Модел. и анализ информ. систем. Т.**15**, №**1** (2008) 3–9

УДК 681.327

Хореография сервисов в семантической сервис-ориентированной архитектуре

Кашалкин Д.Ю., Курчидис В.А.
Ярославский государственный университет
150 000, Ярославль, Советская, 14

получена 10 сентября 2007

Аннотация

Вводятся основные понятия и архитектурные компоненты, используемые при решении задачи хореографии сервисов в семантической сервис-ориентированной архитектуре. Разработан алгоритм решения задачи хореографии, позволяющий обнаружить и интегрировать элементы CCOA, необходимые для выполнения запроса к системе. Алгоритм использует модель семантического реестра, представленную в виде сети Петри, и анализирует дерево достижимости сети на предмет соответствия семантической аннотации сервисов и семантического описания запроса к системе.

1. Введение

Вопросы интеграции и интероперабельности разнородных компонентов являются одними из наиболее важных при построении распределенных информационных систем. Эффективное решение данных вопросов позволит построить гибкие системы, способные адаптироваться к изменяющимся условиям динамичной среды. В настоящее время существуют различные подходы к созданию интеграционных решений в распределенных системах, среди которых следует выделить сервис-ориентированный подход (COA) [1,2] и Semantic Web [3,4]. Сервисы создают сеть агентов, программ, которые обрабатывают информацию и позволяют реализовать автоматическое взаимодействие компонентов систем. Semantic Web предоставляет средства для эксплицитного задания контента распределенной системы. Существенным недостатком систем, построенных на базе COA или Semantic Web, является неспособность таких систем эффективно решать задачу хореографии компонентов. Задача хореографии возникает, когда ни один из отдельно взятых компонентов не способен выполнить запрос к системе. В этом случае необходимо обнаружить и интегрировать несколько компонентов системы, совместное выполнение которых позволит реализовать целевую функцию, характеризующую собой запрос к системе. Для решения задачи хореографии в распределенных информационных системах предлагается объединить существующие интеграционные решения COA и Semantic Web для создания новой архитектуры и разработать алгоритм решения задачи хореографии для данной архитектуры. Предлагается рассматривать объединение COA и Semantic Web в виде новой семантической сервис-ориентированной архитектуры (CCOA, Semantic SOA, SSOA), основная идея которой заключается в добавлении семантического описания к компонентам COA [5]. Расширение элементов сервис-ориентированной архитектуры семантической аннотацией позволяет реализовать автоматический поиск и селекцию основных компонентов системы - сервисов, что дает возможность повысить гибкость информационных систем, построенных на основе CCOA. Принципы построения данной архитектуры, ее основные элементы и базовые типы взаимодействия между ними рассматриваются в работе [6]. В статье предлагается алгоритм решения задачи хореографии в CCOA, основанный на обнаружении и интеграции компонентов системы на базе сопоставления их семантической аннотации и семантического описания целевой функции.

2. Основные понятия

В CCOA сервис (S) формально представляется как некоторая функция, правило, позволяющее преобразовать набор входных параметров в выходные [6]. В терминах архитектуры сервис рассматривается как событийно-управляемый компонент: он реагирует на входящие сообщения, выполняет внутренние действия и формирует исходящие сообщения. Формализованное описание сервиса имеет вид:

$S : Inp \rightarrow Out$, где Inp - конечное множество входных параметров $\{Inp^1, Inp^2, ..., Inp^n\}$, а Out - конечное множество выходных параметров $\{Out^1, Out^2, ..., Out^m\}$.

Расширенное представление сервиса заключает в себе описание условий, предшествующих выполнению сервиса (так называемых предусловий - Pre), и изменений предметной области, порождаемых выполнением сервиса. В этом случае в множестве Out выделяется два подмножества:

- $Output$ - непосредственно выходные параметры;

- $Effects$ - эффекты, которые сервис может оказывать на предметную область, изменяя состояние окружающей его среды.

$S : Pre \rightarrow Output \cup Effects$

Данная формула отражает необходимое условие выполнения сервиса. Необходимое и достаточное условие имеет вид:

$S : Pre \cup Inp \rightarrow Output \cup Effects$

Множества $Inp, Pre, Output, Effects$ задаются в терминах онтологий [7, 8, 9], которые определяют понятия и отношения предметной области. Для задания онтологий используется формализм дескриптивной логики [10]. Далее вводятся основные понятия, используемые при решении задачи хореографии сервисов в ССОА.

Определение 1. Семантический реестр представляет собой хранилище множества семантических сервисов $S = \{S_1, S_2, ..., S_k\}$, подмножество $S^{'} = \{S^{'}_1, S^{'}_2, ..., S^{'}_{k^{'}}\}$ которого составляют взаимосвязанные сервисы.

Определение 2. Два сервиса $S_i, S_j : S_i, S_j \in S^{'}; i, j \in [1, k^{'}], i \neq j, S_{i,j} : Inp_{i,j} \rightarrow Out_{i,j}, Inp_{i,j} = \{Inp^1_{i,j}, Inp^2_{i,j}, ..., Inp^{n_{i,j}}_{i,j}\}, Out_{i,j} = \{Out^1_{i,j}, Out^2_{i,j}, ..., Out^{m_{i,j}}_{i,j}\}$ определяются как взаимосвязанные, если они связаны отношением выводимости или отношением частичного вывода.

Определение 3. Два сервиса $S_i, S_j : S_i, S_j \in S^{'}; i, j \in [1, k^{'}], i \neq j, S_{i,j} : Inp_{i,j} \rightarrow Out_{i,j}, Inp_{i,j} = \{Inp^1_{i,j}, Inp^2_{i,j}, ..., Inp^{n_{i,j}}_{i,j}\}, Out_{i,j} = \{Out^1_{i,j}, Out^2_{i,j}, ..., Out^{m_{i,j}}_{i,j}\}$ связаны отношением выводимости (сервис S_j выводится из S_i), если множество/подмножество выходных данных сервиса S_i может служить в качестве всего множества входных данных сервиса S_j, т.е. выполняется условие $\forall p \in [1, n_j] : \exists l \in [1, n_i] : Inp^p_j \subseteq Out^l_i$ (рис. 1а).

Определение 4. Два сервиса $S_i, S_j : S_i, S_j \in S^{'}; i, j \in [1, k^{'}], i \neq j, S_{i,j} : Inp_{i,j} \rightarrow Out_{i,j}, Inp_{i,j} = \{Inp^1_{i,j}, Inp^2_{i,j}, ..., Inp^{n_{i,j}}_{i,j}\}, Out_{i,j} = \{Out^1_{i,j}, Out^2_{i,j}, ..., Out^{m_{i,j}}_{i,j}\}$ связаны отношением частичного вывода (сервис S_j частично выводится из S_i), если множество/подмножество выходных данных сервиса S_i может служить в качестве подмножества (но не всего множества) входных данных сервиса S_j, т.е. выполняется условие $\exists p \in [1, n_j] : \exists l \in [1, n_i] : Inp^p_j \subseteq Out^l_i$ (рис. 1б).

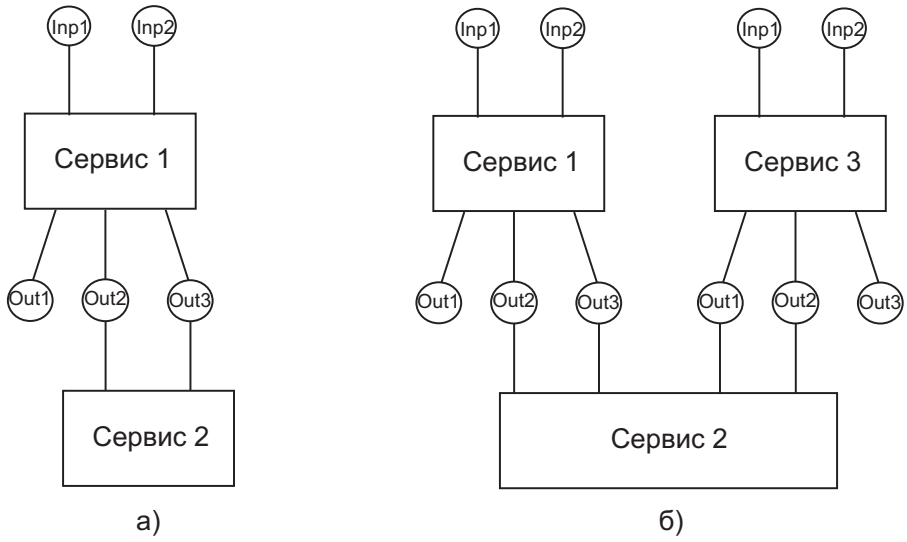

Рис. 1. Взаимосвязанные сервисы

Определение 5. Хореографией множества сервисов $S_{comp} = \{S^1_{comp}, S^2_{comp}, ..., S^{k_{comp}}_{comp}\}, S_{comp} \in S^{'}, S_{comp} : Inp_{comp} \rightarrow Out_{comp}, Inp_{comp} = \{Inp^1_{comp}, Inp^2_{comp}, ..., Inp^{n_{comp}}_{comp}\}, Out_{comp} = \{Out^1_{comp}, Out^2_{comp}, ..., Out^{n_{comp}}_{comp}\}$ по отношению к целевому сервису $S_g : Inp_g \rightarrow Out_g, Inp_g =$

$\{Inp_g^1, Inp_g^2, ..., Inp_g^{n_g}\}, Out_g = \{Out_g^1, Out_g^2, ..., Out_g^{n_g}\}$ называется совокупность сервисов семантического реестра (метасервис) таких, что выполняются следующие два условия:

1. $\forall S_c^i omp \in S_{comp} : Inp_{S_c^i omp} \subseteq Inp_{comp} \rightarrow Inp_{S_c^i omp} \subseteq Inp_g$.
2. $Out_{S_{comp}^i} \subseteq Out_{comp} \rightarrow (Out_g \subseteq Out_{S_{comp}^i}) \cup (Out_{S_{comp}^i} \subseteq Out_g)$.

При этом сервисы S_{i-1}, S и S, S_{i+1} являются взаимосвязанными $\forall i \in [2, k-1]$. Графически хореографию сервисов можно изобразить в виде схемы, представленной на рисунке 2. Исполнение хореографии

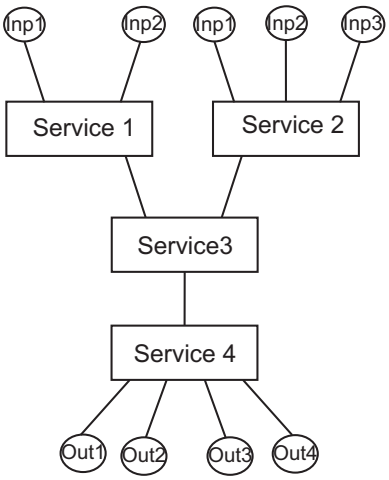

Рис. 2. Хореография сервисов

представляет собой инициализацию входных параметров метасервиса, собственно выполнение метасервиса и возврат его выходных данных.

3. Сценарий взаимодействия элементов ССОА при решении задачи хореографии

Для решения задачи хореографии рассматриваются следующие основные элементы ССОА: пользователь, реестр, машина логического вывода, онтологии. Предлагается выделить следующие элементы реестра: анализатор публикаций, хранилище сервисов, парсер запроса пользователя, блок работы с сетью Петри, транслятор в BPEL. Используя такую модель семантического реестра, можно строго определить хореографию сервисов, проверить ее выполнимость. Взаимодействие элементов ССОА при решении задачи хореографии сервисов представлено в виде схемы на рис. 3.

Можно предложить следующий сценарий взаимодействия элементов ССОА:

1. Провайдер публикует сервис в реестре, анализируются связи публикуемого сервиса с другими сервисами реестра, происходит сохранение описания сервиса в БД реестра. При размещении сервиса в реестре и описании его входных и выходных параметров анализатор публикаций анализирует уже содержащиеся в реестре сервисы на отношение выводимости нового сервиса и уже опубликованных в реестре. Определение взаимосвязанных сервисов на стадии их опубликования в реестре позволяет задать машину вывода, которая при инициализации входных данных целевого сервиса способна вывести возможные хореографии сервисов. Работа анализатора публикаций основана на использовании табличного алгоритма дескриптивной логики машины логического вывода.

2. Пользователь задает некоторую цель, конечный результат, описывает целевой сервис - т.е. тот сервис, который он хочет найти. Описание конечной цели пользователя задается в том же формате, что и описание сервиса: $S_g : Inp_g \rightarrow Out_g$

3. Парсер запроса выделяет в целевом сервисе входные и выходные данные, которые передает в блок работы с сетью Петри. На основании этих данных осуществляется поиск целевого сервиса. В случае

Рис. 3. Взаимодействие элементов ССОА при решении задачи хореографии сервисов

успешного обнаружения сервиса возвращается ссылка на его описание; в противном случае осуществляется попытка составить требуемый сервис из имеющихся описаний. Определение этих сервисов и порядок их запуска представляет собой цель хореографии. Поиск нужных сервисов происходит на основе определения семантического соответствия существующих описаний сервисов запросу пользователя, поэтому хореография носит семантический характер. В случае успешного решения задачи хореографии BPEL транслятор преобразует описания интегрируемых сервисов в BPEL описание и возвращает его пользователю. Множество взаимодействующих сервисов порождает систему со сложной динамической структурой, анализ поведения которой невозможно осуществить без моделирования. Если рассматривать сервисы как события в системе, а наличие входных данных сервисов - как условия выполнения событий, то динамика поведения такой системы может быть удобно отражена при помощи сети Петри, определяемой следующим образом:

$R = (P, S, I, O)$,

где $P = \{p_1, p_2, ..., p_{n+m}\}, n, m \geq 0$, ( )- множество позиций, моделирующих условия, т.е.

$$p_i = \left\{ \begin{array}{l} Inp_i^j \\ Out_i^j \end{array} \right.$$

$S = \{S_1, S_2, ..., S_k\}, k \geq 0$ - множество переходов (сервисов);

$P \cup S = \emptyset$;

I - входная функция $I : P \to S$;

O - выходная функция $O : S \to P$.

Пусть сеть, построенная таким образом, является свободно помеченной с некоторой функцией помечения $\sigma : S \to \Sigma$, где Σ - алфавит. Схема работы сервиса в сети: сервис Si получает сообщение, выполняет некоторые действия, посылает сообщение. После этого сервис готов к принятию новых сообщений. Выполнение действий сервисом возможно только при условии наличия входных параметров, что соответствует наличию фишек в соответствующих позициях p_i сети R. Поэтому процесс выполнения сервиса в сети R представляет собой запуск разрешенного перехода. Взаимосвязанные сервисы в сети являются строками алфавита Σ.

Ключевым элементом решения задачи хореографии является алгоритм, который позволяет обнаружить и запустить на выполнение одну или несколько хореографий, выходные данные которых представляют собой выходные данные целевого сервиса. Предлагаемый алгоритм состоит из следующих этапов:

1. Определение начальной маркировки сети Петри семантического реестра.

2. Построение и анализ дерева достижимости с целью поиска сервиса(ов), удовлетворяющих запросу пользователя, и определения последовательности выполнения сервисов (в случае если цель пользователя может быть выполнена реализацией нескольких сервисов).

3. Завершение алгоритма представляет собой возврат пользователю ссылки на сервис или последовательность сервисов.

4. Алгоритм решения задачи хореографии

4.1. Определение начальной маркировки сети Петри семантического реестра

Этап определения начальной маркировки M необходим для определения сервисов, которые пользователь может запустить, используя множество начальных параметров Inp_g. Далее эти данные используются для построения пространства состояний сети Петри и его анализа на предмет семантического соответствия позиций сети R цели пользователя Out_g.

$M = \{\mu_1, \mu_2, ..., \mu_{n+m}\}, \mu_i \in \{0, 1\}, i = 1, ..., n + m,$

где μ_i - маркировка позиции p_i , т.е. $\mu : P \to N$, где N - множество неотрицательных целых чисел.

Основная информация, получаемая на данном этапе, касается того, может ли пользователь запустить сервис или нет, т.е. обладает ли он нужными входными данными или нет. Поэтому следует рассматривать ординарную сеть, т.е. кратность любой позиции 0 или 1: $\sharp(p_i, I(S_j)) \leq 1$ и $\sharp(p_i, O(S_j)) \leq 1$.

Определения семантического соответствия данных, которыми обладает пользователь, и данных, которые необходимы для запуска сервисов, сводятся к решению задачи выводимости Inp_g из дескриптивного логического выражения, описывающего в терминах предметной области условия запуска сервиса, соответствующего позиции p_i в сети, моделирующей реестр. В свою очередь в формализме ДЛ задача выводимости сводится к задаче категоризации. Поэтому задача $Inp_g \models p_i$ сводится к задаче $Inp_g \subseteq p_i$. При решении данной задачи возможны следующие варианты:

1. Пользователь может ввести данные, которые категоризуют входные данные сервиса S_i , т.е. $Inp_g \subseteq I(S_i)$. В этом случае пользователь вводит более общие данные.

2. Пользователь может ввести данные, которые категоризуются входными данными сервиса S_i , т.е. $I(S_i) \subseteq Inp_g$. В этом случае пользователь не может ввести все необходимые данные.

3. Пользователь может ввести те данные, которые необходимы для запуска сервиса, т.е. $Inp_g \subseteq I(S_i)$ и $I(S_i) \subseteq Inp_g$.

4. Сервис использует входные данные, которые пользователь не может ввести, т.е. $Inp_g \cap \neg I(S_i) = \emptyset$.

На этапе определения начальной маркировки реестра необходимо проверить множество всех входных позиций переходов $p_i : p_i \in I(S_j), \forall i, \forall j$ на отношение категоризации Inp_g и p_i и провести маркировку фишек. Тогда

$$\mu_i = \begin{cases} 0, \forall j : Inp_{g_j} : (Inp_{p_i} \subseteq Inp_{g_j}) \cup (Inp_{p_i} \cap Inp_{g_j}) \subseteq \bot \\ 1, \exists j : (Inp_{g_j} \subseteq Inp_{p_i}) \cup (Inp_{g_j} \equiv Inp_{p_i})) \end{cases}$$

4.2. Построение и анализ дерева достижимости

Следующий этап решения задачи хореографии состоит в построении дерева достижимости сети Петри R, анализ которого используется для поиска целевого сервиса или определения сервисов, участвующих в хореографии. Целесообразно задать следующие правила построения дерева :

1. Определяется начальная маркировка сети R (корневая вершина дерева) и разрешенные переходы.

2. Любая вершина M_i дерева представляет собой маркировку $\{\mu_1, \mu_2, ..., \mu_{n+m}\}$ позиций сети R, полученную запуском какого-либо разрешенного перехода, удалением фишек из входных позиций этого перехода и помещением фишек в его выходные позиции.

3. Если переход S_i разрешен, то после его выполнения переход S_j также будет разрешен, при условии, что соответствующие сервисы S_i, S_j связаны отношением вывода;

4. Если переход S_i разрешен, а соответствующие сервисы S_i, S_j и S_h, S_j связаны отношением частичного вывода, то после выполнения перехода S_i переход S_j также будет разрешен, при условии, что переход S_h уже был выполнен на одном из этапов построения дерева ; в противном случае переход S_j будет запрещен.

После построения дерева происходит анализ его вершин, конечной целью которого является обнаружение целевого сервиса или сервисов, необходимых для хореографии. Соответственно выбираются только те вершины, семантическое описание которых соответствует описанию выходных параметров сервиса пользователя. Выбор осуществляется на основании оценочной функции F, которую предлагается задавать следующим образом:

$$F(M_i) = \sum_{j=1}^{n} V(\mu_j),$$

где

$$V(\mu_j) = \begin{cases} 0, \exists q : (Out_j^q \equiv Out_p^j) \cup (Out_j^q \subseteq Out_p^j) \\ 1, \exists q : Out_p^j \subseteq Out_j^q \\ 2, \forall q : Out_p^j \cap Out_j^q \subseteq \perp \end{cases}$$

При таком задании оценочной функции выбор вершин с минимальным значением F на каждом этапе анализа гарантирует получение маркировки сети, которая наилучшим образом удовлетворяет запросу пользователя. Анализ вершин дерева с использованием оценочной функции осуществляется на основании жадного алгоритма [11]. Перед запуском алгоритма создается список целей пользователя, т.е. $Goals = \{Out_g^1, Out_g^2, ..., Out_g^{g_k}, \}$, список отобранных вершин $Selected = \{M_s^1, M_s^2, ..., M_s^{s_k}\}$ и список посещенных вершин $Visited = \{M_v^1, M_v^2, ..., M_v^{v_k}\}$.

Работа алгоритма осуществляется следующим образом:

1) для текущей вершины M_i рассчитывается оценочная функция вершин-потомков $F(M_j)$;

2) выбираются вершины-потомки M_j с минимальным значением функции $F(M_j)$;

3) вершина M_i помещается в список $Visited$;

4) проверяются правила:

   - если для вершины $p_i : p_i = mark^{-1}(\mu_j), \exists j : (Out_g^j = Out_p^i) \cup (Out_g^j \subseteq Out_p^i)$, тогда из списка $Goals$ исключается Out_g^j , а вершина M_i заносится в список $Selected$;
   - если для вершины $p_i : p_i = mark^{-1}(\mu_j), \exists j : Out_p^i \subseteq Out_g^j$, тогда из списка $Goals$ исключается Out_g^j и заносится новый элемент $Out_g^j - Out_p^i \equiv Out_g^j \cap \neg Out_p^i$, а вершина M_i заносится в список $Selected$;
   - если ни одно из вышеперечисленных правил не выполняется, то M_i не заносится в список $Selected$.

4.3. Завершение алгоритма

Завершение алгоритма происходит, когда все цели пользователя выполнены, т.е. список $Goals$ пуст, а список $Selected$ имеет вид: $Selected = \{M_1, M_2, ..., M_h\}$. Следует отметить, что вершины данного списка в силу особенностей его заполнения обладают следующими свойствами:

1. $(O(S_i^k) \subseteq Out_g) \cup (Out_g \subseteq O(S_i^k))$;
2. $\forall M_i \exists \{S_i^1, S_i^2, ..., S_i^k\} : I(S_i^l) \subseteq Inp_g$, где $\{S_i^1, S_i^2, ..., S_i^k\}$ - последовательность переходов, выполнение которых приведет состояние сети R от начальной разметки (корневой вершины дерева достижимости) до разметки, обозначенной вершиной M_i.

Очевидно, что, заменив переходы $\{S_i^1, S_i^2, ..., S_i^k\}$ на соответствующие сервисы, получим, что S_{i-1}, S_i и S_i, S_{i+1} являются взаимосвязанными $\forall i \in [2, k-1]$. Таким образом, последовательность сервисов $\{S_i^1, S_i^2, ..., S_i^k\}$ представляет собой хореографию сервисов, а множество таких хореографий (совокупность последовательностей переходов для вершин дерева , включенных в список $Selected$) представляет собой метасервис, выполнение которого означает реализацию целевого сервиса. Т.е. по построенному списку $Selected$ определяются интегрируемые сервисы, реализующие целевой сервис. Семантические описания сервисов, участвующих в хореографии, транслируются в язык описания процессов, например, BPEL. Ссылка на описание возвращается пользователю.

5. Заключение

Построение распределенных систем на основе слабосвязанных компонентов семантической сервис-ориентированной архитектуры позволяет повысить гибкость информационных систем. Важным фактором повышения адаптивности таких систем является эффективное решение задачи хореографии сервисов. В работе предложены алгоритмы обнаружения и хореографии сервисов на основе определения семантического соответствия между описаниями запросов пользователей и описаниями сервисов семантического реестра. В настоящее время использование ряда технологий (OWL DL, BPEL, Pellet) позволяет реализовать семантические сервис-ориентированные системы и семантическую хореографию их элементов. Эффективное решение задачи хореографии сервисов является особенно актуальным для систем с непрерывно изменяющимся окружением, таким как, например, приложения электронного бизнеса. На сегодняшний день в подобных системах наметилась тенденция перехода от интеграции данных и бизнес-процессов к "решениям по запросу". Это приложения, которые автоматически компонуются из доступных элементов распределенной системы для выполнения запроса пользователей. Алгоритм, рассмотренный в статье, позволяет реализовать подобные приложения.

Список литературы

1. Stojanovic, Z. Service-oriented Software System Engineering / Z. Stojanovic. - Idea Group Publishing, 2005. - 435 с.

2. Черняк, Л. SOA - шаг за горизонт / Л. Черняк // Открытые системы. - 2003. - №9. - С. 31-35.

3. Berners-Lee, T. What the Semantic Web can represent. -http://www.w3.org/DesignIssues/RDFnot.html.

4. Кашалкин, Д.Ю. Принципы организации Semantic Web /Д.Ю. Кашалкин, А.Н. Лататуев // Модел. и анализ информ. систем. - 2005. - Т.12. N2. - С. 7-11.

5. Курчидис, В.А. Принципы организации Web-служб в системах с семантической сервис-ориентированной архитектурой / В.А. Курчидис, Д.Ю. Кашалкин, А.С. Назанский; Яросл. гос. ун-т. - Ярославль, 2006. - Деп. в ВИНИТИ 20.04.2006, № 526-В2006.

6. Курчидис, В.А. Принципы построения семантической сервис-ориентированной архитектуры / В.А. Курчидис, Д.Ю. Кашалкин // Модел. и анализ информ. систем. - 2007. - Т.14. N1. - С. 47-52.

7. Кашалкин, Д.Ю. Принципы создания метаописаний ресурсов во всемирной сети /Д.Ю. Кашалкин, А.Н. Лататуев // Модел. и анализ информ. систем. - 2005. - Т.12. N2. - С. 12-16.

8. Taniar, D. Web semantics and ontology / D. Taniar. - Idea Group Publishing, 2006. - 423 с.

9. Fensel, D. Ontologies: Silver Bullet for Knowledge Management and Electronic Commerce / D. Fensel. - Springer, 2003. - 104 с.

10. Baader, F. The description logic handbook - Theory, implementation and applications / F. Baader. - Cambridge University Press, 2003. - 583 с.

11. Люггер, Д. Искусственный интеллект / Д. Люггер. - М., 2003. - 848 с.

Service choreography in semantic service-oriented architecture

Kashalkin D.Y., Kurchidis V.A.

Basic concepts and architecture's components which are using in decision of service's choreography problem in the semantic service-oriented architecture are introduse in the article. The decision algorithm of the service choreography problem has been developed, which lets find and integrate service-oriented system's components, necessary to fulfil system's query. The algorithm uses semantic register's model, presented in the form of Petri net and analyzing net's accessibility tree to find the compliance of service's semantic annotations and system's query semantic description.