Requisitos básicos del sistema Raspberry Pi a construir

Diego Martín Arroyo

15 de marzo de 2015

Resumen

Resumen de los aspectos a considerar en el desarrollo del Trabajo de Fin de Grado

1. Histórico de cambios

2. Introducción

El presente documento recoge los diferentes aspectos a valorar en el modelado del sistema a construir teniendo en cuenta diferentes criterios que se detallarán más adelante.

Este documento no tiene como objetivo establecer de forma definitiva la arquitectura y aspectos a considerar del sistema, si no que su principal uso es la elicitación de diferentes decisiones de diseño y el uso del mismo para exponer las ideas reflejadas a terceros como tutores o colaboradores.

La técnica utilizada en la versión actual se basa remotamente en [1], si bien la influencia de dicha publicación será mayor en posteriores versiones.

2.1. Objetivos del proyecto

Este proyecto cuenta con varios objetivos muy diferentes entre sí, que se agrupan en tres categorías:

- Arquitectura subyacente
  Definición de los componentes hardware a utilizar en el sistema, interconexión de los mismos, soluciones de alimentación eléctrica, almacenamiento....
- Servicios a proveer
- Componente didáctico

3. Definiciones[1]

3.1. Definición del dominio del problema

El sistema se ubica en una Facultad universitaria con aproximadamente 600 alumnos 0 con varias asignaturas en las que se imparten áreas de conocimiento relacionados con la Computación Distribuida, en particular las asignaturas **Arquitectura de Computadores** y **Sistemas Distribuidos** [2].

[1] Esta parte abarca únicamente el componente didáctico del sistema debido a que no se cuenta con ningún tipo de trasfondo para el resto de partes del dominio del problema.

3.2. Modelado del sistema actual

La Facultad cuenta con varias aulas y laboratorios informáticos donde los alumnos disponen de la intraestructura necesaria para realizar los ejercicios y prácticas asignadas. Dichos espacios permiten utilizar cualquier equipo como nodos, pues pertenecen a la misma red, siendo incluso factible la comunicación directa entre equipos situados en diferentes aulas o edificios. La conexión es relativamente rápida, contando con un cableado capaz de soportar teóricamente transferencias de hasta 100Mb/s *full-duplex*. La gestión de usuarios se realiza mediante un protocolo LDAP (*Lightweight Directory Access Protocol*) [3], contando con un sistema de ficheros centralizado que permite acceder a la información de un usuario desde cualquier equipo, facilitando las tareas de replicación menos sofisticadas.

La mayoría de las prácticas son programadas en el lenguaje **Java**, que es ya conocido por la totalidad de los estudiantes gracias a asignaturas previamente cursadas y que facilita el despliegue y la compatibilidad entre diferentes equipos de trabajo sustancialmente.

Problemas conocidos Estos son varios de los problemas identificados en los diferentes usuarios de la infraestructura:

- Cada pareja de alumnos necesita tres estaciones de trabajo para poder realizar algunos de losejercicios propuestos.
- El servidor LDAP constituye un "cuello de botella", pues todos los alumnos acceden a él de forma intensiva, provocando caídas en el mismo.
- Las técnicas de programación utilizadas hasta la fecha tienen un rendimiento bajo y son en ocasiones relativamente complejas.

3.3. Identificación de usuarios participantes

- Estudiantes de tercero y cuarto curso del Grado en Ingeniería Informática
- Doctentes de las asignaturas Arquitectura de Computadores y Sistemas Distribuidos
- Administradores del sistema

4. Identificación de las necesidades de cada parte

4.1. Necesidades de los alumnos

- Entorno de trabajo útil y sencillo.
- Posibilidad de observar los resultados de las ejecuciones de forma sencilla.

4.2. Docentes

- Entorno versátil sobre el cual puedan llevarse a cabo **todas** las prácticas y ejercicios propuestos, aportando si es posible algún tipo de ventaja sobre el sistema en uso.

4.3. Administrador

- Sistema integrable en la intraestructura actual cuyo mantenimiento sea sencillo.

5. Propuestas para la búsqueda de necesidades

- Encuestas o entrevistas a todas las partes.
- Evaluación de la experiencia de uso en las diferentes etapas de desarrollo del sistema.

6. Identificación de requisitos

6.1. Requisitos de almacenamiento de la información

- Gestión de usuarios (credenciales de autenticación)
- Gestión de los datos de cada usuario
- *Logs* del sistema

6.2. Identificación de requisitos funcionales

6.3. Identificación de actores

6.4. Identificación de requisitos no funcionales

- El *software* debe ser mantenible y robusto[2].
- Reducción de los costes de desarrollo.
- Definición de los protocolos de comunicación.
- Definición de los protocolos de seguridad y confidencialidad.
- Definición de la interacción con el usuario.
- Integridad del sistema y fiabilidad (*uptime*, recuperación frente a fallos).
- Productos a crear.
- Compatibilidad con las prácticas y ejercicios.

Referencias

[1] A. D. Toro, B. B. Jimenez, A. R. Cortes, and M. T. Bonilla, "A requirements elicitation approach based in templates and patterns," 1999.

[2] U. de Salamanca, "Titulación y programa formativo - grado en ingeniería informática." `http://http://www.usal.es/webusal/files/Grado_en_Ingenieria_Informatica_2014_1%C2%AA%20parte-actualizado%202-10-14.pdf`, 10 2014.

[3] N. W. Group, "Comment on rfc 4516 - lightweight directory access protocol (ldap)," *RFC*, June 2006.

[2]Siendo dicha robustez garantizada mediante el uso de *software* utilizado por una base de usuarios significativa, una arquitectura conocida, pruebas realizadas sobre él o un equipo de desarrollo en activo, entre otras