

Coordinando componentes. Un modelo basado en conectores

Silvia Amaro

samaro@uncoma.edu.ar

Dpto. de Ciencias de la Computación, Universidad Nacional del Comahue, Argentina

Ernesto Pimentel

ernesto@lcc.uma.es

Dpto. de Lenguajes y Ciencias de la Computación, Universidad de Málaga, España

Ana M. Roldán

amroldan@diesia.uhu.es

Dpto. de Ingeniería Electrónica y Sistemas Informáticos. Universidad de Huelva, España

Resumen

En la construcción de aplicaciones a partir de componentes software es necesario entender como interactúan las componentes de manera individual con su entorno y especificar como debería integrarse para que su composición se comporte como un todo coordinado. En este sentido es de principal importancia la detección temprana de problemas de compatibilidad entre componentes. En este proyecto proponemos el uso de un modelo de coordinación basado en canales Reo, como un mecanismo para describir el comportamiento interactivo de componentes software, orientado a resolver problemas de interoperabilidad y analizamos su potencia expresiva comparándolo con otras alternativas basadas en otros modelos de coordinación.

Palabras clave: Componentes, coordinación, expresividad, interoperabilidad.

1. Introducción

Construir aplicaciones a partir de componentes software existentes sigue siendo un desafío importante en el campo de la Ingeniería de software. Su importancia surge por el continuo crecimiento de la provisión y demanda de software para Internet y la computación móvil. Utilizar componentes significa comprender como interactúan de manera individual con su entorno y especificar como deberían ser sus interacciones mutuas y cooperativas para que su composición sea correcta [11]. Por otro lado al trabajar con componentes heterogéneas, uno de los objetivos a lograr es conseguir una separación clara entre los aspectos de interacción y computación, para así favorecer la reutilización y el análisis global de la aplicación. En este contexto se manifiesta la necesidad de disponer de modelos de coordinación [3, 10] que puedan ser utilizados para describir explícitamente protocolos de coordinación complejos en términos de primitivas simples y constructores.

Reo es un modelo de coordinación basado en canales en el cual la comunicación sólo es posible

en presencia de conectores [4]. El carácter composicional de los conectores y la posibilidad de su reconfiguración dinámica, junto a la disponibilidad de una amplia variedad de tipos de canales con semánticas diferentes de las tradicionales, y las propiedades formales de su cálculo coinductivo ponen de manifiesto la potencia expresiva del lenguaje. Nos interesa tratar el problema de la interoperabilidad de componentes a nivel de protocolos de interacción, es decir atendiendo a la forma en que los servicios ofrecidos y requeridos por las componentes deben ser utilizados. Un lenguaje de alto nivel de abstracción basado en Reo, con esta capacidad permitiría, a partir del análisis de las descripciones, determinar de forma anticipada la compatibilidad entre componentes, e incluso la reemplazabilidad, asegurando así el correcto ensamblado de los sistemas basados en componentes. Actualmente existen varias propuestas en este sentido y se han definido formalismos que establecen relaciones para medir el nivel de compatibilidad entre componentes, utilizando álgebras de procesos y determinados modelos de coordinación [7, 8, 9, 12].

Por otro lado con el objetivo de analizar la potencia expresiva del modelo, realizamos un estudio comparativo de diversos modelos de coordinación utilizados en la descripción del comportamiento de componentes. En particular, trabajamos sobre el modelo basado en canales y el modelo basado en datos. Para ello aplicamos la noción de inmersión modular propuesta en [6] sobre el cálculo de procesos basado en Reo, propuesto en este trabajo y el cálculo de procesos basado en Linda, propuesto en [2],

El resto del trabajo se organiza de la siguiente forma. En la sección 2 se presenta el modelo de interacción. En la sección 3 mostramos el estudio de la potencia expresiva del modelo. Finalmente indicamos algunas conclusiones y trabajos futuros.

2. Modelo de Interacción basado en conectores

En el contexto de los lenguajes y modelos de coordinación, nos interesa analizar el modelo basado en la composición de canales de comunicación, introducido por Arbab et. al [4], en el cual el "entorno" progresa teniendo en cuenta las acciones de entrada y salida que se realizarán sobre los canales que están involucrados en la conexión entre componentes coordinadas. En este contexto cada conector impone un patrón de comunicación entre las entidades que conecta.

Reo es un modelo de coordinación que permite la construcción de sistemas por la composición de componentes que interaccionan y cooperan a través de conectores. Los conectores se construyen a partir de un conjunto de canales, síncronos y asíncronos, con comportamiento bien definido y el operador *merge*. Los extremos de un conector son representados por secuencias de datos temporizadas (SDT) $\langle \alpha, a \rangle$ que dan su comportamiento potencial, cuyos cambios se reflejan por medio de su derivativa $\langle \alpha', a' \rangle$. La comunicación entre instancias de componentes se logra exclusivamente por la aplicación de operaciones de entrada y salida sobre los puntos de conexión, afectando directamente a los SDT's que los representan. De forma general, $c^{(k)} = \langle \alpha^{(k)}, a^{(k)} \rangle$ identifica un extremo en el que se han aplicado con éxito k operaciones y sobre el cual una nueva operación exitosa producirá un cambio de $c^{(k)}$ a $c^{(k+1)}$ [5].

2.1. El language \mathcal{R}

Proponemos un álgebra de procesos \mathcal{R} basado en las primitivas de comunicación de Reo . Los agentes en \mathcal{R} se construyen a partir de los operadores prefijo, elección no determinista y

composición paralela. A continuación definimos formalmente la sintaxis de \mathcal{R} :

$$\begin{aligned} P &::= 0 \mid A.P \mid P + P \mid P \parallel P \mid \text{rec}X.P \\ A &::= \text{wr}(c, v) \mid \text{tk}(c, v) \mid \text{rd}(c, v) \end{aligned}$$

dónde 0 denota el proceso vacío, y c un extremo de entrada o salida de un conector. Los prefijos wr , tk y rd son las abreviaciones de las operaciones *write*, *take* y *read* respectivamente.

Como en Reo la comunicación es posible sólo en presencia de un conector, en la definición de la semántica operacional del cálculo, la semántica del conector es una pieza esencial. Consideramos el comportamiento de un conector C dado por un conjunto de transiciones que tienen la forma

$$\langle C, \overline{\text{act}} \rangle \xrightarrow{\overline{\text{act}}_1}_C \langle C', \overline{\text{act}}_2 \rangle$$

dónde $\overline{\text{act}}$ denota el conjunto de acciones que aplicadas en paralelo sobre los extremos del conector pueden producir un progreso sobre el mismo, dando un nuevo estado representado por C' . El conjunto $\overline{\text{act}}_1$ denota las acciones cuya aplicación tiene éxito inmediato, y $\overline{\text{act}}_2$ representa el subconjunto de acciones que han quedado suspendidas en algún extremo del conector. Un ejemplo de un conector simple en Reo es $LR = \{c_i, c_o\}$, las transiciones (1), (2) y (3) describen su comportamiento.

$$\langle \{c_i, c_o\}, \{\text{wr}(c_i, d)\} \rangle \xrightarrow{\{\text{wr}(c_i, d)\}}_C \langle \{c_i', c_o\}, \emptyset \rangle \quad (1)$$

$$\langle \{c_i^k, c_o^j\}, \{\text{tk}(c_o, d)\} \rangle \xrightarrow{\{\text{tk}(c_o, d)\}}_C \langle \{c_i^k, c_o^{j+1}\}, \emptyset \rangle \quad (k > j) \quad (2)$$

$$\langle \{c_i^k, c_o^j\}, \{\text{rd}(c_o, d)\} \rangle \xrightarrow{\{\text{rd}(c_o, d)\}}_C \langle \{c_i^k, c_o^j\}, \emptyset \rangle \quad (k > j) \quad (3)$$

El sistema de transiciones del cuadro 1 describe el modelo computacional de \mathcal{R} . En cualquier caso el agente sólo podrá proceder en presencia de un conector que contenga en su comportamiento transiciones etiquetadas con el mismo conjunto de acciones que se está intentando aplicar.

3. Potencia expresiva del modelo

Para analizar la potencia expresiva del lenguaje \mathcal{R} , en primera instancia lo comparamos con el lenguaje \mathcal{L} propuesto en [] para el modelo de coordinación Linda. El criterio adoptado para la comparación de ambos modelos se basa en la noción de inmersión modular [], o sea una traducción composicional que preserve algunos aspectos de observables, en particular éxito y fracaso. Sean dos lenguajes L y L' , y las funciones (observables) $\mathcal{O} : L \rightarrow \text{Obs}_L$ y $\mathcal{O}' : L' \rightarrow \text{Obs}_{L'}$, definidas sobre sus semánticas, tal que Obs_L y $\text{Obs}_{L'}$ son algunos de los posibles dominios. Entonces, decimos que L es más expresivo que L' (o que L' puede ser embebido por L), si existe un mapping \mathcal{C} (compilador) de los estados de L' a los estados de L ($\mathcal{C} : L' \rightarrow L$) y un mapping \mathcal{D} (decodificador) de Obs_L a $\text{Obs}_{L'}$ tal que para todo estado $A \in L' : \mathcal{D}(\mathcal{O}(\mathcal{C}(A))) = \mathcal{O}'(A)$.

$$\begin{array}{ccc} L' & \xrightarrow{\mathcal{O}'} & \text{Obs}_{L'} \\ \downarrow \mathcal{C} & & \uparrow \mathcal{D} \\ L & \xrightarrow{\mathcal{O}} & \text{Obs}_L \end{array}$$

(1) \mathcal{R}	$\langle act \cdot P, C \rangle \xrightarrow{act} \langle P, C \rangle$
(2) \mathcal{R}	$\frac{\langle P_1, C \rangle \xrightarrow{\overline{act}} \langle P'_1, C \rangle}{\langle P_1 + P_2, C \rangle \xrightarrow{\overline{act}} \langle P'_1, C \rangle}$
(3) \mathcal{R}	$\frac{\langle P_1, C \rangle \xrightarrow{\overline{act}} \langle P'_1, C \rangle}{\langle P_1 \parallel P_2, C \rangle \xrightarrow{\overline{act}} \langle P'_1 \parallel P_2, C \rangle}$
(4) \mathcal{R}	$\frac{\langle P_1, C \rangle \xrightarrow{\overline{act}_1} \langle P'_1, C \rangle \quad \langle P_2, C \rangle \xrightarrow{\overline{act}_2} \langle P'_2, C \rangle}{\langle P_1 \parallel P_2, C \rangle \xrightarrow{\overline{act}_1 \uplus \overline{act}_2} \langle P'_1 \parallel P'_2, C \rangle}$
(5) \mathcal{R}	$\frac{\langle P, C \rangle \xrightarrow{\overline{act}} \langle P', C \rangle \quad \langle C, \overline{act} \rangle \xrightarrow{\overline{act}}_C^* \langle C', \emptyset \rangle}{\langle P, C \rangle \longrightarrow \langle P', C' \rangle}$
(6) \mathcal{R}	$\frac{\langle P_1, C \rangle \xrightarrow{\overline{act}_1} \langle P'_1, C \rangle \quad \langle P_2, C \rangle \xrightarrow{\overline{act}_2} \langle P'_2, C \rangle \quad \langle C, \overline{act} \rangle \xrightarrow{\overline{act}_1}_C^* \langle C', \overline{act}_2 \rangle}{\langle P_1 \parallel P_2, C \rangle \longrightarrow \langle P'_1 \parallel P'_2, C' \rangle}$

Cuadro 1: Sistema de transiciones para el cálculo \mathcal{R}

De Boer y Palamidessi propusieron en [6] tres nuevas restricciones sobre el codificador \mathcal{C} y el decodificador \mathcal{D} , para obtener una definición de inmersión modular adecuada para comparar lenguajes

Definición 1 *Los observables en \mathcal{R} son dados por la función $O_{\mathcal{R}} : \mathcal{R} \rightarrow Obs_{\mathcal{R}}$ definida por :*

$$ss \in O_{\mathcal{R}}(P, C) \text{ if } \langle P, C \rangle \rightarrow^* \langle 0_{\mathcal{R}}, C' \rangle$$

$$ff \in O_{\mathcal{R}}(P, C) \text{ if } \exists Q \neq 0_{\mathcal{R}}. \langle P, C \rangle \rightarrow^* \langle Q, C' \rangle \not\rightarrow$$

Comparamos la potencia expresiva de \mathcal{L} y \mathcal{R} en presencia de un conector LR como el introducido en la sección previa. El compilador que consideramos es el siguiente:

Definición 2 (Compilador) *Definimos el compilador $\mathcal{C} : \mathcal{L} \longrightarrow \mathcal{R}$ por:*

$$\begin{aligned} \mathcal{C}(in(t).P) &=^{def} tk(c_o, t).\mathcal{C}(P) \\ \mathcal{C}(out(t).P) &=^{def} wr(c_i, t).\mathcal{C}(P) \\ \mathcal{C}(rd(t).P) &=^{def} rd(c_i, t).\mathcal{C}(P) \\ \mathcal{C}(P_1 +_{\mathcal{L}} P_2) &=^{def} \mathcal{C}(P_1) +_{\mathcal{R}} \mathcal{C}(P_2) \\ \mathcal{C}(P_1 \parallel_{\mathcal{L}} P_2) &=^{def} \mathcal{C}(P_1) \parallel_{\mathcal{R}} \mathcal{C}(P_2) \\ \mathcal{C}(0_{\mathcal{L}}) &=^{def} \mathcal{C}(0_{\mathcal{R}}) \end{aligned}$$

Proposición 3 () *Dados 2 procesos $P, P' \in \mathcal{L}$,*

1. Si $P \parallel_{\mathcal{L}} St \xrightarrow{\alpha} P' \parallel_{\mathcal{L}} St'$ ($\alpha \neq \bar{t}$) entonces $\langle \mathcal{C}(P), LR_{St} \rangle \xrightarrow{\alpha_{\mathcal{R}}} \langle \mathcal{C}(P'), LR_{St'} \rangle$, donde

$$\alpha_{\mathcal{R}} = \begin{cases} \tau & \text{if } \alpha = \tau \\ tk(c_o, t) & \text{if } \alpha = t \\ rd(c_o, t) & \text{if } \alpha = \underline{t} \end{cases}$$

2. Si $P \parallel_{\mathcal{L}} St \xrightarrow{\bar{t}} P' \parallel_{\mathcal{L}} St'$ entonces $\exists j > 0$ donde $LR_{St}^o = c_o^j$ y $LR_{St'}^o = c_o^{j+1}$.

Prueba: La prueba es por inducción estructural sobre la complejidad de las reglas en el language \mathcal{L} .

El resultado previo nos muestra la relación entre los sistemas de transición de ambos cálculos. Esta relación emerge debido a que el comportamiento del conector tipo *bag* en Reo permite simular las acciones de Linda sobre el espacio de tuplas.

Considerando la definición del compilador \mathcal{C} , la noción de observables y el decodificador \mathcal{D} que fue definido como la identidad se puede probar el teorema siguiente:

Teorema 4 *El compilador \mathcal{C} es una inmersión modular*

Prueba:

Las propiedades enunciadas en \square se satisfacen considerando la noción de observables, el hecho de que \mathcal{D} es la identidad y la definición del compilador. Para probar $\mathcal{O}(\mathcal{C}(A)) = \mathcal{O}'(A)$ se procede probando la doble inclusión, considerando los elementos posibles de los conjuntos de observables.

Es posible definir una inmersión modular en el sentido opuesto, considerando el mismo conector. Ambas inmersiones nos habilitan a afirmar que ambos modelos (basados en Linda y Reo) son igualmente expresivos al considerar el caso asíncrono y un conector de tipo *bag*.

4. Conclusions

Los trabajos realizados sobre lenguajes y modelos de coordinación tienen como objetivo principal la interoperabilidad de aplicaciones software, normalmente descritas en lenguajes de programación distintos. El modelo de interacción presentado permite ampliar la información que proporcionan las interfaces de componentes con el fin de evitar multitud de problemas, como los que surgen a nivel de protocolos, y así asegurar que el comportamiento cooperativo será el esperado. La noción de canales móviles nos permite establecer comunicaciones entre componentes y habilita reconfiguraciones dinámicas de conexiones en un sistema. Esto refuerza la reusabilidad de sistemas en los que es posible reemplazar una componente por otra versión sin tener que cambiar otras componentes en el sistema. Reo posee una gran potencia expresiva de por sí. Sin embargo hemos comparado la expresividad de ambos modelos de descripción de sistemas concurrentes, y entre las diversas herramientas disponibles utilizamos la inmersión modular estudiando el caso asíncrono, en el que el conector empleado en el modelo Reo presenta un comportamiento equivalente a un canal de tipo *bag*. Como consecuencia del estudio, observamos que, para este caso concreto, ambos modelos son igualmente expresivos. Aunque el resultado principal obtenido era previsible, dado que el conector considerado en Reo se comporta de forma similar a como lo hace el espacio de tuplas compartido de Linda, este trabajo constituye un punto de partida para

analizar la expresividad entre las dos alternativas más influyentes en el contexto de los modelos de coordinación. Como conclusión de este trabajo hemos de resaltar la viabilidad del enfoque seguido para realizar la comparación mencionada, que nos permitirá analizar la expresividad de otros conectores de Reo frente a primitivas adicionales de Linda. Dicho enfoque se basa en (1) homogeneizar ambos modelos a través de la definición de respectivos cálculos de procesos y (2) el uso de la noción de inmersión modular.

En particular, se demuestra que ambos modelos son igualmente expresivos para un tipo concreto de conector asíncrono. Sin embargo el enfoque utilizado y la homogeneización de las dos familias de lenguajes bajo cálculos similares permitirá realizar una comparación exhaustiva de ambos modelos de coordinación.

Nuestro trabajo futuro estará destinado a estudiar profundamente mas conectores de Reo y definir un lenguaje de descripción de interacciones basado en Reo para la coordinación de componentes al estilo de lo realizado en el contexto de Linda. A partir del análisis exhaustivo de la expresividad de los dos modelos descritos, nuestra intención es explorar si uno es más adecuado que el otro en determinadas circunstancias para expresar la interacción entre componentes. No descartamos la posibilidad de que partes de un sistema puedan ser convenientemente descritos con un formalismo, mientras que otras sean más susceptibles de ser especificadas con el otro modelo.

Referencias

- [1] S. Amaro, E. Pimentel and A.M. Roldán. Modelos de Interacción para la Coordinación de Componentes. In *Proceedings of IDEAS'03*, 2003.
- [2] S. Amaro, E. Pimentel and A.M. Roldán. Modelos Basados en tuplas compartidas y en canales de comunicación. In *Proceedings of PROLE'03*, 2003.
- [3] F. Arbab. The IWIM model for coordination of concurrent activities. In *First International Conference on Coordination Models, Languages and Applications (Coordination'96)*, LNCS, pages 34–56, 1996.
- [4] F. Arbab. A Channel-based Coordination Model for Component Composition. In *Proceedings of the 16th European Conference on Object-Oriented Programming 2002*.
- [5] F. Arbab and J.J.M.M. Rutten. A coinductive calculus of component connectors. *CWI Report SEN- R0216 ISSN 1386-369X* 2002.
- [6] F.S. de Boer and C. Palamidesi. Embedding as a tool for language comparison. In *Information and Compt*, 108(1):128-157, 1991.
- [7] A. Bracciali, A. Brogi, and F. Turini. Coordinating Interaction Patterns. In *Proceedings of 16th ACM Symposium on Applied Computing*, 2001.
- [8] C. Canal. *Un Lenguaje para la Especificación y Validación de Arquitecturas de Software*. PhD thesis, Dept. Lenguajes y Ciencias de la Computación. University of Málaga, 2001.
- [9] C. Canal, L. Fuentes, E. Pimentel, J. Troya, and A. Vallecillo. Extending Corba Interfaces with Protocols. *The Computer Journal*, 44(5):448–462, 2001.

- [10] D. Gelernter and N. Carriero. Coordination Languages and Their Significance. *Communications of de ACM*, 35(3):97–107, 1992.
- [11] G. T. Leavens and M. Staraman, editors. *Foundations of Component-Based Systems*. Cambridge University Press, 2000.
- [12] A. Roldán, E. Pimentel and A. Brogi. Safe Composition of Linda-based Components, TACoS'03(ETAPS'03). *Electronic Notes in Theoretical Computer Science*, 82 No. 6, 2003.