Everything you need to know about PLM in the Cloud

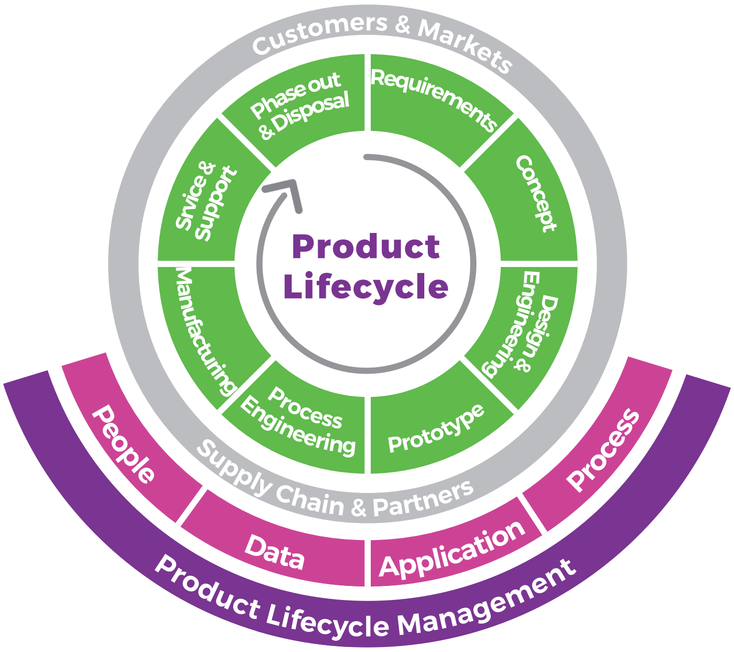
Table of contents

Product Lifecycle Management (PLM) in the Cloud

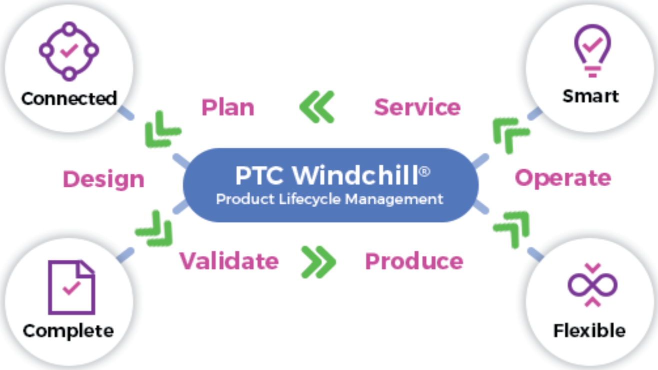
With Cloud technologies developing rapidly in recent years, so came the thought of using the Cloud for hosting various systems in it, from computationally demanding, through mission critical and more. Product Lifecycle Management (PLM) systems were no different and a large number of both PLM vendors and customers attempted to leverage benefits the Cloud enables to gain that competitive edge world's best manufacturers always strive for. Does this approach make sense and what benefits (and drawbacks) does it have over traditional on-premise hosting? What are the differences between various XaaS ("*Something*"-as-a-Service) models for Cloud PLM solutions? How can we make sure that a Cloud PLM system is efficient both cost- and performance-wise?

What are Product Lifecycle Management (PLM) systems?

Product Lifecycle Management (PLM) systems control the life cycle of a product and virtually all data regarding it from the very first idea ("Let's build a new car/washing machine/wardrobe/airplane!") through releasing the final design to manufacturing and even beyond. These systems are the backbone of mostly any product design and manufacturing company these days. Created as a natural successor of Product Data Management (PDM) systems, which aimed at storing and managing CAD data, they introduced so much more: BOM management, collaboration tools, workflows and engineering change management, and even compliance tracking and supply chain management.



This allowed drastically decreasing time-to-market for new and updated products and significantly streamlined information flow within OEMs – thus greatly improving collaboration, even among globally distributed teams.



PTC Windchill is one of world's most popular – and best – PLM systems.

You can find out more about it and the value it creates for organizations here:

LEARN MORE

In most cases, we are talking about huge enterprise-class IT systems, first created in the 1990s and gradually built upon an enhanced through the last 20+ years. As you can imagine, all that development has made them extremely powerful – and at least as extremely complex. Running a PLM system in an enterprise with thousands of engineers feeding it daily with fresh CAD data, which then undergoes acceptance processes and later publication to, for example, an ERP system, will require some capable infrastructure, including dedicated machines for the database, publishing engine (workers), content (vaults) and the application itself. Add to that the need to make, store (and restore) backups on a daily basis and you'll see that hosting and maintaining a PLM system can be quite a challenge even for skilled IT departments – especially if they are not very familiar with the system itself.

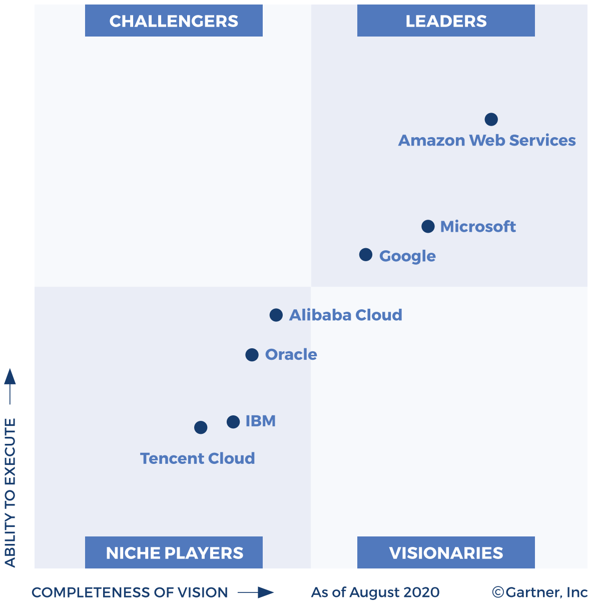
Automation of various tasks and activities (these are IT systems, so by definition they are susceptible to automation) is an obvious way to eliminate at least some of the burden of managing a PLM system. But what can we do about scalability: the need to serve more end-users, or spin up new environments for developers working on new versions of our systems?

Enter the Cloud.

The Public Cloud

The Cloud, or more specifically, Public Cloud, is a service provided by (usually) technology giants such as Amazon, Microsoft, Google, Alibaba, Oracle and the likes, which is basically providing computational, data storage, networking and other IT resources to customers for a periodic (monthly) fee or in a pay-as-you-go model which are usually a fraction of the cost of setting up an entire equivalent infrastructure. Instead of paying for server rooms and equipping them with high-end computers, switches and the likes, you can simply rent the computing power you need to host your website, database, PLM system or even a cat-video sharing portal.

Cloud hosting takes the burden of purchasing and managing a physical infrastructure off the customer/user. Cloud vendors take care of purchasing servers, enhancing computing and storage capabilities, network management, and so on, across (usually) dozens or hundreds of locations around the globe. Furthermore, they provide many tools which can make our life with their solution easier, ranging from complete website templates for retail businesses, through solutions allowing code to run without defining a particular machine to run it on (so-called *serverless*, more on that later) and tools enabling automation of most complex (or mundane) tasks. At the same time, it provides nearly unmatched reliability, security and stability.



Magic Quadrant for Cloud Infrastructure and Platform Services – Amazon, Microsoft and Google are clear leaders of the Public Cloud market. Others are so far lagging in functionality, robustness and offering.

Source: ***https://www.gartner.com/doc/reprints?id=1-1ZDZDMTF&ct=200703&st=sb***

Makers of various PLM systems thought the same thing as you probably are now: "*The Cloud seems like a magnificent asset to put our solutions on! It will run faster, scale almost indefinitely, be easier to manage, and so on...*"

And so, they did just that.

What is Cloud PLM?

Truth be told, Cloud PLM is generally no different (functionally) from the PLM you may know well. It provides the same BOM management, workflow and Change Management, CAD data management and other capabilities we are used to from a full-blown, mature PLM system. Most likely it even has an identical User Interface as the on-premise version, so for many end users the difference will be negligible, if at all noticeable.

The main difference is in the underlying infrastructure, which also introduces new system management strategies and models.

What are the types of Cloud PLM systems?

There are two types of "Cloud PLM" systems out there:

Traditional PLM systems which you know well, simply hosted differently. Examples include:

- PTC Windchill
- Siemens Teamcenter
- Dassault Enovia
- Oracle Agile

Cloud-only PLM systems, which are not available for On-Premises installations. Examples include:

- GrabCAD
- Autodesk PLM 360
- Oracle PLM Cloud
- Upchain Cloud PLM

Before we get to details, we need to understand that "Cloud" is quite a generic term which can mean one of at least 4 approaches: Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS), Function-as-a-Service (FaaS) or Software-as-a-service (SaaS). These models, although all cloud-focused, vary significantly from one another and provide you with different benefits – and drawbacks. Let's dig in.

Differences between Cloud-based and on-site solutions

In a traditional On-Premises setting it is an organization's responsibility to set up basically everything: First, you must prepare a server room (which can be quite a sophisticated – and expensive – place, with specific air conditioning, fire safety, power backup and other requirements). Then, you put your physical servers and other equipment (storage, networking, etc.) into that server room and connect them to the network. Once that's done, you must install and configure all pre-required software, such as Operating Systems (OS), IT management tools and so on. Most organizations also introduce some form of virtualization, so they don't have to procure another physical computer for the server room. That also needs to be installed and configured prior to usage. When that's done, you can usually start setting up your PLM system, which includes installing a database, any third-party software required (for example: Java runtime environment) and then – the PLM application itself.

But if you think your work is done there, you're mistaken. Gravely. That's where the "fun" truly begins.

I won't cover all topics here, because that could make up an entire book. I will only give you a couple of examples to give you an understanding of the effort required to make sure your On-Premises PLM system is running smoothly.

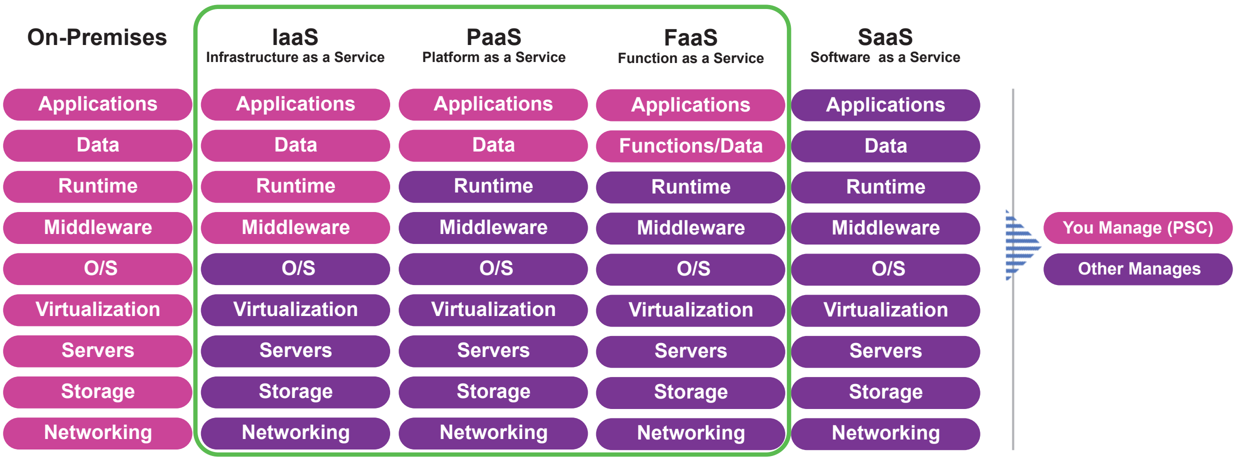
Some of the things you need to do more-or-less frequently, are:

- Configure backup strategies, run backups and ensure these backups actually happen – and contain what they are supposed to contain.
- Manage and maintain the database. If you don't, performance will take a hit (which is still the best-case scenario).
- Manage and maintain the PLM system, including:
  - Daily, weekly and onthly ***maintenance activities***, which we will touch on later. This page describes Cloud administration and maintenance activities for PTC systems offered by TTPSC.
  - Apply patches, critical updates, version updates, etc.
  - Upgrade the system to newer versions as yours will become outdated and lose support. You can find a more detailed description of what upgrades entail ***here***, though please be mindful this is a rather simple case.
  - Create/develop and maintain any configuration or customization specific to your organization's needs and your PLM instance. Read ***this*** to understand what configuration and customization of PLM are and what are the consequences of using them.

Had enough? That's only scratching the surface.

Flavors of Cloud

The Cloud takes some of the pain above off you, but to a varying degree. There are multiple "flavors" of Cloud solutions and I think the best way to describe them is to present it using a graph (which you may be familiar with, as it's one quite commonly used to show differences between On-Prem, IaaS, PaaS, SaaS and FaaS):



On-Premises	IaaS Infrastructure as a Service	PaaS Platform as a Service	FaaS Function as a Service	SaaS Software as a Service
Applications	Applications	Applications	Applications	Applications
Data	Data	Data	Functions/Data	Data
Runtime	Runtime	Runtime	Runtime	Runtime
Middleware	Middleware	Middleware	Middleware	Middleware
O/S	O/S	O/S	O/S	O/S
Virtualization	Virtualization	Virtualization	Virtualization	Virtualization
Servers	Servers	Servers	Servers	Servers
Storage	Storage	Storage	Storage	Storage
Networking	Networking	Networking	Networking	Networking

As you go from left to right you will notice that more and more of responsibility falls on the vendor, offloading it from your organization's IT department.

It seems that SaaS is clearly the go-to approach (who wants to pay those pesky IT guys anyway?), but it is necessary to consider all benefits and drawbacks of each solution before making the final decision. You may find it surprising (or not) but moving between different models is not as simple as it looks.

Cloud PLM SaaS benefits and drawbacks

Software-as-a-Service (SaaS) means you pay a monthly subscription fee for access to software hosted and managed by someone else, usually the vendor/creator. This fee will likely depend mostly on the number of users you wish to have access to the system.

With SaaS you don't care what needs to be done to make sure the instance is running. You don't have to configure backup strategies, apply patches and so on. You simply pay and start using the system.

Sounds awesome, right? However, this approach also has its drawbacks.

1. You can only (and not even always) configure your system. By default, customization is not allowed (or even possible). Please refer to ***this article*** (also linked above) to understand what this entails.

2. SaaS does not allow you to access the infrastructure. You will not have the ability to connect to a Remote Desktop or Shell. In many cases you won't even have full administrative access to the application itself.

3. The roadmap of the product depends on the provider. You do not decide on system updates, security patches, etc. The vendor applies them according to his schedule. You will not be able to decide to skip an update, immediately deploy a patch once it becomes available, etc.

4. Since you're not managing databases or file vaults, it may be quite difficult to access and extract your data if you ever decide to switch to a different solution (different PLM system), hosting model (for example: from SaaS to IaaS) or Cloud vendor (for example: AWS to Azure or vice versa). It's not in the vendor's best interest to let a customer leave easily, so you can expect them not to be too helpful during that process and you may end up having to run a full-scale data migration project.

Cloud PLM PaaS benefits and drawbacks

Platform-as-a-Service (PaaS) is somewhat a bridge between SaaS and IaaS. This model allows delivery of complete development and deployment environments in the Cloud, with resources that enable delivering anything from simple Cloud-based apps to advanced enterprise applications. In the PaaS model, developers can rent everything they need to build an application. A PaaS vendor provides hardware and software on its own infrastructure. This approach frees developers from having to install On-Premises hardware and software to develop or run a new application. PaaS significantly simplifies web application development as, from a developer's perspective, all backend management takes place behind the scenes. PaaS also includes servers, storage, and networking as well as middleware, development tools, database management systems, and more. PaaS is designed to support complete web application lifecycles: building, testing, deploying, managing, and updating.

Various vendors include other services in a PaaS model, but the basics are:

- Development tools
- Middleware
- Operating systems
- Database management
- Infrastructure

An example might help better understand PaaS (and differentiate it from other "flavors"): Imagine you wish to build a web application (or an entire web system) and you wish to do it using JavaScript and additional frameworks like Node.js. In a PaaS model, the vendor will provide you with the entire environment, so your development team may start coding right away. In the background (invisible to you) these vendors will usually utilize services of large vendors (AWS, Azure, GCP, etc.) to host and run whatever they provide to you.

PaaS has an edge when building an application from scratch. You can use one environment for development, through tests and even production deployment. It allows you to forget about dependencies and such. Simply build your application on top of a well-designed system. On the other hand, PaaS doesn't play nice if you need to deploy and maintain a "ready-to-use" application which has specific requirements. You'd be often forced to make modifications to the code of that application to be able to run it in an environment provided by a PaaS vendor. In some cases, it may even be impossible to run an application in such a closed and strictly specified environment.

Alas, if you wish to build a new system as fast as possible – PaaS might be the best choice for you. However, if we are talking about hosting a mature PLM system, only certain parts of PaaS make sense – more on that later.

Examples of services offered in a PaaS model include: AWS Elastic Beanstalk, Heroku, Google App Engine, Apache Stratos which are runtime environments for many popular technologies like Python, Node.js, Java, .NET or PHP. Databases are also offered in a similar model, for example a NoSQL database which can run using the MongoDB Atlas service. In case of "ordinary" relational databases, Microsoft Azure SQL Database is a traditional SQL database offered as a PaaS product.

The benefits of utilizing a PaaS model include:

- **Faster time to market** – PaaS allows building web applications more quickly, as developers don't have to worry about servers, configurations or deployment. All they need to do is write code.
- **Price** – PaaS may be more cost-efficient comparing to IaaS. Many vendors offer pay-as-you-go pricing models, where clients are charged only for computing resources used by the application.
- **Licensing** – PaaS vendor provides all necessary licenses for OS, frameworks and development tools.

There are a couple of drawbacks as well:

- **Vendor lock-in and vendor dependency** - It could be difficult to switch PaaS vendors, because the application is built using a given vendor's toolset and specific mechanisms for their platform. Other PaaS providers might not support the same programming languages, libraries, APIs, architectures used to build the application, and so on. Thus, sometimes changing PaaS vendors results in the need to rebuild the entire application.
- **Compliance** - In a PaaS model, a third-party company will store all data an application generates. From time to time the vendor may persist databases via yet another third-party, usually an IaaS provider. This situation could make it difficult to fully assess and test security mechanisms protecting application's data. Companies which must comply with strict data security regulations (e.g. banks, insurers and healthcare entities among others) may find verifying the compliance of additional external vendors an issue – or even potentially a roadblock.

Cloud PLM IaaS benefits and drawbacks

Infrastructure-as-a-Service (IaaS) should be the easiest one to understand from an IT perspective. In this case we are not dealing with applications automatically updating themselves, or backups doing themselves, etc. IaaS is simply about replacing your physical infrastructure (your server room) with Cloud resources. In other words, there are still servers running virtual machines which you can install your systems on, but you no longer care about managing those servers and virtual machines. You let your Cloud vendor do that for you and simply bill you the cost as you go.

While that doesn't seem much like an improvement over On-Premises, it does have a few key benefits.

First of all, IaaS eliminates the need for up-front investment into the physical infrastructure: buildings, rooms, equipment, etc. It also eliminates the need for maintaining that infrastructure, so if a disk burns out in one of Amazon's (or Microsoft's, or Google's, or …) data centers, they have to replace it to make sure they are able to continue delivering services to you. You don't care.

Scalability is another important aspect. With traditional (physical) infrastructure, you **will** run out of processing power, storage space, network throughput. That may mean you have no space to store new designs, or your number of users will exceed what the machinery can reasonably handle. Or, you may need to do some very demanding computing (for example: simulation) which will cook your processors after the first 10 hours out of 10 years of estimated execution time. No matter how future-proof your infrastructure will be, eventually time will catch up with it and speed past it like a blazing rocket.

NOTE: Every Cloud model eliminates that issue, not just IaaS. Using Public Cloud you're not bound to any specific hardware resting (or working) in your server room, but rather to the vast capabilities offered by large data centers located worldwide, providing as much computing power, storage, etc. as you can imagine.

Obviously, drawbacks are present too.

IaaS can be understood (in a simplified way) as "offloading the need to buy and set up physical machines," but otherwise these machines act like "real ones." This means they will require nearly the same level of attention and software maintenance you would expect from an On-Premises system. It's a little bit like driving a Ferrari through a town with a strictly enforced 20mph speed limit. Luckily, a lot of these things can be significantly automated, but more on that later.

Function as a Service (FaaS, Serverless)

Function as a Service (FaaS) or "serverless", seems on the other end of the spectrum to IaaS.

FaaS is a type of cloud computing where you don't handle provisioning resources for the back-end code to run, but simply access these services on an as-needed basis. The cloud provider is responsible for starting and stopping a container platform the service needs to run on. FaaS eliminates the complexity of building and maintaining the infrastructure – which was until recently a requirement for developing and launching any application.

To better understand what that means, think this: "*I need to run some simple code somewhere, and I don't want to care about the operating system, CPU and all that boring stuff.*" This is where FaaS (in AWS: Lambda, in Azure: Functions) comes in: you write some code and tell it to execute. That's it. No worrying about anything else.

These pieces of code in the serverless module are supposed to be super-simple. They are not to be considered as entire applications, but rather atomic actions performed within it. Grouped together they can form something called "microservices" (modular components of an applications, which are independent of each other). These, in turn can be further bundled into entire applications.

Furthermore, such serverless apps are usually super-cheap to run, as you don't have to maintain an entire infrastructure 24/7. You call them when you need them, and then they're gone.

Is it an effective way to build a PLM system?

No. There would be too much effort involved into building something like that from scratch. Way too much. Maybe a simple app, but not a PLM system.

What is serverless good for in PLM, then?

Executing simple tasks, often related to managing your cloud infrastructure, for example: creating backups, spinning up new environments from templates, executing health checks, etc. See where I'm going? Simple, automatable administrative tasks.

Mix and match

Luckily, most of the services offered by cloud vendors can be used in a "fragmented" way, meaning you can use some of them in an IaaS model, while others in a PaaS model, adding FaaS wherever it makes sense.

As an example, let's dissect what we did for one of customers from the Automotive sector:

> **We used IaaS to define and run the entire infrastructure for PLM:**

- Virtual network and routing
- Virtual machines (which host the application and all its components)
- Databases (although these could be run in a PaaS model as well).

> **We used FaaS to automate "simple", repeatable administrative actions to reduce costs, for example:**

- Creating and reverting backups
- Creating new environments from backups and templates, deleting unneeded ones.

Note: There are many serverless functions executed for each of these, and they are executed in groups, for example: First group of functions creates a copy of the current Production instance; second deletes current QA instance; third copies the production backup to a different availability zone; fourth recreates a new instance from that copy. In addition, several validations and checks are executed as functions along the way.

> **We used PaaS especially for:**

- Our DevOps process, which includes build creation and deployment automation, and is running using Jenkins through Docker containers (AWS Fargate / Azure Container Instances / Google Cloud Run)
- PingFederate, which serves as a proxy for the Administrative Domain controller and enables Single Sign-on (SSO) for users.

This approach allowed us to **minimize human effort** for administration **and costs** (through automating and utilizing serverless functions wherever possible), and introduce **proper System Change Management** procedures to ensure high solution quality (build automation and validation on every step), giving our customer a near-SaaS experience. At the same time, we maintained full control of all infrastructure elements, which SaaS does not allow. How did we achieve this?

Cloud PLM Automation

The beauty of the Cloud is that it was basically designed with automation and DevOps in mind. Not getting into too much details about that (this could be yet another book), PaaS and even IaaS instances of many systems – Windchill PLM included – can be automated to a degree which makes them nearly as easy to manage as SaaS (i.e. almost no management at all!), while providing full control of all environments (which SaaS by definition cannot provide). Let's look at an example:

You start by defining generic templates for a Windchill system: a so-called "golden image" of the system with default settings, proper vault, DB configuration, OS hardening, network, etc. – anything you might require from every single environment you're running, whether it's development, test, QA or production. Once an environment like this is ready, you can create a snapshot of it, which will be the baseline for creating new instances.

For Windchill, PTC Rehost Utility can be used to (as the name suggests) rehost a new image created from the template we have just created, ensuring the system can launch without errors. As you can imagine, the operations of PTC Rehost Utility can also be automated.

Having these two, you can create multiple environments (stages: dev, QA, etc.) using the same baseline image serving as a template, and then automatically rehosted to match network configuration, server names, etc.

If you enhance the above with a few setup scripts (bash/PowerShell – depending on the OS running the virtual machine), you can even diverse from one golden image, creating multiple different configurations, builds and so on, depending on needs. At the same time, you are still maintaining consistency between instances, significantly improving manageability of even a large number of instances. This approach of describing architecture using scripts is called "Infrastructure-as-a-Code" (IaaC). Yes, I know it's not really proper English, but it follows the "XaaY" naming scheme, right?

The table below depicts services and technologies used for this part in Amazon Web Services and Microsoft Azure.

Task	AWS	Azure
Describe infrastructure to make it easily reproducible	**CloudFormation**	**ARM Templates**
Computing / database	**VM/EC2 in IaaS and DB in PaaS**	
Monitor infrastructure	**CloudWatch**	**Azure Monitor**
Storage (logs, baselines, snapshots, etc.)	**S3**	**Azure Storage**
DevOps automation and tooling (Container Services)	**ECS**	**AKS**
Serverless Solutions	**Lambda**	**Azure functions**

There are other options out there, as well, so don't limit yourself to the above. As Cloud Services are constantly being developed and new features are added, it's good to explore them to optimize what you are using, even after your system went live.

You may claim setting all this is quite a bit of effort, but after doing that spinning up a new Windchill instance with full configuration and the latest build deployed takes 15-30 minutes. Compare that to a traditional (manual) process, which could take days (or weeks in worst cases) and the benefit becomes clear and undeniable.

Next, let's look at backups and Disaster Recovery (DR) because... let's face it: **Every. System. Will. Fail.** It is not a matter of "if", but rather "when." Your best bet, then, is to be prepared in advance to mitigate any negative results of such a failure.

First, public Cloud offers tools for automating system health checks to make sure your instances are up and running – and responsive. If a system starts experiencing issues, you get notified and can always replace it with a fresh instance (if you feel like it – you can always try to debug and fix whatever the problem is).

Secondly, you can apply third-party tools and services, such as DataDog (and/or PTC System Monitor for Windchill), which can provide you with a very thorough overview of your system's condition – up to details such as queue usage, cache statistics, memory usage, Java exceptions and other metrics that may help improving performance. Such tools can help you predict potential problems which may occur in the future and allow you to react upfront, preventing any disaster from happening in the first place, or diagnose problems with custom code thanks to aggregation of exceptions and flow analysis.

Furthermore, it's relatively easy to establish a centralized log with backup and versioning. Such a log will show every action performed around your infrastructure. This is important not only from a maintenance perspective, but even more so when it comes to security. You should always know (or be able to check) who did what and when with your infrastructure.

Lastly, if you did your homework and implemented the system creation automation which I described a few paragraphs earlier, your whole infrastructure is described as code. This makes backups simple to automate and you can easily and efficiently test Disaster Recovery scenarios (yes, you should have those too), thanks to your infrastructure being very easily reproducible. Then, even if a meteor strikes the server farm which physically hosts your system's instances, you can quickly spin them up again in a completely different region, thus mitigating any negative effects such a meteor strike would have on your organization. Try doing that with On-Premises...

Reasons to move your PLM to the cloud

You may think, "I already have a PLM system. Its running on-premise and my organization is doing just fine the way it is. I don't need this."

Why would you at all consider the cloud for hosting your PLM system? A few example reasons:

3x time faster recovery in case of hardware failure when compared to on-premise (minutes instead of hours or even days)

Up to 90% faster rehost procedure thanks to automation and DevOps processes (hours instead of days)

Up to 80% less downtime during deployments

Up to 200% release frequency improvement thanks to more optimized and stable processes

Eliminate cost of expensive Cloud/DevOps specialists in your internal IT teams

CapEx to Opex model (simple monthly subscription model)

If above reasons aren't convincing enough, I suggest setting up an overview of your environments with a Cloud and PLM expert. Such an overview will provide you with numbers personalized to your systems and will be as close to achievable reality as possible. ***TTPSC's Cloud Team*** can do such an overview for you – for free!

Cloud is already mainstream

76%

have used PLM for 10+ years

95%

will purchase PLM Cloud in next 24 months, 72% in next 13 months

68%

have a defined IT roadmap for PLM Cloud deployment

Companies are achieving substantial benefits

1,7x

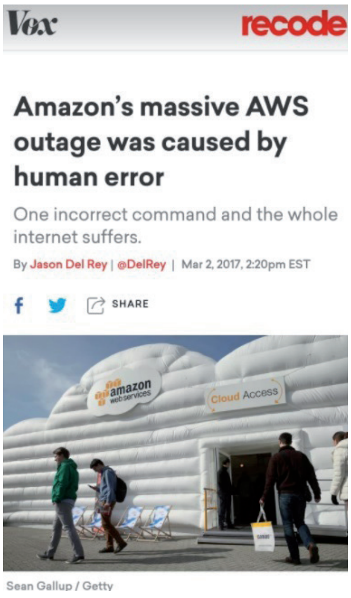
amount of ROI cloud-based SaaS delivers vs on-premise applications

Source: *CIMdata, https://www.cimdata.com/en/resources/complimentary-reports-research/white-papers*

Cloud PLM Security

Let's face it: there are very few companies who have as big IT budgets (entirety) as cloud giants spend on the topic of security alone.

Those cloud giants are high profile targets, very attractive to anyone trying to do some bad things, yet we don't hear too often about their systems being hacked and the entire infrastructure failing because of that. The last time that happened to Amazon it was actually caused by a human error (of an Amazon employee, nonetheless). Please note that, as of the date of me writing this article, that was almost 3.5 years ago, and we haven't heard of anything similar since. What's more, there are actually news of successfully thwarted attacks, such as the one mentioned below, which I think may be over the heads of most in-company IT departments.



Vox recode

Amazon's massive AWS outage was caused by human error

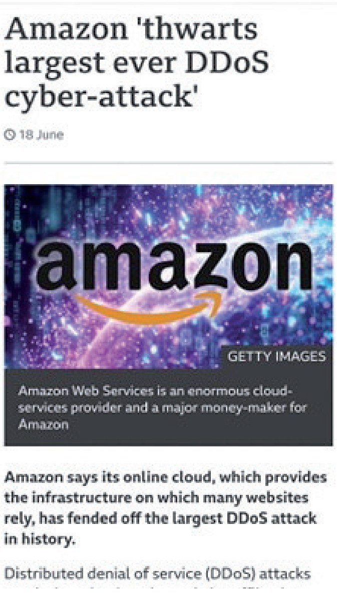
One incorrect command and the whole internet suffers.

By Jason Del Rey | @DelRey | Mar 2, 2017, 2:20pm EST

SHARE

Sean Gallup / Getty

In 2017 a big outage hit AVS' services. It turned out it was caused by a simple mistake made by one of Amazon's own employees link as reported by on March 2nd 2018



Amazon 'thwarts largest ever DDoS cyber-attack'

18 June

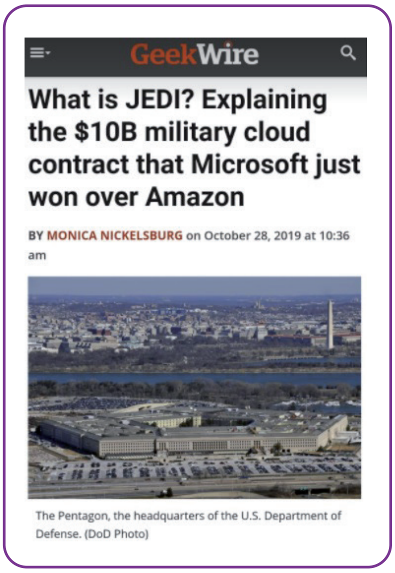
GETTY IMAGES

Amazon Web Services is an enormous cloud-services provider and a major money-maker for Amazon

Amazon says its online cloud, which provides the infrastructure on which many websites rely, has fended off the largest DDoS attack in history.

Distributed denial of service (DDoS) attacks

In June Amzon reported it thwarted the largest DDoS (Distributed danial of service) attack ever recorded – as large as almost half of traffic the entire British Telecom sees on its entire UK network during a normal working day. As reported by BBC on June 18th, 2020.



GeekWire

What is JEDI? Explaining the $10B military cloud contract that Microsoft just won over Amazon

BY MONICA NICKELSBURG on October 28, 2019 at 10:36 am

The Pentagon, the headquarters of the U.S. Department of Defense. (DoD Photo)

I sometimes employ a Jedi example here. No, not the folks from the Galaxy Far Far Away, waving their "laser swords" and shooting lightnings from their fingers. I'm talking about the US Department of Defense JEDI program.

In short, the US DoD will make a huge investment into making the public cloud its de facto main IT resource. If they feel fine keeping their data in the public cloud, why are you worried? Especially since that same level of security is available to anyone – obviously it's simply more expensive than what most of us mere mortals need.

What's more, Windchill – even a public-cloud-hosted instance – can successfully leverage Single Sign-On (SSO) with Active Directory at very little additional cost, as Tomasz Breś explained in ***one of his articles***. This makes using the system easier and more secure at the same time.

If you want to learn more on the topics of Cloud security, you can start by reading ***this piece*** by Kamil Ulezło on security of AWS serverless applications (so-called Lambdas). **Warning:** this article requires technical knowledge to fully understand and appreciate.

The Single Source of Truth - Collaboration

I provided a simplified introduction into what PLM is at the beginning of this article, so even if you didn't previously, by now you surely do understand how important PLM is to organizations developing, maintaining and selling products. If you do need a refresher, though, please refer to this page.

What I would like to focus on here is the fact that this Single Source of Truth, once established within your organization, becomes vital to its operations. If you lose it (for whatever amount of time and whatever reason) many departments will feel the disruption such failure causes in their operations, and thus generate avoidable cost. With so many companies dispersed globally it was a requirement for many years now to make sure such kind of critical infrastructure is always operating at peak efficiency and is available from companies' locations around the globe. The current work-from-home situation pushed this need to the next level, while at the same time straining corporate networks to the point of breaking.

Cloud computing (in case of PaaS and IaaS – supplemented with a heavy dose of automation) ensures that one of your most important systems is available 24/7/365, enabling your global organization to work whenever (and wherever) they choose.

Let’s also consider another example: Imagine trying to download a large, complex CAD file from a server in China if you are located in Germany (or the other way around). In the scenario of an on-premises installation, a PLM system, which is responsible for storing and providing access to large amounts of data, usually utilizes data silos and replication mechanisms. This means that data is first accessed from the nearest server to a user, ensuring as low latency and as high bandwidth as possible. When that user finishes his work and checks the updated file in, it gets uploaded to the same server. To make that file **efficiently** available to users on the other side of the globe, this file needs to be replicated to a physical site much closer to its destination. Otherwise you are facing severe waiting times when trying to download or upload anything, no matter how fast your Internet connection is. Data replication for on-premises PLM systems is usually scheduled “overnight”, which means there is a possibility that teams in different parts of the world might not necessarily access the latest information – because it has not yet been synchronized between corresponding data silos.

Cloud infrastructure simplifies that. Firstly, it allows you to automate data replication across multiple sites (and these sites are located worldwide, so there is always one relatively close to your desired location) without the need to dive deep into PLM vendors’ tools – in many cases you can simply “tell” a cloud service it is to replicate itself to a different region with a given frequency. Since you are no longer limited to your own IT infrastructure, this replication is also much faster than the traditional approach, making your data available to globally distributed teams much faster, thus minimizing the risk of accessing outdated information.

Data Protection (incl. backup & DR)

There are two types of people in this world: those who do backups, and those who will do backups.

Traditionally, IT had to take care of backups: define strategies, write scripts, ensure backups actually happened, and then store them. If something went wrong, these backups were “manually” brought up to life, which might have taken anything from minutes to days, depending on the complexity of whatever went down. With the Cloud, once you configure and automate it properly, you’ll get notified if any part of that process goes wrong, so outside of strategy and scripting, you only need to act (do anything, really) if any part of the process doesn’t work at a certain iteration – and you’ll actually get a notification that you might want to look into it. No more worrying about storage space, backup execution… or recovery, which (with so much computing power at our disposal) is significantly faster than copying files between hard drives. Oh, the dream world… except it’s here!

Cost optimization

This part is tricky.

Yes, the up-front cost of a cloud-based solution is significantly lower compared to building and equipping an entire server room. Yes, the cloud in many cases eliminates the need for costly IT systems' maintenance. Yes, if you want to spin up a new environment and then trash it after a couple of days, you don't need to buy an extra physical machine.

However, it is important to note that moving on-premise systems "as-is", without properly adjusting the architecture they're built on and somewhat changing the principles of how we look at such resources, the cloud can generate costs which, in the long run, can be twice (or even thrice) as much as on-prem!

No, cloud vendors won't tell you that bit.

There are ways, however, which allow you to save even 40-70% on infrastructure costs, for example:

- **Using spot instances wherever possible / viable.** Spot instances are resources "free at this moment in time" which are dirt-cheap, but don't give you any guarantee that when big demand comes into the system, they won't be "taken away".[1] They are not designed to work as production systems, but rather development or test environments.
- **Periodic reviews of instances and upgrading to latest instances released by Cloud providers** – which, even though usually more powerful, can often be cheaper than older offerings.
- **Limiting reserved instances and databases:** The cost of cloud solutions drops every year/quarter/month, so a year/quarter/month from now you may end up paying less for an on-demand environment than you would for a pre-paid reserved instance back then.
- **Most importantly:** Shutting down unused resources. If development is done Mon-Fri 9am-5pm, why would you need to pay for that environment running on a Saturday evening? Resources should be turned on only when needed and shut down immediately once they are needed no longer. Yet another reason for automation.

Infrastructure scaling

We live in times of a pandemic and even this article can't get away from that. The last couple of months showed us how important a dependable and **scalable** infrastructure can be the difference between make or break of remote work. Having thousands of employees quickly switch to work-from-home meant thousands of new daily connections for many servers, some of which didn't handle the stress, effectively causing downtimes for entire enterprises.

If your company owns a near-limitless amount of resources, you can build such near-limitless scaling capacity yourself. However, it'll cost you dearly. And you'll run out of space for new racks in your server room eventually. And another. And another. Amazon, Microsoft, Google and others – won't. They've got more than enough to sustain any surge in demand. They've been building for that for years now.

[1] This is yet another great reason to build automation into your cloud solution, because if a spot instance gets "taken away", you can quickly and easily spin up a new one, minimizing any disruption to your work.

Cloud PLM Services

Now that we know and understand the value that Public Cloud can bring to enterprises, let's consider what is needed to make that happen.

Traditional approach to IT within many organizations is to have a (relatively) small unit which takes care of all things IT: computers and other hardware, software, network, printers, VOIP (Voice Over IP), etc. These "magicians," often looked upon as Roy Trenneman and Maurice Moss from "The IT Crowd," are often essential to making sure that business can run at all. Usually it falls within their responsibilities to manage and administer an organization's PLM system, which is fine, especially if they have the necessary knowledge regarding that particular application.With the Cloud things start to get a little bit more complicated*. The sheer amount of various technologies and services offered by major vendors has led to a point where it's not really possible for any single person to "know the Cloud." One may rather say that he knows some services offered by a given vendor and has awareness of several others – even that level means this person is highly skilled in this area. Having that in mind – and the ongoing responsibility for maintaining other IT systems and services within an organization – it would be difficult for an internal IT department to fully leverage the potential of the Cloud, whether due to lack of knowledge, skill in that particular area, or simply time.

* Outside of SaaS, where your organization is not tasked with maintenance and administration of the system.

In most cases companies turn to external sources to fill these gaps and provide a complex and comprehensive approach to utilizing Public Cloud. PLM is no exception here, but due to its specifics it requires additional product and process knowledge, as well as industry expertise, which most Cloud service providers lack. Finding a reliable partner who can not only deliver a service, but actually help define a strategic approach and flawlessly execute on it should then be considered a priority for any organization planning to at least consider using Public Cloud.

Approach and Cloud Strategy

Every organization is different. It's as simple as that. Just like there are no one-size-fits-all PLM systems, there is no such way of migrating an organization to the Cloud. Each case (company, department, system) should be thoroughly evaluated and analyzed to find out what would be the best and most effective way of moving it to the Cloud – or whether it should be moved there at all!

A **good** partner, before suggesting anything, should spend time with an organization, learning its working methods, infrastructure, internal processes, and so on. Thorough understanding of an organization is key to being able to find areas of improvement. This is true not only for PLM, but for most other aspects of business as well. Therefore, a good partner will first talk to your IT department, internal customers (end-users of IT systems), management, etc. before suggesting anything. If an organization appears on your doorstep with an "out of the box cloud strategy" – you should show them out.

Once the assessment is complete, a vision of a strategy for using the Cloud should start to emerge. Such a strategy should describe in detail:

- systems (or their elements) to be addressed, i.e. moved to the cloud
- priorities, dependencies, risks and mitigation methods
- high-level architecture of resulting cloud solutions
- action plan and schedule for executing
- clear estimate of cost (note here that the cost of using Public Cloud is hard to predict exactly, but the point here is that your partner should present his estimate to you in a detailed and transparent way, not just "a single number")
- long-term maintenance plan
- working methods for development, deployment and maintenance.

Please note the above is not a complete list. It is rather a "must have minimum" you should be looking for.

Cloud Migration

Once the strategy is in place and a decision is made to execute on it, the migration phase starts. This migration should not be considered as a traditional ETL data migration, as it is focused on migrating not datasets, but entire systems to a Cloud infrastructure. For PLM this usually includes items like:

- **Preparing Cloud infrastructure:** detailed architecture, templates, networking, security, backup and recovery mechanisms, etc.
- **Installing systems and components:** database, file vaults, CAD and other Workers (for publishing), System Change Management components and the PLM application itself.
- **Developing (and/or deploying) configuration and customization.** (This part will depend on how far a given organization needs to go away from OOTB. Obviously, the less – the cheaper and faster. Equivalently, the further – the longer and more expensive. More on that later.)
- **Migrating existing data from On-Premises to Cloud** (which is more-or-less a traditional Data Migration process).
- **Testing and validation:** not to be mistaken with testing performed during development, this phase will focus on verifying integrity of your PLM environment and its dependencies to ensure you have a stable system and data is intact.
- **Go-Live:** once everything is verified, you "flip the switch" to shut down your On-Premises system and use the Cloud-based one.
- **Ongoing support:** even Cloud systems need maintenance and support, and in many cases also development. This must be managed in a proper way.

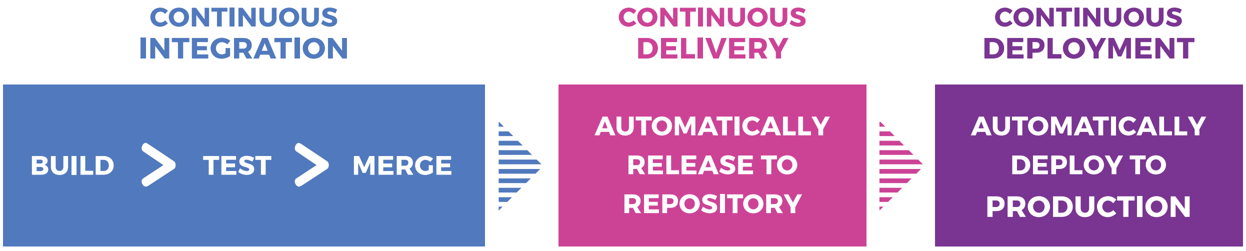
As you can imagine, this is the most complex and time-consuming part of the process, as it is actually executing on the plan established earlier. The more important, then, that the plan is solid and based on an organization's true needs in the first place in order to minimize waste and increase impact.

System Change Management

As your PLM environment matures, which often includes development of new functionalities, it is important to manage this process through System Change Management (SCM) procedures. Appropriate approach to SCM ensures your systems' stability and thus availability to end users through enforcing certain standards and multiple levels of validation of any change before a change makes its way to the production server. Furthermore, proper SCM includes tracking every change made to the solution, so if things do go awry, it is easy to pinpoint what precisely caused the issue and thus drastically decrease the time needed to address and fix it.

In general, this is similar in On-Premises and Cloud-based approaches. The main difference is in the cost which would – or would not – be incurred by an organization due to bugs or server crashes: spinning up new environments from backups, potential loss of some (latest) data, disruption to engineers working on product development, and so on.

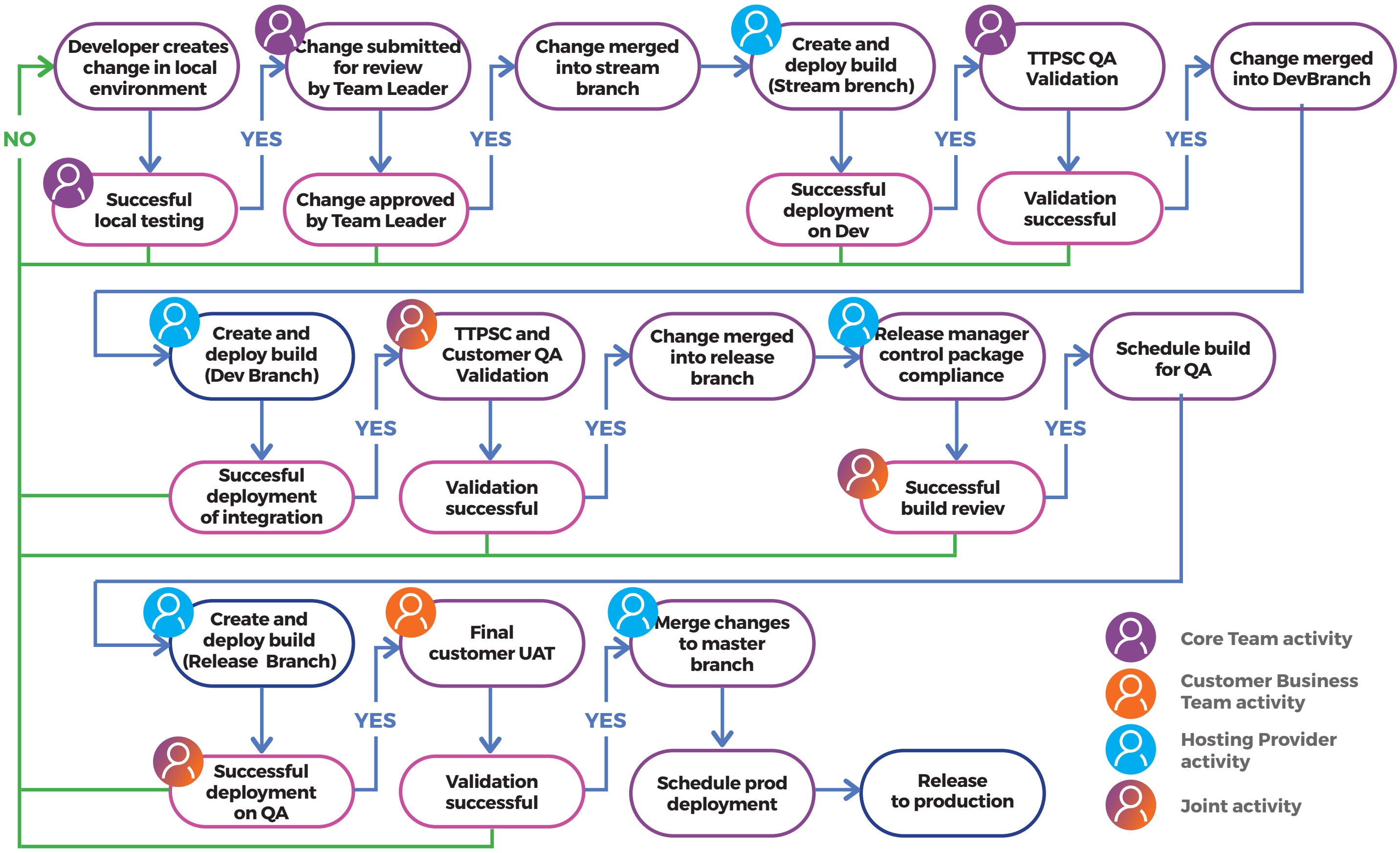
This idea is generally aligned to the Continuous Deployment (CD) philosophy, which builds on Continuous Integration (CI) and Continuous Delivery (another CD). It encourages automation at every step of the process to increase the speed of releasing new features to users to gather feedback as soon as possible while maintaining quality thanks to... yes, automation. CI focuses on submitting new code as often as possible and utilizing automatic testing to verify if it works. Continuous Delivery adds to that further automated testing and merging that code to a repository, which can then be deployed to a production system by the System Operations (SysOps) team. Continuous Deployment then takes that code and actually releases it to a production system in a frequent, regular manner, making it usable by customers (in case of PLM – mostly internal customers).



Applying this process lets you tackle some common problems in software development, ranging from too many branches which may conflict each other, through early detection of bugs, visibility between development and business and overloading teams with manual processes which may slow down progression.

The whole idea of "automating everything" does not mean, however, that manual verification should be left out of the process. On the contrary – it is key at certain stages but should focus on functionalities and business requirements rather than on technicalities, which should be checked without human intervention. Especially with PLM systems, such as Windchill, where not everything can be tested using frameworks such as Junit, Selenium and others.

A successful SCM process for a PLM system uses CD to the fullest extent possible – automating as much as can be automated – and mixes in testing done by users (and developers, and testers) where required to ensure system development is in-line with business requirements. An example SCM procedure we applied at a few accounts which proved quite successful may look like this:



Developer creates change in local environment
Change submitted for review by Team Leader
Change merged into stream branch
Create and deploy build (Stream brench)
TTPSC QA Validation
Change merged into DevBranch
NO
YES
Succesful local testing
Change approved by Team Leader
Successful deployment on Dev
Validation successful
Create and deploy build (Dev Branch)
TTPSC and Customer QA Validation
Change merged into release branch
Release manager control package compliance
Schedule build for QA
Succesful deployment of integration
Validation successful
Successful build reviev
Create and deploy build (Release Branch)
Final customer UAT
Merge changes to master branch
Successful deployment on QA
Validation successful
Schedule prod deployment
Release to production
Core Team activity
Customer Business Team activity
Hosting Provider activity
Joint activity

System Operations (SysOps)

Regardless of how much development your system is undergoing or how complex its architecture is, or even whether it's in the Cloud or on your premises, a PLM system needs to be properly maintained and managed to ensure its performance and availability to an organization.

For Windchill, PTC devised a list of monitoring and maintenance activities which it suggests should be performed regularly to make sure the PLM platform is running smoothly.

TABLE 1 Scheduled monitoring activities

Activity	Frequency / Occurrence
Configure Notification Settings ● JMX Administrators list ● Database Administration ● Windchill Directory Server ● PTC System Monitor or another tool optional)	**Initial Deploment**
Investigate and Respond to Notifications and Alerts ● PTC System Monitor (optional) ● Windchill Log File Viewer ● Windchill Server Status Page ● Oracle OEM or Microsoft SQL Server Management Studio ● Windchill Directory Server Control-Panel	**Daily / As Needed**
Monitor System Health Using Dashboards and Key Performance Indicators ● Windchill Server Status Page ● Windchill Queue Management ● Oracle OEM/ADDM or Microsoft SQL Server Database Tuning Advisor ● Java Tools (JConsole/VisualVM) ● Operating System tools ● PTC System Monitor (optional)	**Daily**
Check Network Health ● Ping ● PTC System Monitor User Experience Monitor (optional) ● Third-party Network Monitors	**Daily**
Benchmark System Response Times ● Multiuser Load Generator ● Single-user Performance Tester ● Windchill Creo Data Management Performance Benchmark Test ● Business Transactions in PTC System Monitor (otional) ● Automated web transactions using PTC System Monitor (optional)	**Weekly**
Generate Throughput and Security Audit Reports ● Report Management ● Usage and License Reporting ● Security Audit Reporting ● Windchill Business Reporting (if needed) ● PTC System Monitor (optional)	**Monthly**

TABLE 2 Scheduled maintenance activities

Activity	Frequency / Occurrence
Backup System ● SQL Server Backup & Log Shipping (RDS snapshot in AWS) ● Windchill Directory Server Backup, Export, & Replication (automated and kept in AWS S3 or SVN or whatever suits customer needs) ● Third-party Backup & Replication utilities (optional if needed) ● Snapshots (AMI & RDS snapshot)	**Daily**
Maintain Database Statistics and Indexes ● Automated database mechanisms	**Daily**
Perform Database Maintenance ● Oracle OEM or Microsoft SQL Server Management Studio	**Weekly**
Apply Maintenance Patches and Updates ● PTC software patches ● Operating system and hardware patches ● Database patches	**Monthly**
Check system configuration using analysis tools ● WinDU ● Windchill Configuration Assistant ● Windchill Client Inspector ● Database analysis tools ● Oracle or Microsoft SQL Server Tuning Report	**Monthly**
Execute Windchill Site Maintenance Utilities ● Vault Configuration - Remove Unreferenced Files ● Participant Administration utility - Check for Disconnected Participants ● Personal Cabinet Administration utility	**Monthly**
Purge obsolete or unused data ● Purge Management utility	**Monthly**
Rehearse Recovery scenarios ● Third-party backup and recovery software	**After establishing system baseline / As needed**
Maintain Multiple Windchill Environments ● Windchill Rehosting Utility, third-party vrtualization software ● Third-party source code management software ● Database export & import tools ● Snapshots	**Monthly**

Activities falling under SysOps vary from monitoring and prevention, all the way through disaster recovery. Luckily, many of them can be managed by various tools, such as PTC System Monitor, Datadog and others. If you add commissioning new instances (for development, recovery after a system outage, or other reasons and purposes) to that, the homework done by ensuring appropriate levels of automation during preparing your first Cloud PLM environment will quickly pay off.

As I described in the section Cloud PLM Automation, many usually mundane tasks occupying your team for days can be limited to minutes, performed mostly using scripts, with only minimal human supervision. So even if your system does go down for whatever reason (like a meteor crashing directly into one of Amazon's, Microsoft's or Google's data centers), users may not even notice anything happened, continuing to work on things that generate value for an organization.

If On-Premise, your IT department usually handles that, as mentioned earlier, and their automation options are somewhat limited. For example, it is impossible (or at least very hard) to describe infrastructure using scripts, so whenever a new machine needs to be spun up, a lot of it has to be done either using snapshots, or manually.

However, managing a Cloud instance may require knowledge and skills outside of the usual around networking, hardware, etc. To be able to do SysOps effectively, you either have to build such expertise in-house, or use a third-party provider, who can offload entire responsibility for server and application management off of you, eliminating the need to employ Cloud specialists and ensuring that your systems are always safe, available and up to date.

Hiring such a service provider may seem like a significant cost at first. However, if you look at the cost of hiring (and maintaining) a good Cloud expert these days (not to mention a Cloud **and** PLM expert), you may find it actually much more cost-effective. After all, the service provider who you'd be engaging with should have a team which shares their time between various customers, thus you'd not be tasked with paying for the entirety of their employment cost (plus service vendor's margin).

Services related to Cloud and PLM management may be as important, if not more, than the choice of the actual Cloud vendor. Even the best Cloud PLM can be extremely expensive, or even limiting an organization, if not managed properly. At the same time, proper services can ensure you the peace of mind known from SaaS while giving you full power (and control) over your entire infrastructure – regardless of what it is all running on. Whether you build capabilities to handle that in-house or turn to a third-party is up to you but bear in mind that it's not always best to operate on yourself – unless you're Leonid Rogozov[2].

[2] Leonid Rogozov, a Russian surgeon and the only doctor on the team during an expedition to the Antarctic, fell ill with acute appendicitis and had to operate on himself – or face almost certain death. The surgery was successful and after just two weeks he returned to his normal duties. ***https://ichef.bbci.co.uk/news/624/media/images/82708000/jpg/_82708605_88323765-20b8-4fea-9685-305f40bd9103.jpg***

Debunking myths

By now we managed to address the most important aspects of Cloud PLM, or rather what you need to know before you come to a decision to move your On-Premises instance of a PLM system to the Cloud. It's time, then, to focus on straightening out a few things you might have read on the Internet before you got here.

On the Internet you can find whatever it is you want to be convinced off – or somebody who pays enough wants you to be convinced of. Especially these days. Whether you're a Republican or a Democrat (or an Easter Egg for that matter), whenever you jump on the Internet, you'll find whatever content is **FED** to you – not necessarily what you want to learn.

Same thing for Cloud PLM – there is many false claims out there. Let me straighten some of them out for you:

1. If you search the web for "Cloud PLM" you may end up thinking only Cloud PLM has BOM management, etc. Not true. The software and its functionalities are virtually the same On-Premise and in the cloud. It's the underlying infrastructure that is different. Actually, some Cloud-only PLM platforms offer limited functionality compared to "traditional" PLM systems.

2. Some claim that On-Premise PLM is usually over-customized and thus impossible to upgrade, and thus SaaS is the only sensible way forward. This is based on a very wrong assumption – or very poor working methods when it comes to configuration and customization of PLM. You can read my take on that here (although chances are that if you got this far you have already read it).

3. That same article busts another myth PLM vendors would like you to believe: that their solutions are a one-size-fits-all, which is a straightforward lie.

4. Pre-built 3rd party system integrations: just as systems are configured and customized, their integrations with other solutions in most cases have to be done so as well. Therefore, it's not like you purchase licenses for a new ERP system and tomorrow it's up and running, humming along while it efficiently processes your company's data. Enterprise systems integrations just don't work that way. Some may, but few and far between.

5. SaaS is easy to manage – true, until the first time you need to make a change to the system which is not straightforwardly supported by the vendor. Then the pain starts, and you wish you had full access to servers the system is running on to simply make that damned small change yourself.

When not to move your PLM to the Cloud?

1 If you want to limit recurring costs (while incurring the in-advance cost, which is usually higher).

2 If legal regulations require you to have **complete** control over all data (i.e. to a level when you can say “*data for project A is in this server room, this rack, this hard drive*”). In some cases, this may not be an issue. For example, if a requirement states that all data must be physically stored on servers located in the United States of America, you’re fine, as all major Cloud vendors have data centers here. Similar with Germany, France and a few other countries. Heck, Microsoft even has some underwater if you need that. However, if that requirement asked you to have all data stored inside the territory of, say, Madagascar, it may get tricky.

Microsoft pulling one of its submerged data centers out of the water after 2 years of operating on the seabed.

Source: *Microsoft.com*

3 If you need to keep your data completely off the grid/network. I get it, you may be designing a hyperspace engine and you don’t want anyone else in the world to find out – even your suppliers, which you don’t have any, so secretive you are.

Basically, that’s it. If none of the above sounds familiar, you should at least consider moving your PLM to the Cloud. That doesn’t mean you have to necessarily end up migrating to the Cloud. You should, however, run an assessment, estimate costs and evaluate all pros and cons. An informed decision in business is always better than a blind one.

Summary

There are many definitions, technologies and services hidden under the common name of "Cloud." With the explosive growth of both demand and supply in the last few years (and especially in the last few months) it may be easy to get lost in all those fancy names and abbreviations. There is no denying that the Public Cloud, with all its flavors, brings enormous value to various organizations worldwide, regardless of their – or their products' – level of maturity. Even if you are not aware, you are using it daily, as most major business-to-customer services rely heavily on the Cloud.

In the case of a mature PLM system for an enterprise the, though, Cloud can be more complicated. Out of all those technologies out there some make more sense – while others not so much. Whether you opt-in for a SaaS model, where you agree to limited control and customization capabilities for the peace of mind it brings, go full-IaaS with adding automation by yourself, or have someone who knows what they're doing do it for you, in most cases a move to the Cloud may make sense. Not many organizations have the resources and capacity to ensure the levels of data protection, security, scalability and nearly unlimited storage and computing power to rival the Big Tech out there.

However, "may" is not "will," "in most cases" is not "always" and the way you proceed is equally important to whether you proceed in the first place. Before you come to a decision, I strongly advise to perform a thorough evaluation of your existing infrastructure and all related consequences: timelines, cost, resources and expertise among the most important ones. There are many ways of improving your system's performance, stability of cost-effectiveness which you might not even be aware of. If you are not confident enough to do all that in-house, find a partner who can help: figure out what would be the best approach for your organization and guide you along the entire journey or, if you choose to, take the heavy-lifting off your shoulders entirely.

About the Author

Pawel Pacewicz, M.Sc., started his career holding various IT and software development roles.

He joined TTPSC in 2011 and through the next few years worked on small- and large-scale PLM projects for customers in Korea, Japan, Germany, Austria, USA, holding roles ranging from consultant through technical team leader and technical architect. In 2014 he took a management role and helped start one of TTPSC's offices in Kielce, Poland, which went to become one of the leading R&D and PLM units in the organization. He also participated in launching several research projects around Internet of Things (IoT) and Augmented Reality (AR) technologies. In 2017 Pawel moved to the United States to take on a business development role. He is currently responsible for establishing and managing relationships with customers and partners in North America.

Pawel's areas of expertise include **software development, PLM, IoT and AR**. He uses this expertise and a strong problem-solving drive to help organizations transform their businesses, gain competitive advantage and achieve new levels of efficiency through implementing the concepts of Industry 4.0.

Pawel is a seasoned public speaker, taking the stage at events as large as LiveWorx. He is an advocate of sharing knowledge with others, having led extracurricular high-school programming classes and multiple PLM-related trainings.

Do you need a trusted partner?

EXPERIENCE YOU CAN TRUST!

Our experts are creating advanced technologies on a daily basis, which allows them react faster and better to technology and market changes. We are constantly cooperating with some of the world's biggest brands, supporting them in PLM, IoT, AR, Cloud solutions, or providing IT and software development outsourcing services.

500+	professionals in Transition Technologies PSC
2000+	professionals in the Transition Technologies capital group
3000+	professionals available in the TT PSC partner network

LOCATIONS CLOSE TO YOU

we operate on three continents: Europe (Poland, Scandinavia, France, Germany, Ukraine), North America and Asia (Taiwan).

CUSTOMER-ORIENTATION

our experience of working with many customers taught us how to focus on your needs. You can be certain that our approach always keeps your best interest in mind.

AGILITY

we are very flexible in adopting to complex environments and fast pace. We know that every organization is different and understanding which initiatives may be useful will depend on where you are in your integration journey.

LOCATIONS CLOSE TO YOU

high-performance teams are essential to deliver highest quality on-time and on-budget. Our specialized, dedicated teams provide efficient support thanks to extensive knowledge of business processes, software tools and industrial standards. Working with us means working with world's best talent.

CONTACT:
https://ttpsc.com/en/product-lifecycle-management/services/plm-cloud

Any quastions? CONTACT US NOW!

UNITED STATES OF AMERICA	**Nate Taylor**	**E:** Nate.Taylor@ttpsc.pl
EUROPE	**Andrzej Nowak**	**E:** Andrzej.Nowak@ttpsc.pl

If you want to know more please check our website:
ttpsc.com or follow our blog: **https://ttpsc.com/en/blog/**
If you are interested in working for TTPSC check our latest offers: **https://kariera.ttpsc.com/**