



# DEVOPS ESSENTIALS PROFESSIONAL CERTIFICATE (DEPC)



**CertiProf®**  
Professional Knowledge

[www.certiprof.com](http://www.certiprof.com)

CERTIPROF® is a registered trademark of CertiProf, LLC in the United States and/or other countries.

## Objetivos

- Conocer la importancia de **DevOps** y aprender cómo aplicar esta metodología en diferentes proyectos.
- Certificación profesional.

## ¿Quién es CertiProf®?

CertiProf® ofrece una amplia gama de certificados profesionales para personas y empresas. Nuestra misión es preparar a los profesionales de la más alta calidad reconocidos a nivel internacional.

Con un equipo internacional que se especializa en la implementación de material, nuestro instituto es uno de los proveedores líderes que no solo brinda educación excepcional en el mercado de los EE.UU., sino que también se está expandiendo a las regiones de América Latina.

Potenciamos a las personas y las ayudamos a alcanzar su nivel óptimo al proporcionarles las herramientas y capacitación necesaria para aumentar su desempeño, habilidades y mejorar su desarrollo profesional.

## ¿Quién debe asistir a este taller de certificación?

Cualquier persona que esté interesada en ampliar sus conocimientos en **DevOps**.

## Agenda

<b>¿Qué es DevOps?</b>	<b>6</b>
¿Por qué DevOps?	7
Definiciones de DevOps	8
¿Qué no es DevOps?	8
Definición de DevOps Según sus Líderes	8
<b>Historia</b>	<b>11</b>
<b>Propósito de DevOps</b>	<b>14</b>
DevOps Adopción	15
<b>Beneficios</b>	<b>16</b>
Estabilidad	17
<b>Taller</b>	<b>21</b>
Análisis en Grupos	20
<b>Pilares de DevOps</b>	<b>22</b>
<b>Conceptos</b>	<b>23</b>
Just-in-time (JIT) o Justo a Tiempo	24
Sistema de Producción Toyota (SPT)	24
Kaizen	24
BIMODAL - Gartner	25
DevSecOps	26
Nuevo Modelo Operacional	27
Scrum	27
Work-in-Progress (WIP) (Trabajo en Proceso)	28
Acuerdo de Nivel de Servicio (SLA) y OLAS	28
Ciclo Plan-Hacer-Verificar-Actuar (Plan-Do-Check-Act Cycle/PDCA Cycle)	28
Definición de Listo (en Agile/Scrum)	29
Kanban para Equipos DevOps	29
<b>Desarrollo</b>	<b>31</b>
Ciclo de Vida del Desarrollo del Software	32
Desarrollo Agile del Software	32
Modelo en Cascada	33
Modelo V	33
DevOps	34
<b>Integración Continua</b>	<b>37</b>
Beneficios de la Integración Continua	37
<b>Entrega Continua</b>	<b>41</b>
<b>Aprendizaje Continuo</b>	<b>44</b>
<b>Modelo CALMS</b>	<b>46</b>
<b>DevOps, Otras Prácticas Recomendadas y Frameworks (Marcos)</b>	<b>48</b>
DevOps y Agile	47
DevOps y Scrum	48

DevOps e ITSM (ITIL)	48
<b>Taller</b>	<b>53</b>
<b>Cultura DevOps</b>	<b>55</b>
Cultura Orientada en DevOps	54
<b>Equipo DevOps</b>	<b>57</b>
Organización - Legado	56
Operadores y Desarrolladores Tradicionales	57
Se Crean Equipos Dedicados de DevOps	57
DevOps Total	58
Roles de los Equipos	58
Oficina de Gestión de Servicios SMO	58
<b>Adopción Incremental</b>	<b>61</b>
<b>System Thinking</b>	<b>63</b>
Experimentación y Aprendizaje	63
<b>Feedback</b>	<b>66</b>
<b>Herramientas para el Apoyo de la Cultura DevOps</b>	<b>68</b>
Herramientas DevOps - GIT	68
Plataforma Nube	68
Docker	69
JS	70
Chef	70
Jenkins	71
Puppet	71
<b>Modelo Gartner DevOps</b>	<b>74</b>
P.P.T y Cultura	73
<b>Lista de Chequeo DevOps y Estado del Reporte DevOps</b>	<b>77</b>
Modelo de Madurez DevOps	76
Lista de Chequeo - DevOps	76
Autoevaluación	76
Reporte DevOps 2017	78
<b>Taller</b>	<b>81</b>
Análisis del Estado DevOps de Puppet	80
<b>Referencias</b>	<b>83</b>
Literatura	82
Referencias	82
<b>Preguntas de Apoyo</b>	<b>85</b>



# ¿Qué es DevOps?

**CertiProf®**  
Professional Knowledge

[www.certiprof.com](http://www.certiprof.com)

CERTIPROF® is a registered trademark of CertiProf, LLC in the United States and/or other countries.

## ¿Qué es DevOps?

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



**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 sin 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”.

Es fundamental tener las herramientas adecuadas y combinar servicios, **DevOps** se preocupa por si 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.

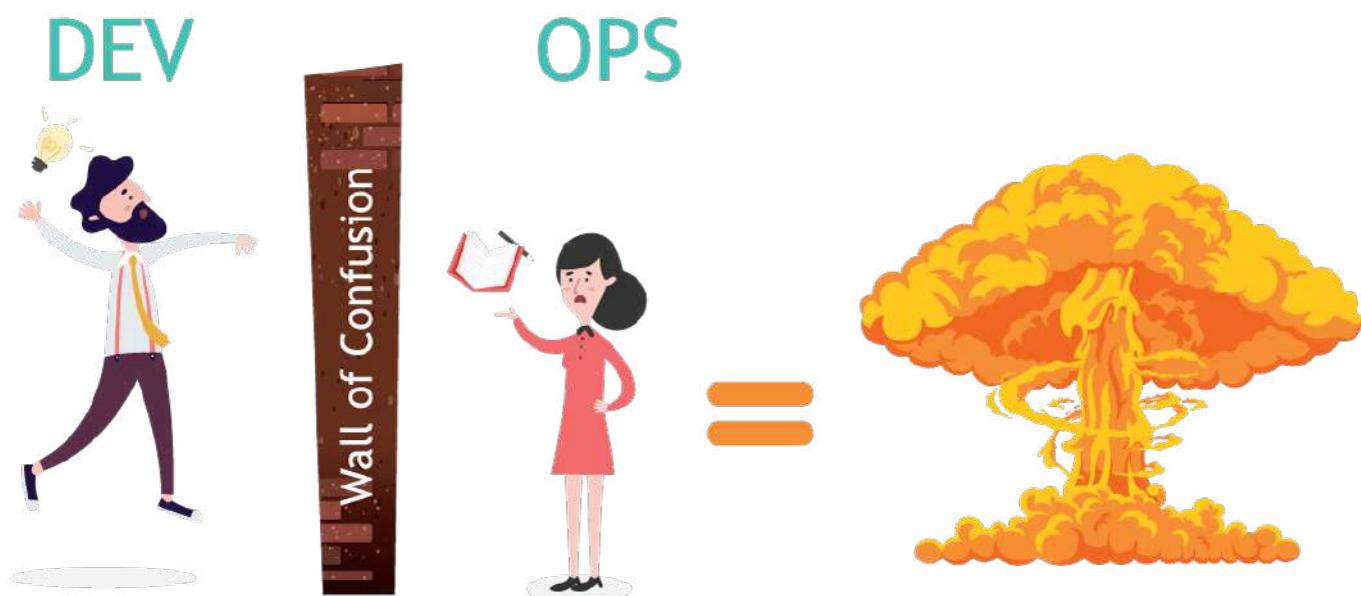
No existe una sola herramienta de **DevOps** que trabaje en la colaboración e integración entre los equipos de desarrolladores, testers y operaciones.

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.

**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.**



*“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.**



# Historia

**CertiProf®**  
Professional Knowledge

[www.certiprof.com](http://www.certiprof.com)

CERTIPROF® is a registered trademark of CertiProf, LLC in the United States and/or other countries.

## Historia

### Año 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.

### Año 2009

**John Allspaw** y **Paul Hammond**, dos empleados en jefe en **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.

### Año 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.

### Año 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.

### Año 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**.

### Año 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.

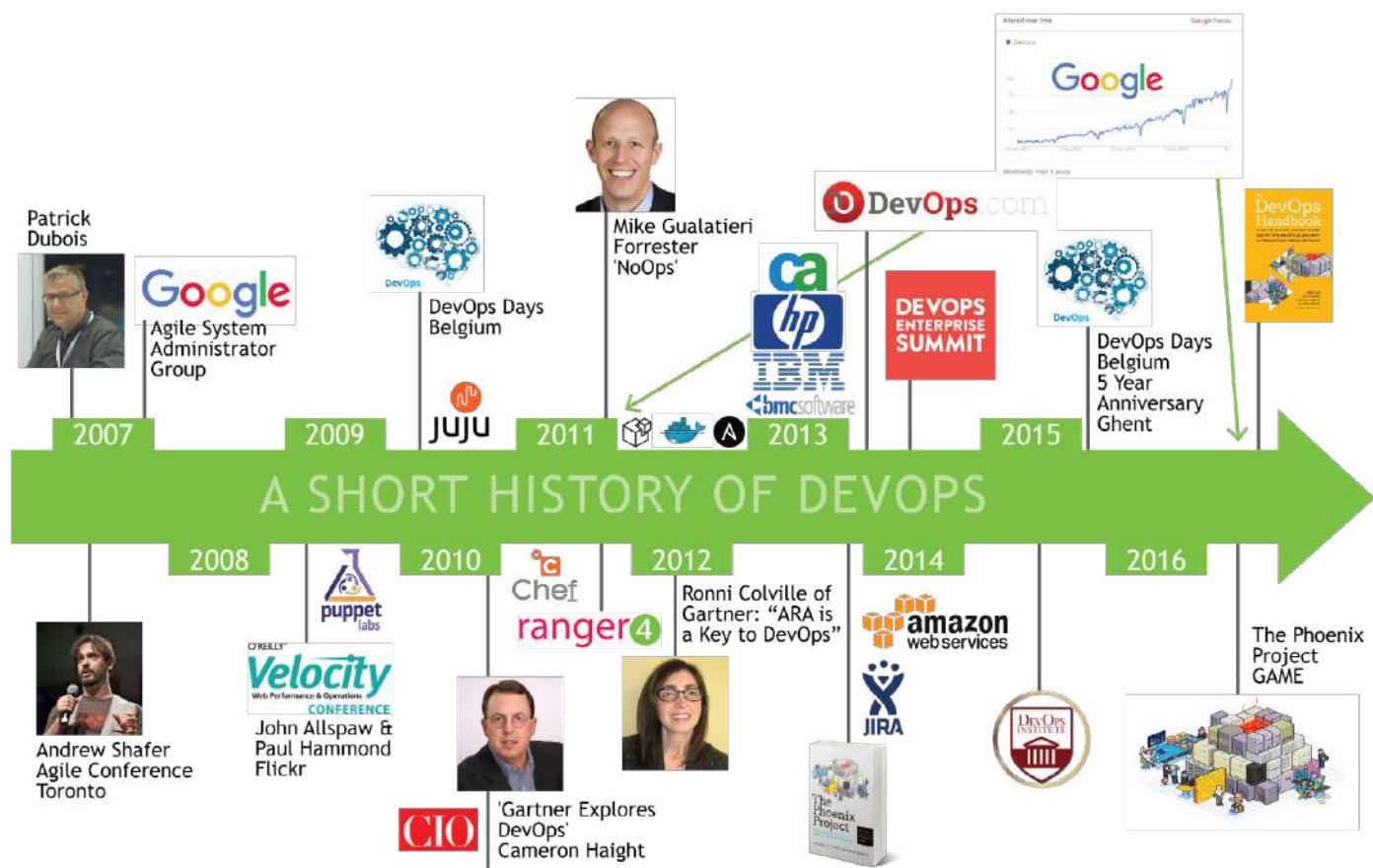
## Año 2014

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.

## Año 2015+

**DevOps** es presente y futuro...





# Propósito de DevOps

**CertiProf®**  
Professional Knowledge

[www.certiprof.com](http://www.certiprof.com)

CERTIPROF® is a registered trademark of CertiProf, LLC in the United States and/or other countries.

## Propósito de DevOps

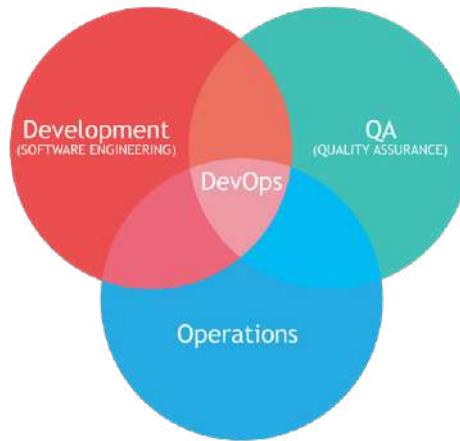
El mejor propósito ofrecido por **DevOps**, es iterar de manera más rápida durante la fase de desarrollo.

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).

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.

- **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.

### Algunos modelos planteados.

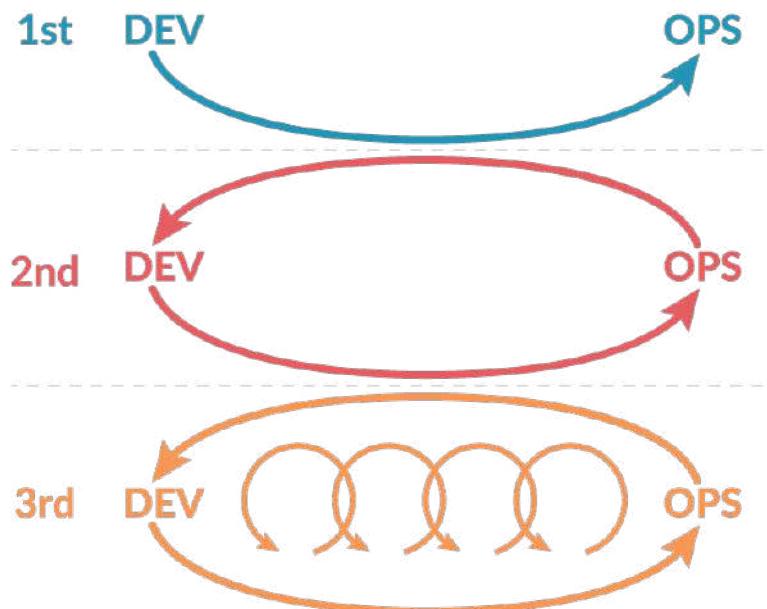
- DevOps Implementation Framework (DIF).
- DevOps Roadmap.
- DevOps Journey.
- DevOps Process.

## 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

**CertiProf®**  
Professional Knowledge

[www.certiprof.com](http://www.certiprof.com)

CERTIPROF® is a registered trademark of CertiProf, LLC in the United States and/or other countries.

## 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 y comunicarse y colaborar con otros equipos de **TI** en el entorno dinámico actual.

**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.

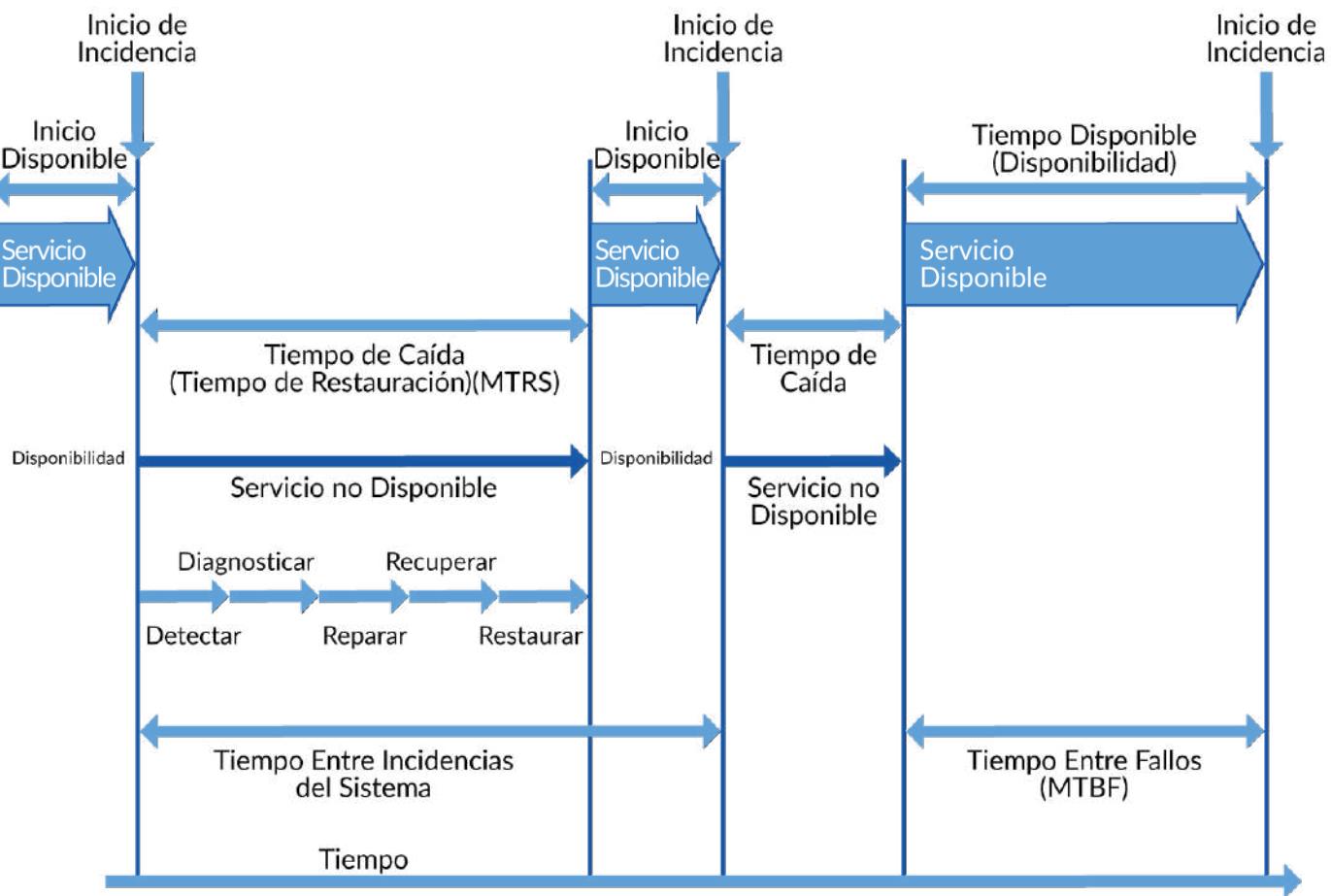
El desarrollo de software actual requiere que los equipos se involucren en la entrega continua sin fallas, en un período reducido para los marcos de tiempo de salida al mercado y ciclos de lanzamiento más cortos.

## 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.





# Taller



Tiempo Estimado:  
30 Minutos

**CertiProf®**  
Professional Knowledge

[www.certiprof.com](http://www.certiprof.com)

CERTIPROF® is a registered trademark of CertiProf, LLC in the United States and/or other countries.

## Análisis en Grupos

Informe State of DevOps 201x





# Pilares de DevOps

**CertiProf®**  
Professional Knowledge

[www.certiprof.com](http://www.certiprof.com)

CERTIPROF® is a registered trademark of CertiProf, LLC in the United States and/or other countries.

## 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

**CertiProf®**  
Professional Knowledge

[www.certiprof.com](http://www.certiprof.com)

CERTIPROF® is a registered trademark of CertiProf, LLC in the United States and/or other countries.

## 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 desarrolló 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 principales de **SPT** son mencionados en el libro "*La Manera de Toyota*".

## Kaizen

**Kaizen**, 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.

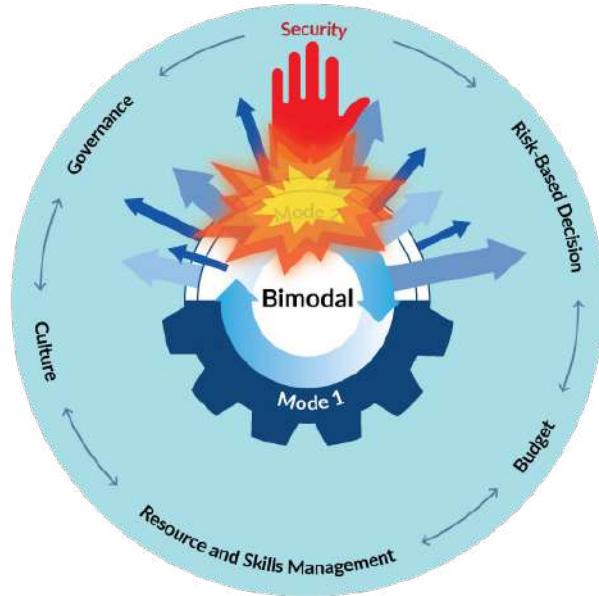
## BIMODAL - Gartner

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>



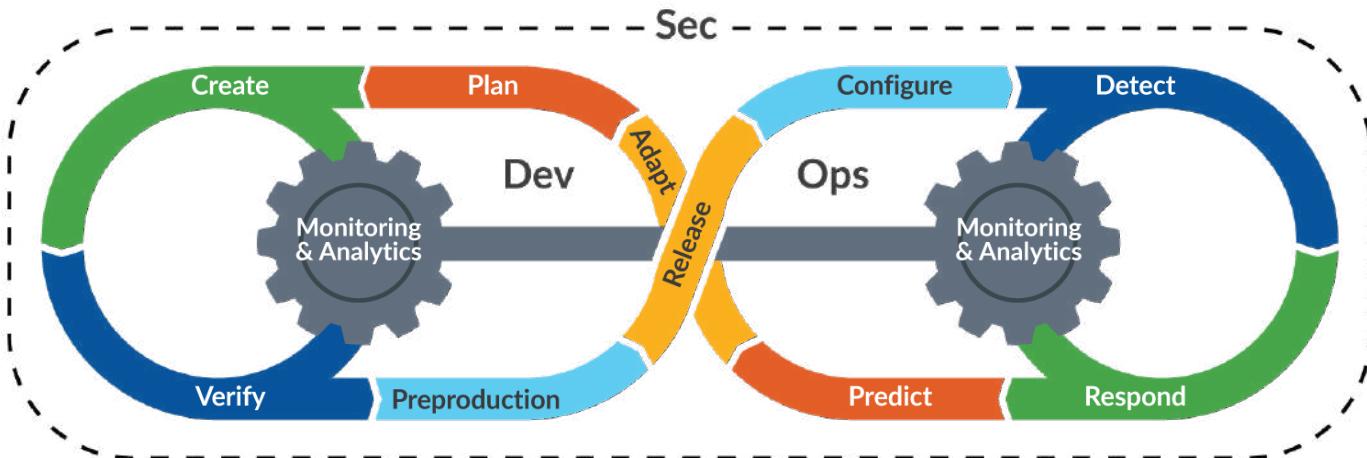
**IT Bimodal:** Dos enfoques distintos pero coherentes, bastante diferentes, ambos esenciales.

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

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

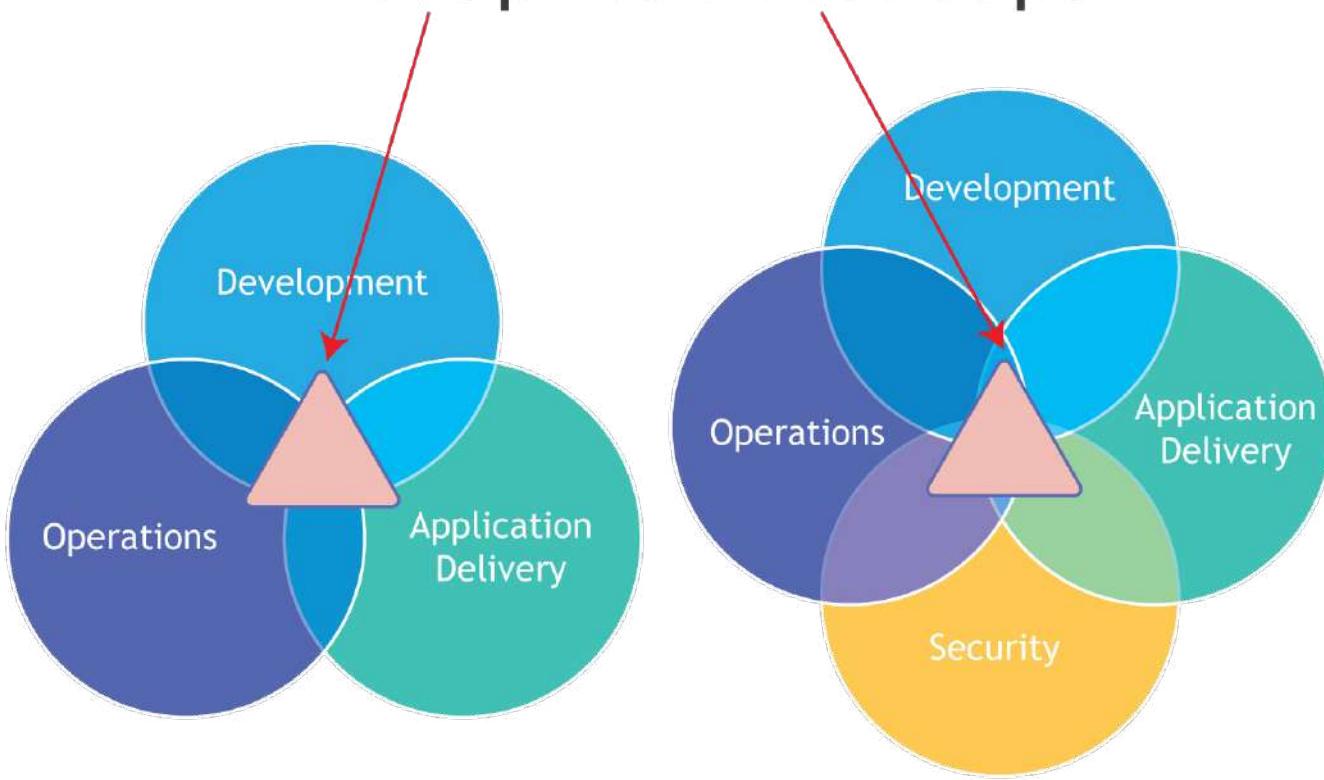
## DevSecOps

DevOps tiene una relación nada fácil con seguridad.



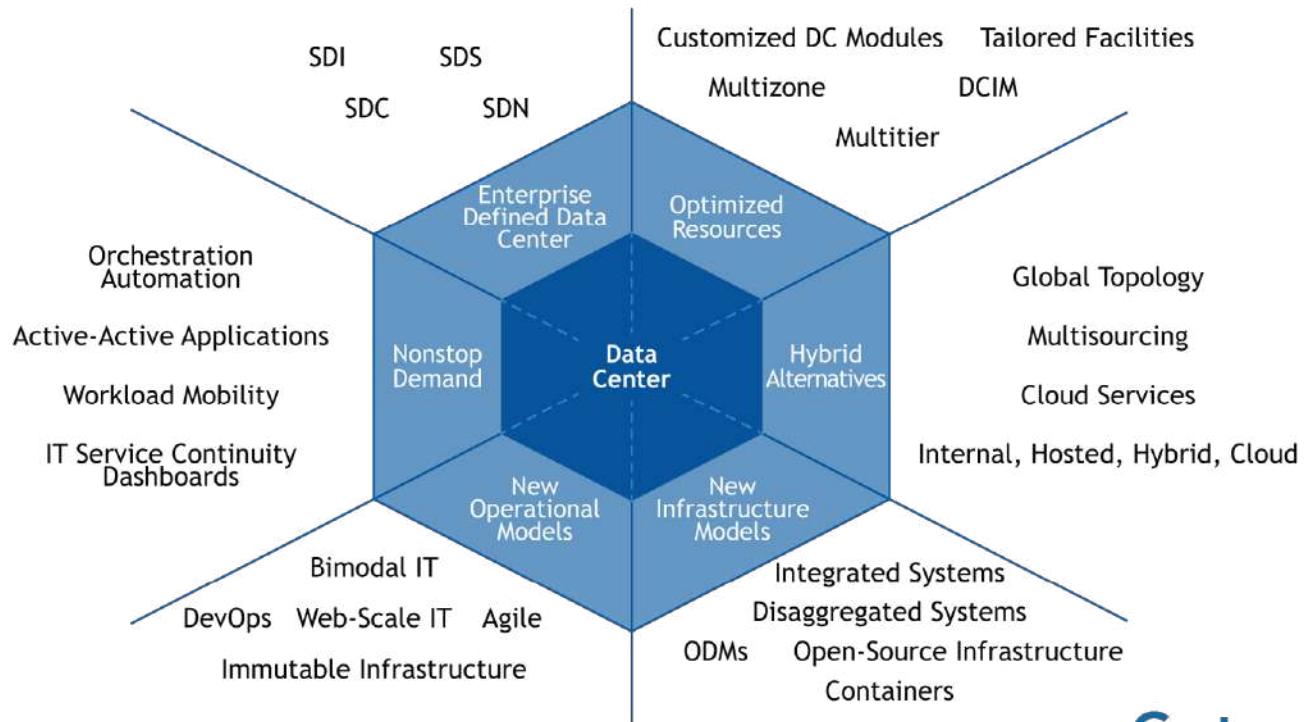
Si DevOps no presta atención a la seguridad puede facilitar la rápida introducción de las vulnerabilidades.

## DevOps vs DevSecOps



## Nuevo Modelo Operacional

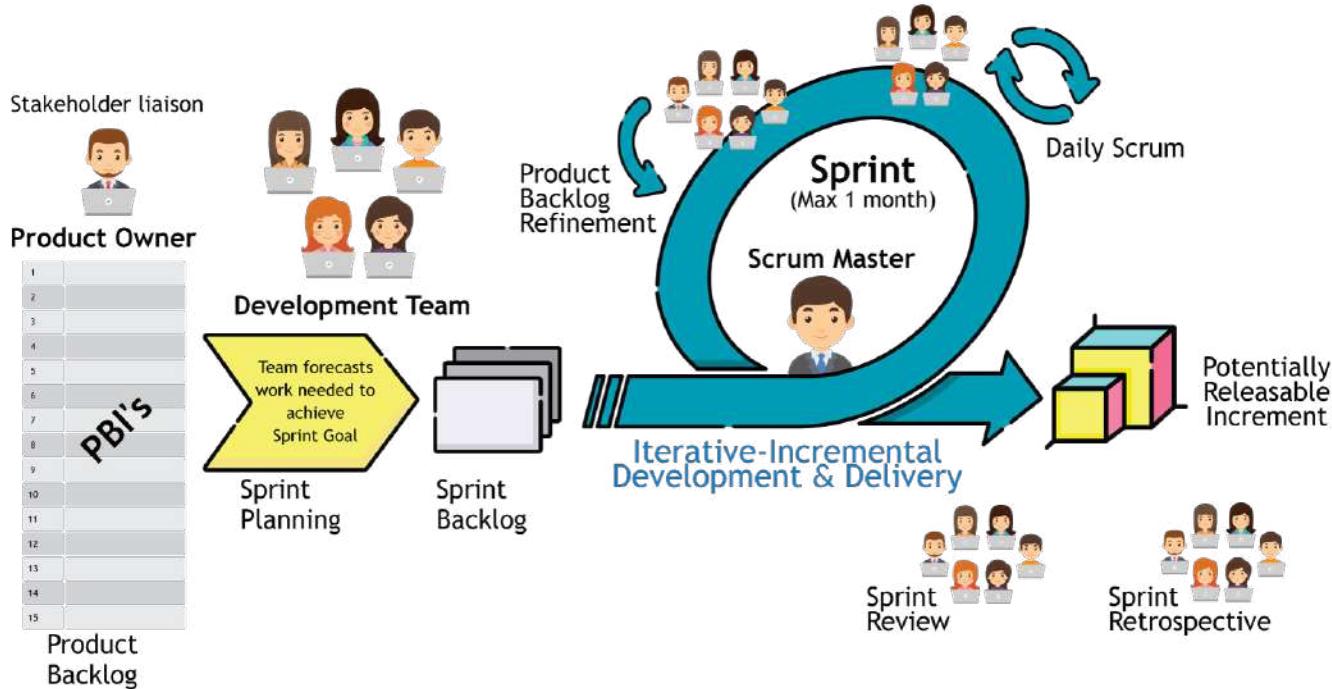
# ¿Cómo lucirá el Centro de Datos en el futuro?



© 2016 Gartner, Inc. and/or its affiliates. All rights reserved.

**Gartner**

## Scrum



**Scrum** es un marco para desarrollar y mantener productos complejos. Consiste en roles **Scrum**, eventos, artefactos y las reglas que los unen. **Ken Schwaber** y **Jeff Sutherland** desarrollaron **Scrum**.

Fuente: [www.scrumguides.org/docs/scrumguide](http://www.scrumguides.org/docs/scrumguide)

## 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.

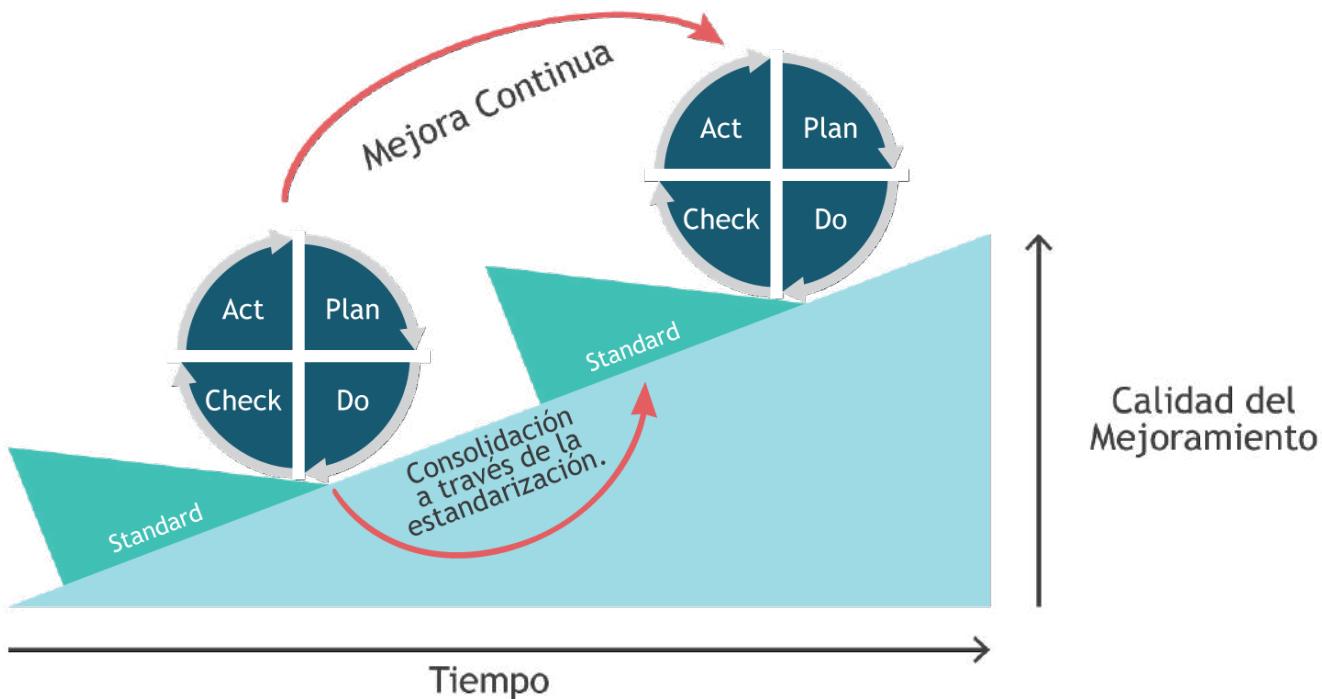
**Acuerdos de Nivel Operacional** (Operational-level agreements u OLAs) pueden ser utilizados por grupos internos para apoyar **SLAs**.

## 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**.



## Definición de Listo (en Agile/Scrum)

Definición de “Listo” Cuando un artículo o **Producto Backlog** o un Incremento se describe como “Listo”, todo el mundo debe entender lo que significa “Listo”.

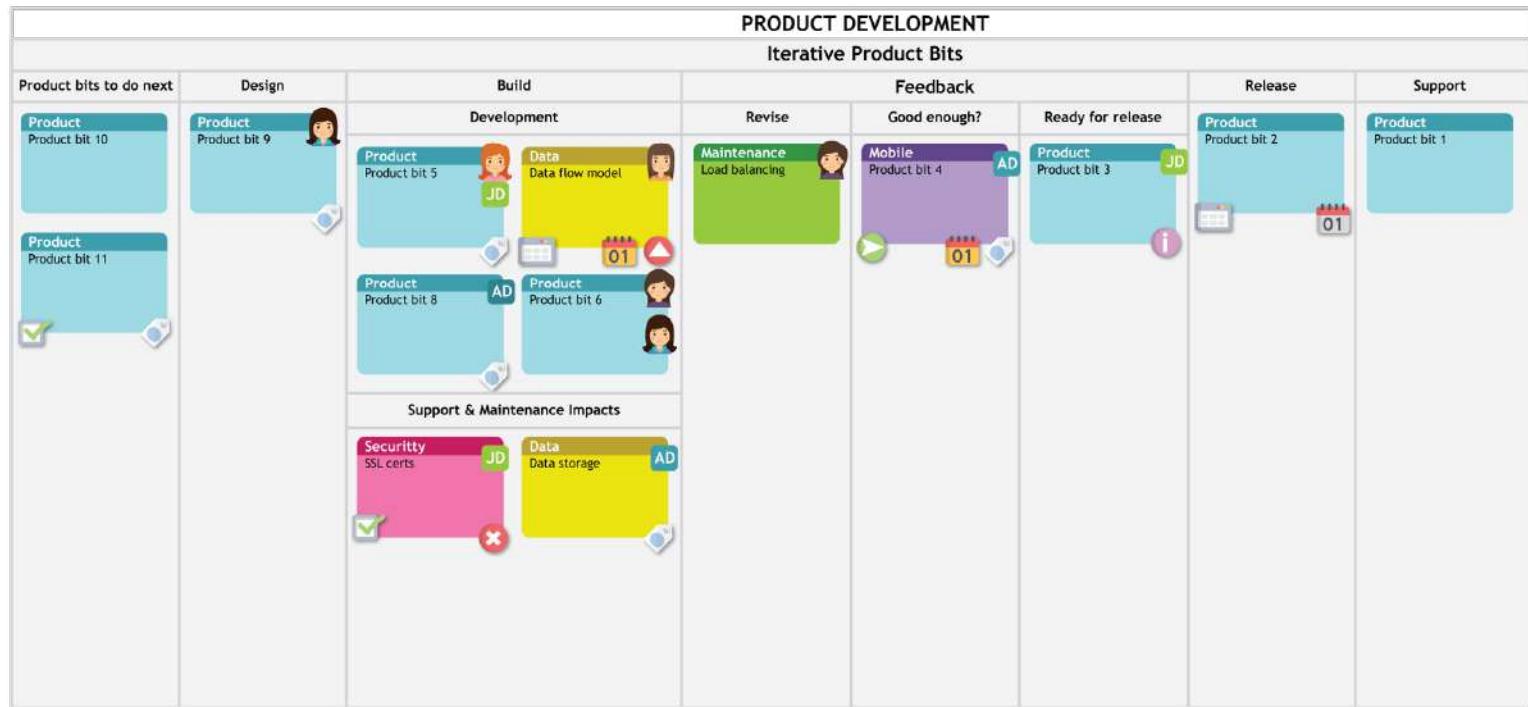
Aunque esto varía considerablemente según el **Equipo Scrum**, los miembros deben tener un entendimiento compartido de lo que significa que el trabajo sea completo, para asegurar la transparencia.

Esta es la definición de “Listo” para el **Equipo Scrum** y se utiliza para evaluar cuándo se completa el incremento del trabajo. La misma definición guía al equipo de desarrollo para saber cuántos elementos de la cartera de productos se pueden seleccionar durante una planificación de **Sprint**.

El propósito de cada **Sprint** es entregar Incrementos de funcionalidad potencialmente liberable que se adhieran a la definición actual del **Equipo de Scrum** de “Listo”.

## Kanban para Equipo 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**.





# Desarrollo

**CertiProf®**  
Professional Knowledge

[www.certiprof.com](http://www.certiprof.com)

CERTIPROF® is a registered trademark of CertiProf, LLC in the United States and/or other countries.

## 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.

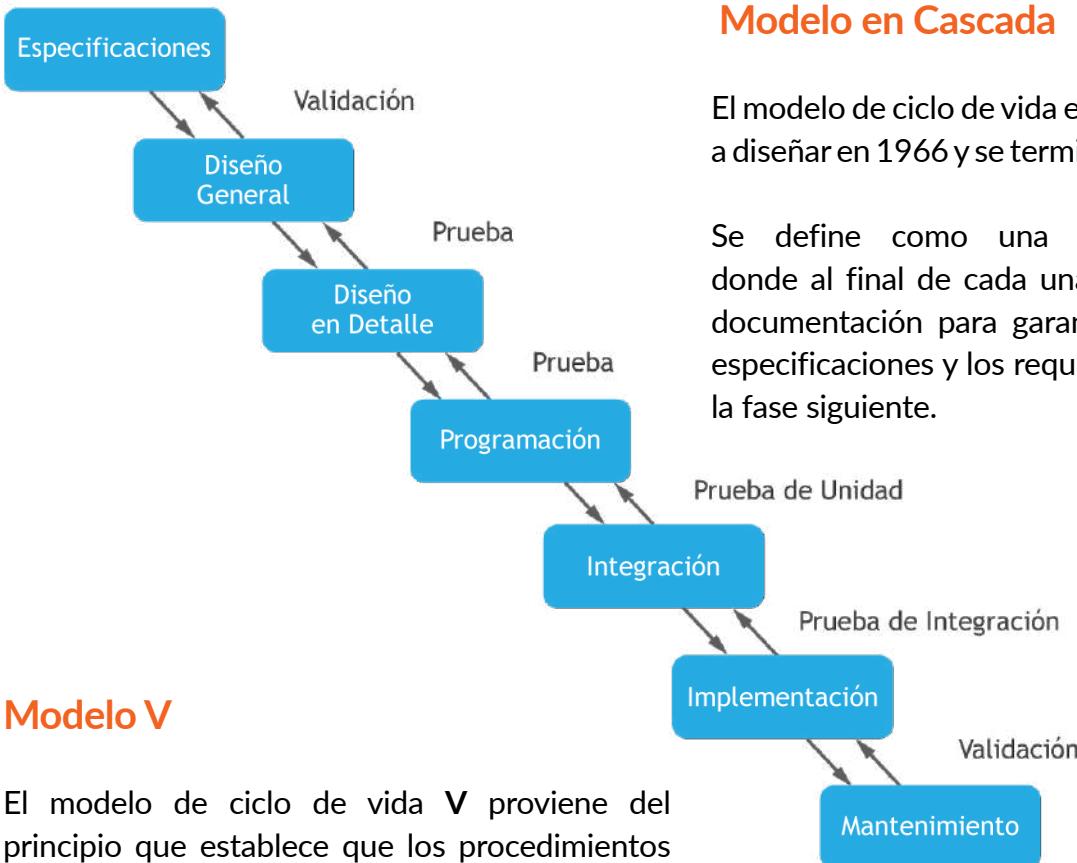
Las metodologías **Agile** ofrecen una solución para casi todos los proyectos que tienen estas características.

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.

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.

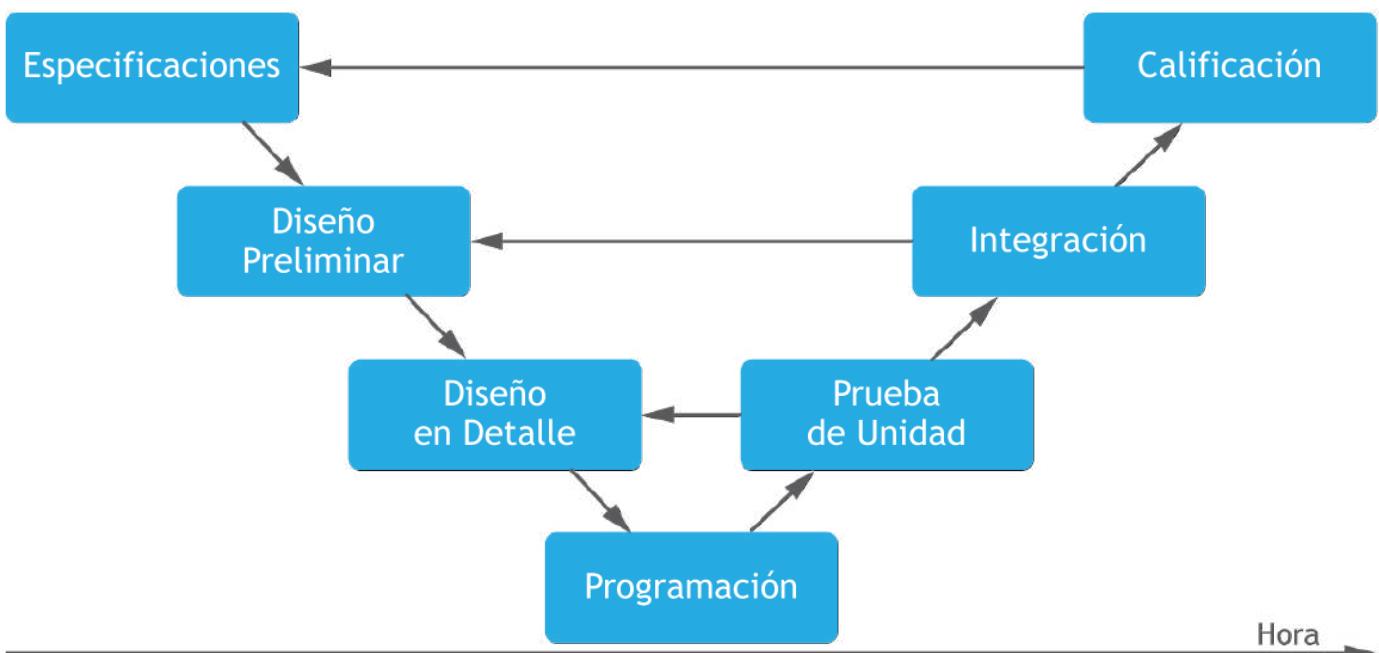
Este enfoque se ha encontrado eficaz en proyectos que tienen requisitos dinámicos y la necesidad de reducir drásticamente el tiempo de desarrollo, mientras mantiene la alta calidad.

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.

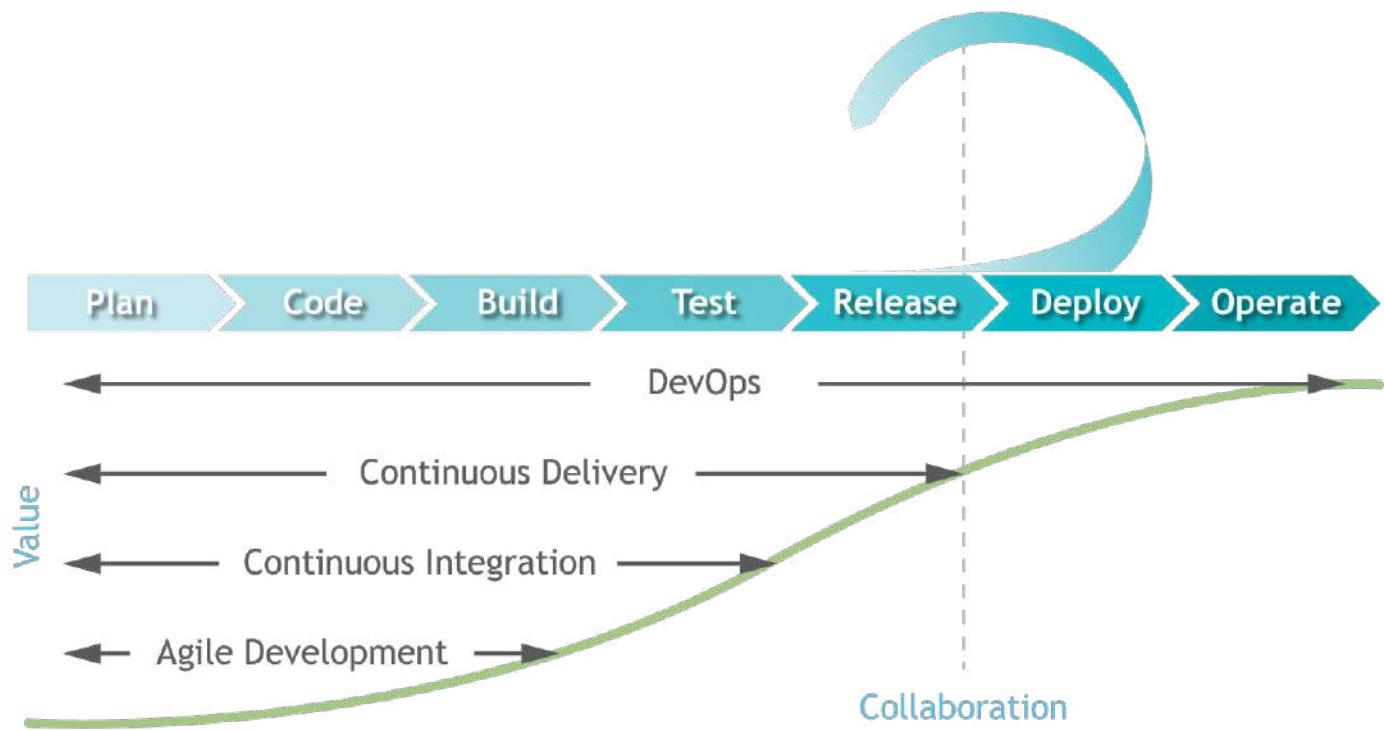


## Modelo V

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





# Integración Continua

**CertiProf®**  
Professional Knowledge

[www.certiprof.com](http://www.certiprof.com)

CERTIPROF® is a registered trademark of CertiProf, LLC in the United States and/or other countries.

## 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.

### 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.

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**.

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.

**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**.



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

**CertiProf®**  
Professional Knowledge

[www.certiprof.com](http://www.certiprof.com)

CERTIPROF® is a registered trademark of CertiProf, LLC in the United States and/or other countries.

## 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.



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 está 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.

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.

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

**CertiProf®**  
Professional Knowledge

[www.certiprof.com](http://www.certiprof.com)

CERTIPROF® is a registered trademark of CertiProf, LLC in the United States and/or other countries.

## 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

**CertiProf®**  
Professional Knowledge

[www.certiprof.com](http://www.certiprof.com)

CERTIPROF® is a registered trademark of CertiProf, LLC in the United States and/or other countries.

<b>Cultura</b>	<ul style="list-style-type: none"> <li>• Cambiar la manera en que pensamos y nos comportamos en la organización.</li> <li>• Convertirnos en uno.</li> <li>• Grassroots.</li> <li>• Cooperación.</li> </ul>
<b>Automatización</b>	<ul style="list-style-type: none"> <li>• Configurar Items.</li> <li>• Infraestructura como código.</li> </ul>
<b>Lean</b>	<ul style="list-style-type: none"> <li>• Enfocado en el valor y el cliente.</li> <li>• Reduciendo el tiempo gastado en actividades sin valor.</li> </ul>
<b>Medidas</b>	<ul style="list-style-type: none"> <li>• Medir todo el tiempo.</li> <li>• Mostrar mejoras.</li> </ul>
<b>Compartir (Sharing)</b>	<ul style="list-style-type: none"> <li>• Compartir.</li> <li>• Colaborar.</li> <li>• Transparencia.</li> </ul>



# DevOps, Otras Prácticas Recomendadas y Frameworks (Marcos)

**CertiProf®**  
Professional Knowledge

[www.certiprof.com](http://www.certiprof.com)

CERTIPROF® is a registered trademark of CertiProf, LLC in the United States and/or other countries.

## 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** 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.

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.

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**.



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.

**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](http://www.axelos.com).*

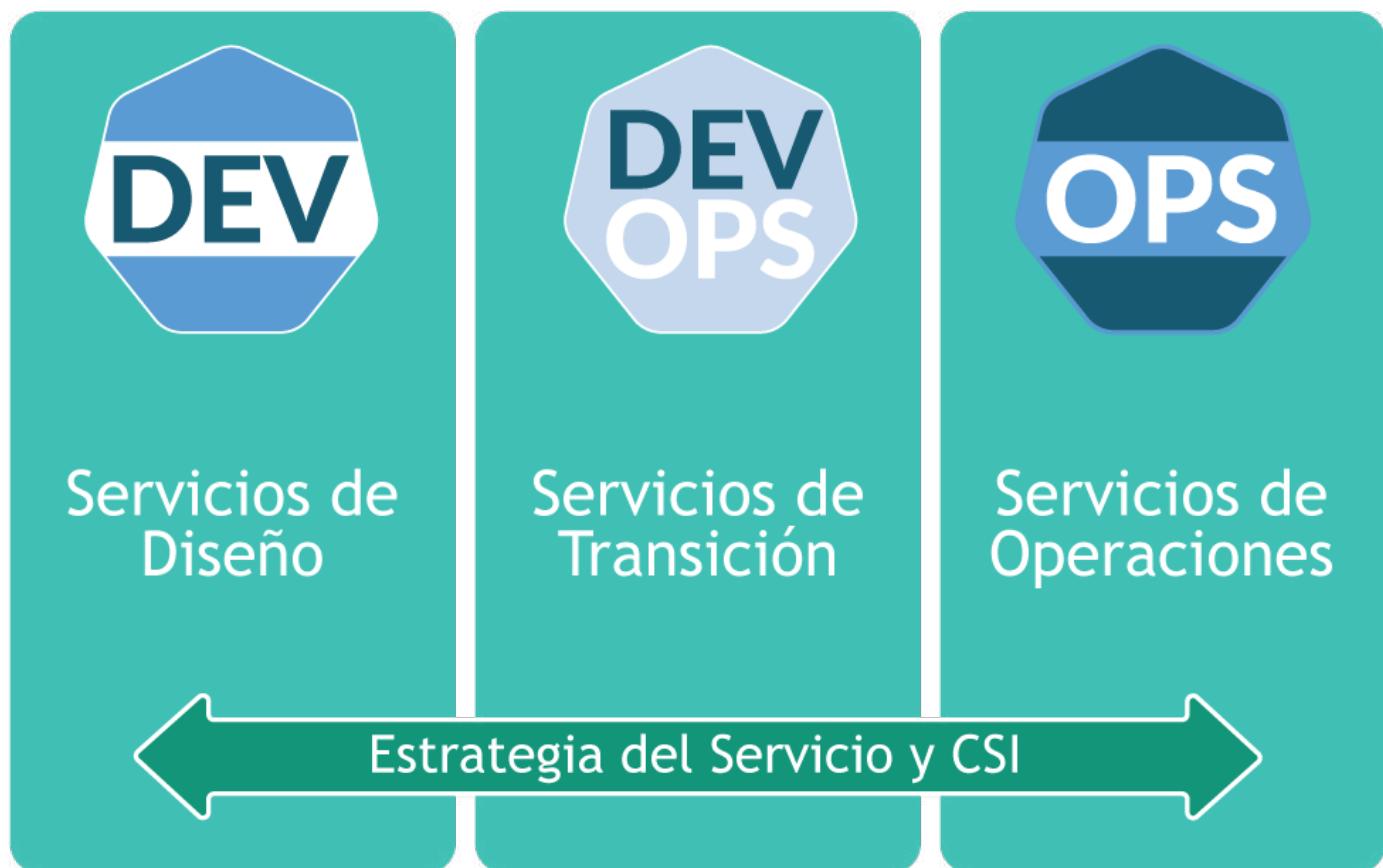


Otra percepción es que **ITIL** y **DevOps** no pueden trabajar juntos porque no son compatibles.

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 industry architecture, figure 9.

"ITIL® is a (registered) Trade Mark of AXELOS Limited. All rights reserved.

©Copyright AXELOS.



# Taller



Tiempo Estimado:  
30 Minutos

**CertiProf®**  
Professional Knowledge

[www.certiprof.com](http://www.certiprof.com)

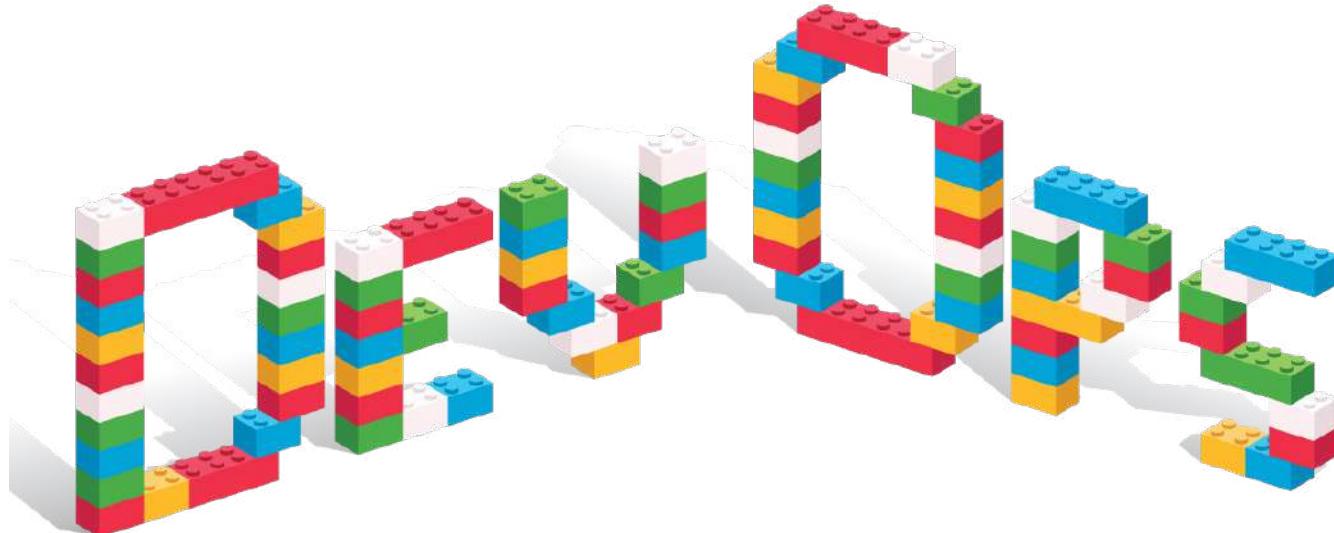
CERTIPROF® is a registered trademark of CertiProf, LLC in the United States and/or other countries.

## ¿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.



Esta facilitación se hace basada en el libro de **Daya Playeba**. Duración máxima 3 horas.

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



# Cultura DevOps

**CertiProf®**  
Professional Knowledge

[www.certiprof.com](http://www.certiprof.com)

CERTIPROF® is a registered trademark of CertiProf, LLC in the United States and/or other countries.

## 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.

- 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.

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

**CertiProf®**  
Professional Knowledge

[www.certiprof.com](http://www.certiprof.com)

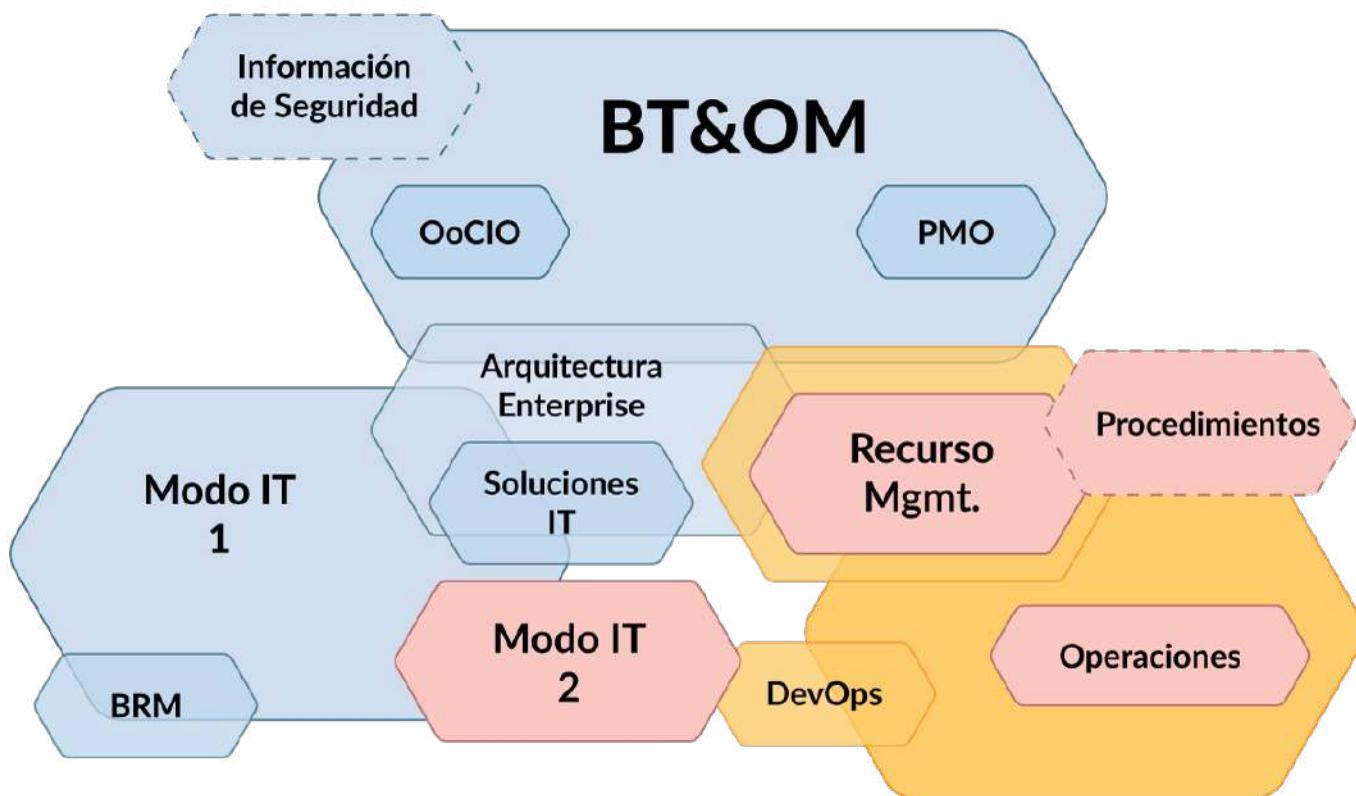
CERTIPROF® is a registered trademark of CertiProf, LLC in the United States and/or other countries.

## Organización-Legado

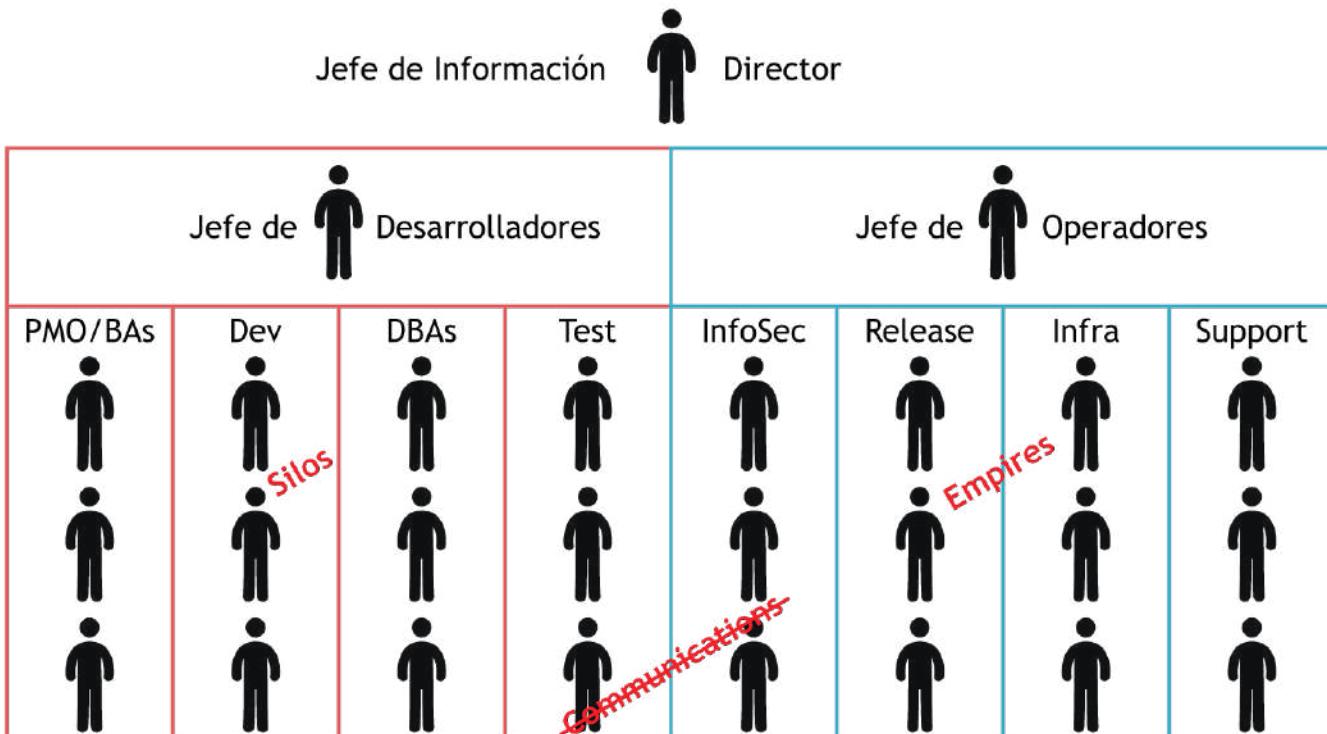
Rendimiento, optimización de recursos, estructura y líneas de reporte, comando y control.



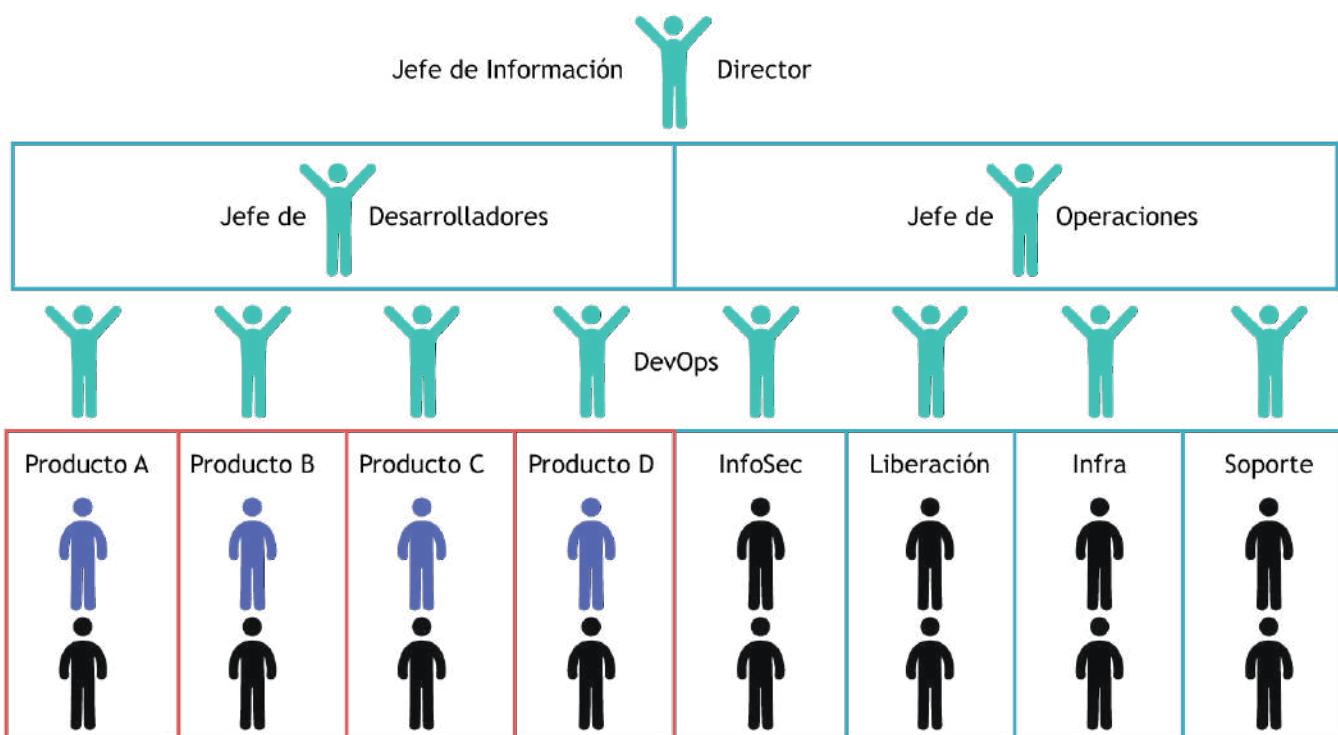
Alineación, integración, coordinación de procesos y reglas gubernamentales, liderazgo y colaboración.



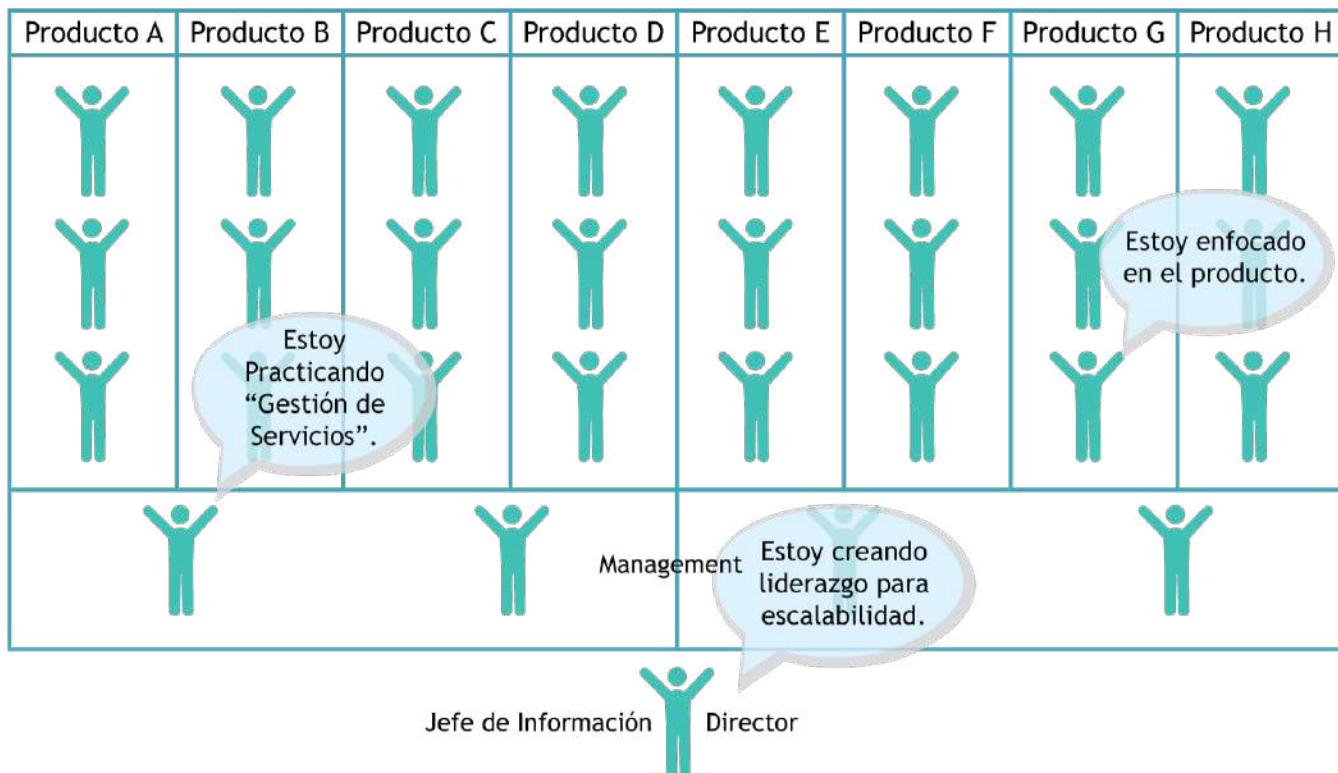
## Operadores y Desarrolladores Tradicionales



## 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

**CertiProf®**  
Professional Knowledge

[www.certiprof.com](http://www.certiprof.com)

CERTIPROF® is a registered trademark of CertiProf, LLC in the United States and/or other countries.

## 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.

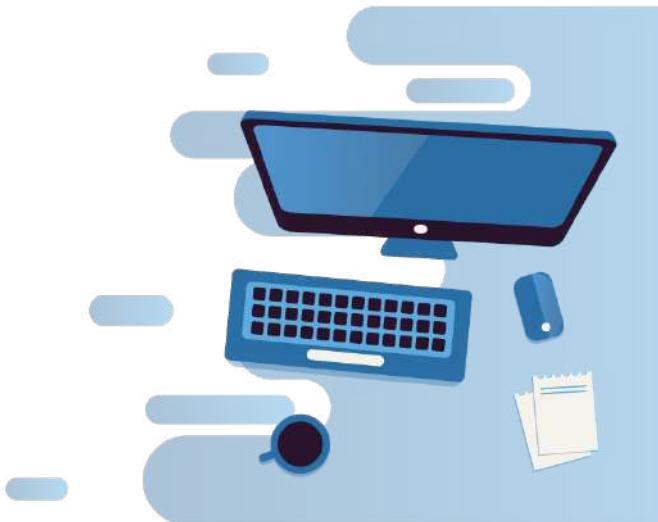
El desafío clave sigue siendo encontrar un balance entre los complejos entornos de TI de la organización, la velocidad de implementación y el riesgo, que deben hacerse para que trabajen juntos.

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.

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

**CertiProf®**  
Professional Knowledge

[www.certiprof.com](http://www.certiprof.com)

CERTIPROF® is a registered trademark of CertiProf, LLC in the United States and/or other countries.

## 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.

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.

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.

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.

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

**CertiProf®**  
Professional Knowledge

[www.certiprof.com](http://www.certiprof.com)

CERTIPROF® is a registered trademark of CertiProf, LLC in the United States and/or other countries.

## 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.

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

**CertiProf®**  
Professional Knowledge

[www.certiprof.com](http://www.certiprof.com)

CERTIPROF® is a registered trademark of CertiProf, LLC in the United States and/or other countries.

## Herramientas para el apoyo de la cultura DevOps

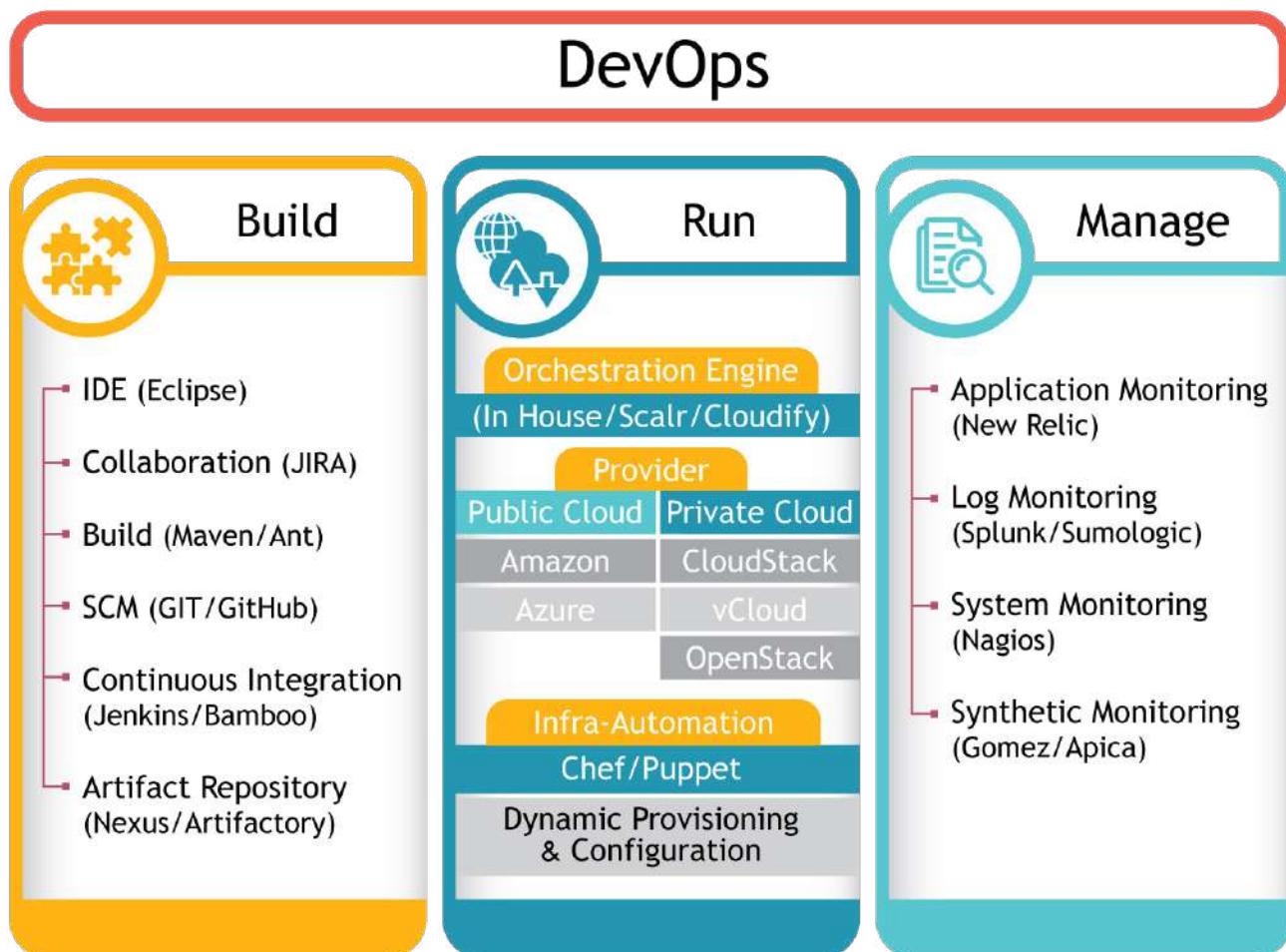
El uso de herramientas no es hacer DevOps.

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 inician a tiempo. Si la compañía compra un software de gestión de reuniones, ¿logrará que las reuniones inician 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?



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.

## 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.

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 %.

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. 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.



# Jenkins

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.

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

**CertiProf®**  
Professional Knowledge

[www.certiprof.com](http://www.certiprof.com)

CERTIPROF® is a registered trademark of CertiProf, LLC in the United States and/or other countries.

# P.P.T y Cultura



# 1. Personas



## 2. Procesos



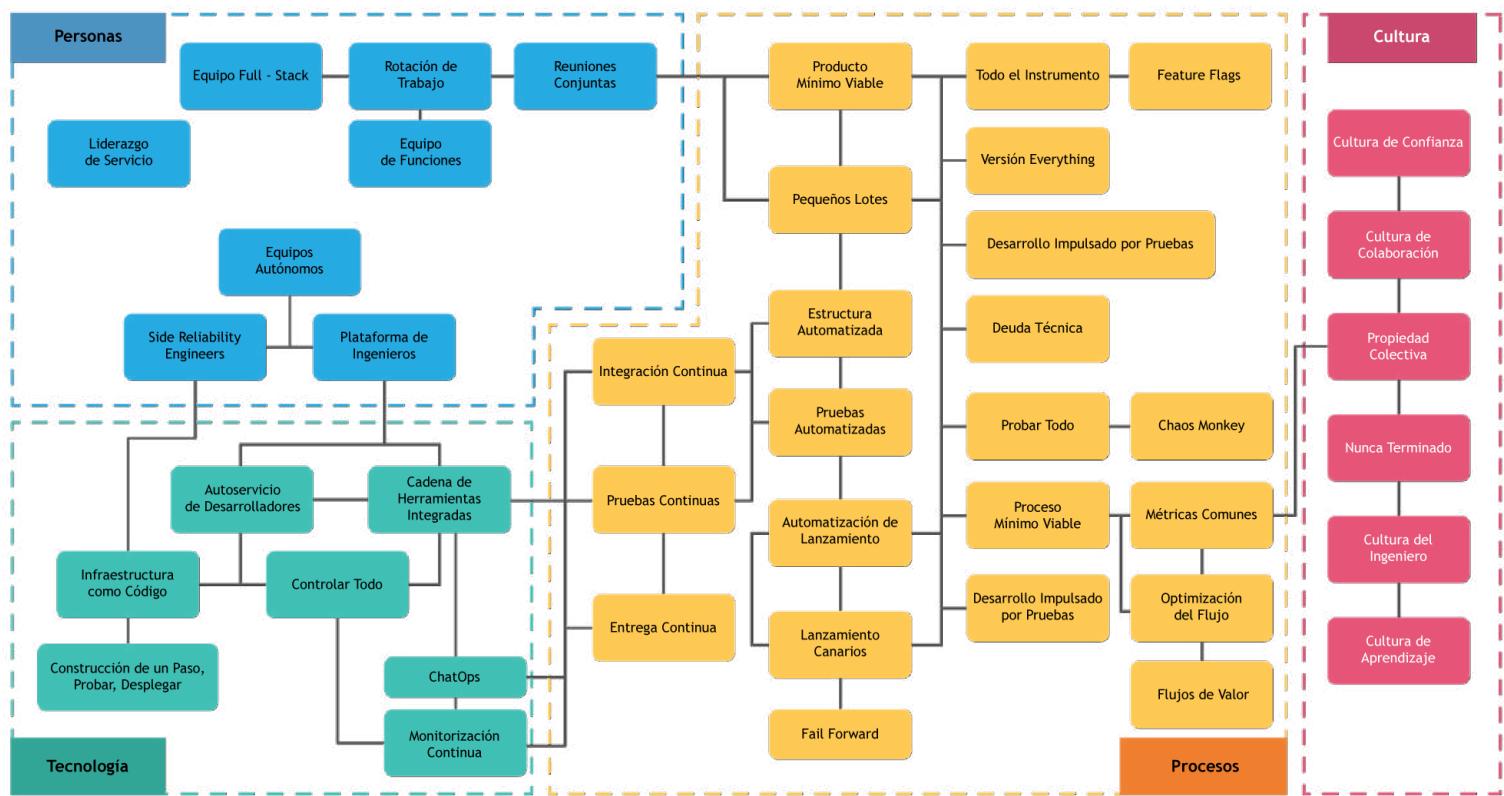
### 3. Tecnología

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](http://www.gartner.com)

# Principles and Practices of DevOps





# Lista de Chequeo DevOps y Estado del Reporte DevOps

**CertiProf®**  
Professional Knowledge

[www.certiprof.com](http://www.certiprof.com)

CERTIPROF® is a registered trademark of CertiProf, LLC in the United States and/or other countries.

## 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.

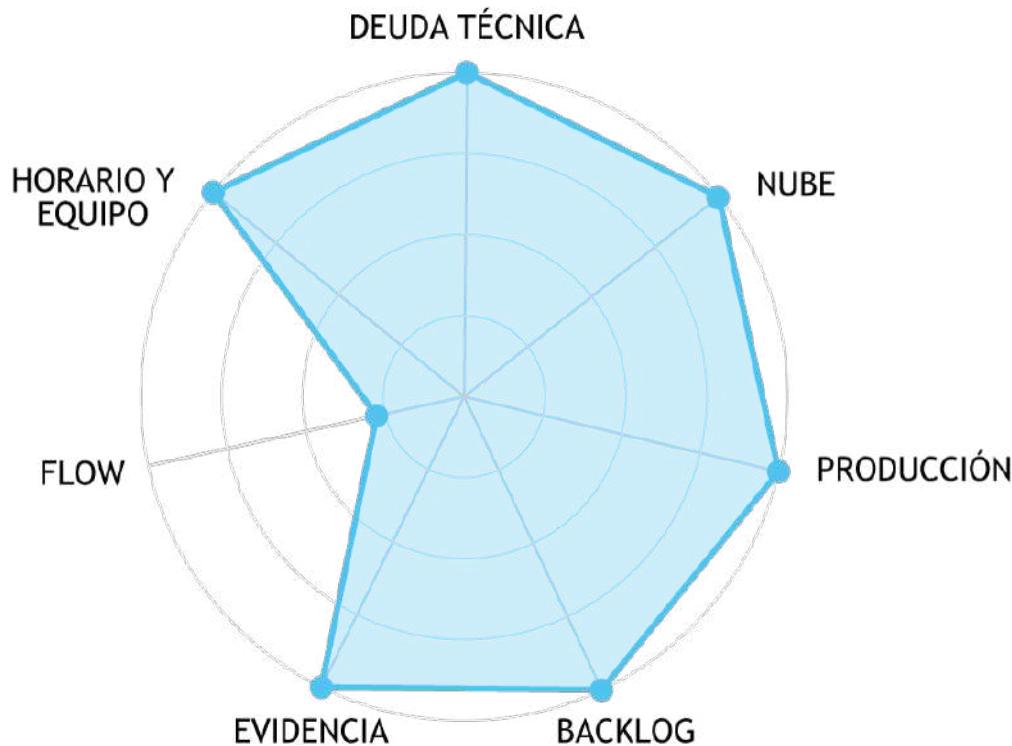
## Autoevaluación

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.

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



Fuente

<http://devopsassessment.azurewebsites.net/es-ES/>

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 2017

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®.

This year's report explores new areas:

- The influence leadership has on **DevOps** transformations.
- How high- and low-performing teams automate differently.
- The impact of architecture and team structure on IT performance.

[www.puppet.com](http://www.puppet.com)



# Taller



Tiempo Estimado:  
30 Minutos

**CertiProf®**  
Professional Knowledge

[www.certiprof.com](http://www.certiprof.com)

CERTIPROF® is a registered trademark of CertiProf, LLC in the United States and/or other countries.

## Análisis del Estado DevOps de Puppet





# Referencias

**CertiProf®**  
Professional Knowledge

[www.certiprof.com](http://www.certiprof.com)

CERTIPROF® is a registered trademark of CertiProf, LLC in the United States and/or other countries.

## 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)

★★★★★ – 950 customer reviews

#1 Best Seller in Production & Operations

• See all 9 formats and editions

Kindle \$8.00	Hardcover \$29.95	Paperback \$15.11	Audible \$16.95 or Free
Read with Our Free App	17 Used from \$15.20 18 New from \$29.95	19 Used from \$11.04 51 New from \$12.09	or Free with Audible 30-day 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.



# Preguntas de Apoyo

**CertiProf®**  
Professional Knowledge

[www.certiprof.com](http://www.certiprof.com)

CERTIPROF® is a registered trademark of CertiProf, LLC in the United States and/or other countries.

**DevOps Essentials  
Professional Certificate (DEPC)**

**Examen de ejemplo V092018**

1. La palabra **DevOps** es una contracción de “Desarrollo” (Development) y “Operaciones” (Operations).
  - a) Verdadero.
  - b) Falso.
2. Tanto **ITIL** como **DevOps** están destinados a apoyar la prestación de servicios de calidad.
  - a) Verdadero.
  - b) Falso.
3. Los orígenes del movimiento **DevOps** se encuentran alrededor de 2005.
  - a) Verdadero.
  - b) Falso.
4. ¿Uno de los principios del proceso de desarrollo ágil es entregar software funcional en entregas más pequeñas y frecuentes?
  - a) Verdadero.
  - b) Falso.
5. El objetivo de **DevOps** no sólo es incrementar el ritmo de cambio, sino implementar exitosamente nuevas funcionalidades en producción, sin causar caos ni interrumpir otros servicios.
  - a) Verdadero.
  - b) Falso.
6. **DevOps** tiene un enfoque holístico hacia la erradicación sistemática del trabajo no planificado.
  - a) Verdadero.
  - b) Falso.

7. Los resultados de la Tercera Vía incluyen comprensión y respuesta a las necesidades de los clientes, internos y externos, el acortamiento y amplificación de los bucles de retroalimentación e incluir conocimiento donde sea necesario.
  - a) Verdadero.
  - b) Falso.
8. Uno de los beneficios de negocios para las organizaciones que adoptan DevOps es: Reducción en el tiempo de comercialización, es decir, con tiempos de ciclo más reducidos y tasas de despliegue más altas.
  - a) Verdadero.
  - b) Falso.
9. Las organizaciones de **TI** de alto rendimiento sacan una ventaja inicial a las otras organizaciones, pero esa ventaja inicial tiende a mantenerse estable.
  - a) Verdadero.
  - b) Falso.
10. **DevOps** es incompatible con la metodología **Lean**.
  - a) Verdadero.
  - b) Falso.
11. La entrega continua de soluciones de servicio desarrolladas debe ser independiente de la capacidad del cliente para absorber el beneficio.
  - a) Verdadero.
  - b) Falso.
12. El objetivo final para el desarrollo de aplicaciones es adoptar un enfoque de gestión de servicios de negocio (BSM).
  - a) Verdadero.
  - b) Falso.
13. **DevOps** hace énfasis en la comunicación transparente, la colaboración e integración entre **Desarrolladores y Operadores**.

- a) Verdadero.  
b) Falso.
14. Una gran inspiración para **DevOps** fue el movimiento **Agile**, cuyo gran inconveniente era que dejaba de lado al equipo de **Desarrolladores**.
- a) Verdadero.  
b) Falso.
15. Los equipos **DevOps** logran el éxito por el uso de dos componentes claves llamados “comunicación” y “visibilidad en tiempo real”.
- a) Verdadero.  
b) Falso.
16. **DevOps** es automatización, es un error considerar que no es completamente automatizado.
- a) Verdadero.  
b) Falso.
17. **DevOps** busca maximizar los resultados del negocio, tales como incrementar las ventas, la rentabilidad y mejorar la velocidad del negocio maximizando los costos operativos.
- a) Verdadero.  
b) Falso.
18. Los pilares de **DevOps** son **Agile**, la gestión de servicios de tecnologías de la información (en inglés **IT Service Management**, ITSM), y la **Entrega Continua**.
- a) Verdadero.  
b) Falso.
19. El objetivo principal del ciclo de vida del software es definir todas las fases intermedias necesarias para validar el desarrollo de la aplicación.
- a) Verdadero.  
b) Falso.

20. En la Entrega Continua, la idea es producir rápidamente versiones del sistema que estén operativas, aunque no tengan toda la funcionalidad prevista para el sistema, pero son un resultado de valor para el negocio.
- a) Verdadero.  
b) Falso.
21. La Entrega Continua representa la capacidad de desplegar el software en cualquier entorno específico en un momento específico.
- a) Verdadero.  
b) Falso.
22. La importancia de tener la capacidad de hacer Entrega Continua está en poder solucionar cualquier problema potencial que un usuario pueda experimentar mientras usa el software.
- a) Verdadero.  
b) Falso.
23. **CALMS** es un acrónimo desarrollado originalmente por **John Willis** y **Damon Edwards** en el 2010 y posteriormente refinado por **Jez Humble** como un recurso para describir a **DevOps**.
- a) Verdadero.  
b) Falso.
24. La letra C de **CALMS** significa “Ciclo” en **DevOps**.
- a) Verdadero.  
b) Falso.
25. La **S** de **CALMS** es **Share/Compartir** se basa en crear responsabilidad compartida y sentido de pertenencia. Las personas están dispuestas a trabajar juntas si sus pensamientos y opiniones están siendo escuchados.
- a) Verdadero.  
b) Falso.

26. 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.
  - a) Verdadero.
  - b) Falso.
27. **DevOps** mantiene los canales de comunicación cerrados para no permitir una infiltración de los clientes a los niveles de desarrollo y operación.
  - a) Verdadero.
  - b) Falso.
28. Las herramientas más populares de **DevOps** son: **Git, Jenkins, Selenium, Puppet, Nagios, Docker**.
  - a) Verdadero.
  - b) Falso.
29. **DevOps** significa que los desarrolladores tomarán las responsabilidades de los operadores.
  - a) Verdadero.
  - b) Falso.
30. Según **Gartner**, **DevOps** se trata de personas, procesos y tecnología, también de aplicar principios y métodos para la mejor colaboración entre la entrega de software y tus equipos de operaciones TI.
  - a) Verdadero.
  - b) Falso.
31. **DevOps** también tiene un enfoque hacia la erradicación sistemática del trabajo no planificado, aplicando principios de la ingeniería resiliente, la gestión y reducción responsable de la deuda técnica.
  - a) Verdadero.
  - b) Falso.

32. Uno de los Beneficios de **DevOps** es la reducción en el tiempo de comercialización.
- a) Verdadero.  
b) Falso.
33. **DevOps** realiza un incremento de la calidad, es decir; incremento de disponibilidad, incremento del ritmo de cambios exitosos, reducción de los fallos, entre otros.
- a) Verdadero.  
b) Falso.
34. **DevOps** mejora la transparencia que es esencial para una toma de decisiones efectiva a través de colaboración entre el desarrollador y los equipos de operación.
- a) Verdadero.  
b) Falso.
35. Seguridad de la Información debe mantenerse involucrada en **Desarrollo y Operaciones** y esto es llamado actualmente **DevSecOps**.
- a) Verdadero.  
b) Falso.
36. En la cultura de **DevOps** es importante fomentar el respeto. Cuando puedes entender y empatizar en ambos equipos de **Desarrolladores y Operadores**, puedes centrar todas tus energías en resultados positivos.
- a) Verdadero.  
b) Falso.
37. Los principios fundamentales de Lean se centran en la identificación de flujos de valor, que en el caso de **DevOps**, son aplicaciones que forman servicios prestados a los clientes, los empleados o los socios.
- a) Verdadero.  
b) Falso.
38. **System Thinking** o pensamiento sistemático trata de identificar y culpar al individuo responsable de cualquier incidente y no al sistema que permitió que este sucediera.

- a) Verdadero.  
b) Falso.
39. **ITIL y DevOps** no pueden trabajar juntos ya que no son compatibles. No se puede integrar los temas ágiles con marcos tradicionales de gestión de operaciones.
- a) Verdadero.  
b) Falso.
40. El acrónimo **CALMS** significa: **Cultura, Automatización, Lean, Medios de Comunicación, Sharing/Compartir.**
- a) Verdadero.  
b) Falso.

## Respuesta

- |            |   |            |   |
|------------|---|------------|---|
| <b>1.</b>  | A | <b>21.</b> | A |
| <b>2.</b>  | A | <b>22.</b> | A |
| <b>3.</b>  | B | <b>23.</b> | A |
| <b>4.</b>  | A | <b>24.</b> | B |
| <b>5.</b>  | A | <b>25.</b> | A |
| <b>6.</b>  | A | <b>26.</b> | A |
| <b>7.</b>  | B | <b>27.</b> | B |
| <b>8.</b>  | A | <b>28.</b> | A |
| <b>9.</b>  | B | <b>29.</b> | B |
| <b>10.</b> | B | <b>30.</b> | A |
| <b>11.</b> | B | <b>31.</b> | A |
| <b>12.</b> | A | <b>32.</b> | B |
| <b>13.</b> | A | <b>33.</b> | A |
| <b>14.</b> | B | <b>34.</b> | A |
| <b>15.</b> | A | <b>35.</b> | A |
| <b>16.</b> | B | <b>36.</b> | A |
| <b>17.</b> | B | <b>37.</b> | A |
| <b>18.</b> | A | <b>38.</b> | B |
| <b>19.</b> | A | <b>39.</b> | B |
| <b>20.</b> | A | <b>40.</b> | B |



# DEVOPS ESSENTIALS PROFESSIONAL CERTIFICATE (DEPC)

**CertiProf®**  
Professional Knowledge



[certiprof.com](http://certiprof.com)



@Certiprof



@CertiProf



CertiProf



Certiprof\_llc

[www.certiprof.com](http://www.certiprof.com)

CERTIPROF® is a registered trademark of CertiProf, LLC in the United States and/or other countries.