Guía para la Documentación de Arquitecturas de Software Como Base Para el Desarrollo de Sistemas de Información en la Iglesia Adventista del Séptimo Día

Jazmín Hernández
jazminpalom@gmail.com

Technical Report COMP-029-2009

Abstract

Al diseñar una nueva arquitectura de software es necesario especificar puntualmente cada aspecto del mismo; por lo tanto, contar con una guía o plantilla para documentar la arquitectura de software usada en el desarrollo de sistemas de información es muy importante ya que ayuda a agilizar esta fase, además de mantener la consistencia y la facilidad de mantenimiento entre las distintas arquitecturas que se usen. Tomando en consideración que actualmente no existe en la Iglesia Adventista del Séptimo Día (IASD) ninguna pauta que especifique la forma de documentar arquitecturas de software se propone una guía a seguir durante la fase de documentación de las arquitecturas de software en la IASD.

INTRODUCCION

Con el avance de los paradigmas, estilos arquitectónicos y filosofías en el desarrollo de software, también se va desarrollada la IASD, y se ha visto la necesidad de establecer marcos teóricos y prácticos para el mejor desempeño de la misma; entre estos avances se encuentran las mejores prácticas de desarrollo de software. Estas prácticas promueven las arquitecturas de software que proveen un marco, guías y políticas para el desarrollo de sistemas de software desde alto nivel, ensamblando elementos arquitectónicos de manera que se logren satisfacer requerimientos funcionales y no funcionales para los sistemas.

De acuerdo a la explicación de un autor, una arquitectura de sistema define la "estructura" básica del sistema, por ejemplo, los módulos de nivel más alto que comprenden las mayores funcionalidades del sistema la administración y distribución de los datos, el tipo y estilo de su interfaz de usuario, sobre qué tipo de plataformas correrá y así sucesivamente [1].

Además de esta definición existen otras más que surgen de acuerdo a distintos puntos de vista mencionado en cierta publicación [2] las cuales definen a una arquitectura de software como un diseño de alto nivel, la totalidad de la estructura de un sistema, la estructura de los componentes de un programa o sistema, sus interrelaciones y los principios o guías que gobiernan su diseño y evolución a través del tiempo, es componentes y conectores, y en general, en la esencia de estas definiciones se encuentran todas aquellas cuestiones estructurales del sistema.

Cabe señalar también que la propuesta estará basada en la experiencia y argumentos de arquitectos profesionales indicando el propósito de la documentación, qué documentar y, en general, cómo realizar una buena documentación de cualquier arquitectura de software.

El presente artículo se divide en tres secciones, la conclusión y los trabajos futuros: En la primera sección, importancia de la documentación de una arquitectura de software, se explica por qué se debe documentar una arquitectura de software, categorizando las razones y la importancia de hacerlo. En la segunda sección, documentación de una arquitectura de software, se explican las reglas que debe cumplir la documentación de las arquitecturas de software para su fácil lectura, comprensión y adopción. En la tercera sección, definición de la plantilla de documentación de arquitecturas de software para la IASD, sea hace una descripción de la plantilla que se propondrá detallando la información que debiera contener.

IMPORTANCIA DE LA DOCUMENTACIÓN DE UNA ARQUITECTURA DE SOFTWARE

La documentación de una arquitectura puede ser usada para distintos propósitos a corto, mediano o largo plazo, los cuales se pueden agrupar en tres propósitos globales de acuerdo a algunos autores [3]:

1. Como medio de educación. Ya que comunicar la arquitectura a los interesados es tan importante como crearla, la documentación de una arquitectura de software puede usarse para explicar el sistema a otras personas como pueden ser miembros del equipo, analistas externos o nuevos arquitectos. Esto parte de la premisa de
2. Como un vehículo de comunicación entre los interesados, ya que define de qué forma se llevará a cabo la comunicación entre los interesados del proyecto.
3. Como base para el análisis del sistema. La documentación de una arquitectura de software puede usarse para dar soporte al análisis del sistema; de manera particular a ingenieros de rendimiento y durante el diseño del sistema para proveer de características de calidad, para analizar arquitecturas alternativas, desarrollo de nuevos sistemas a partir de sistemas legados, planeación y presupuesto de otras etapas de desarrollo, especificación de líneas o familia de producto, etc.

REGLAS PARA DOCUMENTAR UNA ARQUITECTURA DE SOFTWARE

Para realizar una buena documentación, según ciertos autores [3], se deben seguir siete sencillas reglas que hace la diferencia entre una buena o mala documentación. Estas reglas son:

1. La documentación debe ser escrita desde el punto de vista del lector, no del escritor: La documentación debe ser escrita una sola vez de manera eficiente, cortés y clara pensando en las personas para las cuales se está escribiendo de manera que estas aprecien el esfuerzo.
2. Evitar repeticiones innecesarias: Toda información debe ser escrita en un solo lugar dependiendo de su tipo para hacerlo más fácil de usar y actualizar, evitando confusiones al usuario.

3. Evitar ambigüedades: Una documentación es totalmente útil cuando elimina la gran cantidad de detalles que se necesitan resolver para poder liberar un sistema. Y debe hacerlo evitando que ocurran múltiples interpretaciones, de las cuáles varias pueden ser incorrectas. Para lograrlo se debe explicar claramente las notaciones que se hagan.
4. Usar una organización estándar: Establecer una organización planeada y estandarizada de la documentación ayuda al lector a navegar más fácilmente a través de él y mejora el entendimiento del mismo; además de hacer que se pueda referenciar fácilmente para su estudio.
5. Registro razonado: Se deben documentar las rutas alternativas rechazadas y su explicación después de haber tomado una decisión para poder usarlos en futuros interrogatorios acerca de tales rutas.
6. Mantener la documentación actualizada, pero no demasiado actualizada. Se debe mantener actualizado la documentación con los cambios hechos a través del tiempo para proyectar de esta forma la realidad y obedecer a la regla de consistencia interna de la documentación.
7. Revisar la documentación para conveniencia de propósito: Se debe hacer una revisión de la documentación y liberar aquellos cambios que sean aceptados por la comunidad de lectores a la que esté dirigida. Y solo aquellos usuarios para los cuáles fue escrito podrán decir si la información está representada de la forma correcta.

DEFINICIÓN DE LA PLANTILLA DE DOCUMENTACIÓN DE ARQUITECTURAS DE SOFTWARE PARA LA IASD

A continuación se detallará la plantilla propuesta para la documentación de las arquitecturas de software en la IASD. La plantilla se dividirá en temas y subtemas en forma ordenada y secuencial, y cada una de ellas es detallada con su definición dentro del contexto de la arquitectura de software.

Página de título: La página de título deberá tener como encabezado la frase Documento de Arquitectura de Software, seguido del nombre del proyecto, los realizadores y la empresa o institución a la que pertenece la documentación, el logotipo del proyecto o de la institución a la que pertenece el proyecto y el mes y año e su publicación. Es importante tener al menos los primeros cuatro datos y la fecha de publicación para poder referenciar correctamente la documentación en trabajaos futuros o para actualizaciones a través del tiempo.

Historia de revisión: Las historias de revisión deberán estar en formato de tabla para su mejor lectura, presentando los siguientes datos:

- Fecha: es la fecha de publicación de la versión del documento
- Versión: el número de versión del documento
- Descripción: el nombre del documento en caso de haber cambiado gradualmente y los cambios más generales que hayan ocurrido en esa versión.

- Autor: el nombre(s) de quienes intervinieron en la realización de esa versión de la documentación.

Tabla de contenido: esta presenta una lista de las partes de conforman la documentación organizado de acuerdo al orden en el que aparecen los temas. La tabla de contenido incluirá los nombres de los temas o la descripción del primer tema a nivel de cabecera. Pero, para efectos de facilidad de estudio y referencia, se recomienda que se incluyan la mayor parte de temas y subtemas.

Lista de figuras: en este apartado se incluirá un sumario de todas las figuras usadas en el documento comenzando cada figura con la palabra *Figura* y su número o número de sección-número de figura, su descripción y la página en la que se encuentra dentro del documento.

1. Alcance: aquí se declararán los objetivos específicos del documento, así como el público al que está dirigido y en dónde es aplicable dicho documento. Esta sección es muy importante ya que delimita la información contenida en la documentación permitiendo así que tanto el autor como el lector estén enfocados en los objetivos y la orientación de la misma.

2. Referencias: Las referencias son las referencias se mostrarán en el orden en el que aparecen en la documentación conteniendo un formato simple, del tipo IEEE en el caso de ser libros o en el caso de ser un documento de la organización o de alguna otra, se debe especificar el título, número de reporte si aplica, fecha y la organización que lo publicó. Las referencias servirán para permitir al lector hallar una fuente de información en caso de no entender algún concepto o definición comentado en el documento; además, el uso de una estructura estandarizada de las referencias hará su lectura más fácil y acorde con otras revisiones o documentaciones de arquitectura pertenecientes a la misma institución.

3. Arquitectura de software: en esta sección se detalla de forma general la representación arquitectónica haciendo énfasis en los principios arquitectónicos que se estén usando en la definición de la arquitectura del software listando también las vistas que contendrá, así como el tipo de notaciones, diagramas y herramientas que usará en cada vista; y si el autor lo prefiere, también podrá añadir la definición de los tipos de vistas que contendrá la documentación. Es importante que el lector tenga en mente el tipo de diagramas que hallará en cada representación de vistas y los principios arquitectónicos que deberá seguir para estar preparado para su lectura y comprensión

4. Metas y restricciones arquitectónicas: ya que el software existirá en un contexto del mundo real y el mundo real tiene restricciones; esta parte deberá contener los requerimientos principales, que también pueden ser nombrados como requerimientos especiales dada su especial importancia para la arquitectura, de forma clara y cómo estos requerimientos afectan directamente a la arquitectura; y también contendrá restricciones de sistema que tendrá la arquitectura que se está

documentando. En las restricciones arquitectónicas pueden mencionarse el tipo de modelo de diseño en la que estará basado el software, así como los requerimientos de diseño con los que deberá cumplir. Entre las restricciones que podrían mencionarse se encuentran las tecnologías aprobadas para ser usadas en la arquitectura, estándares de la institución, estándares públicos requeridos, formato de mensajes, perfil de habilidades y conocimientos del equipo de desarrollo, plazos establecidos y la naturaleza del proyecto [4].

5. **Arquitectura lógica:** la sección de arquitectura lógica explicará este ámbito de la arquitectura. Aquí se explicarán los requisitos funcionales. Se diagramará la descomposición del sistema en una serie de abstracciones clave aplicando los principios definidos en la sección 3 (Arquitectura de software). La arquitectura lógica es importante por su uso en el análisis funcional de la arquitectura. También se define el tipo de notación utilizado por el autor, como bien pueden ser diagramas de clases. Si se usa UML, la arquitectura lógica puede representarse con diagramas de clases, de objetos, de interacción, estados y actividades [5].

6. **Arquitectura de procesos:** aquí se mostrará cómo funciona la arquitectura desde un nivel más alto, así como la interacción entre los pasos que siguen los procesos para unirse. La documentación de la arquitectura de procesos se enfocará en temas de concurrencia y distribución, integridad del sistema y de tolerancia a fallas. De forma general, se muestra la descomposición en procesos de la arquitectura, considerando los requerimientos no funcionales. Se detallan los grupos de tareas llevados a cabo por procesos. Si se usa UML, la arquitectura de procesos puede ser representada con diagramas de clases, objetos, de interacción, de estados y actividades.

7. **Arquitectura de desarrollo:** aquí se documentará la organización de los módulos del software en un ambiente de desarrollo como una jerarquía de niveles, mostrando el empaquetamiento que se realiza formando bibliotecas o subsistemas. Si se usa UML, la vista de procesos puede ser representado con diagramas componentes, interacción, de estados y de actividades. Si se usa UML, la arquitectura de desarrollo puede ser representada con diagramas de componentes, de interacción, de estados y actividades.

8. **Arquitectura física:** en esta sección se detalla cómo será desplegado el software en la infraestructura física. En otras palabras, se documentará cómo se mapea el software al hardware. Para estor se toman en cuenta principalmente los requisitos no funcionales entre los que se encuentran la disponibilidad, confiabilidad, desempeño y escalabilidad. Por lo tanto, aquí se mapean las redes, procesos, tareas y objetos a nodos de procesamiento [6]. Si se usa UML, la arquitectura física puede ser representada con diagramas de componentes, de interacción, de estados y actividades.

9. **Escenarios:** aquí el autor deberá documentar los son de alguna manera una abstracción de los requisitos más importantes. Aquí es donde se representará la conjunción de

todas las vistas en determinadas situaciones y casos de uso. Su diseño se expresa mediante el uso de diagramas de escenarios y diagramas de interacción de objetos.

10. Tamaño y desempeño: en esta parte se documentarán las características principales de que impactan la arquitectura así como los objetivos de las restricciones de desempeño. Esta sección es opcional, si hay pocos o ningún dato persistente, o las traducciones entre el modelo de diseño y el modelo de datos es trivial.

11. Calidad: se detallará cómo la arquitectura del sistema contribuye a todas las capacidades (distintas de la funcionalidad) del sistema: extensibilidad, fiabilidad, portabilidad, disponibilidad, usabilidad, interoperabilidad, y demás que apliquen. Si las características tienen un significado especial, por ejemplo, el salvado, o implicaciones de seguridad o privacidad, estas deben ser claramente delineadas.

Apéndices:

A. Acrónimos y abreviaciones: el autor deberá proporcionar los acrónimos y abreviaturas requeridas para interpretar correctamente el documento de la arquitectura del sistema; esta información puede proporcionarse por la referencia en un glosario del proyecto.

B. Definiciones: Esta subsección debe proporcionar las definiciones de los términos usados por el autor de forma que pueda interpretar correctamente el documento de la arquitectura del sistema.

CONCLUSIONES

Este trabajo ha sido únicamente la propuesta de una plantilla para arquitecturas de software para los departamentos de desarrollo de la IASD; sin embargo, es posible hallar otras alternativas viables para el mismo objetivo. Pero como se ha podido notar, la propuesta presenta una forma muy sencilla de documentar arquitecturas de software.

TRABAJOS FUTUROS

Tomando como base esta propuesta se pueden realizar algunos trabajos más como pueden ser:

1. Desarrollo de una aplicación que permita a los autores de una arquitectura documentarla de tal forma que la aplicación genere un archivo estándar y bien estilizado, ahorrando tiempo y evitando errores.
2. Investigación de la efectividad de la plantilla propuesta en los centros de desarrollo de la IASD.
3. Evaluación de la plantilla mediante casos de uso en situaciones y proyectos distintos.

4. Evaluación de la plantilla mediante encuestas a centros de desarrollo de la IASD que hayan hecho uso de la plantilla propuesta.
5. Mejoramiento y optimización de la plantilla para hacer más eficiente la documentación de arquitecturas de software.

Se espera que los planteamientos aquí presentados puedan verificarse con el tiempo; y como estos posibles trabajos futuros pueden surgir otras ideas no planteadas en este documento, ideas que muy probablemente se presentarán como la solución en casos, situaciones y proyectos específicos.

BIBLIOGRAFÍA

[1] Beyond Software Arquitecture, Luke Hohmann, Addison-Wesley, 2003.

[2] Software Architecture in Practice, Second Edition, Len Bass, Paul Clements, Rick Kazman, Addison-Wesley, 2003.

[3] Documenting Software Architectures, Paul Clements, Felix Bachmann, Len Bass, David Garlan, James Ivers, Reed Little, Robert Nord, Judy Stafford, Addison-Wesley, 2002.

[4] Simon Brown, Software Architecture Document Guidelines, disponible vía Web en el sitio http://twurl.nl/tpn3ue, visitado el 6 de octubre de 2009.

[5] Architectural Blueprints—The "4+1" View Model of Software Architecture, Philippe Kruchten, IEEE Software, November 1995, pp. 42-50.

[6] Large-Scale Software Architecture, A Practical Guide using UML, Jeff Garland, Richard Anthony, Ed. Wiley, 2003.