Lawrence Chung1 y Julio César Sampaio do Prado Leite2

[1] Departamento de Ciencias de la Computación, Universidad de Texas en
[2] Dallas Departamento de Informática, Pontifícia Universidade Católica do Rio de
www.utdallas.edu/~chung/, www.inf.puc-rio.br/~jul

Abstracto. Esencialmente, la utilidad de un sistema de software está determinada tanto por su funcionalidad como por sus características no funcionales, como usabilidad, flexibilidad, rendimiento, interoperabilidad y seguridad. Sin embargo, ha habido un énfasis insuficiente en la funcionalidad del software, a pesar de que lo funcional no es útil o utilizable sin las características no funcionales necesarias. En el capítulo, revisamos el estado del arte sobre el tratamiento de requisitos no funcionales (en adelante, NFR), al tiempo que brindamos algunas perspectivas para direcciones fu.

Palabras clave: Requisitos no funcionales, NFR, objetivos blandos, satisfacción, ingeniería de requisitos, ingeniería de requisitos orientada a objetivos, criterios de selección alternativos.

1. Introducción

"Es más difícil lidiar con lo blando que con lo duro". [Anónimo]

Esencialmente, la utilidad de un sistema está determinada tanto por su funcionalidad como por sus características funcionales, como usabilidad, flexibilidad, rendimiento, interconexión y seguridad. Sin embargo, ha habido un énfasis desigual en la función del sistema, a pesar de que la funcionalidad no es útil o utilizable sin las características no funcionales.

Al igual que casi todo lo demás, el concepto de calidad también es fundamental en la ingeniería de software, y se tienen en cuenta tanto las características funcionales como las no funcionales en el desarrollo de un sistema de software de calidad. En parte debido a la corta historia detrás de la ingeniería de software, en parte debido a la necesidad de tener sistemas en ejecución rápidamente que satisfagan las necesidades básicas, y también a la naturaleza “blanda” de las cosas no funcionales, la mayor parte de la atención en la ingeniería blanda en el pasado se ha centrado en centrado en notaciones y técnicas para proporcionar las funciones que debe realizar un sistema de software.

Una práctica frecuentemente observable, como resultado de este aspecto funcional desequilibrado de un artefacto de software, es que las características de calidad necesarias se tratan sólo como cuestiones técnicas relacionadas principalmente con el diseño detallado o el d

AT Borgida et al. (Eds.): Mylopoulos Festschrift, LNCS 5600, págs. 363–379, 2009.

propuesta como solución y también cuáles podrían ser los requisitos específicos de la solución de software. Y los problemas del mundo real están más orientados no a lo funcional que a lo funcional, por ejemplo, baja productividad, lentitud, alto costo, baja calidad y cliente insatisfecho.

Aunque la comunidad de ingeniería de requisitos ha clasificado los requisitos como funcionales o no funcionales, la mayoría de los modelos de requisitos existentes y los lenguajes de especificación de requisitos carecían de un tratamiento adecuado de las características de calidad. El tratamiento de las características de calidad como un todo, y no sólo como funcionalidad, ha sido un foco clave de los trabajos en el sector. área del motor de requisitos orientado a objetivos [2], y en particular el Marco NFR [3] que trata el nivel de abstracción no funcional tanto para el problema como para la solución.

Este capítulo presenta una revisión de la literatura sobre NFR, con énfasis en diferentes definiciones, esquemas de representación y conceptos u más avanzados. Al final, concluimos el capítulo discutiendo temas abiertos sobre el tratamiento de los NFR y sus impactos en la construcción de software.

2 ¿Qué son los requisitos no funcionales?

En la literatura, se puede encontrar una gran cantidad de definiciones de re no funcional (NFR).

Hablando coloquialmente, a los NFR se les ha denominado “-ilidades” (p. ej., nosotros “-idades” (por ejemplo, integridad), es decir, palabras que terminan con la cadena “-ilidad” o “-idad”. de tales palabras se puede encontrar, por ejemplo, en [3]. Hay muchos otros tipos que no terminan ni con “-ilidad” ni con “-idad”, como la amistad y la coherencia performa.

Un trabajo importante sobre NFR es el NFR Framework [1] [3], que combina el concepto de funcionalidad con otros atributos de calidad y coproductividad, tiempo y costo, mediante un mayor nivel de abstracción. Centrándose en expresar los requisitos en términos de funciones detalladas y atributos constantes, el Marco NFR ideó la distinción de NFR mediante el uso de meta y meta blanda. Se describirán más detalles sobre el Marco NFR en la Sección 4.

En el área de Arquitectura de Software, es frecuente encontrar atributos clave” [4], entendidos como un conjunto de preocupaciones relacionadas con la calidad. Para una definición de calidad, aquí se utiliza un estándar IEEE [5] como ac "La calidad del software es el grado en que el software posee los atributos de comunicación deseados (por ejemplo, confiabilidad, interoperabilidad)".

Varios otros autores también han tratado este tipo de preocupaciones. Por ejemplo, la calidad (funcionalidad, confiabilidad, facilidad de uso, economía y seguridad) se diferencia de la calidad adicional (flexibilidad, reparabilidad, adaptabilidad, comprensibilidad, capacidad y mejora) en [6].

requisitos y maximiza la calidad "positiva" (como la facilidad de uso, la diversión, la creación de valor. Los sistemas de calidad tradicionales tienen como objetivo minimizar los defectos de calidad negativos y el mal servicio)". Una de las técnicas utilizadas por las estrategias QFD es la Calidad [8], en la que el proceso comienza "...con el cliente, cuyo requerimiento se llama atributos del cliente (CA´s) - frases que los clientes usan para describir las características del producto... ". Por cierto, ninguno de los ejemplos de CA se relaciona con la funcionalidad o simplemente con la funcionalidad.

En el área de requisitos de software, el término requisito no funcional se ha utilizado para referirse a preocupaciones no relacionadas con la funcionalidad del software, diferentes autores caracterizan esta diferencia en informal y desigual. Por ejemplo, en [10] se resume una serie de definiciones de este tipo:

a) "Describir los aspectos no conductuales de un sistema, capturando las restricciones bajo las cuales un sistema debe operar. b) "Los atributos generales requeridos del sistema, incluida la capacidad de portabilidad, la eficiencia, la ingeniería humana, la capacidad de prueba y la modificabilidad de la comprensión". c) "Requisitos que no se refieren específicamente al funcionamiento de un sistema. Imponen restricciones al proceso de desarrollo del producto que se está desarrollando y especifican restricciones externas que debe cumplir". d) "...requisitos globales sobre su desarrollo o costo operativo, confiabilidad, mantenibilidad, portabilidad, robustez y no existe una definición formal o una lista completa de no funciones". e) "Las propiedades de comportamiento que deben tener las funciones especificadas, desempeño, usabilidad". f) "Una propiedad o cualidad que debe tener el producto, como una propiedad de velocidad o precisión". g) "Una descripción de una propiedad o característica que un software puede exhibir o una restricción que debe respetar, que no sea un comportamiento observado".

Luego de argumentar que las definiciones no son claras y que carecen de consenso, dice: "Para las personas que no quieran deshacerse del término 'infunción', podemos definir este término adicionalmente como: DEFINICIÓN. Un no requisito es un atributo o una restricción de un sistema". [10].

Hay otras definiciones adicionales en la literatura que vale la pena agregar a la lista.

h) "... tipos de inquietudes: inquietudes funcionales asociadas con el suministro y inquietudes no funcionales asociadas con la calidad, como seguridad, precisión, desempeño, etc.". [2]. i) "El término "requisito no funcional" se utiliza para delinear el enfoque en "qué tan bien" el software hace algo, en contraposición a los requisitos funcionales, que se centran en "qué" hace el software". [1

A propósito, dejamos la cita de [12] como la última definición de varios. Esta definición de requisitos no funcionales es de gran importancia, como se comenta más adelante en la Sección 4.

Dado que estamos revisando tantas definiciones, podría ser útil centrarnos en cuatro palabras que parecen centrales en todas las definiciones: calidad, funcional y no funcional. Las definiciones fueron seleccionadas de WordNet [13 Net es una base de datos léxica del inglés, donde sustantivos, verbos, adjetivos y a se agrupan en conjuntos de sinónimos cognitivos (synsets), cada uno de los cuales expresa una distinción. Aquí enumeramos la definición principal de cada una de estas palabras como aparecen en Hacemos esto para llamar la atención sobre el término no funcional.

Calidad: Sustantivo - S: (n) calidad (un atributo esencial y distintivo de alguien)

Funcional: Adjetivo -- S: (adj) funcional (diseñado o capaz de una función o uso)

No funcional: Adjetivo -- S: (adj) no funcional (no tener ni realizar); S: (adj) mal funcionamiento, no funcional (no realiza o no puede realizar una función normal)

Funcionalidad: Sustantivo - S: (n) funcionalidad (capaz de cumplir un propósito w

Si examinamos cuidadosamente estas definiciones, notamos que la palabra “functi adjetivo y “calidad” es a la vez sustantivo y adjetivo, mientras que “funcionalidad” También notamos que “funcional” se refiere al uso y “funcionalidad” se refiere a p.

Aplicamos estas definiciones, ya que si entendemos los términos “requisitos divertidos” y “requisitos no funcionales” fuera de contexto, tienen una semántica diferente. Por supuesto, el término "requisitos no funcionales" significa requisitos que no son capaces de realizar una función, pero si se interpreta en el contexto, puede crear confusión. Si la literatura hubiera sido más cuidadosa con los nombres, esta posible confusión podría haberse evitado.

A pesar de estas observaciones y del hecho de que la funcionalidad también puede ser calidad, como también se señala en [6], entendemos que la distinción entre tipos de calidad es extremadamente útil e importante en la ingeniería de software.

Un punto central para la distinción de funcionalidad y otras cualidades en la construcción del software es que el propósito del sistema de software debe estar bien relacionado con las funciones que realizará el software. Puede parecer extraño que la distinción no sea tan evidente en otras áreas de la ingeniería, como podríamos ver en la estrategia QFD. En negocios e ingeniería, la función de un artefacto se ocupa principalmente de entidades físicas y suele ser clara y directa. En la ingeniería de software c cuyos productos son entidades conceptuales, esto no es una cuestión de hecho, sería extraño detallar, por funciones, el hecho de que un automóvil tiene que transportar personas, pero, para que se construya un sistema de software, se necesita un software. engi entiende qué funciones debe realizar, normalmente con mayor dificultad no son evidentemente visibles, medibles, táctiles, etc. Además, los soft

que otros tipos de ingenieros normalmente no tienen que enfrentar.

La distinción entre funcionalidad y otras cualidades en el campo de la ingeniería de software tiene un beneficio importante: deja claro que los requisitos de ingeniería de software están destinados a tratar con atributos de calidad y no solo con A medida que la industria del software se volvió más madura y los ingenieros de software dominaban diferentes dominios , quedó más claro que no bastaría sólo con la descripción de la funcionalidad deseada, sino que también se deberían pensar plenamente en los atributos de calidad desde el principio.

Entonces, en presencia de tantas definiciones diferentes de NFR, ¿cómo debemos proceder? Queremos que nuestra definición de trabajo sea coherente y acorde con otras definiciones. Como definición de trabajo, comenzamos con el coloquial de NFR, como en el marco NFR [1] [3], es decir, cualquier "-ilidad", "-idad", muchas otras cosas que no necesariamente terminan con ninguna de ellas. , como la facilidad de uso y la coherencia, así como preocupaciones por la productividad y la felicidad personal. En vista de las funciones matemáticas, en forma de,

$$f: yo \rightarrow O \text{ (por ejemplo, suma: int x int } \rightarrow \text{ En t)},$$

casi cualquier cosa que aborde las características de f, I, O o relaciones y O se considerará NFR. Por ejemplo, si la suma se puede encontrar fácilmente en una calculadora, si la función se puede construir fácilmente o de manera rentable y en tiempo, si la función devuelve la salida, se pueden ver la función, las entradas o la salida, por ejemplo.

3 Algunos esquemas de clasificación

Como se vio en la sección anterior, varios trabajos proporcionan formas de distinguir entre diferentes tipos de preocupaciones sobre la calidad. Una es la apuesta de distinción y calidad extra [6]. Otra es la distinción entre preocupaciones (atributo de calidad subatributivo), factores (o impedimentos - posibles propiedades del sistema), políticas y mecanismos integrados en el sistema, que tienen un impacto en los y métodos (medios utilizados por el sistema para alcanzar las preocupaciones).) [4].

También destaca la norma ISO/IEC 9126 [14] que distingue niveles de calidad: calidad en uso, calidad externa, calidad interna y proceso.
Basado en estos tipos, [11] proporciona una clasificación orientada a procesos.

1) "La identificación de NFR desde diferentes puntos de vista y diferentes detalles".

2) "El apoyo para descubrir dependencias y conflictos entre ellos y discutirlos y priorizarlos en consecuencia".

3) "La documentación del NFR y la evaluación de este documento. 4) "El apoyo para identificar los medios para satisfacer el NFR, incluso discutir los medios y tomar decisiones de compensación en consecuencia. Esta estimación de costos", y 5) "El apoyo al cambio y la gestión de proyectos".

objetivos blandos funcionales y objetivos blandos no funcionales.
Otro esquema de clasificación se introduce en [16]:

- Requisitos de interfaz: describe cómo es el sistema con el entorno, los usuarios y otros sistemas. Por ejemplo, cualidades de la interfaz de usuario (por ejemplo, facilidad de uso).
- Requisitos de rendimiento: describen las limitaciones de rendimiento entre límites de tiempo/espacio, como cargas de trabajo, tiempos de respuesta y espacio de almacenamiento disponible. Por ejemplo, "el sistema debe realizar transacciones por segundo".

o confiabilidad que involucra la disponibilidad de los componentes. La cantidad de información mantenida y suministrada al sistema debe tener menos de 1 hora de inactividad cada 3 meses.
o seguridad, como los flujos de información permitidos. o capacidad de supervivencia, como la resistencia del sistema bajo fuego y catástrofes.

- Requisitos operativos: incluyen restricciones físicas (tamaño, disponibilidad de nuestro personal, consideraciones sobre el nivel de habilidad, mantenimiento del acceso al sistema, etc. • Requisitos del ciclo de vida: se pueden clasificar en dos subcategorías o Calidad del diseño: medida en términos tales como idad, mejora, portabilidad. o límites al desarrollo, como el tiempo de desarrollo l disponibilidad de recursos, estándares metodológicos, etc.
- Requerimientos económicos: costos inmediatos y/o a largo plazo. • Requerimientos políticos.

La Figura 1 muestra un árbol de calidad del software [17] que tiene como objetivo abordar los tipos de NFR y, lo que es más importante, las posibles correlaciones entre ellos.

FURPS es un acrónimo que representa un modelo para clasificar el software qua utes o requisitos no funcionales, desarrollados en Hewlett-Packard y agregados, de ahí FURPS+, para ampliar el acrónimo y enfatizar varios atributos

- Funcionalidad: conjunto de características, capacidades, generalidad, seguridad.
- Usabilidad: factores humanos, estética, coherencia, documentación. • Fiabilidad: frecuenci gravedad de fallos, recuperabilidad, pre. Precisión, tiempo medio hasta el fallo
- Rendimiento: velocidad, eficiencia, consumo de recursos, tiempo de respuesta. • Soportabilidad: comprobabilidad, extensibilidad, adaptabilidad, principal. Compatibilidad, configurabilidad, capacidad de servicio, instalabilidad, L-ity, portabilida

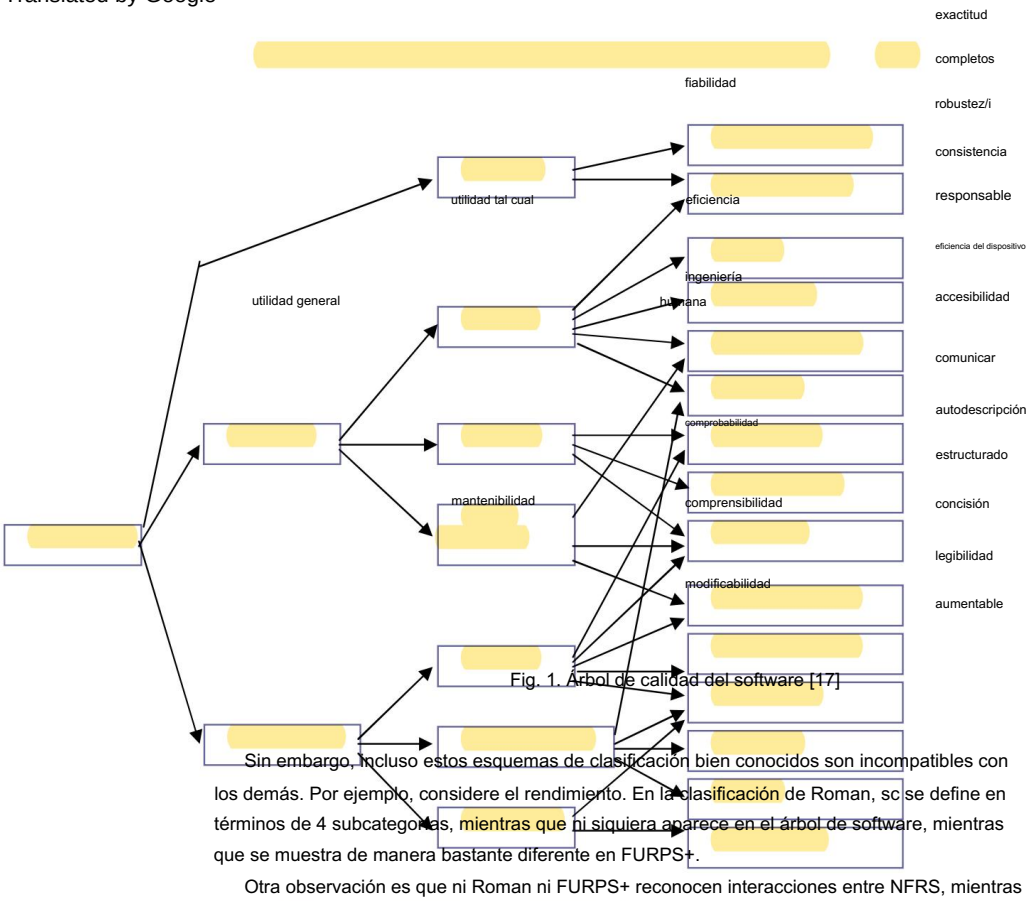

Fig. 1. Árbol de calidad del software [17]

Sin embargo, incluso estos esquemas de clasificación bien conocidos son incompatibles con los demás. Por ejemplo, considere el rendimiento. En la clasificación de Roman, se define en términos de 4 subcategorías, mientras que ni siquiera aparece en el árbol de software, mientras que se muestra de manera bastante diferente en FURPS+.

Otra observación es que ni Roman ni FURPS+ reconocen interacciones entre NFRS, mientras que el árbol de calidad del software lo hace, por ejemplo, en el árbol de calidad del software, la portabilidad y la confiabilidad están relacionadas a través de un subconcepto común de autocontención.

Se pueden hacer observaciones similares sobre otros tipos de NFR además del rendimiento, solo con respecto a los pocos esquemas de clasificación que se muestran aquí, pero también a muchos otros que se muestran aquí, incluido el de [19].

No es sorprendente que los NFR desempeñen un papel crítico en el área de la arquitectura. El esquema de clasificación que se proporciona en el contexto de las evaluaciones ATAM [20] distingue las cualidades del tiempo de ejecución (disponibilidad, rendimiento, seguridad) a partir de las cualidades del tiempo (modificabilidad, integración). ATAM también aborda el riesgo, al mismo tiempo que establece una jerarquía de arquitectura, proceso y organización.

Un profesional de software debe conocer algunos de los esquemas cla más conocidos, como ISO 9126, un estándar internacional para la evaluación de la calidad, y considerar uno o más para adoptar con cierta adaptación. Independientemente del esquema de comunicación que un profesional de software decida adoptar, lo más importante a tener en cuenta es que debe saber qué entiende por NFR como rendimiento, de modo que el significado de dicho NFR pueda comunicarse al usuario como así como con los desarrollador de sistemas/software para que el producto final sea el esperado.

4 Representaciones de requisitos no funcionales

Los requisitos relacionados con los NFR suelen estar separados de los requisitos de las funciones. Una forma habitual de representarlos es mediante requisitos que se enumeran por separado en diferentes secciones del requisito técnico. El estándar IEEE 830 (Práctica recomendada para especificaciones de software) es un buen ejemplo, donde la Sección 3.2 se utiliza para requisitos específicos, mientras que el resto de la Sección 3 se utiliza para describir diferentes NFR.

Algunos autores proponen una estructura en torno a las oraciones de requisitos propuesta por [21] que se compone de: número de identificación, tipo de NFR relacionado con él, descripción, justificación, originador, criterio de ajuste, satisfacción del cliente, insatisfacción del cliente, prioridad, conflictos, material de soporte y historia. son información textual informal.

En el trabajo clásico con SADT [22], una definición de requisitos debe plantear un tipo de preguntas: por qué se necesita un sistema (análisis de contexto), qué sistema servirá para satisfacer este contexto (requisitos funcionales del sistema) y cómo se va a construir (requisitos no funcionales del sistema). Aunque existe una referencia lícita a los requisitos orientados a la funcionalidad frente a los requisitos de calidad, los actigramas de SADT se pueden utilizar para abordar indirectamente algunos de los coactigramas de NFR que se pueden asociar con cuatro tipos de información que interactúan con la entidad: entrada, control, salida y mecanismo. El espíritu de la información de control está muy relacionado con la idea de no funcionalidad, ya que la flecha de control en SA tiene como objetivo limitar cómo se realiza la actividad. Como tal, SADT proporciona atributos o restricciones de calidad de dirección.

Los NFR también se representan comúnmente mediante árboles, que expresan el concepto de agrupamiento o descomposición [4] y también mediante listas.

Algunos autores han utilizado NFR junto con una notación de representación más estructurada, por ejemplo, la combinación de NFR con casos de uso o mi (por ejemplo, ver [23], [24],[25], [26] y [27]).

Los NFR también se representan como restricciones sobre diferentes partes de una escena con tiempo y ubicación como información contextual en la descripción del escenario. Aquí, la representación de un escenario se compone de: título, objetivo, contexto, actores, episodios, excepciones y la restricción de atributo, que se aplica.

el primero en tratar los NFR con más profundidad.

En KAOS [2], que tal vez fue pionero en promover la reingeniería orientada a objetivos, al menos desde una perspectiva de objetivos funcionales, los objetivos son: "…m características intrínsecas como su tipo y atributos, y por sus vínculos con otros elementos de un modelo de requisitos". KAOS aborda objetivos tanto funcionale como no funcionales, que se formalizan en términos de operadores Lograr, Mantener y Evitar, y mediante actividades basadas en lógica temporal con algunos operadores temporales especiales. Por ejemplo, KAOS ofrece ofertas especiales para conceptos como "en algún momento en el futuro" y "siempre en el futuro". Gracias a su naturalez formal, las representaciones en KAOS son susceptibles de verificación y razonamiento. En KAOS, los objetivos se caracterizan como un conjunto de limitaciones sobre los estados. Por ejemplo, un objetivo informal: {{Goal [DoorsClosedWhileMoving]}} [2] se puede expresar mediante la siguiente fórmula

{{ tr: Tren, loc, loc': Ubicación En (tr, loc) o En (tr, loc') loc <> loc tr.Doors = 'cerrado' o (tr.Doors = 'cerrado')}}.

La fórmula anterior, donde "o" es un operador temporal que denota el siguiente estado, mientras un tren se mueve de un lugar a otro, sus puertas se mueven mucho durante el movimiento.

Aunque el lenguaje de representación de KAOS no diferencia los objetivos apostacionales y no funcionales, el gráfico Y/O de KAOS sí la diferencia. Los objetivos que pueden asignarse a agentes individuales no necesitan composición y pueden "operacionalizarse", es decir, pueden describirse mediante restricciones.

Al igual que KAOS, el Marco NFR también promueve la orientación hacia objetivos y pone el énfasis principal en los NFR. En el Marco NFR, los requisitos no funcionales se tratan como objetivos blandos, es decir, objetivos que deben abordarse no en absoluto pero sí con suficiente sentido. Esto es un reconocimiento de las dificultades asociadas al problema y la correspondiente solución. En cuanto al estado del problema, muchas veces es extremadamente difícil, si no imposible, definir un término NFR sin ambigüedades sin utilizar ningún otro término NFR, que a su vez habrá sido multado. En cuanto a la solución, también suele ser extremadamen difícil completar la lista de posibles soluciones y elegir la mejor u óptima solución, debido a diversas limitaciones de recursos, como el tiempo, la mano de obra y el dinero disponibles para una exploración.

Reflejando el sentido de "suficientemente bueno", el Marco NFR introduce el concepto de satisfacción y, con esta noción, se dice que una meta blanda satisface (en lugar de otra meta blanda). El Marco NFR ofrece varios tipos diferentes de co mediante los cuales una meta blanda satisface, o niega, Otro objetivo blando: HACER, AYUDAR, ROMPER son los más destacados, así como Y y O (estos también tienen la noción de "suficientemente bueno" en lugar de "satisfacción absoluta"). representar un objetivo blando que satisface (o niega) negativamente a otro.

En el Marco NFR, cada meta blanda o contribución está asociada con indicar el grado en que se satisface o se niega. Se ofrece una propagación de etiquetas en el marco NFR para determinar el efecto de diversas decisiones, independientemente de si son a nivel de sistema o de software. En una etiqueta, cada objetivo o contribución también se puede asociar con un elemento crítico, como extremadamente crítico, crítico y no crítico.

Cuando se utiliza el Marco NFR, los NFR se publican como objetivos blandos que deben alcanzarse y se aplica un proceso iterativo y entrelazado para alcanzarlos, a través de descomposiciones Y/O, operacionalizaciones y argumentos. A lo largo del proceso, se utiliza una representación visual, SIG (intermedio de objetivo blando). gráfico), se crea y mantiene y realiza un seguimiento de los objetivos suaves y las dependencias t, junto con el impacto de varias decisiones a través del sentido de la etiqueta, un SIG muestra cómo se racionalizan varias decisiones (de diseño).

Para aliviar las dificultades asociadas con la búsqueda de kno, el marco NFR Tratar con NFR , ofrece métodos para capturar kn. formas de satisfacer NFR y reglas de correlación para el conocimiento del lado que tales métodos inducen.

Al igual que con los objetivos en KAOS, los objetivos blandos en el Marco NFR se logran en última instancia mediante las acciones de los agentes (humanos, hardware o software), de acuerdo con el espíritu del modelo de referencia [29], en el que las funciones se satisfacen mediante la colaboración entre El sistema de software y los fenómenos ambientales que son causados por agentes en el medio ambiente. Aquí también nos ocupamos de los objetivos suaves que los requisitos (funcionales) deben ayudar a lograr junto con los fenómenos ambientales. Tenga en cuenta que requieren parte de la solución a algún problema en una parte de la realidad. , y la noción o se puede utilizar para representar cualquier cosa que no sea funcional, ya sea el problema o la solución.

La familia i*: i* [30], Tropos [31] y GRL (Requisitos de objetivo Lang heredó el concepto de objetivo suave del marco NFR, apuntando a objetivos blandos, o atributos no relacionados con la funcionalidad, como un modelado de primera clase. Como se mencionó en la Sección 2, la última definición [12] presentada fue algo diferente del resto: una NFR se describe como una justificación de una decisión de diseño y una restricción en la forma en que se puede realizar una funcionalidad requerida. Los NFR desempeñan un papel clave en la ingeniería, ya que se dice que la ingeniería de software tiene que ver con la toma de decisiones. Como varios autores han señalado, los NFR necesitan transformarse en algunos medios, métodos u operaciones. es muy adecuado para los NFR: inicialmente se expresan en términos generales como requisitos abstractos, pero luego gradualmente se van refinando en términos más concentrados y detallados.

Cuando los NFR finalmente se ponen en práctica, en términos de operaciones o restricciones de software, se convierten en la justificación de por qué dichas operaciones existen en el sistema de software, es decir, para servir a los atributos de calidad que especifican objetivos suaves. Si los ingenieros de software son lo suficientemente cuidadosos

Reificar un problema de calidad afecta cómo se logra la funcionalidad general. Actualmente, se toman diferentes tipos de decisiones de diseño a lo largo de un proceso de software, y los NFR actúan como criterio para dichas decisiones de diseño (consulte la discusión sobre estas decisiones de diseño en el contexto de los casos de uso). Como [12], una NFR no es sólo una justificación a posteriori, sino que limita cómo se realiza fu.

Entonces, si el ingeniero de software utiliza atributos de calidad desde el principio durante el proceso de desarrollo, habrá una red de explicaciones, limitadas por la racionalidad, que vinculan los resultados de las decisiones con los atributos de calidad. La justificación del trabajo [34], basándose en trabajos anteriores sobre la idea de Ibis [35], se centra en la decisión, pero sin tomar en consideración factores preexistentes en la decisión. El trabajo sobre la justificación del diseño puede beneficiarse de mejores formas de aprovechar la naturaleza dinámica y contextual del diseño de software y trabajar con la infinidad de alternativas posibles y sus justificaciones [36].

Se dice que la integración de funcionalidad y otras cualidades es esencial. Aunque uno podría cuestionar esa perogrullada, pocos han propuesto o defendido un proceso que entrelaza estas dos clases de requisitos. Los métodos orientados a objetivos, s NFR Framework, KAOS y la familia i*, son las pocas excepciones que consideran la no funcionalidad como un concepto de primera clase, estando entrelazadas la funcionalidad, ya que están cosificadas. Sin embargo, no debería sorprender que el enfoque tenga sus propias formas particulares de hacer este entrelazamiento de atributos de calidad funcional, con varias variaciones distintas y sin necesariamente una historia de desarrollo

¿Así que lo que? En pocas palabras, el punto que estamos tratando de señalar es doble: no es necesario establecer los requisitos funcionales desde el principio, pero pueden ayudar al ingeniero a tomar decisiones de diseño y, al mismo tiempo, justificar dichas decisiones. Para tomar esto en consideración, es necesario que los atributos de calidad se consideren solo como un conjunto separado de requisitos, pero con la consideración de funcionalidad durante todo el proceso de desarrollo.

El Marco NFR y la familia i* tienen una característica intrínseca que los diferencia de otros métodos NFR: la dependencia de un enfoque cualitativo (NFR o objetivos blandos). En el corazón del Marco NFR se encuentra el concepto de que son idealizaciones y, como tales, sin todas sus propiedades definitorias necesarias. Esta característica es similar a la noción de “relación acotada hacia adelante” de Herbert Simon [37] cuando explica su comprensión de lo que los diseñadores usan para tomar una decisión dada información incompleta. Esta característica se basa en las ideas de contribución y correlación entre las explicadas anteriormente. A través de contribuciones, un objetivo blando puede descomponerse en su nivel de operacionalización y, mediante correlaciones, diferentes objetivos blandos se pueden descomponer entre sí. Una red de tales contribuciones y correlaciones permite llevar a cabo diferentes tipos de razonamiento cualitativo. La semántica del trabajo de s está dada por relaciones en tres dimensiones a) descomposición Y/O árbol, b) contribuciones entre subárboles yc) correlaciones entre objetivos blandos, todo lo que conduce a la formación de un gráfico de interdependencia de objetivos blandos (

NFR Framework podría ayudar a lograr mejores modelos UML se describe en [presenta un proceso para vincular gráficos NFR con modelos UML. Las centrales realizan cualitativamente objetivos blandos en los modelos UML. La realización de U, por ejemplo, se basó en introducir atributos a clases, métodos o sobre los atributos.

En i* [30], los vínculos entre objetivos suaves y operaciones (tareas o recursos son explícitos, ya que el modelado se lleva a cabo simultáneamente en el contexto del diagrama de Justificación (SR). En un diagrama SR, las relaciones medios-fines (i* operador spe) puede mostrar cómo las opciones (tareas) se relacionan con diferentes objetivos suaves, mostrando los pros y los contras de cada selección.

El Marco NFR presentado en esta Sección se adapta a cualquier esquema cla que se analizó en la Sección 3, a través de descomposiciones Y/O más allá, al ofrecer esos conceptos de operacionalizaciones y argumentos juntos con correlaciones positivas/negativas. En términos de estos conceptos, Framework ayuda a racionalizar las decisiones de diseño, tanto a nivel de sistema como suaves. Para la representación de NFR, reconoce los NFR como objetivos suaves y los relaciona como contribuciones positivas/negativas parciales/totalmente.

5 direcciones futuras

Ha habido varios trabajos que exploraron más a fondo los objetivos conceptuales, al tiempo que dieron forma a algún escenario para direcciones futuras. Clasificamos áreas: variabilidad del software, análisis de requisitos, obtención de requisitos, reutilización, trazabilidad de requisitos y desarrollo orientado a aspectos. De estas áreas se han explorado aspectos particulares de la idea de un SIG (Softgo pendency Graph).

Cuando se trata de líneas de productos de software, la idea de variabilidad es crítica en algunas partes de una línea de productos, existirán puntos de variación de arquitectura para permitir diferentes alternativas necesarias para componer diferentes productos a partir de una arquitectura común. Los trabajos de [39] [40] [41] exploraron el hecho de que a son intrínsecos en los modelos SIG, ya que son gráficos Y/O. El uso de un enfoque continuo para las líneas de productos brinda una manera fluida de producir un arco de línea de productos, ya que las características no solo se identifican, sino que también se justifican, en términos de objetivos bá

Una de las principales ventajas del enfoque cualitativo de un SIG es el análisis de estados. La idea misma de que puede haber diferentes tipos de relaciones entre objetivos blandos y entre objetivos blandos y operacionalizaciones en un SIG lleva a la multitud a realizar análisis, propagando el impacto de las decisiones a lo largo de las relaciones de contribución. Utilizando el concepto de propagación de etiquetas sobre un SI es posible evaluar cómo una operacionalización dada de un NFR formará un gráfico completo. El proceso original fue ideado en el Marco NFR [3] y seguido [42] [43]. Utilizando la idea de propagación de etiquetas, se podrían realizar diferentes análisis desde el principio antes de comprometerse con una arquitectura, como se ve en la exploración de preocupaciones de seguridad [44], en la elección visual de opciones [45] y en la elaboración de un análisis de modelo i*. como un problema SAT [46].

en un esquema de elicitación que parte de un léxico extendido [47], técnicas de representación para ayudar a aclarar la denominación durante el proceso de elicitación. Teoría del constructo personal para provocar la contribución entre los objetivos blandos [49].

El marco NFR [3] ha identificado que los catálogos NFR y los gráficos de composición eran un aspecto importante de la creación de software utilizando softgoa. Un trabajo posterior [50] exploró más las ideas al enfatizar el aspecto de la reutilización y propuso un método para mantener una organización de objetivos blandos destinada a En [5 se explora la forma de trazabilidad centrada en objetivos basada en gráficos de objetivos blandos, en los que la red de objetivos blandos se utiliza como base para explicar los cambios en la evolución.

La relación entre los atributos de calidad y el desarrollo orientado a aspectos se ha explorado en [53] y [54]. Más adelante, otros han identificado la importancia que desempeña el Marco NFR - en particular, los objetivos blandos - en el tratamiento temprano como [55] [56] [57] [58] [59] (Ver [60] para una encuesta sobre el tema).

Si bien estos resultados recientes ayudaron a comprender mejor el género de los requisitos de calidad y abrieron nuevos caminos para futuras investigaciones, otros también avanzaron aún más, como la integración de los NFR en otros requisitos, como el modelo de referencia [29] y los cuatro modelo de variable [61] que influye significativamente en el área de la ingeniería de requisitos. Aunque abordan cuestiones de rendimiento o precisión, son esencialmente funcionales y sin la noción de objetivos. Como se mencionó brevemente en la Sección 4, el modelo de referencia ex establece que los requisitos (funcionales) se satisfacen mediante la relación entre el comportamiento funcional del sistema de software y los fenómenos en el entorno. KAOS [2] va más allá de estos funcionales, introduce tipos generales de objetivos suaves para el sistema en general, mientras que el rendimiento, la precisión y la seguridad son preocupaciones para el sistema de software.

También estamos de acuerdo con [11] en la necesidad de realizar más investigaciones empíricas sobre los NFR durante la ingeniería de requisitos y en el uso de métodos etnográficos sobre cómo los equipos de software abordan los problemas de calidad como requisitos. Entendemos que la investigación debe realizarse con proyectos reales, tanto en situaciones de laboratorio como en proyectos industriales, para mejorar nuestro conocimiento sobre cómo abordar problemas c

Agradecimientos. Agradecemos los comentarios de Barbara Paech sobre el borrador, que nos ayudaron significativamente a mejorar el documento de una manera más comprensible.

Referencias

1. Mylopoulos, J., Chung, L., Nixon, B.: Representación y uso de no funciones: un enfoque orientado a procesos. Traducción IEEE. Software. Ing. 18(6), 483– http://dx.doi.org/10.1109/32.142871
2. van Lamsweerde, A.: Ingeniería de requisitos orientada a objetivos: una guía para las conferencias del 5º Simposio internacional IEEE sobre ingeniería de requisitos 27-31, 2001, p. 249. Sociedad de Computación IEEE, Washington (2001)

. aracc, ., ongsa, .., en, .., ensoc, ..: uay ru cal Informe CMU/SEI-95-TR-021, ESC-TR-95-021 (diciembre de 1995)
5. Estándar IEEE 1061-1992 Estándar para un método de métricas de calidad de software para ingenieros eléctricos y electrónicos, Nueva York (1992)
6. Freeman, PA: Perspectivas del software: el sistema es el mensaje. Addis Leyendo (1987)
7. QFD Institute, Quality Function Deployment, http://www.qfdi.org/ 8. Hauser Jr., Clausing, D.: La casa de la calidad. Revisión de negocios de Harvard 6 (1988)
9. Yeh, RT, Zave, P., Conn, AP, Cole, GE: Recciones y perspectivas del análisis de requisitos de software. En: Vick, CR, Ramamoorthy, CV (eds.) Handbo ware Engineering, Van Nostrand Reinhold Co. (1984)
10. Glinz, M.: Sobre requisitos no funcionales. En: 15° IEEE Internacional R Conferencia de ingeniería (RE 2007), págs. 21-26 (2007)
11. Paech, B., Kerkow, D.: Ingeniería de requisitos no funcionales - Calidad i En: Taller internacional del décimo aniversario sobre ingeniería de requisitos: para la calidad del software, REFSQ 2004 (2004), http://www.sse.uni- e refsq/downloads/toc-refsq04.pdf
12. Landes, D., Studer, R.: El tratamiento de los requisitos no funcionales en MI tella, P., Schäfer, W. (eds.) ESEC 1995. LNCS, vol. 989, págs. 294–306. Primavera Berg (1995)
13. Una base de datos léxica del inglés, http://wordnet.princeton.edu/
14. ISO/IEC 9126-1:2001(E): Ingeniería de software - Calidad del producto - Parte 1: Qu (2001)
15. Jureta, IJ, Faulkner, S., Schobbens, P.-Y.: Un concepto de objetivo blando más expresivo para el análisis de requisitos de calidad. En: Embley, DW, Olivé, A., Ram, S. (eds. LNCS, vol. 4215, págs. 281–295. Springer, Heidelberg (2006)
16. Roman, G.-C.: Una taxonomía de problemas actuales en ingeniería de requisitos. computadora, 14-21 (abril de 1985)
17. Boehm, BW, Brown, JR, Kaspar, H., Lipow, M., MacLeod, GJ, Merritt, M teristics of Software Quality. Holanda Septentrional, Ámsterdam (1978)
18. Grady, R., Caswell, D.: Métricas de software: establecimiento de un Pro tice-Hall para toda la empresa, Englewood Cliffs (1987)
19. Bowen, TP, Wigle, GB, Tsai, JT: Especificación de atributos de calidad del software RADC-TR-85-37, vol. I (Introducción), vol. II (Libro de especificaciones de calidad del software), vol. III (Guía de evaluación de la calidad del software), Rome Air Develop Griffiss Air Force Base, Nueva York (febrero de 1985)
20. Bass, L., Nord, R., Wood, W., Zubrow, D.: Temas de riesgo descubiertos a través de evaluaciones, Informe técnico CMU/SEI-2006-TR-012, ESC-TR-2006-012 (20 21 Robertson, S., Robertson, J.: El proceso de requisitos de Volere, Mastering t ments Process, Addison-Wesley, Londres (1999).
22. Ross, DT: Análisis estructurado (SA): un lenguaje para comunicar ideas. Yo software. Ing. 3(1), 16–34 (1977), http://dx.doi.org/ 10.1109/TSE.1977.229900 23. Chung, L., Supakkul, S.: Representación de nFR y fR: una estrategia de uso y orientada a objetivos acercarse. En: Dosch, W., Lee, RY, Wu, C. (eds.) SERA 2004. LNCS, vol. 36 41. Springer, Heidelberg (2006)
24. Herrmann, A., Paech, B.: MOQARE: requisitos de calidad orientados al mal uso e Requir. Ing. 13(1), 73–86 (2008)

Erencia, vo. . Press

26. Alexander, I.: Los casos de uso indebido ayudan a generar requisitos no funcionales. computar Revista de ingeniería de control 14 (1), 40–45 (2003)
27. de Sousa, TGMC, Castro, JFB: Hacia unos requisitos orientados a objetivos Basado en el Principio de Separación de Preocupaciones. En: Anais do WER 2003 - Works genharia de Requisitos, Piracicaba-SP, Brasil, 27 y 28 de noviembre de 2003, pp. 223– http:// wer.inf.puc-rio.br/WERpapers/artigos/artigos_W georgia_souza. pdf
28. Leite, JC, Hadad, G., Doorn, J., Kaplan, G.: un proceso de construcción de escenarios Comentarios Engineering Journal 5(1), 38–61 (2000)
29. Gunter, C., Gunter, E., Jackson, M., Zave, P.: un modelo de referencia para Requir Especificaciones. Software IEEE, 37–43 (2000)
30. Yu, ESK: Hacia el apoyo al modelado y el razonamiento para la fase inicial requiere neering. En: Actas del Tercer Simposio Internacional IEEE sobre Ingeniería Requir, págs. 226-235 (1997)
31. Castro, J., Kolp, M., Mylopoulos, J.: Hacia la ingeniería informática basada en requisitos: el proyecto Tropos. Sistemas de información 27 (6), 365–389 (2002)
32. Amyot, D., Mussbacher, G.: URN: Hacia un nuevo estándar para los requisitos visuales. En: Sherratt, E. (ed.) SAM 2002. LNCS, vol. 2599, págs. 21-3 Heidelberg (2003)
33. Dutoit, AH, Paech, B.: Especificación de casos de uso basada en fundamentos. Requisito artículo 7(1), 1–3 (2002)
34. Potts, C., Bruns, G.: Registro de los motivos de las decisiones de diseño. En: Procee 10ª Conferencia Internacional sobre Ingeniería de Software. International Confere ware Engineering, Singapur, 11 al 15 de abril de 1988, págs. 418–427. IEEE Comp Press, Los Alamitos (1988)
35. Kunz, W., Rittel, HWJ: Issues as Elements of Information Systems, Workin 131 (julio de 1970); Studiengruppe für Systemforschung, Heidelberg, Alemania, mayo de 1979)
36. Dutoit, AH, McCall, R., Mistrík, I., Paech, B. (eds.): Gestión racional Ingeniería. Springer, Heidelberg (2006)
37. Simon, HA: Las ciencias de lo artificial, 3ª ed. La prensa del MIT, Cam (1977)
38. Cysneiros, LM, Leite, JC: Requisitos no funcionales: de la obtención a los modelos. Traducción IEEE. Software. Ing. 30(5), 328–350 (2004), http://dx.doi.org/10.1109/ TSE.2004.10
39. Liaskos, S., Lapouchnian, A., Yu, Y., Yu, ESK, Mylopoulos, J.: Sobre la adquisición y el análisis de la portería. En: RE 2006, págs. 76–85 (2006)
40. Yu, Y., Lapouchnian, A., Liaskos, S., Mylopoulos, J., Leite, JC: Desde el diseño de software de variabilidad go. En: An, A., Matwin, S., Raś, ZW, Ślęzak, D. (ed ciones de Intelligent Systems. LNCS, vol. 4994, págs. 1–16. Springer, Heidelberg (41. González-Baixauli, B ., Laguna, MA, Leite, JC: Usando modelos de objetivos para analizar ity. En: Primer taller internacional sobre modelado de variabilidad de software-inttems, VaMoS 2007, Actas, Limerick, Irlanda, 16-18 de enero de 2007, p Lero Informe Técnico 2007-01 2007 (2007)
42. Giorgini, P., Mylopoulos, J., Nicchiarelli, E., Sebastiani, R.: Razonamiento con G En: Spaccapietra S., March, ST, Kambayashi, Y. (eds.) ER 2002. LNCS, vo 167–181. Springer, Heidelberg (2002)

ey, asngon 44.
Liu, L., Yu, E., Mylopoulos, J.: Configuración social de análisis de requisitos de seguridad y privacidad. En: Actas de la 11ª Conferencia Internacional IEEE sobre Ingeniería R, 8-12 de septiembre de 2003, IEEE Computer Society, Washington (20 45. González-Baixauli, B., Leite, JC, Mylopoulos, J.: Visual Variability Analys Models. En: Actas de la Conferencia de Ingeniería de Requisitos, 12.° IE tional, 6 al 10 de septiembre de 2004, págs. 198 a 207. IEEE Computer Society, Washin http://dx.doi.org/10.1109/RE.2004.56

46. Horkoff, J., Yu, ESK: análisis cualitativo, interactivo y retrospectivo de i* Mod 2008, págs. 43–46 (2008)
47. Oliveira, APA, Leite, JC, Cysneiros, LM: AGFL - Objetivos de los agentes de Lexi ing Intencionalidad de sistemas multiagentes. En: iStar 2008, págs. 29-32 (2008)
48. Niu, N., Easterbrook, SM: Gestión de la interferencia terminológica en la cuadrícula del repertorio de objetivos. En: RE 2006, págs. 296-299 (2006)
49. González-Baixauli, B., Leite, JC, Laguna, MA: Obtención de interacciones R no funcionales mediante la teoría del constructo personal. En: RE 2006, págs. 340–341 (2 50. Cysneiros, LM, Werneck, V., Kushniruk, A.: Conocimiento reutilizable para requisitos de satisfacción. En: RE 2005, págs. 463–464 (2005)
51. Cleland-Huang, J., Settimi, R., BenKhadra, O., Berezhanskaya, E., Trazabilidad centrada en Christin para la gestión de requisitos no funcionales. En: Actas de la Conferencia Internacional sobre Ingeniería de Software, ICSE 2005, St. Louis, MO, 15-21, 2005, págs. ACM, Nueva York (2005), http://doi.acm.org/10.1145/1062455.1062525
52. Cleland-Huang, J., Marrero, W., Berenbach, B.: Trazabilidad centrada en objetivos: U Plumblines para mantener cualidades sistémicas críticas. Traducción IEEE. Software. Ing. 699 (2008), http://dx.doi.org/10.1109/TSE.2008.45
53. Grundy, JC: Ingeniería de requisitos orientada a aspectos para sistemas basados en componentes. En: Actas del 4º Simposio internacional del IEEE sobre ingeniería Requir, RE, 7 al 11 de junio de 1999, págs. IEEE Computer Society, Washington 54. Moreira, A., Araújo, J., Brito, I.: Atributos de calidad transversales para los requisitos. En: Actas de la 14ª Conferencia Internacional sobre Ingeniería del Conocimiento de Software Engi, SEKE 2002, Ischia, Italia, 15 al 19 de julio de 2002, vol. 27, pág. ACM, Nueva York (2002), http://doi.acm.org/10.1145/568760.56
55. Yu, Y., Leite, JC, Mylopoulos, J.: De las metas a los aspectos: Descubriendo los modelos de metas de los requisitos de Aspec. En: 12.ª Actas internacionales del IEEE de la Conferencia de Ingeniería R, 6 al 10 de septiembre de 2004, págs. IEEE Compu Washington (2004), http://dx.doi.org/10.1109/RE.2004.23
56. Brito, I., Moreira, A.: Integración del marco NFR en un modelo de ER. En: Taller EA-A sobre aspectos iniciales: ingeniería y diseño de requisitos orientados a aspectos, celebrado junto con la 3ª Conferencia Internacional sobre Desarrollo de Software Aspe, Lancaster, Reino Unido, 22 al 26 de marzo (2004), http://trese.cs .utwente.nl/workshops/early-aspects-20 Papers/BritoMoreira.pdf

57. Alencar, F., Silva, C., Moreira, A., Araújo, J., Castro, J.: Identificación de Candid con enfoque I-star. En: Early Aspects 2006: Trazabilidad de aspectos en el ciclo, págs. 4-10 (2006)

oaorave eseaic, , oronó, naro, anaa, coer
pag. 16. ACM, Nueva York (2006), http://doi.acm.org/10.1145/
1188988

59. da Silva, LF, Leite, JC: Generación de vistas de requisitos: una transformación
acercarse. ECEASST 3 (2006)
60. Yu, Y., Niu, N., González-Baixauli, B., Mylopoulos, J., Easterbrook, S., Leit quirements
Ingeniería y Aspectos. En: Ingeniería de requisitos de diseño: perspectiva. Apuntes
de conferencias sobre procesamiento de información empresarial, págs. 432–45
Heidelberg (2009)
61. Parnas, DL, Madey, J.: Documentación funcional para sistemas informáticos.
Programación informática 25 (1), 41–61 (1995)