Plan de Trabajo Final
Carrera Ingeniería de Sistemas

Facultad de Ciencias Exactas – UNICEN

Tema: Análisis de Información de Documentos de Arquitectura de Software en base a Procesamiento de Lenguaje Natural

Alumno/s: Marcos Basso, Eduardo Ismael Solís

Director: Dr. Andres Diaz Pace

Codirector: Mg. Alejandro Rago

1. Introducción

La arquitectura de software de un programa o sistema de computación es la estructura o estructuras del sistema, que comprenden elementos de software, las propiedades externamente visibles de esos elementos y las relaciones entre ellos [1]. Otra definición interesante considera que la arquitectura es, a grandes rasgos, una vista del sistema que incluye los componentes principales del mismo, el comportamiento de esos componentes según se la percibe desde el resto del sistema, y las formas en que los componentes interactúan y se coordinan para alcanzar la misión del sistema [2].

Esto significa que la arquitectura de software comprende el diseño de alto nivel de una solución. Como en toda disciplina que involucra diseño, es esencial contar con una correcta documentación de la misma, ya que dicho artefacto servirá para involucrar a los distintos stakeholders durante el ciclo de vida del proyecto y para guiar la implementación del sistema. De aquí en adelante se hará mención a la sigla SAD para denotar el Documento de Arquitecturas (o Software Architecture Document), que es el artefacto donde comúnmente se captura la información arquitectural de un proyecto.

Un SAD sirve para comunicar las diferentes facetas de la arquitectura a los stakeholders de forma tal de que puedan realizar sus trabajos apropiadamente [12]. Los SAD deben estar escritos utilizando un lenguaje claro y sin ambiguedades. Además, los SAD suelen estar organizados en secciones para que los lectores puedan buscar rápidamente información relevante.

2. Motivación

Generalmente, un SAD tiene un volumen considerable de información, proporcional al tamaño y/o complejidad del sistema al cual describe. Cada stakeholder puede ser responsable o estar afectado por un área específica del sistema, por lo que centrará su atención en una determinada parte de la arquitectura y por ende de su correspondiente documentación.

Los stakeholders interesados en un SAD y sus propósitos al utilizar este documento son diversos [12]. Por ejemplo, un arquitecto puede revisar el SAD para negociar tradeoffs, un programador para asegurar que se respeten las decisiones de diseño, un tester para crear pruebas de sistema que validen los atributos de calidad, o un administrador de redes para determinar si el sistema soporta la carga de operación, entre otros. Adicionalmente, una organización de desarrollo de software podría tener la necesidad de definir un template interno para documentar un SAD, y utilizar para ello distintos formatos: Wikis, documentos de texto, fotografías, etc.

Consecuentemente, cuando los stakeholders deben inspeccionar un SAD para informarse antes de realizar una tarea pueden llegar a tener dificultades. Esto ocurre principalmente porque existen diferentes formatos para documentar la arquitectura y la estructuración utilizada para organizar la información suele cambiar de empresa a empresa. En este contexto, las herramientas que faciliten el acceso a información de interés para un stakeholder particular podrían ser de gran ayuda para reducir los tiempos de búsqueda y filtrar partes del SAD que no son relevantes.

3. Objetivos

El objetivo principal de esta propuesta de tesis es construir una herramienta que brinde soporte a un arquitecto o analista en a las actividades de comprensión de SADs independientemente de su formato o estructura. De este objetivo principal se desprenden tres objetivos específicos. El primer objetivo consiste en analizar diferentes formatos de SAD, especialmente de aquellos codificados en archivos PDF o en una Wiki compuesta por páginas Web. El segundo objetivo es definir estrategias para procesar la información textual, identificando determinadas secciones de interés, abstracciones y conceptos importantes de los SADs que luego podran ser consultados. Por último, el tercer objetivo es proveer un mecanismo que facilite la búsqueda de información relevante contenida dentro de los SADs.

Para lograr estos objetivos, se propone el desarrollo de una herramienta denominada **SADAnalyzer (Software Architecture Document Analyzer).** SADAnalyzer pretende orientar a un arquitecto o analista y facilitarle su trabajo a la hora de tomar decisiones en la construcción de un sistema, basándose en el procesamiento de SADs mediante técnicas de

procesamiento de lenguaje natural. Para ello, se prevé un desarrollo estructurado en 3 elementos fundamentales: a saber: un módulo de lector de documentos, un módulo de procesamiento de texto, y un módulo de consulta.

El ***Módulo lector de documentos*** será responsable de leer la documentación proveniente de las distintas fuentes de información (Wiki, PDF) y generar un modelo bien definido del contenido del documento seleccionado independientemente de la estructura y del origen de la información. Es importante destacar la independencia entre el origen de datos y el modelo generado, de manera que el procesamiento de los datos no quede ligado al formato original.

El ***Módulo de Procesamiento del SAD*** tiene la función de procesar el texto obtenido por el Módulo lector. Para ello, se tomará como punto de partida la herramienta *REAssistant*[i], que se enfoca en la identificación de crosscutting concerns en las especificaciones de casos de uso [4, 5]. SADAnalyzer adaptará el pipeline de NLP en inglés implementado en REAssistant, adicionando un pipeline en español [4, 6, 7, 8]. De esta manera se amplía el espectro de documentos a procesar. Cada una de las técnicas NLP generará *anotaciones* de manera de estructurar el documento de entrada con la arquitectura UIMA [9] para su posterior consulta.

El ***Módulo de Consulta*** tiene como propósito asistir al usuario de la herramienta, es decir, cualquier stakeholder interesado en algún aspecto del SAD, en la obtención de información relevante contenida en el SAD procesado. Para efectuar las búsquedas, se trabajará en la definición de scripts de consulta con UIMA RUTA [10, 11] con los que se exploraran los conceptos y abstracciones obtenidos con el Módulo de Procesamiento. La herramienta contará con scripts de consulta que se ajusten a las preferencias de cada stakeholder, como por ejemplo una consulta para encontrar secciones del SAD que discutan aspectos del sistema relacionados con el atributo de calidad *Disponibilidad* para un stakeholder del tipo *Manager*.

4. Cronograma de actividades

Se planifica realizar el trabajo final en aproximadamente 8 meses, las actividades y su respectiva duración aproximada están enumeradas a continuación.

- Análisis y relevamiento bibliográfico de variantes de documentación de arquitecturas de software. Tiempo estimado: 30 días.
- Desarrollo del módulo lector de SADs a partir de PDFs y parsing de HTML. Tiempo estimado: 15 días.
- Análisis de técnicas de procesamiento de texto NLP y de la implementación preexistente en la herramienta REAssistant. Tiempo estimado: 30 días.

- Análisis de taxonomías de atributos de calidad, tácticas de diseño y otros conceptos relevantes para la búsqueda en los SAD. Tiempo Estimado: 15 días.
- Desarrollo de pipeline de NLP en español e inglés para el procesamiento de los SAD y el reconocimiento de conceptos. Tiempo estimado: 30 días.
- Desarrollo de la herramienta SADAnalyzer como un plugin de Eclipse. Tiempo estimado: 30 días
- Análisis y desarrollo del módulo de consultas con el lenguaje de scripting RUTA. Tiempo Estimado: 15 días.
- Codificación de un conjunto de scripts de consultas de RUTA personalizadas para diferentes tipos de stakeholders. Tiempo Estimado: 15 días.
- Evaluación de la herramienta SADAnalyzer con casos de estudio. Tiempo Estimado: 30 días.
- Escritura del informe final de tesis (en paralelo con algunas actividades anteriores). Tiempo estimado: 60 días.

5. Bibliografía

[1] Len Bass et al. “Software Architecture in Practice (Third Edition)”. Addison Wesley. 2012. ISBN 978-0321815736

[2] Paul Clements. “A Survey of Architecture Description Languages”. Proceedings of the International Workshop on Software Specification and Design. Alemania, 1996. DOI 10.1109/IWSSD.1996.501143

[3] David Garlan. “Software Architecture: A Roadmap”. The future of software engineering. ACM Press. 2000. DOI 10.1145/336512.336537

[4] Alejandro Rago et al. “Assisting Requirements' Analysts to Find Latent Concerns with REAssistant”. Automated Software Engineering. 2014. DOI 10.1007/s10515-014-0156-0

[5] https://code.google.com/p/reassistant/

[6] http://opennlp.sourceforge.net/projects.html

[7] http://nlp.stanford.edu/software/corenlp.shtml

[8] http://code.google.com/p/mate-tools/

[9] http://uima.apache.org/

[10] http://uima.apache.org/d/ruta-current/tools.ruta.book.pdf

[11] http://uima.apache.org/downloads/gscl2013/2013-GSCL-Ruta.pdf

[12] Paul Clements et al. “Documenting Software Architectures: Views and Beyond (2nd Edition)”. Addison-Wesley Professional. 2010. ISBN 978-0321552686.

[i] https://code.google.com/p/reassistant/

Prof. Dr. J. Andrés Díaz-Pace

Alejandro Rago