Planificación Continua basada en PDDL: un caso de estudio

Germán Braun Mario Moya Gerardo Parra

email: {german.braun,mario.moya,gparra}@fi.uncoma.edu.ar

Grupo de Investigación en Lenguajes e Inteligencia Artificial
Departamento de Teoría de la Computación
Facultad de Informática
Universidad Nacional del Comahue
Buenos Aires 1400 - (8300)Neuquén - Argentina

Resumen

Este trabajo se desarrolla en el contexto del proyecto de investigación *Agentes Inteligentes en Ambientes Dinámicos*.

La composición de servicios Web es aún una tarea altamente compleja. Varios métodos han sido propuestos con el objetivo de construir herramientas automáticas para esta tarea. Uno de los enfoques más importantes consiste en aplicar algunas técnicas de planificación referenciadas en la literatura de Inteligencia Artificial. En este trabajo, proponemos un nuevo framework basado en planificación continua para la composición de servicios Web. Además dicho framework utiliza el lenguaje PDDL para especificar los problemas de planificación. Consideramos que esta es una alternativa muy atractiva para abordar el problema de composición debido a las características dinámicas, no determinísticas y parcialmente observables del ambiente Web.

Palabras Clave: Composición de Servicios Web, PDDL, Planificación Continua.

Contexto

Este trabajo está parcialmente financiado por la Universidad Nacional del Comahue, en el contexto del proyecto de investigación *Agentes Inteligentes en Ambientes Dinámicos*. Este proyecto tiene una duración de cuatro años, ha comenzado en enero del 2013 y finaliza en diciembre de 2016.

1. Introducción

Podemos considerar a un servicio Web como una aplicación de software autocontenida, modular y auto-definida que puede ser publicada, localizada e invocada a través de la Web. Durante los últimos años, un gran número de empresas y organizaciones solo implementan su núcleo central de negocios y el resto de los servicios ofrecidos es derivado a través de Internet. Por lo tanto, la capacidad de seleccionar e integrar de manera eficiente y efectiva servicios heterogéneos e inter-organizacionales es un aspecto muy importante para el desarrollo de aplicaciones para servicios Web. En particular, si ningún servicio por sí mismo puede satisfacer una determinada funcionalidad requerida por un usuario, debería existir la posibilidad de componer o combinar varios servicios existentes con el fin de atender la necesidad planteada.

Esta tendencia ha motivado un considerable esfuerzo de investigación dedicado a la composición de servicios Web.

Han surgido varias iniciativas con la finalidad de proveer plataformas y lenguajes que permitan una fácil integración de sistemas heterogéneos. En particular, lenguajes como UDDI[1], WSDL[11], SOAP[2], entre otros, definen protocolos estándar para la descripción, descubrimiento e invocación de servicios Web.

A pesar de estos esfuerzos, la composición de servicios Web es aún una tarea altamente compleja y su procesamiento manual está más allá de la capacidad humana. El número de servicios se ha incrementado muy significativamente durante los últimos años y, por lo tanto, existe un enorme repositorio de servicios Web disponibles. Además, estos servicios pueden ser creados y actualizados *on the fly*[1], lo que implica que el sistema de composición necesita detectar la actualización en tiempo de ejecución y la decisión de utilizar o no un determinado servicio debe basarse en la información más reciente.

Varios métodos han sido propuestos con el objetivo de construir herramientas automáticas o semi-automáticas para la composición de servicios Web. Uno de los enfoques más importantes consiste en aplicar técnicas de planificación de Inteligencia Artificial.

En esta aproximación, los métodos de composición intentan construir un plan automáticamente. En un contexto de planificación, podemos asumir que cada servicio Web puede ser especificado por sus precondiciones y sus efectos. Además, cada servicio altera el estado del ambiente luego de su ejecución. El estado del ambiente requerido para la ejecución de un servicio constituye su precondición, mientras que sus efectos se componen por estados alcanzados a partir de la ejecución.

En [4], los autores proponen un framework general para la composición automática de servicios Web. El framework constituye una abstracción de alto nivel, sin considerar ninguna plataforma o lenguaje en particular. Además, no se indica ningún algoritmo de planificación o de composición en especial.

En este trabajo, proponemos planificación continua como el algoritmo de planificación a utilizar. Además, proponemos la utilización de PDDL[2] como lenguaje para la composición de servicios Web. Dado un entorno en constante cambio, la planificación continua se entiende como el proceso de actuar en una serie de fases que se repiten de manera coordinada y que incluyen las acciones de planificar, ejecutar y monitorear. En un ambiente con marcadas características dinámicas, no determinísticas y parcialmente observables como la Web, consideramos una alternativa muy atractiva el uso de planificación continua para abordar el problema de composición de servicios.

El trabajo está estructurado de la siguiente manera. En la sección siguiente presentamos las líneas de investigación del proyecto vigente. En la subsección 2.1 se introduce una descripción del framework propuesto. Comentamos la traducción desde WSDL a PDDL y describimos el proceso de composición de servicios mediante técnicas de planificación continua. En la sección 3 indicamos algunos resultados obtenidos y esperados. Finalmente, comentamos aspectos referentes a la formación de recursos humanos.

2. Líneas de Investigación y Desarrollo

El proyecto de investigación *Agentes inteligentes en ambientes dinámicos* tiene varios objetivos generales. Por un lado, el de desarrollar conocimiento especializado en el área de Inteligencia Artificial. Además, se estudian técnicas de representación de conocimiento y razonamiento, junto con métodos de planificación y tecnologías del lenguaje

[1] En castellano, *sobre la marcha*.

[2] Lenguaje de Definición de Dominios de Planificación (en inglés, Planning Domain Definition Language)

natural aplicadas al desarrollo de sistemas multiagentes.

Específicamente, en la línea Planificación se investiga sobre temas afines a la planificación en ambientes altamente dinámicos y en este sentido es que se ha escogido experimentar con el dominio de los servicios web.

2.1. Framework para la Composición de Servicios Web

El framework propuesto incluye las fases de traducción del lenguaje WSDL a PDDL y la generación de la composición mediante técnicas de planificación. Las fases de evaluación y ejecución de la composición son atacadas directamente por el planificador continuo, por tratarse de un agente que actúa indefinidamente en su entorno. Es decir, el planificador no intercambia etapas de planificación y ejecución, sino que persiste en el entorno actualizando el plan según las nuevas percepciones[9].

2.1.1. Traducción de WSDL a PDDL

En [4], Rao y Su, indican que la mayoría de los sistemas de composición de servicios Web hacen una distinción entre los lenguajes internos y externos para la especificación de los servicios. Los externos son los usados por los usuarios e intentan expresar las entradas y salidas de los servicios de una manera relativamente amigable. Entre los más importantes podemos nombrar al Lenguaje de Descripción de Servicios Web (WSDL)[11], el cual es un estándar basado en XML para la descripción sintáctica de interfaces. Una especificación WSDL provee una descripción de cómo los servicios pueden ser invocados, qué parámetros espera y qué valores retorna.

Los lenguajes internos son utilizados para generar la composición de servicios, por lo tanto, es necesario que las especificaciones descriptas en estos lenguajes sean manipulables por los respectivos algoritmos. Aquí es donde surge la necesidad de desarrollar una traducción entre el lenguaje estándar de servicios Web y dicho lenguaje interno. En nuestro trabajo, utilizaremos a PDDL[7] como lenguaje interno, debido a que, en la actualidad, es ampliamente aceptado y considerado un estándar para la especificación de dominios y problemas de planificación.

Uno de los aportes realizados hasta el momento y, ampliamente referenciado en la bibliografía, es la implementación propuesta por Peer en [5]. Peer propone un mapeo entre los elementos de una especificación WSDL a operaciones PDDL usando anotaciones. Posteriormente, este trabajo fue retomado y mejorado en [6], donde Peer formaliza su primer propuesta mediante un lenguaje de marcado para servicios Web, basado en XML, llamado SESMA. El marcado permite agregar semántica a la descripción de las interfaces de los servicios para poder expresarlos en PDDL y así componerlos automáticamente. Utilizaremos este enfoque propuesto por Peer para especificar dominios en PDDL, dada una entrada WSDL.

2.1.2. Proceso de Composición con Planificación Continua

En el problema de composición de servicios Web, las acciones para el plan están determinadas por cada servicio. Un servicio Web es un componente de software que toma datos de entrada y produce datos de salida. De este modo, las precondiciones y efectos de las acciones del plan conforman, respectivamente, los parámetros de entrada y de salida del servicio [4]. Este dominio, en contraste con el Fútbol de Robots, está caracterizado por su mayor dinamismo dado que las acciones disponibles varían constantemente. Del mismo modo, puede ocurrir que la ejecución de un servicio Web falle o que no tenga el resultado esperado y entonces el plan de servicios compuestos también fallará.

El algoritmo de planificación continua, toma los principios de `continuouspop-agent` propuestos por Russell [10] y extiende esos conceptos para llegar a una implementación real del agente, como una adaptación del pla-

nificador de orden parcial, más conocido como POP [9].

Así como el algoritmo POP resuelve precondiciones abiertas y conflictos causales, nuestro agente continuo debe además tener capacidades para poder enfrentar con eficiencia los cambios propios de los ambientes dinámicos y no determinísticos. El agente puede agregar un nuevo servicio Web que resuelva las precondiciones abiertas o puede poner restricciones en el orden de ejecución de las acciones para evitar conflictos causales. Además, puede eliminar algún servicio Web que ya no esté disponible.

Para mostrar el funcionamiento del planificador, vamos a comenzar proponiendo un plan inicial determinado por el algoritmo de planificación continua, tal como muestra la figura 1.

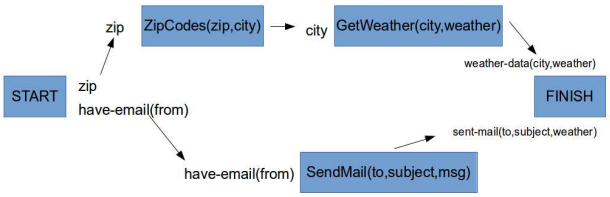

Figura 1: Plan inicial

Ahora supongamos que se comienza a ejecutar el plan y cuando se quiere consultar el estado del tiempo invocando al servicio `GetWeatherService`, éste no se encuentra disponible. Ocurre entonces que la acción del plan `GetWeather` se ejecuta, pero no tiene el efecto esperado. En el plan quedan precondiciones abiertas que deben ser resueltas. Tal situación se puede observar en la figura 2.

Figura 2: Plan actual con acción fallida

Notar que la acción `ZipCodes(city,zip)` es ejecutada y, por lo tanto, el átomo `city` se hace verdadero en el estado actual.

El planificador debe buscar una nueva acción que satisfaga la precondición abierta. Para ello, busca un nuevo servicio que pueda resolver la precondición `weather-data`. Lo encuentra y lo agrega al plan. Este nuevo servicio presenta nuevas precondiciones abiertas (`lat` y `lng`) y, por lo tanto, el planificador debe agregar otras nuevas acciones, repitiendo el proceso de búsqueda anterior. El plan final se muestra en la figura 3. Los servicios `findNearByPostalCode` y `findNearByWeather` son una adaptación para el ejemplo de los servicios ofrecidos por el proyecto GeoNames[3].

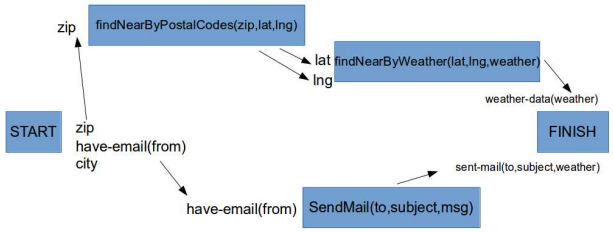

Figura 3: Plan final

3. Resultados Obtenidos y Esperados

La arquitectura de control basada en planificación continua se encuentra en estado de desarrollo. Algunos resultados de esta investigación han sido publicados en [8, 9].

Uno de los resultados esperados de nuestra investigación es que se pueda lograr una simulación real de la composición de servicios web, similar a la brindada en el ejemplo de la sección anterior. Para ello es necesario atacar dos puntos en paralelo de nuestra propuesta. Uno de ellos es lograr la traducción de servicios WSDL a acciones en el lenguaje PDDL. El otro punto es que aún es necesario adaptar el planificador continuo desarrollado para que pueda leer una especificación escrita PDDL.

4. Formación de Recursos Humanos

Durante la ejecución del proyecto se espera lograr la culminación de dos tesis de postgrado y la finalización de seis tesis de grado dirigidas y/o co-dirigidas por los integrantes del proyecto. Ya se han iniciado, en el contexto del grupo y dirigidas por algunos de sus miembros, dos tesis de Licenciatura en Ciencias de la Computación. Finalmente, se aspira a la formación como investigadores de los miembros más recientes del grupo.

Referencias

[1] Belwood, T., et al. Universal Description, Discovery and Integration specification (UDDI) 3.0. http://uddi.org/pubs/uddi-v3.00-published-20020719.htm, accedido en Julio de 2013.

[2] Box, D., et al. Simple Object Access Protocol (SOAP) 1.1, 2001. http://www.w3.org/TR/SOAP/, accedido en Julio de 2013.

[3] GeoNames. The GeoNames geographical database. http://www.geonames.org/about.html, accedido en Julio de 2013.

[4] Jinghai Rao and Xiaomeng Su. A Survey of Automated Web Service Composition Methods. In *Proceedings of the First International Workshop on Semantic Web Services and Web Process Composition, SWSWPC 2004*, pages 43–54, 2004.

[5] Joachim Peer. A PDDL Based Tool for Automatic Web Service Composition. In *Proceedings of the Second Intl Workshop on Principles and Practice of Semantic Web Reasoning (PPSWR*, pages 149–163. Springer Verlag, 2004.

[6] Joachim Peer. Semantic Service Markup with SESMA. In *Proceedings of Workshop Web Service Semantics: Towards Dynamic Business Integration, Chiba, Japan (10th May 2005)*, pages 100–116, 2005.

[7] D. McDermott. PDDL, the Planning Domain Definition Language. Technical report, Yale Center for Computational Vision and Control, 1998.

[8] M. Moya, P. Kogan, G. Parra, L. Cecchi, and C. Vaucheret. Sistemas Multiagentes en Ambientes Dinámicos: Planificación. In *XII Workshop de Investigadores en Ciencias de la Computación*, El Calafate, Santa Cruz, Argentina, 2010. Universidad Nacional de la Patagonia San Juan Bosco.

[9] M. Moya and C. Vaucheret. Agentes Deliberativos Basados en Planificación Continua. In *X Workshop Agentes y Sistemas Inteligentes (WASI)*, Martiarena esquina Italia - S.S. de Jujuy, Octubre 2009. Universidad Nacional de Jujuy - Facultad de Ingeniería.

[10] S. Russell and P. Norvig. *Artificial Intelligence: A modern approach*. Prentice Hall, New Jersey, third edition, 2009.

[11] World Wide Web Consortium (W3C). Web Services Description Language (WSDL) Version 2.0 Part 2: Adjuncts, 2007. http://www.w3.org/TR/wsdl20-adjuncts/, accedido en Julio de 2013.