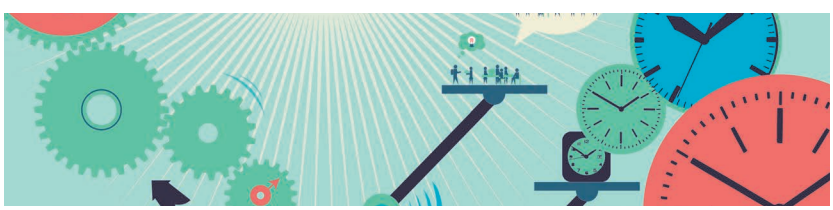
Relationships Between Project Size, Agile Practices, and Successful Software Development

Results and Analysis

Magne Jørgensen, Simula Metropolitan and the University of Oslo

// *Large-scale software development succeeds more often when using agile methods. Flexible scope, frequent deliveries to production, a high degree of requirement changes and more competent providers are possible reasons.* //



Digital Object Identifier 10.1109/MS.2018.2884863
Date of publication: 22 February 2019

A TYPICAL RESPONSE when confronted with increased size and complexity of work is to implement more planning and management formalism.[1] Agile software development methods, on the other hand, aim to remove or reduce much of the traditional project management formalism. Does this mean that agile works mainly for smaller projects,[2] or do agile methods also work well for larger projects?[3, 4] The available empirical evidence does not provide a definitive answer. Additionally, the evidence does not give much insight into when, if at all, agile methods tend to work well for larger projects. This shortage of empirical evidence is the motivation for the survey presented in this article, which attempts to answer the following two questions:

1) How well do larger agile software projects perform compared to smaller projects and nonagile projects?
2) Which agile practices and characteristics are associated with better performance?

The Survey

Respondents and Data Collection

The survey's participants, Norwegian software professionals who visited three different project management seminars in 2016 and 2017, provided information about their last completed projects. Two-hundred sixteen responses were received. After removing the responses that lacked the minimum information needed for analysis, i.e., the budget size category, the development method, and the perceived performance of the project, 196 unique responses remained. The project information was given anonymously, in Norwegian, using

the survey tool Qualtrics. There was a "Don't know" option for all of the project information questions to ensure that respondents answered only when they felt they had sufficient knowledge.

The software professionals had, on average, 13 years of experience, with 70% having eight or more years. Sixty-nine percent the respondents were from the provider side and 31% from the client side. Seventy-one percent had technical roles in the reported project, e.g., architects or developers, and 29% had managerial roles, e.g., product owners, team leaders, and project managers.

Table 1. The project's characteristics.*

Characteristic	Categories
Budget size (used as measure of project size)[1]	Small (<€1 million)
	Medium (€1–10 million)
	Large (>€10 million)
Development method[2]	Agile
	Nonagile
Requirement volatility[3]	High (>30% changes)
	Low (≤30% changes)
Perceived flexibility of scope	High
	Low
Perceived detail of upfront project plan	High
	Low
Perceived detail of upfront requirement specification	High
	Low
Frequency of deliveries to production[4]	>4 per year
	≤4 per year
Contract type	Time and materials
	Fixed price
Perceived provider competence	High
	Low
Perceived client competence	High
	Low

*The full questionnaire is available to interested readers upon request.
[1]The budget size categories (small, medium, and large) are the same as those that separated the effect of agile practices.[4]
[2]There is no commonly accepted definition of what it means to work "agile." I used the respondents' own perception of whether they worked agile or not in the first analysis and added analyses of the effects of different agile practices and characteristics in the second analysis.
[3]The threshold of 30% is based on what was closest to the median level of the perceived amount of requirement change of the projects.
[4]The original categories were "none," "1–4," and "more than 4," where the two first were joined. Note that even nonagile projects, e.g., incremental or timeboxing-based projects, may have deliveries to production during the project execution.

Project Characteristics

The project characteristics requested from the participants are shown in Table 1. The variables shown are those that distinguish successful software projects from failed software projects in an earlier survey.[5] To avoid having too few observations for some categories, the analyzed category "high" ("low") includes both "very high" ("very low") and "high" ("low") responses.

Project Performance

After describing the characteristics of the project, each participant assessed the performance of his or her last completed project as he/she perceived it, using the scale "Very successful—Successful—Acceptable—Problematic—Very problematic" for each of the success dimensions: client benefits (value), cost control, time control, productivity, and technical quality.

To define the project's overall performance, we used the following categorization:

- successful = successful or better on all five success dimensions
- acceptable = acceptable or better on all five success dimensions
- failed = very problematic on at least one success dimension.

Data Collection Challenges

Different participants may have been involved in the same projects, leading to the possibility of duplicate projects in our data set. The variance in organizations of the participants, as analyzed from the list of seminar participants and the typically large

size of their organizations, indicates that the number of duplicates, if any, is very low.

Participants from the client and provider sides as well as participants in different roles, may have different knowledge and perceptions of a project's performance. While this subjectivity may affect the accuracy of the reported success and failure rates, it is less likely to change the direction of the connection between development methods, project size, and project performance.

An examination of the list of participants shows that the majority belong to or worked for large organizations with mainly administrative software applications. Consequently, the results may be valid mainly in this context.

Results

In total, 16% of the software projects were categorized as successful, 52% as acceptable, and 7% as failed. The small- and medium-sized projects had the best performances with 15 and 22% categorized as successful, 55 and 50% as acceptable, and 7 and 4% as failed, respectively. The larger projects had 5% categorized as successful, 41% as acceptable, and 14% as failed. The decrease in project performance with increased project size corresponds to findings in other studies.[6]

Seventy-four percent of the projects were categorized as agile. These projects, as shown in Table 2, had a better average success rate than the nonagile projects for all three size categories. Figure 1 illustrates the effect of this interaction for projects with acceptable performance. An analysis using a general linear model with the variable development method (i.e., agile and nonagile) nested into the variable budget size (i.e., small, medium, and large) suggests that the difference in proportion of acceptable projects is statistically significant, with agile being more successful for small- ($p < 0.01$) and medium-sized ($p = 0.03$) projects, but not for large-sized projects ($p = 0.12$).

The analysis of practices and context characteristics (i.e., factors) potentially connected with better performance of agile projects was conducted. First, the factors more frequently observed in agile than in nonagile projects were identified through a chi-square analysis. These factors may explain the better performance of agile projects even though they have a similar, positive effect on nonagile projects. Secondly, the connection between all of the factors and acceptable project performance

Table 2. The relationship among budget size category, development method, and project performance.*

Project performance	Development method	Small (n = 120)	Medium (n = 54)	Large (n = 22)
Successful (n = 31)	Agile	19%	24%	7%
	Nonagile	0%	19%	0%
Acceptable (n = 102)	Agile	65%	58%	50%
	Nonagile	19%	31%	25%
Failed (n = 13)	Agile	2%	3%	14%
	Nonagile	23%	6%	13%

*The percentages are the proportion of successful, acceptable, and failed projects for projects of the same budget size, category, and development method.

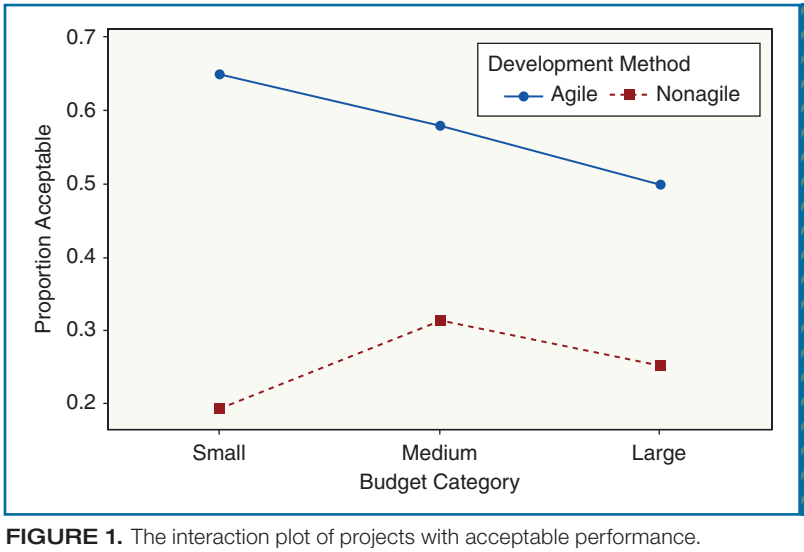


FIGURE 1. The interaction plot of projects with acceptable performance.

Table 3. The proportion of projects with acceptable performance.*

Factor	Category	Agile (n = 146)	Nonagile (n = 50)
Requirement volatility	High (n = 80)	58%	13%
	Low (n = 97)	61%	29%
Delivery frequency	>4 per year (n = 99)	70%	25%
	≤4 per year (n = 60)	49%	21%
Scope flexibility	High (n = 71)	85%	33%
	Low (n = 26)	50%	40%
Detail of project plan	High (n = 59)	67%	18%
	Low (n = 81)	53%	21%
Detail of requirement specification	High (n = 63)	55%	13%
	Low (n = 76)	61%	26%
Contract type	Fixed price (n = 66)	60%	17%
	Time and materials (n = 60)	60%	23%

** The percentages are the proportion of acceptable projects for projects with the same factor category and development method. There are too few observations (low statistical power) for some of the combinations of categories to conduct meaningful tests of statistical significance for the interactions in Table 3. Consequently, the differences should be interpreted as indications of relationships, not as strong evidence.*

(i.e., the performance category with most observations) was analyzed.

The factors associated with a statistically significant (here, set as $p < 0.05$) higher proportion of agile projects were high requirement volatility (50% of agile projects and 33% of nonagile projects had more than 30% requirement changes, i.e., $p = 0.04$), frequent deliveries to production (68% of agile projects and 32% of nonagile projects had more than four deliveries to production per year, i.e., $p < 0.01$) and flexible scope (79% of agile projects and 47% of nonagile projects had a perceived high degree of scope flexibility). There were no statistically significant differences in the proportion of projects with detail of project plan (60% of agile and 53% of nonagile projects were perceived to have little detail in project plans, i.e., $p = 0.75$), detail of requirement specification (55% of agile and 54% of nonagile projects were perceived to have little detail in requirement specification, $p = 0.96$), and contract type (51% of agile and 58% of nonagile used fixed price contracts, $p = 0.53$).

Table 3 shows the results for the proportion of projects with acceptable performance for the analyzed factors. Note that the sum of observations is lower than the full data set of 196 projects because of "Don't know" answers.

The results in Table 3 suggest that experiencing high requirement volatility did not greatly affect the proportion of acceptable agile projects, while the proportion of acceptable nonagile projects decreased from 29 to 13%. Frequent delivery to production seems to have had a much stronger positive connection with better performance for agile than for nonagile projects. This practice was also much more common among agile projects and may therefore contribute to a better performance of agile projects both by being more frequently used and by having a stronger positive connection. Higher scope flexibility was connected with a much higher proportion of acceptable performance for agile projects, and a lower proportion for nonagile projects. The factors, including detail of project plan, detail of requirement specification, and contract type, did not contribute significantly to explaining the improved performance of agile projects.

Table 4 suggests that as the project size increased from small to medium/large, a high degree of requirement changes further increased the superior performance of the agile projects. A higher delivery frequency was associated with a larger increase in acceptable agile than in acceptable nonagile projects. Similarly, a higher flexibility of scope was associated with increased performance of small agile and decreased performance of small nonagile projects.

If agile projects attract more competent providers or clients, this may contribute to the difference between agile and nonagile projects. An analysis of the project data indicates that the agile software projects were perceived to have more competent clients and providers (a chi-square test of independence gives $p = 0.02$ and $p = 0.01$, respectively). A binary logistic regression model with the elements client competence

(high versus low), provider competence (high versus low), development method (agile versus nonagile), requirement volatility (high versus low), delivery frequency (high versus low), and scope flexibility (high versus low), using the performance measure acceptable (1 = acceptable and 0 = not acceptable) as the dependent variable, give much higher likelihoods (odds ratios of 5.7 and 2.4, respectively) of observing an acceptable project when having a high compared to a low/medium competent provider ($p = 0.046$) and client ($p = 0.27$, i.e., not statistically significant). Additional studies are needed to analyze how client and provider competence interact with agile practices and contexts to explain differences in project performance.

Table 4. The proportion of projects with acceptable performance, per size category.*

Factor	Category	Agile		Nonagile	
		Small (n = 94)	Medium/ large (n = 52)	Small (n = 26)	Medium/ large (n = 24)
Requirement volatility	High (n = 80)	62%	54%	13%	14%
	Low (n = 97)	65%	47%	20%	38%
Delivery frequency	High (n = 99)	73%	65%	—	38%
	Low (n = 60)	54%	41%	13%	27%
Scope flexibility	High (n = 71)	86%	84%	14%	—
	Low (n = 26)	55%	40%	57%	—

**The percentages are the proportion of acceptable projects for projects with same factor category, budget size, and development method. There are too few observations (low statistical power) in some of the categories to conduct meaningful tests of statistical significance for the interactions in Table 4. Consequently, the differences should be interpreted as indications of relationships, not as strong evidence. The fields with "—" have fewer than five observations because of missing data about a project or because of few occurrences, and the proportions were not calculated.*

The survey of 196 Norwegian software projects provides empirical support for the use of agile methods on larger as well as smaller software projects, especially when including flexible scope and frequent delivery to production and in contexts with high requirement changes. A contributing factor may be that agile projects tend to have more competent providers and clients.

ABOUT THE AUTHOR

MAGNE JØRGENSEN is a professor of software engineering at the University of Oslo and a chief research scientist at Simula Metropolitan. He has extensive industry experience as a consultant and manager and currently serves on the Norwegian Digitalization Board, where he advises governmental software projects. His research interests include project management, evidence-based software engineering, and expert judgment. He received a Ph.D. from the University of Oslo. Contact him at magnej@simula.no

References

1. J. Child, "Predicting and understanding organization structure," *Administ. Sci. Quart.*, vol. 18, no. 2, pp. 168–185, 1973.
2. T. Dybå and T. Dingsøyr, "Empirical studies of agile software development: A systematic review," *Inf. Softw. Technol.*, vol. 50, no. 9, pp. 833–859, 2008.
3. T. Dingsøyr, N. B. Moe, T. E. Fægri, and E. A. Seim, "Exploring software development at the very large-scale: A revelatory case study and research agenda for agile method adaptation," *Empirical Softw. Eng.*, vol. 23, no. 1, pp. 1–31, 2018.
4. M. Jørgensen, "Do agile methods work for large software projects?" in *International Conference on Agile Software Development*, J. Garbajosa, X. Wang, and A. Aguiar, Eds. Porto, Portugal: Springer-Verlag, 2018, pp. 179–190.
5. M. Jørgensen, "A survey on the characteristics of projects with success in delivering client benefits," *Inf. Softw. Technol.*, vol. 78, pp. 83–94, 2016.
6. C. Sauer, A. Gemino, and B. H. Reich, "The impact of size and volatility on IT project performance," *Commun. ACM*, vol. 50, no. 11, pp. 79–84, 2007.