Trying to Increase the Mature Use of Agile Practices by Group Development Psychology Training — An Experiment

Lucas Gren
Chalmers and the University of Gothenburg
Gothenburg, Sweden 412–92 and
University of São Paulo
São Paulo, Brazil 05508–090
Email: lucas.gren@cse.gu.se

Alfredo Goldman
University of São Paulo
São Paulo, Brazil 05508–090
Email: gold@ime.usp.br

***Abstract*—There has been some evidence that agility is connected to the group maturity of software development teams. This study aims at conducting group development psychology training with student teams, participating in a project course at university, and compare their group effectiveness score to their agility usage over time in a longitudinal design. Seven XP student teams were measured twice (43+40), which means 83 data points divided into two groups (an experimental group and one control group). The results showed that the agility measurement was not possible to increase by giving a 1.5-hour of group psychology lecture and discussion over a two-month period. The non-significant result was probably due to the fact that 1.5 hours of training were not enough to change the work methods of these student teams, or, a causal relationship does not exist between the two concepts. A third option could be that the experiential setting of real teams, even at a university, has many more variables not taken into account in this experiment that affect the two concepts. We therefore have no conclusions to draw based on the expected effects. However, we believe these concepts have to be connected since agile software development is based on teamwork to a large extent, but there are probable many more confounding or mediating factors.**

I. Introduction

Agile Project Management and its methods evolved during the nineties on ideas from lean production and more flexible product development [1], but also from practical experience saving IT projects that were about to fail [2]. The main difference between lean production and agile project management is that both management ideas admit they do not know what the best end-product would look like far in advance [3]. The agile development processes are often intimately connected to high performing, self-managing and mature teams [4] and the way group norms are set has been shown to increase performance [5]. Agile development, as compared to plan-driven ditto, implies more communication and focus on human factors, which make the group psychology aspects of teams a key ingredient [6]. However, the agile processes do not explicitly include the temporal perspective of what happens to all teams over time from a group maturity perspective.

In this experiment, we conducted a longitudinal study of seven agile teams to see if the group development affects process agility. By giving half of the teams training in group psychology theory we hoped to see an effect on their measured agility. However, by only giving a 1.5-hour lecture, we did not see such an effect. We instead discuss reasons for our non-significant results and suggest next steps for future attempts at finding such effects in complex social systems.

We follow Jedlitschka, Ciolkowski and Pfahl's [7] guidelines on how to to report experiments on software engineering throughout this paper. We will therefore start by giving a theoretical background (Section II), describe the experiment in detail (Section III), analyze the data and show descriptive statistics and tests (Section IV), and, finally, discuss the result (Section V) and provide conclusions and suggestions for future work (Section VI).

A. Context

When software development teams transition to an agile approach (i.e. more team-based work) more of the process is dependent on how well the team cooperates [4]. The agile adoption sometimes fails due to the fact that an agile transition is a cultural change as well, which impose new constellations of teams [8], [9]. To further explore the causal relationship between the group dynamics and agile practices over time, would therefore be interesting, both from a research and an industrial perspective, in order to guide agile adoptions better.

B. Problem statement

Many aspects of group dynamics come into play in the team-based workplace [10]. There are studies showing a correlation between group maturity and agile concepts (see e.g. [11]), however, little is known of any causal relationship between them. Correlation analysis only show the connection between the two. If the mature usage of agile practices are directly dependent on group development aspects has not yet been investigated. Therefore, it would be interesting to see if group psychology training of agile software development teams could increase the adoption of concrete agile practices.

II. Background

A. Agile methods (processes)

Agile methodologies can be seen as an approach rather than a technique that mostly change the culture and values behind managing projects. There are some more concrete agile methods, but they all basically share the same values. However, in order to understand how these methods work in practice, we will now shortly present some of the agile practices and how the values are implemented.

a) eXtreme programming (XP): eXtreme programming was the first method created by the agile community and is the most researched method [12] and is considered relatively strict and controlled. The practices that implement the agile principles are [13]:

1) *The planning game.* In the beginning of each iteration, the team, managers, and customers meet and write requirements in form of user stories (written in clear natural language and in a way that everybody can understand). During these meetings the whole group estimates and prioritizes the requirements.
2) *Small releases.* Working software is up and running and delivered very fast and new versions are released continuously, from every few days to every few weeks.
3) *Metaphor.* Customers, managers, and developers model the system after a constructed metaphor or set of metaphors.
4) *Simple design.* Developers are asked to keep design as simple as possible.
5) *Tests.* The development is test-driven (TDD), i.e., the test are written before the code.
6) *Re-factoring.* The code should be revised and simplified over time.
7) *Pair-programming.* All code is written by having two developers per machine.
8) *Continuous integration.* The developers integrate new code into the system as often as possible. However, all code must pass the testing otherwise it is discarded.
9) *Collective ownership.* Developers can change code wherever necessary and the overall code is assessed.
10) *On-site customer.* A customer is in the team all the time to answer questions so the team always works according to what is needed.
11) *40-hour work week.* The team works with a sustainable pace defined as a 40 hour work week. The requirement selected for each iteration should never mean that the team needs to work overtime.
12) *Open workspace.* The team should be collocated and fit in the same room. The layout of the room should make cooperation and communication easy.

b) Scrum: Scrum is based on XP and is one of the more common methodologies and is built on embracing change and focus a lot on delivering value. In Scrum, the project has a prioritized backlog of requirements and use iterative development (called "sprints") to get basic working software for the customer to view as soon as possible. Scrum uses self-organizing teams that get coordinated through daily meetings called "scrums." The manager is called a "Scrum Master" to clarify that it is a facilitating role and not a directive one.

The Scrum methodology consists of three main phases: Pre-sprint planning, sprint (iteration), and post-sprint meeting. All work to be done is kept in a "release backlog" where from requirements (user stories) are taken to the current "sprint backlog." The requirements are usually broken down from a higher abstraction level when the sprint backlog is made. The actual sprint (usually 2–4 weeks) is when the implementation is performed. Here, the sprint backlog is frozen and the team "sprints" to complete what was planned. The team members choose tasks they want to work on themselves. "Scrum meetings" also called "Daily scrums" are 15-minute meetings every morning were the team members check status, report problems, and keep the whole team focused on a common goal. The post-meeting is done to evaluate the process and demonstrate the current system. One important aspect of Scrum is to have small working teams in order to maximize communication, minimize overhead, and maximize the sharing of informal (or tacit) knowledge. The team should also agree and be able to define when something is considered "done" [14].

c) Lean and Kanban: The flexible project management techniques and focus on customer value is not new. Within lean manufacturing these aspects have existed a long time (for more information about lean manufacturing see for example [15]). Many companies combine the process of Scrum with Kanban (Scrum-ban). It is important to note that Kanban is a signal card to pull products through the process within Lean production but has become a software development tool itself [16]. Scrum is a more strict process and can be modified by changing the WIP (work in progress) in each sprint into being connected to the work-flow state to prevent too much WIP. Kanban also allows adding items within each sprint. Another aspect is to change the sprint backlog owned by the team into a Kanban board with multiple teams with work-flow state instead. The Kanban board is never reset after a sprint and can be followed over time, and is also less dependent on collocation. Scrum only allows three different roles of the team, while Kanban does not have a limit. Therefore, larger teams in larger organization with a diversity of specializations often use Kanban or Scrum-ban when possible [17].

d) Crystal: We will not describe the Crystal methodologies in detail but, generally speaking, they are built on the assumption that the main problem in software development is poor communication. Crystal focuses on people, interaction, community, skills, talents, and communication as main effects on performance [18].

The twelve agile principles are a very high-level description of a work environment. Agile software development is an ambiguous concept with descriptions on various levels of abstraction. Many of these are obviously connected to group dynamics. The problem is that these psychological aspects are not described in detail in the methods (processes). This means that this dimension is left out for practitioners to figure our for themselves to a large extent. In order to try to operationalize agility and correlate the measurement to group maturity over time, we enforced the twelve original XP practices (described in Section II-A0a) on all the participating student teams and then opted to use the Perceptive Agile Measurement developed by So and Scholl [19] in order to measure this "agile" behavior over time. All the items are included in Section III-E.

B. Groups and Teams

A group can be defined as: "three or more members that interact with each other to perform a number of tasks and achieve a set of common goals" [20]. If the group is larger all the members might not have a common goal, which means that larger groups often consist of subgroups. Some studies have shown that smaller groups are more productive than larger groups with a threshold at around eight individuals [21]. In psychology, a "work-group" is a group that has a shared view of the group goal and has developed a structure that enables goal achievement. A team, on the other hand, is an effective work-group, however, we will use the terms somewhat interchangeably in this paper, since agile work-groups are called "teams" no matter their actual effectiveness. In social psychology, though, only 17% of all groups were considered teams according to one study [22].

The group research in psychology received much attention after the second world war and before the sixties. After that, the focus in research was on the individual instead of groups [22]. The start of the human factors research in software engineering has also mostly focused on individuals and their personalities and traits for 40 years without finding any coherent results [23]. Therefore, we have reason to believe that much of what happens in software engineering is set on team-level, which means that "agility" is hard to obtain if we do not understand the group dynamics of agile teams, or as Wheelan and Hochberger [22] very adequately put it: "before one jumps to fix something, one has to know what is broken."

During so many years of research on groups in psychology, there are, of course, a diversity of group development models [24]. However, there seems to be a reoccurring patterns of what happens to all types of groups when humans get together in order to solve a task. The first researchers to aggregate models into a general group development model were Tuckman and Jensen [25] in the seventies. In the nineties Susan Wheelan did a similar aggregation of existing models that resulted in the Integrated Model of Group Development that we used in this study. However, Tuckman and Jensen's [25] model with the phases; Forming, Storming, Norming, and Performing correspond well to the stages suggest by Wheelan [22].

C. Wheelan's Integrated Model of Group Development

The Integrated Model of Group Development (or IMGD) describes four different stages that all groups go through in their journey towards becoming a well-functioning high performing team. These stages are illustrated in Figure 1 and described next. The Group Development Questionnaire (the GDQ), that is a measurement of how much energy the group is spending on each development stage, is described afterward.

a) Stage 1 — Dependency and Inclusion: During the first stage of group development (i.e. when the group is new) the group members have more focus on safety and inclusion, a dependency on the designated leader, and more of a wish for order and structure, than in more mature stages. A group at stage one can still get work done, but will focus more on figuring out who the other people are. There is a lack of structure and the group needs to become organized, being able to do efficient work, and achieve the group goals. The group members need to create a sense of belonging and lay the foundation for how to interact within the group. At this first stage, there is a lack of the feeling of belonging to a group, but after this stage people start feeling safe enough to state their ideas and contribute to how they think the group should work in order to achieve its goals. If this does not happen groups stagnate, which is often noticed when group members stop doing work between meetings and even stop attending the group meetings [22].

b) Stage 2 — Counter-Dependency and Fight: During the second stage the group starts having conflict. These differences in opinion is a must in order to create clear roles based on real competence and to make it possible to work together in a constructive manner. The group members have to go through this more turbulent stage in order to build trust. After feeling safe and therefore daring to have these conflicts, a sense of loyalty emerges, which is needed to create cohesion. Since we do not have a clear picture of goals and roles in the beginning, we need this emotional and hard work in order to get shared perceptions of values, norms, and goals, which need to be set on group-level. Since everybody needs to believe in the group values and norms for them to fill their purpose, the rules of the game need to be negotiated so that all members thoroughly believe in them. The more shallow discussions about goals probably present in the first stage, will now be more emotional or include disagreements [22].

c) Stage 3 — Trust and Structure: During the third stage the structure is getting into place and the roles are now actually based on competence instead of status, power, or safety concerns. The communication patterns are more open and also more task-oriented. In this stage the role, organization, and process negotiations are most often more mature and there will be an evident clarification and consensus regarding the group goals. The group members also spend time solidifying positive relationships, and when the tasks are adjusted to competence the likelihood of goal achievement is higher. At this stage the leader's roll goes from needing to have been more directive to being more consultative. The communication structure is also more flexible (i.e. group members talk to whomever they need). Along the group development the content of the communication is also more and more task-oriented instead of relation-oriented. Groups always need the relation-oriented communication since we always need to do the maintenance of discussing how we work together as a group. Therefore, conflict will still occur but be over much faster since the group has better conflict management techniques. Work satisfaction and cooperation increase together with cohesion and trust. At this stage the individual commitment to the group goal will be higher (i.e. we care about what the group is doing on a personal level). This means we will see a voluntary conformity with the norms and helpful deviation from the group (like sub-grouping) will be accepted if considered helpful for the group as a whole [26].

d) Stage 4 — Work and Productivity: The forth stage of group development is when the group does even better with regards to the purpose of stage three. This means that the group focuses on getting the task done well together as well as maintaining group cohesion over a longer period of time. It is important to realize that there is a large set of variables that can and will disturb the group development. Basically, all changes will have such an effect, e.g. change

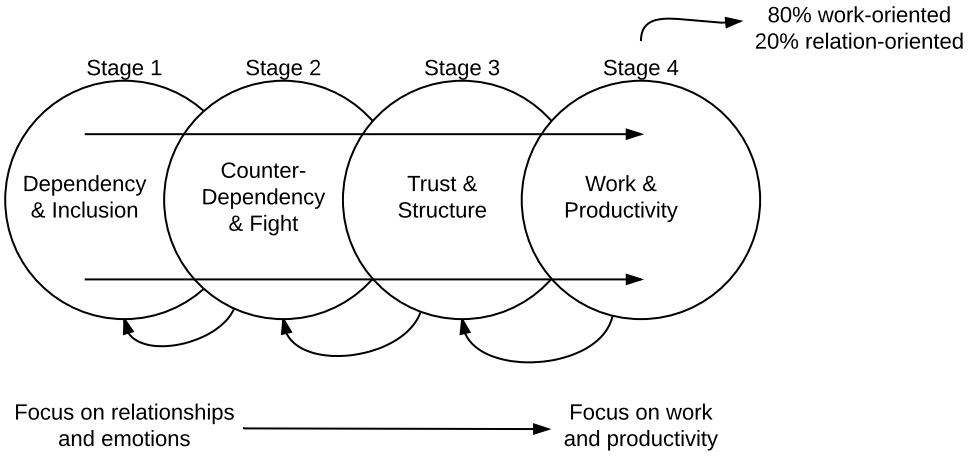

Fig. 1. The Group Development Stages [26]

of demands from the organization, losing staff, getting new staff, and so on and so forth. The challenge in stage four is therefore to try to maintain the effectiveness reached, and the most effective groups do not discuss task-related issues a hundred percent of the time, but actually, still spend around twenty percent discussing how to work together, which is key when maintaining high performance. The characteristics of the decision-making in such teams are participatory and the team encourages task-related conflicts, since they help finding better solutions to problems faced. A person who is or has been on such a team will recognize the intensity of the work and the effectiveness together with a very high interpersonal attraction between group members. People in such high performing teams often look at their work with excitement and joy and getting work done is easy and members have the feeling of being a part of the absolute best team in the world. Getting to stage four takes a lot of work both from internal group members but the group also needs to be given the right conditions from their surrounding ecosystem [26].

D. The Group Development Questionnaire (GDQ)

Wheelan [26] was not the first one who found these characteristic stages of group development, but she contributed with a tool to measure these different stages with four scales put together in a questionnaire. Her tool has made it possible to measure and therefore diagnose where a specific group is focusing its energy from a group developmental perspective. The survey has a total of 60 items and provides a powerful tool for research and interventions in teams. Scale four (GDQ4) is the "work and productivity" and has been shown to correlate with a set of effectiveness measures in different fields. Examples are that groups that have high scores on GDQ4 finish projects faster [27], students perform better on standardized test (SAT scores) if the faculty team scores high on GDQ4 [28], [29], and intensive care staff save more lives in surgery [30].

E. Technology under investigation

The group development stages have been well-known for many years in social psychology [10]. Helping teams to develop and mature in their group development have been shown to increase productivity and effectiveness in a diversity of fields (see Section II-D). Therefore, we want to see if helping teams to mature from a group psychology perspective also gets them to mature in their usage of the XP practices.

There are many group development models, but few have been scientifically validated in the way the group development questionnaire (GDQ) has [22]. Since correlations have been found between the group score on the "work and productivity" scale of the GDQ and other external effectiveness measurements, it would interesting to explore its effect on agile software development teams in their adoption of agile practices. Especially since agile methods have been shown to increase software development project success [31].

Evidence-based interventions within group development with the GDQ have been shown to increase the group maturity in teacher teams [32], and to increase the velocity of group development [33].

F. Relevance to practice

If group development training can be shown to increase the agility of software engineering teams, such aspects would be appropriate to explicitly integrate into the implementation of "agility" in all organizations conducting software development.

III. Experiment planning

A. Goals

The goal of this experiment was to see if training an agile team in group developmental psychology would increase their agility through more mature use of the agile practices.

B. Experimental units

In order to conduct an experiment the study was conducted with 43 students in an agile software development course at the University of São Paulo. Group developmental aspects apply to all group-work and therefore working with students as research subjects is a valid representation of software development conducted by developers on all knowledge levels. However, we would still be careful in generalizing a result to a larger

population than that of developers in the phase of learning an agile approach (i.e. individuals with little experience of agile software development in practice). The course were offered to 3rd year students, however, most students usually take the course in the 4th or 5th (last) year of their software engineering degree. The course is also open to graduate students who are given the possibility to take the course twice during their graduate education.

The student teams in this study comprised students enrolled in a project XP software development course called "The Laboratory of XP" at the Institute of Mathematics and Statistics at the University of São Paulo. The purpose of the course is to introduce agile methods through the use of XP. These methods included, at a minimum, the twelve practices presented in Section II-A0a. Some other staff at the university acted as customers and had to pitch their project ideas to the students, who signed up for the most interesting one from their point of view. All the teams included six to eight members and a more experienced student acting as a an agile coach for the team. The process was put together by the student teams themselves and we allowed any type of additional practices they selected as long as it was within the XP framework. As an example, we enforced collocation of a minimum of eight hours per week.

C. Experimental material

The experimental object was the agile software development team. Group norms and cooperation are set on group level and therefore the actual "team" is the relevant level of analysis.

D. Tasks

The experimental tasks applied in this experiment was for the teams (one team at a time) to listen and reflect on group development theory and discuss its applicability in connection to their own team.

E. Hypotheses, parameters, and variables

The construct used to measure agile practices and the behavior connected to them, was the mature usage of nine agile practices as defined by So and Scholl [19]:

Iterative Planning: (1) All members of the technical team actively participated during iteration planning meetings. (2) All technical team members took part in defining the effort estimates for requirements of the current iteration. (3) When effort estimates differed, the technical team members discussed their underlying assumption. (4) All concerns from team members about reaching the iteration goals were considered. (5) The effort estimates for the iteration scope items were modified only by the technical team members. (6) Each developer signed up for tasks on a completely voluntary basis. (7) The customer picked the priority of the requirements in the iteration plan.

Iterative Development: (1) We implemented our code in short iterations. (2) The team rather reduced the scope than delayed the deadline. (3) When the scope could not be implemented due to constraints, the team held active discussions on re-prioritization with the customer on what to finish within the iteration. (4) We kept the iteration deadlines. (5) At the end of an iteration, we delivered a potentially shippable product. (6) The software delivered at iteration end always met quality requirements of production code. (7) Working software was the primary measure for project progress.

Continuous Integration and Testing: (1) The team integrated continuously. (2) Developers had the most recent version of code available. (3) Code was checked in quickly to avoid code synchronization/integration hassles... (4) The implemented code was written to pass the test case. (5) New code was written with unit tests covering its main functionality. (6) Automated unit tests sufficiently covered all critical parts of the production code. (7) For detecting bugs, test reports from automated unit tests were systematically used to capture the bugs. (8) All unit tests were run and passed when a task was finished and before checking in and integrating. (9) There were enough unit tests and automated system tests to allow developers to safely change any code.

Stand-Up Meetings: (1) Stand up meetings were extremely short (max. 15 minutes). (2) Stand up meetings were to the point, focusing only on what had been done and needed to be done on that day. (3) All relevant technical issues or organizational impediments came up in the stand up meetings. (4) Stand up meetings provided the quickest way to notify other team members about problems. (5) When people reported problems in the stand up meetings, team members offered to help instantly.

Customer Access: (1) The customer was reachable. (2) The developers could contact the customer directly or through a customer contact person without any bureaucratic hurdles. (3) The developers had responses from the customer in a timely manner. (4) The feedback from the customer was clear and clarified his requirements or open issues to the developers.

Customer Acceptance Tests: (1) How often did you apply customer acceptance tests? (2) A requirement was not regarded as finished until its acceptance tests (with the customer) had passed. (3) Customer acceptance tests were used as the ultimate way to verify system functionality and customer requirements. (4) The customer provided a comprehensive set of test criteria for customer acceptance. (5) The customer focused primarily on customer acceptance tests to determine what had been accomplished at the end of an iteration.

Retrospectives: (1) How often did you apply retrospectives? (2) All team members actively participated in gathering lessons learned in the retrospectives. (3) The retrospectives helped us become aware of what we did well in the past iteration/s. (4) The retrospectives helped us become aware of what we should improve in the upcoming iteration/s. (5) In the retrospectives (or shortly afterwards), we systematically assigned all important points for improvement to responsible individuals. (6) Our team followed up intensively on the progress of each improvement point elaborated in a retrospective.

Collocation: (1) Developers were located majorly in... (2) All members of the technical team (including QA engineers, db admins) were located in... (3) Requirements engineers were located with developers in... (4) The project/release manager worked with the developers in... (5) The customer was located with the developers in...

The group maturity (or effectiveness) operationalization was done through using Scale 4 of the GDQ [22]. All the items in the GDQ scale cannot be shared here due to copyright, however, we can include three example items:

- The group gets, gives, and uses feedback about its effectiveness and productivity.
- The group acts on its decisions.
- This group encourages high performance and quality work.

The group development measurement on Scale 4 was assessed on a 5-point Likert scale (1 = low agreement to the statement and 5 = high agreement). The agile items were assesses on a 7-point Likert scale (1 = never and 7 =

always). These scales were used for the simple reason that these measurements were developed and validated using these exact scales.

Both measurements have been validated using a factor analysis [34] and a test for internal consistency (using the Cronbach's α [35]).

The formal research hypothesis for each scale is that the mean values for the scale is different between the two measurements, or $H_1 : \mu_1 \neq \mu_2$.

F. Design

We used a longitudinal research design in order to test differences in group mean value scores on the two measurements over time. The first measurement comprised seven teams and 43 student responses, and the second measurement comprised the same seven teams with 40 responses, i.e. three student were absent during the second measurement.

G. Procedure

The two measurement surveys were distributed to the teams five weeks into their software development projects (during their scheduled and collocated development sessions). The reason was to let the students actually form teams and have done some work before the first measurement. Three of the participating seven teams were randomized into the experimental group and the remaining four teams were used as a control group. The randomization was done by first writing the numbers "3" and "4" on paper slips and letting a person not connected to the experiment draw one folded slip for the research group (three groups were selected). The second step was conducted by writing all team names on other paper slips and letting the person draw three slips to be used for the research group.

On week six, the three selected teams participated in a 1.5-hour group development training with a discussion on the applicability to their own team. During the first hour of the training, the first author of this paper presented The Integrated Model of Group Development [10] and its four group developmental stages. The idea is, briefly, that there are predictable group developmental stages that all groups have to go through in order to work effectively. If team-members are aware of these there is a smaller probability of the team getting stuck on group issues, which leads to quicker and higher quality work [10]. Aspects covered were, for example, goal-setting, role clarification, decision-making, and leadership issues of groups in different development stages.

On week eleven, the second measurement was conducted using the same procedure as in the first measurement.

H. Analysis procedure

The data was analyzed using a general linear model for repeated measures (i.e. a standard repeated measures ANOVA). Such a model assumes normality in data, but since we did not find any significant result, we did not proceed to use non-parametric tests (since these are more restrictive and would therefore neither show any significance).

IV. Results

A. Descriptive statistics

Since we aimed at affecting the agile practices score by conducting group psychology training, we first looked at if we managed to increase the group dynamics score. Since that was not the case we already knew we did not succeed with the intended plan of the experiment. However, we still looked for differences in the agile practices measurement to see if they differed anyways between the two measurements. The only two significant differences we found between the first and second measurements were that the scales "Retrospectives" and "Customer Acceptance Tests." Therefore, we show the descriptive statistics for these scales as well (see Table I).

B. Data set preparation

A mean value was calculated based on the collected data for each individual, and then for the team according the agile practices as defined by So and Scholl [19]. The measured agile practices were: Iteration planning, Iterative development, Continuous integration and testing, Stand-up meetings, Customer access, Customer acceptance tests, Retrospectives and Collocation. The group development Scale 4 individual items were also turned into a mean value for each individual and then for the groups separately. Since we wanted to run the analysis on group-level we only have three mean values in the research group and four values in the control group (seven groups in total).

C. Hypothesis testing

Since we have so few data points, we cannot assess the population distribution based on our sample. However, other studies have shown this kind of data to be normally distributed [19], [22]. Also, since we did not find any significant results based on parametric tests, neither would we for non-parametric tests (since they are more restrictive). We began by testing the group effectiveness score (GDQ4 mean values) for the first and second measurements and can conclude that we did not see a significant effect (see Table II).

We then, still, ran the same analysis for all the agile practices and only found that the scales "Retrospectives" and "Customer acceptance tests" were different between the two measurements overall and not in connection to whether they were in the research group or not (see Table III and Table IV).

We conclude that the group development effectiveness measurement (GDQ Scale 4) was not different between the research group and the control group (not in the first nor the second measurement). The two agile practices "Retrospectives" and "Customer acceptance tests" where both different overall between the two measurements, but not depending on if the teams were in the research group or the control group.

V. Discussion

We did not find any of the expected results in this study. Clearly, just having 1.5 hours of training and discussion is not enough to help the group to develop, even if 1.5 hours of a workweek of 8 hours (like the students in the course had) would be equivalent to 7.5 hours of working full-time 40 hours a week. When taking a closer look at when other experiments

TABLE I. DESCRIPTIVE STATISTICS.

	Research Group	Mean	Std. Deviation	N
GDQ4 1st Measurement	Yes	3.9226	.22378	3
GDQ4 2nd Measurement	Yes	3.8579	.92728	3
GDQ4 1st Measurement	No	3.9007	.19832	4
GDQ4 2nd Measurement	No	4.0055	.33619	4
Retrospectives 1st Measurement	Yes	3.4963	1.54921	3
Retrospectives 2nd Measurement	Yes	5.8500	.42517	3
Retrospectives 1st Measurement	No	4.9280	1.05903	4
Retrospectives 2nd Measurement	No	5.9099	.54201	4
Cust. Accept. Tests 1st Measurement	Yes	3.6319	.38193	3
Cust. Accept. Tests 2nd Measurement	Yes	4.6400	.90598	3
Cust. Accept. Tests 1st Measurement	No	4.3349	.93600	4
Cust. Accept. Tests 2nd Measurement	No	4.8229	.91841	4

TABLE II. ANOVA FOR THE TWO REPEATED GDQ4 MEASUREMENTS.

Source	GDQ4	Type III Sum of Squares	df	Mean Square	F	Sig.
GDQ4	Linear	.001	1	.001	.010	.925
GDQ4 * research_group	Linear	.025	1	.025	.178	.691
Error(GDQ4)	Linear	.694	5	.139		

TABLE III. ANOVA FOR THE TWO REPEATED RETROSPECTIVES MEASUREMENTS.

Source	Retrospective	Type III Sum of Squares	df	Mean Square	F	Sig.
Retrospective	Linear	9.537	1	9.537	19.597	.007
Retrospective * research_group	Linear	1.613	1	1.613	3.314	.128
Error (Retrospective)	Linear	2.433	5	.487		

TABLE IV. ANOVA FOR THE TWO REPEATED CUSTOMER ACCEPTANCE TESTS MEASUREMENTS.

Source	Cat	Type III Sum of Squares	df	Mean Square	F	Sig.
Cat	Linear	1.919	1	1.919	7.018	.045
Cat * research_group	Linear	.232	1	.232	.848	.399
Error(Cat)	Linear	1.367	5	.273		

succeeded in significantly helping the groups to develop and capture their increased maturity, it turns out they did a much larger intervention then was applied in this study. Jacobsson and Wramsten Wilmar [32], for example, gave the groups eight different interventions of group development assessment and enforced improvement points that the group had to work on until the next workshop, plus the students were fully dedicated to only one course. In hindsight, we probably would have needed something similar in order to move the groups forward in the research group. There is also, of course, the possibility of software engineering teams' agility not being as dependent on group maturity as we might think.

A. Threats to validity

We believe the layout of the experiment has potential. Of course, even if we would have found a significant difference, we would still have to have been careful when generalizing the results due to the very small sample size. Regarding construct validity the first author was present during the data collection and could answer any potential questions regarding the questionnaire. However, since we did not want the control group to get training of group development we provided all participants with as little information as possible before the first survey, since we did not want to introduce bias. The trade-off is of course that the participants could have misinterpreted the questions and failed to answer in connection to our intended operationalization of constructs. Regarding learning effects between measurement, the GDQ has been shown to be stable for repeated measurements as such [22]. We have no such studies for the agile practices, which means that we might have seen a learning effect when students answer that part of the survey.

In order to prevent hypothesis guessing, we only informed the participant that the research was about looking at connections between group psychology and agile practices and not more detail on how we expected them to be connected. The internal validity is considered quite high in this experiment since we used validated scales as defined and validated quantitatively by other researchers [19], [22]. However, inter-group communication between the research groups and the control groups is also a threat our experimental research design.

We draw no inference from this experiment. We do not want to state that group development causes more mature use of agile practices, nor the opposite.

B. Lessons learned

The largest lesson learned from this experiment is evidently to check the level of intervention effort needed to move groups forward in their development before conducting this kind of an experiment. We still do not known the effort needed, but the span is more then one 1.5 hours workshop with a second measurement two months later, and less than six to eight workshops of 2–3 hours during a full year with connected action plans and follow-up. By having more time with the teams we could have focused even more concretely on, for example, goal-setting, role clarification, decision-making, functional sub-grouping, or leadership issues, like in [32].

VI. Conclusions and future work

We obtained an insignificant result of this experiment. We therefore have no conclusions to draw based on the expected effects. However, we believe these concepts could still be connected since agile software development is based on teamwork to a large extent. We evidently need a larger intervention effort and, of course there could also be more confounding or mediating factors we have not thought of in the context of agile software development teams.

We would like to redo this experiment with more resources and be able to give the teams in the research group eight times more workshops with connected action plans in order to see if we can get a similar effect as has been shown with teacher teams [32]. It would, of course, be advantageous to include as many teams as possible and at multiple universities and companies to increase the statistical power of the experiment.

References

[1] H. Takeuchi and I. Nonaka, "The new new product development game," *Harvard business review*, vol. 64, no. 1, pp. 137–146, 1986.

[2] J. Sutherland, *Scrum: The art of doing twice the work in half the time*. Random House Business, 2014.

[3] K. Schwaber and M. Beedle, *Agile software development with scrum*. Upper Saddle River, NJ: Prentice Hall, 2002.

[4] G. Melnik and F. Maurer, "Direct verbal communication as a catalyst of agile knowledge sharing," in *Agile Development Conference, 2004*. IEEE, 2004, pp. 21–31.

[5] A. Teh, E. Baniassad, D. Van Rooy, and C. Boughton, "Social psychology and software teams: Establishing task-effective group norms," *IEEE Software*, vol. 29, no. 4, pp. 53–58, 2012.

[6] P. Lenberg, R. Feldt, and L.-G. Wallgren, "Human factors related challenges in software engineering: An industrial perspective," in *Proceedings of the Eighth International Workshop on Cooperative and Human Aspects of Software Engineering*.

[7] A. Jedlitschka, M. Ciolkowski, and D. Pfahl, "Reporting experiments in software engineering," in *Guide to advanced empirical software engineering*. Springer, 2008, pp. 201–228.

[8] J. Iivari and N. Iivari, "The relationship between organizational culture and the deployment of agile methods," *Information and Software Technology*, vol. 53, no. 5, pp. 509–520, 2011.

[9] C. Tolfo and R. Wazlawick, "The influence of organizational culture on the adoption of extreme programming," *Journal of systems and software*, vol. 81, no. 11, pp. 1955–1967, 2008.

[10] S. Wheelan, *Group processes: A developmental perspective*, 2nd ed. Boston: Allyn and Bacon, 2005.

[11] L. Gren, R. Torkar, and R. Feldt, "Group maturity and agility, are they connected? A survey study," in *Proceedings of the 41st EUROMICRO Conference on Software Engineering and Advanced Applications (SEAA)*, 2015.

[12] T. Dybå and T. Dingsøyr, "Empirical studies of agile software development: A systematic review," *Information and software technology*, vol. 50, no. 9, pp. 833–859, 2008.

[13] D. Cohen, M. Lindvall, and P. Costa, "An introduction to agile methods," *Advances in Computers*, vol. 62, pp. 1–66, 2004.

[14] K. Schwaber, "Scrum development process," in *Business Object Design and Implementation*. Springer, 1997, pp. 117–134.

[15] W. Feld, *Lean manufacturing: Tools, techniques, and how to use them*. Boca Raton, Fla.: St. Lucie Press, 2001.

[16] M. Poppendieck, "Lean software development," in *Companion to the proceedings of the 29th International Conference on Software Engineering*. IEEE Computer Society, 2007, pp. 165–166.

[17] C. Ladas, "Scrumban," *Lean Software Engineering-Essays on the Continuous Delivery of High Quality Information Systems*, 2008.

[18] A. Cockburn, *Agile software development: The cooperative game*, 2nd ed. Upper Saddle River, NJ: Addison-Wesley, 2007.

[19] C. So and W. Scholl, "Perceptive agile measurement: New instruments for quantitative studies in the pursuit of the social-psychological effect of agile practices," in *Agile Processes in Software Engineering and Extreme Programming*. Springer, 2009, pp. 83–93.

[20] J. Keyton, *Communicating in groups: Building relationships for group effectiveness*. New York: McGraw-Hill, 2002.

[21] S. Wheelan, "Group size, group development, and group productivity," *Small Group Research*, vol. 40, no. 2, pp. 247–262, 2009.

[22] S. Wheelan and J. Hochberger, "Validation studies of the group development questionnaire," *Small Group Research*, vol. 27, no. 1, pp. 143–170, 1996.

[23] S. Cruz, F. da Silva, and L. Capretz, "Forty years of research on personality in software engineering: A mapping study," *Computers in Human Behavior*, vol. 46, pp. 94–113, 2015.

[24] S. Wheelan and R. Mckeage, "Developmental patterns in small and large groups," *Small Group Research*, vol. 24, no. 1, pp. 60–83, 1993.

[25] B. Tuckman and M. Jensen, "Stages of small-group development revisited," *Group & Organization Management*, vol. 2, no. 4, pp. 419–427, 1977.

[26] S. Wheelan, *Creating effective teams: A guide for members and leaders*, 4th ed. Thousand Oaks: SAGE, 2013.

[27] S. Wheelan, D. Murphy, E. Tsumura, and S. F. Kline, "Member perceptions of internal group dynamics and productivity," *Small Group Research*, vol. 29, no. 3, pp. 371–393, 1998.

[28] S. Wheelan and F. Tilin, "The relationship between faculty group development and school productivity," *Small group research*, vol. 30, no. 1, pp. 59–81, 1999.

[29] S. Wheelan and J. Kesselring, "Link between faculty group: Development and elementary student performance on standardized tests," *The journal of educational research*, vol. 98, no. 6, pp. 323–330, 2005.

[30] S. Wheelan, C. N. Burchill, and F. Tilin, "The link between teamwork and patients' outcomes in intensive care units," *American Journal of Critical Care*, vol. 12, no. 6, pp. 527–534, 2003.

[31] P. Serrador and J. K. Pinto, "Does agile work? – A quantitative analysis of agile project success," *International Journal of Project Management*, vol. 33, no. 5, pp. 1040–1051, 2015.

[32] C. Jacobsson and M. Wramsten Wilmar, "Increasing teacher team effectiveness by evidence based consulting," in *Proceedings of the 14th European Congress of Work and Organizational Psychology (EAWOP)*, May 13–16 2009.

[33] C. Jacobsson and O. Persson, "Group development, what's the speed limit? – Two cases of student groups," in *Proceedings of the 7th Nordic Conference of Group and Social Psychology (GRASP)*, May 20–21 2010.

[34] L. Fabrigar and D. Wegener, *Exploratory Factor Analysis*, ser. Series in understanding statistics. OUP USA, 2012.

[35] L. Cronbach, "Coefficient alpha and the internal structure of tests," *Psychometrika*, vol. 16, no. 3, pp. 297–334, 1951.