Softtek®

Digital Product Lifecycle

Índice

Haz **clic** en el **título** para volver al índice en cualquier momento o en un **bloque** para acceder directamente.

07 - 20

Continuous Product Design

21 - 35

Continuous Product Development

36 - 48

Continuous Product Delivery

EDITORIAL

La visión holística del diseño de producto

David Alejano Hernández

Digital Strategy & Experience Director

Softtek EMEA

Para las compañías era vital la transformación digital, con el objetivo de ofrecer mejores productos y servicios, así como de optimizar sus procesos internos. Pero actualmente, dado el momento en el que nos encontramos y debido a la transición acelerada que se ha experimentado hacia el mundo digital, esto ya no es suficiente. Se ha **acelerado la necesidad de una redefinición de los modelos de negocio** actuales y la aparición de otros nuevos que están cambiando la forma en la que se entrega valor a los usuarios.

Estos nuevos modelos requieren **ofrecer una experiencia omnicanal e híbrida, uniendo el mundo físico y digital.** La clave para conquistar a unos usuarios cada vez más demandantes es generar experiencias asombrosas que aumenten el "engagement" con las compañías, esta es la razón de la creciente relevancia de los productos digitales.

Los productos digitales nacen con el objetivo de responder a los problemas y necesidades de los usuarios. Para poder generar un producto digital exitoso tenemos que estar en **constante búsqueda y entendimiento de los problemas y las necesidades reales que los usuarios.** Debemos ir más allá de la tradicional segmentación de usuarios o del comportamiento de compra o de uso de los usuarios; tenemos que entender la motivación de estos y que es lo que realmente quieren resolver.

Dividimos el flujo de generación de productos digitales en tres ciclos que ocurren de forma continua y simultánea. El primer ciclo, **Continuous Product Design**, está enfocado en entender los problemas reales y la exploración de posibles soluciones que nos ayuden a resolverlos.

El primer paso de este ciclo es Hipotetizar. Es una actividad que generalmente se usa cuando tenemos nociones de grandes problemáticas y queremos refinarlas para poder **acotar mejor los problemas y las posibles soluciones.**

En el siguiente paso, Investiga y Entiende, profundizamos en entender el problema, realizando investigaciones de mercado, arquetipos y generando como resultado las **motivaciones reales de los usuarios expresadas en Jobs.**

A continuación, trabajamos en la Exploración de **soluciones de forma divergente para después converger** en una de ellas, expresándola mediante wireframes navegables que faciliten la conceptualización. En este paso también es necesario revisar la viabilidad técnica. Estas actividades son clave en el diseño de nuestro producto y por lo tanto conllevarán una gran parte del esfuerzo necesario.

A partir de aquí pasaríamos a diseñar gráficamente la solución y generar un **prototipo navegable que nos permita validar con usuarios** finales si la solución buscada es la adecuada. Si la solución parece ser correcta y resuelve el problema, pasaríamos al siguiente ciclo de generación de productos. En caso contrario, seguiríamos iterando hasta encontrar la solución correcta.

El segundo ciclo, **Continuous Product Development**, está enfocado en la implementación de la solución y en el trabajo integrado de todos los equipos necesarios para hacerla realidad. Está basado en el uso de las mejores prácticas de desarrollo, con arquitecturas ligeras y asegurando la calidad a todos los niveles.

Este ciclo nos ayuda a mejorar la calidad, reduciendo el riesgo y estableciendo un **ritmo de desarrollo rápido, fiable y sostenible** generando software potencialmente desplegable en cualquier momento.

El último ciclo, **Continuous Product Delivery**, está enfocado en la entrega de las soluciones y el aprendizaje a través del comportamiento de uso. Es importante diferenciar el despliegue en producción, que debe realizarse de forma continua, a la apertura de las soluciones a los usuarios finales. Esta apertura se realizará cuando el equipo de negocio considere que es el mejor momento para hacerlo.

La solución debe poder ser entregada para todos o solo un grupo de usuarios. También, nos debe permitir tener varias soluciones a la vez que puedan ser usadas por diferentes grupos de usuarios. El objetivo principal es **aprender de la solución y de los datos de comportamiento** sobre la misma. Este último ciclo es una de las entradas de "insights" para el primer ciclo de creación de producto.

El éxito de un producto digital parte de un profundo entendimiento de los problemas que tiene que resolver el usuario y de la búsqueda y validación de soluciones. A su vez, es igual de importante ser capaz de implementar una solución viable y entregarla en el menor tiempo posible. Por lo tanto, podemos decir, que un problema no está resuelto hasta que la solución esta entregada y usada satisfactoriamente por los usuarios finales y el **único modo de lograrlo es tener una visión holística, manteniendo un flujo continuo** entre los distintos ciclos de creación de producto.

Durante la última década las metodologías como Agile o DevOps se han popularizado con gran fuerza, dado que impulsan la capacidad de las organizaciones digitales para escalar el desarrollo y entrega de software continuo en el mercado.

Actualmente, la creación de productos digitales es una de las principales tendencias de mercado a nivel global, y está captando un nivel de inversión sin precedentes. Pero, a pesar de ello, son muchos los productos que no alcanzan la meta establecida o, en ocasiones, implican pérdidas de beneficios o de clientes después de un pico inicial exitoso.

Para la creación del producto **es importante seguir un proceso** y, generalmente se puede dividir en tres ciclos, que adicionalmente se componen de varias etapas que ocurren simultáneamente con el objetivo de descubrir las necesidades de cliente y las posibles soluciones:

CONTINUOUS PRODUCT DESIGN:

*En esta primera etapa, el objetivo es **comprender las necesidades de los usuarios, definir una solución y decidir qué producto** se va a desarrollar, a partir del presupuesto de cada empresa. Así, los factores anteriores definirán el periodo de tiempo que englobará la entrega continua.*

1. Hipotetiza.
2. Investiga y Entiende.
3. Explora e Innova.
4. Diseña.

CONTINUOUS PRODUCT DEVELOPMENT:

*Construcción de soluciones atendiendo a una **experiencia de usuario perfecta, arquitecturas apropiadas** y de alta calidad.*

5. Sintetiza.
6. Construye.
7. Integra.
8. Testea.

CONTINUOUS PRODUCT DELIVERY:

*Gestión de lanzamiento sin fricción (stores, entornos Cloud) y **aprendizaje a través del análisis** de datos de comportamiento.*

9. Despliega.
10. Responde y Recupera.
11. Monitoriza y Mide.
12. Aprende.

01

Continuous Product Design

1. Hipotetiza

A partir del conocimiento de mercado se tienen nociones generales de los problemas y necesidades a solucionar. Este paso nos ayuda a refinar y entender esos problemas y las posibles soluciones a alto nivel, generando como resultado una hipótesis inicial que nos guiará en la creación de los productos digitales.

El **objetivo** es refinar grandes problemas y generar hipótesis iniciales que nos ayuden a priorizar como se van a abordar las soluciones y que en los siguientes pasos del ciclo, se desglosarán y detallarán.

Metodología

DESIGN SPRINT

Se enfoca en ayudar al trabajo conjunto de los equipos en la resolución de un problema concreto, dando como resultado una entrega de soluciones para que sean probadas por los usuarios.

Esta metodología es la aceleración y simplificación del proceso de diseño de un producto a través de la co-creación. De tal forma, se obtienen resultados medibles, que permiten la validación de las soluciones a través del testeo por parte de los futuros usuarios.

En cinco días esta metodología ayuda a:

- Entender y planificar el problema dentro del área elegida.
- Idear las soluciones adecuadas.
- Llevar a cabo la toma de decisiones, convirtiendo las ideas en hipótesis probables.
- Realizar un prototipo realista.
- Pruebas con usuarios reales en vivo.

LEARN STARTUP

Se busca acortar ciclos de desarrollo de productos mediante la **combinación de experimentación impulsada por hipótesis** para medir el progreso, lanzamientos de productos iterativos y aprendizaje validado para medir cuánto se ha aprendido.

Crear productos en un periodo de tiempo corto, definiendo rápidamente si un negocio es viable o no. Las compañías que hagan uso de esta metodología serán capaces de centrarse en el **desarrollo de su producto, a la vez que reciben un feedback de su cliente** y, para ello, es necesario lanzar un producto mínimo viable al mercado (MVP).

El objetivo es poder evitar grandes inversiones de dinero durante las etapas iniciales del desarrollo de un producto, generando la oportunidad de éxito a largo plazo. Las compañías deberán centrarse en obtener la opinión de sus clientes desde el inicio, de manera que esta **retroalimentación ayude a realizar los cambios** oportunos para mejorar en el tiempo.

Por un lado, se debe seleccionar a un pequeño subconjunto de usuarios en una variedad de datos demográficos que puedan usar el producto. Por otro lado, se debe medir el alcance del producto, es decir, se deben **medir de manera efectiva los resultados del producto mínimo viable** mientras que se continúa desarrollando el producto con los comentarios del cliente.

En caso de que el MVP no esté obteniendo buenos resultados con los clientes a los que se le proporcionó, deshacerse del producto básico debería ser sencillo al no haberse utilizado demasiados recursos.

Un pilar fundamental es el aprendizaje, ya que aprender de manera efectiva de los resultados de las pruebas debe permitir el desarrollo de un producto satisfactorio acorde a las necesidades del cliente.

INNOVATION ACCOUNTING

Se utiliza para la puesta en marcha de Lean Startup, creando métricas que surgen a través de la participación del usuario, del ajuste del producto al mercado y de la escalabilidad midiendo el progreso en diferentes niveles:

- **Nivel 1:** mediante métricas simples se realiza un seguimiento del progreso a lo largo del tiempo, dando una idea básica de lo que funciona.
- **Nivel 2:** representa la interacción completa con un cliente e incluye variables de etapas posteriores.
- **Nivel 3:** una vez interiorizado el aprendizaje se generan beneficios económicos que permiten volver a ejecutar el proyecto completo.

Output

Hipótesis inicial: como resultado obtendremos una hipótesis inicial de cliente, de problema y de solución, la cual nos ayudará a refinar y entender mejor los problemas a trabajar. Las hipótesis iniciales nos ayudaran a priorizar entre ellas y nos servirá de base para comenzar el trabajo de diseño de producto.

Este paso es necesario cuando no tenemos clara la priorización y queremos refinar para ver por donde continuar, o porque queremos tener una visión global del problema y de las posibles soluciones aún sin estar trabajadas en detalle. Esta hipótesis se irá refinando en siguientes pasos hasta convertirse en MVP.

2. Investiga y Entiende

Para concretar las necesidades y encontrar las soluciones más adecuadas buscaremos entender las necesidades reales de los clientes identificando los tipos de personas (modelo Persona), cómo piensan (Empathy Maps) y cómo se relacionan con la empresa (Customer Journey).

El **objetivo** es trabajar con múltiples partes interesadas a través del Design Thinking para entender las necesidades de los usuarios finales.

Metodología

GEMBA WALKS

Las entrevistas y Gemba Walks son técnicas que permiten entender cómo una persona se relaciona e interactúa con un producto, comprender los procesos existentes de desarrollo y adquirir una visión en detalle de las dinámicas del sector. Estas técnicas permiten acompañar en todo momento al target establecido de ayuda para mejorar los puntos clave del modelo de negocio mediante las conclusiones obtenidas, generando datos e información para el diseño específico de un producto adaptado a necesidades reales del cliente.

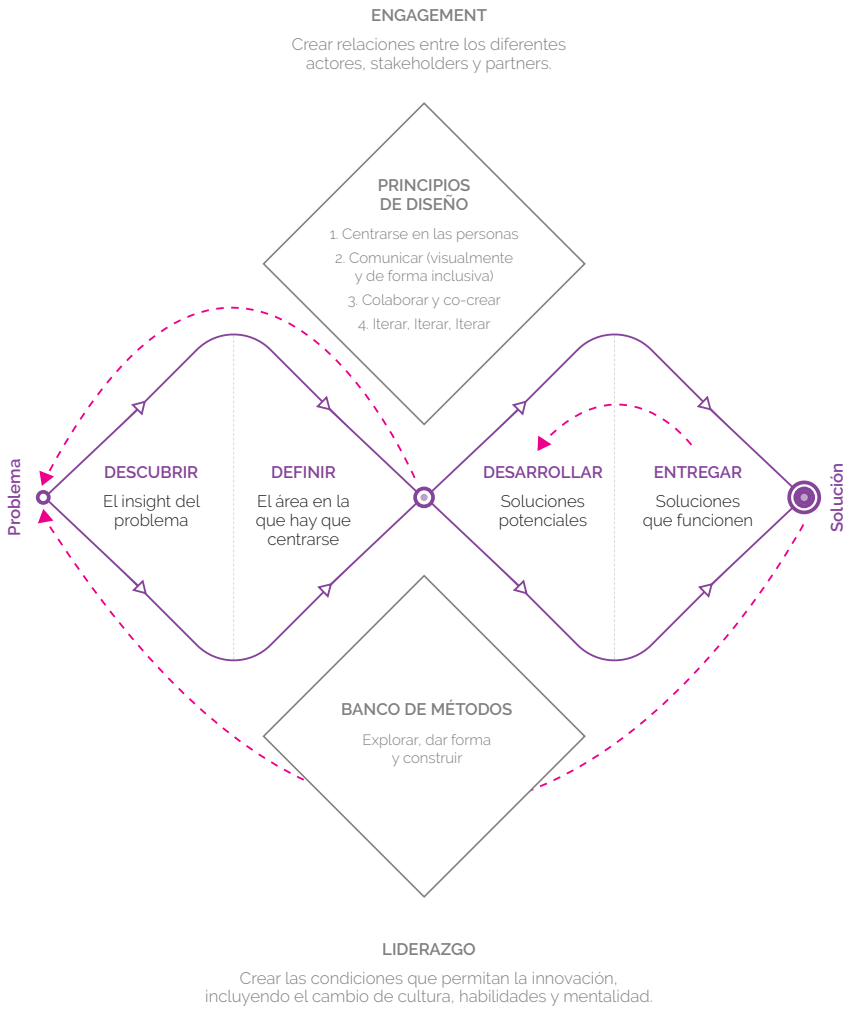
Hay dos enfoques que se pueden aplicar al Gemba Walk, por un lado, el **enfoque efectivo** permite realizar observación de los usuarios y tomar nota de los procesos que se están realizando, obtener información de los usuarios y aprender de ella, y por último, consultar qué enfoques y métodos funcionan bien y cuáles necesitan mejorar. Por otro lado, el **enfoque inefectivo** hace suposiciones o recopila información mientras se encuesta en una sala, juzga el producto y piensa qué se debe corregir en todo lo que está mal, encuentra fallos en lugar de observar el proceso y proporciona una opinión personal, mostrando sesgos.

DOUBLE DIAMOND

La técnica Double Diamond se **encarga de mapear las etapas divergentes y convergentes** de un proceso de diseño: describe los modos de pensar que utilizan los desarrolladores, siendo este un proceso lineal, donde se plantea un diseño inicial significativo antes de su producción. De este modo, las soluciones se perfeccionan antes del lanzamiento pudiendo conocer de antemano su eficacia en un mercado real.

En el caso de los productos digitales es posible **perfeccionar la solución antes de su comercialización** mediante la creación de paquetes de valor más pequeños para su lanzamiento. Según el aprendizaje obtenido, se pueden brindar soluciones en un mercado real y adaptar la arquitectura del producto.

La solución inicial aporta información sobre el futuro y la estrategia se puede adaptar permitiendo la optimización de productos y mejorar en los flujos de trabajo.



ESTUDIOS DE MERCADO

Tanto primarios como secundarios ya que, generalmente, para la venta de cualquier producto hay que desarrollar un **estudio de mercado tanto primario como secundario.** La investigación de mercado primaria la realiza la propia compañía.

En el caso de la investigación de mercados secundarios, se realiza por otras personas, donde se incluyen estudios realizados por investigadores o datos publicados por otras empresas.

QUALITATIVE RESEARCH

Un proceso de investigación cualitativa de clientes correctamente organizado genera resultados válidos, exactos, fiables y completos. De este modo, se reflejan las necesidades de los clientes que ayudarán a mejorar el producto.

Un ejemplo de este tipo de investigación es el Focus Group, consistente en introducir a un número de personas en una sala y grabarlas manteniendo una conversación sobre el tema establecido. En algunos casos, se obtendrán nuevas ideas para los productos y, en otros, se obtendrán diferentes puntos de vista sobre el mismo. A partir de esta situación, se **obtienen insights para manejar la información** obtenida. Normalmente, los participantes son seleccionados a partir de diferentes criterios como la demografía, los intereses o la ocupación.

JOBS TO BE DONE

Se trata de una metodología para comprender mejor el comportamiento de los consumidores a partir de un cambio de perspectiva, buscando que las acciones de marketing se vuelvan más eficientes y que la innovación sea más rentable.

Así pues, las empresas deben identificar cuáles son los factores que motivan al usuario y qué productos y servicios pueden ayudar a resolver estos factores. Mediante esta técnica, una compañía **maximiza las posibilidades de desarrollar una solución exitosa** en el mercado. Aporta los siguientes beneficios:

- Definir claramente los mercados en torno al producto que se va a realizar.
- Brindar mejoras diarias a los clientes en sus necesidades.
- Aumentar el valor del producto según aumenta el número de ayudas que le ofrece al cliente.
- Desarrollo de un negocio capaz de adaptarse a las nuevas necesidades de los clientes.
- Mayor eficacia en la obtención de beneficio: las personas están dispuestas a pagar más para alcanzar su meta de una manera más eficaz.
- Ahorro en inversiones de productos I+D que no darán resultados.
- Obtener predicciones certeras sobre qué productos triunfarán en el futuro.

Output

Definición de los trabajos (Jobs to be Done): a través de esta metodología se definen las tareas que realiza el cliente, mediante la adquisición de un producto en función de sus necesidades. Así pues, identificar esas tareas es fundamental para crear experiencias de marketing adaptadas al usuario y, en este aspecto, la comunicación con el cliente es vital.

Arquetipos (Persona, Mapa de Empatía y Journey Map): consiste en extraer información de diversas entrevistas para generar un agregado de datos, donde cada usuario es una persona con una historia basada en esos mismos datos.

Se define como "Persona" las **imágenes ficticias de clientes reales que detallan aspectos y rasgos** de una persona real que puede usar o comprar un producto. Generalmente, son representaciones de necesidades y deseos de grupos de personas que comparten rasgos comunes como el *Marketing Persona* y *Persona UX*.

En consecuencia, la figura de Persona ayuda a **unificar equipos en torno a una comprensión compartida** de definición de los usuarios reales. Además, esta figura es de naturaleza prescriptiva e incentiva una mejor toma de decisiones tácticas y operativas, pero, a largo plazo, tienen menor contribución de éxito, ya que no ofrecen información sobre clientes futuros ni hablan de las aspiraciones de estos.

Los arquetipos hacen **referencia a una parte de la mente humana que contiene un inconsciente colectivo** compartido por todos los miembros de la especie, una especie de memoria universal y primordial. Así pues, los arquetipos aprovechan este inconsciente colectivo y la memoria primaria para darle sentido al mundo que rodea al ser humano. Pueden ser aplicados tanto a clientes como a situaciones.

La diferencia que existe entre arquetipos y Personas fundamentalmente es que los primeros son patrones comunes en historias que son fáciles de identificar y relacionar. Sin embargo, no son prescriptivos y no ofrecen detalles específicos del cliente, pero ayudan a impulsar modelos aspiracionales mediante la creatividad. Según la teoría narrativa existen doce arquetipos clásicos, cada uno con sus propias fortalezas, motivaciones y debilidades.

En conclusión, las Personas ayudan a comprender quiénes son los clientes y los arquetipos muestran quiénes quieren ser los clientes, por lo tanto, adoptar ambas perspectivas ayuda a impulsar la visión estratégica de la compañía.

3. Explora e Innova

Para brindar una solución innovadora y disruptiva definiremos junto al usuario la solución en base a su problema o necesidad, apoyándonos en las tecnologías de vanguardia.

El **objetivo** es obtener las soluciones adecuadas para cada una de las necesidades.

Metodología

CO-CREACIÓN

La co-creación es la **colaboración entre socios estratégicos** para la creación y entrega de productos centrados en el cliente buscando más valor, de una forma más rápida y que minimice el riesgo respecto a las metodologías más tradicionales.

Así pues, las asociaciones estratégicas y centros conjuntos de innovación proporcionan una situación de mayor control, reducción del coste y del tiempo en el desarrollo de un producto.

VISUAL THINKING

La metodología Visual Thinking busca la transmisión de ideas complejas mediante un mecanismo más simple a través del **uso de gráficos e imágenes.** Se trata de trasladar las ideas en forma de gráficos y diagramas para hacer más simple la transmisión, tratándolas como un concepto más universal y entendible.

La visualización de ideas ayuda a los equipos a **comprender más fácilmente cuáles son los puntos claves** del desarrollo de un producto. Además, con esta técnica, los diferentes equipos pueden colaborar y aportar nuevas ideas a través de otros diagramas e imágenes.

El Visual Thinking ayuda a identificar problemas, **patrones y nuevos enfoques de desarrollo** de producto, facilitando la materialización de ideas que, sin su aplicación, no serían posibles.

LEAN UX

El proceso de creación mediante Lean UX se inicia a partir de supuestos y, a raíz de estos, se **generan hipótesis que deben ser validadas.** Una vez estén validadas las hipótesis, se evalúan los resultados para analizar si son los esperados. Esta metodología se diseñó para la creación de productos que sean capaces de satisfacer las necesidades de los usuarios, agilizando el tiempo de producción y alcanzando la meta establecida en menos tiempo.

De esta técnica destaca la **reducción de tiempos de producción**, lo que involucra la disminución del riesgo en la comercialización. Además, la inversión de tiempo se concentra exclusivamente en el desarrollo del producto. Se trata de una metodología colaborativa y dinámica que trata de maximizar la interacción entre los diferentes equipos y los usuarios.

Asimismo, esta metodología permite conocer las opiniones de los usuarios previamente, durante y después del proceso de producción, mejorando el rendimiento y optimizando el proceso teniendo en cuenta las opiniones de los usuarios.

INTENTIONAL ARCHITECTURE

Este tipo de arquitectura se encarga de definir un conjunto de iniciativas planificadas con el **propósito de mejorar el diseño y el rendimiento del desarrollo** de un producto. Mediante esta técnica, se proporciona un mejor esquema para la planificación entre equipos de los pasos adecuados para la implementación.

Se trata de un enfoque muy estructurado de iniciativas arquitectónicas definidas y bien planificadas. La clave es identificar correctamente las limitaciones, ya que esto proporciona un mayor nivel de éxito y, en consecuencia, la entrega de valor aumenta.

Output

Arquitectura de información transversal: el resultado de esta arquitectura es la **simplificación de procesos de comprensión de los datos** y de las tareas a realizar por los usuarios de los productos desarrollados.

Se trata de un proceso iterativo y transversal que tiene lugar a lo largo del diseño del producto, asegurando que se cumplan los objetivos de la producción y del desarrollo del producto de manera efectiva. De este modo, se conseguirá que la actuación del usuario sea más efectiva y que el producto sea más accesible.

Matriz de funcionalidades: consiste en especificar las características del sistema que cada usuario exigirá para las diferentes tareas a realizar. De este modo, se pueden identificar los pasos más críticos y así, la compañía puede centrarse en cada tarea o paso durante el proceso de diseño.

Generalmente, la creación de matrices de funcionalidades se realiza en las primeras etapas para especificar las necesidades de los usuarios, siendo especialmente útil en los productos que tienen un número elevado de funcionalidades.

Para la creación de estas matrices es necesario reunir a un grupo de usuarios, designar tareas en estos grupos, y definir las funciones y características que se introducirán en la matriz. Según avance el proceso de desarrollo del producto, se podrán ir introduciendo nuevas funciones y características, al mismo tiempo que se eliminarán las funciones innecesarias.

Requisitos no funcionales: son aquellos que se emplean para juzgar las operaciones de un proceso. Describen las características y limitaciones que debe tener un sistema para alcanzar el éxito teniendo en cuenta el rendimiento, tiempo de entrega y seguridad.

Wireframe: son montajes a escala de diseño de un producto sobre una representación de los elementos donde será aplicado, lo que dará al cliente una visión más realista de cómo trabajará con el producto.

4. Diseña

Para generar la mejor experiencia de uso de la solución idearemos una arquitectura de información transversal, crearemos los flujos de navegación y diseñaremos la interacción y la interfaz visual. Buscaremos la optimización de los ratios de conversión testeando diferentes tipos de implantación con usuarios finales.

El **objetivo** son activos de diseño que conformarán la UX de la App y los elementos tecnológicos para llevarla a cabo.

Metodología

ATOMIC DESIGN

Esta metodología **minimiza las inconsistencias y optimiza el diseño** y el desarrollo de productos digitales, evita los errores del proceso y facilita el desarrollo. Es una técnica que se encuentra en la intersección entre el diseño y el desarrollo de producto, encargada del diseño de componentes desde su estado más elemental hasta su incrementación máxima, y puede dividirse en cinco niveles:

- **Nivel atómico**: está compuesto por los elementos UI que individualmente tienen su propia funcionalidad.
- **Nivel molecular**: consiste en la unión de diferentes elementos del primer nivel para formar un elemento UI más complejo.
- **Nivel de los organismos**: hace referencia a elementos más complejos representados por la suma de los niveles anteriores.
- **Nivel plantillas**: wireframes, la unión de diferentes organismos que forman una aplicación.
- **Nivel páginas**: es el diseño final, es decir, el archivo ya preparado para el desarrollo o pruebas con usuarios.

VALIDACIÓN CON USUARIOS

Con esta metodología, se pueden recoger las **necesidades y los problemas que los usuarios ven en el diseño** de un producto. Por ejemplo, los usuarios pueden usar prototipos y la compañía debe observar las reacciones, conociendo poco a poco los problemas con los que se encuentran. Se puede recoger información del producto antes de que salga al mercado y corregir los aspectos que no satisfagan al cliente.

A diferencia de la etapa de descubrimiento continuo, aquí no se está evaluando al usuario, sino que se está **probando el sistema y las propiedades** del producto.

El usuario recibe tareas a realizar y se le entrega el prototipo que empleará, dando feedback a la empresa.

EXPLORING

Es un boceto que involucra una descripción en detalle, diseñado para la representación visual y esquemática de la estructura de un producto digital. Con esto, se busca definir el contenido y la posición de las diversas características que lo constituyen. No obstante, el diseño no es lo importante, sino que se debe priorizar el contenido.

AGILE MODELING

Trata de la práctica para el modelado y la documentación efectiva de sistemas basados en software. **Los pilares sobre los que se fundamenta son la comunicación, la simplicidad y la retroalimentación,** de tal manera que se busca aplicar prácticas eficientes durante el proceso de desarrollo satisfaciendo las necesidades del equipo y de las partes interesadas en el producto para generar una visión completa.

EMERGENT DESIGN

Consiste en entregar funcionalidades de manera cronológica, adoptando e implementando las mejores prácticas de cada parte de la funcionalidad entregada. De este modo, la entrega de valor es continua y, al final del diseño, el producto está compuesto por el conjunto de las partes más útiles de cada funcionalidad. **El resultado es un diseño simple con una base de código pequeña.**

Output

System Design: infraestructura que garantiza la calidad del diseño y la eficiencia del desarrollo de los productos digitales, que agrupa los elementos de prototipado y desarrollo del producto. Es un **conjunto de entregables reutilizables en diferentes combinaciones** que irán evolucionando con el producto y las nuevas tecnologías que se le apliquen. Está compuesto por tokens de diseño que definen la morfología de las características, mediante bibliotecas de estilos, la documentación necesaria sobre las consideraciones de diseño y un lenguaje común para impulsar la comunicación entre desarrolladores y diseñadores.

Diseño Técnico (resultado de Agile Modeling): donde se deciden las opciones técnicas adaptadas para llevar a cabo las funcionalidades diseñadas.

Prototipo: la creación de prototipos es un método de investigación de nuevos conceptos, que busca **comprender la usabilidad del producto y mejorar su funcionalidad.** Actualmente, existen multitud de herramientas para crear muestras con enfoque propio.

Crear prototipos es fundamental dentro del proceso de diseño para poder conocer si el producto funciona o no, junto con las mejoras que se deben realizar para alcanzar la meta buscada. Los prototipos **no son vinculantes y pueden utilizarse durante todo el proceso** de diseño, ya sea para actualizar las mejoras del producto o iterar sobre un diseño ya existente.

02

Continuous Product Development

5. Sintetiza

Para establecer un roadmap del producto y comenzar a implementar la solución estableceremos el MVP, las MMF,y el Backlog de historias.

El **objetivo** es sintetizar las hipótesis, la investigación, la definición y el diseño en una visión, un roadmap y un backlog.

Metodología

FEATURE WRITING

Consiste en **dividir las características del producto en historias de usuarios** durante las fases de planificación y desarrollo. De esta forma, se conseguirá que el producto a desarrollar pueda ser el elegido.

BDD

Behavior-Driven Development (BDD) es una **estrategia de desarrollo basada en el comportamiento**, que presenta un idioma común para todos los stakeholders, mejorando la comunicación entre los diferentes equipos. Este enfoque de prueba deriva de la metodología de desarrollo impulsada por pruebas (TDD), pero en este caso las pruebas se basan en el comportamiento de los sistemas.

Entre sus beneficios destacan el **alcance que tiene para llegar a un mayor público** mediante el uso del lenguaje no técnico y definir cómo debe comportarse el sistema empleando una perspectiva de cliente.

Se lleva a cabo aplicando el principio de los "cinco por qué" a las historias de usuario propuestas, implementando los comportamientos que contribuyan directamente a los resultados comerciales, simplificando la forma de expresar estos comportamientos con el fin de mejorar la comunicación y aplicar estas técnicas hasta los niveles más bajos de abstracción.

WSJF

La técnica Weighted Shortest Job First (WSJF), es una metodología para priorizar y programar el trabajo. Su objetivo es generar el **máximo beneficio económico cuando se aporta el máximo valor**, de forma que se actualicen de manera automática las prioridades en función del usuario y del negocio, teniendo en cuenta los factores de riesgo, de tiempo, etc.

La metodología de obtener el máximo beneficio sigue con la **disminución de los costes de demora y la optimización del valor añadido** al final de cada iteración. Además, la técnica puede resultar muy útil si se detectan dos características o funciones iguales, porque ayuda a la toma de decisión de cuál debería llevarse primero a cabo en función del valor y el tiempo de desarrollo.

WSJF = Cost of Delay / Job Duration (Job size)

Output

Story maps: trata de ordenar las historias de los usuarios en diferentes dimensiones, de tal manera que el mapa organice las actividades del usuario en un orden de prioridad representada en el eje horizontal, mientras que en el eje vertical se representa la sofisticación de la implementación.

Gracias al desarrollo de estos mapas, en caso de existir un fallo su impacto es minimizado, ya que en caso de que una de sus características quede obsoleta, no afecta al producto en su totalidad. Funcionalmente estas características aisladas son de menor valor y, por tanto, no tienen necesariamente que diferir para futuras versiones.

Características con hipótesis de beneficios y criterios de aceptación: son definidos por el equipo de desarrollo y describen los requisitos que el producto debe cumplir para estar terminado. Suelen estar **relacionados con las historias de usuario o Feature Writing**, de manera que permite al equipo de desarrollo adoptar un nuevo enfoque hacia el cliente.

Priorizar el Backlog de historias de usuario: mediante diversas herramientas se deben **seleccionar las historias de usuario de mayor valor para el equipo de desarrollo**, con el objetivo de introducir mejoras, cambios y correcciones, para que pueda asignárseles un orden y sean seleccionadas en el siguiente sprint.

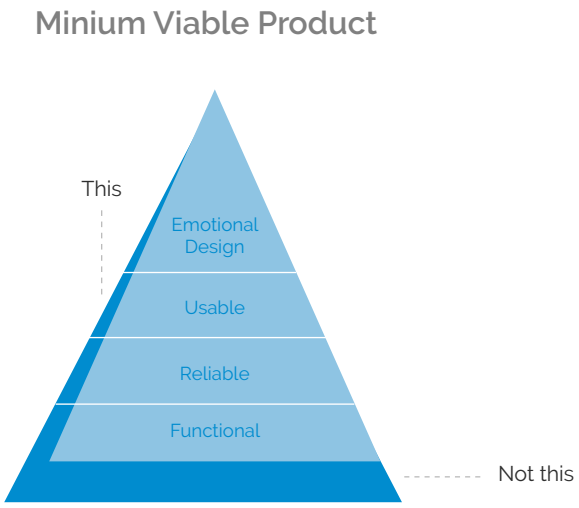
Roadmap: es una hoja de ruta o lista ordenada de tareas y mejoras para la implementación del proceso de desarrollo establecido por los equipos de trabajo que se irán implementando a lo largo del tiempo.

Se visualizan las **acciones del producto en relación con los diferentes puntos de interacción** del usuario con un producto para tener una visión global y rápida del plan de desarrollo del producto.

MVP e Indicadores de aprendizaje: una vez se haya aprobado la hipótesis mediante la creación de prototipos y la obtención de una prueba de concepto, se creará un MVP. Inicialmente, se creará la versión más sencilla del producto que, posteriormente se mejorará, y se compartirá con la mayor cantidad de personas posible.

Un MVP **permite la comprensión total del producto, los usuarios que hacen uso de él y los requisitos mínimos** necesarios para alcanzar la meta establecida. Entre los beneficios de desarrollarlo se encuentran:

- Ganan todas las partes interesadas.
- Ayuda a la evolución del producto.
- Minimiza los costes de desarrollo.
- Acelera la obtención del producto final para su comercialización.



MMF'S: la característica comercial mínima (MMF) es la unidad más pequeña de funcionalidad del producto con valor de mercado intrínseco. Un caso de MMF es el lanzamiento de un producto inicial básico, al cual de forma gradual se le añaden nuevas características adicionales para descubrir cuál de estas características son poco empleadas. Esta metodología permite centrarse en las funciones de alto valor, reduciendo el tiempo de comercialización y aumentando el ROI.

MMF es **capaz de generar una metodología de gestión del trabajo en función de la cantidad de valor** proporcionado.

Las compañías deben buscar un MMF de función pequeña e independiente que se pueda desarrollar rápidamente, brindando un valor significativo al cliente.

Los MMFs se consideran la **mejor unidad de planificación para los lanzamientos**, sobre todo cuando se necesitan múltiples iteraciones para construir partes de una nueva característica, pero es necesario retrasar su entrega hasta que pueda combinarse en un todo conceptual viable.

6. Construye

Para crear una solución viable, factible y sostenible utilizaremos las tecnologías nativas, híbridas o PWA y las arquitecturas y entornos (On Premise, Cloud, Multicloud) más adecuadas en base al enfoque visual, cargabilidad, uso y otros factores funcionales o técnicos.

El **objetivo** es implementar una story o parte de una story y subir el código.

Metodología

CONTROL DE VERSIONES

Hace referencia a la gestión de los cambios que se realiza sobre los elementos del producto desarrollado o la configuración de este, es decir, es la revisión del producto en su estado de desarrollo o modificación.

ENGINEERING PRACTICES

Esta práctica, define las **actividades de la ingeniería que garantizan la calidad** de los productos, asegurándose que el desarrollo es el establecido a partir de la generación de entregables que aseguren los requisitos de validación.

Por tanto, evalúan que la calidad, las características del diseño, la mano de obra y el proceso, sigan y cumplan las condiciones establecidas, mediante un conjunto de criterios aceptados y revisados.

PAIR WORK

Método que hace referencia a que siempre debe haber dos personas trabajando al mismo tiempo en el código. Generalmente, **una de las personas se encarga de escribir el código y la otra debe supervisarlo en tiempo real**, así pueden encontrar soluciones y desarrollar ideas creativas mediante el debate de problemas. Sin embargo, el rol de estas dos personas debe intercambiarse frecuentemente para que no exista una brecha jerárquica. Cada una de las personas trabajará con su propio espacio de trabajo en el que se mostrará la misma información que dispone su compañero.

MODELADO DE AMENAZAS

Proceso estructurado para **identificar las posibles vulnerabilidades de seguridad, conocer su gravedad** y seleccionar las técnicas adecuadas para controlar y proteger los recursos. Las características principales de un modelo de amenazas son que éste es sistemático y estructurado, siguiendo una serie de pasos concretos para comprender el sitio que quieren proteger. Se deben seguir al menos cuatro pasos principales:

1. Descomponer la aplicación.
2. Determinar las amenazas.
3. Determinar las medidas.
4. Clasificar las amenazas.

TEST-DRIVEN DEVELOPMENT

Técnica de ingeniería de software para diseñar productos digitales que funciona mediante la **escritura de pruebas unitarias.** Con esta técnica se obtiene un código más tolerante al cambio, más seguro y resistente. Además, según se empieza a aplicar aparece una mayor velocidad de desarrollo.

La metodología funciona a partir de tres pilares fundamentales:

- No se escribe un código de producción sin antes escribir el código de fallo.
- No se escribe más de un texto unitario que dé lugar a fallo.
- No se debe escribir más código del necesario para realizar la prueba.

APLICACIÓN DE TELEMETRÍA

Se encargan de la **medición automatizada y recolección de datos.** La aplicación suele representar una inversión significativa, pero es la forma idónea para registrar datos y, posteriormente, transformarlos en un análisis de alto valor para la toma de decisiones.

Output

Implementación del código: una vez configurado el producto, se debe llevar a cabo la implementación del código. Para ello, existen diferentes plataformas y metodologías para este entregable.

Pruebas unitarias: es una forma de comprobar que un fragmento de código funciona de la manera adecuada. Generalmente, se trata de aislar una parte de código y comprobar su funcionamiento, mediante tests que validan la lógica del código. Gracias a ellas, se detectan errores antes de realizar pruebas más avanzadas.

Las pruebas unitarias aportan legibilidad al código y **ayudan a mejorar el código base**, afianzan la lógica del código, se realizan en poco tiempo y, posteriormente, servirán como documentación del proyecto.

7. Integra

Con el propósito de agilizar el desarrollo y reducir los costes de construcción, se fomentará la reutilización y componentización del código favoreciendo así la modularidad e interoperabilidad entre las distintas plataformas.

El **objetivo** es compilar los archivos fuente en binarios desplegables, verificar que el código funciona como los desarrolladores pretendían, fusionar las ramas de desarrollo con el tronco y garantizar la integración con terceros.

Metodología

INTEGRACIÓN CONTINUA

La integración continua del código se **emplea para combinar los cambios en el código en un repositorio central de manera periódica**, de modo que así se permite la ejecución de varias versiones y pruebas automáticas.

Al igual que otras metodologías anteriores, se emplea para encontrar errores con mayor agilidad, mejorando la calidad del producto y reduciendo el tiempo que se tarda en validar dentro de la fase de integración.

Usando esta metodología, se envían los cambios con las actualizaciones de forma periódica a un repositorio compartido en un sistema de control. Antes del envío, se ejecutan pruebas unitarias locales en el código como medida de verificación antes de la integración.

Así pues, esta técnica se encarga de **crear y ejecutar automáticamente pruebas unitarias en los nuevos cambios** realizados en el código, y así poder identificar errores.

AUTOMATIZACIÓN

La automatización de la construcción y los tests es la práctica que facilita el control de la ejecución de un producto digital de manera automática, generándose una comparación entre los resultados obtenidos y los resultados esperados.

Normalmente, este proceso emplea pruebas repetitivas y ejecuciones de manera automática, lo que **complementa a las pruebas que se realizan de forma manual**, liberando de carga de trabajo a diversos empleados.

Los beneficios de esta metodología son: una elevada fiabilidad que elimina el error humano, la rapidez, y el permitir la repetición de ejecuciones de la misma operación. Estas pruebas son programables y se pueden reutilizar los scripts empleados para las pruebas.

SEGURIDAD

Integrar un alto nivel de seguridad en el desarrollo de software genera **aplicaciones de calidad y robustas a un coste menor.** Cuando se habla de seguridad en esta fase, se hace referencia a una serie de actividades durante la etapa de implementación sobre el diseño y la codificación del producto.

TRUNK-BASED DEVELOPMENT

Es una técnica de gestión de control de versiones en la que los empleados se encargan de combinar actualizaciones frecuentes y de corta duración en una rama principal del código. Su **función principal es acelerar las fases de fusión e integración.** Trunk-Based Development es especialmente útil cuando aumenta la complejidad del código base, ayudando a mantener el flujo de las versiones de producción. Además, es una metodología simple porque se centra en las correcciones y actualizaciones, asumiendo que la rama central del código siempre es estable y no presenta problemas.

Las principales ventajas de esta metodología son la facilitación de integración de código, la **garantía de una revisión continua del código y la facilidad de lanzamientos consecutivos** de códigos de producción.

GATED COMMIT

Esta metodología consiste en generar un **patrón de integración de software capaz de evitar el fallo de una compilación** al realizar cambios en la rama principal del control de versiones mediante una prueba de simulación. Por tanto, este patrón debe de ser compatible con los servidores de integración continua.

Normalmente, para llevar a cabo un Gated Commit, los desarrolladores tienen que realizar cambios en una rama central. Un servidor se encargará de implantar los cambios locales y de realizar las validaciones en la simulación. Así, el desarrollador podrá ser capaz de comprobar si los cambios realizados hacen que falle la compilación sin haber realizado cambios reales.

Output

Código implementado integrado: tras realizar las etapas anteriores, llegados a este punto, se obtiene finalmente **un código implementado e integrado** listo para su producción.

Prueba de integración: consiste en pruebas avanzadas para verificar que todos los elementos que componen un software funcionen juntos correctamente. Estas pruebas se realizan una vez se hayan aprobado todas las pruebas unitarias para **centrarse en la comunicación entre los diferentes elementos.**

Así, se consigue verificar la correcta construcción de los diversos componentes que construyen un código, una vez las pruebas unitarias son válidas para comprobar que la comunicación entre sus interfaces es correcta, que t**odos los elementos en conjunto realizan las funciones establecidas** y que no existen errores globales.

8. Testea

Para asegurar la robustez de las aplicaciones implementadas verificaremos primero la calidad del código y generaremos pruebas automatizadas para validar la aplicación de forma continua (pruebas de unidad, integración y end2end).

El **objetivo** es validar los cambios con respecto a los criterios de aceptación en un entorno de producción integrado y simulado y, en un segundo paso, alojar el sistema totalmente validado en un entorno de producción desde el que se puedan implementar.

Metodología

AUTOMATION TESTS

Permite realizar pruebas de manera automática y repetitiva, mediante la implementación automática en secciones estables que no serán modificadas a corto plazo, pero sí que requieren que no existan fallos.

Así pues, se **detectan de una manera más ágil los errores y se optimiza el tiempo** para solventarlos, ahorrando costes. Además, esta metodología permite optimizar el tiempo de los empleados liberándoles de cargas de trabajo. Este es un método recomendado cuando el producto es estable, y la compañía se quiere centrar en la CX para lanzar su producto al mercado.

BLUE/GREEN DEPLOYMENT

Este es un modelo de aplicaciones que trabaja transfiriendo lentamente el tráfico de usuarios de una etapa anterior a una etapa siguiente. A la etapa pasada se le conoce como entorno azul y a la nueva etapa se le conoce como entorno verde.

Blue/Green Deployment es una **forma de implementación automatizada que traslada el software desde la etapa final de prueba a la producción** en vivo. Este proceso tiene que ocurrir lo más rápido posible para minimizar el tiempo de inactividad, facilitando esta transición y garantizando que el producto tenga dos entornos lo más idénticos posibles.

Además, esta técnica **hace más sencillo el retroceso** en caso de que exista algún fallo y, por lo tanto, se pueda volver a la etapa anterior. El entorno azul y verde deben ser diferentes pero lo más idénticos posibles, o bien puede ser un único entorno operativo dividido en zonas separadas. Una vez se pasa al verde, el azul se puede utilizar como zona de ensayo para próximas actualizaciones o emplearse como plantilla.

CONGRUENCIAS

Para comprobar la calidad auténtica del producto es necesario garantizar entornos congruentes de prueba y producción: mediante este método se reduce la posibilidad de fallo y se genera confianza en el producto. Además, buscando la congruencia se **garantiza que los desarrolladores comprendan la infraestructura** que existe detrás de sus aplicaciones.

TEST DATA MANAGEMENT

Permite seleccionar y eliminar de forma automática grupos de datos de pruebas extraídos de la producción, de tal forma que los **datos sensibles se protegen con el formato correcto** en un tiempo mínimo.

El fin principal de este test es gestionar los códigos fuentes de una aplicación brindándoles el tratamiento adecuado para realizar pruebas.

SERVICE VIRTUALIZATION

La virtualización de servicios se encarga de añadir valor diferencial al modelo de negocio, acelerando el proceso de desarrollo y reduciendo su coste mediante la simplificación del producto para entornos de prueba.

Haciendo uso de este método se **reduce el tiempo y la complejidad de las pruebas** gracias a la capacidad de utilizar servicios de simulación en lugar de servicios reales.

REQUISITOS NO FUNCIONALES

Los requisitos no funcionales son los que se utilizan para **evaluar las operaciones de proceso**, describiendo las características y limitaciones del proceso.

Output

Código desplegado y validado: una vez testeado el código o producto digital, y tras obtener unos resultados positivos de la prueba, se obtiene el código desplegado y validado que se puede producir de forma masiva, preparado para su comercialización.

Pruebas funcionales y de aceptación: la realización de pruebas funcionales **facilita que existan menos reprocesos y que aumente la velocidad** de desarrollo mediante la comunicación efectiva de los requerimientos. Llegados a este punto en el proceso de la creación de un producto, las pruebas de aceptación componen la última fase del proceso de testing.

Las pruebas de aceptación las realizan los usuarios reales del software para poder verificar que se cumpla con las tareas requeridas en las situaciones reales.

Generalmente, este tipo de pruebas se realizan en entornos intermedios y, a continuación, se entrega la versión beta a los clientes, buscando recoger información sobre detalles y comentarios realizados por medio de encuestas.

03

Continuous Product Delivery

9. Despliega

Para minimizar los errores en el proceso de desligue, automatizaremos los pipelines de entrega y usaremos técnicas de apertura (Feature Toggle, Canary Releases, etc.) de funcionalidades a usuarios.

El **objetivo** es implementar los cambios de producción con alta frecuencia y bajo riesgo.

Metodología

CANARY RELEASES

Esta técnica consiste en reducir el riesgo de introducir una nueva versión en la producción, tratando de implementar lentamente el cambio a un grupo de usuarios reducido antes de la implementación masiva.

Se inicia **implementando el producto en un subconjunto de la infraestructura** al que no se enruta ningún usuario. Cuando se alcanza la satisfacción con el producto por parte de la compañía, se empieza a enrutar a algunos usuarios seleccionados.

Entre las ventajas de las Canary Releases está la capacidad de realizar pruebas en un entorno de producción con una estrategia de retroceso segura en caso de que se detecten fallos. La idea principal es **introducir poco a poco carga de trabajo que se pueda monitorizar** y medir cómo impacta en el entorno de producción. Se suele utilizar para probar nuevas funciones de Back-End.

DARK LAUNCHES

Describe el lanzamiento de funciones a un subconjunto de usuarios, de manera que permita conocer las opiniones de estos y, en consecuencia, realizar las actualizaciones oportunas.

Con este método se busca conocer directamente la **respuesta del usuario a las funciones de la interfaz** facilitadas. Normalmente, se lanza a un grupo de usuarios que desconocen lo que están probando y no tienen ninguna tarea señalada a realizar, únicamente probar el producto.

FEATURE TOGGLES

Permite **modificar el comportamiento de los sistemas sin alterar el código**, mediante la clasificación en diferentes categorías de uso introduciendo conmutadores. De este modo, la compañía puede tomar el control del lanzamiento de sus funcionalidades y gestionar aquellas que no funcionan bien.

Los Feature Toggles son variables que se usan dentro de declaraciones condicionales que se pueden activar o desactivar según el valor que se les asigne. Mediante esta técnica se puede controlar el flujo de software y evitar el lanzamiento de las funcionalidades que no están preparadas. No obstante, también pueden ser empleadas para evitar errores al añadir cambios en un software antes de su lanzamiento.

INFRASTRUCTURE AS CODE

Metodología que describe el lenguaje informático de la infraestructura necesaria para ejecutar un software. Consiste en la **programación de estructuras de hardware para adaptarse a los cambios** fácilmente.

Esta técnica toma como ejemplo la tecnología Cloud, que utiliza sus herramientas definidas por software para gestionar el hardware sin acceder manualmente a los dispositivos.

AUTOMATIZAR

Automatizar las implementaciones permite trasladar el software desde los entornos de prueba a la producción sin utilizar procesos manuales, obteniendo implementaciones seguras irrepetibles en todo el ciclo de distribución del producto.

Entre los beneficios de automatizar las implementaciones se encuentra que se pueden **lanzar nuevas características con mayor velocidad** y se elimina la posibilidad del fallo humano en la intervención de la implementación.

Todo esto conforma un método de distribución y supervisión permanente durante el ciclo de vida del producto. Esta técnica depende del correcto diseño de la automatización de pruebas.

SELF-SERVICE DEPLOYMENT

Cada equipo del proceso de creación puede realizar sus funciones sin tener que esperar al resto de los equipos. Generalmente, se emplean **modelos de dominio descendente para crear silos individuales**, y estos silos pueden ser administrados por diferentes grupos de usuarios. Dicho de otro modo, cada equipo puede tener su propio entorno para administrar sus funciones y cada uno de ellos puede compartir su información con el resto. Así pues, la información se empaqueta y se comparte.

Lo más adecuado para esta metodología es brindar al proceso de un enfoque de arriba hacia abajo que permita modelar los entornos en función de la estructura organizativa. El resultado es una **solución de administración de versiones sostenibles y autoservicio** de los equipos.

CONTROL DE VERSIONES

Es una herramienta que ayuda a los equipos de software a gestionar los pequeños cambios que se produzcan en el producto a lo largo del tiempo, mediante un **seguimiento de las modificaciones realizadas en el código** a través de una base de datos específica.

SELECTIVE DEPLOYMENT

Hace referencia a la capacidad de implementar entornos de producción específicos bajo criterios como la función del usuario, la geografía, etcétera.

Se genera un **método sistemático y selectivo** para realizar la evaluación de la eficacia de la implementación. El grupo de usuarios para la implementación por grupos se seleccionan en función de criterios basados en silos.

Output

Desplegar el código en producción: actualmente, existen diversas maneras para realizar el despliegue del código en producción, como por ejemplo: Rolling Upgrade a través de contenedores, Blue & Green Deployment, Canary Releases, etc. Siempre se debe estudiar la estrategia óptima para cada escenario de producción.

Lanzamiento de Dashboards: se utilizan representaciones gráficas para facilitar la toma de decisiones y así optimizar la estrategia.

10. Responde y Recupera

Para asegurar la estabilidad y minimizar el impacto de posibles problemas durante el despliegue de nuevas funcionalidades debemos de ser capaces de detectar proactivamente los problemas y de recuperar de forma rápida el sistema.

El **objetivo** es detectar y resolver proactivamente los problemas de producción antes de que causen una interrupción de la actividad.

Metodología

DETECCIÓN PROACTIVA

La detección proactiva se puede llevar a cabo de diferentes modos: mediante técnicas de ejecución previa y de tiempo de ejecución, se generan diversas capas de detección de ataques y malwares sin necesidad de actualizaciones. Contar con antivirus y realizar pruebas periódicas brindará una **seguridad proactiva para la detección de fallos** y amenazas.

COLABORACIÓN ENTRE EQUIPOS

Brinda la comunicación adecuada durante los procesos de creación, diseño y producción, junto con la facilidad de compartir conocimientos. Con todo ello, aumenta la creatividad y se obtiene un mayor rendimiento de todo el proceso de creación de un producto.

INGENIERÍA DEL CAOS

Consiste en la identificación de los fallos antes de que se provoquen interrupciones en el proceso. Se trata de **poner a un sistema bajo estrés**, de manera que se logre identificar los fallos y, así, poder corregirlos.

La técnica permite la **comparación de fallos en un entorno simulado frente a un fallo en un entorno real**, es decir, se provoca el fallo para conocer sus soluciones antes de que ocurran en un proceso real.

Para llevar a cabo la Ingeniería del Caos se emplean tres pasos para conocer cómo reaccionará el producto ante un fallo:

1. Se debe plantear una hipótesis sobre lo que ocurriría ante una situación perjudicial para producto o proceso.
2. Se deben diseñar diferentes variables simples de fallo para realizar experimentos de fallos.
3. Medición del impacto del fallo.

Así, se conoce el comportamiento del producto ante situaciones adversas al describir situaciones inestables. Cuanto mayor sea la estabilidad del producto ante las variables negativas introducidas, mejor será la calidad de este. Realizar este tipo de técnica **es idóneo para los sistemas distribuidos**, ya que, aunque independientemente funcionen a la perfección, las interacciones entre ellos pueden dar lugar a fallos no planificados. La ejecución de diversos experimentos de manera automatizada y continua impulsará la orquestación y el análisis continuo.

SESSION REPLAY

Trata de replicar la experiencia del usuario a través de un viaje por el sitio web o una aplicación, mediante las grabaciones o las reproducciones de sesiones, o la repetición de usuarios. De esta manera, se optimiza el proceso y las tasas de conversión. Así pues, se pueden visualizar los recorridos de los usuarios de manera anónima, registrando los movimientos en tiempo real.

Mediante esta técnica se **obtiene información precisa del comportamiento del usuario** y se complementan los mapas de calor y desplazamiento por el sitio gracias a las interacciones del cliente. Asimismo, el Session Replay brinda información sobre el tiempo que pasa el usuario en el sitio y las repeticiones de visita al lugar o producto.

ROLLBACK Y FIX FORWARD

Por un lado, el enfoque de **Rollback** significa que se debe escribir un script de reversión con cada script de actualización generado. De este modo, se deben tener en cuenta las posibles reversiones antes de la implementación, realizando los Rollback previamente. No obstante, existen herramientas de migración que permiten la encadenación de scripts de manera automatizada. Se debe ser cuidadoso al usar este enfoque, ya que implica la eliminación de datos y puede perderse información relevante de los scripts.

Por otro lado, el **Fix Forward** en ocasiones conlleva menos riesgos que el Rollback. Con esta metodología se busca mejorar y alcanzar el estado óptimo de funcionamiento. Implica continuar generando cambios que resuelvan un incidente y mantengan activas las funciones previstas.

INFRAESTRUCTURA INMUTABLE

Este enfoque se refiere a que, una vez se introduce un concepto en el producto, este nunca se modificará. En relación con la creación de un producto digital, hace referencia a que **la infraestructura utilizada forma parte del producto** y una vez esté en funcionamiento no se modificará. Cuando se requiera una nueva actualización, se creará una nueva instancia, pero la anterior estará disponible en caso de tener que realizar un Rollback del propio entorno.

El resultado es la **creación de nuevas versiones con un entorno propio de manera controlada**, eliminando la posibilidad de fallos resultantes de instancias rotas. Además, gracias a este enfoque se consigue evitar costes adicionales asociados al mantenimiento de las infraestructuras mutables.

Output

Detección de riesgos:
facilita la identificación y eliminación de aspectos perjudiciales presentes en el entorno de trabajo y producción. Esta es una herramienta fundamental para la prevención de pérdidas y mantener un elevado nivel de seguridad.

Recuperación automática:
generalmente, la recuperación automática permite reanudar el funcionamiento tras experimentar algún fallo de software o hardware.

11. Monitoriza y Mide

Para detectar posibles problemas y entender los comportamientos de uso de la aplicación, se deben dejar diversas trazas en desarrollo, lo que permitirá una monitorización efectiva.

El **objetivo** principal es medir de forma cuantitativa el comportamiento del sistema y de los usuarios en tiempo real.

Metodología

FULL-STACK TELEMETRY

Mecanismo de feedback automatizado que permite la **rápida recopilación de datos para la medición de diferentes parámetros.** De este modo, la supervisión es realmente eficaz mediante la conexión de las diferentes infraestructuras y las interacciones de los clientes para facilitar una visión E2E.

TESTS A/B

Comparación de dos versiones de un mismo producto digital, sitio web o aplicación para comprobar qué versión puede ser más eficiente. Las diferentes versiones se enseñan a los clientes de forma aleatoria, en la que unos verán la versión A y otros la versión B. Tras la muestra de ambas versiones, se realiza un **test estadístico para conocer la efectividad de cada versión a partir de indicadores de rendimiento** como por ejemplo, los clics realizados o las suscripciones de clientes. Puede ser cuantitativa y cualitativa.

FEDERATED MONITORING

Con esta metodología se busca **monitorizar la disponibilidad de todos los recursos que existen.** Mediante la creación de un usuario ficticio se genera una instancia de una máquina virtual para una monitorización externa, lo que permite monitorizar las capacidades de los recursos disponibles.

VISUAL DISPLAYS

Son representaciones que se emplean para **transmitir información a través de diferentes elementos** mediante el uso de diagramas e interfaces. Generalmente, se basan en elementos de visualización de los datos recogidos por otras herramientas.

AIOPS

AIOps hace referencia a las diferentes **plataformas encargadas de automatizar las operaciones IT a través de Inteligencia Artificial, analítica avanzada y Machine Learning.** Se alimentan del Big Data para detectar y solventar problemas de forma automática en tiempo real, además de generar análisis de históricos. AIOps se aleja de los silos de IT para poder agregar datos de observación junto con los datos de participación.

Output

Informes sobre problemas detectados y comportamiento de los usuarios: una vez el producto está en producción, a través de las diferentes metodologías para medir y monitorizar, se puede profundizar sobre los diferentes problemas encontrados o acerca de las verdaderas necesidades de los usuarios, generando informes que recojan estos aspectos.

Alarmas e informes automáticos de analítica: las alarmas automáticas se encargan de avisar de fallos técnicos, detectando anomalías. Generalmente, mediante este sistema de alarmas se puede conocer el nivel de sensibilidad de la anomalía para tomar las medidas ajustadas a la situación.

Análisis de fidelización, conversión y segmentación: el objetivo de este tipo de análisis es la fidelización de clientes conociendo su comportamiento, las temporadas en las que aumente la venta del producto, la importancia de la marca en redes sociales, etc.

Análisis de costes: por instalación, número de usuarios y dispositivos activos, actualizaciones y desinstalaciones. Estos análisis son el resultante de procesos de identificación de recursos, donde se determina la calidad y la cantidad de recursos necesarios. Generalmente, analiza el coste del proyecto global.

Duración de la sesión e intervalos y errores: mediante la medición de diferentes intervalos de tiempo o la duración del inicio de sesión de un usuario, se puede conocer la distribución de usuarios y la agrupación de estos. Así pues, las compañías pueden conocer que tareas realizan los usuarios en su sitio web y, por tanto, ajustar su producto para que sea más atractivo y efectivo. Asimismo, se puede descubrir que categorías gustan particularmente a los grupos de usuarios para potenciarlas aún más y mejorar las menos llamativas.

Análisis de cohortes: para obtener estos análisis se emplean herramientas de atribución en las que se permite la selección de grupos en periodos de tiempo, filtros de información o métricas de valores concretos. La compañía pude hacer frente a la gestión de usuarios y de los datos que destina a la gestión del sitio web, como el engagement de los usuarios y su comportamiento.

12. Aprende

Para mejorar el proceso de toma de decisiones deben estudiarse los datos obtenidos acerca del comportamiento de los usuarios en la solución implementada.

El **objetivo** es comprender y poder tomar decisiones inteligentes basadas en datos.

Metodología

PENSAMIENTO SISTÉMICO

El pensamiento sistémico hace referencia a una filosofía subyacente que cuenta con diferentes herramientas y métodos para **estudiar diferentes arquetipos** y para conocer cómo se puede entender y visualizar a la compañía o producto.

Generalmente, se puede emplear como una **herramienta de diagnóstico**, donde se examinan las dificultades de una manera detallada antes de realizar ninguna actualización o cambio.

Mediante este tipo de pensamiento se realizan la observación de eventos, la identificación de datos y la búsqueda de patrones de comportamiento en el histórico, que normalmente **generan estructuras subyacentes que ayuden a mejorar el producto o servicio.** Así pues, cuando se modifican las estructuras que no son útiles, se obtiene una mejora de rendimiento y se pueden crear nuevas soluciones para los problemas detectados.

GENERATIVE CULTURE

Es un enfoque proactivo, donde se trata de hacer llegar la información a todas las áreas y personas por diversos medios. En este tipo de cultura, se pone especial interés en el **alcance colectivo de las metas de la organización** más que en los beneficios y las pautas que implican alcanzarlas.

En consecuencia, la Generative Culture viene **determinada por la forma en la que se procesa y maneja la información.** Por tanto, el énfasis en el desempeño afectará directamente al procesamiento de la información y la funcionalidad de la compañía. Generalmente, esta forma de cultura es altamente funcional.

MEJORA CONTINUA

A través de esta metodología, se busca una mejor gestión de la competitividad entre empresas a través de su productividad de un modo sistemático y permanente en todo su recorrido. Gracias a la mejora continua se **eliminan las operaciones innecesarias que no añaden valor al producto**, donde los procesos que añaden este valor son los que se identifican como ejecutores de los procesos de mejora. De este modo, los procesos o productos cada vez tendrán menos problemáticas y adversidades a las que enfrentarse.

Output

Aprendizajes sobre el producto:
una actividad reflexiva que contribuye al cambio y a la mejora continua del producto, planificando los objetivos del aprendizaje. Mediante el aprendizaje continuo se puede constituir una base teórica para un diseño y una construcción más eficiente y sin fallos.

Aprendizajes sobre el usuario:
a la hora de diseñar productos digitales es fundamental atender las necesidades y comportamientos del usuario, aprender cómo gestionarlas y usarlas para generar el máximo valor posible del producto o del proceso de creación del mismo.

Aprendizajes sobre el proceso:
la realización de pruebas continuas durante el proceso de creación de un producto digital permite conocer y mejorar los fallos del proceso, así como la aplicación de diferentes metodologías para añadir valor.

Softtek®