



Active Stakeholder Participation: An Agile Best Practice

Home Start Here Best Practices Disciplines Artifacts Resources Contact Us #AgileModeling

 Search

AM's practice of Active Stakeholder Participation is an expansion of eXtreme Programming (XP)'s On-Site Customer that describes the need to have on-site access to people, typically users or their representatives, who have the authority and ability to provide information pertaining to the system being built and to make pertinent and timely decisions regarding the requirements, and **prioritization** thereof. While this level of participation is required to make your software development efforts effective it often isn't sufficient in many organizations, particularly those where politics and not reason are the order of the day. Project success often requires a greater level of involvement by project stakeholders: domain and technical experts should be actively involved with modeling (yes, creating the actual models, not just giving information to a modeler), senior management needs to publicly and private support your project, operations and support staff must actively work with your project team towards making your production environment ready to accept your system, other system teams must work with yours to support integration efforts, and maintenance developers must work to become adept at the technologies and techniques used by your system. The focus of this article is to explore how to successfully promote active stakeholder participations on your agile delivery teams.

Table of Contents

1. Who are stakeholders?
2. Why is stakeholder participation important?
3. Strategies for working together
4. Gaining access to stakeholders
5. Factors affecting stakeholder participation
6. What are project teams actually doing?
7. Rights and responsibilities

1. Who Are Stakeholders?

My definition of a project stakeholder is anyone who is a direct user, indirect user, manager of users, senior manager, operations staff member, the "gold owner" who funds the project, support (help desk) staff member, auditors, your program/portfolio manager, developers working on other systems that integrate or interact with the one under development, or maintenance professionals potentially affected by the development and/or deployment of a software project.

From this definition you can see that business people, such as direct users and their managers, aren't the only stakeholders of a project. As you know there is a wide range of people potentially affected by a new system, therefore to succeed you must understand and then synthesize their requirements into a cohesive vision. This is one of the things that makes software development hard: each project stakeholder will have their own requirements, their own vision, and their own priorities. But it also makes software development fun.

In this definition I have chosen to exclude the developers who are working on the project. This may seem strange at first because developers clearly have an important stake in the projects that they work on. Yes, **developers are definitely project stakeholders**. Why do I continue to distinguish between developers and project stakeholders? Because I want convenient terms to distinguish them, I really don't like "developer stakeholder" and "non-developer stakeholder", and because they have different roles to play on a project.

My definition of project stakeholder and developer may be different than yours, or perhaps you prefer different terms. For example, **XP** discusses the concepts of customer and programmer, not project stakeholder and developer, and has slightly different definitions because they use the terms differently than AM does. **Scrum**, on the other hand, talks about agile team members (instead of developers) and **product owners** (instead of customers or more accurately stakeholder representatives). The most sophisticated definition of stakeholders that I've seen within the agile community comes from **Outside-In Software Development** because it explicitly indicates that there is a wide range of stakeholders and even organizes them into four categories: Principals (the people who buy your software), end-users (the people who interact with it), partners (the people who will make your software work in production), and insiders (people within your organization that impact how your team works).



2. Why is Stakeholder Participation Important?

People are not very good at defining, particularly in detail, what they want. However, people are fairly good at indicating what they think they want and then when an option is presented to them what they like and don't like about it. In other words, we need to work with our stakeholders to identify what they think they want, produce something which reflects that understanding, get feedback from our stakeholders, and then update our solution to reflect our improved understanding. The implication is we need to work in a more evolutionary and collaborative manner if we're to provide solutions which reflect our stakeholders actual needs, and to do that we must work closely and regularly with stakeholders.

Traditional approaches to software development which are based on defining a detailed requirements specification early in the project, referred to as "**big requirements up front (BRUF)**" strategies, prove to be very risky in practice. Traditional project teams, even the "successful" ones, typically produce less than ideal results when they strive to produce a solution which reflects the specification. Yes, the team may produce something to specification but it likely won't be what the stakeholders actually want, but instead what they thought they needed at some point in the past. I believe that the goal of a disciplined agile delivery project team should be to provide their stakeholders with a solution which fulfills their current understanding of the intent of their stakeholders as effectively as possible given the constraints of the situation, not build something to specification.



3. Strategies for Working Together

It is clear that in order to be successful all project stakeholders must actively work with your team to achieve these goals. There are several implications of this practice:

1. **Timely decisions.** Stakeholders must be prepared to share business knowledge with the team and to make both pertinent and timely decisions regarding project scope and requirement priorities.
2. **Inclusive modeling.** You need to adopt **inclusive modeling techniques**, which are based on **user-centered design (UCD)** and **participatory design** principles, which stakeholders can easily learn and adopt.
3. **Management requires IT skill and knowledge.** For senior managers to effectively support your project they must first understand the technologies and techniques that your team is using, understand why your team is using them, and understand the implications of using them. With this knowledge their efforts within your organization's political arena are far more likely to be effective at the right times in the right ways. Senior managers won't be able to gain this requisite knowledge simply by reading a weekly project status report or by attending a monthly project steering meeting. Instead, they need to invest the necessary time to learn about the things that they manage, they need to actively participate in the development of your system.
4. **Production staff are involved from the start.** Your **operations and support organization** must invest the resources required to understand both your system and the technologies that it uses. Your support staff must take the time to learn the nuances of your system, the implication being that they need to work with your system as it is developed and/or your team will need to provide them with training. Furthermore your operations staff must become proficient with both the installation and operation of your system. You may choose to include one or two operations engineers on your development team or once again to invest project resources to train operations staff as required. Regardless of your approach, both your operations and support organizations will need to be actively involved with your project team.
5. **Take an enterprise view.** You need to work with other project teams if your system needs to integrate with other systems. For example, perhaps your system needs to access a **legacy database**, interact with an online system, work with a data file produced by an external system, or provide an XML data extract for other systems. Integration often proves difficult if not impossible without the active participation of these developers: imagine how difficult it would be to access the information contained in a large legacy database if the owners of that database refuse to provide any information about that database. Some **initial architecture envisioning** will help to drive this.
6. **Don't just hand-off to the maintenance team.** Maintenance developers need to work with you to learn your system. When the intention is to either partially or completely hand-off the maintenance of your system to other developers, it is common to bring in software professionals skilled in maintaining and enhancing existing systems to free up members of the original development team, then your team must work with these people so they can take over the system from you. Even when some original team members are still involved an effort must be made to transfer the knowledge to the new members of the team.

4. Strategies for Gaining Real Access to Stakeholders

I have never been on an IT project, agile or otherwise, where I couldn't get regular, direct access to key stakeholders or their representatives. I've worked on projects in the financial industry (including brokerage), retail, telecommunications, e-commerce, software product development, government (including military), and others. I have, however, worked on many projects where people provided excuses (none of which ever held any water) for why they couldn't get access to stakeholders. Common excuses included:

1. **Understand why stakeholder involvement is important.** First and foremost, you need to understand why active stakeholder involvement is important and what the impact to your project will be if you don't have it. In many organizations you will be required to justify why stakeholders need to be actively involved with an IT project, and the better you understand why it's important the better you'll be able to argue for it.
2. **Get support from the stakeholders themselves.** Some of your stakeholders would love to be actively involved with your project, and many more could likely be easily convinced that they should be involved.
3. **Educate senior management.** In many organizations senior management will need to determine, and then support, the level of involvement of stakeholders. To make this decision they need to understand the alternatives and the trade-offs of each.
4. **Be flexible.** Although you would like to be co-located with your stakeholders and have instant access to them, it doesn't always happen that way. You may only get to interact with them for a few hours a day, or only for a few hours once a week. More access is generally better, but as you can see below there are several reasons why it can be difficult to gain access to them on a continual basis.
5. **Be prepared to work with stakeholder representatives.** Regardless of the inane rhetoric you may have heard, you will almost always end up working with people representing the wider stakeholder community (this is certainly true of **product owners**).
6. **Fight for it.** Active stakeholder participation is crucial to the success of your project because without it you won't know what your stakeholders actually need.
7. **Keep fighting for it.** Throughout the project there will be pressure to divert stakeholders away from the project so that they can focus on their normal "day jobs". You may need to keep up a regular communication stream thanking the stakeholders for their participation (always a polite thing to do regardless) and reminding them of the positive impact that they're having on the effort.

Gaining access to stakeholders, let alone gaining their active participation can be hard for several reasons:

1. **We've trained stakeholders to work in "hands off" manners.** For several decades we've told stakeholders to follow a serial/traditional approach to IT projects -- we'll do some initial requirements gathering, we often ask them to sign off on the requirements and project plan, we'll give them periodic updates of how the project is going, and then months or sometimes years later ask them to be involved with acceptance testing. In other words, have short, defined periods of involvement with longer periods of virtually no involvement in between. With modern approaches, such as agile, we're now asking them to be involved continuously throughout the project.
2. **Many stakeholders don't trust us.** The traditional approaches to solution delivery haven't worked out very well in practice from the point of view of stakeholders. According to the **2008 IT Project Success Survey** traditional strategies have not been very effective at delivering functionality which stakeholders actively want, something backed up by the Standish Group's Chaos report (see the **BRUF** article for details).
3. **It isn't part of their job.** Many stakeholders believe that they don't need to be actively involved with IT projects -- IT work is the job of the IT department after all. This can be a difficult cultural issue to overcome.
4. **They believe it's too hard.** Many traditionalists will use complex diagrams which use notations that stakeholders aren't used to seeing nor all that interested in learning. Agilists prefer to use **inclusive tools**, such as **plain old whiteboards (POWs)** and paper, and sketches to model with stakeholders.
5. **They believe their normal "day jobs" are more important.** From the point of view of the stakeholder this is fair, but from the point of view of your organization it likely isn't. Whenever I run into the situation where I'm told the stakeholders are too busy or their time is too valuable I question the logic of this. If that's true, then what you're being told is that the value to the organization

of their day jobs is greater than that of the IT project project. If that's the actual case then the resources being invested in your project should really be diverted to helping the stakeholders do their day job.

6. **They really aren't available all the time.** The normal activities of your organization still need to occur in parallel to your IT project. Some stakeholders may not be available during normal business hours. For example, I've worked in brokerage firms where we weren't able to interact with traders during trading hours, but could do so during other times. Some stakeholders may only be available part time, perhaps only an hour or two a day at certain parts of the day.

5. Factors Affecting Stakeholder Participation

There are several factors which will impact the nature and quality of how stakeholders participate with a solution delivery team. These factors are summarized in [Table 1](#).

Table 1. The factors impacting stakeholder participation on IT projects.

Factor	Range	Potential Impact(s)
Participation style	Reactive <=> Proactive	Stakeholders who proactively participate with IT project may have a political agenda which they are trying to further. Stakeholders who are reactive to requests for information may slow the project because the team must wait for responses (minimally, the product owner will need to guess at the answer, increasing the chance of potential rework in the future) Reactive stakeholders may be a sign that the stakeholder community has a poor relationship with the IT department
Relationship	Negative <=> Positive	When the relationship between IT and stakeholders is negative the stakeholders will likely participate less frequently and to a lesser extent
Communication channels	Formal <=> Informal	Formal communication, such as written documentation in a specific format, can increase the bureaucratic overhead on the team, increase cost to the project, and increase the time it takes to deliver the solution. Communication will increase in formality in regulatory compliance situations. Informal communication strategies, such as face-to-face communication and sketching, lowers overall complexity and cost and often improves time to market.
Availability	Irregular <=> Continuous	When stakeholders aren't regularly involved with a project team the chance that the team will build the wrong thing increases. With continuous stakeholder participation the feedback cycle is reduced, improving overall chances of project success.
Interaction	Facilitated <=> Active	When interaction with stakeholders needs to be facilitated (someone needs to act as a go between to help the development team and stakeholders to communicate) you run the risk of miscommunication and increasing the team's time to delivery (the facilitator becomes a potential bottleneck).
Location	Co-located <=> Global	When the team is co-located with stakeholders it is much easier for them to interact regularly and actively with the development team. As the team becomes more geographically distributed, the chances of project success decrease (see the 2008 IT Project Success Survey for some figures).

6. What are teams actually doing?

Everything that I've described up to this point is all fine and dandy, but what are agile teams actually doing in practice? Luckily several of my [IT Surveys](#) provides insight. [Figure 1](#) summarizes results from one of the questions in the [2013 How Agile Are You? survey](#) which lists several strategies for working with stakeholders and the adoption rate of those strategies within teams claiming to be agile. As you can see, there are several indications that agile teams are working closely with their stakeholders:

- Over 60% have regular discussions with stakeholders
- Half invest time to identify who their stakeholders are and what their goals are
- 60% take usability issues into account (something you need to work closely with stakeholders to do)
- Over half are actively trying to improve the business process, also something requiring active stakeholder participation

Figure 1. Potential strategies for producing value to stakeholders.

What strategies does your team follow to provide value to your stakeholders? Please select all that apply (if any).

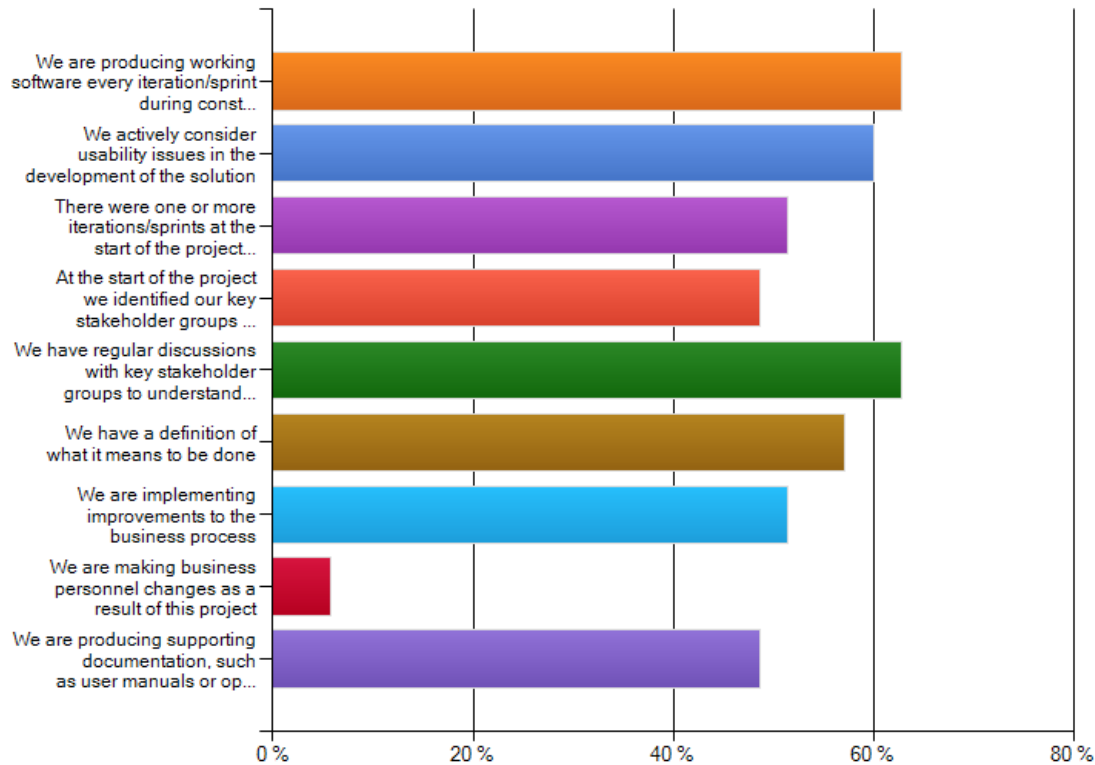
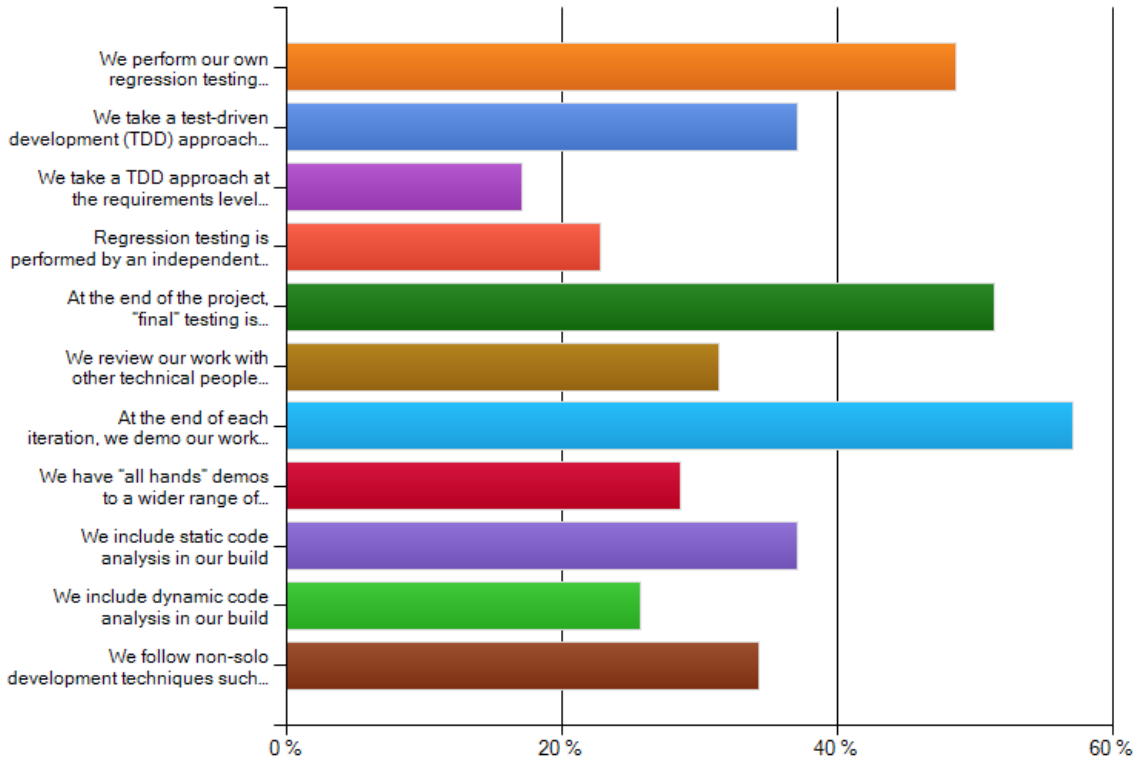


Figure 2 also depicts results from the 2013 *How Agile Are You?* survey, in this case exploring how agile teams are validating their work. Iteration demos, where the team shows the work they did that iteration to key stakeholders, tops the list at almost 60%. Acceptance test-driven development (TDD), also an activity requiring active stakeholder participation, comes in at 18% and "all-hands demos" to a wider group of stakeholders at 28%.

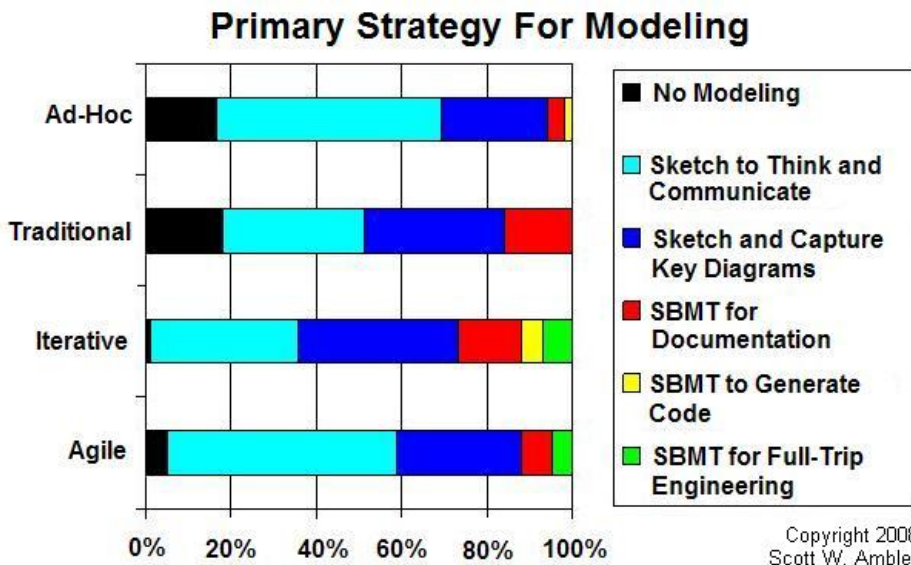
Figure 2. Agile criterion: Validation.

What strategies does your team follow to validate their work? Please select all that apply (if any).



When it comes to modeling, the 2013 Project Initiation survey found that over 90% did some sort of up-front requirements modeling, clearly something requiring stakeholder involvement. Results from the 2008 Modeling and Documentation survey are shown in Figure 3, revealing that the use of inclusive modeling tools such as whiteboards and paper is quite common regardless of development paradigm.

Figure 3. Primary modeling strategy.



7. Rights and Responsibilities

I used to talk about the rights and responsibilities of project stakeholders, a concept that I adopted from Karl Wiegers excellent book entitled *Software Requirements*. Unfortunately, I recently discovered that these rights and responsibilities as written in *Agile Modeling* weren't as clear as I had originally thought and were potentially even divisive. So, after a fair bit of discussion on the AM mailing list I've decided to reword the rights and responsibilities from the point of view of everyone on the project, not just the stakeholders.

The rights of everyone:

1. To be treated with respect.
2. To produce and receive quality work at all times based on agreed to project standards and principles.
3. To estimate the activities you are actively involved with, and to have those estimates respected by others.
4. To be provided adequate resources (time, money, and so on) to do the job that's been asked of you.
5. To determine how your resources will be invested. For people funding the project how the funds will be spent and for people working on the project (e.g. investing time) what tasks they choose to work on.
6. To be given the opportunity to gain the knowledge pertinent to making the project a success. Business people will likely need to learn about the underlying technologies/techniques and technical staff to learn about the business.
7. To have decisions made in a timely manner.
8. To be provided good-faith information in a timely manner. Sometimes this is just the "best guess" at the time, and that's perfectly all right. This includes but is not limited to business information such as prioritized requirements and detailed domain concepts as well as technical information such as designs and detailed technical concepts.
9. To own your organization's software processes, following and actively improving these processes when needed.

The responsibilities of everyone:

1. To produce a system that best meets your needs within the resources that you are willing to invest in it.
2. To be willing to work with others, particularly those outside your chosen specialties.
3. To share all information, including "work in progress".
4. To actively expand your knowledge and skillset.

Share with friends: [Tweet](#) [LinkedIn](#) [Facebook](#) [StumbleUpon](#) [Digg](#) [Baidu](#) [Google +](#)

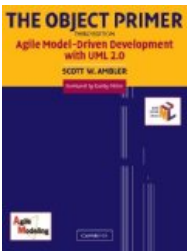
Let Us Help

We actively work with clients around the world to improve their information technology (IT) practices, typically in the role of mentor/coach, team lead, or trainer. A full description of what we do, and how to contact us, can be found at [Scott Ambler + Associates](#).

Recommended Reading



This book, [Disciplined Agile Delivery: A Practitioner's Guide to Agile Software Delivery in the Enterprise](#) describes the [Disciplined Agile Delivery \(DAD\)](#) process decision framework. The DAD framework is a people-first, learning-oriented hybrid agile approach to IT solution delivery. It has a risk-value delivery lifecycle, is goal-driven, is enterprise aware, and provides the foundation for [scaling agile](#). This book is particularly important for anyone who wants to understand how agile works from end-to-end within an enterprise setting. Data professionals will find it interesting because it shows how agile modeling and agile database techniques fit into the overall solution delivery process. Enterprise professionals will find it interesting because it explicitly promotes the idea that disciplined agile teams should be enterprise aware and therefore work closely with enterprise teams. Existing agile developers will find it interesting because it shows how to extend Scrum-based and Kanban-based strategies to provide a coherent, end-to-end streamlined delivery process.



[The Object Primer 3rd Edition: Agile Model Driven Development with UML 2](#) is an important reference book for agile modelers, describing how to develop 35 types of agile models including all 13 UML 2 diagrams. Furthermore, this book describes the fundamental programming and testing techniques for successful agile solution delivery. The book also shows how to move from your agile models to source code, how to succeed at implementation techniques such as [refactoring](#) and [test-driven development \(TDD\)](#). The Object Primer also includes a chapter overviewing the critical database development techniques ([database refactoring](#), [object/relational mapping](#), [legacy analysis](#), and database access coding) from my award-winning [Agile Database Techniques](#) book.

