The Four Prerequisites For DevOps Success

UpGuard

Table of Contents

Executive Overview 3

What Exactly is DevOps? 4

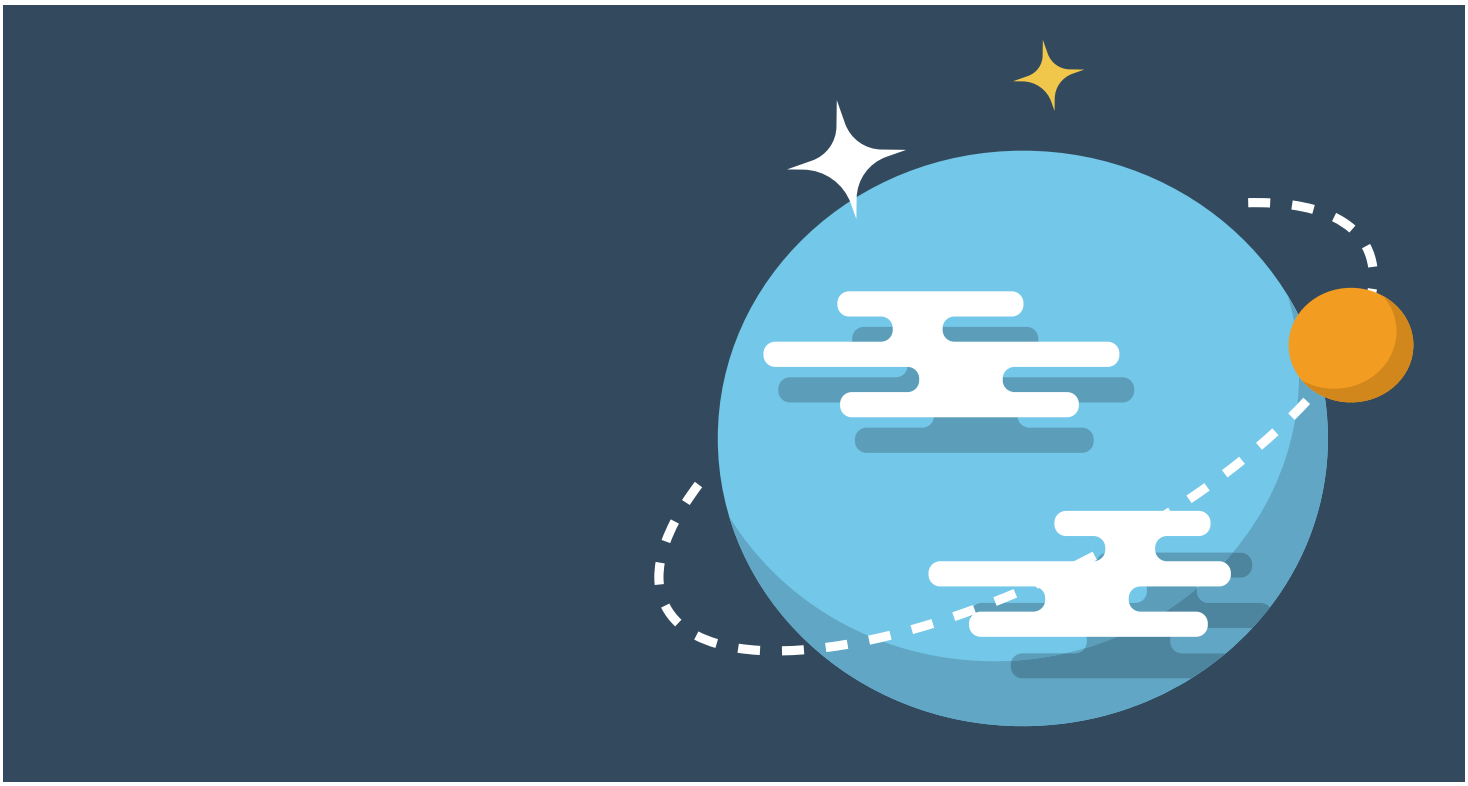
I. Understand What The Business Goals Are 5

II. Get Situational Awareness and Watch for the Drift 7

III. Clearly Define Processes and Stakeholders 10

IV. How Do You Measure It? 12

Executive Overview



Every day, the world is becoming more instrumented, interconnected, and intelligent. Infrastructures, assets and devices across the globe are rapidly being digitized, made available 24/7/365 and transforming everyday products into smarter products that allow people, systems and objects to communicate and interact with each other in entirely new ways. The ability to instrument and interconnect intelligent products and information technology is making it possible for industries to transform their business models and deliver new, innovative services that exceed expectations and deliver unique client value.

In order to win in today's fast paced world, businesses must maintain stability and reliability whilst depending on constantly changing envi-ronments that support them.

This background is driving the adoption of DevOps, and is reinforced by the following trends:

Consumerization of IT

A complete blurring of lines between personal, social and work activities with expectations that everything is accessible, everywhere and at all times.

Cloud Computing

Creates the possibility (and expectation) for better, faster, cheaper and easier. Also, trades complexity for agility and flexibility.

IT as a Service

Cloud vendors are playing an increasing role within Enterprise IT, freeing up internal IT to focus on value added services.

What Exactly is DevOps?

DevOps refers to the emerging professional movement that advocates a collaborative working relationship between Development and IT Operations, resulting in the fast flow of planned work (i.e., high deploy rates), while simultaneously increasing the reliability, stability, resilience of the production environment. Today, we can see how DevOps patterns enable organizations like Etsy, Netflix, Facebook, Amazon, Twitter and Google to achieve levels of performance that were unthinkable even five years ago.

Goals of DevOps

The general goals of any DevOps initiative could be categorized as follows:

People	Process	Technology
Cultural Change - Collaboration, Cooperation and Transformation to a Learning Organization Improved knowledge retention & deeper knowledge understanding of the business.	**More agile change process** Increased rate of acceptance for new features & visibility across the product .	**Improved quality of code** Reduction of defects in each subsequent environment deployment and increase in the test coverage of code
Increased responsiveness to business needs Increased understanding to prioritize for feature requests from the business ability to communicate.	**Improved quality of software deployments** Reduction in the number of incidents/ problems per product per release.	**More agile development** Increased deployment for product features.
	More frequent software releases Increase in ve-locity of product releases to production.	
	Improved visibility into IT Process and requirements Visibility into the stages & underpinning contracts required to release software within the organization.	

I. Understand What The Business Goals Are

DevOps could be accused of being the flavor du jour in today's modern IT shop. Being able to define DevOps is an important first step. The above list of DevOps specific goals is a great framework for understanding where it can be applied. Neither, however, is sufficient for success. They will help frame discussions and categorize work, but if pursued independent of actual business goals their application will be misguided at best, counterproductive at worst.

Before jumping into the deep end of your DevOps initiative, you should aim to have as many conversations with your business counterparts as possible, including executive leadership, to better understand what their specific strategic business initiatives are, what key challenges and friction points they are currently experiencing, and their overall needs and wants of IT. These foundational conversations will help make your DevOps initiative more effective in the long term.

In order to win in today's fast paced world, businesses must maintain stability and reliability whilst depending on constantly changing environments that support them.

This background is driving the adoption of DevOps, and is reinforced by the following trends:

Listen to your quarterly earnings calls

Read your financial disclosure documentation

Review your corporate website for key calls to action, themes, press releases, job postings, etc.

I. Understand What The Business Goals Are

This research will help you understand what the business cares about, but it is only the beginning.

Your goal should be to meet with as many of your business cohorts in short order to interview them and bet-ter understand their challenges, wants and needs.

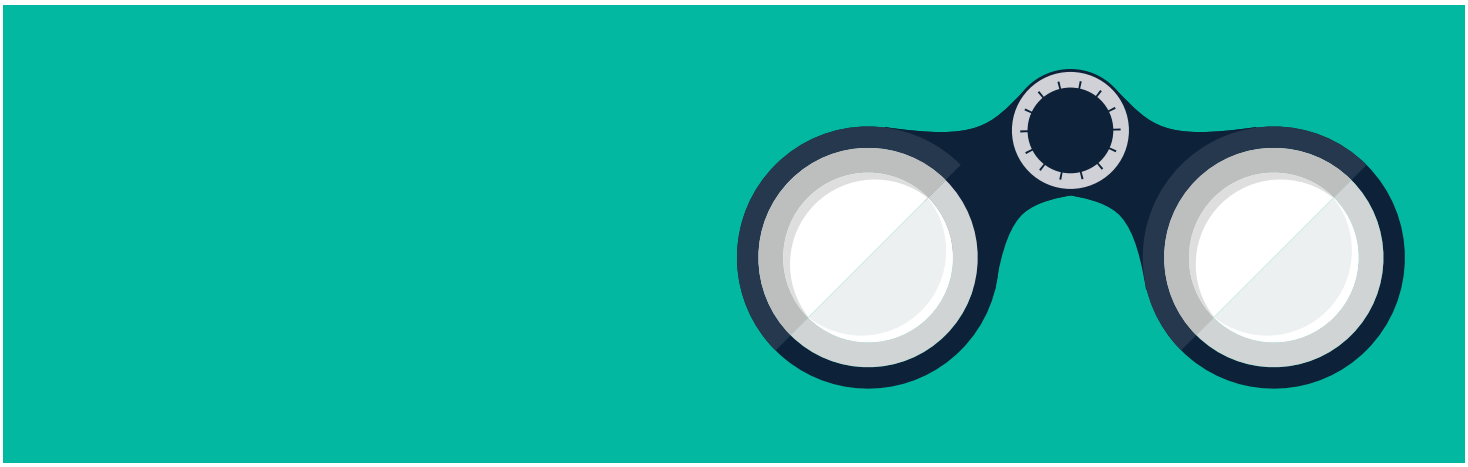
We'll refer to the definitive book on Enterprise DevOps, The Phoenix Project, for the types of questions you will want to use during these conversations:

> ***"To make sure I don't have any incorrect or preconceived notions, could you tell me exactly what you do here at <company name>?"***
>
> ***"What differentiates a good day from a bad day for you?" (You want to extract key pain points from the "bad day" scenario. Address these and you have a supporter for life)***
>
> ***"What are your goals, objectives and measurements for this year?"***
>
> ***"Which of your goals, objectives and measurements are most at risk?"***
>
> ***"For each of these initiatives and measurements, who are the manag-ers held responsible and accountable?"***

Notice that you are not trying to 'sell them' on DevOps – they wouldn't know what they were buying even if you did. The critical part is to ask open ended questions that enable you to better understand the key needs and strategies the business is working on. By having a clear picture of their focus areas, it can allow you to align your DevOps movement on the most strategic plans and have the biggest effect on the organization as a whole.

In reality, DevOps does not solve everything.
It is a philosophy for IT collaboration.

II. Get Situational Awareness and Watch for the Drift



Does this sound familiar... the production release of a deployment that worked like a charm all the way through testing has gone completely off the rails. The first questions are naturally What is different? What changed? But in order to answer these questions, you must first need to establish proper situational awareness. It is impossible to identify deltas without first understanding what you have.

Fundamental to your successful DevOps implementation is the establishment of an understanding of what systems and assets you have across an entire spectrum of physical devices, virtualized systems and hybrid cloud instances. The lack of adequate situational awareness has been identified as one of the primary factors in accidents attributed to human error. It all starts with ensuring you have complete visibility of the state of your heterogeneous environments to ensure more reliable and repeatable delivery of IT services. Having comprehensive, accurate and up-to-the-minute visibility is the foundation for successful decision making across an array of complex and dynamic systems. The net-net is that situational awareness is what you must have in order not to be surprised later.

For your DevOps initiative, situational awareness takes a few shapes and forms:

Discovery
Have a clear picture of all devices in your environment

Configurations
Know the state of your infrastructure

Changes
Understand, in context, how, when and what part of the state has changed

> *"80% of all mission critical IT service outages result from people and process failures, and of those outages, more than 50% result from a lack of coordination between change, release and configuration management processes."*
>
> Source: Gartner

II. Get Situational Awareness and Watch for the Drift

One of the core advantages of DevOps is the alignment and collaboration it can create. By establishing a clear picture of the state of your entire heterogeneous infrastructure, you can now have a conversation in context with the various stakeholders. For example:

Application Development
Integrate deployment tools to ensure accurate and automated provisioning of builds across test lab and production environment.

IT Operations
Ensure that service support and bug tracking tools across operations and development remain synchronized for better problem tracking, customer support and continual service improvement.

Quality Assurance
Test application performance in lab and production environments, and feed root cause back to testing, to speed fixes and reduce mean time to repair.

Enterprise Architecture
Have visibility into the current production environment to ensure effective application and service design.

Security and Audit
Leverage a common set of tools for reduced security risk and improved compliance reporting.

Service Management
Ensure that information is shared and synchronized across asset stores, so that details are current and consistent across audiences and tools.

Third Party Vendors
Improved accountability to measure effectiveness of SLA and more specifically measure your delivery costs.

Business Counterparts
Enhance the customer experience with faster deployments and feed-back loops, all while reduced waste.

Project Management Office (PMO)
Ability to manage the portfolio of enterprise products and release on-time with confidence.

Once visibility is created across key stakeholders, it is critical to understand the gaps and evaluate all changes happening to those systems.

Not for the purposes of inspecting all changes, but to have situational awareness of what is happening.

The reality is that over time, drift from your known, trusted state will occur. Your jobs is not to prevent those changes from occurring per se, but rather to know about them so you can focus on shipping features, not fixing configurations.

Establishing a culture of visibility stems from a culture of quality. When tests fail, there should be blinking lights everywhere. When everyone knows something is broken and collectively works together to fix it, that is when you know you've created a culture of visibility that will serve you well in the short and long term.

III. Clearly Define Processes and Stakeholders

Defining a clear set of accountabilities throughout the lifecycle of any product in an organization is a challenge. The multitude of touchpoints and integrations means that the task of aligning, prioritizing and executing on the requirements of all stakeholders is challenging one.

Before implementing DevOps practices within your business it is critical to have first clearly defined the relevant processes and stakeholders. Securing support from key stakeholders early is critical. Communicat-ing on familiar terms will assist with this, helping to increase buy in, visibility, and often accelerate a DevOps initiative.

The below diagram is a useful framework to start from for your analysis of processes and stakeholders. It clearly lays out the elements in play, and their integration points.

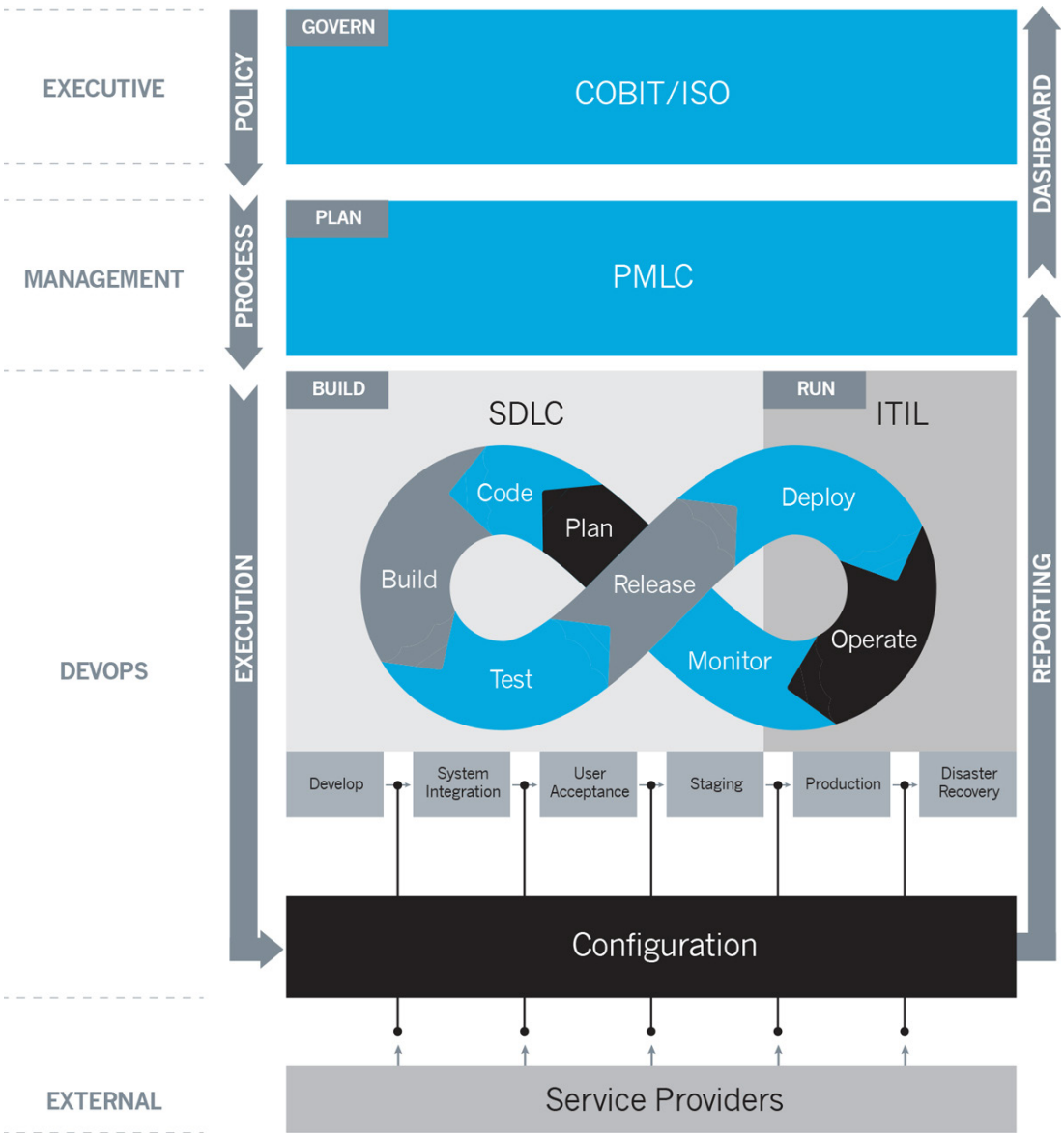
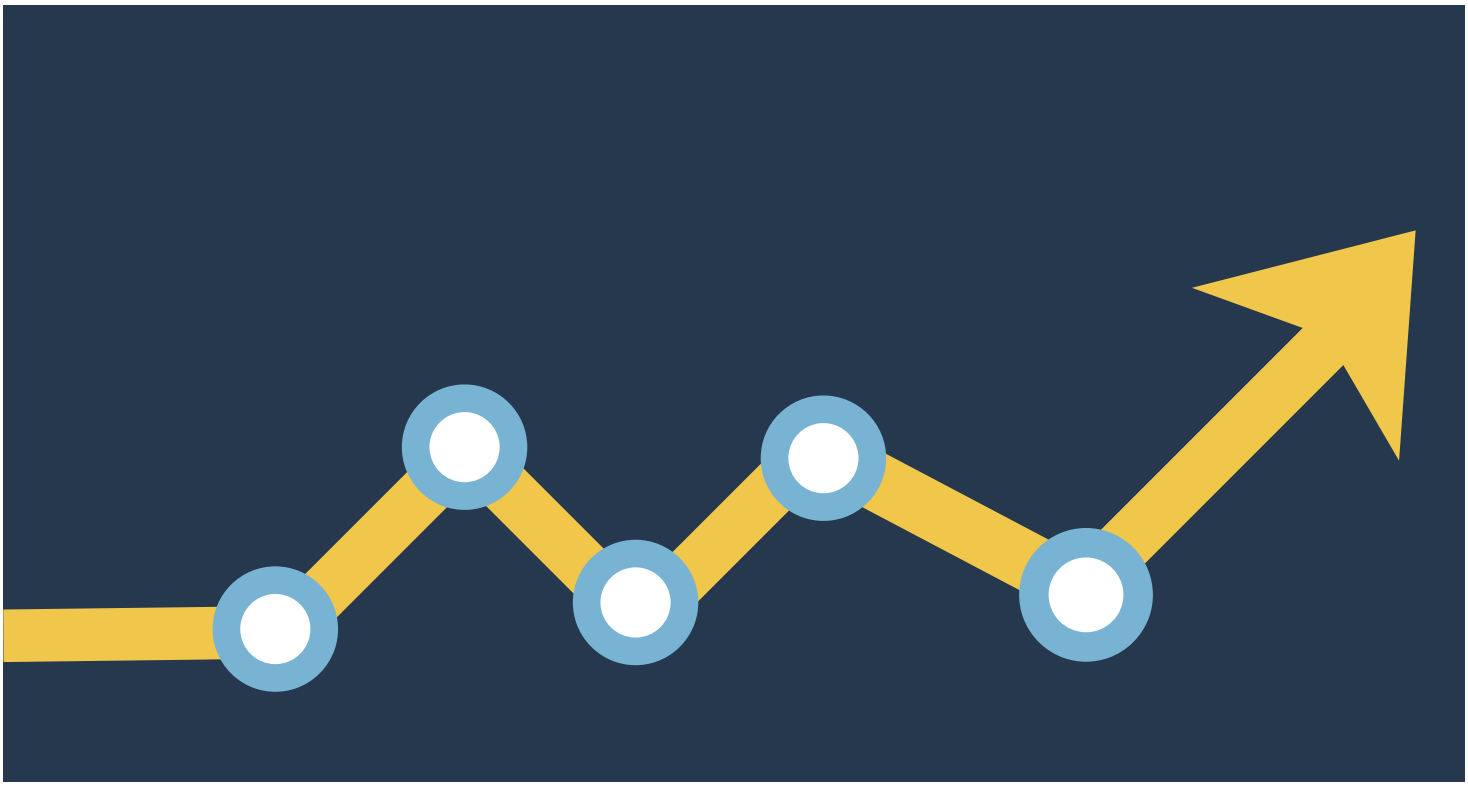


Figure 1: Typical Enterprise model defining operational stages that relate to Govern/Build/Run functions.

Whilst not comprehensive, below is an Enterprise cheat sheet for discovering integration points and technology overlaps between different "factions" that may be in play within your organization. It will help you to better define roles and responsibilities relevant to your DevOps initiative.

Processes	Process Integration Points
Control Objectives for Information Technology (COBIT)	Owner: Chief Risk Officer, Head of Service Mgmt, Risk Analyst Processes: MEA2/MEA1/BAI6/BAI8/DSS2/DSS3
IT Infrastructure Library (ITIL)	Owner: Change Manager, SVP Service Management Office, Release Coordinator Guides: Service Transition, Service Design, Service Operations. Processes: [Configuration Management, Change Management, Release Management, Knowledge Management] Assets: [Definitive Media Library, Definitive Software Library, Definitive Hardware Library, CMDB]
Software Development Lifecycle (SDLC)	Owner: Application Owner, Application Support Manager, Head of Engineering, SVP Application Development Phase: 1. Design Specifications 2. Design/Develop/Test Software 3. Implement Systems 4. Support Operations
Vendor Management - eSourcing Capability Model for Service Providers (ESCM - CP/SP)	Owner: IT Procurement, Head of IT Risk, VP Financial Management, SVP Vendor Management ITSCM - SP CAPABILITY AREA: Knowledge Management Share Knowledge - Capability Level 4 Knowledge Systems - Capability Level 3 Version & Change Control - Capability Level 2 Provide Required Information - Capability Level 2 CAPABILITY AREA: Knowledge Management Workforce Competencies - Capability Level 3 Assign Responsibilities - Capability Level 2 CAPABILITY AREA: Performance Management Capability Baselines - Capability Level 4 CAPABILITY AREA: Technology Management Control Technology - Capability Level 2 CAPABILITY AREA: Threat Management Statutory & Regulatory Compliance - Capability Level 2 CAPABILITY AREA: Service Design & Development Design & Deploy Service - Capability Level 3 Service Design - Capability Level 2 Verify Design - Capability Level 3 Deploy Service - Capability Level 2
The Open Group Architecture Framework (TOGAF)	Owner: CTO, Enterprise Architect, Infrastructure Architect, Solutions Architect. D: Technology Architecture Part V: Enterprise Continuum & Tools: Technical Reference, Model Architecture, Continuum Solution, Continuum
Sherwood Applied Business Security Architecture (SABSA)	Owner: Chief Risk Officer, Chief Security Office, Information Security Manager, Logical Architecture: Information Assets, Component Architecture: Process Tools & Standards.

IV. How Do You Measure It?



As Lord Kelvin said, "If you cannot measure it, you cannot improve it". A statement so obvious that it feels redundant. As anyone who has worked within Enterprise IT knows though, its importance cannot be understated. All too often the results of an initiative are poorly understood. Success is often based on meeting a high level milestone, rather than on an actual analy-sis of impact and results. As cliched as it may seem, the image of an executive declaring success as they emerge from the burning wreck of a disastrous project is an all too familiar one.

So yes, measurement is always important. It is particularly important for a DevOps initiative though. This is because a DevOps initiative targets the very processes that determine how IT work is done. Get-ting it right is a force multiplier. Getting it wrong can be a disaster, especially when recognition of this fact comes from trailing indicators.

Not being able to measure is a problem. Not knowing what to measure is a catastrophe. This is where you must start. In IT we are often quick to rush to buy a solution for a problem we have, but have not properly defined. A seismograph is a great measurement tool. Would you buy one if you were research-ing melting ice caps though? Of course not.

Ask most people in this space about "DevOps Monitoring" and they'll happily tell you that you need log, infrastructure, configuration, application and network monitoring. They'll tell you to buy a dashboard solution. They may also recommend an API integration service to tie everything together.

Would these be bad purchases? Well, yes and no. You could extract value from each and every one of these types of tools. Starting with a tool of measurement though, without clearly defining what you want to measure, is a recipe for time wasting. Success of an IT monitoring initiative cannot be measured in terms of data gathered or metrics surfaced. It can only be measured in terms of its ability to influence business decisions in a positive way.

For example, this is the wrong type of monitoring goal to start out with:

"Ability to measure long running SQL transactions"

Is it a bad thing to have? Well, no, but is it meaningful for your business at this moment? Are the appli-cations relying on SQL having noticeable performance issues? Are customers complaining? Is money being lost? If the answer is "No" to these questions then it should not be a focus. You need to choose the metrics you measure very carefully. Metrics that aren't useful, or aren't important, represent noise and wasted effort to capture.

IV. How Do You Measure It?

All metrics should therefore be based on business goals. It's OK to start out at a relatively high level. You just work your way down from there within the context of the starting point.

Of key importance here is understanding business metrics before diving into IT metrics. It cannot be an afterthought. The process should be:

It would be wrong to be overly prescriptive here by publishing a definitive list of what to measure. It must be fit for purpose.

A company that manufactures medical instruments is unlikely to ever embrace Facebook's "Move fast and break things" motto. Here are some examples of the types of metrics you may want to track:

It is important to understand which of the above matter most to your business. It is important to recognize that some of them are naturally in conflict. You cannot say "We're going to focus on all of them!". Sort them in order of priority and try not to focus on any more than three. If you are unsure then always refer back to the defined business goals.

Understand the business goals your initiative must align with (see #1 above). In an ideal world these will be defined and measured by the business already. If they are not, you will need to work with the business to define them. If they cannot be monitored continually they should at least be benchmarked for comparison later.

With business metrics understood you can now derive and define your key IT metrics.

Now you know what to measure, you can start looking at the how.

Let's take a simple example.

The business has told you that customer satisfaction is their biggest problem at the moment. Your first job is to work out why. If it's due to the slow addition of new features then you will naturally look to increase your development velocity. If, on the other hand, you discover that it satisfaction is in fact low due to poor product quality then your approach would be different. You main focus would be on the number of defects being released, and you would measure this accordingly. Now, the frequency of changes may be a lever you pull to address this, but it would not be the focus of your measurement. You would focus on the number of defects (or MTBF) and its flow on effect on overall customer satisfaction. In some cases reducing the frequency of change may actually have a positive effect.

With a DevOps initiative, as with any IT project, knowing what to measure is always the first step. Starting with an understanding of what the business needs is key. Use that understanding to prioritize the areas you will focus on from a measurement and monitoring per-spective. Beware of vanity metrics. Your business sponsor won't be at all impressed to hear that you've doubled your development velocity if the metrics she cares about are flatlining, or, worse still, getting worse. ■