GESTIÓN Y MANEJO DEL SISTEMA DE APOYO ALIMENTARIO DE LA UNIVERSIDAD DISTRITAL

Daniel Garcia
Daniel Moreno
Edwar Diaz

29 de agosto de 2017

Índice general

Parte I

Proyecto

Capítulo 1

Problema

Problema:
Dificultad que tienen los estudiantes para aprovechar el apoyo alimentario que ofrece la Universidad Distrital Francisco José de Caldas.

Objetivos:
Crear una aplicación que permita agilizar el proceso de entrega de almuerzo Mejorar el servicio de apoyo alimentario Automatizar el proceso registro de asistencia de usuarios

Pregunta: ¿Cómo agilizar el proceso de entrega de almuerzos en la Universidad ?

Hipotesis:
Una estrategia para agilizar el proceso de entrega de almuerzos en la universidad es la construcción de un aplicativo web que permita la entrega de papeles, organización de horarios y la verificación de asistencia al apoyo alimentario.

Es necesario que a través del aplicativo se pueda organizar horarios ya que si solo se gestiona la asistencia y entrega de papeles no habra una solucion al problema a largo plazo pero si gestionan los horarios estudiantiles se pueden crear estrategias para evitar el aglutinamiento de personas a la hora de recibir el almuerzo.

Justificación:
En la Universidad Distrital Francisco José de Caldas existe un serio problema de gestión en la distribución de almuerzo a los estudiante: la lentitud en la entrega ha ocasionado que los estudiantes no puedan aprovechar el servicio sin correr el riesgo de comprometer la llegada a sus clases. Muchos estudiantes necesitan el apoyo alimentario pero el llegar tarde a clase da lugar incidentes entre profesores y alumnos lo que puede afectar, a su vez, el desempeño académico

Capítulo 2

Metodología

2.1. Manifiesto Agil

Antes de decir que es Scrum, vale la pena saber que es el manifiesto agil y de que se compone.

El manifiesto agil se compone por 5 valores y 12 Principios:

2.1.1. Valores

Valorar a las personas y las interacciones entre ellas por sobre los procesos y las herramientas

Las personas son el principal factor de éxito de un proyecto de software. Es más importante construir un buen equipo que construir el contexto. Muchas veces se comete el error de construir primero el entorno de trabajo y esperar que el equipo se adapte automáticamente. Por el contrario, la agilidad propone crear el equipo y que éste construya su propio entorno y procesos en base a sus necesidades.

Valorar el software funcionando por sobre la documentación detallada

La regla a seguir es "no producir documentos a menos que sean necesarios de forma inmediata para tomar un decisión importante". Estos documentos

deben ser cortos y centrarse en lo esencial. La documentación (diseño, especificación técnica de un sistema) no es más que un resultado intermedio y su finalidad no es dar valor en forma directa al usuario o cliente del proyecto. Medir avance en función de resultados intermedios se convierte en una simple ïlusión de progreso".

Valorar la colaboración con el cliente por sobre la negociación de contratos

Se propone que exista una interacción constante entre el cliente y el equipo de desarrollo. Esta mutua colaboración será la que dicte la marcha del proyecto y asegure su éxito.

Valorar la respuesta a los cambios por sobre el seguimiento estricto de los planes

La habilidad de responder a los cambios que puedan surgir a lo largo del proyecto (cambios en los requisitos, en la tecnología, en el equipo, etc.) determina también su éxito o fracaso. Por lo tanto, la planificación no debe ser estricta sino flexible y abierta.

2.1.2. Principios

Los valores anteriores son los pilares sobre los cuales se construyen los doce principios del Manifiesto Ágil. De estos doce principios, los dos primeros son generales y resumen gran parte del espíritu ágil del desarrollo de software, mientras que los siguientes son más específicos y orientados al proceso o al equipo de desarrollo:

1. Nuestra mayor prioridad es satisfacer al cliente a través de entregas tempranas y frecuentes de software con valor.

2. Aceptar el cambio incluso en etapas tardías del desarrollo. Los procesos ágiles aprovechan los cambios para darle al cliente ventajas competitivas.

3. Entregar software funcionando en formafrecuente , desde un par de semanas a un par de meses, prefiriendo el periodo de tiempo más corto.

4. Expertos del negocio y desarrolladores deben trabajar juntos diariamente durante la ejecución del proyecto.

5. Construir proyectos en torno a personas motivadas, generándoles el ambiente necesario, atendiendo sus necesidades y confiando en que ellos van a poder hacer el trabajo.

6. La manera más eficiente y efectiva de compartir la información dentro de un equipo de desarrollo es la conversación cara a cara.

7. El software funcionando es la principal métrica de progreso.

8. Los procesos ágiles promueven el desarrollo sostenible. Los sponsors, desarrolladores y usuarios deben poder mantener un ritmo constante indefinidamente.

9. La atención continua a la excelencia técnica y buenos diseños incrementan la agilidad.

10. La simplicidad –el arte de maximizar la cantidad de trabajo no hecho– es esencial.

11. Las mejores arquitecturas, requerimientos y diseños emergen de equipos auto-organizados.

12. A intervalos regulares, el equipo reflexiona acerca de cómo convertirse en más efectivos, luego mejora y ajusta su comportamiento adecuadamente.

2.2. Scrum

Scrum es una de las metodologia agiles mas conocidas al igual que XP, es una herramienta en la cual se tienen varios grupos de trabajo enfocados en diversas tareas con un unico objetivo, en el cual se espera generar resultados y/o entregas al cliente en cortos periodos de tiempo.

El equipo de trabajo en esta metodologia esta apoyado por 2 roles: El ScrumMaster y el Product Owner. el ScrumMaster o tambien consederado Coach es aquel que vela por que se utilice Scrum , por remover las impedimentos y da asistencia al equipo para que logre el mayor nivel de rendimiento posible. El Product Owner es quien representa al negocio, stakeholders (trabajadores, organizaciones sociales, accionistas y proveedores, entre muchos otros actores clave), cliente y usuarios finales.

2.2.1. En que se basa

Esta basada en tres pilares:

Transparencia: Todos los implicados tienen conocimiento de qué ocurre y en el proyecto y cómo ocurre. Esto hace que haya un entendimiento "común" del proyecto, una visión global.

Inspección: Los miembros del equipo Scrum frecuentemente inspeccionan el progreso para detectar posibles problemas. La inspección no es un examen diario, sino una forma de saber que el trabajo fluye y que el equipo funciona de manera auto-organizada.

Adaptación: Cuando hay algo que cambiar, el equipo se ajusta para conseguir el objetivo del sprint. Esta es la clave para conseguir éxito en proyectos complejos, donde los requisitos son cambiantes o poco definidos y en donde la adaptación, la innovación, la complejidad y flexibilidad son fundamentales.

Parte II

UML

Capítulo 3

Análisis

3.1. Introducción

3.2. Diagrama de Casos de Uso

El diagrama de casos de uso es una forma de representar los requerimientos de un sistema, cada caso de uso reune una serie de requisitos basandose en diferemtes funciones o tareas.

Actor: Es una agrupacion uniforme de personas, sistemas o maquinas que interactuan con el sistema que estamos construyendo. Caso de Uso: Son secuencias de interacciones entre actores y un sistema que usan un servicio.

Cuadro 3.1: Caso de uso 1

Nombre	Recibir apoyo alimentario
Actores	Estudiantes
Escenario	
Primario	Recibir el apoyo alimentario
Secundario	no coinciden horarios,
Excepciones	no estar registrado, no es estudiante, se acabo la comida

Cuadro 3.2: Caso de uso 2

Nombre	Realizar Inscripcion
Actores	Estudiante
Escenario	
Primario	Inscribiruna solicitud al apoyo alimentario
Secundario	Olvido datos de usuario
Excepciones	No es estudiante o es un egresado, tener el semestre aplazado, error con el servidor, error con el cliente No hay convocatoria disponible

Cuadro 3.3: Caso de uso 3

Nombre	Realizar Inscripcion
Actores	Cordinador Beinestar
Escenario	
Primario	Generar convocatoria apoyo alimentario
Secundario	Olvido datos de usuario
Excepciones	No hay acceso a la plataforma, No tiene los permisos necesarios, error con el servidor, error con el cliente

3.3. Interacciones

3.3.1. Diagrama de Secuencia

3.3.2. Diagrama de Comunicación

3.3.3. Diagrama de Temporización

3.4. Diagramas de Actividades

3.5. Diagramas de Actividades

3.6. Diagramas de Workflow

3.7. Diagramas de Descripción de la Interacción

Capítulo 4

Diseño

4.1. Introducción

4.2. Diagrama de Clases de Análisis

4.3. Diagrama de Clases

4.4. Diagrama de Objetos

4.5. Diagrama de Estructura Compuesta

Capítulo 5

Despliegue

5.1. Introducción

5.2. Diagrama de Sistemas

5.3. Diagrama de Componentes

5.4. Diagrama de Artefactos

5.5. Diagrama de Nodos

Parte III

Conclusiones

Capítulo 6

Conlusiones

Capítulo 7

Trabajos Futuros

Anexos

Bibliografía

[1] S Bolanos. *Metaproceso de Software*.

[2] J. B. Castro and R. G. Crespo. A software architecture proposal artistic engineering environment -aee-sandro. In *2012 Workshop on Engineering Applications*, pages 1–6, May 2012.