Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ciencia de la Computación

Clase 21
Pruebas de validación

IIC3745 – Testing

Rodrigo Saffie

rasaffie@uc.cl

18 de noviembre de 2020

Pruebas de validación

- Con las pruebas unitarias y de integración tenemos cierto nivel de certeza que la aplicación “funciona”.

  ¿Funciona como los usuarios esperan/quieren?

- Puede que la lógica de negocio esté correcta, pero es **inútil** si es que los usuarios no saben/pueden interactuar con ella.

Pruebas de usuarios

- Se enfocan en analizar cómo los usuarios interactúan con la aplicación.

- Se definen tareas que los usuarios deben completar para así detectar dificultades o sensaciones de estos.

- No se está evaluando a los usuarios, sino que si la aplicación cumple con sus expectativas.

¿Cuándo se prueba?

- Se pueden realizar pruebas antes, durante o después.
  - *Sketchs*
  - *Wireframes*
  - Prototipos de vistas
  - Aplicación funcional
- Mientras más seguido se realicen más temprano se detectarán mejoras, por lo que serán más fáciles de implementar.
- Una buena práctica es por lo menos una vez al mes, aunque depende del contexto.

¿Con quién se prueba?

- Con personas que sean representativas de los usuarios:
  - No sirve hacer pruebas con gente que no utilizará el sistema ni está familiarizada con los conceptos.
  - Un *subset* de usuarios representativos sirve para detectar casi los mismos problemas que para todos los usuarios.
- Las pruebas serán tan buenas como los usuarios lo sean:
  - representativos
  - dispuestos a entregar *feedback*
- Se deben invertir recursos en seleccionar con cuidado a estos usuarios.

¿Con quién se prueba?

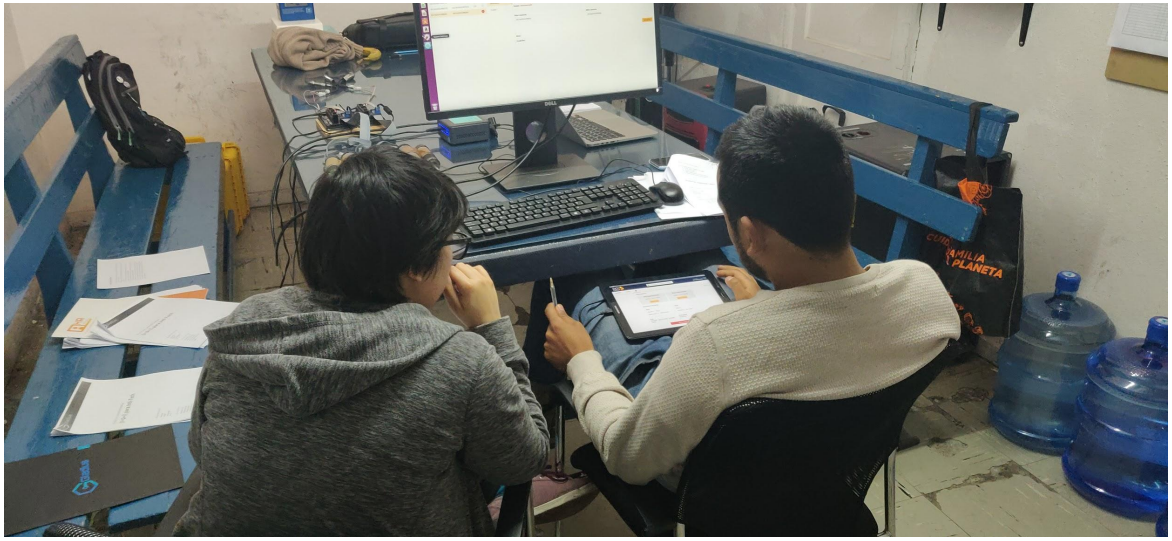
- Problema:

  ¿Cómo se definen los usuarios representativos?
- La audiencia objetiva suele ser más amplia de lo que uno imagina.
- Es mejor probar con “usuarios comunes” que esperar al “usuario modelo”.
  - Se debe ser consciente de los sesgos que estos pueden tener.
- Evitar repetir rondas de pruebas con usuarios repetidos.

¿Qué se prueba?

- Se deben listar las tareas más importantes a evaluar:
  - Las más críticas para el negocio
  - Las menos probadas
  - Las con más criticadas por los usuarios
- Luego, se deben plantear estas como escenarios para que los usuarios interactúen con el sistema.

La prueba misma



La prueba misma

- Un facilitador
  - Guía al usuario en las tareas a realizar.
  - Pregunta constantemente qué piensa y siente el usuario al interactuar.
  - Es importante mantenerse neutral para no sesgar los resultados.
- Para cada escenario:
  - Cronometrar tiempo.
  - ¿El usuario logra ejecutarlo?
  - ¿Cuál es su sensación, qué opina?

La prueba misma

- Gente observando
  - Se puede grabar la pantalla, las interacciones del usuario, donde dirigió su vista, entre otros.
  - Mientras más gente analice las pruebas más perspectivas se obtendrán del uso de la aplicación.
  - Es más fácil convencer a la organización de que un cambio es necesario si es que participaron de la prueba que lo detectó.
- *Hotjar*

Reporte de pruebas

- Listar los problemas más frecuentes y graves que se detectaron entre los usuarios.
- Diseñar opciones para solucionarlos
  - Priorizar las soluciones simples y baratas, que los usuarios decidan cuál es mejor.

A/B testing

- Ampliamente utilizado en sitios *web*.
- Se prueban dos versiones distintas de una página y se evalúa cual resulta mejor:

- Split

Alpha/Beta testing

- Pruebas de la aplicación simulando ambiente real de utilización.
- *Alpha testing*:
  - Usuarios seleccionados o desarrolladores prueban el producto en el sitio de desarrollo bajo la dirección de un encargado de las pruebas.
- *Beta testing*:
  - Número de usuarios reales controlado prueba libremente el producto en su propio ambiente de uso.

Ambiente de pruebas

- *Review apps*
- *Staging*
  - Ambiente de ejecución del sistema que trata de emular lo más posible al ambiente real (*production*)
  - La idea es probar el sistema de manera segura sin afectar los datos reales
  - Se necesita tener claras las dependencias del sistema (ej. DB / servicios externos) y aislarlas del ambiente de pruebas
    - Por ejemplo, en el ambiente de pruebas no se deberían enviar correos => ¿cómo probar correos sin enviarlos?
  - Útil para ejecutar pruebas de humo sobre *release candidates*

Proyecto: Entrega 4

Encuesta docente

PONTIFICIA UNIVERSIDAD CATÓLICA DE CHILE

Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ciencia de la Computación

Clase 21
Pruebas de validación

IIC3745 – Testing

Rodrigo Saffie

rasaffie@uc.cl

18 de noviembre de 2020