Universidad Carlos III de Madrid
www.uc3m.es

TESIS DOCTORAL

Metodología orientada a la optimización automática de la calidad de los requisitos

Autor:

Eugenio Parra Corredor

Director/es:

Valentín Moreno Pelayo

Anabel Fraga Vázquez

Tutor:

Valentín Moreno Pelayo

DEPARTAMENTO DE INFORMÁTICA

Leganés, agosto 2016

Universidad Carlos III de Madrid
www.uc3m.es

TESIS DOCTORAL

Metodología orientada a la optimización automática de la calidad de los requisitos

Autor: Eugenio Parra Corredor

Director/es: Valentín Moreno Pelayo
Anabel Fraga Vázquez

Firma del Tribunal Calificador:

Firma

Presidente:

Vocal:

Secretario:

Calificación:

Leganés, de

A mi madre

RESUMEN

Las fases iniciales en los proyectos software marcan su desarrollo y resultado final. Defectos provocados en las fases iniciales afectan considerablemente a la calidad y alteran las fechas de finalización. Las organizaciones internacionales se han hecho eco de este problema y se dedican gran cantidad de esfuerzos en investigación para mejorar la calidad en las primeras etapas del desarrollo. Con esta iniciativa surge la ingeniería de requisitos, disciplina encargada de proporcionar procesos de ingeniería en el desarrollo de especificaciones de requisitos necesarias para definir proyectos con cierta complejidad. Por ello han surgido numerosas guías y estándares para asegurar la calidad de los requisitos que componen las especificaciones, evitando así que posibles defectos en los requisitos provoquen errores en el desarrollo y en el producto final.

Una de las mayores dificultades relacionadas con la calidad en las especificaciones de requisitos es su dependencia a las exigencias de los distintos proyectos, y a las restricciones impuestas por los distintos dominios. En esta tesis se presenta una metodología que permite incluir las restricciones impuestas mediante el procesamiento de corpus de requisitos clasificados en función de su calidad por expertos del proyecto y del dominio. El objetivo de la metodología es proporcionar métodos automáticos para la optimización de la calidad en los requisitos de ingeniería. Para ello se propone un proceso para desarrollar un clasificador que permita emular la estimación de la calidad que otorgaría el experto del dominio a un requisito, un sistema de asesoramiento automático para mejorar la calidad de requisitos defectuosos y un método para la generación automática de patrones sintáctico-semánticos, que puedan ser empleados como guía en la redacción de nuevos requisitos asegurando así una composición estructuralmente correcta.

Con el fin de corroborar las propuestas de la investigación, se presentan casos de estudio mediante el tratamiento de un corpus de requisitos proporcionado por el Grupo de Trabajo de la organización INCOSE (“INCOSE (International Council on Systems Engineering)” 2016) y se analizan los resultados obtenidos.

AGRADECIMIENTOS

Quiero dejar reflejado el agradecimiento a todas las personas que han influido en el desarrollo de este trabajo:

- Agradecer a mis tutores su dedicación y apoyo, por transmitirme su sabiduría y su saber hacer, sabiendo que el camino acaba de empezar seguirán siendo modelos de referencia.
- A todo el equipo Knowledge Reuse, grandes profesionales siempre dispuestos a prestar su ayuda desinteresadamente, siendo el valor más destacado su calidad humana, me siento agradecido por trabajar a vuestro lado.
- A mis amigos, por ser un pilar donde apoyarse. No es sencillo encontrar gente tan grande, agradezco la suerte que he tenido.
- Y por supuesto a mi familia, sin *ella* habría sido imposible culminar este trabajo. En los momentos difíciles y en los momentos buenos, siempre hemos ido juntos y seguiremos yendo.

TABLA DE CONTENIDOS

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

CAPÍTULO 1: INTRODUCCIÓN

1.1 MOTIVACIÓN

El desarrollo de proyectos con una significativa complejidad debe contar con procesos de Ingeniería de Requisitos de alta calidad. La ingeniería de requisitos es un "proceso sistemático de desarrollo de requisitos a través de un proceso cooperativo e iterativo de analizar el problema, de documentación, resultado de la observación en una variedad de formatos de representación, y de comprobar la exactitud del conocimiento adquirido" (Loucopoulos and Karakostas 1995). Requisitos de baja calidad pueden provocar errores en el desarrollo del proyecto, siendo considerados los más costosos de corregir si no se detectan a tiempo.

Dependiendo del proyecto, pueden existir sistemas que necesiten una gran cantidad de requisitos, y su especificación involucrar distintos roles como: usuarios o clientes, ingenieros de diseño, equipos de desarrollo, etc. Por este motivo, uno de los enfoques en la ingeniería de requisitos es establecer estándares y guías para facilitar la especificación y mejorar la calidad. En (Génova et al. 2013) los autores “sintetizan los diferentes indicadores cuantitativos de las propiedades de calidad deseadas en los requisitos”; estas propiedades son: verificabilidad, validez, modificabilidad, consistencia, completitud, inambigüedad, comprensibilidad, trazabilidad, abstracción, precisión y atomicidad.

Aunque estas características están claramente definidas, poder cuantificar el valor de cada una en un requisito resulta una tarea compleja. En esta dirección, se han realizado estudios para establecer automáticamente los valores de estas características, usando métricas de calidad sobre los requisitos y estableciendo grados de calidad (Génova et al. 2013; Fabbrini et al. 2001a; Fabbrini et al. 2001b)

No obstante, uno de los factores decisivos en la calidad de los requisitos debe ser la valoración de los expertos involucrados, ya que estas valoraciones están fuertemente vinculadas a las necesidades y exigencias concretas de los diferentes proyectos. Actualmente, las técnicas utilizadas para el análisis de los requisitos, que

establecen automáticamente los valores de las métricas de calidad, no facilitan la flexibilidad de adaptación a las restricciones que exigen los proyectos.

Otro de los factores que influyen considerablemente en la calidad de los requisitos escritos en lenguaje natural es su composición sintáctica y semántica. Mantener en todos los requisitos de una especificación la estructura adecuada al proyecto, influye positivamente en su desarrollo, debido a que se reduce la ambigüedad en la interpretación. En este sentido, se pueden utilizar patrones y plantillas que sirven como guía en la definición de requisitos, asegurando una correcta composición sintáctica y semántica de su contenido, reduciendo así la ambigüedad e incrementando la consistencia.

En esta tesis, se presenta una metodología orientada a mejorar la calidad de los requisitos, incluyendo la valoración de los expertos del proyecto. La metodología está compuesta por dos procesos. El primero proporciona clasificadores para determinar la calidad de los requisitos, y el segundo construye un conjunto de patrones que permiten mejorar la calidad de los requisitos si son empleados en el proceso de redacción.

Explicado con más detalle, el primer proceso que compone la metodología, propone generar emuladores del juicio de expertos humanos para la evaluación de la calidad de requisitos de forma automática, en función de los criterios de calidad que posee el experto del dominio que utilice la metodología. El objetivo es la predicción de la calidad de requisitos que otorgaría el experto. Para llevar a cabo este proceso, el experto debe aportar conjuntos de requisitos previamente clasificados en función de la calidad y que considera apropiados para establecer el grado de exigencia que presente el proyecto. De cada uno de los requisitos que forman el conjunto proporcionado, se extraerán métricas que cuantificarán valores de calidad de los requisitos. En este proceso se utiliza una técnica de aprendizaje automático, concretamente inducción de reglas, para aprender los rangos de valores de las métricas y la forma en que se combinan, para determinar la interpretación de la calidad de los requisitos que posee el experto.

El segundo proceso, propone la generación automática de patrones sintáctico-semánticos mediante el procesamiento del corpus de requisitos, con el objetivo de orientar la redacción de requisitos sintáctica y semánticamente correctos. Durante el procesamiento del corpus, se generan automáticamente patrones binarios en función de la frecuencia de aparición de los elementos gramaticales y categorías semánticas que componen cada uno de los requisitos. Utilizando los patrones binarios se generan patrones sintáctico-semánticos que representan estructuralmente todos los requisitos del corpus. Estos patrones proporcionan una base guía para la redacción de requisitos asegurando la correcta calidad sintáctico-semántica y permitiendo la flexibilidad que implica el uso del lenguaje natural. Se presenta en este trabajo una herramienta que implementa esta metodología. La herramienta procesa corpus de requisitos para generar automáticamente los patrones y proporciona diferentes opciones para gestionar la flexibilidad en las restricciones de los elementos que los componen.

Para demostrar el grado de fiabilidad de la metodología, para cada uno de los procesos, se presenta un caso de estudio utilizando un corpus compuesto por requisitos clasificados por su calidad, proporcionado por el Grupo de Trabajo de Requisitos de la organización INCOSE ("INCOSE (International Council on Systems Engineering)" 2016).

1.2 HIPÓTESIS

Mediante técnicas de inteligencia artificial y de procesamiento de lenguaje natural, es posible establecer un método automático para la optimización de la calidad de los requisitos de ingeniería.

1.3 OBJETO

Proponer un método automático para mejorar la calidad de los requisitos que sea adaptable a cada proyecto y a cada cultura organizacional.

1.4 OBJETIVOS

La propuesta de esta investigación plantea mejorar la calidad de los requisitos en función del tipo de proyecto. Para ello se definen los siguientes objetivos:

- Emular la clasificación de calidad que proporcionaría un experto del proyecto y del domino sobre requisitos.
- Obtener un conjunto de patrones destinados a orientar la redacción de nuevos requisitos asegurando la correcta estructura sintáctico-semántica.
- Optimizar la calidad de los requisitos con el menor esfuerzo mediante sugerencias de corrección automáticas.

1.5 RESTRICCIONES DE LA INVESTIGACIÓN

En este punto se describen los requisitos asociados a la investigación y que restringen el ámbito de estudio.

Como restricción temporal que delimita el estudio encontramos las versiones de las herramientas utilizadas. Debido a su evolución es posible que los resultados puedan verse alterados si se utilizan otras versiones.

Otra restricción es el corpus de requisitos empleado para realizar los casos de estudio. El contenido del corpus afecta a los resultados, por lo que otro conjunto de requisitos podría proporcionar resultados diferentes.

No obstante, a pesar de las consecuencias previsibles por las restricciones expuestas, seguirían vigentes las aportaciones metodológicas de esta tesis.

CAPÍTULO 2: ESTADO DEL ARTE

En esta sección se expone el estado de arte de los ámbitos relacionados con la tesis. El capítulo comienza delimitando el marco de la investigación, desde la ingeniería de sistemas, hasta la ingeniería de requisitos. Varios apartados se centran en los requisitos ya que es el factor clave tratado en la investigación.

Como se ha comentado en la introducción, la metodología propuesta se compone de dos procesos diferenciados. El primero se centra en la clasificación de la calidad de los requisitos, así como en su optimización. En este capítulo se expone el estado del arte relacionado con este proceso; explicando las técnicas de aprendizaje utilizadas, así como trabajos relacionados con la calidad de requisitos, su relevancia en ingeniería y trabajos que utilizan técnicas automáticas para abordar problemas de calidad en los requisitos.

El segundo proceso propone generar automáticamente patrones compuestos por las estructuras sintácticas y semánticas de requisitos. Se hace necesario, por tanto, explicar las técnicas de procesamiento del lenguaje natural utilizadas y los trabajos relacionados, centrándonos en aquellos donde se realicen técnicas de generación automática de patrones.

El capítulo continúa con la definición de la disciplina SBSE, siglas de (Search Based Software Engineering), citando también trabajos donde se emplea esta disciplina para resolver problemas en la Ingeniería de Software.

Para finalizar, se presenta una definición completa de metodología y de detalla las partes que la componen.

2.1 MARCO DE LA INVESTIGACIÓN

En este apartado se encuadra el marco de la investigación con la finalidad de focalizar la localización del estudio dentro del ámbito de la ingeniería. El apartado comienza con la descripción de la ingeniería de sistemas, reflejando la importancia de la calidad en la ingeniería, para desarrollar proyectos de complejidad elevada. Concretando la localización de la investigación, el apartado continúa con la descripción de la ingeniería de requisitos, una de las ramas que completa la ingeniería de sistemas. A continuación, se expone una sección sobre la reutilización del conocimiento, relacionada directamente con la ingeniería de requisitos y de sistemas.

En el siguiente punto se explican las tareas en las que se divide la Ingeniería de Requisitos, continua con la explicación de los beneficios derivados de las especificaciones de requisitos de alta calidad, para concluir con la exposición de trabajos sobre la medición y optimización de la calidad de los requisitos, siendo este el ámbito concreto donde se establece la investigación.

2.1.1 Ingeniería de sistemas

El ámbito general donde se ubica la presente investigación es la ingeniería de sistemas. A continuación, se presenta su definición, se detalla la composición y se expone su relevancia.

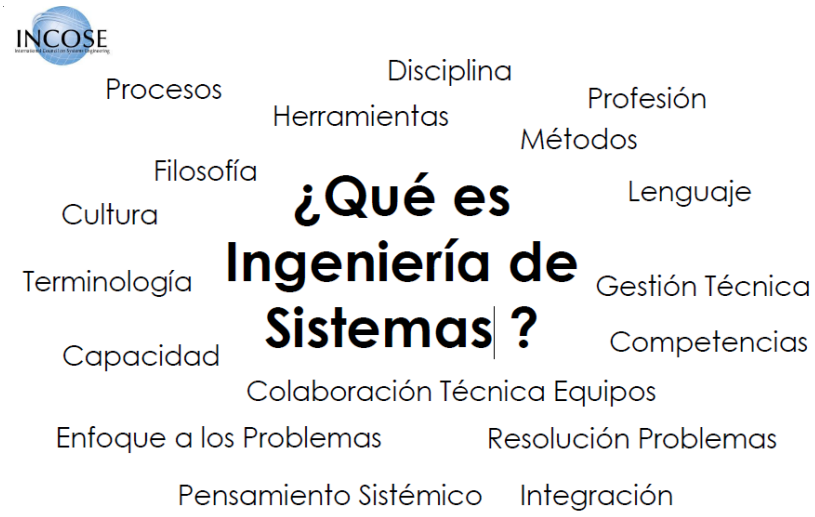

Fig. 1 ¿Qué es la ingeniería de sistemas? (International Council on Systems Engineering 2011)

“La ingeniería de sistemas es un enfoque interdisciplinar cuyo objetivo es conseguir la realización de sistemas de forma satisfactoria. Se centra en la definición de las necesidades de los clientes y la temprana funcionalidad requerida en el ciclo de desarrollo, documentación de requisitos, seguido de la síntesis de diseño y validación de sistemas teniendo en cuenta el problema completo: operaciones, coste y planificación, rendimiento, formación y soporte, pruebas, desarrollo y eliminación. La ingeniería de sistemas considera tanto las necesidades de negocio como las técnicas de todos los clientes, con el objetivo de proporcionar un producto de calidad que satisfaga las necesidades de los usuarios. (International Council on Systems Engineering 2011)”

La definición de este concepto se ha recopilado del libro *Systems Engineering Handbook* (International Council on Systems Engineering 2011) donde se garantiza que: “proporciona una referencia autorizada para entender la ingeniería de sistemas”. Este handbook es consistente con la normativa: (ISO/IEC-15288 2008)

Ya que el objetivo de la ingeniería de sistemas es la “realización satisfactoria de sistemas”, conviene incluir la definición que la guía de INCOSE hace de *sistema:*

“Un sistema es una combinación de elementos organizados que interaccionan para conseguir uno o más propósitos establecidos. Está compuesto por un conjunto de elementos o subsistemas para conseguir un objetivo definido. Estos elementos incluyen productos (hardware, software, firmware), procesos, personas, información, técnicas, comodidades, servicios y otros elementos de soporte.”

Como se hace referencia en la definición, un sistema puede estar compuesto por subsistemas, y estos a su vez compuesto por otros subsistemas. En un sistema complejo abordado por la ingeniería de sistemas, el nivel de jerarquía de niveles entre los sistemas puede ser muy alto. En la siguiente figura se muestra un ejemplo de las jerarquías en un sistema complejo.

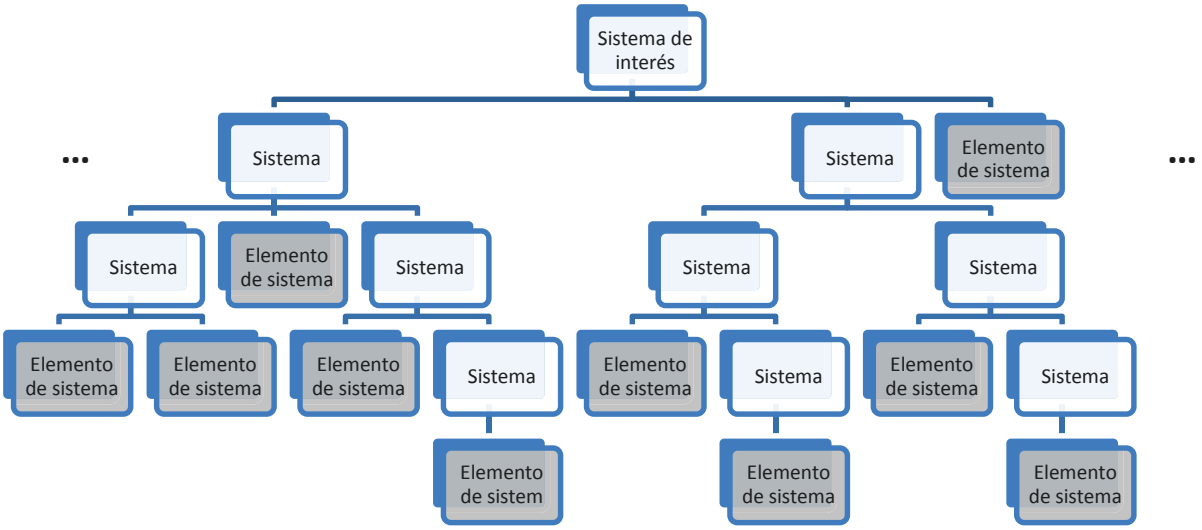

Fig. 2 Jerarquía dentro de un sistema

Para evitar un desequilibrio en el balanceo de los niveles en la jerarquía, se establece *la regla 7 ± 2*, por la que un sistema no debe contener, más de 7 + 2, ni menos de 7 - 2 subsistemas. Un nivel de diseño con demasiadas entidades subordinadas, puede aumentar demasiado su complejidad, por lo que, el mismo diseño y las correspondientes actividades de verificación se correrá el riesgo de estar fuera de control. Con un nivel de diseño con pocas entidades subordinadas puede que no se distingan las actividades de diseño, provocando que tanto el diseño como las actividades de verificación contengan redundancia.

Con un diseño jerárquico adecuado, la ingeniería de sistemas gestiona de forma efectiva la complejidad y los cambios de proyecto desde las fases iniciales, y continua durante su desarrollo con el objetivo de reducir el riesgo asociado a los nuevos sistemas o modificaciones de sistemas complejos. En la siguiente figura se ilustra el coste que se compromete en las distintas fases del desarrollo del proyecto a largo del tiempo. Los datos están asociados a proyectos reales y son proporcionados por el Departamento de Defensa de los Estados Unidos y reportados por la institución *Defense Acquisition University.*

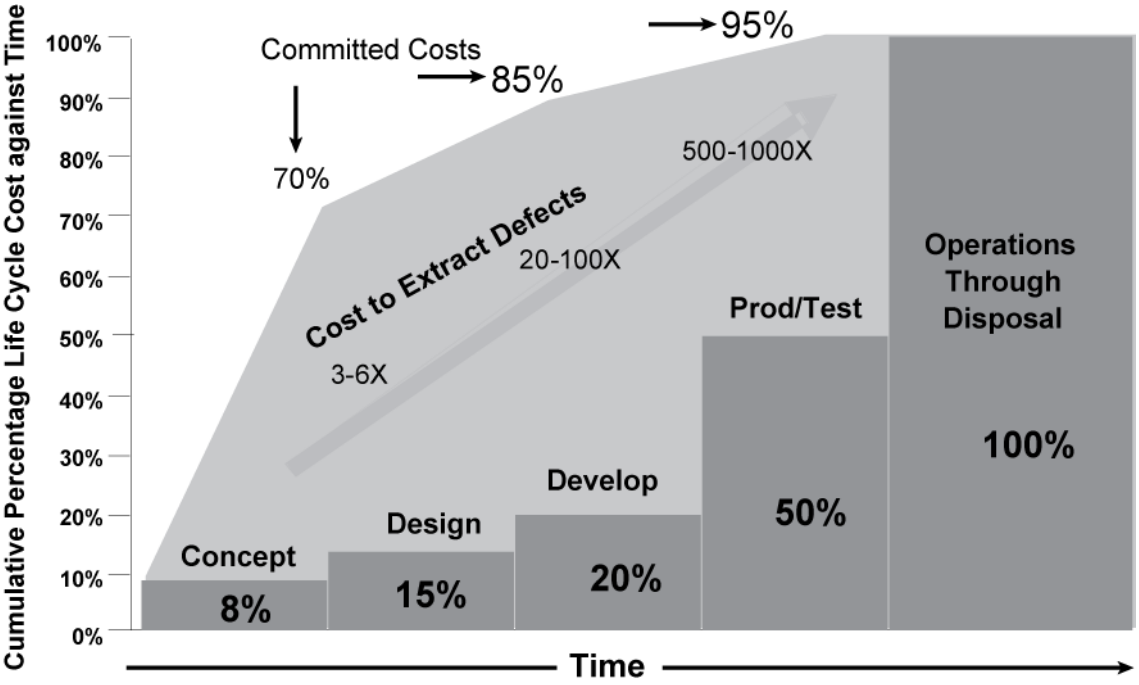

Fig. 3 Coste acumulado por errores detectados en las fases del ciclo de vida (International Council on Systems Engineering 2011)

Los porcentajes a lo largo de la línea de tiempo representan el coste del ciclo de vida real acumulados en el tiempo. Se observa en la figura como el concepto de un sistema nuevo acumula el 8% del coste total del ciclo de vida. El mayor porcentaje del coste se compromete en las fases iniciales del proyecto, esto se observa en la curva *Commited costs* que representa la cantidad de costes comprometidos durante el ciclo de vida. El gráfico indica que cuando el coste total acumulado es del 20%, que corresponde a las primeras fases del proyecto, el 80% del coste comprometido del ciclo de vida ya ha sido determinado. Se compromete el otro 20% del coste durante las demás fases del proyecto por lo que la ingeniería de sistema se debe aplicar desde el inicio del proyecto, en la fase de concepción, y continuar hasta el final.

La flecha diagonal bajo la curva nos recuerda que el coste de los errores se incrementa durante el ciclo de vida.

La mayoría de los errores que suceden en los proyectos son provocados por defectos en las fases iniciales, pero no se detectan hasta que los errores son provocados, normalmente en las fases avanzadas.

En el estudio *An Information Systems Manifesto* (J. Martin 1984), James Marint concluye que más del 50% de los defectos ocurren en las fases iniciales de los proyectos, concretamente en los requisitos del sistema. Y que más del 80% de la distribución de esfuerzos en reparar errores son provocados por defectos en los requisitos. La siguiente figura refleja estos porcentajes.

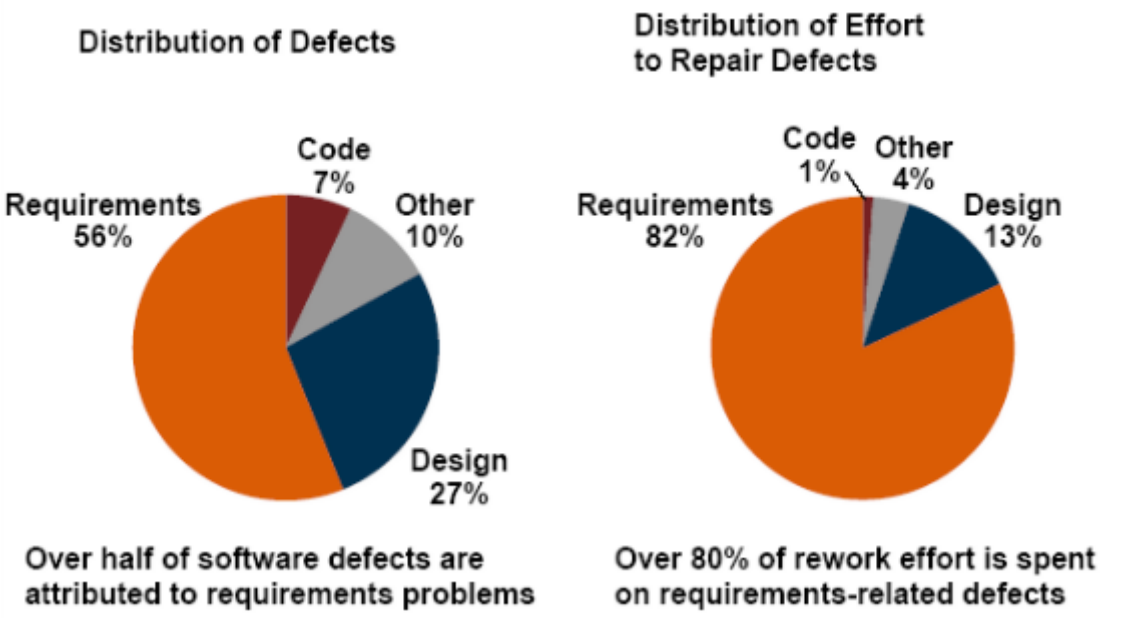

Fig. 4 Distribución de defectos y esfuerzo destinado a corregirlos (J. Martin 1984)

Debido a la importancia que tienen los requisitos en los proyectos, una nueva rama surgió de la ingeniería de sistemas: la ingeniería de requisitos.

2.1.2 Ingeniería de Requisitos

La Ingeniería de Requisitos es una rama de la Ingeniería de Sistemas cuyo objetivo es entender las necesidades de los clientes y definirlas de forma estructurada en conjuntos de requisitos (Shamieh 2011)

La definición formal que la normativa (Iso/Iec/Ieee-29148 2011) recoge sobre la ingeniería de requisitos es:

Una función interdisciplinar que media entre los dominios del cliente y el proveedor para establecer y mantener los requisitos que deber cumplir un sistema, software o servicio de interés. La ingeniería de requisitos se ocupa de descubrir, obtener, desarrollar, analizar, determinar métodos de verificación, validez, comunicación, documentación, y gestión de los requisitos. El resultado de la ingeniería de requisitos es una jerarquía de requisitos que:

- Establece un acuerdo de entendimiento entre stakeholders (clientes, asociaciones, usuarios, proveedores, etc.)
- Se validan con las necesidades del mundo real que pueden ser implementadas.
- Proporciona una base de verificación de diseño y soluciones aceptadas. [ISO 29148]

2.1.3 Reutilización del conocimiento

Gran cantidad de conocimiento adquirido en el desarrollo de un proyecto, puede ser reutilizado en las distintas etapas del ciclo de vida e incluso en distintos proyectos (Frakes and Isoda 1994): informes de viabilidad, descripción de problemas, propuestas de proyectos, artefactos, prototipos, diccionario de datos, entre otros (Krogh, Spaeth, and Haefliger 2005).

En la tesis (Fraga 2010) se define un marco de trabajo y una metodología para la reutilización de cualquier conocimiento a bajo coste. Significa que, independientemente de cuál sea la clase de conocimiento, es posible que sea reutilizado más adelante y en cualquier contexto.

Una de las áreas donde la reutilización presenta mayores dificultades que en otras disciplinas es en la Ingeniería del Software. Esto se debe a la imposibilidad de realizar desarrollos informáticos mediante la unión directa de componentes software [Llorens, 1996], ya que el uso de esto componentes no cumplirá con la especificación requerida. Otra dificultad es que la Ingeniería de Software se puede considerar una disciplina novel, por lo que es difícil poner en práctica la reutilización en una actividad en permanente cambio. No obstante, la reutilización del software es la forma más prometedora para optimizar el desarrollo de los proyectos.

Los componentes software objeto de reutilización son definidos por Neighbors (Neighbors 1984) como: “Aquellos elementos que especifican la semántica de un cierto dominio de aplicación”. Teniendo en cuenta que se puede aplicar la reutilización en cualquier etapa del ciclo de vida del proyecto (análisis, diseño, codificación, pruebas, mantenimiento y nuevos desarrollos) (Neighbors 1984; Ning, Engberts, and Kozaczynski 1993; Jarzabek 1993) se debe extender el grupo de componentes software para ser reutilizados e incluir, toda aquella información útil para el desarrollo de nuevos proyectos, como por ejemplo: resultados de pruebas, estadísticas de evolución, medidas de tiempos de desarrollo, requisitos, etc.

Se puede estructurar la reutilización en tres niveles:

1. Reutilización de especificaciones
2. Reutilización de diseño
3. Reutilización de código

Cuanto más alto sea nivel de abstracción en las etapas de desarrollo, mayores serán las ventajas obtenidas por el proceso de reutilización. Por ejemplo, la reutilización de código es la menos productiva por ser posterior a otras etapas del ciclo de vida [Velasco, 1998]. Por tanto, uno de los principales elementos destinados a la reutilización son los requisitos, ya que estos se definen en las etapas tempranas del proyecto.

Con la reutilización de los requisitos se traspasa conocimiento que ha sido analizado, corregido, modificado y testeado durante todo el ciclo de vida de otros proyectos. Lecciones aprendidas en experiencias previas son utilizadas para mejorar el destino y la ejecución de los nuevos proyectos.

Para mejorar los beneficios de reutilización de requisitos, en los trabajos (Toval et al. 2002; Toval et al. 2008) se presenta SIREN, una aproximación práctica para crear, seleccionar y especificar los requisitos de un sistema software basándose estándares de reutilización e ingeniería del software. SIREN engloba un modelo de proceso en espiral, plantillas de documentos de requisitos, un repositorio de requisitos reutilizables organizado por catálogo, y una herramienta de gestión de requisitos.

En (de Gea et al. 2013) se presenta PANGEA, un método de ingeniería de requisitos focalizado en la reutilización, donde se especifica el conocimiento mediante requisitos escritos en lenguaje natural e incorpora estándares de ingeniería del software. El objetivo de este método es mejorar el intercambio de conocimientos y la colaboración en entornos distribuidos. PANGEA engloba un modelo de procesos, un modelo de referencia de requisitos y una herramienta como soporte de arquitectura

No obstante, la reutilización de los requisitos implica incluir en el nuevo proyecto las restricciones que han sido acordadas en los proyectos previos, por lo que el nivel de exigencia debe ser similar. Esto puede ser un problema ya que medir y conocer estas restricciones no es una tarea sencilla. En esta investigación se propone un método para poder conocer, de forma automática, si los requisitos cumplen con las exigencias de calidad acordadas en otros proyectos, mediante la aportación de conjuntos requisitos que cumplan las restricciones.

Otras de las características de los requisitos que deberían ser utilizadas, son las composiciones de las estructuras sintáctico-semánticas, con el objetivo de crear requisitos estructuralmente correctos y con elementos adecuados a categorías sintácticas y semánticas establecidas. La investigación realizada en esta tesis propone usar patrones para reutilizar estas estructuras. Según el trabajo (Ambler 2000) “un patrón es una solución a problemas comunes, teniendo en cuenta las restricciones pertinentes y permitiendo el uso de técnicas y estrategias probadas”. En el trabajo (Ketabchi, Sani, and Liu 2011), se citan los beneficios asociados al uso de patrones de requisitos:

- Ofrecen orientación sobre qué información debe ser incluida, dando consejos y dirigiendo la atención de los analistas sobre alternativas y la advertencia sobre las dificultades comunes que se podrían encontrar.
- El uso de patrones ahorra tiempo y esfuerzo ya que no hay necesidad de empezar desde cero. De hecho, los patrones proporcionan un buen punto de partida y actúan como una base sobre la cual el resto del proyecto se puede construir.
- Proporcionan consistencia a lo largo del proceso.

- Ayudan a utilizar las mejores prácticas y la experiencia de los expertos en situaciones similares.
- Reutilización del conocimiento.
- Ayudan a recordar a las necesidades y problemas olvidados.
- Patrones bien definidos proporcionan orientación sobre el diseño y también comprueban fases avanzadas que se llevan a cabo en el ciclo de desarrollo.

2.1.4 Estructura del proceso de Ingeniería del Requisitos

En el libro *Software Requirements* (Sawyer and Kontonya 2001) el autor divide el proceso de ingeniería de requisitos en:

- Desarrollo de requisitos: esta actividad se compone de cuatro tareas:
    - Obtención
    - Análisis
    - Especificación
    - Validación
- Gestión de requisitos

Añadimos a esta lista la reutilización de requisitos, que puede ser entendida como la primera tarea en la ingeniería de requisitos, y mantenida en el desarrollo del ciclo de vida del proyecto.

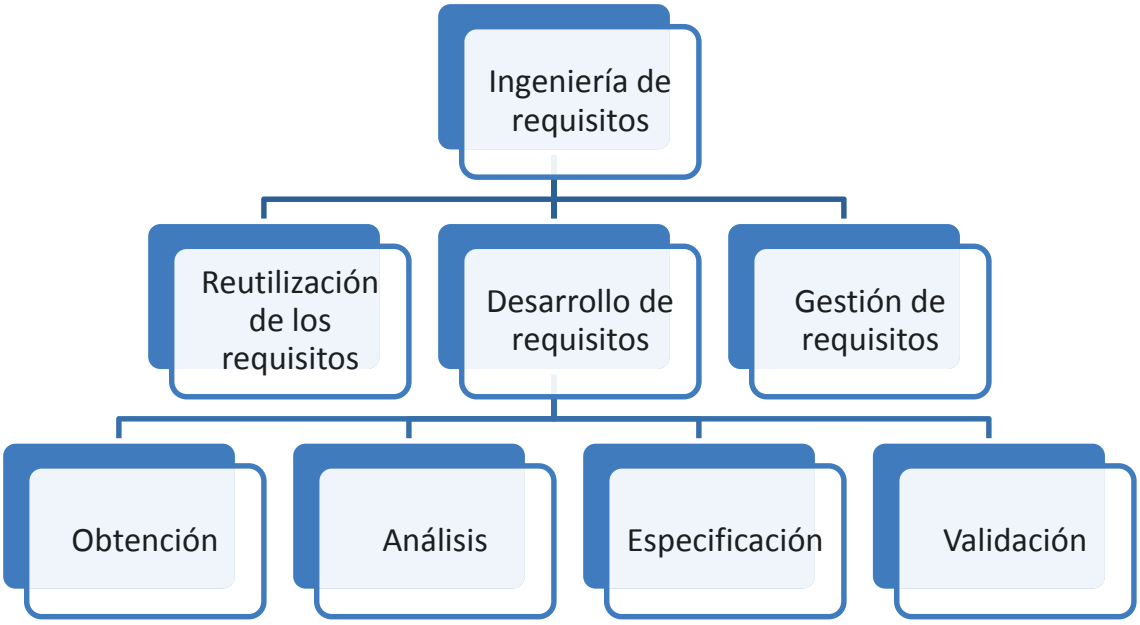

Fig. 5 Actividades en la ingeniería de requisitos

Independientemente de la estructura en el ciclo de vida del proyecto, estas actividades siempre se deben realizar tomando como base el conjunto de requisitos.

A continuación se detallan cada una de las actividades.

2.1.4.1 Reutilización de los requisitos

Como se ha comentado, esta actividad ha sido añadida al conjunto de actividades que aparecían en el libro referenciado (Sawyer and Kontonya 2001). En el apartado de reutilización del conocimiento [sección 2.1.3] se ha expuesto la relevancia de la reutilización de requisitos y de patrones de requisitos. Las tareas que componen esta actividad son:

- Construir un corpus de requisitos que representen las bondades que se quieren mantener y los defectos que se quieren evitar.
- Determinar por cada uno de los requisitos del corpus su nivel de calidad.
- Conocer la adecuación a las exigencias del proyecto de los requisitos que integrarán el nuevo proyecto.

- Obtener una base de patrones que almacenen la información estructural sintáctico-semántica de los requisitos para orientar la redacción de nuevos requisitos.

2.1.4.2 Desarrollo de requisitos

Esta actividad se divide en las siguientes cuatro tareas:

- **Obtención**: agrupación y descubrimiento de requisitos desde los stakeholders y otras fuentes. Se pueden emplear distintas técnicas como entrevistas, workshops, documentos de análisis, prototipos, entre otros. La obtención de requisitos es el primer paso en el desarrollo de requisitos. Las claves de esta actividad son:
    - Identificar las expectativas de los usuarios y stakeholders.
    - Entender las metas de los usuarios y su correspondencia con los objetivos de negocio.
    - Aprender el entorno donde va a ser usado el producto.
    - Trabajar con representantes de cada clase de usuario para entender las necesidades funcionales y las expectativas de calidad.

- **Análisis**: consiste en un entendimiento más preciso y profundo de cada requisito, y una clasificación en función de su nivel de detalle, tipo (informativo, operativo, cualitativo, etc.), área temática, subsistema donde se va a integrar, etc. El análisis identificará deficiencias que serán resueltas con una obtención más profunda. En esta actividad se desempeña:
    - El análisis de la información recibida de los usuarios para clasificar los requisitos en: requisitos funcionales, de calidad, sugerencia de soluciones, reglas de negocio, u otros tipos de información.
    - La descomposición de los requisitos de alto nivel en apropiados niveles de detalle.
    - La extracción de requisitos funcionales desde otros requisitos informativos.

- La comprensión de la importancia relativa a los atributos de calidad.
- La asignación de requisitos a los elementos definidos en la arquitectura del sistema.
- La negociación de las prioridades en el desarrollo.
- Identificar lagunas en los requisitos o requisitos innecesarios.

- **Especificación**: consiste en representar y almacenar el conocimiento recogido de los requisitos de forma persistente y organizada, para permitir una buena comunicación y gestión de cambios. El resultado principal es:
  - Un conjunto de requisitos escritos y diagramas que mejoran la comprensión, revisión y uso por la audiencia a la que va destinada.

- **Validación**: técnicas para confirmar que el correcto conjunto de requisitos ha sido identificado y especificado para construir una solución que satisfaga los objetivos de negocio del proyecto. En esta tarea se realiza:
  - La revisión de los requisitos documentados para corregir cualquier problema antes que el equipo de desarrollo lo acepten.
  - Pruebas y criterios de aceptación para comprobar que el producto basado en los requisitos cumple con las necesidades de los clientes. Requisitos no validados se someterán a una obtención, análisis o especificación más profunda.

2.1.4.3 Gestión de requisitos

Esta es una actividad de soporte cuyo objetivo es abordar el hecho de que los requisitos cambian con el tiempo durante los proyectos. La gestión de los requisitos se realiza durante todas las fases en el desarrollo de los requisitos, así como durante el ciclo de vida del proyecto, con el fin de minimizar el impacto de cambios en los requisitos. Esta actividad incluye:

- Definir una instantánea de los requisitos que representen requisitos funcionales y no funcionales acordados, revisados y aceptados, normalmente de una versión específica o iteración en el desarrollo del proyecto.

- Evaluación del impacto de los cambios propuestos e incorporar, de forma controlada, los cambios aprobados en el proyecto.
- Mantener, en la medida de lo posible, la planificación de los proyectos adaptándose a la evolución de los requisitos.
- Si fuera necesario, negociación de nuevos compromisos basándose en el impacto de los cambios en los requisitos.
- Definir las relaciones y dependencias entre los requisitos.
- Trazar los requisitos individuales con su diseño correspondiente, código fuente y pruebas.
- Trazar el estado de los requisitos y cambios de actividad durante todo el proyecto.

2.1.5 Beneficios derivados de especificaciones de alta calidad

Si se toma en consideración los estudios que indican la cantidad de posibles errores que puede sufrir un proyecto por tener requisitos de mala calidad, y el elevado coste de solucionar los problemas derivados si estos errores no se detectan a tiempo, se hace necesario que el resultado de las actividades de desarrollo y gestión de requisitos sean especificaciones de alta calidad. Sobre esto, el libro *Software Requirements* (Sawyer and Kontonya 2001) recoge los beneficios que se derivan al conseguir especificaciones sin defectos en la calidad.

Realizando procesos metódicos de requisitos, aumenta el grado de inclusión de los stakeholders al proyecto y permite a los equipos de desarrollo entender mejor sus necesidades. Si estas necesidades están claramente definidas desde el inicio del proyecto, durante el desarrollo se evitarán errores debido a malas interpretaciones. Errores que se traducen en un aumento del coste, dilatación en el tiempo de finalización del proyecto, imposibilidad de reutilización de partes desarrolladas ya que se propagarían los errores, insatisfacción de los usuarios, entre otros.

Los beneficios derivados de procesos de requisitos de alta calidad son:

- Reducir defectos en los requisitos y en el producto desarrollado.

- Evitar desarrollo redundante.
- Disminución del tiempo de desarrollo y entrega del producto.
- Eliminar características innecesarias.
- Reducción de coste.
- Mejorar la comunicación entre los stakeholders y el equipo de desarrollo.
- Mejorar la comunicación entre los miembros del equipo de desarrollo.
- Reducir cambios y aumento incontrolados del alcance del proyecto.
- Evitar el caos en el proyecto.
- Satisfacción del equipo de desarrollo y clientes del producto final.
- Productos que realizan lo que se supone que deben hacer

2.1.6 Medición y optimización de la calidad de los requisitos.

Según la guía la escribir requisitos ((Incose) 2015), la mejor forma de expresar las necesidades es utilizar requisitos escritos en lenguaje natural, ya que no hay limitaciones en los conceptos que se pueden expresar, y las oraciones y las estructura gramaticales proporcionan un medio de localización de elementos significativos.

No obstante, la flexibilidad que se consigue al utilizar el lenguaje natural para expresar requisitos, lleva consigo problemas de interpretación y ambigüedad a nivel individual de requisitos, y redundancia e inconsistencia a nivel de especificaciones completas.

Para evitar estos problemas existen guías de distintas organizaciones ((Incose) 2015; Iso/Iec 2010; Hooks 2000; Magee and Tripp 1997; Rosenberg and Linda 2001; Alexander and Stevens 2002) que definen características deseables (A. Fantechi et al. 2003; Rosenberg and Linda 2001; Wilson, Rosenberg, and Hyatt 1997) que deben tener los requisitos escritos en lenguaje natural.

En el trabajo (Génova et al. 2013), se sintetizan las propiedades deseables que se pueden encontrar en la literatura, estas propiedades son:

- Validez: el cliente debe poder confirmar (validar) que los requisitos expresan correctamente todas sus necesidades.

- Verificabilidad: los ingenieros deben poder comprobar (verificar) que el sistema producido es realmente el sistema especificado.
- Modificabilidad: los requisitos deben ser modificables para facilitar los cambios del sistema provocados por el mantenimiento.
- Completo: todas las necesidades son cubiertas por el conjunto de requisitos.
- Consistente: no hay contradicciones entre los requisitos.
- Comprensibilidad: los requisitos se entienden correctamente sin dificultad.
- Inambigüedad: debe haber una sola interpretación por cada requisito.
- Trazabilidad: debe existir una relación explicita entre cada requisito con el diseño, implementación y artefactos de prueba.
- Abstracción: los requisitos expresan lo que la aplicación debe hacer y no cómo debe hacerlo. Es decir, la especificación debe evitar el exceso de detalles técnicos sobre la implementación
- Precisión: todos los términos usados en los requisitos deben ser concretos y estar bien definidos.
- Atomicidad: cada requisito debe ser claramente identificado unitariamente, para no confundirlo con otros requisitos.

En el mismo trabajo se presentan indicadores para medir estas propiedades. Con este aporte se proporcionan indicadores cuantitativos a las propiedades cualitativas, permitiendo así medir el grado de calidad de un requisito.

Se van a resumir a continuación los indicadores que se utilizan en el artículo; donde se clasifican en las siguientes categorías:

- Indicadores morfológicos: aquellos que miden propiedades desde un punto de vista puramente formal, sin considerar el contenido.
- Indicadores léxicos: aquellos que miden propiedades relativas al contenido del texto y que requieren alguna clase de información relacionada.
- Indicadores analíticos: aquellos que requieren un análisis textual de los requisitos empleando herramientas lingüísticas de cierta complejidad.
- Indicadores relacionales: aquellos que miden las propiedades estructurales del conjunto de requisitos, en lugar de propiedades de los requisitos individuales.

2.1.6.1 Indicadores morfológicos

Aparte de los indicadores que comúnmente se utilizan, como la longitud y la legibilidad, también se incluyen el número de signos de puntuación y el número de acrónimos y abreviaciones

- **Longitud:** la longitud de un requisito afecta directamente a la propiedad deseable "atomicidad", y como consecuencia afecta también a las propiedades de: trazabilidad, verificabilidad y modificabilidad. Puede ser medida en caracteres, palabras (la más común), frases o párrafos.

- **Legibilidad:** el índice de legibilidad trata de medir el grado de dificultad de lectura del texto. No se debe confundir la legibilidad tipográfica con la legibilidad lingüística, que es en la que se centra este indicador. La legibilidad se mide mediante una formula compuesta por la media de silabas en las palabras y media de palabras en las frases del requisito.

- **Puntuación:** este indicador está también relacionado con la comprensión de los requisitos y mide el número los signos de puntuación por frase, dividido entre la longitud de la frase. Un número elevado de signo de puntuación dificulta la comprensión y la escasez de ellos indica que la frase debe ser dividida en varias frases, y así analizar si es conveniente dividir el requisito.

- **Acrónimos y abreviaciones**: el exceso de acrónimos (NASA, E.S.A, etc.) y abreviaciones ("nº" por "número") se puede usar como indicador de falta de calidad, debido a que, un número elevado de ellos dificultad la comprensión.

2.1.6.2 Indicadores léxicos

Como se ha mencionado, estos indicadores necesitan de listas de términos definidas por el usuario. En base a estas listas se pueden obtener indicadores que miden ciertas propiedades de la calidad relacionadas con los requisitos.

- **Términos conectores:** el uso excesivo de términos conectores disminuye la calidad. Afecta concretamente a las propiedades de atomicidad, precisión, abstracción, compresión, e inambigüedad. Dentro del conjunto de estos términos podemos encontrar:
    - *Número de términos disyuntivos-copulativos:* relacionados con la falta de atomicidad de los requisitos. El uso de conjunciones, concretamente "y", "o", puede indicar que el requisito está compuesto realmente por dos o más requisitos. Y el uso de términos disyuntivos exclusivos, inclusivos y explicativos, puede provocar falta de precisión, ambigüedad y mala comprensión.
    - *Términos negativos:* incluir muchos términos negativos como: no, tampoco, nunca, nada, de ninguna manera, etc., puede dificultar la comprensión del significado de las frases, incrementando además el riesgo de inconsistencias lógicas, afectando especialmente a la propiedad de comprensibilidad.
    - *Número de términos de control de flujo:* el uso excesivo de términos de control de flujo como: mientras, cuando, si-entonces, puede provocar un exceso de detalle del control del flujo de los procesos y funciones, sobrepasando la abstracción que necesitan las especificaciones de requisitos. Este indicador está relacionado con las propiedades abstracción y atomicidad.
    - *Número de términos anafóricos:* estos términos se emplean para sustituir otros términos, normalmente son pronombres personales (lo), pronombres relativos (que, cual, donde), pronombres demostrativos (este, estos), etc. Los términos anafóricos aumentan el riesgo de imprecisión y ambigüedad en textos de carácter técnico.

- **Términos imprecisos:** una fuente típica de defectos es el uso de términos imprecisos que introducen ambigüedades en los requisitos. La comparación con listas de términos prohibidos proporciona una métrica interesante. Algunos términos imprecisos que componen estas listas pueden ser: bueno, adecuado, eficiente (términos cualitativos), suficiente, bastante, aproximadamente (términos cuantitativos), generalmente, típicamente, casi siempre (términos de frecuencia), posiblemente, probablemente, opcionalmente (términos probabilísticos), etc.

- **Términos de diseño:** abusar de términos relacionados con el diseño y la tecnología (métodos, parámetros, bases de datos) denota falta de abstracción en los requisitos.

2.1.6.3 Indicadores analíticos

Para obtener indicadores analíticos se necesita un análisis textual de los requisitos, empleando con frecuencia herramientas lingüísticas.

- **Tiempo verbal y de modo:** el uso de formas verbales y de modo, que requieren un análisis de la flexión verbal, depende fuertemente del lenguaje del requisito. El número de formas verbales imperativas es un buen indicador de la atomicidad del requisito: un exceso de formas imperativas indicaría que el requisito no es lo suficientemente atómico. Por otro lado, el número de modos condicionales en lugar de futuro es un error típico originado por el deseo de expresar necesidades de bajo nivel del requisito. También, es interesante considerar que los requisitos no deben ser escritos en voz pasiva, ya que ganan cierto grado de imprecisión.

- **Términos del dominio**: otra medida de calidad relevante es el número de términos del dominio encontrados en los requisitos. Para obtener esta medida es necesario normalizar los términos (conjugaciones de verbos, género y número de los nombres) para encontrar formas canónicas definidas en el

domino. Una vez que los nombres y verbos han sido normalizados, son comparados con los términos definidos del dominio, encontrados en glosarios de términos, tesauros u ontologías. El número de términos en el dominio no debe ser ni demasiado alto (indicando perdida de atomicidad) ni demasiado bajo (indicando requisitos imprecisos).

2.1.6.4 Indicadores relacionales

Estos indicadores están relacionados con medidas sobre el conjunto de requisitos. Estas relaciones no se pueden determinar automáticamente, por lo que, para poder obtener estos indicadores es necesario que las relaciones ya estén determinadas.

- **Número de versiones de un requisito**: un elevado número de versiones en un requisito es un indicador de la volatilidad o inestabilidad de los requisitos, influyendo tanto en la validez como en la verificabilidad.

- **Grado de anidación:** si los requisitos están jerárquicamente estructurados, se puede obtener medidas sobre la profundidad de los niveles de jerarquía. Grados de anidación demasiados altos o demasiado bajos pueden afectar negativamente al nivel de comprensión de la información contenida en el conjunto de requisitos.

- **Número de dependencias de un requisito:** estas dependencias afectan a otros requisitos u otros artefactos creados en la fase de desarrollo. Un elevado número de dependencias dificulta el mantenimiento y reutilización, afectando a la atomicidad, comprensión y trazabilidad, por tanto a las propiedades de validez, verificabilidad y modificabilidad.

- **Número de solapamiento entre requisitos:** este número indica la cantidad de requisitos que expresan la misma información. Los problemas derivados por este solapamiento son la redundancia y conflicto en la información que expresan los requisitos.

2.2 TÉCNICAS DE APRENDIZAJE AUTOMÁTICO

Las técnicas de aprendizaje automático son estrategias para la obtención de patrones a partir de recopilaciones de datos. Los datos se estructuran en los llamados ejemplos de aprendizaje para ser procesados por los algoritmos. Se pueden dividir las técnicas en dos categorías, supervisadas o no supervisadas, dependiendo si se conocen o no la clasificación de los elementos (Weiss and Indurkhya 1997). En las técnicas supervisadas se conoce la clase a la que pertenece cada ejemplo de aprendizaje.

2.2.1 Técnicas de inducción reglas

Las técnicas de inducción de reglas reciben como ejemplos de aprendizaje conjuntos de casos. Estos ejemplos de aprendizaje se representan por un conjunto de atributos común e incluye un atributo de clase. Los casos se distinguen entre sí mediante los valores de los atributos. Procesando estos datos de entrada, las técnicas de inducción de reglas generan árboles de decisión o conjuntos de reglas que proporcionarán la clasificación de los nuevos ejemplos (Hong, Mozetic, and Michalski 1986; Clark and Niblett 1989). Por tanto, la inducción de reglas contempla dos estrategias:

- Generación de un árbol de decisión para la extracción de reglas (J. Quinlan 1993).

- Realizar una estrategia de *covering* generando reglas para cubrir en conjunto de ejemplos de una única clase y, una vez eliminados los ejemplos cubiertos, proseguir con otra clase.

Una implementación de la primera estrategia es el algoritmo C 4.5 (J. Quinlan 1993), una extensión del ID3 (J. R. Quinlan 1986), que obtiene reglas mediante la generación de un árbol de decisión. Otros algoritmos como el PRISM (Cendrowska 1987) se basan en una estrategia de *covering*, y otros algoritmos como el PART (Frank and Witten 1998) combinan ambas estrategias.

A continuación, se presentan las ventajas derivadas de estas estrategias:

- La robustez frente al ruido (ya sea por errores en la clasificación o insuficiencia de datos).
- Identificación de atributos irrelevantes.
- Detección de la ausencia de atributos discriminantes o vacíos de conocimiento.
- Generación de reglas comprensibles y de gran expresividad.
- Posibilidad de modificación de las reglas por parte de expertos debido a su legibilidad (Major and Mangano 1995).

2.2.2 Conjuntos de Clasificadores

El objetivo de los conjuntos de clasificadores es mejorar la precisión que obtendrían de forma individual cada uno de los clasificadores pertenecientes al conjunto. La decisión final de cada clasificador se obtiene por la combinación de las decisiones de los clasificadores del conjunto (Dietterich 1997).

Las técnicas más comunes de conjuntos de clasificadores son: los clasificadores homogéneos y los clasificadores heterogéneos.

2.2.2.1 Clasificadores homogéneos

Los clasificadores homogéneos generar clasificadores a partir de un algoritmos de aprendizaje base (Dietterich 2000). Los métodos más relevantes de construcción de clasificadores homogéneos son Bagging (Breiman 1996) y Boosting (Schapire 1990). Estos métodos generan diferentes hipótesis mediante la manipulación de los ejemplos de entrenamiento.

En el proceso, diferentes conjuntos de aprendizaje entrenan al algoritmo base para construir los diferentes clasificadores del conjunto. La decisión final de la clase establecida para un nuevo ejemplo se realiza por un sistema de votación. La variedad en los clasificadores proporciona diversidad en las hipótesis generadas, cubriendo así un mayor abanico de posibilidades de acierto. Esta variedad es proporcionada por el grado de inestabilidad del algoritmo base. Los métodos de inducción de reglas son

algoritmos de aprendizaje inestables y por tanto son adecuados para ser utilizados como algoritmos base.

El método, Bagging (Bootstrap Aggregation) (Breiman 1996) crea conjuntos de muestras a partir del conjunto de ejemplos de entrenamiento. Cada muestra contiene el 63,2% de instancias del conjunto total, y se generan los conjuntos necesarios hasta cubrir todas las instancias de entrenamiento. La elección de la clasificación final será aquella que tenga mayor un número de votos del total propuesto por los clasificadores.

El algoritmo Boosting (Schapire 1990) en la versión más representativa (AdaBoost) (Y Freund and Schapire 1995; Yoav Freund and Schapire 1996), construye los clasificadores de forma secuencial, priorizando aquellos que proporcionan clasificaciones erróneas. A cada clasificador se le asocia un peso representativo del grado de acierto de la clasificación de las instancias de aprendizaje. La decisión final de los nuevos elementos se realiza teniendo en cuenta la ponderación de cada clasificador.

2.2.2.2 Clasificadores heterogéneos

Los clasificadores heterogéneos generan clasificadores usando distintos algoritmos de aprendizaje. El método más representativo de esta clase de clasificadores es Stacking (Stacked Generalization) (Wolpert 1992).

La decisión de clasificación final de las nuevas instancias la realiza un clasificador generado mediante otro algoritmo de aprendizaje. Este algoritmo, aprende a combinar los resultados de los otros clasificadores. La dificultad de este método radica en la elección de los algoritmos.

2.3 CALIDAD DE REQUISITOS

Como se ha mencionado, gran parte de los errores en el desarrollo de los proyectos de ingeniería son originados por deficiencias en la obtención, especificación y análisis de requisitos. Estos errores son los más costosos y más difíciles de corregir si no son detectados en las fases iniciales del proyecto (Brooks 1987; Braude 2000; Glass 2002; Bourque and Dupuis 2004; Think Big 2013). Se exponen a continuación distintos trabajos que revelan el interés de la comunidad científica y las organizaciones por la producción de requisitos de alta calidad.

Con el objetivo de mejorar la calidad de los requisitos existen aproximaciones basadas en modelos conceptuales que tienen como finalidad la detección de errores y e incrementar la formalización en las especificaciones (Gorschek and Wohlin 2006; Racheva, Daneva, and Herrmann 2010; Alencar, Giachetti, and Pastor 2009) Destacan trabajos como (Vicente-Chicote, Moros, and Toval 2007) donde se propone un metamodelo de requisitos llamado REMM (Requirements Engineering MetaModel) que incluye los elementos que deben aparecer en los modelos de requisitos (requisitos, stakeholders, casos de usos, etc.) junto a las relaciones que debe existir entre ellos. En este trabajo se presenta *REMM-studio*, un entorno que permite: (1) la creación de modelos de requisitos gráficos, (2) la validación de los mismos con metamodelos y conjunto de restricciones OCL, y (3) la generación automática de navegabilidad en especificaciones de requisitos de software. En el trabajo (Moros, Vicente-Chicote, and Toval 2008a), se amplía la propuesta anterior, incorporando mecanismos de variabilidad en el modelado para permitir la reutilización de los requisitos. Y en (Moros, Vicente-Chicote, and Toval 2008b), se incluye la especificación de: catálogos de modelos de requisitos reutilizables, y requisitos de productos específicos, para la reutilización de requisitos previamente definidos.

Directamente relacionado con la calidad de los requisitos escritos en lenguaje natural, se encuentra el trabajo (Génova et al. 2013) ya presentado en la sección [2.1.6 Medición y optimización de la calidad de los requisitos]. Donde “se sintetizan diferentes indicadores cuantitativos de las propiedades deseadas en los requisitos”.

Con el fin de asegurar que los requisitos cumplen con estas propiedades, varias organizaciones internacionales han creado guías y estándares que ayudan a los ingenieros a especificar requisitos de calidad. Se han publicado varios libros y artículos indicando la estructura y las métricas de calidad que se deben tener en cuenta en los requisitos de las especificaciones (Magee and Tripp 1997; Hooks 2000; Rosenberg and Linda 2001; Alexander and Stevens 2002; Turk 2006). Organizaciones internacionales como la asociación IEEE ha reeditado varios estándares para la especificación de requisitos (Ieee830 1998; Iso/Iec/Ieee-29148 2011). La Organización Internacional de Normalización o ISO ha creado estándares que complementan los estándares IEEE para la especificación de requisitos de calidad de software y de sistemas (Bøegh 2008; Iso/Iec 2010). Ivy F. Hooks publicó para la NASA una "guía para el mantenimiento y especificación de requisitos" (Hooks 2000) y la Agencia Espacial Europea (ESA) ha publicado una guía similar para la especificación de requisitos ((Esa) 1995). La organización International Council on Systems Engineering (INCOSE) dedicada al evolución y progreso de la ingeniería de sistemas, publicó en 2012 otra guía para escribir requisitos ((Incose) 2015).

También con el objetivo de ayudar en la creación y desarrollo de los proyectos, existen herramientas software que posibilitan la gestión de requisitos (Company 2015b; IBM 2015; Reqtify 2005; Caliber 2015; Solutions 2015).

Las características principales que ofrecen las herramientas sobre la gestión de requisitos son:

- Gestión del almacenamiento.
- Ayuda en la especificación.
- Trazabilidad.
- Validación.
- Gestión de la calidad.

2.4 AUTOMATIZACIÓN EN LA CALIDAD DE REQUISITOS

En el apartado anterior se han mencionado propiedades que deben cumplir los requisitos y que determinan su calidad. La investigación en la ingeniería de requisitos ha llevado a presentar estudios donde se evalúan estas propiedades de forma automática. Existen estudios donde la atención es puesta en la detección y evaluación de una sola característica; así se han presentado trabajos para detectar ambigüedades (Chantree et al. 2006; Popescu et al. 2008; Kiyavitskaya et al. 2008; Wang et al. 2013), inconsistencias (de Sousa et al. 2010), y conflictos (Sardinha et al. 2013).

En (Ali, Dalpiaz, and Giorgini 2013) los autores presentan un conjunto de mecanismos de análisis automatizados para ayudar a los ingenieros a detectar y analizar errores de modelado en modelos de requisitos. Presentan una herramienta llamada *Re-Context* que ayuda a los analistas en la comprobación de inconsistencias y conflictos.

En el trabajo (Aceituna et al. 2014) se presenta un método de verificación de requisitos basados en modelos, llamado NLtoSTD, para verificar documentos de requisitos. El método transforma requisitos escritos en lenguaje natural en diagramas de transición de estados que permiten ayudar a detectar problemas en las propiedades de ambigüedad y completitud. Los autores evalúan la propuesta realizando dos experimentos controlados y los resultados se comparan con un método estándar de inspección de requisitos basado en fallos en listas de verificación.

Otros estudios desarrollan métodos cuantitativos para evaluar la calidad de los requisitos. Mediante el procesamiento del lenguaje, se pueden determinar valores de distintas métricas que sirven de evaluación de las propiedades de los requisitos y de este modo proporcionar valoraciones de calidad. En algunos de estos estudios, además se presentan herramientas que realizan la cuantificación de las métricas de forma automática (Génova et al. 2013; Fabbrini et al. 2001b; Fabbrini et al. 2001a). Usando métodos cuantitativos se puede mejorar la calidad del proyecto realizando una priorización automática de los requisitos (Thakurta 2013; Otero et al. 2010).

El Centro Tecnológico de Calidad de Software de la NASA, propone la conjunción de varias herramientas destinadas a la ingeniería de requisitos de alta calidad (McCoy 2001). Estas herramientas son: ARM (medición automática de requisitos), SCAT (herramienta de análisis de seguridad crítica) y RUT (herramienta case de requisitos de usuario).

En el trabajo (Ott 2013), Daniel Ott presenta un método automático de categorización de requisitos que se encuentran en distintos documentos. El método analiza el texto del requisito y establece una categoría que debe concordar con el tema del documento al que pertenece.

Para resolver la clasificación de requisitos en varias categorías, en el trabajo (Ko et al. 2007) se propone un método que clasifica automáticamente las frases de requisitos de cada categoría usando sólo palabras asociadas a las categorías como representación del punto de vista de los analistas. Los autores evalúan la eficiencia del método realizando experimentos con dos conjuntos de requisitos reales.

Algunos trabajos utilizan técnicas de inteligencia artificial para analizar información relativa a la calidad de requisitos. En el trabajo de investigación publicado en (Mat Jani and Tariqul Islam 2012), se propone que es posible emplear técnicas de Razonamiento Basado en Casos (CBR) y redes de neuronas para mejorar la calidad de requisitos. Los casos correspondientes incluyen el análisis de calidad generado por un software de especificación de requisitos y usando redes de neuronas se pueden encontrar casos similares almacenados en el sistema para comparar la calidad y así proporcionar una respuesta de calidad más acertada. No obstante, el trabajo es sólo una propuesta, no se presenta una demostración real del método.

En el trabajo (Hussain, Ormandjieva, and Kosseim 2007), los autores proponen usar herramientas de detección de ambigüedades en documentos de especificación de requisitos software, donde las frases y párrafos son previamente clasificadas en función de su ambigüedad. Con los resultados de las herramientas y los valores de la clasificación se emplean técnicas de aprendizaje automático para realizar clasificadores en forma de árboles de decisión para detectar ambigüedades de otros documentos del mismo tipo. Este trabajo tiene aspectos similares a la metodología propuesta en la

tesis, pero se diferencia en que sólo se detectan fragmentos ambiguos en documentos de especificación de requisitos software.

En el artículo (Dargan et al. 2014), los autores emplean modelos estadísticos usando métricas de calidad de requisitos para predecir el rendimiento operativo de sistemas de programas de adquisición del gobierno. El artículo describe los resultados esperados de una investigación de tesis doctoral donde se establece que la calidad de los requisitos es de hecho un factor de predicción del rendimiento operativo del sistema final. Las fuentes de los datos que se utilizan son Documentos de Requisitos Operacionales e Informes de Test Operacionales. El análisis de los requisitos textuales se realiza con una herramienta Tiger-Pro (Australia, n.d.), que detecta problemas de calidad lingüísticos. Esta herramienta contiene un diccionario de términos clasificados que si se encuentran en los requisitos pueden revelar problemas. El resultado de la herramienta es una valoración binaria de las características: ambigüedad, completitud, imprecisión y capacidad de testeo. Con estos datos los autores emplean una herramienta software de estadística para hacer una regresión logística con los valores de las características, determinado así un modelo de predicción lineal. El artículo citado es una propuesta y los resultados no han sido aún publicados. Este trabajo tiene similitudes con la metodología propuesta ya que se usan herramientas para obtener métricas de calidad y así crear modelos. Estas métricas de calidad que se utilizan en el artículo, son consideradas en esta tesis como propiedades de los requisitos. En esta tesis, las métricas de calidad son cuantificadores que adquieren valores concretos por el procesamiento de los requisitos, y que se emplean para evaluar estas propiedades.

Otra diferencia es que el objetivo del artículo es mostrar una relación estadística significativa entre la calidad de los requisitos y el rendimiento de sistemas. En la metodología propuesta el objetivo es aprender de un conjunto de métricas que representan la calidad de los requisitos, para predecir la calidad que otorgaría un experto.

La especificación de requisitos en lenguaje natural es un tema complejo que está fuertemente vinculado a las exigencias del proyecto y la opinión de los expertos. Las propuestas citadas en este punto que determinan o mejoran de forma automática la

calidad de requisitos, no reflejan la flexibilidad necesaria para incluir la información proporcionada por el experto adaptándola a cada proyecto. La metodología propuesta en esta investigación se compone de proceso que propone analizar la información de un conjunto de requisitos clasificados que proporciona un experto. Se emplean técnicas de aprendizaje automático para estimar la calidad de los requisitos, con la finalidad de predecir la clasificación que otorgaría el experto. El resultado de la metodología es un clasificador de requisitos generado automáticamente que emula la estimación de calidad del experto adaptándose a la información presentada, por lo que es flexible a las exigencias de calidad en diferentes proyectos, siendo esta la aportación diferenciable de la investigación en relación con los trabajos citados.

2.5 PROCESAMIENTO DEL LENGUAJE NATURAL

El procesamiento del lenguaje natural estudia la interpretación de textos en lenguaje natural intentando superar la ambigüedad del lenguaje mediante el vocabulario, la sintaxis y la semántica del mismo.

Surge de la necesidad de indizar los textos en lenguaje natural y siendo sus aplicaciones muy diversas:

- Interfaces de bases de datos: acceso a la información de bases de datos por consultas en lenguaje natural
- Minería de textos y de contenido: determinar el conocimiento implícito de los documentos a partir de una colección documental.
- Extracción de información: identificar proposiciones relevantes del texto (Cowie and Lehnert 1996).
- Comprensión del lenguaje: representar formalmente texto en lenguaje natural mediante conceptos y sus relaciones.
- Traducción automática: traducción de texto a otro idioma.
- Procesamiento de voz: interacción hombre máquina por medio de la voz.
- Generación de texto: crear un nuevo texto a partir de otros. Una aplicación típica son los generadores de resúmenes.

- Sistemas de corrección ortográfica y de estilo: corrigen los errores gramaticales y de estilo y suelen estar incorporados a los editores de texto como correctores ortográficos.
- Indización automática: Destaca los descriptores representativos de un documento para representar su contenido.

El procesamiento del lenguaje natural (PLN) se divide en varias etapas (Dale 2000): tokenización, análisis léxico, análisis sintáctico, análisis semántico y el análisis pragmático.

2.5.1 Tokenizadores

La Tokenización junto con la Segmentación de Frases se consideran fases de preproceso dentro del PLN, no son por tanto objetivos prioritarios en los sistemas de PLN sin olvidar que repercuten sobre los demás procesos.

La Tokenización se encarga de delimitar las palabras (tokens) identificando sus secuencias de caracteres y agrupándolas por las dependencias establecidas entre ellas.

En las lenguas con alfabeto latino se puede considerar como regla general que los tokens corresponden a secuencias de caracteres delimitadas por separadores como el espacio, la coma, el punto, el punto y coma y otros signos de puntuación.

La tipología morfológica de la lengua, la determinación del sentido contextual de las palabras, la particularidad de la aplicación, el conjunto de caracteres y el corpus documental son las principales causas de la problemática de esta etapa.

Entre los ejemplos de formaciones problemáticas destacan las abreviaturas (Dr., D., etc.), las siglas (O.N.U., U.S.A., etc.), los datos estructurados como fechas, números y direcciones (24-08-2008, dos billones, C\ Azahar 28), las contracciones (I'm, o'clock, etc.), locuciones y términos compuestos (in front of, look at, etc.).

Tradicionalmente las técnicas empleadas para la segmentación de frases y la tokenización se basaban de empleo de gramáticas regulares. Los avances con miras a mejorar los resultados frente a formaciones atípicas han consistido en la inclusión de

reconocedores de abreviaturas y de listas de excepciones. Además de atender al corpus seleccionado para realizar una implementación específica. Riley (Riley 1989) alcanzo una precisión del 99,8% (sobre el Brown Corpus) entrenando sistemas mediante árboles de regresión que clasificaban aspectos contextuales (longitud de la palabra, signos de puntuación y tipología de las abreviaturas).

Reynar y Ratnaparkhi (Reynar and Ratnaparkhi 1997) alcanzaron 98% de precisión entrenado un modelo estadístico con corpus anotados a mano. En el Sistema Satz, construido por Palmer y Hearst (Palmer and Hearst 1997) fue del 99,5% en la desambiguación de casos de abreviaturas en inglés especialmente difíciles. También el Sistema Alembic (Aberdeen et al. 1995) obtuvo una precisión del 99,1% incorporando reglas, listas de abreviaturas y marcas para detectar fechas, números y datos temporales.

2.5.2 Análisis léxico

La misión del análisis léxico consiste en la obtención de las etiquetas candidatas de cada palabra (token).

En los lenguajes naturales que presentan un paradigma de inflexión complejo es conveniente realizar un estudio previo que identifique los diferentes grupos de inflexión (género, número, irregularidad verbal, etc.). De esta forma nos podremos referir a un término mediante las raíces involucradas en él y las inflexiones desde las que se realiza la derivación de formas. Esta aproximación, también nos permite la lematización, la reducción de las formas flexivas de los términos a sus lexemas correspondientes (Marti and Llisterri 2002).

Una forma eficiente de implementar analizadores léxicos es mediante autómatas finitos (Hopcroft and Ullman 1979). Aunque en su uso tradicional de construcción de compiladores (Aho, Sethi, and Ullman. 1986) se reconocen lenguajes limitados y perfectamente estructurados, su aplicación al PLN es adecuada.

La construcción de estos autómatas finitos se puede realizar combinando dos etapas:

1. La construcción de un autómata parcial que va creciendo conforme se incorporan términos a reconocer.
2. La minimización del autómata parcial resultante con el fin de obtener un autómata con menos estados y transiciones y que sea equivalente al original (Hopcroft and Ullman 1979).

Una forma construcción más rápida es mediante un método incremental que realice las operaciones de minimización en línea, es decir, conforme se inserta las palabras a reconocer (Daciuk, Jan, Stoyan Mihov, Bruce W. Watson 2000). Otros trabajos se basan en reglas, el de Kaplan y Martin en Xerox PARC (Kaplan and Kay 1994).

En este tipo de procesos hay que tener en cuenta todas las posibles irregularidades producidas en su mayoría por alternancias vocalistas en la raíz o en las desinencias (Sproat 2000).

Otros métodos de análisis léxico surgen de la necesidad de tratar palabras desconocidas. Los procesos productivos y derivativos pueden proporcionar palabras que no están incluidas en el diccionario morfológico estático.

El modelo de morfología de (Koskenniemi 1983) ha sido la base de multitud de trabajos. Construye analizares léxicos a partir de la especificación de reglas morfológicas de flexión y derivación. Se basa en la comparación de cada cadena presente en el lenguaje (por ejemplo, la siguiente forma del verbo volar, vuelo) con su correspondiente cadena teórica, es decir, la que presentaría de no existir irregularidades (para este supuesto, volo).

En este ejemplo vuel y vol se consideran alomorfos, es decir, formas alternativas del mismo morfema. Igualmente, de los términos en inglés sky y skies se determinan que sky y ski son alomorfos.

El modelo de Koskenniemi se basa en la representación de cualquier palabra mediante la correspondencia directa entre la cadena que tiene asignada y su cadena teórica. Las transformaciones letra a letra se suelen realizar mediante traductores de estado finito (Beesley and Karttunen 2003).

Basadas en este tipo de modelos surgen herramientas como Mmorph (Russell 1995) de libre distribución enmarcada en el proyecto MULTEXT.

2.5.3 Análisis sintáctico

El principal objetivo del análisis sintáctico es explicitar las relaciones sintácticas de los textos (Marti and Llisterri 2002) facilitando su interpretación semántica en la siguiente etapa. El conjunto de técnicas de análisis sintáctico que determinan la estructura sintáctica de las frases se denomina Parsing (Christer and Wiren 2000).

El análisis sintáctico presenta una serie de problemas como son: la definición de una gramática con la suficiente cobertura del lenguaje, la resolución de las ambigüedades estructurales y los costes asociados a la definición de la gramática por los lingüistas y al procesado computacional.

Los problemas mencionados, y el hecho de que no todas las aplicaciones requieren un análisis completo de la oración, han propiciado tipos de análisis más superficiales. Por ejemplo, en extracción de información o en búsqueda de respuestas la identificación de sintagmas nominales o verbales sencillos mediante analizadores de frases (spotters y chunkers).

La estrategia más habitual suele ser realizar análisis globales a nivel superficial y limitar los análisis precisos a ámbitos locales. No obstante, la fragmentación del texto en unidades analizables, la decisión del análisis más adecuado y la adaptación de gramáticas generales a corpus específicos sigue presentado dificultades (Marti and Llisterri 2002).

En el análisis parcial (Abney 1997) se emplea tanto técnicas deductivas como inductivas.

1. Las deductivas se basan en el conocimiento, definiendo un conjunto de reglas gramaticales que representan las estructuras sintácticas y que realizan el análisis utilizando técnicas de estados finitos. Se han desarrollado con estas técnicas analizadores para el inglés (Abney 1996), francés (Salah and Chanod. 1997) y el español (Gala 1999).

2. Las inductivas se apoyan en el aprendizaje automático sobre corpus de entrenamiento. Entre las aproximaciones más relevantes destacan:
    - Aprendizaje basado ejemplos (Veenstra and Bosch 2000)
    - Métodos estadísticos (Koeling 2000); (Molina and Pla 2002)
    - Aprendizaje de reglas (Déjean 2000)
    - Soporte vectorial (Kudo and Matsumoto 2001)
    - Redes de neuronas (Carreras and Lluis 2004)
    - Métodos combinados (Tjong, Hallam, and Hartley 2006)

La mayoría de estos métodos considera como características relevantes para el proceso de desambiguación: las palabras, las etiquetas morfosintácticas y las etiquetas en un entorno de contexto de hasta tres posiciones sobre el término a analizar

2.5.4 Análisis semántico y desambiguación semántica

El objetivo del análisis semántico es interpretar el significado de expresiones. Su principal problema es la ambigüedad semántica. Para solucionarlo necesita el análisis de este proceso de aspectos contextuales.

La ambigüedad se puede presentar de tres formas:

1. Ambigüedad léxica: se produce por la coexistencia de más de un sentido o más de una categoría gramatical en la misma palabra. Está inducida por la polisemia o por la homonimia.
2. Ambigüedad referencial: se debe a las anáforas, es decir, a que algunos términos son referenciados por partículas como los pronombres haciendo imposible su interpretación sin considerar el contexto.
3. Ambigüedad de alcance: se produce por la presencia de elementos, como las partículas negativas, que alteran el sentido de la interpretación.

2.5.5 Análisis Pragmático

Dentro del PLN, el análisis pragmático es de las fases menos desarrolladas ya que depende de las etapas de análisis anteriores. Su misión es la interpretación de cada

frase en función del contexto (Poesio 2000) por lo que debe tener en cuenta las informaciones semánticas de las frases próximas.

2.6 PATRONES EN EL LENGUAJE NATURAL

La segunda metodología propuesta en esta tesis tiene como objetivo la generación automática de patrones sintáctico-semánticos mediante el procesamiento de corpus de requisitos. Los patrones sintáctico-semánticos contienen en su estructura una relación entre restricciones de categorías gramaticales y una posición concreta dentro del patrón.

Esta idea de patrón lo propuso el autor Jeremy Dick en el libro Requirements Engineering (Hull, Jackson, and Dick 2010) donde presentó un tipo de patrones llamados *boilerplates* definidos como: "un lenguaje para expresar requisitos". El autor expuso que un conjunto de Boilerplates permiten coleccionar y clasificar las diferentes maneras de expresar ciertas clases de requisitos.

Los boilerplates, dentro del campo de ingeniería de requisitos, han sido definidos como "una plantilla textual de especificación de requisitos, que están basados en patrones predefinidos con el fin de reducir ambigüedad y asegurar la consistencia en la forma de expresar los requisitos " (Daramola, Sindre, and Stalhane 2012).

Se muestra a continuación un ejemplo de boilerplates contenido en el libro Requirements Engineering (Hull, Jackson, and Dick 2010):

- The <system> shall <function><object> every <performance><units>

Los términos que aparecen entre los símbolos “<” y “>” representan un espacio contenedor donde se puede insertar vocabulario que cumpla la restricción indicada. Así, se podrían ajustar al boilerplate de ejemplo los siguientes requisitos:

- The coffee machine shall produce a hot drink every 10 seconds

  Siendo:
  <System> = coffee machine
  <function> = produce

<object> = a hot drink
<performance>= 10
<units> = seconds

- The coffee machine shall produce a cold drink every 5 seconds

  Siendo:
  <system> = coffee machine
  <function> = produce
  <object> = a cold drink
  <performance>= 5
  <units> = seconds

Con este ejemplo se deduce que se pueden generar una gran multitud de requisitos caracterizados todos por las restricciones que impone el boilerplate. Como expresa el autor Jeremy Dick en la página web *Requirements Engineering* (Hull, Jackson, and Dick, n.d.), "un requisito es por tanto un boilerplate más una serie de atributos seleccionados que encajan con la significación de los espacios contenedores".

En la misma página web, el autor amplia las ventajas que expone en el libro *Requirements Engineering* (Hull, Jackson, and Dick 2010) sobre el uso de boilerplates en la definición de requisitos. Estas ventajas son:

- Una ayuda en la articulación: ser capaz de aprovechar una paleta de boilerplates clasificados ayuda a encontrar maneras efectivas de expresar determinados tipos de requisitos o restricciones.
- Uniformidad del lenguaje: teniendo en cuenta siempre la reutilización de boilerplates existentes, los mismos tipos de requisitos siempre serán expresados de la misma forma, lo que conlleva a un uso consistente del lenguaje para expresar requisitos.
- La abstracción de los valores clave: cuando se expresa un boilerplate más valores de los atributos cada atributo que participa en los boilerplates se convierte en una enumeración, por ejemplo, de todos los "modos de

funcionamiento", o todas las "funciones" del sistema. Esto puede ayudar considerablemente en el procesamiento y el modelado de los requisitos.

- Control sobre las expresiones: con la ayuda de una herramienta, los cambios globales en la forma de los requisitos, que afecta a lo que se quiere expresar, se pueden realizar con sólo cambiar una parte del boilerplate, por ejemplo, cambiar "will" por "shall".
- La protección de la información clasificada: a menudo, la única parte clasificada en un requisito son cantidades asociadas a él, por ejemplo, la velocidad de una embarcación, la tasa de fuego. Usando boilerplates, la clasificación de la información se puede focalizar únicamente en los atributos secretos, permitiendo que el resto del requisito pueda ser visto sin uso restringido.

Los boilerplates también pueden ser usados para comprobar si la estructura de un requisito ya definido es correcta. Si la estructura del requisito coincide con la composición del boilerplate, puede establecerse que la estructura es correcta al cumplir con las restricciones establecidas por el boilerplate.

Si la estructura de un requisito no está establecida en ningún boilerplates, deberá ser modificado hasta conseguir una estructura adecuada, o bien, si consideramos que el requisito es correcto, se deberá crear un nuevo boilerplate que lo represente y añadirlo a la colección. Con este fin, el libro *Requirements Engineering* (Hull, Jackson, and Dick 2010) recoge la idea de que un conjunto de boilerplates puede ser ampliado y reutilizado en la consecución de proyectos con el fin de integrar la experiencia adquirida.

Desde su presentación, los boilerplates han sido utilizados en la ingeniería de requisitos para asegurar que los requisitos cumplen con una estructura adecuada. Se han presentado trabajos relacionados con el uso de los boilerplates en la ingeniería de requisitos, pero la mayoría de los trabajos se centran en la optimización de los procesos de validación y verificación de los requisitos almacenados en el sistema, y del uso de los boilerplates para redactar nuevos requisitos de calidad.

En (Daramola, Sindre, and Stalhane 2012) se presenta un enfoque ontológico que usa plantillas de patrones predefinidas (boilerplates de requisitos) para facilitar a los ingenieros la especificación de requisitos de seguridad. En el trabajo se presenta una herramienta como forma de implementación de la ontología propuesta.

En el trabajo (Farfeleder et al. 2011) se presenta la herramienta DODT. Esta herramienta transforma semiautomáticamente requisitos escritos en lenguaje natural, dentro de sistemas embebidos, para ser reconocidos estructuralmente con boilerplates de requisitos. El objetivo es reducir el esfuerzo de realizar manualmente la verificación y validación de requisitos con boilerplates. DODT se sustenta de un dominio ontológico de alta calidad que se tiene que proporcionar antes de usar la herramienta.

En el trabajo desarrollado en la universidad de Luxemburgo (Arora et al. 2013), se presenta RUBRIC, una herramienta para chequear requisitos de forma automática usando boilerplates. En el trabajo citado los boilerplates tienen que estar previamente definidos en el sistema y la propuesta se centra en mejorar la optimización en el reconocimiento de la estructura de los requisitos por los boilerplates. Los autores diferencian su propuesta con la herramienta DODT, en que no se necesita una base ontológica para realizar proceso de chequeo de requisitos y enfatizan que el resultado no se ve comprometido por tener glosarios de términos incompletos.

Generación de patrones automáticamente

Se han realizado multitud de trabajos en el ámbito del procesamiento del lenguaje natural donde el objetivo es la generación automática de patrones. El objetivo de estos patrones es mejorar las bases del conocimiento por el tratamiento de textos permitiendo, entre otras características, optimizar los análisis estadísticos sobre el texto y mejorar la búsqueda y la indización de partes relevantes.

En el trabajo (Ellen Riloff 1996) se generan patrones automáticamente de texto sin anotaciones. La autora en un trabajo anterior (E Riloff 1996) presenta "AutoSlog": un sistema de construcción de diccionarios que se crea mediante la extracción automáticamente de patrones usando reglas heurísticas sobre textos con anotaciones

sintácticas y semánticas. Adaptando este trabajo y combinándolo con técnicas estadísticas posibilita la eliminación de las dependencias de los patrones con las anotaciones del texto. Los patrones que se generan en el proceso son ordenados en un ranking para eliminar aquellos menos relevantes. Se obtiene como resultado un diccionario de patrones independientes de anotaciones semánticas y sintácticas.

En la patente (Rehberg, C. 2012) con título "Automatic pattern generation in natural language processing" se presenta una invención relativa al significado de las palabras dentro de una frase, con el propósito de mejorar la búsqueda y reutilización de información basada en la comprensión de los componentes de las frases de un corpus. En la propuesta se generan patrones de equivalencia del significado de las palabras que componen las frases. Mediante la base de conocimiento, de diccionarios, base de datos, categorías de palabras y asociación semántica, se crean unidades de patrones para relacionar palabras con el significado en función de los términos que componen las frases.

La generación de patrones que se presenta en esta tesis se realiza sobre el ámbito de los requisitos, donde el lenguaje, aun formando parte del lenguaje natural, presenta mayores formalizaciones y restricciones. Con esto se consigue reducir la ambigüedad y mejorar así la calidad del conjunto de los requisitos.

En la invención (Moreno et al. 2013) con título “Métodos de generación de patrones sintácticos” (en explotación desde 2014) se desarrolla un método en el que se implementan distintos pasos que permiten la generación automática de patrones de indexación, teniendo como origen un corpus y como salida una lista de patrones ordenados por frecuencia. Incluyendo además opcionalmente, varias funciones especiales que organizan y representan la información intermedia obtenida, y capacidades de expansión de los patrones generados a otros formatos jerárquicos.

La invención citada permite disponer de patrones complejos a nivel de frase, que se pueden expresar por otros más simples, con la ventaja de identificar zonas más amplias de los textos de las que extraer semántica con mayor precisión. En consecuencia, tienen una mayor riqueza semántica. Además, su obtención es totalmente automática tomando como base documentos en lenguaje natural.

La investigación desarrollada en esta tesis incorpora el contenido de la invención propuesta anteriormente para generar automáticamente patrones sintáctico-semánticos de requisitos. Estos patrones representarán estructuras correctas de requisitos por lo que podrán ser usados en la orientación en el momento de redactar requisitos con estructuras sintáctico-semánticas correctas.

2.7 INGENIERÍA DEL SOFTWARE BASADA EN BÚSQUEDAS

La ingeniería de software basada en búsquedas conocida por sus siglas *SBSE* (Search Based Software Engineering) es un enfoque de esta disciplina donde se aplican técnicas de optimización para resolver problemas con la ingeniería del software (Mark Harman et al. 2012).

El término SBSE fue acuñado en el trabajo (M. Harman and Jones 2001) los autores establecen que la ingeniería de software es ideal para la aplicación de las técnicas de búsqueda meta-heurísticas, tales como algoritmos genéticos, recocido simulado y aprendizaje automático, entre otros.

SBSE ha sido usado en multitud de problemas en Ingeniería de Software. Por ejemplo, en el trabajo (London et al. 2007) los autores abordan el problema conocido como MONRP, por sus siglas en inglés Multi-Objetive Next Release Problem, en la que se extienden de la función fitness simple sobre el problema Next Release Problem (NRP) para resolver el problema multi-objetivo. El problema (NRP) es un ejemplo del problema de búsqueda de selección de subconjunto de características directamente relacionado con SBS

Uno de los objetivos planteados en la tesis es la clasificación de requisitos atendiendo a su calidad, y de la forma en la que se resuelve el problema, se puede establecer que la meta es encontrar una función fitness que minimice las diferencias entre la calidad estimada y la clasificación real. Y dado que se utilizan técnicas de aprendizaje automático para poder determinar y corregir la calidad de los requisitos, y siendo la investigación de la Ingeniería de Software uno de los principales demandantes de requisitos, se ha dedicado un apartado [*3.2.1.1 Formalización*

matemática de modelos de optimización de los requisitos] a la definición formal basada en SBSE de la investigación propuesta.

2.8 DEFINICIÓN DE METODOLOGÍA

El término metodología es usado frecuentemente en las investigaciones y su significado en ocasiones no está claro. En general no se distingue entre metodología, proceso, método y técnica; tanto en los significados como entre sus diferencias. En este punto vamos a aclarar los conceptos ya que son usados en el desarrollo de esta investigación.

La palabra metodología es considerada erróneamente sinónimo de la palabra proceso (J. N. Martin 1996) (Estefan 2008). Las definiciones acuñadas por Martin (ver Fig. 6) sobre metodología, método, proceso y herramientas son:

- "Un proceso (P) es una secuencia lógica de tareas realizadas para lograr un objetivo particular. Un proceso define *qué* se debe hacer, sin especificar *cómo* lo realizará cada tarea. La estructura de un proceso se compone de múltiples niveles agregados, que permiten el análisis y la definición de varios niveles de detalle, para soportar las diferentes necesidades.
- Un método (M) consiste en las técnicas para realizar una tarea, en otras palabras, define el *cómo* de cada tarea. En cualquier nivel, las tereas de los procesos son realizadas por métodos. Sin embargo, cada método es también un proceso en sí mismo, con una secuencia de tareas a ser desarrolladas por ese método en particular. En otras palabras, el *cómo* en un nivel de abstracción se convierte en el *qué* del siguiente nivel.
- Una herramienta (H) es un instrumento que, cuando es aplicado a un método particular, puede mejorar la eficiencia de la tarea; aplicándose correctamente y por alguien con las apropiadas habilidades y suficiente entrenamiento. El propósito de una herramienta debe ser facilitar el cumplimiento de la tarea. En un sentido más amplio, una herramienta mejora el *qué* y el *cómo*. La mayoría de las herramientas usadas para apoyar la ingeniería de sistemas son herramientas software, también conocidas como herramientas de ingeniería

asistidas por ordenador, CAE (por sus siglas en inglés: Computer Aided Engineering).

- Según estas definiciones, una metodología puede ser definida como una colección relacionada de procesos, métodos, y herramientas. Una metodología es esencialmente una receta y puede ser considerada como aplicación de los procesos, métodos y herramientas en una clase de problemas que tengan características comunes (Bloomberg and Schmelzer 2006)".

Fig. 6 PMTE Pirámide y sectores

Asociado a estas definiciones, el Entorno (E) son los alrededores, los objetos externos, condiciones, u otros factores que influyen en las acciones de un objeto, persona o grupo (J. N. Martin 1996). Estas condiciones pueden ser sociales, culturales, personales, físicas, organizacionales o funcionales. El propósito del entorno de un proyecto debe ser integrar y apoyar el uso de las herramientas y métodos utilizados en ese proyecto. En consecuencia, un entorno permite (o no permite) el *qué* y el *cómo*.

A continuación, se muestra una representación gráfica de las relaciones entre los llamados elementos PMHE (Procesos, Métodos, Herramientas y Entorno) y los efectos de la tecnología y las personas.

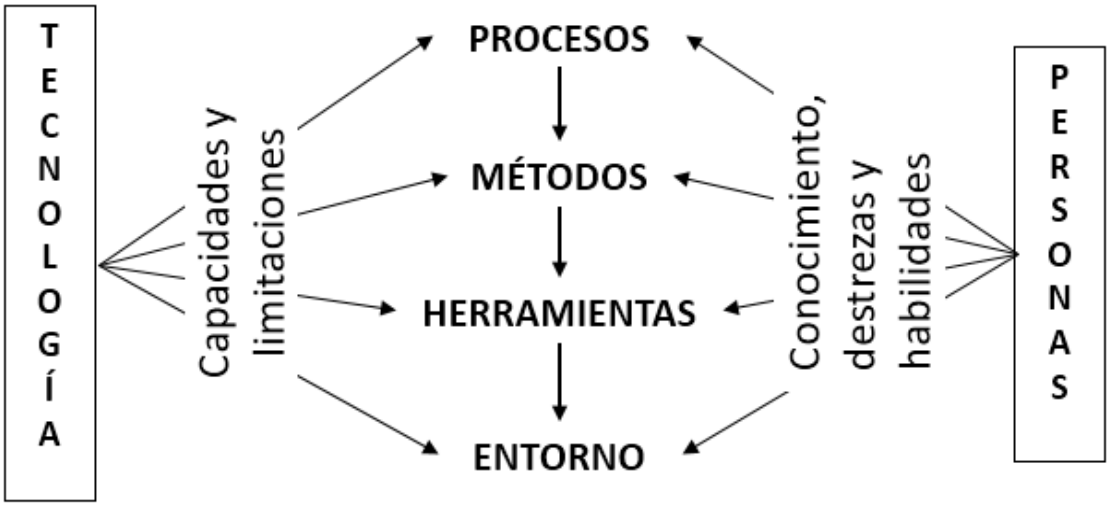

Fig. 7 PMHE elementos

Las capacidades y las limitaciones de la tecnología deben tenerse en cuenta al desarrollar el entorno de desarrollo de ingeniería. La tecnología no debe ser utilizada sólo por el bien de la tecnología. Al elegir la combinación adecuada de elementos PMHE, hay que considerar los conocimientos, habilidades y destrezas (CHD) de las personas involucradas. Cuando son utilizados nuevos elementos PMHE, con frecuencia los CHDs de las personas deben ser mejorados mediante una formación especial (J. N. Martin 1996).

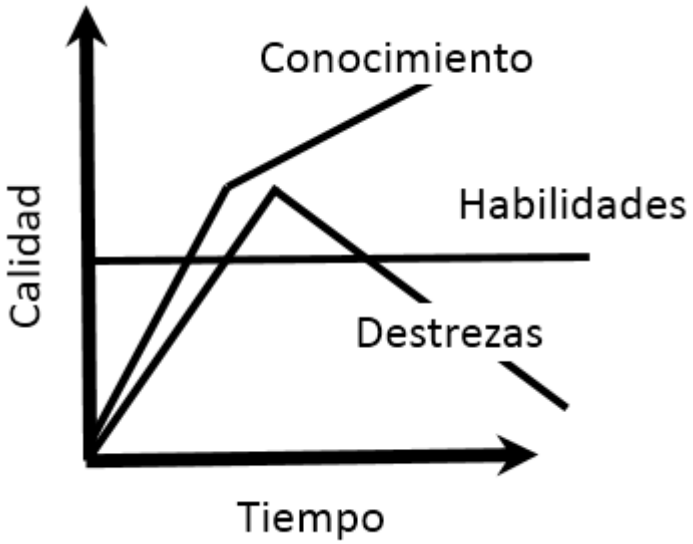

Fig. 8 Tiempo en CHD

La habilidad es más o menos constante durante toda la vida, mientras que el conocimiento por lo general aumenta (ver Fig. 8). Las destrezas, tales como el uso de ciertas herramientas, tienen su pico hasta después de la universidad y decrece a partir de entonces, a menos que se tenga una atención especial para mantener estas habilidades al día. La compra de costosas herramientas sin proporcionar entrenamiento adecuado, conlleva a empeoramiento en el desarrollo del proceso, ya que las herramientas pueden ser mal utilizadas, o no utilizarse en absoluto (J. N. Martin 1996).

En los siguientes capítulos se presenta la metodología propuesta en esta investigación siguiendo las definiciones desarrolladas por Martin.

CAPÍTULO 3: DESARROLLO DE LA INVESTIGACIÓN Y MARCO EXPERIMENTAL

Como se ha citado en la sección, una metodología se compone de procesos, tareas, métodos y herramientas. En este capítulo se expone el contenido de la investigación detallando cada una de las partes que componen la metodología.

3.1 FASES DE LA METODOLOGÍA

A continuación, se presentan las fases de la metodología.

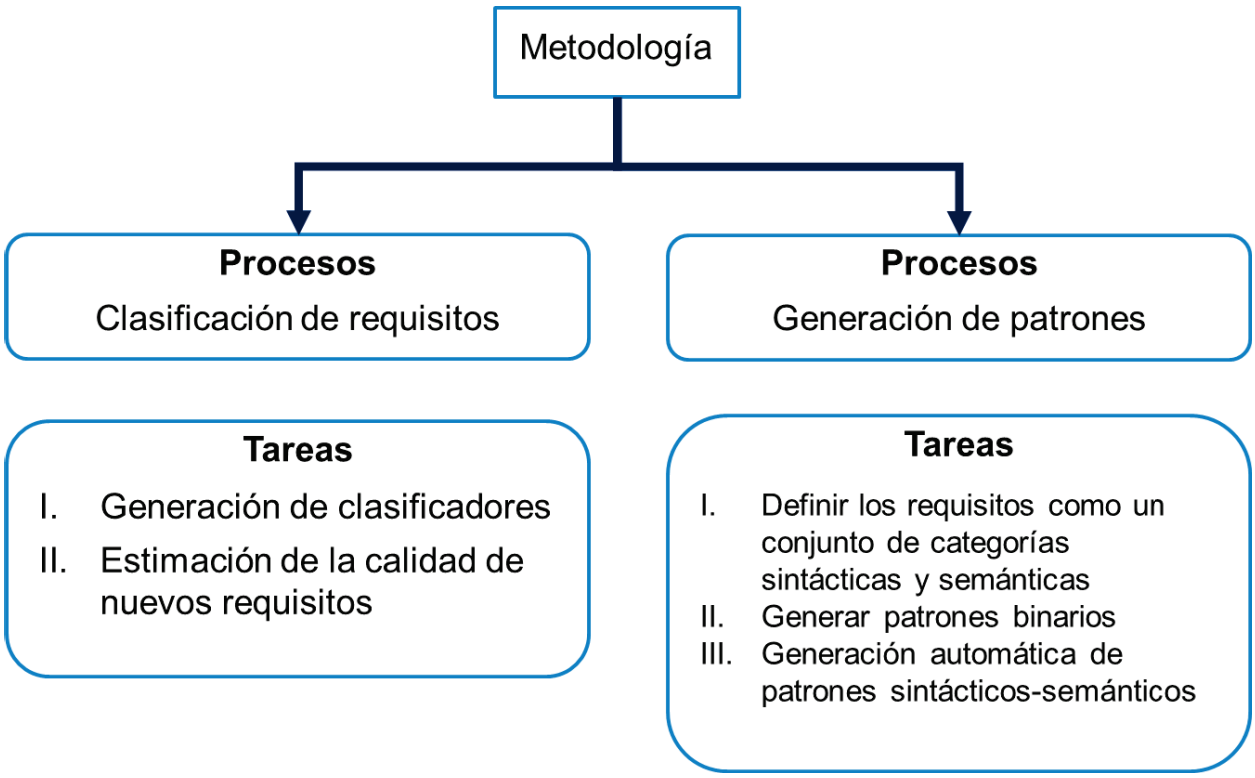

Fig. 9 Metodología: procesos y tareas

La metodología se compone de dos procesos destinados a mejorar la calidad de requisitos realizando procesamiento automático. El proceso llamado Clasificación de requisitos propone la generación automática de clasificadores para estimar la calidad

de nuevos requisitos. Este proceso dará a conocer la calidad de los requisitos que componen las especificaciones de los proyectos.

Con el fin de mejorar la calidad de nuevos requisitos, la metodología se compone de un proceso que tiene como fin la generación automática de patrones sintáctico-semánticos. Estos patrones permitirán la orientación en la redacción de nuevos requisitos.

Para cada uno de los procesos se presentarán las tareas y métodos que los componen, un ejemplo de uso, los resultados del caso de estudio, y los beneficios derivados del desarrollo de la investigación.

3.2 PROCESO PARA LA CLASIFICACIÓN DE REQUISITOS ATENDIENDO A SU CALIDAD

El objetivo del primer proceso que se presenta de la metodología es emular a un experto en la asignación de calidad de requisitos. No obstante, la calidad es un concepto ambiguo que depende de la valoración de distintos criterios, así diferentes usuarios pueden valorar de forma distinta la calidad de un mismo requisito. Para estimar la calidad, el proceso recibe como entrada un conjunto de requisitos clasificados por su calidad. Esta valoración debe ser realizada por el experto de requisitos del proyecto. Es sobre este conjunto donde se realizará un proceso de aprendizaje con la finalidad de conocer los criterios que el experto emplea para determinar la calidad de los requisitos.

Los criterios de calidad son caracterizados por un conjunto de métricas que han sido elaboradas por distintos autores y organizaciones (Génova et al. 2013; Alexander and Stevens 2002; (Incose) 2015) y que representan en valores numéricos la calidad de cada requisito. Basándose en estos valores, las tareas del proceso proponen construir un clasificador que determine qué valores tienen que tener las métricas para designar a los requisitos con buena o mala calidad.

Los métodos de la metodología utilizan una representación de cada requisito del corpus que el experto clasificó en función de la calidad. Esta representación es constituida por el conjunto de métricas. Utilizando algoritmos de aprendizaje automático se puede identificar la relación de las métricas de cada requisito con la clasificación del experto, dando como resultado un clasificador que clasificará con buena o mala calidad a los requisitos en función de los valores del conjunto de métricas que los representan.

A continuación, se muestra una representación gráfica de las tareas y los métodos que componen el proceso de clasificación de requisitos

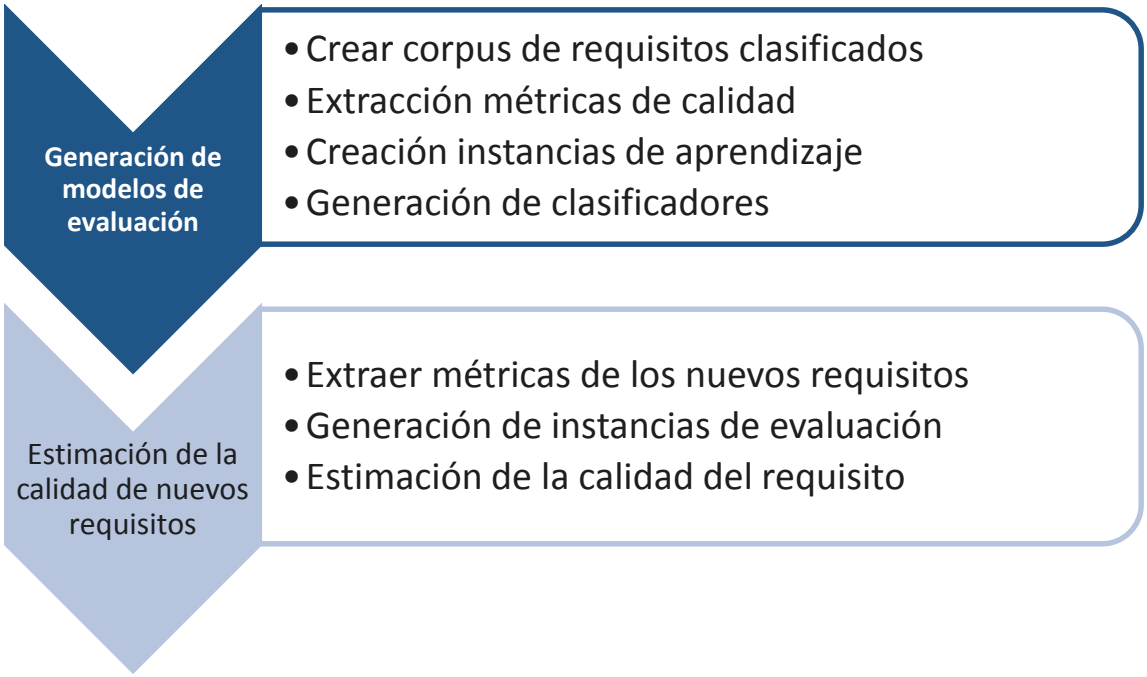

Fig. 10 Proceso de clasificación de requisitos: tareas y métodos

3.2.1 Definición del proceso para la clasificación de requisitos atendiendo a su calidad

En esta sección se detalla el proceso de la metodología para la clasificación de calidad de requisitos. El proceso se divide en dos tareas: a) generación de modelos de evaluación y b) estimación de la calidad de nuevos requisitos.

- *Tarea 1 - Generación de modelos de evaluación*: se generan clasificadores a partir de un conjunto de métricas asociadas a requisitos previamente clasificados por un experto en función de su calidad.
- *Tarea 2 - Estimación de la calidad de nuevos requisitos*: los clasificadores generados en la tarea anterior se utilizan para estimar la calidad de otros requisitos.

3.2.1.1 Formalización matemática de modelos de optimización de los requisitos

A continuación, se presenta el problema en la formalización SBSE (Search-Based Software Engineering), mediante una definición, una representación y la función de fitness.

Dado:

- Un conjunto textual de requisitos $R = \{r_1, \dots, r_n\}$
- Un conjunto de métricas de corrección aplicadas a los requisitos $M = \{m_1, \dots, m_k\}$ tal que $m_j: R \to \mathbb{R}\ (1 \leq j \leq k)$
- El conjunto de medidas de calidad de corrección proporcionadas por el experto sobre el conjunto de requisitos $C = \{C_1, \dots, C_n\}$, tal que $C_i \in \{0, 1\}$ es la calidad proporcionada por el experto a los requisitos $r_i\ (1 \leq i \leq n)$ donde 0 representa mala calidad y 1 representa buena calidad

Problema:

Encontrar una función $f: \mathbb{R}^k \to \{0, 1\}$ tal que minimice $\sum_{i=1}^{n} | f\big(m_1(r_i), \dots, m_k(r_i)\big) - C_i |$

Representación:

La representación de cada posible solución en una función a intervalos. Esta función puede ser definida mediante un conjunto de reglas.

Función fitness:

El mismo sumatorio expuesto en la definición del problema, es la función de fitness que debe ser minimizada.

3.2.1.2 Métodos de la tarea 1 - Generación de modelos de evaluación

El desarrollo de esta tarea proporciona como resultado un conjunto de clasificadores que podrán estimar la calidad de los requisitos. A continuación, se muestran gráficamente los métodos de esta tarea del proceso de la metodología.

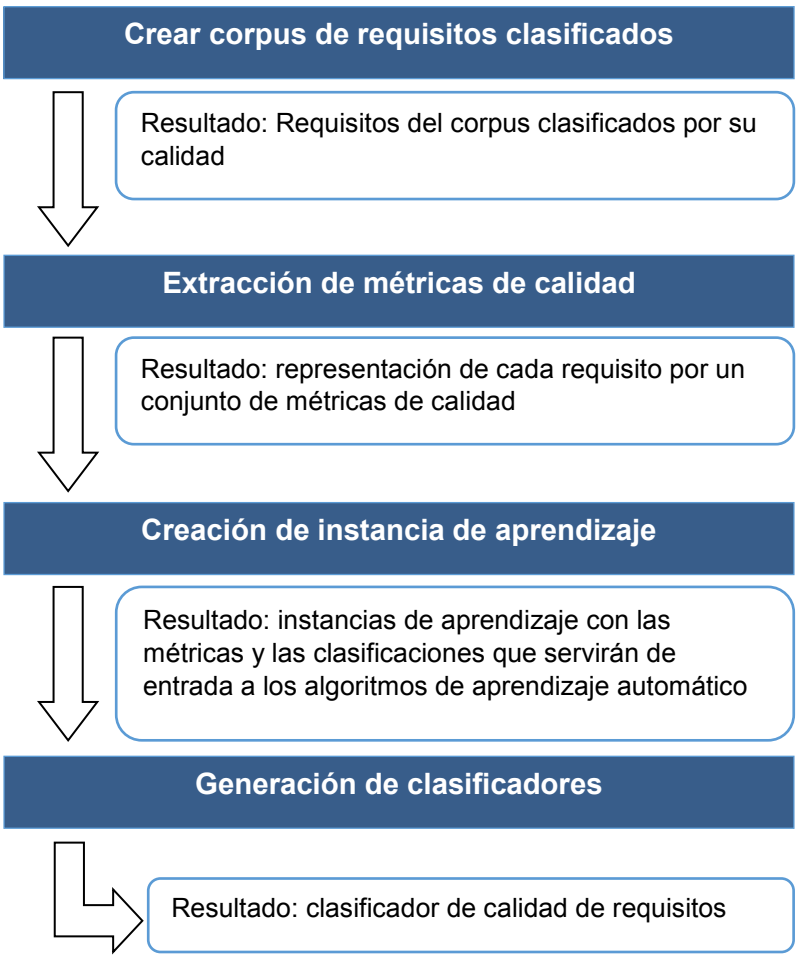

Fig. 11 Métodos de la tarea 1 – Generación de modelos de evaluación

- *Crear corpus de requisitos clasificados*: el primer método de la tarea es crear un corpus de requisitos valorados en función de su calidad por un experto. Así, todo requisito del corpus tendrá una valoración de calidad asociada.
- *Extracción de métricas de calidad*: se deben extraer valores de métricas de calidad de cada requisito del corpus. A partir de este paso cada requisito estará

representado por los valores de las métricas de calidad y la valoración de la calidad estimada por el experto.

- *Creación de instancias de aprendizaje*: con el conjunto de métricas y la valoración de la calidad asociada a los requisitos, se construyen instancias de aprendizaje que servirán para generar los clasificadores. Las instancias de aprendizaje estructuran la información para que sea procesada por los algoritmos de aprendizaje automático.

  La forma de representar la información en las instancias de aprendizaje influye en la capacidad de acierto y en el tiempo invertido en la generación de los clasificadores. Para este proceso se han realizado dos experimentos que implementan dos maneras distintas de construir las instancias de aprendizaje.
- *Generación de clasificadores*: mediante técnicas de aprendizaje automático se usarán las instancias de aprendizaje para construir clasificadores que serán capaces de estimar la calidad de nuevos requisitos.

3.2.1.3 Métodos de la tarea 2 – Estimación de la calidad de nuevos requisitos

Para estimar la calidad de otros requisitos usando los clasificadores generados en la tarea anterior, se debe generar una instancia por cada nuevo requisito con las métricas de calidad empleadas en la primera tarea. La estructura de esta instancia dependerá de la estructura utilizada en las instancias de aprendizaje.

Se muestra gráficamente a continuación los métodos de esta segunda tarea del proceso de la metodología.

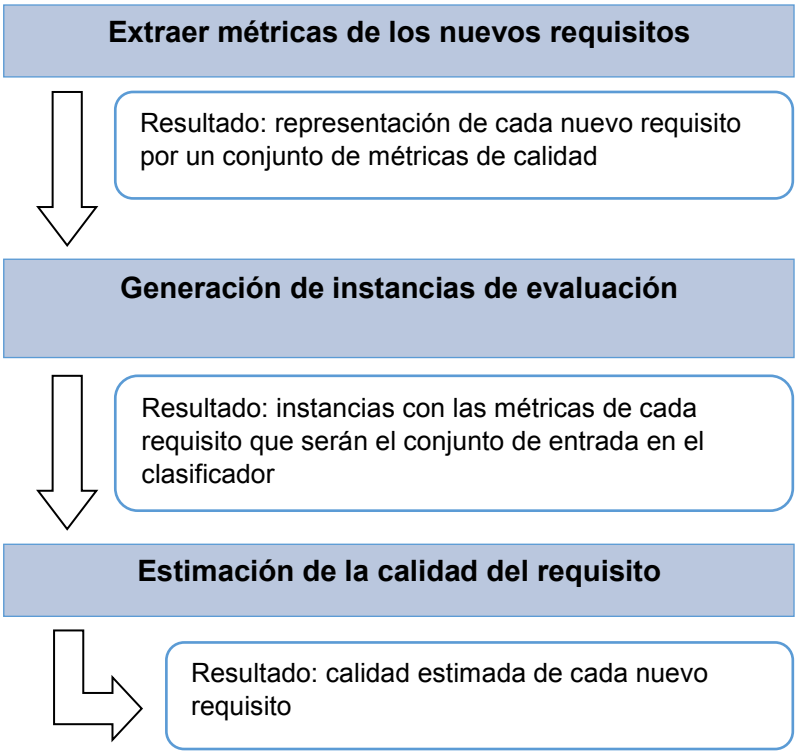

Fig. 12 Métodos de la tarea 2 – Estimación de la calidad de nuevos requisitos

- *Extraer métricas de los nuevos requisitos*: para estimar la calidad de un requisito nuevo con los clasificadores, se deben extraer sus valores de métricas de calidad. Las métricas tienen que ser las mismas que las utilizadas en los métodos de la primera tarea.
- *Generar instancias de evaluación*: una vez extraídas las métricas de un nuevo requisito se pueden usar los clasificadores para estimar su calidad. Para ello, con la información de las métricas, se tienen que construir instancias adecuadas a los clasificadores para que estos reconozcan la información y así poder predecir una valoración de la calidad.
- *Estimación de la calidad del requisito*: la forma de obtener la calidad de un requisito nuevo dependerá de la forma de las instancias de aprendizaje utilizadas en la generación de los clasificadores. Así, puede haber clasificadores que estimen la calidad directamente como resultado de la instancia del nuevo requisito y otros donde la información proporcionada por el clasificador tenga que ser procesada para poder realizar la estimación de la calidad.

3.2.1.4 Herramientas del proceso 1: clasificación de requisitos

En esta sección se exponen las herramientas necesarias en el proceso de la metodología y las utilizadas en el desarrollo de esta tesis.

Extracción de métricas de calidad

Para desarrollar los métodos donde se extraen las métricas de calidad de los requisitos del corpus, se ha utilizado la herramienta Requirements Quality Analyzer (RQA) (Company 2015b). Esta herramienta permite la definición, medición, evolución y gestión de la calidad en la especificación de requisitos. Con esta herramienta se pueden procesar los requisitos y extraer los valores de las métricas de calidad.

Generación de los clasificadores

Para generar los clasificadores que estimarán la calidad de los requisitos, se utilizan los algoritmos de aprendizaje automático que integra la herramienta de análisis de datos Weka (Hall et al. 2009). Concretamente los algoritmos utilizados son los algoritmos de inducción de reglas PART y C 4.5. Con la intención de mejorar la precisión, también se han utilizado combinaciones de clasificadores homogéneos mediante las técnicas Bagging y Boosting, tomando como base los algoritmos anteriores.

3.2.2 Desarrollo del proceso 1: Clasificación de calidad de requisitos

En esta sección se explican los detalles de la propuesta de desarrollo del proceso realizada en esta investigación. Se han realizado dos enfoques distintos para el mismo proceso. Los enfoques difieren en la implementación de las instancias de aprendizaje. Estos dos experimentos se orientan en distintas prioridades: el tiempo de creación y el porcentaje de acierto de los clasificadores. Así, se establecen experimentalmente dos criterios de prioridad para distintos intereses en la resolución del problema. A continuación, se expone la implementación de los dos enfoques; cada uno contendrá la implementación de las dos tareas de la metodología. En ambos enfoques existen métodos donde la implementación tiene que ser la misma, como la composición del

corpus de requisitos o la extracción de métricas de calidad, ya que es necesario un punto de partida común con el fin de obtener distintas soluciones sobre el mismo problema.

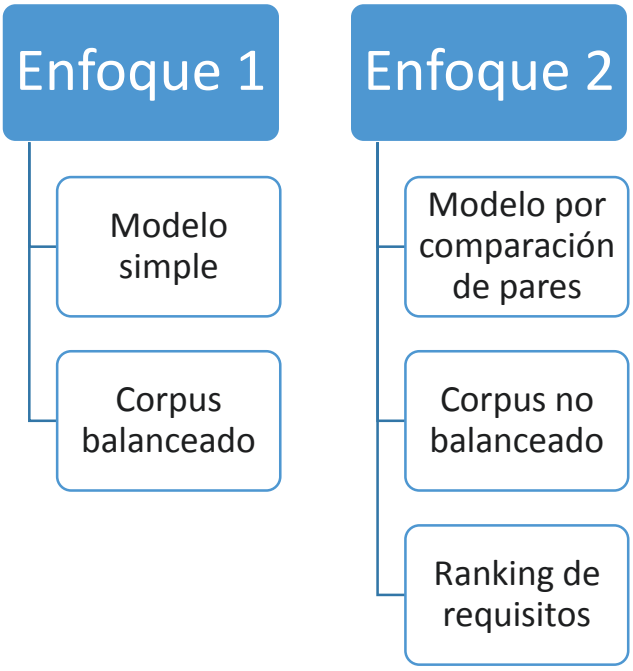

Fig. 13 Enfoques propuestos para la implementación de las instancias

Respecto a la experimentación, se presentan por cada uno de los enfoques los resultados obtenidos, para al final de la sección comparar las dos propuestas. En estos experimentos no se cuantifican indicadores de tipo relacional [2.1.6.4 Indicadores relacionales] ya que no existen relaciones entre los conjuntos de requisitos proporcionados.

3.2.2.1 Enfoque 1: Modelo simple

A continuación, se definen las tareas que componen la implementación del primer enfoque.

3.2.2.1.1 Tarea 1 – Generación de modelos de evaluación

Método: Crear corpus de requisitos clasificados

El primer método de la metodología indica que es necesario contar con un corpus de requisitos clasificados previamente por un experto en función de la calidad. Para el caso de estudio presentado, se ha utilizado un corpus de requisitos facilitado por el Grupo de Trabajo de Requisitos de INCOSE ("INCOSE (International Council on Systems

Engineering)" 2016). Este corpus contiene 1035 requisitos y cada uno de ellos ha sido clasificado por un experto con buena calidad o mala calidad. De los 1035 requisitos 545 tienen buena calidad y 490 mala. Este corpus de requisitos es la base inicial en los dos experimentos presentados para esta metodología.

Método: Extracción de métricas de calidad

Cada requisito del corpus se ha procesado con la herramienta indicada en este capítulo [3.2.1.4 Herramientas de la proceso 1: clasificación de requisitos], Requirements Quality Analyzer (RQA) (Company 2015b), para extraer los valores de las métricas calidad.

El corpus de requisitos está protegido bajo acuerdo de confidencialidad, pero en el Anexo 2 se proporciona una tabla con las métricas extraídas con la herramienta, junto a la clasificación dada por los expertos excluyendo el texto de los requisitos.

A partir de este momento, en los siguientes métodos de la metodología, cada requisito del corpus es representado por su correspondiente conjunto de métricas y la clasificación de la calidad que ha determinado el experto. Los algoritmos de aprendizaje automático procesarán este conjunto de métricas y clasificaciones de calidad para generar clasificadores que permitan predecir la calidad de otros requisitos.

Método: Creación de instancias de aprendizaje

Las instancias de aprendizaje son almacenadas en un formato que es compatible con Weka (Hall et al. 2009), herramienta que forma parte del conjunto de herramientas que indica para este proceso [3.2.1.4 Herramientas de la proceso 1: clasificación de requisitos]. El formato consta de un conjunto de atributos y un conjunto de instancias. El conjunto de atributos corresponde a las métricas de calidad más un valor de clase que representa las dos opciones de calidad (buena calidad o mala calidad). El conjunto de instancias está formado por los valores numéricos de las métricas del conjunto de requisitos de aprendizaje y los valores de la calidad que asignó el experto.

La forma en que se combinan los conjuntos de valores de las métricas de los requisitos para crear el formato de las instancias de aprendizaje, determinan distintas

formas de entrenamiento de los algoritmos, lo que da lugar a los dos enfoques presentados para este proceso.

El primer enfoque ha sido denominado “modelo simple” ya que cada instancia corresponde con el conjunto de los valores de las métricas de sólo un requisito, más la calidad asignada por el experto (ver Fig. 14).

De esta manera, el número de instancias que contiene el modelo simple es el número de requisitos que forman el conjunto de aprendizaje.

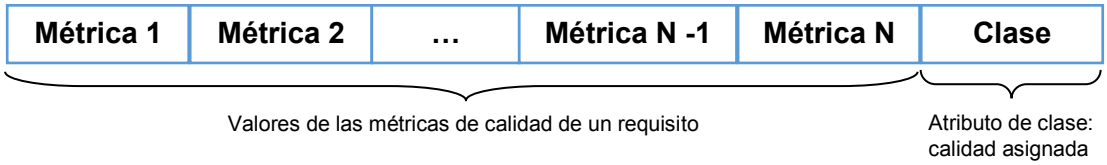

Fig. 14 Formato de una instancia del modelo simple

Método: Generación de clasificadores

Los algoritmos usados para la generación de los clasificadores son los algoritmos PART, C 4.5 y, los algoritmos de clasificador homogéneos Bagging y Boosting que toman como base los algoritmos anteriores.

Los algoritmos reciben como entrada de casos de aprendizaje las instancias creadas con los valores de las métricas de los requisitos. En función de estas instancias los algoritmos inducen reglas con los atributos de las métricas para determinar los valores de clase que representan valores de calidad.

Los experimentos propuestos en esta metodología proporcionan conjuntos de reglas que difieren en la interpretación de la resolución del problema. En este primer enfoque, modelo simple, las reglas determinan la calidad de un requisito en función de los valores que tengan las métricas que lo representa, es decir, el algoritmo aprende los rangos de valores y combinación entre ellos para estimar la calidad de un requisito.

En el segundo enfoque, que se explicará con detalle en el punto [3.2.2.2 Enfoque 2: modelo por comparación de pares], las reglas determinan una valoración de comparación de dos requisitos.

3.2.2.1.2 Tarea 2 – Estimación de la calidad de nuevos requisitos

Para usar los clasificadores en la estimación de la calidad de nuevos requisitos es necesario generar instancias con sus correspondientes valores de métricas y usarlas como casos de entrada en los clasificadores.

Método: Extracción de métricas de calidad

Antes de usar los clasificadores, los nuevos requisitos deben ser procesados para extraer las métricas que los representan. Las métricas usadas son las mismas que las utilizadas en el corpus de requisitos de aprendizaje.

En la implementación de este método de la metodología, los nuevos requisitos han sido procesados por la herramienta Requirements Quality Analyzer (RQA) (Company 2015b) utilizada en la tarea anterior.

Método: Generar instancias con las métricas de cada nuevo requisito

A continuación, se generan las instancias con el conjunto de métricas que representan a los requisitos. El formato de las instancias tiene que ser adecuado a la forma de las instancias de aprendizaje utilizadas en la construcción del clasificador. La implementación de este paso ha sido diferente para los dos enfoques propuestos en este proceso.

Para utilizar los clasificadores generados en este primer enfoque, el formato de las instancias debe contener los valores de las métricas de calidad de los nuevos requisitos en el mismo orden que las instancias de aprendizaje. En función de estos valores, el clasificador podrá estimar un valor de calidad.

Método: Estimar la calidad de los nuevos requisitos

La estimación de la calidad de los clasificadores con el modelo simple es inmediata. Como salida, el clasificador otorga un valor de clase al conjunto de métricas del nuevo requisito; este valor es la calidad estimada por el clasificador como la más probable que otorgaría el experto.

3.2.2.1.3 Ejemplo de estimación de la calidad de un nuevo requisito

A continuación, se muestra un ejemplo de uso de la metodología donde se estima la calidad de dos nuevos requisitos.

Para estimar la calidad de los nuevos requisitos, se asume para el ejemplo que se ha generado un clasificador con el algoritmo C 4.5, que llamaremos Clas-C4.5, procesando un corpus de requisitos clasificados en función de la calidad y proporcionado por un experto.

Los nuevos requisitos son:

- Requisito 1: Typically, the **interface module** must be able to handle at least one fast connection with the **service module** and one connection with the predefined **data base module**.
- Requisito 2: Typically, the **interface component** must be able to handle at least one fast connection with the **service component** and one connection with the predefined **data base component**.

El proceso comienza con la extracción de las métricas de cada requisito. En la Tabla. 1 se muestran las métricas y valores, enfatizando aquellas métricas donde difieren los dos requisitos.

Tabla. 1 Valores de las métricas de los requisitos del ejemplo

Métricas	Requisito 1	Requisito 2
Nº párrafos	1	1
Nº palabras	27	27
Legibilidad	**14.6**	**16**
Caracteres entre signos de puntuación	**163**	**171**
Términos Conectores	1 – and	1 – and
Términos negativos	0	0

Términos de control de flujo	0	0
Términos implícitos	0	0
Términos ambiguos	2 – typically, fast	2 – typically, fast
Términos incompletos	0	0
Términos especulativos	0	0
Términos subjetivos	0	0
Términos racionales	0	0
Términos de diseño	1 – data base	1 – data base
Términos imperativos	1 – must	1 – must
Términos condicionales	0	0
Verbos voz pasiva	0	0
Conceptos del dominio	**4 - interface module, connection, service module, data base module**	**1 – connection**
Verbos del dominio	1 – handle	1 – handle
Reconocido por patrón	0	0

Con la representación de los requisitos mediante los valores de las métricas de calidad, se pueden generar instancias de evaluación que serán procesadas por el clasificador Clas-C4.5.

Durante el procesamiento de las instancias de evaluación, una regla del clasificador es disparada en base a los valores de las métricas:

- Para el requisito 1, se dispara la siguiente regla:

```
Términos de diseño > 0
|     Términos incompletos <= 0
|     |     Verbos del domino <= 1
|     |     |     Conceptos del dominio > 3: Buena calidad (17/1)
```

- Para el requisito 2:

```
Términos de diseño > 0
|     Términos incompletos <= 0
|     |     Verbos del domino <= 1
|     |     |     Conceptos del dominio <= 3
|     |     |     |     Caracteres puntuación > 127: Mala calidad (4)
```

Se puede observar, que bajo las métricas asignadas a los requisitos, el clasificador determina la calidad en función del número de conceptos del domino. En este ejemplo, parece que el experto que utilizó la metodología priorizó la inclusión de los requisitos al dominio en la clasificación de los requisitos del corpus, no dando tanta relevancia a otras características como el número de expresiones ambiguas.

Así, de acuerdo con la metodología propuesta, la estimación de la calidad que otorgaría el experto que proporcionó el corpus de requisitos sería: buena para el primer requisito y mala para el segundo.

Este mismo ejemplo será usado en la sección [3.2.5 Optimizar la calidad de los requisitos con el menor esfuerzo mediante sugerencias de corrección automáticas] ampliando las conclusiones sobre la calidad de estos requisitos.

3.2.2.1.4 Experimentación del enfoque 1: modelo simple

Para realizar la evaluación de los clasificadores de los distintos algoritmos se han utilizado conjuntos de test que no forman parte del conjunto de aprendizaje con los que entrenan los algoritmos. Se han generado varios conjuntos de test porque se ha realizado validación cruzada para evaluar el grado de acierto de los clasificadores.

Cada conjunto de test contiene con una selección estratificada de 101 requisitos de los cuales 54 tienen buena calidad y 47 mala. Por tanto, cada conjunto de aprendizaje contiene 934 requisitos, donde 491 requisitos tienen buena calidad y 443 mala.

Los resultados del porcentaje de acierto de los clasificadores se han obtenido por la función de evaluación que implementa la herramienta Weka (Hall et al. 2009).

Aunque la herramienta WEKA podría realizar la validación cruzada directamente en el experimento 1 (modelo simple), se ha decidido utilizar los mismos grupos empleados que en el experimento 2 y hacer manualmente la validación cruzada con el fin obtener una comparativa adecuada de los resultados.

En esta sección se explican los resultados que se han obtenido en el experimento planteado en la implementación del primer proceso que compone la metodología propuesta en la tesis. Existen dos factores clave para evaluar los distintos algoritmos: el porcentaje de acierto de los clasificadores sobre el conjunto de test y el tiempo empleado en la generación de los clasificadores.

La evaluación de estos dos factores se realiza sobre cada uno de los algoritmos de aprendizaje utilizados: PART, C4.5 y las combinaciones de clasificadores homogéneos mediante las técnicas Bagging y Boosting, que toman como base los algoritmos anteriores.

3.2.2.1.5 Resultados del experimento del enfoque 1: modelo simple

Los resultados del porcentaje de acierto de los clasificadores generados con los algoritmos de aprendizaje automático en los distintos grupos de validación cruzada se muestran en la Tabla. 2. Esta tabla contiene los resultados sobre 10 conjuntos de test de los clasificadores generados por los algoritmos de aprendizaje automático en cada uno de los grupos que forman la estratificación para la validación cruzada.

Tabla. 2 Resultados del porcentaje de acierto de los clasificadores con el modelo simple

Conjunto de test	PART	C4.5	Bagging PART	Bagging C4.5	Boosting PART	Boosting C4.5
Conjunto 1	78,22	78,22	80,2	81,19	81,19	87,15
Conjunto 2	83,17	85,13	87,13	90,1	87,13	86,14
Conjunto 3	86,14	83,17	92,08	86,14	85,15	88,12
Conjunto 4	90,1	89,11	91,09	90,1	92,08	88,12
Conjunto 5	86,14	77,23	86,14	85,65	87,13	82,18
Conjunto 6	83,17	86,14	85,15	86,14	85,15	86,14
Conjunto 7	82,18	79,21	79,21	80,2	83,17	80,2
Conjunto 8	88,12	84,16	90,1	87,13	90,1	85,15
Conjunto 9	84,26	83,17	88,12	82,18	88,12	86,14
Conjunto 10	86,14	86,14	89,11	84,16	89,11	90,1
Total (media aritmética)	**84,76**	**83,17**	**86,83**	**85,3**	**86,83**	**85,94**
Mediana	**85,2**	**83,66**	**87,62**	**85,89**	**87,13**	**86,14**
Desviación estándar	**3,34**	**3,85**	**4,33**	**3,41**	**3,27**	**2,91**

Se observa en la tabla que los mejores porcentajes de acierto son los correspondientes a los algoritmos Bagging PART y Boosting PART con un 86,83% de acierto y con una mediana del 87%. El algoritmo con los resultados más próximos a la media es Boosting-C 4.5 con una desviación típica de 2,91, lo que significa que es el algoritmo más estable.

En la Tabla. 3, se muestra la eficiencia de estos algoritmos en el proceso de aprendizaje. Los valores de la tabla fueron calculados realizando la media aritmética

del tiempo en segundos empleados por los algoritmos en la generación de los clasificadores.

Tabla. 3 Tiempo medio en segundos utilizado en la generación de los clasificadores por los distintos algoritmos con el modelo simple

	PART	C4.5	Bagging PART	Bagging C4.5	Boosting PART	Boosting C4.5
Tiempo en segundos	1.56	1.44	5.26	5.10	4.99	4.82

Se puede destacar que los algoritmos PART y C4.5 requieren menos tiempo que la combinación de clasificadores homogéneos Boosting y Bagging, en la generación de los clasificadores debido a que estos algoritmos generan un conjunto de clasificadores como resultado de su proceso de aprendizaje.

3.2.2.2 Enfoque 2: Modelo por comparación de pares

En esta sección se explica la implementación del segundo enfoque de la primera metodología propuesta en la tesis. Este experimento ha sido denominado "*por comparación de pares"* ya que las instancias de aprendizaje son la comparación de dos requisitos. Como ya se ha comentado, este enfoque implementa algunos métodos de forma similar al primer enfoque (modelo simple), por lo que no se volverán a explicar con detalle.

3.2.2.2.1 Tarea 1 – Generación de modelos de evaluación

Método: Crear corpus de requisitos clasificados

El corpus de requisitos es el mismo que en el enfoque 1, así como la calidad asignada por el experto. Es necesario tener el mismo corpus inicial para poder hacer una comparación coherente de los resultados de los experimentos. Se recuerda que el corpus contiene 1035 requisitos, donde 545 tienen buena calidad y 490 mala.

Método: Extracción de métricas de calidad

La representación de cada requisito por los conjunto de métricas de calidad es también igual en los dos enfoques y se usa la misma herramienta Requirements Quality Analyzer (RQA) (Company 2015b).

Método: Creación de las instancias de aprendizaje

En la implementación de este método, los dos enfoques difieren respecto al formato de las instancias de aprendizaje. La implementación de este método para este enfoque es una adaptación a la ingeniería de requisitos de la metodología propuesta en la patente presentada en (Moreno, Morato, and Sánchez-Cuadrado 2009), donde se presenta "un procedimiento sistematizado que puede predecir la posición que ocuparía un recurso, entre otros ordenados que se consideran competencia, si su características fuesen modificadas.". Haciendo uso de esta patente, en la tesis doctoral (Moreno 2010) se desarrolla una metodología para la estimación de la relevancia documental asignada por los buscadores en el entorno WEB. En el trabajo que se referencia, se utilizan algoritmos de aprendizaje automático sobre instancias de aprendizaje formadas por pares de comparación de valores de características de

posicionamiento de páginas WEB en buscadores de internet. En el trabajo de esta tesis, los valores que forman las instancias son las métricas de calidad de los requisitos. Cada instancia es una concatenación de dos conjuntos de métricas pertenecientes a dos requisitos distintos, más un valor de clase que indica cuál de los dos conjuntos de métricas tiene mejor calidad. En la figura (Fig. 15) se muestra una representación gráfica de una instancia.

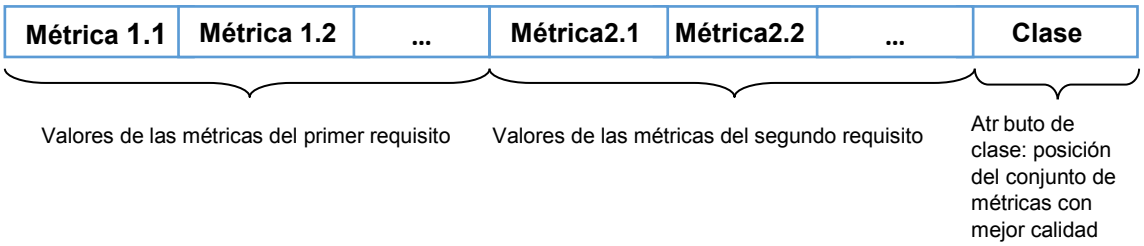

Fig. 15 Formato de una instancia en el modelo por comparación de pares

El valor del atributo de clase que se ha establecido en la implementación es: 1 si el primer requisito tiene mejor calidad asignada por el experto que el segundo o 2 si el segundo requisito tiene mejor calidad que el primero.

Por cada instancia de pares de comparación se realizan permutaciones en el orden de los requisitos y se cambia el valor de clase. Por ejemplo, si la calidad de un requisito A es mejor que la calidad de un requisito B, las instancias tendrán la permutación mostrada en las figuras (Fig. 16) y (Fig. 17).

Métricas del requisito A	Métricas del requisito B	Valor de clase: 1

Fig. 16 Instancias sin permutar

Métricas del requisito B	Métricas del requisito A	Valor de clase: 2

Fig. 17 Instancias permutadas

Realizando esta permutación los algoritmos pueden establecer claramente cuándo un conjunto de métricas de un requisito es mejor que otro independientemente de la posición que ocupa dentro de la instancia. El número de instancias de aprendizaje con pares de comparación son todas las posibles parejas de requisitos de distinta calidad multiplicadas por dos debido a las permutaciones.

Método: Generación de clasificadores

Se utilizan los mismos algoritmos de aprendizaje automático para generar los clasificadores que en el modelo simple. Recordemos que los algoritmos son PART y C4.5, así como combinaciones de clasificadores homogéneos usando las técnicas Bagging y Boosting, que toman como base los anteriores. Este método se realiza también usando la herramienta Weka (Hall et al. 2009).

En este experimento el algoritmo aprende qué conjunto de métricas hacen que un requisito tenga mejor calidad en comparación con otro. Por este motivo hace falta una interpretación de los resultados para poder determinar la calidad de un nuevo requisito.

3.2.2.2.2 Tarea 2 – Estimación de la calidad de nuevos requisitos

Método: Extracción de métricas de calidad

La implementación del primer método de esta etapa es común en los dos enfoques propuestos. Se trata de representar los nuevos requisitos con las mismas métricas de calidad que se usaron en el conjunto de aprendizaje.

Método: Generar instancias con las métricas de cada nuevo requisito

Como se ha comentado, los clasificadores generados usando instancias de aprendizaje por comparación de pares, no estiman la calidad de los requisitos directamente, ya que los algoritmos son entrenados para determinar si un requisito tiene mejor calidad en comparación con otro. Es necesario crear un procedimiento que permita estimar la calidad de un requisito nuevo mediante la comparación de requisitos.

El procedimiento que se ha elegido es comparar cada nuevo requisito con los requisitos utilizados en el conjunto de aprendizaje en la generación del clasificador. Por tanto, la forma de cada instancia es la comparación entre el nuevo requisito y un requisito del conjunto de aprendizaje. Al igual que en el entrenamiento, por cada instancia se realiza una permutación, por lo que el número de instancias necesarias para estimar la calidad de un nuevo requisito es el número de requisitos del conjunto de aprendizaje multiplicado por dos debido a las permutaciones. A este conjunto de instancias se denominará instancias de evaluación.

El resultado al procesar el conjunto de instancias de evaluación por el clasificador es que, por cada instancia se estima cuál de los dos requisitos, el nuevo del que se desea saber la calidad o el requisito de aprendizaje, tiene mejor calidad. En (Fig. 18) se muestra gráficamente la composición del conjunto de instancias de evaluación.

Métricas nuevo requisito	Métricas del requisito A	Valor de clase: Por determinar
Métricas del requisito A	Métricas nuevo requisito	Valor de clase: Por determinar
Métricas nuevo requisito	Métricas del requisito B	Valor de clase: Por determinar
Métricas del requisito B	Métricas nuevo requisito	Valor de clase: Por determinar
. . .	. . .	. . .
Métricas nuevo requisito	Métricas del requisito Z	Valor de clase: Por determinar
Métricas del requisito Z	Métricas nuevo requisito	Valor de clase: Por determinar

Fig. 18 Conjunto de instancias de evaluación

Los requisitos caracterizados con letras del abecedario representan requisitos del conjunto de aprendizaje. Obsérvese que se realizan permutaciones de todas las instancias.

Método: Estimar la calidad de los nuevos requisitos

Una vez introducido el conjunto de instancias de evaluación en el clasificador se obtiene como resultado una valoración por cada una de las instancias indicando cuál de los requisitos tiene mejor calidad: el nuevo requisito o el requisito con el que es comparado. En este trabajo se ha creado un conjunto de reglas que utiliza esta información para establecer la calidad de nuevos requisitos.

El conjunto de reglas utiliza como variables el número de requisitos con buena calidad del conjunto de aprendizaje que han sido valorados con mejor calidad que el nuevo requisito y viceversa, el número de requisitos con mala calidad que han sido valorados con peor calidad que el nuevo requisito. Se utilizan estas variables porque los algoritmos han sido entrenados con la comparación de requisitos con distinta calidad, por lo que el grado de acierto será mayor.

A continuación, se presenta el conjunto de reglas que se han establecido para determinar la calidad de un nuevo requisito.

```
Si Porcentaje_Buenos_Requisitos = 100% y Porcentaje_Malos_Requisitos = 100%
    Calidad_Estimada = Buena_Calidad
Sino
    Si Porcentaje_Buenos_Requisitos = 100%
```

```
        Calidad_Estimada = Mala_Calidad
    Sino
        Si Porcentaje_Malos_Requisitos = 100%
            Calidad_Estimada = Buena_Calidad
        Sino
            Si Porcentaje_Buenos_Requisitos >= 99%
                Calidad_Estimada = Mala_Calidad
            Sino
                Si Porcentaje_Malos_Requisitos >= 99%
                    Calidad_Estimada = Buena_Calidad
                Sino
                    Si Porcentaje_Buenos_Requisitos > 91.65% y Porcentaje_Malos_Requisitos > 90.97%
                        Calidad_Estimada = Mala_Calidad
                    Sino
                        Si Porcentaje_Malos_Requisitos > 96.61%
                            Calidad_Estimada = Buena_Calidad
                        Sino
                            Si Porcentaje_Buenos_Requisitos > 88.93%
                                Calidad_Estimada = Mala_Calidad
                            Sino
                                Si Porcentaje_Malos_Requisitos > 49.89%
                                    Calidad_Estimada = Buena_Calidad
                                Sino
                                    Calidad_Estimada = Mala_Calidad
```

Fig. 19 Conjunto de reglas para determinar la calidad de un nuevo requisito

Donde:

- *Porcentaje_Buenos_Requisitos*: representa la cantidad de requisitos, con buena calidad del conjunto de aprendizaje, valorados por el clasificador con mejor calidad que el nuevo requisito.
- *Porcentaje_Malos_Requisitos*: representa la cantidad de requisitos, con mala calidad del conjunto de aprendizaje, valorados por el clasificador con peor calidad que el nuevo requisito.
- *Calidad_Estimada*: calidad asignada al nuevo requisito.

Para definir estas reglas se ha utilizado los clasificadores con los mismos requisitos del conjunto de aprendizaje con los que han sido entrenados los algoritmos. Las instancias de aprendizaje están formadas por la comparación de cada requisito con todos los demás. Como resultado se obtiene una valoración de la calidad de cada requisito con respecto al resto. Agrupando esta información se puede obtener por

cada requisito del conjunto de entrenamiento, el porcentaje de requisitos de buena calidad que valoran al requisito tratado con peor calidad y viceversa, el porcentaje de requisitos de mala calidad que valoran al requisito tratado con mejor calidad. Una vez hecho esto, se puede generar un conjunto de instancias de aprendizaje usando por cada requisito de entrenamiento, los porcentajes arriba mencionados más la valoración de la calidad que asignó el experto. Las instancias de aprendizaje han sido procesadas por el algoritmo Bagging PART dando como resultado un conjunto de reglas que asocian la calidad estimada por el experto con los porcentajes de los requisitos del conjunto de entrenamiento que evalúan con mala o buena calidad a un requisito. Con este proceso se consigue utilizar la información del conjunto de aprendizaje para generar reglas que permitan estimar una evaluación de la calidad de nuevos requisitos.

3.2.2.2.3 Experimentación del enfoque 2: modelo por comparación de pares

Al igual que en la experimentación del primer enfoque, se han utilizado conjuntos de test que no pertenecen a conjunto de aprendizaje. Se recuerda que estos test es una selección estratificada de 101 requisitos de los cuales 54 tienen buena calidad y 47 mala. Por tanto, cada conjunto de aprendizaje contiene 934 requisitos, donde 491 requisitos tienen buena calidad y 443 mala.

Para realizar una comparativa adecuada de los enfoques se han obtenido los resultados mediante validación cruzada. Debido a que la estimación de la calidad de los requisitos del segundo enfoque (modelo por comparación de pares) no es inmediata como salida de los clasificadores, no se ha podido utilizar la validación cruzada proporcionada por la herramienta utilizada en la generación de los clasificadores, WEKA (Hall 2009). Ha sido necesario realizar los grupos de validación cruzada manualmente y obtener los resultados de acierto en los distintos grupos por la evaluación estándar de la herramienta. Para determinar el resultado final de cada algoritmo de aprendizaje se realiza la media aritmética de los resultados. La evaluación estándar muestra como resultados el porcentaje de acierto del clasificador, evaluado con un conjunto de test que se ha generado para realizar la validación cruzada

Como se ha dicho en el primer enfoque, la validación cruzada ha sido creada con 10 grupos estratificados del corpus de requisitos. Los algoritmos son entrenados con los conjuntos de aprendizaje formados por 9 de los 10 grupos y se utiliza el décimo como conjunto de test. Este proceso se realiza 10 veces seleccionando un conjunto de test distinto cada vez para que los algoritmos aprendan con todos los requisitos y sean evaluados también con todos ellos. El resultado final de la validación cruzada es la media aritmética de los porcentajes de acierto sobre los distintos conjuntos de test.

3.2.2.2.4 Resultados del experimento del enfoque 2: modelo por comparación de pares

En la Tabla. 4 se muestran los resultados obtenidos con el segundo enfoque por los distintos algoritmos sobre los grupos creados para hacer la validación cruzada.

Tabla. 4 Resultados del porcentaje de acierto de los clasificadores con el modelo por comparación de pares

Conjunto de test	PART	C4.5	Bagging PART	Bagging C4.5	Boosting PART	Boosting C4.5
Conjunto 1	80,2	81,19	85,15	84,16	79,21	81,19
Conjunto 2	90,1	88,12	88,12	87,13	92,08	85,15
Conjunto 3	85,15	84,16	88,12	86,14	86,14	85,15
Conjunto 4	89,11	100	89,11	88,12	90,1	93,07
Conjunto 5	84,16	88,12	86,14	87,13	86,14	86,14
Conjunto 6	81,19	85,15	86,14	85,15	86,14	86,14
Conjunto 7	82,18	79,21	83,17	78,22	85,15	78,22
Conjunto 8	83,17	88,12	90,1	88,12	89,11	90,1
Conjunto 9	82,18	83,17	84,16	88,12	85,15	82,18
Conjunto 10	84,16	100	90,1	85,15	86,14	87,13
Total (media aritmética)	**84,16**	**87,72**	**87,03**	**85,74**	**86,53**	**85,45**
Mediana	**83,66**	**86,63**	**87,13**	**86,63**	**86,14**	**85,64**
Desviación estándar	**3,23**	**7,13**	**2,45**	**2,1**	**3,47**	**4,28**

El mejor resultado se obtiene con el algoritmo C4.5 con un 87,72% de acierto. No obstante, el algoritmo C4.5 obtiene la desviación típica más alta 7,13, lo que significa que es el algoritmo más inestable. El algoritmo Bagging-PART consigue el segundo mejor porcentaje de acierto 87,03; el valor más alto con respecto a la mediana 87,13; y se puede considerar el algoritmo más estable al obtener el menor resultado en la desviación típica 2,45.

En la Tabla. 5, se muestra la eficiencia de los algoritmos en la generación de los clasificadores. Se presentan los tiempos en segundos, minutos y horas.

Tabla. 5 Tiempo utilizado en la generación de los clasificadores por los distintos algoritmos con el modelo por comparación de pares

	PART	C4.5	Bagging PART	Bagging C4.5	Boosting PART	Boosting C4.5
Tiempo en segundos	1676,7	275,7	13094,6	3120,6	21985,5	4756,5
Tiempo en minutos	27,95	4,6	218,24	52,01	366,43	79,28

Tiempo en horas	0,46	0,077	3,64	0,87	6,11	1,32

La tabla anterior muestra la media aritmética del tiempo que emplean los distintos algoritmos en la generación del clasificador sobre los grupos de la validación cruzada. Se observa que el algoritmo más eficiente es el C4.5 con algo menos de 5 minutos. Los algoritmos Bagging-PART y Boosting-PART necesitan varias horas para terminar el proceso.

Los algoritmos han sido ejecutados en un ordenador con un microprocesador Intel Core i3-3225 a 3.30 GHz y una capacidad RAM de 8GB.

En (Fig. 20), se presenta un gráfico comparativo de la eficiencia de los algoritmos.

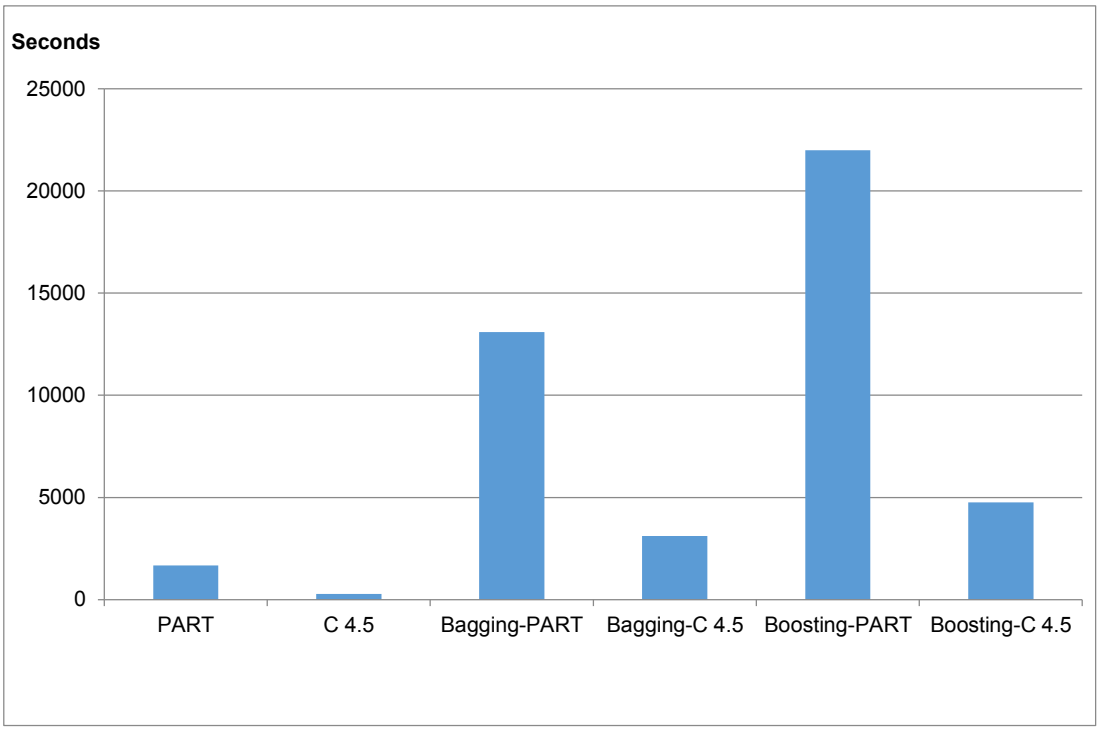

Fig. 20 Gráfico comparativo de la eficiencia en segundos de los algoritmos con el modelo por comparación de pares

3.2.3 Comparación de resultados de la experimentación del proceso 1

En esta sección se comparan los resultados obtenidos en los dos enfoques planteados en el primer proceso de la metodología.

En (Fig. 21) se muestra en un gráfico de barras los resultados. Para cada algoritmo se indican los porcentajes de acierto obtenidos en cada experimento para facilitar la comparación.

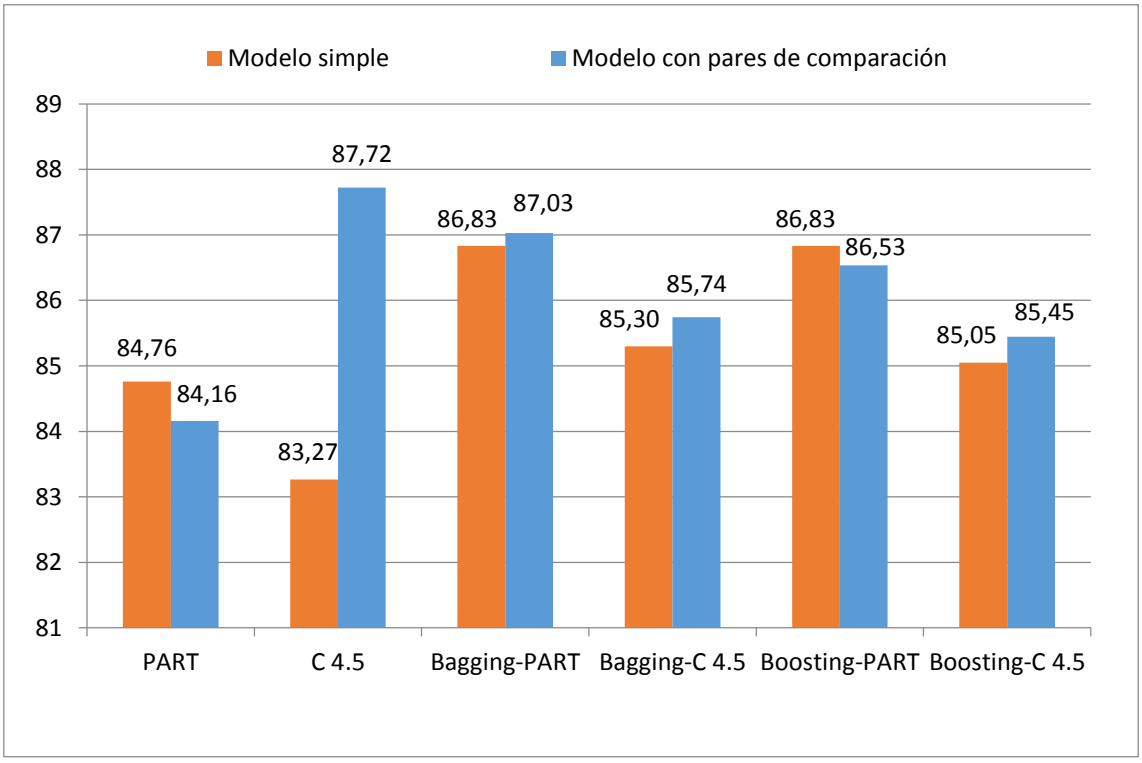

Fig. 21 Porcentaje de acierto de los clasificadores del enfoque uno y del enfoque dos

En la figura 22 (Fig. 22), se muestran los mismos resultados que en la figura 21 (Fig. 21) pero en un gráfico radial. Este tipo de grafico permite un mejor análisis para la comparación de los resultados de los dos enfoques.

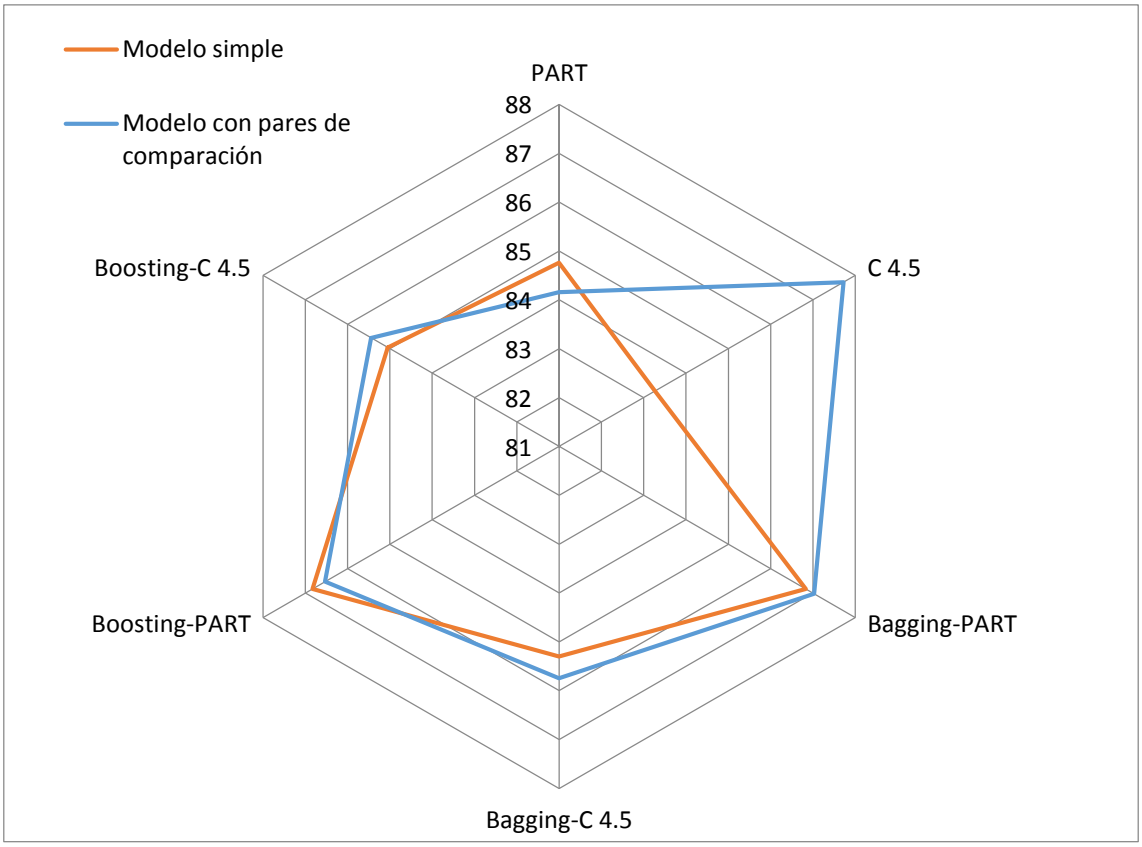

Fig. 22 Porcentaje de acierto de los clasificadores del enfoque uno y del enfoque dos

En (Fig. 23) se muestra en un gráfico de barras el tiempo empleado por los algoritmos de los dos enfoques en la generación de los clasificadores. El gráfico se presenta en perspectiva 3D para distinguir los valores obtenidos con el modelo simple.

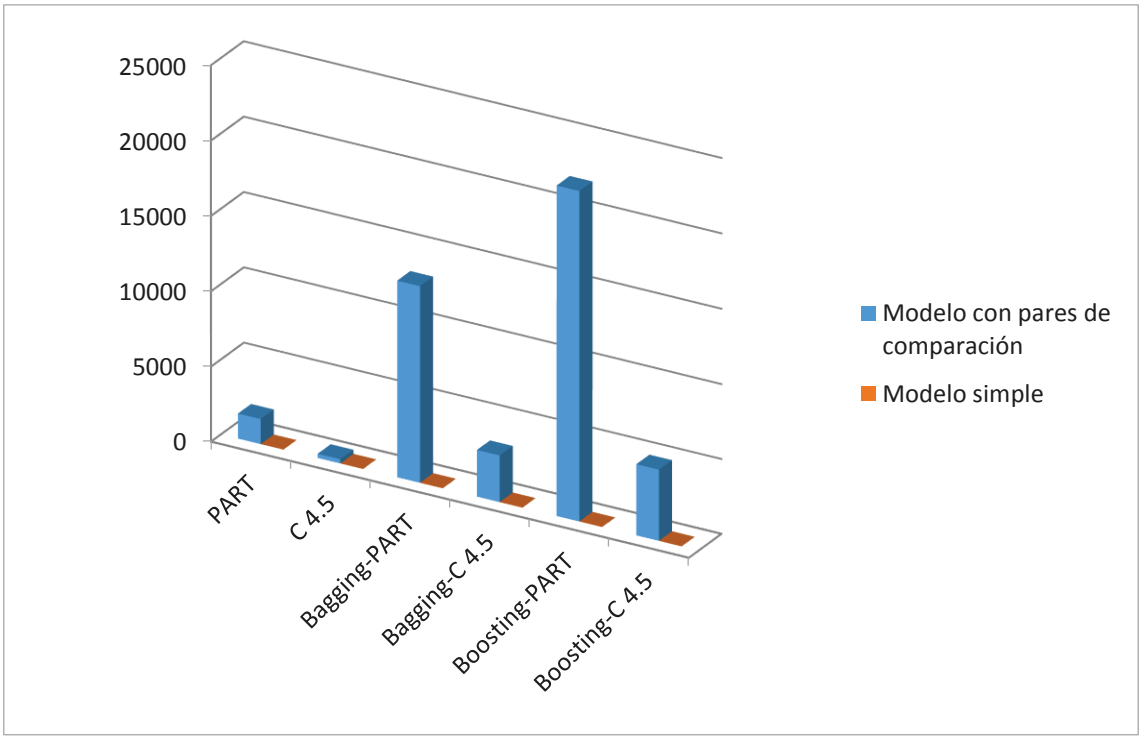

Fig. 23 Eficiencia en segundos de los algoritmos entre los dos enfoques

La Tabla. 6 y Tabla. 7 muestran la media aritmética, mediana y desviación estándar de los dos enfoques presentados.

Tabla. 6 Valores estadísticos de los resultados con el modelo simple

Modelo simple						
	PART	C4.5	Bagging PART	Bagging C4.5	Boosting PART	Boosting C4.5
Total (media aritmética)	84,76	83,17	86,83	85,3	86,83	85,94
Mediana	85,2	83,66	87,62	85,89	87,13	86,14
Desviación estándar	3,34	3,85	4,33	3,41	3,27	2,91

Tabla. 7 Valores estadísticos de los resultados con el modelo por comparación de pares

Modelo por comparación con pares						
	PART	C4.5	Bagging PART	Bagging C4.5	Boosting PART	Boosting C4.5
Total (media aritmética)	84,16	87,72	87,03	85,74	86,53	85,45
Mediana	83,66	86,63	87,13	86,63	86,14	85,64
Desviación estándar	3,23	7,13	2,45	2,1	3,47	4,28

Analizando los resultados, se observa que el modelo que consigue el mejor porcentaje de acierto es el modelo por comparación de pares, mejorando en cuatro de los seis algoritmos. Aunque el algoritmo C4.5 en este modelo obtiene el mejor porcentaje de acierto 87,72 también presenta el valor más alto de desviación típica 7,13. El algoritmo Bagging PART del segundo enfoque obtiene el menor valor de desviación típica 2,45, siendo el algoritmo más estable y tiene el segundo mejor porcentaje de acierto de todos los resultados 87,03.

No obstante, aunque el modelo por comparación de pares consigue el mayor porcentaje de acierto, no consigue una diferencia significativamente alta en la mayoría de los algoritmos, aun cuando el número de instancias es considerablemente más elevado que en el modelo simple (435026 instancias en el modelo por comparación de pares frente a 934 en el modelo simple). Esta observación se puede justificar teniendo en cuenta que las instancias del modelo por comparación de pares están formadas por comparación de requisitos de diferente calidad, y en el proceso de evaluación los nuevos requisitos son comparados con todos los requisitos que forman el conjunto de entrenamiento, provocando que se comparen requisitos con la misma calidad. Esta situación puede provocar estimaciones erróneas que provoquen un descenso en el porcentaje de acierto final de los experimentos.

Una vez obtenidos los resultados del porcentaje de acierto y la eficiencia, se pueden sacar conclusiones sobre la elección de las distintas implementaciones del proceso de la metodología en función de las preferencias al abordar los problemas:

- El modelo por comparación de pares es recomendable para aquellos sistemas en los que se pueda concretar un conjunto suficiente de requisitos, categorizados en función de su calidad y siendo este conjunto suficiente para poder determinar qué características han de cumplir los requisitos para establecer la calidad. También debido a que el número de instancias de aprendizaje aumenta considerablemente al realizar los pares de comparación y las permutaciones, este método es adecuado en corpus donde el número de requisitos de distinta calidad no esté balanceado.

- El modelo simple es adecuado para aquellos sistemas donde el conjunto de requisitos proporcionado ha de ser modificado para establecer continuamente nuevas características de calidad. En estos casos, el modelo simple es la mejor elección debido al poco tiempo que necesitan los algoritmos para crear nuevos clasificadores que contemplen las nuevas características.

  Este enfoque es adecuado en aquellos corpus donde el número de requisitos de distinta calidad esta balanceado.

3.2.4 Funciones de calidad en el hiperespacio de métricas

El proceso que se ha planteado tiene como objetivo encontrar una función que permita emular la calidad que otorgaría a un requisito un experto del dominio. Para ello se han utilizado técnicas de inteligencia artificial sobre métricas de conjuntos de requisitos clasificados. Distintas funciones de calidad son utilizadas en las herramientas de gestión de requisitos para valorar la calidad. A continuación, se va a demostrar porque las funciones utilizadas en la mayoría de las herramientas no acotan con suficiente precisión las métricas de los requisitos para valorar su calidad, y así argumentar el motivo de utilizar técnicas de aprendizaje automático.

La forma en la que las herramientas de gestión de calidad de requisitos realizan la valoración de la calidad es empleando combinación lineal de métricas sobre intervalos de bondad en las métricas utilizadas. Estos intervalos son definidos por expertos en calidad de requisitos del proyecto. Mediante este conocimiento es posible determinar la calidad de los requisitos, y se puede comprobar cuáles de las métricas han sido evaluadas con calidad baja para poder corregir los defectos con el fin de conseguir elevar la calidad de los requisitos de la especificación. No obstante, esta solución no es suficiente para acotar adecuadamente las regiones de calidad en el hiperespacio de métricas.

Supongamos que clasificamos requisitos en función de su calidad con una variable simple o métrica, por ejemplo, el Número de Términos del Dominio (NTD) usados en el

requisito. Es posible formular una regla simple para adecuar la métrica dentro de un nivel de calidad (Génova et al. 2013), tal que:

- "Si NTD = 0 entonces Calidad Mala, si no Calidad Buena".

La regla puede ser fácilmente redefinida para que tenga en cuenta varios niveles de calidad, utilizando más intervalos en el valor de la métrica, tal como:

- "Si NTD <= 0 entonces Mala, si no si NTD > 4 entonces Calidad Media, si no Calidad Buena". (Fig. 24)

Esto se puede generalizar a cualquier número de niveles de calidad. Para simplificar el ejemplo se asumirá que el número de niveles es dos:

- "Si NTD <= 0 entonces Mala, si no si NTD > 4 entonces Calidad Mala, si no Calidad Buena".

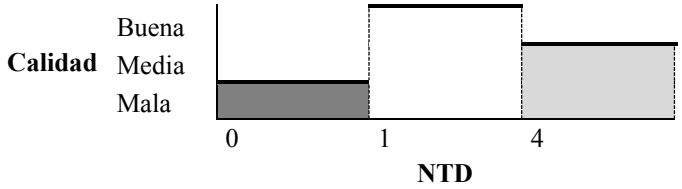

Fig. 24 Regla simple para acotar el número de términos del dominio (NTD) dentro de los niveles de calidad

Ahora, supongamos que queremos combinar dos métricas distintas para asignar un nivel de calidad, por ejemplo, Número de Términos del Dominio (NTD) y Número de Palabras (NP). Se pueden representar ambas variables en un plano X-Y, donde cada punto es un requisito y el color es el grado de calidad. La forma más simple de discriminar la calidad dentro de la nube de puntos es mediante una combinación lineal de variables, es decir, el tradicional método de ponderación media de las métricas:

- "Si (a* NTD + b*NP) <= L, entonces Calidad Mala, si no Calidad Buena". Donde a, b y L son valores convencionales.

Cuando la nube de puntos es naturalmente dividida en dos regiones de calidad separadas por una línea recta, los valores de a, b y L se pueden obtener fácilmente con un método matemático simple (Fig. 25). El método generaliza cualquier número de

dimensiones (métricas) que definen el hiperespacio de requisitos dividiéndolo en dos regiones por un hiperplano.

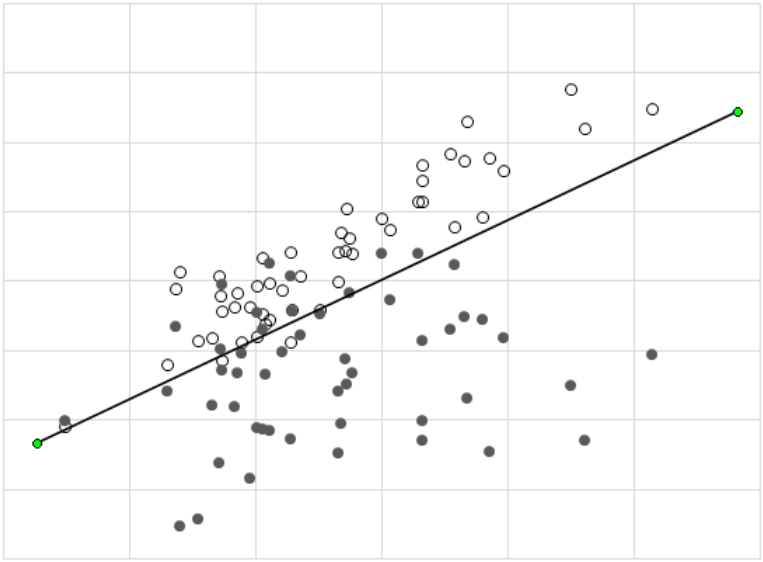

Fig. 25 Combinación de dos métricas arbitrarias usando una línea recta para discriminar la calidad en caso simple (blanco: Calidad Buena, negro: Calidad Mala)

Sin embargo, este caso no es frecuente si las variables presentan relaciones más complejas de calidad, ya que pueden ser muy difíciles de estimar a priori (Fig. 26). En estas situaciones, resulta más conveniente acotar el espacio por regiones rectangulares que utilizar el método simple de combinación lineal de métricas, donde la buena calidad está restringida al espacio dentro del rectángulo, y la mala calidad fuera del mismo. Aclarar que esta región puede ser abierta en uno o más de sus lados: "si NTD entre [1, 4] y NP entre [10, –] entonces Calidad Buena, si no Calidad Mala". Este método también es algo simple y generaliza las regiones a hiperrectángulos en el hiperespacio de requisitos

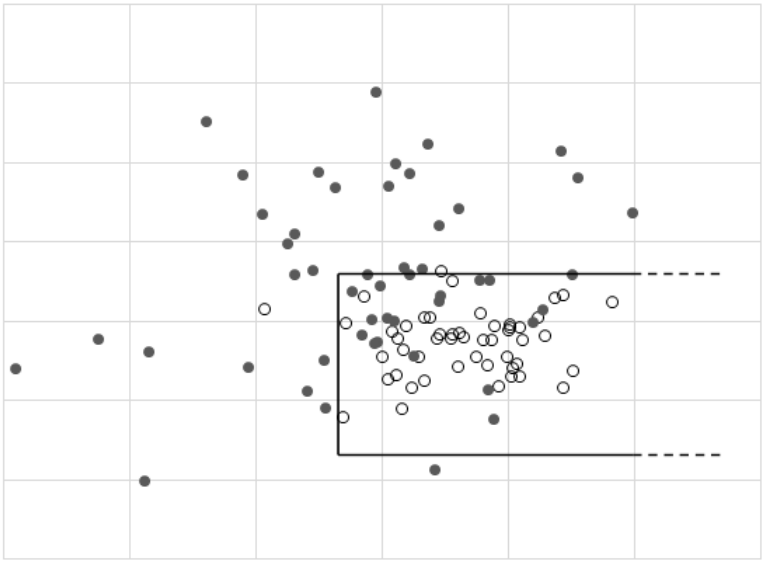

Fig. 26 Combinación de dos métricas usando un rectángulo simple abierto para acotar la calidad en el espacio entre Buena Calidad (blanco) y Mala Calidad (negro)

Sin embargo, el caso más usual en la combinación de métricas no es tan simple, y la nube de puntos no es fácilmente acotable por un (hiper-) rectángulo. En estas situaciones podemos también generalizar el procedimiento con una combinación de regiones que se ajustan mejor a la nube de puntos (Fig. 27), tal que:

- "Si NTD entre [1, 3] y NP entre [10, 30] entonces Calidad Buena, si no si NTD entre [4, 7] y NP [20, –] entonces Calidad Buena, si no Calidad Mala".

Aclarar que las regiones se pueden superponer (exigir que no se superpongan podría producir un peor ajuste de la regla)

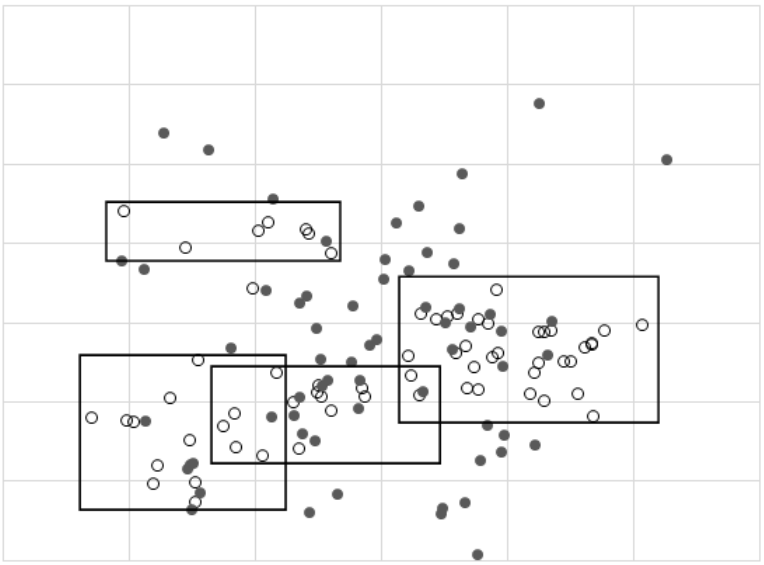

Fig. 27 Combinación de dos métricas usando una combinación de regiones rectangulares para discriminar entre buena calidad (blanco) y mala calidad (negro)

Resumiendo, la discriminación de regiones mediante hiperplanos es generalmente poco apropiada, y el uso de regiones rectangulares simples es también insuficiente. Por otro lado, una combinación de regiones rectangulares es un método versátil cuando las distintas métricas empleadas presentan una relación compleja en las agrupaciones de puntos. Un mayor número de regiones, mejora el ajuste de la discriminación de reglas en los conjuntos de datos (evitando, por supuesto, un excesivo número de regiones, las cuales comprometerían la generalidad). Sin embargo, computar los intervalos concretos por cada métrica resulta un problema complicado. Es por este motivo por lo que se han utilizado técnicas de aprendizaje automático ya que son particularmente útiles para proporcionar conjuntos de reglas.

3.2.5 Optimizar la calidad de los requisitos con el menor esfuerzo mediante sugerencias de corrección automáticas.

Similar al problema de encontrar una función que emule la clasificación de calidad de un experto de requisitos, podemos formular el problema de optimización para encontrar la modificación del requisito que necesite de menos esfuerzos para cambiar

la calidad de mala a buena. En términos de Search-Based Software Engineering, la formulación matemática sería:

- Sea C un clasificador binario basado en reglas que decide la calidad de un requisito. La decisión es tomada entre un conjunto de atributos numéricos de requisitos N, previamente procesados por una herramienta de análisis de calidad, en nuestro caso RQA.
- Sea R un requisito que ha sido clasificado con C con mala calidad.
- Sea M_i, $i \in \{1..N\}$ los valores numéricos obtenidos de R, en base a que C ha clasificado R con mala calidad. R puede ser representado también como un vector (M_1, M_2, …, M_N).
- Sea P_j, $j \in \{1..L\}$ los antecedentes de las reglas positivas en C, es decir, aquellas que clasifican el requisito como Bueno. Entonces, $\forall j \in \{1..L\}$, $\neg P_j(R)$, equivalente a $\neg P_j(M_1, M_2, \ldots, M_N)$. En otras palabras, todos los antecedentes de reglas positivas, cuando aplicadas al vector de métricas de un requisito malo, lo clasifican con mala calidad.

Ahora se quiere reescribir R en R', de modo que R' es evaluado como Bueno; es decir, R' satisface el antecedente de alguna regla positiva: $\exists j \in \{1..L\}, Pj(R')$, o equivalente $\exists j \in \{1..L\}, Pj(M'1, M'2, \ldots, M'N)$.

Además, se quiere minimizar el esfuerzo de obtener R'. Supongamos que existe un coste de corrección al modificar un valor de una métrica, y este coste depende de la magnitud de la modificación. Sea $Fi \in \Re^+ \cup \{0\}$, $i \in \{1..N\}$, el coste de modificación del valor Mi; y sea $Ki \in \Re^+ \cup \{0\}$, $i \in \{1..N\}$, el coste estimado de incrementar o disminuir una unidad al valor M_i.

Entonces el coste de modificar R para que M_i se convierta en $M'i$ es $Fi + Ki \cdot |Mi - M'i|$.

Queremos minimizar la función $\Sigma ni=1\ Fi + Ki \cdot |Mi - M'i|$.

Objetivo:

Encontrar una modificación R' de R tal que:

$$\exists j \in \{1..L\},\ P_j\ (M'_1, M'_2, \ldots, M'_N)$$

Y con un coste mínimo de modificación:

$$\sum_{1 \le i \le n \wedge M_i \neq M'_i}^{n} F_i + K_i \cdot | M_i - M'_i |$$

Hay que tener en cuenta que en la formula anterior se han añadido costes fijos Fi sólo cuando la modificación de los requisitos afecta a las i-ésimas métricas, es decir, cuando Mi ≠ M'i. La formulación se podría también generalizar si se considera que el coste de incrementar el valor de una métrica dada es distinto a los costes de disminuirla (por ejemplo, añadir un término del domino podría ser más costoso que eliminarlo). Se ha implementado un algoritmo mediante valores heurísticos de los coeficientes Fi y Ki, que resuelve este problema de optimización para el caso de reglas generadas con el algoritmo C 4.5.

Ejemplo:

Sea el siguiente requisito que ha sido clasificado como Malo por un clasificador generado con el algoritmo C 4.5.

Typically, *the interface component* ***must*** *be able to* ***handle*** *at least one* ***fast*** *connection with the service component* ***and*** *one connection with the predefined* ***data base*** *component.*

Este requisito contiene las siguientes métricas no nulas:

- Nº párrafos: 1
- Nº palabras: 27
- Legibilidad: 16
- Nº de caracteres entre signos de puntuación: 171
- Conectores: 1 (And)
- Términos ambiguos: 2 (typically, fast)
- Términos de diseño: 1 (data base)
- Términos imperativos: 1 (must)
- Conceptos del dominio. 1 (connection)
- Verbos del dominio: 1 (handle)

Se ha encontrado que la modificación que implica menor esfuerzo es satisfacer la siguiente regla extraída de árbol de decisión:

```
Términos de diseño > 0
|     Términos incompletos <= 0
|     |     Verbos del domino <= 1
|     |     |     Conceptos del dominio > 3: Buena calidad (17/1)
```

El requisito satisface todos los valore dela regla excepto el número de conceptos del domino, los cuales deben ser estrictamente mayores que 3 (esta regla sólo tiene sentido dentro del contexto de todas las reglas, es decir, como la definición de una hiper región dentro de hiper regiones en el hiperespacio de métricas). Por lo tanto, la regla puede ser satisfecha por el requisito con la modificación de una única métrica, simplemente añadiendo tres conceptos del dominio. Entonces se puede proponer la siguiente recomendación: “incrementar el número de conceptos del dominio a 3”. La modificación sobre el requisito para satisfacer la regla del requisito puede ser:

Typically, the interface module must be able to handle at least one fast connection with the service module and one connection with the predefined data base module.

Se ha reemplazado “interface component”, “service component” y “data base component” por sus correspondientes conceptos del dominio en la ontología: “interface module”, “service module” y “data base module”. Existe otra solución, cambiar la ontología en lugar del requisito añadiendo los conceptos modificados. Una vez se ha modificado el requisito, se deben extraer nuevas métricas y comprobar que su calidad es aceptable, como ocurre en el ejemplo.

Se observa en este ejemplo que no es necesario eliminar los términos ambiguos (“typically” y “fast”) para satisfacer la regla. Esto se puede interpretar de la siguiente manera: el clasificador basado en reglas ha aprendido desde el experto que la ausencia

de conceptos del dominio es un defecto grave, mientras que la presencia de términos ambiguos es tolerable; por tanto, la versión original de este requisito es más fácil de corregir añadiendo conceptos del dominio que eliminando términos ambiguos (algo que no sería suficiente para cambiar la calidad el requisito). Por supuesto, para otro experto podría ser más importante corregir los defectos de ambigüedad y estaría reflejado por otro clasificador generado por aprendizaje automático. La cuestión es, que la intención no es evaluar la calidad 'absoluta', si no emular con flexibilidad el juicio particular que posee el experto en un contexto dado.

En resumen:

Los pasos a realizar para modificar un requisito y cambiar su calidad con el menor esfuerzo, utilizando el clasificador como conjunto de sugerencias son:

- Comprobar por cada una de las reglas cuántos cambios son necesarios en el requisito para conseguir buena calidad.
- Se elige la regla donde el coste de modificaciones sea menor.
- Con esta información se puede sugerir al usuario qué cambios concretos se han de realizar para conseguir que el requisito tenga buena calidad.

3.3 PROCESO PARA LA GENERACIÓN DE PATRONES SINTÁCTICO-SEMÁNTICOS

En la sección anterior se ha propuesto un proceso para emular a un experto en calidad de requisitos en el ámbito de la asignación de calidad basándonos en métricas estándar. El siguiente proceso que compone la metodología, se tiene como objetivo la emulación de un experto en el ámbito de la composición estructural y semántica de requisitos. Para llevar esto a cabo, se propone la construcción automática de patrones sintáctico-semánticos mediante el tratamiento de las estructuras sintácticos-semánticas de requisitos, con el fin de ayudar en la redacción correcta de nuevos requisitos.

El proceso de la metodología consta de una serie de tareas que propone el tratamiento de un corpus de requisitos de buena calidad proporcionado por un experto, y se obtiene como resultado un conjunto de patrones relacionados jerárquicamente que componen las estructuras de los patrones sintáctico-semánticos.

A continuación, se muestran un esquema gráfico de las tareas y métodos que componen este proceso.

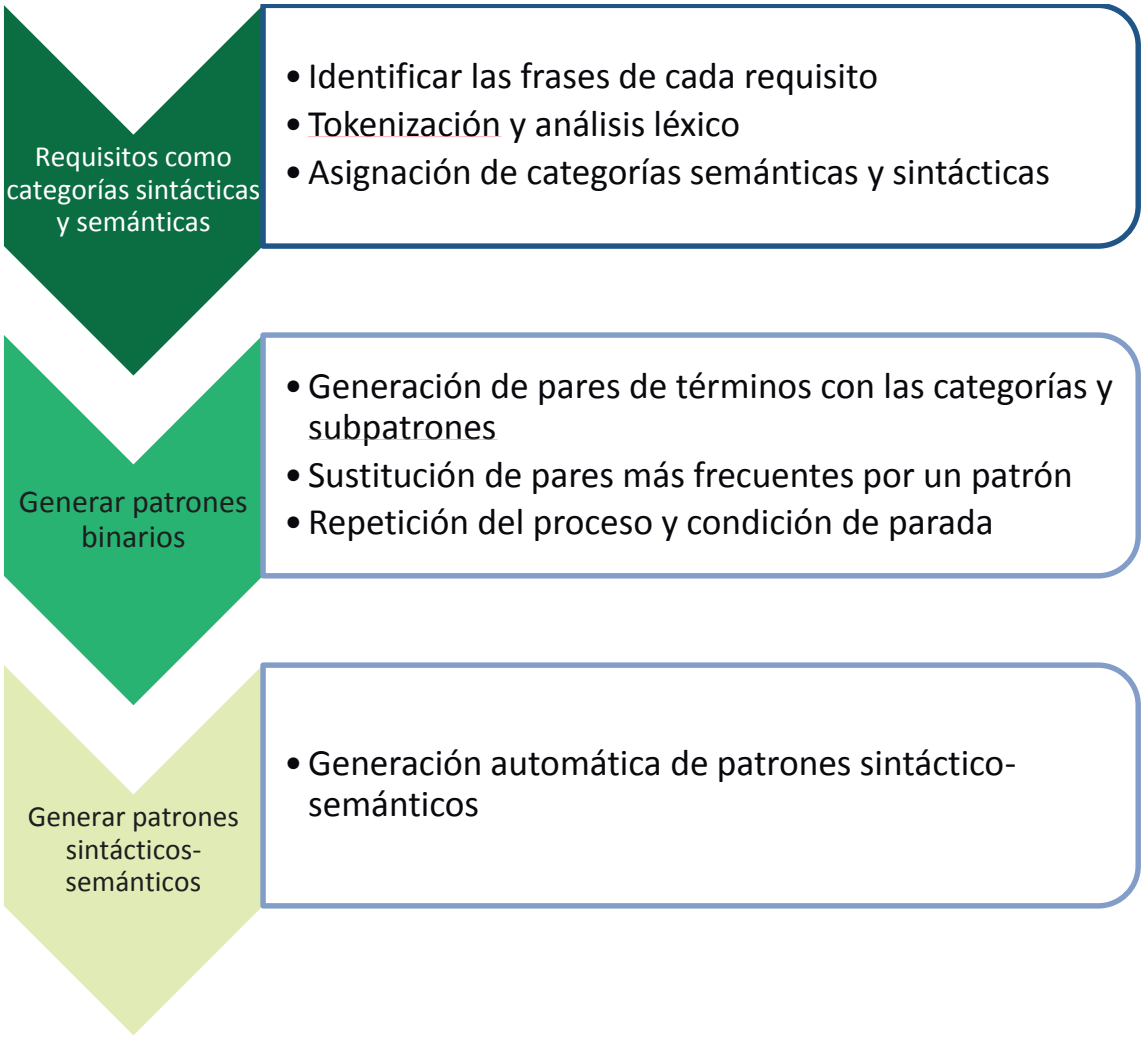

Fig. 28 Proceso de generación de patrones: tareas y métodos

3.3.1 Definición del proceso para la generación de patrones sintáctico-semánticos

En esta sección se detallan las tareas del proceso de la metodología. El proceso está compuesto por tres tareas: a) definir cada requisito como un conjunto de categorías gramaticales y semánticas, b) generar patrones binarios por la frecuencia de repetición de pares de categorías gramaticales en el conjunto de requisitos y c) generación automática de patrones sintáctico-semánticos mediante los patrones binarios. A continuación, se hace una breve descripción de cada tarea:

- Tarea 1 - Definir cada requisito mediante categorías sintácticas y semánticas: se procesa cada requisito para asignar a cada palabra una categoría sintáctica y una familia semántica cuando corresponda.
- Tarea 2 - Generar patrones binarios: se genera un conjunto de patrones binarios jerárquicos en función de la frecuencia de aparición de los pares de categorías gramaticales y semánticas obtenidos en la tarea anterior.
- Tarea 3 - Generar patrones sintáctico-semánticos: se crean automáticamente patrones sintáctico-semánticos de los requisitos con el conjunto de patrones binarios de la tarea anterior.

A continuación, se detallan los métodos de cada tarea del proceso de la metodología.

3.3.1.1 Métodos de la tarea 1 – Definir requisitos mediante categorías sintácticas y semánticas

La primera tarea del proceso de la metodología consiste en representar cada requisito del corpus con buena calidad proporcionado por el experto mediante un conjunto de estructuras que representan las categorías sintácticas de cada palabra del requisito, y una asociación a la familia semántica si fuera pertinente. La figura 29 (Fig. 29) ilustra los métodos de esta tarea del proceso.

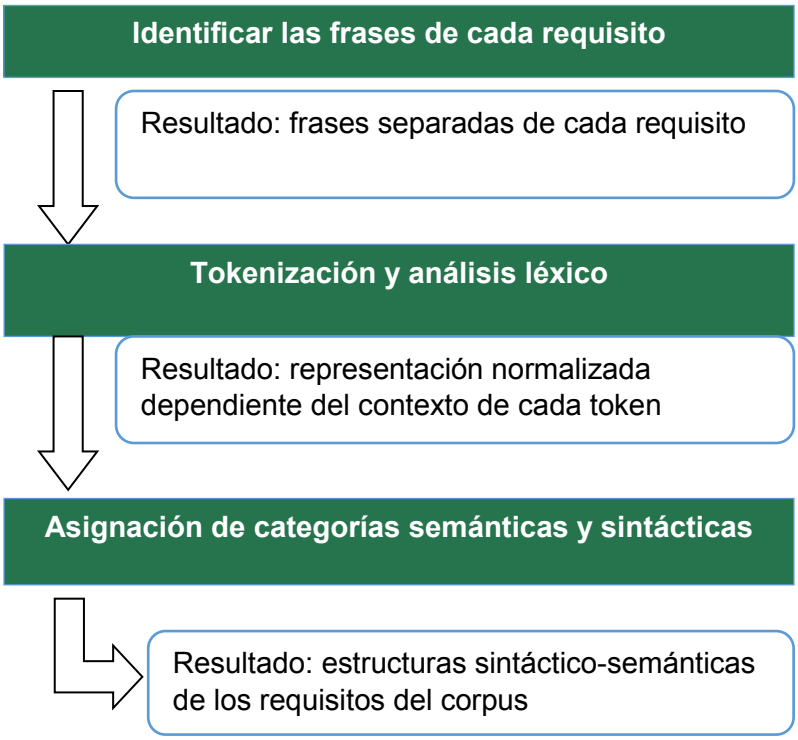

Fig. 29 Métodos de la tarea 1 – Requisitos como categorías sintácticas y semánticas

- ***Identificar las frases de cada requisito*:** un requisito de buena calidad puede estar formado por varias frases. El primer método de esta tarea propone el procesamiento de los requisitos del corpus para diferenciar las distintas frases que los componen, ya que la categoría sintáctica y semántica de cada palabra vendrá determinada por su relación con el resto de las palabras de la frase.

- ***Tokenización y análisis léxico:*** una vez separadas las frases de cada requisito, se realiza un proceso de tokenización para diferenciar los elementos de la frase. Después se efectúa un análisis léxico de cada token teniendo en cuenta su relación con el resto de tokens de la frase.

  El resultado de este método es una representación normalizada dependiente del contexto de cada token de cada una de las frases que componen el corpus de requisitos, obteniendo como resultado términos estandarizados para mejorar el análisis y simplificar la ambigüedad del vocabulario y las formas verbales.

- ***Asignación de categorías gramaticales y semánticas*:** usando la representación estandarizada de los términos de los requisitos, se pueden utilizar algoritmos y bases de datos de representación del conocimiento (tesauros, ontologías, etc.), para determinar sus categorías sintácticas y las categorías semánticas de cada elemento con dependencia del contexto, es decir, la representación de la categoría sintáctica y semántica de cada token en función del resto de las categorías de los otros elementos de la frase de componen cada uno de los requisitos. Se obtienen como resultado el conjunto de estructuras sintáctico-semánticas de los requisitos del corpus.

3.3.1.2 Métodos de la tarea 2 – Generar patrones binarios

Una vez completada la primera tarea de esta metodología se procede a la generación automática de patrones binarios. El objetivo es conseguir un conjunto de patrones jerárquicos generados automáticamente por la determinación de la frecuencia de aparición de pares de categorías sintácticas y semánticas.

En esta segunda tarea se realiza un proceso iterativo de tres métodos que termina cuando se alcanza un número mínimo de frecuencia de aparición determinado. La (Fig. 30) muestra gráficamente los métodos de esta tarea.

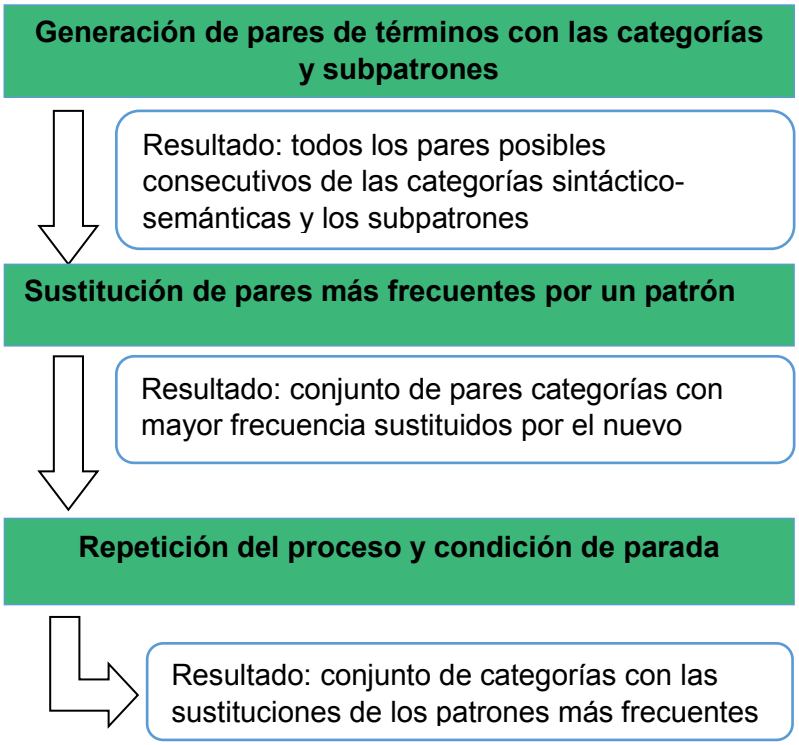

Fig. 30 Métodos de la tarea 2 – Generar patrones binarios

- ***Generación de pares de términos con las categorías y subpatrones*:** en la primera iteración de este método se realiza la extracción de todos los distintos pares consecutivos de las categorías sintácticas junto con sus semánticas, y se ordenan por frecuencia de aparición sobre todo el conjunto de categorías. El objetivo es recuperar los pares de categorías que más se repiten en todo el conjunto.

  En las siguientes iteraciones del método también se incluirán en la extracción de los pares por frecuencia subpatrones que sustituirán a los pares que más se repiten. Este proceso se explicará con más detalle en el tercer paso.

- ***Sustitución de pares más frecuentes por un patrón*:** con la pareja de categorías o subpatrones que más se repiten en todo el conjunto, se crea un patrón binario que sustituirá todas las ocurrencias de esta pareja que aparezcan en el conjunto. El patrón se añade al conjunto de patrones y en el caso de que alguna de las

categorías sintáctica tenga una semántica asociada se le asignará dicha semántica al patrón.

- ***Repetición del proceso y condición de parada*:** para desarrollar la segunda tarea de la metodología es necesario definir una condición de parada al ciclo iterativo. La condición es un número mínimo de frecuencia de aparición de la pareja de elementos. Cuando la pareja que más se repita en el conjunto tenga una frecuencia de aparición igual al valor de la condición de parada definida el proceso finalizará. En cada iteración de la tarea se sustituyen una pareja de elementos consecutivos por un patrón. Mientras no se alcance esta condición de parada, el proceso se repite incluyendo como elementos del conjunto de categorías los patrones binarios generados y por tanto, puede ocurrir que en la pareja que más se repite aparezca un patrón como uno de los elementos. Esta pareja será sustituida por un nuevo patrón, convirtiéndose en subpatrón el patrón que ejerce como término.

Cuando finaliza la tarea, el conjunto de categorías sintácticas y semánticas se ha visto modificado perdiendo elementos que han sido sustituidos por patrones; por tanto, ahora el conjunto está formado por categorías sintácticas, representación de la identificación de un patrón, y las semánticas asociadas a cada uno.

3.3.1.3 Métodos de la tarea 3: Generar patrones sintáctico-semánticos

Una vez alcanzada la condición de parada se obtiene, por un lado, el conjunto de patrones binarios generados, y por otro el conjunto de categorías sintácticas, identificadores de patrones y semánticas asociadas. El conjunto de patrones binarios tiene una estructura jerárquica debido a que los patrones pueden contener subpatrones y estos a su vez otros subpatrones.

Con estos dos conjuntos se pueden generar patrones sintáctico-semánticos de requisitos automáticamente ya que el conjunto de categorías sintácticas, patrones y semánticas representan las frases de cada uno de los requisitos; por tanto, cada requisito puede ser definido por un patrón sintáctico-semántico que estará formado

por los elementos del conjunto que representan las frases de dicho requisito. La figura 30 (Fig. 31) representa gráficamente el método de esta tarea del proceso de la metodología.

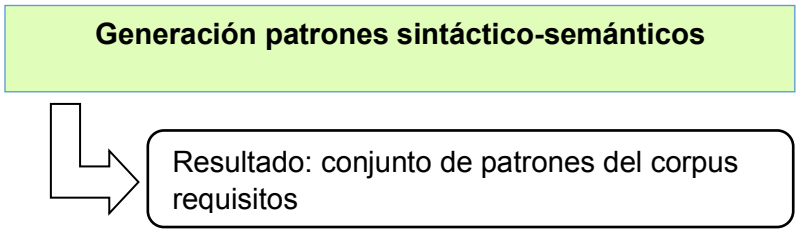

Fig. 31 Método de la tarea 3 – Generar patrones sintáctico-semánticos

En este proceso se han aumentado los niveles de significación generalizando las restricciones de las palabras de los requisitos escritos en lenguaje natural. Estas han sido sustituidas por categorías sintácticas, patrones y semánticas. Por tanto, es posible que un patrón pueda reconocer estructuralmente varios requisitos compuestos con las mismas categorías. Se pueden aumentar la generalización y flexibilidad de los patrones asignando a sus elementos las categorías: *opcional* y *comodín*. En la sección [*3.3.1.1 Herramienta desarrollada para la generación automática de patrones sintáctico-semánticos*] se explica cómo se han incorporado los conceptos de elementos opcionales y elementos comodines en la generación de los patrones sintáctico-semánticos.

3.3.2 Herramientas del proceso 2: generación de patrones sintáctico-semánticos

Se presenta a continuación, las herramientas que componen este proceso de la metodología.

Tokenización y análisis léxico de cada palabra de la frase

Los términos de las frases han sido procesados por funciones de normalización realizadas por el grupo de investigación Knowledge Reuse, del Departamento de Informática de la Universidad Carlos III de Madrid, destinadas al procesamiento del lenguaje natural. Las funciones utilizan bases del conocimiento de ontologías y tesauros de términos orientadas a la ingeniería de requisitos. Estas bases del conocimiento están almacenadas en bases de datos.

Asignación de categorías sintácticas y semánticas de los tokens

El procesamiento de los tokens de las frases de los requisitos se ha realizado con las funciones de procesamiento del lenguaje de la herramienta de gestión de requisitos Knowledge Manager (Company 2015a) partner de la Universidad Carlos III de Madrid. Esta herramienta contiene funciones de análisis sintáctico y semántico para el procesamiento de entidades normalizadas que asignan categorías gramaticales y semánticas a los tokens de los requisitos que previamente han sido normalizados.

Generación de patrones binarios y patrones sintáctico-semánticos

Para la generación de patrones binarios y *patrones sintáctico-semánticos* correspondientes a las tareas 2 y 3, se ha implementado una herramienta que permite el acceso a la base de datos con las categorías sintácticas y semánticas de los requisitos del corpus y desarrolla los métodos de la metodología. La herramienta ha sido desarrollada usando el lenguaje de programación .NET C#.

Almacenamiento y acceso

Los patrones binarios generados, las categorías sintácticas y semánticas de los requisitos y los patrones sintáctico-semánticos son almacenados en tablas de base de datos para permitir el uso de consultas SQL en la recuperación de la información. Estas consultas son programadas en la herramienta desarrollada para esta tesis y citada en el punto anterior.

3.3.2.1 Herramienta desarrollada para la generación automática de patrones sintáctico-semántico

Como resultado de la implementación del proceso de la metodología, se ha desarrollado una herramienta que genera automática de patrones sintáctico-semánticos mediante el tratamiento de un corpus de requisitos.

La herramienta recibe como entrada un corpus de requisitos mediante una base de datos o bien mediante documentos de texto. Tras un procesamiento de tokenización, análisis léxico, análisis sintáctico y semántico se asignan a todos tokens de los requisitos categorías sintácticas y semánticas, para después realizar un proceso

de generación automática de patrones binarios con estructura jerárquica que son utilizados en la generación automática de patrones sintáctico-semánticos.

A continuación, se explican cuáles son las funciones de flexibilidad que se han desarrollado para dar al usuario la posibilidad de mejorar la adaptación de los patrones sintáctico-semánticos al sistema de requisitos.

Funciones de flexibilidad en la estructura de los* patrones sintáctico-semánticos *desarrolladas en la herramienta

La flexibilidad que se puede dotar a los patrones se consigue mediante modificaciones de las restricciones semánticas asociadas a los patrones y por la asignación de las características opcional y comodín a los elementos de los patrones.

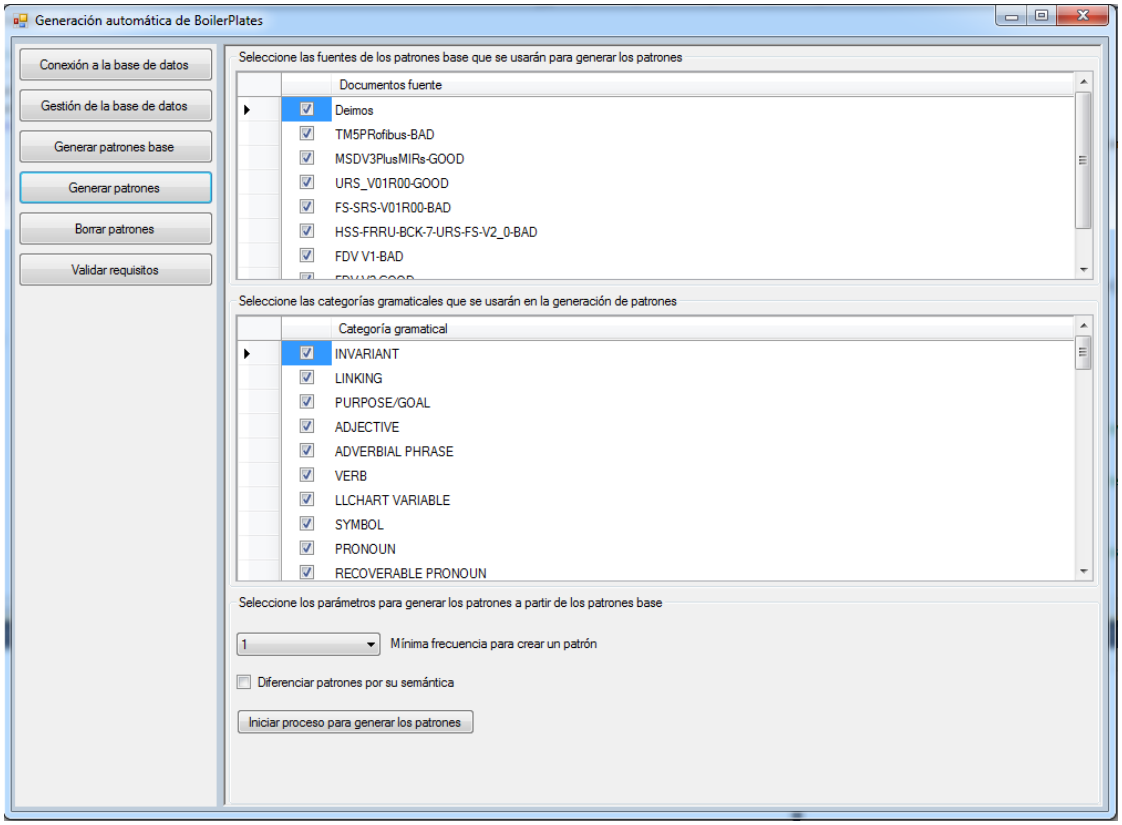

Fig. 32 Herramienta de generación de patrones implementada

- Un elemento que tenga la característica de *opcional*, representa un elemento que puede aparecer o no aparecer en el patrón. Es decir, en la redacción de un nuevo requisito, si se usa un patrón sintáctico-semántico donde alguno de los elementos

tiene la característica opcional asignada, se da la posibilidad de incluir un término que concuerde con la categoría gramatical y semántica del elemento opcional, o bien dejar el espacio vacío. El resto del requisito deberá cumplir con las restricciones de los elementos del patrón sintáctico-semántico.

- Un elemento que tenga la característica de *comodín*, indica que es un espacio libre en el que puede no aparecer ningún elemento o bien incluir cualquier número elementos de cualquier categoría. En la redacción de un requisito si se toma como plantilla un patrón que contiene algún elemento comodín, en esa posición podrá incluirse cualquier número de términos de cualquier categoría sintáctica y semántica. El resto del requisito tiene que cumplir con las restricciones del patrón.

- *Selección de categorías sintácticas:* En una de las interfaces de la herramienta se muestra un listado de todas las categorías sintácticas que están almacenadas en la base de datos. El número de categorías sintácticas a las que pueden pertenecer los token puede ser muy elevado, por lo que es necesario restringir las categorías sintácticas que van a formar parte del proceso. En la herramienta, antes de iniciar el proceso de categorización de los términos normalizados se pueden seleccionar aquellas categorías sintácticas que se consideren relevantes, para que se incluyan en la generación de los patrones. Durante el proceso de generación de los patrones binarios, cuando se encuentran términos con una categoría sintáctica que no ha sido incluida en las categorías relevantes, se convierte en elemento opcional.
- *Diferenciar patrones por su semántica:* Se ofrece esta opción para permitir que dos patrones sean distintos si tienen distintas semánticas asociadas, pero las mismas categorías gramaticales.

  Antes de la ejecución del proceso de la generación de patrones, se permite la posibilidad de elegir si un patrón binario puede tener asociada una o bien varias semánticas. Así, si a un patrón compuesto por las categorías sintácticas A y B sólo se permite tener una semántica asociada a cada categoría, se creará un patrón nuevo cada vez que se tenga que almacenar un patrón con las mismas categorías sintácticas A y B, pero con distintas semánticas asociadas. Por el contrario, si un

patrón puede tener varias semánticas asociadas a las categorías, y se da el caso de almacenar un patrón con las mismas categorías sintácticas, pero con distintas semánticas asociadas, no se almacenará como un nuevo patrón y se incluirán las nuevas semánticas a la lista de semánticas asociadas a ese patrón.

De esta forma, se controlan las restricciones semánticas en la generación de los patrones sintáctico-semánticos ya que, si un patrón tienen como elemento un patrón con varias semánticas asociadas, se podrán redactar varios requisitos donde los elementos sean semánticamente distintos y por tanto dar la flexibilidad en la redacción de requisitos que implica el lenguaje natural. No obstante, si los patrones sintáctico-semánticos contienen sub-patrones con sólo una semántica asociada a los términos, la redacción de los requisitos tendrá mayores restricciones en la redacción.

- *Eliminación de patrones:* Una vez generados todos los patrones binarios, se da la opción de eliminar aquellos que se consideren poco relevantes. La eliminación de un patrón afecta a aquellos patrones que lo contienen como subpatrón. En este proceso el subpatrón eliminado podrá ser sustituido por un elemento opcional o comiden en los patrones que los contengan como subpatrón.

3.3.3 Ejemplo de generación de patrones sintácticos-semánticos

A continuación, se describe un ejemplo de uso de esta metodología para la generación automática de patrones sintáctico-semánticos. En este ejemplo se procesan tres requisitos y se generan los patrones que los representan. Los requisitos son:

- *Requisito*1: The user must be able to save the plan
- *Requisito* 2: The system shall be to perform 10 tests per minute
- *Requisito* 3: The system must be able to search 5 files per second

Se recuerda, que en la primera tarea de este proceso se diferencian las frases de los requisitos, se normaliza cada token y se asignan las categorías sintácticas y semánticas.

A continuación, en la segunda tarea se generan, mediante un proceso iterativo de métodos, los patrones binarios que sustituyen a las categorías de los requisitos en función de la frecuencia de aparición. La condición de parada en el ejemplo será una frecuencia de parición mínima de 1.

En el ejemplo se van a mostrar dos posibilidades de configuración, diferenciando y sin diferenciar los patrones por su semántica. Si se diferencian los patrones por su semántica, los patrones con semánticas distintas, aun teniendo las mismas categorías sintácticas, se considerarán patrones diferentes.

La sucesión de imágenes que se presentan a continuación, muestran el desarrollo secuencial del proceso de la generación de patrones sintáctico-semánticos sobre los requisitos propuestos en el ejemplo.

La siguiente imagen muestra:

- Elementos utilizados en el ejemplo: categoría semántica (caja azul), categoría sintáctica (caja verde), patrón (caja naranja), el token (caja amarilla) y el mismo token normalizado (caja gris).
- Se muestra un ejemplo de la creación de un patrón con dos categorías sintácticas, teniendo la segunda una categoría semántica asociada. Se observa que el patrón hereda la semántica de la categoría.
- Los requisitos que serán tratados en el ejemplo.
- Características del proceso: condición mínima de parada es 1. Esta es la condición más restrictiva, ya que siempre que se repitan dos categorías consecutivas se sustituirán por un patrón. En este primer ejemplo no se diferencian los patrones por su semántica. También en este ejemplo se muestra la composición si patrones sí son diferenciados por su semántica.

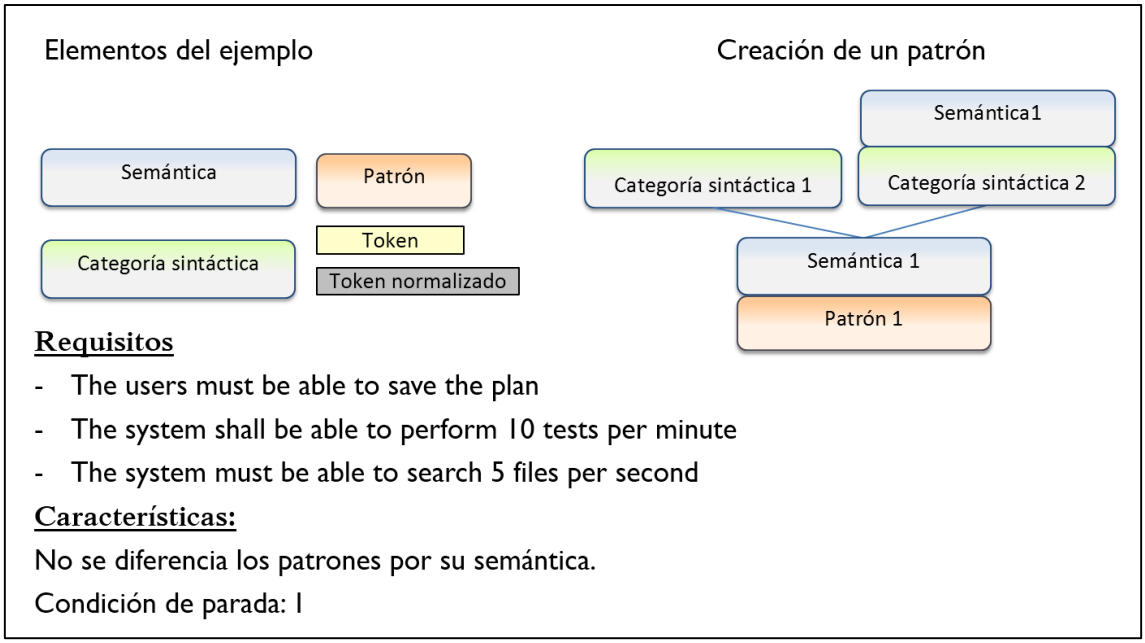

Fig. 33 Elementos del ejemplo de generación de patrones

En la siguiente imagen se muestra por cada palabra de los requisitos, su token, su token normalizado, la categoría sintáctica asignada y en algunos casos la categoría semántica. Son estas dos categorías las utilizadas en la generación de los patrones binarios.

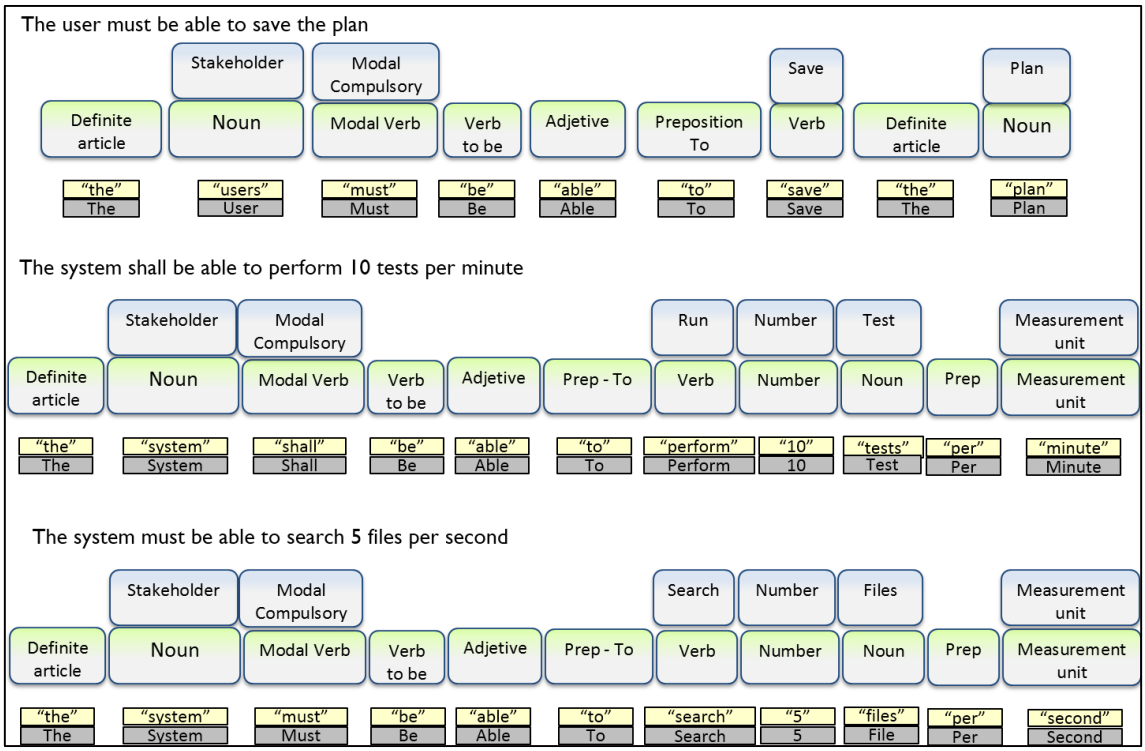

Fig. 34 Resultado de la primera tarea del proceso de generación de patrones

En la siguiente figura se muestra la sustitución de un patrón por las dos categorías sintácticas consecutivas que más se repiten. Es este caso estas categorías son: Artículo definido + Nombre. Los patrones heredan las categorías semánticas, *stakeholder* y *Plan*, si se elige la opción de diferenciar los patrones por su semántica, se deberían crear dos patrones distintos, uno por cada semántica.

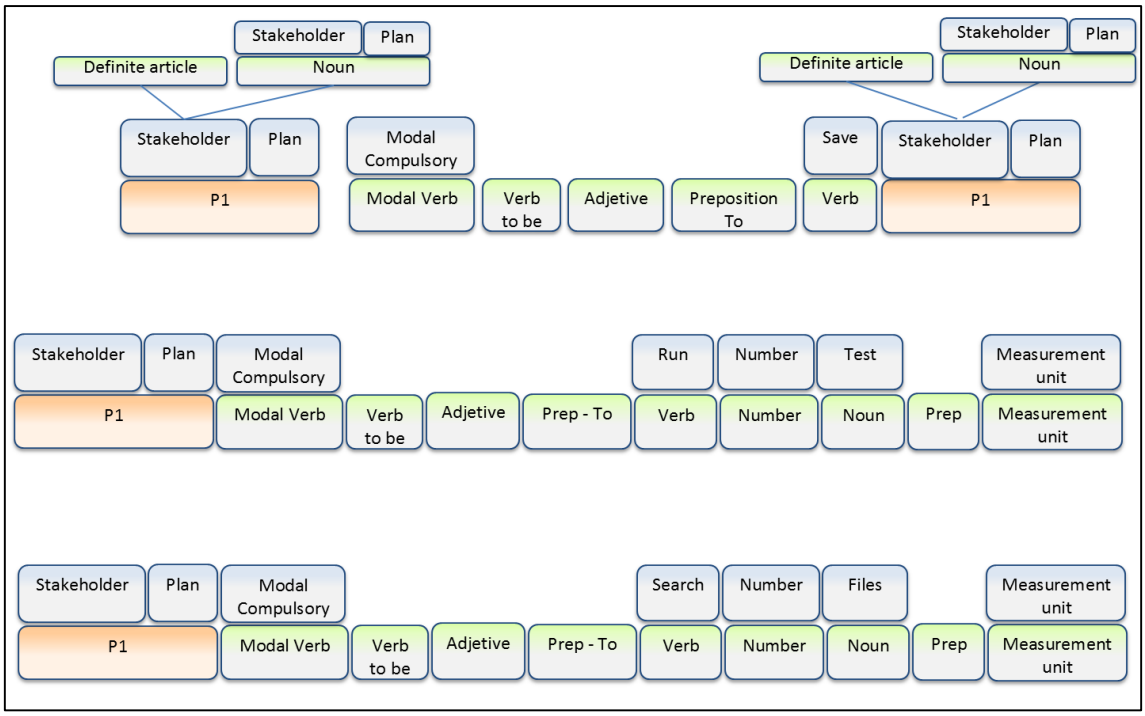

Fig. 35 Generación de patrones binarios - Iteración 1

Una vez sustituido el primer patrón, el proceso se repite ya que no se ha alcanzado la condición de parada. En el proceso ahora se incorpora el patrón *P1*, siendo este, junto con la categoría *Modal Verb,* los pares consecutivos que más se repiten. El nuevo patrón P2 hereda las semánticas del patrón y de la categoría sintáctica.

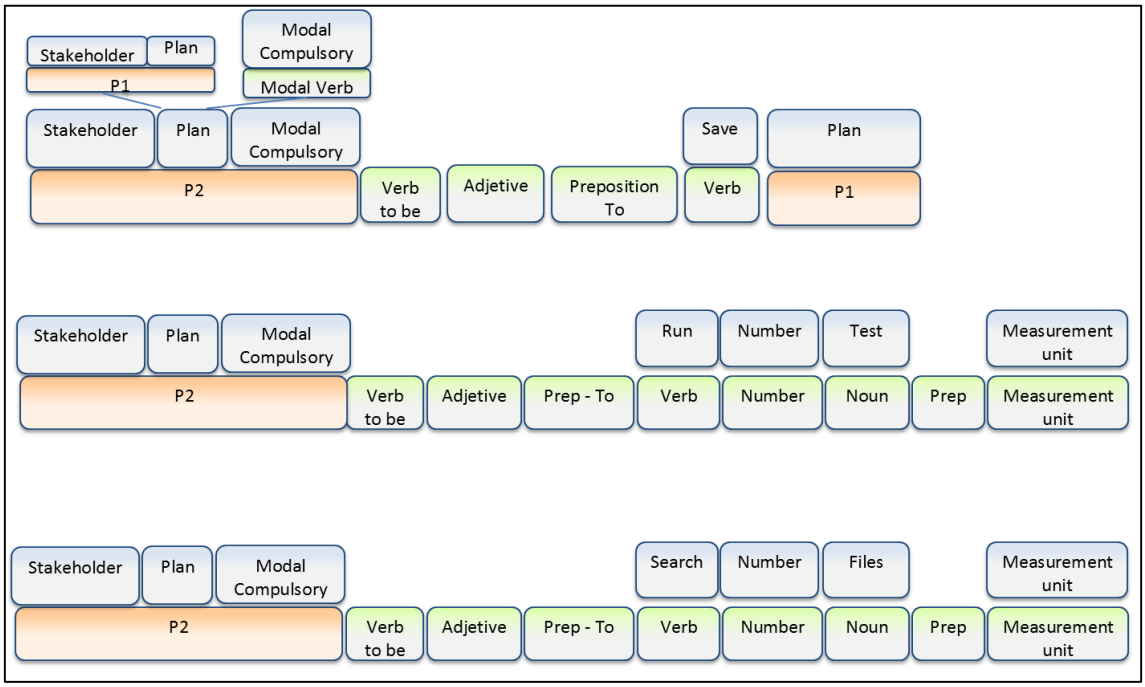

Fig. 36 Generación de patrones binarios - Iteración 2

El proceso continúa del mismo modo con la creación del patrón *P3: patrón P2 + Verb to be*

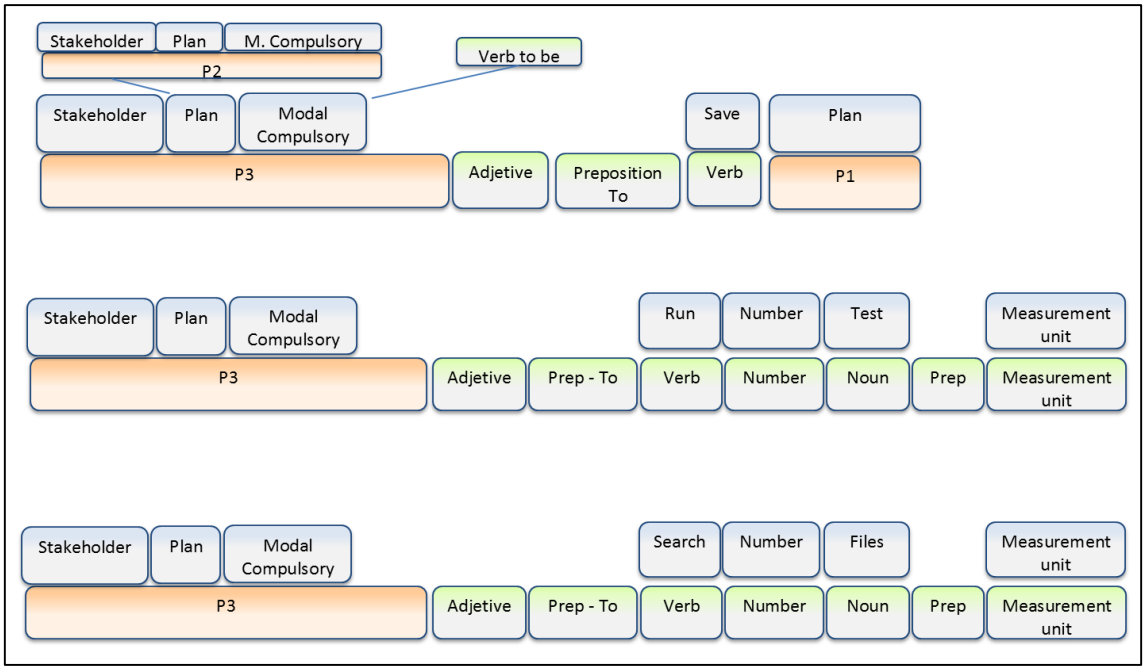

Fig. 37 Generación de patrones binarios - Iteración 3

Sustitución del patrón *P4: P3 + Adjetive*

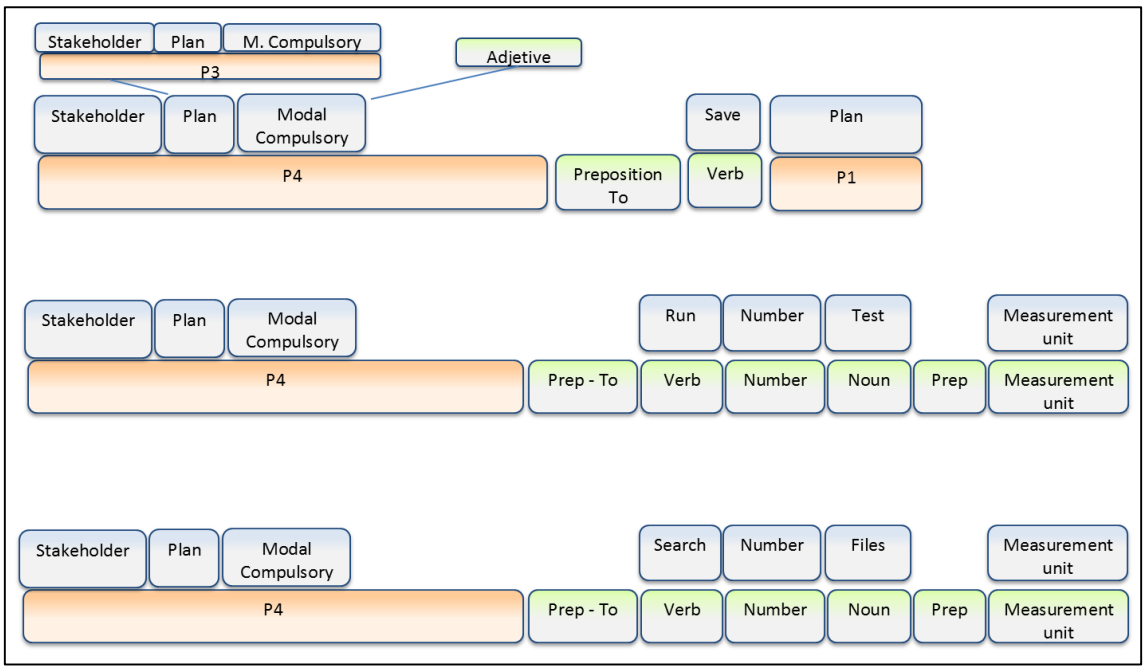

Fig. 38 Generación de patrones binarios - Iteración 4

Creación del patrón *P5: P4 + Preposition To*

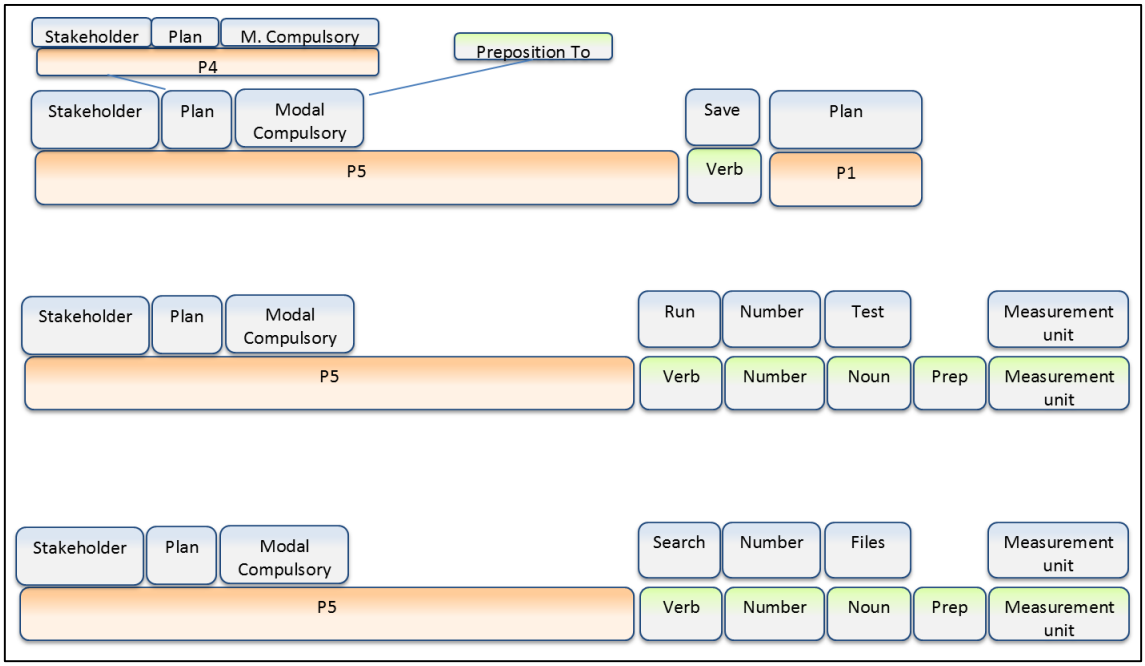

Fig. 39 Generación de patrones binarios - Iteración 5

A continuación, se crea el patrón P6. Este patrón está formado por *P5* y la categoría sintáctica *Verb*. El patrón hereda las semánticas del patrón P5 y las tres semánticas distintas de los verbos de los distintos requisitos. Cuando se vea ejemplo diferenciando los patrones por su semántica se observará que se tienen que crear tres patrones distintos, uno por cada semántica.

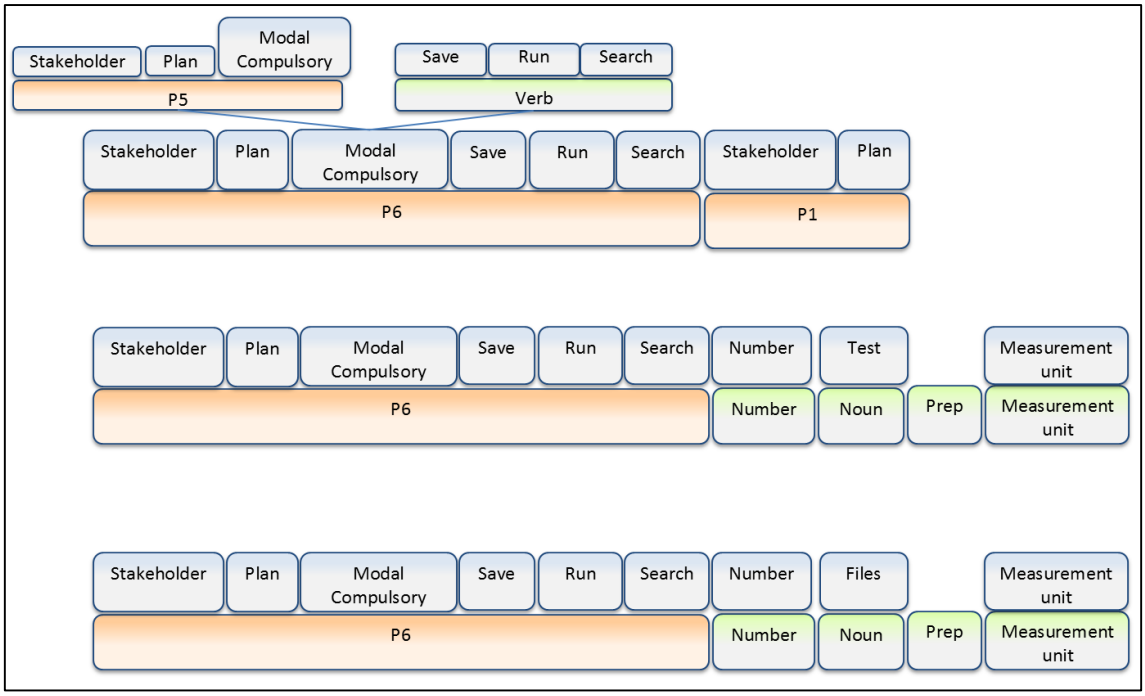

Fig. 40 Generación de patrones binarios - Iteración 6

Para el primer requisito ha terminado el proceso de generación de patrones, ya que no existe otro par consecutivo formado por el patrón *P6* y *P1*. Para los otros dos requisitos continua el proceso ya que se repiten dos veces el patrón *P6 + number*.

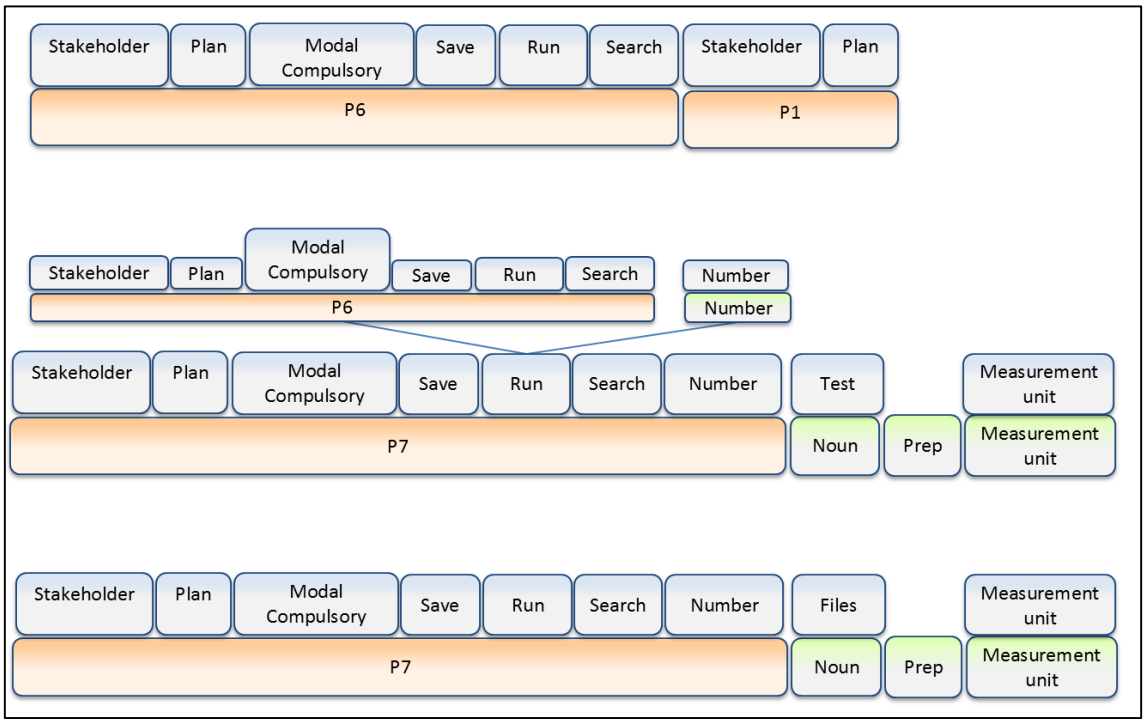

Fig. 41 Generación de patrones binarios - Iteración 7

Continua el proceso con la creación de *P8: P7 + Noun*

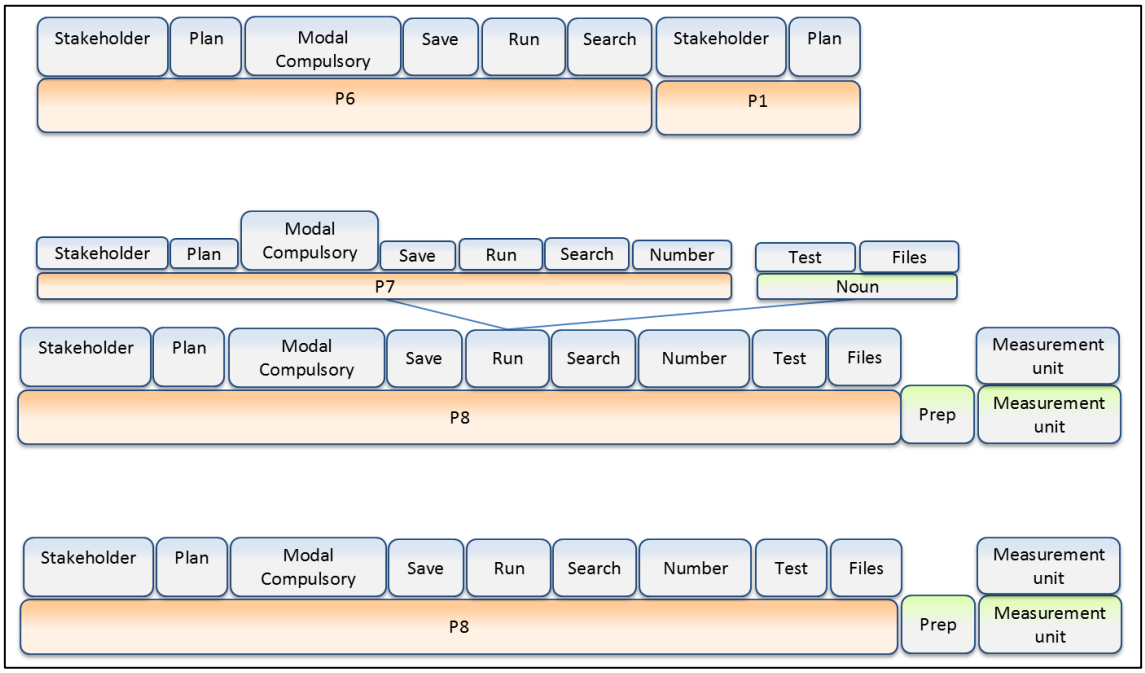

Fig. 42 Generación de patrones binarios - Iteración 8

Creación del patrón *P9: P8 + Preposition*

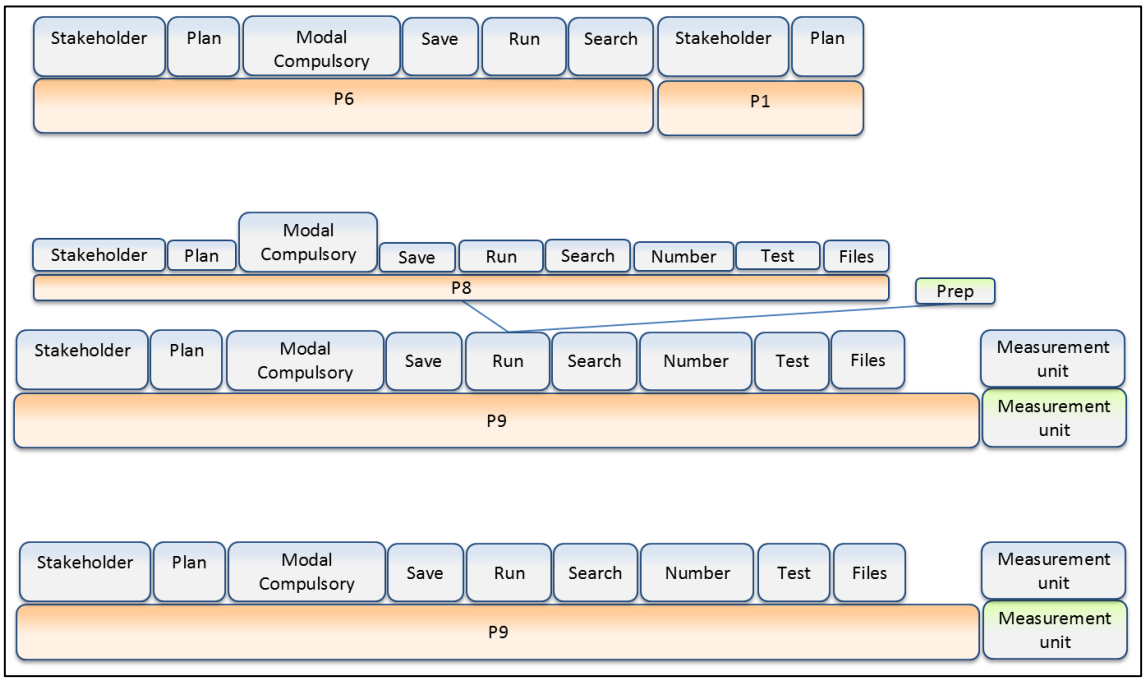

Fig. 43 Generación de patrones binarios - Iteración 9

La siguiente imagen muestra el último paso de creación de patrones, por haber alcanzado la condición de parada. Este último paso es la sustitución del patrón *P9* y la categoría *Measurement unit* por el patrón *P10*.

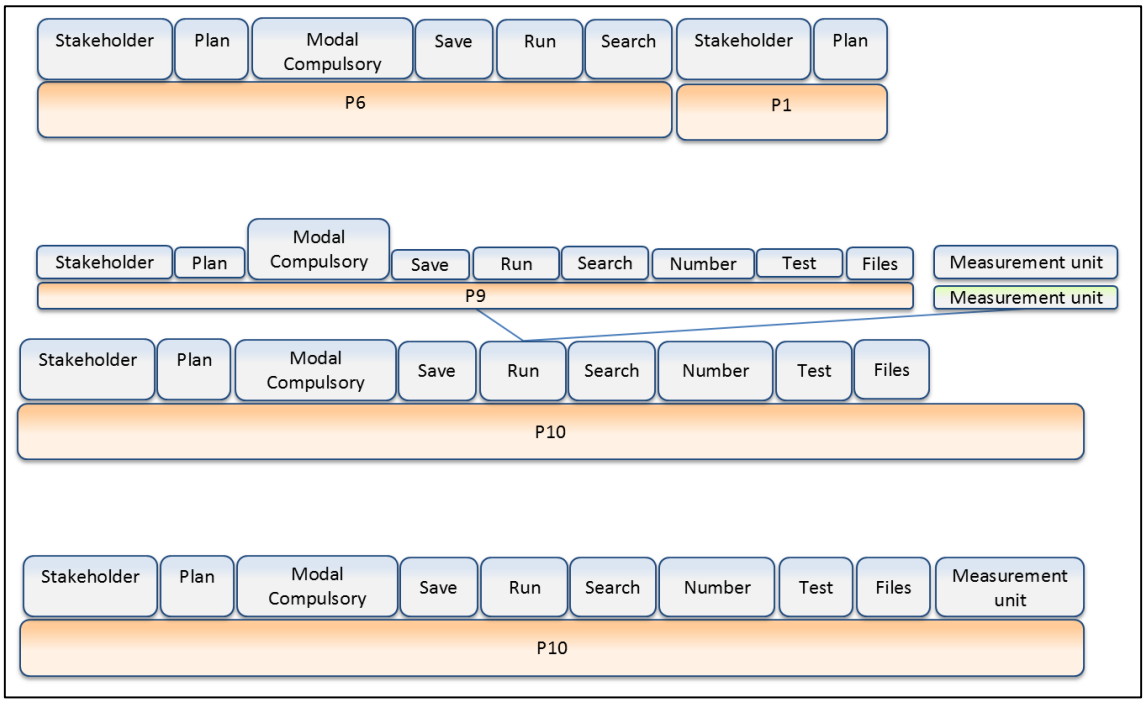

Fig. 44 Generación de patrones binarios - Iteración 10

La siguiente imagen muestra los patrones sintáctico-semánticos que han sido generados y que representan, mediante una composición jerárquica de patrones binarios, las estructuras sintáctico-semánticas de los tres requisitos del ejemplo. Se observa que el segundo representa dos de los tres requisitos.

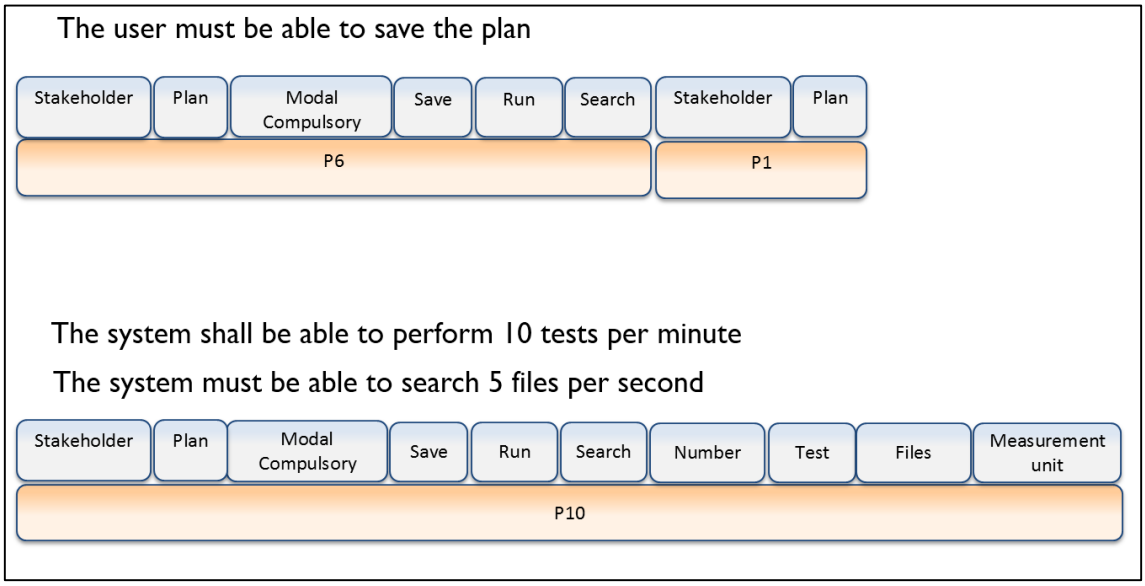

Fig. 45 Resultado: patrones sintáctico-semánticos

A continuación, se muestra la composición jerárquica del primer patrón.

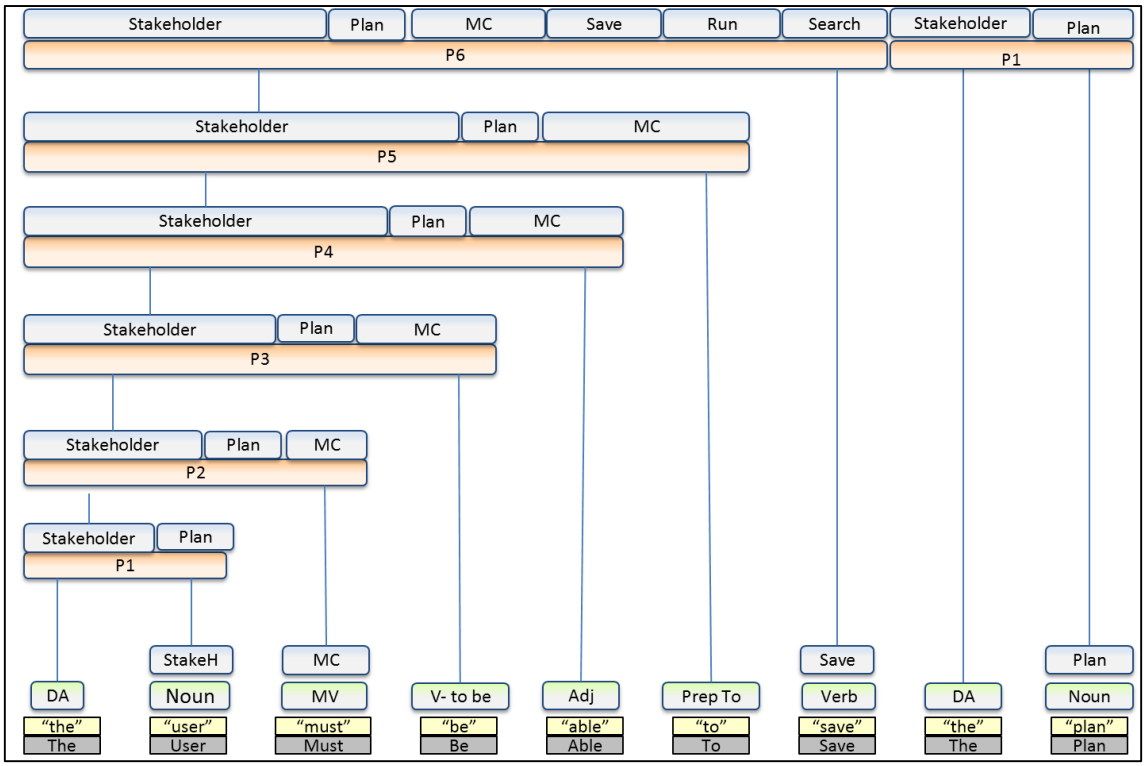

Fig. 46 Composición del patrón 1

La siguiente imagen muestra la composición del segundo patrón.

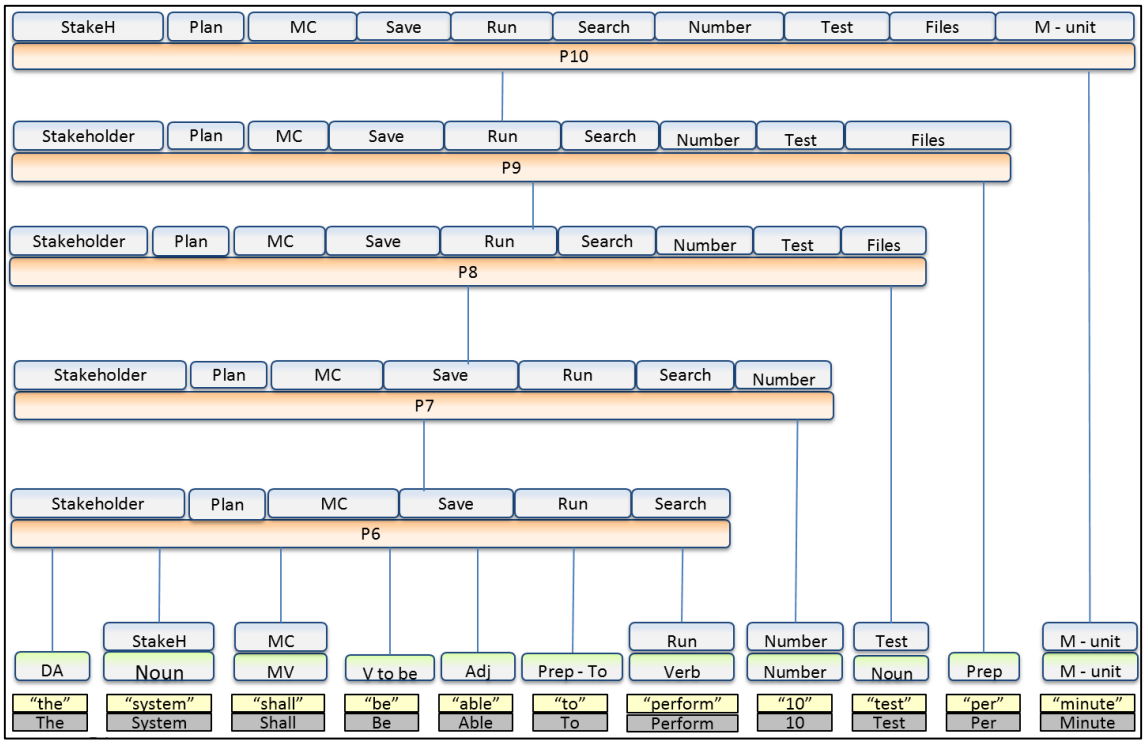

Fig. 47 Composición del patrón 2

A continuación, se muestra el proceso de generación de patrones pero diferenciando los patrones por su semántica.

Comenzamos a partir de la creación del patrón *P5*. Resaltar, que en el primer requisito no se han sustituido las dos últimas categorías *Definite Article y Noun* por el patrón P1, como ocurría en el ejemplo anterior, ya que tienen semánticas diferentes.

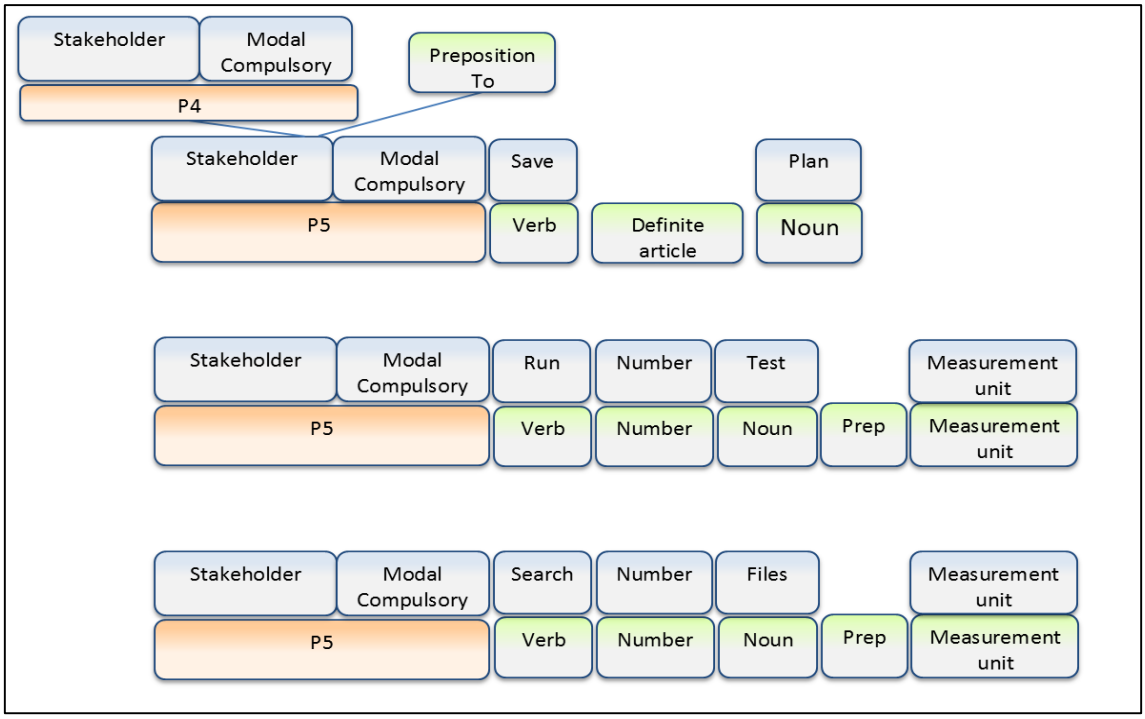

Fig. 48 Generación de patrones ejemplo 2 - Parte 1

Para no repetir el proceso de generación de los patrones, la siguiente imagen muestra el último paso y detalla la composición de cada patrón binario. Al diferenciar los patrones por su semántica, se crea un patrón por cada verbo, asignando la semántica correspondiente. Ocurre lo mismo con los patrones *P11* y *P12* que tiene distintas semánticas aun teniendo las mismas categorías sintácticas.

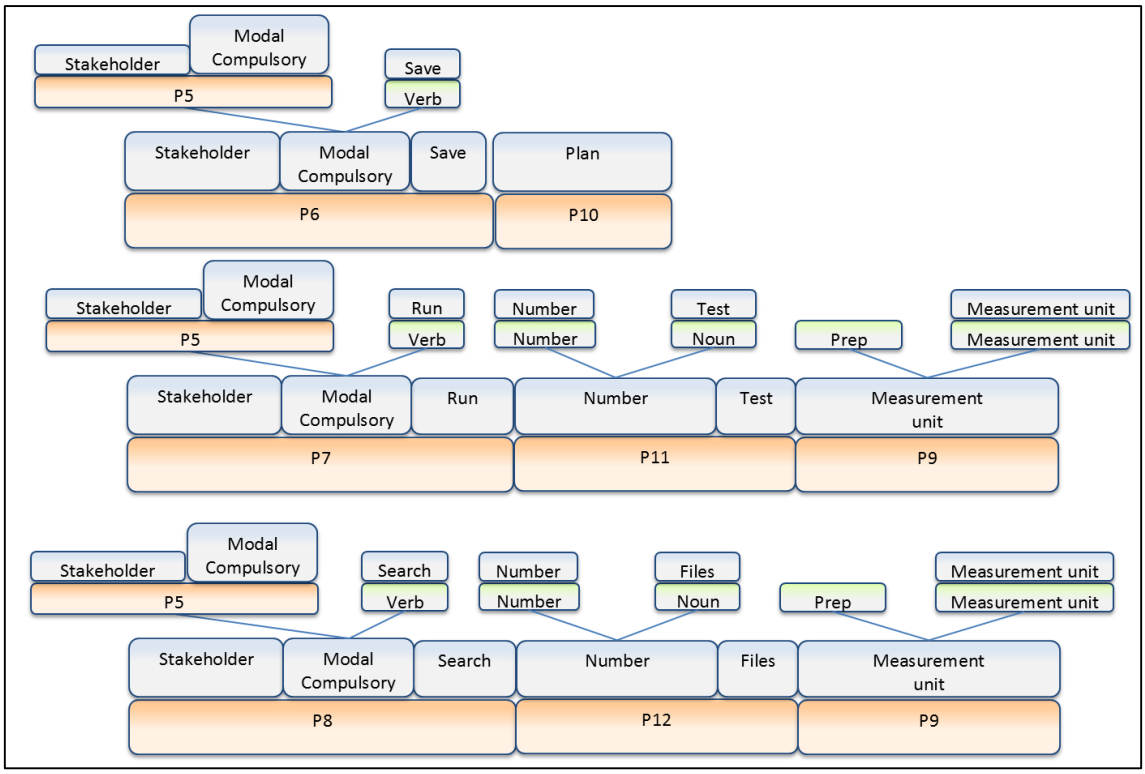

Fig. 49 Generación de patrones ejemplo 2 - Parte 2

La siguiente imagen muestra los tres patrones sintáctico-semánticos que han sido creados si se añade la restricción de diferenciar los patrones por su semántica. Ya que el proceso el más restrictivo, se han creado tres patrones para representar los tres requisitos.

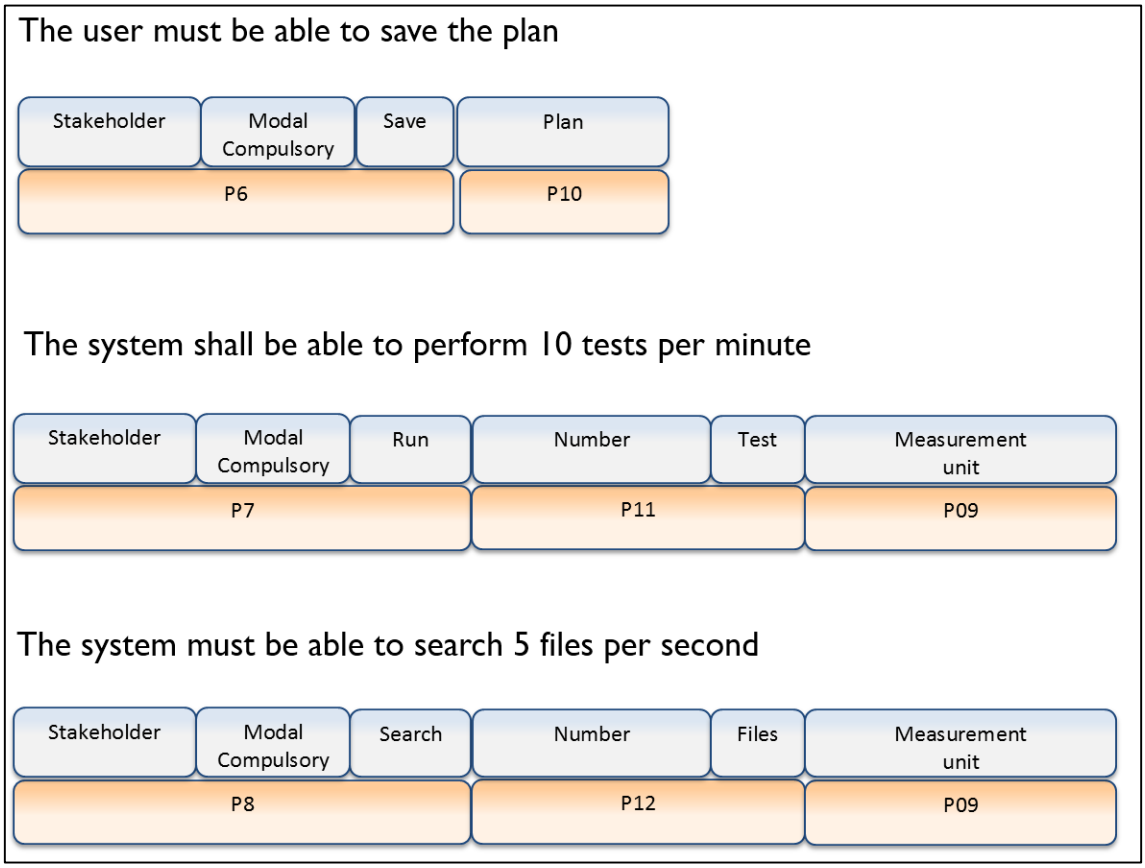

Fig. 50 Generación de patrones ejemplo 2 - Parte 3

3.3.4 Desarrollo del proceso 2: Generación de patrones sintáctico-semánticos

En esta sección se detalla cómo se ha implementado el proceso el desarrollo del caso de estudio de este proceso de la metodología. Como ya se ha comentado, el objetivo de este proceso es la generación de patrones sintáctico-semánticos que representan las estructuras correctas sintáctico-semánticas que deben tener los requisitos. Los patrones sintáctico-semánticos se generan por el procesamiento de un corpus de requisitos con buena calidad proporcionado por el experto. En la experimentación de esta metodología se ha utilizado un corpus de requisitos proporcionados por el Grupo de Trabajo de Requisitos de INCOSE ("INCOSE (International Council on Systems Engineering)" 2016). Este corpus es el mismo que el utilizado en el caso de estudio del primer proceso de la metodología *Generación de clasificadores*, pero sólo se han procesado los requisitos con buena calidad. El motivo de utilizar sólo este conjunto es debido a que el objetivo de este proceso no es comparar la calidad de la estructura de los distintos requisitos, sino obtener un conjunto de patrones que representen la estructura correcta de los requisitos.

Los experimentos realizados tienen la misma implementación, pero se diferencia en las opciones de flexibilidad asociadas en la generación de los patrones. Estas opciones son proporcionadas por la herramienta que implementa para el desarrollo de este proceso. A continuación, se detalla la implementación del proceso, las opciones de flexibilidad y los experimentos propuestos.

3.3.4.1 Tarea 1 – Definir cada requisito como un conjunto de categorías sintácticas y semánticas

Método: identificar frases de cada requisito

En el primer método de esta metodología se realiza la extracción de las frases de los requisitos. En la implementación de este método se ha definido como frase de un requisito los términos que aparecen entre los siguientes signos de puntuación: punto, punto y aparte, dos puntos, signos de interrogación, signos de exclamación.

Método: Tokenización y análisis léxico

Cómo se ha mencionado en la presentación de las herramientas de esta metodología, este método se ha realizado usando las funciones normalizadoras realizadas por el grupo de investigación Knowledge Reuse, del Departamento de Informática de la Universidad Carlos III de Madrid. Como resultados del análisis léxico y la estandarización, se obtienen todos los tokens normalizados de los términos que componen las frases de los requisitos del corpus.

Método: Asignación de categorías sintácticas y semánticas

También mencionada en la sección de herramientas, el procesamiento de los tokens de las frases de los requisitos se ha realizado con las funciones de procesamiento del lenguaje de la herramienta de gestión de requisitos Knowledge Manager (Company 2015a)

3.3.4.2 Tarea 2 – Generar patrones binarios por la frecuencia de repetición de pares de categorías en el conjunto de requisitos

Una vez desarrollada la tarea 1 se obtienen las categorías gramaticales y las semánticas asociadas de los elementos que componen los requisitos del corpus. En la segunda tarea se realizan pares de términos con las categorías y con los subpatrones para sustituir aquella pareja con mayor frecuencia de aparición por un patrón.

Para la implementación de esta segunda tarea se ha desarrollado una herramienta que procesa la información obtenida en la tarea 1. Recordamos que esta segunda tarea es un proceso iterativo de métodos que termina cuando se cumple la condición de parada.

Método: Generación de pares de términos con las categorías sintáctico-semánticas y los subpatrones

Las categorías gramaticales y semánticas son almacenadas en base de datos. Para determinar los pares de categorías consecutivas que más se repiten se ha generado una consulta SQL que selecciona todos los distintos pares de términos consecutivos, los ordena por frecuencia de aparición y devuelve como resultado aquella pareja que

tiene mayor frecuencia. En las iteraciones sucesivas, también son objeto de la consulta SQL los identificadores de subpatrones que han sustituido a los pares con mayor frecuencia.

Método: Sustitución de los pares más frecuentes por subpatrones

La consulta SQL del método anterior devuelve los pares de términos con mayor frecuencia de aparición, y es en este método donde se sustituyen estos pares por el identificador del patrón que es almacenado en el conjunto de patrones.

Este proceso se ha implementado almacenando en base de datos los patrones obtenidos y sustituyendo la pareja de categorías o identificadores de patrones, por el identificador del patrón almacenado. Si alguna de las dos categorías gramaticales tiene asociada una semántica, se almacena dicha semántica en otra tabla de la base de datos y se incluye una referencia a la categoría correspondiente del patrón almacenado. Se considera que un patrón A es subpatrón de otro, si existe un patrón B que los contiene como uno de sus elementos.

Método: Repetición de la tarea y condición de parada

Antes de ejecutar los métodos de esta segunda tarea se pide que se determine el valor mínimo de la frecuencia de parada. Después del proceso de iterativo, cuando la consulta SQL devuelve una pareja de términos con una frecuencia igual al determinado, el proceso iterativo de los métodos finaliza habiendo almacenado en base de datos una jerarquía de patrones binarios sintáctico-semánticos que contienen las categorías gramaticales, semánticas asociadas y subpatrones de los elementos más frecuentes en el corpus de requisitos.

3.3.4.3 Tarea 3 – Generación automática de patrones sintáctico-semánticos

La implementación de esta tarea se ha realizado almacenando en la base de datos los patrones sintáctico-semánticos formados por las categorías y los patrones siguiendo el orden de los elementos de las frases que componen cada requisito. Las categorías y los patrones pueden tener una asociación semántica, lo que significa que los patrones sintáctico-semánticos tienen una mayor representación del conocimiento de los

requisitos. Si varios requisitos son representados por patrones sintáctico-semánticos exactamente iguales, sólo se almacenará en base de datos una entidad de ese patrón.

Ahora cada requisito tiene asociado un conjunto de elementos ordenados en forma de patrón que definen la estructura sintáctico-semántica del requisito. Estos patrones, si son usados como plantillas, servirán de ayuda en la redacción de nuevos requisitos. Al realizar este proceso con un conjunto de requisitos con buena calidad proporcionado por el experto, los patrones sintáctico-semánticos que se generen representarán las estructuras correctas, tanto sintácticas como semánticas, que deben tener todos los requisitos del sistema. Es posible que se necesiten redactar requisitos que no cumplan estrictamente con las restricciones de los patrones generados a partir del corpus, por este motivo se pueden implementar funciones para flexibilizar la composición de los patrones sintáctico-semánticos.

3.3.4.4 Experimentación del proceso 2: Generación de patrones sintáctico-semánticos

Los experimentos desarrollados usando esta metodología se han realizado por el procesamiento de un corpus de 545 requisitos de buena calidad. Los experimentos se han realizado variando las opciones de flexibilidad de la herramienta. A continuación, se exponen las características de estos experimentos.

Experimento 1:

- *Condición de parada en la generación de los patrones binarios*: cuando la mínima frecuencia de aparición sea de 1.
- Los patrones son diferenciados si tienen distintas semánticas asociadas, aunque tenga las mismas categorías gramaticales.

Experimento 2:

- *Condición de parada en la generación de los patrones binarios*: cuando la mínima frecuencia de aparición sea de 1.
- Los patrones no se diferencian por las semánticas asociadas si tienen las mismas categorías gramaticales. Un patrón puede tener varias semánticas asociadas.

3.3.4.5 Resultados de la experimentación del proceso 2: Generación de patrones sintáctico-semánticos

A continuación, se muestran los resultados obtenidos en la experimentación de este proceso de la metodología. Recordamos que se han procesado 545 requisitos con buena calidad del corpus proporcionado por el Grupo de Trabajo de Requisitos de INCOSE ("INCOSE (International Council on Systems Engineering)" 2016). Los resultados contemplan las categorías gramaticales y semánticas más frecuentes, los patrones que más se repiten, y por cada experimento el número de patrones binarios, número de patrones sintáctico-semánticos y características que los componen.

3.3.4.5.1 Asignación de categorías gramaticales y semánticas

Tras el procesamiento de los 545 requisitos escritos en lenguaje natural se han obtenido los siguientes resultados. En la Tabla. 8 se muestran las 20 primeras categorías gramaticales y semánticas que más veces se repiten.

Tabla. 8 Las 20 primeras categorías sintácticas y semánticas

Categoría gramatical	**Frecuencia**	**Semántica**	**Frecuencia**
No clasificado	3612	Modal mandatory	381
Nombre	2389	Modal future	192
Artículo definido	1337	Range (Maximum)	151
Verbo	1095	Allow	114
Verbo modal	686	Modal Optional	110
Símbolo	616	Stakeholder	77
Preposición	579	Range All	73
Verbo "to be"	537	Deny	71
Preposición "to"	491	Range (Minimum)	52
Adjetivo	400	If activation	49
Artículo indefinido	339	Operation	48
Conjunción "and"	335	When activation	33
Preposición "of"	316	Provide	32
Adverbio	308	Modify	28

Not grouping noun	281	Verify	27
Preposición "for"	243	Support	26
Quantifier determiner	180	Range little-few-some	24
Absolute verb	176	Rate	23
Número	160	Range any	19
Conjunción "or"	150	Visualization	18

- Número de categorías gramaticales: 15144
- Número de semánticas asociadas: 1789
- Número de términos encontrados que no son reconocidos por la ontología: 3612
- Las categorías gramaticales que más veces se repite es "Nombre" con un total de 2389 ocurrencias.
- Las semánticas que más veces se repite es " Modal Mandatory" con un total de 381 ocurrencias

3.3.4.5.2 Resultados del experimento 1

El experimento 1 se realiza bajo las siguientes condiciones:

- Procesamiento de 545 requisitos de buena calidad.
- Los patrones se diferencian por sus semánticas. Es decir, dos patrones se consideran distintos si aun teniendo las mismas categorías gramaticales tienen distintas semánticas.
- Condición de parada 1: El proceso de generación de patrones finaliza cuando la frecuencia mínima de aparición de los patrones binarios es igual a 1.

A continuación, se muestra en la Tabla. 9 los primeros patrones binarios encontrados en la primera ejecución del proceso iterativo, antes de iniciar el proceso de reemplazo:

Tabla. 9: Los 10 primeros patrones binarios en la primera iteración del proceso

Categoría izquierda	Semántica izquierda	Categoría derecha	Semántica derecha	Frecuencia
No clasificado	--	No clasificado	--	1058
Artículo definido	--	Nombre	--	767
No clasificado	--	Nombre	--	500
Nombre	--	No clasificado	--	434
Artículo definido	--	No clasificado	--	384
Nombre	--	Verbo modal	Obligatoriedad	264
No clasificado	--	Símbolo	--	253
Símbolo	--	No clasificado	--	233
Verbo modal	Obligatoriedad	Verbo "to be"	--	188
Verbo "to be"	--	Adjetivo	--	178
Nombre	--	Nombre	--	178
Nombre	--	Símbolo	--	174
Adjetivo	--	Preposición "to"	--	155
Artículo indefinido	--	No clasificado	--	146
No clasificado	--	Conjunción "and"	--	140
Nombre	--	Verbo modal	Futuro	137
No clasificado	--	Verbo	--	136
Nombre	--	Preposición "of"	--	132
Verbo "to be"	--	Verbo	--	130
Preposición "of"	--	Artículo definido	--	118

El patrón binario que más se repite, con 1058 apariciones, es el formado por términos no clasificados por la ontología, seguido del patrón formado por las categorías gramaticales [*Artículo definido* + *Nombre]* con una frecuencia de 767. El primer patrón con mayor frecuencia con una semántica asociada es *Nombre* + *Verbo modal* con semántica de *obligatoriedad*.

A continuación, se muestran los resultados una vez finalizado el proceso iterativo en la ejecución de los métodos en la creación y sustitución de patrones binarios:

- Número total de patrones binarios generados: 996
- Número de patrones con semántica: 193

El patrón que más veces se ha distinguido por tener distintas semánticas asociadas pero las mismas categorías gramaticales es el formado por el patrón con identificador P4, en su lado izquierdo, y por la categoría gramatical "Verbo", en el lado derecho. En la siguiente tabla se muestran los patrones con estas categorías gramaticales:

Tabla. 10 Patrones compuestos con las mismas categorías gramaticales pero distintas semánticas

Identificador del patrón	Elemento izquierdo	Elemento derecho	Semántica derecha
P108	P4	Verbo	Ninguna
P92	P4	Verbo	Permitir
P176	P4	Verbo	Proporcionar
P202	P4	Verbo	Mantener
P254	P4	Verbo	Comunicar
P332	P4	Verbo	Funcionamiento

El patrón con el identificador P108 que aparece en la tabla anterior no tiene semántica asociada. Los demás han sido distinguidos por tener otras semánticas asociadas. Esto no ocurrirá en el siguiente experimento 2 donde un mismo patrón puede tener varias semánticas.

Patrones sintáctico-semánticos

A continuación, se presentan los resultados que se han obtenido del experimento 1 en la generación automática de patrones una vez finalizado el proceso iterativo de generación de patrones binarios.

- Número de patrones sintáctico-semánticos del primer experimento: 545 (Un patrón sintáctico-semántico por cada uno de los requisitos).
- Número medio de elementos que contienen los patrones sintáctico-semánticos: 10.586
- Máximo número de elementos que contiene un patrón sintáctico-semántico: 52

3.3.4.5.3 Resultados del experimento 2

Las condiciones del experimento 2 son:

- Procesamiento de 545 requisitos de buena calidad.
- Los patrones no se diferencian por sus semánticas. Es decir, un patrón puede tener varias semánticas asociadas.
- Condición de parada 1: El proceso de generación de patrones finaliza cuando la frecuencia mínima de aparición de los patrones binarios es igual a 1.

A continuación, se muestran los resultados una vez finalizado el proceso iterativo de los métodos para la creación y sustitución de patrones binarios:

- Número total de patrones binarios generados: 906
- Número de patrones con semántica: 169

Los patrones con mayor número de semánticas asociadas con los patrones con identificador P16 y el patrón P55 con un total de 16 semánticas cada uno.

El patrón P16 está formado por:

Patrones sintáctico-semánticos

Los resultados de los patrones sintáctico-semánticos que se han generado en este experimento son:

- Número de patrones sintáctico-semánticos del segundo experimento: 442
- Número medio de elementos que contienen los patrones sintáctico-semánticos: 9.81
- Máximo número de elementos que contiene un patrón sintáctico-semántico: 52

El número de patrones sintáctico-semánticos creados es menor que el número de requisitos procesados, habrá por tanto patrones sintáctico-semánticos que reconozcan estructuralmente más de un requisito. Este resultado es consecuencia reducir las restricciones en comparación al primer experimento debido a que no se distinguen los patrones por su semántica.

3.3.5 Comparación de resultados de los experimentos 1 y 2

En esta sección se muestran los resultados de los patrones binarios y patrones sintáctico-semánticos generados en los dos experimentos.

- Patrones binarios generados en el primer experimento: 996
- Número de patrones con semántica del primer experimento: 193
- Patrones binarios generados en el segundo experimento: 906
- Número de patrones con semántica del segundo experimento: 169
- Patrones sintáctico-semánticos generados en el experimento 1: 545
- Patrones sintáctico-semánticos generados en el experimento 2: 442
- Número medio de elementos que contienen los patrones sintáctico-semánticos del experimento 1: 10.586
- Máximo número de elementos que contiene un patrón sintáctico-semántico en el experimento 1: 52
- Número medio de elementos que contienen los patrones sintáctico-semánticos del experimento 2: 9.81
- Máximo número de elementos que contiene un patrón sintáctico-semántico en el experimento 2: 52

3.3.6 Beneficios derivados del proceso 2

Se ha utilizado este proceso de la metodología en diferentes ámbitos para generar patrones sintáctico-semánticos y obtener así una base de conocimiento de las estructuras gramaticales que componen las frases de los documentos tratados. A continuación, se exponen estos trabajos junto con una descripción del problema.

Extracción de patrones sintáctico-semánticos de documentos de patentes

Proyecto Final de Carrera
Ingeniería Técnica en Informática de Gestión
Escuela Politécnica Superior
Autor: Leticia Arroyo Minguela
Tutores: Prof. Dr. Anabel Fraga y Prof. Dr. Valentín Moreno
Octubre 2015
Resumen:

En este trabajo se presenta un análisis realizado sobre el contenido de documentos de patentes. Se obtienen mediante el estudio de frecuencias de aparición los términos más utilizados, categorías sintácticas y semánticas, con el objetivo de generar una base ontológica. Se presenta también el análisis de patrones sintáctico-semánticos generados.

Evaluación de un sistema de procesamiento del lenguaje natural de la banca

Proyecto Final de Carrera
Ingeniería Técnica en Informática de Gestión
Escuela Politécnica Superior
Autor: Nuria de la O Maestro
Tutores: Prof. Dr. Anabel Fraga y Prof. Dr. Valentín Moreno
Octubre 2015
Resumen:

El trabajo investigación la composición de documentos del sector bancario. Mediante el análisis de frecuencia, se obtienen y presentan los siguientes datos: términos del dominio más utilizados, categorías gramaticales más frecuentes, así como un conjunto de patrones sintáctico-semánticos capaces de orientar en la redacción de documentos en el sector bancario.

Evaluation of a natural language processing system in public health

Trabajo Final de Máster

Máster en Ciencia y Tecnología Informática
Escuela Politécnica Superior
Autor: Valeria Rodríguez Barberena
Tutor: Prof. Dr. Anabel Fraga y Prof. Dr. Valentín Moreno
Octubre 2014
Resumen:

Se realiza un estudio analítico de la composición de documentos en el ámbito de la sordera genética. Se presentan los términos del dominio más utilizados, junto con sus categorías y un conjunto de patrones sintáctico-semánticos orientados a reconocer las estructuras, así como orientar en la redacción de este tipo de documentos.

3.4 EJECUCIÓN DE LA METODOLOGÍA PARA OPTIMIZAR LA CALIDAD DE LOS REQUISITOS

Se han presentado los dos procesos que componen la metodología por separado y se han detallado las tareas y métodos que las componen para poder implementarlos. Pero los dos procesos tienen el objetivo común de mejorar la calidad de los requisitos, teniendo cada uno un enfoque de acción distinto.

Para los requisitos ya existentes que forman parte de las especificaciones de los proyectos, se ha presentado el proceso de *generación de clasificadores* con el cual se

puede conocer la calidad de corrección de estos requisitos haciendo uso del clasificador resultado.

Para los nuevos requisitos que se quieran incluir en las especificaciones, se ha propuesto el proceso de *generación de patrones sintáctico-semánticos* que obtiene patrones que podrán ser utilizados para orientar la redacción de los nuevos requisitos asegurando una estructura correcta en su composición.

Aun teniendo dos enfoques distintos, es posible utilizar estos dos procesos de forma iterativa para mejorar la calidad, contemplamos estos dos casos.

- Se redacta un requisito haciendo uso de los patrones para conseguir una estructura correcta. Una vez escrito, es procesado por el clasificador para saber la calidad. Si la calidad del requisito es mala, el sistema de optimización de requisitos propuesto [*3.2.5 Optimizar la calidad de los requisitos con el menor esfuerzo mediante sugerencias de corrección automáticas*] nos advertirá de las métricas que se deben cambiar para transformar la calidad del requisito. Por tanto, el requisito deberá ser modificado, pero teniendo en cuenta la autoría de los patrones que nos guiarán en las posibilidades de cambios en la estructura. El proceso de mejora del requisito termina, cuando el clasificador estima que el requisito tiene buena calidad y es reconocido estructuralmente por los patrones sintáctico-semánticos.
- La otra posibilidad es cuando queremos cambiar la calidad de un requisito existente. El proceso es el mismo que el caso anterior, modificando el requisito para que el clasificador determine que es correcto y teniendo en cuenta las estructuras de los patrones. De hecho, una de las métricas que se pueden utilizar en la generación de los clasificadores es “Reconocimiento por patrones” por lo que el mismo clasificador ya tendría en cuenta la estructura correcta de los requisitos.

CAPÍTULO 4: DISCUSIÓN Y CONCLUSIONES

El siguiente capítulo contiene la discusión y las conclusiones extraídas de la investigación. Se divide la sección en los dos procesos de la metodología.

4.1 DISCUSIÓN Y CONCLUSIONES DEL PROCESO 1: PROCESO PARA LA CLASIFICACIÓN DE REQUISITOS

El primer proceso que compone la metodología propone generar clasificadores utilizando algoritmos de aprendizaje automático de inducción de reglas, sobre conjuntos de requisitos clasificados por un experto en función de su calidad, con el fin de estimar la calidad de nuevos requisitos.

Se ha propuesto utilizar conjuntos de métricas de calidad para representar la calidad de los requisitos escritos en lenguaje natural, y estos conjuntos han sido utilizados por los algoritmos para inducir reglas que permitan valorar la calidad de un requisito en función de las métricas que lo representa.

Dos enfoques de implementación de las instancias de aprendizaje han sido presentados para permitir procesar tanto conjuntos de requisitos clasificados de forma balanceada, es decir, se componen de un mismo número aproximado de requisitos de distinta calidad; como de conjuntos donde existe una diferencia significativa entre las clasificaciones de los requisitos.

Se ha presentado un caso de estudio empleando un conjunto de 1035 requisitos clasificados por el Grupo de Trabajo de Requisitos de la organización INCOSE ("INCOSE (International Council on Systems Engineering)" 2016). El caso de estudio plantea la generación de clasificadores con dos algoritmos de inducción de reglas PART y C 4.5 y dos algoritmos de clasificadores homogéneos Bagging y Boosting que toman como base los dos algoritmos anteriores. Los clasificadores han sido generados con los dos enfoques de implementación de instancias de aprendizaje propuestas obteniendo diferentes resultados tanto en el porcentaje de acierto como en la eficiencia en la

generación de los clasificadores. Los porcentajes de acierto han sido establecidos mediante validación cruzada.

Se han comparado los resultados de los dos enfoques para cada uno de los clasificadores y se ha propuesto el uso de cada implementación del proceso de la metodología en función de la composición del conjunto de requisitos y de las características del problema, priorizando el porcentaje de acierto en la estimación de la calidad de requisitos o la eficiencia en la generación de los clasificadores. El segundo enfoque, donde se ha invertido más tiempo en la generación de los clasificadores, consigue los mejores porcentajes de acierto en la estimación de la calidad que otorgaría el experto, llegando a alcanzar un 87.7 % para el algoritmo C 4.5. Por otra parte, el primer enfoque consigue en su mejor porcentaje de acierto un 86.3% para los clasificadores generados con los algoritmos Bagging PART y Boosting PART, y se invierte tan sólo 5.26 segundos en la generación de los clasificadores con el algoritmo Bagging PART y 4.99 segundos con el algoritmo Boosting PART, mejorando considerablemente la eficiencia si se compara con el tiempo invertido con el enfoque se comparación por pares. Para este enfoque el algoritmo que menos tiempo necesita para generar el clasificador es el C 4.5 con 275.7 segundos, siendo el algoritmo Boosting PART el que más tiempo invierte, necesitando 6.107 horas.

Se ha demostrado mediante ejemplos gráficos que es necesario utilizar funciones complejas para poder acotar los intervalos de calidad en métricas de los requisitos cuando el objetivo es estimar adecuadamente la calidad de los mismos. El uso de funciones simples como la ponderación de reglas lineales no es suficiente para establecer regiones diferenciadas en relaciones complejas de métricas, aumentando considerablemente su falta de precisión cuando las relaciones se producen entre varias métricas. Sin embargo, encontrar estas funciones complejas resulta una tarea complicada, siendo una de las mejores opciones de búsqueda el uso de técnicas de aprendizaje automático.

Se ha formalizado un sistema de sugerencias de modificación de requisitos para transformar la calidad. Este sistema utiliza las reglas de un clasificador que valoran los requisitos con buena calidad, y selecciona la regla que implica un menor esfuerzo para

transformar un requisito de mala calidad en buena calidad. Este esfuerzo obtiene por el número de métricas que se han de cambiar en el requisito y mediante pesos asignados a cada métrica. Una vez transformado el requisito para que cumpla con alguna de las reglas se podrá considerar que el requisito tiene las condiciones para ser considerado de calidad por el experto que utilice la metodología.

Realizado este trabajo de investigación para el primer proceso de la metodología, se ha demostrado que es posible aprender, mediante métricas de corrección, el juicio de un experto del dominio sobre la calidad de los requisitos, con el fin de emular automáticamente la calidad de nuevos requisitos. Además, se proporciona sistema de sugerencias para modificar la calidad de los requisitos, alcanzando así dos de los objetivos planteados en el inicio de esta tesis:

- Emular la clasificación de calidad que proporcionaría un experto del proyecto y del domino sobre requisitos.
- Optimizar la calidad de los requisitos con el menor esfuerzo mediante sugerencias de corrección automáticas.

4.2 Discusión y conclusiones del proceso 2: Proceso para la generación de patrones sintáctico-semánticos

El segundo proceso de la metodología propone el procesamiento de un corpus de requisitos para la generación de patrones sintáctico-semánticos que los representen estructuralmente. La finalidad de estos patrones es que sirvan de ayuda en la redacción de nuevos requisitos con estructuras sintáctico-semánticas correctas.

Este proceso comienza con un análisis léxico, tokenización y asignación de categorías sintácticas y semánticas sobre requisitos clasificados con buena calidad y que proporciona el experto del dominio. La siguiente tarea consiste en generar patrones binarios mediante un ciclo iterativo de métodos que sustituyen a las categorías en función de la frecuencia de aparición, hasta alcanzar una condición de parada. Por último, con la información obtenida se genera un conjunto de patrones sintáctico-semánticos que representan la estructuralmente a los requisitos del corpus. Un mismo patrón puede contener la estructura de varios requisitos, la cantidad de

patrones generados dependerá de las restricciones impuestas en proceso. Cuantas más restricciones se exijan, mayor número de patrones deberán ser generados para reconocer el conjunto de los requisitos.

Para realizar un caso de estudio, se ha desarrollado una herramienta que implementa el proceso de la metodología. La herramienta incluye diferentes opciones para flexibilizar las restricciones en la generación de patrones y para comprobar estas opciones se han presentado dos experimentos de generación de patrones sobre el conjunto de requisitos utilizado en el primer proceso pero seleccionando aquellos requisitos con buena calidad. Los resultados obtenidos han revelado distintas formas de generar patrones sintáctico-semánticos diferenciándose en las restricciones incorporadas en el proceso.

La investigación desarrollada en este proceso demuestra que es posible generar patrones de requisitos, pudiéndose usar estos patrones como guía en la redacción de requisitos asegurando una estructura sintáctico-semántica correcta, por lo que se ha alcanzado unos de los objetivos planteados en la tesis:

- Obtener un conjunto de patrones destinados a orientar la redacción de nuevos requisitos asegurando la correcta estructura sintáctico-semántica.

Este proceso de generación de patrones es una primera aproximación de representación de la estructura de requisitos. Los experimentos realizados demuestran que es posible generar automáticamente patrones sintáctico-semánticos de requisitos mediante composiciones jerárquicas, pero el número de patrones, aun siendo inferior al número de requisitos, es demasiado elevado. Tomando como punto de partida esta investigación, se pueden analizar otras opciones de flexibilización para reducir el número de patrones, como la generalización del uso de elementos opcionales y comodines, o el análisis detallado de las composiciones semánticas para determinar elementos comunes semánticos, proporcionar así un mayor peso al contenido semántico y alejar las restricciones de contenido sintáctico.

4.3 DISCUSIÓN Y CONCLUSIONES DE LA METODOLOGÍA

Con los dos procesos desarrollados en la metodología de esta investigación se han alcanzado los objetivos planteados al inicio de la tesis. Se han utilizado técnicas de inteligencia artificial y se ha realizado procesamiento del lenguaje natural sobre corpus de requisitos facilitados por expertos del proyecto y del dominio, para proporcionar automatización en la optimización de requisitos, siendo verificados mediante casos de estudio, demostrando así la hipótesis planteada:

- Mediante técnicas de inteligencia artificial y de procesamiento de lenguaje natural, es posible establecer un método automático para la optimización de la calidad de los requisitos de ingeniería.

CAPÍTULO 5: LÍNEAS FUTURAS

A continuación, se proponen líneas futuras de la metodología. Para mejorar el porcentaje de acierto de los clasificadores del primer proceso, se pueden variar o idear nuevas formas de implementación de las instancias de aprendizaje. Una modificación que se podría realizar en el enfoque por comparación de pares, es adaptar con mayor fidelidad el trabajo presentado en (Moreno 2010), donde los clasificadores son generados a partir de recursos ordenados. Los cambios que se proponen son utilizar los clasificadores para evaluar los mismos requisitos con los que han sido entrenados los algoritmos. Todos los requisitos serán comparados con todos para poder establecer una medida de calidad de cada requisito con respecto al grupo y obtener así una ordenación de los requisitos en función de su calidad. Con el conjunto de aprendizaje ordenado se pueden generar nuevas instancias de aprendizaje comparando cada requisito con aquellos que tenga una valoración inferior en el conjunto y establecer que dicho requisito es mejor que todos con los que ha sido comparado. Utilizando los algoritmos de aprendizaje automático es posible generar un clasificador que permita estimar la posición de un nuevo requisito y así poder establecer una valoración de la calidad en función de esa posición en el conjunto. El número de instancias del modelo sería todas las posibles parejas de requisitos del conjunto de aprendizaje, no sólo las parejas de requisitos de distinta calidad como en el trabajo propuesto, por lo que el tiempo empleado en la generación del clasificador aumentaría considerablemente.

Con el fin de mejorar la eficiencia en la generación de los clasificadores en el segundo enfoque, se puede crear un conjunto de aprendizaje compuesto sólo por requisitos que representen a todos los requisitos del corpus. Este proceso se puede realizar, escogiendo aleatoriamente dos requisitos de distinta calidad del conjunto y generando un clasificador con ellos. Una vez generado el clasificador, se introducen los requisitos restantes para estimar la calidad. Aquellos donde la calidad haya sido acertada se extraen del conjunto, y se repite el proceso añadiendo otros dos requisitos para generar un nuevo clasificador. Este proceso se repite hasta que todos los requisitos del conjunto proporcionado hayan sido correctamente clasificados, con el

resultado final de un conjunto reducido de requisitos que formarán el conjunto de aprendizaje. De esta manera se puede reducir el tiempo en la generación de los clasificadores con los modelos de comparación por pares.

Se pueden emplear otras técnicas de inteligencia artificial para la generación de los clasificadores, como máquinas vectoriales o redes bayesianas, lo que conllevaría idear nuevas técnicas de asesoramiento automático para mejorar la calidad de los requisitos clasificados como defectuosos.

Debido a que la clasificación de la calidad se realiza en función de las opiniones de los expertos, otra posible línea futura de investigación podría ser aplicar lógica difusa y comprobar resultados al generar clasificadores difusos.

Las líneas futuras planteadas en la generación de patrones sintáctico-semánticos se enfocan en la composición de los patrones. El proceso planteado en esta investigación es una primera aproximación de generación de patrones, una posible modificación sería la generación de patrones partiendo de patrones básicos de N términos no sólo con dos términos como se plantean en el desarrollo.

Se podría analizar mediante un procesamiento automático los patrones binarios con mayor frecuencia, los elementos que los separan, con el fin de sustituirlos por elementos opcionales y comodines. De esta manera se podrían reducir el número de patrones consiguiendo que cada patrón reconociera un mayor número de requisitos.

La verdadera fortaleza de los patrones es la asociación de semánticas a las categorías, sería posible realizar procesamiento del contenido semántico de los patrones y generalizar patrones semánticos que sinteticen el significado de los requisitos, con lo que se permitiría orientar la redacción de nuevos requisitos centrándose en la composición semántica y alejándose de las restricciones sintácticas.

CAPÍTULO 6: APORTACIONES DE LA INVESTIGACIÓN

La realización de la tesis ha derivado en varios trabajos de investigación que completan los fundamentos propuestos en las hipótesis y los objetivos.

A methodology for the classification of quality of requirements using machine learning techniques (Parra et al. 2015)

Resumen:

Se propone en esta publicación, una metodología orientada a la generación de clasificadores de calidad de requisitos, mediante el procesamiento automático de corpus de requisitos clasificados en función de su calidad. El corpus de requisitos es proporcionado por el experto del proyecto que quiera usar la metodología, incorporando así, las restricciones de calidad que se consideren, oportunas para establecer el grado de exigencia que requiera el proyecto. Debido a esto, el resultado de la metodología se adapta a las exigencias de calidad de los diferentes proyectos y culturas organizacionales.

Method for generating semantic patterns (Moreno et al. 2013)

Invención en explotación por la empresa: CISET (Centro de Innovación y Soluciones Empresariales y Tecnológicas).

Nº de publicación: WO/2014/049186
Nº de solicitud internacional: PCT/ES2013/070638
País de prioridad: España
Fecha de publicación: 03 /04/ 2014
Entidad titular: Universidad Carlos III de Madrid

Resumen:

Sistema en el que se implementaron distintos pasos metodológicos que permiten la generación automática de patrones de indexación, teniendo como origen un corpus y como salida una lista de patrones semánticos ordenados por frecuencia.

Applying INCOSE Rules for writing high-quality requirements in Industry (Fuentes et al. 2016)

Resumen:

El artículo presenta la implementación de algunas pautas establecidas por la organización INCOSE, para escribir requisitos en la industria con alta calidad. Se expone también, los beneficios derivados de aplicar las técnicas, métodos y herramientas, en la mejora del proceso en la ingeniería de sistemas y se evalúa de impacto en la calidad de los requisitos.

Automatic pattern generator of natural language text applied in public health (Fraga et al. 2016)

Aceptado en Special Issue y Pendiente de publicación

Resumen:

Debido a que la mayoría de los documentos en investigación se redactan de forma similar en cada nicho de domino, la investigación presenta un análisis realizado mediante el estudio de la frecuencia de repetición de categorías gramaticales, términos del dominio más utilizados, así como el descubrimiento de patrones y sus frecuencias. Esta investigación emplea una base ontológica del ámbito de salud y analiza 800 documentos que pertenecen a este ámbito.

Application of machine learning techniques to the flexible assessment and improvement of requirements quality (Moreno et al., n.d.)

Pendiente de revisión en revista

Resumen:

En este artículo se muestra cómo una combinación lineal de métricas es insuficiente para calcular adecuadamente una calidad global de requisitos de ingeniería, y se propone el desarrollo de una metodología flexible para evaluación y mejora de la calidad de los requisitos, que puede ser fácilmente adaptable a diferentes contextos, proyectos, organización y estándares de calidad, con alto grado de automatización.

Proyecto CRYSTAL - CRitical sYSTem engineering AcceLeration (CRYSTAL-ARTEMIS 2016)

Esta investigación ha recibido financiación del séptimo programa del marco de trabajo de la unión europea (FP7/2007 - 2013) CRSYTAL - CRITICAL SYSTEM ENGINEERING ACCELERATION del compromiso común concedido nº 332830 y desde los programas nacionales específicos y/o autoridades de financiación. Esta tesis se considera una aportación al citado proyecto.

CAPÍTULO 7: REFERENCIAS

(Esa), European Space Agency. 1995. *Guide to the Software Requirements Definition Phase*.

(Incose), Requirements Working Group. 2015. "Guide for Writing Requirements," no. April: 59. http://www.incose.org.

A. Fantechi, S. Gnesi, G. Lami, and A. Maccari. 2003. "Applications of Linguistic Techniques for Use Case Analysis." *Journal Requirements Engineering*, 161–170. doi:10.1007/s00766-003-0174-0.

Aberdeen, John, John Burger, David Day, Lynette Hirschman, Patricia Robinson, and Marc Vilain. 1995. "MITRE: Description of the {Alembic} System Used for {MUC-6}." In *Proceedings of the 6th Conference on Message Understanding*, 141–55.

Abney, Steven. 1996. "Partial Parsing via Finite-State Cascades." In *Natural Language Engineering 2*, 337–44.

Abney, Steven. 1997. "Part-of-Speech Tagging and Partial Parsing." In *Corpus-Based Methods in Language and Speech Processing*, 118–36. Springer Netherlands.

Aceituna, Daniel, Gursimran Walia, Hyunsook Do, and Seok-Won Lee. 2014. "Model-Based Requirements Verification Method: Conclusions from Two Controlled Experiments." *Information and Software Technology* 56 (3): 321–34. doi:10.1016/j.infsof.2013.11.004.

Aho, Alfred V., Ravi Sethi, and Jeffrey D. Ullman. 1986. *[1] A. V. Aho, R. Sethi, J.D. Ullman., Compilers: Principles, Techniques and Tools, Addison Wesley, 1986.Compilers: Principles, Techniques and Tools*. Addison wesley.

Alencar, Fernanda, Giovanni Giachetti, and Oscar Pastor. 2009. "From I* Requirements Models to Conceptual Models of a Model Driven Development Process." *The Practice of Enterprise Modeling: Second IFIP WG 8.1 Working Conference, PoEM 2009, Stockholm, Sweden, November 18-19, 2009, Proceedings*, 99–114. doi:10.1007/978-3-642-05352-8_9.

Alexander, I.F., and R. Stevens. 2002. *Writing Better Requirements*. *Chemistry Biodiversity*. Vol. 1. Addison-Wesley. http://onlinelibrary.wiley.com/doi/10.1002/cbdv.200490137/abstract\nhttp://scholar.google.com/scholar?hl=en&btnG=Search&q=intitle:Writing+Better+Requirements#0.

Ali, Raian, Fabiano Dalpiaz, and Paolo Giorgini. 2013. "Reasoning with Contextual Requirements: Detecting Inconsistency and Conflicts." *Information and Software Technology* 55 (1): 35–57. doi:10.1016/j.infsof.2012.06.013.

Ambler, Scott. 2000. “Requirements Engineering Patterns: Three Approaches to Motivate Your Developers to Invest Time in Taking Care of First Things First.” http://www.drdobbs.com/architect/184414612.

Arora, C, M Sabetzadeh, L Briand, F Zimmer, and R Gnaga. 2013. “RUBRIC: A Flexible Tool for Automated Checking of Conformance to Requirement Boilerplates.” *2013 9th Joint Meeting of the European Software Engineering Conference and the ACM SIGSOFT Symposium on the Foundations of Software Engineering, ESEC/FSE 2013 - Proceedings*, 599–602. doi:10.1145/2491411.2494591.

Australia, University of South. n.d. “TigerPro.”

Beesley, Kenneth R., and Lauri Karttunen. 2003. “Two-Level Rule Compiler.” *History*.

Bloomberg, Jason, and Ronald Schmelzer. 2006. *Service Orient or Be Doomed!: How Service Orientation Will Change Your Business*. John Wiley & Sons.

Bøegh, Jørgen. 2008. “A New Standard for Quality Requirements.” *IEEE Software* 25 (2): 57–63. doi:10.1109/MS.2008.30.

Bourque, P, and R Dupuis. 2004. *Guide to the Software Engineering Body of Knowledge 2004 Version*. *SWEBOK 2004 Guide to the Software Engineering Body of Knowledge*. Vol. 1. doi:10.1109/SESS.1999.767664.

Braude, Eric J. 2000. *Software Engineering: An Object-Oriented Perspective*. John Wiley \\& Sons, Inc.

Breiman, Leo. 1996. “Bagging Predictors.” *Machine Learning* 24 (421): 123–40. doi:10.1007/BF00058655.

Brooks, Frederick P. 1987. “Essence and Accidents of Software Engineering.” *Computer* 20 (4): 10–19. doi:10.1109/MC.1987.1663532.

Caliber. 2015. “Borland.” *Borland*. Accessed May 1. www.borland.com/.

Carreras, Xavier, and Marquez Lluis. 2004. “Phrase Recognition by Filtering and Ranking with Perceptrons.” In *Recent Advances in Natural Language Processing III: Selected Papers from RANLP*.

Cendrowska, Jadzia. 1987. “PRISM: An Algorithm for Inducing Modular Rules.” *International Journal of Man-Machine Studies* 27 (4): 349–70. doi:10.1016/S0020-7373(87)80003-2.

Chantree, Francis J, Anne De Roeck, Bashar Nuseibeh, and Alistair Willis. 2006. “Identifying Nocuous Ambiguity in Natural Language Requirements.” *14th IEEE International Requirements Engineering Conference (RE’06)* Doctor of: 203. doi:10.1109/RE.2006.31.

Christer, Samuelsson, and Mats Wiren. 2000. “Parsing Techniques.” In *Handbook of Natural Language Processing*, 59–91.

Clark, P., and T. Niblett. 1989. “The {CN}2 Rule Induction Algorithm.” *Machine Learning*

3 (4): 261–84. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.51.3672&rep=rep1&type=pdf.

Company, The Reuse. 2015a. “Knowledge Manager.” *The Reuse Company*. Accessed May 1. www.thereusecompany.com.

———. 2015b. “RQA Requirements Quality Analyzer.” *The Reuse Company*. Accessed May 1. www.reusecompany.com.

Cowie, Jim, and Wendy Lehnert. 1996. “Information Extraction.” *Communications of the ACM* 39 (1): 80–91. doi:10.1145/234173.234209.

CRYSTAL-ARTEMIS. 2016. “CRitical sYSTem Engineering AcceLeration (CRYSTAL EU Project).” Accessed September 4. http://www.crystal-artemis.eu/.

Daciuk, Jan, Stoyan Mihov, Bruce W. Watson, and Richard E. Watson. 2000. “Incremental Construction of Minimal Acyclic Finite-State Automata.” *Computational Linguistics*.

Dale, Robert. 2000. “Symbolic Approaches to Natural Language Processing.” In *Handbook of Natural Language Processing*, edited by and Hermann Moisl Robert Dale , Harold Somers. New York: Marcel Dekker.

Daramola, Olawande, Guttorm Sindre, and Tor Stalhane. 2012. “Pattern-Based Security Requirements Specification Using Ontologies and Boilerplates.” *2012 2nd IEEE International Workshop on Requirements Patterns, RePa 2012 - Proceedings*, 54–59. doi:10.1109/RePa.2012.6359973.

Dargan, John L., Enrique Campos-Nanez, Pavel Fomin, and James Wasek. 2014. “Predicting Systems Performance through Requirements Quality Attributes Model.” *Procedia Computer Science* 28: 347–53. doi:10.1016/j.procs.2014.03.043.

de Gea, Juan Manuel Carrillo, Joaquín Nicolás, José Luis Fernández Alemán, Ambrosio Toval, Aurora Vizcaíno, and Christof Ebert. 2013. “Reusing Requirements in Global Software Engineering.” *Managing Requirements Knowledge*, 171–97.

de Sousa, Thiago C., Jorge R. Almeida, Sidney Viana, and Judith Pavón. 2010. “Automatic Analysis of Requirements Consistency with the B Method.” *ACM SIGSOFT Software Engineering Notes* 35 (2): 1. doi:10.1145/1734103.1734114.

Déjean, Hervé. 2000. “Learning Syntactic Structures with XML.” *Proceedings of CoNLL-2000 and LLL-2000*, no. Table 2: 4–6. doi:10.3115/1117601.1117632.

Dietterich, Thomas G. 2000. “Ensemble Methods in Machine Learning.” *Lecture Notes in Computer Science* 1857: 1–15. doi:10.1007/3-540-45014-9_1.

Dietterich, Thomas G. 1997. “Machine-Learning Research.” *AI Magazine* 18 (4): 97. doi:10.1609/aimag.v18i4.1324.

Estefan, Jeff a. 2008. “Survey of Model-Based Systems Engineering (MBSE) Methodologies 2 . Differentiating Methodologies from Processes , Methods , and

Lifecycle Models." *Environment*. doi:10.1109/35.295942.

Fabbrini, F., M. Fusani, S. Gnesi, and G. Lami. 2001a. "The Linguistic Approach to the Natural Language Requirements Quality: Benefit of the Use of an Automatic Tool." *26th Annual NASA Goddard Software Engineering Workshop, IEEE/NASA SEW 2001*, 97–105. doi:10.1109/SEW.2001.992662.

Fabbrini, F, M Fusani, S Gnesi, and G Lami. 2001b. "An Automatic Quality Evaluation for Natural Language Requirements." *REFSQ 2001: Proceedings of the Seventh International Workshop on RE: Foundation for Software Quality* 1: 150–64. http://fmt.isti.cnr.it/WEBPAPER/P11RESFQ01.pdf\nhttp://www.citeulike.org/user/swatipendyala/article/3729653.

Farfeleder, Stefan, Thomas Moser, Andreas Krall, Tor Stålhane, Herbert Zojer, and Christian Panis. 2011. "DODT: Increasing Requirements Formalism Using Domain Ontologies for Improved Embedded Systems Development." *Proceedings of the 2011 IEEE Symposium on Design and Diagnostics of Electronic Circuits and Systems, DDECS 2011*, 271–74. doi:10.1109/DDECS.2011.5783092.

Fraga, Anabel. 2010. "A Methodology for Reusing Any Kind of Knowledge at Low Cost: Universal Knowledge Reuse." University Carlos III de Madrid.

Fraga, Anabel, Juan Llorens, Eugenio Parra, and Moreno Valentín. 2016. "Automatic Pattern Generator of Natural Language Text Applied in Public Health." In *Communications in Computer and Information Science*.

Frakes, W B, and S Isoda. 1994. "Success Factors of Systematic Reuse." *Software, IEEE* 11 (5): 14–19. doi:10.1109/52.311045.

Frank, E, and I Witten. 1998. "Generating Accurate Rule Sets Without Global Optimization." *Proceedings of the Fifteenth International Conference on Machine Learning*, 144–51. http://hdl.handle.net/10289/1047.

Freund, Y, and Re Schapire. 1995. "A Desicion-Theoretic Generalization of on-Line Learning and an Application to Boosting." In *Computational Learning Theory*, 55:119–39. doi:10.1006/jcss.1997.1504.

Freund, Yoav, and Re Robert E Schapire. 1996. "Experiments with a New Boosting Algorithm." In *International Conference on Machine Learning*, 148–56. doi:10.1.1.133.1040.

Fuentes, José, Anabel Fraga, Gonzalo Génova, Eugenio Parra, José María Álvarez, and Juan Llorens. 2016. "Applying INCOSE Rules for Writing High-Quality Requirements in Industry." In *26th Annual INCOSE International Symposium (IS 2016)*.

Gala, Nuria Pavia. 1999. "Using the Incremental Finite-State Architecture to Create a Spanish Shallow Parser." In *Proceedings of XV Congress of SEPLN*.

Génova, Gonzalo, José M. Fuentes, Juan Llorens, Omar Hurtado, and Valentín Moreno. 2013. "A Framework to Measure and Improve the Quality of Textual

Requirements." *Requirements Engineering* 18 (1): 25–41. doi:10.1007/s00766-011-0134-z.

Glass, Robert L. 2002. *Facts and Fallacies of Software Engineering*. *October*. Boston: Addison-Wesley Professional. doi:http://dx.doi.org/10.1109/FOSE.2007.29.

Gorschek, Tony, and Claes Wohlin. 2006. "Requirements Abstraction Model." *Requirements Engineering* 11 (1): 79–101. doi:10.1007/s00766-005-0020-7.

Hall, Mark, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H Witten. 2009. "The WEKA Data Mining Software." *ACM SIGKDD Explorations* 11 (1): 10–18. doi:10.1145/1656274.1656278.

Harman, M., and B. F. Jones. 2001. "Search-Based Software Engineering." *Information and Software Technology* 43 (14): 833–39. doi:10.1016/S0950-5849(01)00189-6.

Harman, Mark, Phil McMinn, Jerffeson Teixeira De Souza, and Shin Yoo. 2012. "Search Based Software Engineering: Techniques, Taxonomy, Tutorial." *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 7007: 1–59. doi:10.1007/978-3-642-25231-0.

Hong, Jiarong, Igor Mozetic, and Ryszard S. Michalski. 1986. "AQ15: Incremental Learning of Attribute-Based Descriptions from Examples: The Method and User's Guide." *Department of Computer Science, University of Illinois at Urbana-Champaign.*

Hooks, Ivy F. 2000. *GUIDE FOR MANAGING AND WRITING REQUIREMENTS*. Compliance Automation.

Hopcroft, John, and Jeffrey Ullman. 1979. "Introduction to Automata Theory and Computation." https://scholar.google.ch/scholar?q=Automata+Theory+hopcroft+1979&btnG=&hl=en&as_sdt=0,5#0.

Hull, Elizabeth, Ken Jackson, and Jeremy Dick. n.d. "Www.requirementsengineering.info."

———. 2010. *Requirements Engineering*. Springer Science & Business Media.

Hussain, Ishrar, Olga Ormandjieva, and Leila Kosseim. 2007. "Automatic Quality Assessment of SRS Text by Means of a Decision-Tree-Based Text Classifier." *Proceedings - International Conference on Quality Software*, no. Qsic: 209–18. doi:10.1109/QSIC.2007.4385497.

IBM. 2015. "Rational DOORS. Rational RequisitePro." *IBM*. Accessed May 1. www.IBM.com.

Ieee830. 1998. *IEEE Recommended Practice for Software Requirements Specifications*. *Practice*. Vol. 1998.

"INCOSE (International Council on Systems Engineering)." 2016. Accessed September 4. http://www.incose.org/.

International Council on Systems Engineering. 2011. “Systems Engineering Handbook v3.2.2 - a Guide for System Life Cycle Processes and Activities,” no. October: 376.

Iso/Iec. 2010. “Software Product Quality Requirements and Evaluation (SQuaRE).” *Quality Requirements, Int’l Organization for Standardization* ISO/IEC 25. http://www.iso.org/iso/catalogue_detail?csnumber=35683.

Iso/Iec/Ieee-29148. 2011. “ISO/IEC 29148 FDIS Systems and Software Engineering—Life Cycle Processes—Requirements Engineering.”

ISO/IEC-15288. 2008. “Systems and Software Engineering – System Life Cycle Processes.”

Jarzabek, Stan. 1993. “Domain Model-Driven Software Reengineering and Maintenance.” *The Journal of Systems and Software* 20 (1): 37–51. doi:10.1016/0164-1212(93)90047-2.

Kaplan, Ronald M, and Martin Kay. 1994. “Regular Models of Phonological Rule Systems.” *Computational Linguistics* 20: 331–78. http://portal.acm.org/citation.cfm?id=204917&dl=GUIDE.

Ketabchi, Shokoofeh, Navid Karimi Sani, and Kecheng Liu. 2011. “A Norm-Based Approach towards Requirements Patterns.” *2011 IEEE 35th Annual Computer Software and Applications Conference*, July. Ieee, 590–95. doi:10.1109/COMPSAC.2011.82.

Kiyavitskaya, Nadzeya, Nicola Zeni, Luisa Mich, and Daniel M. Berry. 2008. “Requirements for Tools for Ambiguity Identification and Measurement in Natural Language Requirements Specifications.” *Requirements Engineering* 13 (3): 207–39. doi:10.1007/s00766-008-0063-7.

Ko, Youngjoong, Sooyong Park, Jungyun Seo, and Soonhwang Choi. 2007. “Using Classification Techniques for Informal Requirements in the Requirements Analysis-Supporting System.” *Information and Software Technology* 49 (11–12): 1128–40. doi:10.1016/j.infsof.2006.11.007.

Koeling, Rob. 2000. “Chunking with Maximum Entropy Models POS-3 POS-3 / POSo TAG-2 TAG _ I POS-2 / POS-1 / POSo / POS + I,” 139–41.

Koskenniemi, Kimmo. 1983. “A General Computational Model for Word-Form Recognition and Production,” 178–81.

Krogh, G., Sebastian Spaeth, and Stefan Haefliger. 2005. “Knowledge Reuse in Open Source Software: An Exploratory Study of 15 Open Source Projects.” *Proceedings of the Proceedings of the 38th Annual Hawaii International Conference on System Sciences-Volume 07* 0 (100012): 198–2. doi:10.1109/HICSS.2005.378.

Kudo, Taku, and Yuji Matsumoto. 2001. “Chunking with Support Vector Machines.” *Proceedings of NAACL 2001* 816: 1–8. doi:http://dx.doi.org/10.3115/1073336.1073361.

London, College, Yuanyuan Zhang, S. Afshin Mansouri, Mark Harman, and S. Afshin Mansouri. 2007. “The Multi-Objective next Release Problem.” *Proceedings of the 9th Annual Conference on Genetic and Evolutionary Computation - GECCO ’07*, no. May 2016: 1129. doi:10.1145/1276958.1277179.

Loucopoulos, P., and Vassilios Karakostas. 1995. *System Requirements Engineering*. McGraw-Hill, Inc.

Magee, Stan, and Leonard L Tripp. 1997. *Guide to Software Engineering Standards and Specifications*. Artech House, Inc.

Major, John a., and John J. Mangano. 1995. “Selecting among Rules Induced from a Hurricane Database.” *Journal of Intelligent Information Systems* 4 (1): 39–52. doi:10.1007/BF00962821.

Marti, M.a., and J. Llisterri. 2002. “Tratamiento Del Lenguaje Natural. Tecnología de La Lengua Oral Y Escrita.” In *Universitat de Barcelona*. https://books.google.com.co/books?id=em69wKZi3pUC&printsec=frontcover&dq=lenguaje+natural&hl=es-419&sa=X&ei=ozlVVfHTDMyhNvr4gPAD&sqi=2&ved=0CBoQ6AEwAA#v=onepage&q=lenguaje natural&f=true.

Martin, James. 1984. *An Information Systems Manifesto*. Prentice H. doi:http://doi.acm.org/10.1145/3166.214926.

Martin, James N. 1996. *Systems Engineering Guidebook: A Process for Developing Systems and Products*. CRC Press, Inc.: Boca Raton.

Mat Jani, H., and a.B.M. Tariqul Islam. 2012. “A Framework of Software Requirements Quality Analysis System Using Case-Based Reasoning and Neural Network.” *Ieeexplore.Ieee.Org*, 152–57. http://ieeexplore.ieee.org/ielx7/6520998/6528391/06528619.pdf?tp=&arnumber=6528619&isnumber=6528391\nhttp://ieeexplore.ieee.org/xpl/articleDetails.jsp?tp=&arnumber=6528619&sortType=desc_p_Citation_Count&matchBoolean=true&pageNumber=2&rowsPerPa.

McCoy, J. R. 2001. “NASA Software Tools for High-Quality Requirements Engineering.” *26th Annual NASA Goddard Software Engineering Workshop, IEEE/NASA SEW 2001*, 69. doi:10.1109/SEW.2001.992657.

Molina, Antonio, and Ferran Pla. 2002. “Shallow Parsing Using Specialized Hmms.” *J. Mach. Learn. Res.* 2: 595–613. http://dl.acm.org/citation.cfm?id=944790.944819.

Moreno, Valentín. 2010. “Análisis de Los Criterios de Relevancia Documental Mediante Consultas de Información En El Entorno Web.” University Carlos III de Madrid. http://e-archivo.uc3m.es:8080/handle/10016/9727.

Moreno, Valentín, Gonzalo Génova, Eugenio Parra, and Anabel Fraga. n.d. “Application of Machine Learning Techniques to the Flexible Assessment and Improvement of Requirements Quality.”

Moreno, Valentín, Jorge Morato, and Sonia Sánchez-Cuadrado. 2009. Procedimiento y sistema de estimación de la posición de un recurso. PCT/ES2009/070517, issued 2009.

Moreno, Valentín, Pablo Miguel Suárez, Anabel Fraga, Juan Llorens, and Eugenio Parra. 2013. Método de generación de patrones semánticos. PCT/ES2013/070638, issued 2013.

Moros, Begoña, Cristina Vicente-Chicote, and Ambrosio Toval. 2008a. "Metamodeling Variability to Enable Requirements Reuse." *CEUR Workshop Proceedings* 337: 140–54. doi:10.1007/978-3-540-87877-3-46.

———. 2008b. "Remm-Studio+: Modeling Variability to Enable Requirements Reuse." In *International Conference on Conceptual Modeling*, 530–31.

Neighbors, James M. 1984. "The Draco Approach to Constructing Software from Reusable Components." *IEEE Transactions on Software Engineering* SE-10 (5): 564–74. doi:10.1109/TSE.1984.5010280.

Ning, J.Q., a. Engberts, and W. Kozaczynski. 1993. "Recovering Reusable Components from Legacy Systems by Program Segmentation." *[1993] Proceedings Working Conference on Reverse Engineering*, 64–72. doi:10.1109/WCRE.1993.287778.

Otero, Carlos E., Erica Dell, Abrar Qureshi, and Luis D. Otero. 2010. "A Quality-Based Requirement Prioritization Framework Using Binary Inputs." *AMS2010: Asia Modelling Symposium 2010 - 4th International Conference on Mathematical Modelling and Computer Simulation*, 187–92. doi:10.1109/AMS.2010.48.

Ott, Daniel. 2013. "Automatic Requirement Categorization of Large Natural Language Specifications at Mercedes-Benz for Review Improvements." *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 7830 LNCS: 50–64. doi:10.1007/978-3-642-37422-7_4.

Palmer, David D., and Marti a. Hearst. 1997. "Adaptive Multilingual Sentence Boundary Disambiguation." *Computational Linguistics* 23: 241–67. doi:10.1.1.57.1897.

Parra, Eugenio, Christos Dimou, Juan Llorens, Valentín Moreno, and Anabel Fraga. 2015. "A Methodology for the Classification of Quality of Requirements Using Machine Learning Techniques." *Information and Software Technology* 67. Elsevier B.V.: 180–95. doi:10.1016/j.infsof.2015.07.006.

Poesio, Massimo. 2000. "Semantic Analysis." In *Handbook of Natural Language Processing*, 93–122. New York: Marcel Dekker.

Popescu, Daniel, Spencer Rugaber, Nenad Medvidovic, and Daniel M. Berry. 2008. "Reducing Ambiguities in Requirements Specifications via Automatically Created Object-Oriented Models." *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 5320 LNCS: 103–24. doi:10.1007/978-3-540-89778-1_10.

Quinlan, J. 1993. *C4.5: Programs for Machine Learning*. Elsevier.

Quinlan, J. R. 1986. “Induction of Decision Trees.” *Machine Learning* 1 (1): 81–106. doi:10.1023/A:1022643204877.

Racheva, Zornitza, Maya Daneva, and Andrea Herrmann. 2010. “A Conceptual Model of Client-Driven Agile Requirements Prioritization: Results of a Case Study.” *Proceedings of the 2010 {ACM-IEEE} International Symposium on Empirical Software Engineering and Measurement*, 39:1–39:4. doi:10.1145/1852786.1852837.

Rehberg, C., P. 2012. Automatic pattern generation in natural language processing. US 2010/0010800 A1, issued 2012.

Reqtify. 2005. “3DS DassaultSystèmes.” *3DS DassaultSystèmes*. Accessed May 20. www.3ds.com/.

Reynar, Jeffrey C., and Adwait Ratnaparkhi. 1997. “A Maximum Entropy Approach to Identifying Sentence Boundaries.” *Proceedings of the Fifth Conference on Applied Natural Language Processing*, 16--19. doi:10.3115/974557.974561.

Riley, Michael D. 1989. “Some Applications of Tree-Based Modelling to Speech and Language.” *Proceedings of the Workshop on Speech and Natural Language (HLT)*, no. 2: 339–52. doi:10.3115/1075434.1075492.

Riloff, E. 1996. “An Empirical Study of Automated Dictionary Construction for Information Extraction in Three Domains.” *Artificial Intelligence* 84: 356. doi:10.1016/0004-3702(96)81367-1.

Riloff, Ellen. 1996. “Automatically Generating Extraction Patterns from Untagged Text.” *Proceedings Of The National Conference On Artificial Intelligence* 2 (Cardie 1993): 1044–49. http://www.aaai.org/Papers/AAAI/1996/AAAI96-155.pdf.

Rosenberg, Linda H., and H. Linda. 2001. “Generating High Quality Requirements.” In , 4524:28–30. Proceedings of the AIAA space 2001 conference and exposition, AIAA paper. http://arc.aiaa.org/doi/pdf/10.2514/6.2001-4524.

Russell, Petitpierre Dominique and Graham. 1995. “Mmorph-the Multext Morphology Program.”

Salah, Ait-Mokhtar, and Jean-Pierre Chanod. 1997. “Incremental Finite-State Parsing.” In *Proceedings of the Fifth Conference on Applied Natural Language Processing*, 72–79. Association for Computational Linguistics.

Sardinha, Alberto, Ruzanna Chitchyan, Nathan Weston, Phil Greenwood, and Awais Rashid. 2013. “EA-Analyzer: Automating Conflict Detection in a Large Set of Textual Aspect-Oriented Requirements.” *Automated Software Engineering* 20 (1): 111–35. doi:10.1007/s10515-012-0106-7.

Sawyer, P., and G. Kontonya. 2001. *Software Requirements*. Edited by Pearson Education. *Guide to the Software Engineering Body of Knowledge - SWEBOK*.

http://proquest.safaribooksonline.com/9780735679658.

Schapire, Robert E. 1990. “The Strength of Weak Learnability.” *Machine Learning* 5 (2): 197–227. doi:10.1023/A:1022648800760.

Shamieh, Cathleen. 2011. *Systems Engineering for Dummies*. *Systems Engineering*.

Solutions, Visure. 2015. “Visure Solutions.” *Visure Solutions*. Accessed May 1. www.visuresolutions.com/.

Sproat, Richard. 2000. *Lexical Analysis*. Marcel Dekker.

Thakurta, Rahul. 2013. “A Framework for Prioritization of Quality Requirements for Inclusion in a Software Project.” *Software Quality Journal* 21 (4): 573–97. doi:10.1007/s11219-012-9188-5.

Think Big, Act Small. 2013. “2 Chaos Manifesto 2013.” *Chaos Manifesto*. http://www.versionone.com/assets/img/files/ChaosManifesto2013.pdf.

Tjong, Sri, Nasreddine Hallam, and Michael Hartley. 2006. “Improving the Quality of Natural Language Requirements Specifications through Natural Language Requirements Patterns.” *The Sixth IEEE International Conference on Computer and Information Technology (CIT’06)*. Ieee, 199–199. doi:10.1109/CIT.2006.103.

Toval, Ambrosio, Begona Moros, Joaquin Nicolas, and Joaquin Lasheras. 2008. “Eight Key Issues for an Effective Reuse-Based Requirements Process.” *Computer Systems Science and Engineering* 23 (6): 373–85.

Toval, Ambrosio, Joaquín Nicolás, Begoña Moros, and Fernando García. 2002. “Requirements Reuse for Improving Information Systems Security: A Practitioner’s Approach.” *Requirements Engineering* 6 (4): 205–19. doi:10.1007/PL00010360.

Turk, Wayne. 2006. “Writing Req Quirements Equirements,” no. July: 20–23.

Veenstra, Jorn, and Antal Van Den Bosch. 2000. “Single-Classifier Memory-Based Phrase Chunking.” *Proceedings of CoNLL-2000 and LLL-2000* 5 (1): 5–7. doi:http://dx.doi.org/10.3115/1117601.1117640.

Vicente-Chicote, Cristina, Begona Moros, and Ambrosio Toval. 2007. “REMM-Studio : An Integrated Model- Driven Environment for Requirements.” *October* 6 (9): 437–54.

Wang, Yue, Irene L. Manotas Gutièrrez, Kristina Winbladh, and Hui Fang. 2013. “Automatic Detection of Ambiguous Terminology for Software Requirements.” In *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 7934 LNCS:25–37. Springer. doi:10.1007/978-3-642-38824-8_3.

Weiss, S, and N Indurkhya. 1997. *Predictive Data Mining*. Vol. 4. Morgan Kaufmann.

Wilson, William M., Linda H. Rosenberg, and Lawrence E. Hyatt. 1997. “Automated

Analysis of Requirement Specifications." *Proceedings - International Conference on Software Engineering*, 161–71. doi:10.1109/ICSE.1997.610237.

Wolpert, D H. 1992. "Stacked Generalization." *Neural Networks* 5 (2): 241–59. doi:10.1016/S0893-6080(05)80023-1.

ANEXOS

Anexo 1: Descripción de las métricas de calidad

El siguiente anexo muestra una descripción de las métricas proporcionadas por la herramienta Requirements Quality Analyzer (RQA) (Company 2015b) y que han sido utilizadas en los experimentos realizados en el proceso de *Clasificación de requisitos*.

Se han omitido aquellas métricas que no aportaban valor en ninguno de los requisitos del conjunto utilizado en los experimentos ya que no tiene efecto en la generación de los clasificadores.

Tabla. 11 Descripción de métricas de calidad

Morphological	
Paragraphs	El requisito no se debe expresar con demasiados párrafos para evitar sobre especificación, redundancia de información y expresar varias necesidades en un mismo requisito.
Words	El requisito no debe tener un elevado número de palabras para evitar los problemas similares relacionados con la longitud.
Readability	La legibilidad mede el grado de dificultad en la lectura del texto, se calcula como la media aritmética de sílabas/número de caracteres por palabras más el número de palabras por frase. Una baja legibilidad puede ocasionar confusión.
Punctuation	Medida como el número de caracteres entre signos de puntuación. Measured as the number of characters between punctuation marks. Una Puntuación incorrecta dificulta la legibilidad.
Lexical	
Connectors	Número de conectores disyuntivos copulativos. El uso de múltiples conectores puede indicar diferentes necesidades en un mismo requisito y por tanto comprometer la atomicidad.
Negative	Expresiones negativas puede provocar que el requisito sea difícil de entender.
Control-flow	El requisito debe evitar seudocódigo y expresiones de control de

	flujo para evitar la especificación de soluciones al problema.
Implicit	El requisito debe ser explícito, se debe evitar el uso de pronombres personales.
Ambiguous	Usar expresiones ambiguas dificulta la comprensión del requisito.
Incomplete	Frase con enumeraciones incompletas (‘etc’, ‘como mínimo’, ‘no limitado a’, etc.) demuestra que el requisito no plantea claramente el objetivo o no es atómico.
Speculative	El uso se expresiones especulativas (‘bastantes’, ‘suficiente’, ‘aproximadamente’, etc.) indica que la verdadera necesidad no está clara.
Subjective	Se debe evitar expresar el punto de vista del autor en el requisito.
Rationale	Se debe evitar expresar justificaciones en el requisito.
Design	El requisito se debe expresar las necesidades y evitar especificar las soluciones.
Analytical	
Imperative	El requisito debe tener al menos un verbo en modo imperativo.
Conditional	El requisito debe ser escrito en modo asertivo.
Passive voice	Usar verbos en voz pasiva dificulta la comprensión del requisito.
Domain concepts	Un número elevado de conceptos del dominio en un mismo requisito puede indicar sobre especificación.
Domain verbs	Usar demasiados verbos del dominio puede indicar que el requisito expresa demasiadas necesidades.
Patterns	Indica si el requisito es representado mediante patrones sintáctico-semánticos.

Anexo 2: Valores de métricas

En este anexo se muestran los valores de las métricas obtenidas con la herramienta Requirements Quality Analyzer (RQA) (Company 2015b) del conjunto de 1035 requisitos proporcionado por el Grupo de Trabajo de Requisitos INCOSE ("INCOSE (International Council on Systems Engineering)" 2016)

Tabla. 12 Valores de métricas

																				Patterns Count ('No')	938
																				Patterns Count ('Yes')	97
Count 0		**0**	**0**	**0**	**0**	**554**	**939**	**920**	**839**	**782**	**729**	**958**	**1022**	**987**	**174**	**831**	**760**	**285**	**447**		
Min		**1**	**6**	**2**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**No**	
Max		**19**	**230**	**32**	**158,67**	**13**	**2**	**4**	**5**	**4**	**22**	**2**	**1**	**5**	**6**	**2**	**7**	**28**	**21**	**Yes**	

Req	Quality	Para	Words	Read	Punct	Conn	Neg	Flow	Impl	Amb	Inc	Spec	Rat	Des	Imp	Cond	Pass	DomC	DomV	Pat
1	Good	1	10	19	39	0	1	0	0	0	0	0	0	1	1	0	0	1	0	No
2	Good	1	29	15	38,75	2	1	0	1	0	0	0	0	0	2	0	0	2	1	No
3	Good	1	26	14	48	2	1	2	2	0	0	0	0	0	1	0	1	2	2	No
4	Good	1	17	15	38,67	1	0	0	0	0	0	0	0	0	0	1	0	2	1	No
5	Good	1	15	15	48,50	1	0	0	0	0	0	0	0	1	1	0	0	1	1	No
6	Good	1	21	14	60	1	0	0	0	1	0	0	0	0	1	0	0	2	3	No
7	Good	1	15	14	44,50	0	0	0	0	0	0	0	0	0	1	0	0	1	1	No
8	Good	1	19	9	30,67	0	0	0	1	0	0	1	0	0	1	1	1	0	2	No
9	Good	1	23	11	36,33	0	0	1	1	0	0	1	0	0	1	1	1	0	2	No
10	Good	1	13	12	27	0	0	1	0	0	0	0	0	0	1	0	1	0	0	No

11	Good	1	13	11	27	0	0	1	0	0	0	0	0	0	1	0	1	0	0	No
12	Good	1	12	14	37	1	0	0	0	0	0	0	0	0	1	0	0	2	0	No
13	Good	1	17	13	15	1	0	1	0	0	2	0	0	0	1	0	2	0	1	No
14	Good	1	19	14	29,25	2	0	1	0	0	0	0	0	0	1	0	2	0	1	No
15	Good	1	11	16	39,50	0	1	0	0	0	0	0	0	0	1	0	0	0	0	No
16	Good	1	16	12	24	1	0	1	0	0	0	0	0	0	1	0	2	0	1	No
17	Good	1	19	13	28,50	2	0	1	0	0	0	0	0	0	1	0	2	0	1	No
18	Good	1	12	12	39	0	1	0	0	0	0	0	0	1	1	0	0	0	0	No
19	Good	1	15	14	48,50	1	1	0	0	0	0	0	0	1	1	0	0	0	0	No
20	Good	1	12	9	30,50	0	0	0	0	0	0	0	0	0	1	0	0	1	2	No
21	Good	1	12	9	31	0	0	0	0	0	0	0	0	0	1	0	0	1	2	No
22	Good	1	12	8	31	0	0	0	0	0	0	0	0	0	1	0	0	1	2	No
23	Good	1	13	9	34,50	0	0	0	0	0	0	0	0	0	1	0	0	1	2	No
24	Good	1	13	10	35	0	0	0	0	0	0	0	0	0	1	0	0	2	2	No
25	Good	1	17	14	50,50	0	0	0	0	1	0	0	0	1	1	0	0	1	3	No
26	Good	1	18	15	55,50	0	0	0	0	1	0	0	0	1	0	1	0	1	3	No
27	Good	1	20	15	60,50	1	0	0	1	1	0	0	0	0	0	1	1	2	2	No
28	Good	1	17	13	47	0	0	0	1	1	0	0	0	0	0	1	1	1	3	No
29	Good	1	11	9	34	0	0	0	0	0	0	0	0	0	1	0	0	1	0	No
30	Good	1	11	9	33,50	0	0	0	0	0	0	0	0	0	1	0	0	1	0	No
31	Good	1	11	9	33,50	0	0	0	0	0	0	0	0	0	1	0	0	1	0	No
32	Good	1	14	9	39	0	0	0	0	0	0	0	0	0	1	0	0	2	0	No
33	Good	1	24	10	64	2	0	0	0	1	0	0	0	0	1	0	0	2	3	No
34	Good	1	15	11	40,50	0	0	0	1	1	0	0	0	0	1	0	1	1	1	No
35	Good	1	23	7	33,33	1	0	0	1	0	0	0	0	0	1	0	0	2	3	No
36	Good	1	16	11	49,50	1	0	0	1	1	0	0	0	0	1	0	0	1	1	No
37	Good	1	12	9	34,50	1	0	0	0	1	0	0	0	0	0	1	0	1	1	No
38	Good	1	12	11	35,50	1	0	0	0	0	0	0	0	0	1	0	0	3	1	No

39	Good	1	15	8	41,50	1	0	0	0	0	0	0	0	0	1	0	0	2	2	No
40	Good	1	17	5	43	1	0	0	1	0	0	0	0	0	1	0	0	0	0	No
41	Good	1	12	13	37	0	0	0	0	1	0	0	0	0	1	0	0	2	1	No
42	Good	1	26	11	65	1	0	1	0	0	0	0	0	0	1	0	0	0	2	No
43	Good	1	13	12	36	0	0	1	0	0	0	0	0	0	1	0	0	1	0	No
44	Good	1	12	11	32	0	0	0	0	0	0	0	0	0	1	0	0	1	1	No
45	Good	1	19	7	43,50	1	1	0	0	0	0	0	0	0	1	0	1	3	0	No
46	Good	1	24	11	19,13	1	0	0	0	0	3	0	0	0	2	0	0	4	2	Yes
47	Good	1	29	13	78,50	1	0	0	0	1	0	0	0	0	1	0	0	3	3	Yes
48	Good	1	16	13	51	1	0	0	0	0	0	0	0	0	1	0	0	2	2	No
49	Good	1	31	18	66	1	0	0	0	0	0	0	0	1	0	0	0	9	3	No
50	Good	1	16	13	51,50	0	0	0	0	0	0	0	0	0	1	0	0	3	1	No
51	Good	1	15	8	39,50	1	0	0	0	0	0	0	0	0	1	0	0	2	1	No
52	Good	16	83	12	18,27	1	0	0	2	1	4	0	0	2	2	0	1	22	7	No
53	Good	1	18	12	49,50	0	0	0	0	0	0	0	0	0	1	0	0	1	1	No
54	Good	6	49	13	50,71	3	0	0	0	0	0	0	0	5	1	0	0	5	4	No
55	Good	1	13	8	14,20	0	0	0	0	0	1	0	0	0	1	0	0	3	1	No
56	Good	1	16	13	50,50	2	0	0	0	1	0	0	0	0	0	1	0	3	2	No
57	Good	1	19	12	13,13	1	0	0	1	2	3	0	0	0	0	1	0	3	0	No
58	Good	9	68	10	20,77	4	0	1	2	2	7	0	0	0	2	0	1	12	3	Yes
59	Good	1	24	11	40,75	0	0	0	0	0	0	0	0	0	1	1	0	1	1	No
60	Good	1	27	9	16,30	0	0	0	0	0	3	0	0	0	0	1	0	6	2	Yes
61	Good	1	23	11	12,50	1	0	0	0	1	6	0	0	0	1	0	0	1	1	No
62	Good	1	34	12	52,25	2	0	0	1	2	0	0	0	0	3	0	0	5	0	No
63	Good	1	50	11	26,25	2	0	1	0	4	3	0	0	0	1	0	0	5	5	Yes
64	Good	1	21	10	53	0	0	0	0	0	0	0	0	0	1	0	1	2	1	No
65	Good	1	19	10	47,50	0	0	1	0	0	1	0	0	0	1	0	0	2	1	No
66	Good	1	14	13	31,67	0	0	0	0	0	0	0	0	0	1	0	0	1	1	No

67	Good	1	16	12	50	1	0	0	0	1	0	0	0	0	1	0	0	0	0	No
68	Good	1	12	8	33,50	0	1	0	0	0	0	0	0	0	1	0	0	1	1	No
69	Good	1	43	14	38	1	0	0	1	0	2	2	0	0	1	2	2	2	4	Yes
70	Good	1	18	10	47,50	0	0	0	0	0	0	0	0	0	1	0	0	1	1	No
71	Good	1	19	10	49,50	0	1	0	0	0	0	0	0	0	1	0	0	2	2	No
72	Good	13	93	8	28,33	6	0	0	0	2	7	2	0	0	2	2	1	18	2	Yes
73	Good	1	38	13	59,25	1	0	2	0	1	0	1	0	0	1	1	1	3	2	No
74	Good	1	17	6	41	0	0	0	0	0	0	0	0	0	1	0	0	2	0	Yes
75	Good	1	24	12	42,33	1	0	0	0	1	0	0	0	0	1	0	0	3	1	Yes
76	Good	1	24	14	49,67	2	0	0	0	0	0	0	0	0	1	0	0	2	1	No
77	Good	1	13	9	26	1	0	0	0	0	0	0	0	0	1	0	0	3	1	No
78	Good	1	24	12	64,50	1	0	0	0	0	0	0	0	0	1	0	0	3	2	No
79	Good	1	23	13	66,50	2	0	1	0	0	0	0	0	0	1	0	0	1	1	No
80	Good	5	43	12	25,82	0	1	1	0	1	3	0	0	0	1	0	0	6	2	No
81	Good	1	23	15	72	1	0	0	0	0	0	0	0	0	1	0	0	5	1	No
82	Good	1	24	7	33,25	1	0	0	0	0	0	0	0	0	2	0	0	4	2	No
83	Good	1	18	13	52	0	0	0	0	1	0	0	0	0	1	0	0	1	1	No
84	Good	1	35	16	33,83	2	0	0	0	0	2	0	0	0	1	0	0	5	1	No
85	Good	1	24	11	35	1	0	0	0	1	0	0	0	0	2	0	2	2	1	No
86	Good	1	27	15	53,33	1	0	0	0	0	2	0	0	0	1	0	0	2	1	No
87	Good	1	32	16	20,56	2	0	0	0	1	2	0	0	0	1	0	0	6	2	No
88	Good	1	26	14	49,67	1	0	0	0	0	0	0	0	0	1	0	0	1	1	No
89	Good	1	43	11	75,33	2	1	1	1	0	0	1	0	0	2	1	1	2	2	No
90	Good	1	60	12	39,11	3	1	2	1	0	4	0	0	0	4	0	3	3	4	Yes
91	Good	1	37	11	23,56	1	0	0	0	0	2	0	0	0	2	0	0	3	1	No
92	Good	1	31	10	12,86	0	0	0	1	0	6	1	0	0	1	1	1	5	2	No
93	Good	1	18	10	48	1	0	0	0	0	0	0	0	0	1	0	1	1	0	Yes
94	Good	1	44	10	64,50	0	0	0	2	0	0	1	0	0	1	1	1	5	2	No

95	Good	1	43	12	25,50	3	0	0	0	0	6	0	0	0	1	0	1	2	3	No
96	Good	1	18	12	27	0	0	0	0	0	2	0	0	0	1	0	0	1	0	No
97	Good	1	30	10	47,75	1	0	0	1	0	0	1	0	0	1	1	1	1	1	No
98	Good	1	20	12	55	0	0	0	1	0	0	0	0	0	1	0	0	2	0	No
99	Good	1	12	11	24	0	0	0	0	1	2	0	0	0	1	0	0	1	2	Yes
100	Good	1	16	13	34,33	1	0	0	0	0	0	0	0	0	1	0	0	3	2	No
101	Good	1	24	12	38,75	1	0	0	1	0	0	0	0	1	2	0	0	3	1	No
102	Good	1	28	10	26	1	0	0	1	0	2	0	0	0	2	0	1	2	3	No
103	Good	1	25	19	17,30	2	0	0	0	1	3	0	0	0	1	0	0	4	2	No
104	Good	14	79	14	22,38	2	0	0	1	1	14	0	0	0	1	0	1	19	2	Yes
105	Good	6	43	11	36,63	2	0	0	0	1	2	0	0	0	1	0	0	10	0	No
106	Good	1	17	17	14,50	0	1	0	0	0	1	0	0	0	1	0	0	2	0	No
107	Good	1	25	13	20,63	2	0	0	0	0	3	0	0	0	1	0	0	5	1	Yes
108	Good	1	26	12	21,83	2	1	0	0	0	2	0	0	0	1	0	0	3	0	No
109	Good	1	22	14	35	2	0	0	0	0	2	0	0	0	1	0	0	2	2	Yes
110	Good	1	17	9	44,50	1	0	0	0	0	0	0	0	0	1	0	0	3	0	No
111	Good	1	23	10	35,67	1	1	1	1	0	0	2	0	0	0	1	1	1	1	Yes
112	Good	1	26	15	21,43	1	0	0	0	0	4	0	0	0	1	0	0	3	0	No
113	Good	1	36	19	26,63	0	0	0	0	1	0	0	0	0	2	0	0	3	1	Yes
114	Good	1	32	17	30,83	1	0	0	0	1	3	1	0	0	1	1	0	3	2	Yes
115	Good	1	9	10	26,50	0	1	0	0	1	0	0	0	0	0	0	0	1	1	No
116	Good	1	14	6	39,50	1	0	0	0	0	0	0	0	0	1	0	0	1	0	No
117	Good	1	22	12	22,50	2	0	0	0	0	3	0	0	0	1	0	0	2	0	No
118	Good	1	34	16	23,75	4	0	0	0	0	0	0	0	0	1	0	0	2	2	No
119	Good	1	33	17	49	1	0	0	0	1	2	0	0	0	2	0	0	7	2	No
120	Good	1	18	12	26,50	0	0	0	0	0	2	0	0	0	1	0	0	3	1	No
121	Good	1	17	9	44,50	1	0	0	0	2	0	0	0	0	1	0	0	5	1	Yes
122	Good	1	13	11	36,50	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No

123	Good	1	25	13	23,67	1	0	0	0	0	3	0	0	0	1	0	0	4	1	No
124	Good	1	17	10	49	0	0	0	0	1	0	0	0	0	1	0	0	2	0	No
125	Good	4	36	11	17,15	2	0	0	0	0	5	0	0	0	2	0	0	2	0	No
126	Good	1	36	11	51,50	2	0	0	0	1	0	0	0	0	2	0	1	5	1	No
127	Good	1	36	11	51,50	2	0	0	0	1	0	0	0	0	2	0	1	5	1	No
128	Good	1	17	8	44	1	0	0	0	1	0	0	0	0	1	0	0	3	3	Yes
129	Good	1	16	10	44,50	0	0	0	0	0	0	0	0	0	0	1	0	3	1	No
130	Good	4	45	12	41,14	3	1	0	0	1	1	0	1	0	0	1	1	10	0	No
131	Good	1	23	11	22,60	0	0	1	0	0	4	0	0	0	1	0	0	2	1	Yes
132	Good	2	28	13	31,60	0	0	0	0	0	2	0	0	0	1	0	0	2	0	No
133	Good	1	22	10	56,50	1	0	1	0	0	0	0	1	0	1	0	1	4	1	Yes
134	Good	1	18	14	38,33	0	0	1	0	0	0	0	0	0	2	0	1	3	1	No
135	Good	1	17	10	23,50	0	0	0	0	0	2	0	0	0	1	0	0	2	3	Yes
136	Good	1	14	8	37	0	0	0	0	1	0	0	0	0	1	0	0	3	1	No
137	Good	1	28	15	79	1	0	1	0	0	0	0	0	0	1	0	1	3	1	Yes
138	Good	1	26	13	45,67	0	0	0	0	1	2	1	0	0	1	1	0	3	2	Yes
139	Good	1	23	12	33	1	0	0	0	0	0	0	0	0	0	1	0	6	1	No
140	Good	1	53	32	36,64	3	0	1	0	1	4	0	0	1	1	0	0	14	0	No
141	Good	5	33	7	23,50	0	0	0	0	0	2	0	0	0	1	0	1	3	1	No
142	Good	1	20	15	26	3	0	0	0	2	2	0	0	0	1	0	0	3	1	No
143	Good	1	15	11	20,25	1	0	0	0	1	2	0	0	0	1	0	0	2	1	No
144	Good	1	12	6	21,33	1	0	0	0	1	2	0	0	0	1	0	0	1	1	No
145	Good	1	27	13	38	2	0	0	0	1	2	0	0	0	1	0	0	6	1	No
146	Good	1	35	19	24,22	2	1	0	0	0	3	0	0	0	2	0	0	7	3	No
147	Good	1	31	13	33,33	0	0	0	1	1	0	2	0	0	1	2	0	3	2	No
148	Good	1	27	6	27	0	1	1	1	0	2	0	0	0	2	0	1	3	1	No
149	Good	1	50	24	29,10	1	1	0	0	1	5	1	1	0	1	1	0	5	5	No
150	Good	1	20	11	56,50	0	0	0	0	0	0	0	0	0	1	0	0	1	0	No

151	Good	5	30	10	35,17	1	0	0	0	2	0	0	0	0	1	0	0	4	1	No
152	Good	1	20	13	40,67	0	0	0	0	0	0	0	0	0	1	0	0	2	0	No
153	Good	1	25	11	50,67	1	0	0	0	3	0	0	0	0	2	0	0	4	0	No
154	Good	4	34	8	34	0	0	0	0	0	1	0	0	0	1	0	0	8	0	No
155	Good	1	17	6	24	0	0	0	0	1	2	0	0	0	1	0	0	2	1	No
156	Good	4	38	9	22,30	0	0	0	1	1	2	0	0	0	2	0	1	5	2	Yes
157	Good	1	18	12	51	0	0	0	0	0	0	0	0	0	1	0	0	3	0	No
158	Good	1	52	10	41,14	2	0	0	1	0	1	0	0	0	2	1	0	5	0	Yes
159	Good	1	29	11	60	1	0	0	0	1	0	0	0	0	2	0	1	4	1	No
160	Good	1	23	8	46,33	0	0	0	0	0	0	0	0	0	2	0	1	3	1	No
161	Good	1	64	11	33,55	2	0	0	1	3	3	1	0	0	3	1	0	3	0	No
162	Good	1	11	7	30	0	0	0	0	0	0	0	0	0	1	0	0	2	0	No
163	Good	4	62	12	40,22	3	0	1	0	1	3	1	0	0	3	1	2	10	4	No
164	Good	6	53	9	28,33	3	0	0	1	1	3	0	0	0	2	0	0	10	0	Yes
165	Good	1	20	10	53,50	1	0	0	0	1	0	0	0	0	1	0	0	3	0	No
166	Good	1	15	11	33	0	0	0	0	0	0	0	0	0	1	0	0	1	0	No
167	Good	1	19	12	60	1	0	0	0	0	0	0	0	0	1	0	0	3	0	Yes
168	Good	1	49	14	36,50	0	2	1	1	0	2	0	0	1	1	0	1	8	4	No
169	Good	1	34	17	38	1	0	0	0	0	3	0	0	0	1	0	2	4	3	No
170	Good	1	27	16	80	1	0	0	0	0	0	0	0	0	1	0	1	2	1	No
171	Good	1	28	8	30,20	1	0	0	0	1	2	0	0	0	1	0	0	4	1	No
172	Good	1	28	8	29,60	1	0	0	0	2	2	0	0	0	1	0	0	4	1	No
173	Good	1	17	12	50	0	0	0	0	0	0	0	0	0	1	0	0	2	0	No
174	Good	4	26	10	32,20	1	0	0	0	1	0	0	0	0	1	0	0	4	0	Yes
175	Good	1	17	12	50,50	0	0	0	0	0	0	0	0	0	1	0	0	3	1	No
176	Good	1	19	7	48	0	0	0	0	1	0	0	0	0	1	0	0	2	0	No
177	Good	1	51	24	29,70	1	1	0	0	1	5	1	1	0	1	1	0	5	6	No
178	Good	1	34	10	48	0	1	1	2	1	0	0	0	0	1	1	0	2	4	No

179	Good	1	21	12	58,50	0	0	0	0	0	0	0	0	0	1	0	0	2	1	No
180	Good	1	13	11	40	0	0	0	0	0	0	0	0	0	1	0	0	1	0	Yes
181	Good	4	28	3	24	0	0	0	0	0	0	0	0	0	1	0	0	4	1	No
182	Good	1	32	10	38,60	2	0	1	0	1	2	0	0	0	1	0	1	4	1	No
183	Good	1	26	14	32,20	1	0	0	0	1	2	0	0	0	1	0	0	4	1	No
184	Good	1	30	13	32,80	1	0	0	0	0	2	0	0	0	1	0	0	2	1	Yes
185	Good	1	21	14	23,33	1	0	0	1	0	2	0	0	1	0	1	0	2	1	No
186	Good	1	29	12	52,33	2	0	0	0	2	0	0	0	2	0	1	0	2	1	No
187	Good	1	15	12	13,86	0	0	0	0	0	3	0	0	0	0	1	0	3	3	No
188	Good	1	11	8	30,50	0	0	0	0	0	0	0	0	0	1	0	0	2	0	No
189	Good	1	30	13	40,75	2	0	0	0	1	2	0	0	0	0	1	0	3	2	No
190	Good	1	35	17	68	1	0	0	0	0	0	0	0	0	0	1	0	3	3	No
191	Good	1	19	10	52,50	0	0	0	0	0	0	0	0	0	0	1	0	3	1	No
192	Good	1	14	8	26,67	1	0	0	0	0	0	0	0	1	1	0	0	2	1	No
193	Good	1	23	12	44	1	0	0	0	0	0	0	0	0	1	0	0	4	0	No
194	Good	1	67	12	38,10	4	0	0	1	1	4	0	0	1	2	0	0	10	8	Yes
195	Good	1	18	14	29,25	1	0	1	0	0	0	0	0	0	1	0	0	1	1	No
196	Good	8	44	7	17,88	4	0	0	0	0	6	0	0	0	2	0	0	3	3	No
197	Good	1	28	8	39	0	0	1	1	0	0	0	0	0	3	0	1	3	1	No
198	Good	1	22	11	20,17	0	0	0	0	0	3	0	0	0	1	0	0	2	1	No
199	Good	2	52	15	64,60	4	0	0	1	1	2	0	0	0	1	1	0	8	3	No
200	Good	1	10	11	20	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
201	Good	1	34	10	69,33	1	0	0	0	0	0	0	0	0	2	0	1	4	2	No
202	Good	2	55	11	40,75	1	0	0	3	3	1	0	1	0	3	0	2	5	5	No
203	Good	1	31	18	27,71	1	0	0	0	1	4	0	0	0	1	0	0	5	1	No
204	Good	1	26	15	32	0	1	1	0	1	4	0	0	0	1	0	0	3	3	No
205	Good	1	24	8	42,33	1	0	0	1	0	0	0	0	0	2	0	0	3	0	No
206	Good	1	18	7	28	0	0	0	0	1	0	0	0	0	1	0	0	2	1	Yes

207	Good	1	17	12	32,33	0	0	0	0	1	0	0	0	0	0	0	0	2	1	No
208	Good	1	32	15	26,50	0	0	1	0	0	3	0	0	0	0	0	1	2	1	No
209	Good	1	28	16	33,80	1	0	0	0	0	2	0	0	0	1	0	0	4	1	No
210	Good	1	29	15	83,50	2	1	2	1	0	0	0	0	0	1	0	2	6	0	No
211	Good	1	19	9	50	0	0	0	0	0	0	0	0	0	1	0	0	3	2	No
212	Good	1	24	17	52,67	1	0	0	0	0	2	0	0	0	1	0	0	6	1	No
213	Good	1	19	10	56,50	1	0	1	0	0	0	0	0	0	2	0	0	3	0	No
214	Good	1	16	14	55	1	0	0	0	0	0	0	0	0	1	0	0	1	1	No
215	Good	2	26	5	11,25	1	0	0	0	0	2	0	0	0	1	0	0	5	2	No
216	Good	1	58	9	34,44	2	0	1	1	2	3	0	0	0	1	2	1	9	3	Yes
217	Good	1	26	15	15	0	0	0	0	0	3	0	0	0	1	0	0	8	3	No
218	Good	1	20	10	21,40	0	0	0	0	0	2	0	0	0	1	0	0	3	0	No
219	Good	2	26	9	11,85	2	0	0	0	0	5	0	0	0	1	0	0	3	2	No
220	Good	1	45	10	35	3	0	0	0	0	3	0	0	1	2	0	0	5	4	No
221	Good	2	30	7	22,29	2	0	0	0	0	2	0	0	0	1	0	0	1	2	No
222	Good	1	52	9	31,22	2	0	1	0	1	3	0	0	0	3	0	2	4	3	No
223	Good	1	19	10	54	1	0	0	0	0	0	0	0	0	1	0	0	3	0	No
224	Good	2	23	6	6,61	1	0	0	0	0	2	0	0	0	1	0	0	4	0	No
225	Good	2	35	10	11,50	1	0	0	0	1	4	0	0	0	1	0	0	10	0	No
226	Good	1	56	14	34,33	2	0	1	1	0	5	0	0	0	2	0	0	7	1	No
227	Good	1	22	10	43,67	1	0	0	0	1	0	0	0	0	2	0	0	2	1	No
228	Good	1	41	11	34,43	2	0	0	0	0	4	0	0	0	2	0	0	6	3	No
229	Good	1	12	8	33	0	0	0	0	1	0	0	0	0	1	0	0	2	1	No
230	Good	1	25	15	40,75	2	0	1	0	0	2	0	1	0	2	0	1	3	1	No
231	Good	1	58	12	38,67	3	1	1	0	0	4	0	0	0	3	0	1	7	2	No
232	Good	2	105	15	42,56	5	2	0	5	1	7	1	0	0	3	2	2	14	12	Yes
233	Good	1	43	17	51	1	1	1	0	2	3	0	0	0	1	1	0	4	3	No
234	Good	1	30	8	18,44	0	0	0	1	1	4	0	0	0	1	1	1	4	2	Yes

235	Good	1	18	12	16	0	0	0	0	0	3	0	0	0	1	0	0	2	2	No
236	Good	1	24	13	32,50	1	0	0	0	1	0	0	0	0	1	0	0	1	1	No
237	Good	6	36	9	15	2	0	1	0	3	8	0	0	0	1	0	0	5	1	No
238	Good	1	25	14	30,33	1	0	0	0	1	4	0	0	0	1	0	0	3	2	No
239	Good	1	28	16	21,50	1	0	0	0	0	2	0	0	0	1	0	0	3	0	No
240	Good	1	24	20	28,33	1	0	0	0	0	3	0	0	0	1	0	0	5	1	No
241	Good	1	14	11	29,67	1	0	0	0	0	1	0	0	0	1	0	0	3	1	No
242	Good	1	55	10	46,29	1	0	0	1	0	2	0	0	0	3	0	1	5	3	No
243	Good	1	23	11	29,60	0	0	0	0	0	2	0	0	0	2	0	1	4	2	No
244	Good	1	43	12	35,57	4	0	1	1	0	2	0	0	0	2	0	0	6	0	No
245	Good	2	29	10	17	2	0	0	0	1	4	0	0	0	1	0	0	4	2	No
246	Good	1	22	14	31,50	1	0	1	0	0	0	0	0	0	0	0	2	3	1	No
247	Good	1	18	9	23,75	0	0	1	0	0	0	1	0	0	0	1	1	2	0	Yes
248	Good	2	24	11	19,63	1	0	0	0	0	4	0	0	0	1	0	0	3	2	No
249	Good	1	60	15	48,29	1	0	2	0	1	4	0	0	0	2	0	0	9	3	No
250	Good	2	46	14	54,60	1	0	1	1	0	0	0	0	0	2	0	1	4	2	No
251	Good	1	31	10	58,67	0	0	0	1	1	0	0	0	0	2	0	1	5	3	No
252	Good	1	27	8	49,67	0	0	0	1	1	0	0	0	0	2	0	1	6	2	No
253	Good	1	19	10	34,33	0	0	1	0	0	0	0	0	0	1	0	0	2	1	No
254	Good	1	53	8	48,83	0	0	0	2	0	0	0	0	0	4	0	1	6	2	No
255	Good	1	29	11	25,71	1	1	0	2	0	5	0	0	0	2	0	0	4	0	No
256	Good	1	41	21	59,50	2	0	0	0	0	0	0	0	0	1	0	0	5	3	No
257	Good	1	15	14	46	1	0	0	0	0	0	0	0	0	1	0	0	2	0	No
258	Good	1	22	10	58,50	1	0	0	1	0	0	0	0	0	1	0	1	1	1	Yes
259	Good	1	30	15	56	2	0	0	0	2	2	0	0	0	0	1	0	1	1	No
260	Good	1	22	14	64,50	0	0	1	0	0	0	0	0	0	1	0	0	3	0	Yes
261	Good	1	33	9	19,11	1	0	0	0	0	3	1	0	0	1	1	1	5	0	No
262	Good	1	40	10	43,40	1	0	0	0	1	0	0	0	0	2	0	0	10	4	No

263	Good	1	43	6	22,20	1	0	0	0	1	4	0	0	0	1	0	0	8	1	No
264	Good	1	34	16	29,83	0	1	1	0	0	2	0	0	0	0	0	2	4	2	No
265	Good	1	18	16	58	1	0	0	0	1	0	0	0	0	0	0	0	3	2	No
266	Good	1	19	9	37,67	1	0	0	0	0	0	0	0	0	1	0	1	4	2	No
267	Good	1	21	13	20,17	0	0	0	0	0	3	0	0	0	1	0	1	1	3	Yes
268	Good	1	31	11	44,25	1	0	1	0	0	0	0	0	0	1	0	2	2	1	No
269	Good	1	32	19	23,88	2	0	0	0	2	2	0	0	0	1	0	1	5	1	No
270	Good	1	27	15	50,33	1	0	0	0	1	0	0	0	0	0	0	1	5	2	No
271	Good	1	30	9	39,25	0	0	0	0	0	0	0	0	0	1	0	0	4	1	No
272	Good	1	20	12	37,33	0	0	0	0	0	2	0	0	0	1	0	0	4	0	No
273	Good	1	16	9	44,50	0	0	0	0	0	0	0	0	0	1	0	0	1	1	No
274	Good	1	24	16	71	0	0	1	0	0	0	0	0	0	1	0	1	1	1	Yes
275	Good	1	23	11	39,67	1	0	1	1	0	0	0	0	0	1	0	1	2	1	No
276	Good	1	26	16	28,17	0	0	0	0	0	3	0	0	0	1	1	0	2	0	No
277	Good	1	35	11	52,50	0	0	1	1	0	0	0	0	0	2	0	1	4	0	No
278	Good	1	37	21	54,75	1	0	1	0	0	2	0	0	0	1	0	1	6	1	No
279	Good	1	19	10	34,33	0	0	1	0	0	0	0	0	0	1	0	0	3	1	No
280	Good	1	43	24	29,70	3	0	0	1	1	4	1	1	0	1	1	1	7	2	No
281	Good	1	35	11	42	0	0	1	1	0	0	0	0	0	2	0	1	4	0	No
282	Good	1	24	13	19,14	0	0	1	0	0	2	0	0	0	1	0	1	4	0	No
283	Good	1	32	17	60,67	3	0	1	1	0	0	0	0	0	1	0	1	4	1	No
284	Good	1	33	19	101,50	3	0	0	1	1	0	0	0	0	1	0	0	1	2	No
285	Good	1	20	12	57	0	0	0	0	1	0	0	0	0	1	0	0	1	2	No
286	Good	1	26	14	50,67	2	0	0	0	0	0	0	0	0	1	0	0	4	1	No
287	Good	1	14	8	40	1	0	0	0	0	0	0	0	0	0	1	0	3	1	No
288	Good	1	37	13	32,57	2	0	0	1	0	4	0	0	0	1	1	0	6	1	No
289	Good	1	29	10	59,67	0	0	0	1	0	0	0	0	0	0	2	0	3	2	No
290	Good	1	20	9	50	0	0	0	0	1	0	0	0	0	1	0	0	3	2	No

291	Good	6	36	11	27,89	0	0	0	0	3	4	0	0	0	1	0	0	8	2	Yes
292	Good	1	32	17	47,50	2	0	1	1	0	2	0	0	0	1	0	1	4	2	No
293	Good	4	30	12	27,14	0	0	0	0	1	2	0	0	0	1	0	0	6	2	No
294	Good	10	56	10	21,88	1	0	0	1	1	4	1	0	0	1	1	0	14	0	No
295	Good	1	22	15	43,67	0	0	0	1	0	0	0	0	0	1	0	0	1	0	No
296	Good	1	16	12	19,20	0	0	0	0	0	2	0	0	0	1	0	0	2	0	No
297	Good	1	27	13	35	1	0	0	0	1	0	0	0	0	1	0	1	1	1	No
298	Good	1	12	6	16	0	0	0	0	0	2	0	0	0	1	0	0	1	0	No
299	Good	1	16	12	46,50	0	0	0	0	0	0	0	0	0	1	0	0	2	2	No
300	Good	1	31	15	29,33	1	0	0	0	0	2	0	0	0	1	0	0	1	0	No
301	Good	2	55	13	32,50	1	1	1	2	0	6	0	0	1	3	0	0	4	4	No
302	Good	1	42	13	37,14	0	0	0	0	1	4	0	0	0	2	0	1	4	1	Yes
303	Good	9	77	11	27,53	1	0	0	1	1	8	1	0	1	2	1	1	13	1	No
304	Good	1	33	10	64,33	2	0	1	0	1	0	0	0	0	3	0	0	2	1	No
305	Good	1	22	13	21,67	1	0	0	0	0	3	0	0	0	1	0	0	6	0	No
306	Good	1	19	12	34	0	0	0	0	0	0	0	0	0	1	0	0	3	0	No
307	Good	1	13	9	39,50	1	0	0	0	1	0	0	0	0	0	1	0	2	1	No
308	Good	1	16	10	31,33	1	0	0	0	0	0	0	0	0	0	1	0	3	0	No
309	Good	1	57	27	34	4	0	0	0	1	5	0	0	0	1	0	0	8	3	Yes
310	Good	1	32	16	32,33	2	0	0	0	0	2	0	0	0	1	0	0	7	2	Yes
311	Good	1	18	10	49,50	1	0	0	0	1	0	0	0	0	1	0	0	3	1	Yes
312	Good	1	19	9	49	0	0	0	0	0	0	0	0	0	1	0	0	3	0	Yes
313	Good	1	13	7	35,50	0	0	0	0	0	0	0	0	0	1	0	0	2	1	No
314	Good	6	63	11	15,96	2	0	0	0	0	10	0	0	0	1	0	0	6	5	No
315	Good	1	20	8	19,60	0	0	0	0	0	3	0	0	0	1	0	0	3	0	No
316	Good	1	29	16	42,50	1	0	0	0	0	2	0	0	0	1	0	0	5	0	No
317	Good	1	21	12	23,20	1	0	0	0	1	3	0	0	0	1	0	0	2	1	No
318	Good	1	15	15	51,50	0	0	0	0	0	0	0	1	0	1	0	0	3	2	No

319	Good	1	18	17	13,89	0	0	0	0	0	3	0	0	0	1	0	0	2	1	Yes
320	Good	1	22	9	55,50	0	0	0	0	0	0	0	0	0	1	0	0	4	2	No
321	Good	1	14	8	37,50	0	0	0	0	0	0	0	0	0	1	0	0	3	0	No
322	Good	1	27	15	22,25	1	0	0	0	0	2	0	0	0	1	0	0	6	1	No
323	Good	1	14	5	34	0	0	0	0	0	0	0	0	0	1	0	0	2	1	No
324	Good	1	27	13	24	1	1	0	0	0	2	0	0	0	1	0	0	3	1	Yes
325	Good	1	17	14	13,50	0	0	0	0	1	3	0	0	0	1	0	0	3	1	No
326	Good	1	12	10	37,50	0	0	0	0	0	0	0	0	0	1	0	0	3	1	No
327	Good	1	9	10	26	0	0	0	0	0	0	0	0	0	0	0	0	1	0	No
328	Good	1	24	10	40,67	1	0	0	0	0	1	0	0	0	1	0	0	4	1	No
329	Good	1	22	14	26,20	0	0	0	0	1	2	0	0	0	1	0	1	3	1	No
330	Good	1	11	8	31,50	1	0	0	0	1	0	0	0	0	1	0	0	2	0	No
331	Good	1	14	11	43	1	0	0	0	0	0	0	0	0	1	0	0	3	1	No
332	Good	1	27	13	36,75	2	0	0	0	0	0	0	0	0	1	0	0	4	1	Yes
333	Good	1	17	8	44	1	0	0	0	1	0	0	0	0	0	1	0	3	1	Yes
334	Good	1	41	12	41,17	2	0	0	0	0	2	0	0	0	2	0	0	3	2	No
335	Good	1	25	6	17,86	1	0	0	0	0	4	0	0	0	1	0	0	2	1	No
336	Good	1	10	10	29,50	0	0	0	0	0	0	0	0	0	1	0	0	1	1	No
337	Good	1	23	14	71	1	0	0	0	0	1	0	0	0	1	0	0	4	3	No
338	Good	1	26	14	74,50	1	0	0	0	0	0	0	0	0	2	0	1	3	3	No
339	Good	1	24	15	74	0	0	0	0	1	0	0	0	0	1	1	1	2	2	No
340	Good	1	23	8	16,75	0	0	0	0	0	5	0	0	0	1	0	0	6	1	No
341	Good	2	49	11	55,40	4	0	0	0	2	2	0	0	0	3	0	2	6	2	No
342	Good	1	17	13	21	1	0	0	0	0	2	0	0	0	1	0	0	4	1	No
343	Good	1	24	13	46,33	1	0	0	0	0	0	0	0	0	1	0	0	3	0	No
344	Good	1	22	12	30,75	3	0	0	0	0	2	0	0	0	1	0	0	3	0	No
345	Good	1	12	11	36,50	1	0	0	0	0	0	0	0	0	1	0	0	1	0	No
346	Good	1	28	14	74,50	3	0	0	1	0	0	0	0	0	0	1	0	4	2	No

347	Good	1	23	12	63,50	1	0	0	0	0	0	0	0	0	1	0	1	4	3	No
348	Good	1	11	9	34	1	0	0	0	0	0	0	0	0	1	0	0	3	1	No
349	Good	1	16	11	46,50	0	0	0	0	0	0	0	0	0	1	0	0	4	2	No
350	Good	1	15	13	48,50	1	0	1	1	0	0	0	0	0	0	1	0	3	1	No
351	Good	1	37	10	70	1	1	1	1	0	1	0	0	0	1	0	2	4	4	No
352	Good	1	25	10	25,80	1	0	0	0	0	3	0	0	0	1	0	0	4	1	No
353	Good	1	37	17	30,57	1	0	0	1	0	3	0	0	0	1	0	1	4	1	No
354	Good	1	28	18	56	1	0	1	1	0	0	0	0	0	0	1	0	3	0	No
355	Good	1	19	11	22,60	0	0	0	0	0	3	0	0	0	1	0	0	5	0	Yes
356	Good	1	39	8	70	1	0	0	0	0	0	0	0	0	2	0	2	8	2	Yes
357	Good	1	41	12	26,67	2	1	1	1	0	3	0	0	0	1	1	0	10	2	No
358	Good	1	25	13	73,50	1	2	1	1	0	0	0	0	0	1	0	2	5	1	No
359	Good	1	37	11	53,25	1	0	1	1	0	0	0	0	0	2	0	2	3	2	No
360	Good	1	43	12	46,20	0	0	1	0	0	0	1	0	0	1	1	2	4	2	No
361	Good	1	12	13	41,50	1	0	0	0	0	0	0	0	0	1	0	0	2	1	No
362	Good	1	33	11	51,50	1	0	0	0	0	2	0	0	0	2	0	0	4	2	No
363	Good	1	22	13	42	1	0	0	0	0	2	0	0	0	1	0	0	2	2	No
364	Good	1	43	10	40	0	0	0	1	1	2	0	0	0	2	0	0	5	1	No
365	Good	1	29	8	38	1	0	0	0	0	0	0	0	0	3	0	1	4	2	No
366	Good	1	18	12	56	0	0	0	0	0	0	1	0	0	1	1	1	3	2	No
367	Good	1	29	14	54,67	0	2	1	0	0	0	0	1	0	1	0	0	4	1	No
368	Good	1	44	20	62,50	1	1	1	1	0	0	0	0	0	2	0	0	4	4	No
369	Good	1	22	11	31,50	2	0	0	1	1	2	0	0	0	1	0	1	2	2	No
370	Good	4	58	9	32,50	0	2	0	1	0	6	1	0	0	2	1	0	10	1	No
371	Good	3	53	13	36,88	5	0	0	1	1	2	0	0	0	1	0	1	8	1	No
372	Good	1	25	14	49,67	0	1	1	0	0	0	1	0	0	0	1	0	1	1	No
373	Good	1	21	9	32,67	0	1	1	0	0	0	0	0	0	1	0	0	2	0	No
374	Good	2	73	19	53,25	3	1	2	1	0	3	1	0	0	1	1	1	7	1	No

375	Good	1	27	15	53,67	5	0	0	0	0	0	0	0	0	1	0	0	3	0	No
376	Good	1	25	22	96,50	2	0	0	2	0	0	0	0	1	0	1	0	4	1	No
377	Good	1	16	8	41,50	0	0	0	0	0	0	0	0	0	0	1	1	3	0	Yes
378	Good	1	22	14	65	0	0	0	0	0	0	0	0	0	1	0	0	2	1	No
379	Good	1	18	12	53	0	0	0	0	0	0	0	0	0	1	0	0	2	2	Yes
380	Good	1	38	12	27,75	2	0	1	0	0	3	1	0	0	1	1	2	8	1	No
381	Good	1	19	12	37,67	1	0	0	0	0	2	1	0	0	1	0	0	2	1	No
382	Good	1	18	10	48,50	1	1	0	0	0	0	0	0	1	1	0	1	2	2	No
383	Good	1	10	8	31	0	0	0	0	0	0	0	0	0	1	0	0	1	0	No
384	Good	1	11	19	28	1	0	0	0	1	0	0	0	0	1	0	0	1	0	No
385	Good	1	17	12	51,50	0	0	0	0	0	0	0	0	0	1	0	0	2	1	No
386	Good	1	23	15	47,33	2	0	0	0	0	0	0	0	0	1	0	1	4	3	No
387	Good	1	18	12	51,50	0	0	0	0	0	0	0	0	0	0	1	0	4	3	Yes
388	Good	1	37	19	35,33	3	0	1	0	1	2	0	0	0	2	0	2	4	3	No
389	Good	1	36	14	56,75	2	0	0	2	0	0	0	0	0	1	1	0	6	0	No
390	Good	1	19	18	62	0	0	0	0	0	0	0	0	0	1	0	0	2	0	No
391	Good	1	19	13	36,33	1	0	1	0	0	0	0	0	0	1	0	0	2	1	No
392	Good	1	22	16	67	1	0	1	1	0	0	0	0	0	1	0	0	3	1	No
393	Good	1	16	8	41,50	0	0	0	0	0	0	0	0	0	0	1	1	3	0	Yes
394	Good	1	15	12	33,67	1	0	0	0	0	0	0	0	0	0	1	0	5	1	No
395	Good	1	22	8	45	2	1	0	1	2	0	0	0	0	0	2	0	3	1	No
396	Good	1	16	12	32,67	1	0	0	0	0	0	0	0	0	1	0	0	2	1	No
397	Good	11	92	8	24,58	4	1	0	3	0	9	0	0	1	1	0	0	24	1	Yes
398	Good	1	31	10	59,67	0	0	0	2	2	0	0	0	0	2	0	0	7	3	No
399	Good	1	26	13	29,20	2	0	0	0	2	2	0	0	0	0	1	0	4	1	No
400	Good	1	38	7	14,93	1	0	0	0	0	3	0	0	0	0	2	1	4	1	Yes
401	Good	1	35	17	18,09	2	0	0	2	0	6	0	0	0	0	1	0	6	0	No
402	Good	1	15	7	38	0	0	0	0	0	0	0	0	0	0	1	0	4	0	No

403	Good	1	12	5	30	0	0	0	0	0	0	0	0	0	0	1	0	2	0	No
404	Good	1	14	12	39	0	0	0	1	0	0	0	0	0	0	1	0	1	1	No
405	Good	1	29	14	15	1	0	0	0	0	7	0	0	0	0	1	0	5	0	No
406	Good	9	47	11	12,63	1	1	1	0	3	10	0	0	0	1	0	0	11	2	Yes
407	Good	1	13	12	18,75	1	0	0	0	1	2	0	0	0	0	1	0	2	0	No
408	Good	1	22	18	31,40	0	0	0	0	0	3	0	0	0	1	0	0	4	1	No
409	Good	1	42	22	119	2	0	0	0	0	0	0	0	0	2	0	0	1	1	No
410	Good	1	24	16	19,25	0	0	0	0	0	1	1	0	0	0	1	0	4	0	No
411	Good	1	39	27	29,70	3	0	0	3	3	3	0	0	1	1	0	2	6	3	No
412	Good	1	17	18	59,50	2	0	0	1	0	0	0	0	0	0	0	0	4	0	No
413	Good	1	26	17	87,50	0	0	0	1	1	0	0	0	0	1	0	0	6	1	No
414	Good	3	32	18	49	2	0	0	0	0	0	0	0	1	1	0	0	5	1	No
415	Good	1	35	16	61	3	0	0	0	0	2	0	0	1	0	0	1	6	1	No
416	Good	1	29	21	33,50	2	0	1	1	1	2	1	0	1	1	1	0	6	3	Yes
417	Good	1	22	12	48,67	1	0	0	0	1	0	0	0	0	0	0	1	3	0	No
418	Good	1	28	15	80	2	0	0	0	0	0	0	0	0	1	0	0	5	2	No
419	Good	1	28	20	32,50	2	0	0	1	0	0	0	0	0	1	0	0	2	3	No
420	Good	1	16	14	54	0	0	0	0	0	0	0	0	0	1	0	0	4	0	No
421	Good	8	105	8	24,92	4	0	3	0	1	8	0	0	0	1	0	1	9	11	No
422	Good	1	31	18	31,50	1	1	0	1	0	1	0	0	0	1	0	0	2	2	No
423	Good	1	19	12	27,50	1	1	0	0	0	0	0	0	0	1	0	0	0	4	No
424	Good	1	18	12	53	2	0	0	0	1	0	0	0	0	1	0	0	0	1	No
425	Good	1	17	6	24	0	0	0	0	0	3	0	0	0	0	1	0	2	0	No
426	Good	1	26	15	77	0	0	0	0	0	0	0	1	0	0	1	0	2	2	No
427	Good	1	18	13	57	0	0	0	1	1	0	0	0	0	1	0	0	0	1	No
428	Good	1	12	12	37,50	0	0	0	0	0	0	0	0	0	1	0	0	1	0	No
429	Good	1	16	10	48	0	1	0	0	1	0	0	0	0	1	0	1	0	0	No
430	Good	5	128	16	73,70	9	0	1	1	0	0	1	0	0	6	1	7	12	8	Yes

431	Good	1	30	10	57,33	0	1	0	1	0	0	0	0	0	1	0	1	2	3	Yes
432	Good	1	15	8	39,50	0	0	0	1	2	0	0	0	0	1	0	0	2	2	No
433	Good	1	19	11	39,33	0	1	0	2	1	0	0	1	0	1	0	0	4	0	No
434	Good	1	27	15	40,25	0	0	0	0	2	2	0	0	0	1	0	0	4	0	No
435	Good	1	27	14	39,75	0	0	0	0	1	2	0	0	0	0	1	0	3	0	Yes
436	Good	1	37	19	46,60	2	0	0	1	0	4	0	0	0	2	0	1	9	1	Yes
437	Good	1	18	12	52,50	1	0	0	0	0	0	0	0	0	1	0	0	2	1	Yes
438	Good	1	44	21	87,33	1	0	0	1	1	0	0	0	0	1	0	0	4	1	No
439	Good	1	23	13	68	2	0	0	0	1	0	0	0	0	1	0	0	5	0	No
440	Good	1	22	12	26,80	0	0	0	0	0	2	0	0	0	1	0	1	5	2	Yes
441	Good	1	16	9	30,33	1	0	0	0	1	0	0	0	0	1	0	0	2	0	No
442	Good	1	30	17	61,33	0	1	0	1	0	2	1	0	0	1	1	1	4	3	Yes
443	Good	1	26	18	43	0	1	1	0	1	2	0	0	0	1	0	0	3	2	No
444	Good	1	38	18	30,43	2	0	0	0	1	2	0	0	0	2	0	0	4	2	Yes
445	Good	1	31	23	38,50	2	0	0	0	2	1	0	0	0	1	0	0	4	1	No
446	Good	1	32	18	32	3	0	0	0	2	2	0	0	0	1	0	0	3	0	Yes
447	Good	1	15	10	40,50	0	0	0	0	0	0	0	0	0	1	0	0	1	0	No
448	Good	1	15	12	50	0	0	0	1	0	0	0	0	0	1	0	0	1	0	Yes
449	Good	1	18	17	41,67	1	0	0	0	0	2	0	0	0	1	0	1	1	2	No
450	Good	1	40	19	56	1	1	0	0	2	2	0	0	0	1	0	0	6	0	No
451	Good	1	24	15	25,67	1	1	1	0	0	2	0	0	0	1	0	1	3	0	No
452	Good	1	38	19	76	2	1	1	1	1	0	0	0	0	1	0	0	5	4	No
453	Good	1	23	12	67,50	0	1	0	1	1	0	0	0	0	1	0	0	2	0	Yes
454	Good	1	10	11	32	0	0	0	0	0	0	0	0	1	1	0	0	1	2	No
455	Good	1	32	16	46,25	1	0	0	0	1	2	0	0	0	1	0	0	5	2	No
456	Good	1	47	12	64,25	0	0	0	1	0	0	0	0	0	1	0	1	6	4	Yes
457	Good	5	66	31	45,11	3	0	1	0	1	6	0	0	0	1	1	1	10	3	No
458	Good	1	20	16	66,50	0	0	0	1	0	0	0	0	0	1	0	0	2	1	No

459	Good	1	38	19	46	0	0	1	2	1	1	1	0	0	1	1	1	5	1	No
460	Good	1	22	17	76,50	1	0	0	0	1	1	0	0	0	1	0	0	3	1	No
461	Good	1	46	24	73	0	0	1	2	1	0	0	0	0	1	0	1	5	4	No
462	Good	1	26	15	76,50	0	1	0	2	0	0	1	0	0	1	1	0	2	0	Yes
463	Good	1	18	12	55	1	0	0	0	0	0	0	0	0	1	0	0	1	1	No
464	Good	1	28	12	20,63	1	0	0	2	0	3	0	0	0	1	0	0	5	1	No
465	Good	1	29	9	56,33	0	0	0	1	0	0	0	0	0	0	0	1	3	2	No
466	Good	1	17	8	48,50	0	0	0	0	0	0	0	0	0	1	0	1	2	2	Yes
467	Good	1	47	15	29,20	1	0	0	1	1	7	0	0	0	2	0	0	7	2	No
468	Good	1	46	13	45,33	2	1	0	0	0	1	0	0	0	2	0	0	6	2	Yes
469	Good	1	13	10	40,50	0	0	0	0	0	0	0	0	0	1	0	0	3	2	No
470	Good	1	35	16	45,25	1	1	1	0	0	2	0	1	0	1	0	0	2	0	Yes
471	Good	1	15	9	43	0	0	0	0	0	0	0	0	0	1	0	0	2	1	No
472	Good	1	18	13	23,60	1	0	0	0	1	2	0	0	0	1	0	0	3	2	No
473	Good	1	19	12	30,50	0	1	0	2	1	0	0	0	0	1	0	0	3	0	No
474	Good	1	21	11	54,50	0	0	0	0	0	0	0	0	1	1	0	0	3	1	No
475	Good	1	13	17	46,50	0	1	0	0	0	0	1	0	0	0	1	0	1	1	No
476	Good	1	13	12	39	1	1	0	0	0	0	1	0	0	0	1	0	1	0	No
477	Good	1	28	15	47,50	0	0	0	0	1	0	0	0	0	2	0	0	0	0	No
478	Good	1	20	15	15,13	0	1	0	0	0	3	0	0	0	1	0	0	5	2	No
479	Good	1	19	9	37,33	1	0	0	0	0	0	0	0	0	0	0	0	2	0	No
480	Good	1	14	16	54	1	0	0	0	1	0	0	0	0	0	0	0	2	1	No
481	Good	1	13	11	26,67	0	0	0	0	0	2	1	0	0	0	1	0	2	1	No
482	Good	1	28	11	12,69	1	0	0	0	1	8	0	0	0	0	0	1	4	0	No
483	Good	1	18	14	37,67	0	0	1	0	1	0	0	0	0	1	0	0	2	1	No
484	Good	1	23	14	71,50	1	1	0	0	0	0	1	0	0	0	1	1	4	1	No
485	Good	1	13	9	36	0	0	0	0	0	0	0	0	0	1	0	1	1	1	No
486	Good	1	21	4	9,91	0	0	0	1	0	0	0	0	0	1	0	1	4	0	No

487	Good	1	10	11	30	0	0	0	0	0	0	0	0	0	0	0	0	1	1	No
488	Good	1	21	11	46,33	2	0	0	0	0	0	0	0	0	2	0	1	3	1	No
489	Good	19	102	11	11,73	0	0	0	1	0	6	0	0	2	1	0	0	20	3	Yes
490	Good	4	28	14	25,83	0	0	0	0	1	0	0	0	0	2	0	0	5	2	Yes
491	Good	1	10	6	25	1	0	0	0	0	0	1	0	0	0	1	1	0	1	No
492	Good	1	25	10	52	1	0	0	0	1	0	1	0	0	1	1	1	3	0	Yes
493	Good	3	14	10	24	0	0	0	0	0	0	0	0	0	0	0	1	1	1	No
494	Good	1	17	10	45	1	0	1	2	0	0	1	0	0	0	1	0	2	2	No
495	Good	13	69	6	14,46	0	0	0	2	0	4	0	0	0	0	0	0	10	1	No
496	Good	1	10	13	23	0	0	0	0	0	0	0	0	0	0	0	0	0	1	No
497	Good	1	17	10	32	0	0	0	0	0	0	0	0	0	1	0	0	2	0	No
498	Good	1	28	13	51	1	0	1	0	0	0	0	0	0	1	0	1	3	1	No
499	Good	1	10	7	11,75	1	0	0	0	0	0	0	0	0	1	0	0	2	0	No
500	Good	1	11	7	28,50	0	0	0	0	0	0	0	0	0	1	0	0	1	0	No
501	Good	1	18	12	35	0	1	0	0	0	0	1	0	0	0	1	1	2	0	Yes
502	Good	1	15	6	36	1	0	0	0	0	0	0	0	0	0	0	1	0	2	No
503	Good	1	12	7	32,50	0	0	0	0	0	0	0	0	0	0	0	0	0	0	No
504	Good	1	23	13	69	1	1	0	0	0	0	0	0	0	2	0	0	3	2	No
505	Good	1	6	16	21	0	0	0	0	0	0	0	0	0	0	0	1	1	1	No
506	Good	1	11	9	29,50	0	1	0	0	0	0	0	0	0	0	0	1	3	1	No
507	Good	1	13	9	34,50	1	0	0	0	0	0	0	0	0	0	0	0	2	1	No
508	Good	1	21	13	37,33	0	0	1	0	0	0	0	0	1	1	0	1	1	0	No
509	Good	1	39	20	31,71	1	0	0	2	0	3	1	0	0	1	1	2	3	1	Yes
510	Good	1	29	15	41,50	0	1	0	2	0	2	0	0	0	1	0	1	0	1	No
511	Good	1	11	9	31	0	0	0	0	1	0	0	0	0	1	0	0	1	2	No
512	Good	1	20	11	28,25	0	0	0	0	0	0	0	0	0	1	0	0	2	1	No
513	Good	1	26	15	50,33	2	0	0	0	1	0	0	0	0	1	0	0	3	0	No
514	Good	1	27	12	69	0	1	0	0	0	0	0	0	0	1	0	1	2	0	No

515	Good	1	35	13	35,67	1	0	0	2	1	4	0	0	0	2	0	0	6	0	No
516	Good	1	16	10	31,67	1	0	0	0	0	2	0	0	0	1	0	0	3	0	No
517	Good	1	21	13	22	1	1	0	0	0	3	1	0	0	0	1	0	1	0	No
518	Good	1	19	12	16,14	0	0	0	0	0	3	1	0	0	0	1	1	2	1	No
519	Good	1	27	11	55,33	1	0	0	0	0	0	0	0	0	2	0	1	3	0	No
520	Good	1	15	10	43	1	0	0	0	0	0	0	0	0	1	0	0	3	0	No
521	Good	1	18	16	61	1	0	0	0	0	0	0	0	0	1	0	0	1	1	No
522	Good	1	18	11	56	1	0	0	0	1	0	0	0	0	1	0	1	2	1	No
523	Good	1	13	9	26,33	0	0	0	0	1	0	0	0	0	1	0	1	0	0	No
524	Good	1	11	14	24	0	0	0	0	0	2	0	0	0	1	0	0	1	0	No
525	Good	1	20	14	40,67	3	0	0	0	0	2	0	0	0	1	0	1	3	2	No
526	Good	1	15	12	16,33	1	0	0	0	0	2	0	0	0	1	0	1	3	2	No
527	Good	1	21	13	66,50	1	0	0	0	0	0	0	0	0	1	0	0	3	0	No
528	Good	1	27	15	27,67	0	0	0	0	0	2	0	0	0	1	0	1	1	0	No
529	Good	1	33	19	69,67	0	0	0	0	0	0	0	0	0	1	0	0	3	0	No
530	Good	1	18	18	23,17	0	0	0	0	0	2	1	0	0	1	0	0	5	0	No
531	Good	1	19	10	26,50	0	0	0	0	1	2	0	0	0	1	0	0	5	0	No
532	Good	1	13	17	49	1	0	0	0	1	0	0	0	0	1	0	0	3	2	No
533	Good	1	25	12	45	1	0	0	0	0	1	0	0	0	1	0	0	4	1	No
534	Good	1	42	13	21,27	2	0	0	1	3	5	0	0	0	2	0	0	9	2	No
535	Good	1	23	12	44,67	3	0	1	1	0	0	0	0	0	1	0	0	2	0	No
536	Good	1	14	9	38,50	0	0	0	0	1	0	0	0	0	1	0	0	2	0	No
537	Good	1	31	16	45,25	1	0	0	0	1	0	0	0	0	1	0	0	2	0	No
538	Good	1	22	13	60,50	2	0	0	0	0	0	0	0	0	1	0	1	3	1	Yes
539	Good	1	13	7	32,50	0	0	0	0	0	0	0	0	1	1	0	0	2	0	No
540	Good	1	9	14	35	0	0	0	0	0	0	0	0	0	0	1	0	2	0	No
541	Good	4	59	9	32,80	1	1	0	2	0	6	0	0	0	2	0	0	9	4	No
542	Good	4	35	8	25,38	0	0	0	0	2	1	0	0	0	1	0	0	6	1	No

543	Good	1	44	11	40,50	1	0	0	3	0	1	0	0	0	3	0	0	6	4	Yes
544	Good	1	25	11	25,40	2	0	0	0	0	1	0	0	0	1	0	0	8	1	No
545	Good	1	55	11	22,14	4	0	0	1	0	9	2	0	0	2	2	0	12	2	No
546	Bad	1	10	13	32	0	0	0	0	0	0	0	0	0	1	0	0	0	1	No
547	Bad	1	8	13	24	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
548	Bad	1	12	10	32,50	0	0	0	0	0	0	0	0	0	1	0	0	1	0	No
549	Bad	1	12	11	34,50	0	0	0	0	0	0	0	0	0	1	0	1	0	1	No
550	Bad	1	13	10	38	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
551	Bad	1	17	13	31	0	0	0	0	0	0	0	0	0	1	0	1	1	0	No
552	Bad	1	11	12	35,50	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
553	Bad	1	12	13	36,50	0	0	0	0	0	0	0	0	0	1	0	0	1	1	No
554	Bad	1	27	12	72,50	1	0	0	0	2	0	0	0	0	1	0	0	0	1	No
555	Bad	1	20	16	62	0	0	0	0	0	0	0	0	0	1	0	0	1	1	No
556	Bad	1	18	10	20,40	0	0	0	0	0	4	0	0	0	1	0	1	0	1	No
557	Bad	1	24	13	66	0	1	0	0	0	0	0	0	0	1	0	2	0	2	No
558	Bad	1	23	13	65	0	0	0	0	0	0	0	0	0	0	1	2	0	2	No
559	Bad	1	10	5	24	0	0	0	0	0	0	0	0	0	1	0	1	0	2	No
560	Bad	1	9	10	26,50	0	0	0	0	0	0	0	0	0	1	0	1	0	1	No
561	Bad	1	7	9	10,75	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
562	Bad	1	13	12	26,67	0	0	0	0	0	0	0	0	0	1	0	0	1	1	No
563	Bad	1	11	14	24,33	0	0	0	0	0	0	0	0	0	1	0	0	1	1	No
564	Bad	1	12	20	33	0	0	0	0	0	0	0	0	0	1	0	0	1	1	No
565	Bad	1	11	12	23,33	0	0	0	0	0	0	0	0	0	1	0	0	1	1	No
566	Bad	1	12	11	24,33	0	0	0	0	0	0	0	0	0	1	0	0	1	1	No
567	Bad	1	15	15	34,33	0	0	0	0	0	0	0	0	0	1	0	0	1	1	No
568	Bad	1	19	17	25,20	0	1	1	0	0	1	0	0	0	1	0	0	1	1	No
569	Bad	1	11	12	23	0	0	0	0	0	0	0	0	0	1	0	0	1	1	No
570	Bad	1	11	13	22,67	0	0	0	0	0	0	0	0	0	1	0	0	1	2	No

571	Bad	1	11	11	21,67	0	0	0	0	0	0	0	0	0	1	0	0	1	1	No
572	Bad	1	17	13	48,50	0	0	0	1	1	0	0	0	0	1	0	1	2	1	No
573	Bad	1	17	13	49,50	0	0	0	0	1	0	0	0	0	1	0	0	2	0	No
574	Bad	1	17	12	48	0	0	0	0	1	0	0	0	0	1	0	0	2	0	No
575	Bad	1	12	14	39	0	0	0	0	0	0	0	0	0	1	0	0	1	0	No
576	Bad	1	12	12	37,50	0	0	0	0	0	0	0	0	0	1	0	0	1	0	No
577	Bad	1	15	11	44	0	0	0	0	0	0	0	0	0	1	0	1	0	0	No
578	Bad	1	16	10	21,25	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
579	Bad	1	17	8	42	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
580	Bad	1	12	9	31	0	0	0	0	0	0	0	0	0	1	0	1	0	1	No
581	Bad	1	7	14	26	0	0	0	0	0	0	0	0	0	1	0	1	0	1	No
582	Bad	1	27	12	69,50	0	0	0	3	1	0	0	0	0	1	0	0	1	0	No
583	Bad	1	18	7	42,50	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
584	Bad	1	17	6	40	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
585	Bad	1	21	7	33,33	0	0	0	0	0	0	0	0	0	1	0	0	1	0	No
586	Bad	1	16	5	41	0	0	0	0	1	0	0	0	0	1	0	0	1	0	No
587	Bad	1	25	11	63	0	0	0	1	0	0	0	0	0	1	0	1	0	0	No
588	Bad	1	30	14	77,50	0	0	0	1	0	0	0	0	0	1	0	1	0	0	No
589	Bad	1	38	19	107,50	1	0	0	2	0	0	0	0	0	1	0	1	0	1	No
590	Bad	1	36	18	1	0	0	0	2	0	0	0	0	0	1	0	1	1	1	No
591	Bad	1	34	18	49,50	0	0	0	1	0	2	0	0	0	1	0	1	1	2	No
592	Bad	1	22	5	28,50	0	0	0	0	0	0	0	0	0	2	0	0	1	0	No
593	Bad	1	14	9	12,83	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
594	Bad	1	9	8	25	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
595	Bad	1	16	8	42,50	1	0	0	0	0	0	0	0	0	1	0	0	0	0	No
596	Bad	1	12	7	35,50	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
597	Bad	1	12	6	31	0	0	0	0	0	0	0	0	0	1	0	0	0	1	No
598	Bad	1	14	10	44	0	0	0	0	0	0	0	0	0	0	1	0	0	0	No

599	Bad	1	10	7	20	0	0	0	0	0	0	0	0	0	1	0	0	0	1	No
600	Bad	1	9	6	27,50	0	0	0	0	0	0	0	0	0	1	0	0	0	1	No
601	Bad	1	15	8	16,40	0	0	0	0	0	2	0	0	0	1	0	0	1	0	No
602	Bad	1	16	7	29,33	0	0	0	0	0	2	0	0	0	1	1	0	0	0	No
603	Bad	1	10	5	30,50	0	0	0	0	0	0	0	0	0	0	1	0	0	0	No
604	Bad	1	22	6	40	0	0	0	0	0	0	0	0	0	2	0	0	1	0	No
605	Bad	1	12	9	17,50	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
606	Bad	1	9	8	25	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
607	Bad	1	15	8	40	1	0	0	0	0	0	0	0	0	1	0	0	0	0	No
608	Bad	1	12	7	35,50	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
609	Bad	1	12	6	31	0	0	0	0	0	0	0	0	0	1	0	0	0	1	No
610	Bad	1	14	10	44	0	0	0	0	0	0	0	0	0	0	1	0	0	0	No
611	Bad	1	10	7	20,33	0	0	0	0	0	0	0	0	0	1	0	0	0	1	No
612	Bad	1	15	8	16,40	0	0	0	0	0	2	0	0	0	1	0	0	1	0	No
613	Bad	1	15	7	27,67	0	0	0	0	0	2	0	0	0	0	1	0	0	0	No
614	Bad	1	10	5	30,50	0	0	0	0	0	0	0	0	0	0	1	0	0	0	No
615	Bad	1	24	8	37	0	0	0	0	0	0	0	0	0	2	0	0	2	0	No
616	Bad	1	17	6	38	0	0	0	0	0	0	0	0	0	2	0	0	0	0	No
617	Bad	1	9	2	14,33	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
618	Bad	1	12	10	39,50	0	0	0	0	0	0	0	0	0	1	0	0	1	0	No
619	Bad	1	12	4	13,60	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
620	Bad	1	9	9	19	0	0	0	0	0	0	0	0	0	1	0	0	0	1	No
621	Bad	1	13	9	42,50	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
622	Bad	1	13	8	25	0	0	0	0	0	0	0	0	0	1	0	0	1	0	No
623	Bad	1	19	9	35	0	0	0	0	0	2	0	0	0	0	1	0	0	0	No
624	Bad	1	14	8	41,50	0	0	0	0	0	0	0	0	0	0	1	0	0	0	No
625	Bad	1	48	21	65,50	1	0	0	0	1	2	0	0	0	1	0	1	1	3	Yes
626	Bad	1	23	13	38	0	0	0	0	0	0	0	0	0	1	0	0	1	2	No

627	Bad	1	23	13	37,50	0	0	0	0	0	0	0	0	0	1	0	0	1	0	No
628	Bad	1	29	15	47,75	2	0	0	0	1	0	0	0	0	1	0	0	0	0	No
629	Bad	1	42	20	87,67	2	0	0	0	1	0	0	0	0	1	0	0	1	2	No
630	Bad	1	33	17	73	0	0	0	0	0	2	0	0	0	1	0	2	2	1	No
631	Bad	1	15	13	57	0	0	0	0	0	0	0	0	1	1	0	0	0	0	No
632	Bad	1	15	12	56	0	0	0	0	0	0	0	0	1	1	0	0	0	0	No
633	Bad	1	12	9	40,50	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
634	Bad	1	15	11	52	1	0	0	0	0	0	0	0	0	1	0	1	0	1	No
635	Bad	1	12	9	41,50	0	0	0	0	1	0	0	0	0	1	0	0	0	0	No
636	Bad	1	13	9	46	0	0	0	0	1	0	0	0	0	1	0	0	0	0	No
637	Bad	1	15	7	30,33	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
638	Bad	1	10	7	35,50	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
639	Bad	1	10	8	36	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
640	Bad	1	10	10	35,50	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
641	Bad	1	22	13	50,67	1	1	0	0	0	0	0	0	0	1	0	1	0	1	No
642	Bad	1	17	8	56,50	1	0	0	0	0	0	0	0	0	1	0	0	0	0	No
643	Bad	1	19	9	17	0	0	0	0	0	2	0	0	0	1	0	0	0	0	No
644	Bad	1	7	2	19	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
645	Bad	1	7	2	23,50	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
646	Bad	1	7	2	25	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
647	Bad	1	7	2	24	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
648	Bad	1	15	4	27	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
649	Bad	1	7	2	25,50	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
650	Bad	1	7	2	24,50	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
651	Bad	1	13	8	14,33	0	0	0	0	0	2	0	0	0	1	0	0	0	0	No
652	Bad	1	9	2	30	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
653	Bad	1	10	4	35	0	0	0	0	0	0	0	0	0	0	1	0	0	0	No
654	Bad	1	20	11	66	2	0	0	0	0	0	0	0	0	1	0	0	0	0	No

655	Bad	1	13	11	53	1	0	0	0	0	0	0	0	0	0	1	0	0	1	No
656	Bad	1	17	9	54	1	0	0	0	0	0	0	0	0	1	0	0	0	0	No
657	Bad	1	7	4	26,50	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
658	Bad	1	16	5	22	1	0	0	1	3	0	0	0	0	1	0	0	0	0	No
659	Bad	1	13	9	46,50	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
660	Bad	1	9	9	34,50	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
661	Bad	1	8	5	31	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
662	Bad	1	17	10	57	0	0	0	0	1	0	0	0	0	1	0	0	1	0	No
663	Bad	1	20	11	65,50	2	0	0	0	1	0	0	0	0	1	0	1	1	0	No
664	Bad	1	19	12	44,33	2	0	0	0	0	0	0	0	0	1	0	0	1	0	No
665	Bad	1	17	15	71,50	1	0	0	0	1	0	0	0	0	1	0	0	1	0	No
666	Bad	1	33	16	101	2	0	0	1	0	0	0	0	0	1	0	0	1	1	No
667	Bad	1	35	15	98	3	0	0	3	0	0	0	0	0	1	0	0	1	1	No
668	Bad	1	34	14	98,50	2	0	0	1	0	0	0	0	0	1	0	0	1	1	No
669	Bad	1	12	16	31,33	1	0	0	0	0	0	0	0	0	1	0	0	1	0	No
670	Bad	1	23	10	63,50	1	0	0	0	1	0	0	0	0	1	0	0	0	0	No
671	Bad	1	36	22	131,50	1	0	0	0	0	0	0	0	0	1	0	0	0	3	No
672	Bad	1	13	7	25,33	0	0	0	0	0	2	0	0	0	1	0	0	0	0	No
673	Bad	1	12	4	22	0	0	0	0	0	2	0	0	0	1	0	0	0	0	No
674	Bad	1	16	5	13,50	0	0	0	0	0	2	0	0	0	1	0	0	0	0	No
675	Bad	1	16	3	17,50	1	0	0	0	0	2	0	0	0	1	0	0	0	0	No
676	Bad	1	18	10	50	0	0	0	1	0	0	0	0	0	1	0	1	0	1	No
677	Bad	1	11	10	33	0	0	0	1	0	0	0	0	0	1	0	0	0	0	No
678	Bad	1	21	16	78,50	0	0	0	1	1	0	0	0	0	1	0	1	2	1	No
679	Bad	1	10	7	32	0	0	0	0	0	0	0	0	0	0	1	0	0	1	No
680	Bad	1	13	7	23	0	0	0	0	0	0	0	0	0	1	0	0	1	0	No
681	Bad	1	15	7	28	0	0	0	0	0	2	0	0	0	0	1	0	0	0	No
682	Bad	1	10	5	30,50	0	0	0	0	0	0	0	0	0	0	1	0	0	0	No

683	Bad	1	20	12	60	0	0	0	0	0	0	0	0	1	2	0	0	0	0	No
684	Bad	1	15	14	58,50	0	0	0	1	0	0	0	0	0	1	0	0	1	1	No
685	Bad	1	21	6	42	0	0	0	0	0	0	0	0	0	2	0	0	0	0	No
686	Bad	1	23	11	61,50	0	0	0	1	0	0	0	0	0	1	0	1	0	0	No
687	Bad	1	28	13	76	0	0	0	1	0	0	0	0	0	1	0	1	0	0	No
688	Bad	1	12	8	36	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
689	Bad	1	12	7	31,50	0	0	0	0	0	0	0	0	0	1	0	0	0	1	No
690	Bad	1	12	8	33,50	0	0	0	0	0	0	0	0	0	1	0	0	0	1	No
691	Bad	1	10	8	31,50	0	0	0	0	0	0	0	0	0	0	1	0	0	1	No
692	Bad	1	13	8	34	0	0	0	0	0	0	0	0	0	1	0	0	1	0	No
693	Bad	1	15	8	28,33	0	0	0	0	0	2	0	0	0	0	1	0	0	0	No
694	Bad	1	10	6	31	0	0	0	0	0	0	0	0	0	0	1	0	0	0	No
695	Bad	1	12	14	21	0	0	0	0	0	0	0	0	0	1	0	0	0	1	No
696	Bad	1	14	10	17,20	0	0	0	0	0	1	0	0	0	1	0	0	0	0	No
697	Bad	1	21	15	32,75	1	0	0	0	0	2	0	0	0	1	0	0	2	3	No
698	Bad	1	16	11	23	0	0	0	0	1	0	0	0	0	1	0	0	1	2	No
699	Bad	1	21	17	29,20	1	0	0	0	0	2	0	0	0	1	0	0	2	1	No
700	Bad	1	14	12	22	0	0	0	0	0	0	0	0	0	1	0	0	1	0	No
701	Bad	1	15	13	24,25	0	0	0	0	0	0	0	0	0	1	0	0	1	0	No
702	Bad	1	13	10	20	0	0	0	0	0	0	0	0	0	1	0	0	2	0	No
703	Bad	1	12	12	20	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
704	Bad	1	29	7	48,33	0	0	0	0	0	0	0	0	0	2	0	0	0	0	No
705	Bad	1	14	11	21,75	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
706	Bad	1	14	12	17,60	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
707	Bad	1	21	11	34,67	1	0	0	0	0	0	0	0	0	1	0	0	0	0	No
708	Bad	1	16	11	30,33	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
709	Bad	1	11	7	34,50	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
710	Bad	1	11	6	30	0	0	0	0	0	0	0	0	0	1	0	0	0	1	No

711	Bad	1	11	7	32	0	0	0	0	0	0	0	0	0	1	0	0	0	1	No
712	Bad	1	9	8	30	0	0	0	0	0	0	0	0	0	0	1	0	0	1	No
713	Bad	1	12	7	22,33	0	0	0	0	0	0	0	0	0	1	0	0	1	0	No
714	Bad	1	14	7	27,33	0	0	0	0	0	2	0	0	0	0	1	0	0	0	No
715	Bad	1	9	5	29,50	0	0	0	0	0	0	0	0	0	0	1	0	0	0	No
716	Bad	1	23	10	62	0	0	0	1	0	0	0	0	0	1	0	1	0	0	No
717	Bad	1	28	13	76,50	0	0	0	1	0	0	0	0	0	1	0	1	0	0	No
718	Bad	1	36	18	106,50	1	0	0	2	0	0	0	0	0	1	0	1	0	1	No
719	Bad	1	34	17	99	0	0	0	2	0	0	0	0	0	1	0	1	1	1	No
720	Bad	1	33	17	49,75	0	0	0	1	0	2	0	0	0	1	0	1	1	2	No
721	Bad	1	31	5	25,50	0	0	0	2	0	0	0	0	0	2	0	0	1	0	No
722	Bad	1	47	21	28,67	2	0	1	0	0	7	1	0	0	1	0	0	1	2	No
723	Bad	1	27	15	77,50	1	0	0	0	0	0	0	0	0	1	0	0	0	2	No
724	Bad	1	13	11	20,25	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
725	Bad	1	23	11	61,50	0	0	0	1	0	0	0	0	0	1	0	1	0	0	No
726	Bad	1	28	13	76	0	0	0	1	0	0	0	0	0	1	0	1	0	0	No
727	Bad	1	11	8	34	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
728	Bad	1	11	7	29,50	0	0	0	0	0	0	0	0	0	1	0	0	0	1	No
729	Bad	1	11	8	31,50	0	0	0	0	0	0	0	0	0	1	0	0	0	1	No
730	Bad	1	9	9	29,50	0	0	0	0	0	0	0	0	0	0	1	0	0	1	No
731	Bad	1	12	8	22	0	0	0	0	0	0	0	0	0	1	0	0	1	0	No
732	Bad	1	14	8	27	0	0	0	0	0	2	0	0	0	0	1	0	0	0	No
733	Bad	1	9	6	29	0	0	0	0	0	0	0	0	0	0	1	0	0	0	No
734	Bad	1	12	14	21	0	0	0	0	0	0	0	0	0	1	0	0	0	1	No
735	Bad	1	13	9	39	0	0	0	0	0	0	0	0	0	0	1	0	0	0	No
736	Bad	1	12	11	20	0	0	0	0	0	0	0	0	0	1	0	0	0	2	No
737	Bad	1	11	16	24,67	0	0	0	0	0	2	0	0	0	1	0	0	0	0	No
738	Bad	1	13	12	27	1	0	0	0	0	2	0	0	0	1	0	0	0	0	No

739	Bad	1	11	13	23	0	0	0	0	0	2	0	0	0	1	0	0	1	0	No
740	Bad	1	17	14	27,75	2	0	0	0	0	0	0	0	0	1	0	0	2	0	No
741	Bad	1	13	12	41	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
742	Bad	1	33	23	30,38	0	0	0	0	0	0	0	0	0	1	0	1	1	3	No
743	Bad	1	7	14	13,25	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
744	Bad	1	9	14	12,60	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
745	Bad	1	10	21	20,25	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
746	Bad	1	9	8	12,50	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
747	Bad	1	8	10	12,25	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
748	Bad	1	7	6	9,75	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
749	Bad	1	8	8	12,75	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
750	Bad	1	10	8	15,75	0	0	0	0	0	0	0	0	0	1	0	0	2	1	No
751	Bad	1	9	12	15,75	0	0	0	0	0	0	0	0	0	1	0	0	0	1	No
752	Bad	1	13	11	21,75	0	0	0	0	0	0	0	0	0	1	0	0	0	1	No
753	Bad	1	14	13	24,25	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
754	Bad	1	22	13	27,20	0	0	0	0	1	0	0	0	2	1	0	0	0	0	No
755	Bad	1	49	21	65,50	1	0	0	0	1	2	0	0	0	1	0	1	1	3	Yes
756	Bad	2	22	14	36,25	0	0	0	0	0	0	0	0	0	1	0	0	1	2	No
757	Bad	2	22	14	35,75	0	0	0	0	0	0	0	0	0	1	0	0	1	0	No
758	Bad	2	28	16	36,60	2	0	0	0	1	0	0	0	0	1	0	0	0	0	No
759	Bad	2	42	21	65,50	2	0	0	0	1	0	0	0	0	1	0	0	1	2	No
760	Bad	2	32	18	52,75	0	0	0	0	0	2	0	0	0	1	0	2	2	1	No
761	Bad	1	14	15	52,50	0	0	0	0	0	0	0	0	1	1	0	0	0	0	No
762	Bad	1	13	16	51	0	0	0	0	0	0	0	0	1	1	0	0	0	0	No
763	Bad	1	11	11	36,50	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
764	Bad	1	14	13	48	1	0	0	0	0	0	0	0	0	1	0	1	0	1	No
765	Bad	1	11	11	37,50	0	0	0	0	1	0	0	0	0	1	0	0	0	0	No
766	Bad	1	12	12	42	0	0	0	0	1	0	0	0	0	1	0	0	0	0	No

767	Bad	1	14	8	27,67	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
768	Bad	1	9	10	31,50	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
769	Bad	1	9	12	32	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
770	Bad	1	9	13	31,50	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
771	Bad	1	21	15	48	1	1	0	0	0	0	0	0	0	1	0	1	0	1	No
772	Bad	1	16	10	52	1	0	0	0	0	0	0	0	0	1	0	0	0	0	No
773	Bad	1	18	10	15,86	0	0	0	0	0	2	0	0	0	1	0	0	0	0	No
774	Bad	1	6	6	16,33	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
775	Bad	1	6	6	19	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
776	Bad	1	6	6	20,50	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
777	Bad	1	6	6	19,50	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
778	Bad	1	14	5	24	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
779	Bad	1	6	6	21	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
780	Bad	1	6	6	20	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
781	Bad	1	12	10	12,83	0	0	0	0	0	2	0	0	0	1	0	0	0	0	No
782	Bad	1	8	5	25,50	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
783	Bad	1	9	6	30,50	0	0	0	0	0	0	0	0	0	0	1	0	0	0	No
784	Bad	1	19	12	61,50	2	0	0	0	0	0	0	0	0	1	0	0	0	0	No
785	Bad	1	12	14	48,50	1	0	0	0	0	0	0	0	0	0	1	0	0	1	No
786	Bad	1	16	11	49,50	1	0	0	0	0	0	0	0	0	1	0	0	0	0	No
787	Bad	1	6	8	21,50	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
788	Bad	1	16	7	20,50	1	0	0	1	3	0	0	0	0	1	0	0	0	0	No
789	Bad	1	15	12	54,50	0	0	0	0	0	0	0	0	0	1	0	0	0	1	No
790	Bad	1	12	11	42	0	0	0	0	0	0	0	0	0	1	0	0	0	1	No
791	Bad	1	14	8	45	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
792	Bad	1	16	11	52	0	0	0	0	1	0	0	0	0	1	0	0	1	0	No
793	Bad	1	20	11	60,50	2	0	0	0	1	0	0	0	0	1	0	1	1	0	No
794	Bad	1	18	13	40	2	0	0	0	0	0	0	0	0	1	0	0	1	0	No

795	Bad	1	16	16	66	1	0	0	0	1	0	0	0	0	1	0	0	1	0	No
796	Bad	1	32	16	96	2	0	0	1	0	0	0	0	0	1	0	0	1	1	No
797	Bad	1	33	16	93	3	0	0	3	0	0	0	0	0	1	0	0	1	1	No
798	Bad	1	32	15	93	2	0	0	1	0	0	0	0	0	1	0	0	1	1	No
799	Bad	1	11	17	22,25	1	0	0	0	0	0	0	0	0	0	0	0	1	0	No
800	Bad	1	15	14	50	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
801	Bad	1	35	23	127	1	0	0	0	0	0	0	0	0	1	0	0	0	3	No
802	Bad	1	21	11	27,50	0	0	0	1	0	0	0	0	0	1	0	0	0	0	No
803	Bad	1	21	10	27,50	0	0	0	1	0	0	0	0	0	1	0	0	0	0	No
804	Bad	1	29	14	53,67	1	0	0	1	0	0	0	0	0	1	0	0	0	0	No
805	Bad	1	18	12	40,67	0	0	0	0	0	2	0	0	0	1	0	0	0	0	No
806	Bad	1	21	11	25,60	0	0	0	0	0	2	0	0	0	1	0	0	0	0	No
807	Bad	1	27	15	59	0	0	0	0	0	0	0	0	0	1	0	0	0	1	No
808	Bad	1	28	14	85,50	0	0	0	1	0	0	0	0	0	1	0	1	0	0	No
809	Bad	1	32	16	96,50	0	0	0	1	0	0	0	0	0	1	0	1	0	0	No
810	Bad	1	40	20	125,50	1	0	0	2	0	0	0	0	0	1	0	1	0	1	No
811	Bad	1	38	19	119	0	0	0	2	0	0	0	0	0	1	0	1	1	1	No
812	Bad	1	37	19	59,25	0	0	0	1	0	2	0	0	0	1	0	1	1	2	No
813	Bad	1	24	12	69	0	0	0	0	0	0	0	0	0	2	0	0	0	0	No
814	Bad	1	18	8	10,78	0	0	0	0	0	3	0	0	0	1	0	0	0	0	No
815	Bad	1	12	8	34,50	1	0	0	0	0	0	0	0	0	1	0	0	0	0	No
816	Bad	1	18	10	51,50	1	1	0	1	0	0	0	0	0	1	0	2	0	1	No
817	Bad	1	10	6	28,50	0	0	0	0	0	0	0	0	0	1	0	0	1	0	No
818	Bad	1	15	8	29	0	0	0	0	0	2	0	0	0	0	1	0	0	0	No
819	Bad	1	10	6	32	0	0	0	0	0	0	0	0	0	0	1	0	0	0	No
820	Bad	1	24	12	70	0	0	0	0	0	0	0	0	0	2	0	0	0	0	No
821	Bad	1	17	8	10,11	0	0	0	0	0	3	0	0	0	0	0	0	0	0	No
822	Bad	1	12	8	34,50	1	0	0	0	0	0	0	0	0	1	0	0	0	0	No

823	Bad	1	18	10	51,50	1	1	0	1	0	0	0	0	0	1	0	2	0	1	No
824	Bad	1	10	6	28,50	0	0	0	0	0	0	0	0	0	1	0	0	1	0	No
825	Bad	1	15	8	29	0	0	0	0	0	2	0	0	0	0	1	0	0	0	No
826	Bad	1	10	6	32	0	0	0	0	0	0	0	0	0	0	1	0	0	0	No
827	Bad	1	25	12	37	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
828	Bad	1	30	13	12,54	0	0	0	0	0	1	0	0	0	0	0	0	0	0	No
829	Bad	1	12	8	35	1	0	0	0	0	0	0	0	0	1	0	0	0	0	No
830	Bad	1	18	10	52	1	1	0	1	0	0	0	0	0	1	0	2	0	1	No
831	Bad	1	10	6	29	0	0	0	0	0	0	0	0	0	1	0	0	1	0	No
832	Bad	1	15	8	29,33	0	0	0	0	0	2	0	0	0	0	1	0	0	0	No
833	Bad	1	10	6	32,50	0	0	0	0	0	0	0	0	0	0	1	0	0	0	No
834	Bad	1	21	7	20	0	1	0	0	1	4	0	0	0	1	0	0	0	0	No
835	Bad	1	27	6	45	0	0	0	0	0	0	0	0	1	2	0	0	0	0	No
836	Bad	1	24	9	29,25	0	1	0	0	0	2	0	0	0	1	0	0	0	0	No
837	Bad	1	20	10	28,25	0	0	0	0	0	2	0	0	0	1	0	0	0	1	No
838	Bad	4	37	21	21,09	1	0	0	0	0	5	0	0	0	3	0	1	0	4	No
839	Bad	1	10	5	13,75	0	0	0	0	0	0	0	0	0	1	0	0	1	0	No
840	Bad	1	15	7	28,33	0	0	0	0	0	2	0	0	0	0	1	0	0	0	No
841	Bad	1	10	5	31	0	0	0	0	0	0	0	0	0	0	1	0	0	0	No
842	Bad	1	46	20	132,50	1	0	0	0	1	0	0	0	0	1	0	1	1	3	Yes
843	Bad	1	25	15	46	0	0	0	0	0	0	0	0	0	1	0	0	1	2	No
844	Bad	1	25	14	45,25	0	0	0	0	0	0	0	0	0	1	0	0	1	0	No
845	Bad	1	31	16	55,50	2	0	0	0	1	0	0	0	0	1	0	0	0	0	No
846	Bad	1	39	19	87,67	2	0	0	0	1	0	0	0	0	1	0	0	1	1	No
847	Bad	1	35	18	83,33	0	0	0	0	0	2	0	0	0	1	0	2	2	1	No
848	Bad	1	17	15	71	0	0	0	0	0	0	0	0	1	1	0	0	0	0	No
849	Bad	1	17	14	70	0	0	0	0	0	0	0	0	1	1	0	0	0	0	No
850	Bad	1	14	11	55	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No

851	Bad	1	17	13	66,50	1	0	0	0	0	0	0	0	0	1	0	1	0	1	No
852	Bad	1	14	11	56	0	0	0	0	1	0	0	0	0	1	0	0	0	0	No
853	Bad	1	15	12	60,50	0	0	0	0	1	0	0	0	0	1	0	0	0	0	No
854	Bad	1	17	9	40	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
855	Bad	1	12	10	50	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
856	Bad	1	12	11	50,50	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
857	Bad	1	12	12	50	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
858	Bad	1	24	15	60,33	1	1	0	0	0	0	0	0	0	1	0	1	0	1	No
859	Bad	1	19	10	71	1	0	0	0	0	0	0	0	0	1	0	0	0	0	No
860	Bad	1	21	11	21,14	0	0	0	0	0	2	0	0	0	1	0	0	0	0	No
861	Bad	1	9	6	28,67	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
862	Bad	1	9	6	37,50	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
863	Bad	1	9	6	39	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
864	Bad	1	9	6	38	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
865	Bad	1	17	6	36,33	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
866	Bad	1	9	6	39,50	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
867	Bad	1	9	6	38,50	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
868	Bad	1	15	10	19	0	0	0	0	0	2	0	0	0	1	0	0	0	0	No
869	Bad	1	11	6	44	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
870	Bad	1	12	7	49	0	0	0	0	0	0	0	0	0	0	1	0	0	0	No
871	Bad	1	12	7	27,33	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
872	Bad	1	22	12	80,50	2	0	0	0	0	0	0	0	0	1	0	0	0	0	No
873	Bad	1	15	13	67,50	1	0	0	0	0	0	0	0	0	0	1	0	0	1	No
874	Bad	1	19	11	68,50	1	0	0	0	0	0	0	0	0	1	0	0	0	0	No
875	Bad	1	13	8	34	0	1	0	0	0	1	0	0	0	1	0	0	0	0	No
876	Bad	1	18	8	29,25	1	0	0	1	3	0	0	0	0	1	0	0	0	0	No
877	Bad	1	17	13	68,50	0	0	0	0	0	0	0	0	0	1	0	0	0	1	No
878	Bad	1	13	11	56	0	0	0	0	0	0	0	0	0	1	0	0	0	1	No

879	Bad	1	12	8	51	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
880	Bad	1	19	12	71	0	0	0	0	1	0	0	0	0	1	0	0	1	0	No
881	Bad	1	22	12	79	2	0	0	0	1	0	0	0	0	1	0	1	1	0	No
882	Bad	1	20	14	52,33	2	0	0	0	0	0	0	0	0	1	0	0	1	0	No
883	Bad	1	19	16	85	1	0	0	0	1	0	0	0	0	1	0	0	1	0	No
884	Bad	1	35	17	115	2	0	0	1	0	0	0	0	0	1	0	0	1	1	No
885	Bad	1	37	16	112	3	0	0	3	0	0	0	0	0	1	0	0	1	1	No
886	Bad	1	36	15	112,50	2	0	0	1	0	0	0	0	0	1	0	0	1	1	No
887	Bad	1	12	16	23,50	1	0	0	0	0	0	0	0	0	1	0	0	1	0	No
888	Bad	1	17	10	62	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
889	Bad	1	35	21	132,50	1	0	0	0	0	0	0	0	0	1	0	0	0	3	No
890	Bad	1	8	13	25,50	0	0	0	0	0	0	0	0	0	1	0	0	0	0	No
891	Bad	12	230	27	40,58	13	0	1	2	0	22	1	0	1	1	2	1	28	21	No
892	Bad	1	17	15	58	2	0	0	1	0	0	0	0	0	0	0	0	2	2	No
893	Bad	3	78	13	84,67	3	0	4	0	0	0	0	0	0	0	2	0	10	4	No
894	Bad	1	31	21	103	1	0	0	0	0	0	0	0	0	1	0	0	4	1	No
895	Bad	3	75	22	158,67	3	1	3	0	0	0	1	0	0	2	1	1	8	4	No
896	Bad	7	76	15	45,75	1	0	0	0	1	0	0	0	1	0	0	0	20	3	No
897	Bad	1	17	16	55	0	0	0	0	0	0	1	0	0	0	1	0	2	2	No
898	Bad	1	36	21	109,50	4	0	0	0	0	0	2	0	0	0	1	1	3	1	Yes
899	Bad	11	83	19	35,63	2	0	2	0	1	10	1	0	0	1	2	0	12	0	No
900	Bad	8	104	14	31,67	2	0	3	2	2	9	1	0	1	2	1	0	11	0	No
901	Bad	1	25	16	76	0	0	0	0	0	0	0	0	0	1	0	0	4	1	No
902	Bad	16	175	28	132,38	7	2	3	0	0	0	0	0	0	3	1	1	18	4	No
903	Bad	1	30	19	66,67	2	0	0	1	0	0	0	0	0	0	1	0	3	2	No
904	Bad	11	201	24	45,81	10	0	1	2	0	20	1	0	2	1	2	1	26	19	No
905	Bad	1	17	15	58,50	2	0	0	1	0	0	0	0	0	0	0	0	2	2	No
906	Bad	3	77	14	84,17	3	0	4	0	0	0	0	0	0	0	2	0	10	4	No

907	Bad	1	29	21	99,50	1	0	0	0	0	0	0	0	0	1	0	0	4	1	No
908	Bad	3	65	15	103,75	3	1	3	1	0	0	1	0	0	2	1	1	7	4	No
909	Bad	5	62	15	43,80	1	0	0	0	1	0	0	0	1	0	0	0	17	3	No
910	Bad	1	17	16	55,50	0	0	0	0	0	0	1	0	0	0	1	0	2	2	No
911	Bad	9	77	17	43	2	0	1	0	1	6	1	0	0	1	2	0	12	0	No
912	Bad	1	25	16	76	0	0	0	0	0	0	0	0	0	1	0	0	4	1	No
913	Bad	16	159	26	119,25	6	2	2	0	0	0	0	0	0	3	1	1	17	3	No
914	Bad	1	29	19	98	2	0	0	1	0	0	0	0	0	0	1	0	3	2	No
915	Bad	2	60	16	91	2	0	1	1	0	0	0	0	0	2	0	0	10	3	No
916	Bad	1	35	21	79,67	3	0	0	0	0	0	0	0	1	0	1	0	7	2	No
917	Bad	1	36	22	124	3	1	0	0	0	0	0	0	1	0	1	0	5	2	No
918	Bad	1	21	13	66,50	0	0	0	0	0	0	0	0	0	0	1	0	5	0	No
919	Bad	1	51	12	72,25	4	0	1	1	0	0	0	0	0	2	0	0	6	3	No
920	Bad	2	57	18	120,33	2	0	0	2	1	0	1	0	0	2	1	1	10	6	No
921	Bad	9	62	32	40,10	4	0	0	0	1	6	0	0	0	2	0	0	15	0	No
922	Bad	4	53	18	91	2	0	2	1	0	0	0	0	0	2	0	0	7	1	No
923	Bad	13	94	17	40,67	5	0	0	4	0	8	1	0	0	3	1	0	18	2	No
924	Bad	1	21	15	64	1	0	0	0	0	0	1	0	0	0	1	0	2	0	No
925	Bad	1	22	14	70,50	2	0	0	0	0	0	0	0	0	1	0	0	1	0	No
926	Bad	2	49	13	74,25	3	0	0	1	1	0	0	0	0	2	0	1	8	0	No
927	Bad	1	20	12	31,75	0	0	1	1	0	2	0	0	2	1	0	0	2	0	No
928	Bad	16	79	22	78,86	5	1	0	0	0	2	0	0	0	1	2	0	11	5	No
929	Bad	1	15	13	50,50	1	0	0	0	0	0	0	0	0	1	0	0	2	0	No
930	Bad	9	53	16	67,40	4	0	0	1	0	2	0	0	0	2	0	1	8	4	No
931	Bad	2	58	17	120,33	2	0	0	2	1	0	1	0	0	2	1	1	9	6	No
932	Bad	7	40	22	24,50	3	0	0	0	0	6	0	0	0	1	0	1	9	0	No
933	Bad	2	48	22	144,50	4	0	0	0	0	0	0	0	0	2	1	0	10	0	No
934	Bad	1	27	17	58,33	1	0	0	1	0	0	0	0	0	0	2	1	3	1	No

935	Bad	1	20	11	59,50	0	0	0	1	0	0	0	0	0	1	0	1	2	0	No
936	Bad	1	24	13	66,50	1	0	0	0	0	0	0	0	0	1	0	1	2	0	No
937	Bad	1	19	17	63,50	0	0	0	0	0	0	0	0	0	1	0	0	3	0	No
938	Bad	1	19	17	70	1	1	0	0	0	0	0	0	0	1	0	1	2	1	No
939	Bad	1	25	11	53	1	1	0	0	0	0	0	0	0	1	0	1	3	2	No
940	Bad	1	18	16	65	0	0	0	0	0	0	0	0	0	1	0	0	2	1	No
941	Bad	1	20	16	66	1	0	0	0	1	0	0	0	0	2	0	0	0	0	No
942	Bad	1	18	14	54,50	0	0	0	0	0	0	0	0	0	1	0	1	4	1	No
943	Bad	1	15	15	49,50	1	0	0	0	0	0	0	0	0	1	0	0	2	1	No
944	Bad	2	48	9	31,78	0	0	0	0	3	2	0	0	0	2	1	1	1	2	No
945	Bad	1	15	9	30,33	0	1	0	0	0	0	0	0	0	1	0	1	3	1	No
946	Bad	1	12	11	33,50	0	0	0	0	0	0	0	0	0	1	0	1	2	2	No
947	Bad	1	18	10	47	0	0	0	0	1	0	0	0	0	1	0	1	1	1	No
948	Bad	1	28	15	32	1	0	0	0	1	2	0	0	0	1	1	1	1	3	No
949	Bad	1	25	15	49	0	0	0	0	0	2	1	0	0	0	1	1	4	3	No
950	Bad	1	20	14	61,50	0	0	0	0	0	0	0	0	0	0	1	1	3	1	No
951	Bad	1	23	13	32,25	2	0	0	0	0	2	0	0	0	0	1	1	2	1	No
952	Bad	1	14	11	40	0	0	0	2	0	0	0	0	0	1	0	0	1	1	No
953	Bad	1	25	17	54	0	0	0	0	0	0	0	0	0	1	0	1	2	0	No
954	Bad	1	13	10	41	0	0	0	0	1	0	0	0	0	1	0	0	0	0	No
955	Bad	1	23	16	48	0	0	2	0	1	0	0	0	0	1	0	1	2	0	No
956	Bad	1	16	15	55	0	0	0	1	0	0	0	0	0	0	1	0	2	1	No
957	Bad	1	13	9	39,50	0	0	0	0	0	0	0	0	0	0	1	1	1	1	No
958	Bad	1	26	16	38,75	3	0	0	0	0	2	0	0	0	0	1	1	1	3	No
959	Bad	1	13	11	39	0	0	0	2	0	0	0	0	0	1	0	0	1	1	No
960	Bad	1	12	14	29,67	1	0	0	0	0	0	0	0	0	0	1	1	0	1	Yes
961	Bad	1	19	14	33	2	0	0	0	1	0	1	0	0	1	1	1	0	0	No
962	Bad	1	10	10	28	0	0	0	0	0	0	0	0	0	0	1	1	1	0	No

963	Bad	1	21	12	28,75	2	0	0	0	0	2	0	0	0	0	1	1	2	1	No
964	Bad	1	19	17	63,50	0	0	0	0	0	0	0	0	0	1	0	0	3	0	No
965	Bad	1	21	18	77,50	2	1	0	0	0	0	0	0	0	1	0	1	3	1	No
966	Bad	1	25	11	53	1	1	0	0	0	0	0	0	0	1	0	1	3	2	No
967	Bad	1	18	16	65	0	0	0	0	0	0	0	0	0	1	0	0	2	1	No
968	Bad	1	13	14	44,50	0	0	0	0	1	0	0	0	0	1	0	0	0	0	No
969	Bad	1	18	14	54,50	0	0	0	0	0	0	0	0	0	1	0	1	4	1	No
970	Bad	1	15	15	50	1	0	0	0	0	0	0	0	0	1	0	0	2	1	No
971	Bad	2	48	9	31,78	0	0	0	0	3	2	0	0	0	2	1	1	1	2	No
972	Bad	1	22	10	45	1	0	0	0	1	0	1	0	0	0	2	0	5	0	No
973	Bad	1	14	12	38,50	0	0	0	0	0	0	0	0	0	1	0	1	2	1	No
974	Bad	1	18	10	48	0	0	0	0	1	0	0	0	0	1	0	1	1	1	No
975	Bad	1	28	15	32	1	0	0	0	1	2	0	0	0	1	1	1	1	3	No
976	Bad	1	25	15	49	0	0	0	0	0	2	1	0	0	0	1	1	4	3	No
977	Bad	1	20	14	61,50	0	0	0	0	0	0	0	0	0	0	1	1	3	1	No
978	Bad	1	23	13	31,75	2	0	0	0	0	2	0	0	0	0	1	1	2	1	No
979	Bad	1	14	11	40	0	0	0	2	0	0	0	0	0	1	0	0	1	1	No
980	Bad	1	25	18	82,50	0	0	0	0	0	0	0	0	0	1	0	1	1	0	No
981	Bad	1	13	10	41	0	0	0	0	1	0	0	0	0	1	0	0	0	0	No
982	Bad	1	26	14	50,67	0	0	0	0	1	0	1	0	0	1	1	1	2	0	No
983	Bad	1	12	15	44,50	0	0	0	0	0	0	0	0	0	1	0	0	1	0	No
984	Bad	1	12	14	29,67	1	0	0	0	0	0	0	0	0	0	1	1	0	1	Yes
985	Bad	1	19	14	33	2	0	0	0	1	0	1	0	0	1	1	1	0	0	No
986	Bad	1	10	10	28	0	0	0	0	0	0	0	0	0	0	1	1	1	0	No
987	Bad	1	21	12	28,75	2	0	0	0	0	2	0	0	0	0	1	1	2	1	No
988	Bad	1	19	12	54,50	1	0	0	0	0	0	0	0	0	1	0	1	3	1	Yes
989	Bad	1	13	9	24,67	1	0	0	0	1	0	0	0	0	0	0	0	2	0	No
990	Bad	2	27	13	56,33	2	0	0	0	0	0	0	0	0	2	0	1	3	2	No

991	Bad	3	48	9	36	0	0	0	0	3	2	0	0	0	2	1	1	1	2	No
992	Bad	1	23	10	47,33	1	0	0	0	1	0	1	0	0	0	2	0	5	0	No
993	Bad	1	13	11	36	0	0	0	0	0	0	0	0	0	1	0	0	1	0	No
994	Bad	1	18	10	47	0	0	0	0	1	0	0	0	0	1	0	1	1	1	No
995	Bad	1	29	15	41,25	1	0	0	0	1	2	0	0	0	1	1	2	1	3	No
996	Bad	1	25	15	49	0	0	0	0	0	2	1	0	0	0	1	1	4	3	No
997	Bad	1	17	13	53,50	0	0	0	0	0	0	0	0	0	0	1	1	2	1	No
998	Bad	1	26	14	23,17	3	0	0	0	0	4	0	0	0	0	1	1	3	1	No
999	Bad	1	14	11	40	0	0	0	2	0	0	0	0	0	1	0	0	1	1	No
1000	Bad	1	10	10	28	0	0	0	0	0	0	0	0	0	0	1	1	1	0	No
1001	Bad	1	24	13	21,17	3	0	0	0	0	4	0	0	0	0	1	1	3	1	No
1002	Bad	1	15	22	62,50	1	0	0	0	1	0	0	0	0	1	0	0	2	0	No
1003	Bad	1	22	22	56,33	2	0	0	0	0	0	0	0	0	1	0	0	3	1	No
1004	Bad	4	26	24	33,71	0	0	0	0	0	0	0	0	0	0	0	1	2	0	No
1005	Bad	1	13	25	60	1	0	0	0	1	0	0	0	0	1	0	0	2	0	No
1006	Bad	1	20	23	54,67	2	0	0	0	0	0	0	0	0	1	0	0	3	1	No
1007	Bad	5	30	24	32,33	0	0	0	0	0	0	0	0	0	0	0	1	4	0	No
1008	Bad	1	12	21	47	0	0	0	0	1	0	0	0	0	1	0	0	2	0	No
1009	Bad	1	23	20	30,17	1	0	0	0	0	0	0	0	0	0	0	1	5	2	No
1010	Bad	1	14	19	55	0	0	0	0	0	0	0	0	0	1	0	0	4	0	No
1011	Bad	1	23	13	52,67	0	0	0	1	0	0	0	0	0	1	1	0	4	0	No
1012	Bad	1	19	19	50,33	1	0	0	0	1	0	0	0	0	1	0	0	6	2	No
1013	Bad	1	13	17	50,50	1	0	0	0	0	0	0	0	0	1	0	0	2	3	No
1014	Bad	1	19	18	24,67	1	1	0	0	0	3	0	0	0	1	0	0	6	2	No
1015	Bad	1	18	20	48,67	2	0	0	0	0	0	0	0	0	0	1	0	4	3	No
1016	Bad	1	19	16	31	0	0	0	0	1	2	0	0	0	0	0	0	2	0	No
1017	Bad	1	24	17	50,67	0	0	0	1	1	0	0	0	0	0	0	0	6	1	No
1018	Bad	1	27	19	44	1	0	1	1	0	0	0	0	0	2	0	1	7	0	No

1019	Bad	1	19	13	38	0	0	0	0	0	0	0	0	0	1	0	1	2	1	No
1020	Bad	1	22	15	45,67	1	0	1	0	1	0	0	0	0	1	0	1	2	2	Yes
1021	Bad	1	9	8	26	0	0	0	0	0	0	0	0	0	1	0	0	1	0	No
1022	Bad	1	13	10	25	0	0	0	0	3	0	0	0	0	0	0	0	0	0	No
1023	Bad	1	15	16	51	2	0	1	0	2	0	0	0	0	1	0	1	4	1	Yes
1024	Bad	1	27	16	48,75	1	0	0	0	0	0	0	1	0	1	0	2	8	2	Yes
1025	Bad	1	13	13	42	0	0	0	0	0	0	1	0	0	0	1	1	2	0	No
1026	Bad	1	45	10	32,25	3	1	0	0	1	0	1	0	0	1	2	2	6	1	Yes
1027	Bad	1	10	11	32	1	0	0	0	0	0	0	0	0	1	0	0	4	0	No
1028	Bad	1	11	8	29,50	0	0	0	0	0	0	0	0	0	1	0	1	1	0	No
1029	Bad	1	18	11	50,50	0	0	0	0	1	0	0	0	0	0	0	0	3	1	No
1030	Bad	1	19	13	55	1	0	0	0	0	0	0	0	0	0	0	0	2	0	No
1031	Bad	1	18	15	54,50	0	0	0	0	0	0	0	0	0	0	0	0	2	0	No
1032	Bad	1	23	13	39	0	0	0	0	1	2	1	0	0	0	1	0	1	0	No
1033	Bad	1	14	11	38,50	0	0	0	0	0	0	0	0	0	0	0	0	2	0	No
1034	Bad	1	14	18	50,50	1	0	0	0	0	0	1	0	0	0	1	1	3	1	Yes
1035	Bad	1	17	17	57,50	2	1	0	0	0	0	1	0	0	0	1	0	1	1	No

																		Patterns Count ('No')	938	
																		Patterns Count ('Yes')	97	
Count 0		**0**	**0**	**0**	**0**	**554**	**939**	**920**	**839**	**782**	**729**	**958**	**1022**	**987**	**174**	**831**	**760**	**285**	**447**	
Min		**1**	**6**	**2**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**No**
Max		**19**	**230**	**32**	**158,67**	**13**	**2**	**4**	**5**	**4**	**22**	**2**	**1**	**5**	**6**	**2**	**7**	**28**	**21**	**Yes**