Centro Universitario de la Defensa
en la Escuela Naval Militar

TRABAJO FIN DE MÁSTER

DevOps qué es y cómo puede mejorar la Gestión de TI en el Ministerio de Defensa

Máster Universitario en Dirección TIC para la Defensa

ALUMNO: Francisco Escalante Martínez

DIRECTORES: Miguel Ángel Ares Tarrío

José María Núñez Ortuño

CURSO ACADÉMICO: 2020-2021

Universidade Vigo

Centro Universitario de la Defensa
en la Escuela Naval Militar

TRABAJO FIN DE MÁSTER

DevOps qué es y cómo puede mejorar la Gestión de TI en el Ministerio de Defensa

Máster Universitario en Dirección TIC para la Defensa

Especialidad de Sistemas y Tecnologías de la Información

Universidade Vigo

RESUMEN

Las empresas y las organizaciones se desenvuelven hoy en día en unos ambientes caracterizados por su complejidad, sujetos a cambios cada vez más rápidos, intensos y profundos, y con un elevado grado de incertidumbre, lo que dificulta el proceso de toma de decisiones y de ejecución y control de acciones. Su capacidad para adaptarse continuamente al entorno y a los requisitos que este exige mediante respuestas rápidas, reconfigurándose y transformándose con flexibilidad en caso necesario, constituye un objetivo estratégico fundamental.

En estas circunstancias la información disponible, interna y externa, así como la posesión de unos mecanismos que aseguren su buena gestión y su transformación en conocimiento útil para la toma de decisiones, se convierten en factores claves para asegurar la supervivencia, el éxito y el progreso constantes de cualquier entidad. Aquí adquieren un papel clave los servicios ofrecidos por las TIC y, en especial, el software específico que soporta los procesos y la gestión de datos de la organización. Por eso las organizaciones, incluyendo a las administraciones públicas en general y al Ministerio de Defensa y las Fuerzas Armadas en particular, han tomado conciencia de la necesidad de modernizarse acometiendo procesos de transformación digital. El Ministerio, en esta transformación, debe ser capaz de dotarse de aplicaciones de una forma ágil y continua.

Para ello el mundo tecnológico actual ofrece la adopción de la cultura DevOps, que preconiza el establecimiento de una mentalidad y el empleo de herramientas y técnicas avanzadas que favorezcan el establecimiento de una flujo continuo, rápido y seguro de desarrollo y despliegue de aplicaciones de calidad.

Este trabajo pretende mostrar cuál es la situación actual en el Ministerio en el sector del desarrollo y despliegue de aplicaciones y cómo este mejoraría al adoptar DevOps.

PALABRAS CLAVE

DevOps, cultura, ágil, software, automatización.

AGRADECIMIENTOS

Al comandante Arroyo de la JCISAT del ET, por dedicarme su tiempo e ilustrarme sobre el desempeño y actividades del personal que se dedica al desarrollo de aplicaciones en el ET.

Al teniente coronel Rodríguez del CESTIC, por atenderme y proporcionarme, entre un par de reuniones de alto nivel y unos cuantos «fuegos urgentes que apagar», la visión de los equipos de operaciones del ciclo de vida del software creado en el MINISDEF.

A mi mujer, Natalia, por mostrarse tan comprensiva con las horas dedicadas al estudio y elaboración de este trabajo.

A mis hijos, por aguantar, sin muchas protestas, mis ausencias pegado al ordenador durante las vacaciones y las horas libres.

CONTENIDO

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

1 INTRODUCCIÓN Y OBJETIVOS

1.1 Introducción

Hoy en día podemos considerar que la información es el recurso clave de cualquier organización. El sistema de información se constituye en el elemento coordinador y director del resto de sistemas. La obtención de datos y su trasformación en conocimiento e inteligencia útil es lo que va a permitir a una entidad sobrevivir, progresar y perdurar en ambientes cada vez más indefinidos y competitivos. La información en sí misma no tiene por qué ser un elemento esencial, pero adquiere valor y resulta determinante en cuanto se utiliza de forma eficiente para comprender qué está pasando y qué podría pasar y, en consecuencia, para la toma de decisiones y para el diseño y ejecución de los procesos que modelan las actividades. Por tanto, son los sistemas creados para tratar esa información, las aplicaciones informáticas como conjunto de software que se utiliza para tratarla y gestionarla, los que constituyen los elementos que en muchos casos van a determinar el éxito o fracaso de una entidad y de sus actividades. El ser capaz de proveerse de un software útil y de calidad, con capacidad de cambio y adaptación constante a las necesidades y al entorno altamente variable y competitivo en el que las organizaciones se desenvuelven hoy en día, es, por tanto, uno de los objetivos estratégicos fundamentales que, en la actualidad y en el futuro, siempre debería contemplarse[1].

El Ministerio de Defensa (MINISDEF) y las Fuerzas Armadas (FAS) no pueden ser ajenos a esta premisa que está implícita en el camino de transformación digital que han emprendido. El conjunto del Ministerio, tanto para el funcionamiento en su vertiente administrativa, como organización que cuenta con un gran número de personal y de recursos materiales que gestionar, como para su funcionamiento operativo, en la ejecución de misiones militares, debe dotarse de software de calidad de forma ágil y de manera que pueda mantener una alta capacidad de adaptación, asegurándose el éxito en el cumplimiento de sus objetivos.

¿Cómo se proveen las organizaciones de un software adecuado? La tendencia actual es la adopción de paradigmas para el desarrollo de aplicaciones denominados ágiles y de las buenas prácticas definidas por DevOps en el diseño de la vida del software. DevOps es un acrónimo, compuesto por las palabras inglesas «development» y «operations», que hace referencia a los equipos que intervienen a lo largo del ciclo de vida del software: los de desarrollo, que lo diseñan y producen, compuestos por expertos tales como ingenieros de software, programadores y diseñadores web; y los de operaciones, dedicados a la gestión de las infraestructuras y plataformas asociadas a las tecnologías de la información y las comunicaciones (TI, TIC o CIS, según la terminología civil o militar empleada) y compuestos por administradores de bases de datos, administradores de sistemas, administradores de redes y otros tipos de administradores que son responsables y mantienen en funcionamiento las infraestructuras TIC y los servicios de plataforma. DevOps describe, en esencia, una filosofía que pretende extender las prácticas ágiles, empleadas normalmente solo por los equipos de desarrollo, a todo el ciclo de vida de las aplicaciones y al resto de equipos que intervienen en el proceso de su puesta en explotación, rompiendo las barreras de comunicación existentes tradicionalmente entre «desarrollo» y «operaciones».

Esta memoria puede encuadrarse en el ámbito teórico de estudio de la ingeniería del software y sus contenidos se adaptan a los objetivos de un máster dedicado a la gestión y dirección TIC. No obstante, puede tener una aplicación real en cuanto a la adopción de determinadas prácticas, procesos, relaciones de trabajo y herramientas tecnológicas. En concreto analiza cuáles son actualmente los hábitos, costumbres, procedimientos y tecnologías que se utilizan en el desarrollo y puesta en explotación de software en el ámbito de uno de los componentes específicos del MINISDEF, el Ejército de Tierra (ET), y cómo estos podrían mejorarse adoptando la cultura DevOps. Aunque se ha escogido al ET para realizar este trabajo, las conclusiones alcanzadas son de aplicación, con pequeñas variaciones en su caso, al resto

[1] Esta idea podemos encontrarla no solo en trabajos relacionados con TI sino en estudios en otros campos, como por ejemplo en este árticulo del Departamento de Organización de Empresas de la Universidad Politécnica de Valencia [41].

de componentes del Ministerio que producen código. El estudio se realiza teniendo en consideración el ciclo definido por DevOps, tratando de encontrar las características o aspectos que la adopción de las prácticas de esta cultura podría perfeccionar resultando en beneficio para el conjunto de la organización.

1.2 Organización de la memoria

Esta memoria se organiza de la siguiente forma:

- Un capítulo introductorio, en el que se incluye este apartado, que contextualiza y motiva el trabajo desarrollado. Trabajo cuyo objetivo general es justificar la adopción de la cultura DevOps en el MINISDEF como la mejor forma de adaptar el proceso de desarrollo de aplicaciones en el Ministerio a la transformación digital actualmente en curso.
- Un segundo capítulo dedicado al estado del arte en el sector de la ingeniería informática de creación de programas, repasando la evolución de los modelos empleados para el desarrollo de aplicaciones hasta llegar a las metodologías ágiles, introduciendo DevOps y analizando en qué consiste, para finalizar con el encuadramiento de DevOps dentro de la gestión integral de servicios TIC dentro de una organización.
- En el siguiente capítulo se pone de manifiesto, por un lado, la necesidad expuesta en un amplio desarrollo normativo de la transformación digital en el conjunto del MINISDEF y, más en concreto, en el ET; y, por otro, se analiza cuál es el actual modelo de desarrollo de aplicaciones en el Ministerio y cuál es la visión detallada que de este modelo tienen los equipos de desarrollo, limitando el análisis a los equipos de desarrollo del ET, y los equipos de operaciones.
- En el capítulo 4 se exponen los resultados de este análisis, considerando que DevOps sería la opción ideal en el Ministerio para evolucionar las prácticas y el concepto existentes sobre la creación de aplicaciones, alineándolos con la transformación digital. Se muestra lo que actualmente resulta conforme y lo que no con la cultura DevOps, así como los cambios que se podrían emprender, proponiendo un modelo final a alcanzar.
- Por último, a modo de resumen, se exponen las principales conclusiones alcanzadas, exponiendo por qué se debería evolucionar adoptando DevOps en el MINISDEF para mejorar el paradigma de desarrollo de aplicaciones y cómo esto se podría realizar.

2 ESTADO DEL ARTE

2.1 La creación de software, del arte a la ciencia y la industria

Desde mediados del siglo XX tanto la creación de máquinas de computación[2] como la de los programas[3] que las gobiernan y utilizan se inició de una forma creativa y artesanal, al igual que ha sucedido con otros muchos aspectos del desarrollo tecnológico humano, aunque en este caso se hizo con una fuerte base en los conocimientos teóricos previos. El hardware y el software que configuran los sistemas informáticos automatizados de tratamiento de la información han ido avanzando en paralelo. Inicialmente con una fuerte interdependencia, pero, conforme han evolucionado, con tendencia a abstraerse el uno del otro.

En el campo del software, a medida que su uso se iba expandiendo y generalizando, a la vez que se consolidaba la cultura científica asociada a su desarrollo, creció también la necesidad de aplicar y repetir las técnicas productivas que mejor funcionaban. De esta forma se conseguía cubrir la creciente demanda de uso y mejora de estos sistemas por parte de empresas, instituciones y usuarios en general. El paradigma inicial en el que el software se diseñaba casi a medida para cada aplicación y máquina y en el que la misma persona lo desarrollaba, explotaba y mantenía, prácticamente sin utilizar documentación ni un método reglado, resulta rápidamente inoperativo y queda obsoleto.

Surge así la ingeniería del software[4] con el objetivo fundamental de crear software que resulte útil y fiable, con un enfoque sistemático y disciplinado, aplicando los principios básicos de cualquier ingeniería. Van apareciendo los modelos, estándares, técnicas y procesos de producción asociados a esta ingeniería que van a ser utilizados por una gran cantidad de empresas dedicadas total o parcialmente a la creación de software. Se pasa del arte a la ciencia y la industria. Es el nacimiento de toda una rama científica que podríamos denominar como informática y un mundo empresarial productivo asociado. Conforme el volumen y complejidad de los sistemas informáticos crece, aquí se van a agrupar no solo los ingenieros de software sino también un gran número de especialistas, como los ingenieros de sistemas, de calidad del software, de seguridad informática, diseñadores, programadores y administradores, que intervienen a lo largo del proceso de creación, puesta en explotación, operación y mantenimiento del software.

2.1.1 Evolución de la distribución del esfuerzo en la creación de software

Los componentes clásicos de todo sistema de información son los recursos humanos, los recursos materiales, la información y el conjunto de normas, procedimientos y documentación del sistema. Bajo una concepción simplista podemos decir que los recursos materiales de todo sistema informático se caracterizan por dos grandes elementos que los diferencian de otros sistemas de información. Por un lado, nos encontramos con el hardware, todo el conjunto de equipos y dispositivos físicos que dan soporte a la recopilación, tratamiento, trasmisión y almacenamiento de la información; por otro, tenemos el software, el conjunto de instrucciones lógicas y matemáticas agrupadas en programas que, siguiendo un orden determinado en forma de algoritmos, nos permiten tratar y operar sobre los datos y la información, transformándolos y empleándolos para alcanzar los objetivos del sistema. Podríamos pensar que con estos dos elementos es suficiente, junto a las personas y los procesos, para configurar un sistema eficiente. En mi opinión, esto no es así, y a lo largo del tiempo ha surgido otro «componente» relacionado con la creciente complejidad y la necesidad de evolución y mejora de estos sistemas: el

[2] En la versión on-line del Diccionario de la Real Academia Española [40] podemos encontrar el termino «hardware» que se define como: equipo, conjunto de aparatos de una computadora.

[3] En la versión on-line del Diccionario de la Real Academia Española [40] podemos encontrar el termino «software» que se define como: conjunto de programas, instrucciones y reglas informáticas para ejecutar ciertas tareas en una computadora.

[4] Los conceptos de «crisis del software» e «ingeniería del software» surgen en el ámbito de la OTAN durante la Conferencia de Roma de 1968 [45].

mantenimiento. El orden expuesto, hardware, software y mantenimiento, responde a la evolución temporal no solo de la importancia que se les ha dado a estos elementos sino también de los costes y recursos empleados en cada uno de ellos, como puede verse en la Tabla 2-1.

	1950-60	1970-80	1985-
EQUIPOS	85%	25%	10%
DESARROLLO	10%	35%	25%
MANTENIMIENTO	5%	40%	65%

Tabla 2-1 Evolución de la distribución del coste de los sistemas informáticos en EEUU (tomada de [1])

Un determinado software es valorado positivamente por los usuarios si satisface los requisitos establecidos inicialmente. Pero lo que realmente marca la diferencia es que el usuario perciba que puede cambiar esos requisitos, no solo durante el desarrollo inicial, sino también durante la fase de explotación; y que los cambios se producen de una forma rápida y eficaz. De esta forma su sistema de información se convierte en una herramienta ágil y útil en su papel fundamental de orquestador de todos los procesos que se desarrollen en la empresa u organización.

En la industrialización y adopción de la ingeniería para el proceso de producción de software cabe destacar que existen ciertas diferencias con la ingeniería industrial clásica inherentes al producto final obtenido. El software debe ser diseñado, construido y probado para ser utilizado, pero es invisible, normalmente complejo y modificable. No tiene piezas de repuesto y no se desgasta por el uso, aunque tiene un ciclo de vida y deja de ser útil cuando las necesidades del cliente cambian o se detectan fallos en su diseño. No se fabrica en serie, excepto las herramientas ofimáticas y de uso personal para gestión o entretenimiento, sino que se desarrolla para satisfacer las necesidades específicas del sistema de información de un cliente.

El mantenimiento de las aplicaciones puede ser considerado como un asunto de mayor complejidad que el de cualquier otro producto material, como puede ser el hardware. La necesidad de uso de estándares de análisis y diseño, propios de la ingeniería del software, se hacen patentes para disminuir la generación de errores tanto en las fases iniciales de producción como durante el mantenimiento, una vez se alcanzan los estadios de producción.

2.2 Los paradigmas clásicos de desarrollo de software

En la referencia [2], página 15, podemos encontrar las fases generales a seguir en la resolución de problemas y, por tanto, la esencia en la práctica de la ingeniería del software para el desarrollo de aplicaciones:

- Entender el problema (fases de comunicación y análisis).
- Planear la solución (modelado y diseño del software).
- Ejecutar el plan (generación del código).
- Examinar la exactitud del resultado (probar y asegurar la calidad).

Estas fases determinan y agrupan, en términos generales, un conjunto de tareas y actividades que hay que realizar para obtener el producto final deseado, esto es, una aplicación útil y confiable. No obstante, el flujo de ejecución de este proceso creativo del software puede ser variable. Esta variabilidad es lo que nos determina los diferentes modelos de desarrollo de software existentes o ciclos de vida. Sin intentar ser exhaustivos, veamos en síntesis los más característicos (extraídos de la referencia [3]).

2.2.1 Ciclo de vida clásico

También denominado «en cascada» o «por fases» presenta un enfoque sistemático y secuencial en el desarrollo. Hasta que una de las fases no se ha completado no se aborda la siguiente. Cada fase requiere una información de entrada, unos procesos y proporciona unos resultados que sirven de entrada a la fase siguiente. Cada paso del ciclo se documentada de forma detallada y exhaustiva. El desarrollo se realiza de arriba abajo mientras que la realimentación se efectúa de abajo hacia arriba.

El nombre y número de fases varía de unos autores a otros, aunque en general pueden encontrarse las siguientes:

- **Análisis previo**. Corresponde a la definición del sistema a desarrollar y su planificación. Se realiza el análisis del sistema y de los requisitos del cliente (especificaciones generales, descripción funcional, restricciones…) así como la viabilidad y planificación del proyecto de desarrollo del software (necesidades de recursos humanos y materiales, estimación de tiempos, estimación de costes…).
- **Análisis funcional**. Se fijan los límites del sistema y se determina qué va a hacer. Se descompone el sistema en subsistemas, se determinan las entradas, procesos y salidas, se especifican los flujos y almacenes de datos. La fase finaliza con el documento «Especificación de Requisitos de Hardware».
- **Diseño**. Se determina cómo lo va a hacer el sistema, construyendo el modelo físico de datos y procesos que van a funcionar. Para ello se determinan detalladamente los datos a usar y se organiza su estructura, y se diseñan en detalle los procesos. Se finaliza con los documentos denominados «Cuadernos de Carga», que son los que se entregan a los programadores para la codificación de los programas.
- **Codificación**. En esta fase, también llamada de programación, se utilizan diversos lenguajes de programación para crear los programas y bases de datos, haciendo comprensibles para las máquinas de computación las especificaciones y procesos obtenidos en la fase anterior.
- **Pruebas**. La fase de pruebas puede dividirse, en su concepto más amplio, en otras dos, las unitarias y las de conjunto. En general cada aplicación consta de múltiples programas o módulos. Las pruebas unitarias realizan la validación de cada uno de esos programas o módulos comprobando que el código es correcto y no produce fallos funcionando de forma aislada. Una vez realizadas las pruebas unitarias es preciso ensamblar toda la aplicación y probarla en su conjunto, realizando lo que también se llaman pruebas integrales o de integración y pruebas funcionales. Además de las pruebas de validación o aceptación finales a realizar por los usuarios, puede considerarse necesario realizar también unas pruebas de verificación en las que un departamento de calidad realice un control exhaustivo del código para comprobar, no ya que los programas o módulos funcionen, sino que se cumplen los estándares o normas de desarrollo y programación establecidos en la organización. De esta forma se asegura la calidad del producto y se disminuyen los costos de mantenimiento.
- **Integración e instalación**. Una vez que finalizan las pruebas de forma satisfactoria y el cliente acepta el producto hay que ponerlo en servicio. Si la aplicación desarrollada es única o la primera de un sistema el proceso finaliza con la instalación de la misma en el entorno hardware en que va a ser utilizada. Si por el contrario hay que añadirla a un sistema que ya se encuentra en explotación hay que realizar su integración, comprobando que no se producen incompatibilidades con los productos ya en explotación. La necesidad de que un programa sea integrado en un sistema ya existente es algo a tener muy presente ya en las fases de análisis y, sobre todo, de diseño.
- **Explotación y mantenimiento**. Cuando la aplicación se entrega al cliente y se integra en el sistema informático existente se inicia la fase de uso y explotación de la misma por parte de los usuarios. Con el paso del tiempo es posible que surjan necesidades de cambios en sus funcionalidades o bien se detecten fallos por una mala definición de requisitos o por un mal diseño. Ese es el momento en que procede realizar el mantenimiento. Si los cambios a

realizar son de tal magnitud que aconsejasen la retirada de la aplicación y su sustitución, podemos considerar que hemos llegado al final del ciclo de vida del producto.

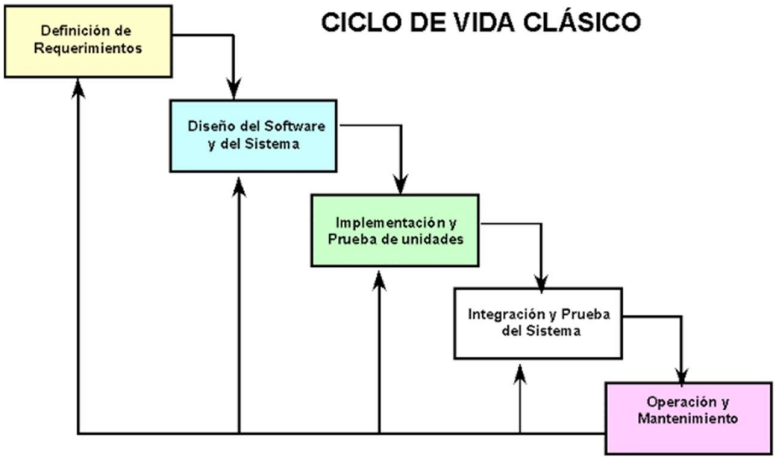

Figura 2-1 Ciclo de vida clásico (tomado de [4])

El ciclo clásico es el más antiguo y quizás el más utilizado todavía en ingeniería del software. Sus fases coinciden de forma genérica con las definidas para cualquier otro modelo, pero presenta una serie de problemas, como la rigidez en el establecimiento de requisitos, la falta de producto hasta que no se finalizan todos los procesos o los importantes retrasos provocados por cualquier fallo, que pueden aconsejar la adopción de otros paradigmas.

2.2.2 Ciclo de vida en V

Es una variante del modelo clásico en el que a medida que se avanza hacia abajo, por el lado izquierdo de la V (ver Figura 2-2), los requisitos que definen el problema y su solución van mejorando, dando lugar a los modelos oportunos. Una vez que se ha creado el código, se sube por el otro lado de la V realizando las pruebas unitarias, de integración, funcionales y de verificación y validación correspondientes[5].

[5] En [53] encontramos una breve explicación de los diferentes tipos de pruebas de software que se pueden realizar. Entre ellos se destacan: las pruebas unitarias, que comprueban el correcto funcionamiento de unidades individuales de código, sin dependencias, es decir, de las funciones o métodos de las clases, componentes o módulos del software, constituyendo las pruebas más fáciles de automatizar; las pruebas de integración, que comprueban que los diferentes módulos o servicios de la aplicación funcionan correctamente cuando trabajan en conjunto y considerando todas sus dependencias, se realizan tras las unitarias y son más costosas de automatizar; a continuación se realizarían las pruebas funcionales o de sistema, que se centran en comprobar que los componentes o módulos de la aplicación, considerados como una caja negra, funcionan de forma adecuada proporcionando las salidas esperadas; las pruebas de verificación son pruebas «end-to-end», que replican el comportamiento de los usuarios en un entorno lo más parecido al final, y responden a la pregunta de si se está construyendo el producto correctamente, cumpliendo con los requisitos especificados; las pruebas de validación o aceptación son también pruebas «end-to-end» pero tratan de responder a la pregunta de si se construye el producto correcto, es decir, si lo que se ha especificado es lo que realmente se quería.

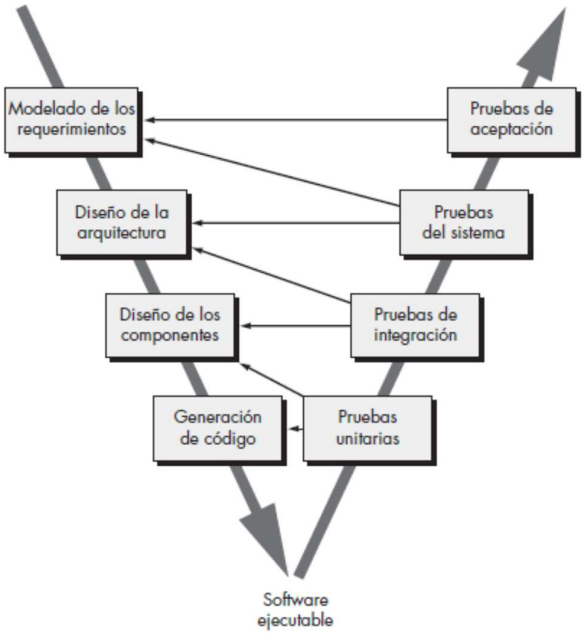

Figura 2-2 Ciclo de vida en V (tomado de [2])

2.2.3 Ciclo de vida prototipado

Este paradigma, también denominado iterativo, viene a intentar solventar el problema expuesto del ciclo clásico de la rigidez en la definición inicial de requisitos por parte del cliente. Se parte de una exposición de requisitos genérica por parte del usuario y se pasa a un diseño rápido enfocado principalmente en los aspectos del producto final más visibles para el usuario (interfaces gráficas, formatos de entrada y salida…). Se genera un prototipo lo antes posible, que no es más que un modelo o representación del producto final que incorpora de forma parcial parte de los componentes que lo definirán al finalizar el proceso. El prototipo es evaluado por el cliente con una doble finalidad: comprobar que lo desarrollado se ajusta a sus necesidades y servir como elemento para identificar los requisitos faltantes. Este ciclo se repite retocando y refinando el modelo que se va obteniendo hasta que se obtiene el producto final. Una vez obtenido se procede, como en el modelo clásico, a realizar la integración, instalación, explotación y mantenimiento.

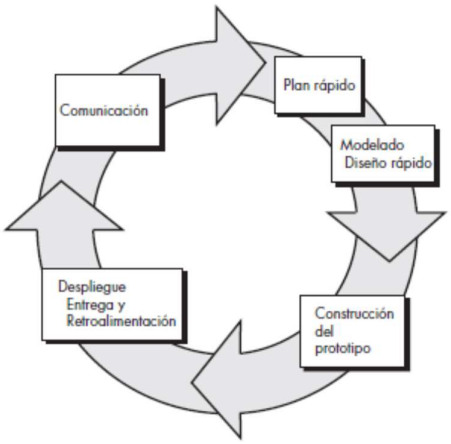

Figura 2-3 Ciclo de vida prototipado (tomado de [2])

2.2.4 Ciclo de vida en espiral

Puede considerarse como una mejora que aúna los ciclos clásico y prototipado. En este paradigma se definen cuatro fases fundamentales que se van repitiendo en una espiral de perfeccionamiento progresivo o evolutivo. En la primera de comunicación y planificación se recogen los requisitos y se evalúan los objetivos, restricciones y alternativas. Seguidamente se procede al modelado y la construcción del producto. Finalmente se produce la entrega al cliente para valoración del producto y sugerencia de modificaciones con los que iniciar una nueva vuelta en la espiral con la que se genera una nueva versión del producto.

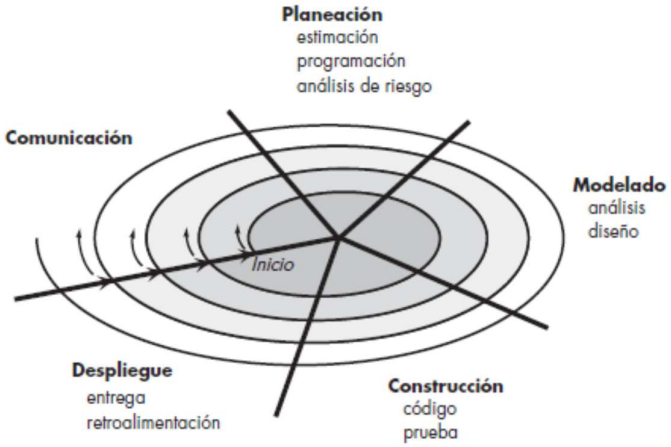

Figura 2-4 Ciclo de vida en espiral (tomado de [2])

2.2.5 Ciclo de vida orientado a objetos

Este paradigma toma en consideración el uso de técnicas de programación orientada a objetos, introducidas en los años 80 del siglo pasado, para el ciclo de vida del software. Sus fases no se diferencian de las generales de un modelo clásico, pero se desarrollan desde una óptica diferente. En lugar de considerar los problemas desde un punto de vista de entrada-proceso-salida, se introduce la clase, que es una representación abstracta en la aplicación de un ente o concepto real y que viene caracterizada por un conjunto de variables o propiedades y de métodos o funciones. Un objeto es la materialización o instancia de una clase, que puede realizar una serie de acciones a petición de otros objetos.

2.3 El desarrollo de software ágil

Desde el inicial ciclo de vida clásico, los diferentes modelos de desarrollo de software han intentado lidiar con una serie de problemas que lastraban a las organizaciones que los utilizaban. Se tenía una confianza excesiva en las especificaciones iniciales y se era poco flexible con las necesidades expresadas por el usuario una vez iniciado el proyecto, lo que provocaba, en muchos casos, insatisfacción con los productos finales. Se tendía a generar productos monolíticos, poco propensos a cambios y actividades de mantenimiento. Los retrasos y la imposibilidad de realizar una estimación de tiempos adecuada se incrementaban con la complejidad de los proyectos. Se producía un gran volumen de documentación, que consumía mucho tiempo y recursos, sin que se apreciase claramente el beneficio de esta actividad en la obtención de un software útil y de calidad.

En la búsqueda de una solución a todo esto surge el concepto de desarrollo ágil («Agile» en su denominación en inglés) a principios del siglo XXI. En general, estas metodologías de producción de software pueden considerarse como variaciones del ciclo de vida en espiral en las que se enfatiza la sencillez, rapidez, iteración y agilidad en la creación de aplicaciones. El objetivo es poder entregar código utilizable en intervalos que se miden en unas pocas semanas. Sus principios los podemos encontrar en el denominado «manifiesto ágil» [5] que es un documento alumbrado por 17 destacados expertos y altos directivos en el ámbito de las TI en 2001. En síntesis, estos principios son:

- La primera prioridad es satisfacer al cliente, y esta satisfacción se consigue siendo capaz de entregar un producto con valor, el software, de forma temprana y continua.
- Se puede entregar software que resulte funcional y útil frecuentemente, en periodos de entre dos semanas y dos meses.
- Los usuarios y los desarrolladores trabajan juntos durante todo el proyecto.
- El componente humano es muy importante en el desarrollo de los proyectos y hay que lograr contar con individuos motivados.
- La forma más eficaz de comunicación con los equipos de desarrollo es la conversación cara a cara, que es también la forma más eficaz de comunicación interna.
- La obtención de software que funciona es la principal medida de progreso.
- Hay que buscar un desarrollo sostenible, entendido como la capacidad de mantener un ritmo constante de actividades tanto por parte de desarrolladores como por usuarios y promotores.
- La búsqueda continua de la excelencia técnica y el buen diseño mejora la agilidad.
- La simplicidad es esencial.
- Las mejores arquitecturas, requisitos y diseños surgen de equipos que se autoorganizan.
- La realimentación es fundamental. Los equipos deben reflexionar a intervalos regulares sobre como ser más efectivos, ajustando y perfeccionando su comportamiento.

Las ideas fuerza en todo desarrollo ágil son:

- Que las personas y sus interacciones deben predominar sobre los procesos y las herramientas.
- Que es más importante tener software que funcione que un gran número de documentos detallados y extensos.
- Que es más importante mantener el contacto con el cliente que atenernos al contenido de un contrato y a unos requisitos.
- Y, finalmente, que es mejor cualidad la capacidad de adaptación al cambio que el ceñirnos a un plan elaborado.

Como veremos más adelante, estos principios e ideas están totalmente en línea con la cultura DevOps y, en mi opinión, son estas metodologías de desarrollo de principios de siglo las que darán pie al surgimiento de DevOps a partir de la segunda década.

En un artículo publicado en la página web del BBVA [6], que podemos considerar como una de las grandes empresas en el sector bancario español y un ejemplo del crecimiento y uso extensivo de las TI como forma de aumentar el valor ofrecido a los clientes, podemos leer que ágil trasciende lo que es una metodología, una mera herramienta de desarrollo, y comienza a parecerse a una filosofía que busca implantar formas de trabajar, de organizarse y de concebir los proyectos y el software como productos que tienen que satisfacer unas necesidades de los clientes que cambian a velocidades cada vez mayores.

2.3.1 Los paradigmas ágiles

En la actualidad existen diferentes paradigmas de desarrollo que siguen los preceptos ágiles. Entre ellos se pueden citar, como ejemplos, los siguientes:

- «Programación Extrema (XP)», formulada por Kent Beck en 1999 y basada en un conjunto de reglas y buenas prácticas enfocadas en la retroalimentación continua y en la mejora de la comunicación para el desarrollo en ambientes donde los requisitos son muy cambiantes e

imprecisos. Durante el desarrollo del software se hace más hincapié en la capacidad de adaptación, considerando como natural los cambios de requisitos, que a la rígida planificación. Las características fundamentales que pueden destacarse son: la utilización de un ciclo de desarrollo iterativo, con la producción de pequeñas mejoras de forma incremental; el empleo de entregas frecuentes para corregir los errores; la utilización de pruebas unitarias continuas y la programación en parejas para favorecer estas pruebas; la integración del cliente en el equipo de desarrollo, que va a permitir, junto a los ciclos cortos, una realimentación realmente efectiva; el empleo de las revisiones y la reprogramación como manera de mejorar la legibilidad y mantenibilidad del código; y, finalmente, la simplicidad en el código, como la mejor característica que asegura el funcionamiento de las aplicaciones [7].

- «Kanban»[6], que se basa en el uso de tarjetas (pueden ser simplemente «post-it») sobre un panel para visualizar las tareas a realizar y su estado dentro del flujo general de ejecución de tareas definido. Se caracteriza por la búsqueda de la calidad por encima de la rapidez, lo que la distingue de otras metodologías ágiles; en la búsqueda de la simplicidad, eliminando todo lo que pueda resultar superfluo; en la mejora continua; y en la flexibilidad, siendo capaz de aceptar y priorizar nuevas tareas en cualquier momento [8].
- «Scrum», es quizás el más extendido hoy en día, como se ve en el gráfico de la Figura 2-5, elaborado en 2020 a partir de una encuesta realizada en un ámbito de más de 40.000 profesionales.

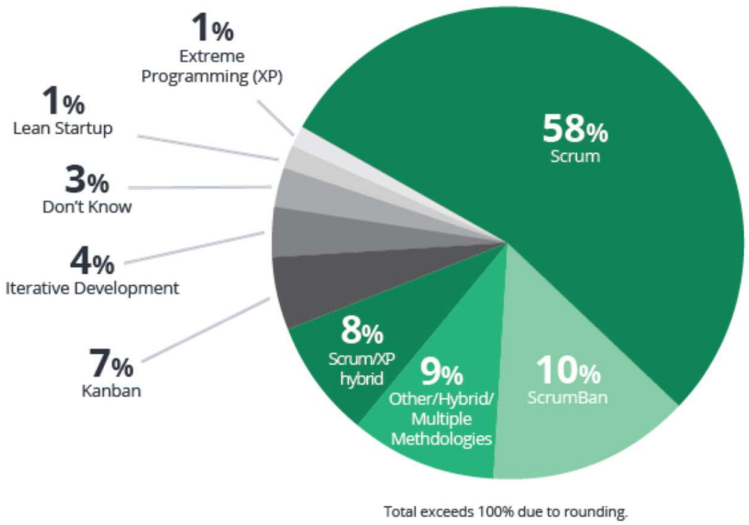

Figura 2-5 Uso de metodologías ágiles (tomado de [9])

«Scrum» parte de la creación de una lista priorizada de características y capacidades necesarias para desarrollar el producto buscado denominada «product backlog» en inglés. A partir de aquí, equipos autoorganizados y multifuncionales realizan las tareas identificadas mediante iteraciones temporales sucesivas denominadas «sprints». Al final de la iteración hay que obtener un producto cuyas funcionalidades o características se van incrementando con cada nueva iteración. Las reuniones que marcan el comienzo y final de los «sprints», así

[6] El origen de esta metodología se encuentra en el proceso productivo utilizado por la empresa Toyota denominado «Just-in-Time» (JIT), en el que se utilizan tarjetas para identificar las necesidades de material en la cadena de producción [46].

como las reuniones diarias de seguimiento, son esenciales para el funcionamiento de este modelo poniendo de manifiesto la importancia de la comunicación entre los diferentes actores [10]. En la Figura 2-6 puede verse un esquema del funcionamiento de esta metodología.

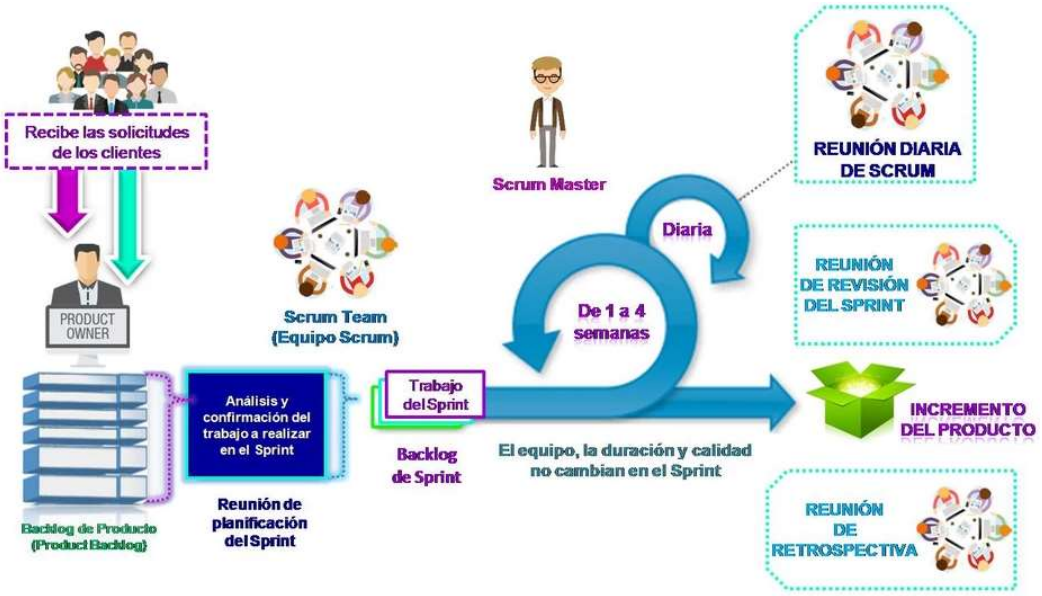

Figura 2-6 Scrum (tomado de [11])

2.4 DevOps

Como comenté en la introducción, DevOps es un acrónimo compuesto por las palabras «development» y «operations» que hacen referencia a los equipos de desarrollo y de operaciones que intervienen a lo largo del ciclo de vida del software. Estos equipos son responsables, respectivamente, de crear o modificar las aplicaciones, y del mantenimiento de la infraestructura TIC y la puesta en explotación y utilización de esas aplicaciones. DevOps es mucho más que un paradigma de desarrollo de software y podría definirse como toda una «cultura», una forma de entender y vivir la producción y entrega del software, que busca la eliminación de barreras y compartimentos estancos[7], como los que tradicionalmente surgen entre los equipos de desarrollo[8] y operaciones[9], y la consecución de un flujo continuo de desarrollo, entrega y despliegue de software que resulte ágil y lo más automatizado posible, no solo durante el diseño y entrega inicial sino también durante el ciclo de vida completo.

La adopción de metodologías ágiles por los equipos de desarrollo en su búsqueda para satisfacer a los usuarios, crear valor añadido[10] de forma continua y resultar altamente competitivos, lleva a que se incremente de forma considerable la rapidez en la producción de software por parte de estos equipos. Las funcionalidades ofrecidas por las aplicaciones no se ven como algo dogmático e inamovible, sino que pueden y deben modificarse continuamente adaptándose a nuevos requerimientos que, a su vez, son

[7] Tradicionalmente las organizaciones tienden a adoptar estructuras funcionales en las que los equipos se conforman en función del tipo de trabajo que realizan. Esto puede llevar a la formación de elementos aislados que en inglés se denominan «silos».

[8] Los equipos de desarrollo necesitan el cambio. Producen nuevo software o lo modifican y quieren que esté en producción lo antes posible.

[9] Los equipos de operaciones tienden a evitar el cambio. Son los responsables de que en los entornos de producción todo funcione de forma estable por lo que una vez que todo corre de forma adecuada no son receptivos a que se modifiquen cosas.

[10] «El valor añadido o valor agregado es la utilidad adicional que tiene un bien o servicio como consecuencia de haber sufrido un proceso de transformación» (https://economipedia.com/definiciones/valor-anadido.html, visitado el 29/12/2020).

volátiles y limitados en el tiempo. No obstante, su capacidad para responder a las necesidades de los usuarios de modificación y mejora continuas en cortos periodos de tiempo encuentra un significativo cuello de botella en la puesta en explotación. Los equipos de operaciones y otros, como los de aseguramiento de la calidad y los de seguridad, no sienten la necesidad de cambio continuo y rápido en las aplicaciones; por el contrario, su mundo ideal es el de la estabilidad y seguridad que se consigue con largos procesos de comprobación, configuración y puesta en explotación.[11]

2.4.1 Los orígenes de DevOps

Los años 50 y 60 del siglo XX vieron surgir la metodología «Lean» en el mundo de la producción industrial de bienes[12]. En síntesis, se buscaban tres objetivos principales: eliminar cualquier aspecto que pudiese resultar accesorio y que no aportase valor durante la producción; revisar y mejorar continuamente los procesos utilizados; y la promoción de la innovación y la iniciativa, sobre todo entre el personal de los escalones más bajos de la organización relacionados directamente con la producción. En la creación de valor[13] adquiere gran importancia la capacidad de aprender, de adaptarse y mejorar; y para aprender y adaptarse lo mejor es ser capaces de producir rápidamente. Los cimientos de la metodología descansan en dos conceptos claves: la mejora continua y la importancia del factor humano. Lean perdura en nuestros días y se ha expandido a otros ámbitos como el de la producción de software, pudiendo considerarse como el precursor de los paradigmas ágiles[14] [12].

Desde principios del siglo XXI se han venido utilizando las metodologías ágiles en la creación de aplicaciones, pero estas no lograban los resultados esperados, sobre todo en grandes empresas donde en la producción y explotación de software intervenían no solamente equipos de desarrollo sino también de operaciones. La existencia de intereses contrapuestos, dinamismo frente a estabilidad, y la falta de comunicación efectiva conformaban la tradicional barrera entre estos equipos. La producción de nuevas funcionalidades podía medirse en semanas mientras que su puesta en explotación lo hacía en meses, produciéndose efectos negativos tanto en los procesos internos, que se retardaban y perdían sincronismo, como de cara al usuario final, que no veía cumplidas sus perspectivas. Se hacía evidente la necesidad de romper esa barrera.

El concepto DevOps fue alumbrado por el consultor independiente de TI, de origen belga, Patrick Debois. Este consultor, que desde el mundo de los servidores, las redes y la seguridad se introdujo también en el mundo del desarrollo software, participó en 2007 en un proyecto de migración de un centro de procesos de datos (CPD) del gobierno belga durante el que pudo apreciar que numerosos problemas eran causados, fundamentalmente, por la falta de comunicación y cooperación entre los equipos de

[11] El libro «The Phoenix Project» [25], a pesar de ser una novela de ficción sobre el mundo TI, muestra muy bien la decadencia a la que se ve sometida una empresa que sigue anclada en los viejas metodologías y procesos, la tensión a que se ven sometidos los equipos de operaciones casi a diario para que todo siga funcionando, los desencuentros entre los compartimentos estancos existentes en el departamento de las TI dentro de una empresa (desarrollo, seguridad de la información, operaciones…) y como la falta de comunicación y de compromiso entre ellos y con el resto de departamentos aboca al desastre. La novela muestra estrategias de adopción de la cultura DevOps y como al incrementar la agilidad en todos los procesos, mejorar la comunicación rompiendo con los compartimentos estancos y compartiendo información y experiencias, se contribuye enormemente a que la organización alcance sus objetivos y progrese.

[12] La metodología «Lean» surgió en la empresa japonesa Toyota con la creación y uso del llamado «Toyota Production System (TPS)», que significó toda una revolución en la forma de entender el mundo de la producción de bienes en las décadas de los 50 y 60 del siglo XX. Los conceptos de este sistema se mantienen hoy en día y se han expandido desde el mundo productivo a otros ámbitos.

[13] «La creación de valor es la capacidad que tienen las empresas o sociedades para generar riqueza o utilidad por medio de su actividad económica. En el ámbito de la dirección estratégica, se define como el principal objetivo de las sociedades mercantiles, así como su razón de ser» (https://economipedia.com/definiciones/creacion-de-valor.html, visitado el 29/12/2020).

[14] Una de las diferencias más notables entre Lean y Agile es la forma en que se considera que puede satisfacerse mejor a los clientes: mientras en el primero se considera que la eliminación de lo superfluo es la principal forma de mejorar la aportación de valor para el cliente, en Agile se estima que lo es la flexibilidad en los procesos para responder rápidamente a la retroalimentación procedente de cualquier participante en el desarrollo [51].

desarrolladores de aplicaciones y los equipos de operaciones (básicamente los equipos de administradores de sistemas) [13]. En 2008, en una conferencia celebrada en Toronto dentro de un ciclo dedicado a las metodologías ágiles, Debois impartió una presentación titulada «Agile Infraestructure & Operations» [14] en la que introdujo los conceptos de separación y problemas entre los equipos de desarrollo, por un lado, y los encargados de infraestructuras y operaciones, por el otro; y como, mientras en los primeros se iban implementando los conceptos ágiles en sus metodologías, los segundos carecían de ellos. Este deseo de incluir a los equipos de operaciones en el mundo «Agile» del desarrollo y de mejorar su comunicación con el resto de equipos que intervenían en la producción de software es lo que dio pie al nacimiento del concepto DevOps. En la Figura 2-7 pueden observarse de forma gráfica las barreras existentes entre los diferentes grupos de actores que intervienen en el desarrollo de aplicaciones y los ámbitos, no exactamente iguales, que abarcan las metodologías ágiles y DevOps.

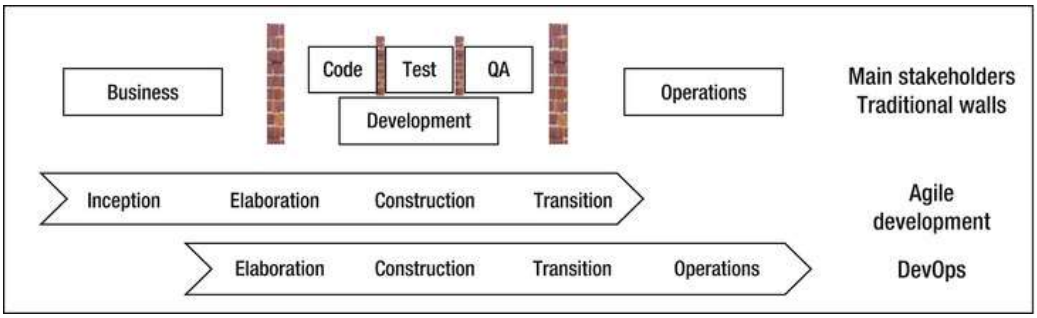

Figura 2-7 Ámbitos en los que se usa desarrollo ágil y DevOps (tomado de [15])

En 2009 Debois promovió el evento denominado «DevOpsDays» [16], siendo esta la primera vez que aparece el término DevOps. El evento tuvo lugar en Gante, Bélgica, y desde entonces se ha venido celebrando varias veces al año en diferentes ciudades del mundo. El conjunto de conferencias técnicas que lo conforman abarca temas muy diversos relacionados con el desarrollo de software, las operaciones de infraestructura TI y las intersecciones entre ellos, en una amalgama en la que podemos encontrar asuntos relacionados con la automatización, las pruebas, la seguridad y la cultura organizativa, pero siempre con el objetivo principal de incrementar la calidad en la entrega del software al usuario final.

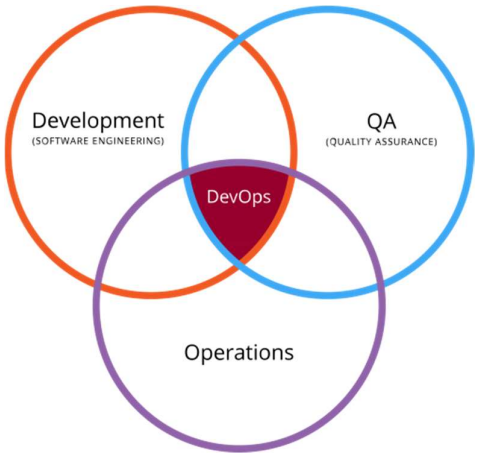

Figura 2-8 DevOps como cooperación entre desarrollo, operaciones y calidad (tomado de [17])

2.4.2 En qué consiste DevOps

DevOps puede definirse como una cultura para el desarrollo y puesta en explotación de software de calidad de forma rápida, alineando el desarrollo con la entrega, operación y mantenimiento de las aplicaciones, y favoreciendo la comunicación efectiva entre los equipos de desarrollo y los de operaciones. Esta definición como cultura, como forma de entender y actuar, implica también que es algo ambiguo y falto de concreción. Es decir, no es una metodología en la que encontremos directrices para definir los pasos a seguir en la producción de aplicaciones, no es una herramienta o conjunto de herramientas o tecnologías concretas que se puedan instalar en una determinada infraestructura TI como soporte para los procesos de producción, no es un nuevo departamento a implantar en la organización ni un nuevo puesto técnico o directivo a cubrir. Como cultura se compone de un conjunto de prácticas que se consideran adecuadas y buenas, de orientaciones para diseñar procesos, de una determinada mentalidad que se debe compartir entre los diversos equipos que intervienen y que facilitará la comunicación entre ellos.

En una fecha tan próxima a los inicios de DevOps como 2013, una de las grandes empresas productoras de software del momento, CA Technologies[15], encargó a Vanson Bourne, una empresa especializada en la investigación en el mercado tecnológico, una encuesta que se realizó a 1.300 ejecutivos de grandes organizaciones del ámbito de las TI, repartidas en 21 países diferentes, en la que se exploraban elementos clave a cerca de la conciencia existente sobre DevOps, su adopción, implementación y beneficios. Un resumen de los resultados de esta encuesta puede consultarse en la referencia [18]. Para hacernos una idea de en qué consiste DevOps podemos observar la Figura 2-9, extraída de la encuesta, donde se muestran, de acuerdo con la opinión de los encuestados, los componentes principales que definen DevOps.

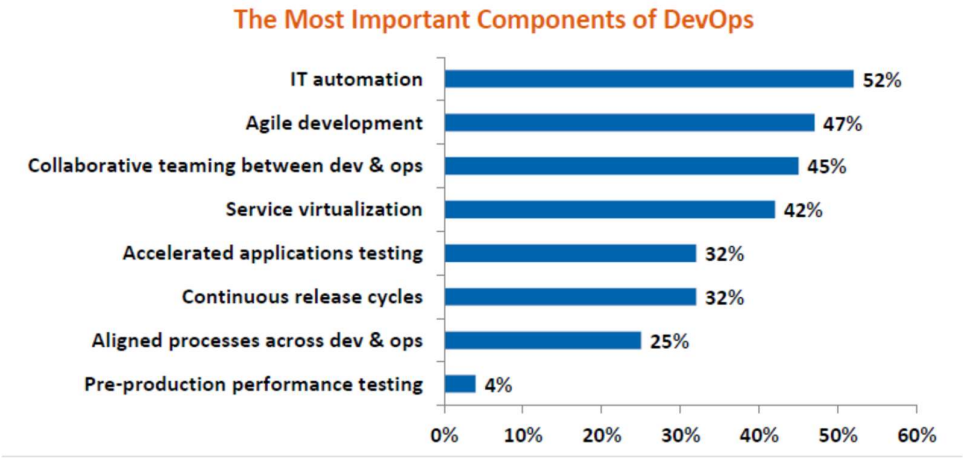

Figura 2-9 Los componentes más importantes de DevOps (tomado de [18])

Aquí podemos encontrar los pilares en los que se sustenta DevOps y que se van a utilizar siempre con la doble finalidad de obtener aplicaciones de forma rápida y asegurando su calidad:

- La búsqueda de la automatización en todas las fases del proceso de desarrollo y explotación, que es lo que lleva al amplio uso de herramientas tecnológicas, al objeto de evitar tareas repetitivas que pueden llevar mucho tiempo e introducir errores sin aportar un valor real.

[15] CA Technologies (Computer Associates) era una de las compañías productoras de software más grandes del mundo en entorno mainframe. En 2018 fue adquirida por Broadcom.

- El empleo de metodologías ágiles, principalmente en el desarrollo, pero no necesariamente circunscrito a este ámbito puesto que adaptándolas pueden utilizarse, por ejemplo, en proyectos de evolución de la infraestructura TI y en otro tipo de proyectos en la organización [19].
- La formación de equipos en los que el personal de desarrollo y el de operaciones trabajen de forma colaborativa, así como la mejora y alineamiento de los procesos en los que intervienen estas dos áreas de las TI.
- El amplio uso de la virtualización. DevOps en su búsqueda de tecnologías y métodos que mejoren la rapidez y automatización favorece el empleo de la virtualización y de las últimas tendencias en este sector como son los servicios en la nube, los contenedores y la computación sin servidor[16]. Estas tecnologías presentan a los servicios de la infraestructura TI de una forma más abstracta y simplificada, ayudando a que los equipos de desarrollo sean capaces de autoabastecerse de lo que necesiten de forma más sencilla y permitiéndoles focalizarse en escribir código.
- El empleo de técnicas para pruebas que sean más rápidas y ágiles. Desde el inicio del desarrollo se debe tener conciencia de favorecer el automatismo para la realización de todo tipo de test sobre el código.
- La creación de un ciclo continuo de entregas. El ciclo de vida de una aplicación no termina con la entrega inicial. El diseño de esta debe permitir un ciclo continuo de variaciones y mejoras favorecido por mecanismos efectivos de realimentación durante su uso. Las primeras entregas deben realizarse de forma rápida. Las sucesivas también deben ser lo suficientemente ágiles como para satisfacer las necesidades de los clientes y mejorar la competitividad de la organización.

El dinamismo característico que propugna DevOps conlleva como corolario la necesidad de descomponer las aplicaciones en pequeños fragmentos, con el menor grado de acoplamiento posible, ya que esta es la forma más eficiente y sencilla de aumentar la agilidad tanto en el desarrollo como para el mantenimiento. Esto lleva a considerar la arquitectura orientada a servicios (SOA) y, en mayor medida, a su evolución en arquitectura de microservicios[17] (ver la Figura 2-10) como las arquitecturas para desarrollo de software que mejor se adaptan a la filosofía DevOps.

[16] En la referencia [47] encontramos un artículo con una buena explicación de qué es la computación sin servidor o arquitectura «serverless». Esta se basa en el uso de contenedores en la nube para alojar de forma dinámica, creándo y destruyendo contenedores según la ocurrencia de determinados eventos, las denominadas «funciones» (que son los trozos de código, servicio o microservicio, que los programadores quieren poner en explotación). Este tipo de servicios en la nube se denominan FaaS («Functions as a Service»). El programador solo debe preocuparse de desarrollar las funciones y puede abstraerse de la gestión de la infraestructura sobre la que se ejecutarán. Serverless favorece el empleo de arquitecturas basadas en microservicios, facilita la gestión del ciclo de vida y los despliegues continuos de las aplicaciones, y simplifica los «rollbacks» (vuelta atrás en la instalación de nuevo software) en caso necesario. Las principales compañías de servicios en la nube ofrecen esta tecnología y, como ejemplo, podemos citar AWS Lambda de Amazon, Google Cloud Functions y Azure Functions de Microsoft.

[17] En un artículo publicado por la empresa Red Hat [48] podemos leer que los microservicios aportan una serie de ventajas que están alineados con los objetivos de DevOps. Entre estas ventajas puede destacarse el disponer de aplicaciones listas para comercializarse más rápidamente, ya que se reducen los ciclos de desarrollo significativamente; su gran capacidad de expansión, dada la facilidad para multiplicar su implementación en la infraestructura disponible; su alta capacidad de recuperación, en base a su alto nivel de desacoplamiento; su facilidad de implementación, dado que aportan el mayor grado de modularidad a las aplicaciones, aunque también implican mayores necesidades de coordinación entre componentes; el alto grado de accesibilidad, ya que al ser de pequeño tamaño y orientadas a satisfacer pequeñas funcionalidades son fáciles de entender por los programadores y, por tanto, son más fáciles de actualizar y mejorar utilizando ciclos rápidos.

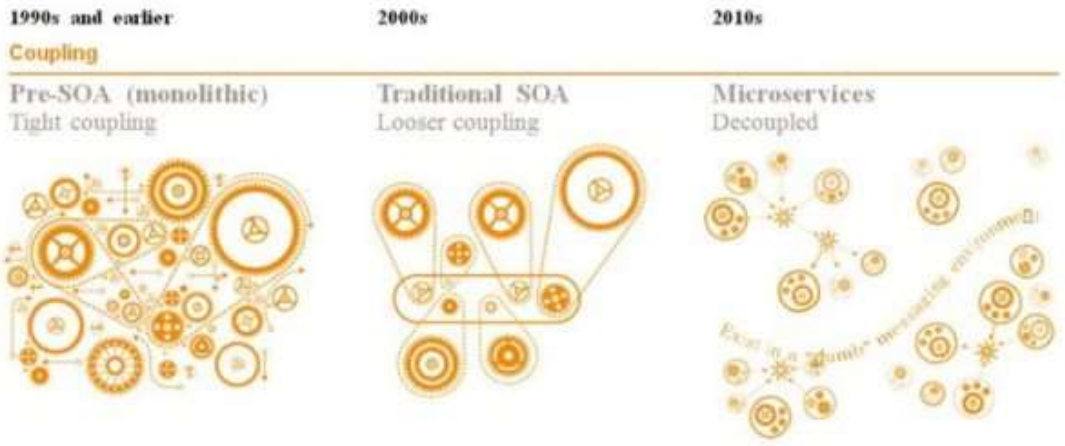

Figura 2-10 Evolución de las arquitecturas de desarrollo de software (tomado de [20])

Otra consecuencia que se podría destacar afecta al cambio de mentalidad con la que se afrontan determinados temas que interesan a los programadores y a los técnicos de sistemas. Ya he destacado en el párrafo anterior el cambio de concepción en la arquitectura para desarrollar los productos desarrollados, pasándose de una visión clásica de crear aplicaciones completas, que pueden estar compuestas por funciones, clases o utilizar alguna técnica determinada para conseguir algo de modularidad, a aplicaciones concebidas como el resultado de la interacción de una serie de servicios, o, mejor aún, de microservicios. Otro cambio importante es la concepción de los errores. Se pasa de tratar que estos no aparezcan a tratar de descubrirlos cuanto antes. El dinamismo y el valor positivo que se da a la iniciativa hacen que los errores ya no se vean como un fracaso, pasan a ser algo propio del proceso creativo y simplemente hay que descubrirlos y corregirlos lo antes posible. Con DevOps, la confianza en el sistema automatizado de pruebas y en la velocidad a la que pueden completarse los ciclos de creación y mejora hace que la aparición de errores no sea algo crítico como sucede en el desarrollo clásico, donde la aparición de fallos suele ocasionar grandes retrasos y obliga a retrotraerse, de forma no prevista, en el flujo general de desarrollo.

Otro gran cambio de mentalidad propio de DevOps es la visión del producto creado. Este ya no se concibe como el resultado de la ejecución de un proyecto que termina con la entrega del software solicitado por el cliente. La entrega del producto en DevOps supone un paso más en un ciclo de vida que no finaliza hasta su retirada del servicio. Ciclo durante el que cualquier miembro del equipo que lo ha creado y puesto en funcionamiento seguirá involucrado.

2.4.3 La actualidad y el futuro de DevOps

Existen diversos estudios que se publican anualmente que informan de la evolución de DevOps en el mundo de las TI, de las empresas dedicadas a la producción de software o con importantes departamentos en este sector, y que se realizan mediante encuestas entre los profesionales correspondientes. Entre ellos podemos destacar el de la empresa «Puppet»[18], que comenzó sus informes en 2013 y que ha publicado recientemente el «2020 State of DevOps Report» [21], contando con más de 2.400 expertos; y el «Accelerate State of DevOps 2019» [22], realizado entre 1.000 profesionales por «DevOps Research & Assessment» (DORA), empresa que a finales de 2018 fue adquirida por «Google

[18] «Puppet» es una empresa que dispone de productos, con un alto grado de implantación, dirigidos a la gestión de la configuración de los recursos en infraestructuras TI y que puede considerarse muy orientada a la automatización, tanto en el ámbito de la gestión de esos recursos como en el del proceso de despliegue de aplicaciones. Asimismo, puede considerarse como un importante valedor de los conceptos de DevOps y de la infraestructura como código («Infraestructure as Code» o IaC).

Cloud» y que supone el último de una serie iniciada en 2014. Estos informes, que aparecieron desde casi los inicios de DevOps, llegan a un conjunto de conclusiones que nos dan una idea de cuál es la actualidad y cuál puede ser el futuro de DevOps. Algunas de estas conclusiones las desarrollo a continuación.

Se ha incrementado considerablemente el número de empresas que declaran adoptar la cultura DevOps y, en particular, este incremento es superior entre las que lo hacen de forma más completa[19]. En los últimos años, como puede apreciarse en la Figura 2-11 y la Figura 2-12, el número de entidades que pueden considerarse con un mayor compromiso y desempeño en el uso de DevOps es el que ha experimentado un aumento más significativo, aunque la mayoría de organizaciones siguen manteniéndose en niveles medios de evolución en la implantación de esta cultura. Esto significa que DevOps sigue un proceso de implantación progresivo pero constante y cada vez más amplio en el sector de las TI. No constituye una revolución sino una evolución que se va imponiendo poco a poco.

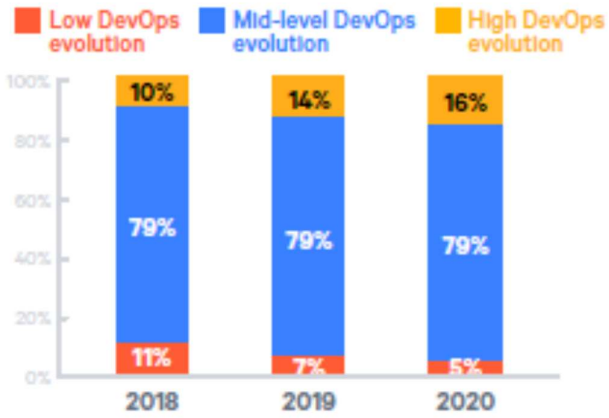

Figura 2-11 Agrupación de organizaciones en función del grado de adopción de DevOps (tomado de [21])

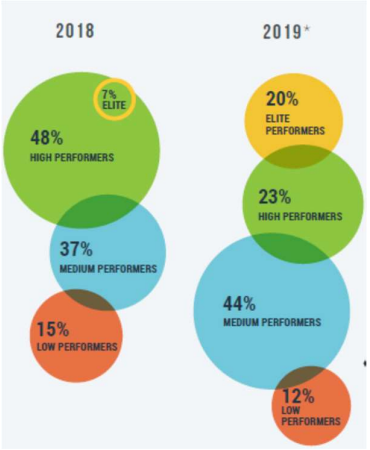

Figura 2-12 Agrupación de organizaciones en función del grado de adopción de DevOps (tomado de [22])

Una de las dos áreas que mejor pueden ayudar a las organizaciones que quieren progresar en la implantación de DevOps, y en las que, por tanto, deberían focalizarse, es el uso de servicios

[19] Los informes suelen clasificar a las organizaciones en función del grado de implantación en ellas de la cultura DevOps. El informe de Google Cloud contempla cuatro grupos: bajo, medio, alto y élite; y el de Puppet tres niveles: bajo, medio y alto.

proporcionados por plataformas de recursos TI internas[20] para la creación y entrega de software, lo que justifica su mantenimiento y el de los equipos que las gestionan. Un resultado destacable de los análisis en este campo es que aquellas organizaciones en que los equipos de operaciones consideran la plataforma como un producto sobre el que hay que reflejar los requisitos de los usuarios (los equipos de desarrollo) y al que hay que mejorar y evolucionar, proporcionando ayuda al uso y posibilidades de experimentación de sus capacidades a esos usuarios, son los que obtienen mejores resultados. Los principales problemas para proveer una plataforma interna son la falta de tiempo, la falta de estandarización y la falta de suficientes conocimientos técnicos entre el personal dedicado a su gestión. En las organizaciones que se ubican en los niveles más bajos, a los problemas anteriores hay que añadir la falta de impulso desde los puestos directivos.

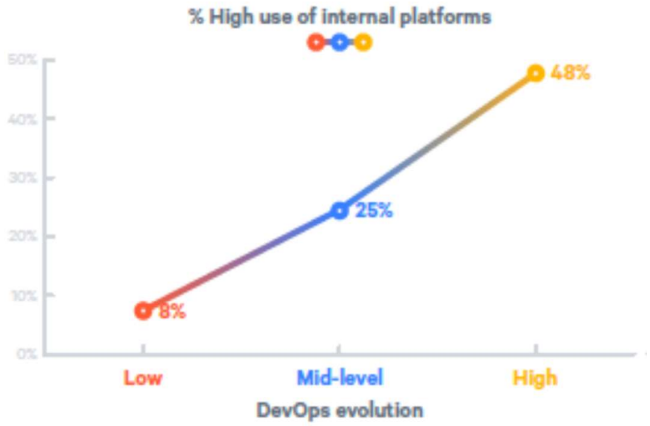

Figura 2-13 Relación entre uso de servicios de plataformas interna y nivel de DevOps (tomado de [21])

La segunda área en la que centrarse para mejorar el nivel de implantación de DevOps es en la mejora y aumento de la agilidad en la gestión del cambio[21], que puede ser la única cuando no es viable la plataforma interna. La gestión del cambio supone un cuello de botella común en el ritmo de entrega del software y puede afirmarse que su efectividad es del orden de tres veces superior en las organizaciones con mayor implantación de DevOps que en las de un grado bajo. Para mejorar esta efectividad las empresas enfatizan el empleo de la automatización en los procesos de pruebas y despliegue del software, la mitigación automática de riesgos[22], el uso de procesos de aprobación y validación del software menos rígidos y menos manuales (como ejemplo, en la Figura 2-14 puede observarse que en organizaciones con mayor uso de los procesos de aprobación rígidos y ortodoxos el nivel de ineficiencia es también muy alto), el aumento de la participación e influencia de los empleados en la gestión del cambio y, obviamente, la adopción de procesos acordes con la cultura DevOps. Los principales retos a afrontar para automatizar el proceso de gestión del cambio son la falta de una adecuada selección de pruebas a

[20] Estas plataformas internas a la organización proporcionan la infraestructura interna y los servicios de nube, los entornos, los flujos continuos de integración y de despliegue de software y, en general, todos los servicios que necesitan los equipos de desarrollo para crear, construir y ejecutar las aplicaciones.

[21] La gestión del cambio es responsable de controlar el ciclo de vida de los cambios introducidos en una organización sin que los servicios TI ofrecidos se vean afectados [30]. En los informes sobre DevOps citados está relacionada con los procesos de validación, aprobación y autorización de cambios en el software, realización automática de pruebas y entregas, y técnicas avanzadas de reducción de riesgos.

[22] La automatización para la reducción de riesgos incluye diversas técnicas, como el empleo de clusters activo-activo, despliegues limitados o «canary release» en que el nuevo código convive con el viejo y solo se pone a disposición de unos pocos usuarios hasta que se comprueba que funciona sin problemas, despliegues azul/verde, en los que los entornos o servidores de preproducción y de producción se van alternando, y otros, que en general buscan compartimentar los riesgos y posibilitar los cambios mientras se mantienen los servicios funcionando.

realizar sobre el software que cubran todos los posibles escenarios, la falta de mentalización en la organización para cambiar y el fuerte acoplamiento existente en la arquitectura de las aplicaciones.

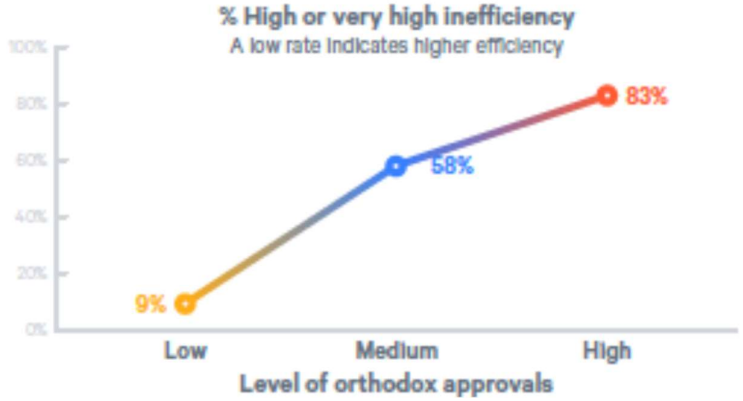

Figura 2-14 Relación entre nivel de aprobación ortodoxa e ineficiencia en la gestión del cambio (tomado de [21])

La adopción de DevOps tiene un impacto positivo en cuatro métricas[23] que indican el buen desempeño de las organizaciones en lo relativo al despliegue y operación de aplicaciones: el tiempo necesario para implantar cambios, que se reduce, la frecuencia con la que se es capaz de desplegar nuevo software, que aumenta, el tiempo necesario para restaurar servicios, que disminuye, y la ratio de fallos en los cambios, que disminuye.

Aspect of Software Delivery Performance*	Elite	High	Medium	Low
Deployment frequency For the primary application or service you work on, how often does your organization deploy code to production or release it to end users?	On-demand (multiple deploys per day)	Between once per day and once per week	Between once per week and once per month	Between once per month and once every six months
Lead time for changes For the primary application or service you work on, what is your lead time for changes (i.e., how long does it take to go from code committed to code successfully running in production)?	Less than one day	Between one day and one week	Between one week and one month	Between one month and six months
Time to restore service For the primary application or service you work on, how long does it generally take to restore service when a service incident or a defect that impacts users occurs (e.g., unplanned outage or service impairment)?	Less than one hour	Less than one day[a]	Less than one day[a]	Between one week and one month
Change failure rate For the primary application or service you work on, what percentage of changes to production or released to users result in degraded service (e.g., lead to service impairment or service outage) and subsequently require remediation (e.g., require a hotfix, rollback, fix forward, patch)?	0-15%[b,c]	0-15%[b,d]	0-15%[c,d]	46-60%

Figura 2-15 Métricas obtenidas en función del nivel de implantación de DevOps (tomado de [22])

También es muy interesante observar cómo las organizaciones y equipos obtienen y extienden los conocimientos DevOps y Agile en función del grado de implantación de DevOps. Según puede verse en

[23] El informe de Google Cloud denomina al conjunto de estas métricas como «software delivery and operational (SDO) performance».

la Figura 2-16 el modelo[24] que puede considerarse de mayor éxito es el de Comunidades de Práctica («Communities of Practice»), seguido del De Abajo hacia Arriba («Bottom-up or Grassroots») y el Mixto («Mashup»). Esto lleva a pensar que en estas organizaciones donde DevOps está más consolidado es muy importante, por un lado, el valor dado a la colaboración, el espíritu de equipo y la puesta en común de conocimientos que suponen las Comunidades de Práctica, y, por otro, el dado a la iniciativa y empuje individual que supone el modelo De Abajo hacia Arriba.

	Low	Medium	High	Elite
Training Center	27%	21%	18%	14%
Center of Excellence	34%	34%	20%	24%
Proof of Concept but Stall	41%	32%	20%	16%
Proof of Concept as a Template	16%	29%	29%	30%
Proof of Concept as a Seed	21%	24%	29%	30%
Communities of Practice	24%	51%	47%	57%
Big Bang	19%	19%	11%	9%
Bottom-up or Grassroots	29%	39%	46%	46%
Mashup	46%	42%	34%	38%

Figura 2-16 Formas de propagación de DevOps y Agile en función del grado de implantación de DevOps (tomado de [22])

Otra de las metas de DevOps es incrementar la productividad de sus equipos haciendo que estos sean capaces de completar trabajos que puedan resultar complejos y muy demandantes en un tiempo breve y con el menor número posible de distracciones, interrupciones o actividades superfluas. La productividad se mide en esto y no, conforme a la visión tradicional, en el número de líneas de código escritas o en el bajo número de errores producidos. De acuerdo con la encuesta [22], esto puede lograrse mediante la adecuada selección de herramientas que ayuden a automatizar todas las tareas que, aunque necesarias, son tediosas e introducen retardos y complicaciones, requiriendo esfuerzos que realmente no aportan valor. Además, la existencia de estas herramientas refuerza la confianza de los equipos que de esta forma pueden aumentar su iniciativa y tomar riesgos calculados donde realmente son importantes, en la creación de software, aumentándose también de esta forma la productividad. Estas herramientas

[24] Los modelos considerados son: Centro de Entrenamiento («Training Center»), un lugar fuera del centro habitual de trabajo dedicado específicamente a la enseñanza de nuevas herramientas, tecnologías, prácticas, etc, para educar a los trabajadores, esperando que a su vuelta implementen y propaguen lo aprendido; Centro de Excelencia («Center of Excellence»), un lugar donde concentrar a los expertos y desde el que poder consultarlos; Prueba de Concepto con Parada («Proof of Concept but Stall»), proyecto donde un equipo recibe total libertad para crear incluso rompiendo las normas de la organización pero deteniéndose y no yendo más allá al finalizar el proyecto; Prueba de Concepto como Plantilla («Proof of Concept as a Template»), igual al anterior pero sirviendo el equipo como modelo para replicar su actuación en otros equipos; Prueba de Concepto como Semilla («Proof of Concept as a Seed»), se crea un equipo para el proyecto de pruebas y luego se rompe, diseminando sus componentes entre otros equipos para difundir el conocimiento y prácticas aprendidas; Comunidades de Práctica («Communities of Practice»), donde se promueve que los grupos que comparten unos mismos intereses, en herramientas, lenguajes o metodologías compartan sus conocimientos y experiencias entre ellos, los equipos y en el conjunto de la organización; Big Bang, en los que la organización al completo se transforma de golpe para seguir las metodologías DevOps, como quiera que las definan, siguiendo directrices de arriba hacia abajo; De Abajo hacia Arriba («Bottom-up or Grassroots»), la transformación se inicia al nivel de pequeños equipos que poco a poco e informalmente van compartiendo y escalando sus éxitos sin el apoyo formal de la organización y sus recursos; y, finalmente, puede considerarse el Mixto («Mashup»), donde la organización implementa varios de los descritos anteriormente pero solo de manera parcial o con recursos insuficientes o priorizándolos.

tienen que resultar sencillas y estar orientadas a mejorar el flujo de entrega de software, cuyas etapas generales podemos ver en la Figura 2-17, a través de la integración continua (CI)[25], la entrega o despliegue continuo (CD)[26] y la automatización de pruebas. Tanto CI/CD como la automatización de pruebas tienen por objetivo reducir los tiempos de desarrollo y el tiempo necesario para que los requisitos o funcionalidades requeridas por los usuarios estén a su disposición, lo que está plenamente conforme con los preceptos de DevOps.

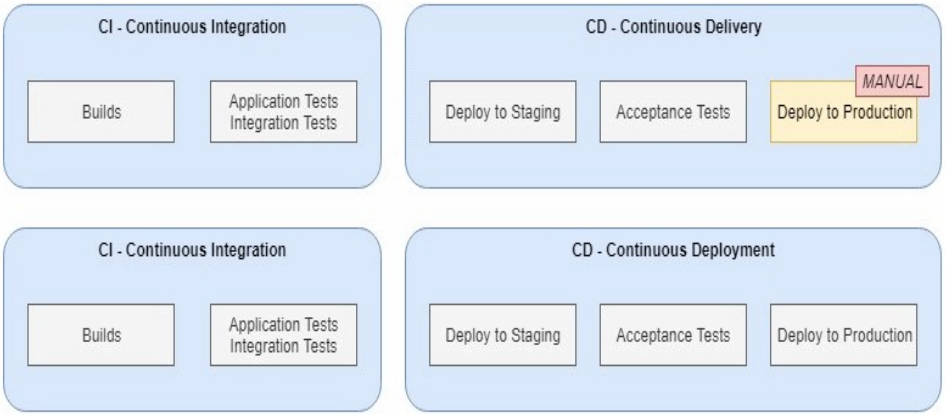

Figura 2-17 Etapas en CI/CD (tomado de [23])

A la hora de escoger estas herramientas de ayuda a la automatización se huye de soluciones particulares ya que estas, aunque en general resulten valiosas, suponen un alto coste de mantenimiento, introduciendo una mayor complejidad en la organización y añadiendo un gran número de nuevas tareas a los equipos, lo que les resta efectividad en la producción de software para los clientes. Conforme a los datos de la Figura 2-18, tomados de la encuesta de Google de 2019 [22], el software propietario para estas tareas está más concentrado en las empresas con los niveles más bajos de implantación de DevOps mientras que resulta menos usado en las de niveles altos; los productos comerciales «off-the-shelf» (COTS) son, más o menos, igual de usados, y las herramientas «open source» son más empleadas en las organizaciones de niveles altos. No obstante, la mezcla de todo tipo de herramientas es lo más habitual en cualquier nivel, quizás como resultado de que, por un lado, la implantación de DevOps es un proceso largo y, por otro, al mantenimiento de herramientas heredadas, a las que hay que sacar rendimiento por la inversión realizada o por los conocimientos adquiridos por el personal para usarlas.

[25] La integración continua («continuous integration» o CI) es un concepto de ingeniería de software que hace referencia a la integración automática de manera frecuente de trozos de código, compilado y probado, en la aplicación que se está desarrollando, manteniendo un adecuado control de versiones y, normalmente, en un contexto donde varios programadores pueden trabajar en el mismo proyecto. Esta integración frecuente, además, nos permite una detección temprana de errores y la vuelta atrás en caso necesario.

[26] La entrega o despliegue continuo («continuous delivery», «continuous deployment» o CD) es un concepto de ingeniería de software que hace referencia a la posibilidad de que cada cambio producido en la aplicación esté disponible lo antes posible para ser entregado automáticamente al usuario. Implica la integración continua (por lo que muchas veces se encuentra el acrónimo CI/CD), la automatización de las pruebas de validación y la automatización de la puesta en explotación. La diferencia esencial entre «continuous delivery» y «continuous deployment» es que en el primero el paso a explotación requiere una aprobación manual mientras que en el segundo este paso se produce automáticamente de forma inmediata o programada siempre que no haya surgido ningún error en el proceso.

	Low	Medium	High	Elite
A mix of proprietary tools, open source, and commercial off-the-shelf (COTS) software	30%	34%	32%	33%
Mainly open source and COTS, heavily customized	17%	8%	7%	10%
Mainly open source and COTS, with little customization	14%	21%	18%	20%
Primarily COTS packaged software	8%	12%	8%	4%
Primarily developed in-house and proprietary to my organization	20%	6%	5%	6%
Primarily open source, heavily customized	6%	7%	5%	12%
Primarily open source, with little customization	5%	12%	24%	15%

Figura 2-18 Tipo de herramientas usadas en función del grado de implantación de DevOps (tomado de [22])

Estas dos últimas ideas reflejadas en los párrafos anteriores, la importancia de la formación del personal y de las herramientas tecnológicas que favorecen la automatización y la creación de un ambiente positivo y de seguridad, ya quedaban reflejadas en el trabajo de 2013 de Vanson Bourne citado anteriormente [18]. En él se señalaba que los dos aspectos donde mayoritariamente se pensaba que las organizaciones tendrían que invertir para implantar DevOps de forma exitosa, según se ve en la Figura 2-19, eran precisamente estos dos.

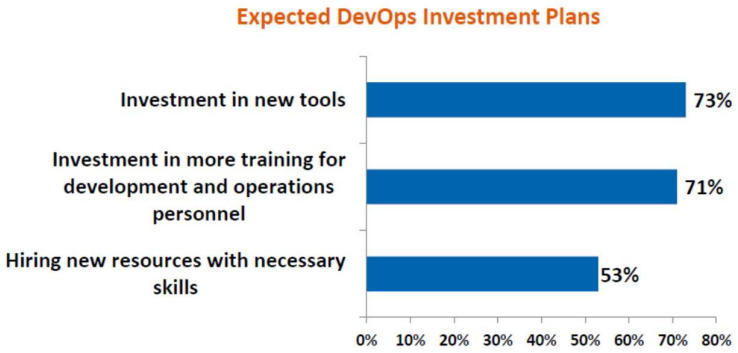

Figura 2-19 Expectativas de inversión para implementar DevOps (tomado de [18])

La experiencia aportada por los informes realizados en la segunda década del siglo XXI, a lo largo de los años en que DevOps se ha ido aposentando en el mundo de la producción de software, lleva a destacar una serie de principios que parecen haberse mantenido de forma constante, como podemos ver en [24]:

- Las organizaciones con un nivel alto de implantación de DevOps aúnan una elevada velocidad en la producción de software con una mayor estabilidad de sus sistemas. Esto es debido principalmente a la automatización, el trabajo colaborativo y menos burocrático, y el

tratamiento de la infraestructura como código (IaC)[27], diseñándola como parte del proceso de desarrollo. Con DevOps la velocidad de desarrollo no está reñida con la estabilidad para operaciones, todo lo contrario. Además, a la velocidad y estabilidad se une otro parámetro de suma importancia, la satisfacción por parte de los usuarios, que también mejora (ver Tabla 2-2).

- La implantación de DevOps es un proceso largo que necesita tiempo para establecer nuevas tecnologías, procesos, mentalidades y organización, y en el que se requiere realimentación, aprendizaje y mejora efectiva para avanzar.
- Las empresas que adoptaron DevOps en época temprana, como Google, Netflix, Amazon y otras, son las que muestran más dinamismo, más competitividad y están resultando líderes en sus ámbitos.
- La colaboración y el compartir experiencias y conocimiento entre diferentes equipos es un elemento esencial en el éxito de la implantación de la cultura DevOps.
- El uso de herramientas que posibiliten la automatización de los procesos de desarrollo y entrega es un valor crítico para DevOps. Cuanto mejor y mayor sea el nivel de automatización conseguido por una organización, esta podrá entregar a los usuarios productos de más calidad y de forma más ágil. La automatización refuerza la idea de que es posible optimizar la velocidad de producción sin sacrificar la estabilidad de los sistemas.
- Es frecuente la falta de apoyo a niveles directivos altos. Estos se encuentran lejos de comprender los problemas de los equipos técnicos, de desarrollo u operaciones, como, por ejemplo, la necesidad de contar con herramientas que les liberen de tareas manuales repetitivas, de una formación adecuada o de unas arquitecturas y productos tecnológicos avanzados.

Compañía	Frecuencia Despliegue	Tiempo Despliegue	Fiabilidad	Respuesta Usuarios
Amazon	23.000/día	Minutos	Alta	Alta
Google	5.500/día	Minutos	Alta	Alta
Netflix	500/día	Minutos	Alta	Alta
Facebook	1/día	Horas	Alta	Alta
Twitter	3/semana	Horas	Alta	Alta
Empresa típica	1 cada 9 meses	Meses o trimestres	Baja/Media	Baja/Media

Tabla 2-2 Desempeño de grandes empresas tecnológicas con DevOps frente a empresas no DevOps (tomada de [25])

[27] En [49] puede leerse una buena definición de qué es la IaC. Esta técnica permite controlar de forma más eficiente y sencilla las configuraciones de las infraestructuras, la gestión de los servidores, contenedores y redes virtuales, en local o en la nube, así como sus cambios. La idea subyacente es que los desarrolladores puedan solicitar los contenedores, las máquinas o los recursos virtuales que requieran mediante la generación de archivos donde declaran sus necesidades de una forma parecida a la que utilizan para generar código. Estas solicitudes son tratadas y gestionadas de forma automatizada consiguiéndose una reducción en el riesgo de producción de errores, generalmente asociados a las configuraciones manuales de los recursos virtuales, así como un incremento de la velocidad en el desarrollo y puesta en producción de aplicaciones. Con el uso de IaC los equipos de desarrolladores deben preocuparse más de la configuración de la infraestructura y los de operaciones deben involucrarse de forma más temprana en el proceso de desarrollo, puesto que deberían asesorar a los anteriores en la creación de los archivos de configuración; en síntesis, la utilización de esta técnica obliga a mejorar la comunicación entre estos equipos, lo que se alinea con la cultura DevOps.

2.4.4 El ciclo continuo de desarrollo DevOps

Desde el punto de vista de DevOps puede considerarse que son una minoría las aplicaciones que son puestas en explotación y no requieren un trabajo constante y rápido de soporte y mejora durante toda la vida del software. Esto conforma la idea de ciclo de vida que se inicia con los primeros trabajos para crear la aplicación y finaliza con su retirada del servicio. Según los autores que se consulten pueden encontrarse ciclos con pequeñas variaciones y cambios en los nombres de las fases, pero, en general, se sigue el flujo y las fases representadas en la Figura 2-20. Este flujo adopta la forma del símbolo matemático de infinito que está muy asociado al precepto DevOps de continuidad. El trabajo no se acaba con la primera entrega del producto, sino que continuamente se supervisa su funcionamiento y se reinicia el ciclo para perfeccionarlo, pudiendo realizarse sucesivas entregas en poco tiempo, de una forma muy rápida y asegurando la calidad final. Excepto quizás en la primera iteración, este ciclo no se ejecuta paso a paso, sino que se realiza de forma continua, con trozos de código en diferentes fases y permitiendo que el software aporte valor a la vez que se modifica y evoluciona. Para ello es esencial que la aplicación esté dividida en pequeños fragmentos lo más desacoplados posible.

Las principales actividades que se realizan en cada una de las fases son [26]:

- Planeamiento: En esta fase se define el proyecto, debiendo contarse con una visión completa en que se especifique el motivo del mismo y el fin a alcanzar. Durante la planificación también se analizan cuáles son las necesidades requeridas del producto a desarrollar y su viabilidad. Es decir, el conjunto de funcionalidades que van a aportan valor y los criterios de acabado y aceptación finales. Asimismo, se perfila cómo van a realizarse los trabajos para desarrollar el producto buscado en las fases restantes. Con el uso de metodologías ágiles, en que las reuniones de planificación y revisión de reiteraciones abarcan todo el proceso, podríamos considerar que esta fase se prolonga y superpone junto a las demás de forma más o menos simplificada.
- Codificación: Se produce el análisis, diseño y la programación de los módulos que componen la aplicación, utilizando los lenguajes, tecnologías y arquitecturas seleccionadas.
- Construcción: En esta fase se produce la compilación e integración de los diferentes componentes desarrollados o modificados. La integración continua (CI) permite automatizar el proceso de revisión y envío de los nuevos productos desarrollados.
- Pruebas: Durante las dos fases anteriores se realizan las pruebas unitarias, funcionales y de integración correspondientes, siendo deseable que se hagan de forma automatizada e introduciendo los mínimos retardos posibles. No obstante, también resulta óptimo que exista un equipo de aseguramiento de la calidad, diferente del de desarrollo, que utilice sus propias herramientas para identificar y corregir errores en el nuevo código de manera continua y sin introducir demoras en el flujo. Igual que sucede con el planeamiento esta fase tiene límites difusos. Las mejores prácticas son las que permiten que cada actividad implique sus propias pruebas automatizadas, para comprobar que se cumplen los criterios de aceptación, seguridad y calidad, de forma que los errores son detectados y corregidos cuanto antes.
- Entrega: Es el momento en que, tradicionalmente, el equipo de desarrollo remitía el producto final al de operaciones para su puesta en explotación. En esta fase, el nuevo código se integra con el código previamente existente en un entorno lo más parecido al entorno final de producción y se llevan a cabo las pruebas funcionales, de validación y aceptación finales. Las nuevas funcionalidades o características, o las mejoras realizadas sobre las existentes, se remiten desde el entorno y la infraestructura de desarrollo, donde se construyen, hacia los servicios que las integran con el resto de la aplicación y aseguran su funcionamiento.
- Despliegue: Es la fase que siempre se ha considerado más crítica en el proceso de producción de una aplicación. El despliegue implica preparar y configurar tanto el entorno software como la infraestructura hardware necesaria para el funcionamiento real de la aplicación. El desarrollo, la integración continúa (CI) y las entregas o despliegues continuos (CD) y rápidos buscados por los equipos DevOps no deben interferir en el funcionamiento de la aplicación

y son posibles debido a la automatización y continuidad en los mecanismos de integración, pruebas, entregas, y configuración y gestión de la infraestructura.

- Operación: En esta fase las aplicaciones se encuentran en su entorno real de empleo, produciendo y entregando el valor buscado por los usuarios. Los equipos de operaciones se encargan de que presenten un funcionamiento sólido y estable, corrigiendo los comportamientos inadecuados o los errores que se produzcan durante la explotación debidos a la plataforma, la convivencia de diferentes sistemas de información y la acción de los usuarios. Aquí también es fundamental la automatización en la optimización de los escenarios, buscando que los recursos empleados se ajusten a las demandas y se recuperen ante fallos de forma dinámica y óptima.
- Monitorización: El equipo de operaciones normalmente se encargará de definir las características y los parámetros a vigilar, así como el rango y tipo de medidas a tomar, para poder controlar el estado de salud de las aplicaciones y la infraestructura. No solo se monitoriza el software desarrollado, el propio ciclo DevOps es supervisado para comprobar su funcionamiento, buscando perfeccionarlo de forma continua. Podemos afirmar que la mejora solo es posible sobre aquello que es medible por lo que resulta prioritario marcar cuáles serán los indicadores[28]. Esta última etapa del proceso DevOps es también una fase permanente, que se aplica a todo el ciclo completo y que no es exclusiva del periodo en que las aplicaciones se encuentran en operación. La medición y análisis de indicadores ha de ser continua en todos los ámbitos y niveles del ciclo DevOps buscando que todos los procesos se desarrollen de forma adecuada y se mejoren de forma continua, corrigiéndose cualquier desviación y descubriendo e implementando avances en el desempeño de cualquier actividad.

Figura 2-20 Ciclo de vida iterativo DevOps (tomada de [26])

2.4.5 Las herramientas DevOps

Ya he expresado que en la cultura DevOps el uso de herramientas se considera esencial. Estas van a focalizarse en conseguir la automatización de procesos y tareas, liberando a los equipos de realizar

[28] En el mundo de los procesos se utiliza el término indicador clave de rendimiento («key performance indicator» o KPI) para denominar a los datos utilizados para, en función del grado de cumplimiento o no de un objetivo o de realización de un producto, medir el nivel de rendimiento de un proceso y señalar si está bien diseñado y si las tareas que se realizan son las adecuadas y se ejecutan correctamente. En esencia estos indicadores miden de forma cuantitativa o cualitativa un determinado parámetro y comparan esa medición con el valor objetivo marcado para ese parámetro, expresando el resultado en tantos por ciento. Como ejemplo de estos indicadores podemos señalar el número de despliegues de software, el tiempo necesario para el despliegue, el número de fallos, la duración de una actividad, el número de consultas del cliente, la satisfacción del cliente, etc.

actividades repetitivas y demandantes en esfuerzo y tiempo, a la vez que disminuyen la ratio de errores y aumentan la confianza en el entorno de trabajo y, en definitiva, incrementan la eficacia y productividad.

Estas herramientas pueden ser propietarias, elaboradas por la misma organización que las usa, aunque ya hemos visto que esta no suele ser la mejor solución, consistir en productos comerciales («off-the-shelf» o COTS) e incluso ser de código abierto con un mayor o menor grado de soporte y personalización. Las herramientas disponibles en el mercado son muy variadas y pueden centrarse en una sola fase del ciclo de vida del software, aunque lo habitual es que su uso se alargue a lo largo de él abarcando múltiples fases.

Las estadísticas de uso y los estudios de tendencias que se pueden consultar son también muy numerosos. A modo de ejemplo he incluido en la Figura 2-21 el resultado de un informe de la empresa americana Axosoft[29] elaborado a partir de una encuesta realizada a unos 2.700 profesionales del sector del desarrollo de software. Las herramientas que aquí aparecen no son, ni mucho menos, todas las que existen[30], y en un mercado tan dinámico como el de la producción de software pueden aparecer nuevos programas, evolucionar o desaparecer los existentes e incluso crearse «suites» completas con el adjetivo de DevOps para cubrir todos los aspectos susceptibles de automatización. No obstante, sirve de orientación y ejemplo para determinar qué tipo de herramientas se emplean en las diferentes fases a lo largo del ciclo de vida del software.

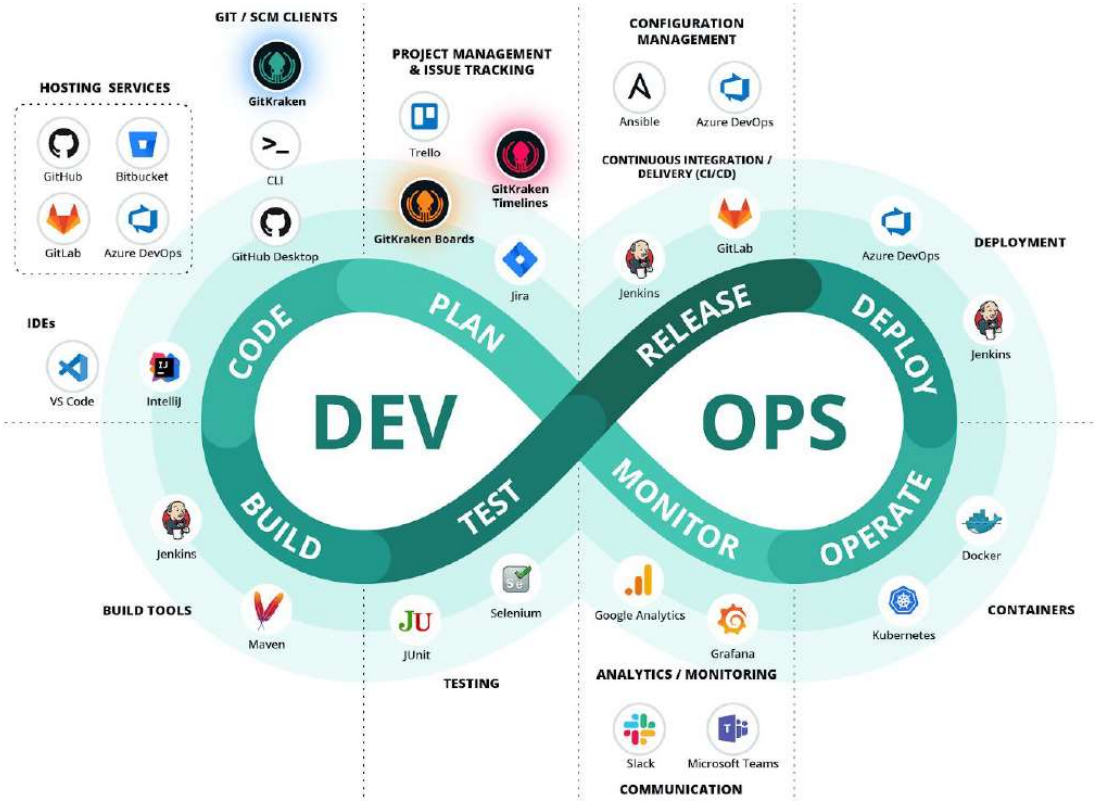

Figura 2-21 Herramientas en el ciclo DevOps (tomado de [27])

[29] Esta empresa, creada en 2000 en el estado de Arizona, es la creadora y mantenedora principalmente de dos productos: Axosoft, un software de gestión de proyectos «Agile»; y GitKraken, una suite para equipos de desarrolladores cuyo principal componente es un cliente «Git» (extraído de https://en.wikipedia.org/wiki/Axosoft el 13/12/2020).

[30] Por ejemplo, en el grupo de herramientas para gestión de la configuración, en la fase de «release» o entrega, aparecen solo las muy conocidas «Ansible», de Red Hat, y «Azure DevOps», de Microsoft, aunque existen otras muchas que también entrarían en este apartado y pueden resultar adecuadas en la gestión de configuración de la infraestructura y la IaC tales como «Puppet», «Chef», «Terraform» de HashiCorp, «AWS CloudFormation» de Amazon, o «Cloud Deployment Manager» de Google.

Para la fase de planeamiento se suelen utilizar aplicaciones dedicadas a la gestión de proyectos y al seguimiento de incidencias o desviaciones en su ejecución. Deben ser capaces de utilizarse con metodologías ágiles de desarrollo y realizar el seguimiento de las tareas e incidencias a través de todo el ciclo. Entre ellas se pueden destacar «Jira» y «Trello», ambas de la empresa Atlassian.

Durante la creación de código, además de la elección de un entorno de desarrollo integrado (IDE)[31] que permita utilizar los lenguajes, librerías y «frameworks» seleccionados, lo más importante es quizás el uso de una herramienta tipo «Git», como, «Bitbucket», «GitLab» o «GitHub» entre otras. Estas aplicaciones funcionan como un repositorio de código, con control de versiones y posibilidad de revertir los cambios, permitiendo la programación en paralelo y concurrente[32] de varios programadores sobre un mismo software. Algunas han escalado en las funcionalidades y capacidad de integración que ofrecen dentro de la cadena de desarrollo DevOps, como por ejemplo «GitLab», que se ofrece como una plataforma completa DevOps para la automatización y gestión de todas las fases del flujo de desarrollo, desde el planeamiento, gestión del software, realización de test, empaquetamiento del software, securización, CI/CD, hasta la configuración y supervisión de la infraestructura, y la producción de análisis e informes durante todo el proceso[33].

Entre las herramientas de construcción de código la más destacable es «Jenkins». Esta herramienta permite establecer procedimientos automatizados a lo largo del ciclo DevOps incluyendo la compilación, realización de pruebas y el despliegue. Puede, por ejemplo, detectar cambios en el código de los repositorios «Git» y automáticamente utilizar otras herramientas[34] para compilar, realizar pruebas y gestionar contenedores tipo «Docker» para entregas en producción, realizando vueltas atrás en caso de detección de fallos.

Para la corrección de errores existen también múltiples aplicaciones, orientadas a los diversos lenguajes de programación y entornos. En general estas herramientas ayudan a los equipos de pruebas y aseguramiento de la calidad a diseñar y generar los casos de prueba y a automatizar la realización de estos test sobre el código, reduciendo de esta forma el tradicional cuello de botella que supone la actuación de estos equipos. Entre estas herramientas podemos citar a «JUnit» y «Jest», para Java y JavaScript respectivamente, «Selenium», para aplicaciones basadas en la web, y «PHPUnit», para aplicaciones basadas en «PHP».

Para las fases de entrega y despliegue, además de las comentadas «Jenkins» y «GitLab» que implementan CI/CD, podemos señalar a «Ansible»[35] entre las herramientas de automatización de la gestión de la configuración para infraestructura, redes, contenedores, seguridad y otros recursos de plataforma, habiéndose convertido en un producto líder en este sector. «Travis CI» es otra herramienta destacable de CI/CD y de ejecución de test que presenta una buena integración con «GitHub» y «Bitbucket».

---

[31] Se entiende por entorno de desarrollo integrado («integrated development environment» o IDE) el software que, a través de una única interfaz gráfica de usuario, proporciona al programador la ayuda y todas las herramientas necesarias para escribir código y crear la estructura de ficheros que conformarán la aplicación desarrollada. Existe un gran número de IDE disponibles, optimizados para diferentes lenguajes (como Java, C, C++, C#...) aunque la mayoría permiten el uso de varios, como son «VS Code», «Microsoft Visual Studio», «NetBeans», «Eclipse», «IntelliJ» etc.

[32] «El desarrollo concurrente o en paralelo provee de los mecanismos necesarios para aislar una versión de un desarrollo software con el objeto de realizar modificaciones a esta sin alterar otras versiones que puedan coexistir en ese momento, puesto que cubre la necesidad de realizar cambios simultáneos con distintos objetivos en una misma aplicación» (https://devopsti.wordpress.com/2014/09/10/que-es-el-desarrollo-en-paralelo-o-concurrente/, visitado el 31/12/2020).

[33] En la página web https://about.gitlab.com/devops-tools/ (consultada el 13/12/2020) puede verse una comparación de multitud de herramientas con GitLab en las diversas fases del ciclo de vida de software de DevOps.

[34] La integración entre herramientas es común, facilitando y simplificando su uso por parte del usuario. «Jenkins» por ejemplo puede utilizar «Maven» para construir aplicaciones basadas en Java utilizando las posibilidades de esta última de compilación, testeo (integrándose a su vez con «Junit» para ejecutar test automáticos) y empaquetamiento en ficheros con extensiones jar o war.

[35] En la referencia [50] podemos encontrar una guía al uso de «Ansible» y sus posibilidades de automatización en un ambiente DevOps.

En el entorno de producción las tecnologías de virtualización basadas en contenedores «Docker» y la de orquestación de contenedores «Kubernetes»[36] se han convertido casi en unos estándares. Los contenedores nos permiten desplegar aplicaciones con cada uno de los módulos o servicios que la componen alojados en su propio contenedor, de tal forma que en ellos tienen todo lo necesario para ejecutarse, entorno de ejecución, librerías, sistema operativo reducido y dependencias de cualquier tipo. De esta forma se proporciona un nivel de abstracción superior al de la virtualización clásica y un aprovechamiento de los recursos maximizado. La orquestación de contenedores permite su gestión automatizada de forma eficiente en entornos distribuidos, creando, eliminando y escalado los contenedores de acuerdo con las necesidades.

En el área de monitorización y análisis[37] encontramos herramientas como «Google Analytics», orientadas a proporcionar información sobre el uso de aplicaciones web; «Grafana», que provee datos para monitoreo y análisis con un uso extensivo de todo tipo de gráficos y permite establecer alertas; y «Slack», específicamente dedicada a agilizar y mejorar la comunicación no solo entre los equipos sino también entre todo el personal que interviene en el proceso, y que se integra muy bien con otras herramientas como «Jira» o «Trello», para el seguimiento de fallos y tareas, así como con «GitHub», «GitLab» y «Bitbucket».

Por último, es de destacar que, además del conjunto de diversas herramientas aquí expuestas, la mayoría de las grandes empresas dedicadas a ofrecer servicios en la nube, como Microsoft con «Azure», Amazon con «AWS», IBM con «IBM Cloud», Google con «Google Cloud» o SAP con «SAP Cloud Platform», cuentan con un conjunto de servicios específicos que posibilitan o favorecen la aplicación de la cultura DevOps y tratan de abarcar el ciclo completo de desarrollo y puesta en explotación conforme a las premisas de esta filosofía, implicándose en general de forma muy significativa en la expansión del conocimiento y uso de DevOps.

2.4.6 Algunas malinterpretaciones con DevOps

Considerar a DevOps solamente como un nuevo paradigma en la producción de software o como una filosofía que propugna la mera inversión económica en una serie de herramientas tecnológicas para ayuda al desarrollo y a la gestión de la infraestructura TI es una aproximación demasiado simplista e incluso errónea. Ya se he señalado que hay que verlo más como una cultura, como una forma de concebir y llevar a la práctica el desarrollo de software en que resulta de mayor importancia las relaciones entre equipos, la colaboración e iniciativa, la creación de valor y la búsqueda de agilidad y rapidez, a la vez que se mantienen estándares elevados de calidad, frente a otras consideraciones.

En los últimos años algunas empresas han creado equipos especializados DevOps e incluso puestos específicos denominados DevOps [24]. No obstante, hay que recordar que DevOps no se sustancia en la creación de nuevas estructuras a incluir en el organigrama de los departamentos de TI. La adopción de esta cultura lleva a formar equipos únicos de trabajo o que colaboran de forma estrecha, equipos orientados a la misión de producir y mantener un determinado software compuestos por ingenieros de software y programadores, administradores de sistema, probadores, técnicos de ciberseguridad y aseguradores de la calidad.

[36] «Docker» también dispone de su propia herramienta para orquestación de contenedores, denominada «Docker Swarm», con la que gestionar clústeres de nodos de Docker en una arquitectura maestro-esclavo similar a la de Kubernetes, y de una herramienta llamada «Docker Compose» que permite definir aplicaciones multicontenedor concebidas como grupos de servicios o microservicios interconectados que comparten dependencias de software y se escalan y orquestan de forma conjunta [52].

[37] Esta área podría denominarse de forma más sintética «realimentación» e incluiría el establecimiento de elementos y las medidas a tomar sobre ellos durante el proceso con el doble objetivo de detectar errores o malos funcionamientos, pudiendo establecerse alertas, y de poder realizar análisis para incluir mejoras.

¿Qué son entonces los denominados «ingenieros DevOps»?[38] Estos puestos, en general, suelen corresponden a ingenieros dedicados a la arquitectura de sistemas, tanto de redes como de servidores e infraestructuras TI en general, con un enfoque DevOps, haciendo hincapié en la formación en tecnologías de aplicación en la nube, a la gestión de recursos virtuales y contenedores, a la experiencia en el uso y gestión de IaC, a la comprensión y conocimientos de la arquitectura de microservicios y a la experiencia en general de todo tipo de herramientas que puedan emplearse en el ciclo DevOps. Realmente las organizaciones solo necesitan a expertos que realicen una labor de consultaría para estudiar, en función de sus particularidades y objetivos, cuáles serían los mejores procedimientos a utilizar y las tecnologías a adquirir, así como cuáles serían los cambios a introducir, tanto organizativos como procedimentales, para adoptar con éxito una cultura DevOps orientada, en último extremo, a mejorar la calidad en la producción y provisión de software.

Otra idea equivocada es que adquirir un conjunto suficiente de aplicaciones tecnológicas avanzadas, que nos permitan construir flujos de CI/CD y automatizar la mayoría de tareas del ciclo de vida del software, es suficiente para conseguir todas las ventajas de la cultura DevOps. Si recordamos la «ley de Conway», que venía a decir algo similar a que el software tiende a desarrollarse de forma que refleja las estructuras y relaciones existentes en los equipos que lo desarrollan, podemos afirmar, como corolario, que los equipos acaban empleando las herramientas disponibles de forma que implementen los patrones de organización y comunicación ya existentes y no modificándolos o mejorándolos como podría pensarse (esta idea la podemos encontrar en el apartado «Why Tools Don't Matter», pág. 113 de [28]). Por tanto, sigue siendo fundamental que DevOps se establezca como una cultura efectiva, en la mentalidad y estructura de la organización, y no como un mero modernismo o un conjunto de potentes herramientas tecnológicas a adquirir.

2.4.7 Algunos problemas de DevOps

Primero veamos algunos problemas que pueden surgir durante la adopción. Como podemos observar en la Figura 2-22, extraída de la encuesta de 2013 de Vanson Bourne [18], las principales dificultades giran en torno a las personas y los procesos internos. Puede destacarse la falta de unos roles y responsabilidades claramente definidos entre el personal de desarrollo y operaciones que favorezcan la interacción entre ellos y la creación de equipos orientados a conseguir un ciclo de vida de las aplicaciones ágil y seguro. Hay una carencia de formación, tanto técnica como de mentalidad, que ayude al desarrollo y entrega ágil de software. En los puestos directivos no se aprecian las ventajas y beneficios que puedan obtenerse al adoptar DevOps, ya que no se tiene una visión clara de qué es ni se realizan análisis de casos de estudio o investigaciones de prospectiva que orienten en este sentido. La falta del impulso desde arriba, fundamental para el necesario esfuerzo inversor y de cambio organizativo, dificulta su implementación. Y, por último, puede señalarse que el diseño de la propia organización suele resultar complejo y es difícil de cambiar, en caso de que se considere necesario, condicionando asimismo el diseño de los procesos que en muchas ocasiones no se encuentran suficientemente analizados ni documentados.

También es difícil realizar el esfuerzo inversor necesario para dotarse del conjunto de herramientas adecuado para obtener una cadena altamente automatizada de creación y entrega de software. Como ya he comentado anteriormente la inversión en automatización es una prioridad en la implantación de

[38] Existen en el mercado muchas empresas que ofrecen certificaciones de «ingenieros DevOps» como, por ejemplo, el certificado de «ingeniero de herramientas de DevOps» ofrecido por el Linux Professional Institute (https://www.lpi.org/es/our-certifications/devops-overview, visitado el 20/12/2020). Las grandes empresas tecnológicas ofrecen asimismo certificaciones con el apellido DevOps orientadas hacia el conjunto de aplicaciones y servicios específicos que ofrecen y pueden ser utilizados en este ámbito; entre ellas pueden señalarse el «Microsoft Certified: DevOps Engineer Expert» de Microsoft (https://docs.microsoft.com/en-us/learn/certifications/devops-engineer, visitado el 20/12/2020), el «Professional Cloud DevOps Engineer» de Google (https://cloud.google.com/certification/cloud-devops-engineer, visitado el 20/12/2020), el «AWS Certified DevOps Engineer – Professional» de Amazon (https://aws.amazon.com/es/certification/certified-devops-engineer-professional/, visitado el 20/12/2020) o el «IBM Certified Solution Advisor - DevOps V1» de IBM (https://www.ibm.com/certify/cert?id=50001301, visitado el 20/12/2020).

DevOps, aunque no lo es todo en esta cultura. Estas herramientas liberan al personal de desarrollo y operaciones de tareas manuales reiterativas y que pueden ser una fuente de fallos evitables.

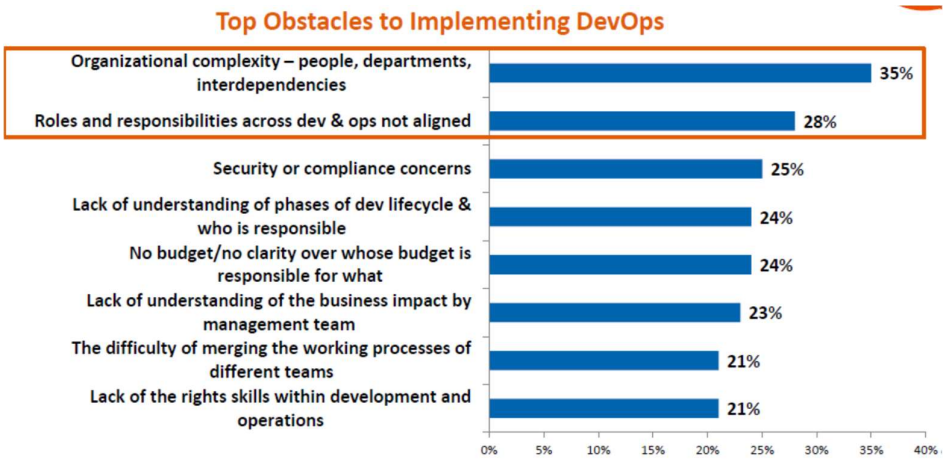

Figura 2-22 Principales problemas para adoptar DevOps (tomado de [18])

En segundo lugar, señalaré algunos de los problemas que pueden presentarse al adoptar DevOps. Su implantación no nos asegura que podamos obtener todas las ventajas productivas que se le asocian. Con esta cultura también podemos encontrarnos con una serie de problemas a los que enfrentarnos y que deberemos gestionar para que no resulten perjudiciales. Entre ellos podemos destacar los siguientes:

- Va en contra de la cultura de «proyecto». Todo proyecto por definición tiene un principio y un fin. Se ajusta a un presupuesto y a unos límites temporales. Tanto su planificación como su control se orientan a no traspasar esas fronteras. Para aquellos productos en los que el cliente pone énfasis en que solo se necesita un producto concreto con unas funcionalidades que no van a evolucionar y en donde no se desea invertir en su mantenimiento, DevOps ve muy limitadas sus posibilidades de poner en juego sus características de agilidad y mejora continua.
- Arrastra los problemas inherentes al desarrollo ágil como es la disolución de responsabilidades en el equipo y una posible falta de control. Promover el trabajo colaborativo y en equipo, y la iniciativa dentro de este, no significa que se encuentren soluciones innovadoras o que se resuelvan siempre los problemas que puedan encontrarse. Por el contrario, podemos encontrarnos con un panorama de exceso de democracia, donde las tareas se dilatan o no se resuelven, por lo que las figuras de los líderes o coordinadores siguen siendo esenciales.
- Puede generarse un exceso de dependencia de las herramientas tecnológicas. El incremento de la confianza y el ambiente de seguridad generado por el uso de las herramientas tecnológicas que, en general, abstraen y simplifican una gran variedad de tareas que antes se realizaban de forma manual puede llevar a perder el conocimiento y experiencia de cómo realizar esas tareas. Esto puede conducir a situaciones no deseadas en el caso de que las herramientas, por el motivo que sea, dejen de estar disponibles de forma puntual o prolongada.

2.5 La gestión de servicios TI y DevOps

Una sencilla definición de qué es la gestión de las tecnologías de la información[39] la encontramos en el portal web de la Agencia Española para la Calidad donde podemos leer que «gestionar las TI consiste en tomar decisiones operativas dentro del gobierno de las TI. La gestión de la TI se refiere a los aspectos operativos para el suministro de productos y servicios de TI en la forma más eficaz» [29]. Es decir, está muy relacionada con la dirección y gobernanza de las TI en su conjunto.

Un servicio puede ser definido como un conjunto de actividades desarrolladas con la finalidad de entregar valor a unos clientes, facilitándoles unos resultados en forma de productos, tangibles o no, que se obtienen sin la propiedad de los costes y riesgos asociados a su producción [30]. Específicamente, un servicio TI es el servicio proporcionado a los clientes por un proveedor de servicios TI, interno o externo a la organización, que se basa para ello en el uso de las tecnologías de la información, dando soporte, además, a los procesos de negocio del cliente [30]. Un servicio TI se compone de una combinación de personas, procesos y tecnología y debería estar definido en un acuerdo de nivel de servicio (SLA) [30]. Hoy en día toda organización dedicada al ámbito de las tecnologías de la información y las comunicaciones, orienta sus actividades para proporcionar servicios TI útiles y eficaces a sus usuarios o clientes, ya sean los internos, propios de su organización, o los externos, con los que la organización se relaciona y a los que ofrece servicios en su concepto más amplio.

Podemos concluir, por tanto, que el concepto de gestión de TI es más amplio y va mucho más allá que el mero desarrollo y gestión del ciclo de vida del software en el que se desenvuelve DevOps. Los servicios TI no son solo las aplicaciones que dan soporte a procesos específicos, sino que incluyen también, por ejemplo, el correo electrónico, el espacio de almacenamiento de ficheros en red, la telefonía, las videoconferencias, el establecimiento de redes de datos, los servicios de cifrado de documentación, las utilidades de ofimática y un largo inventario que depende de cada organización. No obstante, dentro del amplio catálogo que suponen los servicios TI nos encontramos con el software como elemento tecnológico básico, junto al hardware, para dar soporte y materializar estos servicios. La capacidad de una organización, a través de su departamento o componente TIC/CIS, para proveerse, operar y mantener el software específico que da soporte a sus procesos de negocio resulta hoy en día esencial para determinar el éxito o fracaso de la misma en la consecución de sus objetivos, así como en la apreciación por parte de los usuarios y clientes de su utilidad y superioridad frente a competidores. El MINISDEF y las FAS no escapan a esta visión moderna y están obligados a dotarse de todas las herramientas que permitan no solo su supervivencia y adaptación a un mundo tecnológicamente avanzado, sino también alcanzar la superioridad en la información que les asegure la consecución de sus objetivos frente a todo tipo de amenazas[40].

En la gestión de las TI podemos encontrar diversos marcos de trabajo estandarizados que proponen una serie de buenas prácticas o procesos a implantar. Entre ellos podemos señalar algunos como ISO/IEC 20000[41], COBIT[42] y eSCM_SP[43], pero quizás el más conocido y de mayor implantación es ITIL[44]. ITIL

[39] El término más preciso sería gestión de servicios de las tecnologías de la información («IT service management» o ITSM).

[40] El concepto de superioridad en la información, que permite a una fuerza militar tener un mejor conocimiento propio y de la situación y actividades del enemigo, acelerando el ciclo de toma de decisiones e incrementando el ritmo de ejecución de las operaciones, así como su relación con la modernización de las TI utilizadas por estas fuerzas, podemos encontrarlo en una entrevista al director de Estrategia de Servicios e Innovación de la Agencia de Información y Comunicaciones (NCI) de la OTAN en este artículo de 2017 [54].

[41] https://www.normas-iso.com/iso-20000/, visitado el 19/12/2020.

[42] https://es.wikipedia.org/wiki/Objetivos_de_control_para_la_informaci%C3%B3n_y_tecnolog%C3%ADas_relacionadas, visitado el 19/12/2020.

[43] https://en.wikipedia.org/wiki/ESourcing_Capability_Model, visitado el 19/12/2020.

[44] Biblioteca de infraestructura de tecnologías de la información («IT infraestructure library» o ITIL). Este ITSM, como expondré más adelante, es el citado dentro de la normativa de desarrollo del «Plan Estratégico de los CIS/TIC» del

es un conjunto de libros, creados inicialmente en la década de los 80 del siglo pasado por el gobierno británico, donde se desarrollan un conjunto muy amplio y complejo de conceptos y buenas prácticas, y cuyo alcance dentro de las actividades de la organización es muy grande. Inicialmente eran ocho libros: dos de ellos estaban dedicados a la gestión específica de TI –mejores prácticas para la provisión de servicio y mejores prácticas para el soporte de servicio–, cinco eran guías operativas sobre temas diversos, desde las infraestructuras TI a los activos software –gestión de la infraestructura de TI, gestión de la seguridad, perspectiva de negocio, gestión de aplicaciones y gestión de activos software–, y un libro final que contenía guías de ayuda de implementación, centrado principalmente en la gestión de servicios y que posteriormente se completó con otra guía para implementación en pequeños departamentos TI [31]. ITIL no especifica el diseño concreto del proceso a establecer, pero sí cuáles serían los procesos con los que habría que contar para conseguir una gestión adecuada de los servicios TI.

De forma parecida al ciclo de vida de DevOps que hemos visto, ITIL también desarrolla el concepto de ciclo de vida del servicio con unas fases similares [31]:

- Estrategia: Estudio de las necesidades de servicios y posibilidades de satisfacerlas. Que comprende los procesos de gestión financiera, del portfolio, de la demanda, de relaciones con el negocio, gobernanza y cloud computing.
- Diseño: Análisis de viabilidad de servicios. Con los procesos de gestión del catálogo de servicios, de niveles, de disponibilidad, de capacidad, de continuidad, de proveedores, de seguridad de la información y de coordinación del diseño.
- Transición: Realización de pruebas y evaluación del desempeño del servicio. Los procesos que lo componen son de gestión de la configuración y activos, del cambio, del conocimiento, planificación y apoyo a la transición, gestión de entrega y despliegue, de validación y pruebas y evaluación del cambio.
- Operación: Monitorización activa y pasiva del funcionamiento del servicio. Con los procesos de gestión de incidentes, de problemas, de solicitudes, de eventos y de accesos.
- Mejora continua: Medición del funcionamiento del servicio, resultados, problemas y soluciones adoptadas con la idea de realimentar y proporcionar información útil a los procesos necesarios para mejorar la prestación del servicio.

La última versión de ITIL, la 4, lanzada a principios de 2019, ha introducido una serie de cambios notables buscando abandonar las rigideces anteriores y ser más pragmática para facilitar su implantación. Entre ellos, como ejemplo, destaca la descripción de prácticas[45] en vez de procesos. No obstante, se ha buscado un cambio de actitud más que un cambio de contenidos, usando, por ejemplo, conceptos que resultan familiares a las prácticas en alza de desarrollo ágil y de DevOps, a la vez que se muestra como un marco en cuyo interior pueden convivir este tipo de nuevas tendencias [32]. En este sentido, se introducen una serie de siete principios rectores a seguir como recomendaciones por las organizaciones en cualquier circunstancia y con independencia de las diferencias en objetivos, estrategias, tipos de trabajo o estructura de gestión, y que, en mi opinión, se alinean bien con DevOps [33]:

- Enfocarse en producir valor, considerando no solo el beneficio para el cliente o usuario sino también para cualquiera que pueda verse afectado o intervenga en la producción como son los empleados, proveedores e incluso la sociedad en su conjunto.
- Comenzar desde la situación actual y no hacerlo desde cero rompiendo con lo conseguido hasta el momento.

MINISDEF como ejemplo de marco de trabajo para tener en cuenta en la definición de procesos en los ciclos de vida de los servicios.

[45] La práctica de gestión se define en ITIL 4 como un conjunto de recursos de la organización diseñados para la ejecución de un trabajo o para la consecución de un objetivo [34].

- Progresar en ciclos no excesivamente largos o ambiciosos, iterativamente y mediante realimentación.
- Colaborar y promover la visibilidad, buscando romper con los grupos o elementos aislados y favoreciendo la transparencia y la compartición de experiencias y conocimiento.
- Trabajar y pensar de forma holística, considerando que ningún servicio, práctica, proceso, departamento, o proveedor trabaja de forma aislada, sino que todo interactúa de una forma compleja para crear valor.
- Mantenerlo todo simple y práctico, eliminado lo superfluo que no aporta valor.
- Optimizar y automatizar, siempre en ese orden y buscando no solo la eficacia sino también la eficiencia en el uso de los recursos existentes.

Otro concepto importante en ITIL 4 es el de cadena de valor del servicio, que funciona como el ciclo de vida, describiendo seis actividades clave –planificar, interactuar, diseño y transición, obtener/crear, entregar y proporcionar soporte, y mejorar– interconectadas y que reciben aportaciones tanto del exterior como del interior de la cadena, para la producción de valor mediante la creación de un producto o servicio [34]. Estas actividades se alinean aún mejor con el ciclo DevOps que los ciclos de vida de los servicios de versiones anteriores como puede verse en la Figura 2-23.

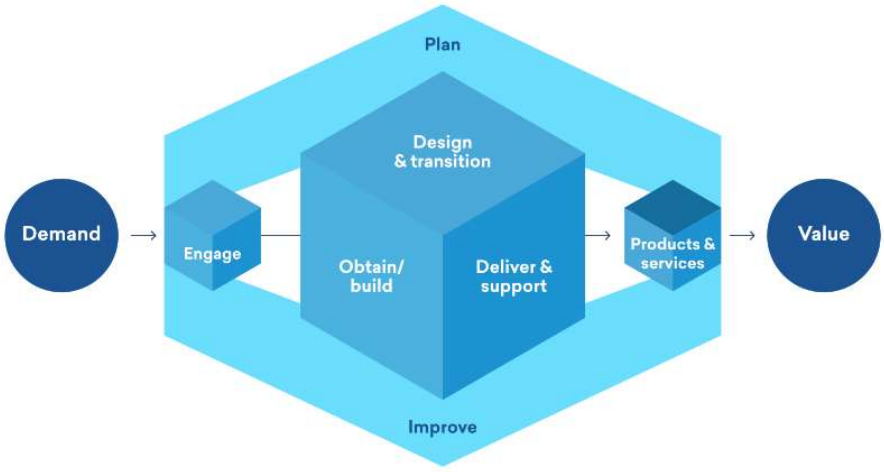

Figura 2-23 Cadena de valor de ITIL v4 (tomado de [34], a su vez tomado de «ITIL Foundation: ITIL 4 Edition (2019)», de Axelos)

En esencia, con la implantación de la gestión de servicios TI se busca establecer el orden y la productividad en un sector que tiende a ser complejo y caótico, sobre todo en las grandes organizaciones con fuertes componentes TI. La implementación y certificación de ITIL en una entidad es un proceso arduo que supone una gran cantidad de trabajo, de documentación y de esfuerzo. Estos conceptos de minuciosidad documental, control estricto, complejidad, poca flexibilidad y lentitud asociados a los procesos de ITIL hacen que, en principio, parezca que este marco de gestión y dirección de TI colisione con la cultura de DevOps.

En mi opinión esto no es así y la evolución de ITIL a su cuarta versión refuerza la idea de convivencia entre ITIL y DevOps. En primer lugar, hay que huir de la implantación rígida de ITIL, de tal forma que habría que evitar que los procesos de ITIL (o prácticas, como se denominan en su última versión) se convirtiesen, en sí mismos, en el objetivo de la organización, dedicando más esfuerzos a su implantación que a la provisión de servicios TI de calidad. Cada entidad debe estudiar, en función de sus actividades y posibilidades, los procesos necesarios y establecerlos con la profundidad adecuada buscando siempre

la mejora continua, pero sin que esto se convierta en la meta fundamental que debe ser siempre la provisión de servicios.

Por otro lado, hay que señalar que ITIL proporciona a las organizaciones una visión más completa del sector de las TI, permitiéndoles estructurarlo y gestionarlo en su conjunto de forma más eficiente. Posibilita que se entiendan mejor y de forma más exhaustiva no solo sus propias capacidades TI y cómo estas se relacionan con el resto de actividades de la entidad, sino también el mundo de sus clientes, de los proveedores y otros aspectos que afectan y orbitan alrededor de los servicios TI como pueden ser los asuntos financieros o legales. Además, facilita el establecimiento de mecanismos que aseguran no solo la mera prestación de los servicios sino también que esto se haga en un contexto adecuado y que se contemple su mejora.

ITIL puede existir en una organización sin implementar la cultura DevOps. Pero también DevOps puede adoptarse en una organización dedicada a la producción de software sin tener por qué considerar a ITIL. Pero en este último caso se perdería la visión holística y las posibilidades de dirección y gobernanza que sobre el conjunto de los servicios TI nos ofrece ITIL. Por tanto, bajo mi punto de vista, puede concluirse que ITIL y DevOps no solo pueden convivir, sino que resulta muy ventajoso que lo hagan, sobre todo en grandes organizaciones, como puede ser el MINISDEF, donde un amplio conjunto de servicios TI son necesarios, incluyendo entre ellos a la producción y mantenimiento de aplicaciones específicas. Tan solo hay que considerar que las altas frecuencias de despliegue asociadas a DevOps no deben ser entorpecidas, por lo que los procesos de ITIL relacionados con el cambio, la configuración y la entrega de productos deberían automatizarse y agilizarse al máximo.

3 DESARROLLO DEL TFM

3.1 La necesidad de digitalización en el MINISDEF

Un ministerio, en su desempeño, no deja de actuar como cualquier otro organismo, configurándose como un sistema. En consecuencia, se puede afirmar que, en términos generales, reciben unas entradas que son tratadas a través de procesos ejecutados por diversos subsistemas para obtener las salidas deseadas, todo ello guiado por unos objetivos concretos. En la Figura 3-1 puede verse un ejemplo con un esquema de una organización como sistema. En este mundo sistémico podemos afirmar que, hoy en día, el subsistema de información, cuyo esquema general puede verse en la Figura 3-2, es el núcleo que nos va a permitir dirigir, coordinar y analizar al conjunto de la organización y su funcionamiento, y que de su actuación va a depender, en muchos casos, el éxito o fracaso de la organización.

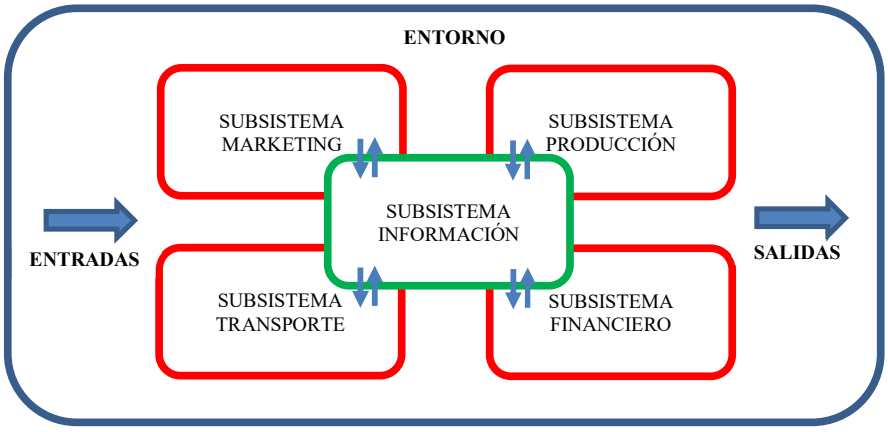

Figura 3-1 Ejemplo de organización como sistema (elaboración propia)

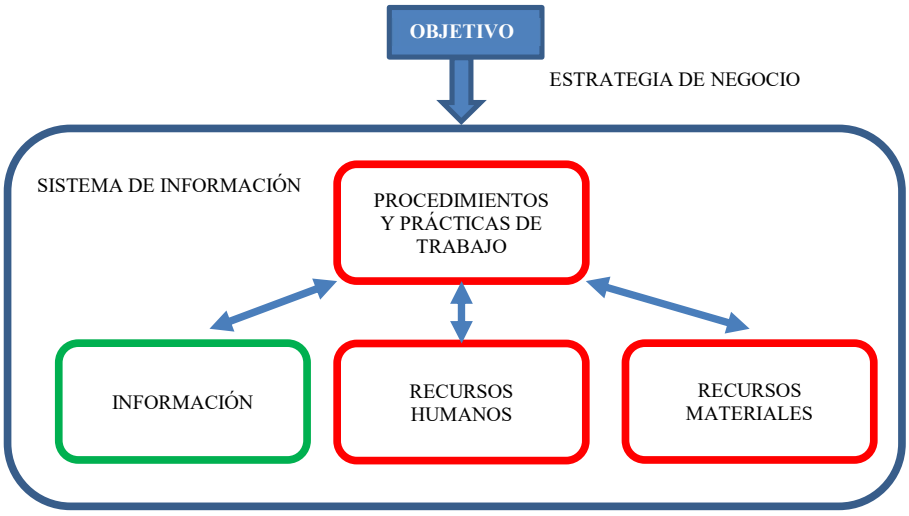

Figura 3-2 Elementos de un sistema de información (elaboración propia)

El MINISDEF, y en particular las FAS[46], como toda la Administración General del Estado (AGE)[47], han de ofrecer el mejor servicio posible a la sociedad española. Por eso deben alcanzar los objetivos que se les marquen diseñando unos procesos flexibles y modernos que puedan ejecutarse de la forma menos costosa sin perder por ello nada de eficacia. Asimismo, no deben permanecer aislados del resto de la administración y, salvaguardando las necesidades específicas derivadas de la Defensa y de las organizaciones internacionales de Defensa en que se encuadran, sus sistemas de información tienen que buscar el máximo de interoperabilidad tanto en los procesos exclusivamente internos, como en los que relacionan el Estado Mayor de la Defensa (EMAD), los diferentes Ejércitos y el Órgano Central. Esta interoperabilidad entre procesos también debe estar presente en aquellos que marcan sus relaciones con otras administraciones y organizaciones, ya sean a nivel nacional o internacional.

Para ello el MINISDEF y las FAS tienen que avanzar en su digitalización, tanto en el ámbito operativo como en el funcional o administrativo. Hay que procurar que todos los procesos y actividades, tanto internos, propios de un órgano, como externos, que relacionen varios órganos o a estos con entes externos a la administración, como ciudadanos o empresas, estén digitalizados y soportados por software. La transformación digital a acometer implica un cambio cultural y en el funcionamiento de las organizaciones basado en la gestión por procesos y en la orientación al dato[48], añadiendo como tercer elemento un importante soporte de servicios TI.

En este sentido, con la «Agenda Digital»[49], aprobada por el Gobierno español en 2013, podemos afirmar que se inicia el proceso para lograr la transformación y modernización de la administración mediante un uso eficaz e intensivo de las TI. El «Plan de Transformación Digital de la Administración General del Estado y sus Organismos Públicos» [35] de 2015, estrategia TIC de la AGE para el periodo 2015-2020, es la referencia fundamental para modernizar la administración transformándola digitalmente. Este Plan establece cinco grandes objetivos estratégicos como puede verse en la Figura 3-3, cada uno con sus líneas de acción asociadas, objetivos que constituyen el marco inicial para definir los diferentes «Planes de Acción Departamentales» de cada ministerio. Planes que, en consecuencia, deben converger con los principios, objetivos y líneas de acción del «Plan de Transformación Digital». El MINISDEF, como veremos más adelante, ya ha desarrollado su «Plan de Acción de Transformación Digital» que se encuentra alineado con la estrategia TIC de la AGE y con la propia política y estrategia CIS/TIC del Ministerio.

[46] Las FAS están compuestas por el Estado Mayor de la Defensa (EMAD) y los Ejércitos: el Ejército de Tierra (ET), la Armada (AR) y el Ejército del Aire (EA).

[47] La necesidad de modernización de la AGE a través de la transformación digital queda claramente establecida en la Ley 39/2015, de 1 de octubre, del «Procedimiento Administrativo Común de las Administraciones Públicas» y la Ley 40/2015, de 1 de octubre, de «Régimen Jurídico del Sector Público».

[48] La orientación al dato («data centric») es el reconocimiento por parte de las organizaciones de que lo más importante en sus sistemas de información reside en los datos manejados y no en otros elementos como pueden ser las aplicaciones que los gestionan. En este contexto puede señalarse que el MINISDEF estableció su «Estrategia de la Información» mediante la Orden DEF/1196/2017, de 27 de noviembre, (https://www.boe.es/diario_boe/txt.php?id=BOE-A-2017-14417, visitado el 21/12/2020) en la que se reconoce a la información como un recurso estratégico y se señala, en la motivación de la Orden, que «la administración debe adoptar una nueva cultura de la información y estar preparada para recoger, generar y tratar grandes volúmenes de información digital sobre sus operaciones, procesos y resultados. Igualmente, subraya que el desarrollo de las capacidades de análisis trasversal de la información permitirá optimizar la gestión, mejorar la toma de decisiones y ofrecer servicios interdepartamentales de manera independiente a la estructura administrativa».

[49] En la página web de la Moncloa podemos encontrar una reseña de la «Agenda Digital» aprobada por el Gobierno en 2013 (https://www.lamoncloa.gob.es/consejodeministros/Paginas/enlaces/150213enlaceagendadigital.aspx, visitada el 21/12/2020)

OBJETIVOS ESTRATÉGICOS	LÍNEAS DE ACCIÓN ASOCIADAS
OE1. Incremento de la productividad y eficacia en el funcionamiento interno de la Administración	LA1. Transformar los procesos de gestión internos de las unidades administrativas en electrónicos LA2. Desarrollar el puesto de trabajo digital
OE2. El canal digital ha de ser el medio preferido por ciudadanos y empresas para relacionarse con la Administración	LA3. Proveer servicios públicos digitales adaptados a las nuevas tecnologías LA4. Mejorar la satisfacción del usuario en el uso de los servicios públicos digitales LA5. Promover la innovación en la prestación de servicios
OE3. Mayor eficiencia en la prestación de los servicios TIC en el seno de la Administración	LA6. Proveer de manera compartida servicios comunes
OE4. Gestión corporativa inteligente del conocimiento, la información y los datos	LA7. Publicar la información disponible para ciudadanos y empresas LA8. Disponer de sistemas de análisis de datos para la toma de decisiones
OE5. Estrategia corporativa de seguridad y usabilidad	LA9. Garantizar la seguridad de los sistemas de información de la AGE y sus organismos públicos

Figura 3-3 Objetivos Estratégicos y Líneas de Acción de la estrategia TIC en la AGE (tomada de [35])

La pirámide normativa propia del MINISDEF que define su estrategia de transformación digital se establece en su cúspide a través de la «Política CIS/TIC»[50]. Esta política establece 6 grandes ejes estratégicos, que agrupan 20 objetivos estratégicos, y estos, a su vez, 63 líneas de acción. Derivada de esta normativa de alto nivel tenemos:

- Por un lado, el desarrollo normativo asociado que va concretando las acciones a realizar para alcanzar los objetivos, y que comprende el «Plan Estratégico CIS» y, al nivel más bajo, los «Planes de Acción», tanto los sectoriales comunes al MINISDEF como los específicos del EMAD y los Ejércitos.
- Por otro, el desarrollo normativo técnico que define la arquitectura CIS/TIC a utilizar y que comprende la «Arquitectura Global CIS/TIC» (capacidades CIS/TIC), las «Arquitecturas de Referencia» (sistemas CIS/TIC) y las «Arquitecturas Objetivo» (componentes CIS/TIC).

En toda esta estructura normativa, que parte del año 2015 y que todavía se encuentra en desarrollo en sus niveles inferiores, subyace la necesidad de normalizar y modernizar el uso de las TIC en el MINISDEF.

De esta pila normativa me gustaría destacar para el objeto de este trabajo el «Plan de Acción del Ministerio de Defensa para la Transformación Digital»[51]. Dicho plan identifica las 5 unidades TIC que

[50] Orden DEF/2639/2015, de 3 de diciembre, por la que se establece la Política de los Sistemas y Tecnologías de la Información y las Comunicaciones del Ministerio de Defensa [42]. Esta norma viene a sustituir al anterior «Plan Director CIS», orden DEF/315/2002, como punto de partida de las políticas y estrategias CIS/TIC en el ámbito del MINISDEF

[51] Aprobado en sus dos partes mediante la Instrucción 25/2018 [43], de 25 de abril, y la Instrucción 14/2020 [44], de 15 de abril, del Secretario de Estado de Defensa.

gestionan y facilitan servicios en los 7 ámbitos funcionales del MINISDEF (ver Tabla 3-1) y contextualiza el punto de partida de la digitalización del Ministerio: más de 600 sistemas de información (en 150 de los cuales intervienen ciudadanos, empresas u otras administraciones), 47 páginas web en Internet y 31 en la Intranet, y 102 procedimientos para ciudadanos en sedes electrónicas. El plan destaca que se han conseguido avances en la automatización de procedimientos y en la gestión por procesos, pero podemos preguntarnos si realmente estos avances pueden considerarse suficientes y si están en línea con los adelantos y tendencias tecnológicas actuales.

Ámbitos funcionales	Unidades TIC
Estado Mayor de la Defensa (EMAD)	Jefatura de sistemas de Información y Telecomunicaciones (JCISFAS)[52]
Ejército de Tierra (ET)	Jefatura de los Sistemas de Información, Telecomunicaciones y Asistencia Técnica del Ejército de Tierra (JCISAT-ET)
Armada (AR)	Jefatura de Servicios Generales, Asistencia Técnica y sistemas de Información y Telecomunicaciones de la Armada (JCIS-AR)
Ejército del Aire (EA)	Jefatura de Servicios Técnicos y Sistemas de Información y Telecomunicaciones del Ejército del Aire (JSTCIS-EA)
Secretaría de Estado de la Defensa (SEDEF)	Centro de Sistemas y Tecnologías de la Información y las Comunicaciones (CESTIC)
Subsecretaría de Defensa (SUBDEF)	CESTIC
Secretaría General de Política de Defensa (SEGENPOL)	CESTIC

Tabla 3-1 «Plan de Acción del MINISDEF de Transformación Digital». Ámbitos del MINISDEF y unidades que proporcionan servicios TIC (elaboración propia).

En lo referente a la organización, puesta de manifiesto con la identificación de las denominadas unidades TIC, podemos decir que el EMAD y los ejércitos disponen de unas Jefaturas CIS, que son las responsables de analizar las necesidades y proveer y mantener los servicios CIS para cada uno de ellos. Para el resto del Ministerio, incluyendo órganos especiales como puede ser la Casa de S.M. el Rey, el CESTIC desempeña los mismos cometidos. No obstante, en línea con los grandes objetivos estratégicos del departamento, los servicios de infraestructura y plataforma son proporcionados de forma centralizada por CESTIC[53]. Por tanto, ciñéndonos al campo de los productos software en el ámbito administrativo, podemos aseverar que las Jefaturas CIS y el propio CESTIC actúan como equipos de desarrollo mientras que el CESTIC es el único órgano que desempeña las tareas de un equipo de operaciones.

Al preguntarnos si en este desarrollo normativo, incluyendo a la normativa técnica de Arquitecturas CIS/TIC mencionada, se señala un modelo a adoptar para definir el ciclo de vida del software, la respuesta rápida sería que no. Existen, no obstante, ciertas referencias importantes que muestran el

[52] Órgano que ha desaparecido en 2020 y cuyos cometidos han sido absorbidos por el Mando de Ciberdefensa del EMAD.

[53] Uno de los ejes estratégicos de la «Política CIS/TIC» es avanzar hacia una única Infraestructura Integral de Información para la Defensa, I3D, gestionada por CESTIC y común tanto para los servicios administrativos del conjunto del Ministerio como para los servicios operativos del EMAD y las FAS.

interés en proporcionar de forma adecuada los servicios CIS necesarios a los usuarios[54]. En concreto el «Plan Estratégico de los CIS/TIC» especifica que en uno de los Planes de Acción a desarrollar, el de Planeamiento y Obtención de los Recursos CIS, se deben definir los procesos a seguir de acuerdo con las etapas correspondientes del ciclo de vida de los servicios, sus procesos y funciones, según las buenas prácticas en la materia, nombrando explícitamente como ejemplo a ITIL[55] v3 de 2011. Cabe recordar que un servicio TI es un concepto más generalista que el de aplicación o software.

Asimismo, el Plan Estratégico establece que el proceso de obtención de los CIS/TIC forma parte del de los recursos de armamento y material o de I+D del Ministerio y deberá tomar en consideración las fases e hitos establecidos en la Instrucción 72/2012, de 2 de octubre, del Secretario de Estado de Defensa, en la que se regula el proceso de obtención del armamento y material y la gestión de sus programas. Este proceso, en mi opinión, resulta demasiado rígido y prolongado, con un elevado número de hitos documentales, estudios exhaustivos, complejos procedimientos de aprobación y muy orientado a las adquisiciones y al ciclo de vida de armamento y recursos materiales, y no aplicable directamente al software o a recursos CIS en general. El propio «Plan Estratégico de los CIS/TIC» lo aprecia así cuando establece que, no obstante lo anterior, CESTIC, en coordinación con la Dirección General de Armamento y Material (DGAM), desarrollará los procedimiento de colaboración y coordinación necesarios, los cuales se integrarán en el ciclo de vida de los CIS/TIC; contemplará procedimientos especiales y abreviados que tomen en consideración las especificidades de los recursos CIS; posibilitará que los procesos de planeamiento y obtención de estos recursos CIS sean ágiles y permitan una adaptación eficaz a situaciones sobrevenidas; y determinará los hitos documentales que deben generarse en cada una de las fases y etapas del modelo de ciclo de vida. Como conclusión puede decirse que para la obtención de los recursos TIC mediante inversiones financieras que impliquen adquisiciones de bienes o servicios se reconoce la necesidad de establecer procedimientos propios que además resulten ágiles, pero que no deben resultar totalmente ajenos al proceso general de obtención de recursos en el Ministerio. Asimismo, es de señalar que estos procedimientos, ligados a adquisiciones, no rigen para los desarrollos propios puesto que estos no suponen una inversión económica sino un empleo de los recursos TI disponibles.

3.2 La transformación digital en el Ejército de Tierra

La transformación digital, vista en el apartado anterior en la AGE y en el MINISDEF, ha sido recogida de forma particular en el Ejército de Tierra mediante la emisión de la Directiva 08/20, de 28 de octubre, del Jefe de Estado Mayor del Ejército (JEME), sobre la «Gestión de la Información y el Conocimiento (GIC) y Transformación Digital en el ET».

La transformación digital se considera un plan estratégico en el ET, fundamental para lograr el Ejército del futuro junto a la iniciativa «Fuerza 35» [36], proyecto a largo plazo de transformación de las fuerzas terrestres donde se están llevando a cabo experimentos, en el campo operativo, con estructuras, procedimientos y tecnologías emergentes y disruptivas que permitirán actuar a las unidades operativas que se formen para el combate de forma más eficaz y muy rápida, sin pausas operacionales ni tácticas, para lograr sus objetivos en tiempos muy cortos.

La transformación digital en el ET se compone de tres pilares:

[54] En el contexto del desarrollo de software en el MINISDEF los usuarios o clientes forman parte de la propia organización, es decir, son los organismos y unidades que constituyen el Ministerio.

[55] Como ya se he expuesto, la «Information Technology Infraestructura Library» o «Biblioteca de Infraestructura de Tecnologías de Información» que engloba un conjunto de conceptos y buenas prácticas usadas para la gestión de los servicios de tecnologías de la información, el desarrollo de tecnologías de la información y las operaciones relacionadas con la misma en general [31]. Las etapas que se definen en el ciclo de vida de los servicios TI, cada una a su vez compuesta por una serie de procesos, son cinco: estrategia, diseño, transición, operación y mejora continua del servicio.

- Los procesos operativos y funcionales[56]. Se ha de pasar del actual modelo sistémico basado en la gestión por funciones al modelo de gestión por procesos. Según el modelo actual, cada necesidad se resuelve por medio de un sistema. Así contamos, por ejemplo, con un sistema para el apoyo logístico, con un sistema de acuartelamientos, con un sistema de administración económica, etc. Sistemas que en su conjunto componen el denominado sistema de mando y dirección del ET (SIMADE). Cada uno de estos sistemas implica la existencia de una estructura funcional asociada de órganos apoyados y órganos de apoyo. En el modelo futuro cada necesidad tendrá respuesta mediante un servicio de usuario en base a procesos, cuyos elementos de entrada y salida serán los datos y productos de información, y servicios TIC comunes a todo el MINISDEF[57].
- Los datos y productos de información. La transformación debe tener un enfoque centrado en la persona, en el combatiente, orientada a la misión a cumplir y basada en la gestión de la información y los datos. Se pretende obtener superioridad en la información[58] y acortar el ciclo de decisión. Es más importante la cultura del dato que la tecnología que se emplee.
- Los servicios TIC. Se deben aprovechar las ventajas de las tecnologías para disminuir las cargas burocráticas, ganar en agilidad y automatizar las tareas susceptibles de ello. El ET cuenta con 52 aplicaciones o sistemas de información[59] que deben ser analizados para comprobar su orientación a los procesos y a servicios, y su no duplicidad con otros servicios incluidos en el catálogo común que se elabore en el MINISDEF. Durante la implantación de la estrategia de transformación digital en el Ministerio, según se refleja en la Directiva 08/20 citada, como medida para conseguir la alineación del software con los procesos que se definan, el desarrollo de nuevas aplicaciones estará restringido y el ET solo podrá gestionar necesidades que supongan pequeñas modificaciones, adaptaciones meramente tecnológicas (no consideradas modificaciones de alto impacto), o peticiones sobre el funcionamiento de servicios en operación, y que puedan responderse mediante mantenimientos correctivos o evolutivos, y gestionarse sobre la base de componentes programados y expedientes en curso.

Todo lo visto hasta ahora casa perfectamente con la adopción de DevOps en el área de servicios TIC como la cultura de referencia en el desarrollo de aplicaciones. El concepto de transformación digital en el ET gira entorno al cambio de cultura que nos saque del apego a las formas de organizarnos, de pensar y actuar tradicionales; propugnando la adopción de todo lo que favorezca la agilidad, flexibilidad y capacidad de adaptación; así como al empleo de técnicas y tecnologías avanzadas y disruptivas[60].

[56] Los operativos se refieren a los empleados en operaciones militares y los funcionales, también denominados administrativos o corporativos, se refieren a los procesos no operativos que se desarrollan para permitir las actividades diarias y en cuanto a organización componente de la administración general.

[57] La plataforma ARGO (Armonización para la Gestión de la Organización) es la herramienta de CESTIC, a disposición de todos los ámbitos del MINISDEF, para la racionalización de sistemas, procesos, datos y servicios TIC.

[58] La Orden DEF/1196/2017 sobre «Estrategia de la Información» en el MINISDEF define la superioridad en la información como «ventaja relativa que se genera mediante el empleo de información relevante, principios y capacidades disponibles, de forma continua y dirigida adaptándose a cada situación. Este concepto debe entenderse más como la capacidad de adaptación al entorno cambiante en el que se actúe, que con el control de la información en sí misma. La superioridad ya no solo depende de la tecnología ni de la rapidez con la que la información se explote, sino con su oportunidad y capacidad de adaptación al contexto en el que se opere».

[59] Estas 52 aplicaciones se agrupan en 16 áreas: gestión de personal; planeamiento, control y decisión estratégica; gestión sanitaria; gestión económica-financiera; gestión logística y de recursos materiales; gestión administrativa, apoyo, vida y funcionamiento; enseñanza, formación y adiestramiento; gestión de infraestructuras; gestión documental y mensajería; gestión TIC; apoyo a la decisión política y relaciones internacionales; seguridad; y operaciones.

[60] Según el Diccionario de la Real Academia Española, disruptivo significa que produce rotura o interrupción brusca. En el contexto de la transformación digital en que se emplea se usa para calificar a las tecnologías avanzadas e innovadoras que suponen nuevas tendencias y rompen con los viejos conceptos tecnológicos.

3.2.1 JCISAT como órgano impulsor de DevOps

La Directiva 08/20 del JEME marca misiones a diferentes organismos para alcanzar los objetivos de la transformación digital en el ET. A la JCISAT, como gran unidad CIS/TIC del ET y responsable técnico, le asigna, entre otros cometidos, el de constituir una unidad con un núcleo de personal del ET apto para el diseño técnico y automatización de los procesos modelados, su codificación y despliegue en la próxima Plataforma para la Armonización de la Gestión de la Organización (ARGO) de CESTIC. Con la transformación digital en curso le seguirá correspondiendo, por tanto, el papel de equipo de desarrollo. Como tal su capacidad fundamental será la de poder producir y mantener de forma ágil y con un elevado nivel de calidad las aplicaciones demandadas. Y en conjunción con el equipo de operaciones, que seguirá localizado en CESTIC, su objetivo esencial consistirá en ofrecer productos de alto valor de forma constante a los usuarios finales.

En mi opinión JCISAT es, en consecuencia, la entidad del ET adecuada para, como interlocutor de CESTIC, impulsar la adopción de DevOps en el ámbito del MINISDEF con el objetivo de ofrecer de forma ágil a cualquier usuario aplicaciones útiles y de calidad a la vez que se consigue una cultura que facilita el trabajo y ofrece seguridad y un mayor nivel de satisfacción a todos los profesionales de las TI que desempeñen sus cometidos tanto en equipos de desarrollo como de operaciones.

3.3 Procedimiento formal de solicitud de modificación o desarrollo de aplicaciones

El procedimiento de solicitud de modificación o desarrollo de servicios TIC[61] en el ámbito del Ministerio está regulado por la Instrucción Técnica (IT) 01/20 de CESTIC, de 26 de junio de 2020, «"Gestión de la Demanda". Solicitud de nueva necesidad de servicio CIS/TIC en el MINISDEF». La adquisición, desarrollo o modificación de aplicaciones está, por tanto, incluido en el proceso descrito en esta IT, aunque solo afecta a los nuevos desarrollos o a las modificaciones de envergadura, dejando que las modificaciones menores o las pequeñas tareas de mantenimiento evolutivo o correctivo sigan sus propios procesos.

En síntesis, el proceso comprende las siguientes tareas:

- Fase 1, solicitud (duración no superior a 15 días hábiles):
  - Elaboración de la solicitud por cualquiera de los siete ámbitos del MINISDEF (Jefaturas CIS o el propio CESTIC) con toda la información relevante necesaria (técnica, legal, etc.) y remisión a CESTIC.
  - Comprobación por CESTIC de que no existe ya un servicio que satisfaga la necesidad.
  - Estudio inicial y propuesta al resto de ámbitos de inclusión en el proyecto por si el servicio les fuera de utilidad.
- Fase 2, análisis de la solicitud (duración no superior a un mes):
  - Análisis de la viabilidad técnica y decisión sobre si realizar un desarrollo propio o adquirir un producto ya existente[62].
  - Valoración de los recursos humanos, materiales y financieros necesarios tanto para el desarrollo o adquisición como para su mantenimiento y operación, así como evaluación de otras necesidades o aspectos que puedan ir asociados como formación, aspectos legales, análisis de riesgos, impacto sobre otros servicios ya en explotación, etc.
  - Posibilidad de realizar el desarrollo con recursos, humanos, materiales o financieros, aportados por el solicitante (en el caso de las Jefaturas CIS).

[61] Estos servicios pueden ser: servicios y aplicaciones de usuario, servicios de infraestructura tecnológica, servicios de seguridad de la información y servicios de gestión.

[62] Este producto puede ser comercial («Comercial off-the-shelf» o COTS), gubernamental (GOTS), de la OTAN (NOTS) o de la Unión Europea (UEOTS).

- Fase 3, aprobación de la solicitud (duración no superior a 15 días hábiles):
    - Se aprueba o se rechaza la solicitud formalmente. En caso de aprobación y de requerirse adquisición de productos o servicios, la necesidad se incorpora al planeamiento financiero y al proceso de obtención de recursos CIS conforme a la Instrucción 67/2011 del Secretario de Estado de Defensa que regula el proceso de obtención de recursos materiales.

Tras este proceso, que correspondería con la fase de planeamiento en el ciclo DevOps, se llevaría a cabo el diseño, implantación y puesta en producción del servicio. Como vemos, los tiempos de este planeamiento pueden ser superiores a los dos meses. Además, en el caso de solicitudes desde ámbitos distintos a CESTIC como, por ejemplo, desde el ET, habría que sumar los tiempos para el planeamiento y estudios iniciales propios. Aunque este proceso pueda resultar muy laborioso y rígido para la cultura DevOps, solo se llevaría a cabo de forma completa en la primera iteración del ciclo de vida y no es descartable que pueda ser automatizado en cierta medida.

3.4 El modelo actual de desarrollo e implantación de aplicaciones en el MINISDEF

En la actualidad el Ministerio mantiene dos grandes entornos CIS de carácter privado, separados y diferenciados (ver Figura 3-4): el de propósito general, permanente y estático, donde encontramos todas las redes locales de los acuartelamientos y la intranet corporativa en que funcionan todos los servicios de carácter administrativo y funcional; y el de mando y control, donde se establecen las redes de comunicaciones y datos así como los sistemas de información y las aplicaciones relacionados con las operaciones militares, con requisitos de seguridad más elevados, y en las que estarían incluidas las redes móviles y temporales propias de las unidades. Con la futura I3D (Infraestructura Integral de Información de la Defensa) se busca una mayor integración de estos dos entornos, el funcional y el operativo, avanzando hacia una única infraestructura (ver Figura 3-5).

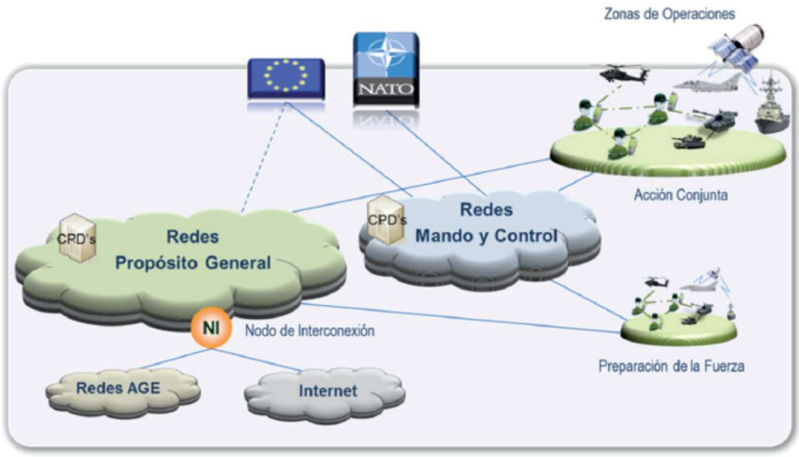

Figura 3-4 Situación actual de las infraestructuras CIS/TIC del MINISDEF (tomada de [37])

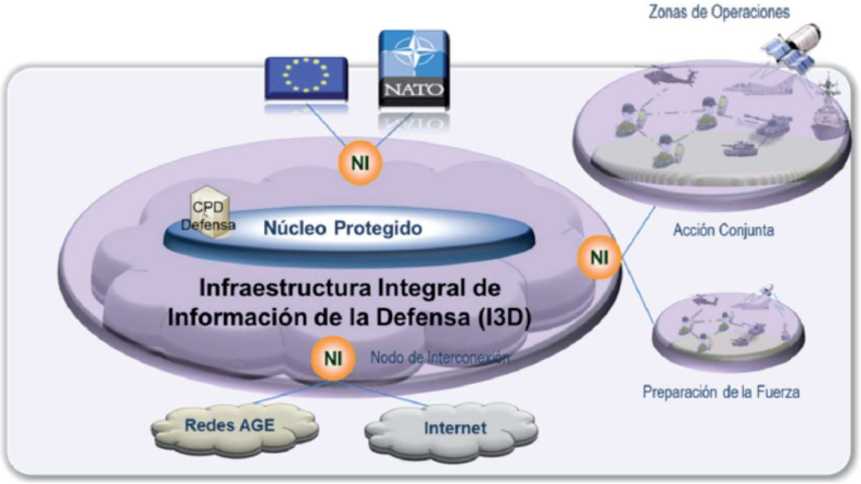

Figura 3-5 Situación objetivo de las infraestructuras CIS/TIC del MINISDEF (tomada de [37])

En la Tabla 3-1 se exponen las unidades que proporcionan servicios TIC en el MINISDEF. De ellas solo en CESTIC puede considerarse que encontramos al equipo de operaciones que da servicio a todo el Ministerio, puesto que los centros de procesos de datos dispersos pertenecientes a los ejércitos y otros órganos han ido desapareciendo. Esto corresponde con parte de las funciones que se le asignan al CESTIC en el Real Decreto 372/2020, de 18 de febrero, por el que se desarrolla la estructura orgánica básica del MINISDEF. En concreto se le atribuye «controlar la operación y el mantenimiento de los sistemas y tecnologías de la información y las comunicaciones para la provisión directa de los servicios correspondientes en el marco de la I3D, asumiendo los cometidos de la Autoridad Operacional de todos sus sistemas».

Además de controlar la operación, la vocación del CESTIC es también dirigir el diseño de los CIS y gestionar los desarrollos que se realicen[63]. No obstante, en la actualidad el resto de unidades TIC del Ministerio, en mayor o menor medida, incluyendo al propio CESTIC[64], cuentan con equipos de desarrollo y gestionan la contratación o la realización de aplicaciones, tanto para el ámbito funcional como para el operativo, siendo coordinados desde CESTIC según el procedimiento expuesto en el apartado anterior.

Para poder evaluar desde el punto de vista DevOps el desempeño de los equipos de desarrollo y operaciones en el MINISDEF realicé unas entrevistas a responsables de estos equipos. En primer lugar, entrevisté a un representante de la unidad TIC del ET[65] cuyo trabajo fundamental es el desarrollo y

[63] El Real Decreto 372/2020 también asigna al CESTIC las funciones de «dirigir el diseño, la obtención y la configuración de los sistemas y las tecnologías de la información y las comunicaciones y de la seguridad de la información para garantizar la normalización, homologación y estandarización de dichos sistemas y su plena interoperabilidad, en el marco de la Infraestructura Integral de Información para la Defensa (I3D) y de los acuerdos nacionales e internacionales en los que España sea Parte que afecten a dichos sistemas» y «gestionar los recursos, programas, proyectos, desarrollos y la implantación de los sistemas y tecnologías de la información y las comunicaciones en el marco de la I3D y de los acuerdos nacionales e internacionales vigentes, así como garantizar su integración e interoperabilidad, y la gestión y supervisión de los correspondientes servicios». Si CESTIC dirige el diseño, obtención y configuración de las TIC (arquitectura TIC), gestiona los recursos, programas, proyectos, desarrollos e implantación de las TIC (desarrollo) y controla la operación y el mantenimiento de los TIC (operaciones), cabe pensar que es posible que en el futuro desaparezcan las unidades TIC del resto de organismos del MINISDEF siempre que el CESTIC sea capaz de asumir de forma completa y eficiente todas estas funciones.

[64] CESTIC cuenta con una unidad denominada DIVINDES (División de Infraestructuras y Desarrollos) dedicada a los desarrollos.

[65] Con destino en la JCISAT, Subdirección CIS (SUBCIS), Sección de Aplicaciones y Procesos Técnicos, localizada en el acuartelamiento «Capitán Sevillano» en Pozuelo de Alarcón, Madrid.

mantenimiento de aplicaciones en el ámbito de propósito general [38], y, posteriormente, mantuve una reunión con uno de los responsables de la unidad de CESTIC[66] dedicada a la gestión y mantenimiento del centro de procesos de datos del Ministerio [39]. La posición orgánica relativa de CESTIC y del ET dentro de la estructura del MINISDEF puede verse en la Figura 3-6. Aunque pueda parecer bastante alejada la posición de CESTIC con el resto de órganos TIC del Ministerio, existe una relación funcional directa, lo que facilita la comunicación sin seguir las cadenas orgánicas.

Figura 3-6 Organigrama simple del MINISDEF con CESTIC y el ET (elaboración propia)

Los guiones seguidos para el desarrollo de las entrevistas se encuentran recogidos en los Anexos I y II a este documento. Las conclusiones principales de los asuntos y temas debatidos, que se trataron intentando seguir un hilo argumental basado en el ciclo continuo DevOps, se incluyen a continuación.

3.4.1 La visión del equipo de desarrollo

La entrevista con los representantes de los equipos de desarrollo de la Jefatura de los Sistemas de Información, Telecomunicaciones y Asistencia Técnica (JCISAT) del Ejército de Tierra tuvo lugar el 26 de octubre de 2020. A continuación, expongo los principales asuntos tratados en ella.

Actualmente se dispone en la JCISAT del ET de hasta tres equipos de desarrollo (ver la Figura 3-7 para su localización orgánica) formados, en general, por un número variable de analistas/ingenieros de categoría de oficial con el diploma de informática militar y programadores de categoría de suboficial con la especialidad de informática. Antes solía externalizarse en mayor medida, acudiéndose a la contratación de empresas para el desarrollo de aplicaciones, pero el elevado coste y la disminución de los créditos disponibles para estos cometidos en el ET hace que se prefiera el desarrollo y mantenimiento propios siempre que existan capacidades suficientes. La idea general es que ante una nueva necesidad y antes de iniciar un desarrollo el ET lo plantee primero al CESTIC con el objetivo de que este estudie si ya se dispone de una aplicación que, con las mínimas modificaciones, en su caso, la satisfaga, o bien si ya existe otra unidad TIC del MINISDEF que se encuentre desarrollándola, evitándose así las duplicidades[67].

[66] Con destino en el CESTIC, División de Operaciones en Red (DIVOPER), localizado en la c/ Arturo Soria 289 de Madrid.

[67] Ver el apartado 3.4 de este documento donde se expone el proceso en detalle.

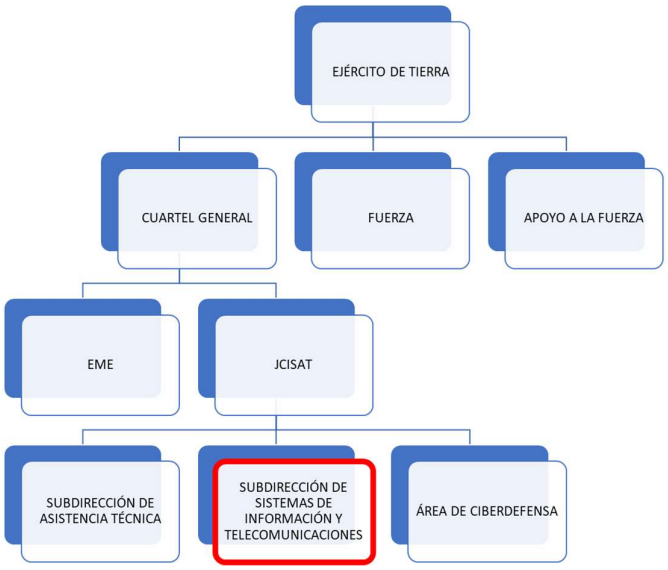

Figura 3-7 Organigrama simple del ET con localización de la SUBCIS (elaboración propia)

Para los desarrollos, integración y pruebas disponen de unos entornos propios, con una pequeña plataforma de hardware separada físicamente del CPD del CESTIC y gestionada por ellos mismos y un reducido equipo de operaciones de una unidad de Transmisiones del ET que trabaja en estrecha colaboración. Una vez entregan la aplicación a CESTIC esta pasa por dos entornos: el de preproducción, donde se realizan las pruebas finales y de validación; y, finalmente, por el de producción.

Durante el análisis y diseño de las aplicaciones se siguen paradigmas clásicos en cascada y, a veces, en espiral, generando la documentación manual reflejada en «Métrica»[68], aunque en la actualidad han abandonado esta forma de documentar y en su lugar están utilizando «Enterprise Architect»[69] para el modelado de las aplicaciones. En el futuro su intención es implantar una metodología ágil basada en «Scrum», habiendo ya realizado cursos formativos. Utilizan como herramientas de apoyo «Jira» para la gestión de tareas y «Confluence»[70] para compartir conocimientos e información. El mantenimiento de las aplicaciones lo realizan siguiendo procedimientos ágiles completamente.

No existe un equipo de pruebas o de control de calidad específico, sino que los equipos disponibles ejercen ese cometido con el software que desarrollan los otros equipos. Las pruebas unitarias y de calidad del software son realizadas internamente por el propio equipo que desarrolla y las pruebas funcionales suelen realizarse por un equipo de desarrollo distinto, siguiendo un modelo en V, en el caso de usar el paradigma clásico, o realimentando los ciclos de «Scrum», en desarrollo ágil. Se utiliza «SonarQube»[71] para el análisis y la evaluación de la calidad del software. Para las pruebas se emplea la herramienta

[68] «Métrica» es una metodología de planificación, desarrollo y mantenimiento de sistemas de información propia de la administración española (https://administracionelectronica.gob.es/pae_Home/pae_Documentacion/pae_Metodolog/pae_Metrica_v3.html#.X-jtxhZ7mUk, visitado el 27/12/2020).

[69] «Enterprise Architect» es una herramienta de la empresa «Sparx Systems» para el diseño y modelado de sistemas de información basada en UML (https://sparxsystems.com/, visitado el 27/12/2020).

[70] «Confluence» es una herramienta de la empresa Atlassian para el trabajo colaborativo en equipo y la gestión del conocimiento común que se integra bien con otros productos de la empresa como «Jira» o «Trello» (https://www.atlassian.com/es/software/confluence, visitado el 27/12/2020).

[71] «SonarQube» es un software de código libre que dispone de diversas herramientas para analizar código y obtener métricas, como código duplicado, estándares de codificación a utilizar, errores potenciales, etc., que permiten mejorar la calidad del mismo. Se integra bien con varias herramientas de ayuda a la programación como «GitHub», «GitLab» o «Bucket» y de construcción y CI/CD como por ejemplo «Maven» y «Jenkins» (https://www.sonarqube.org/, visitado el 28/12/2020).

«JUnit», puesto que se utiliza el lenguaje Java o sus variantes para programación. Una vez realizadas las pruebas unitarias y funcionales se realizan con el usuario las pruebas de validación con el uso de casos de prueba previamente diseñados.

Los productos a entregar se empaquetan en ficheros «war»[72] a los que se les añade, en caso necesario, los scripts de configuración de las bases de datos[73]. La entrega al equipo de operaciones de CESTIC se realiza mediante un servicio FTP. Una vez entregado el software pasa por un entorno de preproducción en que es sometido a más pruebas funcionales, por parte del equipo de desarrollo, y de validación, por parte de los clientes, y donde, obligatoriamente, debe pasar una auditoría de seguridad realizada por personal específico del área de seguridad de CESTIC. Esto último supone un cuello de botella significativo y, actualmente, las aplicaciones están en cola unos cinco meses para pasar la auditoría, que suele durar unas dos semanas, y posteriormente ser certificadas antes de poder ser desplegadas en el entorno de producción.

Se dispone a nivel ministerial de un «Centro de Atención a Usuarios» (CAU), atendido permanentemente (24x7), y de una herramienta corporativa denominada «Sistema de Control de Acuerdo de Nivel de Servicio» (SCANS) para proporcionar servicio de soporte TIC con gestión de incidencias, peticiones y consultas[74]. Es la herramienta que se utiliza para relacionar al equipo de desarrollo con el equipo de operaciones y, una vez en el entorno de producción, con los administradores funcionales y los usuarios, realizándose a través de ella la administración y el mantenimiento de las aplicaciones. El mantenimiento correctivo se realiza mediante la generación de incidencias en SCANS mientras que las peticiones de mantenimiento evolutivo necesitan una solicitud formal desde el órgano que ejerce de cliente mediante el sistema de correo oficial[75]. En caso de grandes cambios se inicia un ciclo de desarrollo nuevo completo, con análisis, diseño, casos de uso, pruebas, etc.

Las reuniones con los usuarios se suelen limitar a la inicial para recoger los requisitos y a las necesarias para realizar las validaciones cuando se dispone de prototipos de la aplicación a desarrollar. En general no se mantienen reuniones o contactos con otros equipos que puedan participar en el ciclo de vida del software. Se utiliza la herramienta SCANS para establecer los contactos normalizados con esos equipos siguiendo los procedimientos establecidos en ella. Asimismo, se utilizan ciertos mecanismos que reducen la necesidad de esos contactos como puede ser el empleo de librerías estandarizadas en el Ministerio para «front-end» de aplicaciones, que ya están probadas y validadas para uso general. Esta falta de contactos iniciales supone que no existan restricciones, limitaciones o recomendaciones específicas trasladadas al equipo de desarrollo desde otros equipos. Cuando se realiza la petición de los entornos de preproducción y producción a CESTIC, al realizar la entrega de la aplicación, se puede pasar una estimación de necesidades de recursos, número de usuarios, requisitos de servidores, copias de seguridad, etc.

En relación con los tiempos de desarrollo se estima que una aplicación puede tardar entre uno y dos años desde que se inicia el proyecto hasta que se pone en producción, dependiendo, sobre todo, de la experiencia de los programadores con los que se cuente ya que estos suelen ser escasos y puede que no tengan continuidad en el puesto. Tan solo la generación de los requisitos y la documentación inicial puede llevar más de dos meses. Una vez en producción, los fallos se abordan de inmediato y la corrección, envío de los nuevos paquetes compilados con la aplicación completa y su puesta en explotación en sustitución de la versión anterior puede ser cuestión de horas o días a lo sumo. Las mejoras o nuevas funcionalidades pueden tardar meses o años, en función de ventanas programadas para ello y de la carga de trabajo de analistas y programadores. Las entregas por mejoras o nuevos requisitos

[72] Los archivos con extensión «.war» son archivos «.jar» de empaquetado de aplicaciones en lenguaje Java específicos para aplicaciones web (https://es.wikipedia.org/wiki/WAR_(archivo), visitado el 28/12/2020).

[73] Se utiliza «SQL Server» de Microsoft como sistema de gestión de bases de datos.

[74] Este tipo de herramientas también se denominan en inglés «ticketing system» o «helpdesk system».

[75] El MINISDEF dispone de un sistema de correo electrónico denominado SIMENDEF que permite el envío de oficios y mensajes oficiales en formato electrónico entre las diversas autoridades telegráficas del Ministerio.

suelen producirse cada dos o más años y se estima que las solicitudes de los usuarios en este sentido se atrasan o no se producen porque tardan mucho en ver los resultados.

El porcentaje de estas modificaciones que resulta fallido al ponerlas en producción y requiere de una vuelta atrás es prácticamente de cero debido a los controles de calidad y a la realización de múltiples pruebas. Entre un 10% y un 15% de los fallos detectados en las fases iniciales se debe a la falta de experiencia de los programadores, incluyendo en este caso a programadores contratados que, además, suelen cambiar con mayor frecuencia que los programadores militares.

Los lenguajes de programación, librerías y «frameworks» que utilizan están basados en Java, con «SQL Server» como gestor de bases de datos. Se dispone de librerías y de un «framework» propios que homogenizan y facilitan los desarrollos y que se utilizan también cuando se realizan contrataciones con empresas de desarrollo externas. En el futuro tienen previsto adoptar la tecnología de contenedores con «Docker» y «Kubernetes». Actualmente no tienen una arquitectura de programación definida, aunque básicamente desarrollan aplicaciones web y utilizan los servicios de SOA disponibles en CESTIC.

Las principales herramientas tecnológicas que emplean para apoyar el desarrollo son «Jira», para gestionar las tareas que comprenden el proyecto, «GitHub», como repositorio de código y gestor de versiones, y «Jenkins», para la construcción y empaquetado de las aplicaciones. La gestión y control de la documentación asociada al desarrollo se realiza a través de «Confluence».

El equipo de desarrollo prácticamente se desentiende de la aplicación una vez se realiza su puesta en producción y solo atiende las incidencias que se les asignen desde SCANS. No existen indicadores de los que tengan conocimiento sobre el desempeño de la aplicación tales como el volumen de datos almacenado y el tanto por ciento de variación periódico, el flujo de datos habitual y los periodos en que este varía, recursos hardware consumidos, número de accesos a la aplicación por horas/días, etc. No se desarrolla previendo una variación importante en las demandas de uso o para asegurar la continuidad del servicio, modulando y preparando el software para la creación y eliminación dinámicas de instancias de forma que se facilite el trabajo del equipo de operaciones. No se realiza ninguna acción para comprobar la satisfacción del usuario frente a la aplicación y ver que esta aporta, al menos, el valor buscado. No existe una forma de comunicación ágil entre todos los actores implicados en el desarrollo y funcionamiento de la aplicación; el procedimiento de comunicación se establece formalmente mediante una instrucción operativa particular[76] (IOP) y se implementa a través de la herramienta SCANS.

En lo relativo a los contactos y comunicación con el resto de actores en el proceso de desarrollo creen que podría mejorarse algo las relaciones no con los usuarios finales, que probablemente saturarían al equipo, pero sí con usuarios intermedios como los administradores funcionales o con quien desempeñase el papel de cliente de la aplicación. Estas relaciones podrían materializarse no solo con SCANS sino también a través del correo oficial y el correo electrónico particular. No existen problemas con los equipos que realizan las pruebas, el control de calidad y la integración y empaquetado ya que o son el mismo equipo o es uno de los equipos de la unidad con los que existe un contacto directo diario. Con los equipos de seguridad y de operaciones no existen en la práctica relaciones.

3.4.2 La visión del equipo de operaciones

La entrevista con los representantes del equipo de operaciones del CESTIC tuvo lugar el 16 de noviembre de 2020. A continuación, expongo los principales asuntos tratados en ella.

La unidad de operaciones (ver la Figura 3-8 para su localización orgánica) dispone de hasta 38 militares y 96 civiles. Estos últimos tienen diversos perfiles que abarcan ingenieros superiores y técnicos y diversos grados y especialidades de formación profesional, e incluyen al personal que gestiona algunos

[76] Una instrucción operativa particular (IOP) de SCANS es un documento que establece el flujo de comunicación y de incidencias y peticiones entre los diferentes roles de usuarios que se implementan en una aplicación o sistema, comprendiendo a los usuarios, administradores funcionales y administradores técnicos, estos últimos pertenecientes a los equipos de operaciones y desarrollo.

de los servicios TIC externalizados por el MINISDEF según los contratos en vigor[77]. El personal militar está constituido por oficiales de la especialidad de transmisiones o con diploma de informática y suboficiales de transmisiones o especialistas en informática. Este personal, en general, tiene experiencia o ha recibido formación en una amplia variedad de temas como SOA, almacenamiento y virtualización, correo electrónico, videoconferencias, sistemas legados (mainframe), directorio activo de Microsoft Windows, seguridad en los puestos de trabajo, sistemas específicos del EMAD, etc. Este personal proporciona servicios propios de los equipos de operaciones permanentemente (24x7).

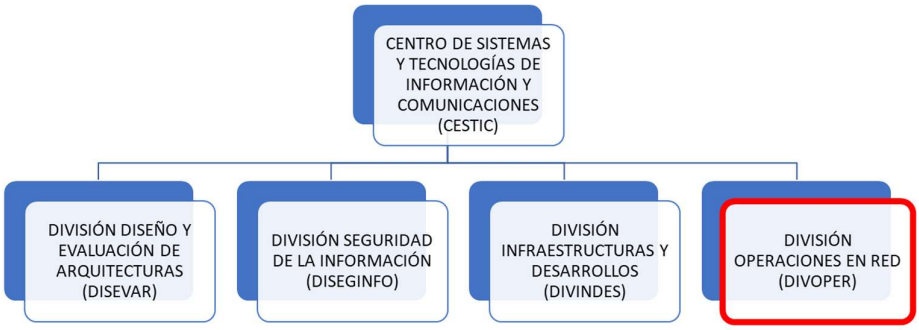

Figura 3-8 Organigrama simple del CESTIC con localización de la DIVOPER (elaboración propia)

Se han realizado cursos en el ámbito de la gestión de servicios TI y el personal puede decirse que está preparado para esta gestión. Las únicas herramientas de ayuda con las que se cuenta en este ámbito son un sistema de gestión del inventario hardware y la aplicación SCANS para gestionar las incidencias y peticiones.

No existen «Service Level Agreement» (SLA) de CESTIC, como proveedor de servicios, con los ejércitos y el resto de órganos del Ministerio, como clientes, donde se definan cuáles son los servicios que se proporcionan y cuáles son los criterios de calidad, disponibilidad, seguridad, asistencia y mantenimiento. Desde los clientes tampoco se demandan estos acuerdos. Lo que sí se demanda desde el equipo de operaciones a los desarrolladores son las características para la realización de las copias de seguridad. Además, cuando una aplicación entra en producción se genera una IOP, donde se plasman los flujos de trabajo relacionados con la atención al usuario y las incidencias, que se implementa en SCANS.

Durante la planificación de las aplicaciones el equipo de operaciones solo interviene para el análisis de viabilidad con los equipos de desarrollo propios de CESTIC, aunque son conscientes de que podría resultar útil participar en etapas tempranas para proporcionar asesoramiento desde el punto de vista de operaciones. De los desarrollos realizados por otras unidades TIC del MINISDEF se tiene conocimiento porque inicialmente están obligados a contactar con CESTIC para comprobar que no se producen duplicidades. Los equipos de desarrollo tienen que ajustarse a las directrices en cuanto a infraestructura

[77] De acuerdo con el «Pliego de Cláusulas Administrativas Particulares» y el «Pliego de Prescripciones Técnicas» para la contratación de servicios e infraestructuras de telecomunicaciones de la I3D del MINISDEF, se ha establecido un acuerdo marco para la contratación de estos servicios dividiéndolos en una serie de 12 lotes que incluyen servicios e infraestructuras para datos, telefonía fija, telefonía móvil, comunicaciones satelitales, acceso a Internet, videoconferencia, etc. En estos documentos que datan de 2017 podemos encontrar la definición de los equipos mínimos de trabajo por cada lote, así como el personal que debe componerlos y su cualificación. Los pliegos están disponibles para consulta en la plataforma de contratación del estado (https://contrataciondelestado.es/wps/wcm/connect/7ed99b29-7da6-4cab-9aaa-9c19d3d675f7/DOC20170609100916PCAP+AM+I3D+01+06+2017.pdf?MOD=AJPERES y https://contrataciondelestado.es/wps/wcm/connect/1ac5e293-7e08-4552-8680-c6d0f4328ab1/DOC20170609100551PPT+I3D.pdf?MOD=AJPERES, visitados el 28/12/2020).

a utilizar y arquitectura a implementar que se marcan desde el Área de Planes y Políticas de la División de Diseño y Evaluación de Arquitecturas (DISEVAR) de CESTIC.

Desde el equipo de operaciones se controla solamente el entorno de desarrollo y pruebas de los equipos de desarrollo de CESTIC. Sí se encargan de gestionar los entornos de preproducción y el de producción, que son comunes y únicos para todo el Ministerio. En el entorno de preproducción es reseñable que, además de las pruebas técnicas que se realizan antes de pasar al de producción, las aplicaciones son sometidas obligatoriamente a una auditoría de seguridad que es realizada por equipos de la División de Seguridad de la Información (DISEGINFO) de CESTIC, representando actualmente esta última acción un cuello de botella significativo.

Puede considerarse que las relaciones con los equipos de desarrollo propios de CESTIC es continuo a nivel técnico, realizándose reuniones con carácter semanal. Con los equipos de los ejércitos y otros órganos del Ministerio no se suelen mantener contactos excepto si se considera necesario debido a causas extraordinarias. Las relaciones con ellos suelen establecerse a través de la herramienta SCANS. Además de con los equipos de desarrollo se pueden establecer contactos con equipos técnicos específicos relacionados con la puesta en producción de las aplicaciones en función de las características de las mismas.

La infraestructura TI para soporte de las aplicaciones consiste en un CPD principal y otro de respaldo en localizaciones físicas diferentes, además de otro CPD, separado de los anteriores, para sistemas clasificados y que tienen requisitos de seguridad más exigentes. Todos los CPD son propiedad del MINISDEF y no existen servicios en la nube ni externalizados en CPD no propietarios. Se están realizando pruebas para crear una nube privada y para la implementación de una arquitectura basada en contenedores, tipo «Docker» con «OpenShift»[78] de Red Hat, que ya cuenta con el software y hardware necesarios para entornos de desarrollo, preproducción y producción. Las principales soluciones y productos tecnológicos utilizados son[79]:

- Para los servidores, «Linux» de Red Hat como sistema operativo y «Apache Tomcat» como servidor web.
- Para bases de datos, «Oracle Database» de Oracle y «SQL Server» de Microsoft.
- Para virtualización y otros servicios en servidores, las soluciones de VMware como software de virtualización, los productos de F5 Networks y «A10 Networks» de TechnologyPartners para disponibilidad y balanceo de carga de servidores, y «Docker» y «OpenShift» para contenedores.
- Todavía se mantienen aplicaciones en entorno mainframe.
- Para monitorización y seguridad, «Spectrum» y «Nagios» para monitorizar las redes, servidores y aplicaciones. CESTIC dispone, además, de un Centro de Operaciones y Seguridad (COS) específico, dado el alto número de ataques que se detectan diariamente, que cuenta con un nodo de interconexión a internet y herramientas de todo tipo.
- Para almacenamiento, cabinas de discos de la familia de IBM para todos los sistemas, así como una red SAN. También disponen de robot de cintas para copias de seguridad que está siendo sustituido por un sistema de cintas virtuales tanto en entorno mainframe como en el resto.

La arquitectura a utilizar es SOA, aunque todavía perviven sistemas heredados como las aplicaciones del entorno mainframe y otros servicios y aplicaciones específicos. Más allá de los servidores de ficheros y el mantenimiento de los diferentes entornos, no se proporciona ninguna tecnología o herramienta para apoyo al desarrollo tipo repositorio y control de código, pruebas automatizadas o CI/CD. La determinación de la infraestructura TI necesaria para dar servicio a las aplicaciones en producción se

[78] Puede considerarse que «OpenShift» es la versión propietaria y con soporte de Red Hat del sistema de código libre para orquestación de contenedores «Kubernetes».

[79] Existen otros productos que se utilizan o prueban pero que no están listados aquí por ser menos significativos.

realiza durante el estudio de viabilidad, al inicio del desarrollo, en que sí participa el equipo de operaciones (solo para desarrollos de CESTIC). Para la puesta en producción de las aplicaciones se necesita una documentación previa, compuesta por el denominado «documento de despliegue», que contiene los requisitos para realizar el despliegue y que es elaborado por la DISEVAR. No se realiza un control de las versiones de las aplicaciones en producción o en el resto de entornos y tan solo se gestionan las versiones de los productos que componen la propia plataforma TI.

El ritmo aproximado de puesta en producción de aplicaciones es de unas cinco diarias por mantenimiento correctivo o evolutivo de pequeña entidad y de dos o tres al mes nuevas o tras ser sometidas a una gran modificación. La transición de las aplicaciones entre entornos y las configuraciones de los recursos necesarios se hacen de forma manual y no se realizan confirmaciones o contactos con los equipos de desarrollo para este proceso.

Solo se mantienen unos pocos indicadores del desempeño de las aplicaciones que tienen incidencia directa con el buen funcionamiento de la infraestructura TI como pueden ser el volumen de datos y su tasa de crecimiento, así como el número de accesos para aplicaciones grandes. En caso de problemas determinados, como incidentes sobre ralentización en el uso, se realizan estudios específicos. Los fallos en las aplicaciones detectados durante la instalación de las mismas suelen deberse a errores en el diseño mientras que los fallos en producción suelen tener como origen la saturación de los recursos asignados, siendo fácilmente solucionables con la dedicación, por ejemplo, de más memoria o procesadores en las máquinas virtuales.

En lo relativo a los contactos y relaciones con el resto de actores que intervienen en el desarrollo y puesta en producción de las aplicaciones consideran que existe buen entendimiento con los equipos de desarrollo[80] a bajo nivel pero que a un nivel superior debería ser más ágil y efectivo. También se identifican como problemas y fuente de retrasos las actuaciones de los equipos de DISEVAR, encargados de la definición y control de las arquitecturas y los documentos de despliegue de las aplicaciones, y como un importante cuello de botella los retrasos introducidos por los equipos de DISEGINFO al realizar las auditorías de seguridad y certificaciones previas al despliegue en el entorno de producción.

[80] Siempre referido a los equipos de desarrollo de CESTIC puesto que con otros equipos del resto de órganos del Ministerio no existen prácticamente relaciones.

4 RESULTADOS

4.1 El desarrollo de aplicaciones en la transformación digital del MINISDEF

Tanto la estrategia TIC de la AGE como la propia estrategia y política CIS/TIC del MINISDEF obligan al Ministerio a su transformación digital como una necesidad para alinearse con el resto de administraciones en el ámbito funcional y de relaciones con los ciudadanos. Esta transformación también está obligada desde el punto de vista de las operaciones militares, dado el contexto actual en el que estas se desarrollan, para poder obtener y mantener la superioridad sobre el adversario, alcanzar los objetivos que se marquen y enfrentar con éxito cualquier posible amenaza.

Esta transformación digital se basa en pasar de una organización sistémica a una organización orientada a procesos funcionales y operativos, centrada en los datos y en los productos de información, y sostenido todo ello por unos servicios CIS/TIC avanzados y eficaces.

El desarrollo de aplicaciones es una de las principales formas de proveerse de estos servicios por lo que cabe concluir que encontrar la mejor forma de hacerlo, adoptando las mejores y más avanzadas costumbres y usos, metodologías y tecnologías que puedan encontrarse en el ámbito de la ingeniería del software es fundamental para que uno de los objetivos estratégicos y pilar de la transformación sea alcanzable y lo más sólido posible.

4.2 El empleo de DevOps en el MINISDEF

Lo primero que habría que destacar es que DevOps no es una moda en la ingeniería del software. Junto a las metodologías ágiles de desarrollo ha venido para quedarse y puede convertirse en un estándar en la producción de aplicaciones en la próxima década. No obstante, hay que tener en cuenta que DevOps no es la solución a todos los problemas y requiere de un largo proceso para establecerse, pero es una filosofía claramente orientada a solucionarlos y no a imponer nuevas complicaciones, trabas y dificultades, sino todo lo contrario.

En mi opinión, y una vez vistas las necesidades estratégicas de transformación digital, DevOps sería la cultura ideal a adoptar por el MINISDEF para dotarse del conjunto de prácticas y herramientas necesarias para obtener el modelo de desarrollo de aplicaciones moderno y eficaz, orientado a satisfacer de forma rápida y flexible las necesidades de los usuarios en la ejecución de procesos y gestión de la información, que el Ministerio requiere. Como hemos podido comprobar muchas de las iniciativas que ya se toman y los avances tecnológicos que se van adoptando desde los equipos de desarrollo y operaciones están conformes con ella.

Para esta adopción no se parte de cero como ha quedado de manifiesto en el capítulo anterior. Bajo mi punto de vista, existen tanto elementos que indican malas prácticas en función de lo que preconiza DevOps, que habría que modificar para adoptar esta cultura, como otros que ya están alineados con ella y favorecen su implantación. Repasémoslos a continuación, pudiéndose extrapolar las consideraciones hechas sobre los equipos de desarrollo del ET a los equipos de desarrollo de los demás ámbitos TIC del Ministerio.

4.2.1 Malas prácticas y hábitos frente a DevOps

La primera y más evidente es el muro existente entre los equipos de desarrollo y el de operaciones. Prácticamente no existen relaciones. Las aplicaciones son creadas en entornos aislados, «lanzadas» al equipo de operaciones para su puesta en explotación y, de facto, olvidadas por el equipo de desarrollo. Esta situación fue precisamente el origen de DevOps. Además, la organización y dispersión de órganos TI refleja la propia realidad del MINISDEF como entidad compleja y es un contexto difícil de cambiar, lo que dificulta la implantación de DevOps y favorece la creación de compartimentos estancos. La situación en relación con los equipos de desarrollo de CESTIC es algo mejor, pero la mera proximidad

física no es la solución. Para mejorar las relaciones habría que adoptar dos conjuntos de medidas: por un lado, establecer un flujo continuo y automatizado de integración y despliegue que incluyese herramientas de configuración automática de los recursos de la infraestructura, y, por otro, implicar a representantes de operaciones en las fases de diseño y desarrollo que, a su vez, deberían estar gobernadas por paradigmas ágiles. La automatización lograda con las primeras medidas liberaría al personal de operaciones de una elevada carga de trabajo mejorando así su disponibilidad para reforzar las relaciones con los equipos de desarrollo.

Los productos que se generan, a pesar de utilizar la arquitectura SOA y el diseño en forma de aplicaciones web, tienen los defectos propios de las aplicaciones monolíticas. Hay una confianza excesiva en los requisitos iniciales y hay poca flexibilidad ante cambios o nuevas necesidades expresadas por los usuarios una vez iniciado el proyecto o con el software en producción. Como resultado se obtiene una insatisfacción por parte de los usuarios y clientes[81]. No obtienen valor añadido e incluso no se cumplen las expectativas desde el primer momento dado los tiempos de desarrollo. Estas herramientas son percibidas en muchas ocasiones más como un lastre que como un beneficio y su uso, en ocasiones, debe ser impuesto mediante órdenes impartidas vía mando. También da lugar a que los organismos usuarios adopten soluciones particulares como, por ejemplo, la descarga local de bases de datos para su tratamiento mediante paquetes ofimáticos. La adopción de metodologías ágiles, tanto en el proceso inicial de desarrollo como durante todo el ciclo de vida, contribuiría en gran medida a resolver este problema. Estas metodologías ágiles tienen como ideas fuerza que las personas e interacciones deben predominar sobre procesos y herramientas, que es más importante tener software que funcione que un gran volumen de documentos extensos y detallados, que resulta más provechoso mantener el contacto con el cliente que atenernos de forma estricta a un contrato y unos requisitos, y que es mejor cualidad la de adaptarse al cambio que la de ceñirse a un plan. Asimismo, entre sus principios se contempla que los usuarios y los desarrolladores trabajen juntos durante todo el proyecto y no solo durante la fase de determinación de requisitos funcionales, siendo la mejor forma de establecer relaciones los contactos directos cara a cara. En el ET encontramos oficiales diplomados en informática en todas las estructuras orgánicas que pueden requerir software específico lo que facilitaría en gran medida la aplicación de este principio.

Las auditorías de seguridad suponen un cuello de botella significativo que retarda en exceso el despliegue de las aplicaciones. Los requisitos impuestos por estas auditorías deberían afrontarse desde etapas tempranas en el desarrollo y no esperar al final del proceso para comprobarlos y certificarlos, implicando al personal de seguridad de la información en el ciclo lo antes posible. DevOps también implica que culturalmente se acepte que los requisitos no funcionales (calidad del código, escalabilidad, manejabilidad, sencillez, seguridad, reutilización, etc.) son de la máxima importancia para la organización, de forma que idealmente, en un ambiente colaborativo y de alta confianza, donde todos se sienten responsables de su trabajo, las actividades de auditorías de seguridad sobre los productos podrían simplificarse y sustituirse por aseguramiento de la calidad en los procesos de desarrollo.

Como ya he expuesto, los flujos de información de izquierda a derecha (desarrollo-operaciones-usuarios) se interrumpen y solo fluyen localmente en cada grupo. Hay que identificar y eliminar o modificar lo que constituye el impedimento a este flujo, ya sean personas, tecnologías o procesos. El flujo de información de realimentación de derecha a izquierda (usuarios-operaciones-desarrollo) sencillamente no existe. No se tiene información de lo que funciona mejor o peor, no se crea conocimiento que mejore la calidad de los procesos y productos, no se generan nuevos objetivos comunes ni iniciativas de mejora y valor añadido, y no se comparten experiencias, positivas o negativas, ni lecciones aprendidas. Para mejorar este aspecto desde el punto de vista DevOps hay que definir e implantar unas métricas e indicativos, junto a herramientas de monitoreo y análisis, para que todos

[81] Durante este trabajo se considera al usuario como el rol desempeñado por quienes utilizan las aplicaciones y a los clientes como los propietarios de las aplicaciones frente a los proveedores de las mismas. También puede emplearse el término usuario para referirse a los dos.

puedan apreciar si el software y las infraestructuras están funcionando de acuerdo a como fueron diseñados. En este sentido, tanto los equipos de desarrollo como los de operaciones deberían tener acceso a herramientas del tipo de «Nagios», «Grafana» o «Elastic Observability», que permiten la supervisión de métricas y parámetros de funcionamiento. Asimismo, se deberían utilizar mecanismos, como sondeos o encuestas, que indiquen si los usuarios alcanzan satisfactoriamente sus objetivos y perspectivas o como podrían mejorarse las aplicaciones.

Hay que aceptar que la experimentación y la aceptación del riesgo que conlleva conducen a la mejora. Esto, junto a la práctica y la repetición, es la mejor forma de alcanzar la excelencia. Se debería promover la cultura de la innovación y la aceptación del riesgo, frente a la comodidad de seguir órdenes e instrucciones de forma rígida, como buenas prácticas para favorecer el progreso de la organización. No hay que ver el error como un fracaso sino como una etapa lógica del aprendizaje y mejora. Tan solo tenemos que asegurarnos que no se producen fallos catastróficos y que ante la aparición de un error se puede retrotraer el sistema a un estado seguro. La implantación de herramientas de pruebas y de una cadena de CI/CD automatizadas contribuye en gran medida a que esto sea posible, aumentando la percepción de seguridad durante todo el proceso de desarrollo y despliegue de aplicaciones.

La asignación de recursos de infraestructura TI y su configuración no puede ser la más eficiente y óptima ya que la falta de comunicación entre los equipos de desarrollo y operaciones, así como el uso de procedimientos manuales, lleva al empleo de asignaciones estandarizadas que solo se modifican en caso necesario ante fallos. En operaciones faltan herramientas que permitan implantar flujos de CI/CD, como «Jenkins» o «GitLab», y para implementar la infraestructura como código (IaC), como «Ansible» o «Puppet», de forma que se facilite la mejora de las relaciones con desarrollo, rompiendo las barreras actualmente existentes, y se automaticen las últimas fases del ciclo DevOps. Hay que tender a emplear la IaC y la tecnología de contenedores de forma que los equipos de desarrollo, convenientemente asesorados, sean capaces de autoabastecerse de recursos sin perjudicar la estabilidad de las infraestructuras TI, reduciendo los tiempos de puesta en explotación y la aparición de errores de configuración de estos recursos.

4.2.2 Buenas prácticas y hábitos frente a DevOps

Los equipos de desarrollo disponen de herramientas de automatización suficientes y las utilizan de forma adecuada en el entorno de desarrollo. Esto significa que están preparados para implantar un flujo automatizado de CI/CD.

Están en proceso de adoptar metodologías ágiles de desarrollo. Si lo hacen e implican al equipo de operaciones y a los usuarios en el proceso es indudable que no solamente lograrán mejorar la velocidad de producción y la calidad de sus productos, sino que empezarán a romper las barreras de comunicación actualmente existentes.

Pretenden adoptar la tecnología de contenedores tipo «Dockers» y «Kubernetes». Esta iniciativa está alineada con el equipo de operaciones, que ya dispone de ella para pruebas. Esta tecnología por un lado facilita el despliegue y la introducción de automatismos al final de la cadena de desarrollo, al suponer una mayor seguridad y sencillez en la puesta en explotación de las aplicaciones, así como un empleo más eficiente de los recursos, y por otro favorece la adopción de arquitecturas que contemplan el empleo de aplicaciones con elementos altamente desacoplados, tales como los microservicios.

El personal de operaciones cuenta con un buen bagaje formativo, de experiencias y conocimiento. Esto puede aprovecharse para la implantación de nuevas herramientas y tecnologías y para compartir sus conocimientos con el resto de equipos, que podrán emplearlos desde etapas más tempranas en los desarrollos conforme vayan mejorando los mecanismos de relación. Cuando un conocimiento se emplea en las fases iniciales de un proceso resulta un elemento constructivo, pero cuando se utiliza al final suele convertirse en una herramienta para descubrir errores y defectos que podían haberse evitado con anterioridad.

Los avances tecnológicos que se pretenden adoptar en operaciones, como servicios en la nube y tecnología de contenedores, coinciden con las iniciativas de los equipos de desarrollo y favorecen la adopción de DevOps al ayudar a la automatización de tareas, la auto provisión de servicios y al aumento en la modularidad de las aplicaciones.

4.3 Estrategias y acciones para adoptar DevOps

Para impulsar el establecimiento de una cultura DevOps hay que huir de la micro gestión, de los planes excesivamente detallados, y utilizar una aproximación holística y flexible. Hay que considerar que los equipos de la organización son los verdaderos expertos en sus campos y los que cuentan con la experiencia adecuada tanto en el trabajo que se requiere de ellos como en el contexto donde lo desempeñan. Siguiendo las técnicas ágiles, es mejor tener un plan global, a grandes rasgos, de los objetivos que queremos alcanzar y a dónde queremos llegar, y, a partir de aquí, ir diseñando y ejecutando acciones a corto plazo que nos permitan alinearnos con ese plan. En mi opinión, el gran objetivo estratégico en una cultura DevOps es el de producir valor para el cliente, que en el caso del MINISDEF es la propia organización, de forma ágil y continua. Lo importante no es el obtener buenos productos sino el poner a disposición del usuario de forma rápida el producto que necesita en cada momento.

Para impulsar su implantación la organización debe tener un conocimiento lo más claro posible de en qué consiste esta filosofía y cuáles son los beneficios que va a aportar, que deben alinearse con los objetivos buscados, debiendo favorecerse su establecimiento desde los puestos directivos. En general, puede decirse que se requerirá introducir cambios significativos en la forma de pensar de los miembros de los departamentos TI, así como un esfuerzo formativo, con la adopción de nuevas herramientas y la enseñanza de nuevas habilidades. La transformación y la mejora deben ser continuas, comenzando con pequeñas acciones y con elementos básicos para posteriormente ir identificando y corrigiendo las limitaciones y defectos que se vayan encontrando y descubriendo dónde se debería actuar para progresar en la adopción de DevOps, volviendo a repetir el proceso en un ciclo de optimización continuo.

Creo que los principales problemas del MINISDEF en la concepción del desarrollo del software son la existencia de compartimentos aislados y la visión de esta tarea, en sí misma, como el principal objetivo. Cada equipo solo aprecia su trabajo y no comprende el de los demás. Al no existir comunicaciones efectivas entre ellos no se resuelven los problemas y trabas de forma holística, contemplando el proceso en su totalidad, sino que se tiende a soluciones locales que siempre resultan de peor calidad y no pueden resolver los problemas de conjunto que afectan a más de un equipo o cuya solución está en la colaboración entre ellos. Las herramientas no son la única solución porque no se puede establecer una cadena automatizada de integración y despliegue continuo si no se establece también la cadena de relaciones continua entre los diferentes equipos que cree el ambiente de colaboración y comprensión mutua necesario. Por otro lado, no existe una cultura de crear valor centrada en los usuarios, sino que se da mayor importancia a cumplir con las tareas recibidas, a cumplir con los objetivos marcados y finalizar los proyectos, cerrándolos y pasando al siguiente. No importa lo que se tarde en alcanzar las metas siempre que se haga bien porque se le da más valor al producto obtenido que a la satisfacción del usuario. Esto es un error evidente que provoca que en la actualidad se puedan tardar hasta dos años en proporcionar una aplicación a un usuario que, además, no participa muy activamente en el desarrollo por lo que al recibirla es muy probable que ya no satisfaga todas sus necesidades o no lo haga de forma adecuada. Además, la lentitud, rigidez y dificultad que entrañan los procesos de modificación o evolución hace que los usuarios desistan de intentar hacerlo y empiecen a ver la aplicación más como un problema y una carga que como una solución y una herramienta útil.

Además de la mentalidad, la adecuada formación del personal y la dotación de las herramientas tecnológicas y las infraestructuras TI adecuadas, otro aspecto que considero muy interesante es la arquitectura genérica a adoptar en el desarrollo de software. En el caso del MINISDEF, y a la espera del desarrollo de los Planes de Acción y las Arquitecturas de Referencia y Objetivo, este aspecto no está claramente definido. En el Plan de Acción para la Transformación Digital, que es el único aprobado hasta el momento, podemos leer en la descripción de la actuación dedicada a «desarrollar nuevos

Sistemas de Información», dentro del grupo de planeamiento y programación de los servicios del MINISDEF, que se ha de adoptar una Arquitectura Orientada a Servicios (SOA) y que los Sistemas de Información existentes (heredados/legados) deberán integrarse a la nueva arquitectura a través del mecanismo de interfaces de servicios web. Este es un buen enfoque que en la actualidad se está viendo superado por el más moderno de Arquitectura Orientada a Microservicios (MSA). Esta última es totalmente compatible con SOA y, en muchos aspectos, puede considerarse como una evolución de la misma en la que los conceptos de sencillez, independencia y desacoplamiento de los servicios desarrollados se eleva a su máxima expresión.

4.3.1 Cambios a implementar

Un resumen de los principales cambios a afrontar en la adopción de DevOps y que hay que tener presentes en el MINISDEF son los que se presentan en la Tabla 4-1.

Nivel en la organización	Mejores prácticas para DevOps
A nivel organización	Proceso de gestión del cambio ágil
	Paradigmas ágiles
	Arquitectura de muy bajo acoplamiento
	Legibilidad y mantenibilidad de código
A nivel equipo	Desarrollo concurrente con repositorios (Git)
	Test automatizados
	Integración continua (CI)
	Despliegue automatizado (CD)
	Infraestructura como Código (IaC)
	Monitorización
A nivel organización y equipos	Cultura centrada en satisfacer al usuario
	Ruptura de compartimentos aislados
	Uso de servicios en la nube
	Uso de virtualización con contenedores
	Pruebas de recuperación ante desastres
	Formación e intercambio de conocimientos

Tabla 4-1 Algunas de las mejores prácticas para adoptar DevOps (tomado de [22] y modificado)

En general se puede decir que los cambios a nivel organización son los que conllevan decisiones sobre arquitecturas y modelos a implantar o políticas a seguir con resultados a largo plazo, que implican imponer estándares que se empleen por todos los equipos, así como directrices globales de funcionamiento. Por el contrario, las capacidades que pueden implementarse a nivel equipo son las que tienen un más rápido impacto, aunque algunas de ellas necesitan o se benefician de la coordinación e impulso desde los niveles organizativos. En esencia y con independencia de donde figuren en la tabla,

puede afirmarse que tanto los esfuerzos a nivel de organización como los de nivel equipo son en la mayoría de los casos interdependientes y concurrentes puesto que se apoyan y necesitan unos en otros.

4.3.2 Situación final objetivo

A la hora de establecer una situación final ideal, es la propia organización la que mejor debería conocer sus capacidades, limitaciones y ambiciones, el entorno en que se desenvuelve y qué es lo que demandan sus usuarios. Es la que debería analizar hasta qué nivel o de qué forma la implantación de DevOps le resultaría beneficioso y cómo debería afrontarla.

Conforme al trabajo desarrollado y expuesto hasta ahora, teniendo en cuenta las buenas prácticas y las tecnologías que mejor se adaptan a la cultura DevOps ya señaladas, así como las prácticas y herramientas que actualmente se utilizan por los equipos de desarrollo y operaciones en el MINISDEF, el modelo de la Tabla 4-1 que propongo podría considerarse como el objetivo a alcanzar para implantar de forma efectiva DevOps en el conjunto del Ministerio.

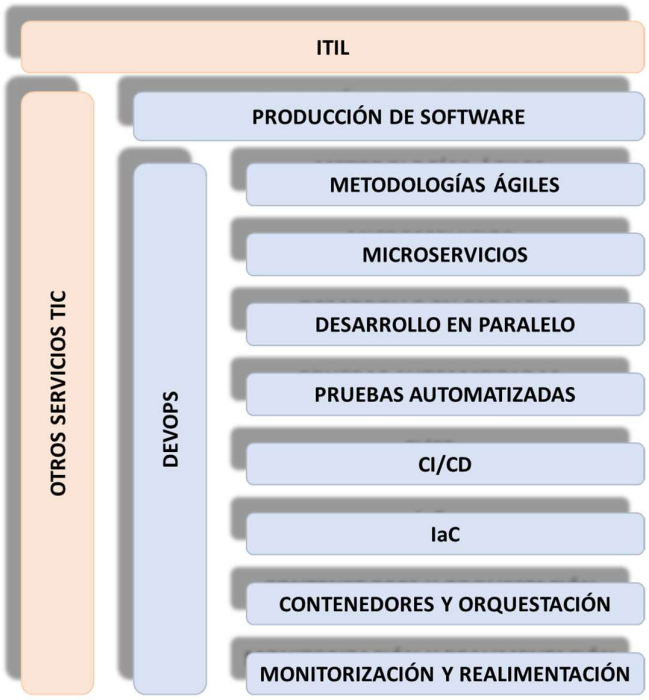

Figura 4-1 Modelo de prácticas y tecnologías DevOps para creación de software (elaboración propia)

En primer lugar, en un organismo tan complejo y extenso como el MINISDEF que necesita un elevado y muy diverso número de servicios TI, es esencial contar con un ITSM, apoyado por herramientas tecnológicas que automaticen sus procesos. ITIL es un buen marco de referencia, pero su utilización debe hacerse de forma que no suponga un impedimento para adoptar DevOps, y ya he expuesto que incluso su versión 4 parece alinearse aún más con los preceptos de esta cultura.

Las metodologías ágiles son el origen y el hilo conductor de DevOps. «Scrum» parece la más consolidada y con mayor aceptación en diversos organismos y ámbitos, aunque otros paradigmas, como por ejemplo «Kanban», puede adaptarse mejor a los procedimientos de trabajo a desarrollar por los equipos de operaciones.

Las arquitecturas con menor acoplamiento son las que presentan un impacto más positivo para favorecer el establecimiento de un flujo CI/CD y para ser empleadas en las tecnologías de virtualización

con contenedores. Con estas arquitecturas los equipos de desarrollo pueden realizar más fácilmente, a mayor velocidad y con menores riesgos, cambios en el código, pruebas y despliegues. También se reduce la dependencia de estos equipos del apoyo de otros equipos, servicios y recursos, y se evitan procesos de revisión y aprobación. No obstante, hay que tener en consideración que se va a necesitar alguna forma de orquestación de estos servicios a alto nivel para que funcionen como una aplicación. SOA es una buena aproximación, aunque los servicios pueden acabar presentando una alta dependencia del «Bus de Servicios» (ESB) que gestione la comunicación entre ellos y que suele terminar implementando gran parte de la lógica de negocio. La actual tendencia es a usar microservicios, que son los que tienen un grado de acoplamiento más bajo. Los microservicios representan, al incluir en puridad cada uno la información que maneja de forma privada, un cambio muy grande en cuanto a la visión tradicional de almacenamiento y administración de datos, por lo que escoger un patrón de gestión de datos con microservicios resulta fundamental para mantener la integridad y coherencia de la información.

El desarrollo en paralelo y concurrente realizado por varios programadores se ve favorecido por las arquitecturas anteriores y aumenta la velocidad y seguridad con la que se desarrolla y se modifica el código. Las herramientas tipo «Git» que se utilizan facilitan también la gestión de versiones y constituyen el primer elemento en la automatización de la cadena de entrega del software.

Las pruebas automatizadas y el conjunto de herramientas para CI/CD están en el núcleo del flujo continuo de despliegue automatizado que se pretende con DevOps, entregando software de calidad de una forma ágil.

Ya puse de manifiesto que el uso de la infraestructura como código (IaC) y la tecnología de contenedores favorece la automatización de los despliegues y permite que los equipos de desarrollo se provean de recursos al tiempo que se mejora la estabilidad de las infraestructuras TI, reduciendo los tiempos de puesta en explotación y la aparición de errores de configuración de estos recursos. Una vez que las aplicaciones están en el entorno de producción, la tecnología de contenedores y su orquestación facilita también que los recursos disponibles se usen de forma lo más eficiente posible a la vez que se mejora la escalabilidad de la aplicación y su robustez ante fallos.

Y, por último, hay que implementar herramientas para que los equipos de desarrollo y de operaciones puedan estar informados de cómo se comportan las aplicaciones y las infraestructuras para actuar en caso necesario. El propio usuario debería ser incluido como elemento del que recibir realimentación puesto que el objetivo fundamental de DevOps de producir software de forma ágil y segura tiene como meta el satisfacer los objetivos de la organización produciendo el valor buscado.

5 CONCLUSIONES Y LÍNEAS FUTURAS

5.1 La transformación digital y el desarrollo de software como servicio TI fundamental

El MINISDEF está empeñado en su transformación digital. Esta transformación, en el ámbito administrativo y funcional, busca equipararlo al resto de la AGE y presentarlo como un organismo moderno, evolucionado y eficaz, tanto ante la sociedad civil que entabla relaciones con él, como ante sus propios miembros. En el ámbito operativo se pretenden conseguir unas Fuerzas Armadas ágiles y decisivas, tecnológicamente avanzadas, capaces de obtener la superioridad de la información, acortando sus ciclos de decisión e incrementando el ritmo de sus acciones, ante todo tipo de amenazas y en los escenarios complejos, variados y cambiantes en que pueden desarrollarse las operaciones militares hoy en día.

Para alcanzar los objetivos de la transformación digital se han definido tres pilares fundamentales:

- La gestión por procesos. El paso de una organización basada en sistemas funcionales, que se desarrollan verticalmente creando estructuras funcionales con tendencia a funcionar de forma aislada, conforme a las teorías predominantes a finales del siglo pasado, a otra regida por procesos, en que la gestión se realiza de forma horizontal, involucrando a los departamentos necesarios y dándole mayor importancia a los indicadores para controlar y verificar que estos procesos se alinean con la estrategia de la organización y las necesidades de los usuarios.
- La importancia del dato y su gestión, adquiriendo consciencia de que, igual que en cualquier otra organización de nuestros días, la información es uno de los activos fundamentales, junto a los recursos humanos, para conseguir las metas marcadas y progresar alcanzando la superioridad en la información que le permita responder eficazmente a los retos y amenazas.
- El proveerse de servicios TIC adecuados como sustento de los dos puntos anteriores. Y en este sentido, el poder disponer de las aplicaciones específicas que en todo momento satisfagan las necesidades de los usuarios, dando el soporte adecuado y automatizando los procesos y la gestión de la información de la organización, es, en mi opinión, uno de los elementos esenciales para alcanzar el éxito en la transformación digital y constituye el fundamento del trabajo expuesto en esta memoria.

5.2 La cultura DevOps

El mundo actual surgido con el nuevo siglo está dominado por las TIC en todos sus ámbitos y se caracteriza por la rápida evolución y transformación en entornos cargados de incertidumbres. En este contexto podemos afirmar, sin riesgo a equivocarnos, que lo que verdaderamente valoran los clientes y usuarios de las aplicaciones que dan soporte a los procesos de las empresas y organizaciones hoy en día no es que el software específico satisfaga unos requerimientos iniciales, comportándose como un producto COTS, sino que sea una herramienta adaptable, capaz de satisfacer unos requisitos variables y volátiles que deben responder a un entorno cada vez más competitivo, indefinido y cambiante. Es decir, los usuarios demandan que las aplicaciones ofrezcan valor añadido constantemente y que, además, lo hagan de forma ágil y rápida.

Por tanto, todo sistema informático que pretenda ser una herramienta útil y práctica, orquestadora eficaz de los procesos desarrollados en la organización, debería tener como cualidad esencial el poder crearse y transformarse en cortos períodos de tiempo. De esta forma no solo se podrían alcanzar los objetivos marcados, satisfaciendo a los usuarios, sino que posibilitaría la obtención de la superioridad en el manejo de la información frente a competidores y adversarios.

Para poder lograr esta capacidad de producción y modificación rápida de aplicaciones, a principios de siglo aparecen las metodologías ágiles que empiezan a utilizar los equipos de desarrollo junto a

múltiples nuevas herramientas tecnológicas que favorecen su uso. Y es en ese proceso de intentar obtener software útil y de calidad a un ritmo elevado de producción cuando se detecta que las principales dificultades para poder llevarlo a la práctica se encuentran en la propia cadena de creación y puesta en explotación, concretamente en las relaciones entre los equipos de desarrollo y los de operaciones y en el paso de las aplicaciones desde los entornos de desarrollo a los de producción. Es aquí donde surge DevOps.

5.2.1 Qué es DevOps

DevOps es una cultura, un conjunto de hábitos y usos, una de forma de hacer las cosas y de entender el desarrollo y despliegue del software de manera que produzca valor a los usuarios, ya sean estos internos o externos a la organización. Con su origen en las metodologías ágiles, tiene como finalidad producir código de calidad y desplegarlo en los entornos de producción de forma rápida y segura. Su nombre deriva de los términos «development» y «operation» que hacen referencia a los equipos de desarrollo y a los de operaciones, dos de los conjuntos principales de profesionales participantes en el proceso de creación y puesta en explotación de los programas. Entre ellos suele existir tradicionalmente una barrera invisible que es identificada como el mayor problema para la producción y entrega rápida de software.

La implantación de DevOps supone el establecimiento de un ciclo continuo que comprende todas las fases del desarrollo de software (planificación, codificación, construcción, pruebas, entrega, despliegue, operación, monitorización y realimentación) y que se ejecuta hasta el final de la vida útil de las aplicaciones. Este ciclo se caracteriza por la ruptura de las barreras entre los participantes, acabando con los compartimentos estancos y facilitando los contactos, la colaboración y la puesta en común de conocimiento y experiencias; así como por la automatización de las tareas repetitivas que no generan valor y son fuente de errores, promoviendo el uso de herramientas tecnológicas de apoyo a lo largo de todo el ciclo que aceleran el proceso a la vez que incrementan la calidad y nivel de seguridad del código así como la estabilidad de la infraestructura.

Una organización que utiliza y ha madurado DevOps es capaz de generar y poner en explotación rápidamente código de calidad en condiciones seguras. Esto permite, por un lado, que los usuarios, internos o externos, vean satisfechas sus necesidades y se aumente la confianza en las TIC; y que, por otro, la organización se asegure de que el soporte principal para sus procesos y datos, las aplicaciones específicas, se adapte rápidamente y de forma flexible a las condiciones imperantes en cada momento, proporcionándole una ventaja competitiva frente a amenazas y adversarios y permitiéndole progresar de forma continua y acelerada.

5.2.2 Principales problemas en el MINISDEF frente a DevOps

El principal escollo es la casi total falta de relaciones entre los equipos de desarrollo y el de operaciones. La situación es igual a la descrita en la mayoría de documentación existente sobre el tema como origen de DevOps. Los equipos de desarrollo crean el software y lo «lanzan» al de operaciones para que lo ponga en producción dando por finalizado el proyecto y considerando que lo demás no es de su incumbencia a menos que haya que modificar el código por la aparición de errores.

No se han implantado modelos ágiles y los plazos de desarrollo o de mantenimiento evolutivo se miden en años y semestres. Esta falta de agilidad provoca que los usuarios vean las aplicaciones como recursos inmodificables que en función de cómo evolucionen sus propias necesidades se convierten más en una carga que en una herramienta eficaz de apoyo.

La seguridad de la información no se afronta desde las etapas iniciales en el desarrollo y acaba constituyendo un claro cuello de botella al ser necesario realizar auditorías complejas con carácter previo al pase a producción de las aplicaciones. Esto es un síntoma más de la existencia de barreras y de falta de conocimiento mutuo, de confianza y de trabajo colaborativo entre los distintos equipos.

No existen tecnologías en uso que faciliten a los equipos de desarrollo el auto aprovisionamiento y configuración de recursos de infraestructura, ni herramientas para automatizar la parte final del ciclo de desarrollo creando cadenas efectivas de integración y despliegues continuos (CI/CD).

No existe monitorización ni realimentación por parte de los equipos de desarrollo y el equipo de operaciones solo monitoriza determinados parámetros para asegurar la estabilidad y funcionamiento de las infraestructuras TIC.

Por último, aunque no menos importante, hay que destacar que el desarrollo de software se concibe como un proyecto, como una tarea que hay que completar generando un buen producto final, pero sin centrar el foco en la creación de valor para el usuario.

5.2.3 Cómo adoptar DevOps en el MINISDEF

No existe una única forma de implementar DevOps en una organización. Las acciones a realizar van a depender mucho de los objetivos que se pretendan alcanzar, de las capacidades ya presentes y de las que se identifiquen como necesarias, así como de la posibilidad de realizar inversiones y cambiar la estructura organizativa, lo que incluye la remodelación de los procesos, en caso necesario. También hay que tener claro desde el principio que la implantación de la cultura DevOps es un proceso complejo, que va a llevar tiempo y esfuerzo y que no se realiza como un proyecto sino como una suma de ellos con objetivos limitados. Quizás la mejor aproximación sea contar con un plan global, a grandes rasgos, con los objetivos que queremos alcanzar y la situación final deseada, para, a continuación, ir diseñando y ejecutando planes a corto plazo que nos permitan cumplir las metas finales.

Los ejes en los que debe centrarse la organización para implantar DevOps son:

- Realizar un esfuerzo en el cambio de mentalidad. Este debería impulsarse desde los puestos directivos y orientarse en la toma de conciencia de la importancia de producir y mantener aplicaciones de forma rápida y con calidad. También debería incidirse en el trabajo colaborativo y en la ruptura de barreras que suprima la existencia de compartimentos estancos en la organización. Por último, habría que resaltar que el objetivo final no se encuentra en la generación de buenos productos, sino en la satisfacción de las necesidades de los usuarios.
- Realizar un esfuerzo en la adquisición de herramientas que permitan la automatización de todo el ciclo DevOps. Este esfuerzo también conlleva un cambio en conceptos previamente asumidos, como, por ejemplo, que los equipos de desarrollo no participan en la configuración de la infraestructura o que los de operaciones no intervienen en el proceso de desarrollo. Así mismo hay que contemplar las necesidades de formación que las nuevas herramientas y procedimientos a aplicar impliquen.
- Realizar un esfuerzo en el cambio de modelos. Hay que adoptar modelos lógicos que se alineen con DevOps, como pueden ser las metodologías ágiles, las arquitecturas de muy bajo acoplamiento y el empleo de código legible y fácil de mantener sin exceso de documentación, pero también es recomendable utilizar modelos tecnológicos que faciliten la implantación de DevOps como pueden ser la virtualización con contenedores o la IaC.

5.2.4 Ventajas de la adopción de DevOps

La ventaja más evidente en la adopción de una cultura DevOps es el de producir valor para el cliente, que en el caso del MINISDEF es la propia organización, de forma ágil y continua. Lo importante no es que se disponga de buenos productos, sino que se va a poder poner a disposición del usuario de forma rápida el producto que necesita en cada momento.

Con la automatización y las nuevas tecnologías también se van a emplear de forma más eficiente tanto los recursos humanos especializados en TIC, que verán cómo aumenta su productividad, como los recursos materiales TIC, cuyas capacidades serán mejor aprovechadas y se incrementarán sus niveles de fiabilidad y estabilidad.

Con DevOps se consiguen la cooperación, comunicación y compartición de conocimientos y experiencias que hacen posible la mejora continua en el proceso de creación de software y, por extensión, en el conjunto de los procesos de la organización soportados por este software.

Y, por último, hay que señalar que, tal como se expuso en el capítulo inicial como objetivo de este trabajo, DevOps puede contribuir de forma muy significativa a mejorar los hábitos, costumbres, procedimientos y tecnologías que se utilizan en el desarrollo y puesta en explotación de software, facilitando que el MINISDEF alcance sus objetivos de transformación digital. El soporte de los procesos y de la gestión de la información del Ministerio se podrá realizar con aplicaciones de calidad que se adaptarán rápidamente a las necesidades que en cada momento requieran los usuarios en un mundo lleno de incertidumbres y en constante cambio.

6 BIBLIOGRAFÍA

[1] R. Fairly, Ingeniería del Software, México: McGraw-Hill, 1988.

[2] R. S. Pressman, Ingeniería del Software. un enfoque práctico (7ª edición), México: McGraw-Hill, 2010.

[3] *Apuntes de Ingeniería de Software, curso de Informática Militar,* 1999.

[4] «Ciclo de vida de los Sistemas de Información,» [En línea]. Available: http://itisistemasdeinformacion.blogspot.com/2015/09/. [Último acceso: 10 noviembre 2020].

[5] «Manifiesto por el Desarrollo Ágil del Software,» [En línea]. Available: https://agilemanifesto.org/iso/es/manifesto.html. [Último acceso: 16 noviembre 2020].

[6] M. Tena, «¿Qué es la metodología 'agile'?,» [En línea]. Available: https://www.bbva.com/es/metodologia-agile-la-revolucion-las-formas-trabajo/. [Último acceso: 16 noviembre 2020].

[7] «Programación extrema (wikipedia),» [En línea]. Available: https://es.wikipedia.org/wiki/Programaci%C3%B3n_extrema. [Último acceso: 10 noviembre 2020].

[8] «¿En qué consiste la metodología Kanban y cómo utilizarla?,» [En línea]. Available: https://www.apd.es/metodologia-kanban/. [Último acceso: 16 noviembre 2020].

[9] «14th annual State of Agile report,» [En línea]. Available: https://stateofagile.com/#ufh-i-615706098-14th-annual-state-of-agile-report/7027494. [Último acceso: 10 noviembre 2020].

[10] K. S. Rubin, Essencial Scrum. A practical guide to the most popular agile process, Ann Arbor: Pearson, 2013.

[11] Y. Muradas, «Conoce las 3 metodologías ágiles más usadas,» 8 marzo 2018. [En línea]. Available: https://openwebinars.net/blog/conoce-las-3-metodologias-agiles-mas-usadas/. [Último acceso: 10 noviembre 2020].

[12] R. Lynn, «Lean Methodology,» [En línea]. Available: https://www.planview.com/resources/articles/lean-methodology/. [Último acceso: 8 diciembre 2020].

[13] M. J. Martín, «¿Qué es DevOps? Aclarando el significado,» [En línea]. Available: https://profile.es/blog/que-es-devops-harder-better-faster-stronger/. [Último acceso: 26 noviembre 2020].

[14] P. Debois, «Agile Infraestructure & Operations,» [En línea]. Available: http://www.jedi.be/presentations/agile-infrastructure-agile-2008.pdf. [Último acceso: 26 noviembre 2020].

[15] M. Hüttermann, DevOps fpr Developers, Apress, 2012.

[16] «DevOpsDays,» [En línea]. Available: https://devopsdays.org/about. [Último acceso: 26 noviembre 2020].

[17] «DevOps (wikipedia),» [En línea]. Available: https://es.wikipedia.org/wiki/DevOps. [Último acceso: 26 noviembre 2020].

[18] CA Technologies, «What Smart Businesses Know About DevOps,» septiembre 2013. [En línea]. Available: https://dsimg.ubm-us.net/envelope/331183/290922/1404151341_techinsights-research-what-smart-businesses-know-about-devops.pdf. [Último acceso: 10 diciembre 2020].

[19] D. A. Marcos, «Todos para uno y agile para todos,» julio 2019. [En línea]. Available: https://www.tendencias.kpmg.es/2019/07/metodologia-agile/. [Último acceso: 10 diciembre 2020].

[20] A. Hevia, «Diferencias entre SOA y Microservicios,» 2018. [En línea]. Available: https://andreshevia.com/2018/04/22/diferencias-entre-soa-y-microservicios/. [Último acceso: 10 diciembre 2020].

[21] A. Brown, M. Stahnke y N. Kersten, «2020 State of DevOps Report (Puppet),» [En línea]. Available: https://puppet.com/resources/report/2018-state-devops-report/. [Último acceso: 9 diciembre 2020].

[22] N. Forsgren, D. Smith, J. Humble y J. Frazelle, «Accelerate State of DevOps 2019,» [En línea]. Available: https://services.google.com/fh/files/misc/state-of-devops-2019.pdf. [Último acceso: 9 diciembre 2020].

[23] J. Serrano, «Integración, entrega y despliegue continuo. Diferencias y similitudes,» 25 febrero 2019. [En línea]. Available: https://geeks.ms/jorge/2019/02/25/integracion-entrega-y-despliegue-continuo-diferencias-y-similitudes/. [Último acceso: 10 diciembre 2020].

[24] Puppet, «A Decade of DevOps,» 2020. [En línea]. Available: https://puppet.com/devops/state-of-devops-report-retrospective/. [Último acceso: 12 diciembre 2020].

[25] G. Kim, K. Behr y G. Spafford, The Phoenix Project, Portland: IT Revolution Press, 2013.

[26] «Introducción a DevOps. Qué es y cómo implementarlo,» [En línea]. Available: https://tech.tribalyte.eu/blog-introduccion-devops. [Último acceso: 19 noviembre 2020].

[27] GitKraken, «DevOps Tools Report 2020,» 2020. [En línea]. Available: https://www.gitkraken.com/resources/devops-report-2020. [Último acceso: 13 diciembre 2020].

[28] J. Davis y K. Daniels, Effective DevOps, Sebastopol, California: O'Reilly, 2016.

[29] «Agencia Española para la Calidad,» [En línea]. Available: https://www.aec.es/web/guest/centro-conocimiento/gestion-tic. [Último acceso: 12 octubre 2020].

[30] «ITIL Glossary v01, 1 May 2006: Acronyms,» 1 mayo 2006. [En línea]. Available: https://web.archive.org/web/20110218082813/http://www.itil-officialsite.com/nmsruntime/saveasdialog.aspx?lID=925&sID=242. [Último acceso: 9 noviembre 2020].

[31] «ITIL (wikipedia),» [En línea]. Available: https://es.wikipedia.org/wiki/Information_Technology_Infrastructure_Library. [Último acceso: 09 noviembre 2020].

[32] Freshworks, «How ITIL 4 incorporated practices of Agile, Lean, and DevOps,» [En línea]. Available: https://freshservice.com/how-itil-4-incorporated-practices-of-agile-lean-and-devops-blog/. [Último acceso: 19 diciembre 2020].

[33] S. Rance, «The 7 Guiding Principles of ITIL 4,» marzo 2019. [En línea]. Available: https://www.sysaid.com/blog/entry/the-7-guiding-principles-of-itil-4-practical-advice-to-help-you-make-decisions. [Último acceso: 19 diciembre 2020].

[34] Freshworks, «ITIL V4,» [En línea]. Available: https://freshservice.com/es/itil/itil-v4/. [Último acceso: 19 diciembre 2020].

[35] «Plan de Transformación Digital de la Administración General del Estado y sus Organismos Públicos (Estrategia TIC 2015-2020),» [En línea]. Available: https://administracionelectronica.gob.es/pae_Home/pae_Estrategias/Estrategia-TIC-AGE.html#.X-DYJBZ7mUk. [Último acceso: 3 noviembre 2020].

[36] Ejército de Tierra, «Resumen ejecutivo 'FUERZA 35',» [En línea]. Available: https://ejercito.defensa.gob.es/estructura/briex_2035/resumen_ejecutivo_fuerza_35.html. [Último acceso: 21 diciembre 2020].

[37] MINISDEF, *Instrucción 58/2016, Arquitectura Global de Sistemas y Tecnologías de Información y Comunicaciones del MINISDEF,* 2016.

[38] F. J. Arroyo, Interviewee, *Entrevista con equipo de desarrollo.* [Entrevista]. 29 octubre 2020.

[39] A. Rodríguez Jiménez, Interviewee, *Entrevista con equipo de operaciones.* [Entrevista]. 17 noviembre 2020.

[40] Real Academia Española, «Diccionario de la lengua española, actualización 2019,» [En línea]. Available: https://dle.rae.es/. [Último acceso: 12 octubre 2020].

[41] R. F. Oltra Bardenes, «Sistemas de Información, un factor estratégico para la gestión de empresas,» [En línea]. Available: https://riunet.upv.es/bitstream/handle/10251/84467/Oltra%20-%20Los%20Sistemas%20de%20Informaci%C3%B3n.%20Un%20factor%20estrat%C3%A9gico%20para%20las%20empresas.pdf?sequence=1. [Último acceso: 10 noviembre 2020].

[42] MINISDEF, *Orden DEF/2639/2015, Política de los Sistemas y Tecnologías de la Información y las Comunicaciones del MINISDEF,* 2015.

[43] SEDEF, *Instrucción 25/2018, Plan de Acción del MINISDEF para la Transformación Digital (1ª parte),* 2018.

[44] SEDEF, *Instrucción 14/2020, Plan de Acción del MINISDEF para la Transformación Digital (2ª parte),* 2020.

[45] «Crisis del software (wikipedia),» [En línea]. Available: https://es.wikipedia.org/wiki/Crisis_del_software. [Último acceso: 14 octubre 20201].

[46] J. Medina, «Los origenes del Just-in-Time,» [En línea]. Available: https://blog.toyota-forklifts.es/origenes-just-in-time. [Último acceso: 9 diciembre 2020].

[47] C. Gallego Rodríguez, F. Álvarez y M. Evgeniev, «Serverless,» 11 abril 2019. [En línea]. Available: https://www.bbva.com/es/serverless/. [Último acceso: 10 diciembre 2020].

[48] Red Hat, «¿Qué son los microservicios?,» [En línea]. Available: https://www.redhat.com/es/topics/microservices/what-are-microservices. [Último acceso: 10 diciembre 2020].

[49] Trend Micro, «¿Qué es la infraestructura como código?,» [En línea]. Available: https://www.trendmicro.com/es_es/what-is/cloud-security/infrastructure-as-code.html. [Último acceso: 13 diciembre 2020].

[50] J. Geerling, Ansible for DevOps, Midwestern Mac, LLC, 2017.

[51] R. Half, «6 Basic SDLC Methodologies: Which One is Best?,» mayo 2019. [En línea]. Available: https://www.roberthalf.com/blog/salaries-and-skills/6-basic-sdlc-methodologies-which-one-is-best. [Último acceso: 20 diciembre 2020].

[52] Ionos, «Docker Compose y Docker Swarm: orquestación de contenedores Docker,» julio 2019. [En línea]. Available: https://www.ionos.es/digitalguide/servidores/know-how/docker-compose-y-swarm-gestion-multicontenedor/. [Último acceso: 20 diciembre 2020].

[53] «Los diferentes tipos de testing en el desarrollo de software,» [En línea]. Available: https://programacionymas.com/blog/tipos-de-testing-en-desarrollo-de-software. [Último acceso: 28 diciembre 2020].

[54] El Mundo, «La guerra moderna se basa en la superioridad del dato,» 2017. [En línea]. Available: https://www.elmundo.es/economia/2017/03/21/58d0f6fa46163f814c8b458e.html. [Último acceso: 30/12/2020 diciembre 2020].

ANEXO I: CUESTIONARIO PARA ENTREVISTA CON EL EQUIPO DE DESARROLLO

El 26 de octubre de 2020 me entrevisté con representantes de los equipos de desarrollo de la Jefatura de los Sistemas de Información, Telecomunicaciones y Asistencia Técnica (JCISAT) del Ejército de Tierra. El conjunto de cuestiones que configuraron la entrevista se incluye en este anexo. Para preparar los asuntos a discutir utilicé como esquema básico el ciclo de vida iterativo para desarrollo de software de DevOps que se puede ver en la Figura A1-0-1.

Figura A1-0-1 Ciclo de vida iterativo DevOps (tomada de [26])

FASE 1: PLANIFICACIÓN

1. ¿Qué metodología de desarrollo se sigue? (En cascada, en espiral, ágil…)
2. ¿Esta metodología se usa tanto para el desarrollo inicial como para cualquier cambio?
3. ¿Cuáles son las etapas del flujo de entrega de la aplicación y quién interviene en cada una de ellas? (desarrollo, pruebas, puesta en explotación, operación/administración y mantenimiento)
4. ¿Se mantienen reuniones/contactos periódicos con el usuario? ¿En qué momentos del desarrollo inicial o del desarrollo causado por un cambio se producen esos contactos? ¿De qué se trata en ellas con el usuario?
5. ¿Se mantienen reuniones/contactos periódicos con otro personal implicado en el despliegue de la aplicación como personal de operaciones, de seguridad, de calidad del software, de pruebas? ¿En qué momentos del desarrollo inicial o del desarrollo causado por un cambio se producen esos contactos? ¿De qué se trata en ellas?
6. ¿Existen restricciones previas al desarrollo impuestas por el personal de operaciones, seguridad, etc.?
7. ¿Qué tiempo estimado llevó o se prevé que lleve el desarrollo inicial?
8. ¿Existe una vía ágil y práctica para que el usuario comunique necesidades de cambios o fallos?

FASES 2-3-4: CODIFICACIÓN, CONSTRUCCIÓN Y PRUEBAS

9. ¿Qué tecnologías se emplean para programar? (lenguaje de programación, BBDD, dockers)
10. ¿Qué arquitectura tiene la aplicación? (cliente-servidor, multicapa, servicios o microservicios web)
11. ¿Qué número de personal interviene en el desarrollo y cuál es su rol? (ingenieros de software, diseñadores, programadores, probadores, aseguramiento de la calidad…)

12. ¿Qué tecnologías se emplean para apoyo al desarrollo y entrega? (repositorios de software, pruebas automatizadas, herramientas de integración y despliegue continuos CI/CD, servicios en la nube, dockers, kubernetes)
13. ¿Cómo se realizan la compilación, pruebas unitarias, revisión de calidad del código y detección de vulnerabilidades?
14. ¿Qué documentación se mantiene de la aplicación y cómo se guarda? (repositorios de documentación)
15. ¿Cómo se realiza el control y gestión de versiones de la aplicación y la documentación asociada?
16. ¿Qué infraestructura TI necesita la aplicación? (número y tipo de servidores, requisitos para los servidores o las máquinas de los usuarios, requisitos de red)

FASE 5-6: ENTREGA Y DESPLIEGUE

17. ¿Qué entornos de despliegue se utilizan desde el desarrollo hasta la puesta en explotación? (desarrollo, pruebas, integración, explotación)
18. ¿Cómo se realiza la transición entre entornos y qué herramientas se utilizan?
19. ¿Qué personal es el responsable de cada entorno?
20. ¿Cuáles son las vías de comunicación entre los diferentes responsables?
21. ¿Qué tiempo se tarda desde que se decide hacer un cambio (por fallo o nuevo requisito) hasta que la aplicación modificada está disponible para el usuario?
22. ¿Ante un fallo de la aplicación que imposibilite su uso, cuál es el tiempo estimado para que vuelva a estar disponible?
23. ¿Cada cuánto hay que hacer una entrega de nuevo software por modificaciones causadas por fallos, mejoras o nuevos requisitos?
24. ¿Qué porcentaje de esas modificaciones realizadas resultan fallidas inicialmente y requieren de otra modificación o vuelta atrás?
25. ¿Cuál suele ser la causa de modificaciones fallidas? (fallos durante las pruebas, fallos en la puesta en explotación)
26. En caso de fallos de la aplicación en la puesta en explotación o durante su operación ¿Cuál suele ser el motivo o causa que da el equipo de operaciones de estos fallos?

FASES 7-8: OPERACIÓN, MONITORIZACIÓN Y REALIMENTACIÓN

27. ¿Se tienen indicadores del desempeño de la aplicación? (volumen de datos almacenado, volumen del flujo de datos, % de crecimiento de datos periódico, número de accesos a la aplicación por horas/días de la semana, empleo de recursos hardware como nº de procesadores/servidores…)
28. ¿Se tiene información de variaciones periódicas en la demanda de uso de la aplicación? (diferencia entre horario de trabajo o no, periodos de vacaciones, etc.) ¿La aplicación está preparada para absorber estas variaciones? (uso de balanceadores de carga, creación y eliminación dinámica de instancias)
29. ¿Se realiza alguna acción para comprobar la satisfacción del cliente? (encuestas, cuestionarios)
30. ¿Existe alguna forma de comunicación ágil entre todos los actores (desarrolladores, usuarios, equipo de operaciones…) y un procedimiento reglado de comunicación?

CICLO EN SU CONJUNTO

31. Cuáles son los principales problemas/inconvenientes que se detectan por parte del equipo de desarrollo, y cómo cree que se podrían solventar, en los siguientes aspectos:
    a. Relaciones con el usuario de la aplicación
    b. Relaciones con los equipos que realizan la compilación, integración y pruebas (caso de que existan)
    c. Relaciones con los equipos de operaciones (administradores del entorno hardware)

d. Relaciones con los equipos de seguridad de la información
e. Relaciones con los equipos de aseguramiento de la calidad
f. Relaciones con los administradores de la aplicación

ANEXO II: CUESTIONARIO PARA ENTREVISTA CON EL EQUIPO DE OPERACIONES

El 16 de noviembre de 2020 me entrevisté con el jefe del equipo de operaciones que proporciona servicios de infraestructuras TI a todos los órganos del MINISDEF, encuadrado en la División de Operaciones y Redes (DIVOPER) del Centro de Sistemas y Tecnologías de la Información y las Comunicaciones (CESTIC). El conjunto de cuestiones que configuraron la entrevista se incluye en este anexo. Para preparar los asuntos a discutir utilicé como esquema básico el ciclo de vida iterativo para desarrollo de software de DevOps que se puede ver en la Figura A2-0-1

Figura A2-0-1 Ciclo de vida iterativo DevOps (tomada de [26])

FASE 1: PLANIFICACIÓN

1. ¿Cuál es el perfil profesional del personal de operaciones? (militares o civiles, qué tipo de ingeniería o formación profesional)
2. ¿Se utiliza alguna metodología de gestión de servicios TI? (ITIL, COBIT)
3. ¿Existe un «Service Level Agreement» (SLA) entre el CESTIC y los ejércitos/órgano central que defina qué servicios proporciona CESTIC, con qué calidad, disponibilidad y los criterios de seguridad, asistencia y mantenimiento?
4. ¿Intervienen de alguna forma en la fase de planificación con los equipos de desarrollo (en la recogida de requisitos, análisis y diseño de los programas)?
5. ¿Cómo conoce el equipo de operaciones que se va a desarrollar una aplicación que van a tener que poner en explotación y quién autoriza esa puesta en explotación?
6. ¿Se marcan restricciones o se dan indicaciones a los equipos de desarrollo sobre cuál debería ser la infraestructura a utilizar o la arquitectura a seguir de acuerdo con las posibilidades de la infraestructura existente?
7. ¿En qué etapas del flujo de entrega de la aplicación interviene el equipo de operaciones y cuál es su cometido básico en cada una de ellas? (desarrollo, pruebas, puesta en explotación, operación/administración y mantenimiento)
8. ¿Se mantienen reuniones/contactos periódicos con el equipo de desarrollo? ¿En qué momentos del desarrollo inicial o del desarrollo causado por un cambio se producen esos contactos? ¿De qué se suele trata en ellas?
9. ¿Se mantienen reuniones/contactos periódicos con otro personal implicado en el despliegue de la aplicación como personal de seguridad, de calidad del software, de pruebas? ¿En qué

momentos del desarrollo inicial o del desarrollo causado por un cambio se producen esos contactos? ¿De qué se trata en ellas?
10. ¿Existen requisitos previos al desarrollo o a la puesta en explotación impuestos por el personal de desarrollo, seguridad, calidad del software, etc.? ¿Existe la posibilidad de modificar la infraestructura para atender estos requisitos?
11. ¿Existe una vía ágil y práctica para que el equipo de operaciones comunique con el equipo de desarrollo necesidades de cambios debido a fallos?

FASES 2-3-4: CODIFICACIÓN, CONSTRUCCIÓN Y PRUEBAS

12. ¿Cuál es la infraestructura disponible para la puesta en explotación de las aplicaciones? (CPD, nube, nodos locales, infraestructura externalizada…)
13. ¿Cuáles son las principales tecnologías que se emplean en la infraestructura para puesta en explotación de aplicaciones?
    a. Tipos de servidores web o de aplicaciones y sistemas operativos empleados.
    b. Tipos de BBDD y gestores de BBDD.
    c. Tipo de virtualización, balanceadores de carga, containers, sistemas de orquestación
    d. Sistemas de monitorización de la infraestructura, sistemas de seguridad -IDS, IPS, firewalls, etc.-
    e. Sistemas de almacenamiento de datos -SAN, cabinas de discos, tipo de RAID-
    f. Otras tecnologías destacables.
14. ¿Cuáles son las arquitecturas que emplean las aplicaciones? (cliente-servidor, multicapa, servicios o microservicios web)
15. ¿Qué número de personal interviene en operaciones y cuál es su rol? (ingenieros de sistema, programadores, operadores, supervisores de seguridad…)
16. ¿Qué tecnologías se emplean para apoyo al desarrollo y entrega? (repositorios de software, pruebas automatizadas, herramientas de integración y despliegue continuos CI/CD, servicios en la nube, dockers, kubernetes…)
17. ¿Cómo se realizan la compilación, pruebas unitarias, pruebas funcionales, revisión de calidad del código, auditorías de seguridad y detección de vulnerabilidades?
18. ¿Se exige algún tipo de documentación para la puesta en explotación?
19. ¿Cómo se realiza el control y gestión de versiones de las aplicaciones?
20. ¿Cómo se determina qué infraestructura TI necesita la aplicación? (número y tipo de servidores, requisitos para los servidores o las máquinas de los usuarios, requisitos para copias de seguridad, requisitos de red…)

FASE 5-6-7: DESPLIEGUE

21. ¿Cuál es el proceso genérico de puesta en explotación de un programa? (Cómo llega desde los equipos de desarrollo y hasta que está en explotación)
22. ¿Qué entornos de despliegue se utilizan desde el desarrollo hasta la puesta en explotación? (desarrollo, pruebas, integración, explotación)
23. ¿Qué tareas fundamentales se realizan en cada uno de estos entornos?
24. ¿Qué volumen de aplicaciones se encuentran en cada entorno y qué volumen aproximado de usuarios y datos tiene cada una?
25. ¿Cuál es el ritmo aproximado de llegada de nuevas aplicaciones a cada entorno?
26. ¿Cómo se realiza la transición entre entornos y qué herramientas se utilizan?
27. ¿Qué personal es el responsable de cada entorno?
28. ¿Cuáles son las vías de comunicación entre los diferentes responsables?
29. ¿Ante un fallo de la aplicación que imposibilite su uso, cuál es el tiempo estimado para que vuelva a estar disponible? (en caso de fallo del hardware o software de la infraestructura, o en caso de fallo de la propia aplicación)

30. ¿Cada cuánto suelen recibirse entregas de nuevo software por modificaciones causadas por fallos, mejoras o nuevos requisitos de aplicaciones ya en explotación?
31. ¿Qué porcentaje de esas modificaciones realizadas resultan fallidas inicialmente y requieren de otra modificación o vuelta atrás?
32. ¿Cuál suele ser la causa de modificaciones fallidas? (fallos durante las pruebas, fallos en la puesta en explotación)
33. En caso de fallos de la aplicación en la puesta en explotación o durante su operación ¿Cuál suele ser el motivo o causa que da el equipo de operaciones de estos fallos?

FASES 8-9: OPERACIÓN Y REALIMENTACIÓN

34. ¿Se tienen indicadores del desempeño de la aplicación? (volumen de datos almacenado, volumen del flujo de datos, % de crecimiento de datos periódico, número de accesos a la aplicación por horas/días de la semana, empleo de recursos hardware como nº de procesadores/servidores…)
35. ¿Se tiene información de variaciones periódicas en la demanda de uso de la aplicación? (diferencia entre horario de trabajo o no, periodos de vacaciones, etc.) ¿La infraestructura y el diseño de las aplicaciones están preparados para absorber estas variaciones? (uso de balanceadores de carga, creación y eliminación dinámica de instancias de contenedores…)
36. ¿Existe una política única para copias de seguridad o depende de cada aplicación?
37. ¿En la gestión de copias de seguridad intervienen los equipos de desarrollo?
38. ¿Se realiza alguna acción para comprobar la satisfacción del cliente? (encuestas, cuestionarios)
39. ¿Existe alguna forma de comunicación ágil entre todos los actores (desarrolladores, equipo de operaciones, usuarios …) y un procedimiento reglado de comunicación?

CICLO EN SU CONJUNTO

40. Cuáles son los principales problemas/inconvenientes que se detectan por parte del equipo de operaciones, y cómo se podrían solventar, en los siguientes aspectos:
    a. Relaciones con los equipos de desarrollo (responsables técnicos)
    b. Relaciones con el usuario de la aplicación
    c. Relaciones con los administradores (responsables funcionales)
    d. Relaciones con los equipos de seguridad de la información
    e. Relaciones con los equipos de aseguramiento de la calidad