Gestión de Proyectos Software

Detectar falso agilismo (*agile bullshit*)

Agile bullshit

- El Defense Innovation Board del Departamento de Defensa de EE.UU. contrata mucho software
- Como el término ágil está de moda y todos los proveedores de software presumen de serlo, publicaron una breve guía que usan internamente para detectar lo que ellos llaman *agile bullshit*
  - Formas de detectar que un proveedor de software que presume de ser ágil, no lo es realmente

Claves para detectar que un proyecto no es ágil de verdad

- Nadie habla con los usuarios del software en acción ni les observa usándolo
  - Usuarios reales con el software real
- No hay feedback continuo desde los usuarios hasta el equipo de desarrollo
  - No basta con hablar con ellos al principio del desarrollo para verificar los requisitos
- Cumplir los requisitos se considera más importante que entregar algo útil lo antes posible

Claves para detectar que un proyecto no es ágil de verdad

- Los interesados (desarrollo, test, seguridad, subcontratas, usuarios finales...) no están coordinados, actúan cada uno por su cuenta
  - La frase "eso no es mi trabajo" surge frecuentemente al hablar con ellos
- Los/as usuarios/as finales no aparecen por ningún sitio durante el desarrollo
  - Como mínimo deberían participar en la planificación de lanzamientos y los tests de aceptación por usuarios/as
- No hay cultura DevSecOps si se tolera que se hagan cosas a mano que podrían y deberían ser automatizadas
  - Tests, integración continua, entrega continua

Herramientas habituales en equipos ágiles

- Usarlas no garantiza nada, pero un equipo que no usa ninguna de estas cosas (o equivalentes) seguro que no es ágil
- Las herramientas cambian con el tiempo, pero señalan algunas actuales (no las pongo todas)
  - Git u otro sistema de control de versiones
  - BitBucket o GitHub para alojar repositorios y otras cosas como la gestión de incidencias
  - Jenkins, Circle CI, Travis CI, algo que ayude a montar sistemas CI
  - Chef, Ansible, Puppet para la configuración automática de servidores
  - Docker
  - Kubernets o Docker Swarm
  - Jira (gestión de incidencias, seguimiento y gestión del estado del proyecto)

Preguntas

- El documento propone unas cuantas preguntas que se les puede hacer a los contratistas para determinar su grado de agilidad real
- Pongo unas cuantas a continuación

Preguntas a los equipos de desarrollo

- ¿Cómo probáis el código?
- Respuestas malas
  - Tenemos un grupo de testers
  - El equipo de operaciones e ingeniería es responsable de las pruebas
- Si se quiere indagar un poco más se les puede preguntar
  - Por las herramientas de testing que usan para pruebas unitarias, funcionales, de seguridad etc.

Preguntas a los equipos de desarrollo

- ¿Cómo de automatizados están vuestro desarrollo, pruebas, seguridad o pipelines de despliegue?
- Si se quiere indagar un poco más se les puede preguntar
  - Por las herramientas que usan para CI/CD
  - Por las herramientas que usan para la documentación del software
  - Si su infraestructura de hardware está definida mediante código

Preguntas a los equipos de desarrollo

- ¿Quiénes son vuestros usuarios y cómo interactuáis con ellos?
- Si se quiere indagar un poco más se les puede preguntar
  - ¿Qué mecanismos usáis para tener feedback directo de los usuarios?
  - ¿Qué herramienta usáis para que los usuarios reporten incidencias?
  - ¿Cómo se asignan incidencias a los desarrolladores?
  - ¿Cómo se informa a los usuarios de que sus incidencias se están tratando, o que han sido resueltas?

Preguntas a los equipos de desarrollo

- ¿Cuál es vuestro tiempo de ciclo, actual y futuro, para entregar a vuestros usuarios?
  - La pregunta es que cuánto tarda una funcionalidad nueva entre que se la piden y les llega a los usuarios
- Si se quiere indagar un poco más se les puede preguntar
  - ¿Usáis contenedores?
  - ¿Qué herramientas de gestión de configuraciones usáis?

Preguntas a clientes y usuarios

- ¿Cómo os comunicáis con los desarrolladores?
- ¿Os observan usando la aplicación y hacen preguntas que demuestran que comprenden vuestras necesidades?
- ¿Cuándo es la última vez que se sentaron a hablar con vosotros sobre funcionalidades que os gustaría tener?
- ¿Cómo enviáis sugerencias o informes de errores?
- ¿Cómo os dan el feedback de esas sugerencias o informes?
- Desde que pedís un cambio hasta que lo tenéis, ¿cuánto tiempo pasa?

Preguntas a la gerencia

- ¿Cuáles son las métricas de gestión sobre desarrollo y operaciones?
  - Se buscan respuestas como estas:
    - Velocidad de los equipos
    - Tiempo de ciclo (desde que se pide una funcionalidad hasta que llega a los usuarios)
    - Tiempo de *lead* (desde que se sube el código al VCS hasta que llega a los usuarios)
    - Cobertura de tests
    - Tasa de cambio del código (*code churn*)
    - Tiempo de recuperación del servicio después de un corte
    - ...
- ¿Cómo se usan para informar sobre prioridades, detectar problemas etc.?
- ¿Con qué frecuencia los managers y la dirección técnica consultan estas métricas?

Preguntas a la gerencia

- ¿Los equipos entregan software a, al menos, una parte de los usuarios reales en cada iteración para obtener feedback?
- ¿Hay una guía/plan de producto con los objetivos estratégicos del mismo? ¿La entienden todos los miembros del equipo?
- ¿Se convierte el feedback de los usuarios en items de trabajo concretos para los equipos en plazos inferiores a un mes?
- ¿Se anima a los equipos a cambiar requisitos en base al feedback de los usuarios?
- ¿Se anima a los equipos a cambiar sus procesos en base a lo que aprenden?
- ¿Es todo el ecosistema de tu proyecto ágil?
  - Por ejemplo, no tiene sentido usar desarrollo ágil y luego tener despliegues tradicionales, manuales y tardíos
- **Las respuestas a estas 6 preguntas deberían ser "sí"**

Bibliografía

- DIB Guide: Detecting Agile BS
- https://media.defense.gov/2019/May/02/2002127286/-1/-1/0/DIBGUIDEDETECTINGAGILEBS.PDF