УДК 004.89
DOI:10.25729/ESI.2024.34.2.007

Использование диаграмм переходов состояний для автоматизированного создания баз знаний

Дородных Никита Олегович, Юрин Александр Юрьевич

Институт динамики систем и теории управления имени В.М. Матросова СО РАН,
Россия, Иркутск, *nikidorny@icc.ru*

Аннотация. Построение баз знаний в форме продукций, онтологий или графов знаний продолжает оставаться достаточно трудоемкой задачей в рамках разработки различных предметно-ориентированных интеллектуальных систем. В данной статье рассмотрены подход и программное средство автоматизации создания баз знаний на основе анализа и преобразования концептуальных моделей в виде диаграмм переходов состояний. Подход основан на выделении структурных элементов диаграмм и их трансформации в конструкции целевого языка представления знаний. Приведено описание основных этапов подхода, анализируемых конструкций рассматриваемого формата диаграмм переходов состояний, а также реализация подхода в форме веб-ориентированной программной системы – Knowledge Modeling System (KMS). Представлен иллюстративный пример преобразования диаграмм переходов состояний для формирования плана анализа отказа технической системы.

Ключевые слова: инженерия знаний, получение знаний, база знаний, онтология, диаграмма переходов состояний, трансформация моделей, генерация кода, продукции

Цитирование: Дородных Н.О. Использование диаграмм переходов состояний для автоматизированного создания баз знаний / Н.О. Дородных, А.Ю. Юрин // Информационные и математические технологии в науке и управлении. – 2024. – № 2(34). – С. 69-81. – DOI:10.25729/ESI.2024.34.2.007.

Введение. Создание интеллектуальных систем является перспективным направлением в сфере информационных технологий. Системы подобного типа способны обрабатывать большие объемы данных, анализировать информацию и принимать решения на основе полученных результатов, что позволяет значительно ускорить процессы в различных отраслях экономики, повысить эффективность работы предприятий и улучшить качество жизни людей. Кроме того, интеллектуальные системы могут быть использованы для решения сложных и слабо-формализованных задач в области медицины [1-3], энергетики [4, 5], техники [6-8], природной и техногенной безопасности [9-11]. Как правило, в основе подобного рода систем лежит База Знаний (БЗ), содержащая формализованные знания предметной области (например, в форме продукций, онтологий или графов знаний), полученные в результате практической деятельности и профессионального опыта, позволяющие специалистам ставить и решать задачи в этой области [12]. Однако, задача создания эффективных методов и средств разработки БЗ интеллектуальных систем не решена в полном объеме до сих пор, поэтому исследования, направленные на развитие методов обработки информации для создания элементов систем, основанных на знаниях, при решении практических задач в различных предметных областях являются актуальными.

На сегодняшний день для построения БЗ используют различные специализированные редакторы, которые позволяют представлять формализованное описание понятий предметной области и структур БЗ на определенном языке представления знаний (ЯПЗ) [13]. Однако такие системы обладают низкой интеграционной способностью со средствами визуального моделирования и модулями интерпретации знаний, в лучшем случаев поддерживая один определенный ЯПЗ. Более того, данные редакторы направлены на хорошо подготовленных специалистов (например, инженеров по знаниям, программистов) и не ориентированы на экспертов предметной области, которые являются основным источником знаний и опыта. Это порождает проблему переноса предметных неформализованных знаний (компетенций) от эксперта через инженера по знаниям в БЗ. Для преодоления данной проблемы разрабатываются разные методики и средства. В частности, очень часто на стадиях извлечения

и структурирования знаний активно используются различные концептуальные (информационные) модели (например, концепт-карты, диаграммы Исикавы, деревья событий или отказов, семантические модели), имеющие общесистемную направленность и ориентированные на систематизацию знаний или поддержку принятия решений [14]. Эти модели являются удобным и понятным для эксперта способом представления знаний. К таким моделям можно также отнести диаграммы переходов состояний (*state transition diagram* или *statechart diagram*) [15], используемые для моделирования поведения системы в зависимости от ее состояний и внешних воздействий. Каждая подобная модель представляет собой граф, где *узлы* – это состояния системы, а *дуги* (*переходы*) – это события или действия, которые приводят к изменению состояния системы. На диаграмме также могут быть указаны условия, выполняемые при каждом переходе. Диаграмма Переходов Состояний (ДПС) используется для проектирования и анализа сложных систем, таких, как программное обеспечение, аппаратные устройства, бизнес-процессы и т.д. Она помогает понять логику работы системы и выявить возможные ошибки в ее работе. Также ДПС может использоваться для документирования уже существующей системы или процесса. Это позволяет легко описывать и передавать знания о работе системы другим специалистам. В целом, ДПС является очень полезным инструментом для любого проекта или задачи, где необходимо четко определить последовательность действий и состояний системы.

Для построения ДПС существуют различные программные средства, выполняющие роль визуальных редакторов и осуществляющие расчеты на основе обработки данных диаграмм. Редакторы обычно представлены как в виде настольных приложений, так и веб-сервисов, в частности:

- Flexberry Designer [16];
- Visual Paradigm Online [17];
- Enterprise Architect [18];
- MATLAB & Simulink – State Diagram [19];
- EASE: State diagram editor [20];
- Draw.io [21];
- Borland CaliberRM [22].

Однако эти системы, хоть и позволяют разрабатывать визуальные диаграммы разной степени абстракции, соответствующие знаниям эксперта, но не предназначены для разработки БЗ и поэтому не могут обеспечить полноту процесса разработки от моделей предметной области до формализованного кода БЗ на определенном ЯПЗ. Эта особенность затрудняет практическое использование построенных диаграмм для создания БЗ при разработке интеллектуальных систем.

Таким образом, целью данной работы является повышение эффективности разработки БЗ на основе анализа и преобразования ДПС. Для этого предлагаются новый подход и программная система – Knowledge Modeling System (KMS) [23], основное назначение которой – поддержка моделирования знаний экспертов в различных предметных визуальных нотациях, в том числе в виде ДПС, и синтез кодов БЗ на основе построенных диаграмм. В качестве целевых формализмов представления знаний выбраны продукции, в частности, ЯПЗ CLIPS (C Language Integrated Production System) [24], таблицы решений (*decision table*) [25] и онтологии, в частности, ЯПЗ OWL (Web Ontology Language) [26]. Данный подход использовался для прототипирования БЗ для решения задач в области техногенной и природной безопасности.

1. Постановка задачи. Задача анализа ДПС и их преобразования в БЗ может быть сведена к задаче трансформации моделей [27]. Трансформация моделей является одной из основных составляющих модельно-ориентированного подхода к разработке программного

обеспечения (*Model-Driven Engineering*) [28]. В общем случае, под трансформацией моделей понимается процесс автоматической генерации целевой модели по исходной модели, в соответствии с некоторым набором правил трансформации. При этом под правилом трансформации подразумевается описание того, как одна или более конструкций на исходном языке моделирования может быть преобразована (отображена) в одну или более конструкций на целевом языке моделирования [29].

В рамках предлагаемого подхода постановку задачи можно формализовать следующим образом:

$$T: CM^{STD} \rightarrow KB, \quad (1)$$

где T – это оператор преобразования исходной концептуальной модели в целевую БЗ; CM^{STD} – это исходная концептуальная модель в виде ДПС (*STD*); KB – это целевая БЗ, при этом: $KB = \langle Code^{CLIPS}, Code^{DT}, Code^{OWL} \rangle$.

Для решения поставленной задачи предлагается специализированный подход, реализованный в виде веб-ориентированной программной системы (KMS), которая обеспечивает поддержку визуального моделирования ДПС и их трансформацию в код БЗ на целевом ЯПЗ. Далее подробнее рассмотрим данный подход и систему.

2. Предлагаемый подход.

2.1. Структура диаграммы переходов состояний. ДПС являются одним из мощных инструментов моделирования и документирования поведения системы. Они представляют собой графическое изображение последовательности состояний системы (*узлов* или *вершин*) и переходов между ними (*стрелки* или *дуги*) при определенных событиях (*условиях*). Таким образом, основная идея ДПС заключается в том, что система имеет некоторые возможные состояния, и при определенных условиях она может перейти из одного состояния в другое. Каждый переход описывается событием, которое вызывает изменение состояния системы. При этом у состояний и переходов могут быть определенные свойства (*характеристики*). Также на данных диаграммах в зависимости от особенностей реализации (например, в UML) изображают начало и конец переходов состояний. Абстрактный пример ДПС, содержащий все основные элементы, представлен на рисунке 1.

Основные особенности ДПС:

- позволяют легко представить поведение системы в виде последовательности состояний и переходов между ними;
- могут использоваться для моделирования как простых, так и сложных систем;
- помогают выявить возможные ошибки в работе системы на этапе ее проектирования;
- используются для документирования работы уже существующих систем, чтобы облегчить понимание их работы и упростить процесс отладки;
- могут быть использованы для создания тестовых сценариев и проверки работоспособности системы или оценки риска.

ДПС широко применяются в различных областях, таких, как: разработка и тестирование программного обеспечения, проектирование и разработка аппаратных (электронных) устройств, автоматизация бизнес-процессов, моделирование работы сложных технических систем и др.

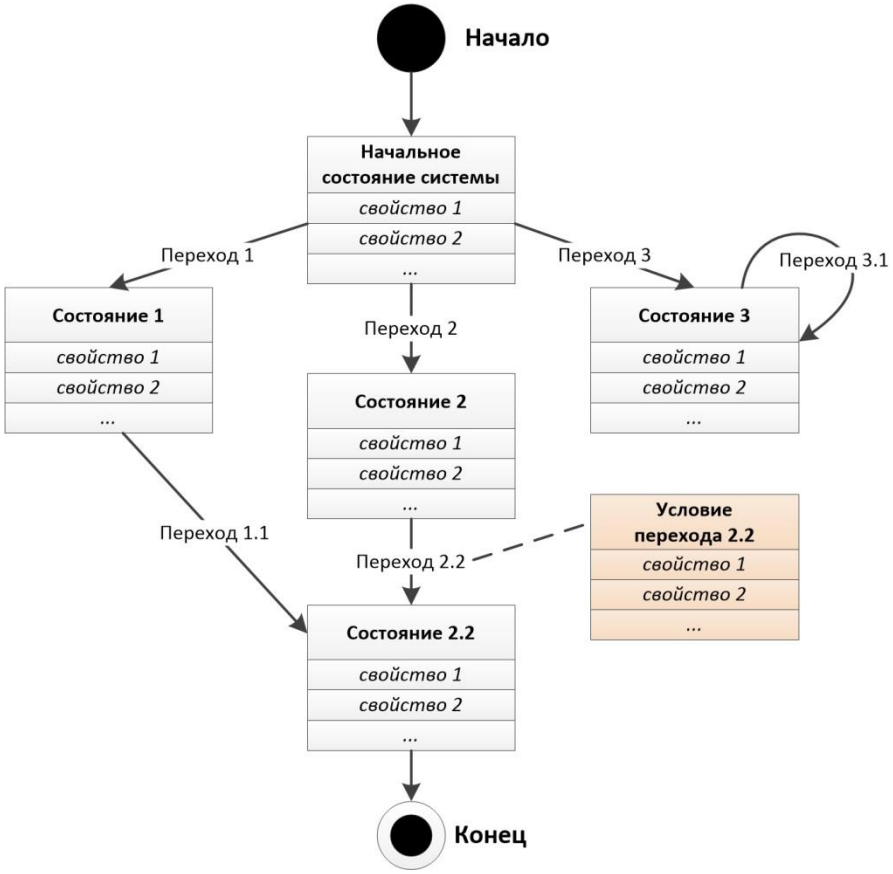

Рис. 1. Абстрактный пример ДПС

Метамодель (*abstract syntax*) ДПС, определяющая в абстрактной форме основные концепты, из которых состоит ДПС, представлена на рисунке 2. Построенная метамодель соответствует мета-метамодели Ecore [30] и в дальнейшем используется в качестве исходной метамодели при разработке правил трансформации, описывающих соответствия между элементами данной метамодели и целевой метамодели БЗ.

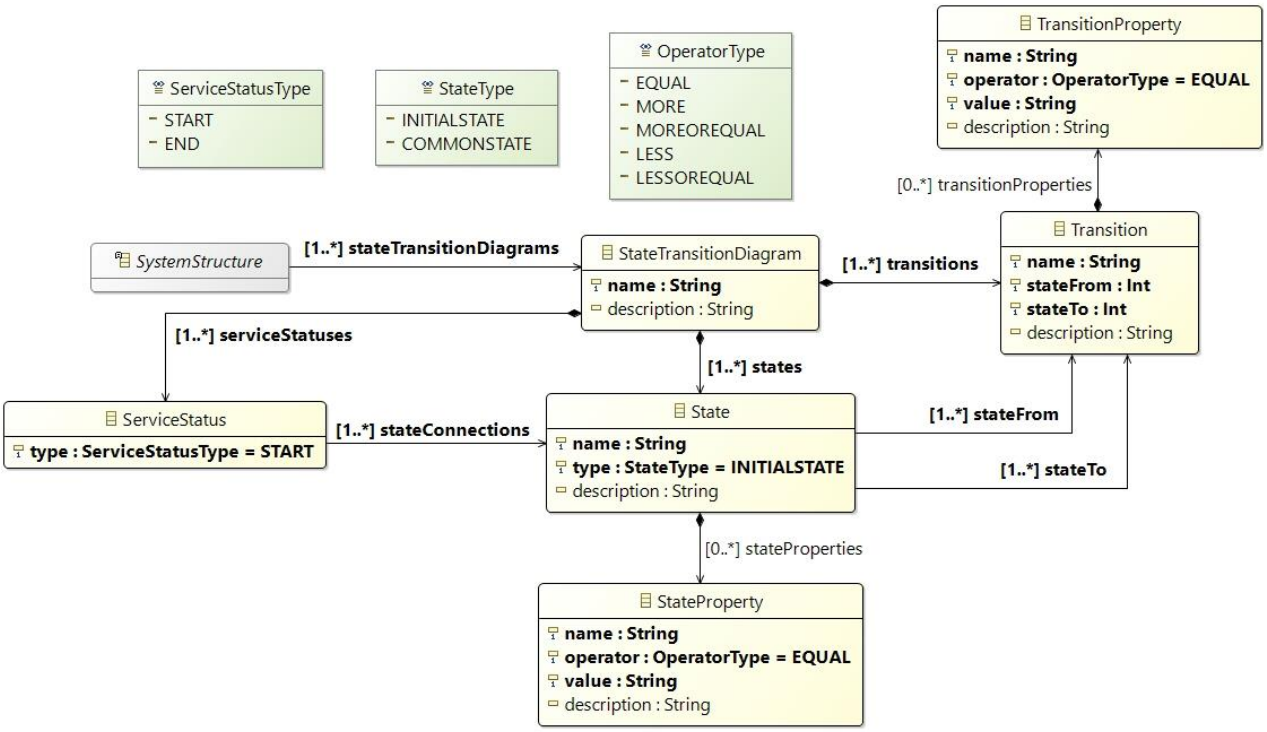

Рис. 2. Метамодель ДПС

На настоящее время не существует единого общепринятого формата или стандарта текстовой нотации для представления ДПС. Учитывая это, а также исходя из специфики ДПС, принято решение о разработке собственного формата (*спецификации*) сериализации ДПС с использованием языка XML, который является универсальным и наиболее распространенным способом для интеграции программных систем и обеспечения обмена информацией между приложениями.

Описание разработанного XML-формата (*concrete syntax*) ДПС, определяющее основные XML-конструкции, представлено в Таблице 1.

Таблица 1. XML-формат представления ДПС

Элементы XML-структуры ДПС	**Описание**
Diagram	Общее описание ДПС, содержащее информацию об исследуемой системе.
State	Информация о состоянии. Также содержит тип состояния с возможными значениями: начальное состояние (*initial state*) и обычное состояние (*common state*).
StateProperty	Информация о свойстве (характеристике) состояния. Также содержит оператор и возможное значение свойства.
Transition	Информация о переходе одного состояния в другое.
TransitionProperty	Информация о свойстве (условии) перехода. Также содержит оператор и возможное значение условия перехода.

3.2. Основные этапы подхода. Для решения поставленной задачи (1) специализируем обобщенный алгоритм трансформации концептуальных моделей в код БЗ, описанный в [31]. Таким образом, основной задачей является преобразование (отображение) элементов исходной ДПС, представленной в формате XML, в конструкции целевого представления БЗ, что может быть представлено в виде последовательности действий (рис. 3).

На этапе 1 при помощи визуального редактора – State Transition Diagram Editor (STDE) [32], входящего в состав системы KMS, пользователь (эксперт предметной области) строит ДПС, описывающую последовательность переходов состояний некоторой системы. На этапе 2 построенная диаграмма представляется (сериализуется) в формате XML с использованием разработанной спецификации. На этапе 3 процесса анализа XML-структуры исходной ДПС выделяются элементы диаграммы и их отношения. Далее (этап 4) на их основе автоматически формируется (генерируется) либо модель продукций, либо модель онтологии (пользователь выбирает необходимую). Данные модели выступают в качестве универсального абстрактного средства представления знаний, не зависящего от целевого ЯПЗ (например, CLIPS, Jess, Drools, SWRL, OWL, RDF) или источника моделей.

При помощи специальной графической нотации – Rule Visual Modeling Language (RVML) [33] предоставляется возможность визуализации, модификации (проверки) полученных продукций (этап 5.1). При этом на этапе 6.1 происходит генерация либо целевой таблицы решений в формате CSV, либо кода БЗ в формате CLIPS на основе сформированной модели продукций.

Возможность модификации онтологической модели осуществляется средствами системы – Knowledge Base Development System (KBDS) [34] (этап 5.2). А на этапе 6.2 происходит генерация кода онтологической БЗ в формате OWL2 DL.

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

$$T = \langle T_{CM-RM}, T_{RM-RKB}, T_{CM-OM}, T_{OM-OKB} \rangle,$$
$$T_{CM-RM}: CM_{XML}^{STD} \to RM,\ T_{RM-RKB}: RM \to RKB,\ RKB = \langle Code^{CLIPS}, Code^{DT} \rangle,$$
$$T_{CM-OM}: CM_{XML}^{STD} \to OM,\ T_{OM-OKB}: OM \to Code^{OWL},$$

где T_{CM-RM} – оператор преобразования ДПС в модель продукций; T_{RM-RKB} – оператор преобразования модели продукций в код БЗ на ЯПЗ CLIPS или таблицу решений в формате CSV; CM_{XML}^{STD} – представление ДПС в XML-формате; RM – представление полученных знаний в виде модели продукций; $Code^{CLIPS}$ – код БЗ на ЯПЗ CLIPS; $Code^{DT}$ – таблица решений в формате CSV; T_{CM-OM} – оператор преобразования ДПС в модель онтологии; T_{OM-OKB} – оператор преобразования модели онтологии в код БЗ на ЯПЗ OWL2 DL; OM – представление полученных знаний в виде модели онтологии; $Code^{OWL}$ – код БЗ на ЯПЗ OWL2 DL.

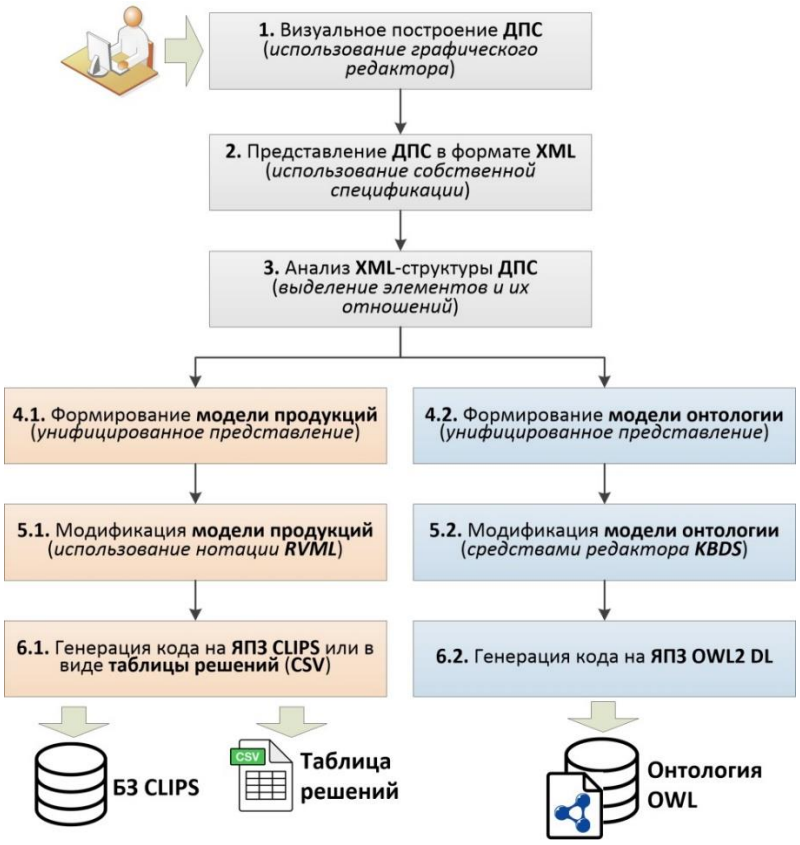

Рис. 3. Основные этапы предлагаемого подхода

3. Программная реализация. Разработанный подход реализован в виде веб-ориентированной программной системы – KMS [26], разработанной на языке PHP с использованием фреймворка Yii2 и паттерна проектирования «Model-View-Controller». Данное средство обладает клиент-серверной архитектурой и ориентировано на непрограммирующих пользователей (например, предметных экспертов, аналитиков данных).

В состав системы KMS входят два визуальных редактора:

– *Extended Event Tree Editor (EETE)* – это графический редактор для визуального моделирования (проектирования) семантических древовидных структур в виде классических или расширенных деревьев событий. Данные диаграммы могут быть также преобразованы в коды БЗ [35];
– *State Transition Diagram Editor (STDE)* – это графический редактор для визуального моделирования (проектирования) ДПС [32].

Основные функции KMS:

– создание, редактирование, просмотр и удаление пользователей;
– аутентификация (вход и выход) и авторизация пользователей (определение прав доступа);
– создание, редактирование, просмотр и удаление проектов, в рамках которых ведется работа с различными диаграммами;

– создание, редактирование, просмотр и удаление ДПС, в рамках определенного проекта;
– создание, редактирование, просмотр и удаление диаграмм деревьев событий (как классических, так и расширенных), в рамках определенного проекта;
– импорт и экспорт диаграмм в виде сериализованных файлов в формате XML;
– автоматическое преобразование диаграмм в модель продукций и онтологий;
– генерация (экспорт) построенных моделей в файлы форматов ЯПЗ CLIPS и OWL2 DL и электронные таблицы формата CSV.

4. Пример практического применения. Рассмотрим пример автоматизированного формирования БЗ на примере фрагмента ДПС. В [7] решалась задача создания программы анализа отказа технической системы для подсистемы-планировщика. В частности, на основе модели динамики технического состояния [7], алгоритма анализа отказа и с учетом структуры программы и алгоритма планирования с помощью специального редактора STDE [35] была построена визуальная модель БЗ в форме ДПС (Рис. 4).

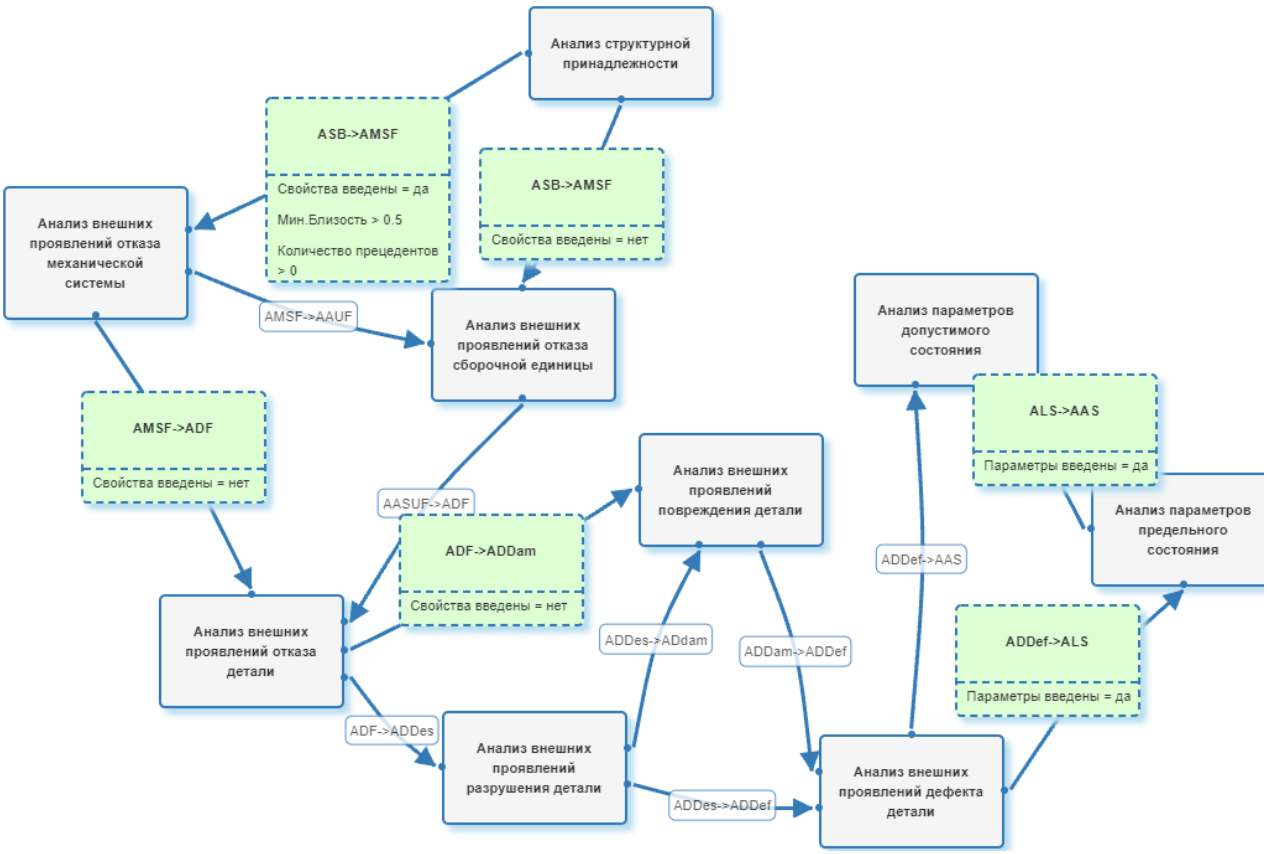

Рис. 4. Пример фрагмента ДПС, описывающей алгоритм анализа отказов

Графическому представлению ДПС (рис. 4) соответствует фрагмент XML-документа (листинг 1). При этом выделены ключевые конструкции (табл. 1), на основе которых осуществлялось извлечение необходимых элементов ДПС.

Листинг 1. Фрагмент XML-кода ДПС, описывающего алгоритм анализа отказов

```
<?xml version="1.0" encoding="UTF-8"?>
<Diagram id="3" name="Д 6.3.5 База знаний планировщика анализа отказа"
description="Диаграмма для базы знаний планировщика анализа отказов">
  <State id="26" name="Анализ структурной принадлежности" type="Initial state"
description=""/>
  <State id="27" name="Анализ внешних проявлений отказа механической системы"
type="State" description=""/>
…
  <Transition id="28" name="ASB-&gt;AMSF" state-from="26" state-to="27"
description="">
```

```
        <TransitionProperty id="30" name="Свойства введены" operator="=" value="да"
description=""/>
        <TransitionProperty id="32" name="Мин.Близость" operator="&gt;" value="0.5"
description=""/>
        <TransitionProperty id="40" name="Количество прецедентов" operator="&gt;"
value="0" description=""/>
      </Transition>
    …
```

На основе извлеченных элементов ДПС сгенерирована модель продукций и онтологии. Далее приводится фрагмент полученной модели продукций в форме диаграммы RVML (рис. 5).

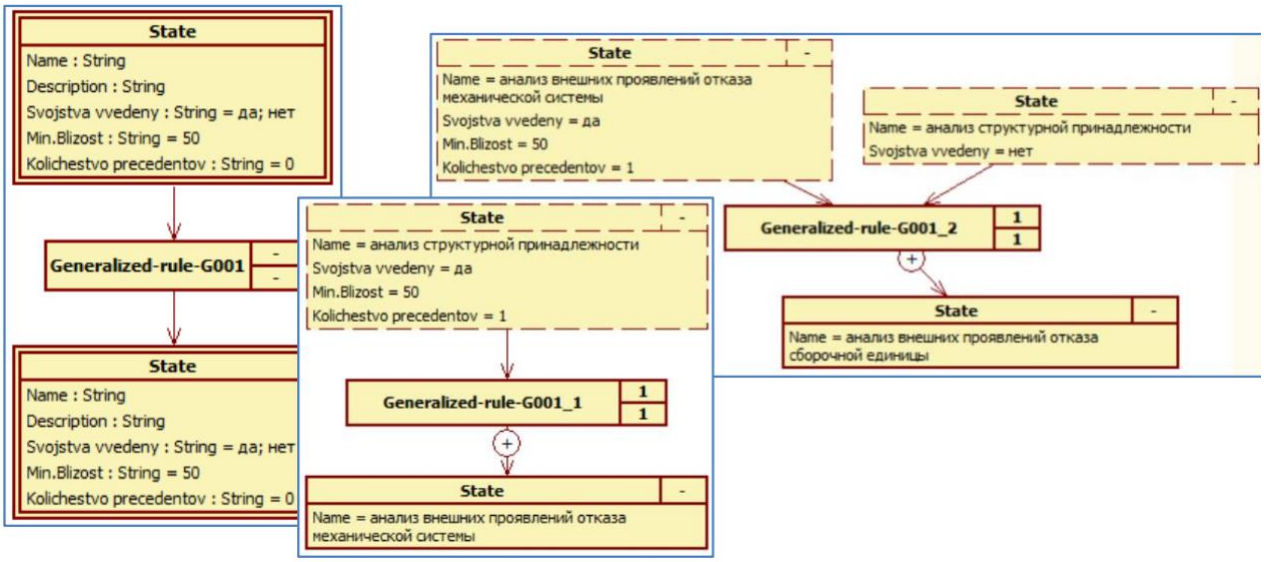

Рис. 5. Пример фрагмента полученной модели продукций в виде диаграммы RVML: шаблон правила и пример двух конкретных правил

Полученная модель продукций была преобразована в код БЗ на ЯПЗ CLIPS с уточнением знаков операторов, определяющих ограничения на значения слотов, и добавлением дополнительных условий. Пример одного из полученных правил приводится в листинге 2. На рисунке 6 также приводится пример фрагмента полученной БЗ в виде таблицы решений в формате CSV.

Листинг 2. Фрагмент CLIPS-кода правила из алгоритма анализа отказов

```
(defrule Generalized-rule-G001-2 "Description of the rule: Generalized-rule-
G001 2"
(declare (salience 1))
 (State ;State
 (Name "АНАЛИЗ ВНЕШНИХ ПРОЯВЛЕНИЙ ОТКАЗА МЕХАНИЧЕСКОЙ СИСТЕМЫ")
 (Svojstva-vvedeny "ДА")
 (Min.Blizost >50)
 (Kolichestvo-precedentov >0)
)
(State ;State
 (Name "АНАЛИЗ СТРУКТУРНОЙ ПРИНАДЛЕЖНОСТИ")
 (Svojstva-vvedeny "НЕТ")
)
(State ;State
 (Name "АНАЛИЗ ВНЕШНИХ ПРОЯВЛЕНИЙ ОТКАЗА МЕХАНИЧЕСКОЙ СИСТЕМЫ")
 (Svojstva-vvedeny "ДА")
 (Min.Blizost <50)
)
=>
(assert
(State ;State
 (Name "АНАЛИЗ ВНЕШНИХ ПРОЯВЛЕНИЙ ОТКАЗА СБОРОЧНОЙ ЕДИНИЦЫ")
```

```
 (GUI-form-name "AAUF")
 ))
)
```

	State	Stat	State	State::Name	#State::Name
2	да	0.5	0	Анализ структурной принадлежности	Анализ внешних проявлений отказа механической системы
3	нет			Анализ структурной принадлежности	Анализ внешних проявлений отказа сборочной единицы
4	да	0.5	0	Анализ внешних проявлений отказа механической системы	Анализ внешних проявлений отказа сборочной единицы
5	нет			Анализ внешних проявлений отказа механической системы	Анализ внешних проявлений отказа детали
6	да	0.5	0	Анализ внешних проявлений отказа сборочной единицы	Анализ внешних проявлений отказа детали
7	да	0.5	0	Анализ внешних проявлений отказа детали	Анализ внешних проявлений разрушения детали
8	нет			Анализ внешних проявлений отказа детали	Анализ внешних проявлений повреждения детали
9	да	0.5	0	Анализ внешних проявлений разрушения детали	Анализ внешних проявлений повреждения детали
10	нет			Анализ внешних проявлений разрушения детали	Анализ внешних проявлений дефекта детали
11	да	0.5	0	Анализ внешних проявлений повреждения детали	Анализ внешних проявлений дефекта детали
12	да			Анализ внешних проявлений дефекта детали	Анализ параметров предельного состояния
13	нет			Анализ внешних проявлений дефекта детали	Анализ параметров допустимого состояния

Рис. 6. Фрагмент полученной таблицы решений в формате CSV

В свою очередь, полученная модель онтологии была преобразована в код онтологической БЗ в формате OWL2 DL (Листинг 3). При этом все состояния интерпретировались как *классы*, а переходы как *объектные-свойства* (*ObjectProperty*). Свойства (*характеристики*) переходов выражались через *свойства-значения* (*DatatypeProperty*), которые задаются для специального служебного класса перехода.

Листинг 3. Фрагмент OWL-кода БЗ, описывающего алгоритм анализа отказов

```
<owl:Class rdf:ID="Состояние"/>
<owl:Class rdf:ID="Переход"/>
<owl:Class rdf:ID="АнализСтруктурнойПринадлежности">
      <rdfs:subClassOf rdf:resource="#Состояние"/>
</owl:Class>
<owl:Class rdf:ID="АнализВнешнихПроявленийОтказаМеханической">
      <rdfs:subClassOf rdf:resource="#Состояние"/>
</owl:Class>
<owl:DatatypeProperty rdf:ID="минБлизость">
      <rdfs:domain rdf:resource="#Переход"/>
      <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#double"/>
</owl:DatatypeProperty>
<owl:ObjectProperty rdf:ID="имеетПереход">
      <rdfs:domain rdf:resource="#Состояние"/>
      <rdfs:range rdf:resource="#Переход"/>
</owl:ObjectProperty>
<owl:ObjectProperty rdf:ID="ASB->AMSF">
      <rdfs:subPropertyOf rdf:resource="#имеетПереход"/>
      <rdfs:domain rdf:resource="#АнализСтруктурнойПринадлежности"/>
      <rdfs:range rdf:resource="#АнализВнешнихПроявленийОтказаМеханической"/>
</owl:ObjectProperty>
…
```

Заключение. Создание предметно-ориентированных интеллектуальных систем является одним из ключевых направлений развития современных информационных технологий, при этом автоматизация разработки БЗ для этих систем является одной из актуальных и перспективных областей в сфере искусственного интеллекта, особенно, когда речь идет об использовании уже накопленной информации, представленной, в частности, в форме концептуальных моделей.

В статье рассмотрен новый подход к автоматизации разработки БЗ на основе ДПС, основанный на анализе структурных элементов ДПС и их преобразовании в конструкции целевого продукционного или онтологического ЯПЗ. Такой подход позволяет, с одной стороны, избежать ошибок программирования на этапе формализации знаний за счет автоматической кодогенерации, а также сократить время, затрачиваемое на разработку БЗ, с другой – непосредственно вовлечь экспертов предметной области в процесс разработки прототипов БЗ, позволяя им создавать программный код, оперируя понятными для них

предметно-ориентированными моделями. Предлагаемый подход реализован в виде веб-ориентированной программной системы KMS, которая может быть использована в качестве инструмента для моделирования предметных знаний и получения сгенерированного кода прототипов БЗ, которые в дальнейшем могут быть доработаны в специализированных средствах и использованы в различных интеллектуальных системах.

Благодарности. Работа выполнена в рамках госзадания Минобрнауки России по проекту «Методы и технологии облачной сервис-ориентированной цифровой платформы сбора, хранения и обработки больших объёмов разноформатных междисциплинарных данных и знаний, основанные на применении искусственного интеллекта, модельно-управляемого подхода и машинного обучения» (№ гос. регистрации: 121030500071-2).

Список источников

1. Грибова В.В. Семантические модели для оценки влияния комплекса факторов на развитие заболеваний / В.В. Грибова, Д.Б. Окунь, Е.А. Шалфеева // Онтология проектирования, 2021. – Т.11. – №4(42). – С.464-477.
2. Ясницкий Л.Н. Нейроэкспертная система диагностики, прогнозирования и управления рисками сердечнососудистых заболеваний / Л.Н. Ясницкий, Ф.М. Черепанов // Прикладная математика и вопросы управления, 2018. – №3. – С.107-126.
3. Alam F., Giglou H.B., Malik K.M. Automated clinical knowledge graph generation framework for evidence based medicine. Expert systems with applications, 2023, vol .233, 120964.
4. Воропай Н.И. ИТ-инфраструктура для построения интеллектуальных систем управления развитием и функционированием систем энергетики на основе цифровых двойников и цифровых образов / Н.И. Воропай, Л.В. Массель, И.Н. Колосок, А.Г. Массель // Известия Российской академии наук. Энергетика, 2021. – №1. – С.3-13.
5. Zhang L., Su H., Zio E., Jiang L., Fan L., Zhang J. A graph structure feature-based framework for the pattern recognition of the operational states of integrated energy systems. Expert systems with applications, 2023, vol.213, 119039.
6. Берман А.Ф. Технология создания основанных на знаниях систем для исследования динамики состояния технических объектов / А.Ф. Берман, Н.О. Дородных, О.А. Николайчук, А.Ю. Юрин // Вычислительные технологии, 2023. – Т.28. – №4. – С.94-108.
7. Берман А.Ф. Интеллектуальная информационная система анализа отказов / А.Ф. Берман, О.А. Николайчук, А.Ю. Юрин // Проблемы машиностроения и надежности машин, 2012. – №4. – С.88-96.
8. Zhang J., Pu X., Zhao R., Li J., Nie Z. Implicit contradictions identification and solution process model for complex technical systems. Computers & industrial engineering, 2023, vol.177, 108822.
9. Ноженкова Л.Ф. Создание комплексной системы безопасности региона на основе системной интеграции технологий / Л.Ф. Ноженкова, В.В. Ничепорчук, А.И. Ноженков // Информатизация и связь, 2013. – №2. – С. 122-124.
10. Nicheporchuk V., Favorskaya M.N., Gryazin I. Framework for intelligent wildlife monitoring. Smart innovation, Systems and technologies, 2020, vol.193, pp. 167-177.
11. Nikolaychuk O.A., Pestova J.V., Yurin A.Yu. Wildfire susceptibility mapping in Baikal natural territory using random forest. Forests, 2024, vol.15, no.1, 170.
12. Гаврилова Т.А. Инженерия знаний. Модели и методы / Т.А. Гаврилова, Д.В. Кудрявцев, Д.И. Муромцев. – СПб.: Лань, 2016. – 324 с.
13. Джарратано Дж. Экспертные системы: принципы разработки и программирования, 4-е издание: пер. с англ. / Дж. Джарратано, Г. Райли. – М.: Вильямс, 2007. – 1152 с.
14. Гаврилова Т.А. Визуально-аналитическое мышление и интеллект-карты в онтологическом инжиниринге / Т.А. Гаврилова, Э.В. Страхович // Онтология проектирования, 2020. – Т.10. – №1(35). – С.87-99.
15. Хопкрофт Дж. Введение в теорию автоматов, языков и вычислений, 2-е издание: пер. с англ. / Дж. Хопкрофт, Р. Мотвани, Дж. Ульман. – М.: Вильямс, 2008. – 528 с.
16. Flexberry Designer. Available at: https://flexberry.net/ru/developers-flexberry-designer.html (accessed: 01/21/2024).
17. Visual paradigm online. Available at: https://online.visual-paradigm.com/ (accessed: 01/21/2024).
18. Enterprise Architect. Available at: https://sparxsystems.com/ (accessed: 01/21/2024).
19. MATLAB & Simulink – State Diagram. Available at: https://www.mathworks.com/discovery/state-diagram.html (accessed: 01/21/2024).

20. EASE: State diagram editor. Available at: https://www.hdlworks.com/products/ease/state_diagram.html (accessed: 01/21/2024).
21. Draw.io. Available at: https://app.diagrams.net/ (accessed: 01/21/2024).
22. Borland CaliberRM. Available at: https://borland-caliber.software.informer.com/11.4/ (accessed: 01/21/2024).
23. Knowledge Modeling System (KMS). Available at: http://kms.knowledge-core.ru/ (accessed: 01/21/2024).
24. CLIPS: A tool for building expert systems. Available at: https://www.clipsrules.net/ (accessed: 01/21/2024).
25. Еремеев А.П. Развитие аппарата таблиц решений в контексте создания гибридных интеллектуальных систем / А.П. Еремеев // Гибридные и синергетические интеллектуальные системы: матер. III Всерос. Поспеловской конф. с междунар. участием, 2016. – С.121-132.
26. OWL 2 web ontology language document overview (Second Edition). 2012, available at: https://www.w3.org/TR/owl2-overview (accessed: 01/21/2024).
27. Mens T., Gorp P.V. A taxonomy of model transformations. Electronic notes in theoretical computer science, 2006, vol.152, pp. 125-142.
28. 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.
29. Czarnecki K., Helsen S. Feature-based survey of model transformation approaches. IBM Systems Journal, 2006, vol.45, no.3, pp. 621-645.
30. Ecore structure description (Metamodelling Language), available at: https://download.eclipse.org/modeling/emf/emf/javadoc/2.11/org/eclipse/emf/ecore/package-summary.html (accessed: 01/21/2024).
31. Бычков И.В. Подход к разработке программных компонентов для формирования баз знаний на основе концептуальных моделей / И.В. Бычков, Н.О. Дородных, А.Ю. Юрин // Вычислительные технологии, 2016. – Т. 21. – №4. – С.16-36.
32. Дородных Н.О. State transition diagram editor / Н.О. Дородных, Д.Н. Шпаченко, А.Ю. Юрин // Свидетельство о государственной регистрации программы для ЭВМ. М. Рег. № 2022664023 от 22.07.22.
33. Rule visual modeling language (RVML. Available at: http://knowledge-core.ru/index.php?p=rvml&lan=ru (accessed: 01/21/2024).
34. Knowledge base development system (KBDS). Available at: http://www.kbds.knowledge-core.ru/ (accessed: 01/21/2024).
35. Дородных Н.О. Автоматизированное создание продукционных баз знаний на основе деревьев событий / Н.О. Дородных, О.А. Николайчук, А.Ю. Юрин // Информационные и математические технологии в науке и управлении, 2017. – №2(6). – С.30-41.

Дородных Никита Олегович. *К.т.н., старший научный сотрудник Института динамики систем и теории управления им. В.М. Матросова СО РАН (ИДСТУ СО РАН). Основные направления исследований: автоматизация создания интеллектуальных систем и баз знаний, получение знаний из документов, таблиц, концептуальных моделей, семантическая интерпретация таблиц. ORCID: 0000-0001-7794-4462, Author ID (RSCI): 979843, Author ID (Scopus): 57202323578, Researcher ID: E-8870-2014, tualatin32@mail.ru.*

Юрин Александр Юрьевич. *Д.т.н., заведующий лабораторией Информационных технологий исследования природной и техногенной безопасности ИДСТУ СО РАН, профессор Института информационных технологий и анализа данных ИрНИТУ. Основные направления исследований: разработка интеллектуальных систем и баз знаний, использование прецедентного подхода и семантических технологий при проектировании интеллектуальных диагностических систем. ORCID: 0000-0001-9089-5730, AuthorID (RSCI): 174845, AuthorID (Scopus): 16311168300, Researcher ID: A-4355-2014. iskander@icc.ru.*

UDC 004.89
DOI:10.25729/ESI.2024.34.2.007

Using state transition diagrams for automated creation of knowledge bases

Nikita O. Dorodnykh, Alexander Yu. Yurin

Matrosov institute for system dynamics and control theory SB RAS,
Russia, Irkutsk, *nikidorny@icc.ru*

Abstract. Building knowledge bases in the form of rules, ontologies, or knowledge graphs continues to be a rather time-consuming task in the development of various domain-specific intelligent systems. This article discusses an approach and software for automating the creation of knowledge bases using the analysis and transformation of conceptual models in the form of state transition diagrams. The approach is based on the identification of structural elements of diagrams and their mapping into constructions of the target knowledge representation language. The main stages of the approach are described, as are the analyzed constructions of the considered format of state transition diagrams, as well as the implementation of the approach in the form of web-oriented software, namely, Knowledge Modeling System (KMS). An illustrative example of converting state transition diagrams to form a failure analysis plan is presented.

Keywords: knowledge engineering, knowledge acquisition, knowledge base, ontology, state transition diagram, model transformation, code generation, rules

Acknowledgements: The reported study was supported by the Ministry of Education and Science of the Russian Federation (Project no. 121030500071-2 "Methods and technologies of a cloud-based service-oriented platform for collecting, storing and processing large volumes of multi-format interdisciplinary data and knowledge based upon the use of artificial intelligence, model-driven approach and machine learning").

References

1. Gribova V.V., Okun D.B., Shalfeeva E.A. Semanticheskie modeli dlya ocenki vliyaniya kompleksa faktorov na razvitie zabolevanij [Semantic models for assessing the influence of a combination of factors on the development of diseases]. Ontologiya proyektirovaniya [Ontology of designing], 2021, vol.11, no.4, pp. 464-477.
2. Yasnickij L.N., Cherepanov F.M. Nejroekspertnaya sistema diagnostiki, prognozirovaniya i upravleniya riskami serdechnososudistyh zabolevanij [Neuroexpert system for diagnostics, prediction and risk management of cardiovascular diseases]. Prikladnaya matematika i voprosy upravleniya [Applied mathematics and management issues], 2018, no.3, pp. 107-126.
3. Alam F., Giglou H.B., Malik K.M. Automated clinical knowledge graph generation framework for evidence based medicine. Expert systems with applications, 2023, vol.233, 120964.
4. Voropai N.I., Massel L.V., Kolosok I.N., Massel A.G. IT-infrastruktura dlya postroeniya intellektual'nyh sistem upravleniya razvitiem i funkcionirovaniem sistem energetiki na osnove cifrovyh dvojnikov i cifrovyh obrazov [IT-infrastructure for construction of intelligent management systems of development and functioning of energy systems based on digital twins and digital images]. Izvestiya Rossiyskoy akademii nauk. Energetika [Proceedings of the Russian academy of sciences. Power engineering], 2021, no.1, pp. 3-13.
5. Zhang L., Su H., Zio E., Jiang L., Fan L., Zhang J. A graph structure feature-based framework for the pattern recognition of the operational states of integrated energy systems. Expert systems with applications, 2023, vol.213, 119039
6. Berman A.F., Dorodnykh N.O., Nikolaychuk O.A., Yurin A.Yu. Tekhnologiya sozdaniya osnovannyh na znaniyah sistem dlya issledovaniya dinamiki sostoyaniya tekhnicheskih obektov [Technology for creating knowledge-based systems for studying the dynamics of the state of technical objects]. Vychislitel'nyye tekhnologii [Computational technologies], 2023, vol.28, no.4, pp. 94-108.
7. Berman A.F., Nikolaychuk O.A., Yurin A.Yu. Intellektual'naya informatsionnaya sistema analiza otkazov [Intellectual data system for analyzing failures]. Problemy mashinostroyeniya i nadezhnosti mashin [Journal of machinery manufacture and reliability], 2012, vol. 41 (4), pp. 337-343.
8. Zhang J., Pu X., Zhao R., Li J., Nie Z. Implicit contradictions identification and solution process model for complex technical systems. Computers & Industrial Engineering, 2023, vol.177, 108822.
9. Nozhenkova L.F., Nicheporchuk V.V., Nozhenkova A.I. Sozdanie kompleksnoj sistemy bez-opasnosti regiona na osnove sistemnoj integracii tehnologij [Creating the comprehensive regional safety system based on system integration of technologies]. Informatizatsiya i svyaz' [Informatization and communication], 2013, no.2, pp. 122-124.
10. Nicheporchuk V., Favorskaya M.N., Gryazin I. Framework for intelligent wildlife monitoring. Smart innovation, Systems and technologies, 2020, vol.193, pp. 167-177.
11. Nikolaychuk O.A., Pestova J.V., Yurin A.Yu. Wildfire susceptibility mapping in baikal natural territory using random forest. Forests, 2024, vol.15, no.1, 170.
12. Gavrilova T.A., Kudryavtsev D.V., Muromtsev D.I. Inzhenerija znanij. Modeli i metody [Knowledge Engineering. Models and methods]. SPb., Lan, 2016, 324 p.
13. Giarratano J.C., Riley G.D. Ekspertnyye sistemy: printsipy razrabotki i programmirovaniya, 4-ye izdaniye: per. s angl. [Expert systems: Principles and Programming. 4th ed. trans. from English]. M., Williams, 2007. – 1152 p.
14. Gavrilova T.A., Strakhovich E.V. Vizual'no-analiticheskoe myshlenie i intellekt-karty v ontologicheskom inzhiniringe [Visual analytical thinking and mind maps for ontology engineering]. Ontologiya proyektirovaniya [Ontology of designing], 2020, vol.10, no.1, pp. 87-99.
15. Hopcroft J.E., Motwani R., Ullman J.D. Vvedeniye v teoriyu avtomatov, yazykov i vychisleniy, 2-ye izdaniye: per. s angl. [Introduction to automata theory, languages, and computation. 2rd ed.]. M., Williams, 2008, 528 p.

16. Flexberry Designer. Available at: https://flexberry.net/ru/developers-flexberry-designer.html (accessed: 01/21/2024).
17. Visual paradigm online. Available at: https://online.visual-paradigm.com/ (accessed: 01/21/2024).
18. Enterprise Architect. Available at: https://sparxsystems.com/ (accessed: 01/21/2024).
19. MATLAB & Simulink – State Diagram. Available at: https://www.mathworks.com/discovery/state-diagram.html (accessed: 01/21/2024).
20. EASE: State diagram editor. Available at: https://www.hdlworks.com/products/ease/state_diagram.html (accessed: 01/21/2024).
21. Draw.io. Available at: https://app.diagrams.net/ (accessed: 01/21/2024).
22. Borland CaliberRM. Available at: https://borland-caliber.software.informer.com/11.4/ (accessed: 01/21/2024).
23. Knowledge Modeling System (KMS). Available at: http://kms.knowledge-core.ru/ (accessed: 01/21/2024).
24. CLIPS: A tool for building expert systems. Available at: https://www.clipsrules.net/ (accessed: 01/21/2024).
25. Eremeev A.P. Razvitie apparata tablic reshenij v kontekste sozdaniya gibridnyh intellektual'nyh sistem [The development of decision tables apparatus in the context of creating hybrid intelligent systems]. Gibridnyye i sinergeticheskiye intellektual'nyye sistemy: mater. III Vseros. Pospelovskoy konf. s mezhdunar. uchastiyem [Proc. 3rd All-Russ. Pospelov's conf. on hybrid and synergistic intelligent systems], 2016, pp. 121-132.
26. OWL 2 web ontology language document overview (Second Edition). 2012, available at: https://www.w3.org/TR/owl2-overview (accessed: 01/21/2024).
27. Mens T., Gorp P.V. A taxonomy of model transformations. Electronic notes in theoretical computer science, 2006, vol.152, pp. 125-142.
28. 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.
29. Czarnecki K., Helsen S. Feature-based survey of model transformation approaches. IBM Systems Journal, 2006, vol.45, no.3, pp. 621-645.
30. Ecore structure description (Metamodelling Language), available at: https://download.eclipse.org /modeling/emf/emf/javadoc/2.11/org/eclipse/emf/ecore/package-summary.html (accessed: 01/21/2024).
31. 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'nyye tekhnologii [Computational technologies], 2016, vol.21, no.4, pp.16-36.
32. Dorodnykh N.O., Shpachenko D.N., Yurin A.Yu. State transition diagram editor. Certificate of state registration of a computer program. M. Reg. No. 2022664023, dated 07/22/22.
33. Rule visual modeling language (RVML). Available at: http://knowledge-core.ru/index.php?p=rvml&lan=ru (accessed: 01/21/2024).
34. Knowledge base development system (KBDS). Available at: http://www.kbds.knowledge-core.ru/ (accessed: 01/21/2024).
35. 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]. Informatsionnyye i matematicheskiye tekhnologii v nauke i upravlenii [Information and mathematical technologies in science and management], 2017, vol.2, no.6, pp. 30-41.

Dorodnykh Nikita Olegovych. *Ph.D., senior associate researcher at Matrosov institute for system dynamics and control theory SB RAS (ISDCT SB RAS). Main research domains: computer-aided development of intelligent systems and knowledge bases, knowledge acquisition based on documents, tables and conceptual models, semantic table interpretation. ORCID: 0000-0001-7794-4462, Author ID (RSCI): 979843, Author ID (Scopus): 57202323578, Researcher ID (WoS): E-8870-2014, tualatin32@mail.ru.*

Yurin Alexander Yurievich. *Ph.D., head of a laboratory "Information and telecommunication technologies for investigation of natural and technogenic safety" at ISDCT SB RAS and associate professor of INRTU. Main research domains: development of intelligent systems and knowledge bases, application of the case-based reasoning and semantic technologies in the design of diagnostic intelligent systems, maintenance of reliability and safety of complex technical systems. ORCID: 0000-0001-9089-5730, Author ID (RSCI): 174845, Author ID (Scopus): 16311168300, Researcher ID (WoS): A-4355-2014, iskander@icc.ru.*

Статья поступила в редакцию 09.02.2024; одобрена после рецензирования 08.04.2024; принята к публикации 06.06.2024.

The article was submitted 02/09/2024; approved after reviewing 04/08/2024; accepted for publication 06/06/2024.