certiprof®

DEVOPS ESSENTIALS

PROFESSIONAL CERTIFICATION

DEPC® Versión 072023

DevOps Essentials Professional Certificate

DEPC® Versión 072023

¿Quién es Certiprof®?

Certiprof® es una entidad certificadora fundada en los Estados Unidos en 2015, ubicada actualmente en Sunrise, Florida.

Nuestra filosofía se basa en la creación de conocimiento en comunidad y para ello su red colaborativa está conformada por:

- **Nuestros Lifelong Learners (LLL)** se identifican como Aprendices Continuos, lo que demuestra su compromiso inquebrantable con el aprendizaje permanente, que es de vital importancia en el mundo digital en constante cambio y expansión de hoy. Independientemente de si ganan o no el examen.
- Las universidades, centros de formación, y facilitadores en todo el mundo forman parte de nuestra red de aliados **CPLS (Certified Partner For Learning Solutions).**
- **Los autores (co-creadores)** son expertos de la industria o practicantes que, con su conocimiento, desarrollan contenidos para la creación de nuevas certificaciones que respondan a las necesidades de la industria.
- **Personal Interno:** Nuestro equipo distribuido con operaciones en India, Brasil, Colombia y Estados Unidos está a cargo de superar obstáculos, encontrar soluciones y entregar resultados excepcionales.

Nuestras Afiliaciones

Memberships

Digital badges issued by

IT Certification Council - ITCC

Certiprof® es un miembro activo de ITCC.

Una de las ventajas de hacer parte del ITCC es como líderes del sector colaboran entre sí en un formato abierto para explorar maneras nuevas o diferentes formas de hacer negocios que inspiran y fomentan la innovación, estableciendo y compartiendo buenas prácticas que nos permiten extender ese conocimiento a nuestra comunidad.

Certiprof ha contribuido a la elaboración de documentos blancos en el Career Path Ways Taskforce, un grupo de trabajo que se implementó internamente para ofrecer a los estudiantes la oportunidad de saber qué camino tomar después de una certificación.

Algunos de los miembros del ITCC

- **IBM**
- **CISCO**
- **ADOBE**
- **AWS**
- **SAP**
- **GOOGLE**
- **ISACA**

Agile Alliance

Certiprof® es un miembro corporativo de Agile Alliance.

Al unirnos al programa corporativo Agile Alliance, continuamos empoderando a las personas ayudándolas a alcanzar su potencial a través de la educación. Cada día, brindamos más herramientas y recursos que permiten a nuestros socios formar profesionales que buscan mejorar su desarrollo profesional y sus habilidades.

https://www.agilealliance.org/organizations/certiprof/

Credly

Esta alianza permite que las personas y empresas certificadas con Certiprof® cuenten con una distinción a nivel mundial a través de un distintivo digital.

Credly es el emisor de insignias más importante del mundo y empresas líderes en tecnología como IBM, Microsoft, PMI,Nokia, la Universidad de Stanford, entre otras, emiten sus insignias con Credly.

Empresas que emiten insignias de validación de conocimiento con Credly:

- **IBM**
- **Microsoft**
- **PMI**
- **Universidad de Stanford**
- **Certiprof**

Insignias Digitales

Insignias Digitales:
¿Qué Son?

- Según el estudio del IT Certification Council (ITCC), años atrás, la gente sabía muy poco sobre las insignias digitales. Hoy, grandes empresas e instituciones educativas de todo el mundo expiden insignias.
- Las insignias digitales contienen metadatos detallados sobre quién las ha obtenido, las competencias requeridas y la organización que las ha expedido. Algunas insignias incluso están vinculadas a las actividades necesarias para obtenerlas.
- Para las empresas e instituciones educativas, las insignias y la información que proporcionan son tan importantes que muchas decisiones, como las de contratación o admisión, se basan en los datos que aportan.

¿Por qué son importantes?

- **Facilidad de Compartir y Verificar Logros:**

  Las insignias digitales permiten a los profesionales mostrar y verificar sus logros de manera instantánea y global. Según un informe de Credly, **los perfiles de LinkedIn con insignias digitales reciben un 40% más de atención por parte de reclutadores y empleadores.**

- **Visibilidad en Plataformas Digitales:**

  En una encuesta realizada por Pearson y Credly, el **85%** de los usuarios que obtuvieron insignias digitales **las compartieron en LinkedIn,** y el **75%** reportó que esto mejoró su **credibilidad profesional en sus redes.** Además, el **76%** de los empleadores encuestados afirmó que las insignias digitales les ayudan a identificar rápidamente habilidades específicas.

¿Por qué son importantes?

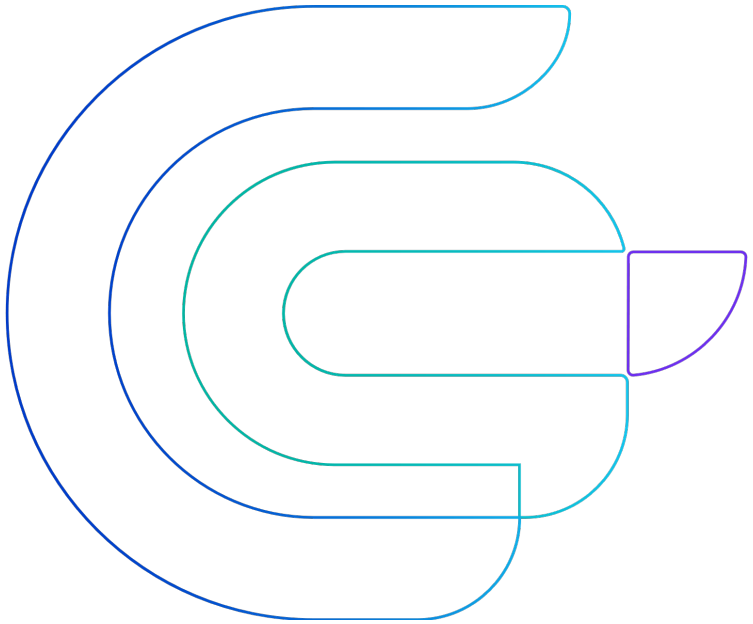
- **Impacto en la Contratación:**

  Un estudio de la **Asociación Internacional de Gestión de Proyectos (PMI)** encontró que los candidatos que muestran insignias digitales de gestión de proyectos tienen **un 60%** más de probabilidades de ser contratados en comparación con aquellos que solo mencionan sus habilidades sin verificación digital.

¿Por qué son importantes?

- **Empoderamiento de la Marca Personal:**

La visibilidad y verificación instantánea proporcionada por las insignias digitales permiten a los profesionales no solo demostrar sus habilidades, sino también construir una marca personal fuerte. Según un estudio de LinkedIn, los profesionales que utilizan insignias digitales tienen un 24% más de probabilidades de avanzar en sus carreras.
La certificación y las insignias digitales no son solo una validación del conocimiento, sino también una herramienta poderosa para la mejora continua y la empleabilidad. En un mundo donde el aprendizaje permanente se ha convertido en la norma, estas credenciales son clave para el desarrollo profesional y la competitividad en el mercado laboral global.



- No todas las insignias son iguales, y en **Certiprof,** estamos comprometidos con ofrecerte más que un simple reconocimiento digital. Al obtener una insignia emitida por certiprof, estarás recibiendo una validación de tu conocimiento respaldada por una de las entidades líderes en certificación profesional a nivel mundial.

- **Da el siguiente paso y obtén la insignia que te abrirá puertas y te posicionará como un experto en tu campo.**

¿Por qué es importante obtener su certificado?

- **Prueba de experiencia:** Su certificado es un reconocimiento formal de las habilidades y conocimientos que ha adquirido. Sirve como prueba verificable de sus cualificaciones y demuestra su compromiso con la excelencia en su campo.
- **Credibilidad y reconocimiento:** En el competitivo mercado laboral actual, las empresas y los compañeros valoran las credenciales que le distinguen de los demás. Un certificado de una institución reconocida, como Certiprof, proporciona credibilidad instantánea e impulsa su reputación profesional.
- **Avance profesional:** Tener tu certificado puede abrirte las puertas a nuevas oportunidades. Ya se trate de un ascenso, un aumento de sueldo o un nuevo puesto de trabajo, las certificaciones son un factor diferenciador clave que los empleadores tienen en cuenta a la hora de evaluar a los candidatos.

¿Por qué es importante obtener su certificado?

- **Oportunidades de establecer contactos:** Poseer un certificado le conecta con una red de profesionales certificados. Muchas organizaciones cuentan con grupos de antiguos alumnos o de trabajo en red en los que puede compartir experiencias, intercambiar ideas y ampliar su círculo profesional.

- **Logro personal:** Obtener una certificación es un logro importante, y su certificado es un recordatorio tangible del trabajo duro, la dedicación y el progreso que ha realizado. Es algo de lo que puede sentirse orgulloso y mostrar a los demás.

Insignia

DevOps Essentials Professional Certification - DEPC®

Issued by Certiprof

DevOps Essentials Professional Certification - DEPC® earners have a fundamental understanding of DevOps that encompasses culture, processes and people. This knowledge allows for a higher level of awareness and contributes to a change in mindset and process from traditional approaches, which helps even more if the organization is already adopting agile principles.

Learn more

Earn this Badge

Certification $ Paid

Skills

Collaboration | Communication | Culture | DevOps | Operations | Practice-based | Principles | Software Development

https://www.credly.com/org/certiprof/badge/devops-essentials-professional-certification-depc

Aprendizaje Permanente

- Certiprof ha creado una insignia especial para reconocer a los aprendices constantes.
- Para el 2024, se han emitido más de 1,000,000 de estas insignias en más de 11 idiomas.

Propósito y Filosofía

- Esta insignia está destinada a personas que creen firmemente en que la educación puede cambiar vidas y transformar el mundo.
- La filosofía detrás de la insignia es promover el compromiso con el aprendizaje continuo a lo largo de la vida.

Acceso y Obtención de la Insignia

- La insignia de Lifelong Learning se entrega sin costo a aquellos que se identifican con este enfoque de aprendizaje.
- Cualquier persona que se considere un aprendiz constante puede reclamar su insignia visitando:

https://certiprof.com/pages/certiprof-lifelong-learning

COMPARTE Y VERIFICA TUS LOGROS DE APRENDIZAJE FÁCILMENTE

#DEPC #certiprof

certiprof®

¿Qué es DevOps?

¿Qué es DevOps?

La palabra **DevOps** es una contracción de “Desarrollo” (Development) y “Operaciones” (Operations).

¿Qué es DevOps?

DevOps es una nueva tendencia en la industria **TI** dirigida a mejorar la agilidad del servicio de entregas en **TI**. El movimiento hace énfasis en la comunicación transparente, la colaboración junto con la integración entre el software de desarrolladores y las operaciones de **TI**.

DevOps reconoce que los desarrolladores y los operadores de **TI** son grupos con relación que pueden interactuar entre sí, pero no realmente trabajar juntos.

DevOps ayuda a la organización a crear servicios **TI** y software de manera rápida lo que resulta en la reducción del número de iteraciones.

Los equipos **DevOps** logran el éxito por el uso de dos componentes claves llamados *"comunicación"* y *"visibilidad en tiempo real"*.

¿Qué es DevOps?

Es fundamental tener las herramientas adecuadas y combinar servicios, **DevOps** se preocupa por sí una herramienta provee la capacidad de interactuar y funcionar eficazmente.

DevOps es relativamente un nuevo desarrollo en la industria **TI**, que enfatiza en la comunicación y colaboración entre el software de desarrolladores y los otros profesionales de **TI** como el equipo de operadores, con el objetivo de automatizar el proceso de entrega de software y los cambios a la infraestructura.

Los objetivos básicos de **DevOps** son establecer un ambiente donde realizar códigos, probar y desarrollar software pueda realizarse rápidamente, de manera frecuente y segura.

¿Qué es DevOps?

No existe una sola herramienta de **DevOps** que trabaje en la colaboración e integración entre los equipos de desarrolladores, testers y operaciones.

Se utilizan una cadena de herramientas **DevOps** que consiste en un número de herramientas que se ajustan en varias categorías del proceso en las fases desde desarrollo a la implementación.

Estas herramientas son usadas en los procesos que involucran a los equipos de código, construcción, test, empaque, liberación, configuración y monitorización.

¿Por qué DevOps?

Comunicación tradicional entre desarrollo y operaciones.

Definiciones de DevOps

No hay un acuerdo claro y universal sobre su definición.

Hay varias opiniones sobre qué es y qué no es **DevOps**, generalmente se define como una nueva forma de organización, una cultura o incluso una nueva forma de pensar.

¿Qué no es DevOps?

DevOps no es una estrategia para todos.

Hay gran diversidad de tecnologías empresariales y drivers a ser considerados para establecer la estrategia de adopción para **DevOps**.

DevOps no es automatización.

DevOps implica automatización. **DevOps** es más que automatización.

¿Qué no es DevOps?

DevOps no es una herramienta implementada.

Aunque hay herramientas que son usadas en **DevOps**, no deberíamos limitar su alcance a herramientas específicas como **Chefs** o **Jenkins**. Esto limita el amplio alcance como si una sola herramienta de automatización se equiparara con **DevOps**.

DevOps no es equipo de trabajo nuevo y separado de las demás áreas de TI.

Tener un equipo **DevOps** separado, anula el propósito de evitar las posibles fricciones y falta comunicación entre los desarrolladores y operadores de **TI** ya que crea un silo más.

Definición de DevOps Según sus Líderes

Nos referimos a "**DevOps**" como el resultado de la aplicación de principios eficientes a la corriente de valor de **TI**.

Libro de cocina de DevOps (DevOps Cookbook).

*"Una mezcla de patrones destinados a mejorar la colaboración entre desarrolladores y operadores. **DevOps** se dirige a compartir metas e incentivos, así como procesos compartidos y herramientas".*

Michael Hüttermann.

Definición de DevOps Según sus Líderes

"Un movimiento de personas quienes se preocupan por desarrollar y operar sistemas fiables, seguros y de alto rendimiento a escala".

Jez Humble.

*"**DevOps** es una cultura o un movimiento profesional".*

Adam Jacob, CTO at Chef.

*"**DevOps** es como un movimiento filosófico".*

Gene Kim, Fundador de TripWire, CTO y Autor.

Definiciones de DevSecOps

DevSecOps es una metodología que **integra la seguridad en todas las etapas del ciclo de vida** del desarrollo de software.

Combina los principios de DevOps con prácticas de seguridad, fomentando la colaboración y la **responsabilidad compartida entre los equipos de desarrollo, operaciones y seguridad.**

El objetivo principal de DevSecOps es crear un enfoque proactivo para abordar la seguridad, evitando retrasos y problemas de seguridad en las fases finales del desarrollo.

Beneficios de DevSecOps

Mayor seguridad: Al integrar la seguridad desde el principio, DevSecOps ayuda a identificar y abordar los problemas de seguridad de manera más temprana en el ciclo de desarrollo, reduciendo los riesgos y las vulnerabilidades en el software.

Mayor agilidad: DevSecOps permite un desarrollo y despliegue más rápidos y frecuentes, ya que las pruebas de seguridad se automatizan y se incorporan en los flujos de trabajo, evitando retrasos innecesarios.

Mayor colaboración: DevSecOps fomenta la colaboración continua entre los equipos de desarrollo, operaciones y seguridad, mejorando la comunicación y evitando la fragmentación en los procesos de desarrollo.

Beneficios de DevSecOps

Cumplimiento normativo: Al incluir la seguridad desde el principio, DevSecOps facilita el cumplimiento de los requisitos normativos y las políticas de seguridad, reduciendo el riesgo de infracciones y sanciones.

Mayor calidad del software: Al abordar los problemas de seguridad desde el principio, DevSecOps ayuda a mejorar la calidad general del software, evitando la introducción de vulnerabilidades y mitigando los riesgos asociados con brechas de seguridad.

Historia

Historia

2008

Las semillas del movimiento **DevOps** fueron plantadas durante la conferencia de **Agile**, 2008 celebrada en **Toronto**, por el desarrollador de software **Patrick Debois**, quien tenía experiencia en múltiples funciones en una gran organización de la industria **TI** como desarrollador, administrador de sistema, especialista de red, gerente de proyecto e incluso tester.

Él demostró que podía haber mejores maneras de obtener un gran trabajo al resolver los conflictos entre los equipos de desarrollo y operaciones. Pronto fue reconocido como el líder de la idea detrás del concepto de **DevOps** y otros continuaron resolviendo estos desafíos.

Historia

2009

John Allspaw y **Paul Hammond**, dos empleados en jefe de **Flickr**, presentaron una charla clave, titulada *"Diez despliegues en el día: cooperación de Dev y Ops en Flickr"*. En esta charla **Allspaw** y **Hammond** señalaron poderosamente como el conflicto llevó a *"apuntarse con el dedo"* entre los desarrolladores y operadores al culparse entre ellos.

Ellos señalaron que la única manera de construir y desplegar un software viable era hacer a operaciones y desarrollo integrados y transparentes.

Inspirado por esto, **Debois** organizó su propia conferencia (*DevOpsDay*). El nombre de este movimiento fue reducido a **DevOps** después de esta convención.

El primer evento **DevOps** fue organizado por **Debois** en Ghent, Bélgica.

Historia

2010

Primer **DevOpsDay** organizado en los Estados Unidos tuvo a defensores de **DevOps** como **Andrew Clay Shafer**, **Damon Edwards**, entre otros. Los eventos llamaron la atención global y fueron avanzando hacia la comunidad **DevOps**.

Introducción del hashtag ***#DevOps*** que resultó ser una rica fuente de información.

2011

Muchos análisis pronosticaron el surgimiento de **DevOps** a nivel global para el 2020. Se desarrollaron herramientas de código abierto tales como **Vagrant** que funcionaba con **Chef**, **Puppet** y herramientas de administración de configuraciones similares.

Historia

2012

Los **DevOpsDays** fueron organizados alrededor del mundo y se convirtieron en eventos **TI** muy concurridos y deliberados sobre el pensamiento innovador en el dominio **DevOps**.

2013

Mike Loukides, una figura prominente en el mundo **DevOps**, junto con **Debois** editaron algunos textos fundamentales de **DevOps**. Él afirmó que es fácil pensar en **DevOps** en términos de las herramientas que se utilizan en el mismo. Pero, en realidad este es un acuerdo íntimo entre los equipos de desarrollo y operaciones.

Gran cantidad de libros sobre **DevOps** aparecieron. Algunos de los más notables son *"El Proyecto Phoenix"* (The Phoenix Project), *"Implementando la Eficiencia del Desarrollo de Software"* (Implementing Lean Software Development), *"Operaciones Web"* (Web Operations) y *"El Inicio Eficiente"* (The Lean Startup), etc.

Historia

2014

El mundo tecnológico en evolución presenta nuevas oportunidades para el concepto de **DevOps** en forma de explosión de nuevas aplicaciones, dispositivos y comunicaciones en entornos móviles y **Cloud Computing**.

En una encuesta realizada por **Puppet Labs**, 16 % de 1486 encuestados afirmaron que ellos son parte de **DevOps** en sus organizaciones.

2015

DevOps es presente y futuro...

Historia

2019

El mundo tecnológico en evolución presenta nuevas oportunidades para el concepto de **DevOps** en forma de explosión de nuevas aplicaciones, dispositivos y comunicaciones en entornos móviles y **Cloud Computing**.

En una encuesta realizada por **Puppet Labs**, 16 % de 1486 encuestados afirmaron que ellos son parte de **DevOps** en sus organizaciones.

2020

La pandemia de COVID-19 acelera la adopción de DevOps, ya que las empresas se enfrentan a la necesidad de adaptarse rápidamente y entregar software de manera eficiente en un entorno remoto.

Historia

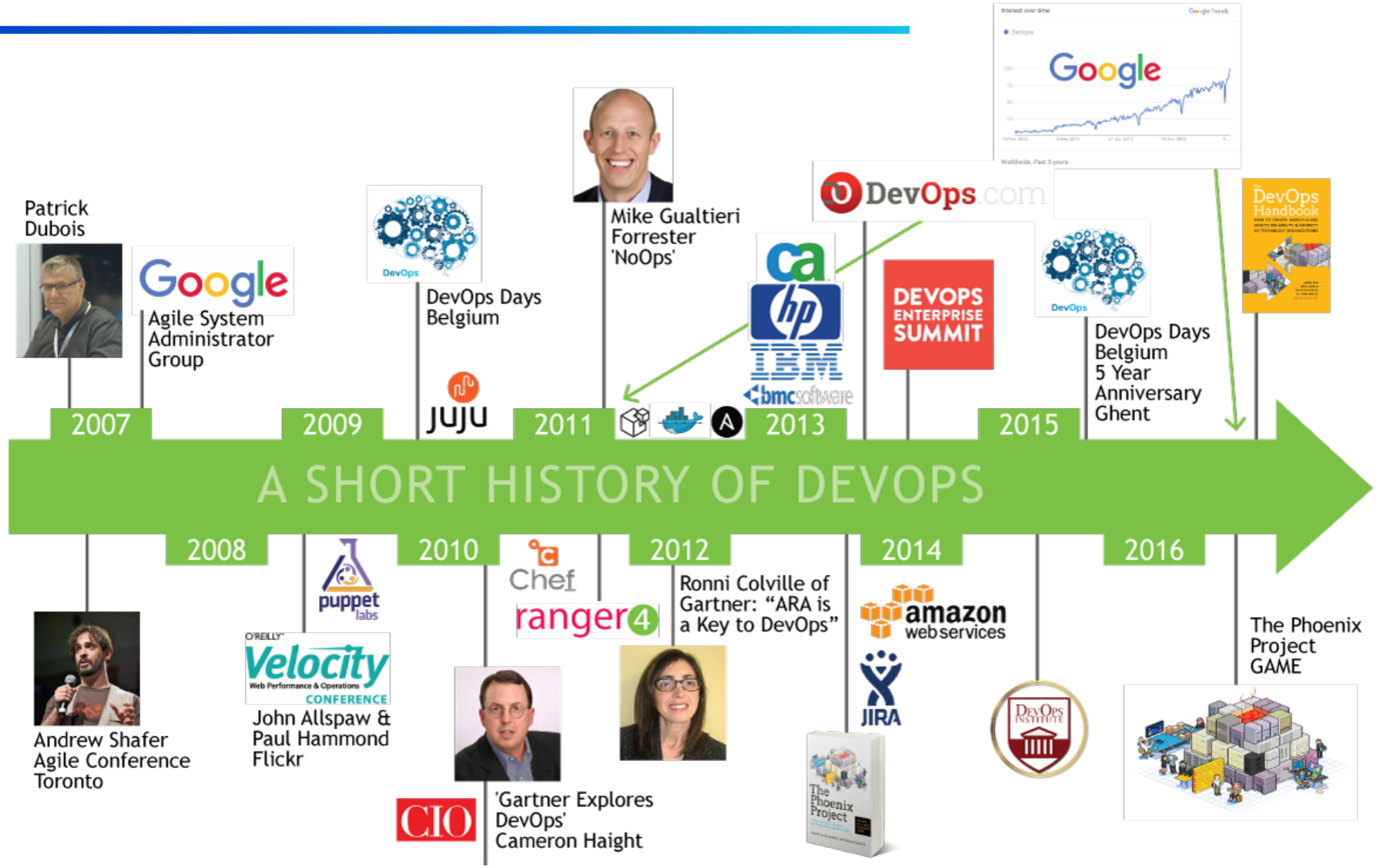
2021

El informe State of DevOps 2021, publicado por Puppet y CircleCI, destaca los beneficios comerciales significativos obtenidos por las organizaciones que han implementado prácticas DevOps maduras.

2022

El informe de Agile Adoption Report de CertiProf muestra aun las grandes posibilidades que hay en la adopción de DevOps en las organizaciones y el estado actual.

Historia


Patrick Dubois
Google
Agile System Administrator Group
DevOps Days Belgium
Mike Gualtieri Forrester 'NoOps'
DevOps.com
DEVOPS ENTERPRISE SUMMIT
DevOps Days Belgium 5 Year Anniversary Ghent
2007
2009
2011
2013
2015
A SHORT HISTORY OF DEVOPS
2008
2010
2012
2014
2016
Andrew Shafer Agile Conference Toronto
John Allspaw & Paul Hammond Flickr
'Gartner Explores DevOps' Cameron Haight
Ronni Colville of Gartner: "ARA is a Key to DevOps"
The Phoenix Project GAME

Propósito de DevOps

Propósito de DevOps

El mejor propósito ofrecido por **DevOps**, es iterar de manera más rápida durante la fase de desarrollo.

Esto se logra al evitar la fricción entre los desarrolladores y operadores tanto como sea posible.

Esto se logra garantizando la transparencia e integración entre el equipo de desarrollo y operaciones.

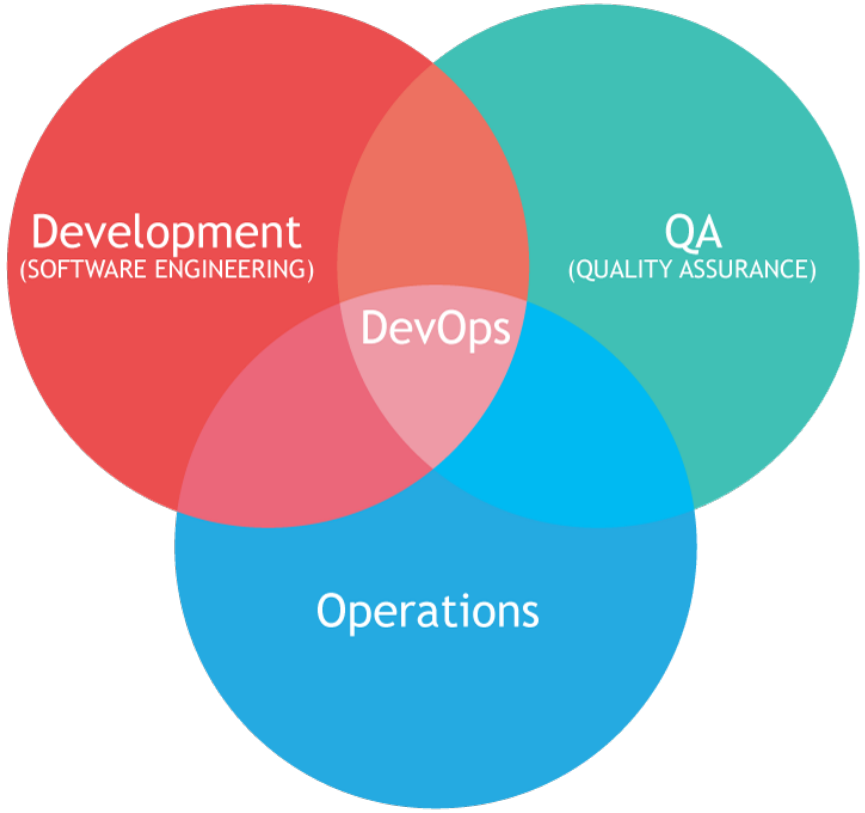
El objetivo de **DevOps** es establecer procesos de negocios alineados en flujo *"justo a tiempo"* (JIT por sus siglas en inglés).

DevOps busca maximizar los resultados del negocio, tales como incrementar las ventas y la rentabilidad, mejorar la velocidad del negocio, o minimizar costos operativos, al alinear los procesos empresariales *"justo a tiempo"* (JIT).

Propósito de DevOps

Las empresas buscan obtener nuevas aplicaciones o servicios con propósitos específicos, pero se necesita un tiempo considerable para codificar el proyecto, algunos días para asegurar la calidad (Quality Assurance), otros más para manejar problemas de implementación, mantenimiento y en muchos casos se necesita regresar atrás debido a la falta de coincidencia entre lo que se esperaba y lo que se entrega.

Muchos proyectos toman meses en esta inevitable ronda de eventos. En este contexto, **DevOps** ayuda a realizar la implementación de manera rápida y sin esfuerzo. Por lo tanto, el propósito general de **DevOps** es lograr la rapidez en el despliegue.



Propósito de DevOps

- **DevOps** desea establecer la cadena de suministros de servicios **TI** en el negocio, de la misma manera que la cadena de suministro para otros productos está incrustado. Es un gran cambio de paradigma desde la entrega del software hasta la prestación de servicios **TI.**

- Desde el punto de vista arquitectónico, **DevOps** necesita establecer un sistema de despliegue automatizado rápido.

- Hay muchas metodologías y herramientas que pueden ser utilizadas.

- **DevOps** no tiene un modelo para la implementación, cada organización tiene que pensar y construir su propio proceso **DevOps** para mejorar su negocio.

Propósito de DevOps

Algunos modelos planteados.

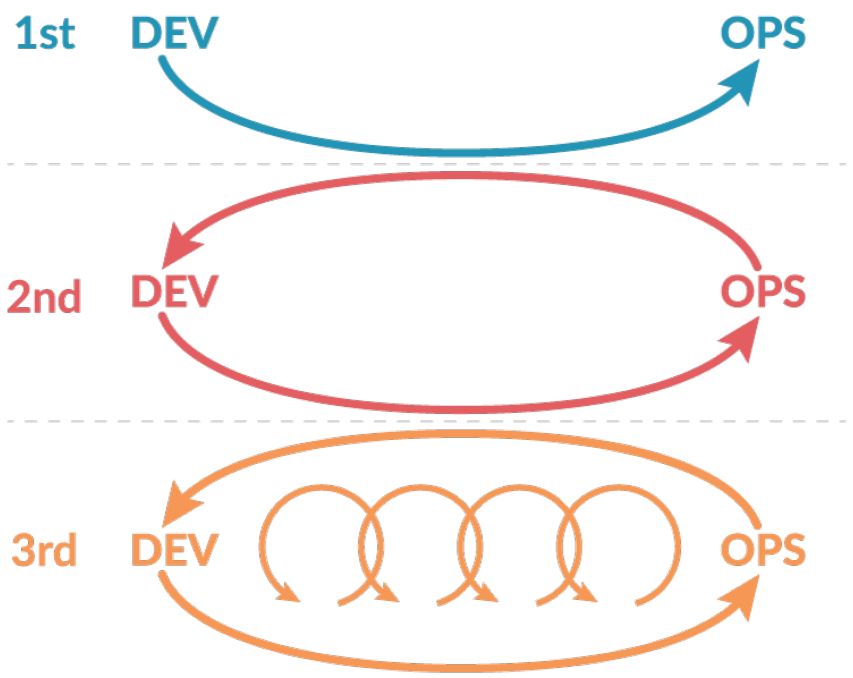
- DevOps Implementation Framework (DIF).
- DevOps Roadmap.
- DevOps Journey.
- DevOps Process.

DevOps Adopción

El enfoque incremental se centra en la idea de minimizar el riesgo y el costo de una adopción de **DevOps**, al tiempo que se construyen las habilidades necesarias y el impulso necesario para lograr una implementación exitosa en toda la empresa.

Los *"Principios de tres maneras"* de **Gene Kim** establecen esencialmente diferentes formas de adopción incremental de **DevOps**:

- **La primera manera:** Pensamiento de sistemas.
- **La segunda manera:** Amplificar bucles de retroalimentación.
- **La tercera manera:** La cultura de la experimentación continua y el aprendizaje.



Beneficios

Beneficios

DevOps garantiza un tiempo más rápido de comercialización de los plazos de entrega y, por lo tanto, mejora la rentabilidad de las inversiones (ROI).

Fundamentalmente, **DevOps** es una aplicación del concepto de desarrollo **Agile** y por lo tanto el principal beneficio de **DevOps** es el desarrollo más rápido de software y la entrega frecuente mejorando la línea de fondo.

DevOps trae el mayor beneficio al mejorar la colaboración entre el desarrollador y los equipos de operación. Esto se logra mejorando la transparencia que es esencial para una toma de decisiones efectiva.

Hoy en día, los equipos de desarrollo deben dividir sus silos departamentales, comunicarse y colaborar con otros equipos de **TI** en el entorno dinámico actual.

Beneficios

DevOps mejora la agilidad ya que ofrece un entorno de colaboración, comunicación e integración en equipos diversamente ubicados en una organización global de **TI**.

Otro beneficio significativo de **DevOps** es la detección temprana y la correspondiente corrección más rápida de los defectos que implica ofrecer los mejores servicios entregados a los clientes.

Uno de los beneficios clave de **DevOps** es el lanzamiento, implementación, supervisión y la corrección continua.

El desarrollo de software actual requiere que los equipos se involucren en la entrega continua sin fallas, en un período reducido para los marcos de tiempo de salida al mercado y ciclos de lanzamiento más cortos.

Estabilidad

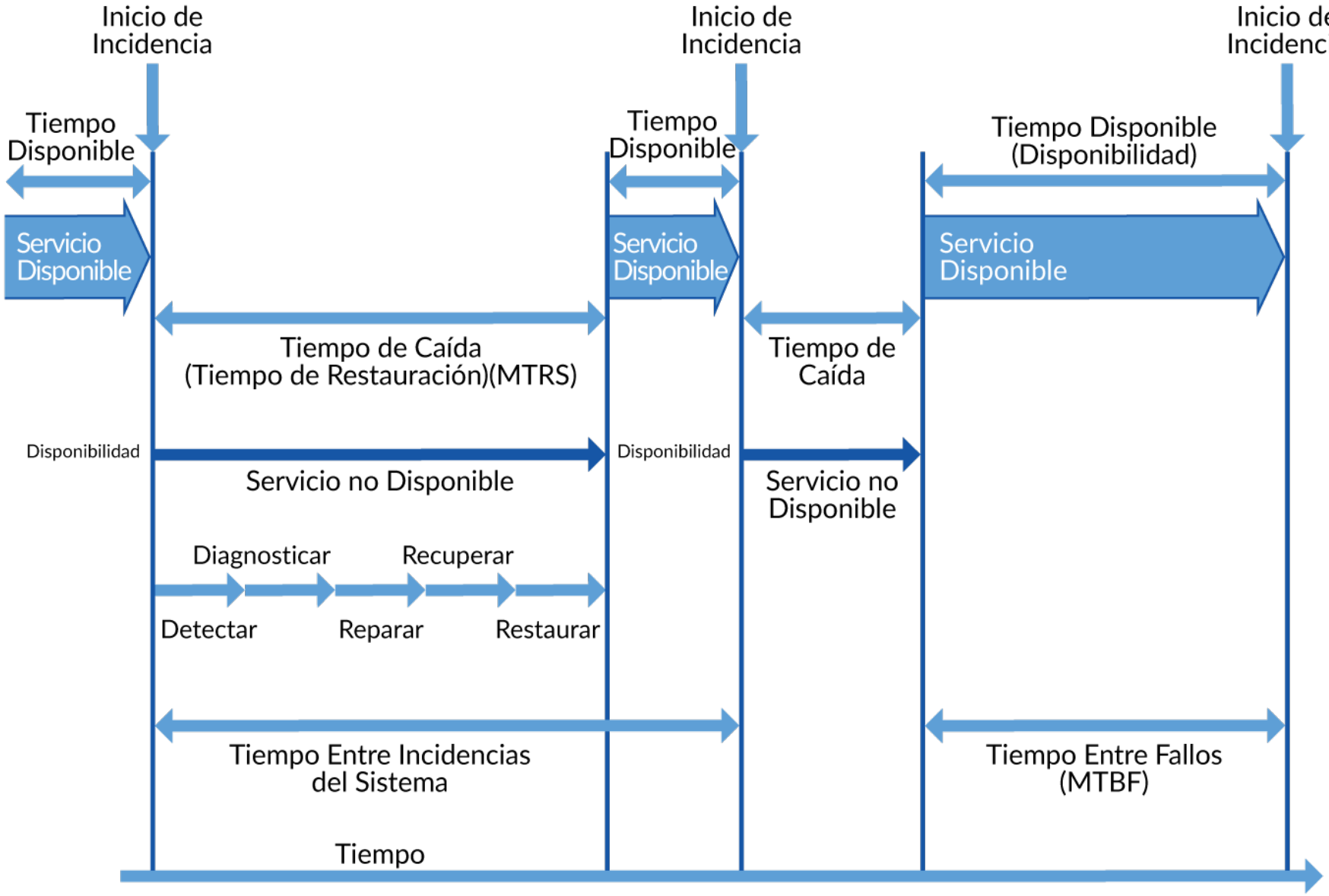
Existen mejoras en los Mean Time To Recover (MTTR).

Es el tiempo promedio para reparar un servicio de **TI** u otro elemento de configuración después de una falla.

El **MTTR** se mide desde el momento en que el elemento de configuración falló hasta que fue reparado.

El **MTTR** no incluye el tiempo necesario para recuperar o restaurar. A veces se usa incorrectamente en lugar del tiempo medio para restablecer el servicio.

Estabilidad



Taller
30 Minutos

DEPC® Versión 072023

Análisis en Grupos

Informe State of DevOps 2021 y Agile Adoption Report de CertiProf

¿Cuáles son los principales beneficios comerciales destacados en el informe y cómo se relacionan con la adopción de prácticas DevOps maduras?

¿Cuáles son los factores clave identificados en el informe que contribuyen al éxito de la adopción de DevOps en las organizaciones?

¿Qué datos o estadísticas del informe respaldan la importancia de la colaboración entre los equipos de desarrollo, operaciones y seguridad en la implementación de DevOps?

¿Cómo se destaca la automatización en el informe como un componente esencial de DevOps y qué beneficios concretos se mencionan?

¿Cuáles son los principales desafíos y obstáculos mencionados en el informe en relación con la adopción de DevOps y cómo se abordan?

Pilares de DevOps

Pilares de DevOps

Agile

1. Velocidad.
2. Adaptabilidad al cambio.
3. Lanzamiento sin errores (JKK Concept).

ITSM

1. Valor del Concepto (ITIL®).
   1.1 Utilidad.
   1.2 Garantía.

Entrega Continua

Conceptos

Just-in-time (JIT) o Justo a Tiempo

La fabricación **Justo a Tiempo** (JIT), también conocida como producción justo a tiempo o el sistema de producción **Toyota** (TPS), es una metodología dirigida principalmente a reducir los tiempos de flujo dentro de la producción, así como los tiempos de respuesta de los proveedores y los clientes.

Siguiendo su origen se desarrollo en Japón, en gran parte en los años 60 y 70 y particularmente en **Toyota**.

JIT permite reducir costos, especialmente de bodega de materias, partes para el embalaje y de los productos finales.

La esencia de **JIT** es que los insumos llegan a la fábrica o los productos al cliente, *"Justo a tiempo",* eso siendo poco antes de que se usan y solo en las cantidades necesarias. Esto reduce o hasta elimina la necesidad de almacenar y luego mover los insumos de la bodega a la línea de producción (en el caso de una fábrica).

Sistema de Producción Toyota (SPT)

El **Sistema de Producción Toyota** (**SPT**) (Toyota Production System o TPS en inglés), es un sistema integral de producción *"Integral Production System"* y gestión surgido en la empresa japonesa automotriz del mismo nombre.

En origen, el sistema se diseñó para fábricas de automóviles y sus relaciones con proveedores y consumidores, sin embargo, este se ha extendido hacia otros ámbitos. Este sistema es un gran precursor para el genérico **Lean Manufacturing**.

El desarrollo del sistema se atribuye fundamentalmente a tres personas: el fundador de Toyota, **Sakichi Toyoda**, su hijo **Kiichiro** y el ingeniero **Taiichi Ohno**, quienes crearon este sistema entre 1946 y 1975. Originalmente llamado *"Producción Justo-a-tiempo"*. Los principios de **SPT** son mencionados en el libro *"La Manera de Toyota"*.

Kaizen

Kaizen, término japonés para *"mejora"*.

Cuando se utiliza en el sentido comercial y se aplica al lugar de trabajo, **Kaizen** se refiere a actividades que mejoran continuamente todas las funciones e implican a todos los empleados desde el **CEO** a los trabajadores de la línea de montaje.

También se aplica a procesos, tales como compras y logística que cruzan los límites de la organización en la cadena de suministro. Se ha aplicado en la asistencia sanitaria, psicoterapia, entrenamiento de vida, gobierno, banca y otras industrias.

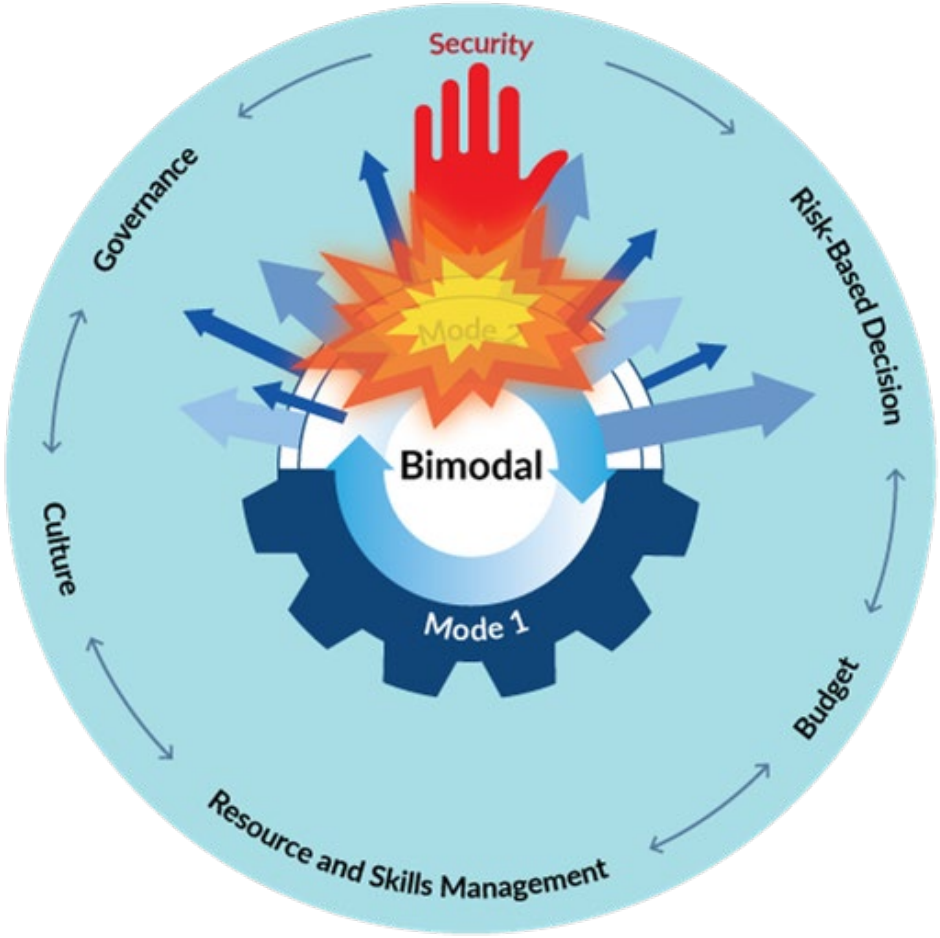
BIMODAL – Gartner

Bimodal está ocurriendo.

Si intentas aplicar los convenios de seguridad del modo 1 en las iniciativas del modo 2, serás sobrepasado y la empresa quedará expuesta a riesgos insostenibles.

Para 2019, el 30 % de **CISCOs** adaptará las prácticas de gestión de riesgos, apoyará la **TI** bimodal y mejorará las tasas de éxito del modo 2, al tiempo que reducirá los costos.

http://www.gartner.com/it-glossary/?s=Bimodal



BIMODAL – Gartner

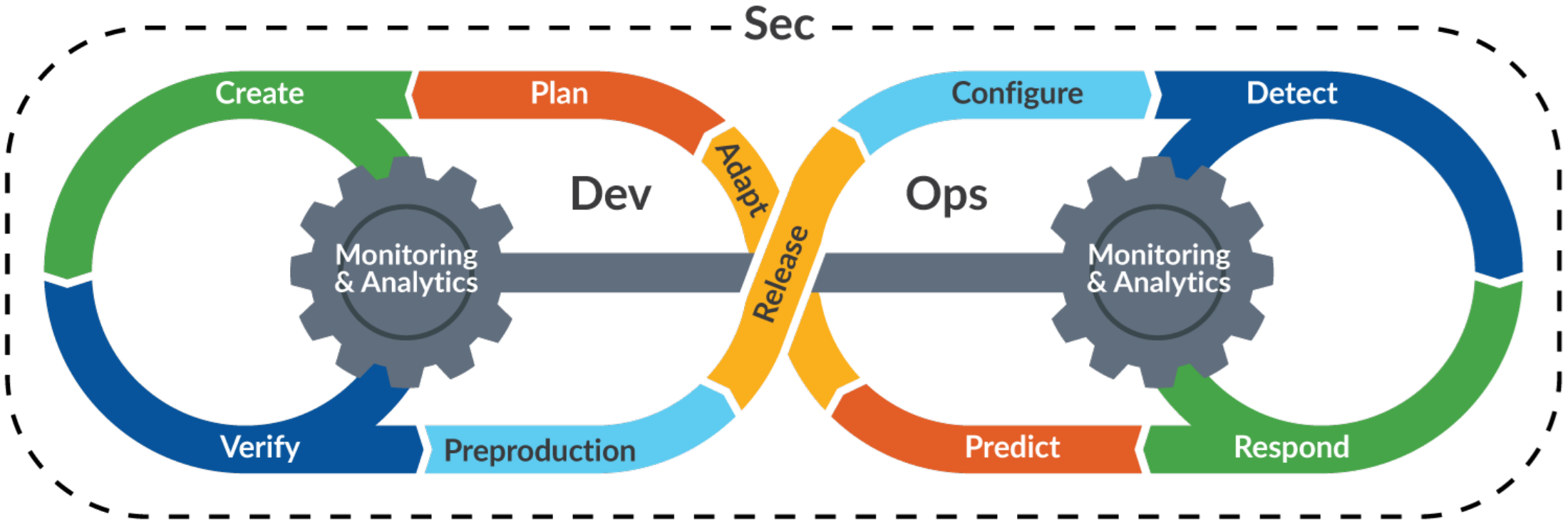
	Modo 1	Modo 2
Meta	Confiabilidad.	Agilidad.
Valor	Precio por rendimiento.	Ingresos, marca, experiencia del cliente.
Enfoque	Lineal, cascada, high-ceremony de la aplicación de desarrollo Agile.	Iterativo, low-ceremony, no lineal, lean startup, kanban, aplicación de desarrollo Agile.
Gobernancia	Planificado, base de aprobación.	Empírica, continua, implícita en el enfoque.
Abastecimiento	Suministros empresariales, tratos de largos términos.	Pequeños, vendedores nuevos, tratos de corto tiempo.
Talento	Buenos en procesos convencionales, proyectos.	Buenos en nuevos enfoques y lidiando con incertidumbres.
Cultura	IT céntrica, arm´s-length para clientes.	Centrado en el negocio, cerrado a clientes.
Tiempos de Ciclo	Largos (meses).	Corto (días, semana).

http://www.gartner.com/it-glossary/?s=Bimodal

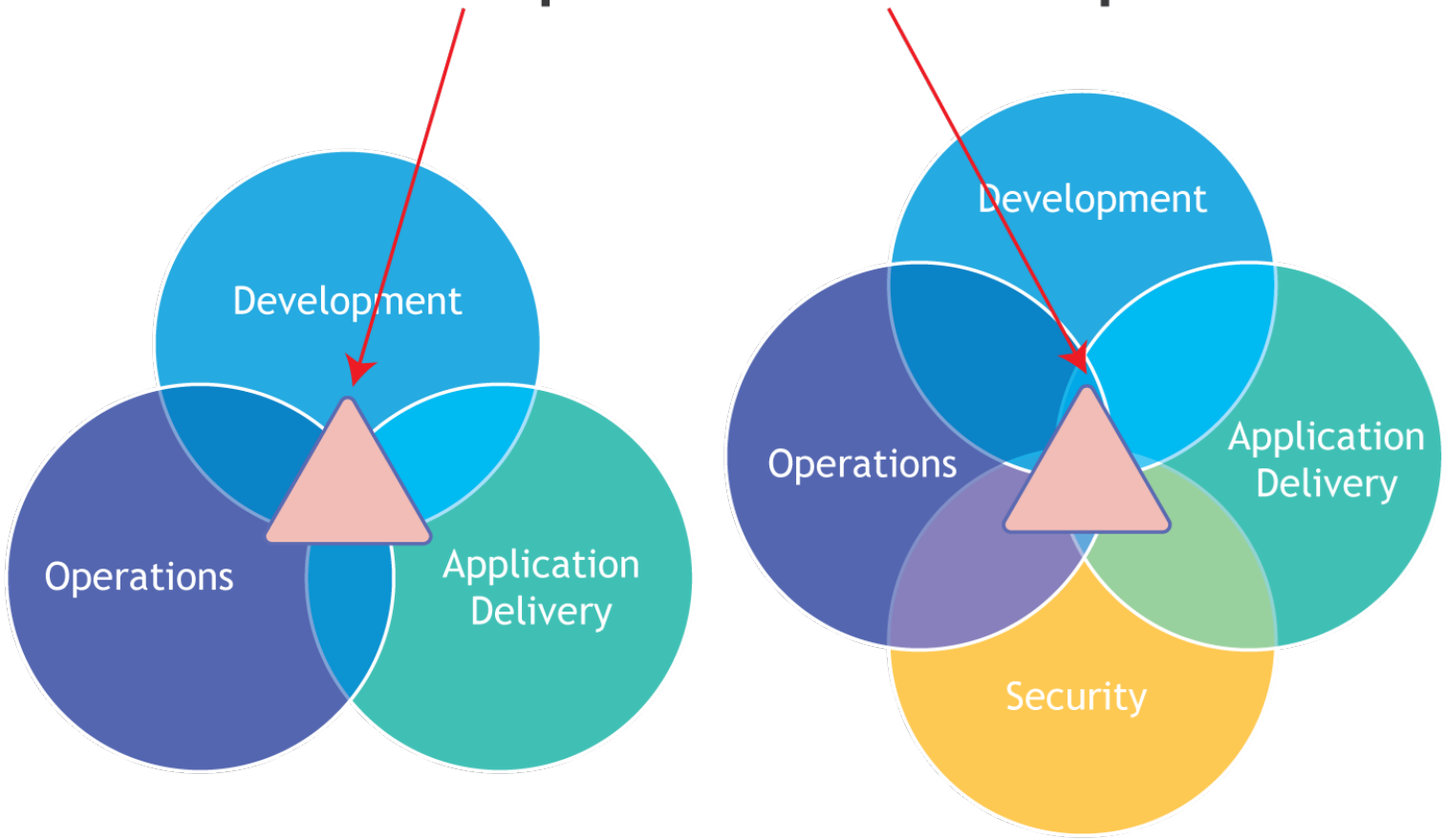
IT Bimodal: Dos enfoques distintos pero coherentes, bastante diferentes, ambos esenciales.

DevSecOps

DevOps tiene una relación nada fácil con seguridad.



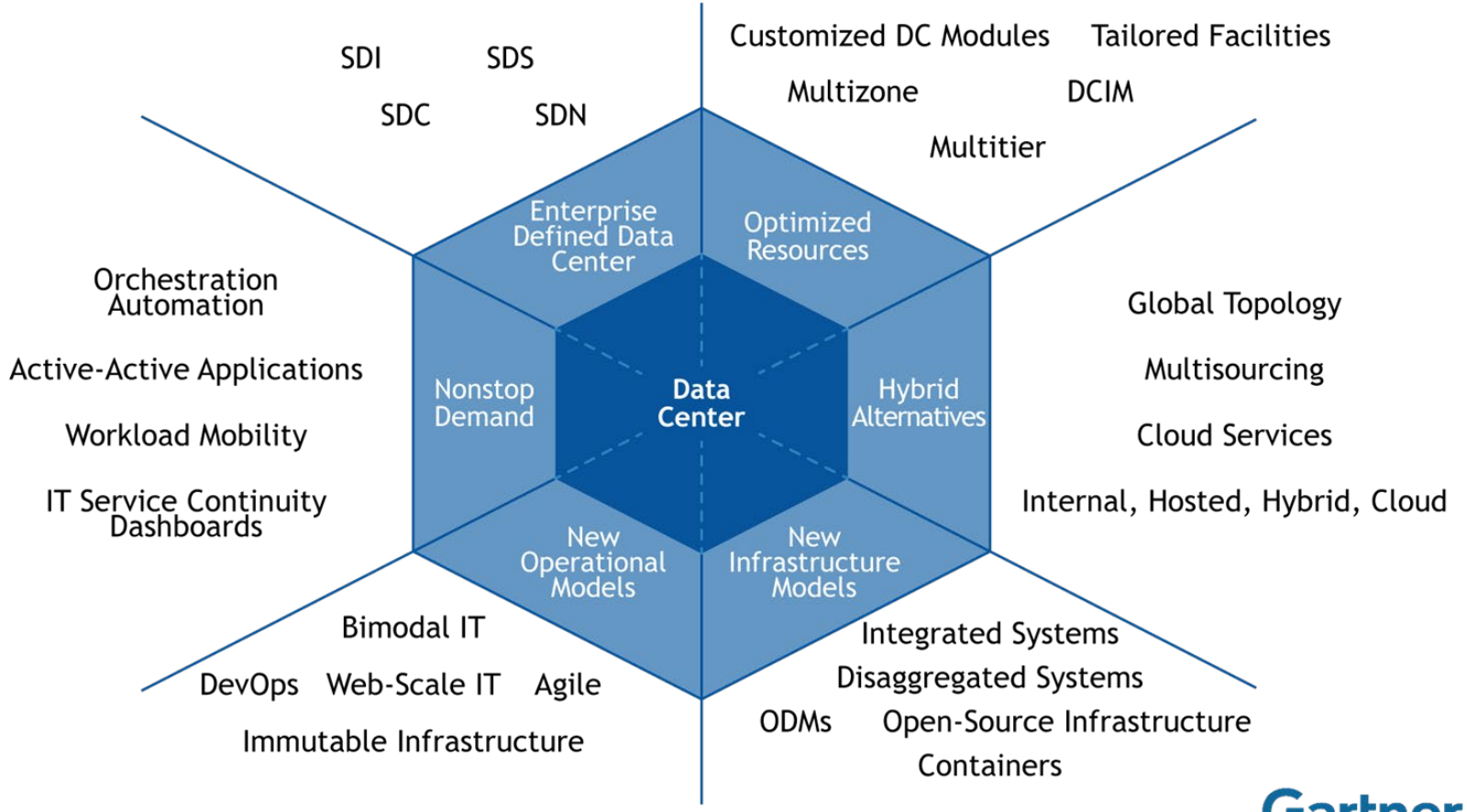
DevOps vs DevSecOps



Si **DevOps** no presta atención a la seguridad puede facilitar la rápida introducción de las vulnerabilidades.

Nuevo Modelo Operacional

¿Cómo lucirá el Centro de Datos en el futuro?



Gartner

Scrum

Scrum es un marco de trabajo ligero para desarrollar, entregar y mantener productos complejos. Se basa en los principios de transparencia, inspección y adaptación. Scrum promueve una colaboración estrecha entre los miembros del equipo y se enfoca en el aprendizaje iterativo y en la entrega de valor de manera incremental.

Scrum se basa en roles definidos, eventos, artefactos y reglas específicas que ayudan a las personas y a los equipos a trabajar de manera efectiva y eficiente

Ken Schwaber y Jeff Sutherland desarrollaron Scrum.

CertiProf tiene su propio programa de certificación en Scrum alineado a la Guía Oficial.

SCRUM PATH

Fuente:
www.scrumguides.org/docs/scrumguide

Scrum


Daily Scrum
Sprint Backlog
Sprint Planning
Product Backlog
Product Idea
Sprint Retrospective
Sprint Review
Potentially Shippable Product Increment
SPRINT

Work in Progress, WIP (Trabajo en Proceso)

Un límite **WIP** (work in progress) es una estrategia para prevenir cuellos de botella en el desarrollo de software.

Se acuerda por el equipo de desarrollo trabajar en progresos limitados antes de que un proyecto comience y sean ejecutados por el facilitador del equipo.

Por ejemplo, un equipo puede dividir las tareas que deben realizarse para una característica en el diseño, código, prueba y despliegue. Cuando se alcanza un límite **WIP** para una determinada tarea, el equipo se detiene y trabaja en conjunto para eliminar el cuello de botella. El objetivo de trabajar de esta manera es asegurar que todo el equipo se apropie del proyecto y produzca códigos de alta calidad.

Acuerdo de Nivel de Servicio (SLA) y OLAS

Un **Acuerdo de Nivel de Servicio** *(Service-level agreement o SLA)* se define como un compromiso oficial que prevalece entre un proveedor de servicios y el cliente.

Aspectos particulares del servicio (calidad, disponibilidad, responsabilidades) son acordados entre el proveedor de servicios y el usuario del servicio.

Acuerdos de Nivel Operacional *(Operational-level agreements u OLAs)* pueden ser utilizados por grupos internos para apoyar **SLAs**.

Nota: Los SLAs ayudan a identificar áreas de mejora y oportunidades para optimizar los procesos y las prácticas de DevOps.

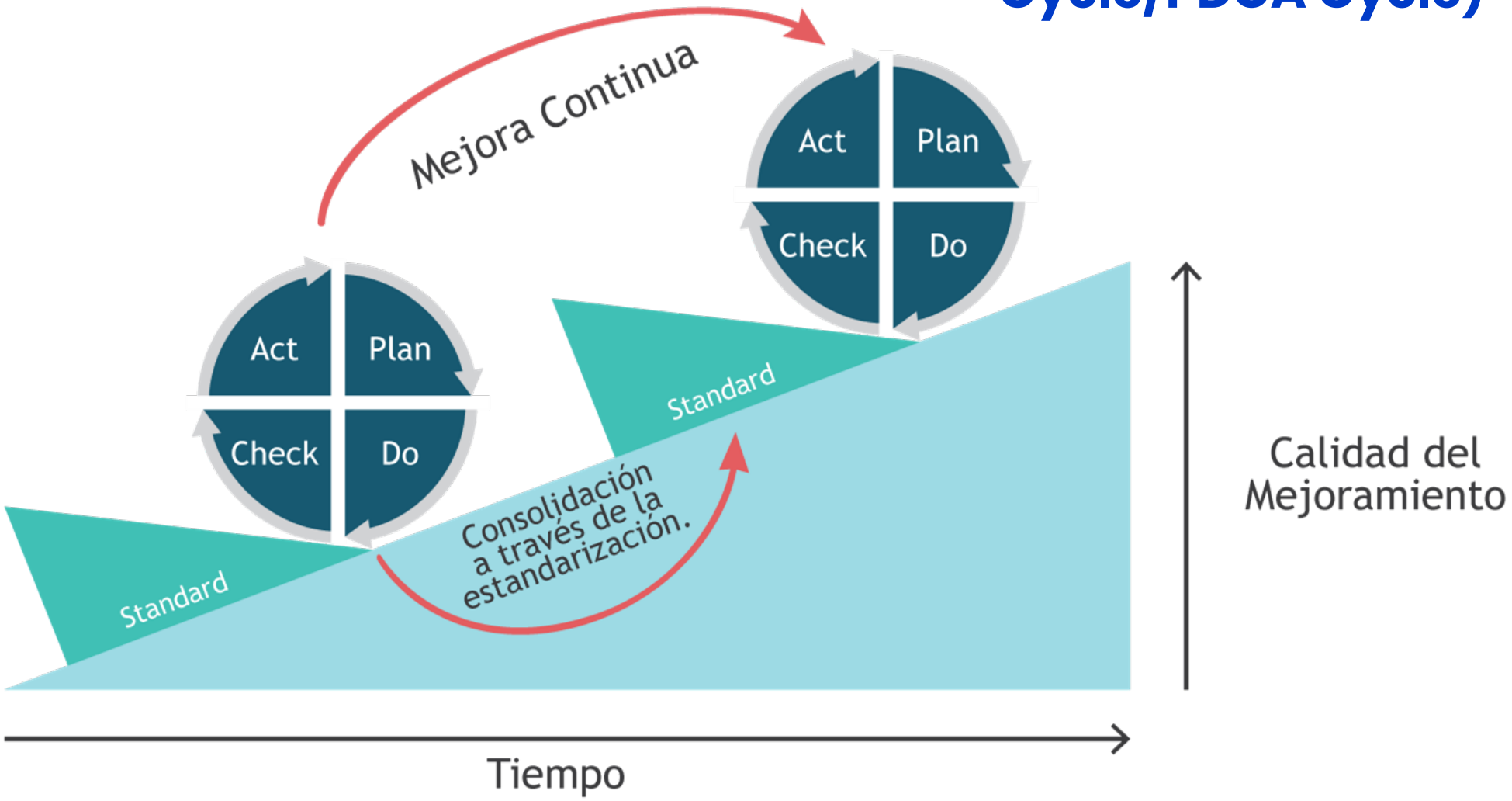
Ciclo Plan-Hacer-Verificar-Actuar (Plan-Do-Check-Act Cycle/PDCA Cycle)

PDCA (Planear-Hacer-Verificar-Actuar o Planear-Hacer-Verificar-Ajustar) es un método de gestión de cuatro pasos iterativo utilizado en los negocios para el control y la mejora continua de procesos y productos.

También se conoce como círculo/ciclo/rueda de **Deming**, ciclo de **Shewhart**, círculo/ciclo de control, o Planear-Hacer-Estudiar-Actuar (**PDSA**).

Otra versión de este ciclo **PDCA** es **OPDCA**. El "O" agregado para la observación o como algunas versiones dicen “Comprender la condición actual”. Este énfasis en la observación y la condición actual tiene relación con la fabricación **Lean** o la literatura del sistema de producción de **Toyota**.

Ciclo Plan-Hacer-Verificar-Actuar (Plan-Do-Check-Act Cycle/PDCA Cycle)



Definición de Terminado (en Agile/Scrum) (Definition of Done)

La Definición de Terminado es una descripción formal del estado del Incremento cuando cumple con las medidas de calidad requeridas para el producto.

En el momento en que un elemento de trabajo pendiente de producto cumple con la Definición de Terminado, se crea un incremento.

La Definición de Terminado crea transparencia al proporcionar a todos una comprensión compartida de qué trabajo se completó como parte del Incremento. Si un elemento de trabajo pendiente de producto no cumple con la Definición de Terminado, no se puede liberar.

Los Desarrolladores deben ajustarse a la Definición de Terminado. Si hay varios Equipos de Scrum trabajando juntos en un producto, deben definir y cumplir mutuamente con la misma Definición de Terminado.

Kanban para Equipos DevOps

El método **Kanban** puede ayudar a los equipos de **DevOps** a poner un poco de orden en su trabajo diario y la teoría **Lean** definitivamente puede ser apalancada para mejorar el flujo a través de los equipos de **DevOps**.

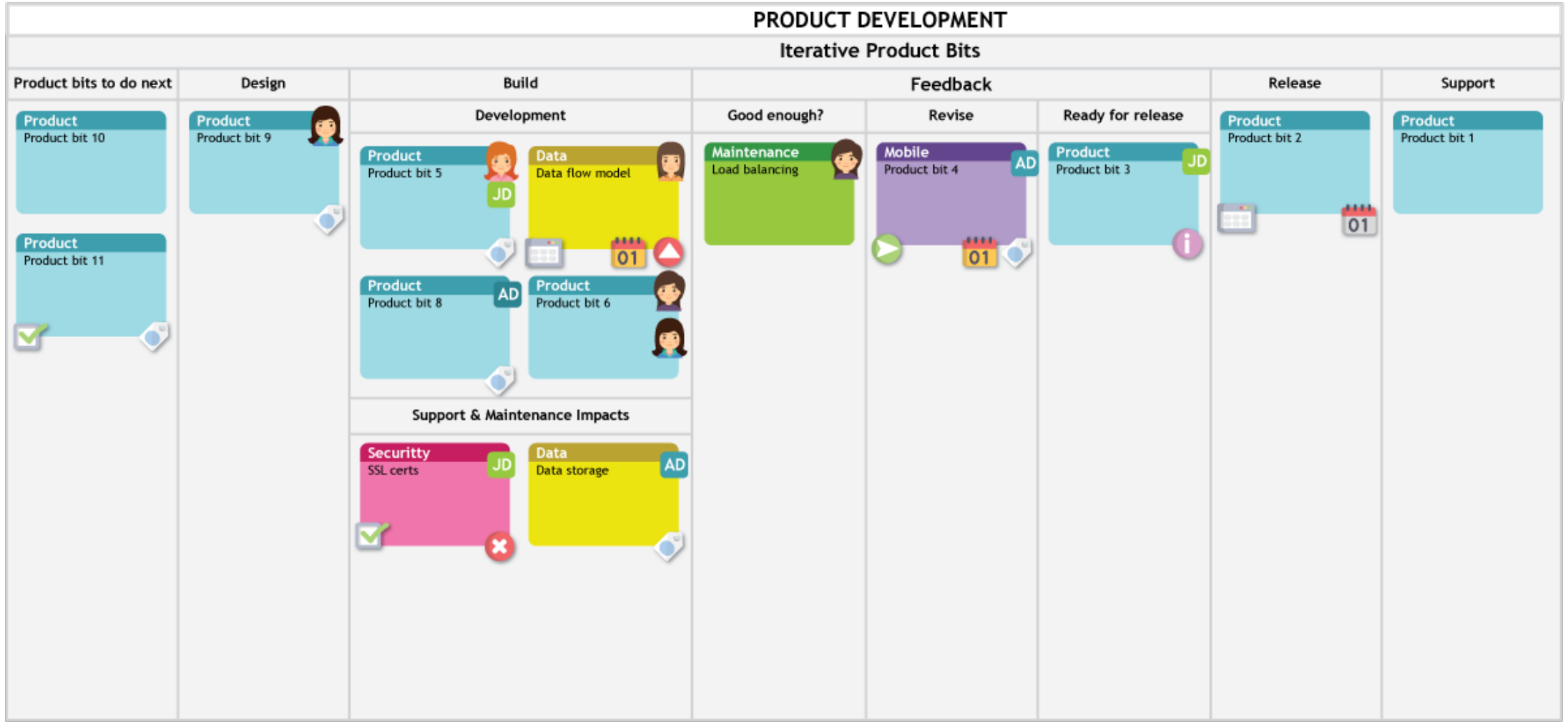
Beneficios clave de utilizar Kanban en equipos que practican DevOps:

1. Visualización del flujo de trabajo: Kanban ofrece una visualización clara y en tiempo real del flujo de trabajo del equipo, lo que facilita la colaboración y el seguimiento de tareas.

2. Eficiencia y productividad: Kanban elimina cuellos de botella y tiempos de espera, optimizando la eficiencia y mejorando la productividad del equipo.

3. Mejora continua: Kanban fomenta la mejora constante al analizar y medir el rendimiento, identificar áreas de mejora y tomar acciones basadas en datos.

Kanban para Equipos DevOps



Desarrollo

DEPC® Versión 072023

Ciclo de Vida del Desarrollo del Software

El ciclo de vida del software significa desarrollar el software, desde la etapa inicial hasta la etapa final.

El objetivo principal es:

Definir todas las fases intermedias necesarias para validar el desarrollo de la aplicación y asegurar que el software cumpla con los requisitos para la implementación y verificación de los procedimientos de desarrollo asegurando la utilización de métodos apropiados.

Al final de cada etapa se realiza una prueba para que se puedan arreglar las revisiones.

Desarrollo Agile del Software

No existe un enfoque universal para dirigir exitosamente cualquier proyecto de desarrollo de software.

Toda metodología debe adaptarse al contexto del proyecto (recursos técnicos y humanos, tiempo de desarrollo, tipo de sistema, etc).

Históricamente, los métodos tradicionales han tratado de abordar el mayor número de situaciones del proyecto, lo que requiere un esfuerzo considerable para adaptarse, especialmente en los pequeños y muy cambiantes requisitos de los proyectos.

Las metodologías **Agile** ofrecen una solución para casi todos los proyectos que tienen estas características.

Una de las cualidades más notables de una metodología **Agile** es su simplicidad lo que reduce los costos de implementación en un equipo de desarrollo.

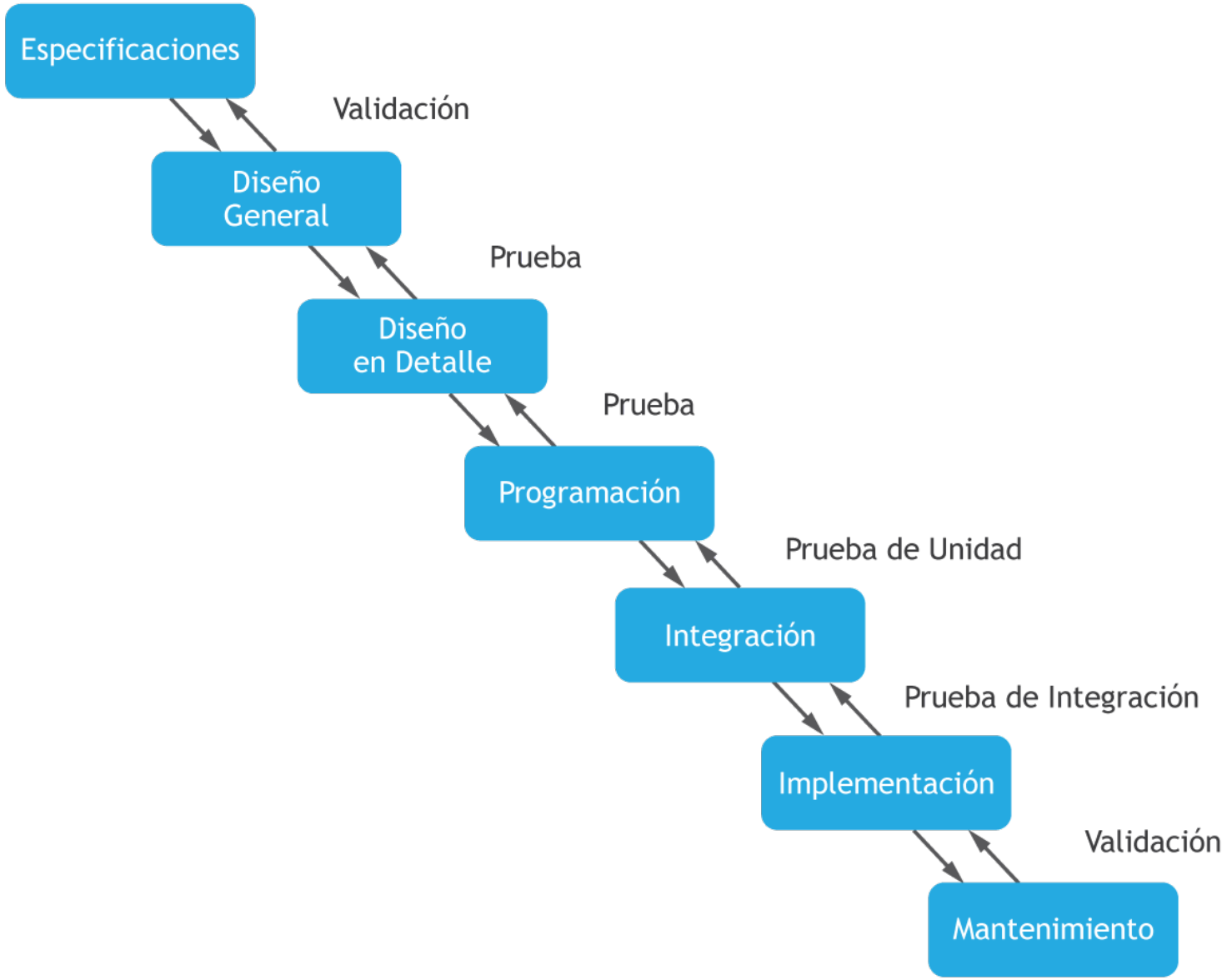
Desarrollo Agile del Software

La metodología **Agile** da mayor valor a lo individual, la colaboración con los clientes y el desarrollo de software de forma incremental mediante iteraciones cortas.

Este enfoque se ha encontrado eficaz en proyectos que tienen requisitos dinámicos y la necesidad de reducir drásticamente el tiempo de desarrollo, mientras mantiene la alta calidad.

Los enfoques de desarrollo **Agile** han revolucionado las formas de producir software. Ha generado un debate entre sus seguidores y escépticos como un enfoque alternativo a las metodologías tradicionales.

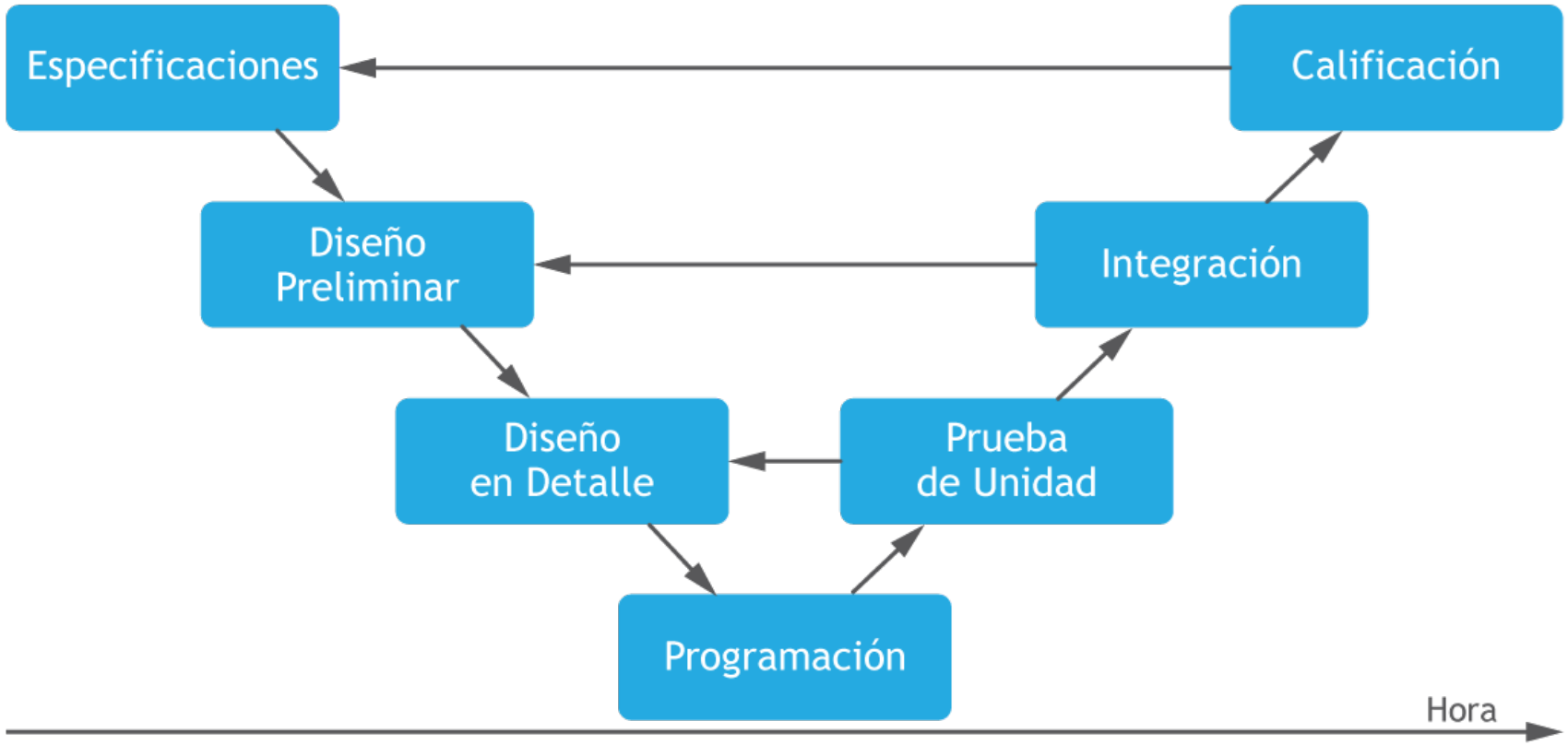
Modelo en Cascada



El modelo de ciclo de vida en cascada se comenzó a diseñar en 1966 y se terminó alrededor de 1970.

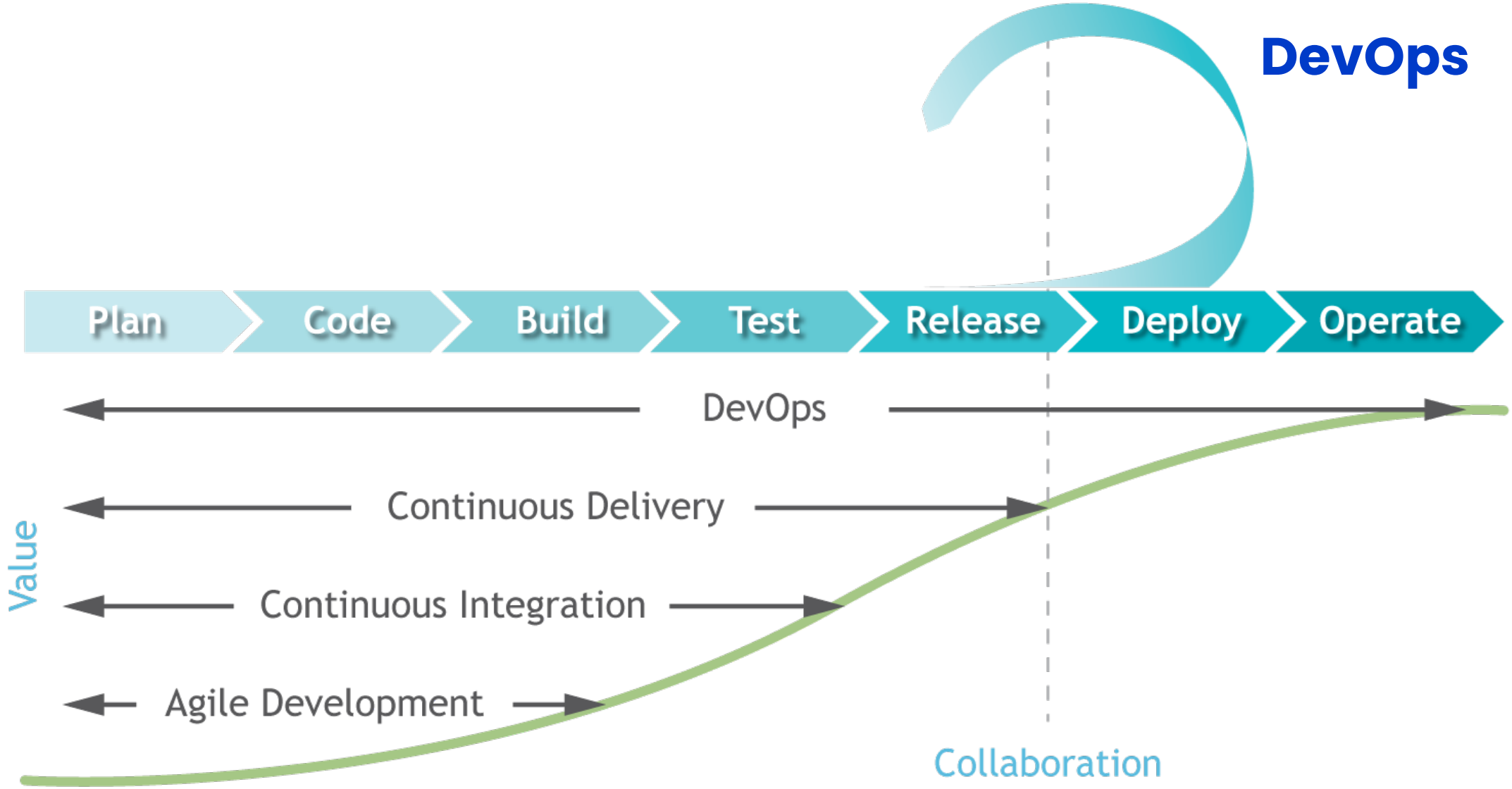
Se define como una secuencia de fases donde al final de cada una de ellas se reúne la documentación para garantizar que cumple las especificaciones y los requisitos antes de pasar a la fase siguiente.

Modelo V



El modelo de ciclo de vida **V** proviene del principio que establece que los procedimientos utilizados para probar si la aplicación cumple las especificaciones ya deben haberse creado en la fase de diseño.

DevOps


Plan
Code
Build
Test
Release
Deploy
Operate
DevOps
Continuous Delivery
Continuous Integration
Agile Development
Value
Collaboration

Integración Continua

DEPC® Versión 072023

Integración Continua

Cada pieza de código está integrada en el sistema una vez que el código está listo.

Los sistemas pueden ser integrados y construidos múltiples veces en un día.

Todas las pruebas se realizan y deben ser aprobadas para que el nuevo código se incorpore definitivamente.

La integración continua a menudo reduce la fragmentación de los esfuerzos de los desarrolladores.

El equipo de desarrollo está más preparado para modificar el código según sea necesario, porque les confiere la identificación y corrección para los errores de integración.

Integración Continua

Pequeñas entregas:

La idea es producir rápidamente versiones del sistema que estén operativas, aunque obviamente no tienen toda la funcionalidad prevista para el sistema, pero son un resultado de valor para el negocio.

Las entregas no deberían tomar más de 3 meses.

Monitorización:

El monitoreo de la carga provee retroalimentación al equipo en el proceso **XP**. Su responsabilidad es verificar el grado de éxito entre el tiempo estimado y el tiempo real empleado, comunicando los resultados para mejorar estimaciones futuras. También rastrea el proceso de cada iteración y evalúa si las metas son alcanzables dentro de las limitaciones de tiempo y los recursos presentados, también determina cuándo hacer cambios para lograr los objetivos de cada iteración.

Integración Continua

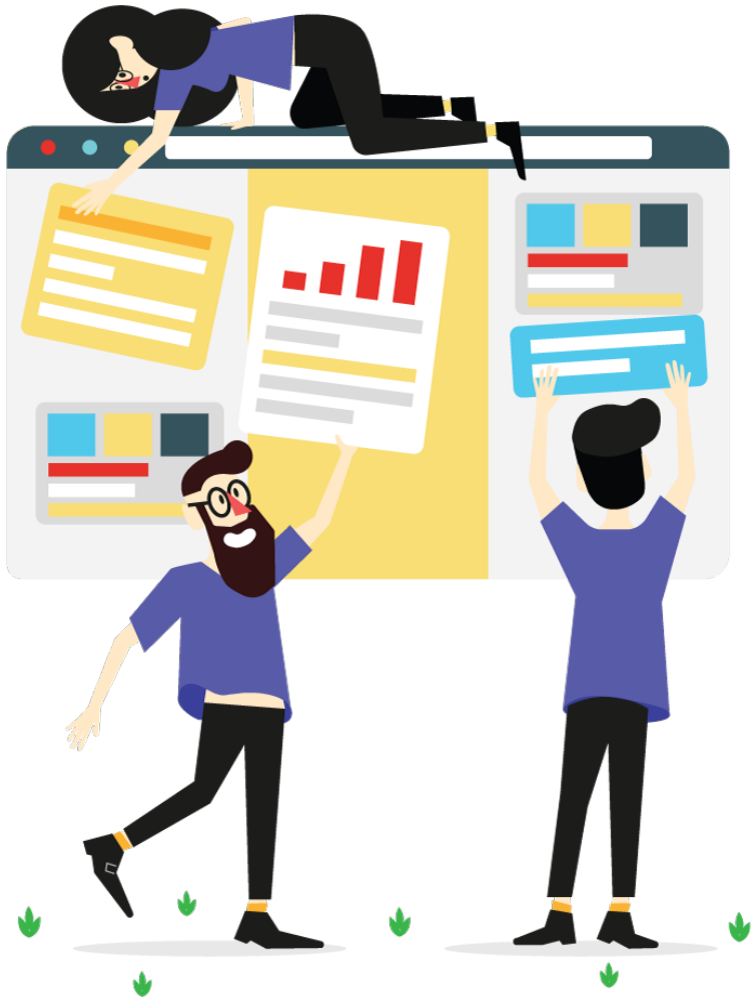
La **Integración Continua** o **CI** son prácticas originadas en el mundo de desarrollo de software, pero también pueden ser muy útiles para el equipo de operaciones.

Un buen ejemplo es un equipo de arquitectos trabajando en el plano de una casa. Cada día trabajan en la creación del diseño de manera independiente y cada viernes tratan de construir la estructura una vez que se han combinado todos los planos.

Este tipo de trabajo en equipo es bueno, pero ¿qué pasa si dos arquitectos trabajan en una misma parte de la estructura y elaboran dos diseños diferentes? Seguramente se producirá un problema.

Sin embargo, esto podría prevenirse, ¿Cómo? A través de la **Integración Continua.**

Integración Continua



Al implementar **CI** cada miembro del equipo deberá entregar diariamente cualquier cambio que realice.

Cada que se añada algo nuevo al plano, se requerirá escribir una prueba que verifique si es correcto lo que se haya agregado.

Cualquier cambio futuro que ajuste el diseño debe ser notificado inmediatamente.

Beneficios de la Integración Continua

- Mantener el repositorio de una sola fuente.
- Automatizar.
- Hacer que cada estructura se evalúe automáticamente.
- Pruebas en producción paralela.
- Automatizar la implementación.

El objetivo principal de la **Integración Continua** es que los miembros del equipo conozcan la necesidad del trabajo integrado. Las pruebas automatizadas pueden detectar errores siempre que un miembro del equipo trate de realizar un cambio.

Beneficios de la Integración Continua

CI requiere que los desarrolladores trabajen integrando códigos en un repositorio compartido varias veces en el día.

Cada chequeo pasará a ser verificado a través de la compilación automatizada.

CI permite a los equipos detectar rápidamente los problemas tan pronto como estos aparecen.

Con **CI** los errores son detectados de manera temprana.

Algunas compañías o equipos creen que es posible construir y entregar sin **CI**, pero hoy día puede ser un requerimiento.

Se puede creer que es posible desarrollar más rápido sin la implementación de **CI**.

Beneficios de la Integración Continua

Con proyectos en aumento y creciendo ni la compañía ni el equipo se harán más eficientes sin **CI**.

Con **CI** se detectan errores rápidamente, la confianza aumenta y esto lleva a una mayor eficiencia en la entrega del software.

Con la integración continua habrá menos **Back-tracking** que hacer para descubrir dónde se originó un error. Esto permite mayor tiempo para ser utilizado en la construcción de las características.

La **Integración Continua** es costo-eficiencia, es decir, es económico. Evitar la integración continua es costoso.

No seguir el enfoque continuo significa periodos de tiempo más largos entre integraciones, por lo que es exponencialmente más difícil encontrar los problemas y resolverlos.

Entrega Continua

DEPC® Versión 072023

Entrega Continua

Esto permite a los clientes utilizar el software y volver con comentarios.

La retroalimentación permite al desarrollador realizar mejoras con el software en desarrollo. Hay básicamente tres puntos de mejoras:

- El primero en ser mejorado es, por supuesto, la entrega del software.
- A continuación, el entorno al que se suministra el software también se puede mejorar para que sea posible mejorar la eficiencia y el rendimiento.
- El tercer factor a mejorar después de la entrega de la retroalimentación es en el proceso al que se está entregando el software.

Estas mejoras permiten al desarrollador de software ser más eficiente, más capaz y más rápido en lo que están entregando y esperamos, a un costo menor.

Entrega Continua

La entrega de software no es un simple proceso de entrega a la producción. Hay una serie completa de ciclos de entregas de software en múltiples entornos por los que tiene que pasar.

Se llama la línea de entrega. Estos ambientes son:

DEV: Comienza con el entorno de desarrollo.

BUILD: El proceso de construcción de software.

QA: También esta el entorno de control de calidad y por lo general hay más de uno en tales ambientes, cada uno para apoyar cada tipo de entorno que se utiliza para el software que se entrega.

Entrega Continua

Otros entornos de pre-producción o no-producción podrían incluir:

- **SIT** (Prueba de Integración del Sistema).
- **UAT** (Prueba de aceptación del usuario) Pre-prod y varios otros que podrían ser útiles en su entrega en función de cómo la organización está estructurada.

Entrega Continua

El entorno final en el ciclo de vida es el entorno de producción, donde se ejecuta el software, el cliente lo utiliza y donde el software realmente debe llegar.

El despliegue automatizado es la posibilidad de que el software se despliegue en cualquier entorno en un momento dado.

La entrega continua representa la capacidad de desplegar el software en cualquier entorno específico en un momento específico.

Aprendizaje Continuo

DEPC® Versión 072023

Aprendizaje Continuo

"El **Aprendizaje Continuo** a nivel de equipo es una profundización y ampliación de las capacidades del grupo en (re)estructuración para satisfacer las condiciones cambiantes, agregando nuevas habilidades y conocimientos y (re)creando en un sistema cada vez más sofisticado a través de la reflexión sobre acciones y consecuencias".

Valerie I. Sessa en su libro: Del aprendizaje continuo, perspectivas individuales, grupales y organizacionales.

Modelo CALMS

DEPC® Versión 072023

Modelo CALMS

Cultura	• Cambiar la manera en que pensamos y nos comportamos en la organización. • Convertirnos en uno. • Grassroots. • Cooperación.
Automatización	• Configurar Items. • Infraestructura como código.
Lean	• Enfocado en el valor y el cliente. • Reduciendo el tiempo gastado en actividades sin valor.
Medidas	• Medir todo el tiempo. • Mostrar mejoras.
Compartir (Sharing)	• Compartir. • Colaborar. • Transparencia.

DevOps, Otras Prácticas Recomendadas y Frameworks (Marcos)

DEPC® Versión 072023

DevOps y Agile

- **DevOps** con **Agile** es una combinación interesante. Siempre comienza con un usuario, el cliente, luego algún concepto para ser llevado al mercado. Esto se conoce como ciclo *concepto-a-efectivo*.

- Para conseguir esto del usuario al desarrollo, la gente desarrolla productos, generalmente que responden a los requisitos del cliente y a sus necesidades.

- Y a través de las operaciones, **Agile** te lleva del usuario al desarrollo y **DevOps** te lleva desde el desarrollo hasta las operaciones en las que tendrás algo que realmente puedas proporcionar a tus clientes.

DevOps y Agile

DevOps tiene varios componentes, algunos de ellos incluyen:

- Fuerte control de la fuente.
- Automatización (automatización del software).
- Pruebas tempranas y frecuentes.
- Los pequeños incrementos en la entrega.
- Mejoras continuas.

Equipos cohesivos: Significa trabajar en estrecha colaboración para producir valor, para sacar productos al mercado.

DevOps y Agile

Estos componentes pueden sonar familiar si usted es un practicante ágil.

Por ejemplo, en **XP/Agile Engineering Practices**, estos componentes ya existen:

- **Test Driven Development:** Similar a las pruebas tempranas y frecuentes.
- **Pequeñas Entregas:** Similar a la entrega de pequeños incrementos.
- **Integración Continua:** Similar a la automatización.
- **Equipo Completo:** Al igual que el equipo cohesivo o el trabajo en equipo que sucede entre los desarrolladores y los clientes.
- **Estándares de Codificación:** Similar al fuerte control de fuentes.

DevOps y Agile

Puede utilizar las prácticas de **Ingeniería Agile** junto con la capacidad de operar y puede terminar con **DevOps**.

Esta es la capacidad de desarrollar y operar productos y software de manera rápida para luego llevarlos al mercado.

Comienza con un concepto y el objetivo de convertirlo en efectivo.

Usted tiene el puente entre los usuarios y los desarrolladores, **Agile** y **DevOps** enlaza desarrolladores y operadores.

DevOps y Scrum

Scrum fue originalmente formulado para proyectos de desarrollo de software.

Se trata de un marco de trabajo **Agile** que permite completar más rápidamente proyectos complejos.

Sin embargo, con **Scrum**, las posibilidades son infinitas, se puede utilizar para cualquier ámbito innovador y proyecto/tarea compleja. Este marco es muy simple, pero sin duda debe estar trabajando en la creación de la cultura **DevOps**.

DevOps y Scrum

Al aprender a implementar las prácticas de **DevOps** junto con las prácticas de **Scrum**, es necesario decidir cuánto tiempo tomará cada iteración: se denominan **Sprints** en **Scrum**.

Cada **Sprint** es una representación del tiempo necesario para que el equipo desarrolle y luego pruebe el código. El equipo debe comprometerse a tener una aplicación ejecutable en cada conclusión del **Sprint**.

El **Sprint** de dos semanas es el más común para algunos equipos **Scrum**.

DevOps e ITSM (ITIL)

DevOps e **ITIL** se necesita mutuamente. ¿Por qué? Porque tienen funciones que benefician a ambos.

DevOps puede proporcionar:

- Trabajo colaborativo.
- Tiempos de despliegue rápido y continuo.
- Entrega más rápida de funciones.
- Enfoque en el trabajo importante.
- Estabilidad en el ambiente.

DevOps e ITSM (ITIL)

ITIL, por otro lado, puede proporcionar:

- Estructura con su ciclo de vida.
- Relaciones de negocio.
- Mejor calidad y fiabilidad de los servicios.

Aunque **DevOps**, por sí solo, es un proceso ya muy útil, se puede mejorar cuando une fuerzas con **ITIL**.

Con **ITIL** y **DevOps**, una organización disfrutará de más beneficios, como un alcance de servicios más vigoroso, una mejor perspectiva de las estrategias, mayores perspectivas sobre las mejoras, mejores perspectivas sobre la actividad de transición y los rigores de los procesos de diseño de servicios.

Si el tiempo lo permite se recomienda en la preparación de la certificación como **DevOps Essentials** hacer un análisis rápido del documento de **AXELOS** sobre **DevOps** e **ITIL** donde se ve una integración de las prácticas de **DevOps** con las fases de **ITIL**. Disponible para descarga el portal de **AXELOS**, www.axelos.com.

Fuente: https://www.axelos.com/resource-hub/white-paper/itil-and-devops-getting-started

DevOps e ITSM (ITIL)

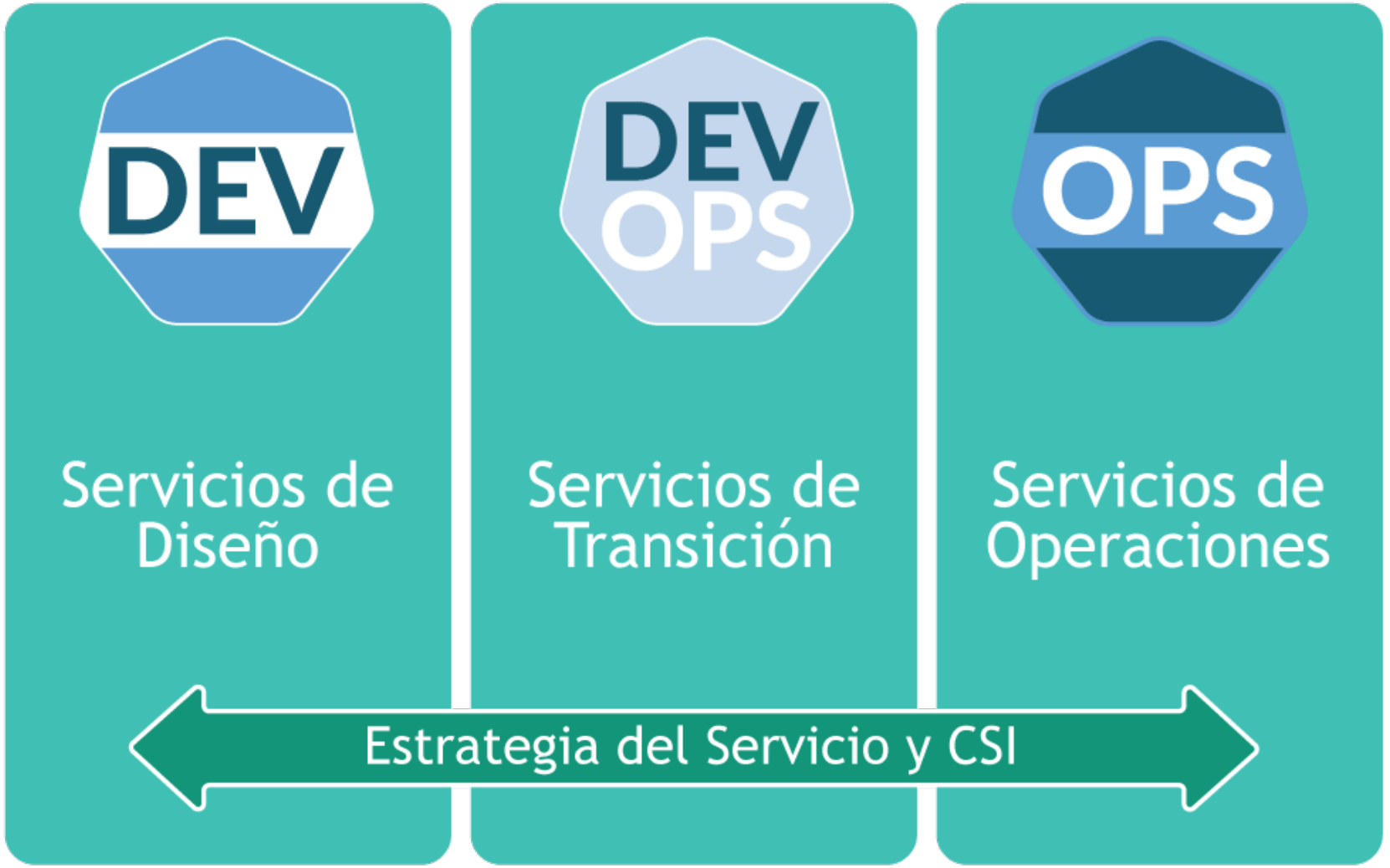
Otra percepción es que **ITIL** y **DevOps** no pueden trabajar juntos porque no son compatibles.

Siempre se ha considerado que la organización debe elegir uno y luego permanecer en ese carril. No es así cómo debería ser.

En realidad, hay más sinergias entre estos dos que diferencias. Sin embargo, muchas organizaciones no se han dado cuenta de esto.

Por lo tanto, están perdiendo mucho en las mejoras de servicio, que podrían introducir y desarrollar con sólo mirar cómo pueden aprovechar y equilibrar estos marcos.

DevOps e ITSM (ITIL)

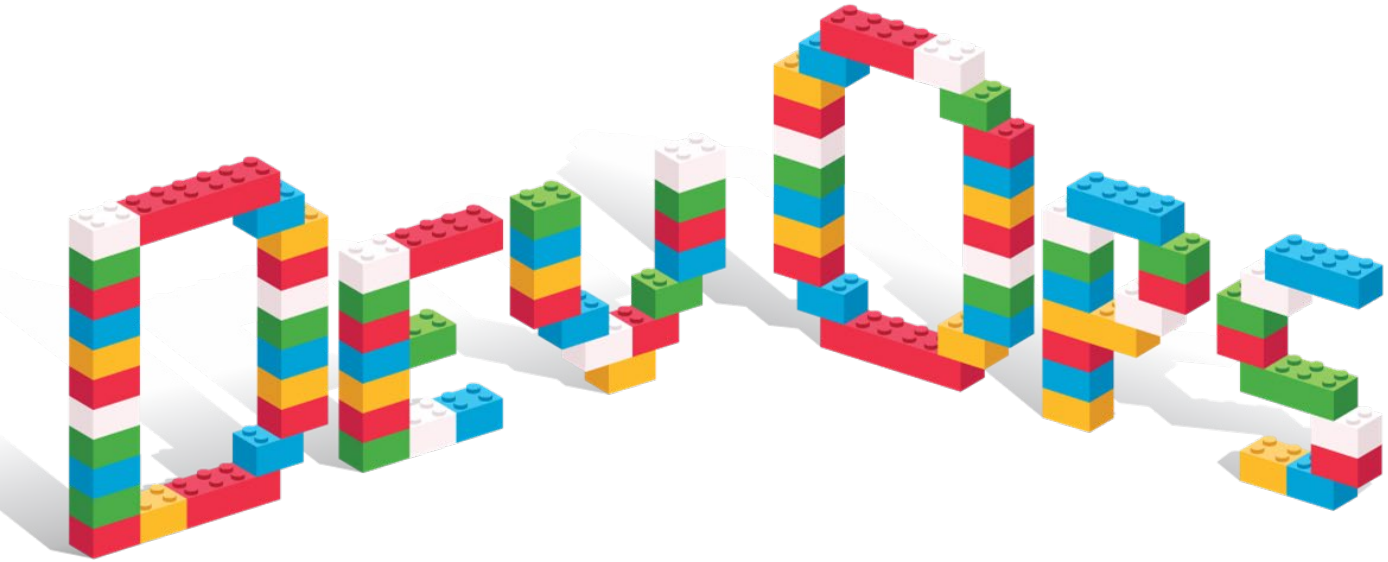


DevOps
industry architecture, figure 9.
"ITIL® is a (registered) Trade Mark of **AXELOS** Limited. All rights reserved
©Copyright **AXELOS**.

Taller
30 Minutos

DEPC® Versión 072023

Taller



A continuación se comparte un ejemplo de facilitación basada en el libro de **Dana Pylayeva**, con una duración máxima de 3 horas.

Se recomienda usar las filminas proporcionadas por la autora para dar contexto a la simulación de ambientes **Pre-DevOps**.

¿En qué consiste el juego?

- Compañía ficticia que desarrolla y produce el software.
- Software Funcional: Animal de LEGO.
- Chocolate: Documentación adjunta.

Equipos:

- Scrum.
- Grupo de **TI** (Administradores de Sistemas, Ingenieros de Seguridad e Ingenieros de Liberación).
- Representantes de la empresa.

Cultura DevOps

DEPC® Versión 072023

Cultura DevOps

El mundo de la tecnología es un entorno en constante cambio y por lo tanto, el desarrollo de software es justo como este. Entre estos cambios está la cultura **DevOps**.

No se puede negar que, con el tiempo, el impacto de **DevOps** ha crecido significativamente. Específicamente en el entorno de **TI** de ritmo rápido y competitivo de hoy.

La amplia disponibilidad de herramientas de programación, idiomas y servicios de software hizo posible crear opciones casi ilimitadas para los desarrolladores de software en la creación de aplicaciones innovadoras. Ser el desarrollador y el creador permite que uno se mantenga ágil.

Cultura DevOps

- Hoy en día, hay una delgada línea entre los roles que los desarrolladores hacen y ya no basta con tener una sola experiencia.

- Las pequeñas, medianas y grandes empresas ahora están adoptando esta nueva cultura **DevOps** para impulsar sus aplicaciones y programas hacia adelante y ser capaz de responder rápidamente a los cambios.

- En la construcción de **DevOps**, hay factores clave a tener en cuenta. En primer lugar, es importante ser generalista. Ser un experto en un campo (tecnología o software) simplemente no funciona más.

- Productos y empresas están cambiando con el tiempo, por lo tanto, es necesario tener la capacidad de saber cómo trabajar en múltiples áreas y crear un mejor valor.

Cultura DevOps

La integración continua también es esencial. Esto significa poder enviar código directamente a los usuarios. Este es un proceso de probar el código mientras está en la etapa de desarrollo.

Esto permite a los desarrolladores actuar de inmediato en cuanto se devuelven los comentarios. Los códigos probados continuamente son menos propensos a errores, y son más fáciles de liberar, también.

Otros factores incluyen el monitoreo continuo, el despliegue continuo (conseguir el nuevo código en el proceso de producción lo más rápido posible) y la resiliencia (la construcción de aplicaciones resilientes para que pueda responder a los errores de forma cuidadosa y exhaustiva).

Cultura Orientada en DevOps

- Alta cooperación (un solo equipo).
- Responsabilidades compartidas y riesgos.
- Entrenamiento continuo.
- Comunicadores recompensados.
- Falla = descubrimiento y mejora.
- La innovación es continua.
- Comprometido.

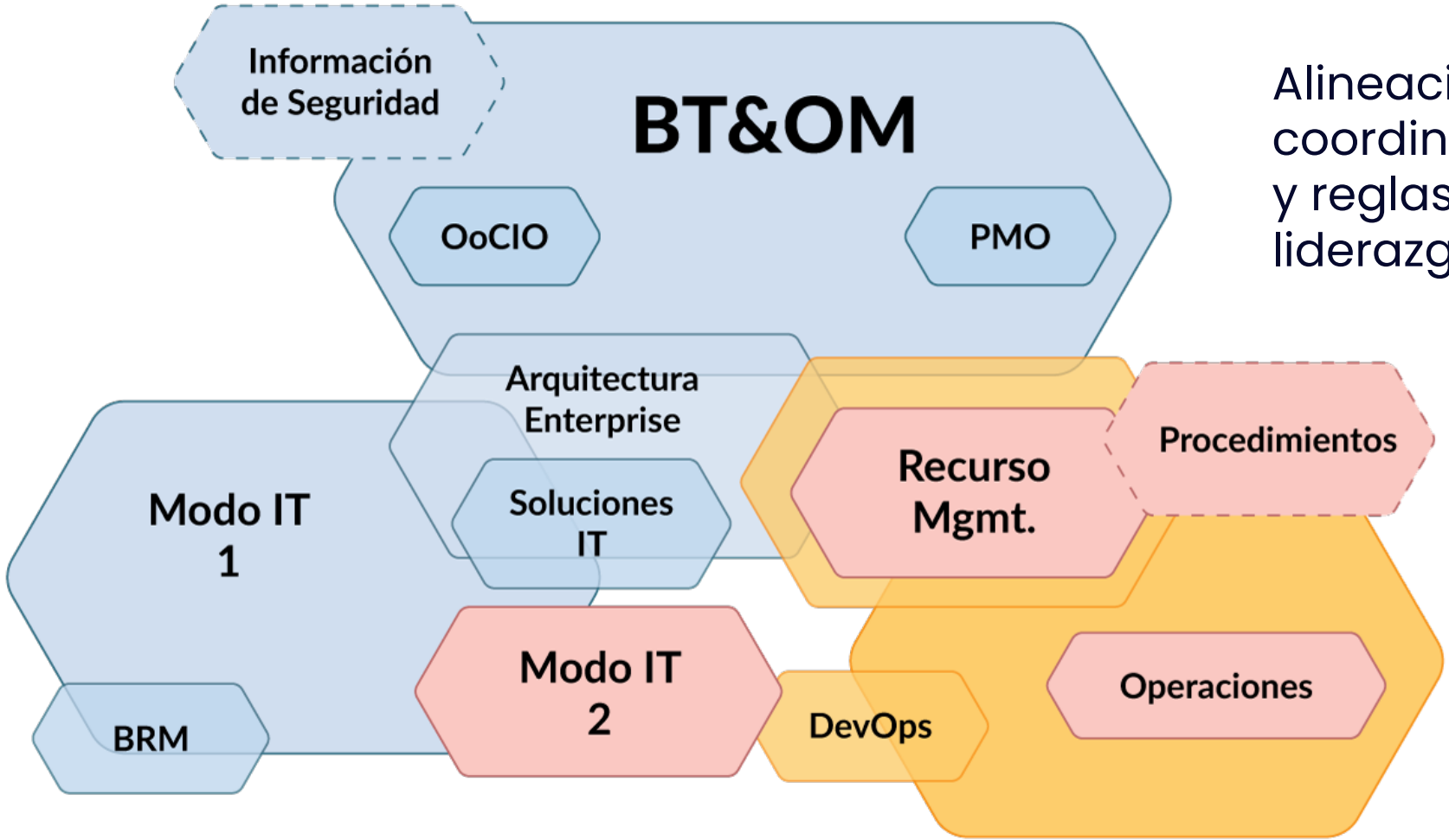
Equipo DevOps

DEPC® Versión 072023

Organización-Legado

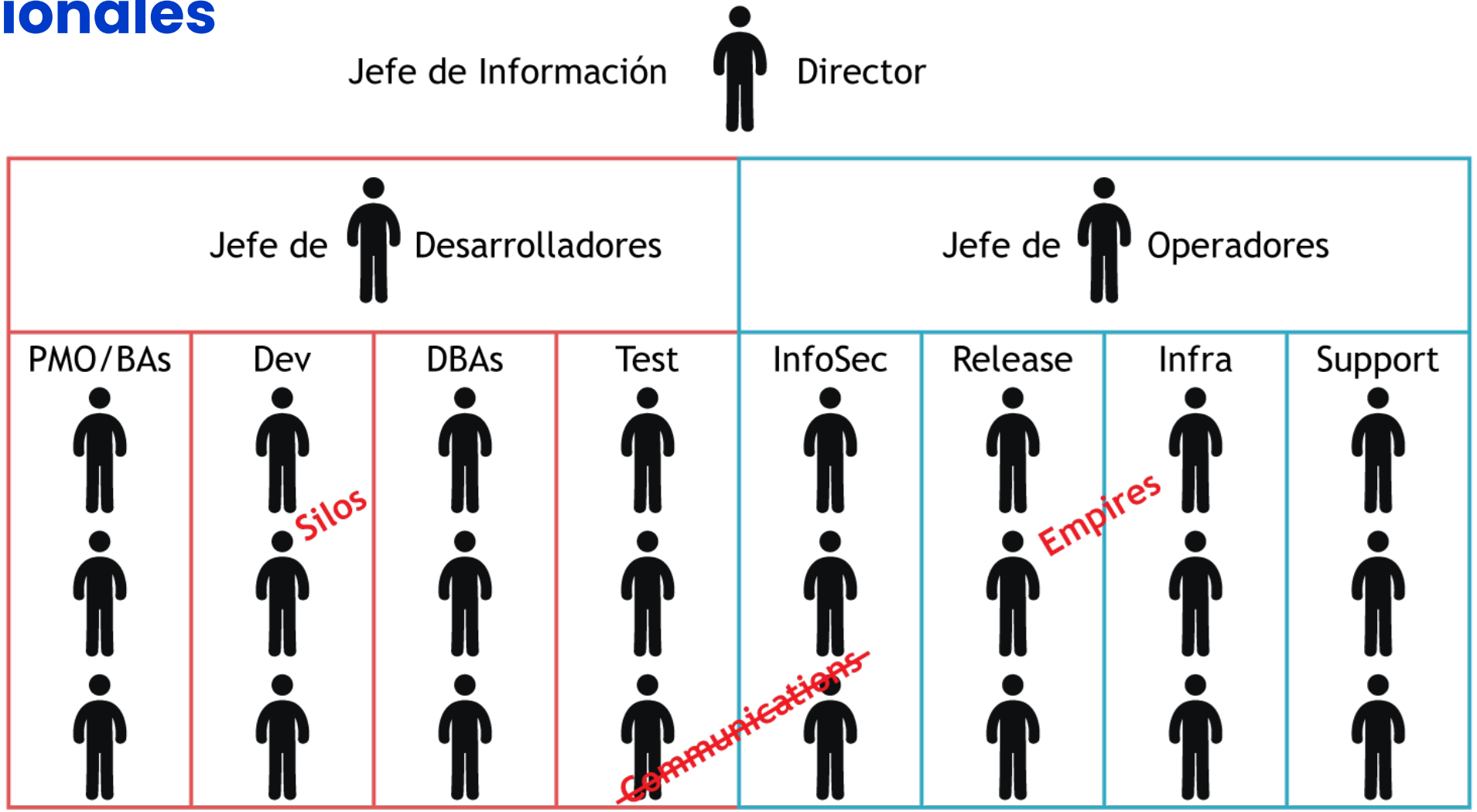
Rendimiento, optimización de recursos, estructura y líneas de reporte, comando y control.

Organización-Legado

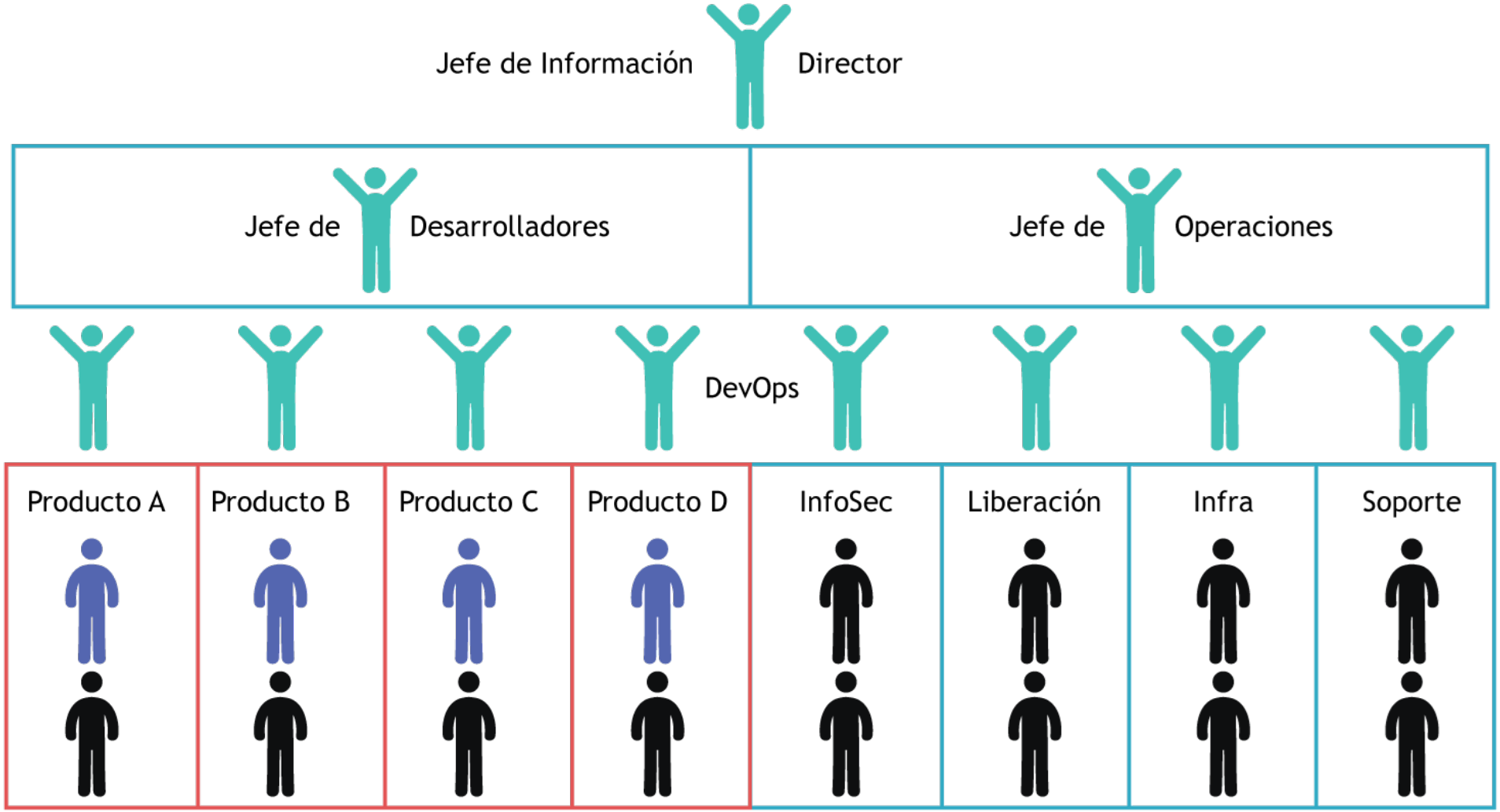


Alineación, integración, coordinación de procesos y reglas gubernamentales, liderazgo y colaboración.

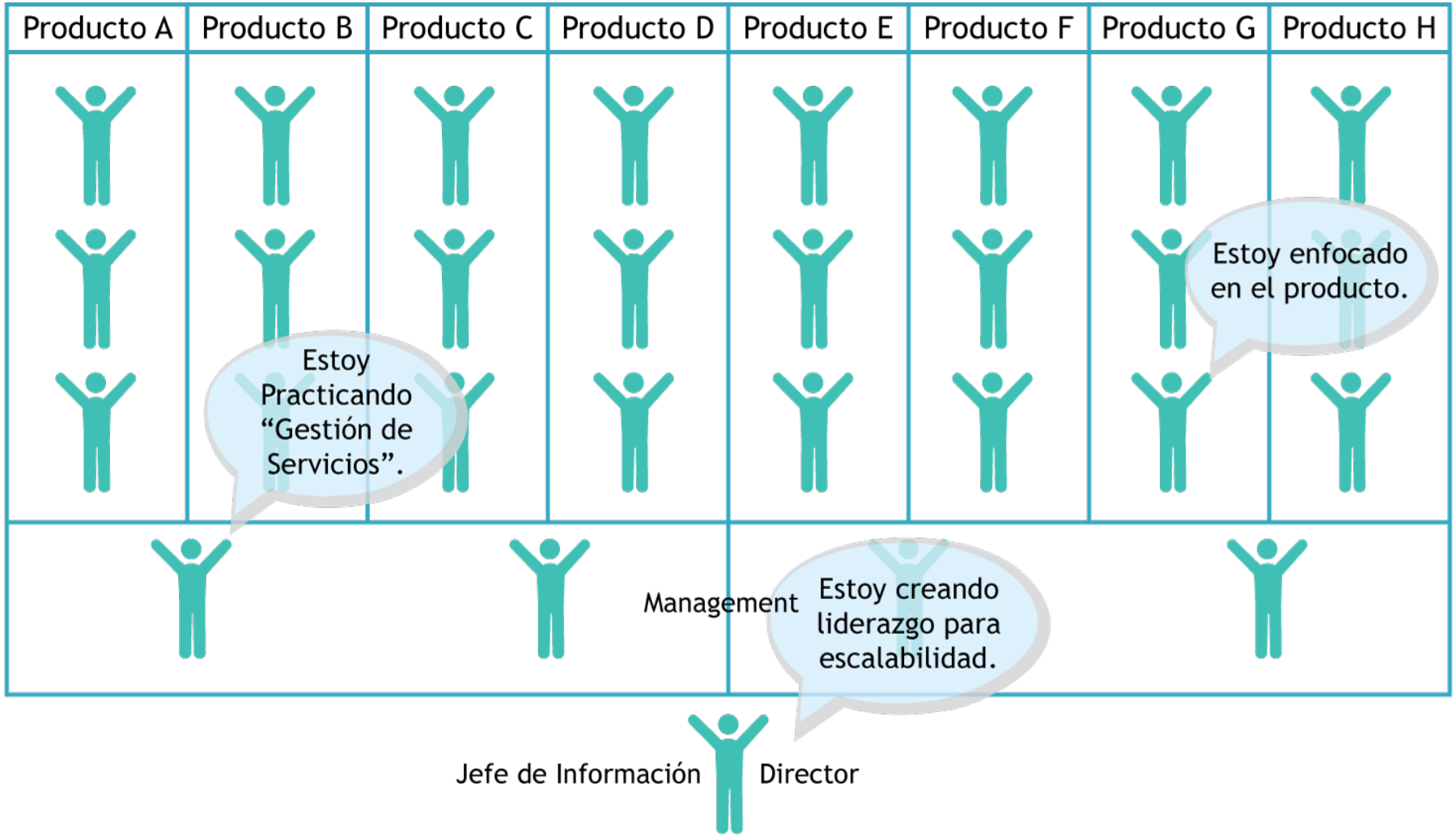
Operadores y Desarrolladores Tradicionales



Se Crean Equipos Dedicados de DevOps



DevOps Total



Roles de los Equipos

- Master de Procesos.
- Master de Servicios.
- Ingeniero de DevOps.
- Coordinador de Lanzamiento.
- Ingeniero de Confiabilidad.
- Equipo de Desarrollo.
- Equipo de Operación.

Fuente: Amazon’s “Two Pizzas Rule”.

Oficina de Gestión de Servicios SMO

- Organización plana para pequeñas organizaciones.
- Organización de matrices para organizaciones grandes y complejas.

Adopción Incremental

DEPC® Versión 072023

Adopción Incremental

Un enfoque alternativo para la adopción agresiva de **DevOps** es una adopción incremental; aunque esto puede tomar más tiempo, representará un riesgo significativamente menor y garantizará una implementación exitosa.

El enfoque comienza con la fijación de objetivos pequeños, como la implementación de **DevOps** en aplicaciones individuales mantenidas por un equipo pequeño de desarrolladores.

Una vez que esto se ha hecho con éxito, las lecciones aprendidas se pueden aplicar para objetivos cada vez más grandes hasta que finalmente hay un equilibrio saludable entre las mejoras del proceso y la ejecución del proceso.

El desafío clave sigue siendo encontrar un balance entre los complejos entornos de **TI** de la organización, la velocidad de implementación y el riesgo, que deben hacerse para que trabajen juntos.

Adopción Incremental

Por lo tanto, **DevOps** debe planearse para la mejora continua y no hacer la implementación de un evento de una sola vez.

Una estrategia incremental tiene que preferiblemente ser hecha en un acercamiento cíclico, escalonado que minimice riesgos y costes de la adopción de **DevOps** para construir los conjuntos de habilidades internas necesarias y el ímpetu para implementar ciclos más grandes y más rápidos.

Cada paso se construirá en el paso previo, en materia de inversión y preparación durante todo el ciclo global.

Las horas dedicadas al desarrollo serán realistas para la mayoría de las empresas y variarán ampliamente de acuerdo con la cultura y la política de una organización.

Adopción Incremental

Es probable que las reuniones de grupo consuman la mayor parte de las horas, con estimaciones basadas en tener pequeños equipos de implementación de 1 a 3 personas que interactúan continuamente con el personal clave de 2-4 personas de una aplicación.

System Thinking

DEPC® Versión 072023

System Thinking

Lo más importante es que las organizaciones de DevOps siempre piensan en términos de Sistemas. Sistemas significa conjuntos de procesos integrados que trabajan hacia objetivos comunes.

Para las organizaciones de DevOps, Sistema significa el negocio como un todo, no sus departamentos o equipos específicos.

Organizar actividades de confianza en el desarrollo del software implica procedimientos y procesos variados.

Los equipos de desarrolladores escriben el código y después los Testers los prueban y los operadores proporcionan la infraestructura para ejecutarlos y finalmente los clientes los consumen.

System Thinking

Puede haber muchas personas y procedimientos involucrados en este proceso.

El pensamiento de **Sistemas** significa que cada equipo debe ser consciente de las acciones de todos los demás equipos que trabajan en las líneas de desarrollo y en última instancia las entregas se realizan al cliente.

DevOps define cómo las acciones pueden afectar no solo al equipo sino a todo el **Sistema**. Esto significa que los desarrolladores podrían tener una mayor visibilidad en el ciclo de vida completo de las piezas del código que escriben.

Esto también significa tomar conciencia de las posibles ramificaciones de un cambio en lugar de olvidarse de las modificaciones después de que se hayan terminados los códigos.

System Thinking

También significa que los administradores de **Sistemas** deben entender completamente el impacto del rendimiento en las aplicaciones una vez que se liberan a los clientes.

El pensamiento sistemático forma un excelente enfoque para pensar también sobre la culpa.

Tradicionalmente, cuando alguien forzaba partes de código hacia la producción, que causaban interrupciones importantes del servicio, eran señalados, reprendidos e incluso potencialmente despedidos.

Un pensamiento sistemático ha sido capaz de eliminar estas inclinaciones iniciales para la culpa individual.

System Thinking

Se trata de no culpar a un individuo que fue lo suficientemente desafortunado al pulsar el botón, sino al sistema que permitió que esto sucediera.

El pensamiento sistemático tratará incidentes como éstos como fallas en el **Sistema**.

El equipo de **DevOps** investigará inmediatamente las causas; no por personas que hicieron esto, sino cómo esto podría producirse en absoluto. Se puede mirar en las causas de por qué las pruebas automatizadas no pudieron atrapar la falla que lleva al principio de aprendizaje y experimentación.

Experimentación y Aprendizaje

Las organizaciones **DevOps** siempre experimentan y aprenden de errores anteriores.

En un caso en que un desarrollador para el sistema, el incidente se investigaría a fondo reuniendo a personas apropiadas e incorporando arreglos para evitar recidivas futuras.

El incidente sería arreglado así que incluso si alguien hace la misma cosa otra vez, el sistema lo evitaría.

Experimentación y Aprendizaje

La gente aprende de sus errores y la organización permite a todos experimentar en lugar de obligarlos a centrarse en un conjunto restringido de tareas.

Las organizaciones de **DevOps** siempre están fomentando la creatividad y el pensamiento fuera de la caja en lugar de preservar los enfoques de la mentalidad de las viejas maneras.

Esto, naturalmente, animaría a todos a ser humildes, no hay ideas perfectas todas se podrían desafiar.

DevOps ha sido capaz de establecer precedentes en la experimentación siendo aceptable, incluso si terminan en los fracasos, ya que eventualmente los fracasos son las mejores maneras de aprender.

Feedback

Feedback

Sin ningún tipo de retroalimentación, el aprendizaje sería ineficiente, por lo que la implementación de **DevOps** siempre anima a diferentes tipos de retroalimentación.

La retroalimentación ocurre entre los pares en los mismos equipos y entre negocio en la forma de retroalimentación de los clientes.

DevOps mantiene los canales de comunicación abiertos para facilitar la retroalimentación y permitir una infiltración de los clientes a los niveles de desarrollo y operación.

DevOps alienta el desarrollo rápido y sus ciclos de retroalimentación no son diferentes.

Para proporcionar las características al cliente en el tiempo posible más corto es crítico que la retroalimentación se reciba lo más pronto posible.

Feedback

Los enfoques de **DevOps** han revertido los enfoques tradicionales y los clientes controlan el desarrollo a través de canales de retroalimentación abiertos.

Los clientes son capaces de proporcionar retroalimentación inmediata que es cribada y priorizada y se convierte en nuevos códigos que se aplican a la infraestructura para salir a los clientes de nuevo a través de un bucle de retroalimentación.

Es igualmente importante para la participación y la interacción de las personas que ofrecen la aplicación como los analistas de negocio, ingenieros de red, desarrolladores, probadores, etc.

Los riesgos de la producción se mitigan con la prueba de conceptos en múltiples fases y la superación de la resistencia al cambio.

Aunque un enfoque de proceso incremental lleva años, los resultados son siempre transformadores, el punto final es que **DevOps** es dinámico.

Herramientas para el Apoyo de la Cultura DevOps

DEPC® Versión 072023

Herramientas DevOps

El uso de herramientas no es hacer DevOps.

Hacer **DevOps** sin el uso de herramientas es complejo si entendemos que **DevOps** tiene componentes de automatización. **DevOps** es y será un tema cultural y no será tan solo el conjunto de herramientas.

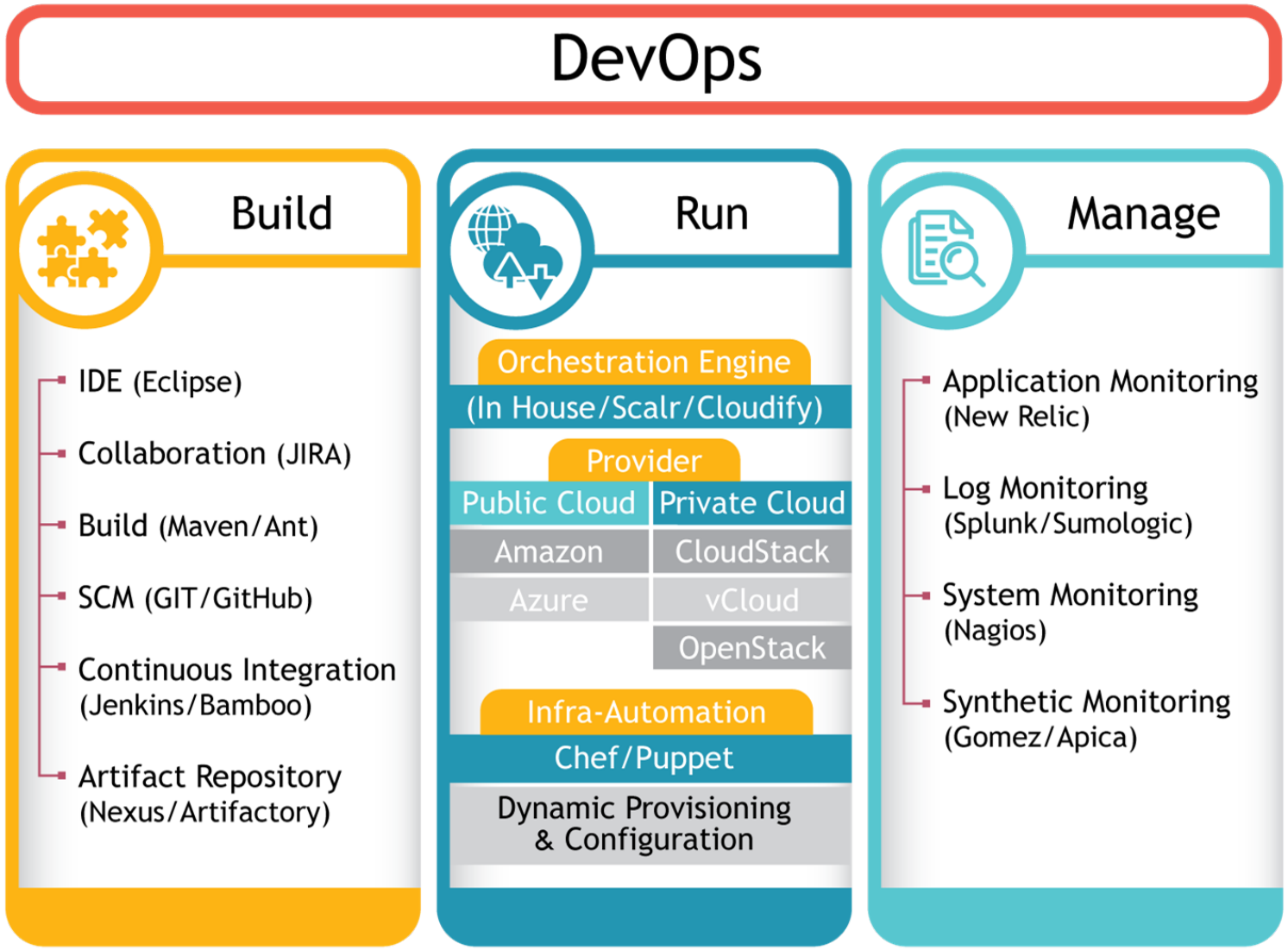
Miremos un ejemplo, las herramientas no hacen cultura, pensemos en una empresa que desea hacer que sus reuniones internas inicien a tiempo. Si la compañía compra un software de gestión de reuniones, ¿logrará que las reuniones inicien a tiempo?

Si la cultura interna es iniciar reuniones tarde, se deberá trabajar en cambiar la cultura interna primero. La herramienta por más buena que sea no logrará hacer que la cultura cambie.

¿Qué herramientas para DevOps o llamadas DevOps conocen los candidatos a la Certificación DevOps Essentials?

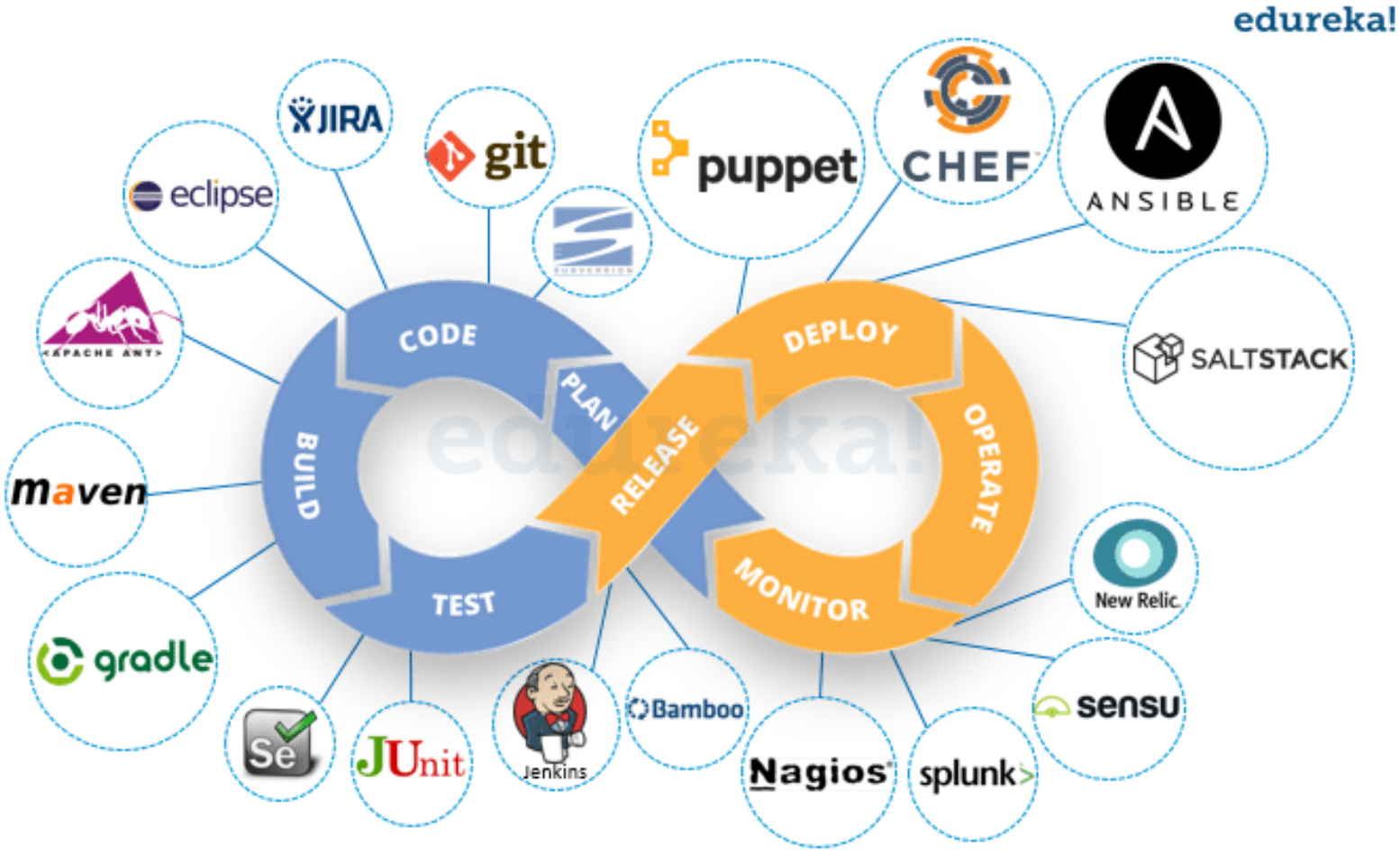
¿Qué hacen puntualmente esas herramientas?

Herramientas DevOps



Sin herramientas adecuadas, la fusión de roles y deberes de **DevOps** puede crear caos y resultados no deseados que incluyen problemas relacionados con escalabilidad, confiabilidad y administración de carga.

Cadena de herramientas de DevOps



Fuente: https://www.edureka.co/blog/devops-tools

Herramientas DevOps-GIT

GIT fue producido siguiendo los requerimientos de la comunidad **Linux** para **SCM**, es decir, el software **Source-Control-Management** que podría soportar sistemas distribuidos.

Esta es la herramienta más comúnmente utilizada para el manejo de fuentes que está disponible hoy en día.

Su ventaja consiste en tener grandes características adicionales en las solicitudes de bifurcación y tracción; Además **GitHub** también proporciona plugins que son capaces de conectar **Jenkins** y facilitar la integración y despliegue.

Las pruebas de velocidad que se ejecutan en una operación de red **GIT** generan resultados impresionantes, ya que los protocolos **GIT** para la transferencia de datos están altamente optimizados.

Sin embargo, con el tiempo, las implementaciones internas de **GIT** revelan limitaciones que pueden tener impactos en la velocidad y eficiencia de las tuberías de entrega.

Plataforma Nube

La automatización de la nube se compone de objetos de automatización discreta o tareas que realizan cosas como la hilatura de servidores de **VM**, la instalación de imágenes de **SO** y el despliegue de aplicaciones y funciones de red.

Las tareas en la automatización de la nube son realizadas por muchas herramientas de automatización como scripts y herramientas de gestión local para la gestión de la configuración.

La automatización de **DevOps** utiliza funciones de automatización discreta como herramientas de integración, compilación y administración continuas para configuraciones de software.

Las orquestas de **DevOps** son una coordinación automatizada por procesos de **DevOps** personalizados en las herramientas y tareas de organización y automatización que se están llevando a cabo en el proceso.

Plataforma Nube

La infraestructura de la nube juega junto con las herramientas de **DevOps** para la automatización y orquestación de despliegues de aplicaciones, además coordina el apoyo a los procesos deseados.

La orquestación va más allá de la simple tracción de la infraestructura de nube conjuntamente para abarcar tareas de automatización de **DevOps** en coordinación con la infraestructura para obtener la orquestación de **DevOps**.

Los desarrolladores que utilizan **Docker** crean entornos **Docker** en el portátil para desarrollar y probar localmente aplicaciones en los contenedores.

Dentro de este entorno se realizan ensayos en pilas de servicio compuestas con múltiples contenedores **Docker**.

Los múltiples contenedores de **Docker** convergen como pilas de un solo servicio, como las pilas **LAMP** y tardan segundos en crear una instancia.

Docker

Docker aporta portabilidad a las aplicaciones a través de la tecnología de contenedores, donde las aplicaciones pueden ejecutarse en unidades autónomas que se mueven a través de las plataformas.

Se compone de un motor **Docker** (una herramienta de tiempo de ejecución y de embalaje ligero, y un **Docker Hub**) que son servicios en la nube para el uso compartido y automatización de flujo de trabajo.

Docker ha formado una parte vital de la próxima generación de pruebas de **Yelp** e infraestructura para la gestión de servicios.

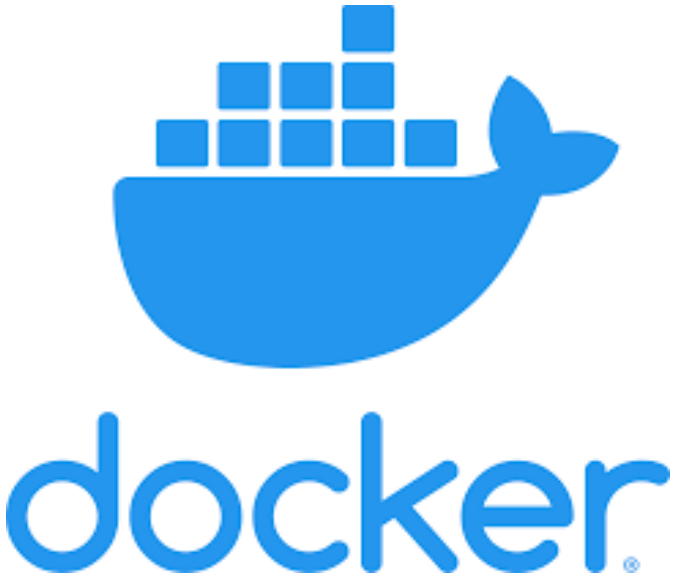
El aislamiento dependiente y los rápidos arranques de los contenedores permiten un ciclo de desarrollo más corto y aumentan las velocidades de prueba en más del 400 %.

Docker

Algunos entornos pueden ejecutar el host **Docker** dentro de otro host **Docker** (*Docker-in-Docker)* en los entornos de compilación.

Docker puede aumentar la velocidad de una tubería de **CI** utilizando **Union-FileSystems** y **Copy-on-Write** *(COW)*.

Para lograr velocidades incrementadas para dar **Entrega Continua** *(CD)* del software, se pueden usar muchas técnicas de **Docker**.



JS

JS permite la creación sencilla de aplicaciones de red rápidas y escalables.

Las plataformas **JS** tienen bibliotecas que permiten a las aplicaciones servir como servidores web sin necesidad de software como el **Apache-HTTP-Server** o el **Microsoft-IIS**, sin tener que hacer que los servidores web acorten las listas de tareas pendientes de **DevOps** y optimicen el flujo de código desde el desarrollo hasta el despliegue. Además, el **JavaScript** se puede utilizar tanto en cliente como en servidor.

Los desarrolladores encuentran una ventaja de eficiencia debido a la reutilización del código. Las aplicaciones **NODE.JS** pueden ejecutarse en tiempo de ejecución **NODE.JS** en **Microsoft Windows**, **OSX**, **Linux** y **FreeBSD**.

Ha sido diseñado para una operación rápida con gran experiencia del usuario. Un modelo de E/S sin bloqueo y controlado por eventos en **NODE.JS** permite aplicaciones ligeras que son altamente eficientes.

Muchas organizaciones líderes de pensamiento abarcaron **NODE.JS**, como **SAP**, **Groupon**, **LinkedIn**, **PayPal** y **Wal-Mart.**

Chef

Las herramientas **Chef DevOps** proporcionan marcos para automatizar y administrar la infraestructura a través de **DSL** simple.

El activo real es un código que trae los servidores y servicios que proporcionan vida.

Chef pone una confianza en sus definiciones reutilizables llamados libros de cocina y recetas escritas en **Ruby**.

Este nivel de abstracción aumenta la productividad y permite a los desarrolladores crear fácilmente configuraciones personalizadas para aplicaciones.

Chef

Las recetas se ejecutan de forma independiente mediante el uso de **CHEF SOLO** o un servidor que tenga **CHEF SERVER** que actúe como hubs para los datos de configuración.

Los servidores almacenan libros de cocina o políticas aplicadas a los nodos y los metadatos que describen el nodo registrado administrado por chef-client.

Chef es una herramienta de gestión multi-plataforma para **Windows**, **Linux**, **Mac OS**, etc., y puede integrarse con los principales proveedores de cloud.

Jenkins

Este es un motor para la integración extensible y continua que se utiliza como una de las principales herramientas de los ingenieros de **DevOps** que desean monitorear repetidas ejecuciones de trabajo.

Con **Jenkins**, el ingeniero de **DevOps** encuentra más fácil integrar los cambios en los proyectos y puede acceder a las salidas fácilmente cuando encuentran que algo va mal.

Jenkins

Las características clave son sus enlaces permanentes, la integración de **RSS**, correo electrónico/mensajería instantánea, un etiquetado después de los hechos, su informe de prueba **Junit/TestNG** y las compilaciones distribuidas.

Puppet

Puppet permite la gestión de la configuración y del software durante la realización de cambios rápidos y repetibles. **Puppet** impone automáticamente la coherencia en entornos y puede trabajar en máquinas físicas y virtuales.

Tiene cadenas de herramientas comunes y respalda las mejores prácticas clave en **DevOps** que incluyen la entrega continua.

Puppet Enterprise ofrece la orquestación de centros de datos mediante la automatización de la configuración y gestión de las máquinas y software.

También existen versiones de **Puppet** de fuente abierta disponibles que han sido utilizadas por la **Universidad de Stanford** para colmar las lagunas en el desarrollo de software para su servicio de biblioteca digital y administración del sistema, necesarias para mantener los servicios funcionando de manera segura.

Modelo Gartner DevOps

DEPC® Versión 072023

P.P.T y Cultura

1. Personas

2. Procesos

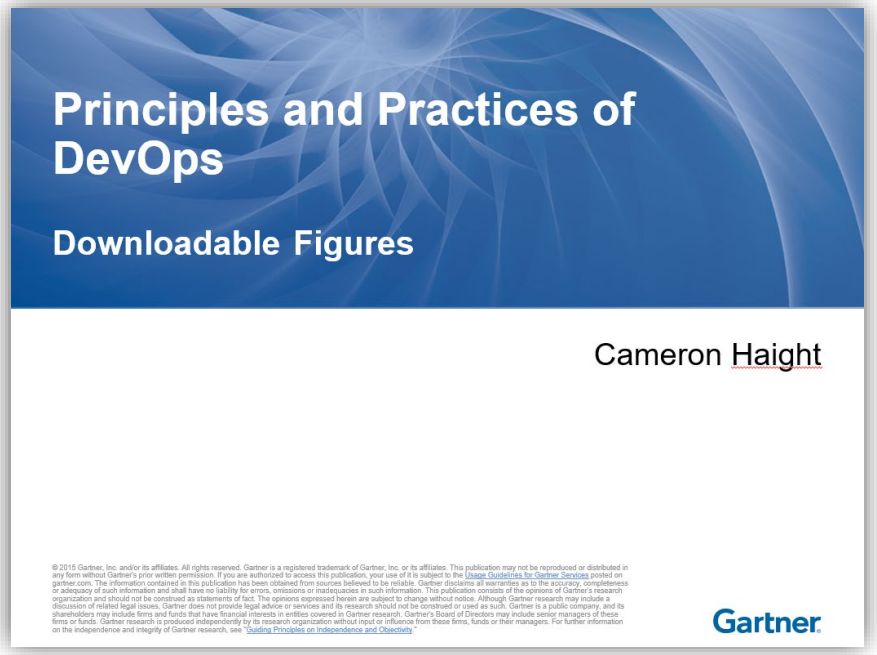
3. Tecnología

Modelo Gartner DevOps

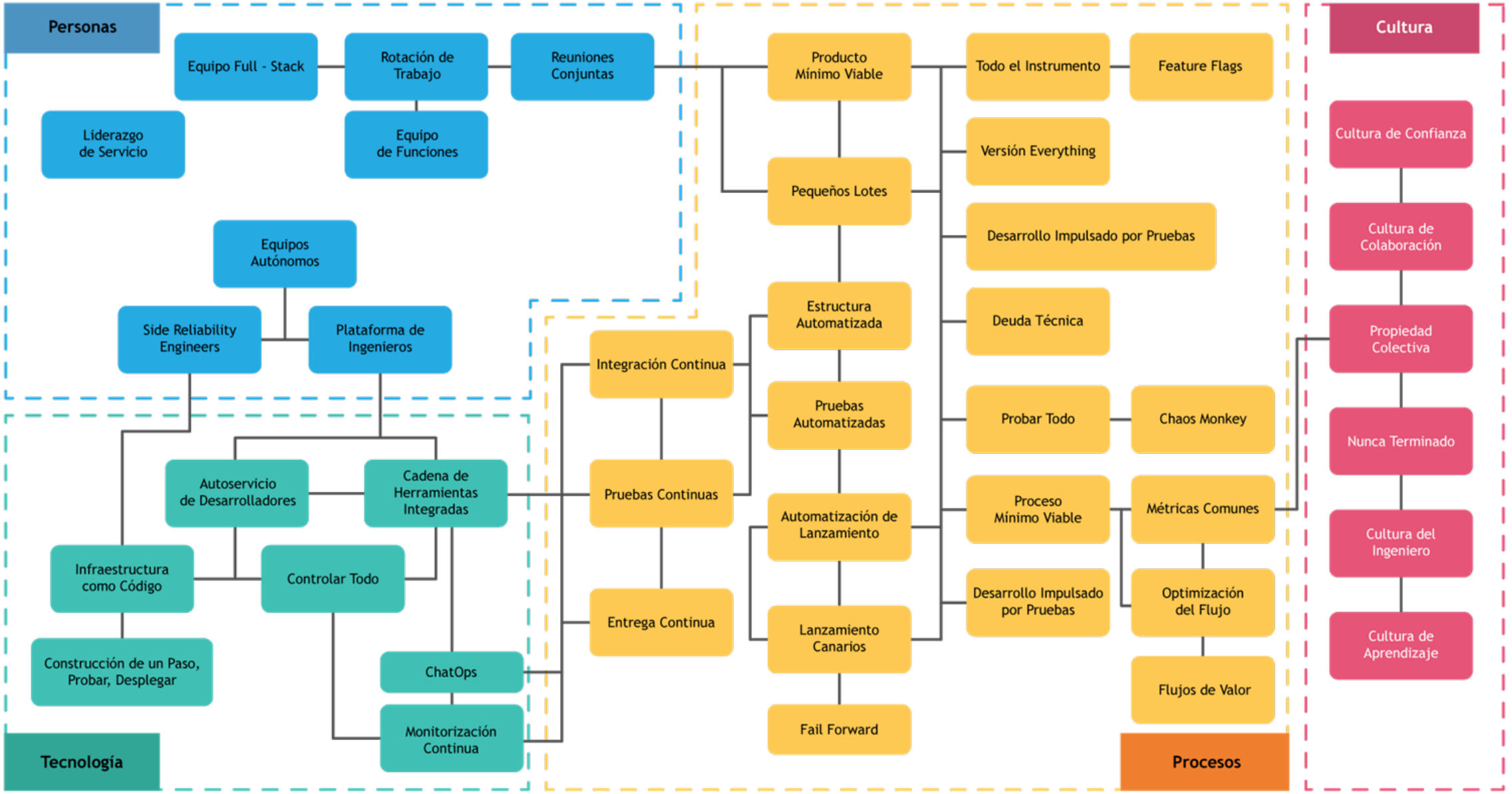
El siguiente slide presenta el modelo **DevOps** planteado por **Gartner Group**.

Los candidatos a la **Certificación de DevOps Essentials de CertiProf®** deben entender los componentes del modelo.

www.gartner.com



Modelo Gartner DevOps



Lista de Chequeo DevOps y Estado del Reporte DevOps

DEPC® Versión 072023

Modelo de Madurez DevOps

Este modelo mira **DevOps** desde tres puntos de vista: procesos, automatización y la colaboración. Se extiende por una serie de estados claramente definidos en el camino hacia un entorno **DevOps** optimizado.

La exploración de la madurez **DevOps** le da con una comprensión del nivel de madurez de tu organización en términos de estandarización de procesos, herramientas de automatización y los enfoques de colaboración, junto con conocimientos sobre tus oportunidades de mejora.

Lista de Chequeo - DevOps

Las listas de comprobación de **DevOps** no son estáticas o únicas y no tienen manifiestos que describan a **DevOps**, pero deben adaptarse a las necesidades de la organización, las interacciones humanas y otros criterios de naturaleza específica.

Las listas de verificación podrían ayudar a un procedimiento con la creación de culturas de **DevOps**, pero no se consideran como formas únicas de proceder con una transformación organizacional.

La configuración de la infraestructura de **DevOps** para las múltiples necesidades de negocio de una organización están enfocadas en procesos de continuidad del negocio y la recuperación de desastres que pueden afectar significativamente las empresas por los costos de tiempo de inactividad. También es necesario entender estas necesidades empresariales para hacer una estrategia de **DevOps** que puede tener impactos significativos en una organización.

Autoevaluación

En las empresas hay una necesidad de evaluación continua que constituye una piedra angular para el desarrollo escalable de la empresa **DevOps**.

- La autoevaluación capacitará a los equipos de desarrollo y operaciones para acelerar la entrega rápida y las iteraciones basadas en comentarios precisos y accionables sobre los productos.
- La evaluación de resultados e impactos constituye una disciplina importante y crítica que los equipos de **TI** deben adoptar durante la implementación de **DevOps**. Este proceso debe ser integral, automático y continuo.
- Sin una evaluación continua en un kit de herramientas de **DevOps**, existe el riesgo de pasar a ciegas y perder grandes oportunidades que pueden estar fácilmente al alcance.
- Esta información será crucial cuando los líderes empresariales y de **TI** prioricen su inversión futura y hagan concesiones que puedan surgir.

Autoevaluación

Microsoft | DevOps Self-Assessment
Helping you become a high-performer

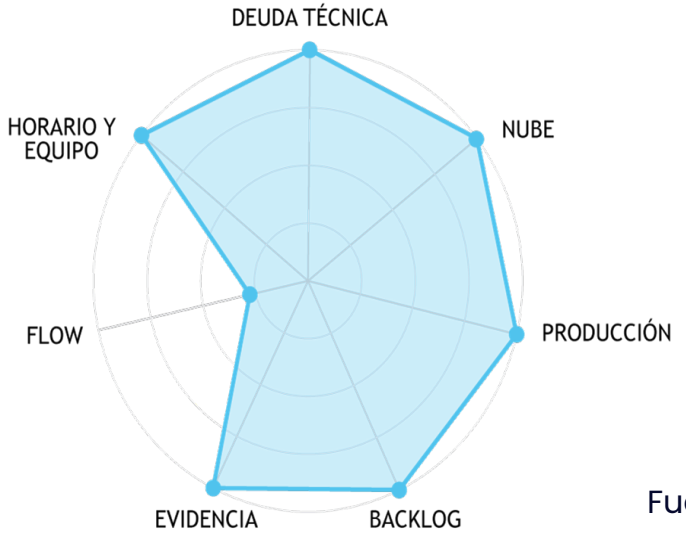
DevOps Self-Assessment

The ability to develop and deliver software is an important piece of any organization's ability to deliver value to customers, pivot when necessary, beat competitors to market, and respond to regulatory and compliance requirements. Delivering value with software often requires a technology transformation, and these transformations necessitate improving key capabilities.

The assessment has questions that touch on several key areas. These areas include:

- Process
- Technology and automation
- Culture
- Measurement
- Outcomes

Al decidir tu próximo paso a tu viaje por **DevOps**, deberías considerar el mercado competitivo, la cultura organizacional, procesos internos y actuales herramientas para fortalecer cada una de tus áreas de práctica de **DevOps**.



El resultado de esta autoevaluación debería ser usada para ayudar a optimizar tus prácticas y herramientas **DevOps**.

Fuente:
http://devopsassessment.azurewebsites.net/es-ES/

Autoevaluación

Las innovaciones rápidas en el negocio podrían actuar como ventajas competitivas.

La autoevaluación ayuda a determinar y medir qué características de **DevOps** son muy valiosas y se deben mejorar en la cultura empresarial.

Es costoso mantener y hace difícil la innovación, lo que ralentiza el equipo de desarrollo, por lo que es obligatorio un proceso de autoevaluación continua.

Reporte DevOps 2023

Es importante conocer cómo se está comportando **DevOps** en la industria Americana y en otros países.

El reporte **DevOps** es producido anualmente por **Puppet** y debería ser conocido y analizado por los Interesados a la **Certificación de DevOps Essentials de CertiProf®.**
El informe de este año explora nuevas áreas:

- La influencia que tiene el liderazgo en las transformaciones de DevOps.
- Cómo los equipos de alto y bajo rendimiento se automatizan de manera diferente.
- El impacto de la arquitectura y la estructura de equipo en el rendimiento de IT.

www.puppet.com

Taller 30 Minutos

Análisis del Estado DevOps de Puppet

https://www.puppet.com/resources/state-of-platform-engineering

DEPC® Versión 072023

Referencias

Literatura

The Phoenix Project

Gene Kim, Kevin Behr, George Spafford.
ISBN-10: 0988262576.
ISBN-13: 978-0988262577.
IT Revolution Press.
(10 de Enero, 2013).

The Phoenix Project: A Novel about IT, DevOps, and Helping Your Business Win Paperback – October 16, 2014

by Gene Kim (Author), Kevin Behr (Author), George Spafford (Author)

990 customer reviews

#1 Best Seller in Production & Operations

See all 9 formats and editions

Kindle	Hardcover	Paperback	Audible
$8.00	$29.95	$15.11	$16.95 or Free
Read with Our Free App	17 Used from $15.20 10 New from $29.95	19 Used from $11.04 51 New from $12.09	or Free with Audible 30-day free trial

Bill is an IT manager at Parts Unlimited. It's Tuesday morning and on his drive into the office, Bill gets a call from the CEO.

The company's new IT initiative, code named Phoenix Project, is critical to the future of Parts Unlimited, but the project is massively over budget and very late. The CEO wants Bill to report directly to him and fix the mess in ninety days or else Bill's entire department will be outsourced.

Referencias

- Jens Smeds, K. N. (2015). IEEE 11th International Conference on Global Software Engineering (ICGSE). Lecture Notes in Business Information Processing Agile Processes in Software Engineering and Extreme Programming , 166-177.
- Kort, W. D. (2016). Dashboards and Reporting. **DevOps** on the Microsoft Stack , 77-96.
- Kort, W. d. (2016). Integrating Testers into **DevOps**. **DevOps** on the Microsoft Stack , 205-229.
- Noureddin Kerzazi, B. A. (2016). Who needs release and devops engineers, and why? Proceedings of the International Workshop on Continuous Software Evolution and Delivery - CSED '16 .
- Stephen Jones, J. N. (2016). Management challenges for **DevOps** adoption within UK SMEs. Proceedings of the 2nd International Workshop on Quality-Aware **DevOps** - **QUDOS**.

¡Síguenos, ponte en contacto!

www.certiprof.com