11705/JISBD/2018/029

Expansión cuantitativa del método MoSCoW para la priorización de requisitos

José del Sagrado, Isabel María del Águila y Alfonso Bosch

Departamento de Informática, Universidad of Almería,
Crtra. de la Playa s/n, 04120 Almería, Spain
{jsagrado,imaguila}@ual.es

Resumen La priorización de los requisitos a ser incluidos en el producto final es un complejo proceso de decisión multicriterio que suele implicar llegar al equilibrio entre el beneficio para el negocio de cada requisito y el consumo de recursos. Existen distintos factores y dimensiones a considerar en la priorización de requisitos, muchos de ellos de carácter cualitativo. Sin embargo, algunos métodos también han utilizado las propiedades cuantitativas estimadas, siendo muchas de estas soluciones del ámbito de las técnicas de optimización. En este trabajo se propone y estudia la validez de un algoritmo de agrupamiento muy conocido, k-medias, junto con el método subjetivo más ampliamente utilizado, el método MoSCoW, para la priorización de requisitos. Los resultados experimentales, sobre dos casos de 20 y 100 requisitos con interdependencias, muestran la validez de la propuesta en la identificación de los requisitos que dan mayor valor al sistema a construir y que aseguran el mayor beneficio en el proyecto, siendo necesario incorporar requisitos para cumplir las restricciones por interdependencias.

Keywords: Ingeniería de requisitos, priorización de requisitos, MoSCoW, k-medias

1 Motivación

Uno de los mayores problemas, cuando se desarrollan sistemas complejos, es la definición, análisis y gestión de los requisitos. Estas etapas relativas a los requisitos son tareas intensivas en conocimiento, que en muchos casos se sustentan con el esfuerzo y la experiencia humana. Los requisitos son fundamentales para el éxito de un proyecto de desarrollo de software, ya que recopilan las necesidades o condiciones que debe cumplir el producto en construcción. Son la base del resto del proceso y el cemento que mantiene unidas las demás etapas del proyecto.

La priorización de requisitos es un campo en el que estas tareas de toma de decisiones son una parte fundamental. Se trata de un proceso complejo de decisión multicriterio, que en muchos casos implica encontrar el equilibrio entre el beneficio para el negocio de cada requisito y su coste e implicaciones sobre la futura evolución del producto. Se presenta cuando las necesidades de los clientes deben ajustarse a la capacidad de trabajo del equipo de desarrollo y se requiere

negociar acerca del alcance del software. Este proceso permite a los gestores del proyecto centrar su atención en aquellas características del producto que añadirán más valor al beneficio global del proyecto.

Se han desarrollado numerosas técnicas de priorización que utilizan distinta información sobre los requisitos. Algunas utilizan un sólo atributo para priorizar y otras combinan varios. Unas tienen en cuenta distintos puntos de vista y otras sólo consideran la visión de un cliente. Varias manejan las dependencias entre requisitos y otras no. Algunos métodos se usan para establecer el orden de implementación de los requisitos, mientras que otros se usan para seleccionar qué requisitos incluir y cuáles descartar. De entre ellos se pueden nombrar el proceso analítico jerárquico (analytical hierarchy process- AHP), análisis de coste valor, priorización con árboles B, voto acumulado, ranking, método MoSCoW, juegos de planificación, top ten, modelo win-win [1, 17].

Uno de los métodos más ampliamente aplicados es MoSCoW que se basa en el hecho de que, aunque todos los requisitos se consideren importantes, es fundamental destacar aquellos requisitos vitales que aportan un mayor valor al negocio y que son considerados obligatorios, de forma que el producto o servicio no se puede poner en producción si incumple alguno de estos requisitos. La técnica MoSCoW propone separar los requisitos en cuatro grandes categorías cada una con un significado intrínseco (Must have, Should have, Could have y Won't have), basandose en el juicio experto y en los atributos o propiedades de los requisitos.

La priorización de requisitos también ha sido formulada como un problema de optimización multiobjetivo donde aparecen objetivos contradictorios. La idea es combinar las métodos computacionales, que utilizan atributos de los requisitos estimados cuantitativamente, con la experiencia de los expertos humanos para obtener mejores resultados. Se han aplicado técnicas de diversa naturaleza como son la satisfacción de restricciones y programación lineal [7, 9] o métodos del ámbito de la ingeniería del software basada en búsqueda [14, 2].

Para tener éxito en la aplicación de cualquiera de las técnicas se require un conocimiento detallado del dominio y buenas técnicas de cuantificación y estimación de las propiedades de los requisitos. Sin embargo, suele ser difícil obtener información fiable que pueda servir de base para la toma de decisiones y en la mayoría de los casos las prioridades son relativas [6]. Esto nos lleva a plantear propuestas combinadas, que exploten la gestión empírica del proyecto junto a métodos más cercanos a los clientes debido al significado intrínseco, como puede ser MoSCoW.

El principal objetivo de este trabajo, es proponer una técnica que mejore la priorización de requisitos y, como consecuencia, obtenga el conjunto de requisitos fundamentales mediante la combinación del método MoSCoW con el algoritmo de agrupamiento k-medias. El primero es de carácter puramente cualitativo y el segundo trabaja con las estimaciones de beneficio y esfuerzo para cada requisito individual.

El trabajo se estructura de la manera siguiente. En la sección 2 se tratan las distintas aproximaciones cuantitativas para representar el problema. Los

métodos MoSCoW, k-medias y la propuesta de combinación se incluyen en la sección 3, para en la sección 4 describir el estudio experimental realizado. La última sección incorpora las principales conclusiones del trabajo.

2 Representación cuantitativa del problema

La selección/priorización de requisitos utilizando las propiedades cuantitativas estimadas sobre los requisitos es un problema que ha sido tratado profusamente en la literatura. No sólo en la terminología, sino en la propia definición del problema existen variaciones de un autor a otro. Si bien todas ellas se basan en la aplicación del criterio coste-valor [11]. Se parte de un conjunto de requisitos, de los que se estima el coste o esfuerzo de desarrollo, y de una serie de clientes, que demandan la inclusión de ciertos requisitos en el producto a desarrollar. Los dos modelos principales son el de Bagnall [5] y el de Van den Akker [4].

Bagnall et al. [5] modeló el problema como un problema de optimización permitiendo relaciones de dependencia entre requisitos. En particular, considera que un requisito puede ser prerrequisito de otro, en cuyo caso la inclusión de este último exige la de aquél. Cuando los requisitos valorados por un cliente se seleccionan para ser incluidos, el beneficio aportado a la compañía se estima según un peso o importancia del cliente. El objetivo es seleccionar un subconjunto de clientes a satisfacer, de forma que se maximice la suma de los pesos de los clientes y el coste total estimado de todos los requisitos a incluir no supere un presupuesto prefijado.

Van den Akker et al. [4] propuso un modelo alternativo en el que la importancia se asigna individualmente a cada requisito, con lo que se dispone de una forma de estimar su beneficio individual. Este último modelo se ajusta mejor a los planteamientos actuales del desarrollo de soluciones software, pero permitiendo interdependencias entre los requisitos, que se han de respetar a la hora de proponer una solución. Estas dependencias se han representado mediante grafos en algunos trabajos [5, 12], si bien la versión que modela todos los tipos de interdependencias no se formuló mediante una representación en forma de grafo y matricial hasta el trabajo de del Sagrado et al. [16].

Greer and Ruhe [8, 15] tratan el problema de la planificación de versiones, importante en el contexto del desarrollo incremental de software. En este problema trata de distribuir los requisitos entre las distintas versiones planificadas teniendo en consideración las opiniones de los clientes, los recursos disponibles, las interdependencias y otros factores. El primer enfoque seguido consistió en resolverlo mediante un algoritmo genético, al que luego siguieron la aplicación de otras técnicas. Sin embargo, puesto que los métodos ágiles e incrementales promueven que en el desarrollo de los proyectos se deben realizar inspecciones frecuentes, lo habitual es que la lista de requisitos cambie drásticamente entre las distintas versiones o entregas. Razón por la que el modelo de Van den Akker también es aplicable en los desarrollos incrementales, una vez al principio de cada incremento, puesto que el conjunto de requisitos de partida habitualmente cambian entre incrementos.

El problema se formula partiendo de $R = \{r_1, \ldots, r_n\}$, el conjunto de requisitos propuestos por m clientes. Cada cliente i tiene distinta importancia para el proyecto, representada por un peso $w_i \in \mathbb{R}$. Todo $r_j \in R$ tiene un esfuerzo de desarrollo e_j y un valor $v_{ij} \in \mathbb{R}$ para cada cliente i. La satisfacción, s_j, o valor añadido por la selección de r_j, se calcula como la suma ponderada $s_j = \sum_{i=1}^{m} w_i \cdot v_{ij}$. Se consideran una serie de interdependencias funcionales que pueden surgir entre los requisitos: $r_i \Rightarrow r_j$ (r_j no puede ser incluido en el software sin incorporar r_i), $r_i \odot r_j$ (los requisitos r_i y r_j deben ser seleccionados conjuntamente), $r_i \oplus r_j$ (r_i y r_j no pueden ser incluidos conjuntamente en el software).

3 Priorización de requisitos

En el desarrollo de un producto software, y más si se aplican metodologías ágiles, se deben priorizar los requisitos, no sólo como un medio para identificar y filtrar los requisitos importantes, sino también para resolver conflictos y planificar las distintas versiones o entregas del producto. Se deben tomar decisiones complejas que requiren un conocimiento detallado del dominio y buenas técnicas de cuantificación y estimación de las propiedades de los requisitos empleando, habitualmente, criterios contradictorios.

Existen distintos factores y dimensiones a considerar en la priorización, algunos fijados por los clientes implicados (e.g. el valor de cada requisito, tiempo hasta la entrega), otros por el equipo de desarrollo (e.g. esfuerzo disponible, número de miembros del equipo), o por ambos (e.g. riesgo, volatilidad) e incluso por factores externos como el valor de mercado.

Una de las técnicas habituales para la priorización de requisitos es AHP que evalua un atributo sobre el que decidir (e.g. esfuerzo, valor, riesgo), utilizando una comparación por pares en una matriz $n \times n$, a partir de la cual se obtiene un orden en los requisitos respecto a ese atributo. Otra de las técnicas, el método de coste-valor, combina AHP para el esfuerzo y la satisfación (también llamada valor) con la posterior representación gráfica de los puntos y la inspección de los valores. El método de votación acumulativa entrega 100 puntos a los clientes para que los distribuyan entre los requisitos e identificar los de más valor. En el método Top-Ten, cada cliente o desarrollador hace una lista de los diez requisitos que ellos creen de mayor prioridad respecto de un atributo a definir y después se unifican. Sin embargo, una de las técnicas más utilizadas en la priorización de requisitos es la conocida como técnica MoSCoW, que no utiliza propiedades cuantitativamente estimadas para los requisitos elicitados sino cuyo objetivo se centra en la clasificación de los requisitos.

3.1 MoSCoW

El método MoSCoW se basa en el hecho de que, aunque todos los requisitos son importantes, es fundamental destacar aquellos que permiten darle un mayor valor al sistema, lo que permite enfocar los trabajos de desarrollo de manera

más eficiente. El acrónimo MoSCoW se obtiene al combinar las primeras letras de: debe tener (Must have), debería tener (Should Have), podría tener (Could Have), y no tendrá en esta ocasión (Would not Have). Lo que la diferencia de otras técnicas es que, en este caso, la escala utilizada tiene un significado intrínseco, de manera que el usuario responsable de asignar la prioridad conoce el efecto real que producirá su elección. Se separan los requisitos en cuatro grandes categorías que marcan una "línea roja", identificando aquellos requisitos que obligatoriamente deben ser incluidos en el desarrollo. Es fácil de entender por todas las partes interesadas y de poner en práctica, ya que definimos los requisitos básicos sin los cuales no se pondrá el producto software en producción. La tabla 1 muestra el significado de cada una de las categorías.

Tabla 1. Categorías del método MoSCoW

Etiqueta	Significado
M Debe tener	Requisitos fundamentales y obligatorios para satisfacer las necesidades del negocio. Si estos no se cumplen, se verá comprometido el éxito del proyecto
S Debería tener	Requisitos que deberían ser cumplidos en la medida de lo posible. El éxito del proyecto no dependerá directamente del cumplimiento de estos requisitos que pueden ser satisfechos mediante soluciones temporales o, llegado el momento si fuera necesario, podrían ser prescindibles si existiera alguna causa que lo justificara
C Podría tener	Requisitos que son interesantes que se incluyan. Se trata de requisitos adicionales que se implementarán en el caso de disponer de tiempo y presupuesto para ello. Estos requisitos mejorarían el valor del producto pero podrían ser eliminados fácilmente
W no tendrá esta vez	Hace referencia a los requisitos que están descartados de momento, pero que en un futuro podrían ser tenidos de nuevo en cuenta y ser reclasificados en una de las categorías anteriores

3.2 Agrupamiento por K-medias

Los métodos de agrupamiento buscan una manera de dividir un conjunto de datos dado en grupos que contengan datos que sean similares entre sí. Este es un problema computacionalmente difícil (NP-duro). El agrupamiento por k-medias es un algoritmo de aprendizaje no supervisado, ya que no conoce el grupo al que pertenece cada observación, que se emplea para encontrar patrones en los datos (p.ej. segmentos de clientes o productos). Para construir los grupos con el algoritmo k-medias, primero hemos de indicar el número k de grupos en los que queremos agrupar los datos. A continuación, el algoritmo asigna de forma

aleatoria cada observación a uno de los k grupos y encuentra el centroide de cada uno de ellos. A partir de aquí, el algoritmo realiza un proceso de refinamiento iterativo. Primero, asigna cada observación al grupo cuyo centroide esté más próximo. Para después calcular los nuevos centroides de los grupos. Estos dos pasos se repiten para minimizar la variación dentro de cada grupo.

El algoritmo de las k-medias [10] reparte las observaciones $\{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n\}$, donde cada observación es un vector d-dimensional (cada dimensión correponde a un atributo en los datos), en k grupos, $\mathbf{C} = \{C_1, C_2, \cdots, C_k\}$, de manera que la suma de los cuadrados de las diferencias entre las observaciones y el centroide del grupo, al que se han asignado, sea mínima. Formalmente, el algoritmo busca encontrar:

$$\min_{\mathbf{C}} \sum_{i=1}^{k} \sum_{\mathbf{x_j} \in \mathbf{C_i}} ||\mathbf{x}_j - \mu_i||^2, \tag{1}$$

donde μ_i es la media de las observaciones en el grupo C_i. Así, al emplear la distancia euclídea para medir la variación dentro de los grupos, el algoritmo intenta encontrar grupos esféricos d-dimensionales.

Al emplear este algoritmo hay que tener en cuenta una serie de consideraciones adicionales:

- Las variables han de ser contínuas y han de reescalarse para garantizar la comparabilidad. Antes de aplicar k-medias *normalizaremos* (centraremos y reduciremos) las variables para comparar las variaciones con independencia de las unidades de medida utilizadas al recopilar los datos.
- Hay que indicar cuál es el número k de grupos. El método del *codo* puede ayudarnos a encontrar el número apropiado de grupos en un conjunto de datos, al interpretar el porcentaje de varianza explicada en función del número de grupos. El número de grupos escogido debe ser aquel tal que añadir otro grupo no proporcione un mucho mejor modelado de los datos. Así, si representamos el porcentaje de varianza explicado por los grupos frente al número de grupos, los primeros grupos proporcionarán mucha información (explicarán mucha varianza), pero en algún punto la ganancia marginal caerá, creando un ángulo (codo, de ahí el nombre del método) en el gráfico. En este punto se elige el número de grupos.
- Los resultados del algoritmo dependen de los k centroides iniciales que se establecen de forma aleatoria. En la práctica debemos ejecutar el algoritmo partiendo de diferentes condiciones iniciales (i.e. centroides) y quedarnos con el mejor resultado (i.e. partición en grupos con menor varianza).

3.3 Priorización: MoSCoW y *k*-medias

El objetivo que persigue este trabajo es identificar el conjunto de requisitos que constituirá el núcleo del producto software. En terminología MoSCoW, estos son los requisitos fundamentales que *"debe tener (Must have)"* el software que va a desarrollarse. Para poder abordar esta tarea de manera cuantitativa, tenemos que

Tabla 2. Cuantificación de las categorías del método MoSCoW

	Categoría	Cuantificación
M	Debe tener	Alta satisfacción y bajo esfuerzo
S	Debería tener	Alta satisfacción y alto esfuerzo
C	Podría tener	Baja satisfacción y bajo esfuerzo
W	No tendrá esta vez	Baja satisfacción y alto esfuerzo

establecer una relación entre las categorías para priorizar requisitos que define el método MoSCoW y las propiedades cuantitativas asociadas a los requisitos (i.e. satisfacción y esfuerzo). La tabla 2 recoje la caracterización de estas categorías en términos de satisfacción y esfuerzo. El algoritmo k-medias puede emplear estas dos propiedades cuantitativas asociadas a los requisitos para encontrar 4 grupos de requisitos (ya que son 4 las categorías que establece MoSCoW para la priorización de requisitos). Una vez que k-medias encuentra los grupos de requisitos, resta identificar la categoría MoSCoW en la que se encuadran. La identificación se lleva a cabo a partir del centroide de cada grupo en base a la correspondencia de la tabla 2. Así, además de identificar, también priorizaremos los requisitos en grupos, de forma similar a MoSCoW.

Tabla 3. Caso 1: prioridad asignada para cada requisito, esfuerzo de desarrollo e interdependencias

	r_1	r_2	r_3	r_4	r_5	r_6	r_7	r_8	r_9	r_{10}	r_{11}	r_{12}	r_{13}	r_{14}	r_{15}	r_{16}	r_{17}	r_{18}	r_{19}	r_{20}	w_i
c_1	4	2	1	2	5	5	2	4	4	4	2	3	4	2	4	4	4	1	3	2	1
c_2	4	4	2	2	4	5	1	4	4	5	2	3	2	4	4	2	3	2	3	1	4
c_3	5	3	3	3	4	5	2	4	4	4	2	4	1	5	4	1	2	3	3	2	2
c_4	4	5	2	3	3	4	2	4	2	3	5	2	3	2	4	3	5	4	3	2	3
c_5	5	4	2	4	5	4	2	4	5	2	4	5	3	4	4	1	1	2	4	1	4
Esf.	1	4	2	3	4	7	10	2	1	3	2	5	8	2	1	4	10	4	8	4	

$$r_3 \odot r_{12} \; r_{11} \odot r_{13}$$
$$r_4 \Rightarrow r_8 \; r_4 \Rightarrow r_{17} \; r_8 \Rightarrow r_{17} \; r_9 \Rightarrow r_3 \; r_9 \Rightarrow r_6$$
$$r_9 \Rightarrow r_{12} \; r_9 \Rightarrow r_{19} \; r_{11} \Rightarrow r_{19}$$

4 Estudio experimental

Para realizar el estudio de la validez del método propuesto se han realizado experimentos sobre dos conjuntos de datos que han sido utilizados en numerosas ocasiones para estudiar este problema [8, 16, 13, 3]. El primer conjunto de datos, cuyos detalles se recogen en la tabla 3, comprende 20 requisitos, mientras que el segundo conjunto, especificado en la tabla 4, contiene 100 requisitos.

Tabla 4. Caso 2: prioridad asignada para cada requisito, esfuerzo de desarrollo e interdependencias

	r_1	r_2	r_3	r_4	r_5	r_6	r_7	r_8	r_9	r_{10}	r_{11}	r_{12}	r_{13}	r_{14}	r_{15}	r_{16}	r_{17}	r_{18}	r_{19}	r_{20}	r_{21}	r_{22}	r_{23}	r_{24}	r_{25}	w_i
c_1	1	2	1	1	2	3	3	1	1	3	1	1	3	2	3	2	2	3	1	3	2	1	1	1	3	1
c_2	3	2	1	2	1	2	1	2	2	1	2	3	3	2	1	3	2	3	3	1	3	3	3	2	3	5
c_3	1	1	1	2	1	1	1	3	2	2	3	3	3	1	3	1	2	2	3	3	2	1	2	3	2	3
c_4	3	2	2	1	3	1	3	2	3	2	3	2	1	3	2	3	2	1	3	3	1	1	1	2	3	3
c_5	1	2	3	1	3	1	2	3	1	1	2	2	3	1	2	1	1	1	1	3	1	1	3	3	3	1
Esf	16	19	16	7	19	15	8	10	6	18	15	12	16	20	9	4	16	2	9	3	2	10	4	2	7	

	r_{26}	r_{27}	r_{28}	r_{29}	r_{30}	r_{31}	r_{32}	r_{33}	r_{34}	r_{35}	r_{36}	r_{37}	r_{38}	r_{39}	r_{40}	r_{41}	r_{42}	r_{43}	r_{44}	r_{45}	r_{46}	r_{47}	r_{48}	r_{49}	r_{50}
c_1	3	3	3	1	2	2	3	2	1	2	2	1	3	3	2	2	2	3	1	1	1	2	2	3	3
c_2	1	2	2	3	3	1	3	2	2	1	2	3	2	3	3	3	3	1	1	3	2	2	2	1	3
c_3	3	3	1	3	3	3	2	1	2	2	1	1	3	1	2	1	3	1	3	3	3	3	1	3	2
c_4	3	2	1	1	1	1	2	2	2	3	2	2	3	1	1	3	1	1	3	1	2	1	1	3	2
c_5	2	2	3	2	3	1	1	3	3	2	2	1	1	2	1	3	1	1	2	1	2	3	3	2	2
Esf	15	8	20	9	11	5	1	17	6	2	16	8	12	18	5	6	14	15	20	14	9	16	6	6	6

	r_{51}	r_{52}	r_{53}	r_{54}	r_{55}	r_{56}	r_{57}	r_{58}	r_{59}	r_{60}	r_{61}	r_{62}	r_{63}	r_{64}	r_{65}	r_{66}	r_{67}	r_{68}	r_{69}	r_{70}	r_{71}	r_{72}	r_{73}	r_{74}	r_{75}
c_1	3	3	1	3	2	1	3	1	3	1	2	2	3	3	1	3	1	3	2	3	1	3	2	3	1
c_2	3	3	1	2	2	3	3	2	1	1	1	3	2	3	1	2	1	2	3	1	1	3	1	3	2
c_3	3	1	2	3	2	3	2	1	2	3	1	1	2	3	3	1	3	3	3	1	3	1	3	1	1
c_4	2	1	3	2	1	3	3	1	2	3	2	2	3	3	3	1	2	1	2	1	2	3	3	2	2
c_5	1	3	3	2	3	1	2	1	3	2	2	2	1	2	1	3	2	1	2	1	2	2	3	2	1
Esf	6	2	17	8	1	3	14	16	18	7	10	7	16	19	17	15	11	8	20	1	5	8	3	15	4

	r_{76}	r_{77}	r_{78}	r_{79}	r_{80}	r_{81}	r_{82}	r_{83}	r_{84}	r_{85}	r_{86}	r_{87}	r_{88}	r_{89}	r_{90}	r_{91}	r_{92}	r_{93}	r_{94}	r_{95}	r_{96}	r_{97}	r_{98}	r_{99}	r_{100}
c_1	1	2	3	3	1	2	1	3	1	2	2	2	1	3	2	2	3	1	1	1	2	1	3	1	1
c_2	1	3	3	1	2	1	2	1	2	2	1	3	2	2	2	3	2	2	3	2	2	1	3	1	1
c_3	2	3	3	1	2	1	2	3	2	3	1	2	2	3	3	3	3	2	1	1	2	3	3	2	3
c_4	2	1	3	3	1	3	1	2	2	2	1	1	1	3	1	1	3	3	1	2	1	2	3	1	3
c_5	3	2	3	1	3	3	2	1	2	2	2	2	1	3	3	3	1	1	3	1	3	3	3	3	3
Esf	20	10	20	3	20	10	16	19	3	12	16	15	1	6	7	15	18	4	7	2	7	8	7	7	3

$r_{21} \odot r_{22}$ $r_{32} \odot r_{33}$ $r_{46} \odot r_{47}$ $r_{65} \odot r_{66}$

$r_2 \Rightarrow r_{24}$ $r_3 \Rightarrow r_{26}$ $r_3 \Rightarrow r_{27}$ $r_3 \Rightarrow r_{28}$ $r_3 \Rightarrow r_{29}$ $r_4 \Rightarrow r_5$ $r_6 \Rightarrow r_7$ $r_7 \Rightarrow r_{30}$ $r_{10} \Rightarrow r_{32}$
$r_{10} \Rightarrow r_{33}$ $r_{14} \Rightarrow r_{32}$ $r_{14} \Rightarrow r_{34}$ $r_{14} \Rightarrow r_{37}$ $r_{14} \Rightarrow r_{38}$ $r_{16} \Rightarrow r_{39}$ $r_{16} \Rightarrow r_{40}$ $r_{17} \Rightarrow r_{43}$
$r_{29} \Rightarrow r_{49}$ $r_{29} \Rightarrow r_{50}$ $r_{29} \Rightarrow r_{51}$ $r_{30} \Rightarrow r_{52}$ $r_{30} \Rightarrow r_{53}$ $r_{31} \Rightarrow r_{55}$ $r_{32} \Rightarrow r_{56}$ $r_{32} \Rightarrow r_{57}$
$r_{33} \Rightarrow r_{58}$ $r_{36} \Rightarrow r_{61}$ $r_{39} \Rightarrow r_{63}$ $r_{40} \Rightarrow r_{64}$ $r_{43} \Rightarrow r_{65}$ $r_{46} \Rightarrow r_{68}$ $r_{47} \Rightarrow r_{70}$ $r_{55} \Rightarrow r_{79}$
$r_{56} \Rightarrow r_{80}$ $r_{57} \Rightarrow r_{80}$ $r_{62} \Rightarrow r_{83}$ $r_{62} \Rightarrow r_{84}$ $r_{64} \Rightarrow r_{87}$

A partir de los datos asociados a cada caso, para agrupar los requisitos en las cuatro categorías de MoSCoW, se ha aplicado el proceso siguiente:

i) Calcular el valor de satisfacción de cada requisito teniendo en consideración a los clientes y su peso en el proyecto.
ii) Estandarizar los valores de esfuerzo y satisfacción. Estas dos propiedades conforman las variables cuantitativas que emplea el método de las k-medias.
iii) Generar aleatoriamente tantas configuraciones iniciales para el algoritmo de las k-medias como requisitos haya en el conjunto de datos. Así, intentamos evitar que el algoritmo se quede atrapado en un mínimo local.
iv) Ejecutar k-medias sobre cada una de las configuraciones generadas y devolver como resultado la mejor partició en 4 grupos encontrada.
v) Asociar una categoría MoSCoW a cada uno de los grupos obtenidos. Primero, se calcula el centroide de cada grupo, para, a continuación, asignarle una categoría en base a la correspondencia establecida en la tabla 2.

Para la interpretación de resultados recurriremos tanto a tablas numéricas, como a representaciones gráficas que reflejen información sobre los grupos obtenidos. Las tablas recogerán la categoría asociada a cada grupo, su tamaño los valores de esfuerzo y satisfacción del centroide. Los grupos de requisitos obtenidos, junto con sus centroides, se representarán en una gráfica de puntos para ayudar a corroborar visualmente la asociación con las categorías de MoSCoW.

4.1 Resultados e interpretación

En el primer caso estudiado, para el problema formado por 20 requisitos (recogido en la tabla 3), la tabla 5 incluye la categoría asociada a cada uno de los grupos encontrados por k-medias, junto con su centroide. Toda esta información se muestra en la figura 1, donde cada punto representa un requisito y cada grupo se representa con un color y forma distinto; los centroides aparecen representados por un punto de mayor tamaño en el mismo color y forma del grupo. El núcleo de requisitos, proporcionado por el proceso de agrupamiento, que se han identificado como que deben estar presentes en el proyecto de desarrollo, está formado por $\{r_1, r_8, r_9, r_{10}, r_{11}, r_{14}, r_{15}\}$. Este conjunto tiene unos valores de satisfacción y esfuerzo asociados de 372 y 12, respectivamente. No obstante, si queremos obtener un producto software mínimo viable, hemos de tener en cuenta las dependencias presentes en el problema que, en concreto, afectan a r_8 y a r_{11}. Así, en el producto mínimo viable tendremos que incluir los requisitos r_4 y r_{13} además de los que forman el núcleo. La inclusión del primero se debe a la dependencia de implicación $r_4 \Rightarrow r_8$, mientras que el motivo para incluir al segundo se encuentra en la dependencia de combinación $r_{11} \odot r_{13}$. Completar el grupo obtenido en la priorización teniendo en cuenta el resto de elementos presentes en el problema nos ha proporcionado un producto mínimo viable con una satisfacción de 448 y un esfuerzo de 23. Aunque, respecto al núcleo, el producto mínimo viable aumenta la satisfacción un 20% a costa de casi duplicar el esfuerzo de desarrollo.

Tabla 5. Categorías, grupos y centroides obtenidos para el problema de 20 requisitos

MoSCoW	Grupo	Tamaño	Esfuerzo	Satisfacción
M	2	7	1.71	53.14
S	1	4	5.00	56.25
C	4	5	3.40	30.40
W	3	4	9.00	36.00

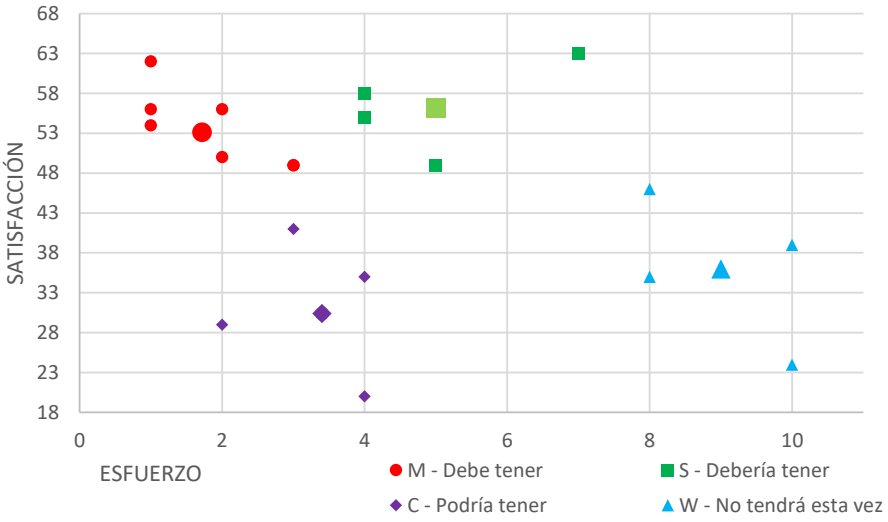

Fig. 1. Grupos encontrados en el problema de 20 requisitos

En el caso estudiado del problema formado por 100 requisitos (véase tabla 4), la categoría asociada a cada uno de los grupos encontrados por k-medias, junto con su centroide se recogen en la tabla 6. En la figura 2 cada punto representa un requisito. Los puntos de mayor tamaño representan los centroides de cada grupo. El proceso de agrupamiento ha identificado como núcleo de requisitos el conjunto formado por $\{r_8, r_9, r_{16}, r_{18}, r_{19}, r_{20}, r_{21}, r_{23}, r_{24}, r_{25}, r_{27}, r_{29}, r_{32}, r_{40}, r_{41}, r_{46}, r_{49}, r_{50}, r_{51}, r_{52}, r_{54}, r_{56}, r_{62}, r_{72}, r_{73}, r_{77}, r_{89}, r_{90}, r_{93}, r_{98}, r_{100}\}$, con valores de satisfacción y esfuerzo asociados de 935 y 173, respectivamente. El conjunto de requisitos que definen el producto software mínimo viable para este problema estará formado por el núcleo, los requisitos $\{r_{22}, r_{33}, r_{47}\}$ para que se satisfagan las dependencias de combinación y $\{r_2, r_3, r_7, r_6, r_{10}, r_{14}, r_{30}\}$ para que se satisfagan las dependencias de implicación. El producto mínimo viable propuesto alcanza una satisfacciónde 1170 (un incremento del 25%), con un esfuerzo de desarrollo de 323 (que supone un 88% de esfuerzo adicional).

Los resultados obtenidos en este trabajo están sujetos a diversas limitaciones derivadas del la naturaleza empiríca del proceso de validación. Por una parte la subjetividad implícita en la estimación de esfuerzo y satisfación. Si bien este

Tabla 6. Categorías, grupos y centroides obtenidos para el problema de 100 requisitos

MoSCoW	Grupo	Tamaño	Esfuerzo	Satisfacción
M	3	31	5.58	30.16
S	1	20	15.25	31.45
C	2	26	6.00	22.50
W	4	23	17.52	22.04

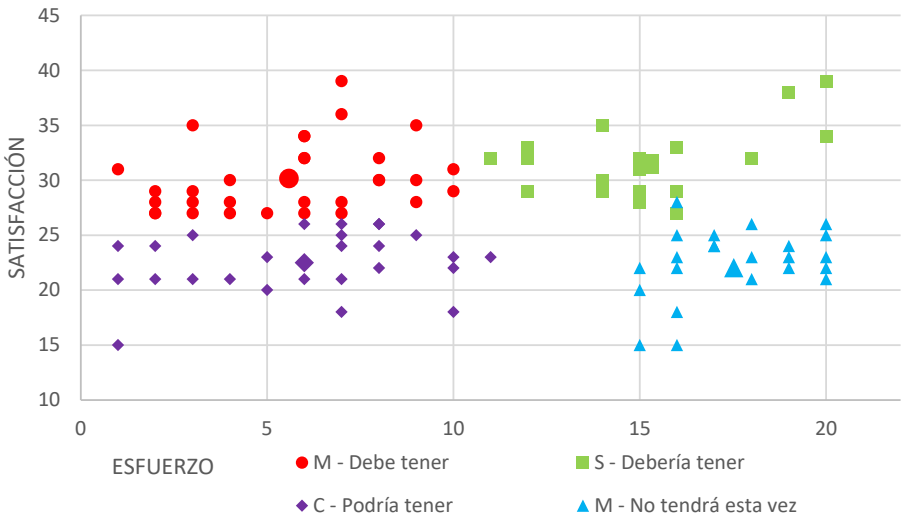

Fig. 2. Grupos encontrados en el problema de 100 requisitos

tipo de medidas son utilizadas cada día por los ingenieros del software, está claro que siempre estarán sujetas a imprecisión, lo que puede tener influencia en la determinación de las fronteras de las categorías. La estandarización de las variables viene a paliar, en la medida de lo posible, este efecto. Otra posible amenaza se encuentra en el número de categorías utilizado. Optamos por utilizar 4 que son las que identifica el método MoSCoW. Aunque esto no justifica que sea la mejor elección a la hora de interpretar el porcentaje de varianza explicada en función del número de grupos. Para comprobar que tomar $k = 4$ es una elección suficientemente buena, podemos recurrir al método del codo y visualizar la evolución de la varianza explicada en base al número de grupos. En los dos casos estudiados (véase Fig. 3) a partir de $k = 4$ añadir un nuevo grupo no mejora el modelado de los datos.

5 Conclusiones

En este trabajo presentamos la extensión del método MoSCoW de priorización de requisitos, altamente conocido, con el algoritmo k-medias para la definición del conjunto de requistos fundamentales o núcleo de requisitos a incorporar

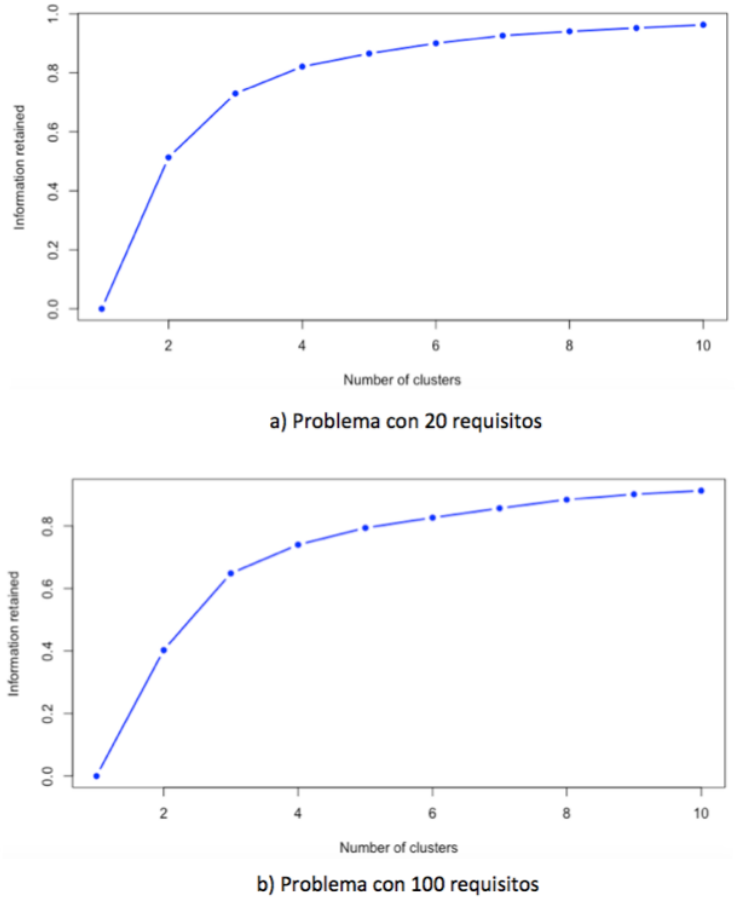

Fig. 3. Porcentaje de varianza explicada por número de grupos

en proyectos de desarrollo de software. La obtención del producto mínimo viable supone la incorporación a la solución de aquellos requisitos necesarios para cumplir las restricciones fijadas por las interdependencias entre los requisitos. Este paquete de requisitos es el punto de partida en las negociaciones contractuales con el cliente para la definición del alcance del proyecto.

Hemos evaluado la propuesta con conjuntos de datos empleados previamente en la literatura de referencia. Los resultados son favorables validando, tanto el número de categorias empleadas, como la aproximación cuantitativa al método MoSCoW. La técnica utilizada es en esencial menos compleja que otras de las propuesta anteriores y tiene la ventaja respecto a resto técnicas cualitativas de similar complejidad de estar fundamentada en datos numéricos.

Entre las posibles líneas de trabajo futuro cabe destacar la ampliación del número de experimentos y buscar aproximaciones alternativas cuando el producto mínimo viable infrinja las retricciones ligadas a los recursos.

Agradecimientos

El trabajo ha sido parcialmente financiado por la Universidad de Almería, a través del grupo de investigación TIC-181, Andalucía Tech y por el Ministerio de Economía y Competitividad mediante el proyecto TIN2017-92480-EXP.

References

1. Achimugu, P., Selamat, A., Ibrahim, R., Mahrin, M.N.: A systematic literature review of software requirements prioritization research. Information and software technology 56(6), 568–585 (2014)
2. del Águila, I.M., del Sagrado, J.: Three steps multiobjective decision process for software release planning. Complexity 21(S1), 250–262 (2016)
3. del Águila Cano, I.M., del Sagrado Martínez, J., Bosch Arán, A.: Análisis de las soluciones guiadas por búsqueda para el problema de selección de requisitos. In: Canós, J. H. y González-Harbour, M. (Eds.), Actas de las XX Jornadas de Ingeniería del Software y Bases de Datos (JISBD 2015). Santander, septiembre. pp. 1–14. Bibilioteca Digital de Sistedes (2015)
4. van den Akker, J.M., Brinkkemper, S., Diepen, G., Versendaal, J.: Determination of the next release of a software product: an approach using integer linear programming. In: Proceeding of the 11th International Workshop on Requirements Engineering: Foundation for Software Quality REFSQ'05. pp. 247–262 (2005)
5. Bagnall, A.J., Rayward-Smith, V.J., Whittley, I.: The next release problem. Information & Software Technology 43(14), 883–890 (2001)
6. Bourque, P., Fairley, R.E. (eds.): SWEBOK: Guide to the Software Engineering Body of Knowledge. IEEE Computer Society, Los Alamitos, CA, version 3.0 edn. (2014)
7. Chicano, F., Dominguez, M.A., del Águila, I.M., del Sagrado, J., Alba-Torres, E.: Dos estrategias de búsqueda anytime basadas en programación lineal entera para resolver el problema de selección de requisitos. In: García Molina, J. (Ed.), Actas de las XXI Jornadas de Ingeniería del Software y Bases de Datos (JISBD 2016). Salamanca, septiembre. pp. 1–14. Biblioteca Digital de Sistedes (2016)
8. Greer, D., Ruhe, G.: Software release planning: an evolutionary and iterative approach. Information and software technology 46(4), 243–253 (2004)
9. Harman, M., Krinke, J., Medina-Bulo, I., Palomo-Lozano, F., Ren, J., Yoo, S.: Exact scalable sensitivity analysis for the next release problem. ACM Transactions on Software Engineering and Methodology (TOSEM) 23(2), 19 (2014)
10. Hartigan, J., Wong, M.: Algorithm as 136: A k-means clustering algorithm. Journal of the Royal Statistical Society, Series C (Applied Statistics) 28(1), 100 – 108 (1979)
11. Jung, H.W.: Optimizing value and cost in requirements analysis. IEEE Software 15(4), 74–78 (1998)
12. Ngo-The, A., Ruhe, G., Shen, W.: Release planning under fuzzy effort constraints. In: Cognitive Informatics, 2004. Proceedings of the Third IEEE International Conference on. pp. 168–175. IEEE (2004)

13. Palomo Lozano, F., del Águila Cano, I.M., Medina Bulo, I.: Un algoritmo híbrido para el problema nrp con interdependencias. In: García Molina, J. (Ed.), Actas de las XXI Jornadas de Ingeniería del Software y Bases de Datos (JISBD 2016). Salamanca, septiembre. pp. 1–14. Biblioteca Digital de Sistedes (2016)
14. Pitangueira, A.M., Maciel, R.S.P., Barros, M.: Software requirements selection and prioritization using sbse approaches: A systematic review and mapping of the literature. Journal of Systems and Software 103, 267–280 (2015)
15. Ruhe, G.: Product release planning: methods, tools and applications. CRC Press (2010)
16. del Sagrado, J., del Águila, I.M., Orellana, F.J.: Multi-objective ant colony optimization for requirements selection. Empirical Software Engineering 20(3), 577–610 (2015)
17. Thakurta, R.: Understanding requirement prioritization artifacts: a systematic mapping study. Requirements engineering 22(4), 491–526 (2017)

View publication stats