

The Complete Guide to Building a Successful Agile Team

John Blanco

 **targetprocess**
publishing

Introduction	5
Overview of Roles	8
Product Owner ^{PO}	10
Product Manager ^{PM}	20
Scrum Master SM	22
Release Train Engineer ^{RTE}	33
Agile Development Team ^{ADT}	37
Closing	46
About Targetprocess	47

Foreword

There's a good chance that no one on your team is able to lift a minivan over their head (or anything else that is 20 times their own body weight for that matter), and you probably don't get together at happy hour and call yourselves a "superorganism." But there are some similarities between your Agile team and a colony of ants.

For one thing, everyone has a role to play.

Ant colonies are called superorganisms because the individual ants work so hard and are so unified in purpose, whether they are the queen, workers, soldiers, or drones, that they behave like a single organism.



The strength of an Agile team is built on the interplay of team members roles as well. When starting a new project, or making the transition to Agile in the first place, defining those roles is a fundamental part of the process.

*Written by
Max Steinmetz*

John Blanco — Enterprise Agile Transformation Leader

John Blanco is a Technology management professional with a proven track record in aligning IT strategies to business objectives and delivering innovative solutions across business disciplines. As an accomplished strategist in portfolio management, systems design and implementation, product development, and business communication, John utilized his skills across numerous industries and *Fortune 500 companies*. These include *NBCUniversal, Scholastic Inc., Direct Brands, Merrill Lynch, iCentral Corp, Alliance Consulting, OpenMetrik Inc., Cablevision, Alyn Hospital, ORC, EMI, and ADP Brokerage Services.*

As a champion of “people and process”, John advocates an Agile approach and methodology for advancing business-IT alliances and partnerships. As a veteran systems designer and applications developer, John developed portfolio management metrics strategies and created software to manage warehouse logistics and tracking of broadband devices, HR management tools for collecting and reporting domestic compensation data, as well as interfaces and databases to manage charity contributions for a world-renowned children’s hospital. As a tactical project manager, John also implemented ERP Financial systems, managed warehouse consolidations, credit derivative trading systems, and e-reader / digital publishing applications.

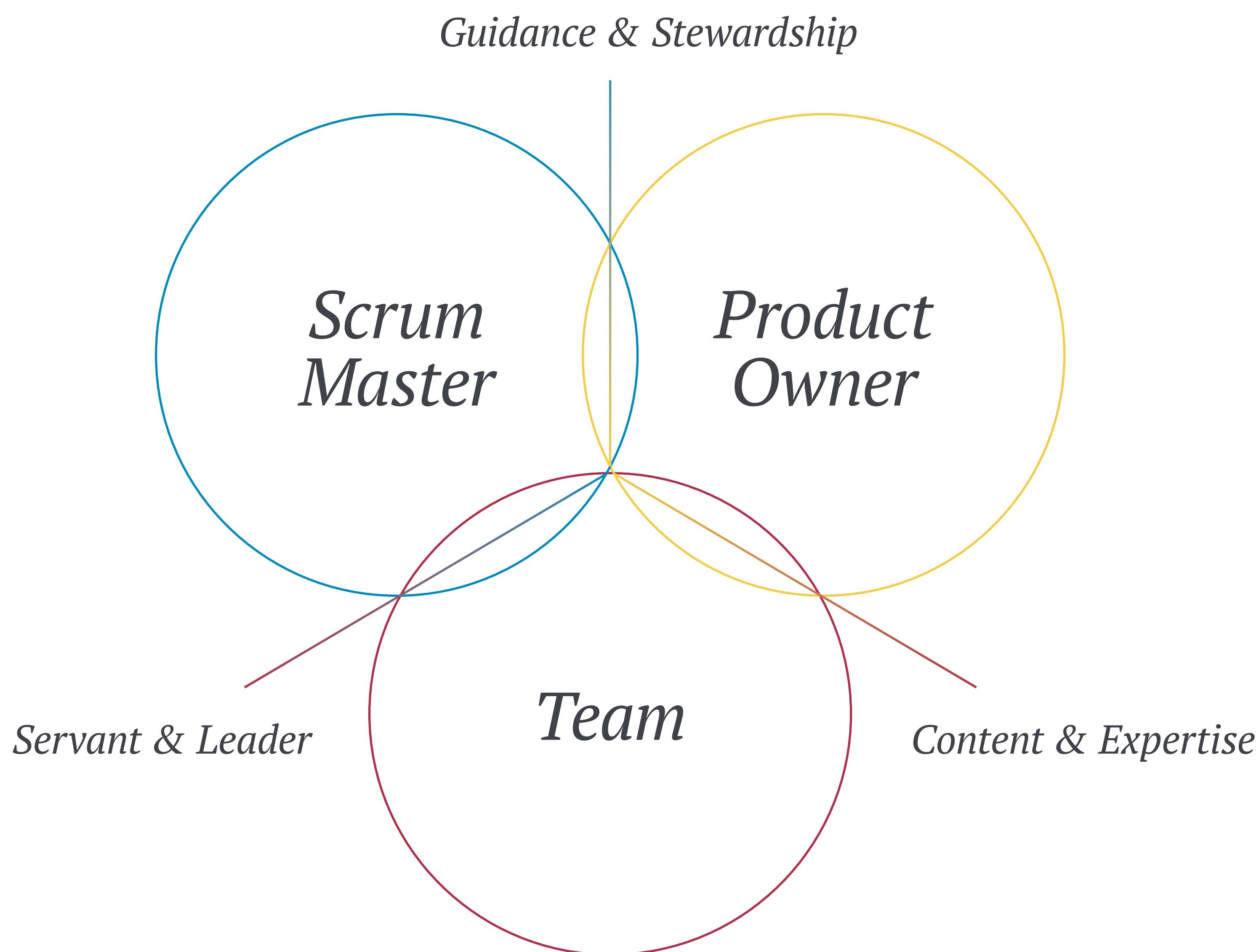
Introduction

When I begin discussions with a new organization or technology group undergoing an Agile transformation, my first step is to understand their present state.

Learn what makes them tick, motivates them, and holds them back.



A successful Agile transition comes down to understanding the nuances and taking an inventory of the data points in four quadrants: *Productivity, Predictability, Responsiveness, and Quality*. These areas of focus must be accounted for by all Agile transformation teams and their fearless leaders.



With that in mind, and for the purpose of starting with a shared foundation, I lead the teams I work with through a formal exploration of the basic Agile roles. And because we live in a world that demands that Agile scale, I convey the importance of each role — carefully articulating the purpose and unique characteristics of each.

Roles in Agile, particularly when applying techniques like Scrum, are critical for establishing, driving, and sustaining the four key fundamentals stated in the [Agile Manifesto](#), along with their underlying [Principles](#). Moreover, you must understand and carry out the responsibilities of each role to ensure the success of your Agile transformation and guarantee that you are establishing an Agile culture. This means placing People at the center.

Henry Ford  once said:

//

Coming together is a beginning

Keeping together is progress

Working together is success

//

I keep this little nugget of wisdom handy because it helps maintain the mantra of teamwork echoed in Bruce Tuckman's project principles of Forming, Storming, Normalizing, and Performing. We do not need to delve into those stages of team maturation here, but let's acknowledge that all successful Agile projects start with the team at the center. This means the identification and clarity of roles within Agile are important for understanding how a team should perform holistically and at all levels of delivery.

Over the course of this guide, we will review each of the key roles in an Agile team and discuss how each plays a critical part in moving the team, and organization, towards success.

Overview of Roles

Unlike its Waterfall or traditional project process counterparts, an Agile team is collaborative and cross-functional. Teams are made up of a variety of roles ranging from developers to quality assurance engineers to systems architects.

Agile teams also stand out as self-organizing -- they receive challenges from team leaders and, within the boundaries and requirements provided, develop a solution that fulfills the need of the organization. Don't assume that self-organizing means a lack of leadership. In fact, it's quite the opposite.

According to Philip Anderson in *The Biology of Business*, "Self-organization does not mean that workers instead of managers engineer an organization design. It does not mean letting people do whatever they want to do. It means that management commits to guiding the evolution of behaviors that emerge from the interaction of independent agents instead of specifying in advance what effective behavior is."

Despite the autonomy of Agile teams, there are *clear roles* for Agile team leadership.

Overview of Roles

For example, in Scrum, three specific roles make up a team:

Product Owner ^{PO}

Product Manager ^{PM}

Scrum Master SM

Release Train Engineer ^{RTE}

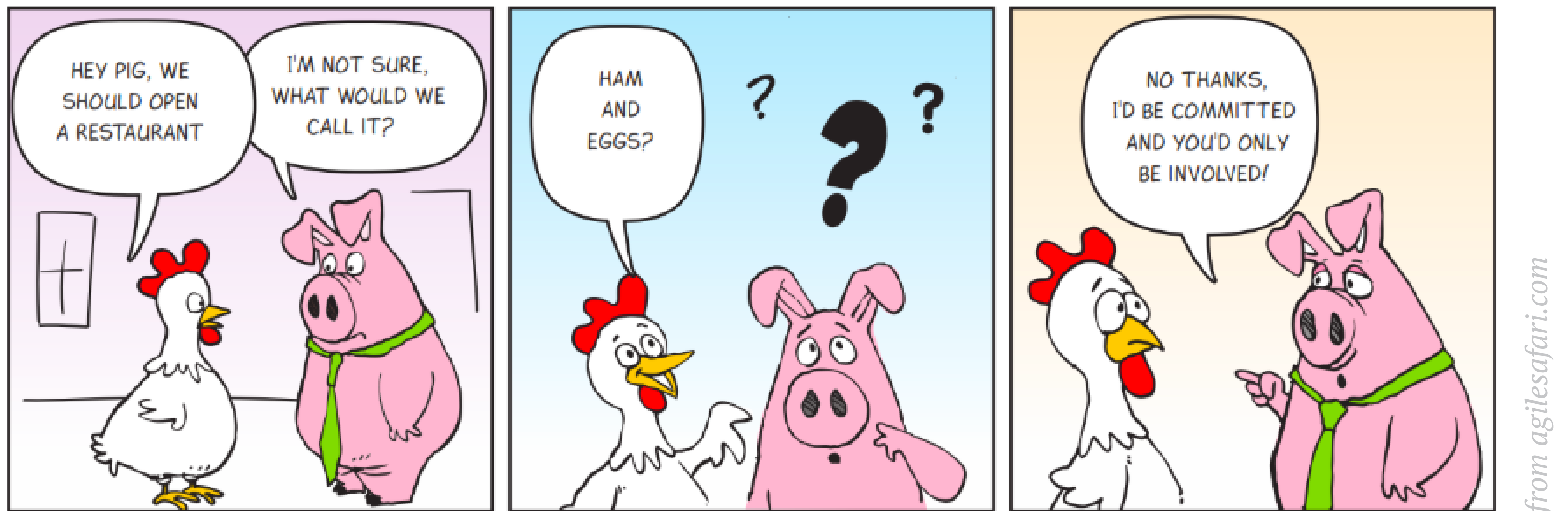
Agile Development Team ^{ADT}

Product Owner ^{PO}

Let's start with what makes a strong and successful product owner in an organization.

In simple terms, the product owner is the “voice of the customer” — or better — our key stakeholder. We rely on such an individual to help the business crystalize its vision and then knowingly, and confidently, convey that vision to the team. As a result, our success relies on the effective engagement of a product owner.

A team lacking someone in this role, or worse, someone ineffectively performing this role, can spell disaster for any development project.




There are hundreds of definitions for the product owner, but for any to be true to the Agile philosophy, they largely share the concepts conveyed above and continue to build on them. Let's focus on the three phrases I've underlined.

Key stakeholder: It's no secret that unlike the old way of running projects, where we leaned towards a more client-vendor relationship, the product owner's (PO) participation on a Scrum team ensures that the business has "skin in the game". In purer Agile implementations, the product owner comes out of the business organization and is assigned to the team to act as the "voice" of the customer. Some of you may remember the old Agile cartoon of the Chicken and the Pig.

Product Owner

Although many see this as a light-hearted jab to express the intent of the scrum master / product owner / team dynamic, it helps explain the principles that empower us. It also (comically) dictates that the requestor (client or “chicken”) can no longer operate outside of our development paradigm, and that the expectations of every team are to have a business representative playing a key role in a product’s development.

At many mid-to large-sized companies, the business may not have the capacity to assign individual product owners to each team. In those situations, we employ the idea of a proxy business product owner, often from the ranks of the technology camp. Regardless of where they originate, the role is the same. The product owner remains responsible for the business content of the product development backlog; and even in cases where architectural features and stories are written, the PO remains the key owner. They must understand the purpose and intent of every requirement committed to by the team.

 **Vision:** One of the key responsibilities of a worthy product owner is to holistically drive the vision of the product’s development. This is the only way that the team can respect the PO and trust that they have a careful eye on not only what the business is asking for at the project’s start, but also the changes that are bound to arise during the product’s evolution.

A good PO understands the industries and markets that their business operates in, and when needed, can be relied upon to voice the details of the requirements and any changes to vision or direction that may occur.

What we avoid, is a PO who stands on the sidelines and says:

//

*I'll just act as my boss' pointer
finger and tell her when you're doing
something wrong*

//

Convey (Communicate): A product owner cannot be effective unless he or she is a good communicator. Conveying and translating business concepts is a key ingredient needed for authoring effective and functional user stories, not to mention, prioritizing the important aspects of all product backlog items (PBIs). This awareness enables the PO to act as a collaborative agent in planning and coordinating all product releases. Of course, communicating a point is one thing, but having a true skill in bartering and negotiating with the team, the scrum master, and even other stakeholders is what makes a good PO a hero!

Agile product owners should follow these *5 basic tips*. It helps the entire team get a better handle on improving how they operate in a true Agile spirit:

1. Have a Solid Concept of the “Big and Small” of Things. In other words, always maintain an awareness of the “big picture.” Backlogs have a tendency to become a dumping ground for ideas. Many may be virtuous ideas, but the team still needs to stay focused on the vision that is driving the product. When teams lose track of that they start letting milestones slip and features become “almost done”.

2. As the Primary User Story Author – Write them Clear and Crisp!

A PO must know the why and who of every user story. This perspective enables teams to develop product features closer to what you want rather than guessing on how the functionality might have to integrate with other stories and features, or who will be the end user.

3. Embrace Change The PO is called many things – some I will not repeat here – but one thing he or she clearly is, is a change agent on the team. Know your markets and do careful research, share when changes are on the horizon, and how an upcoming change might affect the direction of the product. The PO helps the team stay sane when change happens. And it will definitely happen.

4. Split when Splitting Makes Sense Of all the skills a PO can have, this is the one that makes a great PO stand out. Great POs take a big idea and find early high-value, high-learning opportunities. They take a massive thing and break it apart into a core subset of essential user stories that are releasable and marketable sooner. This often means taking a big user story and splitting it into several smaller stories. And if needed, when stories appear to be blocked, they take a small story and help the team build it into even smaller, valuable slices.

5. Know How and When to Say No. You are the product owner because your business leaders trust you to make the right decisions about the direction of your product. Accept the responsibility and be deliberate about what elicits a “yes” or “no”. Great POs have the skill of saying “no” without coming across as arrogant or pig-headed. Here are a few ways to say “no” when someone wants to add something to your backlog.

No

⚠ Delete this message

Wow that's a great idea. I'd love to add it. But... shouldn't we pull something out to make room for it?

Product Owner → Ways to say “no” when someone wants to add to your backlog:

Is that really a priority for us right now? We've been focused on [current focus]. Maybe we should consider this for the next release (or sprint).

Whoa...that actually sounds like it's outside our current vision. Are we really changing business direction?

OK, but help me understand why you want this. Maybe there's another solution.

You know after thinking about it - Here's another thing you can do for this product that might help...

Hey, do you remember this? Here's a similar story from the backlog...

Maybe we can include part of this in the next release (or program increment).

As a customer
I can use a credit
card to pay for
my order

As a customer
I want to know
if I didn't provide
a proper expiration
date

As a customer
I want to know
if I didn't provide
a properly formatted
CC number

As a customer
I want to know
if I didn't provide
a card holder's
name

As a user I should be able to cancel the hotel booking as I would not be traveling

As a privileged user I should be able to cancel the hotel booking without cancelation charges

As a normal user I should be able to cancel the hotel booking without cancelation charges 2 days before

As a normal user I should be able to cancel the hotel booking without cancelation charges 3 days before

The product owner is normally the only role that can accept completed stories. That said, in many environments you may have special technical stories that are accepted by architects and/or QA managers. Nevertheless, the PO should maintain control of the stories backlog: business, technical, and/or QA driven. It only makes sense that if a technical story is introduced into the backlog that the PO keep a firm understanding of the why and reason behind the effort.

I've outlined the fundamental role of the Product Owner first for a reason. Where the PO tends to operate at the team level, in scaled Agile environments (like SAFe) we may also have a product manager who operates at the program level and maintains a keen eye on the big product picture.



Definitions:

Functional Requirements: Define a technical function of the system by specifying behavior between outputs and inputs.

Non-Functional Requirements: Define system attributes such as security, reliability, performance, maintainability, scalability, and usability. They serve as constraints or restrictions on the design of the system across the different backlogs.

Features: A service that fulfills a stakeholder need. Each feature includes a benefit hypothesis and acceptance criteria and can be split as needed.

Enablers: Support the product development plan by accounting for exploratory work related to infrastructure, compliance, and architecture development.

Product Manager PM


The PM is a customer-facing role owning the vision and roadmap(s) of a program — usually at a large or enterprise company. The PM concentrates on defining the strategy through market research, maintaining a solid understanding of company objectives, and building a reasonable plan based on the current state of the product.

Product Manager

This is critical for any Agile Release Train (ART) — a self-organizing group of Agile teams (typically 5–12) that plan, commit, and complete work together. The product manager drives the program increment planning objectives through prioritized features. At the same time, the product manager works with the product owners on the Agile Release Train (ART) to establish a plan of prioritized work for the teams.

Let's move on to the person who teams up with our product owner, the flip side of the Agile management coin and counterpart of this dynamic duo... the scrum master.

Scrum Master SM

One of my favorite Agile quotes comes
from Jeff Sutherland  one of the inventors
of the Scrum

//

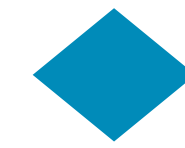
*That absolute alignment
of purpose and trust is
something that creates
greatness*

//

In many ways this is how a scrum master coaches and guides a scrum team.

It is best to recognize that the scrum master is the “glue” that holds a team together. In part, this means acting out the servant component of the servant-leader competency of this role. For example, a scrum master might help team members complete their work by removing impediments that are potential blocks for progress.

The bulk of the SM’s time is spent communicating with team members, helping the team self-organize, remaining autonomous, and providing the right level of governance in the application of Agile practices. They also assist in the nonstop improvement of team performance and conflict management, not to mention working closely with the product owner to drive progress and improve productivity.



A Servant-Leader is someone who inspires action in others by sharing power, putting the needs of others first and helping people develop and perform as highly as possible.

- Agile Experience

Coaching / Mentoring

The Servant / Leader

The “Process Guru”
- Communications
- Motivation

Mobilization

Team-Building

Personal Optimization

A “Success Driver”

Process
Leadership
Skills

People
Management
Skills

The Glue
Scrum Master

Attitude
& Sensitivity

Organizational
Skills

- Roadmap Vision
- Meeting Facilitation
- Retrospective Follow-Through
- Documentation
- The “Block Un-Blocker”

- Respect and Trust
- Self-Starter
- Success-Driven
- Sees the “Aura”
and “Knows the Buttons”

Facilitation

Collaboration

Engagement

Communication

Collaboration

Engagement

Facilitation

A successful SM must exhibit four main qualities:

Be a Process Guru — Leverage Agile best practices and personal experience to lead and coach the team in the Agile process.

Motivate their People — Be an inspirational team builder. Act as the team's rudder on the "boat towards success."

Unblock the Blocks — Follow through on retrospective outcomes, facilitate meetings, intervene when necessary, and run interference when distractions appear. Don't allow disruptions to distract their team. Be the "Block Un-Blocker".

Keep an Eye on the Big Picture — Be a proactive and enthusiastic team leader who has an overarching sense of what matters most. Promote self-governance and team empowerment to earn and maintain the team's respect and trust.

Let's break down these attributes in real life. Consider the following *common activities* for a scrum master:



Facilitate Meetings

- Assist the product owner in preparation for grooming sessions, iteration reviews, iteration planning sessions, etc.
- Be an effective moderator by keeping meetings focused and productive. Assist with logic when emotions go into overdrive.
- Develop their Post-Processing skills to notate and monitor next steps and follow-throughs by the team.
- Facilitate effective retrospectives, planning and closing sessions

Manage Team Dynamics

- Be a confident Agile coach. Keep the team on course and identify and enforce rigidity of rules.
- Be the team psychiatrist and mediate conflicts. In a land of egos and personal agendas, smart, hard-working, and passionate people sometimes lose their tempers, are stubborn, and whine. Learn to cope and help them cope too.
- Help the team make decisions by asking probing questions. Don't answer for them, but assist in finding a solution.
- Foster self-organization principles.
- Maintain open channels of communication. Help reconcile or clarify objectives with the team and the product owner.

Always Be Learning

- Continue personal learning regarding Agile (visit user groups, attend conferences, read books, etc.).
- Consult team members regarding Agile best practices.
- Provide constructive feedback to the team.
- Exchange ideas with other scrum masters (join or create a community of practice).

Participate in Product Planning

- Coach the product owner and team to write better user stories and to split as needed.
- Encourage sound user story writing techniques (the 3 C's, INVEST, etc.).
- Help product owners write or adapt product visions.
- Assist prioritization of product backlog items. Drive practices that effectively prioritize features and stories (WSJF – weighted shortest job first).
- Help with release planning.
- Stay familiar with the team's work (i.e. understand what the product is).

Keep the Big Picture in Mind

- Ensure continuous collaboration.
- Regularly touch base with stakeholders.
- Apply metrics for team reporting.
- Help further Agile adoption within the organization.
- Share insights with the company (publish tip-sheets, blogs, etc.).
- Be the key contact for the team, external teams, and stakeholders.
- Provide internal learning opportunities (conduct briefings or workshops).
- Be an Agile change agent.
- Coach the team in Agile fundamentals, (roles, rituals, acceptance criteria, technical debt, etc.).

Lead by Example

- Reflect Agile and Scrum values.
- Remind the team of their commitments, arrangements, and contracts.
- Seek continuous improvement.
- Act as a communication channel. Share issues and observations with the team.
- Ask open questions to get the team to respond and help make actionable decisions.
- Help the team stay focused by acting as a buffer from external distractions.
- Encourage the team and product owner to implement a suitable definition of done and a definition of ready.

Scrum Master

With a clear understanding of the scrum master's role on a single team, let's expand on how we scale it within larger organizations. Similar to the Product Manager role, which scales the product owner, in SAFe we have the release train engineer (RTE) that scales our scrum master role.



Release Train Engineer RTE

In an environment where Agile Release Trains exist, an additional role is created for the release train engineer. This “chief” scrum master is responsible for facilitating the Agile Release Train processes, program execution, escalating impediments, managing risk, and driving program-level improvements. Additionally, RTE’s are responsible for facilitating program events such as release planning, inspect and adapt, and the weekly scrum of scrums.



RTE responsibilities include:

- ✓ Establish the annual calendars for sprints, program increments, and all other Agile and SAFe artifacts
- ✓ Facilitate program increment (PI) planning events
- ✓ Guide PI planning readiness — vision, backlogs and facilities
- ✓ Facilitate PO sync and scrum of scrums
- ✓ Help manage program risks and dependencies
- ✓ Maintain proper communication with all the technical and non-technical teams involved
- ✓ Participate in enterprise program management improvement and standardization activities
- ✓ Provide input on resourcing to address critical bottlenecks
- ✓ Report status to senior management and other leadership
- ✓ Partner with other scrum masters to coordinate PI execution — this includes tracking team progress and publishing accurate team metrics and progress report.

Robert Greenleaf  Father
of the Servant-Leadership Model:

//

*Good leaders must first
become good servants*

//

Most often, RTEs come out of the program management office (PMO) or Agile management office (AMO) of a company. Although this seems like a natural progression for an existing program or project manager (or scrum master), the person needs to go through a meaningful mindset transition. The servant-leadership model encourages a philosophy that implies a comprehensive view of the quality of people, work, and communication

SAFe further illustrates the required skills and mindset shift with the “from” → “to” qualities:

- *From* coordinating team activities and contributions *to* coaching the teams to collaborate
- *From* deadlines *to* objectives
- *From* driving toward specific outcomes *to* being invested in the program’s overall performance
- *From* knowing the answer *to* asking the teams for the answer
- *From* directing *to* letting the teams self-organize and hit their stride
- *From* fixing problems *to* helping others fix them

As you can see, the RTE plays an incredibly important role in helping ART’s achieve their goals. A resilient and persevering champion — one with a driven spirit for getting the job done through teamwork and association rather than through ego, heroics, or hysterics!

Finally, let’s look at those people who actually do the work...
the Agile development team.

Agile Development Team ADT

The **team members (TM)** are the people responsible for ensuring completion of user stories. This cross-functional team consists of software engineers, architects, analysts, QA experts, testers, UI designers, etc. Typically, where the product owner is responsible for what is being built, the team is focused on how to build it.

TaskWorld, Characteristics of a Good Agile Team Member:

//

Having a great Agile team is integral to the success of any project. At the base of it all are the individual team members who work simultaneously to achieve a common task. A team is only as good as its members.

//

It's important that a scrum team have autonomy to determine how and when to complete its work. It's not unusual for teams to discover that it has more work to do than originally assumed during the first couple of days in a sprint. And it is in cases like this that ongoing communication between the team, product owner, and scrum master must remain intact.

We can summarize a team's responsibilities as →

The team is responsible for managing how the work is done.

No matter how the team is constructed and/or what skills are mixed or joined together — they are our development experts. It's easy for people with long years of technical experience, who are now in the scrum master or product owner roles, to think they should lead, dictate, or direct development. However, this must be avoided. In fact, it's considered an abhorrent practice that gets in the way of the naturally innovative team spirit. Advice, opinions, suggestions? All good, but never devalue the teams authority to define, build, and test their work.

The team estimates the size and complexity of the work.

Another bad habit is when the scrum master, product owner, or other external technical lead disclaims or re-sizes the team's estimation of the work.

Realistically, we will never achieve 100% precision in our estimates, but the estimate of the one doing the work is always more accurate than that of an outsider. An expert may size the work one way, but if the actual developer on the story (who may be more junior) sizes it another way — I'll always go with the actual developer's estimate.

As a project manager with experience in a technical domain on a Waterfall project, it was not uncommon for me to estimate tasks. Unfortunately, I would later wonder why the team wasn't meeting my commitments. There's no problem with questioning or probing a bit to refine estimates, but overall, the team's estimates must hold.

The team determines the technical design in their area of expertise within architectural guidelines.

The team is comprised of either local experts or people who have direct lines into the architectural leadership of the organization. Many teams may even have architectural representation embedded. Part of the Agile movement recognizes how inputs from the team must interact with the architectural standards or guidelines of the enterprise. This is a healthy dynamic that provides recommendations to improve the product's architectural runway. This dynamic must be enabled.

The team commits to the work they can accomplish in an iteration.

At the conclusion of every sprint planning session, the team commits to the work. This happens after the team reviews its estimates, creates and assigns its tasks, and sets a sprint load agreement with the product owner. The scrum master or product owner do not commit on the team's behalf.

The commitment has two parts as it acts as a contract between the team and the business. The parts are:

Team members will do all in their power to fully deliver upon their commitments.

If issues or blocks arise that jeopardize those commitments, the business will do all in its power to work with the team towards an agreed-upon solution. This may mean helping the team re-scope or re-prioritize the work to maintain forward progress and continuous value delivery.

The team is responsible for value and continuously seeks to improve the quality of its deliverables.

A self-organized team clearly understands that its success is not derived by solely completing tasks according to development standards. Rather, the team knows that stories must be fully developed, tested, and accepted by the product owner to be release-ready. Customers want working features in production, not half-baked stories sitting in the backlog. Therefore, discovering ways to improve the development pipeline is integral to our success and improves customer perception.

The team is continuously committed to finding ways to improve.


It is the obligation of every Agile team to constantly reflect on what is or is not working for the purpose of driving total quality improvements. For this reason, we should constantly promote the value of effective retrospective facilitation. The truth is that few improvement suggestions come from above or outside the team. For real improvement look within.

Captain Kirk, Star Trek III — The Search for Spock

*Because
the needs
of the one...
outweigh
the needs
of the many.*

Closing

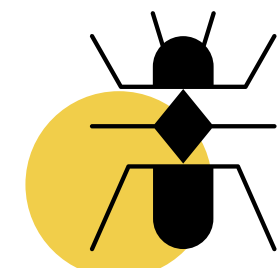
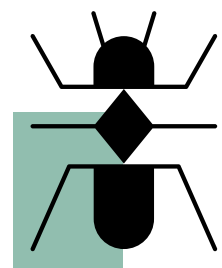
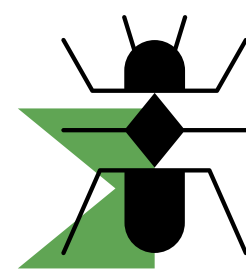
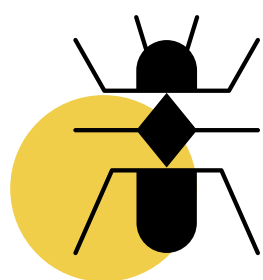
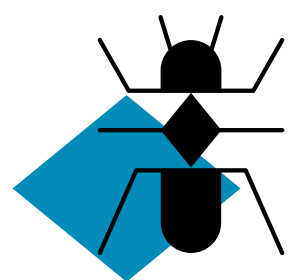
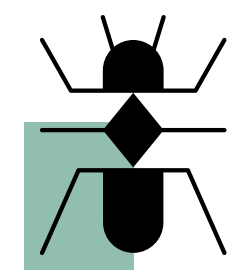
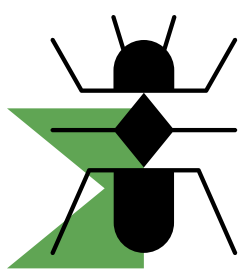
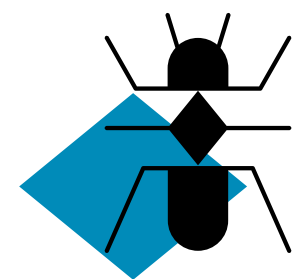
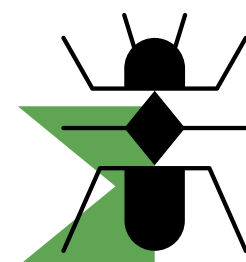
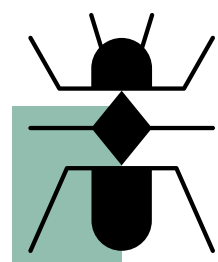
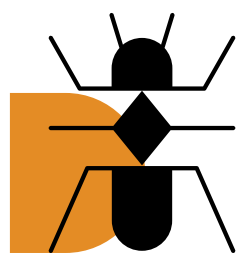
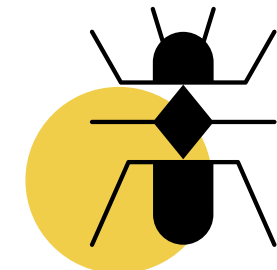
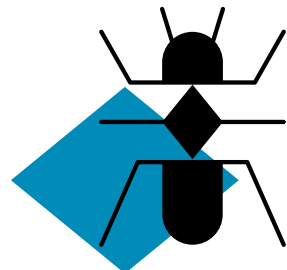
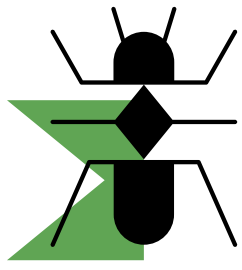
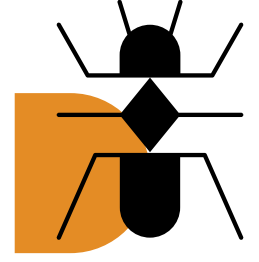
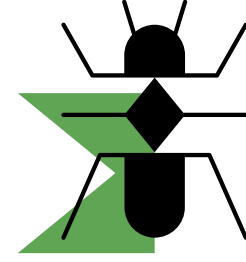
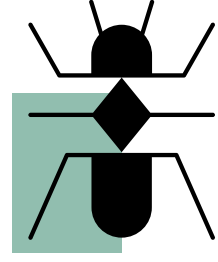
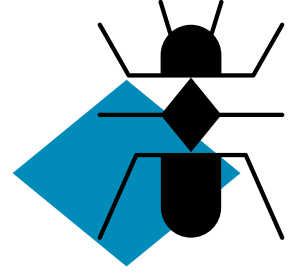
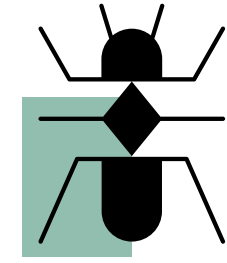
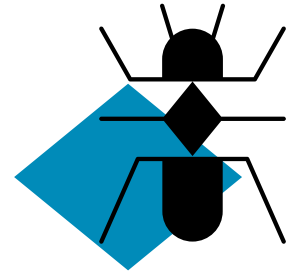
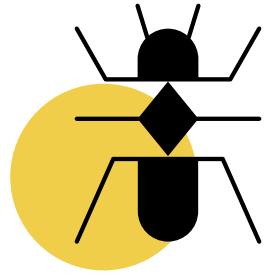
Obviously, the role and responsibilities listed here cannot successfully happen unless a team learns to properly, and adequately, *collaborate with one another*. This only happens when open channels of communication are available and used, sound vision and product context are provided by the product owner, and effective process leadership is facilitated by the scrum master.

There is no “easy” button that you can press to guarantee the creation of a great Agile team.  It takes patience, diligence, and quite a bit of determination. Great Agile transformation leaders recognize that change is always difficult and it takes time to create a self-organizing team that is empowered to confidently assume ownership of projects. With the right distribution of duties across roles and team members, we create an environment that proves our worth to the business. It demonstrates that we understand our actions serve a purpose and that all outcomes impact not only the current project, but the company’s direction as a whole.

About Targetprocess

Targetprocess provides a visual Agile management solution that enables customers to scale Agile from a single team to the enterprise level. Teams can customize Targetprocess to match their ways of working, visualize work from multiple angles to understand the big picture, and present a unified view of progress up to the portfolio levels to drive smarter investment decisions. Founded in 2004, Targetprocess has been selected by more than 1,000 customers and 51,000 licensed users in 80 countries around the world. The company has offices in Buffalo, Toronto, Berlin and Minsk. For more information, visit www.targetprocess.com.

2018



targetprocess
publishing

