УДК 004.89

DOI: 10.15827/0236-235X.120.658-662

Дата подачи статьи: 10.05.17

2017. Т. 30. № 4. С. 658–662

ИСПОЛЬЗОВАНИЕ КОНЦЕПТ-КАРТ ДЛЯ АВТОМАТИЗИРОВАННОГО СОЗДАНИЯ ПРОДУКЦИОННЫХ БАЗ ЗНАНИЙ

Н.О. Дородных, младший научный сотрудник, tualatin32@mail.ru;

А.Ю. Юрин, к.т.н., доцент, зав. лабораторией, iskander@icc.ru

(Институт динамики систем и теории управления им. В.М. Матросова СО РАН,

ул. Лермонтова, 134, г. Иркутск, 664033, Россия;

Иркутский национальный исследовательский технический университет,

ул. Лермонтова, 83, г. Иркутск, 664074, Россия)

Актуальность использования концептуальных моделей в форме концепт-карт для автоматизированного формирования баз знаний интеллектуальных систем обуславливает необходимость разработки специализированного алгоритмического и программного обеспечения.

В данной работе рассмотрен подход, обеспечивающий прототипирование баз знаний экспертных систем продукционного типа на основе анализа концепт-карт IHMC SmartTools. Подход основан на выделении исходных структурных элементов концепт-карт из файлов формата Concept Mapping Extensible Language (CXL) и на их преобразовании в конструкции целевого языка программирования баз знаний, в частности, C Language Production System (CLIPS). Приведены описание основных этапов подхода, анализируемых конструкций CXL (в частности, concept-list, linking-phrase-list, connection-list), а также иллюстративный пример преобразований.

Особенностью предлагаемого подхода является использование онтологической модели в качестве универсальной промежуточной формы представления знаний, полученных из концепт-карт, не зависящей от языка программирования баз знаний, и авторской графической нотации – Rule Visual Modeling Language (RVML), обеспечивающей наглядное отображение и уточнение причинно-следственных отношений в виде логических правил (продукций).

Рассмотренное алгоритмическое обеспечение реализовано в составе исследовательского прототипа программного средства Personal Knowledge Based Designer (PKBD), которое в настоящий момент используется в учебном процессе в Иркутском национальном исследовательском техническом университете.

Ключевые слова: получение знаний, база знаний, концепт-карта, онтология, правила, преобразования, генерация кода, CLIPS, RVML, IHMC SmartTools.

Разработка новых подходов к автоматизации создания систем, основанных на знаниях (knowledge-based systems), и их отдельных компонентов, в частности БЗ, остается перспективной областью научных исследований. Сложность и трудоемкость процесса разработки данных систем связана с этапом разработки БЗ, который традиционно считается узким местом и включает задачи по извлечению (получению), структурированию и формализации предметных знаний путем их описания на определенном языке представления знаний (ЯПЗ) [1, 2]. Автоматизация этих задач является актуальной. Так, в последнее время широкое распространение получили принципы визуального (когнитивного) моделирования БЗ, а на стадиях получения и структурирования знаний активно используются различные концептуальные (информационные) модели, в том числе концепт-карты (concept maps) или интеллект-карты/карты знаний (mind maps) и др. В частности, эта тенденция подтверждается рядом примеров, например в [3–5].

Концептуальные модели являются результатом (основными артефактами) процессов проектирования ПО, анализа и моделирования предметной области. При создании данных моделей используют различные нотации, языки (стандарты), в частности DFD, IDEF0, IDEF5, UML и др. (рис. 1), преобразование которых в программные коды БЗ сильно ограничено.

В данной работе предлагается алгоритмическое и программное обеспечение (подход) для автома-

тизированного создания продукционных БЗ на основе анализа концепт-карт (карт знаний, интеллект-карт) как результатов когнитивного моделирования. Подход основан на выделении структурных элементов концепт-карт и их преобразовании в конструкции целевого языка программирования БЗ. Ранее этот подход был успешно применен для автоматизации создания БЗ на основе UML-моделей (диаграмм классов) [6]. Особенностью предлагаемого подхода является использование онтологии в качестве универсальной промежуточной формы представления знаний, полученных из концептуальных моделей, и авторской графической нотации – Rule Visual Modeling Language

Рис. 1. Наиболее популярные стандарты и нотации создания концептуальных моделей

Fig. 1. The most popular standards and notations for conceptual modeling

(RVML) [7] для уточнения полученных причинно-следственных отношений. Использование онтологии, в свою очередь, позволяет абстрагироваться от особенностей различных форматов представления концептуальных моделей при их последующем преобразовании в программные коды, тогда как RVML обеспечивает наглядность отображения и изменения полученных логических правил (продукций).

Анализ ряда систем концептуального (когнитивного) моделирования (например, IHMC SmapTools, FreeMind, Mindjet, Coggle, Mindjet MindManager, ProTheBrain, Aibase, XMind и др.), обусловивший постановку решаемой задачи, показал отсутствие возможности синтеза программных кодов БЗ и единого общепринятого формата (стандарта) представления концепт-карт (при этом большинство из рассмотренных систем используют XML-подобные форматы, не совместимые между собой).

В данной работе в качестве источника концепт-карт выбрано наиболее распространенное на сегодняшний день программное средство (редактор) – IHMC SmapTools [8], использующее для хранения моделей собственный XML-подобный формат – Concept Mapping Extensible Language (CXL).

Таким образом, постановку задачи можно формализовать следующим образом: необходимо определить оператор T преобразования концептуальной модели:

$$T: CM \rightarrow Code, \quad (1)$$

где CM – исходная анализируемая концептуальная модель, при этом $CM = \langle CM^{CXL} \rangle$, CM^{CXL} – концепт-карта в формате CXL; $Code$ – целевой синтезируемый код БЗ на определенном языке программирования БЗ (ЯПЗ).

Одним из инструментальных средств реализации предлагаемого подхода является Personal Knowledge Based Designer (PKBD) [9], представляющий собой специализированный редактор для прототипирования продукционных экспертных систем (ЭС) и БЗ, ориентированных на C Language Integrated Production Language (CLIPS) [10]. По этой причине $Code = \langle Code^{CLIPS} \rangle$, $Code^{CLIPS}$ – код БЗ в формате CLIPS.

Алгоритмическое обеспечение

Основной задачей предлагаемого алгоритмического обеспечения является преобразование концептуальных моделей (концепт-карт) в конструкции (элементы) языка программирования БЗ (в данном случае CLIPS), что может быть представлено в виде последовательности действий (рис. 2).

На этапах 1 и 2 такого преобразования средствами внешних программ (в данном случае IHMC SmapTools) пользователь строит концептуальную модель предметной области, которая представляется в формате XML. Этот формат является уни-

Fig. 2. An algorithm for knowledge base code generation based on conceptual models

версальным и наиболее распространенным способом интеграции программных систем и обеспечения обмена информацией между приложениями.

На этапе 3 сформированный XML-документ (концепт-карта в формате CXL) импортируется в систему PKBD, где в результате процесса анализа XML-структуры концептуальной модели выделяются понятия предметной области, их свойства и отношения. В частности, основными структурными единицами анализируемого формата CXL являются теги `concept-list`, `linking-phrase-list` и `connection-list`, содержимое которых автоматически извлекается.

На основе извлеченных понятий и их отношений формируется онтологическая модель (этап 4) как универсальное представление знаний, не зависящее от используемого языка программирования БЗ.

На этапе 5 при помощи специальной графической нотации RVML полученные из онтологии причинно-следственные связи визуализируются в виде продукций с последующей возможностью их модификации и проверки (тестирования).

На этапе 6 происходит генерация кода БЗ в формате CLIPS на основе онтологической модели, уточненной RVML-диаграммой.

Таким образом, уточним оператор преобразования концептуальной модели из (1):

$$\begin{aligned}
 T &= \langle T_{CM-ONT}, T_{ONT-Code} \rangle, \\
 T_{CM-ONT}: CM^{CXL} &\rightarrow M^{ONT}, \\
 T_{ONT-Code}: M^{ONT} &\rightarrow Code^{CLIPS},
 \end{aligned} \quad (2)$$

где T_{CM-ONT} – оператор преобразования XML-структур концепт-карты CXL в онтологическую модель; $T_{ONT-Code}$ – оператор преобразования онтологической модели в код БЗ на языке программирования БЗ CLIPS; M^{ONT} – онтологическая модель.

Рассмотрим подробнее преобразование концепт-карт SmartTools (формат CXL) в онтологию из (2): $T_{CM-ONT}: CM^{CXL} \rightarrow M^{ONT}$.

Процесс преобразования представляет собой анализ XML-структур концепт-карты с целью выделения конструкций, описывающих элементы модели. Подробно CXL описан в [8]. Для решения поставленной задачи наибольший интерес представляют разделы:

- **concept-list** – список понятий (концептов) в форме `<concept id="идентификатор" label="наименование"/>`;
- **linking-phrase-list** – наименования связей между концептами в форме `<linking-phrase id="идентификатор" label="наименование"/>`;
- **connection-list** – связи между концептами в форме `<connection id="идентификатор" from-id="идентификатор1" to-id="идентификатор2"/>`, где *идентификатор1* и *идентификатор2* – идентификаторы из списков концептов и наименований связей;
- **concept-appearance-list** – описание особенностей отображения концептов (положение, размер шрифта и др.);
- **linking-phrase-appearance-list** – описание особенностей отображения наименования связей (положение, цвет и др.);
- **connection-appearance-list** – описание особенностей отображения связей (наличие стрелок и др.).

Отметим, что, как правило, в концепт-картах, в том числе и в формате CXL, нет четкого разделения концептов на понятия (классы) и их свойства (атрибуты), поэтому обеспечить автоматическое определение подобных сущностей затруднительно. Одним из решений данной проблемы является введение дополнительных ограничений на именование сущностей в концепт-картах, а именно: при именовании понятий-классов использовать заглавную букву; в свою очередь, имена понятий-свойств начинать со строчных букв.

Используя данное ограничение, формируются множества классов CN_{ONT} , свойств классов PN_{ONT} , объектов Obj_{ONT} , типов данных DT_{ONT} и отношений R_{ONT} онтологии. В частности, происходит преобразование связей в причинно-следственные отношения: $R_A^{CXL} \rightarrow R_{ONT}^C$, при этом используется прямое отображение понятий из левых и правых частей связей, описанных в концепт-картах, в классы онтологии: $r_{A-LHS}^{CXL} \rightarrow cn_{LHS}^{ONT}$, $r_{A-RHS}^{CXL} \rightarrow cn_{RHS}^{ONT}$.

Однако, так как концепт-карта не содержит элементов, которые можно было бы однозначно интерпретировать в качестве значений операторов,

при создании правил с несколькими условиями по умолчанию назначается оператор **and**. На основе сформированной онтологической модели может быть синтезирован код БЗ в формате CLIPS.

Пример преобразования

Рассмотрим пример автоматизированного создания продукционной БЗ на основе преобразования (трансформации) фрагмента концепт-карты (рис. 3) с использованием ранее описанного алгоритма. На первом этапе с помощью программного средства ИМС SmartTools пользователь строит концепт-карту (рис. 3) с учетом рекомендаций по именованию понятий.

Рис. 3. Пример концепт-карты ИМС SmartTools

Fig. 3. An example of an IHC SmartTools concept map

На этапе 2 построенная концепт-карта сохраняется в CXL-файл (фрагмент):

```
...
<concept-list>
  <concept id="1R8PCPYC6-26LL4JG-CJ"
    label="#x442; &#x438; &#x43f;: String"/>
  <concept id="1R8PCPYBQ-DBJJK-CC"
    label="#x41e; &#x441; &#x430; &#x434; &#x43a; &#x438
    ;"/>
  <concept id="1R8PCPYC6-V9L3C3-CL"
    label="#x422; &#x438; &#x43f;: String"/>
  <concept id="1R8PCPYC6-10YHC6L-CH"
    label="#x422; &#x438; &#x43f;: String"/>
...
</concept-list>
<linking-phrase-list>
  <linking-phrase id="1R8PCPYFR-3NF9Z7-HC"
    label="assoc"/>
  <linking-phrase id="1R8PCPYCQ-ZG7CBC-D2"
    label="attr"/>
  <linking-phrase id="1R8PCPYF7-1WH2JTJ-H0"
    label="assoc"/>
...
<connection-list>
  <connection id="1R8PCPYCQ-42P45H-D6" from-
```

```
id="1R8PCPYCQ-ZG7CBC-D2" to-id="1R8PCPYBQ-1YG2W64-C8"/>
<connection id="1R8PCPYDQ-TT1FRX-FZ" from-id="1R8PCPYDQ-129G835-FV" to-id="1R8PCPYC6-1DR0Z73-CD"/>
<connection id="1R8PCPYD7-2D7389G-F2" from-id="1R8PCPYD7-26KHJ1Q-DY" to-id="1R8PCPYBQ-DBBJHK-CC"/>
```

Анализ CXL-файла позволяет сформировать множества понятий и отношений (этапы 3 и 4), которые могут быть представлены в форме онтологической модели для их последующего отображения и модификации (этап 5) в форме RVML (рис. 4).

Рис. 4. Пример RVML-диаграммы – шаблон правила

Fig. 4. An example of RVML diagram. A rule template

В свою очередь, онтологическая модель, включающая описание шаблонов фактов и правил в нотации RVML, при дополнении конкретными данными (рис. 5) может быть отображена (трансформирована) в программный код на CLIPS (этап 6) и протестирована (рис. 6):

```
...
(deftemplate Turisticheskaya-aktivnost
(slot kolichestvo-turistov)
)
(deftemplate Lesopozharnaya-obstanovka
(slot tip)
)
(deftemplate Nasazhdeniya
(slot plotnost)
(slot tip (default "ХВОЙНЫЕ"))
)
...
(defrule Pravilo-1-2 "Описание правила: Правило 1 2"
(Zaselennost ;Заселённость
(plotnost-naseleniya "УМЕРЕННАЯ")
)
(Turisticheskaya-aktivnost ;Туристическая активность
(kolichestvo-turistov "УМЕРЕННОЕ")
)
(Pogodnye-usloviya ;Погодные условия
(naimenovanie "БЛАГОПРИЯТНЫЕ ДЛЯ ПОЖАРА")
)
(Nasazhdeniya ;Насаждения
(plotnost "ВЫСОКАЯ")
(tip "ХВОЙНЫЕ")
)
=>
(assert
(Lesopozharnaya-obstanovka ;Лесопожарная обстановка
(tip "ПОЖАРООПАСНАЯ")
)
)
)
```

Рис. 5. Описание правила

Fig. 5. A description of rules

Рис. 6. Тестирование полученной БЗ – просмотр активированных правил

Fig. 6. Testing the obtained knowledge base. Preview of activate rules

Заключение

Обеспечение возможности синтеза программных кодов на основе концептуальных моделей способствует повышению эффективности процесса разработки БЗ интеллектуальных систем.

В данной работе предлагается автоматизировать анализ концептуальных моделей и формировать программный код БЗ на основе графических примитивов концепт-карт IHMC SmartTools. Такой подход позволяет избежать ошибок программирования, которые, как правило, связаны с большими издержками на последующих этапах жизненного цикла программного продукта.

При решении задач исследования был проведен анализ методов, подходов и систем, а также различных стандартов (нотаций и языков) концептуального (когнитивного) моделирования. Описан алгоритм преобразования концептуальных моделей (концепт-карт IHMC SmartTools) в код продукционной БЗ в формате CLIPS.

Рассмотренное алгоритмическое обеспечение реализовано в составе исследовательского прототипа программного средства PKBD [9].

В настоящий момент PKBD используется в учебном процессе в Иркутском национальном исследовательском техническом университете (ИрНТУ) при выполнении лабораторных работ по курсам «CASE-средства» и «Инструментальные средства информационных систем».

Работа выполнена при частичной поддержке РФФИ, проекты №№ 15-07-05641, 16-37-00041, 16-37-00122.

Литература

1. Гаврилова Т.А., Кудрявцев Д.В., Муромцев Д.И. Инженерия знаний. Модели и методы. СПб: Лань, 2016. 324 с.
2. Джексон П. Введение в экспертные системы; [пер. с англ.]. М: Вильямс, 2001. 624 с.
3. Zulkafli Z., Perez K., Vitolo C., Buytaert W., Karpouzoglou T., Dewulf A., Bievre B.D., Clark J., Hannah D.M., Shaheed S. User-driven design of decision support systems for polycentric envi-

ronmental resources management. *Environmental Modelling & Software*, 2017, vol. 88, pp. 58–73.

4. Gavrilova T.A., Leshcheva I.A. Ontology design and individual cognitive peculiarities: A pilot study. *Expert Systems with Applications*, 2015, vol. 42, pp. 3883–3892.

5. Starr R.R., Parente de Oliveira J.M. Concept maps as the first step in an ontology construction method. *Information Systems*, 2013, vol. 38, pp. 771–783.

6. Дородных Н.О., Юрин А.Ю. Использование диаграмм классов UML для формирования продукционных баз знаний // Программная инженерия. 2015. № 4. С. 3–9.

7. Юрин А.Ю. Нотация для проектирования баз знаний продукционных экспертных систем // Объектные системы. 2016. № 12. С. 48–54.

8. IHMC CmapTools. URL: <http://cmap.ihmc.us/> (дата обращения: 14.06.2017).

9. Юрин А.Ю., Грищенко М.А. Редактор баз знаний в формате CLIPS // Программные продукты и системы. 2012. № 4. С. 83–87.

10. Частиков А.П., Гаврилова Т.А., Белов Д.Л. Разработка экспертных систем. Среда CLIPS. СПб: БХВ-Петербург, 2003. 608 с.

Software & Systems

DOI: 10.15827/0236-235X.120.658-662

Received 10.05.17

2017, vol. 30, no. 4, pp. 658–662

USING CONCEPT MAPS FOR RULE-BASED KNOWLEDGE BASES ENGINEERING

N.O. Dorodnykh¹, Junior Researcher, tualatin32@mail.ru

A.Yu. Yurin^{1,2}, Ph.D. (Engineering), Associate Professor, Head of Laboratory, iskander@icc.ru

¹ *Institute of System Dynamics and Control Theory SB RAS, Lermontov St. 134, Irkutsk, 664033, Russian Federation*

² *Matrosov National Research Irkutsk State Technical University, Lermontov St. 83, Irkutsk, 664074, Russian Federation*

Abstract. Using conceptual models in the form of concept maps for engineering rule-based knowledge bases of intelligent systems remains relevant. This relevance demands the development of specialized algorithmic and software.

This paper considers an approach to prototyping of rule-based knowledge bases of expert systems based on analysis of IHMC CmapTools concept maps. The approach is based on the extracting structural elements of concept maps from the CXL files (Concept Mapping Extensible Language) and their transformation to the elements of a programming language, in particular, the C Language Production System (CLIPS). The paper describes the main stages of the approach, analyzed constructions of CXL files (in particular, concept-list, linking-phrase-list, connection-list). It also presents an illustrative example of transformations.

A distinctive feature of the proposed approach is using an ontological model as a universal intermediate form of knowledge representation derived from concept maps, which is independent to the knowledge base programming language. Another feature is the author's graphic notation – Rule Visual Modeling Language (RVML) that provides visualization and modification of cause-effect relations as logical rules.

The considered algorithms are implemented as a part of a software research prototype called the Personal Knowledge Based Designer (PKBD). Currently, it is used in the educational process at the Irkutsk National Research Technical University (INRTU) in “CASE-tools” and “Software tools of information systems” courses.

Keywords: knowledge acquisition, knowledge base, concept map, ontology, rules, transformations, code generation, CLIPS, RVML, IHMC CmapTools.

Acknowledgements. *The work has been partially supported by RFBR, research projects No. 15-07-05641, 16-37-00041, 16-37-00122.*

References

1. Gavrilova T.A., Kudryavtsev D.V., Muromtsev D.I. *Inzheneriya znany. Modeli i metody* [Knowledge Engineering. Models and Methods]. St. Petersburg, Lan Publ., 2016, 324 p.
2. Jackson P. *Introduction to Expert Systems*. 3rd ed. Addison-Wesley Longman Publ., USA, 1998, 542 p. (Russ. ed.: Moscow, Vilyams Publ., 2001, 624 p.).
3. Zulkafli Z., Perez K., Vitolo C., Buytaert W., Karpouzoglou T., Dewulf A., Bievre B.D., Clark J., Hannah D.M., Shaheed S. User-driven design of decision support systems for polycentric environmental resources management. *Environmental Modelling & Software*. 2017, vol. 88, pp. 58–73.
4. Gavrilova T.A., Leshcheva I.A. Ontology design and individual cognitive peculiarities: A pilot study. *Expert Systems with Applications*. 2015, vol. 42, pp. 3883–3892.
5. Starr R.R., Parente de Oliveira J.M. Concept maps as the first step in an ontology construction method. *Information Systems*. 2013, vol. 38, pp. 771–783.
6. Dorodnykh N.O., Yurin A.Yu. Using UML class diagrams to design knowledge bases of rule-base expert systems. *Programnaya inzheneriya* [Software Engineering]. 2015, no. 4, pp. 3–9 (in Russ.).
7. Yurin A.Yu. Notation for design of knowledge bases of rule-based expert systems. *Obyektnye sistemy* [Object Systems]. 2016, no. 12, pp. 48–54 (in Russ.).
8. IHMC CmapTools. Available at: <http://cmap.ihmc.us/> (accessed June 14, 2017).
9. Yurin A.Yu., Grishchenko M.A. Knowledge Base Editor for CLIPS. *Programmnye produkty i sistemy* [Software & Systems]. 2012, no. 4, pp. 83–87 (in Russ.).
10. Chastikov A.P., Gavrilova T.A., Belov D.L. *Razrabotka ekspertnykh sistem. Sreda CLIPS* [Development of Expert Systems. CLIPS Environment]. St. Petersburg, BHV-Peterburg Publ., 2003, 608 p.