УДК 004.89

РАЗРАБОТКА И ИСПОЛЬЗОВАНИЕ МЕТАМОДЕЛЕЙ ДЛЯ СИНТЕЗА СПЕЦИФИКАЦИЙ И КОДОВ БАЗ ЗНАНИЙ

Дородных Никита Олегович

К.т.н., м.н.с., Институт динамики систем и теории управления имени В.М. Матросова СО РАН, 664033, г. Иркутск, ул. Лермонтова, 134, e-mail: tualatin32@mail.ru

Николайчук Ольга Анатольевна

Д.т.н., с.н.с., Институт динамики систем и теории управления имени В.М. Матросова СО РАН, 664033, г. Иркутск, ул. Лермонтова, 134, e-mail: nikoly@icc.ru

Юрин Александр Юрьевич

К.т.н., зав. лабораторией «Информационных технологий исследования природной и техногенной безопасности», Институт динамики систем и теории управления имени В.М. Матросова СО РАН, 664033, г. Иркутск, ул. Лермонтова, 134, e-mail: iskander@icc.ru

Коршунов Сергей Андреевич

Программист «ООО ЦентраСиб», 664003, г. Иркутск, ул. Карла Маркса, 11, e-mail: info@centrasib.ru

Аннотация. Работа посвящена проблематике повышения эффективности создания баз знаний и интеллектуальных систем. Эффективность данного процесса может быть повышена путем автоматической генерации спецификаций и кодов баз знаний на целевом языке представления знаний путем анализа и трансформации информации из различных источников, в частности, концептуальных моделей, описывающих некоторую предметную область и представленных в разных форматах (например, UML-модели, концепт-карты, древовидные семантические структуры, диаграммы Исикавы и т.д.). Синтез спецификаций и программных кодов методологически основан на применении концепции трансформации моделей – модельно-ориентированного подхода (Model-Driven Engineering) и использовании метамоделей для описания исходных и целевых (CLIPS и OWL) формализмов. В работе приведено описание метода автоматизированного формирования метамоделей анализируемых форматов концептуальных моделей и мета-метамодели для унифицированного представления и хранения метамоделей.

Ключевые слова: интеллектуальная система, база знаний, модельно-управляемый подход, метамоделирование, трансформация моделей, метамодель, концептуальная модель, генерация кода.

Цитирование: Дородных Н.О., Николайчук О.А., Юрин А.Ю., Коршунов С.А. Разработка и использование метамоделей для синтеза спецификаций и кодов баз знаний // Информационные и математические технологии в науке и управлении. 2019. № 2 (14). С. 26–39. DOI: 10.25729/2413-0133-2019-2-03

Введение. На сегодняшний день знания продолжают оставаться стратегическим ресурсом. Актуальность разработки новых методов и средств, повышающих эффективность процессов обработки знаний, в том числе при решении практических слабоформализованных

задач в различных предметных областях, остается высокой. При этом «оцифровка» и представление знаний в виде информационных (концептуальных) моделей, спецификаций и кодов баз знаний (БЗ) обеспечивают их эффективное использование.

Существует множество направлений повышения эффективности создания БЗ и интеллектуальных систем. Однако в последнее время наиболее востребованным является использование гибридных [11] и унифицированных подходов, которые обеспечивают охват всех этапов жизненного цикла систем, основанных на знаниях. В частности, существует ряд крупных методологий, направленных на формализованное описание повторяющихся схем решений задач на абстрактном уровне без привязки к какой-либо конкретной предметной области. Самой популярной методологией инженерии знаний является CommonKADS [22], которая де-факто считается стандартом проектирования и разработки систем, основанных на знаниях. Также можно отметить такие методологические проекты, как: MIKE [24], MOKA [23] и АТ-ТЕХНОЛОГИЯ [12]. Активно развиваются подходы к созданию интеллектуальных систем на основе семантических технологий, в частности онтологий [2, 10, 13]. При наличии в этой области существующих решений, необходимо отметить общую тенденцию к использованию концептуальных моделей при разработке БЗ и ориентацию на непрограммирующих пользователей [10, 14, 21]. В данном контексте перспективными являются подходы, основанные на порождающем программировании, в частности, на модельно-управляемом (-ориентируемом) подходе – Model Driven Engineering (MDE) или Model Driven Development (MDD) [16] и его разновидностях, например, Model Driven Architecture (MDA) [19] – концепции реализации MDE/MDD от консорциума Object Management Group (OMG). MDE/MDD/MDA – это стиль разработки программного обеспечения, когда абстрактные описания программной системы в виде информационных (концептуальных) моделей становятся основными артефактами при разработке, а сам процесс разработки программной системы представляет собой последовательное преобразование (трансформацию) данных моделей. В свою очередь, в основе трансформаций лежит метаописание моделей или их метамодели.

Целью данной работы является создание метода автоматизированного формирования метамоделей для решении задач трансформации и синтеза программных кодов и спецификаций проблемно-ориентированных интеллектуальных систем и их БЗ (предметная область: диагностика технического оборудования в нефтехимии) для CLIPS (C Language Integrated Production System) [15] и OWL (Web Ontology Language) [20] и программная реализация метода в виде веб-ориентированного программного модуля (редактора) визуальной разработки метамоделей в рамках системы поддержки разработки БЗ – Knowledge Base Development System (KBDS) [9].

1. Модельно-ориентированный подход и трансформации. Трансформация моделей является одной из основных составляющих модельно-ориентированного подхода к разработке программного обеспечения (MDE/MDD/MDA) [16, 19]. Под трансформацией модели понимается автоматическая генерация целевой модели из исходной модели в соответствии с некоторым набором правил трансформации.

Базовыми понятиями трансформации моделей являются [16]:

- модель – абстрактное описание на некотором формальном языке характеристик системы (процесса), важных с точки зрения цели моделирования, которое скрывает

информацию о некоторых аспектах с целью представления упрощенного описания остальных;

- метамодель – модель языка, используемого для создания моделей;
- мета-метамодель – язык, на котором описываются метамодели. Для построения метамоделей используются специальные языки метамоделирования. Наиболее распространенными языками метамоделирования являются: MOF (Meta-Object Facility), Ecore, KM3 (Kernel Meta Meta Model) и др.

Для описания базовых понятий и их отношений в MDE/MDD используется 4-х уровневая архитектура моделирования (иерархия моделей). На уровне «M1» (уровень моделей) находится некоторая исходная модель, которую необходимо преобразовать (перевести) в некоторую целевую модель. При этом уровень «M0» соответствует слою данных, т.е. он отражает объекты и процессы реального мира (данный уровень обычно не указывают на подобного рода схемах). Все модели уровня «M1» соответствуют некоторым метамоделям на уровне «M2» (уровень метамоделей), т.е. существуют некоторые формальные языки моделирования для их описания. Модель трансформации представляет собой набор правил преобразования, которые используют соответствующие элементы (конструкции) метамоделей исходной и целевой модели и описываются с использованием различных языков трансформации моделей (например, QVT, ATL, Epsilon, VIATRA2, GReAT, XSLT и др.), т.е. модель трансформации также соответствует некоторой метамодели языка трансформации. Таким образом, трансформация моделей осуществляется на уровне метамоделей («M2»). В свою очередь, все метамодели на уровне «M2» соответствуют некоторой одной мета-метамодели (мета-объектное средство) на уровне «M3» (уровень мета-метамодели), т.е. они описываются с использованием некоторого языка метамоделирования. В подходе MDA в качестве языка метамоделирования используется стандарт MOF, который призван служить мостом между разными метамоделями, поскольку представляет собой основу для их описания.

Таким образом, процесс трансформации моделей базируется на использовании метамоделей и метамоделирования, как одним из основных подходов к определению абстрактного синтаксиса (abstract syntax) языков.

Далее подробнее опишем процесс трансформации моделей в рамках инженерии БЗ, с точки зрения описания возможных метамоделей, участвующих в данном процессе, а также метод создания метамоделей.

2. Метамодели. Применение принципов MDE/MDD/MDA к созданию БЗ позволяет представить этот процесс в виде цепочки преобразований некоторой исходной информационной (концептуальной) модели предметной области, представленной в формате XML, в целевую модель БЗ (модель продукций или онтологии), на основе которой может быть автоматически синтезирован код на языке представления знаний (ЯПЗ) CLIPS или OWL. Данное преобразование описывается моделью трансформации M_T:

$$M_T = \langle MM_{IN}, MM_{OUT}, T \rangle, \quad (1)$$

где MM_{IN} – метамодель исходной (входной) концептуальной модели; MM_{OUT} – метамодель целевой (выходной) модели представления знаний (модели БЗ); T – оператор преобразования моделей. Более детальное описание данного оператора приводится в [1].

Используя (1), подробнее рассмотрим элементы исходной и целевой метамодели в модели трансформации:

$$MM_{IN} \in \{MM_{CM}, MM_{PR}, MM_{ONT}\},$$
$$MM_{OUT} \in \{MM_{PR}MM_{ONT}, MM_{CLIPS}, MM_{OWL}\}, \quad (2)$$

где MM_{CM} – метамодель концептуальной модели, представленной в формате XML; MM_{PR} – метамодель модели продукций; MM_{ONT} – метамодель онтологии; MM_{CLIPS} – метамодель языка CLIPS; MM_{OWL} – метамодель языка OWL.

Исходя из (2), следует, что помимо метамоделей MM_{CM}, MM_{CLIPS} и MM_{OWL}, которые необходимы для описания преобразования исходных концептуальных моделей в целевые БЗ в формате CLIPS или OWL, в модель трансформации также могут входить метамодели MM_{PR} и MM_{ONT}, описывающие некоторые обобщенные модели продукций и онтологии. Данные модели позволяют абстрагироваться от особенностей описания знаний на различных конкретных языках, которые используются при реализации (программировании) БЗ. Таким образом, они представляют собой высокоуровневые абстракции (спецификации), предназначенные для унифицированного представления и хранения знаний, извлеченных из концептуальных моделей, и, в свою очередь, могут выступать как в качестве исходной концептуальной модели для синтеза кода БЗ, так и целевой модели БЗ.

При этом следует отметить, что, в рамках предлагаемого подхода, метамодели MM_{PR}, MM_{ONT}, MM_{CLIPS}, MM_{OWL} предоставляются пользователю по умолчанию для создания модели трансформации.

Исходя из (2), опишем элементы MM_{CM}:

$$MM_{CM} = \langle E_{CM}, R_{CM} \rangle, \quad (3)$$

где E_{CM} – множество элементов метамодели исходной концептуальной модели; R_{CM} – множество отношений между элементами метамодели исходной концептуальной модели.

При этом:

$E_{CM} = \{e_1^{cm}, \ldots, e_n^{cm}\}, e_i^{cm} = \langle name_{i,1}, p_{i,2}, \ldots, p_{i,k} \rangle, i \in \overline{1,n}$, где $name_{i,1}$ – наименование *i*-элемента; $p_{i,2}, \ldots, p_{i,k}$ – остальные свойства *i*-элемента.

$R_{CM} = \langle R_{AS}^{cm}, R_{ID}^{cm}, R_{part-of}^{cm} \rangle$, где R_{AS}^{cm} – ассоциация – бинарное отношение между двумя элементами метамодели; R_{ID}^{cm} – связь по идентификатору – бинарное отношение между двумя элементами метамодели по одному идентификатору (числовому или текстовому значению); $R_{part-of}^{cm}$ – связь «часть-целое» (композиция) – *n*-арное отношение между n элементами метамодели.

При этом:

$R_{AS}^{cm} = \left\{ r_1^{as}, \ldots, r_m^{as} \right\}, r_j^{as} = \left\langle e_{j,1}^{cm}, e_{j,2}^{cm} \right\rangle, j \in \overline{1,m}$, где $e_{j,1}^{cm}$ – левая часть отношения типа ассоциация (левый элемент метамодели), $e_{j,2}^{cm}$ – правая часть отношения типа ассоциация (правый элемент метамодели).

$R_{ID}^{cm} = \left\{ r_1^{id}, \ldots, r_h^{id} \right\}, r_l^{id} = \left\langle r_{lhs_l}^{id}, r_{rhs_l}^{id} \right\rangle, l \in \overline{1,h}$, где $r_{lhs_l}^{id}$ – левая часть связи; $r_{rhs_l}^{id}$ – правая часть связи. В свою очередь $r_{lhs_l}^{id} = e_i^{cm}(p_i), i \in \overline{1,n}$ и $r_{rhs_l}^{id} = e_j^{cm}(p_j), j \in \overline{1,n}$, где $e_i^{cm}(p_i)$ и $e_j^{cm}(p_j)$ – элементы метамодели, участвующие в отношении по идентификатору, т.е. значение свойства p_j равно значению свойства p_i.

Отношение «часть-целое» предполагает, что *i*-элемент метамодели e_i^{cm} состоит из составных частей, которые, в свою очередь, связываются (соединяются) между собой отношением R_{AS}^{cm} или R_{ID}^{cm}.

Для представления и хранения данных метамоделей разработана мета-метамодель (модель уровня «M3») (рис. 1), определяющая концептуальное пространство моделирования (Conceptual Modeling Space) [17].

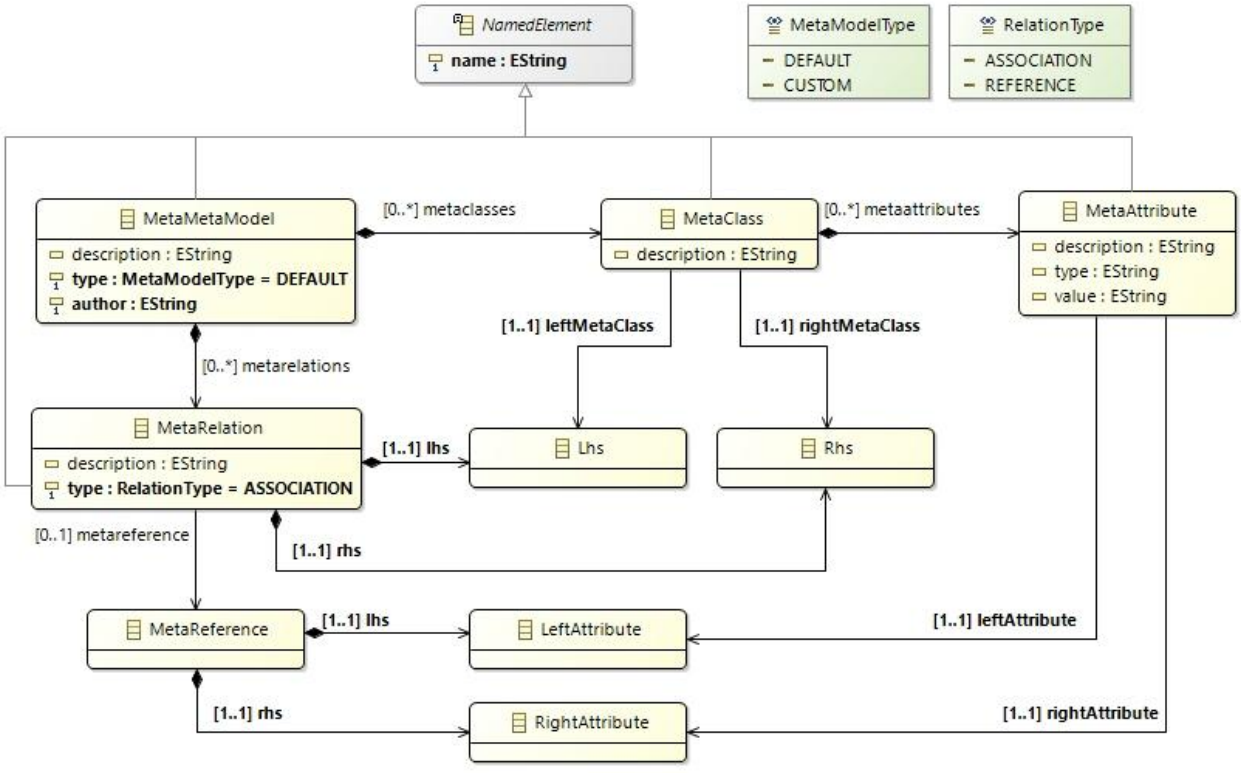

Рис. 1. Унифицированная мета-метамодель (на основе спецификации Ecore [18])

При этом MM_{PR}, определяющая основные концепты, из которых состоит модель продукций, представлена в [5]. MM_{ONT}, определяющая основные концепты, из которых состоит модель онтологии, подробнее описана в [3].

В качестве целевых ЯПЗ предлагается использовать CLIPS [15] и OWL2 [20], как наиболее распространенные на данный момент языки для описания продукций и онтологий.

3. Метод создания метамоделей. Определенные в (2) метамодели и унифицированная метамодель (Рис. 1) определяют особенности метода создания метамоделей для синтеза спецификаций и программных кодов БЗ. При этом метод представляет собой систематизированную совокупность действий (Рис. 2), которые нацелены на решение задачи автоматизированной разработки метамоделей.

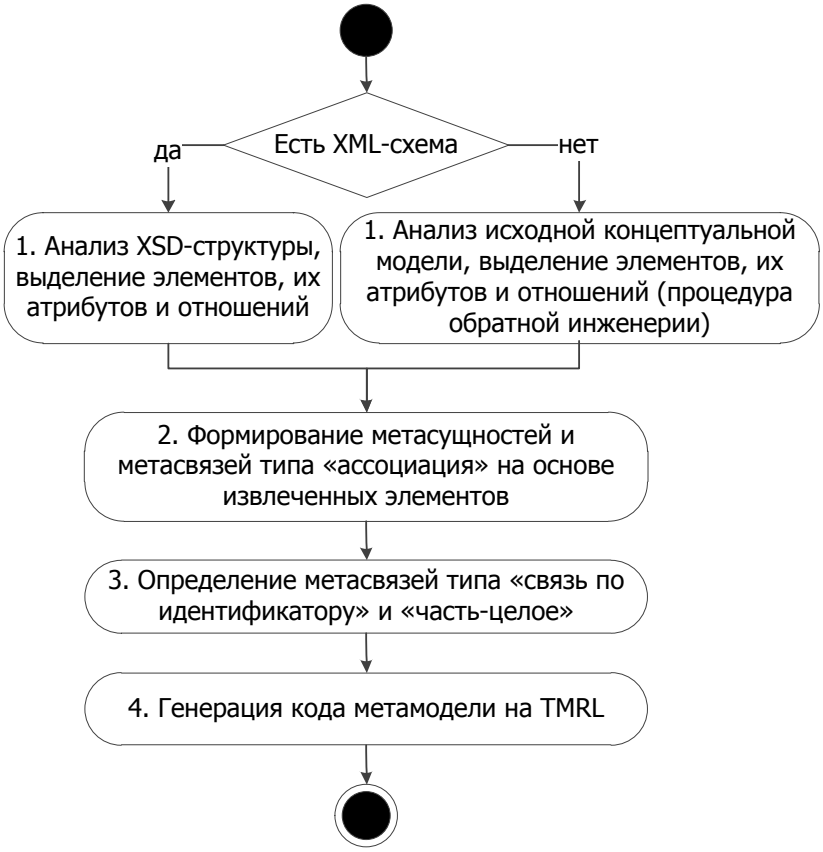

Рис. 2. Этапы создания метамоделей

Трансформация моделей осуществляется на достаточно абстрактном уровне (уровне метамоделей), поэтому определим основные особенности и ограничения предлагаемого метода:

- MM_{CM} может быть сформирована на основе XML-схемы концептуальной модели в формате XML Schema Definition (XSD).
- MM_{CM} может быть получена путем анализа исходных концептуальных моделей и извлечением из них элементов и отношений.
- Полученная MM_{CM} может быть семантически некорректной (например, если в исходной концептуальной модели наименования сущностей отражают семантику предметной области – уровня модели «M1», а не метамодели – уровня «M2»).
- Не все связи элементов в MM_{CM} могут быть получены с использованием XML-схем или процедуры обратной инженерии, а именно: связь «по идентификаторам» R_{ID}^{cm}, установленная путем внутренней индексации элементов; связь «часть-целое» $R_{part-of}^{cm}$, установленная путем переноса части семантики элемента в дочерние элементы.

- Не все элементы XML-схемы могут быть однозначно отображены в элементы MM_{CM} .
- Не все элементы MM_{CM} могут быть однозначно отображены в элементы целевой метамодели (проблема избыточности и дефицита выразительной способности).

Введение вышеперечисленных ограничений позволяет понизить сложность разработки моделей трансформаций при сохранении их выразительной способности.

4. Проверка корректности создаваемых метамоделей и правил трансформаций. В настоящее время разработка любого набора правил трансформации моделей включает их тестирование (верификацию) с целью выявления различного рода ошибок. При этом ошибки могут быть допущены как при создании моделей трансформаций и их метамоделей, так и при их интерпретации.

При создании модели трансформации могут произойти следующие основные ошибки:

- Создание синтаксически неправильных метамоделей: ошибки или опечатки при создании (кодировании) элементов метамодели.
- Создание семантически неправильных метамоделей: ошибки, связанные с неверным заданием структуры метамодели (например, неверные связи между элементами, отсутствие каких-либо связей у элементов и т.д.).
- Неполное покрытие метамоделей: правило трансформации построено без полного охвата элементов исходной и целевой метамодели, что приводит к проблеме того, что некоторые исходные концептуальные модели не могут быть преобразованы (например, правило трансформации работает только для определенных типов элементов и т.д.).
- Создание семантически неверных отображений между элементами исходной и целевой метамодели.
- Ошибки или опечатки из-за неправильного кодирования (уточнения) правил трансформации.

При выполнении (интерпретации) модели трансформации (M_T) могут произойти следующие основные ошибки:

- Генерация синтаксически неправильных моделей БЗ: правила трансформации или их часть реализованы неверно, что приводит к несоответствию целевой модели ее метамодели или нарушению ограничений модели.
- Генерация семантически неправильных моделей БЗ: правила трансформации семантически некорректны, что приводит получению неадекватной синтаксически корректной целевой модели.

Для устранения подобных ошибок используется два подхода: первый основан на предупреждении большинства рассмотренных ошибок, второй – на проверке корректности (тестирования, верификации) полученных БЗ и моделей, как результатов трансформаций.

Первый подход реализован путем автоматизированного контроля спецификаций создаваемых моделей в разработанных редакторах и определения ряда особенностей и ограничений, в частности:

1) При построении метамоделей и правил преобразования используется интерактивное визуальное программирование.

2) Код метамоделей и правила преобразования на языке модельных трансформаций генерируются автоматически.
3) При построении метамоделей автоматически проверяются следующие требования:
    - метамодель не должна содержать несвязанные элементы;
    - метамодель не должна содержать элементы без атрибутов (свойств);
    - пары элементов метамодели не могут связываться одной и той же связью несколько раз;
    - пары элементов не могу связываться кольцевой связью одного типа;
    - элемент метамодели не может связываться сам с собой (не должно быть рекурсивных связей).
4) При построении правил преобразования автоматически проверяются следующие требования:
    - модель трансформации должна содержать хотя бы одно правило преобразования;
    - правила преобразования могут содержать только соответствия между элементами исходной и целевой метамодели;
    - соответствия между атрибутами (свойствами) элементов исходной и целевой метамоделей устанавливаются только после связывания этих элементов;
    - соответствия между атрибутами элементов и самих элементов не допускаются;
    - модель трансформации допускает содержание не связанных соответствием элементов метамоделей, если они принадлежат к крайним случаям соответствий – избыточность и дефицит выразительной способности;
    - у каждого правила преобразования должен быть приоритет, задающий порядковый номер выполнения данного правила в интерпретаторе;
    - приоритеты правил преобразования задаются только натуральными числами;
    - правила преобразования не должны иметь одинаковые приоритеты;
    - приоритеты должны задавать правильную последовательность выполнения правил преобразования исходя из структурных особенностей исходной и целевой метамодели.

Второй подход основан на проверке корректности (тестирования, верификации) полученных БЗ, которая предполагает обнаружение логических ошибок в представлении знаний и структурах вывода. В ряде случаев, рассматривая верификацию интеллектуальных систем по аналогии с верификацией традиционных программ, ее трактуют как доказательство правильности БЗ. В настоящее время кроме нахождения обычных ошибок и опечаток в коде БЗ, можно отметить обнаружение различного рода аномалий, например, нарушения согласованности (consistency) БЗ (например, противоречивость, наличие циклов, избыточность, наличие пересечений и т.д.) или целостности (completeness) БЗ (например, неполнота, отсутствие ссылок, некорректность и т.д.). Данный подход реализован путем использования внешних программных средств, содержащий интерпретаторы и валидаторы полученных целевых моделей и кодов БЗ, в частности:

- Все разработанные в рамках данной работы метамодели (MM_{PR} , MM_{ONT} , MM_{CLIPS} , MM_{OWL}) создавались и проверялись с использованием специального графического редактора [8] и соответствуют мета-метамодели представленной на Рисунке 1.
- Проверка корректности сгенерированных продукционных БЗ в формате CLIPS производилась в системе программирования БЗ – Personal Knowledge Base Designer (PKBD) [4] с использованием подключаемых динамических библиотек машин вывода, в частности, реализующих алгоритм RETE. Проверка отдельных правил БЗ на наличие нарушений целостности и непротиворечивости может быть произведена, например, в системе CHECK или ДИФКЛАСС. Проверка корректности сгенерированных онтологических БЗ в формате OWL может быть произведена в системе онтологического моделирования Protégé с использованием машин вывода (reasoners) - Pellet, FaCT++ или HermiT.

Заключение. В статье в рамках модельно-ориентированного подхода описан метод создания и использования метамоделей при выполнении трансформации исходной модели предметной области в целевую базу знаний. Определены возможные метамодели, участвующие в данном процессе; предложена унифицированная мета-метамодель их описания; описаны основные этапы метода.

Метод основан на анализе структурных элементов XML-схем, а также процедуре обратной инженерии, с целью получения высокоуровневых спецификаций из низкоуровневых текстовых представлений. Такой метод позволяет, с одной стороны, сократить время, затрачиваемое на разработку метамоделей и избежать ошибок, связанных с программированием, за счет автоматического синтеза основных конструкций метамоделей, с другой – приблизить специалистов-предметников к непосредственной разработке интеллектуальных компонентов различных проблемно-ориентированных программных систем, позволяя им создавать спецификации и программный код, оперируя понятными предметно-ориентированными моделями.

Разработанный метод реализован в виде программного модуля (редактора) [8], включенного в состав веб-средства KBDS [9], и использовался для прототипирования продукционных БЗ в задаче прогнозирования развития деградационных процессов аппаратов в нефтехимии. В частности, при помощи разработанного метода и редактора [26] были разработаны метамодели, описывающие форматы (языки) представления исходных концептуальных моделей в виде:

- диаграмм классов UML. Следует отметить, что правильность полученной метамодели была проверена путем её сопоставления с уже существующим описанием спецификации языка UML, что также подтверждает корректность разработанного методологического и программного обеспечения [26];
- причинно-следственных диаграмм Исикавы, представленных в расширенной нотации для отображения деградационных процессов или процессов нарушения техногенной безопасности [25];
- концепт-карт в формате XML Topic Maps (XTM) средства CmapTools [7];
- деревьев событий [6].

Работа выполнена при финансовой поддержке РФФИ (проект № 19-07-00927).

СПИСОК ЛИТЕРАТУРЫ

1. Бычков И.В., Дородных Н.О., Юрин А.Ю. Подход к разработке программных компонентов для формирования баз знаний на основе концептуальных моделей // Вычислительные технологии. 2016. Т. 21. № 4. С. 16–36.
2. Голенков В.В., Гулякина Н.А. Принципы построения массовой семантической технологии компонентного проектирования интеллектуальных систем // Материалы международной научно-технической конференции «Открытые семантические технологии проектирования интеллектуальных систем OSTIS-2011». 2011. С. 21–58.
3. Грищенко М.А., Дородных Н.О., Николайчук О.А., Юрин А.Ю. Применение модельно-управляемого подхода для создания продукционных экспертных систем и баз знаний // Искусственный интеллект и принятие решений. 2016. № 2. С. 16–29.
4. Дородных Н.О., Грищенко М.А., Юрин А.Ю. Система программирования продукционных баз знаний: Personal Knowledge Base Designer. Открытые семантические технологии проектирования интеллектуальных систем. 2016. № 6. С. 209–212.
5. Дородных Н.О., Коршунов С.А., Юрин А.Ю. Средства поддержки моделирования логических правил в нотации RVML // Программные продукты и системы. 2018. № 4. С. 667–672.
6. Дородных Н.О., Николайчук О.А., Юрин А.Ю. Автоматизированное создание продукционных баз знаний на основе деревьев событий // Информационные и математические технологии в науке и управлении. 2017. № 2(6). С. 30–41.
7. Дородных Н.О., Юрин А.Ю. Использование концепт-карт для автоматизированного создания продукционных баз знаний // Программные продукты и системы. 2017. № 4. С. 658–662.
8. Дородных Н.О. Web-ориентированный редактор метамоделей (Web Metamodel Editor). 2018. Свидетельство о государственной регистрации программ для ЭВМ № 2018660815 М.: Федеральная служба по интеллектуальной собственности, патентам и товарным знакам.
9. Дородных Н.О. Web-based software for automating development of knowledge bases on the basis of transformation of conceptual models // Открытые семантические технологии проектирования интеллектуальных систем. 2017. № 7. С. 145–150.
10. Загорулько Ю.А. Семантическая технология разработки интеллектуальных систем, ориентированная на экспертов предметной области // Онтология проектирования. 2015. Т. 15. № 1. С. 30–46.
11. Колесников А.В., Кириков И.В., Листопад С.В. Гибридные интеллектуальные системы с самоорганизацией: координация, согласованность, спор. М.: ИПИ РАН. 2014. 189 с.
12. Рыбина Г.В. Инструментальные средства построения динамических интегрированных экспертных систем: развитие комплекса ат-технология // Искусственный интеллект и принятие решений. 2010. № 1. С. 41–48.
13. Стенников В.А., Барахтенко Е.А., Соколов Д.В. Применение онтологий при реализации концепции модельно-управляемой разработки программного обеспечения для проектирования теплоснабжающих систем // Онтология проектирования. 2014. Т. 14. № 4. С. 54–68.
14. Baumeister J., Striffler A. Knowledge-driven systems for episodic decision support // Knowledge-Based Systems. 2015. Vol. 88. Pp. 45–56.

15. CLIPS: A Tool for Building Expert Systems. 2017. Режим доступа: http://www.clipsrules.sourceforge.net (дата обращения: 2019-04-04).
16. da Silva A.R. Model-driven engineering: A survey supported by the unified conceptual model // Computer Languages, Systems & Structures. 2015. Vol. 43. Pp. 139–155.
17. Djurić D., Gašević D., Devedžić V. The Tao of Modeling Spaces // Journal of Object Technology. 2006. vol. 5. No 8. Pp. 125–147.
18. Ecore structure description (Metamodelling Language). 2019. Режим доступа: http://download.eclipse.org/modeling/emf/emf/javadoc/2.9.0/org/eclipse/emf/ecore/package-summary.html (дата обращения: 04.04.2019).
19. MDA Specifications. 2019. Режим доступа: http://www.omg.org/mda/specs.htm (дата обращения: 04.04.2019).
20. OWL 2 Web Ontology Language Document Overview (Second Edition). 2012. Режим доступа: https://www.w3.org/TR/owl2-overview (дата обращения: 04.04.2019).
21. Ruiz-Mezcua B., Garcia-Crespo A., Lopez-Cuadrado J.L., Gonzalez-Carrasco I. An expert system development tool for non AI experts // Expert Systems with Applications. 2011. Vol. 38. No 1. Pp. 597–609.
22. Schreiber G., Akkermans H., Anjewierden A., Hoog de R., Shadbolt N.R., Velde W. V., Wielinga B. Knowledge Engineering and Management. The CommonKADS methodology // The MIT Press. Cambridge, MA. 2000.
23. Stokes M. Managing engineering knowledge: MOKA: methodology for knowledge based engineering applications (6th ed.). New York: ASME Press. 2001.
24. Studer R., Benjamins V.R., Fensel D. Knowledge engineering: principles and methods // Data & Knowledge Engineering. 1998. Vol. 25. No 1-2. Pp. 161–197.
25. Yurin A.Yu., Berman A.F., Dorodnykh N.O., Nikolaychuk O.A., Pavlov N.Yu. Fishbone diagrams for the development of knowledge bases // Proceedings of the 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO). 2018. Pp. 1136–1141.
26. Yurin A.Yu., Dorodnykh N.O., Nikolaychuk O.A., Grishenko M.A. Designing rule-based expert systems with the aid of the model-driven development approach // Expert Systems, 2018. vol. 35. No 5. Pp. 1–23.

UDK 004.89

METAMODELS FOR SPECIFICATIONS AND CODE GENERATION OF KNOWLEDGE BASES

Nikita O. Dorodnykh

PhD., Matrosov Institute for System Dynamics and Control Theory of Siberian Branch of Russian Academy of Sciences (ISDCT SB RAS)

134, Lermontov Str., 664033, Irkutsk, Russia, e-mail: tualatin32@mail.ru

Olga A. Nikolaychuk

D.Sc., Matrosov Institute for System Dynamics and Control Theory of Siberian Branch of Russian Academy of Sciences (ISDCT SB RAS)

134, Lermontov Str., 664033, Irkutsk, Russia, e-mail: grey.for@gmail.com

Alexander Yu. Yurin

PhD., Head. Laboratory of Information and telecommunication technologies for investigation of technogenic safety, Matrosov Institute for System Dynamics and Control Theory of Siberian Branch of Russian Academy of Sciences (ISDCT SB RAS)

134, Lermontov Str., 664033, Irkutsk, Russia, e-mail: iskander@icc.ru

Sergey A. Korshunov

Programmer, CentrSib LLC, 11, Karl Marx Str., 664003, Irkutsk, Russia,

e-mail: info@centrasib.ru

Abstract. The paper discusses the problem of improving the knowledge bases and intelligent systems design. The efficiency of this process can be improved with the aid of automatic generation of specifications and source codes of knowledge bases for a certain knowledge representation language. The generation, in turn, is based on the analysis and transformation of various information sources, such as conceptual models that describe a subject domain and are presented in different formats (e.g., UML models, concept maps, tree-like semantic structures, fishbone diagrams, etc.). Generation of specifications and codes methodologically is based on the model transformation concept from a model-driven engineering (MDE) and uses metamodels to describe the source and target (CLIPS and OWL) formalisms. The paper describes a method for computer-aided metamodel formation for the analyzed formats of conceptual models, and meta-metamodels for the unified representation and storing of metamodels.

Keywords: intelligent system, knowledge base, model-driven engineering, metamodeling, model transformation, metamodel, conceptual model, code generation

References

1. Bychkov I.V., Dorodnykh N.O., Yurin A.Yu. Podhod k razrabotke programmnyh komponentov dlja formirovanija baz znanij na osnove konceptual'nyh modelej [Approach to the development of software components for generation of knowledge bases based on conceptual models] // Vychislitel'nye tehnologii = Computational Technologies. 2016. vol. 21. No 4. Pp.16–36. (in Russian)

2. Golenkov V.V., Gulyakina N.A. Principy postroeniya massovoj semanticheskoj tekhnologii komponentnogo proektirovaniya intellektualnyh sistem [Principles of building mass semantic technology component design of intelligent systems] // Proceedings of the International scientific-technical conference OSTIS-2011. Minsk: BSUIR. 2011. Pp. 21–58. (in Russian)
3. Grishchenko M.A., Dorodnykh N.O., Nikolaichuk O.A., Yurin A.Yu. Primenenie modelno-upravlyaemogo podhoda dlya sozdaniya produkcionnyh ehkspertnyh system i baz znanij [Application of the model-driven development approach for designing rule-based expert systems and knowledge-bases] // Iskusstvennyj-intellekt-i-prinyatie-reshenij = Artificial Intelligence and decision making. 2016. No 2. Pp. 16–29. (in Russian)
4. Dorodnykh N.O., Grishenko M.A., Yurin A.Yu. Sistema programmirovaniya produkcionnyh baz znanij: Personal Knowledge Base Designer [Software for rule knowledge bases design: Personal Knowledge Base Designer] // Otkrytye-semanticheskie-tekhnologii-proektirovaniya-intellektualnyh-sistem = Open Semantic Technologies for Intelligent Systems. 2016. No 6. Pp. 209–212. (in Russian)
5. Dorodnykh N.O., Korshunov S.A., Yurin A.Yu. Sredstva podderzhki modelirovaniya logicheskih pravil v notacii RVML [Support tools for modeling logical rules in the RVML notation] // Programmnaya inzheneriya = Software engineering. 2018. No 4. Pp. 667–672. (in Russian)
6. Dorodnykh N.O., Nikolaychuk O.A., Yurin A.Yu. Avtomatizirovannoe sozdanie produkcionnyh baz znanij na osnove derev'ev sobytij [Automated creation of rule-based knowledge bases on the basis of event trees] // Informacionnye i matematicheskie tekhnologii v nauke i upravlenii = Information and mathematical technologies in science and management. 2017. No 2(6). Pp. 30–41. (in Russian)
7. Dorodnykh N.O., Yurin A.Yu. Ispol'zovanie koncept-kart dlya avtomatizirovannogo sozdaniya produkcionnyh baz znanij [Using concept maps for rule-based knowledge bases engineering] // Programmnaya inzheneriya = Software engineering. 2017. No 4. Pp. 658–662. (in Russian)
8. Dorodnykh N.O. Web Metamodel Editor. 2018. Certificate of state registration of computer programs No. 2018660815. (in Russian)
9. Dorodnykh N.O. Web-based software for automating development of knowledge bases on the basis of transformation of conceptual models // Otkrytye-semanticheskie-tekhnologii-proektirovaniya-intellektualnyh-sistem = Open Semantic Technologies for Intelligent Systems. 2017. No. 7. Pp. 145–150. (in Russian)
10. Zagorulko Yu.A. Semanticheskaya tekhnologiya razrabotki intellektualnyh sistem orientirovannaya na ehkspertov predmetnoj oblasti [Semantic technology for development of intelligent systems oriented to experts in subject domain] // Ontologiya proektirovaniya = Ontology of design. 2015. vol. 15. No 1. Pp. 30–46. (in Russian)
11. Kolesnikov A.V., Kirikov I.V., Listopad S.V. Gibridnye intellektualnye sistemy s samoorganizaciej: koordinaciya, soglasovannost, spor [Hybrid intelligent systems with self-organization: coordination, consistency, dispute]. Moscow: IPI RAS. 2014. 189 p. (in Russian)
12. Rybina G.V. Instrumentalnye sredstva postroeniya dinamicheskih integrirovannyh ehkspertnyh system razvitie kompleksa at-tekhnologiya [Tools for constructing dynamic integrated expert systems: the development of the complex at-technology] // Iskusstvennyj-intellekt-i-prinyatie-reshenij = Artificial Intelligence and Decision Making. 2010. No 1. Pp. 41–48. (in Russian)

13. Stennikov V.A., Barakhtenko E.A., Sokolov D.V. Primenenie ontologij pri realizacii koncepcii modelno upravlyaemoj razrabotki programmnogo obespecheniya dlya proektirovaniya teplosnabzhayushchih sistem [Usage of ontologies in the implementation of the concept of model-driven engineering for the design of heat supply systems] // Ontologiya proektirovaniya = Ontology of design. 2014. vol. 14. No 4. Pp. 54–68. (in Russian)
14. Baumeister J., Striffler A. Knowledge-driven systems for episodic decision support // Knowledge-Based Systems. 2015. Vol. 88. Pp. 45–56.
15. CLIPS: A Tool for Building Expert Systems. 2017. Режим доступа: http://www.clipsrules.sourceforge.net (дата обращения: 2019-04-04).
16. da Silva A.R. Model-driven engineering: A survey supported by the unified conceptual model // Computer Languages, Systems & Structures. 2015. Vol. 43. Pp. 139–155.
17. Djurić D., Gašević D., Devedžić V. The Tao of Modeling Spaces // Journal of Object Technology. 2006. vol. 5. No 8. Pp. 125–147.
18. Ecore structure description (Metamodelling Language). 2019. Режим доступа: http://download.eclipse.org/modeling/emf/emf/javadoc/2.9.0/org/eclipse/emf/ecore/package-summary.html (дата обращения: 04.04.2019).
19. MDA Specifications. 2019. Режим доступа: http://www.omg.org/mda/specs.htm (дата обращения: 04.04.2019).
20. OWL 2 Web Ontology Language Document Overview (Second Edition). 2012. Режим доступа: https://www.w3.org/TR/owl2-overview (дата обращения: 04.04.2019).
21. Ruiz-Mezcua B., Garcia-Crespo A., Lopez-Cuadrado J.L., Gonzalez-Carrasco I. An expert system development tool for non AI experts // Expert Systems with Applications. 2011. Vol. 38. No 1. Pp. 597–609.
22. Schreiber G., Akkermans H., Anjewierden A., Hoog de R., Shadbolt N.R., Velde W. V., Wielinga B. Knowledge Engineering and Management. The CommonKADS methodology // The MIT Press. Cambridge, MA. 2000.
23. Stokes M. Managing engineering knowledge: MOKA: methodology for knowledge based engineering applications (6th ed.). New York: ASME Press. 2001.
24. Studer R., Benjamins V.R., Fensel D. Knowledge engineering: principles and methods // Data & Knowledge Engineering. 1998. Vol. 25. No 1-2. Pp. 161–197.
25. Yurin A.Yu., Berman A.F., Dorodnykh N.O., Nikolaychuk O.A., Pavlov N.Yu. Fishbone diagrams for the development of knowledge bases // Proceedings of the 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO). 2018. Pp. 1136–1141.
26. Yurin A.Yu., Dorodnykh N.O., Nikolaychuk O.A., Grishenko M.A. Designing rule-based expert systems with the aid of the model-driven development approach // Expert Systems, 2018. vol. 35. No 5. Pp. 1–23.