VSM

Enabling Teams to take Responsibility of their Value Stream

HCL SOFTWARE
DevOps

Enabling Teams to take Responsibility of their Value Stream

Written by:

Steve Boone

Welcome

Have you ever been working and thought, “I wonder where that high priority bug for Customer XYZ is?” As a Product Manager for a large team, this happens to me multiple times a day. What did my process look like at other companies before having HCL Accelerate with filtering and search capabilities on the value stream?

- I get a request from support or a stakeholder asking for an update.
- I then have to figure out which team is working on it.
- I reach out to the team lead to see if their group is working on the item.
- The team lead confirms their team might be working on that item or something like it.
- I then need to go to the team’s Jira board, which tells me that it is “In-Progress” and is assigned to developer X. Being a developer for a long time, I know that status is probably out of date.
- I then need to reach out to that developer to ask some specifics, like “hey, where is this ticket?” only for the developer to respond with something like, “Oh, sorry, that is in the latest build. I need to update that card.”

If you have been on a large or even small software team, this should not be unfamiliar. It almost seems like the normal flow of work, but it’s not the best one. There have been many studies showing developers need to have long periods of uninterrupted time. Long periods of non-context switching allows them to get deep into problems and not lose their train of thought. The happy path of the above steps could take 30 minutes to get my answer, during which I made my team lead and developer stop their train of thought to get me just a status. This tiny distraction unblocks me but could cost them valuable cycles or maybe derail them for the rest of the day.

Enabling your teams to take responsibility of their value stream is about so much more than giving them the autonomy to choose the tools that suite them best (though that is an important part of it). It’s about giving everyone across the software delivery pipeline access to the same information so every individual contributor can see how their work impacts the business goals and every leader can see how to best support their teams. This is accomplished through value stream management that makes your DevOps data big and visible so anyone can quickly find the answers they are seeking, whether that be where an item is, who is working on an item, how long until that item might be done, and so much more!

This equity of information changes everything from stand-ups to just general questions about the software. In this eBook, you’ll learn how to achieve this DevOps data visibility and how to use it to improve your DevOps strategy. If you have any questions about the information in this eBook, please reach out to me at Bryant.Schuck@hcl.com.

Bryant Schuck

HCL Accelerate Product Manager

{Why is enabling teams to take responsibility for their Value Stream important?}

Is your organization having trouble scaling its DevOps adoption? For many organizations, development teams are quickly getting more agile while the rest of the organization holds them back. From database administrators who haven't embraced automation, or who have poor communication from the business to the product teams, there is plenty of room for improvement.

Value Stream Management can be a powerful tool to help streamline your software delivery practices.

A chain is only as strong as its weakest link, and in comparison, an organization can only move as fast as the slowest teams.

One of the key goals of DevOps is to help teams deliver higher quality software faster. Individual product teams often have dependencies on other teams. Knowing how those dependencies work and understanding the team-to-team handoff process is extremely important to determine how long it will take to deliver business value.

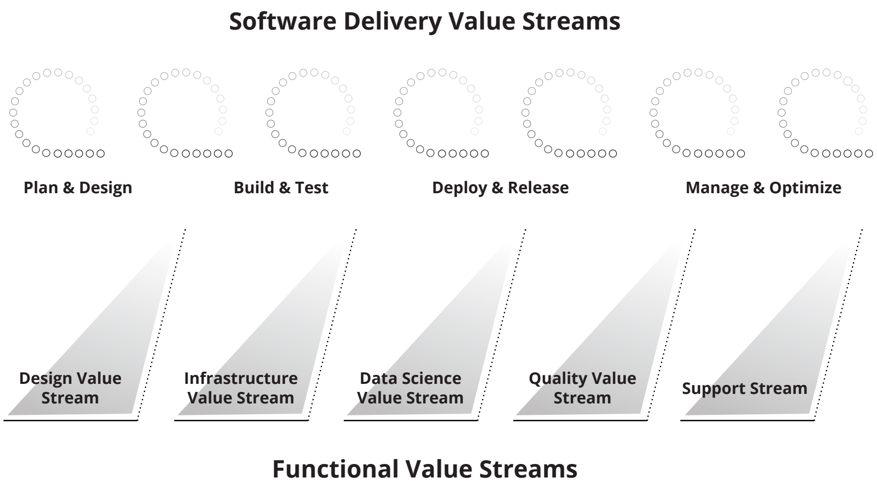
Value Stream Management can help teams get a better understanding of where their work exists at any given time. A proper value stream should help teams uncover time that is wasted, or in other words, help them to identify when works sits idle. These times are usually represented as Lead Time and Cycle Time. Lead Time is a measure of how long it takes business value to propagate from being planned all the way through to when the business value can be actualized. Cycle Time is how long it takes business value to propagate from when the developers are working on the value, to when it is delivered.

Application teams for years have been very good at streamlining their own processes. They've adopted agile best practices such as scrum, as well as planning and pointing meetings. And, they use retrospectives to improve how they internalize their work. They have even embraced processes to help the rest of the business consume their deliverables more efficiently by implementing continuous integration, continuous delivery, and automated testing.

However, that doesn't mean that the rest of the organizations that surround development are equally as efficient. As Dave West pointed out back in 2011, Hybrid Agile or "water-scrum-fall" is most often the norm. "Hybrid Agile methods are a reality in most Agile implementations. This happens in part because Agile adoption has been practitioner-led, leading teams to focus on domains they can influence, mainly the team itself. Areas outside of their control, such as business analysis and release management, continue to follow more-traditional approaches, meaning that Scrum adoption is limited to the development-team level. Compliance requirements are another factor driving hybrid approaches, as they call for strong governance processes before and after development."

(Citation: Dave West, Forrester - http://www.storycology.com/uploads/1/1/4/9/11495720/water-scrum-fall.pdf)"

A simple example of inner-organizational dependencies is how long it takes for a design team to produce the mockups before a development team can start working on the code. When we want to communicate to the business how long it will take before we can deliver value in a particular area, we can't only consider how long it will take for development to complete the code. Several other groups have work associated with the product's value stream including; design teams, testing teams, infrastructure teams, security teams, and QA teams. All of these teams have a hand in contributing to the throughput of business value.



Value Stream Management can play a crucial role in helping to understand and streamline the delivery process across multiple teams. It can even include many parts of the organization from development, testing, documentation, design, marketing, and sales. It is done by mapping out how work progresses from backlog to the customer and then enabling teams to identify and correct areas of improvement. This type of process improvement is what can help turn an average performing team into a high performing team.

By surfacing the areas of unknown waste, teams can work to refine their process to eliminate as much delay as possible. It also gives them the information needed to have intelligent conversations with other groups whom they work alongside so that handoffs and dependencies can be communicated clearly.

{Getting Started Managing your Value Stream}

To help us better understand how to manage our value stream, let's use a fictional product team, the ACME team, as an example.

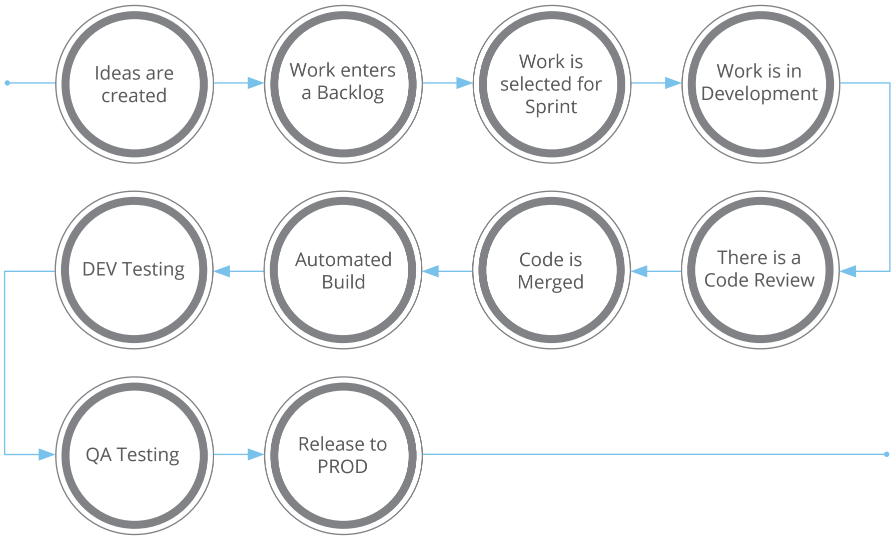
The ACME team is an agile eight-person team in a mid-sized financial company. They embrace scrum and meet daily to review the backlog and make sure they have a clear understanding of the work that is in the current sprint.

To estimate effort, they use story points. The ACME team does some of its own testing, but also relies on business subject matter experts, quality assurance teams, and security groups to deliver software through the business.

Plot out your value stream

The first step in better managing a team's value stream is to have them plot out precisely how they handle work on a day-to-day basis. This should take into consideration all the curveballs that get thrown their way during a given sprint or release.

First, we would have the ACME product team document what they believe is the current process they follow to deliver business value.



A team can usually plot out their value stream relatively quickly as they are the ones primarily doing the work daily. No single person will know the entire value stream. You'll need to work with many people/teams to iron out the full flow.

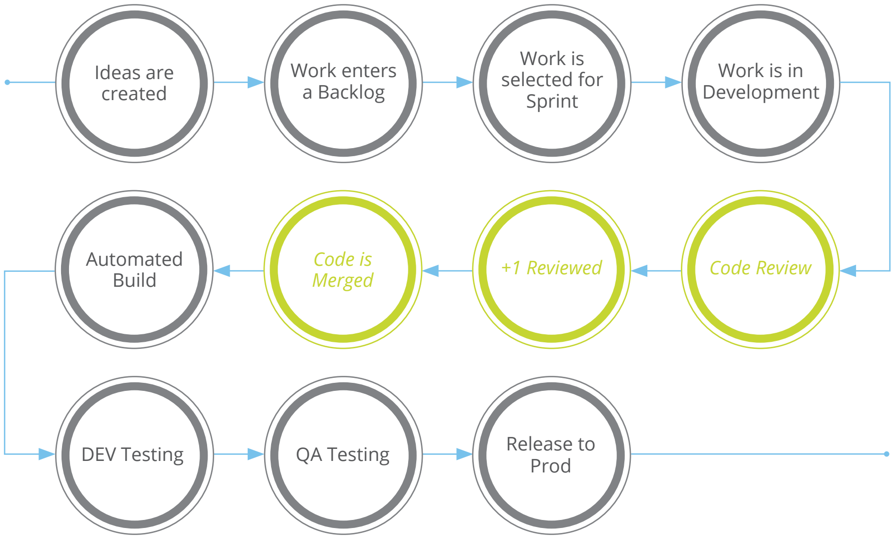
But even when the entire individual team works together to plot the value stream, this first attempt is often very product-focused. And, it may not directly express how teams communicate about their daily activities or implicit dependencies they may have with other teams.

It's important to note that while we are representing the value stream as a logical progression or flow, not every piece of business value will flow through all of the stages of a value stream. Often, we will identify areas where work may end up. We will discuss this more in-depth later when we talk about culture and cross team dependencies. For now, let's look at how we can improve the model.

Your value stream should reflect how your team communicates their work
Let's look at some of the ways teams can make their value streams more granular to reflect the actual work that takes place.

The ACME team has implemented a code review process, and the review requires more than one check before the code can be merged. The initial code review can be done by anyone on the team, followed by a second code review that can only be done by a senior developer. This makes sure that more than two pairs of eyes have looked at the code and dramatically improves the chances that they might find problematic code earlier in the development process. Teams often refer to this as a +1 (initial code review) and a +2 (senior code review).

It is important that our value stream reflects these types of cultural norms. A more granular value stream for the ACME team would be:



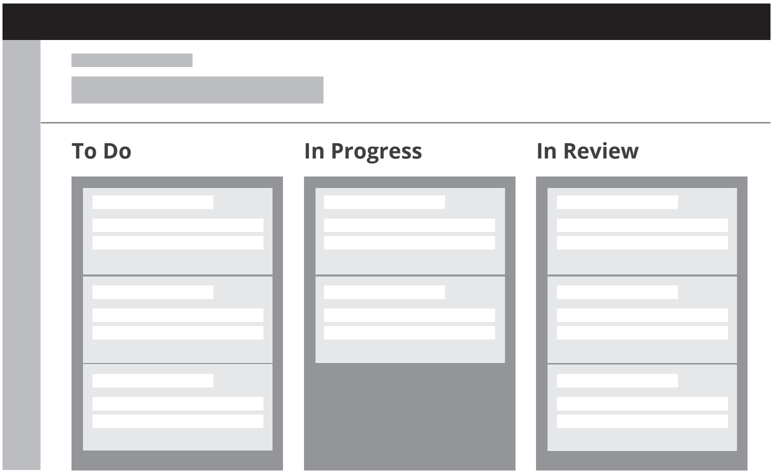
This value stream more accurately reflects the way the ACME team communicates their work, and it allows the team to track their work at a more granular level. This could help expose hidden costs of delay in their process. If the ACME team believes they waste too much time in code review, how would they know which process is holding them up? Are they getting stuck waiting on the initial code review, or are they getting stuck waiting for senior code reviewers? By tracking at a more granular level, they can answer these questions.

Another question we might ask the ACME team is, how do we determine what work will be selected for a given sprint? Many factors typically play into this decision-making process.

- Has the work been adequately evaluated? Do the developers all understand what it takes to complete this work?
- What is its priority? Should other work be prioritized before this?
- Are we waiting on any other deliverables, such as design input, or product management feedback?
- Is this work in line with the business priorities for this release?

The answers to these questions are sometimes found in tools such as JIRA, or GitHub. These types of issue tracking tools allow teams to track their sprints, as well as the points associated with a given story. They also aid in prioritization and can be a great place for development teams and product managers to collaborate. With the growth of "Product Ops" emerging, many product managers are turning to tools such as "Aha!" to plan priority and manage product roadmaps. A well-designed value stream should take into consideration business operations that take place before and after development.

Accurately reflect your Value Stream in the tools you use
Work item management tools often have their own workflows, which can be leveraged to better align with your value stream. For example, the basic out of the box workflow in JIRA is



While this very basic workflow is simple, it identifies three common states of work. First, something needs to be done (To Do). Second, someone is doing the work (In Progress), and finally, the work is complete (Done).

Tracking your work in this very basic model is a disaster for Value Stream Management because it disguises the granular work that is taking place. Let's revisit the example value stream we discussed earlier.

If we were to compare that with the JIRA workflow, there are clearly many steps in our daily process that JIRA would not adequately capture. If our management asks us, "where is the latest work items for the next big feature," we can't simply say "oh, it's in progress."

Is it being developed? Is it in code review? Have we merged it? Do we have an initial build of the functionality? Has the QA team had a chance to test it?

All these questions can be answered if we structure the tools we work with daily to reflect our process. This will require moving away from a simple work item tracking flow of To Do, In Progress, and Done. Instead, we should have meaningful transitions that teams can easily understand.

Culture, communication, and Value Streams
A fundamental part of DevOps success is culture. And a monumental part of culture is communication.

If we are trying to improve how a team organizes and delivers work, it is essential to make sure that the language they use is consistent across the team and the tools. For this reason, one of the first exercises we do with teams is to make sure that the value stream they have mapped out is reflected in the tools they use daily. This means configuring your workflows in JIRA and other tools to represent at a granular state, how work is communicated.

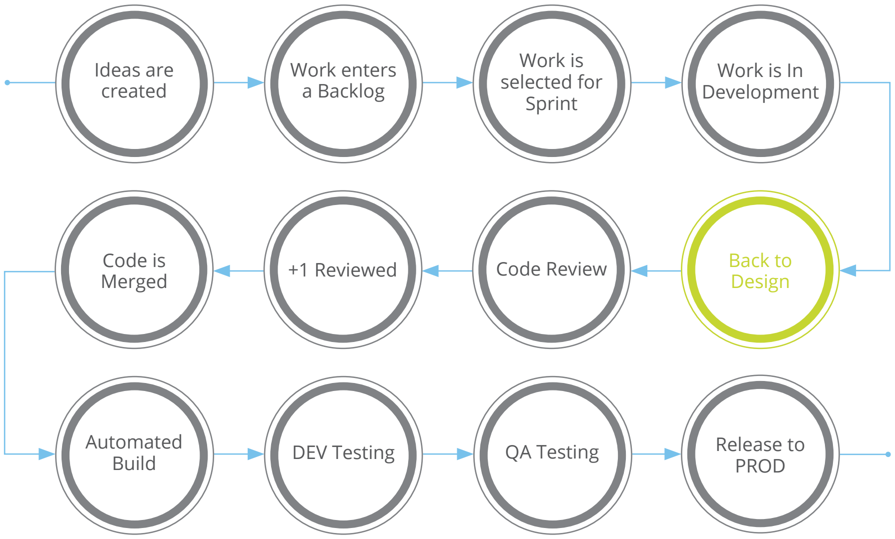
For example, the ACME team during their morning scrum will often refer to work items that need more discussion as something that should go "back to design." By simply saying the phrase "back to design," the entire team knows exactly what this person is trying to convey because it has become a cultural norm. For ACME, "back to design" tells others on the team that they have more questions than answers about how to implement a particular story.

Cultural norms are the standards by which we live in our communities. They are the shared expectations and rules that guide the behavior of people within a social group. Cultural norms are learned and reinforced by the people we interact with daily.

By acknowledging those cultural norms in our value stream, the business can empathize with the day to day tasks. It also works to generate a common language across teams.

One of the benefits of capturing empathy across teams as part of your value stream is that it can nudge everyone to worry about the larger business challenge of rapid delivery, instead of just focusing on their part of the process.

An adjusted ACME team value stream that takes into consideration cultural norms may look like this:



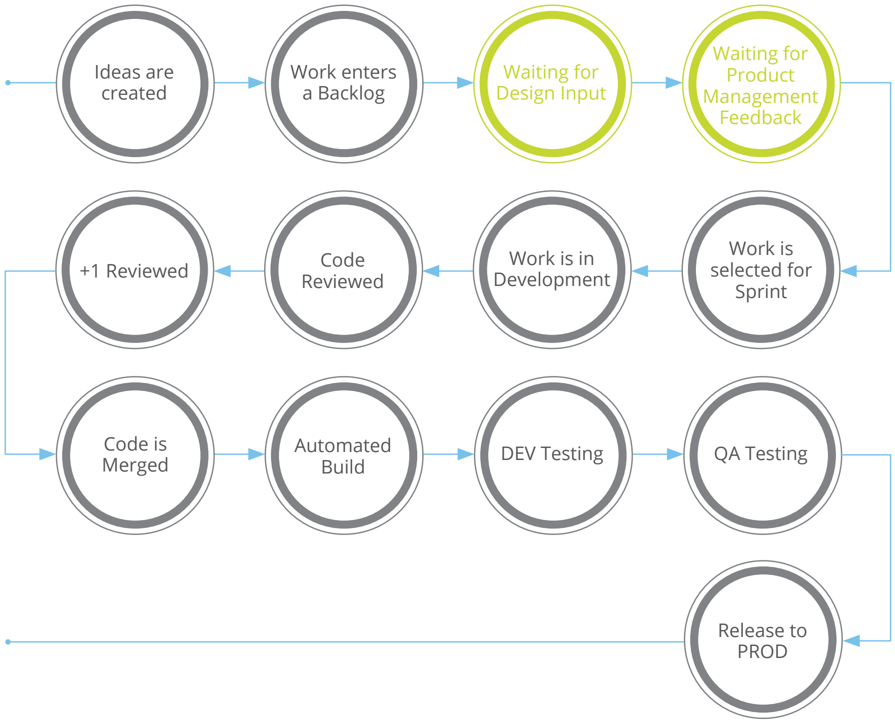
Capture dependencies with other teams
It takes many cross-functional teams to put out a successful release. While the work being done within those groups might not be directly related to your team, it is crucial that we have visibility into how long teams take to come back to us with the information we need.

Let's revisit the ACME value stream: (on opposite page)

Upon further conversations with the ACME team, they have explained that they often need to get additional feedback from designers before working on a solution. They occasionally need to get more information from customers as well, in which case they work directly with their product managers.

Notice that our value stream does not provide a way for the ACME team to represent this part of the delivery process. True; it may not be necessary all the time, but it has an overall impact on our Lead Time, because it effects the amount of time before development can start working on the items.

A more realistic view of the ACME Value Stream might be:



It is also important to realize that just like the ACME product team, the ACME design team, and the product managers all have their own value streams. By identifying those parties in the ACME value stream, they can ensure they know how long a work item typically takes to move through design or to get feedback from stakeholders. This can result in much better estimations for the business when planning a large-scale release.

{Measuring your Value Stream}

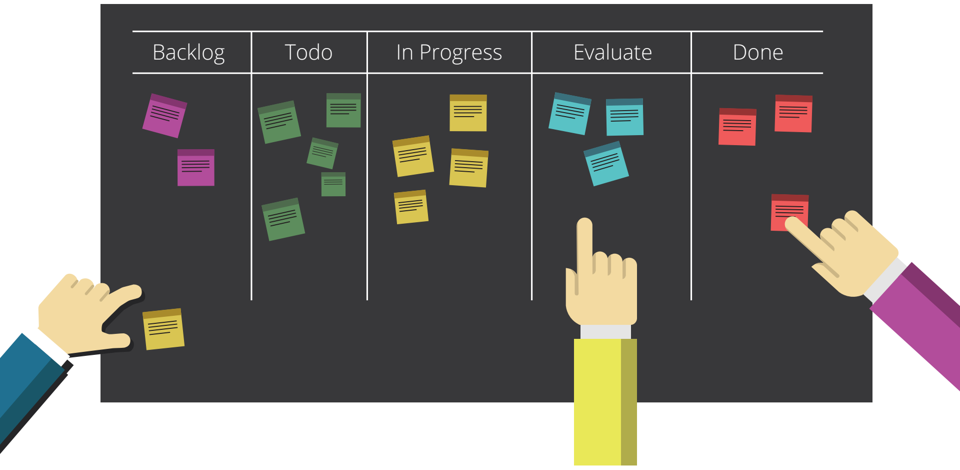
Once you have successfully mapped your value stream and configured your daily development tools to reflect how your teams operate, the next hurdle is optimizing your value stream. Organic conversation is the best place to start. In keeping with the spirit of Agile, your product teams must have regular retrospectives to discuss what they feel they can improve. Even more important is having a culture where the team feels they are responsible for owning any corrective actions they feel should be taken to improve the overall efficiency.

Have your teams discuss the work they are delivering in relation to the value stream. Many teams will keep a printed or whiteboard representation of their value stream in their retrospective room. This allows the team to discuss precisely where they think bottlenecks occur.

If a particular story caused problems during an iteration, it's essential to understand the cause of the issues. Where did it get held up? Could we have planned better for this situation?

Common reasons that work doesn't get completed are:

- The team was impacted by unplanned work (support, excessive meetings, etc.)
- The work was dependent on input or contributions from another team, and it didn't get done (design, QA, other development teams)
- We were waiting for feedback from a customer before finalizing the implementation
- It was helped up in code review
- There was too much WIP (work-in-progress), and the team couldn't contain the story



Many teams find measuring the effectiveness of a value stream difficult. Sometimes it takes many releases for a team to get a feel for a new process and the best way to operate.

In the same way that agile development is iterative, so should be the health and the evolution of your value stream. Teams should always be striving to improve the process they rely on to deliver business value. Teams will typically discuss day to day challenges in retrospectives, but rarely is their time for in-depth reflection on internal processes. Because of this, many teams only reevaluate their value stream once or twice a year.

If we have learned anything from DevOps, we know that fast feedback on our processes is of utmost importance. Taking an audit of our internal processes to ensure they are optimal only twice a year is far from what many agile organizations would consider "fast."

Thankfully, all hope is not lost as the industry has centered around a few key metrics to help us validate that our value stream is functioning optimally.

Value Stream metrics that matter
There are four key metrics that high performing teams, and teams striving to become high performing focus on. These are

- Lead Time: The time it takes for business value to move from planned work, to code successfully running in production
- Change Fail Percentage: How often deployment failures occur in production that require a fix
- Mean Time to Restore (MTTR): The average time it takes to recover from a failure or service outage
- Deployment Frequency: How often a team can successfully execute a deployment into a production environment

One of the key ways to improve these metrics is by optimizing your overall value stream. Let's look at some examples.

For Lead Time, take a look at the work items your development team is currently working on. Ask yourself this question: "When will this work see the light of day?"

In other words, when can we actually get this work into production, or more specifically, when can we get the value to the customer. The value can come in a number of different forms; It could be revenue, or many times it might just be honest feedback, which can help replenish our backlog.

In an optimized value stream, we will have identified areas where work gets held up and will be able to put processes in place to remove bottlenecks and improve collaboration, therefore improving our overall lead time.

Change fail percentage helps us understand how predictable our value stream is operating. Are we catching defects early in our development processes? Can we consistently and reliably deliver into production environments without impacting our customers or the business? Teams with a high change fail percentage typically are capable of moving fast but lack the quality standards in place to allow them to move fast effectively.

A clear value stream can assist in lowering change fail percentages by opening the conversation around how defects manage to make it through the value stream without being identified before production.

Mean time to restore speaks to a team's ability to pivot on demand. All teams want to optimize how much work they can plan for. One of the biggest showstoppers to delivering on time is having to put out production issues. To be able to put out a production issue efficiently, teams need to identify the basics quickly:

- What's currently deployed in production
- What changes recently went into production, and who is responsible for them
- What servers/clients are impacted

A team with an optimized value steam can easily and quickly identify these items. They will have successfully tracked all work items associated with a production release. And, they know precisely who worked on them by being able to associate the work items to both source control and their respective continuous integration and delivery tools.

Additionally, many continuous delivery tools available today will give you precise information as to where the vulnerable code was deployed. Armed with this information, teams can quickly reconcile the defect and push a resolution through their daily pipeline.

Deployment frequency focuses on production deployments only, because again, production is the only place where we can deliver any value.

Many companies have agile product teams that are trapped inside a non-agile organization. Having agile product teams does not make you an agile organization.

However, many teams find it impossible to change the culture and working process of the larger organization. Over the years, we have found one reliable way to help persuade other parts of the organization to operate more effectively, and that is by surfacing data. Teams that can have a factual conversation with other parts of the business and demonstrate how current processes are impeding customer value will typically encourage others to try to look within and determine what they can do better.

One of the exciting areas of Value Stream Management is showing others how your team has taken responsibility for improving their day to day activities. By making teams responsible for their own value streams, the organization can move much more efficiently based on their customer's needs.

By empowering other areas of the business to take responsibility for their value stream, we are, in turn, making the enterprise more agile.

{Digitalize your Value Stream}

Getting the data needed to accurately track the metrics that matter can be challenging. You need to pull data from multiple toolsets, source control, issue tracking, automation delivery tools, security monitoring, testing tools, and more. This leaves many product teams stuck to build dashboards themselves.

Dashboards can be helpful, but the real magic is in the relationships between the data.

UrbanCode Velocity was designed to help teams visualize their value stream in such a way that they can get fast feedback on the efficiency of their day to day practices. Velocity sits on top of your existing toolsets and aggregates multiple types of data. From there, it works on building the key relationships between these data points so that you can start to track the metrics that matter, and much more.

By digitalizing your value stream, you can uncover the real data behind software delivery. No longer can people say, "I think our challenge is..." or "I believe the answer is..." when it comes to process improvements. The data speaks for itself.

Work Tracking: Be certain which business value is being delivered into production. Know the specific work items, test results, and history of each production change

Lead and Cycle Time: Tracking how long it takes for work items to move through your organization becomes easy when you manage the relationships across your DevOps tool sets.

Productivity Trends: Are our teams productive? When are they least productive? What kinds of activities slow progress? By monitoring the movement of business value, we can clearly see where work becomes idle.

Security Visibility: Security is more important than ever. Easily keep track of which teams are running security scans and trend their progress over time.

Quality Trends: Everyone wants to go fast, but going fast with poor quality doesn't help anyone. Make sure that the work progressing through your value stream is quality and use intelligent gating to protect your customers and production environments.

Business Value: Is your product on the hook for delivering a large set of functionalities over the next couple of months? Easily query the work in your value stream to identify what work is completed, what still remains to be done, and what is being held up.

Business Predictability: Forecast with confidence how long it will take to complete a certain set of business value based on the actions and data from your team's past results.

{Introducing UrbanCode Velocity: Who is UrbanCode Velocity for?}

Product Owners and Development Managers

For people close to development teams, the benefit of digitalizing the value stream is huge.

- Accurately predict how long it will take to deliver a feature or set of functionalities
- Know if the team is ahead or behind schedule
- Understand how development teams are spending their time (feature work, defect fixing, team tasks, etc)
- Identify work items and epics across multiple releases

Security and Governance

Keeping our customers and business safe is extremely important, with UrbanCode Velocity you can:

- Make Go or No-Go decisions based on security and quality metrics
- Gain visibility on teams that have not recently executed security scans
- Understand who deployed what with detailed audit report logging

Product Teams

A highly motivated development team is something to be celebrated. By visualizing the backlog product teams gain access to key data points, including:

- How fast are we squashing defects?
- Is quality in this release any better than the previous release?
- What features/fixes are new today?
- What should I work on next?
- What is our WIP (Work in Progress)

Tech Executives

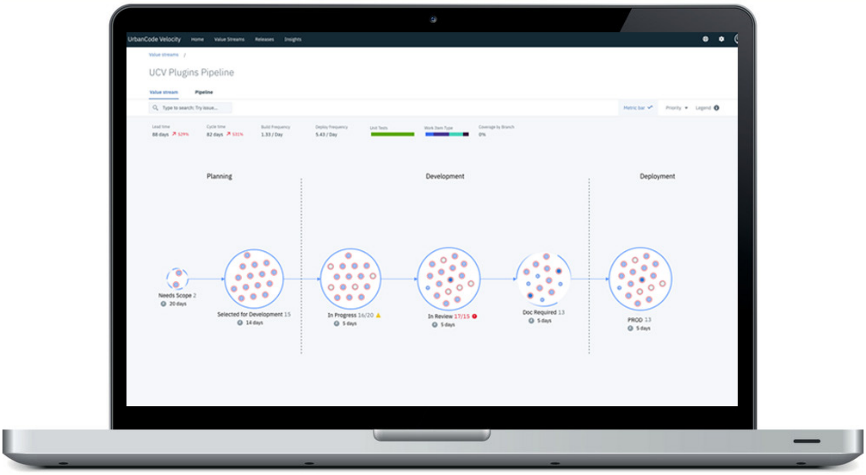
Companies are investing millions of dollars into DevOps initiatives. It is important to make sure you are getting your ROI. By aggregating and looking across multiple value streams executives can now determine:

- From concept to cash, are we delivering value faster?
- Where is my business not keeping up with my agile IT department?
- What makes our best teams "high performing"?
- What are the best practices of our organization?

{Free Trial: UrbanCode Velocity}

If you'd like to learn more about UrbanCode Velocity, you can visit http://www.UrbanCode.com.

Or if you're ready to start applying these principals to visualize your own value stream, visit http://UC-Velocity.com today to claim your free 60-day trial of the standard edition of UrbanCode Velocity.


UrbanCode Velocity
Home
Value Streams
Releases
Insights
Value streams /
UCV Plugins Pipeline
Value stream
Pipeline
Type to search: Try issue...
Metric bar
Priority
Legend
Lead time
88 days 529%
Cycle time
82 days 531%
Build Frequency
1.33 / Day
Deploy Frequency
5.43 / Day
Unit Tests
Work Item Type
Coverage by Branch
0%
Planning
Development
Deployment
Needs Scope 2
20 days
Selected for Development 15
14 days
In Progress 16/20
5 days
In Review 17/15
5 days
Doc Required 13
5 days
PROD 13
5 days