

Seminario de Sistemas Distribuidos

ON DESIGNING AND DEPLOYING INTERNET- SCALE SERVICES

John Hamilton

17 de febrero de 2022

- De Jesús Moreno Yolanda
- Herrera Godina Adriana Jocelyn
- Sanchez Torres Sergio Daniel

AGENDA

1

Introducción

2

Recomendaciones

3

Conclusiones

4

Q&A

INTRODUCCIÓN



Buenas prácticas para el diseño y desarrollo de servicios de operaciones amigables

Sistemas de software centrados en datos a gran escala

Computación Orientada a la Recuperación y Crash-Only Software

Se destacan 3 buenas prácticas, implementadas por Bill Hoffman:

- Esperar fallos
- Mantener las cosas simples
- Automatizar todo

RECOMENDACIONES

Se dividirán en 10 secciones:

- Diseño general de la aplicación
- Gestión automática y aprovisionamiento
- Gestión de la dependencia
- Ciclo de lanzamiento y pruebas
- Selección y estandarización de hardware
- Planificación de operaciones y capacidad
- Auditoría, monitoreo y alertas
- Degradación elegante y control de admisión
- Plan de comunicación con clientes y prensa
- Autoprovisionamiento y autoayuda del cliente

1. DISEÑO GENERAL DE LA APLICACIÓN



Se puede catalogar como una de las secciones más importantes

¿FALLOS?

- Es normal observar primero las operaciones
- Los problemas de operaciones se originan en el diseño y desarrollo
- ¿Es eficaz separar el desarrollo, las pruebas y operaciones?
- ¿Es mejor juntar las secciones?



CONCEPTOS BÁSICOS DE OPERACIONES AMIGABLES

Diseño para fallos

Redundancia y recuperación de fallas

Segmento de hardware básico

Software de una sola versión

Multiusuario

***Las que tienen mayor impacto**

BUENAS PRÁCTICAS PARA EL DISEÑO DE SERVICIOS DE OPERACIONES AMIGABLES

- Servicio rápido de testeo
- Tener una versión de todo
- Permitir la intervención humana
- Analizar la latencia y el rendimiento (throughput)
- Multusuario

2. GESTIÓN AUTOMÁTICA Y APROVISIONAMIENTO



BUENAS PRÁCTICAS EN EL DISEÑO PARA AUTOMATIZAR

- Ser redundante y reiniciable
- Apoyo de la distribución geográfica
- Instalación y aprovisionamiento automático
- Mantener la implementación simple
- Hacer fallar los servicios regularmente



3. GESTIÓN DE LA DEPENDENCIA

No se le ofrece la atención que
merece

BUENAS PRÁCTICAS PARA LA GESTIÓN DE DEPENDENCIAS

- Suponer la latencia
- Aislar las fallas
- Implementar monitoreos y alertas como servicios en el inter
- Usar componentes probados y de envío
- Desacoplar componentes

CICLO DE LANZAMIENTO DE PRUEBAS



Calidad



La mayoría de los servicios tienen al menos una laboratorio de pruebas que es tan similar a la producción como (económicamente) posible y todos los buenos equipos de ingeniería utilizan cargas de trabajo de producción para impulsar los sistemas de prueba de manera realista.



REGLAS

1

El sistema de producción ha de tener suficiente redundancia

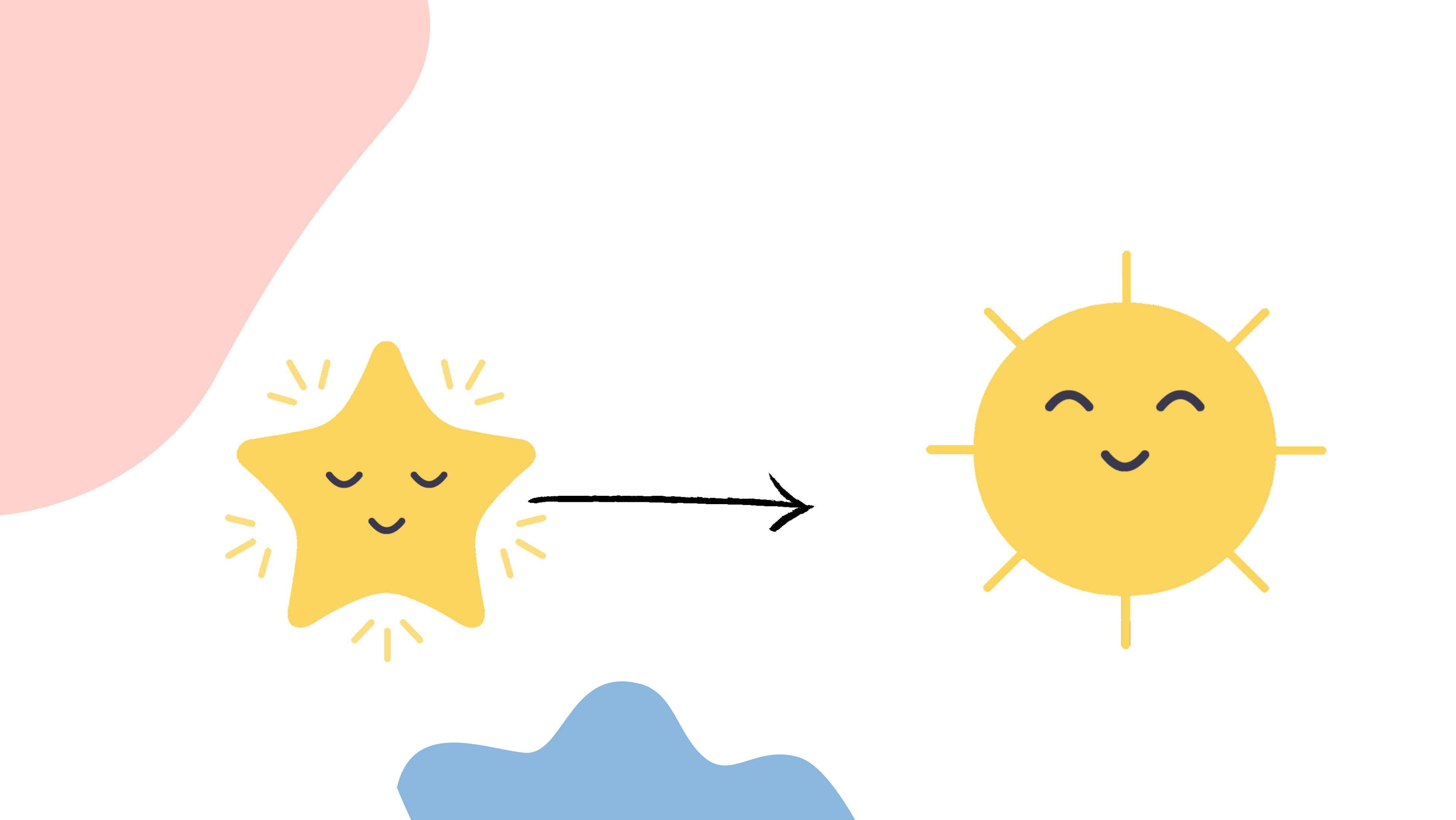
2

Los errores deben ser detectados

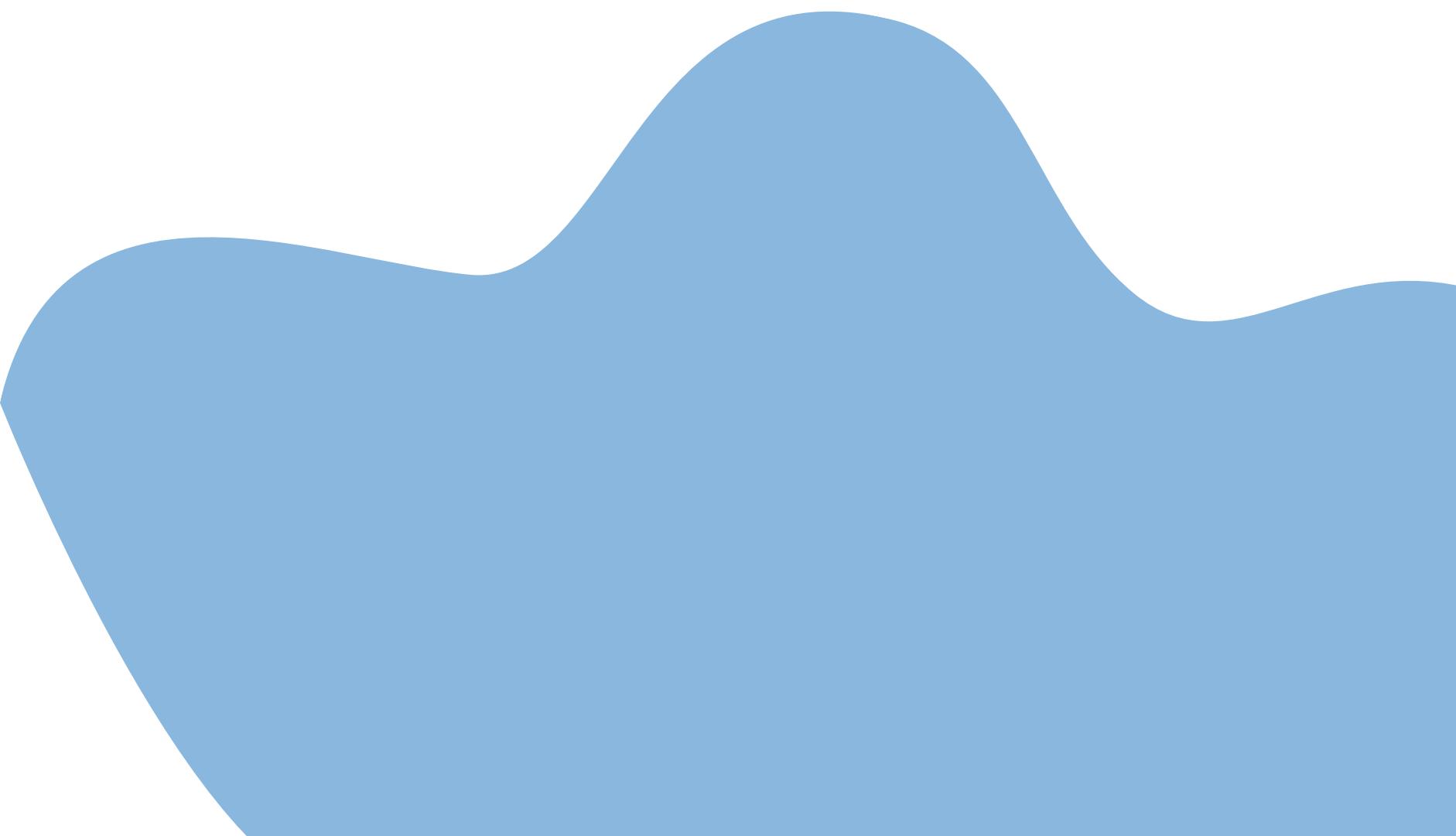
3

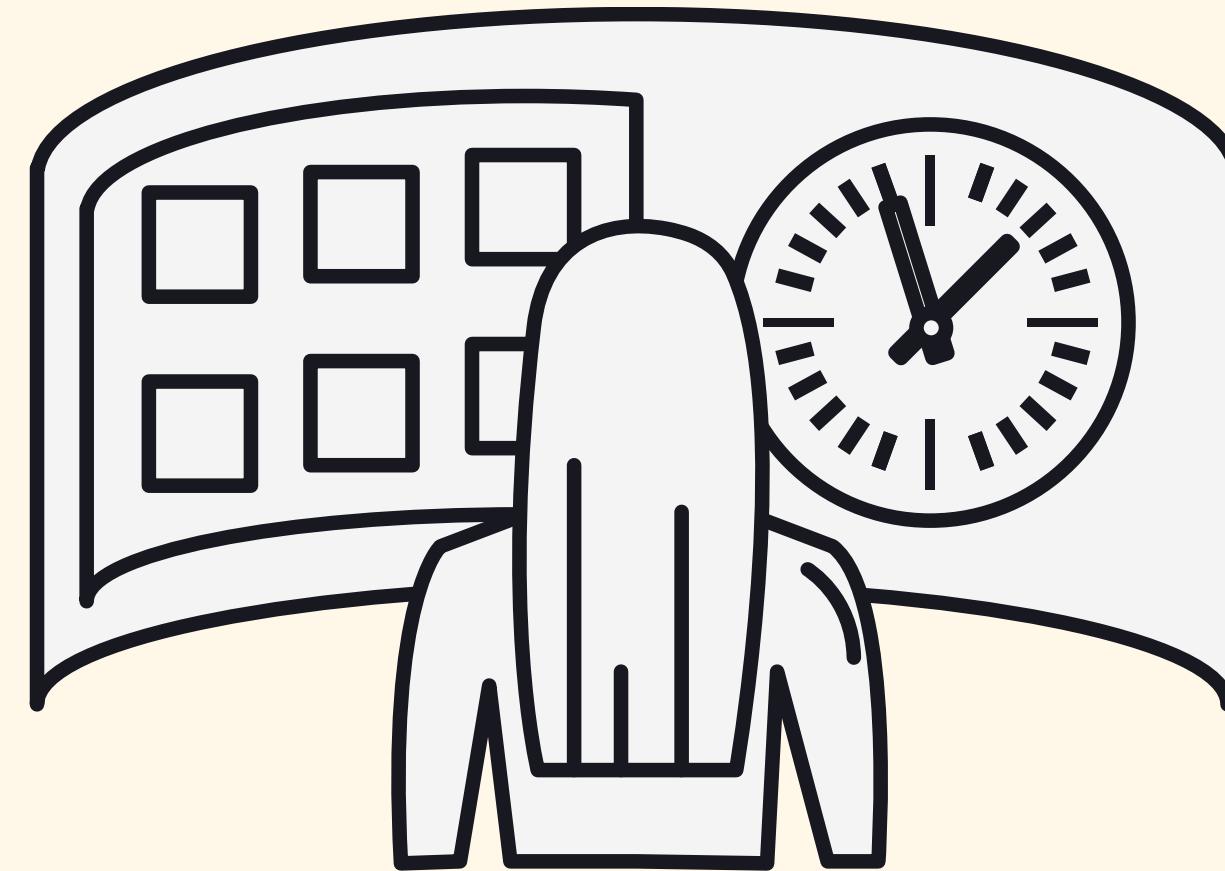
Debe ser posible revertir rápidamente todos los cambios





Mejores practicas





- Usar datos de producción para encontrar problemas.
- Invertir en ingeniería.
- Reversión de la versión de soporte
- Mantener la compatibilidad hacia adelante y hacia atrás.
- Implementación de un solo servidor
- Prueba de estrés por carga



- Realizar pruebas de capacidad y rendimiento
- Cree e implemente de manera superficial e iterativa
- Prueba con datos reales.
- Ejecutar pruebas de aceptación a nivel del sistema
- Probar y desarrollar en entornos completos.

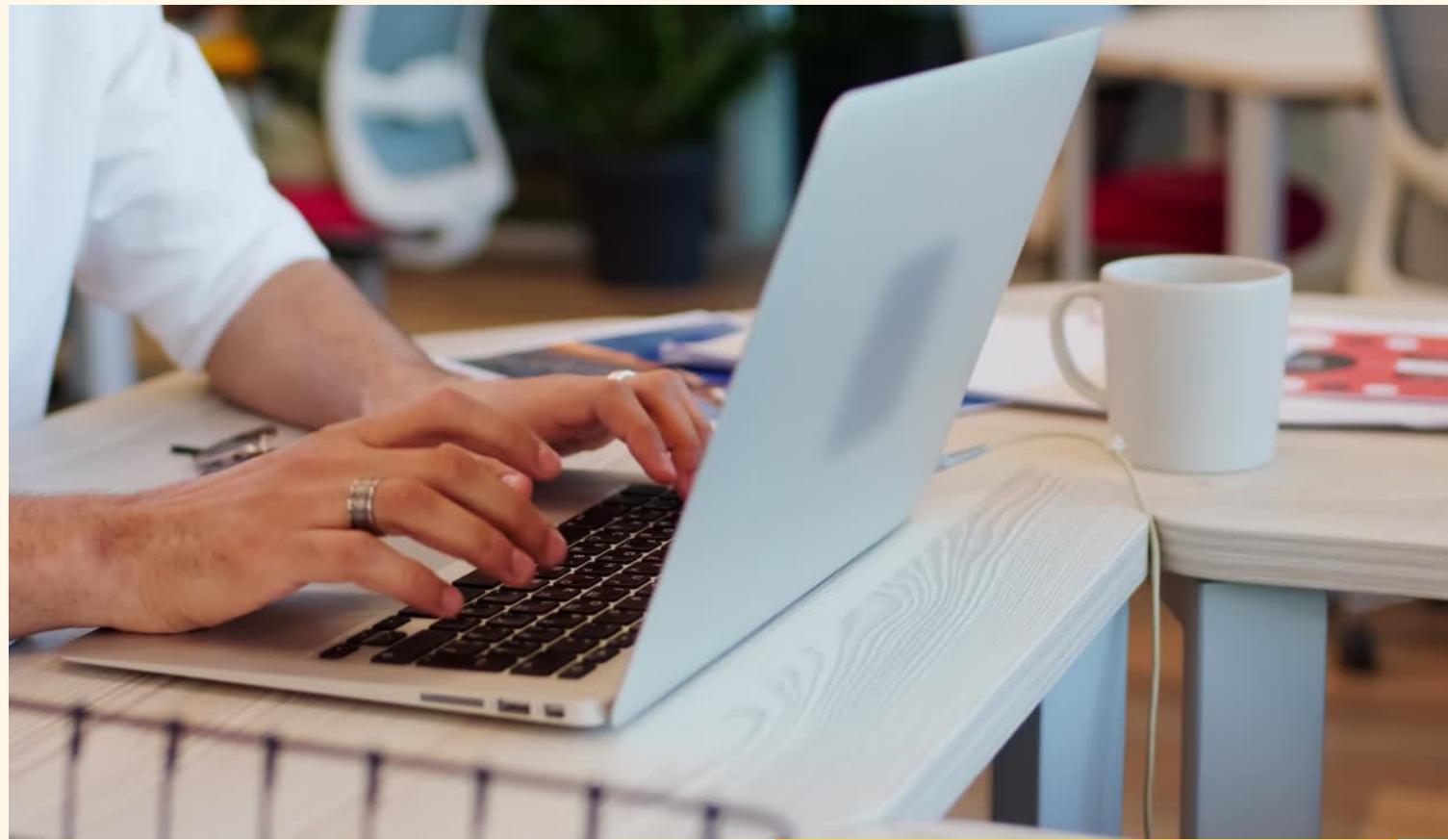
SELECCIÓN Y ESTANDARIZACIÓN DE HARDWARE



- Determinar qué hardware es actualmente el mejor
- Hacer la calificación del hardware y la implementación del software una vez que el hardware esté instalado



MEJORES PRACTICAS



- Use solo SKU estándar
- Compre bastidores completos
- Escribir en una abstracción de hardware
- Abstraer la red y el naming

PLANIFICACIÓN DE OPERACIONES Y CAPACIDAD



```
  $('.count').each(function () {
    $(this).prop('Counter',0).animate({
        Counter: $(this).text()
    }, {
        duration: 4000,
        easing: 'swing',
        step: function (now) {
            $(this).text(Math.ceil(now));
        }
    });
});
```

document.onload = function(){
 document.getElementById('objeto').onclick = function(){
 fadeIn();
 }
};

Responsabilizar al equipo de desarrollo.

Sólo borrado temporal.

Seguimiento de la asignación de recursos

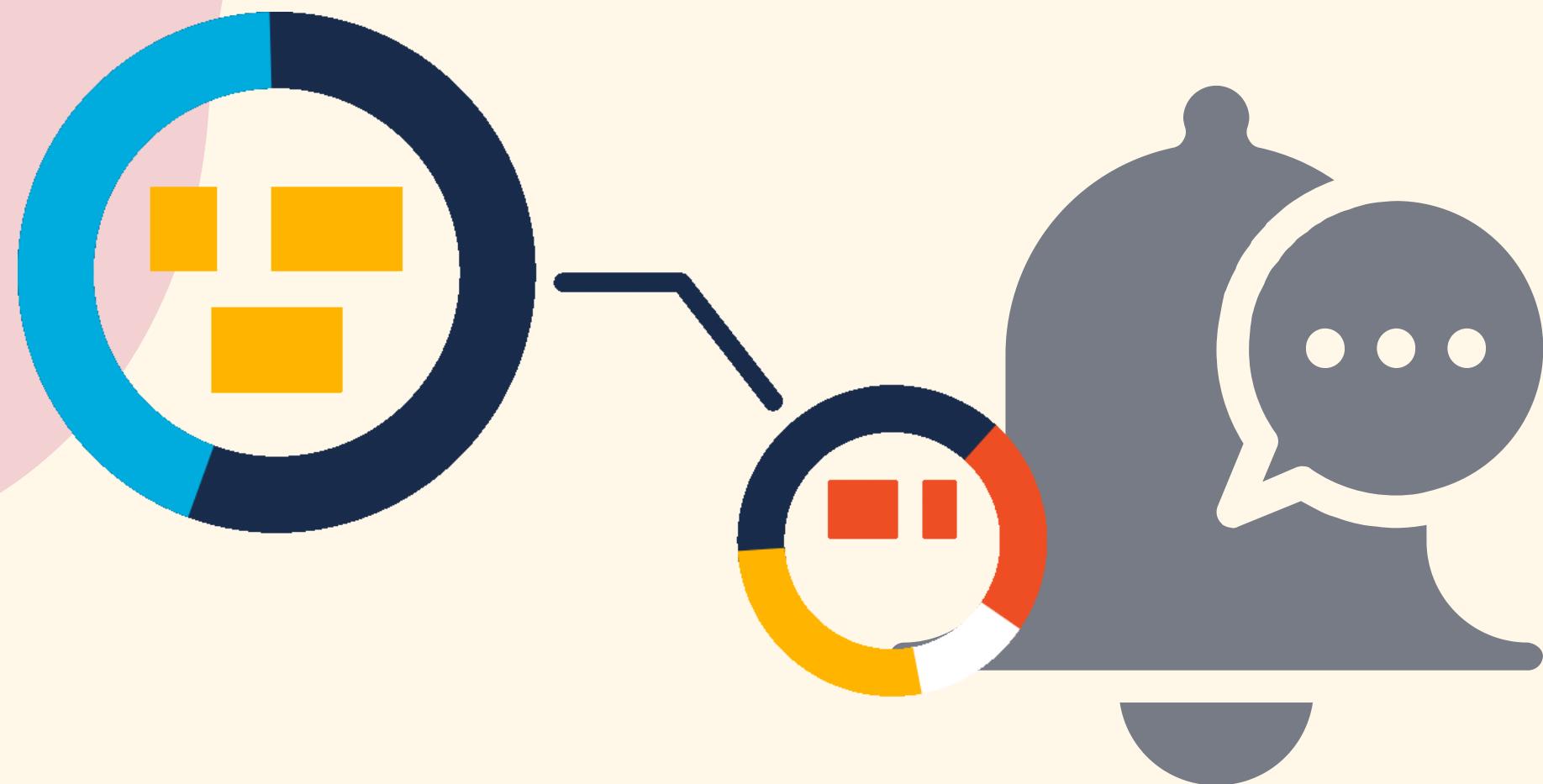
Realice un cambio a la vez

Haga que todo sea configurable



AUDITORÍA, SUPERVISIÓN Y ALERTA

Cada vez que se produzca un cambio de configuración, es necesario tomar los datos en el registro de auditoría el cambio exacto, quién lo hizo y cuándo se hizo.



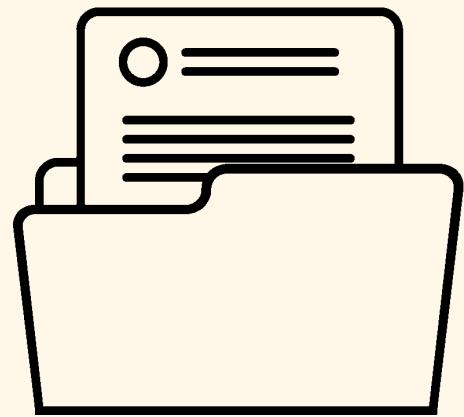
Cada alerta debe representar un problema.

Métricas de alerta:

Rastrear:

- La relación de alertas y notificación de problemas (con un objetivo de casi uno)
- El número de problemas de salud de los sistemas sin las alertas correspondientes (con un objetivo de casi cero).

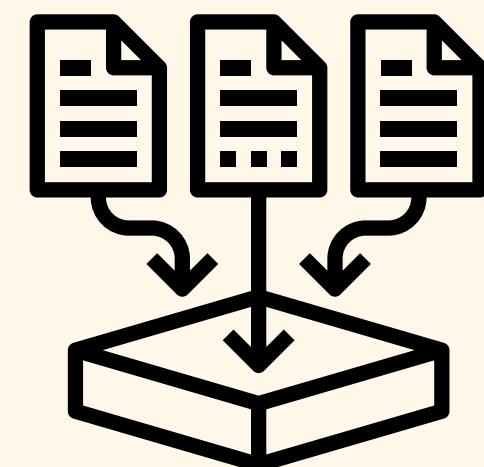
Instrumenta Todo.



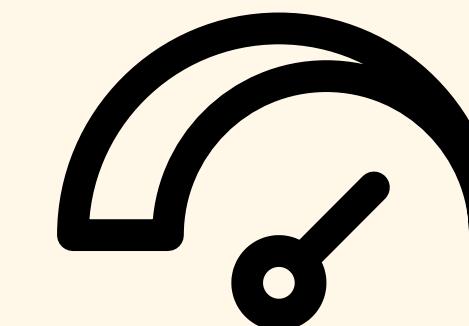
Los datos son el activo más valioso



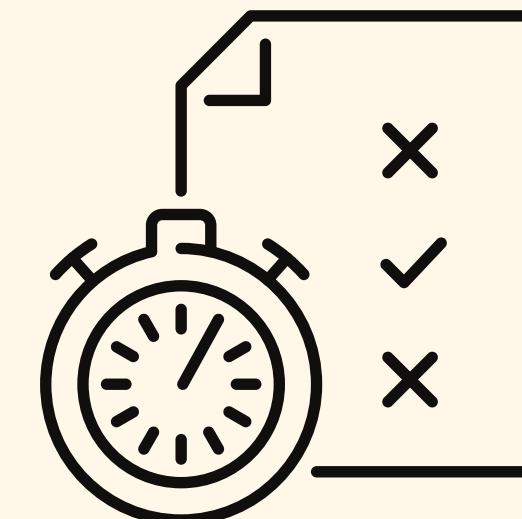
Tener una visión del servicio al cliente



Disponer de suficientes datos de producción

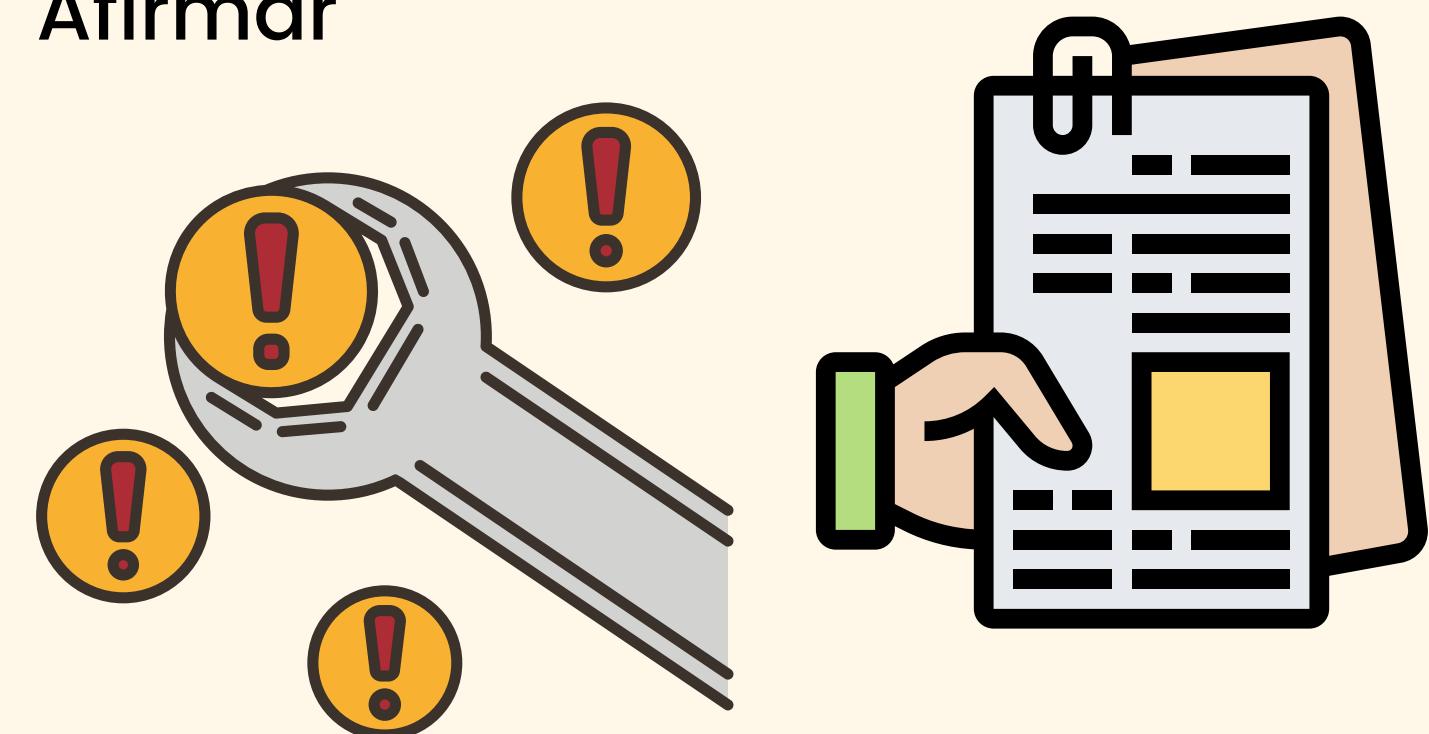


Las latencias son el problema más difícil.



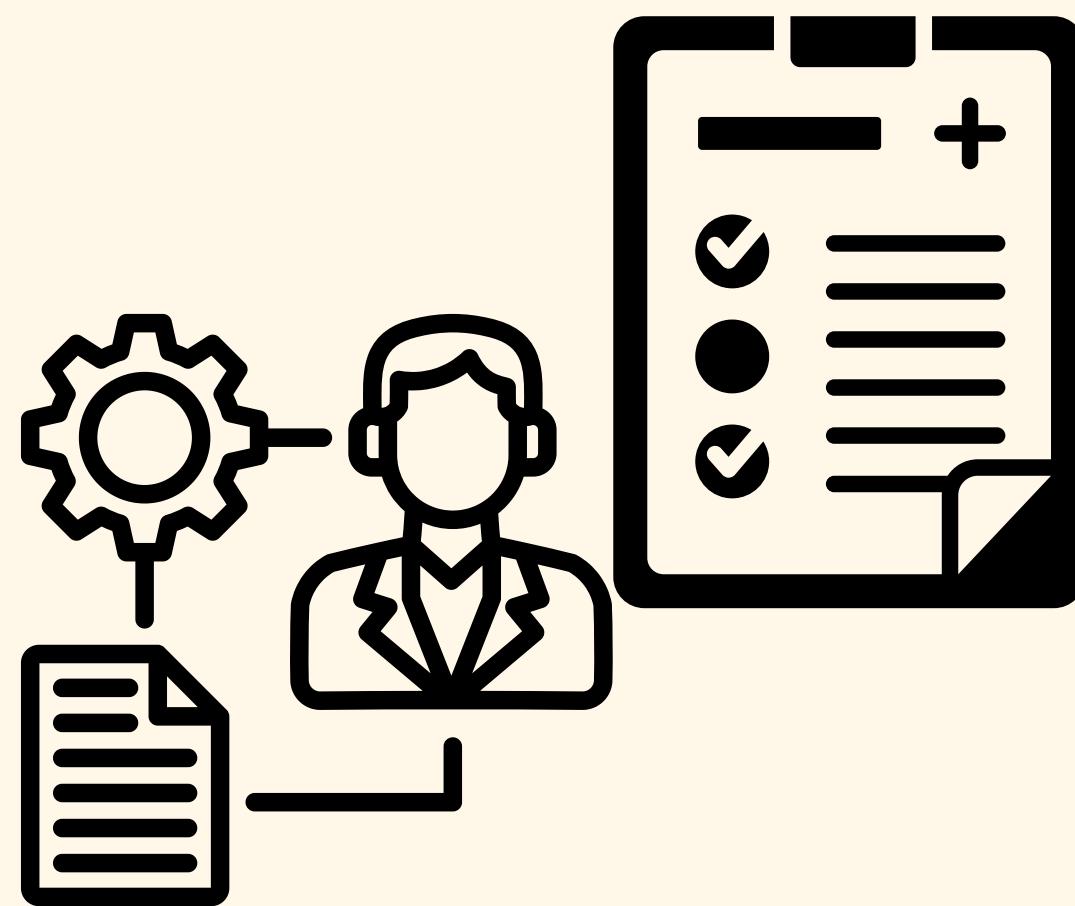
Disponer de suficientes datos de producción

- Utilice contadores de rendimiento para todas las operaciones.
- Auditar todas las operaciones
- Seguimiento de todos los mecanismos de tolerancia a fallos.
- Mantener los datos históricos
- Seguimiento de las operaciones contra entidades importantes
- Afirmar

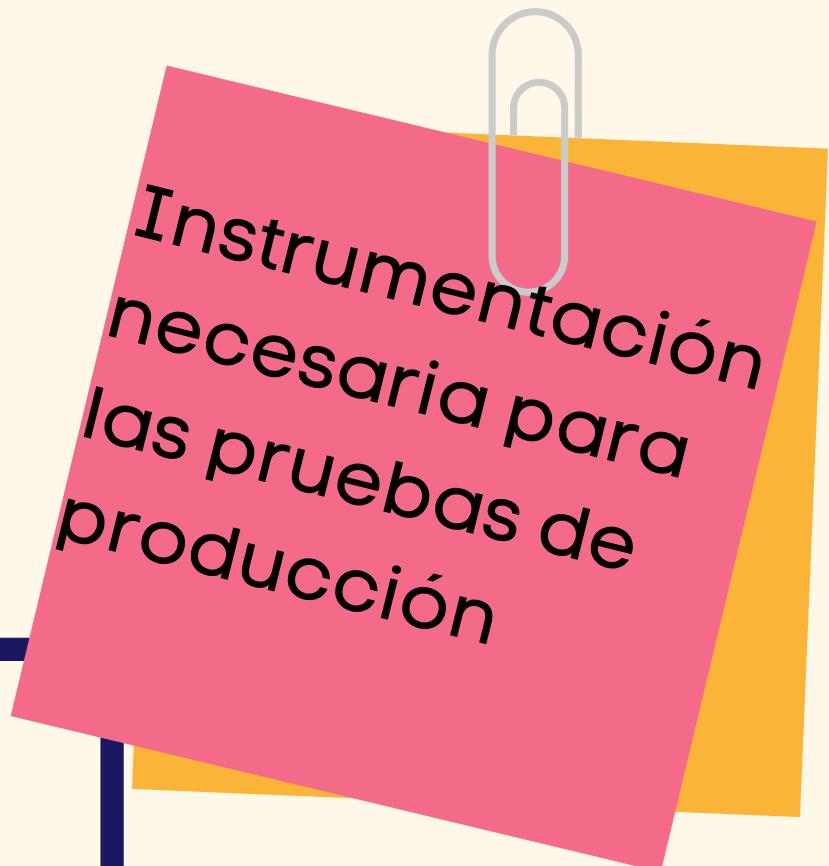


Hacer que todos los errores reportados sean procesables.

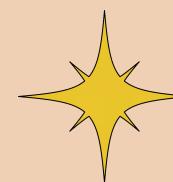
Permitir un diagnóstico rápido de los problemas de producción.



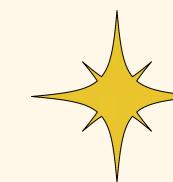
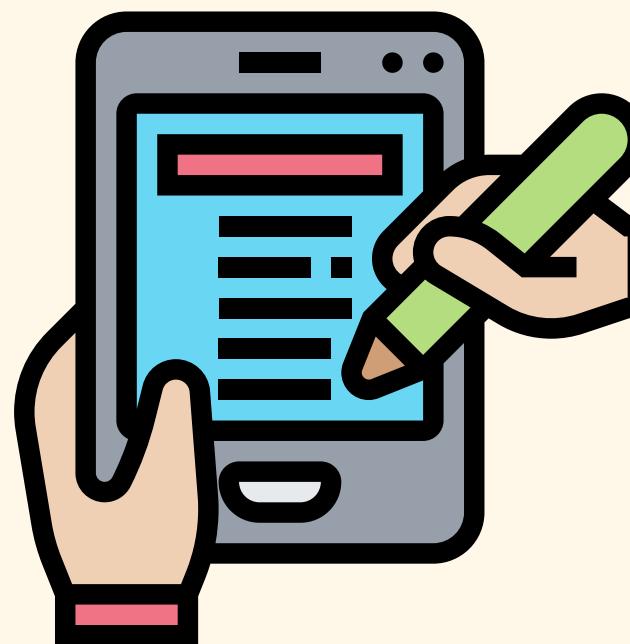
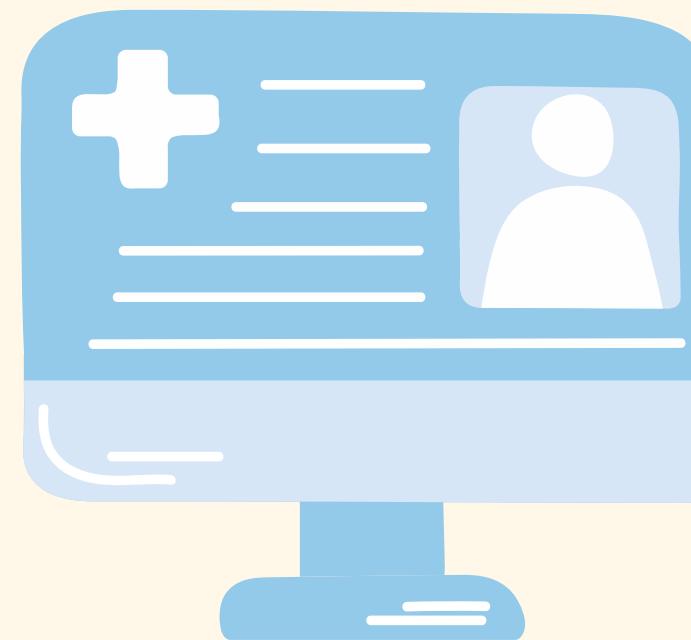
Exponer la información relativa a la seguridad vial para su control



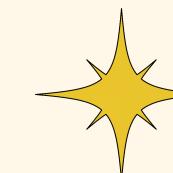
Permitir un diagnóstico rápido de los problemas de producción.



Dar suficiente información para diagnosticar

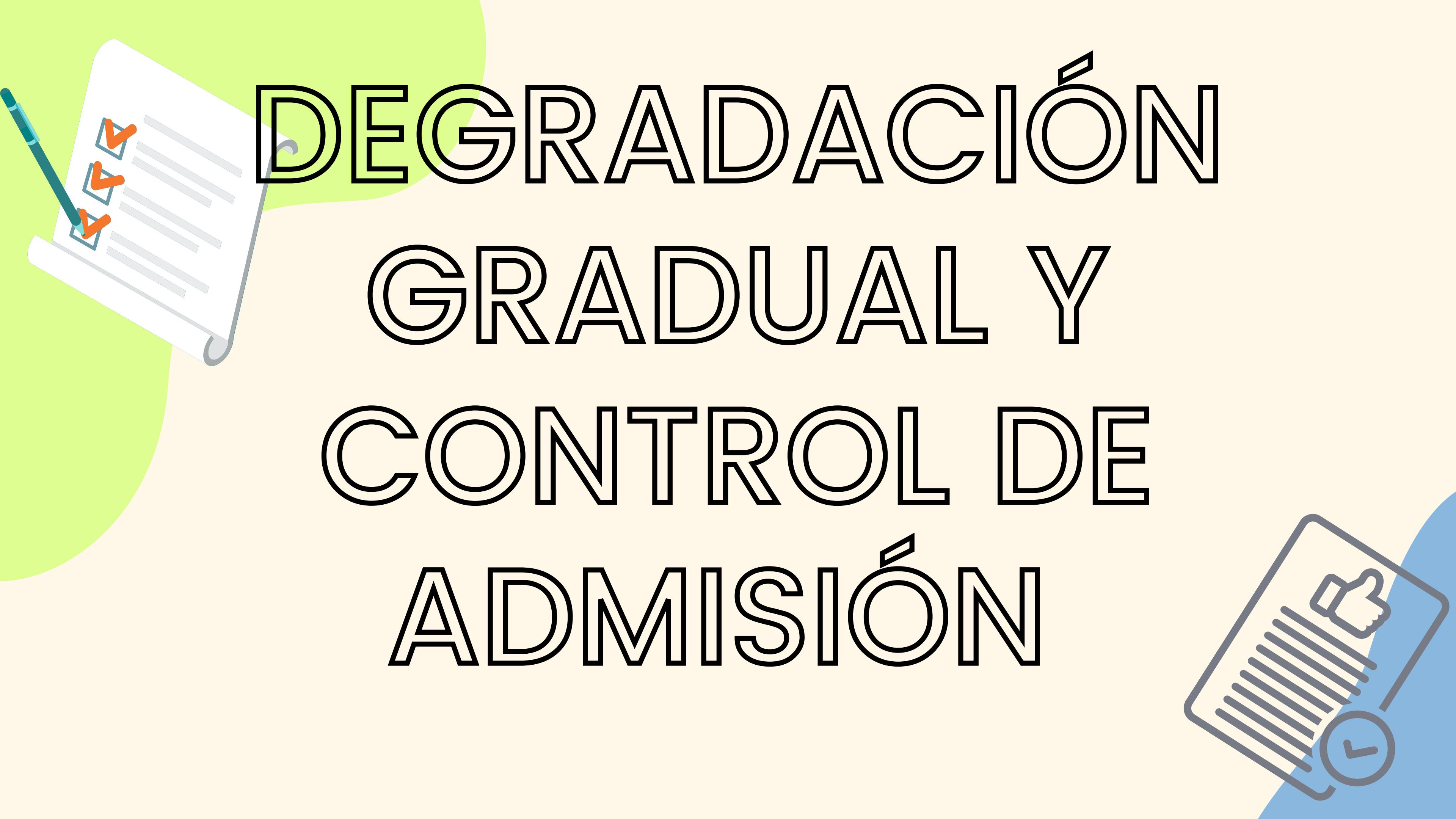


Cadena de evidencia. Depuración en producción.



Registre todas las acciones significativas.

DEGRADACIÓN GRADUAL Y CONTROL DE ADMISIÓN



El servicio debe ser capaz de degradarse con elegancia y controlar las admisiones



La capacidad de eliminar la carga no crítica en caso de emergencia.



Apoyarse un "gran interruptor rojo"

¿Qué mandar al botón rojo?

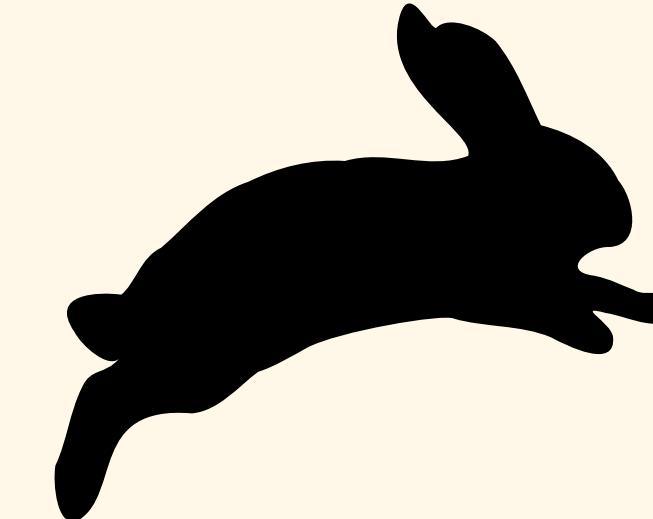
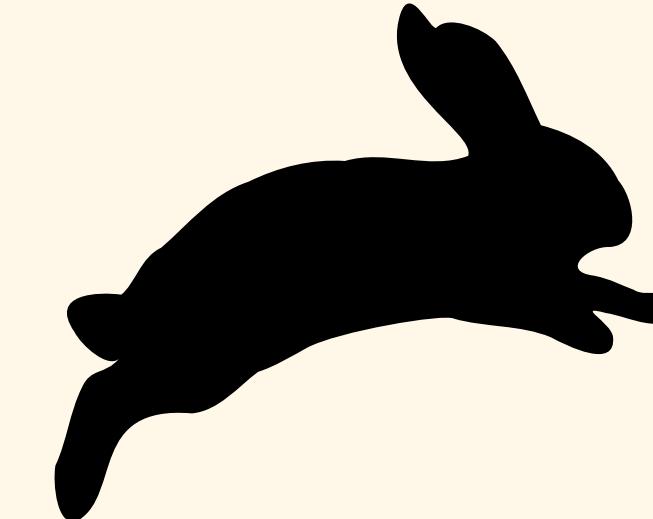
Si hay alguna carga que puede ponerse en cola (importancia) y procesarse más tarde



Control de admisión



Medidor de admisión



Permitir que el cliente siga operando con datos locales, si es el caso, y conseguir hasta que el cliente se retire

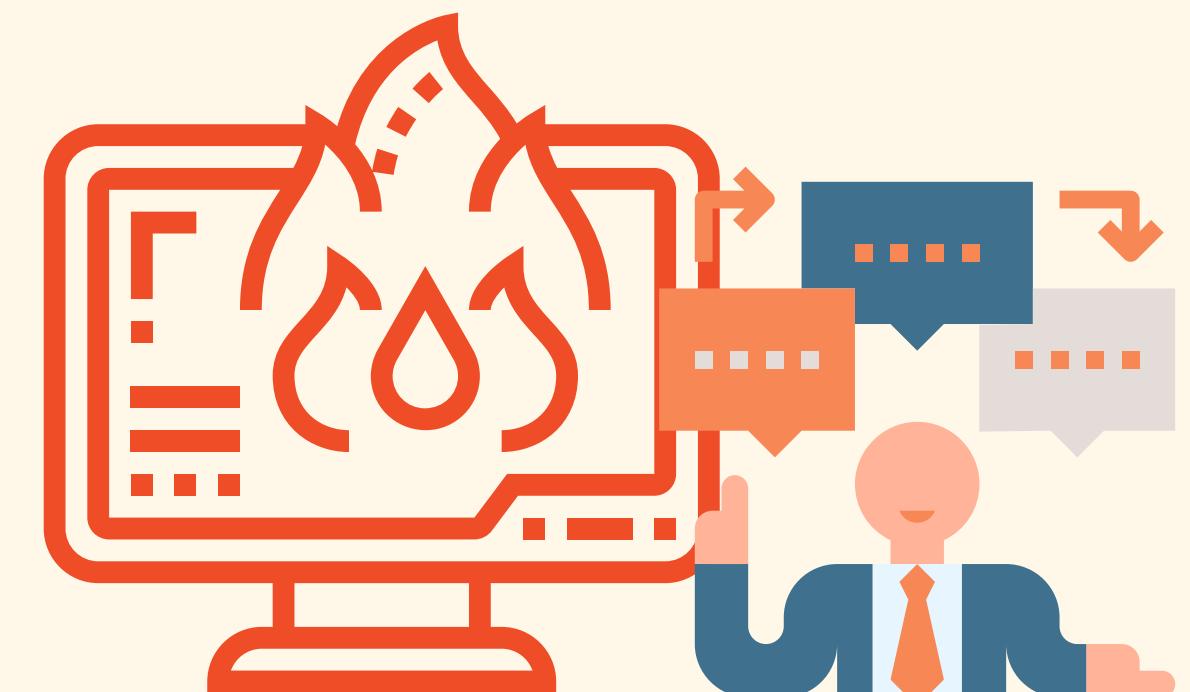
PLAN DE COMUNICACIÓN CON LOS CLIENTES Y LA PRENSA



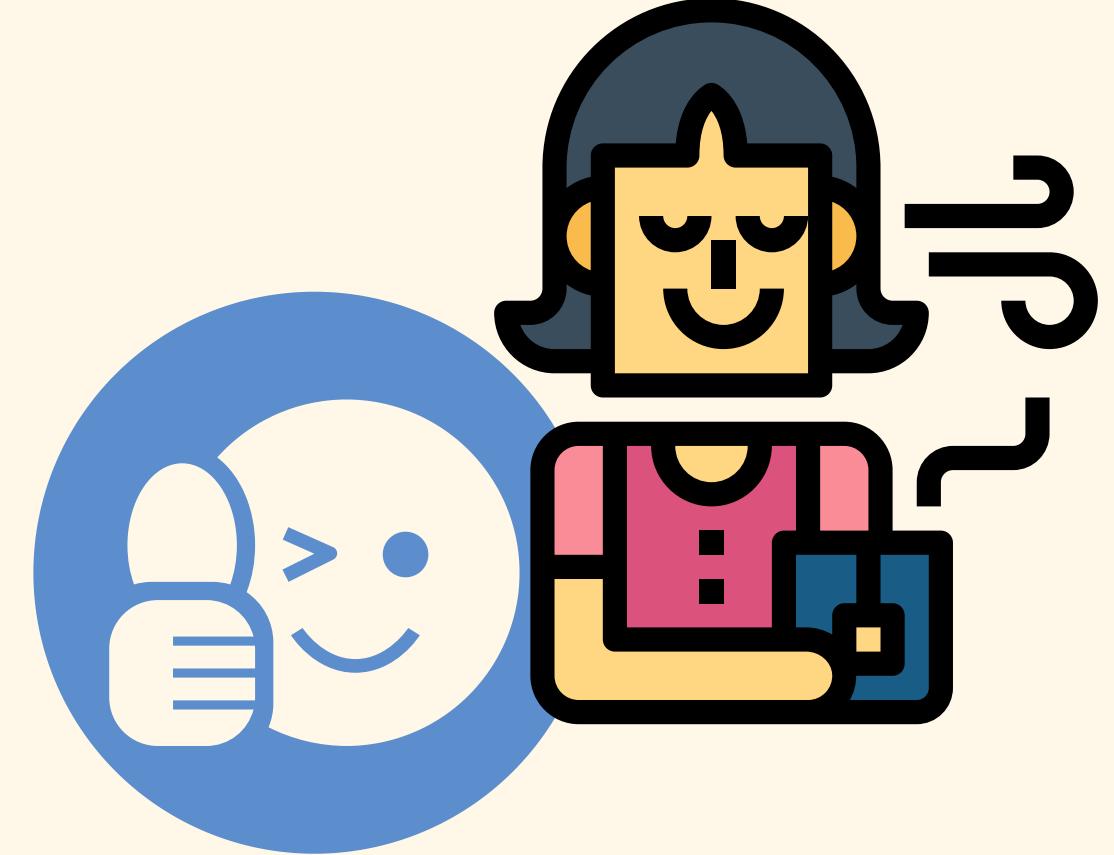
Los sistemas fallan, y habrá ocasiones en las que la latencia u otros problemas deban ser comunicados a los clientes



Algunos tipos de eventos traerán consigo la cobertura de la prensa



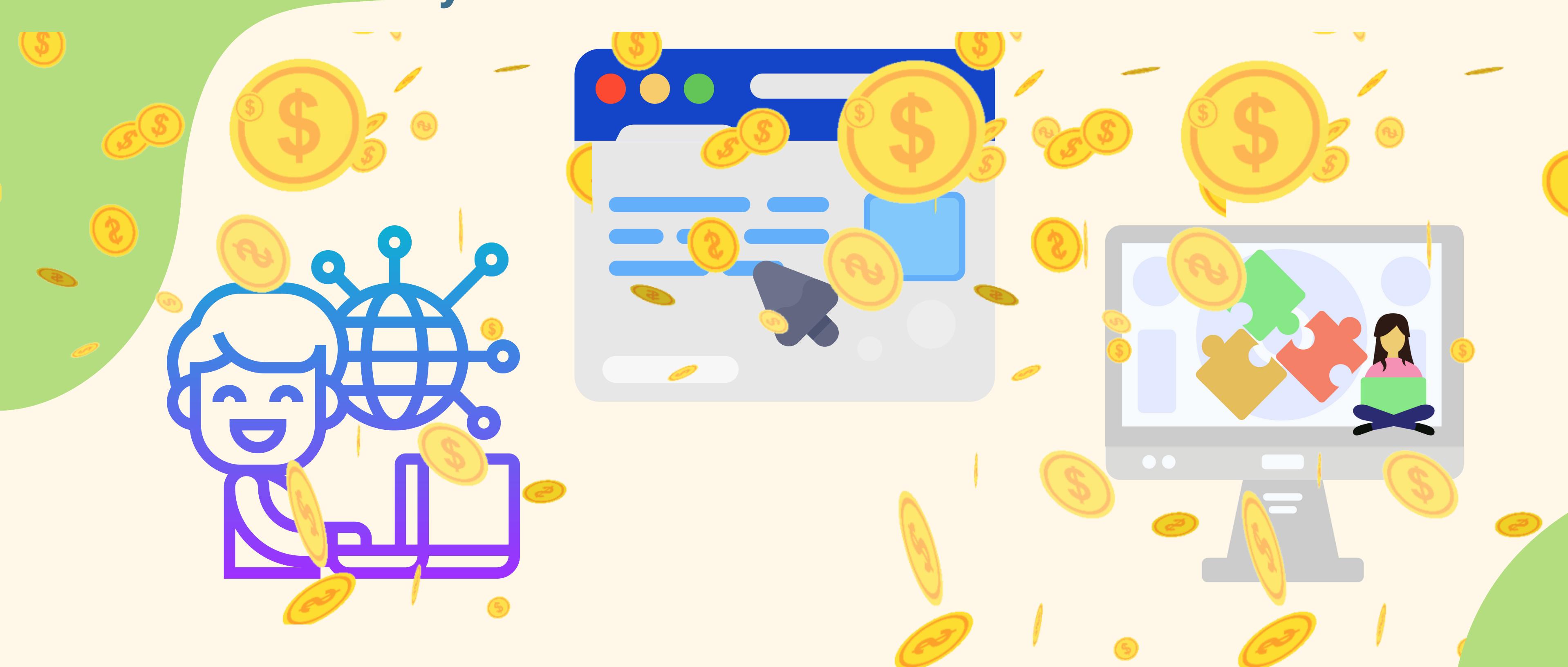
Si los usuarios entienden lo que está pasando y tienen una expectativa razonable de cuándo se restablecerá el servicio



AUTOAPROVISIONA
MIENTO Y
AUTOAYUDA DEL
CLIENTE



El autoaprovisionamiento del cliente reduce sustancialmente los costes y también aumenta la satisfacción del cliente



EJEMPLO



*Diseño para fallos

CONCLUSIONES

Al realizar buenas prácticas podemos evitar costos sumamente excesivos a largo plazo, así reducir costos operativos de igual forma. La confiabilidad del servicio a gran escala hace que sea más fácil de operar.

También al leer el artículo pudimos darnos cuenta que no hay que descuidar reglas importantes para que tengamos mejores prácticas en el diseño, desarrollo, implementación y operación de servicios a gran escala.



REFERENCIAS



Hamilton, James (2016). On Designing and Deploying Internet-Scale Services. Recuperado el 4 de febrero de 2022, de <https://s3.amazonaws.com/systemsandpapers/papers/hamilton.pdf>



<https://github.com/daniel4torres/DistributedSystems>

THANKYOU