The Agile Maturity Map
A Goal Oriented Approach to Agile Improvement

Jay Packlick
Sabre Airline Solutions
Southlake, TX
jay.packlick@sabre.com

Abstract
Fulfilling the potential of Agile development depends on people truly wanting to change how they work. Transitioning to new methods of development is particularly daunting in a large organization that already has a sizeable existing code base and a broad set of existing customers. In our experience, development teams tend to reach a plateau implementing a subset of Agile practices, and those tend to be generally focused on the planning and iterative aspects of Agile. This paper describes our experience in overcoming the barriers to increasing the adoption of Agile practices. Drawing from Agile principles, we have developed an approach for improvement that enables teams to accelerate change, deepen their degree of understanding, and increase their success in execution. We have found this helps teams to overcome the plateau effect.

1. Introduction
There’s an old joke: “How many psychologists does it take to change a light bulb? Just one, but the bulb has to really want to change!” As an Agile coach, I have often felt like the psychologist coaxing the bulb to change. Most development organizations are naturally resistant to change. I call this the ‘Tyranny of Success’ since teams that have successfully delivered software to satisfied customers often find it difficult to see the value in changing what already works. Additionally, learning new methods takes time away from the focus of most teams - getting the next release delivered on time and addressing customer support issues. Nevertheless, the goals of improving product quality, increasing productivity, and reducing ‘Technical Debt’ [1] demand that teams work in new ways.

Within Sabre Airline Solutions (A.S.) we have ample evidence that Agile practices contribute significantly to attaining these goals. Teams that more actively apply such practices do demonstrate higher levels of productivity and quality. In our desire to more broadly leverage this success, we have sought to expand and improve Agile adoption using ‘collaborate-and-enroll’ approaches vs. ‘command and control’.

2. Background
A.S. is a classic candidate for Agile development. Our organization has historically been very lean with many small teams working closely with customers to develop products for the Airline Industry. In 2001 our VP of development, Brad Jensen, chose to expand the use of Extreme Programming (XP). This expansion went beyond a series of successful pilot projects, and into the rest of the development organization. Subsequently, we engaged an outside vendor to train and coach our teams in XP practices.

Gradually, a trend resembling a bell-curve of XP practice adoption began to emerge among the teams. Some teams embraced most of the practices fully while others were more resistant to adopting them.

We found that for the majority of teams it was relatively easy to implement the ‘scrum-like’ subset of XP practices. Nearly all teams transitioned fairly quickly to developing software in iterations, using User Stories, working from backlogs, and integrating continuously. Daily Stand Up meetings were almost universal in adoption.

Unfortunately, many teams tended to plateau at various levels of adoption. Some practices, such as pair-programming, automated acceptance tests, Test Driven Design, on-site customer, and refactoring were more difficult to implement broadly. In general, teams tended to improve for four months to a year and then slowly level out. Teams working on newer development projects tended to embrace the most practices while larger legacy code C++ projects achieved lower levels of adoption. However, even with the teams adopting a smaller subset of practices, we realized significant improvements in productivity and quality.

2.1. Retrospectives were not enough

AGILE 2007
0-7695-2872-4/07 $25.00 © 2007 IEEE
In Agile development regular retrospectives are the primary mechanism for incremental improvement. Unfortunately, we have seen only moderate success achieving significant improvements via team retrospectives alone. Moreover, we have observed that reflective learning is practiced least frequently on teams most in need of improvement. We concluded that reflections alone were not satisfactory for driving the degree of change we were looking for.

### 2.2. Practice-Centric Self Assessments

Toward the end of 2001, a number of months after most of our teams had completed training in XP, we developed a questionnaire with a ranking system for each practice to provide leaders and teams with a tool to gauge their performance against our XP practices. Some higher performing teams used the feedback to focus their improvement efforts while most teams tended to remain relatively static in their post transition levels of adoption. We tried a similar approach with some modifications a year later with comparable results.

We learned that team members often reacted defensively to the practice-centric approach - it seemed the teams were failing to see the underlying value relative to the effort to implement them. Implementing automated unit tests, for example, was a considerable challenge for systems with large C++ code bases. Ultimately, the practice oriented survey approach yielded only minimal improvements.

### 3. Introducing a context for change

In 2005 the context changed; dramatically improving quality became a high priority organization wide goal within A.S. Naturally quality had always been an imperative however, meeting customer expectations had been elevated to a corporate mandate. The ‘Status Quo’ was no longer acceptable. We had embraced a context for driving change and everyone understood that change was necessary. Applying the first principle of quality improvement described by W.E. Deming we had established the basis for ‘constancy of purpose’ [2] in improvement.

Shortly thereafter, we conducted a series of team level ‘Quality Workshops’ to brainstorm ways to improve quality. The scope was broad and inclusive of all aspects of marketing, development, and delivery. All team members participated. We applied Lean development concepts practiced in workshops conducted by Mary and Tom Poppendieck [3] – including Seeing Waste and Value Chain Mapping.

From these workshops quality improvement goals and action plans were developed. Their focus was on achieving specific goals, not on implementing any given practices.

The improvement projects were driven from the bottom-up, with support from senior level management. We strongly encouraged teams to take chances and try new things. Some teams adopted more Agile practices to achieve their goals. After six months we began to see many improvements including significant reductions in defect rates.

From this we learned a number of important lessons: 1) we were able to increase enthusiasm and participation in the improvement projects by encouraging teams to develop their own priorities, and goals, and also their own methods to accomplish them. 2) Team members displayed a great deal of pride in developing their own solutions. 3) Teams that allocated time during each iteration to work the improvement projects achieved the best results. Teams that tried to ‘fit the improvements in’ without allocating time experienced much more limited improvement. 4) Teams that tried to take on more than 2 or 3 goals at a time were less successful.

### 4. Using Agile to increase Agile Adoption

Based on the success of our quality workshops, we wondered if a goal-based approach would better motivate teams to increase their application of Agile principles. Developers are enthusiastic problem solvers by nature. Perhaps, if we established a set of goals based on XP practices, our teams would find their own way to achieve them in an Agile manner. To us, this seemed to be more consistent with the Agile Manifesto principle of focusing on ‘Individuals and Interactions’ [4]. We just needed a good set of goals.

#### 4.1. Creating the Backlog

We enlisted our most experienced Agile practitioners to reexamine the standard XP practices in terms of their underlying goals. From this we would develop a backlog of ‘User Stories’ based on those goals. As an example we derived the goal “Validate the quality of code design and share learning” from the XP practice of ‘Pair Programming’. Of course this statement is an oversimplification of the benefits that can be realized from Pair Programming but, like User Stories, we knew that conversations would be necessary to clarify the goal [5]. Ultimately, we created a backlog of over thirty User Stories based on the set of Agile practices we determined were most in need of increased adoption by our teams.
We piloted implementing the backlog of Agile improvement ‘User Stories’ as an Agile development project with a subset of teams. We identified acceptance criteria for each of the User Stories. The backlog was prioritized and estimated. Team members broke the work down into tasks and signed up for them. Naturally, ‘development’ of the User Stories was performed iteratively. The initial results of this approach were very promising.

4.2. Making the Backlog Sticky

We reviewed the backlog of Agile User Stories with a broad set of developers and leaders to obtain feedback on content and priorities. The response was almost unanimously positive. The goal-based approach to implementing Agile practices resonated very well. We also received feedback that throwing a backlog of over thirty goals at a very busy team trying to get new releases out the door would simply cause their eyes to glaze over. It was too much to swallow at one time. We looked for a way to distill the set of goals down further, creating a more ‘sticky’ concept [6] that teams would easily recall and sustain a focus on; something more marketable.

Ultimately we categorized the User Stories into a set of five high level goals. With a bit of creative license we were able to organize these goals into an easily recalled and obviously self-serving acronym: AGILE. We summarized the five high level AGILE goals as follows:

- **A** = *Acceptance Criteria* – Validation of functional correctness is critical. One of the areas contributing significantly to quality improvement is increasing the fidelity and bandwidth of information communicated to the developers just in time for them to develop within an iteration.
- **G** = *Green-Bar Tests and Builds* - code validates code; development builds are automated. Broken tests and builds are promptly fixed.
- **I** = *Iterative Planning* – Plan continuously, keeping progress visible, engage the entire team, and apply reality based feedback to make informed decisions.
- **L** = *Learning and Adapting* – Real change resulting in quality improvement requires a focus on improving skills and learning from doing. This is the core goal most important to sustaining continuous improvement.
- **E** = *Engineering Excellence* – This was more difficult to state in terms of goals. There are an irreducible set of practices that are necessary to produce high quality software. These processes include refactoring, coding standards, and following S.O.L.I.D design principles [7]. Getting other eyes on code is also a high priority.

Of course there is a lot of detail behind these five high level goals. We articulated that detail as separate User Stories and acceptance criteria.

What makes this list Agile? Anybody well versed in Agile will immediately recognize that the four principles of the Agile Manifesto, Individuals and Interactions, Working Software, Customer Collaboration, and Responding to Change [4] are not explicitly stated in this list. Instead, these principles function as overriding objectives in every activity that our development teams engage in, providing a context for all of our improvement efforts. Also, the spirit of Agile is implicit in the acceptance criteria that we defined for each User Story.

Following is an example of an AGILE User Story from the ‘E’ (Engineering Excellence) goal area derived from the Paired Programming XP practice:

**Validate the quality of code design and share learning** - All new and changed production code is reviewed for correctness by someone other than the developer who coded it. Adherence to software design principles and team standards are considered. Improvements to the design are implemented.

4.3. Creating an AGILE road map

We felt that teams needed a simple way to determine their current state, prioritize the User Stories based on their individual circumstances, and track their progress. We also knew from experience that attempting to implement more than one or two improvement User Stories could overwhelm a team.

To address this, we distributed the User Stories and their acceptance criteria across a continuum of five different maturity levels, each representing one of the different stages of learning a team progresses through in each of the five AGILE goal areas. This was, in effect, a soft of road map suggesting a logical progression of goals for teams to achieve. This formed what we now call the AGILE Maturity Map (AMM).
The five maturity levels that we identified were:

1. **Awareness** – The team understands the goals, and understands the value of pursuing the goals and their acceptance criteria. Awareness of existing ‘better’ practices around the goals typically exists as well. The team may possibly implement basic activities to address the goal.

2. **Transformation** – Knowledge is put to practical use. Development practices to satisfy the goals are regularly applied. The team buys into the practices and both leaders and team members are demonstrating a consistent commitment toward achieving the goal. They are actively working toward putting the knowledge to practical use.

3. **Breakthrough** – The team now consistently uses Agile practices that satisfy the goals and acceptance criteria – even under high pressure situations. The team has found ways to work around their barriers to adoption.

4. **Optimizing** – Improvements are made on a continuing basis in the goal area, and lower level criteria are generally exceeded. A spirit of creative innovation in improving is evident.

5. **Mentoring** - High performing teams coach and mentor other teams in the goal area. This contributes to learning across the organization and beyond with a higher level goal of generally improving the software engineering field.

The Optimizing and Mentoring goals exist to ensure that improvement remains continuous – there is no end state.

Underlying most cells in this map is a set of User Stories and acceptance criteria that we identified to enable teams to transition to the next higher level. For example, under Awareness for the Iterative Planning goal we have these User Stories:

- Requirements are decomposed into units that can be completed in iterations of equal duration.
- The development plan is revised each iteration.

The User Stories under the Breakthrough level for Iterative Planning include the following:

- Release planning is collaborative and involves the extended, cross functional, team members.
- User Stories exhibit ‘INVEST’ properties (Independent, Negotiable, Valuable, Estimatable, Sized Properly, Testable) [8].
- All team members are able to quickly and visibly review actual progress vs. planned completion.

### 4.4. Awareness - Enrolling the team

It’s often said that quality is a management problem. Similarly, improving effectiveness in Agile requires leadership buy-in. One motivation behind the AMM was to shift accountability to improve execution from the Agile coaches to the leadership and the teams. By aligning on a set of goals, estimating the effort, and committing to achieve them; accountability for improvement shifted to the logical leaders within the organization to manage the change; the development managers and their leaders.

Taking a page from the AMM itself, we knew ‘awareness’ was the first step; we needed to educate and enroll the development team leaders and their teams in the value of this approach to improvement.

We test marketed the AMM concept with a few early adopters and received constructive feedback to improve the map. We then presented it to senior development team members. Although we expected some resistance, the map was extremely well received by managers and developers alike. The response was much better than any approach we had attempted previously. They viewed the goal-based approach versus the previous practice-based approaches as pragmatic and empowering.

We met with each development team to walk them through the map in order not to water-down the concept. We explained the purpose of the AMM in detail. Through this we obtained vital feedback to improve the map, the goals, and the acceptance criteria. Virtually every team agreed enthusiastically with the importance and value of the goals. We did, however, encounter one common concern from team members. They feared that management might not allocate resources and time to enable the teams to implement the AMM User Stories.
4.5. Transition – Implementing the Map

The next step, analogous to the ‘transition’ maturity level, was to put the Agile Maturity Map to practical use. Our initial request was for teams to review the acceptance criteria for each of the five goal areas, and determine as a team what level of maturity the team had attained in each goal area. The primary goal here was to provide feedback to teams as to their current level of performance. The value of the feedback was two fold: 1) To enable the team to decide how best to advance in maturity in specific areas, and 2) For the coaches to identify teams that were in breakthrough or higher levels. As a part of this, we hoped to leverage higher performing team skills to assist teams operating at lower levels of maturity.

We were initially concerned that teams would overstate their maturity levels. Instead, we observed a slight bias toward lower than deserved rankings. Still, on the whole, the self assessments were consistent with expectations. We found that teams took their role in achieving these goals very seriously.

In order to keep a focus on the goal based nature of the AMM, we viewed it as critical that there be no audit and no publication of the self assessment results. We also took great pains to clarify that the AMM was not a metrics tool. We stressed that the AMM not be used in this capacity. As an Agile coach I did collect, but didn’t share, the results of this exercise to obtain feedback on the level of coaching support needed.

Each team then chose one or two AMM User Stories to focus on at a time – based on the relative priority as viewed by the team. Teams then developed estimates, task breakdowns, and signed up for tasks to implement the User Stories. As a high level organizational goal, our VP of development urged teams to seek to advance one level of maturity in at least three goal areas before the end of the year. It is premature at this time to report the results of this objective however, the early results are promising.

Although we don’t publish the self assessments by team, we did discover some trends. After teams completed their self assessments and selected the initial AMM User Stories to focus on, we found that the two areas of greatest opportunity for improvement were around Acceptance Criteria and Engineering Excellence. Iterative planning was the goal area with the most teams in ‘breakthrough’ or higher maturity levels.

Within two months of rolling out the AMM, nearly every team had established their baseline and developed some form of plan to address one or two goal areas. Most were actively working on their improvement projects. Following is an example of one team’s task breakdown to implement the User Story derived from Pair Programming:

- Decide how to conduct collaborative peer reviews
- Decide what to review and not to review
- Decide how to handle remote peer reviews
- Create a way to ensure code reviews are done
- Create a way to assess review effectiveness
- Decide how to account for review times in the iteration plan
- Conduct retrospective of Peer Review progress

5. Breakthrough – AMM Results

One Acceptance Criteria for the Agile Maturity Map itself was for teams to realize sustained continuous improvement in the goal areas identified. Achieving this would fall into the ‘Breakthrough’ maturity level for the AMM. Have we achieved this?

Twenty teams have been using the AMM for over six months to focus their efforts to improve quality and increase their performance against the Agile goals. Virtually every team is actively working on two to three AMM User Stories, with ‘Green Bar Tests and Builds’ being the most popular area of focus. Interestingly, demand for coaching support, classes, and tools to facilitate some of the engineering practices are markedly up. Most teams are demonstrating marked improvements in the level of Agile adoption across the board.

- We are seeing remarkable improvements, 60% or greater, in productivity as measured by time spent on new work vs. defect correction on some teams that more fully embrace and achieve the goals.
- The practice of On-Site Customer has increased greatly; real end users are working more often with teams, and in house subject-matter experts are allocating more time to this activity.
- Developer satisfaction in the availability of good acceptance criteria is increasing.
- Teams at lower maturity levels are demonstrating an active interest in learning from teams at or above the ‘Breakthrough’ level.
- Teams that have had difficulties in automating their builds in the past have now found ways to overcome those issues.
- Teams are now conducting retrospectives more regularly and implementing their ideas.
- Code coverage metrics are increasing in teams that have chosen ‘Green Bar Test and Builds’ goals.
- Some teams seeking to fulfill their AMM goals have adopted practices they had previously
resisted, and are now embracing those practices without outside pressure.

- For high pressure projects, some teams found great success in ‘protecting’ improvement tasks from reprioritization and committing to complete them within normal iterations.
- Interest in the use of AMM has spread to other divisions of the Sabre organization where they are actively applying it.
- We have extended this map to develop a Testing Maturity Map, a general ‘Quality’ Maturity Map, and the delivery organization is considering a similar approach.

We observed a tendency for some teams to allow one team member to do most of the work against an improvement goal. This tends to result in much less effective adoption than when multiple members of the team sign up for and implement tasks for the AGILE User Story. It’s also interesting to note that a practice that one might assume to work universally will work well for one team yet work poorly for another. On one large team, a Scrum of Scrum (SOS) [9] approach was tried and worked well while on another large project it added unnecessary overhead without adding much value. By focusing on the AMM goals, our teams have a context for experimentation and invention.

We have observed a strong correlation between progress against AMM goals and the amount of time allocated in iteration plans to support the necessary tasks. Invariably, complaints that there is not enough time to work toward the goal equate to a statement of project priorities.

6. Conclusions

Helping teams overcome plateaus in realizing the full potential of Agile development is a difficult and critical leadership challenge. Teams are naturally resistant to change and lose focus on improvement in the face of urgent operational priorities. Our experience in using the AGILE Maturity Map is that improvements resulting from a goal driven approach for a large mature Agile organization are much more rapid and sustainable than by using more traditional practice-driven approaches. We’ve found that team members value and respect a process that works, and do so far more when they’ve had a hand in its development, rather than having it imposed upon them. Teams can overcome prejudices against specific practices by focusing on goals. As a result, team members understand much better the rationale behind the practice.

Do the teams using the AMM implement approaches that are different than experienced Agile coaches would prefer them to use? Yes! Are these less effective than some they might try? In some cases they are. However, our teams are encouraged to learn and experiment, and most are now actively doing so. With the Optimizing and Mentoring maturity levels before them, they have a motivation to continue to explore other options and improve.

Our version of the AMM is not a methodology, a metrics tool, or an improvement system. We view it as a mindset for thinking about Agile adoption in terms of goals rather than practices favoring people and interactions over process. We find this approach helps teams to ‘want to change’ and at Sabre Airline Solutions the light bulbs are indeed starting to change.

7. References