Reutilización de ontologías en un dominio restringido

Enrique P. Latorres
Universidad ORT del Uruguay
Montevideo, URUGUAY, 11000
latorres@ort.edu.uy
enrique@latorres.org

Resumen

Se plantea un conjunto de hipótesis para la validez de la reutilización de conocimiento. Si los sistemas del futuro deberán estar basados en la reutilización eficiente del conocimiento, esto será posible sólo si efectivamente representan una ventaja contra los sistemas actuales. Así la construcción de nuevo conocimiento a partir de una base de conocimiento de dominios similares debería ayudar a construir nuevas instancias de conocimiento sin grandes esfuerzos y el nivel de reutilización podría ser mejor que simplemente partir de un modelo similar particular. Además la creación de conocimiento muchas veces está basado en la visión operacional del dominio y las sub-clases generalmente son diseñadas de forma algo descuidada o se implementan lo necesario para la infraestructura de lo que se trata de implementar. Esto implica que conocimiento tácito del dominio puede ser implementado de diversas formas y que eventualmente no sea integrado en forma efectiva, debido a visiones divergentes en el diseño de diversas ontologías. Sin embargo comparados estos diseños, algunos de los elementos conceptualizables de conocimiento podrían ser extraídos de la base de conocimiento al identificar patrones que se repiten. Una vez reconocidos podrían ser explícitamente conceptualizados para que sean reutilizados por los expertos del dominio. El experimento trata estos casos en un ejemplo y se presentan los resultados obtenidos. Este trabajo muestra resultados de una experiencia donde se mide la similitud y reutilización de componentes semánticos entre varias ontologías, las cuales están relacionadas por dominios similares. Los resultados indican un mínimo de beneficios esperables para sistemas basados en conocimiento de especificaciones de procesos como herramienta para el desarrollo de nuevas especificaciones y eventualmente el desarrollo de sistemas.

Palabras claves: Gestión de Conocimiento, Dominio de Conocimiento, Dominio de Negocio, Ontologías, Reglas de Negocio, Reglas de Juego, Restricciones, Reutilización, Semántica, Similitud de Ontologías.

Temas: Inteligencia Artificial, Inteligencia Computacional, Sistemas Inteligentes.

Workshop: IV Workshop de Agentes y Sistemas Inteligentes.

1 Introducción

La llamada crisis del software tiene una contraparte en la gestión del conocimiento denominado el cuello de botella de la adquisición de conocimiento o "knowledge acquisition bottleneck" [**1**], Una eventual solución para minimizar los efectos de este problema es que los desarrollos de bases de conocimiento se hagan sobre la base de otros conocimientos reutilizados y mantenidos para el nuevo propósito, y que este proceso sea en su mayor parte automatizable.

En este trabajo se plantea un conjunto de hipótesis para la validez de la reutilización de conocimiento. Si los sistemas del futuro deberán estar basados en la reutilización eficiente del conocimiento, esto será posible sólo si efectivamente representan una ventaja contra los sistemas actuales. Pero la pregunta es si aún con estos temas resueltos en un futuro, la aplicación de semejante paradigma podría ser factible y si significaría una mejora importante a lo que es la crisis del software. Brooks [**2**] presenta un conjunto de los puntos críticos para el desarrollo de software que fácilmente pueden ser extendidos hacia el desarrollo de sistemas basados en conocimiento. En respuesta a este, Cox sugiere en [**3**] que si se deben resolver estos puntos críticos, debe ser a través de un profundo cambio de paradigma. Hay una fuerte corriente que apoya a que este nuevo paradigma necesariamente debe estar basado en conocimiento y debe reutilizar información semántica.

Así la construcción de nuevo conocimiento a partir de una base de conocimiento de dominios similares, con herramientas adecuadas, debería ayudar a construir nuevas instancias de conocimiento sin grandes esfuerzos y el nivel de reutilización podría ser mejor que simplemente partir de un modelo similar particular. Pero la creación de conocimiento muchas veces está basado en la visión operacional del dominio, se tiende a hacer hincapié en los objetos relevantes del dominio, y se descuidan los modelos de subclases, que generalmente disponen de información muy escueta, y los modelos de superclases o de contenedores o modelizadores del sistema en el que los objetos interactúan, muchas veces simplemente son omitidos. Los objetos de subclases generalmente son diseñados de forma algo descuidada o se implementan lo necesario para la funcionalidad de los objetos del dominio de interés. Esto implica que conocimiento tácito del dominio puede ser implementado de diversas formas y que eventualmente no sea simple de integrar en forma efectiva [**4**], debido a visiones divergentes en el diseño de diversas ontologías. Además el uso descoordinado de sub-clases que pueden estar disponibles o no, fomentan la creación de conceptualizaciones difíciles de integrar, como lo muestran los trabajos de Cohen et al. [**5**]. Sin embargo, comparados estos diseños, algunos de los elementos conceptualizables de conocimiento podrían ser extraídos de la base de conocimiento al identificar patrones que se repiten. Una vez reconocidos podrían ser explícitamente conceptualizados e integrados para que sean reutilizados por los expertos del dominio.

Sólo información taxonómica y mereológica, es insuficiente para la identificación de similitudes. Se debe incorporar todo tipo de información relevante al dominio que se está analizando o diseñando, en particular información relativa a los contextos aplicables [**6**]. Así se debe incorporar las reglas de negocios, las restricciones de las clases y las instancias de la ontología modelada. También las intenciones para las que están diseñados y las correlaciones troponímicas de estas [**7**].

El experimento trata estos casos en un ejemplo y se presentan los resultados obtenidos. Este trabajo muestra resultados de una experiencia donde se mide la similitud y reutilización de componentes semánticos entre varias ontologías, las cuales están relacionadas por dominios similares. Los resultados indican un mínimo de beneficios esperables para sistemas basados en conocimiento de

especificaciones de procesos como herramienta para el desarrollo de nuevas especificaciones y eventualmente el desarrollo de sistemas.

2 Planificación de los Experimentos y Aportes de Este Trabajo

Se usó una herramienta de captura de conocimiento, donde se registraron las reglas y especificaciones para un conjunto de programas. En nuestro caso, se analizaron programas de un cierto dominio: Juegos de Cartas (Solitarios) de computadoras. Las reglas que se registraron son las que se cumplen para las jugadas de cada juego según el estado del mismo, entendiendo como estado a una situación arbitraria del juego en cuanto a cartas o montones dispuestos sobre el tablero.

Se codificó las reglas en la herramienta, para cada uno de los juegos seleccionados, llevando adelante criterios de diseño homogéneos entre los modelos. Se compararon las ontologías e hicieron mediciones de reutilización. Las ontologías se construyeron efectivamente reutilizando reglas ya elaboradas para casos anteriores, o partiendo de ontologías elaboradas anteriormente en este experimento y quitando, agregando y modificando declaraciones. Aún así se pueden haber incluido nuevas reglas que sean comparables a otras ya usadas y que no sea trivial su identificación o el origen de su reutilización. También se identificaron ciertas reglas que tienen significado conceptual, reglas básicas de cada juego y se analizó su impacto dentro del conjunto de juegos, como planteo para el análisis de identificación automática de conceptos reutilizables.

3 Diseño Experimental

3.1 Hipótesis de los Experimentos

Dado un dominio de negocio, el cual puede ser tanto un giro de negocio (fabricación, distribución y venta de los productos de un determinado nicho de negocio), como una actividad de negocio (gestión contable y financiera de la empresa), incluyendo todo el know-how implícito o explícito; Definimos como implementación del dominio de negocio, al conjunto de especificaciones de reglas y procesos de negocio de una organización en particular.

Asumiendo que las Ontologías son una herramienta adecuada para la especificación de programas y procesos de negocio, y que si tenemos implementaciones del dominio de negocio de un conjunto grande de organizaciones, entonces:

a) Existe una forma de comparar las especificaciones de las reglas y procesos de negocio, y es posible obtener métricas automáticas de reutilización conceptual.
b) Todas las implementaciones del dominio de negocio tienen un alto grado de similitud entre sí, a excepción de un conjunto relativamente pequeño de reglas o de detalles de las reglas, que hacen a la idiosincrasia de cada organización. Por lo tanto debe ser factible un alto grado de reutilización en las especificaciones.
c) Si se dispusiera de un repositorio con todas las implementaciones del dominio de negocio de un conjunto grande de organizaciones y se debiera desarrollar la implementación del dominio de negocio de otra organización diferente a las anteriores, esta debería tener un conjunto muy grande de especificaciones que coinciden con algunas de las que se encuentran en el repositorio, y esa cantidad debería ser mayor que si se compara la implementación en estudio con cada una de las otras implementaciones del repositorio. O sea que la disponibilidad de un repositorio importante debe garantizar un alto nivel de reutilización en las declaraciones ya disponibles.
d) Además debe haber un conjunto de especificaciones, o partes de especificaciones que corresponden a conceptos importantes de la organización y el negocio, las cuales subyacen dentro de las especificaciones de la implementación (conocimiento tácito), que

no fueron explicitadas o no se les asignó semántica, pero que sin embargo pueden ser identificadas en forma más o menos automática para poder clasificarlas y asignarles una semántica a efectos que sean plausibles de reutilización.

Si se dispone de un par de ontologías que definen o describen dos conceptualizaciones sobre un mismo dominio de conocimiento, potencialmente un dominio de negocio y sus reglas, estas ontologías deben estar constituidas por un conjunto importante de conceptos comunes (clases y restricciones) que pueden ser reutilizados entre ellas y entre otras ontologías del mismo dominio. La integración entre dos ontologías en una tercera, debería ser trivial al menos para las reglas que son compartidas. El resto de las reglas pueden ser ingresadas caso a caso, verificando la coherencia con el resto de las reglas de la ontología que se esta construyendo. Planteado de esta manera la integración de dos ontologías se puede correlacionar al desarrollo de la segunda partiendo de la primera y reutilizando o rediseñando las reglas no compartidas, obviando las reglas que se descartarían. Se asume que subconjuntos de declaraciones (subárboles) de una base de conocimiento se repiten en otros del mismo dominio y que varios de estos deben representar conceptos y elementos importantes que se reutilizan entre ambas bases y que por lo tanto se les puede asignar una semántica. Se investigó la posibilidad de identificar las reglas o subconjunto de declaraciones que puedan tener un interés especial y una semántica para ser reutilizadas, y puedan ser útiles para el desarrollador, fomentando y promoviendo la reutilización.

3.2 Experimentos Realizados

Los modelos elegidos fueron los solitarios Carpet, Klondike, FreeCell, LaBelleLucie y MonteCarlo según la implementación de 123 Free Solitaire 2002 (version 5.9, July 18, 2002, TreeCardGames.com). Luego, Yukon con la implementación de Absol Free Solitaire (Version 8.0, Softgame Company 2002). Y por ultimo el Robamontón, juego popular de cartas, al parecer no implementado en computadora como modelo de control.

3.3 Factores Considerados en la Elección de los Experimentos

El dominio de los juegos de cartas fue sugerido principalmente por haber antecedentes en la creación de programas para familias de juegos, en ser un dominio acotado y manejable en función de los tiempos disponibles para este trabajo. Pero estos antecedentes (bibliotecas, motores de juegos, etc.) no fueron utilizados como base para los experimentos, ya que en su mayor parte no eran aplicables por tener las especificaciones en formatos no extraíbles. Se optó por redefinir los experimentos a partir de las observaciones y descripciones de los juegos. Se acotó el conjunto de juegos a aquellos que se limitan a un solo mazo de cartas, aunque luego de la experiencia se reconoció que los juegos que utilicen varios mazos de cartas no afectarían en forma significativa los resultados generales.

Además, la forma de acción-respuesta de las reglas, es similar a muchas reglas de negocio que se basan principalmente en respuestas a demanda sobre situaciones identificadas.

3.4 Metodología Aplicada

Para cada juego se consideró la modelización de las reglas que rigen sus jugadas. Se definieron las reglas que determinan si un cierto movimiento es válido para todos los escenarios posibles de instancias del juego, esto es todas las posibles situaciones de conjuntos de cartas. En varias metodologías de desarrollo de ontologías se sugiere la elaboración de un conjunto de preguntas para la creación de nuevas ontologías, que se denotan como las "*Competency questions*". Estas preguntas, que podríamos traducir como "preguntas de capacidad", son las que el sistema experto

de gestión de conocimiento será capaz de responder [**8**]. Para el caso de este experimento las preguntas fueron si el juego estaba en un estado que permitía tal o cual movimiento de cartas. Por lo tanto se podría definir un cierto escenario de cartas en los distintos mazos y montones del juego, y consultar al sistema si tal o cual jugada es válida de ser realizada en esa situación para ese juego en particular.

Se definieron conceptualizaciones de los elementos de cada juego, en función de su rol y/o su estructura, y se realizaron ciertos compromisos en la representación por las limitaciones del lenguaje y la herramienta. Estas definiciones y compromisos se reutilizaron a lo largo de todos los modelos desarrollados. Un ejemplo es la carencia de modelizaciones como contenedores abstractos, eso exigió poner un índice a los objetos contenidos en una relación, cuando debían estar en orden. Por lo que se debe definir un *slot* dentro de ciertos objetos que no es intrínseco a la conceptualización de la realidad, por ejemplo para poder representar conjuntos ordenados de cartas. Ciertas tareas constructivas, como ser repartir cartas en forma aleatoria entre varios mazos, al iniciar el juego o al cumplirse una ronda, no fueron consideradas, principalmente porque la herramienta (Protégé) no soporta modelización de reglas que cambian las instancias de conocimiento dentro de la base, además que esas operaciones no son las más interesantes del juego y en general no aportan a la lógica y reglas fundamentales del juego, por lo tanto no afectan mayormente los resultados. Son simplemente mecanismos de dar aleatoriedad a la distribución de cartas en uno o varios montones sin necesidad de precondiciones complejas.

Para cada conjunto de reglas de cada juego, la ontología fue elaborada reutilizando reglas y definiciones de otros juegos modelados anteriormente por lo que se puede asegurar un cierto nivel de reutilización. Incluso, a medida que se disponía de varios juegos, cada vez que se quería modelar uno nuevo se analizaba factores de similitud aparente para basarse en uno de ellos como punto de partida. También se podía copiar partes de otros no usados directamente como base de desarrollo.

La herramienta de modelización es Protégé-2000 Versión 1.7 [**9**][**10**] el cual es un ambiente visual de diseño y registro de ontologías, orientado a *frames* y *slots* desarrollado en Java y que puede correr en forma independiente en un PC. Dispone de un conjunto importante de plugins con orígenes diversos. En particular nos interesa destacar el PAL plugin [**11**]. Consiste en un agregado que permite el ingreso de restricciones y consultas declaradas en un subconjunto de KIF que este módulo soporta. Protégé 2000 con PAL plugin codifica sus bases de conocimiento en un lenguaje derivado de KIF[**12**] [**13**], este a su vez derivado al menos en el formato de LISP[**14**], define que las declaraciones o predicados tengan el formato **(operador operando1 [operando2 ...])** que implica la ejecución de las operaciones en formato prefijo. Así por ejemplo la suma de dos números se codificaría **(+ 1 2)** cuyo resultado sería 3. De la misma manera se pueden anidar predicados: **(and (> valor1 0) (= valor2 valor3))**. Esto hace que el conjunto de predicados anidados se pueda analizar y modelar fácilmente como un árbol sintáctico [**15**]. En particular toda la ontología puede ser modelada de esta manera, incluyendo declaraciones de reglas, consultas y restricciones, la representación de instancias particulares de elementos de los modelos, declaraciones de las clases y ranuras (*slots*) sobre los que se basan. La herramienta Protégé tiene archivos separados para las definiciones de reglas, restricciones e instancias, de las declaraciones de las clases y ranuras. Como las declaraciones de las clases o marcos y ranuras están implícitas en las referencias de los predicados de las reglas y en las declaraciones de las instancias que declaran los valores para las ranuras, se consideró que agregar los archivos de declaración de clases no añadía información extra a las reglas del juego, pues en particular si hay una declaración de las clases y no era inicializada de alguna manera en alguna de las instancias, es simplemente porque la clase o ranura no es utilizada o se usan sus valores por defecto. Toda la información de la base de conocimiento es codificada de

forma que puede ser incorporada en el árbol sintáctico de la ontología.

Sobre la base de las hipótesis definidas se realizaron una serie de experimentos. Las bases de conocimiento generadas fueron procesadas por un programa desarrollado en el experimento para este propósito, que construye el árbol sintáctico y compara las estructuras del árbol para dos ontologías dadas. Se compararon todas las ontologías y se obtuvieron informes de reutilización en función de subárboles similares entre las diferentes ontologías. Es de notar que si bien hubo reutilización en las reglas, por la forma como fueron construidas las ontologías partiendo de otras previas, la comparación de subárboles trata de ser más amplia pues compara similitud entre árboles, que incluirá tanto al reutilización explícita de estructuras y reglas como a la similitud o reutilización implícita de otras estructuras que fueron agregadas en el proceso de desarrollo. Esta información se ingresó a una base de datos también desarrollada para este experimento, donde se analizó la cantidad de veces que cada subárbol de cada ontología es reutilizado en otras ontologías, junto con la información de tamaño y una declaración sintáctica del subárbol considerado. Sobre la base de esta información se desarrollaron las conclusiones de este experimento. Para la comparación de los posibles resultados de un desarrollo basado en un repositorio o base de bases de conocimiento se desarrolló una herramienta capaz de hacer la unión de dos o más ontologías, de forma que el resultado, más allá de que tenga significado o aplicación semántica, contenga todos los subárboles de ambas estructuras. Se realizó dos experimentos, para el mejor y peor caso de índices de reutilización, combinando todas las ontologías menos una, y se analizó el resultado contra la ontología no incluida en la base para comprobar el nivel de reutilización consolidado.

3.5 Elementos de Medición

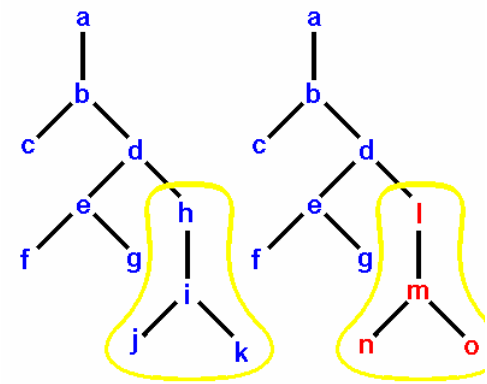
Para cada base de conocimiento correspondiente a cada juego se realizó el árbol sintáctico que modela toda la ontología. Se realizó un programa especial que identifica los términos del lenguaje y los organiza en un árbol sintáctico, para que puedan ser tratados electrónicamente. Una de las hipótesis es que subárboles de una ontología se repiten en otras del mismo dominio y que deben representar conceptos y elementos importantes que se reutilizan entre ambas. Para ello se consideró que se debía comparar los árboles sintácticos de las bases de conocimiento identificando todos los subárboles que se repetían. La comparación es topológica y termina en el primer nodo no coincidente. Las variables ligadas se hacen coincidir con cualquier otra variable ligada. El algoritmo de comparación se describe a continuación:

> *Para cada nodo **a** del árbol **A** y Para cada nodo **b** del árbol **B**, Se compara el subárbol con raíz en **a** con el subárbol con raíz en **b**. Si el término de **a** coincide con el término de **b** incremento la cuenta de términos para el subárbol y se continúa comparando todos los hijos de **a** con todos los de **b** en todas las combinaciones posibles, para cada hijo de **a** se responde con la comparación con el hijo de **b** que dio mayor valor. Cuando todo el subárbol fue analizado registro la cuenta y el subárbol formado con los nodos que coincidieron.*

Por lo tanto lo que devuelve es la cuenta máxima de términos que coinciden entre dos subárboles considerando solo las ramas que tienen nombres iguales o que definen ambas una instancia de una clase. Para el siguiente ejemplo, figura 1, donde se dispone de dos subárboles de once nodos cada uno y suponiendo que los términos de h y l son diferentes, el resto del subárbol no es comparado, aún cuando m e i, y los otros nodos continuando por la rama pudieran tener igual término al correspondiente del otro subárbol. En este caso la cuenta de reutilización daría siete términos. Además para este algoritmo no importa el orden de los subárboles hijo pues compara todos con todos.

Podría suceder que si un subárbol A tiene dos subárboles similares y en cambio el subárbol B tiene dos subárboles diferentes, uno igual a los dos de A, que la comparación de los subárboles se cuente solo la comparación al subárbol de B que dé la mayor similitud, pero lo que estamos considerando es el nivel de reutilización y no tanto la igualdad de la estructura de los árboles. En este caso si dos ramas del subárbol A coinciden con una rama del árbol B, se van a contar ambas ramas. Se considera la comparación de todos los nodos hijos del nodo *a* del subárbol A con todos los nodos hijos de *b* del subárbol B porque se notó, primero que el orden y tipo de los términos en un predicado (nodo) está determinado por el operador, y luego que cuando los operandos son del mismo tipo, el orden de estos no afecta el resultado. Como el operador determina los tipos de los datos se optó por no comparar por tipos de instancias o los nombres que se usan para determinarlos o asociarlos a rangos. Entonces como todas las referencias a instancias (variables ligadas) son de la forma *?nombre*, se hizo que el programa de comparación diera como verdadero la comparación entre dos declaraciones de instancia y continuara con la comparación, independientemente del nombre de la instancia.

figura 1



A medida que el programa va comparando todos los nodos de cada árbol, va creando una base de datos con la cantidad de términos reutilizados (tamaño) asociado al subárbol analizando siempre que al menos haya un nodo similar. Luego la información se ingresa en otro programa creado para este experimento, el programa que compila la información y lleva cuenta de cuantas veces cada subárbol es encontrado dentro de la lista entregada por el programa anterior. Los resultados son una lista de subárboles, con su tamaño y la cantidad de veces que se encuentran para cada combinación de dos bases de conocimiento. Esto nos permite analizar tanto los datos estadísticos de la cantidad de los subárboles reutilizados como de la estructura de las ramas encontradas, más allá de la reutilización explícita que se haya realizado en el proceso de creación de las bases de conocimiento.

4 Resultados Experimentales

Las bases de conocimiento desarrolladas tienen las dimensiones definidas en la tabla 1 en cantidad de. Para cada par de bases de conocimiento se hizo la comparación, 21 en total, y se obtuvo, entre otras, una gran regla que coincidía con la comparación de toda la base de conocimiento desde la raíz con la otra. Esta regla fundamental (Arbol Máximo de Comparación) es la que se tomó para considerarla como principal métrica de reutilización. Los datos obtenidos fueron los siguientes, ver tabla 2.

tabla 1

Base	Tamaño
Klondike	2941
LaBelleLucie	2038
MonteCarlo	1798
Robamonton	1373
Carpet	1412
FreeCell	3152
Yukon	2209

Esta métrica es lo bastante buena como para tener una primer aproximación al nivel de reutilización conceptual de las bases de conocimiento. El algoritmo da peor que una comparación más minuciosa que permitiera además comparar subárboles salteándose algún nodo que no coincide en la comparación pero que algunos niveles más profundo vuelve a mantener su igualdad con el otro subárbol al que se compara o con una interpretación semántica de algunas de sus declaraciones. Por lo tanto el nivel de reutilización real total puede ser mayor o igual al que se obtiene con este algoritmo. Los porcentajes de reutilización obtenidos entonces son los siguientes, ver tabla 3.
La tabla 3 muestra el mayor nivel de reutilización entre dos juegos, o sea el nivel mayor entre el tamaño de un juego dividido por el tamaño del mayor subárbol común de ambos juegos.

tabla 2

Cantidad de Término Comunes

		LaBelle Lucie	Monte Carlo	Roba-monton	Carpet	FreeCell	Yukon
		2038	**1798**	**1373**	**1412**	**3152**	**2209**
Klondike	**2941**	1857	1220	1054	1086	2538	1196
LaBelleLucie	**2038**		1179	1025	1056	1748	1030
MonteCarlo	**1798**			943	934	1257	955
Robamonton	**1373**				1060	1033	1028
Carpet	**1412**					1047	1035
FreeCell	**3152**						1138

tabla 3

Porcentage de Términos reusados entre dos Juegos cualesquiera.

		LaBelle Lucie	Monte Carlo	Roba-monton	Carpet	FreeCell	Yukon
		2038	**1798**	**1373**	**1412**	**3152**	**2209**
Klondike	**2941**	91%	68%	77%	77%	86%	54%
LaBelleLucie	**2038**		66%	75%	75%	86%	51%
MonteCarlo	**1798**			69%	66%	70%	53%
Robamonton	**1373**				77%	75%	75%
Carpet	**1412**					74%	73%
FreeCell	**3152**						52%

tabla 4

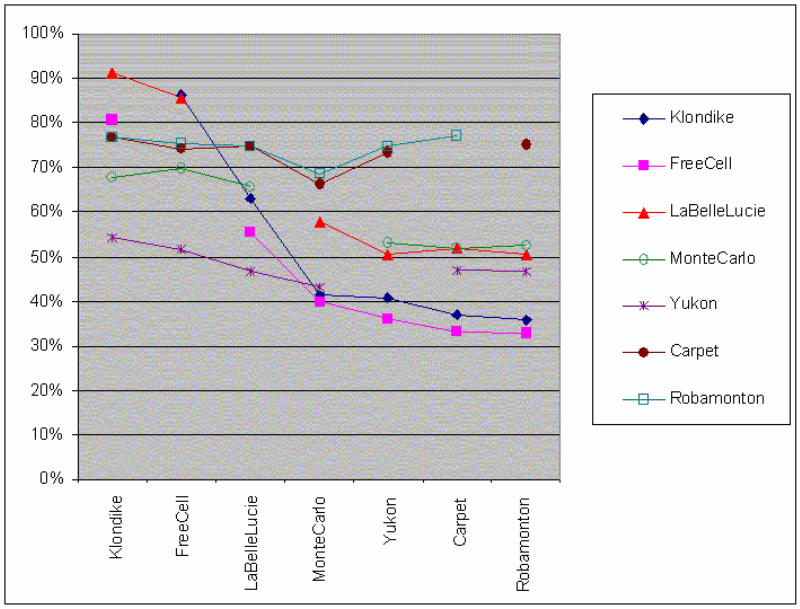
Porcentaje de Términos de la base de la columna que se encuentran en la base de la fila

	Klondike	FreeCell	LaBelle Lucie	Monte Carlo	Yukon	Carpet	Roba-monton	MAX	MIN
Klondike		86%	63%	41%	41%	37%	36%	**86%**	**36%**
FreeCell	81%		55%	40%	36%	33%	33%	**81%**	**33%**
LaBelleLucie	91%	86%		58%	51%	52%	50%	**91%**	**50%**
MonteCarlo	68%	70%	66%		53%	52%	52%	**70%**	**52%**
Yukon	54%	52%	47%	43%		47%	47%	**54%**	**43%**
Carpet	77%	74%	75%	66%	73%		75%	**77%**	**66%**
Robamonton	77%	75%	75%	69%	75%	77%		**77%**	**69%**
MAX	**91%**	**86%**	**75%**	**69%**	**75%**	**77%**	**75%**		
MIN	**54%**	**52%**	**47%**	**40%**	**36%**	**33%**	**33%**		

Es de notar el rango de reutilización de los juegos, donde por ejemplo “La Belle Lucie” reutiliza el 91% de las sentencias con respecto a las del juego “Klondike”, mientras que con “Yukon” es de un 51%. Esto es más curioso aún si se ve ambos juegos “Yukon” y “Klondike” que desde el punto de vista del jugador tienen una similitud muy importante como para esperar que entre ambos hubiera una nivel de reutilización muy grande, sin embargo el “Yukon” carece de ciertas estructuras del juego y un conjunto importante de restricciones que sí tiene “Klondike”, por lo que la similitud aparente para quien no haya analizado con detalle los juegos puede no coincidir con la realidad. Vemos que el porcentaje de reutilización de un juego en otro no es conmutativo, por lo que se construyó el siguiente cuadro, ver tabla 4.

El cuadro en un principio no se ordenó de esta manera, se notó que había una relación entre los juegos de similitud que de alguna manera permitía segmentarlos por una aparente reutilización entre ellos. Al graficar el nivel de reutilización entre los juegos se fueron intercambiando filas y columnas para llevar las líneas de gráficos a una curva más o menos bien definida. El resultado fue el siguiente, ver figura 2:

figura 2



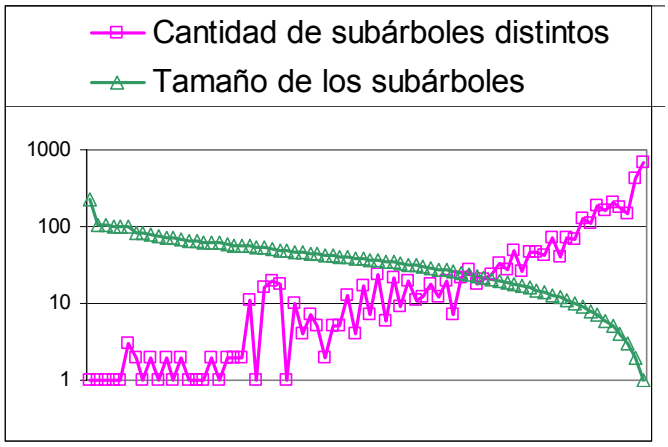
La figura 2 muestra para cada juego (línea de puntos) el porcentaje de reutilización de las declaraciones de los juegos contra los juegos definidos abajo. Sobre esta base vemos que el Robamontón y Carpet son juegos ampliamente distintos a los otros, y al mismo tiempo relativamente sencillos en sus reglas como para que tengan un nivel de reutilización con los otros de alrededor del 70%. Es de notar que el promedio de reutilización es 59,7% similar a lo identificado en [5]. Por lo tanto correlaciones de reutilización podrían ser aplicables para la comparación de bases de conocimiento y una caracterización de ellas para determinar su aplicabilidad de reutilización. Ahora, si consideramos que cualquiera de los juegos se pudo haber desarrollado a partir de una base de bases de conocimiento (repositorio), entonces el nivel de reutilización debería ser mayor que el mayor nivel de reutilización de cada uno de los juegos.

Se hicieron dos bases con todas las ontologías (menos una) a efectos de comparar los resultados con la que quede afuera. Todos menos “La Belle Lucie” quedó de 4555 términos y Todos menos “Yukon” tiene 4077. Nótese que con la fusión de las bases de conocimiento se obtuvo una nueva base de conocimiento que si bien puede no tener sustento conceptual, sí tiene todas las reglas aportadas por todas las bases de conocimiento involucradas, y que en general las bases fusionadas crecieron mucho menos que la concatenación simple de las bases de conocimiento. Se consideraron estas bases, “La Belle Lucie” y “Yukon”, pues son el mejor y el peor caso de los máximos de reutilización, para ver el impacto que tendría la reutilización a partir de un repositorio de reglas que fuera la unión de muchas reglas similares, potencialmente de un dominio determinado de interés. El análisis de “Yukon” y de “La Belle Lucie” contra la suma de los otros juegos da aproximadamente igual que el mejor caso, por lo que no hay mejora significativa. Habrá que estudiar las causas más

adelante, o ver si el criterio de similitud de subárboles extendido que consiste en profundizar un poco por las ramas no iguales para ver si la similitud de los árboles se vuelve a sincronizar, cambia la forma en que se puede considerar los coeficientes de reutilización.

Analizando los subárboles de reglas que se encuentran reutilizados encontramos que en general se destaca un subárbol de gran tamaño, el árbol máximo de comparación, que coincide con la comparación de toda una base de conocimiento con otra. Y sobre esta única gran comparación es con la que hemos basado los estudios presentados antes en este documento. Pero el programa también registra todos los subárboles que son reutilizados a lo largo de toda la base de conocimiento. Por lo que decidimos estudiar esos subárboles para sacar algunas conclusiones. Se realizó la fusión de todas las bases en una y esta se comparó consigo misma para obtener todos los subárboles. Con el resultado se analizó la distribución de los subárboles reutilizados en función de su tamaño, la cantidad de subárboles de ese tamaño reutilizados, y la cantidad de subárboles distintos para ese tamaño. La fusión (merge) de todas las bases de conocimiento dio como resultado una base de 4797 términos, aproximadamente igual a la suma del tamaño de la mayor base con la menor. Se analizaron los subárboles identificados en la comparación. Luego del subárbol máximo de comparación, para la fusión de todas las bases, en orden decreciente de tamaño, aparecen subárboles de dimensiones importantes pero que son solo una fracción de la base de conocimiento. Estos subárboles de 150 términos o menos son en general instancias completas de restricciones o consultas que se han reutilizado, por ejemplo reglas generales que describen el comportamiento de un mazo o un montón de cartas que se reutiliza en muchos de los juegos. Luego comienzan a aparecer las restricciones o consultas solas, mezcladas con declaraciones o partes de las declaraciones fuera del contexto de donde fueron declaradas, en la base de conocimiento. Al nivel de tamaño 58 aparecen las declaraciones de la instancia del objeto más grande del juego, el mazo de cartas, que incluye sus 52 cartas y algunos atributos. Hasta el nivel 31 de tamaño solo aparecen reglas y consultas o pedazos de estas, mezclados con alguna declaración de instancias de juego (montones de cartas con sus listas de cartas correspondientes). En el nivel 30 aparecen las primeras declaraciones de rango mezcladas también con pedazos de restricciones. Así van apareciendo pedazos de restricciones, mezclados con declaraciones de rangos, o sus partes. Ver figura 3.

figura 3



Se notó que con el algoritmo utilizado aparecen en la lista tanto el mayor subárbol encontrado como cualquiera de los subárboles menores dentro del primero lo que causa confusión al aumentar considerablemente la cantidad de subárboles a considerar. En próxima etapa se modificarán los algoritmos para saltearse las ramas de subárboles que claramente ya están incluidos en subárboles previamente evaluados. Otro problema es como identificar subárboles que tengan un significado conceptual de forma que sean plausibles de reutilización en forma automática. Para pequeños proyectos esto parece ser fácil de implementar, al menos en forma manual, pero para grandes desarrollos la reutilización debería hacerse a nivel de componentes conceptuales, con ayuda del sistema para encontrar los conceptos aplicables. Tratar de identificar componentes conceptuales a partir de los subárboles reutilizados parece ser bastante difícil de automatizar pues hay que dar una

interpretación semántica a cada una de las construcciones de subárboles, al menos a las más reutilizadas.

En primera instancia, una fórmula que dio bastante buen resultado para encontrar rápidamente subárboles útiles, y con algún significado conceptual, es buscar por el mayor de un coeficiente que es el producto de la cantidad de reutilizaciones registrados del subárbol por el tamaño del mismo, y filtrando por tipos de objetos (declaraciones de rango, de restricciones, de instancias, de consultas, totales o parciales). Analizamos luego la distribución de subárboles reutilizados en función de su tamaño y consideramos para cada tamaño de subárbol, la cantidad total de subárboles reutilizados y la cantidad de árboles diferentes reutilizados. Vemos que entre los tamaños 42 y 56 hay un pico tanto en variabilidad de los subárboles reutilizados como en la cantidad de reutilizaciones. Estudiando los subárboles vemos que algunos de los más interesantes conceptualmente se encuentran allí, por ejemplo subárboles que determinan si es posible mover una carta a un "suit stack" o sea un montón ordenado de cartas de un mismo mazo, etc. Vemos también que no se previó que hay consultas muy similares a restricciones que cambian el operador raíz de "exists" a "findall" y que se ven como reglas diferentes.

Esto da un criterio para separar ese grupo de árboles de declaraciones cuya variabilidad y cantidad de reutilización es mayor a la de otros en el entorno, para ser analizados en forma manual como propuestas para identificar conceptos reutilizables que no fueron especificados de forma explícita. Se puede mejorar la búsqueda si se hace alguna interpretación de los subárboles identificados eliminando los que contienen declaraciones que muestran que ya se les ha asignado alguna semántica al ser una declaración de una regla o consulta. Esto implica que una primer aproximación para separar algunos de los conceptos implícitos reutilizados dentro de un repositorio de reglas, podría basarse en analizar algunos aspectos de tamaño, cantidad de reutilizaciones y cantidad de variaciones, que se diferencian de otras declaraciones del entorno.

5 Conclusiones

Este experimento de reutilización de ontologías como especificaciones formales muestra que se puede comparar especificaciones y obtener métricas de reutilización en forma automática, aunque consideramos que esta no es la tecnología adecuada para esta función. Es posible obtener obtener un nivel de reutilización de hasta el 91%, para dos aplicaciones diferentes dentro de un mismo dominio con alto nivel de similaridad, aunque habrá que diseñar metodologías adecuadas para medir esta similaridad. En la medida que se disponga de un conjunto importante de reglas y declaraciones para reutilizar, los esfuerzos de desarrollo pueden disminuir en forma importante. En particular un sistema que permita sugerir reglas a ser aplicadas en el proceso de desarrollo puede resolver los problemas presentados por [5], donde declara la dificultad de mantener alineadas varias ontologías si los desarrolladores no disponen fácilmente de acceso a los conceptos que necesitan para ser reutilizados. La reutilización de un conjunto de reglas consolidadas de todos los modelos elaborados, devuelve un coeficiente de reutilización que es mayormente idéntico al nivel de reutilización obtenido con el mejor modelo dentro de la base. Esto puede estar afectado por la forma como se construyeron las ontologías, o la imprecisión de las métricas analizadas, pues era de esperar un coeficiente de mejora en el experimento al identificar nuevas reglas que eran la suma de varias ontologías. Es posible una cierta caracterización de las reglas y especificaciones tácitas dentro de las ontologías para identificar aquellas que tengan una semántica interesante dentro del dominio de conocimiento al que se aplican, y que puedan extraerse y asignarles la semántica para mayor facilidad de los desarrolladores. Aún no tenemos elementos para automatizar el proceso, pero un conjunto de métricas de los subárboles comparados nos permite preseleccionar algunos de los que más probablemente puedan ser aplicados. Luego un tratamiento caso por caso los puede

identificar y clasificar. En el futuro un sistema basado en razonamiento podría automatizar este proceso. Se ve la necesidad de extender estas métricas con comparación extendida, información semántica y sintáctica para analizar los subárboles y afinar las métricas.

El universo de reglas analizado es pequeño, con pocas reglas y de baja complejidad, aunque los resultados animan a indicar que este es un camino factible para los futuros sistemas inteligentes. De todas maneras hay que probar las hipótesis con reglas de negocio reales o con algún otro dominio para comparar los resultados. Este tema necesita más estudio, mejorando las métricas y con otros experimentos necesarios para validar estas conclusiones.

6 Referencias

[1] Feigenbaum, E.A. Knowledge engineering: The applied side of artificial intelligence. *Annals of the New York Academy of Sciences*, pages 91-107, 1984.

[2] Brooks, F.P. No Silver Bullet: Essence and Accidents of Software Engineering. *IEEE Computer*, vol.20, no.4 (April 1987), pp10-19.

[3] Cox, B.J. No Silver Bullet Revisited. *American Programmer Journal*. November 1995.

[4] Pinto, H. S.; Gómez-Pérez, A.; and Martins, J. P. Some Issues on Ontology Integration. In IJCAI99's workshop on Ontologies and Problem Solving Methods: Lessons Learned and Future Trends. 1999.

[5] Cohen, P., Chaudri, V., Pease, A. and Schrag, R. Does Prior Knowledge Facilitate the Development of Knowledge-based Systems? Department of Computer Science, University of Massachusetts, 1999.

[6] Rodríguez, M.A. *Assesing semantic similarity among spatial entity classes*. PhD Thesis, University of Maine. Chapter 7, Conclusions and Future Research Directions. May 2000.

[7] Fellbaum, C. On the semantics of troponymy. Cognitive Science Laboratory, Princeton University. December 2002.

[8] Uschold, M. Building Ontologies: Towards a Unified Methodology. AIAI-TR-197. September 1996. University of Edimburg.

[9] Protégé web site: http://protégé.stanford.edu.

[10] Noy, N.F., Fergerson, R.W. and Musen, M.A. The knowledge model of Protege-2000: Combining interoperability and flexibility. *2th International Conference on Knowledge Engineering and Knowledge Management (EKAW'2000)*. Juan-les-Pins, France. 2000.

[11] PAL plugin. PAL Constraints and Queries Tabs. http://protege.stanford.edu/plugins/paltabs/PAL_tabs.html. The Protégé Axiom Language and Toolset ("PAL") http://protege.stanford.edu/plugins/paltabs/pal-documentation/index.html.

[12] Genesereth, M.R. Knowledge Interchange Format. Draft Proposed American National Standard (dpans). ncits.t2/98-004. http://logic.stanford.edu/kif/dpans.html. http://www-ksl.stanford.edu/knowledge-sharing/kif/

[13] Ginsberg, M. Knowledge Interchange Format. The KIF of Death. *AI Magazine*, 5(63), 1991.

[14] Moyne, A. LISP: A first language for computing. Van Nostrand Reinhold. NY. USA, 1991.

[15] Aho, A.V., Sethi, R. and Ullman, J.D. *Compilers: Principles, Techniques and Tools*. Chapter 5. Bell Telephone Laboratories. Addison Wesley Publishing Company, MA, USA, 1986.