Using Agile Methodologies in People Management

José Manuel Díaz Bossini, Alejandro Ruiz Fernández

Escuela Politécnica Superior

Universidad Carlos III de Madrid

Av. Universidad, 30, 28911 Leganés, Madrid, Spain

jbossini@di.uc3m.es, 100301324@alumnos.uc3m.es

Abstract: *Nowadays, more and more software projects engage Agile Methods. There are so many evidences of success and so many others of failures, but it seems obvious to think that to have a good and strong team is one of the principal reasons to achieve that success. Actually, models like People CMM or Personal Software Process (PSP) focuses on improving the management of the human assets. However our research treat to guess if the Agile Methods could be used to feed the Human Resources Departments to improve the team capabilities and make them stronger. We analyze methods and techniques from some of the most used methodologies like XP, Scrum and Kanban and we study if they would fit into the people and talent management. As result of our study we propose three practices that are susceptible of helping Human Resources Departments by giving extra information that could be used to improve the knowledge about the assets.*

Keywords: *Agile, Human Factors, Talent management, Human Resources, Scrum, Kanban, XP*

1. Introduction

According to the 2009 Report Chaos of the Standish Group [1], fewer than 32 percent of IT projects succeed, nearly 24% are canceled before completion, and the remaining 44 percent are challenged (seriously late, over budget, or lacking expected features).

J.S. Reel points 10 signs that can drive to the failure of a Software project; and that, at least, 7 of those 10 signs are determined before a design is developed or a line of code is written [2]. The signs are:

- Project managers don't understand user needs.
- The project scope is ill-defined.
- Project changes are managed poorly.
- The chosen technology changes.
- Business needs change.
- Deadlines are unrealistic.
- Users are resistant.
- Sponsorship is lost.
- The project lacks people with appropriate skills.
- Managers ignore best practices and lessons learned.

Actually, there are many models that help the software process improvement like the Capability Maturity Model Integration (CMMI), ISO 9000 or even Agile Methodologies. Why then the number of succeed software projects is so reduced?

Well there are so many evidences of both successes [3],[4], [5] and problems or barriers implementing those models in the literature [6],[7]. However, more and more, these and many others articles, point out that one of the success key factors in any Software Process Improvement initiatives is the people factor [8],[9],[10].

In this context, we think that the Agile Methodologies involve both Software Process Improvement and people factors (because strong teams are needed to success). If we use the 10 signs shown by J.S. Reel and searching in the literature for articles, techniques and rules we can see that Agile could fight against at least 8 of them as is shown on table1.

Concerning Human Factors, we will target our focus on two common practices used by the Human Resources Departments; these are the ***competence gaps*** and the ***talent management (TM).***

As Lewis & Heckman says *"practitioners in the field of human resources are now primarily in the business of talent management"* [11]. In their paper they point how hard is to define Talent Management because of the confusion regarding terms and definitions made by authors who write about it.

Why software Project fails?	**Agile solutions**
Project managers don't understand user needs	*"There is often a customer and an end user who wants the resulting product" M. Coram and S. Bohner.*[12] *"Customers decide the scope and timing of releases based on estimates provided by programmers" K. Beck* [13]
The project scope is ill-defined.	*"There is often a customer and an end user who wants the resulting product" M. Coram and S. Bohner.*[12] "You can't program until you know what you're programming" *K Beck.*[13]
Project changes are managed poorly.	*"Responding to change over following a plan" Agile Manifesto* *"By designing for today, an XP system is equally prepared to go any direction tomorrow" K Beck.*[13] *"The increased agility was reflected in the speed with which the developers implemented change requests" Lindvall Mikael et al.* [14]
The chosen technology changes.	*"Before introducing new practices, the development team must understand their effects and implications" Lindvall Mikael et al.* [14]
Business needs change.	*"Responding to change over following a plan" Agile Manifesto.*

Deadlines are unrealistic.	*"*Agile planning is a relatively informal process. For example, deciding what will go into each time-box is accomplished through the daily SCRUM meeting by discussing pending problems, prioritizing work, and assigning resources to the problems*" M. Coram and S. Bohner.*[12]
The project lacks people with appropriate skills.	*"Agile methods derive much of their agility by relying on the tacit knowledge embodied in the team, rather than writing the knowledge down in plans" Barry Boehm.* [15] *"Agile development focuses on the talents and skills of individuals, molding the process to specific people and teams" Alistair Cockburn, Jim Highsmith. [16]*
Managers ignore best practices and lessons learned.	"Agile processes are designed to capitalize on each individual and each team's unique strengths" *Alistair Cockburn, Jim Highsmith.[16]*

Table 3, How Agile can help to reduce the fail factors identified by Reel

However they recognize three different strains of thought regarding TM:

1. Typical human resources practices, functions and activities.
2. A set of processes designed to ensure an adequate flow of employees into jobs throughout the organization.
3. A perspective on TM focuses on talent generically, that is, without regard for organizational boundaries or specific positions.

Nevertheless, another good definition of Talent Management is the one given by Rothwell [16]:

"…a deliberate and systematic effort by an organization to ensure leadership continuity in key positions and encourage individual advancement"

On the other hand, as Rivera-Ibarra, Rodríguez-Jacobo and Serrano-Vargas pointed another important fact to achieve the success, or as they said: "*the quality and innovation of the products and services depend to a great extent on the knowledge, the ability and the talent that software engineers apply in the software development process*"[17].

In this scenario, it seems obvious to think that the management of these competences is not trivial.

As Colomo-Palacios et al. said: *"inefficiencies usually come from inadequate verification of software engineers' competences".*[18]

There are some techniques like the 360-degree and the self-evaluations that help the Human Resources Departments to acquire a better knowledge and manage the skills and the aptitudes of the people that compose their software development teams.

Normally, Human Resources Departments are the ones who should manage Human Factors and try to make strong teams. Our paper, tries to figure out if the reversal could be applied, that is, if Agile techniques applied during the software development could be used to feed the Human Resources Departments helping them to improve the Talent Management and competence gaps.

This paper is structured as follows: Section 2 introduces the Agile methodologies studied Scrum, Kanban and XP and exposes the agile techniques and principles, and the analysis about if they could be used to improve the Management of the Human Factors. Section 3 is used to evaluate our proposal, trying to find the strengths and weaknesses and raises some future works. Finally Section 4 outlines the conclusions.

2. Agile Methodologies

Agile methodologies started to acquire importance in the second half of the 90's. Nevertheless, it wasn't until February 2001 when some developers published the Agile Manifesto. The principles of this Manifesto are:

- Individuals and interactions over process and tools.
- Working code over comprehensive documentation.
- Customer collaboration over contract negotiation.
- Responding to change over following a plan.

These principles don't mean that the second part in the sentence is not important, of course they are, but the items on the left have a little more importance.

As we point above, there are many methodologies that have emerged in the past years, but we center our study in three of them which have been widely used by the software development teams: Extreme Programming (XP), Scrum and Kanban.

2.1. Extreme Programming

Extreme Programming is, maybe, the most recognizable Agile Method [12]. In the XP lifecycle there are five phases and one more known as death:

- **Exploration**: The customer meets the team and they start to create the requirements, called 'user stories'. The team becomes familiar with the tools and technologies that they will use during the development.
- **Planning:** Customer prioritizes the user stories. The development team converts the user stories into small iterations that cover a part of the features and estimates the cost of these iterations.
- **Iterations to release:** This phase includes several iterations over the system to produce a release that must be operational. All the tasks are developed by pairs of developers.
- **Productionizing:** Additional performance and functional tests are performed before deploying the system. In the meantime, new decisions are taken about the features that should be included in next releases.
- **Maintenance:** System must be online while the development team performs new iterations to include the features selected in the productionizing phase.
- **Death:** There are no more user stories to implement, or there is no more budget to accomplish more developments or simply the system is no more profitable. Final documentation is generated and the system won't be evolving anymore.

2.2. KANBAN

The Kanban method has gained momentum recently, mostly due to its linkages to Lean thinking. Lean thinking is normally used in manufacturing, but more and more there is a movement supporting the idea of the use of Kanban in software engineering [19].

Kanban is a Japanese word that means signboard or billboard. Hence that Kanban (applied to the software development) uses a signboard to plan and manage all workflow stages. The basics of Kanban are:

- **Teams can always visualize the workflow:** All the tasks and subtasks are written into post-its and classified into a blackboard. (See fig 1.)
- **The work in progress is limited at each stage.** There are several benefits of limiting the work in progress, for example, early detection of problems, encourage collaboration (if someone is delayed with his/her task and the work in progress limit has been reached, other team members couldn't pass their tasks to the next workflow stage unless the one delayed has been completed).
- **Empirical measure of the average time to complete tasks.** Kanban allow empirically measuring the "*time to completion*" for all those tasks that are planned in the signboard.

2.3. SCRUM

Scrum is an Agile software development that encourages the use of good practices to foment collaborative work, and get the best possible result in a project. Those practices are based in the how to work of high-productive teams.

Scrum is an iterative and incremental methodology. Those iterations are called *sprints.* Each sprint should implement the user stories prioritized by the customer that acts like the project plan. At the beginning of any iteration, the customer could maximize the return of the investment re-planning the user stories that should be implemented in the next sprint.

In the scrum lifecycle we can identify three different stages:

- **The planning meeting:** In which the customer selects the user stories that should be present at the end of the sprint prioritized. The team elaborates the task list necessaries for that sprint to implement those user stories.
- **The sprint itself:** the sprint is restricted to a specific duration (usually 30 days). Every day there is a daily meeting (*daily scrum*) restricted in time as well (15 min) and in which each team member must answer three questions :
  - What have you done since yesterday?
  - What are you planning to do today?
  - Any impediments/stumbling blocks?

- **The retrospective meeting:** After the sprint progress is reviewed and the team lessons learned are identified for the next sprint.

3. Agile practices that could be susceptible of helping Human Resources Departments

According to Hazzan & Hadar there are three aspects that are usually addressed to aim to provide high-quality software:

- **The technical aspect**, whose measures deal with activities such as design, implementation and testing.
- **The managerial aspect** whose measures address, for example, time managerial aspect and schedule.

- **The human aspect** which relates, for instance, to communication among teammates, customer collaboration and learning processes [20].

There are many evidences of measures for technical and managerial aspects but not so many for human aspects [20].

Agile methodologies, stresses the importance of people to succeed as we can see in the agile manifesto principles (Individuals and interactions over process and tools). For this reason, we think that maybe there is a way to reuse the knowledge and expertise acquired during an agile project to generate valuable information for the Human Resources Departments.

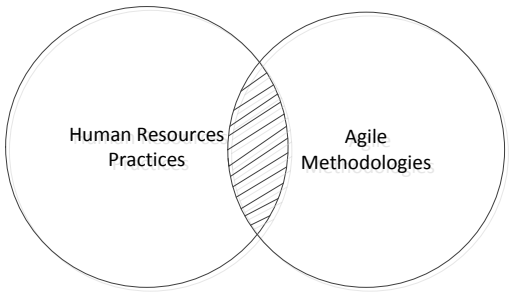

Figure 2. Iis it possible to feed the HR with information of Agile Methodologies?

After the brief review of Scrum, Kanban and XP we have identified some practices that could be susceptible of helping in Talent and Competence gaps management.

The **pair programming** used in XP. The competence gaps management, as we said above, is a hard and even subjective task developed by human resources department. Specific competencies are often ambiguous to managers, and individual perceptions of a firm's competencies may vary significantly. This lack of specificity may mask significant misunderstanding and confusion about competencies [21].

The 360-degree evaluation provides reports from multiple sources and has become a fundamental tool in personnel and human resources management[22]. The 360-degree feedback obtains comprehensive evaluations by considering all those that may reasonably comment on the performance of the individual evaluated, including self-assessment, assessment from below (subordinates/staff), assessment from peers or co-workers as well as assessment from external agents [23].

Based in this kind of evaluations, we propose a cross evaluation between each pair after every time box. It should be agile as well. So, only 3 or 4 questions should be enough to evaluate the work of each partner. The team leader should then record this information and complete it using his own experience. When the project ends this information will be passed to the human resources department, so it could help them to manage teams and evaluate the competence gaps. We propose four questions that could be used in those evaluations:

- Are you comfortable with your partner? Why?
- Point out some technical strengths and some technical weaknesses detected on your partner during this time box.
- What skill of your partner would you improve if you could?
- Did your partner improve the weakness you've detected in the last timebox?

Team leader should record this answers we said above, and in the last timebox make a report with this information to the Human Resource Department and add his/her own evaluations about his team.

Use the empirical measure of the average time to complete task in Kanban to evaluate team members. Kanban allows us to evaluate the average time needed to complete a task [19]; we could use this data to evaluate our team members.

After each task is completed and passed to the next workflow stage, the team leader should record the time needed to complete it, classify it according to the technical field involved (Database, GUI, Core programming and so on) and identify the team member who has completed the task.

If one task is delayed since there is limited work in progress, other team members could help the one who is delayed analyzing and evaluating the task. After their evaluation, they could give the team leader feedback about the issue, who will record the incident inside his evaluation spreadsheet. This empirical information could help Human Resources to manage and make new teams for projects. But also it could help the team leader during the project's lifecycle, helping him to identify strengths and weakness of his team members. This way, Team leader could ask for specific training for his/her team. In addition, recording and managing this information; allows us to build statistical reports to the management.

Use the scrum daily to make a weekly evaluation of the team. The daily scrum gives the scrum master very valuable information that could be grouped and used to manage and evaluate the teams. Our proposal is very similar to that one proposed for Kanban methodology and consists on grouping and analyzing this information according to technical and personal issues. The Scrum Manager at the end of the week will make a review and will fill one spreadsheet with important information about the team. The main differences with the method proposed for Kanban is that the scrum manager should fill a report at the end of each sprint. This report should be used to promote strengths inside the team (exploit the technical knowledge of members to train others, promote those that have aptitudes, and so on) and to fight against their weakness (suggesting training for team members with some technical deficiencies, courses to improve the social skills and so on).

At the end of the project those reports will be sent to the Human Resources Department, and will give them valuable information to manage the talent of their workers.

How hard and complete is the information provided for these methods?

The target of all those proposal practices is to provide as much complete information as they could, and, keep *agilism* inside the project management. So those practices should be agile too, no more 20-30 minutes to group and record the information.

All this information comes from the development process so it is, a priori, empirically true. There should be information about technical strengths and weaknesses as well as social information.

How can Human Resources make use of this information?

Human Resources Departments would have trusted information about the technical knowledge of his resources. So they could use it to assign resources to those projects that need some specific technical skill.

They could use the social information to manage the relationships inside projects. They could form teams based on previously information keeping together people who works well together and moving apart people who have got personal issues.

4. Discussion

As we pointed above, the people factor has been demonstrated to be a critical success factor in software development [8], [4], [7]. So any effort made to improve the management of human factors should be a must.

The main handicap of our approach is that is just a theoretical initiative that should be tested inside an Agile development team to see if the feedback supplied by the project experience could be useful for the Human Resources Departments.

Another important issue is that, our proposals require some extra work for team managers to collect all that information, treat it and make the extra documentation needed.

As strengths we could point that all the information generated will allow team managers to quickly detect issues inside their team, personal or technical. So in last instance, the effort required will be valuable even if it is not useful for the Human Resources Departments.

On the other hand, involving team members into the assessment of competence gaps could be useful to reduce subjectivity of those evaluations performed only by team managers or even high level managers.

The information obtained with our proposal could be useful to Talent Management as well. Those reports should be capable of identifying skills inside team members that weren't identified before: leadership, high technical skills, social skills, and so on. So it could be used to encourage individual advancement as Rothwell said.

5. Future Work

As we said this paper is only a theoretical approach to reuse the information generated by Agile Software Developments to feed the Human Resources Areas. The next obvious step seems to be to select one of the three techniques proposed and implement it inside a real software project and test if the documentation and knowhow acquired helps to improve the Human Resources practices as mentioned.

If this approach succeeds, there are many other methodologies that are susceptible of generating useful information to improve Talent and Competence gaps management. Similarly, there are many practices used in the Human Resources Departments that deserve to be studied like the evaluation of soft and hard skills, and maybe be supplemented with information obtained from Agile projects.

Another different work could be to adapt those techniques to evaluate team leaders as well. By the moment our appraisal is focused mainly on developers but we think it could be used to manage competence of team leaders as well.

It could be interesting; measure the success rate of a development team formed using our techniques and facing it with a development team formed by classic Human Resources practices to figure out if results are better or worst in one or the other team.

Another more ambitious work could be, compare our techniques with some of the most important models in the management of people facts, like People CMM or PSP, and try figure out if there is the possibility of include these practices inside them.

6. Conclusions

According to the 2009 Report Chaos of the Standish Group, there are a high number of failed projects. There are many models used in the Software Process Improvement like CMMI, SPICE, and so on. Nevertheless we center this paper in the Agile context, because as we point above this kind of methodologies could be used to fight against at least 8 of the 10 signs of project failures identified by Reel.

This paper proposes a theoretical and new point of view of Agile Methodologies, trying to use the Agile practices to feed the Human Resources Departments with valuable information that could be used to improve the Talent and the Competence gaps management to build stronger teams.

We made a brief review of the three Methodologies studied: XP, Kanban and Scrum; and we propose three different practices (one for each methodology) to generate useful information for the Human Resources Departments.

Last, we tried to identify strengths and weaknesses of our approach and we proposed some future work that should be made in order to evolve and test the effectiveness of our approach.

References

[1] A. B. Pyster and R. H. Thayer, "Guest Editors' Introduction: Software Engineering Project Management 20 Years Later," *IEEE Software*, vol. 22, no. 5, pp. 18 – 21, Oct. 2005.

[2] J. S. Reel, "Critical success factors in software projects," *IEEE Software*, vol. 16, no. 3, pp. 18 –23, Jun. 1999.

[3] T. Dyba, "An empirical investigation of the key factors for success in software process improvement," *IEEE Transactions on Software Engineering*, vol. 31, no. 5, pp. 410 – 424, May 2005.

[4] M. Niazi, D. Wilson, and D. Zowghi, "Critical success factors for software process improvement implementation: an empirical study," *Software Process: Improvement and Practice*, vol. 11, no. 2, pp. 193–211, 2006.

[5] T. Chow and D.-B. Cao, "A survey study of critical success factors in agile software projects," *Journal of Systems and Software*, vol. 81, no. 6, pp. 961–971, Jun. 2008.

[6] D. B. Huang and W. Zhang, "CMMI in medium #x00026; small enterprises: Problems and solutions," in *2010 The 2nd IEEE International Conference on Information Management and Engineering (ICIME)*, 2010, pp. 171 –174.

[7] A. Shaikh, A. Ahmed, N. Memon, and M. Memon, "Strengths and Weaknesses of Maturity Driven Process Improvement Effort," in *International Conference on Complex, Intelligent and Software Intensive Systems, 2009. CISIS '09*, 2009, pp. 481 –486.

[8] D. E. Perry, N. A. Staudenmayer, and L. G. Votta, "People, organizations, and process improvement," *IEEE Software*, vol. 11, no. 4, pp. 36 –45, Jul. 1994.

[9] N. Baddoo and T. Hall, "Motivators of Software Process Improvement: an analysis of practitioners' views," *Journal of Systems and Software*, vol. 62, no. 2, pp. 85–96, May 2002.

[10] T. Hall, A. Rainer, and N. Baddoo, "Implementing software process improvement: an empirical study," *Software Process: Improvement and Practice*, vol. 7, no. 1, pp. 3–15, 2002.

[11] R. E. Lewis and R. J. Heckman, "Talent management: A critical review," *Human Resource Management Review*, vol. 16, no. 2, pp. 139–154, Jun. 2006.

[12] M. Coram and S. Bohner, "The impact of agile methods on software project management," in *Engineering of Computer-Based Systems, 2005. ECBS '05. 12th IEEE International Conference and Workshops on the*, 2005, pp. 363 – 370.

[13] K. Beck, "Embracing change with extreme programming," *Computer*, vol. 32, no. 10, pp. 70 –77, Oct. 1999.

[14] M. Lindvall, D. Muthig, A. Dagnino, C. Wallin, M. Stupperich, D. Kiefer, J. May, and T. Kahkonen, "Agile software development in large organizations," *Computer*, vol. 37, no. 12, pp. 26 – 34, Dec. 2004.

[15] B. Boehm, “Get ready for agile methods, with care,” *Computer*, vol. 35, no. 1, pp. 64 –69, Jan. 2002.
[16] W. J. Rothwell, *Effective Succession Planning: Ensuring Leadership Continuity and Building Talent From Within*. AMACOM Div American Mgmt Assn, 2010.
[17] J. G. Rivera-Ibarra, J. Rodriguez-Jacobo, and M. A. Serrano-Vargas, “Competency Framework for Software Engineers,” in *2010 23rd IEEE Conference on Software Engineering Education and Training (CSEE T)*, 2010, pp. 33 –40.
[18] R. Colomo-Palacios, C. Casado-Lumbreras, P. Soto-Acosta, F. J. García-Peñalvo, and E. Tovar-Caro, “Competence gaps in software personnel: A multi-organizational study,” *Computers in Human Behavior*, vol. 29, no. 2, pp. 456–461, Mar. 2013.
[19] M. Ikonen, E. Pirinen, F. Fagerholm, P. Kettunen, and P. Abrahamsson, “On the Impact of Kanban on Software Project Work: An Empirical Case Study Investigation,” in *2011 16th IEEE International Conference on Engineering of Complex Computer Systems (ICECCS)*, 2011, pp. 305 –314.
[20] O. Hazzan and I. Hadar, “Why and how can human-related measures support software development processes?,” *Journal of Systems and Software*, vol. 81, no. 7, pp. 1248–1252, Jul. 2008.
[21] A. W. King, S. W. Fowler, and C. P. Zeithaml, “Managing organizational competencies for competitive advantage: The middle-management edge.,” *ACAD MANAGE PERSPECT*, vol. 15, no. 2, pp. 95–106, May 2001.
[22] P. Massingham, T. N. Q. Nguyen, and R. Massingham, “Using 360 degree peer review to validate self-reporting in human capital measurement,” *Journal of Intellectual Capital*, vol. 12, no. 1, pp. 43–74, 2011.
[23] A. H. Church, “Do higher performing managers actually receive better ratings? A validation of multirater assessment methodology,” *Consulting Psychology Journal: Practice and Research*, vol. 52, no. 2, pp. 99–116, 2000.