition”) have yet to be answered decisively. In this article, we are particularly interested in answering the following key questions:

- What different development approaches are used, and in which combinations?
- Why are combinations developed and used?
- Do industry sector and company size matter?

Answers to these questions would enable us to better characterize how and why particular development



approaches are used to address improvement goals and problems in respective areas.

### Objectives and Method

To shed light on the characteristics of hybrid development approaches and to begin answering the previous questions, we initiated an exploratory multistage international research project named *Hybrid Development Approaches in Software Systems Development (HELENA)* and collected data from 69 practitioners across Europe. The questionnaire comprised 25 questions aimed at collecting data on general process use, process use in the context of norms and standards, process improvement, and experiences. Because of the open and explorative nature of the presented study, we intentionally sacrificed full control of the population and accepted the issues that accompany an explorative study. For instance, we did not provide the participants definitions of the terminology used (e.g., method, process, or approach), and we categorized data only during data analysis, according to the definitions from “The Right Degree of Agility in Rich Processes.”<sup>2</sup>

The heart of the questionnaire was a list of 40 different development approaches, which we crafted from several previously conducted studies,<sup>3</sup> including the “State of Agile Report,” found at <https://explore.versionone.com/state-of-agile> (see also “Related Studies”). Of these approaches, we asked the participants to select the ones used in their project. The survey accepted answers from May to June 2016, and the data collection strategy applied was convenience sampling using a number of mailing lists of IT clusters or networks and social media (e.g., Twitter, LinkedIn, Facebook, and ResearchGate) within the relevant



## RELATED STUDIES

In 2016, Vijayasarathy and Butler investigated the question of whether organizational, project, and team characteristics influence the choice of development methods.<sup>S1</sup> They found a huge variety of development methods applied and strong associations between the characteristics and development approaches used. Compared with older studies,<sup>S2, S3</sup> they found an increased use of agile methodologies. However, even in 2003, Jones found substantial diversity in development methods.<sup>S4</sup> In another survey from Turkey, with 202 participants, Garousi et al. found waterfall (53%), agile or lean (45%), and incremental models (38%) to be the most frequently used.<sup>S5</sup> Similar results were found in studies conducted in Germany.<sup>4</sup> We argue that hybrid approaches emerge naturally because of the challenges accompanying a migration to agile.<sup>7, S6</sup>

### References

- S1. L. Vijayasarathy and C. Butler, “Choice of software development methodologies: Do organizational, project, and team characteristics matter?” *IEEE Softw.*, vol. 33, no. 5, pp. 86–94, 2016. doi: 10.1109/MS.2015.26.
- S2. M. Cusumano, A. MacCormack, C. F. Kemerer, and B. Crandall, “Software development worldwide: The state of the practice,” *IEEE Softw.*, vol. 20, no. 6, pp. 28–34, 2003. doi: 10.1109/MS.2003.1241363.
- S3. C. Neill and P. A. Laplante, “Requirements engineering: The state of the practice,” *IEEE Softw.*, vol. 20, no. 6, pp. 40–45, 2003. doi: 10.1109/MS.2003.1241365.
- S4. C. Jones, “Variations in software development practices,” *IEEE Softw.*, vol. 20, no. 6, pp. 22–27, 2003. doi: 10.1109/MS.2003.1241362.
- S5. V. Garousi, A. Coskuncay, A. Betin-Can, and O. Demirörs, “A survey of software engineering practices in Turkey,” *J. Syst. Softw.*, vol. 108, pp. 148–177, Oct. 2015. doi: 10.1016/j.jss.2015.06.036.
- S6. K. Dikert, M. Paasivaara, and C. Lassenius, “Challenges and success factors for large-scale agile transformation: A systematic literature review,” *J. Syst. Softw.*, vol. 119, pp. 87–108, Sept. 2016. doi: 10.1016/j.jss.2016.06.013.



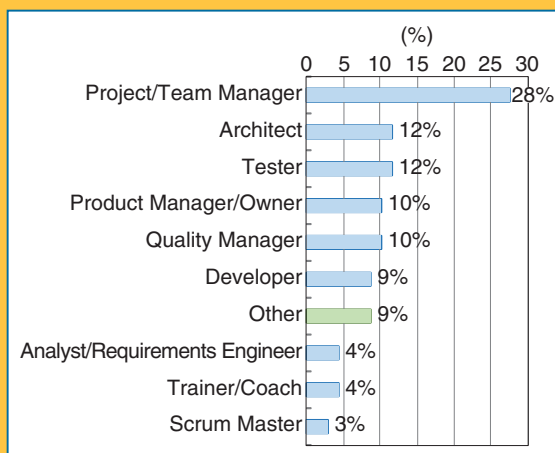
## HYBRID SOFTWARE DEVELOPMENT APPROACH: A DEFINITION

A hybrid software development approach is any combination of agile or traditional (plan-driven or rich) approaches that an organizational unit adopts and customizes to its own context needs (e.g., application domain, culture, processes, project, organizational structure, techniques, and technologies).

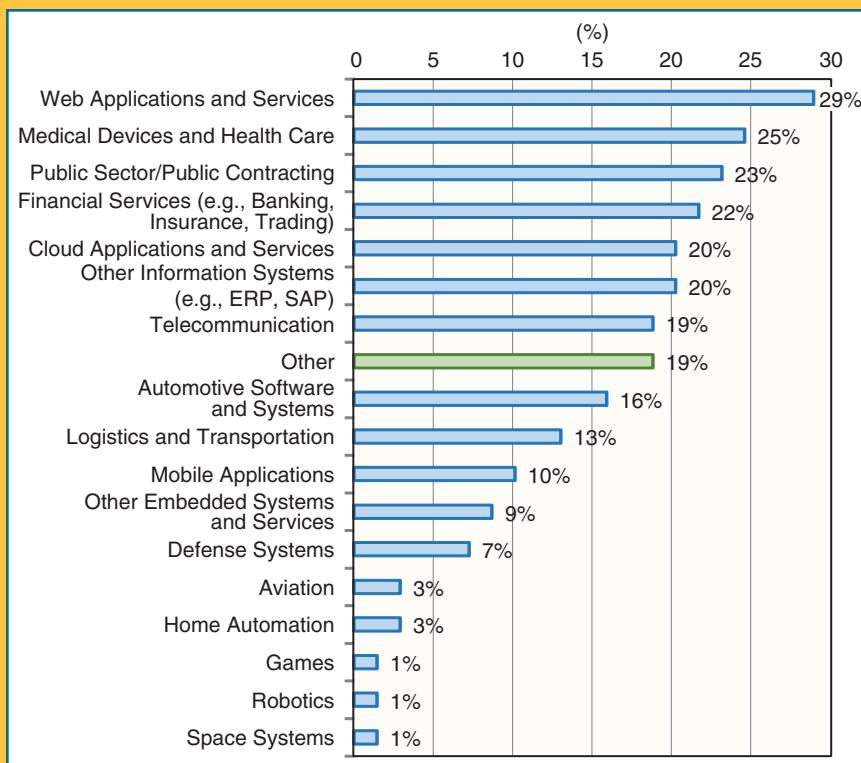
# DEMOGRAPHICS

The Hybrid Development Approaches in Software Systems Development survey comprised 25 questions. It received 69 responses from 16 countries, of which five countries had at least five data points: Germany ( $n = 21$ ), Austria ( $n = 10$ ), Turkey ( $n = 9$ ), Ireland ( $n = 6$ ), and Denmark ( $n = 5$ ). Figure S1 shows the roles of the respondents, of which 28% were project or team managers, representing the largest group. Another 20% were identified as product managers or owners and quality managers (10% each). The more technical roles comprised architects and testers (12% each) and developers (9%). Another 9% were categorized as *other*, which included safety managers, compliance managers, and C-level managers. Hence, the profiles covered the whole software and system development lifecycle.

The respondents' companies covered all sizes and a variety of industry sectors: 17.4% micro-sized (fewer than 10 employees), 20.3% small (11–50 employees), 17.4% medium sized (51–250 employees), 20.3% large (more than 250 employees), and 24.6% very large (more than 2,500 employees). Figure S2 gives an overview of the participants' industry sectors. The data show that the companies, even the small and medium-sized ones, were active in several industry sectors. Finally, our data show that approximately two-thirds of the companies used distributed or virtual teams in software or system development, of which 26.1% worked in a globally distributed way, 20.3% used regionally distributed teams, and 18.8% used nationally distributed teams. The remaining 34.8% did not implement distributed development.



**FIGURE S1.** An overview of the roles held by the survey respondents.



**FIGURE S2.** An overview of the industry sectors covered by the respondents (the questionnaire allowed for multiple selections). ERP: enterprise resource planning.

communities. The full study materials (questionnaire, raw data, etc.) can be obtained from “HELENA SURVEY—Hybrid dEveLopmENt approaches in software systems development.”<sup>8</sup>

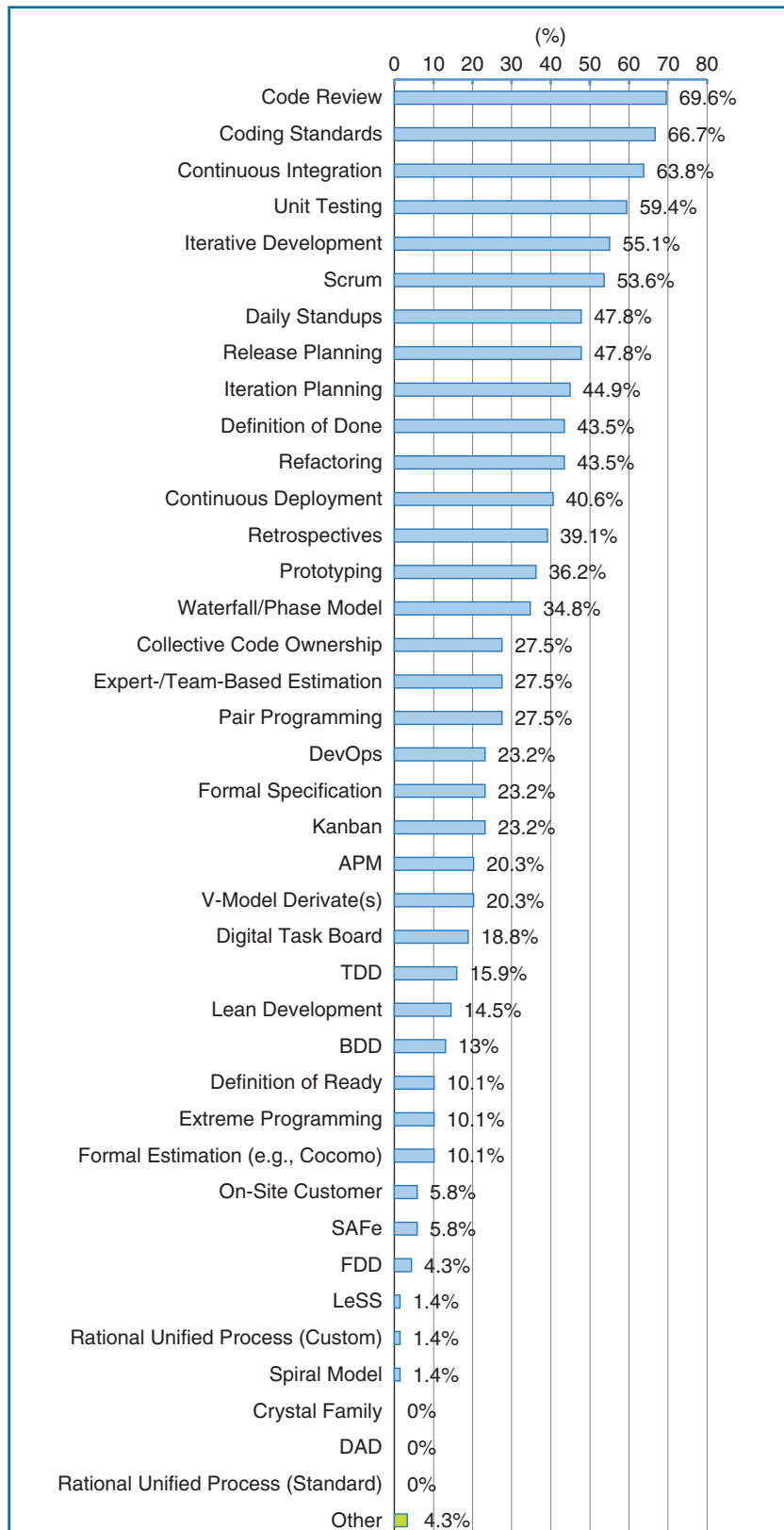
## HELENA Findings

We present the findings of our survey in relation to the key questions presented previously.

### What Different Development Approaches Are Used, and in Which Combinations?

Because the terms *process*, *method*, *approach*, and *practice* are difficult to precisely define, we decided not to provide participants with a precategorized list. Instead, the provided list comprised the 40 development approaches in alphabetical order. The participants were asked to select all of the approaches they used and, if necessary, add further ones. An overview of the basic demographic information is in “Demographics.” Figure 1 shows a large variety of approaches and highlights the ones most frequently selected—for example, code review (69.6%), continuous integration (63.8%), unit testing (59.4%), Scrum (53.6%), and the waterfall or phase model (34.8%).

To analyze the combination of development approaches in detail, we first categorized the different approaches according to the definitions from “The Right Degree of Agility in Rich Processes.”<sup>2</sup> Each approach was categorized as either a method or a practice and as traditional, agile, or both. [The terms *agile* and *traditional* were used in the largely accepted understanding that separates pre-2001 elements (i.e., before the writing of the “Agile Manifesto”<sup>9</sup>) from the ones that emerged afterward. Elements classified as both indicated practices



**FIGURE 1.** An overview of the development approaches applied in practice. The chart is based on 729 selections related to the provided list of 40 different development approaches. APM: agile portfolio management; BDD: behavior-driven development; DAD: disciplined agile delivery; FDD: feature-driven development; LeSS: large-scale Scrum; SAFe: scaled Agile framework; TDD: test-driven development.

Table 1. An overview of the pairwise combination of the different development approaches. The chart is based on 729 selected approaches reported by the 69 survey participants.

			Method															
			Traditional						Agile									
			Rational unified process (custom)	Rational unified process (standard)	Spiral model	V-model derivative	Waterfall or phase model	APM	BDD	DevOps	DAD	Extreme programming	FDD	Kanban	LeSS	Lean development	SAFe	
Method	Traditional	Rational unified process (custom)				1			1			1		1				
		Rational unified process (standard)																
		Spiral model																
		V-model derivative				7	2	1	3		2		3			2		
		Waterfall or phase model					6	2	5	3	2	4			2	2		
	Agile	APM							2	4		2	2	4	1	3	2	
		BDD								5		1		4				
		DevOps										1	1	5		1		
		DAD																
		Extreme programming												1	3	1	2	1
		FDD																
		Kanban														1	5	2
		LeSS															1	1
		Lean development																2
		SAFe																
		Scrum																
	Both	Crystal family																
		Iterative development																
		Prototyping																
Practice	Traditional	Formal estimation (e.g., Cocomo)																
		Formal specification																
	Agile	Collective code ownership																
		Continuous deployment																
		Continuous integration																
		Daily stand-ups																
		Definition of done																
		Definition of ready																
		Digital task board																
		Iteration planning																
		Onsite customer																
		Pair programming																
		Refactoring																
		Retrospectives																
		TDD																
	Both	Code review																
		Coding standards																
		Expert- or team-based estimation																
		Release planning																
Unit testing																		
		Other											2					

© 2013 Pearson Education, Inc. or its affiliate(s). All rights reserved. This publication is protected by copyright. Any unauthorized distribution or reproduction of this work is illegal. All other rights reserved.



and methods that had characteristics of both categories.]

In a second step, we created a matrix to analyze the pairwise combinations. Table 1 shows this matrix and shows our classification of the different approaches as well as the frequency of the combinations. The darker the color of a particular cell, the more often the combination of two approaches occurred in the answers—for example, code review and coding standards (38 instances), the waterfall model and Scrum (10), and Kanban and Scrum (13) were often combined.

Taking the different categories into account, Table 1 clearly shows that traditional and agile approaches were combined with each other and com-

they were combined pairwise. In “Hybrid Software and System Development in Practice: Waterfall, Scrum, and Beyond,” we focused on statistically analyzing different combination patterns.<sup>5</sup> This analysis revealed that a few base methods served as an umbrella for integrating the different approaches. These base methods were the V-model, waterfall or phase model, Scrum, the scaled agile framework, and the generic iterative development. The clusters statistically constructed around these base methods were in line with the two confirmed trends above. In summary, the practitioners used a multitude of different development approaches, which were usually combinations of traditional and agile ones.

projects. In contrast, 20.3% stated that each business unit had its own standards, rules, and policies, and 27.5% stated that each project had the freedom to select the most suitable development approach individually. That is, about half of the participants adopted company-level policies, while the rest selected the development approach more freely.

## Planned Development

On the basis of earlier studies, we expected a combined use, so we asked the participants how a particular combination of approaches was developed. Only 19.6% of the responses stated that the used approach was an outcome of a planned process improvement program. For 83.9% of the responses, the majority of the approaches emerged from experiences collected throughout time. (The percentages add up to more than 100 because participants were allowed to select multiple answers.) That is, the actual approach evolved and was usually not subject to a controlled development—a practice in line with the core principle from the “Agile Manifesto” often referred to as *inspect and adapt*. In this context, 23.2% of the participants also stated that they adapted the approach in response to a specific situation, which is in line with the 27.5% who selected their development approach individually.

## Motivation for a Hybrid Approach

To understand the triggers that motivate practitioners to develop a hybrid approach, we asked the participants for their main reason for such an approach. Regarding the different industry sectors (see “Demographics”), external norms, standards, and rules could be considered drivers behind the development of hybrid approaches. Of the 69 participants, 41 (59.4%) stated

We initiated an exploratory multistage international research project named *Hybrid Development Approaches in Software Systems Development* and collected data from 69 practitioners across Europe.

bbinations occurred among all kinds of development approaches. In summary, with our data, we can confirm the trend previously observed in “Is Water-Scrum-Fall Reality? On the Use of Agile and Traditional Development Practices”<sup>4</sup> that

- different (i.e., traditional and agile) approaches are combined
- agile practices are combined with each other.

Figure 1 and Table 1 show only which approaches were used and how

## Why Are Combinations Developed and Used?

Figure 1 and Table 1 show that different approaches were used and combined. Therefore, we investigated the reasons behind this situation by asking several detailed questions.

### Defined Policies

First, we asked the participants if a company-wide policy concerning process use existed. Of all of the participants, 52.2% stated that their company had a defined process or a set of processes, which was applied to all

**Table 2. The major motivating drivers for implementing hybrid development approaches.\***

Driver	Number of related responses	Examples
Project or product management and commitment	27	Improving the stability of the team, the project type, product lifecycle management, integration of high-level waterfall-like and low-level agile approaches, the lack of readiness of customers (e.g., regarding contracting or pricing), the lack of management buy-in, and business people who are scared of new things
Evolution and pragmatism	18	Constant evolution, a stepwise company transition toward agile development, an increasing number of software parts in complex systems, and the requirements of day-to-day life (“It came up naturally” and “We do what works”)
Project operation and improved flexibility	15	Flexibility of teams and resources, selecting the best-fitting approach, fast product feature evolution, meeting deadlines, and the better handling of volatile requirements
Client constraints	13	The lack of readiness and understanding of agile development, customer satisfaction, and client-domain requirements
Company constraints	Eight	Company size, keeping the existing business running, company history, and company philosophy
Business constraints	Eight	The business context and target domain requirements, e.g., management, standards, and different target domains to be addressed

\* The table shows the major drivers crafted from the 56 participants’ statements of motivating drivers (89 items total).

that standards, norms, and regulations were relevant for their companies or projects. Of these 41 participants, 23 stated that the companies were challenged by implementing the required standards and agility, whereas the remaining 18 stated that implementing agility was not a problem—but fulfilling the requirements of the standards was. In a nutshell, almost two-thirds of the respondents faced requirements brought in by standards, and more than half of these considered the combination of standard-driven and agile development challenging.

### Additional Motivating Drivers

Beyond standards, we asked the participants to report further motivating drivers. In total, 56 participants provided information, which we coded and categorized. Table 2 summarizes the described motivations. The top three categories concerned

project or product management and commitment, evolution and pragmatism, and project operation and improved flexibility. For instance, hybrid approaches were seen as a route toward more stable yet flexible project teams. By including well-known classic approaches, hybrid approaches help improve management commitment, while developers attain flexibility by using agile practices. (This is in line with “Have Agile Techniques Been the Silver Bullet for Software Development at Microsoft?”<sup>6</sup>; see also the section “What Different Development Approaches Are Used, and in Which Combinations?”) Remarkably, 18 statements mentioned evolution and pragmatism. That is, the current approach was neither planned nor designed (see the previous paragraph); it evolved from the different work practices applied, and one participant described it thus: “It is what works.”

## Do Industry Sector and Company Size Matter?

Because of the variety of industry sectors in which software has become key, we were also interested in whether the use of hybrid approaches differed according to either sector or company size and whether a general trend could be identified. An overview of the different industry sectors and company sizes involved in our study can be found in “Demographics.” Table 3 shows a breakdown of our data based on company size and industry sector. The relative distribution of the different approaches is based on the participants’ selection. For example, for the industry sector “web applications and services,” Table 3 reads as follows: “This industry sector was selected by 20 participants, who selected 279 (100%) different development approaches, of which 3.9% were classified as a traditional method, 17.2% as an agile method, 45.2% as an agile practice.”



**Table 3. An overview of the relative use of the different approaches, per company size and industry sector.\***

		Method			Practice				
	Quantity	Traditional	Agile	Both	Traditional	Agile	Both	Other	Approaches selected
Micro (<10)	12	6.6%	15.4%	13.2%	5.5%	33%	25.3%	1.1%	91
Small (11–50)	14	3.2%	16.7%	7.7%	0.6%	47.4%	24.4%	—	156
Medium (51–250)	12	4.9%	16.2%	9.2%	1.4%	40.8%	26.8%	0.7%	142
Large (>250)	14	4.3%	16%	7.4%	3.7%	40.5%	27.6%	0.6%	163
Very large (>2,500)	17	8.5%	15.8%	7.9%	5.1%	38.4%	24.3%	—	177
<b>Average</b>	—	<b>5.5%</b>	<b>16%</b>	<b>9.1%</b>	<b>3.3%</b>	<b>40%</b>	<b>25.7%</b>	<b>0.8%</b>	—
<b>Deviation</b>	—	<b>1.6%</b>	<b>0.3%</b>	<b>1.7%</b>	<b>1.8%</b>	<b>3.5%</b>	<b>1.2%</b>	<b>0.2%</b>	—
Web applications and services	20	3.9%	17.2%	7.2%	3.2%	45.2%	23.3%	—	279
Medical devices and health care	17	7.1%	16.8%	9.2%	4.3%	38%	23.9%	0.5%	184
Public sector or public contracting	16	2.6%	16.2%	8.8%	3.9%	44.3%	24.1%	—	228
Financial services (e.g., banking, insurance, trading)	15	4.2%	16.8%	7.9%	4.2%	42.1%	24.7%	—	190
Cloud applications and services	14	4.5%	21.1%	8%	3%	41.2%	22.1%	—	199
Other information systems (e.g., ERP, SAP)	14	5.7%	19.4%	8.6%	4.6%	39.4%	21.7%	0.6%	175
Telecommunications	13	6%	20.5%	8.6%	4.6%	37.7%	22.5%	—	151
Other	13	7.4%	15.7%	10.2%	0.9%	37%	28.7%	—	108
Automotive software and systems	11	4.4%	17.5%	8%	4.4%	40.1%	25.5%	—	137
Logistics and transportation	9	7.4%	14.8%	8.2%	3.3%	44.3%	22.1%	—	122
Mobile applications	7	2.3%	17.4%	6.8%	2.3%	48.5%	22.7%	—	132
Other embedded systems and services	6	5.4%	18.3%	7.5%	7.5%	36.6%	23.7%	1.1%	93
Defense systems	5	6%	19.4%	9%	4.5%	41.8%	19.4%	—	67
Aviation	2	5.3%	5.3%	15.8%		36.8%	36.8%	—	19
Home automation	2	3.8%	19.2%	11.5%	3.8%	42.3%	19.2%	—	26
Games	1	—	15%	10%		50%	25%	—	20
Robotics	1	4.3%	17.4%	8.7%	4.3%	43.5%	21.7%	—	23
Space systems	1	8.3%	16.7%	8.3%	8.3%	37.5%	20.8%	—	24
<b>Average</b>	—	<b>5.2%</b>	<b>16.9%</b>	<b>9%</b>	<b>4.2%</b>	<b>41.5%</b>	<b>23.8%</b>	<b>0.7%</b>	—
<b>Deviation</b>	—	<b>1.6%</b>	<b>3.3%</b>	<b>2%</b>	<b>1.7%</b>	<b>3.9%</b>	<b>3.8%</b>	<b>0.2%</b>	—

\* For each item, the quantity is given, e.g., the number of companies of a particular size and the number of approaches selected per item. While participants had to select exactly one company size, one company could have been engaged in different industry sectors (multiple selection was enabled). The percentages in this table always refer to the number of selected approaches for the respective category.



**MARCO KUHRMANN** is a senior research associate and division manager at the Clausthal University of Technology's Institute for Applied Software Systems Engineering. Previously, he was an associate professor of software engineering at the University of Southern Denmark. Kuhrmann received a Ph.D. and a habilitation in computer science from the Technical University of Munich. His research interests include hybrid software development methods, product management, software quality and software measurement. Contact him at [kuhrmann@acm.org](mailto:kuhrmann@acm.org).



**PAOLO TELL** is an assistant professor in the IT University of Copenhagen's Computer Science Department. His research interests include software engineering, software processes, (globally) distributed software engineering, computer-supported cooperative work, software architecture, and industry-academia research collaborations. Contact him at [pate@itu.dk](mailto:pate@itu.dk).



**PHILIPP DIEBOLD** is the managing director of Bagilstein GmbH. Previously, he was a researcher in the Process Engineering Department at the Fraunhofer Institute for Experimental Software Engineering. Diebold received an M.S. in informatics from the Fraunhofer Institute for Experimental Software Engineering. His research interests include agility, especially goal- and context-specific agile cherry picking. Contact him at [philipp.diebold@bagilstein.de](mailto:philipp.diebold@bagilstein.de).



**KITIJÄ TREKTERE** is a researcher at the Dundalk Institute of Technology's Regulated Software Research Centre. Trektère received a higher diploma in science in computing. Her research interests include medical device software processes and software process improvement. Contact her at [kitija.trektère@dkit.ie](mailto:kitija.trektère@dkit.ie).



**JÜRGEN MÜNCH** is a professor of software engineering, entrepreneurship, and innovation at Reutlingen University. Previously, he was a professor of software systems in the Department of Computer Science at the University of Helsinki. Münch received a Ph.D. in software engineering from the University of Kaiserslautern, Germany. His research interests include product management, product strategy, business model validation, software startups, Lean Startup, Design Thinking, and agile methods. Contact him at [juergen.muench@reutlingen-university.de](mailto:juergen.muench@reutlingen-university.de).



**FERGAL MCCAFFERY** is the director of the Dundalk Institute of Technology's Regulated Software Research Centre. He is also the competence leader at Lero for medical-device software engineering. McCaffery received a Ph.D. in computer science from the University of Ulster. His research interests include medical device software processes and software process improvement. Contact him at [fergal.mccaffery@dkit.ie](mailto:fergal.mccaffery@dkit.ie).



**VAHID GAROUSI** is an associate professor of software engineering at Wageningen University. Previously, he was an associate professor of software engineering at Hacettepe University, where most of the research reported in this article was conducted. Garousi received a Ph.D. in software engineering from Carleton University. His research interests include software engineering, software testing, model-driven development, and industry-academia research collaborations. He was an IEEE Computer Society Distinguished Visitor from 2012 to 2015. Contact him at [vahid.garousi@wur.nl](mailto:vahid.garousi@wur.nl).



**ECKHART HANSER** is a professor of software engineering at the Baden-Wuerttemberg Cooperative State University (DHBW) Loerrach, the head of DHBW's Competence Center for Agile IT Processes, and a speaker of the German Informatics Society's Process Models group. Hanser received a Ph.D. from the Max-Planck Institute, Stuttgart. His research interests include experimental software engineering, project teams, process models, and quality assurance. Contact him at [hanser@dhbw-loerrach.de](mailto:hanser@dhbw-loerrach.de).



**MICHAEL FELDERER** is a professor at the University of Innsbruck's Institute of Computer Science and a guest professor at the Blekinge Institute of Technology, Sweden. He received a habilitation and a Ph.D. in computer science. His research interests include software quality, software and security processes, as well as data-driven engineering. Contact him at [michael.felderer@uibk.ac.at](mailto:michael.felderer@uibk.ac.at).



**CHRISTIAN R. PRAUSE** is the head of software quality assurance in the Product Assurance Department of the German Aerospace Center's (DLR) Space Administration, Germany's national space agency. His research interests are in ways, methods, and tools to improve software quality, and includes but is not limited to dependability and safety. Contact him at [christian.prause@dlr.de](mailto:christian.prause@dlr.de).

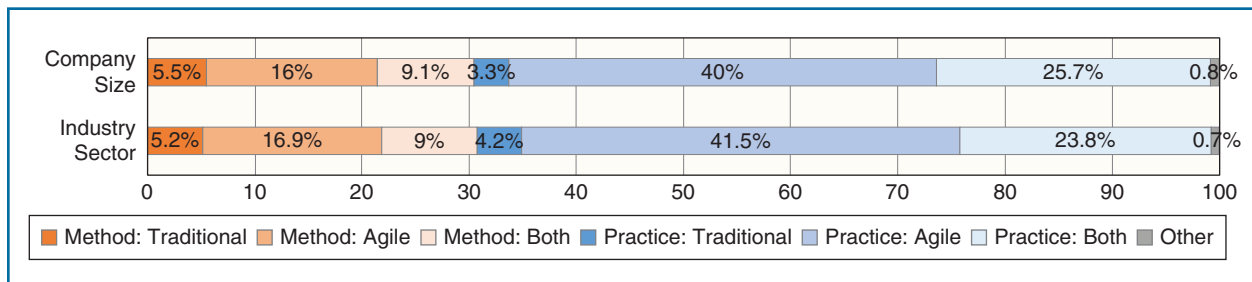


**OLIVER LINSSSEN** is a professor of business informatics at the Institute for IT Management and Digitization, FOM University of Applied Sciences for Economics and Management. He is also an honorary professor at Bergische Universität Wuppertal. Linssen received a Ph.D. in economics. His research interests include project management, requirements engineering, process models and enterprise architecture management. Contact him at [oliver.linssen@fom.de](mailto:oliver.linssen@fom.de).

Based on the data from Table 3, Figure 2 visualizes the averaged data. The figure shows clearly that neither company size nor industry

sector made a difference. Figure 2 shows that traditional and agile approaches were combined with each other regardless of the company size

and industry sector. Moreover, the shares of the different approaches were comparable (for numbers and deviations, see Table 3).




**FIGURE 2.** An overview of the process use categorized according to the approach classification, per company size and industry sector.

**T**raditional software development processes, complemented by norms, standards, and rules, should support development and product quality. However, these have a negative connotation because they limit flexibility to allow for plannability, predictability, compliance, or simpler pricing or contracting models. Compared with agile methods, these approaches are considered heavyweight and inflexible, to the point at which companies are forced to look for alternatives to increase agility.<sup>7</sup>

Hybrid development approaches represent a solution that, regardless of company type and industry sector, enables companies to benefit from both worlds by providing clients and management with a safe environment and developers with the demanded flexibility. Our study shows that hybrid development approaches emerged gradually: 83.9% of the participants stated that the development approach emerged from experience. Moreover, although 52.2% of the participants stated that the company defined a standard approach, more than one-quarter stated that development approaches were selected more individually and adapted during projects in response to given situations.

Our report presents the first step into a deeper investigation on the use of hybrid development approaches.

However, more research is necessary to improve the understanding of the benefits of these approaches. HELENA will continue, and we cordially invite practitioners to participate in future stages of HELENA by sharing their experiences. 

## References

1. D. West, M. Gilpin, T. Grant, and A. Anderson, "Water-Scrum-fall is the reality of agile for most organizations today," 2011. [Online]. Available: <http://www.storycology.com/uploads/1/1/4/9/11495720/water-scrum-fall.pdf>
2. P. Diebold and T. Zehler, "The right degree of agility in rich processes," in *Managing Software Process Evolution: Traditional, Agile, and Beyond—How to Handle Process Change*, M. Kuhrmann, J. Münch, I. Richardson, A. Rausch, and H. Zhang, Eds. Basel, Switzerland: Springer Basel AG, 2016, pp. 15–37.
3. M. Kuhrmann and D. Méndez Fernández, "Systematic software development: A state of the practice report from Germany," in *Proc. 2015 IEEE 10th Int. Conf. Global Software Eng.*, pp. 51–60.
4. G. Theocharis, M. Kuhrmann, J. Münch, and P. Diebold, "Is water-Scrum-fall reality? On the use of agile and traditional development practices," in *Proc. 16th Int. Conf. Product-Focused Software Process Improvement*, 2015, pp. 149–166.
5. M. Kuhrmann et al., "Hybrid software and system development in practice: Waterfall, Scrum, and beyond," in *Proc. 2017 Int. Conf. Software and Systems Process*, pp. 30–39.
6. B. Murphy, C. Bird, T. Zimmermann, L. Williams, N. Nagappan, and A. Begel, "Have agile techniques been the silver bullet for software development at Microsoft?" in *Proc. 7th Int. Symp. Empirical Software Engineering and Measurement*, 2013, pp. 75–84.
7. S. Nerur, R. Mahapatra, and G. Mangalaraj, "Challenges of migrating to agile methodologies," *Commun. ACM*, vol. 48, no. 5, pp. 73–78, 2005.
8. M. Kuhrmann et al., "HELENA SURVEY—Hybrid dEveLopmENt approaches in software systems development," ResearchGate, 2016. [Online]. Available: <https://www.researchgate.net/project/HELENA-SURVEY-Hybrid-dEveLopmENt-Approaches-in-software-systems-development>
9. K. Beck et al., "The agile manifesto," ResearchGate, 2001. [Online]. Available: <https://agilemanifesto.org/>



IEEE COMPUTER SOCIETY  
**DIGITAL LIBRARY**

Access all your IEEE Computer Society subscriptions at  
**computer.org**  
**/mysubscriptions**