УДК 519.876.5

К.О. Устимов, Н.В. Федоров

АЛГОРИТМ ПРЕОБРАЗОВАНИЯ IDEF0-МОДЕЛИ В РАСКРАШЕННУЮ СЕТЬ ПЕТРИ

Рассмотрен алгоритм преобразования IDEF0-модели в раскрашенную сеть Петри. IDEF0-модель используется для функционального моделирования бизнес-процессов. Раскрашенная сеть Петри позволяет проводить имитационное моделирование. Предоставлено математическое описание IDEF0-модели и раскрашенной сети Петри, на основе которых изложен рассматриваемый алгоритм.

Множество цветов в CPN по сути является типом данных языка моделирования, однако синтаксис CPN ML не позволяет оперировать непосредственно типами данных, необходимо определить множества цветов.

Мультимножества применяются в маркировках позиций, выражениях дуг и др. элементов. Мультимножества расширяют понятие фишек сетей Петри.

Рассмотрен алгоритм, позволяющий преобразовать в CPN любую IDEF0-диаграмму стандартного вида, не содержащую туннелей и стрелок вызовов. Полученная модель будет являться базовой имитационной моделью бизнес-процессов.

Ключевые слова: методология IDEF0, раскрашенные сети Петри, алгоритм, математическая модель.

Методология IDEF0 широко применяется для моделирования бизнес-процессов. Функциональная модель IDEF0 позволяет описать структуру и функции системы, а также потоки информации и материальных объектов, преобразуемые этими функциями [1].

В процессе моделирования или реинжиниринга бизнес-процессов очень выгодно использовать имитационное моделирование. Оно позволяет определить, как преобразования повлияют на систему, ставя эксперименты не на «живой» системе, а на ее модели.

Хотя IDEF0-модель и не позволяет проводить имитационное моделирование, она содержит достаточно данных для того, чтобы построить базовую имитационную модель в виде раскрашенной сети Петри (CPN) [2]. Для того чтобы описать алгоритм преобразования IDEF0-модели в CPN, рассмотрим математические описания обеих моделей.

Модель IDEF0

Рассмотрим недекомпозированную IDEF0-модель. Она представляет собой одну IDEF0-диаграмму, состоящую из следующих элементов:

- конечное множество блоков: $B = \{b_0, b_1, b_2, b_3, \dots, b_n\}$, где $n \geq 0$. (n+1) — мощность множества B: $|B| = (n+1)$. b_0 — блок, определяющий саму диаграмму;

- конечное множество элементарных объектов: $0 = \{o_1, o_2, o_3, \dots o_t\},$, где $t \geq 0$. t — мощность множества 0. Элементарный объект означает неделимые данные или материальные объекты;

- конечное множество меток стрелок: $M = \{m_1, m_2, m_3, ...m_t\},$, где $k \geq 0$. k — мощность множества M. Каждая метка представляет собой множество элементарных объектов, обозначающих данные и материальные объекты, которые разделяются или объединяются в соответствующем сегменте стрелки. Метку можно обозначить так: $m_i = \{O_{j_1}, o_{j_2}, O_{j_3}, ...o_{j_{ki}}\},$, где $k_i \geq 0$. k_i — мощность множества m_i Множества m_i состоят из элементов множества O. Таки образом, $m_i \subset O$;

- конечное множество сегментов стрелок: $S = \{s_1, s_2, s_3, ..., s_l\}$, где $I \geq 0I$. l — мощность множества S. Каждому элементу S_i множества S ставится в соответствие элемент m_j множества M : $(s_i, m_j) \in C^{sm}$, где C^{sm} — множество соответствий сегментов стрелок и меток;

- функции F^{src} и F^{snk}, определяющие, соответственно, начальные и конечные точки соединения сегментов стрелок. Эти две функции определяются одинаково, поэтому будем обозначать их буквой F^{ss} , когда можно подразумевать их обе. Функция F^{ss} отображает множество сегментов стрелок в множество, являющееся объединением множеств блоков и сегментов стрелок: $Y = B \cup S$: $F^{ss}: S \rightarrow Y$. Например, выражение $F^{src}(s_i) = \{b_{j_1}\}$, $F^{snk}(s_i) = \{s_{j_1}, s_{j_2}, s_{j_3}\}$ означает, что сегмент стрелки s_i начинается на блоке b_{j_1} и заканчивается ветвлением на сегменты s_{j_1}, s_{j_2} и s_{j_3}.

В случае с декомпозированной моделью будут следующие изменения в описании:

- каждая диаграмма будет иметь свое множество блоков: $B_i = \{b_{i_0}, b_{i_1}, b_{i_2}, b_{i_3}, ..., b_{i_n}\}$, где $i_n \geq 0$. $(i_n + 1)$ — мощность множества B_i. b_{i_0} — блок, определяющий саму диаграмму. Множество $B = \bigcup_{i=1}^{d} B_i$ содержит все блоки всех диаграмм и все диаграммы, где d — общее количество диаграмм;

каждая диаграмма будет иметь свое множество сегментов стрелок: $S_i = \{s_{i_1}, s_{i_2}, s_{i_3}, ..., s_{i_n}\}$, где $i_n \geq 0$. i_n — мощность множества S_i. Множество $S = \bigcup_{i=1}^{d} S_i$ содержит все сегменты стрелок всех диаграмм, где d — общее количество диаграмм;

- функция F^{ss} по аналогии с блоками и сегментами стрелок может быть общая: $F^{ss}: S \rightarrow Y$ и частная: $F_i^{ss}: s_i \rightarrow Y_i$;

- конечное множество элементарных объектов остается общим для всех диаграмм;

- конечное множество меток стрелок остается общим для всех диаграмм;

- функция декомпозиции $D:B \rightarrow B$ определяет какая диаграмма декомпозицией какого блока является. Например, выражение $D(b_{i_k}) = b_{v_0}$ означает, что диаграмма b_{v_0} является декомпозицией блока b_{i_k}. Если блок b_{i_k} не имеет декомпозиции, то $D(b_{i_k}) = \varnothing$.

Раскрашенная сеть Петри

Сеть Петри [3] состоит из четырех элементов: множество позиций P, множество переходов T, входная функция F^I и выходная функция F^0. Сеть Петри C является четверкой: $C = (P, T, F^I, F^O)$. Элементы сети Петри:

- конечное множество позиций: $P = \{p_1, p_2, p_3, \dots, p_n\}$, где $n \geq 0$. n — мощность множества P;

- конечное множество переходов: $T = \{t_1, t_2, t_3, \dots, t_m\}$, где $m \geq 0$. m — мощность множества T;

- входная и выходная функции: $F^I: T \rightarrow P$, $F^O: T \rightarrow P$. Например, $F^I(t_j) = \{p_{k_1}, p_{k_2}, p_{k_3}\}$ означает, что позиции p_{k_1}, p_{k_2} и p_{k_3} являются входными по отношению к переходу t_j. Для общего обозначения входной и выходной функций будем использовать F^{IO} F^{IO}.

Множество позиций и множество переходов не пересекаются: $P \cap T = \varnothing$. Таким образом, сеть Петри является двудольным ориентированным мультиграфом.

Раскрашенная сеть Петри имеет такую же структуру, как и обычная сеть Петри, за исключением того, что в CPN не может быть кратных дуг. Т.е., CPN не является мультиграфом. Понятие кратности дуги обычной сети Петри покрывается понятием выражения дуги в CPN. Рассмотрим основные понятия, которые имеются в CPN в отличие от обычных сетей Петри.

Множества цветов

Множество цветов в CPN по сути является типом данных языка моделирования, однако синтаксис CPN ML не позволяет оперировать непосредственно типами данных, необходимо определить множества цветов. Синтаксис объявления множества цветов в языке CPN ML следующий:

colset <имя множества цветов> = <тип данных>;

Здесь <имя множества цветов> — любой возможный идентификатор CPN ML. <тип данных> — простой или сложный тип данных. Вот несколько примеров объявления множеств цветов:

colset NO = int;
colset DATA = string;
colset NOxDATA = product NO*DATA;

Мультимножества

Мультимножества применяются в маркировках позиций, выражениях дуг и др. элементов. Мультимножества расширяют понятие фишек сетей Петри. Синтаксис определения мультимножества в CPN ML такой:

<кол-во фишек>`<значение> ++ <кол-во фишек>`<значение> ++ …

Элементы мультимножества всегда принадлежат к одному и тому же множеству цветов. Рассмотрим следующий пример:

val v = 1`5 ++ 2`10 ++ 7`11;

В этом примере объявляется константа v, значением которой является мультимножество целого типа: одна фишка со значением «5», две фишки со значением «10» и семь фишек со значением «11». Таким образом, в отличие от обычный сетей Петри, каждая фишка может иметь значение определенного типа, а маркировка позиции может содержать разное количество фишек с различными значениями.

Маркировка позиций

В CPN с каждой позицией должно быть связано множество цветов, которое определяет, фишки какого типа могут находиться в данной позиции. Одно и то же множество цветов может быть связано с разными позициями. Для каждой позиции определяются три описания. Первое описание определяет тип фишек, которые могут содержаться в позиции. Например, перечисление определяется так так: colset items = unit with item;

Второе описание определяет начальную маркировку позиции. Например, позиция может иметь начальную маркировку в виде 10-и фишек со значением «item». Третье описание определяет текущую маркировку позиции.

Выражения дуг

Обычная сеть Петри является мультиграфом, что означает, что между позицией и переходом может существовать несколько дуг. Количество этих дуг определяет количество фишек, исчезающих, или добавляющихся в соответствующую позицию при срабатывании соответствующего перехода.

В CPN не принято изображать параллельные дуги. Вместо этого каждая дуга взвешивается определенным выражением. В простейшем случае выражение является мультимножеством или переменной, значением которой является мультимножество. Выражение дуги должно иметь такой же тип, как и множество цветов, связанное с позицией, с которой эта дуга соединена. Выражения дуг могут быть более сложными, например, они могут содержать условия.

Переходы

Переходы в CPN в отличие от обычных сетей Петри обозначаются прямоугольниками. Срабатывание, как и в обычных сетях Петри, изменяет маркировки связанных с ним позиций в соответствии с выражениями дуг. Переходы могут содержать дополнительные объявления на языке CPN ML, которые в рамках данной работы не рассматриваются.

Алгоритм преобразования IDEF0 в CPN

Рассмотрим алгоритм преобразования модели IDEF0, состоящей из d диаграмм, в CPN.

1. Для каждой метки стрелки IDEF0-модели $\forall m_i \in M$ определить множество цветов CPN следующим образом: colset t_m_i = string;

2. Для каждой IDEF0-диаграммы $\forall b_{i_0} \in B, \ i \in [1,d],$, , где B — множество всех блоков IDEF0-модели, выполнить действия:

2.1. Сформировать множество основных переходов CPN T_i^{base}. Для каждого блока IDEF0-диаграммы $\forall b_{i_j} \in B_i,\ j \in [1, n_i],$, где B_i — множество всех блоков диаграммы b_{i_0}, n_i — общее количество блоков диаграммы b_{i_0}, создать переход CPN $t_{i_j} \in T_i^{base}$..

2.2. Сформировать множество позиций CPN P_i и множество дополнительных переходов T_i^{add}. Для каждого сегмента стрелки IDEF0-диаграммы $\forall S_{i_j} \in S_i,\ j \in [1, L_i]$, где S_i — множество всех сегментов стрелок диаграммы bi_0, Li — общее количество сегментов стрелок диаграммы bi_0, выполнить действия:

2.2.1. Создать позицию CPN $Pi_j \in p_i$.

2.2.2. Если $F_i^{ss}(s_{i_j}) \in S_i$, то:

2.2.2.1. Если $\left|F_i^{ss}(s_{i_j})\right| > 1$, создать дополнительный переход CPN $t_{i_j}^{add} \in T_i^{add}$ и соединить дугой позицию Pi_j и переход $t_{i_j}^{add}$.

2.2.2.2. Если $\left|F_i^{ss}(s_{i_j})\right| = 1$, соединить дугой позицию Pi_j и переход $t_{i_k}^{add}$, соответствующий позиции Pi_k (см. п. 1.2.2.1), соответствующей сегменту стрелки $s_{i_k} = F_i^{ss}\left(s_{i_j}\right)$.

2.2.3. Если $F_i^{ss}(s_{i_j}) \in B_i$, соединить дугой позицию Pi_j и переход $t_{i_k}^{base} = F_i^{ss}\left(s_{i_j}\right)$.

2.3. Множество переходов CPN $T_i = T_i^{base} \cup T_i^{add}$.

2.4. Для каждой позиции CPN $\forall P_{i_j} \in P_i$ выполнить действия:

2.4.1. Связать с позицией Pi_j множество цветов t_m_k, соответствующее метке сегмента стрелки Si_j.

2.4.2. Определить начальную маркировку позиции Pi_j в соответствии с множеством цветов t_m_k следующим образом: val init_p_i_j = 1`"data_k";

2.5. Каждой дуге $\forall a_{i_j} = \left(\forall t_{i_k}, \forall p_{i_{k,l}} = F_i^{IO}\left(t_{i_k}\right)\right)$ назначить выражение val a_i_j = 1`"data_r";, где позиция $P_{i_{k,l}}$ имеет начальную маркировку 1`"data_r".

2.6. Для каждого блока IDEF0-модели $\forall b_{i_k} \in B$ выполнить действия:

2.6.1. Если $b_{j_0} = D\left(b_{i_k}\right) \neq \emptyset$, то для каждого сегмента стрелки диаграммы b_{i_0} $\forall s_{i_l} \in S_i$ выполнить действия:

2.6.1.1. Если $F_i^{ss}(s_{i_l}) \in B_i$ и $F_i^{ss}(s_{i_l}) = b_{i_k}$, то выполнить следующие действия:

2.6.1.1.1. Сделать сокетом позицию Pi_l, соответствующую сегменту Si_l .

2.6.1.1.2. Для каждого сегмента стрелки Sj_t диаграммы bj_0. $\forall Sj_t \in Sj$: пусть сегменту Sj_t соответствует метка m_q, т.е. $(Sj_t, m_q) \in C^{Sm}$, тогда если $(Si_l, m_q) \in C^{Sm}$, т.е. сегменту Si_l соответствует метка m_q , то сделать портом позицию Pj_t, соответствующую сегменту Sj_t.

2.6.1.1.3. Связать сокет Pi_l с портом Pj_t.

3. Конец алгоритма

Рассмотренный алгоритм позволяет преобразовать в CPN любую IDEF0-диаграмму стандартного вида, не содержащую туннелей и стрелок вызовов. Полученная модель будет являться базовой имитационной моделью бизнес-процессов.

СПИСОК ЛИТЕРАТУРЫ

1. Р 50.1.028 — 2001. Рекомендации по стандартизации. Методология функционального моделирования.

2. Kurt Jensen, Lars M. Kristensen. Coloured Petri Nets. Modelling and Validation of Concurrent Systems — Springer, 2009

3. Дж. Питерсон. Теория сетей Петри и моделирование систем. Пер. с англ. М.В. Горбатовой, под ред. В.А. Горбатова. — Москва. МИР, 1984

4. Устимов К.О., Федоров Н.В. Применение раскрашенных сетей Петри для реинжиниринга бизнес-процессов. // Вестник академии промышленности и менеджмента: Выпуск 7. — М.: МГИУ, 2008 — с. 89-96

5. Устимов К.О., Федоров Н.В. Компьютерное моделирование бизнес-процессов на основе IDEF0-моделей. Алгоритмический подход. // Вестник академии промышленности и менеджмента: Выпуск 8. — М.: МГИУ, 2008 — с. 111-116

6. В.А. Горбатов, А.В. Крылов, Н.В. Федоров САПР систем логического управления. М.: Энергоатомиздат, 1988 -232 с.

7. Н.В. Федоров, О.Н. Игнатов Метод имитационного моделирования инвестиционных проектов на основе динамических продуцирующих сетей ГИАБ номер 7 2010

8. Устимов К.О. Автоматизация реинжиниринга бизнес-процессов на основе IDEF0-моделей // Системы управления знаниями. Реинжиниринг бизнес-процессов. Тезисы докладов — М.: МЭСИ, 2010

9. Устимов К.О. Автоматизация реинжиниринга бизнес-процессов на основе современных CASE-технологий // Горный информационно-аналитический бюллетень. Отдельный выпуск № 2. 2010. Труды студентов и молодых ученых-1 — М.: Горная книга, 2010

10. Устимов К.О., Федоров Н.В. Автоматизация построения имитационной модели бизнес-процессов на основе методологии IDEF0 и раскрашенных сетей Петри. // Горный информационно-аналитический бюллетень (научно-технический журнал) № 12 — М.: Горная книга, 2013. ГИАБ

КОРОТКО ОБ АВТОРАХ

Устимов Кирилл Олегович – аспирант, e-mail: avader@yandex.ru

Федоров Николай Владимирович – кандидат технических наук, доцент, зам. зав. кафедрой, e-mail: fnv1@mail.msiu.ru

кафедра информационной безопасности, Московский государственный индустриальный университет (ФГБОУ ВПО «МГИУ»).

UDC 519.876.5

ALGORITHM OF CONVERSION OF IDEF0-MODEL TO COLOURED PETRI NET

Ustimov K.O., Graduate student, e-mail: avader@yandex.ru
Fedorov N.V., Candidate of Engineering Sciences, Assistant Professor, e-mail: fnv1@mail.msiu.ru
Moscow State Industrial University.

The algorithm of conversion of IDEF0-model to Coloured Petri Net is described. IDEF0-model can be used for functional modeling of business-processes. Coloured Petri Net allows to conduct a simulation modeling. Mathematical descriptions of IDEF0-model and Coloured Petri Net are provided. The considering algorithm is based on these descriptions.

The CPN's colour set is per se a modeling language data type, however the syntax of CPN ML doesn't allow to operate with data types directly, it is necessary to define colour sets.

Multisets are applied as position markings, arc expressions and other types of inscriptions. Mutlisets are enlarge the concept of Petri net's tokens.

The described algorithm allows to convert IDEF0-diagram of common kind that doesn't contain tunnels and call arrows, to CPN. The derived model is the basic simulation model of business-processes.

A CPN color set is in a sense the modeling language data type but the CPN ML syntax does not allow operating directly the data types, it is required to define color sets.

Maltisets are used in position identification, expressions of arcs and other elements. Multisets expand definition of tokens in the Petri nets.

The article considers the algorithm of CPN transformation of any standard type IDEF0-diagram without tunnels and arrows. The resultant model will be the reference simulation model of business-processes.

Key words: *IDEF0 methodology, Coloured Petri Nets, algorithm, mathematical model.*

REFERENCES

1. Standardization Guidelines. Functional Modeling Procedure. R 50.1.028, 2001.

2. Kurt Jensen, Lars M. Kristensen. Coloured Petri Nets. Modelling and Validation of Concurrent Systems — Springer, 2009

3. *Peterson J.*, 1981. Petri Net Theory and the Modeling of Systems, Prentice Hall.

4. Ustimov K., Fedorov, N.V. The use of colored Petri nets for re-engineering of busi-ness processes.//Bulletin of the Academy of Industry and Management: Issue 7. - Moscow: MGIU, 2008 - p. 89-96.

5. Ustimov K., Fedorov, N.V. Computer modeling of business processes based on the IDEF0- models. An algorithmic approach.//Bulletin of the Academy of Industry and Management: Issue 8. - Moscow: MGIU, 2008 - p. 111-116.

6. Gorbatov V.A., Krylov, A.V., Fedorov, N.V. CAD logic control systems. M. Energoatomizdat, 1988 - 232 p.

7. Fedorov N., Ignatov O. Simulation method of investment projects based on dynamic networks of producing GIABA room in July 2010.

8. Ustimov K.O. Automation of business process reengineering based on IDEF0- models /systems/knowledge management. Re-engineering of business processes. Abstracts - M: Mesi, 2010.

9. Ustimov KO Automation of business process reengineering based on modern CASE- technology//Mining information - analytical bulletin. A separate issue number 2. 2010. Proceedings of students and young scientists -1 - M: Mining book, 2010.

10. Ustimov K., Fedorov, N.V. Automation of a simulation model of business processes based on the IDEF0 methodology and colored Petri nets.// Mining information - analytical bulletin. № 12/ – 2013.