Driving Process Improvement Via Comparative Agility Assessment

Laurie Williams[1], Kenny Rubin[2], Mike Cohn[3]
North Carolina State University, Raleigh, NC, USA, williams@csc.ncsu.edu
Innolution, USA, krubin@innolution.com
Mountain Goat Software, USA, mike@mountaingoatsofware.com

Abstract—Rather than striving to be "perfectly agile," some organizations desire to be more agile than their competition and/or the industry. The Comparative Agility™ (CA) assessment tool can be used to aid organizations in determining their relative agility compared with other teams who responded to CA. The results of CA can be used by a team to guide process improvement related to the use of agile software development practices. This paper provides an overview of industry trends in agility based upon 1,235 CA respondents in a range of domains and geographical locations. Additionally, the paper goes further in depth on the results of four industrial teams who responded to the CA, explaining why their results were relatively high or low based upon experiences with the teams. The paper also discusses the resultant process improvement reactions and plans of these teams subsequent to reviewing their CA results.

Keywords-component; agile software development, comparative agility survey

I. INTRODUCTION

Organizations primarily strive to beat out their competition rather than to be "perfect". Some organizations may adopt agile practices to improve their ability to more rapidly and affordably create high quality products and services in their quest to obtain advantage over their competitors. These organizations do not pursue agility for the sake of being "perfectly" agile. Rather, these organizations strive to be more agile than their competition and other industrial organizations because of beliefs they have about the benefits of being agile.

How agile is agile enough? Hypothetically, an organization may be aware that their teams are struggling with the adoption of the test-driven development (TDD) [1] practice that is commonly associated with agile software development. This organization may be comforted if it knew that industry trends indicate low adoption of TDD. Conversely, the organization may heighten its intensity in adopting TDD if, instead, its adoption is lagging behind that of other industrial organizations. The Comparative Agility™ (CA) assessment tool developed by the second two authors, as will be explained in Section II, was created to aid organizations in understanding how their agile

adoption compares with others' agile adoption. As such, teams can use CA as a vehicle for process improvement.

In this paper, we share industry trends in agile adoption based upon 1,235 CA respondents in a range of domains and geographical locations. We provide additional insight on these industry trends by sharing survey results and observations of 119 respondents from four industrial[1] teams working in one large US company. The teams were provided detailed comparison with industry trends. The teams were able to utilize the CA results to guide in process improvement.

Section II provides information about the CA assessment instrument. Section III briefly compares CA with two other assessment instruments. Section IV provides contextual information on survey respondents and on the industrial teams. Section V presents and explains survey responses. We summarize in Section VI.

II. COMPARATIVE AGILITY ASSESSMENT

CA is a survey-based assessment tool used by individuals and organizations wanting to compare their own agility to that of others. Any agile practitioner can visit the CA website[2] and, in exchange for investing his or her time to complete the survey, receive a free report that compares his or her survey results to the complete industry dataset. Alternatively, teams can request[3] to have a customized collector. These team members then individually take the survey using a team-specific survey URL. These teams are provided a team-based comparison against some or all of the remaining dataset.

At the highest level, the CA approach assesses agility on seven *dimensions*: Teamwork; Requirements; Planning; Technical Practices; Quality; Culture; and Knowledge Creating. Each dimension is made up of three to six *characteristics* for a total of 32 characteristics. Each characteristic has approximately four *statements* that are assessed by the respondents for a total of 125 statements.

---

[1] The company prefers not to be identified.
[2] http://www.comparativeagility.com/
[3] Those wishing to obtain a customized collector should contact the second author of this paper.

Results can be obtained/aggregated at the dimension, characteristic, or statement level.

Each statement is an agile practice for which the respondent indicates the truth of the statement relative to their team or organization. For example, one of the seven dimensions, Planning, has five characteristics: (1) planning levels; (2) critical variables; (3) progress tracking; (4) sources of dates and estimates; and (5) when do we plan. The "when do we plan" characteristic has four statements:
• Upfront planning is helpful without being excessive.
• Team members leave planning meetings knowing what needs to be done and have confidence they can meet their commitments
• Teams communicate the need to change release date or scope as soon as they are discovered.
• Effort spent on planning is spread approximately evenly throughout the project.

CA respondents choose the appropriate response given their situation, using a five point Likert scale: True; More true than false; Neither true nor false; More false than true; or False.

Through a combination of dimensions, characteristics, and statements, a team or organization can see how they compare to other organizations, or to themselves at an earlier time. For example, a team doing web development may compare itself to all other teams doing web development and find that they lag their competitors at adopting agile technical practices as shown by their score on the Technical Practices dimension. This information on its own could be enough for the team to structure a process improvement plan. But if the team wants more detail, they can look at the specific characteristics where they most lag their competitors. Example output is shown in Figure 1. The individual respondent or team receives information on the number of standard deviations of their response(s) versus the mean.

CA was designed to lead to actionable results. When an organization can see how it compares with other organizations, improvement efforts can be focused. CA does not provide a view to the purported business results of the respondents based upon their agile adoption.
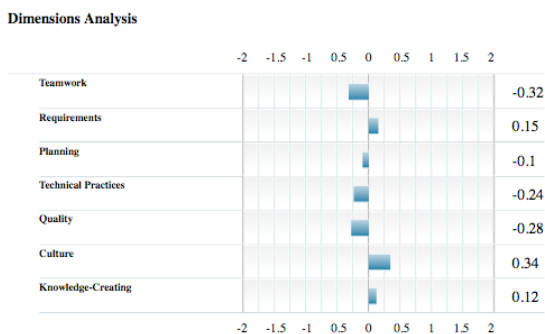

Figure 1: Sample CA Results (number of standard deviations of their response(s) versus the mean)

III. RELATED WORK

Two other assessment frameworks have been used to evaluate agile software development teams. One is the Extreme Programming Evaluation Framework (XP-EF)

[16]. The purpose of the XP-EF is to provide a structure for a case study such that the results of multiple, independent case studies can be combined and compared to create a family of related studies. For example, the results of case studies of industrial Extreme Programming (XP) [2] teams at IBM [16], Sabre Airline Solutions [10], and Tekelec [11] were structured via the XP-EF.

Another assessment framework is the Shodan survey [9]. Similar in intent to CA, the purpose of the Shodan survey is to assess "how XP" a team is. The Shodan survey is specifically focused on the development practices laid out in the XP methodology, and therefore consists of 15 questions rather than CA's more broad 125 questions.

IV. DATA COLLECTION

The industry-wide data reported in this paper are based upon 1,235 respondents who had taken the survey between August 21, 2007 and February 24, 2010. Surveys were deleted from the database if they were anonymous surveys that were not part of customized collectors and the respondent had answered 20% or less of the questions. A slight majority (54%, N=669) of the responses came from teams who asked for their data to be analyzed via a customized collector, such as teams undergoing coaching and/or training by one of the authors. In other cases (46%, N=566), individuals found the Comparative Agility survey site, such as after seeing articles written about the survey [5], and decided to answer the questions. Based upon these circumstances, as a whole the respondents are considered to be part of agile teams or teams beginning an agile transition and not members of anti-agile and/or plan-driven [4] teams.

In this section, we provide demographic and contextual information about survey respondents from the four teams. The teams are all employees of the same successful US-based company with a positive company culture and job security. All teams work on data-intensive applications. Facilities in this company are structured such that each employee has their own office, which the employees have come to value. Office space constraints prevent additional collaborative team spaces to be created. Employees primarily have desktop computers and not laptops, preventing ease of impromptu collaborative work sessions. Top management at the company was supportive of a transition to agile software development, though had not issued a directive for teams to become agile.

A. Team 1

This team was the company's pilot of agile software development beginning in early 2007. The team has focused on the use of the Scrum [12] project management practices rather than on the technical practices. They are a small ten-person team consisting of a development manager, five developers, two testers, a product manager, and a project manager. Seven of the team members are co-located in one building. Three of the developers work out of their homes in other geographic locations in the US, one in the same time zone as the rest of the team and the other with only a one-hour time zone difference.

The product manager works closely with the team on a daily basis to clarify requirements and provide acceptance test criteria. The responsibilities of the ScrumMaster are split between the project manager and the development manager, though the Scrum methodology would not advocate such a split. The team develops a product that

ships in conjunction with a corporate release of a larger product every 18-24 months.

This team's pilot was considered a success, which lead to the spread of agile software development to many other teams in the company. In the last pre-agile release, the team delivered 80% of the functionality committed in the release plan. In the first agile release, the team delivered 137% of the functionality committed in the release plan. In the first agile release, the product manager was able to remove requirements from the original release plan that were no longer desired and to add additional market-driven requirements, a practice that had not happened previously.

Nine of ten team members responded to the survey.

B. Team 2

This team began its transition to agile software development in August 2009. This team consists of approximately 63 members who primarily work in one location. However, the team has strong technical dependencies on components developed at another company location (same time zone) and in India. This team further subdivided into four sub-teams

The team began its agile transition with a new release of a product in which the old product was going to be re-architected from scratch. The team felt, at the time, that it had an unachievable release objective and felt uneasy about also using a radically different development process given this challenge. Some team members were positive about using agile as the only feasible means of delivering a product as scheduled while others were against the use of agile, and remain so after six months using agile. The role of ScrumMaster began with the project managers and then transitioned to development technical leads after approximately four months. Some agile methodologists do not advocate the role of ScrumMaster being filled by a technical team member [6].

The larger team was subdivided into four smaller teams. One of these teams was a "research team." The purpose of the research team was to investigate the technical implications of the stories in the backlog. The research team also determined which stories could be implemented in the next iteration based upon technical dependencies on other teams.

This survey was taken in two phases. During the first offering of the survey, only 12 team members responded. Management then encouraged participation, and all remaining team members took the survey. All responses are included in the paper.

C. Team 3

This team began its agile transition in early 2008. Similar to Team 2, this team was a large team that was subdivided into three sub-teams. The team primarily works in two locations in the same time zone but also have team members in England and India.

The product manager is active in story writing, story clarification, and acceptance test creation on a daily basis. This team also decided that only the acceptance testing would be done in the current iteration and most testing of features would occur in the iteration after the feature was implemented. Converse to the practice used by this team, most often agile teams test stories during the current iteration and leave each iteration with a potentially-shippable product [6].

The survey was completed by 20 of 31 team members.

D. Team 4

Team 4 has just begun its agile transition. Many on the team took agile training at the end of 2009. The product line originated as customized, service-based software developed for customers, and the developers were located around the US to be close to customers. Team 4 organized itself into three teams, each with a separate product in the same product family. Each team was able to create its own agile practice and shared experiences throughout the teams. The products also have to pass US Food and Drug Administration (FDA) audits of its process.

The survey was taken by 27 team members out of 35.

E. All Respondents

As discussed, the industry results reported in this paper are based upon input of 1,235 members of agile teams or teams desiring to be agile and are not representative of the industry at large. The respondents ranged from developer to tester to project manager to Chief Technology Officer. The respondents came from many regions in the world. The geographic distribution of responses is shown in Table 1. A respondent could indicate multiple geographic locations if answering for their larger organization so the total in Table 1 exceeds 100%.

Table 1: Geography of survey respondents.
(Multiple responses allowed)

Continent	% of respondents
Africa	0.4%
Antarctica	0.2%
Asia	32.0%
Australia	1.9%
Europe	35.0%
North America	62.0%
South America	1.5%

As shown in Table 2, the respondents had a range of experience with agile software development practices. The majority of the respondents were relatively new to agile software development. This phenomenon is both likely due to the encouragement of the authors to have the teams they are coaching and training to take the survey and also because agile software development is becoming increasingly popular with companies.

Table 2: Experience of respondents.

Experience	% of respondents
0-6 months	54%
7-12 months	14%
1 year	14%
Longer	18%

V. SURVEY RESULTS

In this section, we discuss survey results structured around the seven dimensions of CA. A comparative view of industry adoption of the dimensions in Figure 2. The number presented for each dimension is the mean score of the responses on the set of statements related to that dimension whereby for each question a response of True=5; More true than false=4; Neither true nor false=3; More false than true=2; and False=1.
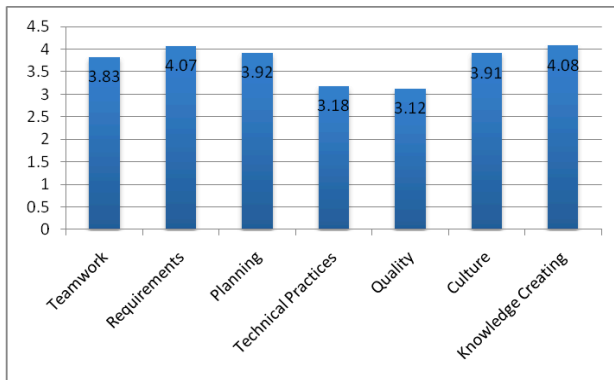
Figure 2: Industry average responses

In the following sub-sections, we will examine each of these dimensions in further depth. The results will reference the survey summary presented in the Appendix.

A. Teamwork

The Teamwork dimension includes the following five characteristics:

- *Team composition*: the extent to which the team operates under a small, focused "whole team" arrangement whereby the multidisciplinary team members all work together for a common goal.
- *Team management:* the extent to which a team is self-empowered and freed from non-value-adding tasks.
- *Focus:* assesses whether the team's priorities change within an iteration and whether the team feels they are all headed toward the same goal.
- *Communication:* the extent to which the team communicates face-to-face daily and has effective daily Scrum meetings.
- *Team member location:* provides a view to whether team members work on the same floor, same building, same campus, same city, or within the same time zone.

1) Industry Trends: The mean response for the Teamwork dimension was 3.83. Some of the particularly strong elements of teamwork among respondents were testers and programmers work on the same team, people were on two or fewer teams, and teams are kept together as long as possible. These teams feel they are all headed toward the same goal. Additionally, these teams have daily Scrum meetings that are most often less than fifteen minutes and are effective at synchronizing work. Finally, more often than not, teams work with substantially overlapping hours, although they less frequently work in close proximity to each other.

Some weaker trends among participants are that teams do not determine who is on or off the team. Additionally, teams deal with non-value-adding tasks and changing management priorities. Lastly, few respondents indicated that they worked in a shared physical environment, such as would be done in a "war room" or team room.

2) Four Teams: As a whole, the four teams are particularly below the industry mean, particularly in the team management characteristic. The teams feel significantly less empowered and freed from non-value-

added tasks relative to others who took the survey. Another trend for all five teams was a continuing emphasis on written documentation and signoffs. This company utilized a gate model for project management that perpetuated the need for documentation and signoffs despite their transition to agile software development. At each gate, the continuation of the development process is decided by the management team based upon a review of pre-determined artifacts. Additionally, the process for releasing their product had not been "agilized" so all releases still ended with at least four months of release tasks. Finally, many of the teams were geographically distributed.

Not all teams shared the same difficulties in transitioning to agile practices. In particular, Teams 2 and 4 had team composition challenges whereby the team members were not working on small, focused teams. Team 2 also had priority changes mid-iteration and did not feel that the whole team was headed toward the same goal. Conversely, Team 3 was ahead of the industry averages relative to the whole team heading toward the same goal.

B. Requirements

The Requirements dimension includes the following four characteristics:

- *Communication focus:* the extent to which the requirements are discussed and clarified in "just-in-time" discussions during an iteration.
- *Level of detail:* the requirements are able to start at a high level with little detail; details are negotiated during the iteration.
- *Emergence:* requirements can change based upon changing needs and development recommendations.
- *Technical design:* the technical design for a product is created collaboratively and occurs iteratively throughout a project.

1) Industry Trends: The mean response for the Requirements dimension was 4.07, among the highest for all dimensions in the survey indicating an overall "more true than false" response to the 14 requirements-related statements. The respondents strongly agreed that written requirements had to be augmented with discussion. Additionally, product owners willingly acknowledge that features can turn out to be bigger than anticipated.

Conversely, the survey results indicate that detailed requirements documents and technical design can sometimes still be written, and product owners can be criticized for requirements changes.

2) Four Teams: To a significant extent, the teams' results mirrored the industry results. Team 2 who has started its agile transition only six months prior to this data collection seems to still value detailed requirements and technical design. Conversely, Teams 1 and 3 has above average results in all requirements characteristics.

C. Planning

The Planning dimension includes the following five characteristics:

- *Planning levels*: assesses whether there is a release plan that is updated throughout the project and a task plan for each iteration.

- *Critical variables:* the extent to which scope is traded off for schedule and that the product owner does the prioritization.
- *Progress tracking:* the extent to which release burndown and iteration burndown charts are used and tracked.
- *Sources of dates and estimates:* the estimates are developed by the people who will do the work.
- *When do we plan:* planning is done prior and throughout the product cycle.

1) Industry Trends: The mean response for the Planning dimension was 3.92. The respondents strongly agreed that the people who are actually doing the work should make estimates and plans. These plans are then broken down into task plans for each iteration. Product owners negotiate tradeoffs between scope and schedule. Upfront planning is helpful without being excessive.

Teams were least likely to create and track to a release burndown chart. A release burndown chart shows the amount of committed features for the release on the y-axis and the number of iterations remaining on the x-axis. Teams were more likely to track to an iteration burndown chart that shows the number of hours of work remaining in the release on the y-axis and the number of days left in the iteration on the x-axis.

2) Four Teams: Many of the industrial trends were similar with the teams. However, the teams were less likely to create an initial plan prior to the first iteration of the release that shows an incremental release of features throughout the release cycle. Team 2 had additional difficulties trading off scope for schedule with the team getting prioritization of their work based upon input from the product managers. Additionally, this team had trouble spreading work evenly and having confidence they could meet their commitments. Team 3 continued the trend of having above average results in all planning characteristics. In particular, Team 3's team members felt especially positive about their participation in the estimation process.

D. Technical Practices

The Technical Practices dimension includes the following six characteristics:
- *Test-driven development:* assesses the extent to which the team writes unit tests before implementation code [1].
- *Pair programming:* the extent to which the team utilizes the pair programming practice whereby two team members work side-by-side at one computer, collaborating on the same work artifact [15].
- *Refactoring:* the extent to which explicit effort is spent to improve the design of code without increasing the functionality [7].
- *Continuous integration:* the extent to which new code is integrated into the team code base at least once per day.
- *Coding standards*: the extent to which the team has a set of rules for writing source code.
- *Collective code ownership*: the extent to which anyone on the team can change any code in the code base.

1) Industry Trends: The mean response for the Technical Practices dimension was 3.18, among the lowest of all dimensions. Other surveys [3, 13, 14] indicate that many agile teams follow the Scrum methodology [12]. Scrum prescribes project management practices but not technical practices. Our results substantiate these prior surveys because the mean for the technical practices dimension was lower than most other dimensions.

The technical practices most adopted by teams, in decreasing order of frequency, are continuous integration, coding standard, collective code ownership, refactoring, TDD, and pair programming. The first four of these are generally considered best practices among both agile and plan-driven teams. The last two, TDD and pair programming, are often associated with agile software development though these practices could be used by plan-driven teams.

2) Four Teams: Similar to many of the respondents, the teams had not generally embraced the technical practices. In particular, none of the teams practice collective code ownership or pair programming. However Teams 2, 3, and 4 did utilize the TDD practice.

E. Quality

The Quality dimension includes the following three characteristics:
- *Automated testing:* assesses the extent to which the team automates their unit tests and runs the tests frequently.
- *Customer acceptance testing:* the extent to which the team has acceptance tests written in conjunction with the product manager and these acceptance test are automated and run each day.
- *Timing:* assesses how early in the development process testing occurs.

1) Industrial Trends: The mean response for the Quality dimension was 3.12, the lowest of all the dimensions. Overall, the quality dimension assesses the team's attention to quality throughout all iterations from the start of the iteration and involving the whole team.

Automated unit testing was not practiced by many teams. Acceptance tests were automated even less frequently than unit tests. Acceptance testing is a formal process that is conducted to determine whether or not a system satisfies a set of criteria (i.e. "acceptance criteria") that are pre-determined by the customer to enable the customer to determine whether or not to accept the system [8]. On agile teams, the acceptance tests are to be written in conjunction with the customer or product owner. Survey results indicated that a significant amount of testing is still done manually. Finally, survey results indicate that performance, integration, and scalability testing occurs at the end of the development process. Many respondents indicated that a feature was not considered done until its acceptance tests pass.

2) Four Teams: Two main trends occurred with all the teams. First, the teams have an above-average practice of obtaining acceptance test criteria from the product owner. Team 2 had a high adoption rate of automated unit and acceptance testing. Additionally, the teams had a greater propensity toward testing later in the process. In

particular, teams are more likely to test completed features in the following or later iterations.

F. Culture

The Culture dimension includes the following six characteristics:

- *Management style*: the extent to which the team feels pressure to meet deadlines without having their autonomy taken away or unreasonable pressured.
- *Responses to stress:* the extent to which the team responds to pressure by re-prioritizing or re-scoping versus by working overtime or adding people.
- *Customer involvement*: assesses the access the team has to its product owner.
- *Title and salary alignment:* assesses the extent to which the team has incentive and a culture to work together as a team rather than as individuals.
- *Infrastructure:* the extent to which the team has technology and a work environment to support agility and synchronous communication between team member.
- *People:* assesses the skill level, accessibility, and agile attitude of the team members.

1) Industry Trends: The mean response for the Culture dimension was 3.91. Most questions had responses around the mean score indicating an overall response of "more true than false." The items that were higher than the average concerned the product owner and the project manager. Teams were satisfied with their access to these team members and with the job they were doing, even though often the product owner was not co-located with the team.

One item that rated lower was bonuses, annual reviews, and compensation promoting team behavior.

2) Four Teams: The teams' responses followed the overall industry trends just discussed though overall more positive. However, Team 2 had concerns related to response to stress and management style. These results indicate the team is experiencing considerable stress and pressure to meet their deadlines.

G. Knowledge Creating

The Knowledge Creating dimension includes the following three characteristics:

- *Reflection:* the extent to which the team conducts iteration reviews and retrospectives.
- *Timeboxes*: the extent to which the team works in short iterations culminating in working software.
- *Team learning:* assesses the culture within the team to question and learn from each other.

1) Industry Trends: The mean response for the Knowledge Creating dimension was 4.08, the highest of all the dimensions. The results indicate that many teams conduct end-of-iteration reviews and retrospectives. Almost all indicate these iterations are no more than 30 days long. The respondents indicated a high propensity toward questioning and learning from each other; valuing new approaches, technologies, skills and practices; sharing a set of common principles; and discussing problems.

2) Four Teams: The teams indicated a higher than average active participation in post-iteration reviews. All teams except Team 3 scored lower than average on having

potentially-deliverable software ready at the end of each iteration and iterations not being mini-waterfalls. Indications are that Team 2 has knowledge-creating challenges relative to learning and gaining from retrospectives and each other.

VI. SUMMARY

Using the results of Comparative Agility assessment tool, this paper provided a view into the "state of agility" in the industry based upon 1,235 responses. Industry trends indicate the highest adoption of agile practices occur in the areas of embracing emergent requirements and creating knowledge throughout the iteration and release. The lowest industry adoption occurs relative to utilizing technical practices and focusing on quality throughout all iterations. The short-term focus of iterations coupled with a lack of prescribed engineering practices in the popular Scrum methodology may lead to trouble. "Flaccid Scrum[4]" refer to teams that utilize only Scrum's project management practices. Progress eventually slows for Flaccid Scrum teams, according to Fowler, because the team has not paid enough attention to the quality of the code produced during each iteration. In many cases, only the easiest scenario of a feature is demonstrated at the end of the iteration. The feature can then be considered to be "done", and focus turns to a new set of features for the next iteration.

The four teams utilized their CA results to structure process improvement. Teams 1 and 3 heightened its awareness on the need to automate tests. Team 2 made several changes. First, they began preparing earlier for the upcoming iteration. To further aid in this preparation, they also decided to disband their dedicated research team and to put these members into each of the sub-teams to prepare for future iterations. All Team 2 sub-teams began to have one joint Sprint Review meeting so that all could become aware of the overall progress toward meeting the release goals. Team 3 reviewed its stature relative to industry trends and its peer teams and was proud that they were generally ahead in their agile adoption. Team 4 determined that it needed more training and coaching in several areas. In particular, they determined that team members sometimes had differing views on some agile practices and more training could help form a more united view.

ACKNOWLEDGMENT

REFERENCES

[1]   K. Beck, *Test Driven Development -- by Example*. Boston: Addison Wesley, 2003.
[2]   K. Beck, *Extreme Programming Explained: Embrace Change*, Second ed. Reading, MA: Addison-Wesley, 2005.
[3]   A. Begel and N. Nagappan, "Usage and Perceptions of Agile Software Development in an Industrial Context: An Exploratory Study," in *Empirical Software Engineering and Measurement Conference (ESEM)*, Madrid, Spain, 2007, pp. 255-264.

---

[4] http://www.martinfowler.com/bliki/FlaccidScrum.html

[4]   B. Boehm and R. Turner, "Using Risk to Balance Agile and Plan-Driven Methods," *IEEE Computer,* vol. 36, pp. 57-66, June 2003.

[5]   M. Cohn, "Determining How Agile You Are Comparatively," January 12, 2010, http://www.agilejournal.com/articles/columns/column-articles/2588.

[6]   M. Cohn, "Suceeding with Agile: Software Development Using Scrum," Massachusetts: Addisson Wesley, 2010.

[7]   M. Fowler, K. Beck, J. Brant, W. Opdyke, and D. Roberts, *Refactoring: Improving the Design of Existing Code.* Reading, Massachusetts: Addison Wesley, 1999.

[8]   IEEE, "IEEE Standard 610.12-1990, IEEE Standard Glossary of Software Engineering Terminology," 1990.

[9]   W. Krebs, "Turning the Knobs: A Coaching Pattern for XP Through Agile Metrics," in *Extreme Programming/Agile Universe*, Chicago, IL, 2002.

[10]   L. Layman, L. Williams, and L. Cunningham, "Exploring Extreme Programming in Context: An Industrial Case Study," in *Agile Development Conference*, Salt Lake City, UT, 2004, pp. 32-41.

[11]   L. Layman, L. Williams, D. Damian, and H. Buresc, "Essential Communication Practices for Extreme Programming in a Global Software Development Team " *Information and Software Technology (IST),* vol. 48, pp. 781-794, 2005.

[12]   K. Schwaber and M. Beedle, *Agile Software Development with SCRUM.* Upper Saddle River, NJ: Prentice-Hall, 2002.
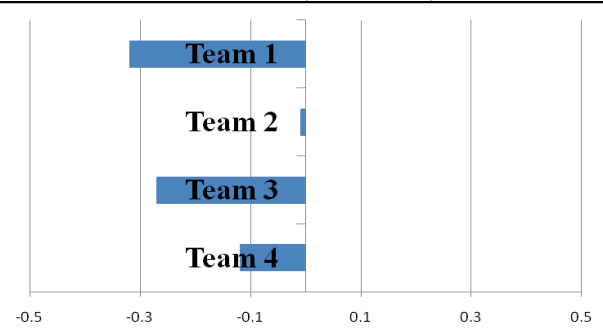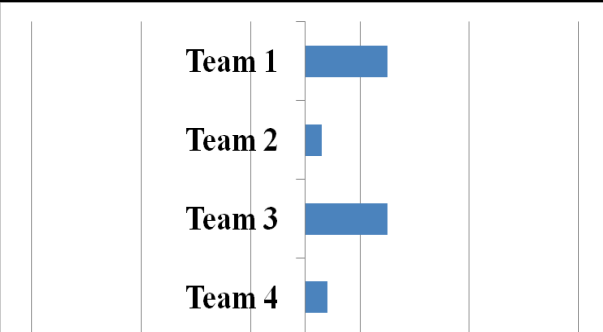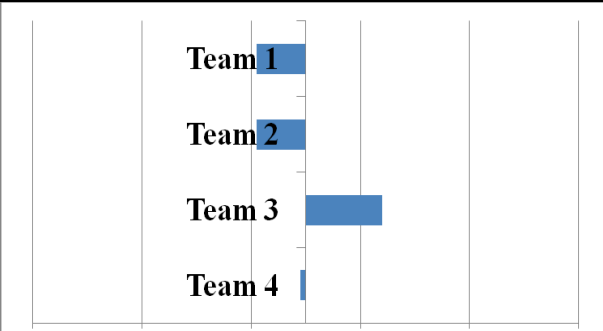
[13]   Version One, "Second Annual Survey 2007 The State of Agile Development," 2007, http://www.versionone.com/pdf/StateOfAgileDevelopmet2_FullDataReport.pdf.
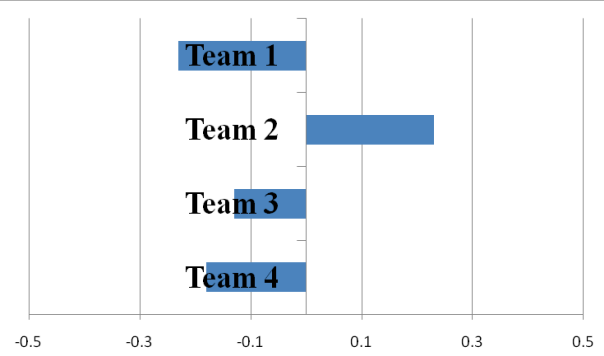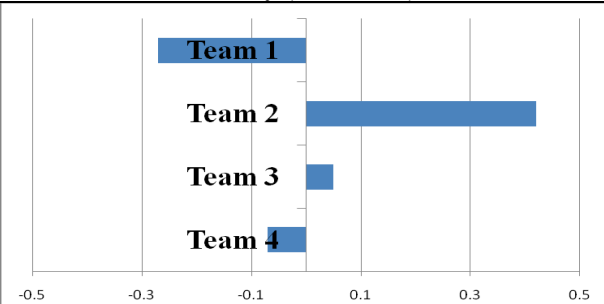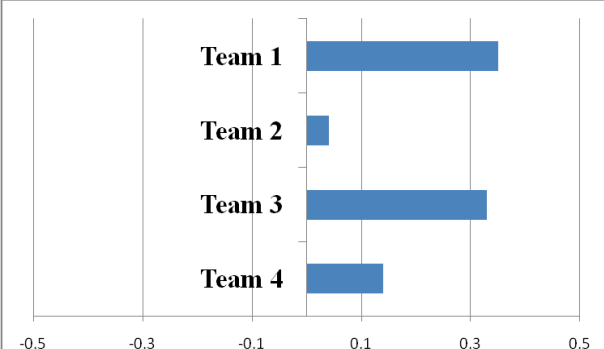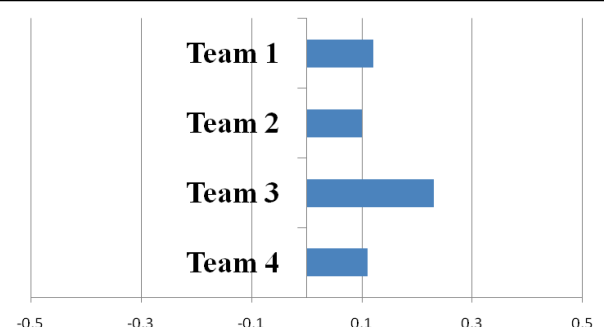
[14]   Version One, "Third Annual Survey 2008 The State of Agile Development," 2008 http://www.versionone.com/pdf/3rdAnnualStateOfAgile_FullDataReport.pdf.

[15]   L. Williams and R. Kessler, *Pair Programming Illuminated.* Reading, Massachusetts: Addison Wesley, 2003.

[16]   L. Williams, L. Layman, and W. Krebs, "Extreme Programming Evaluation Framework for Object-Oriented Languages -- Version 1.4," North Carolina State University, Raleigh, NC Computer Science TR-2004-18   http://www.csc.ncsu.edu/research/tech/reports.php, 2004.

Appendix: Survey summary.

Characteristics of teams below industry average	Dimension (mean score) In the graph, the standard deviations from the average is shown for each team.	Characteristics of teams above industry average
Directed; individuals work in silos; multiple locations; multiple projects	Teamwork (mean 3.83)	Self-organizing; cross-functional teams; dedicated team members; collocated
Document-centric; collected upfront; little acknowledgement of emergence	Requirements (mean 4.07)	Collected at different levels of detail; progressively refined; conversation-focused, augmented with documentation
All-encompassing; task-oriented plans created upfront; reluctance to update plans; little buy-in to dates from teams	Planning (mean 3.92)	Created at multiple levels of detail; frequently updated; created by team with full buy-in

Characteristics of teams below industry average	Dimension (mean score) In the graph, the standard deviations from the average is shown for each team.	Characteristics of teams above industry average
Code written by programmers working alone; little emphasis on testing; code becomes harder to maintain over time; infrequent integration and system builds	**Technical Practices (mean 3.18)** Team 1 Team 2 Team 3 Team 4 -0.5　-0.3　-0.1　0.1　0.3　0.5	Code written in pairs using test-driven development; code not allowed to
Quality is tested in after development; little emphasis on or effective use of automation	**Quality (mean 3.12)** Team 1 Team 2 Team 3 Team 4 -0.5　-0.3　-0.1　0.1　0.3　0.5	Quality is built into the product during each iteration; automated unit and acceptance tests
Satisfied with status quo; meets deadlines through heroic effort; command-and-control	**Culture (mean 3.91)** Team 1 Team 2 Team 3 Team 4 -0.5　-0.3　-0.1　0.1　0.3　0.5	Trusting; collaborative, and adaptive
Infrequent or ineffective reflection and team interaction; inconsistent use of iterations	**Knowledge Creating (mean 4.08)** Team 1 Team 2 Team 3 Team 4 -0.5　-0.3　-0.1　0.1　0.3　0.5	All work performed in strictly adhered-to iterations; frequent reflections; focus on team learning