AGILE OR WATERFALL? IT DEPENDS!

- A DECISION FRAMEWORK FOR AGILE ADAPTION[1]

Joseph Barjis
/PhD, SAFe® SPC, CSP®, PMP®, DEMO® Master/

In the hype of embracing Agile or simply coining to becoming a *pioneer*, companies more likely followed one of these paths:

- Adapted Agile without questioning its rigor to only find out that Agile did not deliver the promised values
- Resisted against it with all means and missed on great opportunities and breakthrough that Agile brought to the industry
- Studied if it makes sense for their business to be Agile and if so, where to start

While the later path sounds more rational, it had its own destiny: Either it was rarely followed OR the study of Agile was based on ad-hoc, personal expertise, or good intuition, and therefore led to less viable conclusions.

In this article, I introduce a framework for decision making and studying when Agile makes sense and when not. Although the proposed framework was developed between 2009 and 2010 and Agile tremendously matured and widely adapted, the principles in this framework can still help Agile leads, mangers and practitioners to make data-driven decision to adapt Agile.

AGILE CHALLENGE

Over the last two decades, a myriad of software development methods[2] has been introduced, which only added to the struggle of organizations in selecting a suitable method or framework. Choosing a software development method is often based on past experiences. For managers to decide what method to use is becoming increasingly difficult. First, software projects are continuously growing in complexity. Second, the lack of a framework for suitability matchmaking forces managers to rely on experience and intuition and even personal preferences.

Formulation of projects characteristics and properties of software development methods used as a framework for decision-making is widely overlooked by researchers and practitioners alike. While most sources discuss the characteristics of software development methods, rarely they make connection between project characteristics and corresponding software development methods.

Here, I present a decision support framework for selecting software development method based on individual projects. In order to do so, first relevant project characteristics are identified. Then, based on the project characteristics, the framework recommends the right software development method.

This framework is designed with two goals in mind:

- Empower Agile transformation practitioners with convincing and data-driven arguments to propose Agile methods.
- Save managers valuable time and money by using the framework for choosing the right method, as far as decision making is concerned.

In the core of this framework lies identification of typical characteristics of software projects that influence the suitability of development methods and mapping both the methods and the characteristics into a framework.

The software development methods considered in this framework are from traditional to Agile such as Waterfall model, Rational Unified Process (RUP), V-model, Spiral model, eXtreme Programming (XP), Scrum, Feature Driven Development (FDD), and Dynamic Systems Development Method (DSDM).

DATA COLLECTION

By interviewing experienced employees from a large organization, Fortune Global 100, and combining the results with information found in literature, a set of major project characteristics has been identified. The interviews were conducted with employees throughout the organization.

Based on this initial composition of project characteristics, a survey was distributed among employees to solicit organization-wide opinion concerning these characteristics. Furthermore, suppliers of the organization were interviewed regarding their insight about the application of various software development methods in practice.

From these interviews and literature study, it became clear that three factors are of significant importance for software projects:

- Project Budget
- Project Duration
- Requirements

The selection and application of software development method has direct impact on these factors. In turn, these factors also influence the suitability of a software development method. Although the influence works both ways, the influence the factors have on the software development method is crucial for the proposed framework, especially the influence they have on the decision of which software development method to use.

From the three factors, it is possible to derive further characteristics that influence the suitability of a software development method. To investigate these additional or derivative characteristics, three methods were used: desktop research or literature review, analysis of existing lessons learned documents, and interviews with professionals.

In order to verify and review the identified characteristics, a follow up survey was conducted among professionals. In this follow up survey, the appropriateness of each characteristic as well as the corresponding weight of each characteristic were established. The final list of characteristics and their averaged weights is presented in Table 1.

Tab. 1: List of characteristics and average weights

Characteristic	Description	Weight
Requirements Maturity	Client wishes maturity. Likelihood of change? Are they feasible?	4,4
Development Stability	Development environment stability. Is technique familiar? Will it change during the project?	4,3
Project Size	The Size of the project. Can be expressed in function points, budget, and project duration.	4,1
Risk Clearness	Are all the possible problems and dangers clear? Will there be no surprises?	3,8
Outsourcing	Are there any outsourcing or vendors used for writing code?	3,5
Scope Clearness	Is project clearly demarcated? Is it likely to change?	3,3
Client's Commitment	The client availability. Is he/she ready and willing to work for the project?	3
Team Relationship	Relationship among all team members. Is communication smooth?	3
Team Size	Architect, programmers, account manager, project manager etc.	2,9
Contracting Method	Fixed Pricing, Time/Material (open contract)	2,9
Stakeholders Flexibility	Is the stakeholder open-minded? Do they want things the familiar way or open for innovations?	2,8

SOLUTION

Based on the characteristics of software development methods and of software projects, the decision support framework was developed. The framework tells what software development method is most suitable regarding a particular project.

Figure 1 illustrates how project size, which is divided into five scales of 1 to 5, is mapped in relation to different methods. Scale 1 represents a smallest size project while scale 5 represents largest size project. For each scale, it is possible to analyze how suitable each software development method is. Three colours indicate a method suitability as follows: Green represents high suitability, yellow medium suitability and orange minimum suitable. If a method is not mentioned on a particular scale, it means that it is not at all suitable in that particular case. On scale 1, small projects, Scrum and XP are found to be most suitable while DSDM and FDD are presented in medium grey grade. Also, Scrum and XP are suitable for small teams. The Waterfall model for example, consists of a high amount of documentation and minimizes face-to-face time, which is therefore less suitable for small projects. On scale 5 of the characteristic project size, the Waterfall model is presented in medium grey. In this case, documentation is highly important.

Scrum	FDD	RUP	RUP	Waterfall
XP	RUP	FDD	Waterfall	V-model
DSDM	DSDM	...	V-model	RUP
FDD	Scrum	...	Spiral	Spiral
1	2	3	4	5

Fig. 1: Mapping development methods on a specific characteristic of a given software project.

The mapping, shown in Figure 1, can be executed in relation to each of the project characteristic in Table 1. We created a spreadsheet-based tool, where all these characteristics, their weights, and development methods are entered.

The first element of the tool, i.e., the decision framework, is the entry form. This entry form consists of the eleven characteristics presented in Table 1. The users will define their project by indicating the scale of each characteristic. The entry form is presented in Figure 2 for a single characteristic.

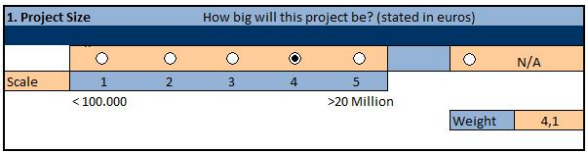

Fig. 2: Example entry form for one characteristic

Almost all of the characteristics are assessed based on 5 scales, plus a 'not available' option. The characteristics *method of contracting* and *outsourcing* consist of two scales. For the method of contracting these are "time & material" and "fixed pricing". For outsourcing the options are "yes" and "no". The "not available" option can be selected when no information regarding a characteristic is available or when it is not a significant factor for a project. If this is selected, the characteristic is removed from the calculation.

Each characteristic is assigned weight suggested by professionals, which can be adjusted project by project based on consensus among developers, clients, project managers, and stakeholders.

When a project is defined as large, more structured and documentation oriented software methods are suitable.

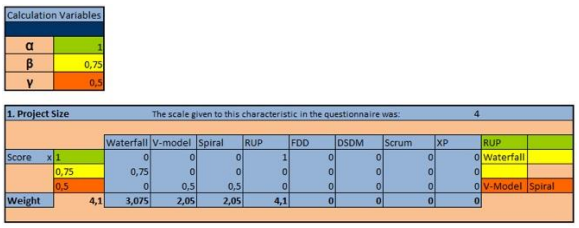

Fig. 3: Example calculation

Figure 3 shows that each software development method receives a score corresponding to its suitability. In the upper part, the calculation scores are presented. When a software development process scores green on a particular scale (suitable), this process receives the score 1. If the process is minimally suitable, the score 0.5 is given, which represents the colour orange. In the example in Figure 3, *project size* is given the scale 4 (which can be found in the blue bar on the far right). This means that it concerns a large project.

The score is then multiplied by the weight of the characteristic, in this case 4.1. This results in higher scores for more important characteristics. For each characteristic this calculation is executed and these are finally summed up. With certain reservation, which will be explained later, a software development method that receives the highest total score is most suitable for that project.

While most of the characteristics are independent, two characteristics are influenced by another characteristic. These dependencies are *team size* on *team relationship* and *requirements maturity* on *scope clearness*, for which a correction is needed. In Figure 4 this is presented for *team relationship*.

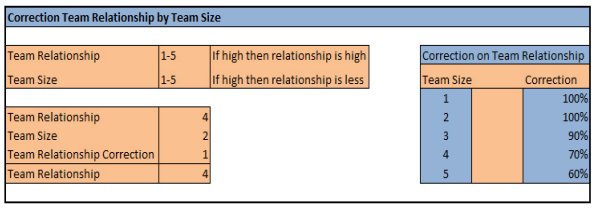

Fig. 4. Example calculation

When the team size becomes big, the relationship within a team gets compromised. Geographical boundaries, lack of communication and opposing values are introduced. This is therefore corrected in the framework. On the right side of Figure 4, the correction percentage is depicted. This shows that when the *team size* is very big, the *team relationship* gets corrected by 60%. In the example in Figure 4, the original score of the relationship was 3. This is then corrected by 60% and results in score 2. This is similar for the dependency between *requirements maturity* and *scope clearness.*

The final element of the decision framework is the result, the main purpose of this framework, to indicate a suitable software development method for a particular project. The conclusion, depicted in Figure 5, presents a final result. In the conclusion five elements are presented. The first element is the table with the total score of the software development methods on all the characteristics. The second element is the graph that represents the same result as given in the table. The conclusion calculation is the third element. This calculates the result given in the fourth element, the final recommendation. Finally, the fifth element states a warning when the most suitable method presented in element four does not at all score on one or more characteristics. The conclusion table and the graph basically show the scoring of each software development method. In this particular example, the RUP scores highest. However, this does not necessarily mean that this process should be used.

Other software development methods, such as FDD and DSDM score high as well. The decision framework does recommend using the RUP for this particular project. The methods FDD and DSDM could be used. It can be concluded that the Waterfall model and the V-model should definitely not be used.

It is possible that the framework shows that RUP is the most suitable process while this process did not at all score (not even minimally) on one or more characteristics. If this is the case it is recommended to analyze the consequences for its suitability. For example, if the RUP scores on all characteristics except on *requirements maturity,* the user should analyze why this is the case.

If the requirements are not mature yet, i.e., an innovative project with many uncertainties is at hand, an agile method is more recommended than RUP. The user can conclude that the activities of requirements engineering (e.g., elicitation, prioritization) need to be more mature. If the requirements are more mature, the suitability of the RUP increases.

APPLICATION

For proof of concept and validation, the framework was applied in two case studies within the same company.

Case Study 1: This case study was considered to be a suitable project for RUP. An RUP expert tested this project on the framework to find whether or not the framework recommends RUP as most suitable and to find which other software development methods might be as well suitable. The framework was therefore tested on its accuracy. The project was defined as Table 2.

When the project was evaluated, the toll concluded as per Figure 6. Only the graph of the conclusion is presented as this is the most intuitive and complete element of the tool.

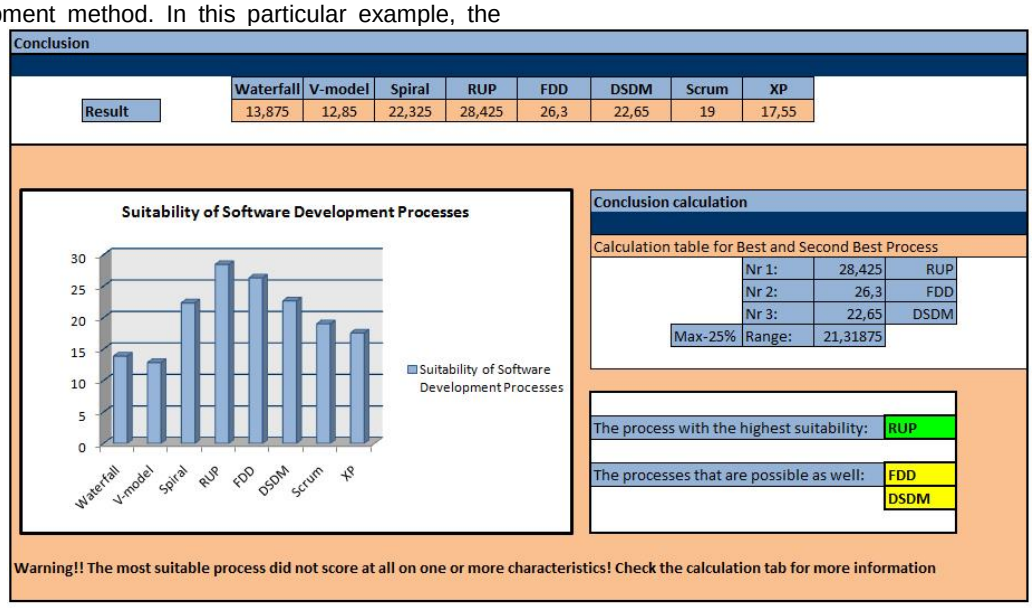

Figure 5: The conclusion frame

Tab. 2: Project for case study 1

Project Characteristic	Scale
Project Size	3
Team Size	3
Requirements Maturity	2
Team Relationship	4
Client's Commitment	3
Scope Clearness	2
Risk Clearness	2
Development Stability	2
Stakeholders Flexibility	2
Method of Contracting	Time/Material
Outsourcing	Yes

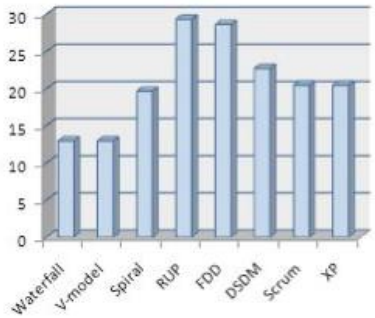

Fig. 6: Suitability of methods for case study 1

The conclusion of the framework was similar to that of the experts' expectations. The RUP is considered the most suitable method for this project. However, FDD is also a very viable for this project. The scales selected are very average (no extremes). The RUP and FDD are both suitable methods to use when a project is between very heavy/traditional and very small/uncertain ones.

Case Study 2: This case study is a project that was considered to be very large with many uncertainties. The project characteristics are defined in Table 3.

For this second project, the suitability graph of the software development methods is depicted in Figure 7. The result of the framework is interesting and complicated. As the graph shows, almost all software development methods are roughly equal in suitability for this particular project. This can be explained by the extreme values of scales given for this project.

However, according to the framework, Scrum and XP both score highest on suitability for the second project.

During the execution of the second project many difficulties and problems arose. This case study is interesting for the decision framework as many organizations struggle with oversized projects in which the risks and scope are difficult to define.

Tab. 3: Project for case study 2

Project Characteristic	Scale
Project Size	5
Team Size	5
Requirements Maturity	1
Team Relationship	2
Client's Commitment	3
Scope Clearness	1
Risk Clearness	2
Development Stability	1
Stakeholders Flexibility	4
Method of Contracting	Time/Material
Outsourcing	Yes

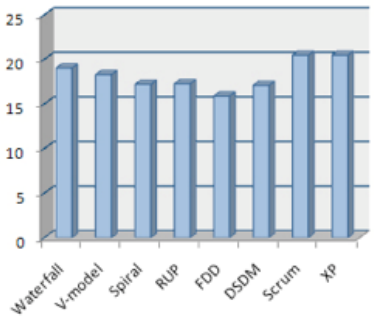

Fig. 7: Suitability of methods for case study 2

FUTURE WORK

This framework was verified with developers, managers, experts, and validated on two real case studies. The case studies were performed in a Global Fortune 100 company, involved in software development. The case studies yielded correct and accurate results that can be considered as a first test and validation of the proposed framework. However, enhancement of this framework and further validation are necessary, for at least two reasons:

- For generalization of the framework and its applicability, it necessitates to conduct more case studies on different organizations, different industries, and different projects.
- Since 2008 and 2010, when the framework was originally published, Agile methods enormously improved both theoretically and practically, grew in popularity, and the developers know more about Agile methods. Even Agile methods made their way into some college courses.

While things certainly changed, maturity, competence, and knowledge of Agile methods have increased, nonetheless, the decision framework is still viable and can be utilized by Agile transformation practitioners.

---

[1] This is a heavily edited and abridged version of our peer refereed and published paper:

Sharon, I., Soares, M.S., Barjis, J., Berg, J. van den, & Vrancken, J. (2010). A Decision Framework for Selecting a Suitable Software Development Process. In the proceedings of ICEIS 2010, June 8 - 12, 2010, Funchal, Madeira – Portugal, volume 3, pp. 34-43.

This framework was developed based on the data and state of the art in software development methods as of 2008-2009, and after 6+ months going through the submission, peer review, and publication process.

[2] Although each has a distinct meaning, the reader may find that the terms such as method, methodology, framework, or process are interchangeably used in software/system development practices that I will mention in this publication. In fact, each of these practices significantly differ in their details, prescriptiveness, and abstractions. Therefore, while in this publication I cumulatively refer to them as a software development method, in fact each might be categorized differently. The same is true in regard to software or system. While I specifically refer to software, the discussed frameworks/methods can be equally used for any system development.