УДК 004.89

ТЕХНОЛОГИЯ АВТОМАТИЗАЦИИ ПРОЦЕССА ФОРМИРОВАНИЯ СИСТЕМ ПОНЯТИЙ И БАЗ ЗНАНИЙ ДЛЯ ПРЕДМЕТНЫХ ОБЛАСТЕЙ С ОБЪЕКТАМИ СЛОЖНОЙ СТРУКТУРЫ

К. А. Гуляева, старший преподаватель департамента ПИиИИ ИМКТ ДВФУ, Владивосток, Россия; orcid.org/0000-0003-0226-5072, e-mail: karinagulyaeva8@gmail.com
И. Л. Артемьева, д.т.н., профессор департамента ПИиИИ ИМКТ ДВФУ, Владивосток, Россия; orcid.org/0000-0003-2088-5259, e-mail: artemeva.il@dvfu.ru

*Рассматривается задача конструирования баз знаний, основанных на онтологии предметной области. **Целью работы** является описание технологии автоматизации процесса формирования систем понятий и баз знаний для предметных областей с объектами сложной структуры. Формализован процесс создания метаграфа системы. Процесс формирования систем понятий автоматизирован посредством разработанных процедуры исследования окрестности каждой вершины метаграфа системы и введения специального множества метапонятий. Описаны категории ситуаций, возникающие при исследовании окрестности вершины метаграфа системы. Инициализирующие шаблоны, отражающие внутреннюю структуру групп свойств систем, определены. Технология позволяет эксперту генерировать промежуточные понятия онтологии регулярным образом. Часть типов аргументов создаваемых экспертом функций и предикатов базы знаний порождается на основании метапонятия автоматически, а часть – выбирается из конечного множества вариантов. Разработанная технология может быть использована при создании редактора базы знаний для предметной области с объектами сложной структуры. Данные проектные решения позволяют получить практически полезные технологии проектирования схем баз знаний.*

Ключевые слова: *база знаний, интеллектуальная система, метаграф, онтология, предметная область, система сложной структуры, меронимическая система, теория.*

DOI: 10.21667/1995-4565-2023-85-65-81

Введение

Интеллектуальная система традиционно включает удобный интерфейс пользователя, решатель задач, интеллектуальный редактор базы знаний (БЗ) и БЗ. Качество интеллектуальной системы и доверие пользователя к ней определяется качеством и доверием пользователя к ее БЗ, структура которой зависит от предметной области (ПрО).

ПрО могут образовывать классы на основании наличия или отсутствия у них определенных признаков. Достаточно широким классом ПрО является класс ПрО, объекты которых связаны отношением меронимии (иначе – холо-партитивные отношения, отношения «часть – целое», «Part Of», «APO» (A-Part-Of)). Многие современные исследования посвящены изучению семантической природы отношения меронимии и образованию различных классификаций данного отношения [1]. В настоящем исследовании изучается класс ПрО, где объекты связаны отношением меронимии и могут представлять собой: элементарные объекты, не имеющие элементов; объекты, имеющие элементами другие объекты; объекты-подмножества и объекты-процессы (участниками процесса являются другие объекты). Указанный класс ПрО далее называется *классом ПрО*, где объекты представляют собой *системы сложной структуры*, а *меронимическая система* кратко называется *системой*.

БЗ для ПрО данного класса должна удовлетворять определённым требованиям, так как содержит информацию об объектах особого рода. У каждого объекта (системы) сложной структуры существует набор свойств, присущих ему как целому, а также наборы свойств каждого из его компонентов в контексте окружения – присутствия или отсутствия других компонентов объекта и их свойств (например, длина связи изолированных атомов и длина

связи атомов в присутствии других атомов молекулы). Кроме этого, свойства объекта и его компонентов могут определяться с учётом условий внешней среды, изменяться во времени и/или пространстве. Примерами таких ПрО могут служить ПрО определения реакционных способностей химических соединений (химическая реакция, химическое соединение и функциональная группа – примеры систем сложной структуры), ПрО развития живых клеток бактерий и архей, ПрО состояния и развития процессов экономической (процессы производства и потребления – примеры систем сложной структуры) или экологической систем. Задача настоящего исследования – описание технологии автоматизации процесса формирования систем понятий и БЗ для ПрО с объектами сложной структуры.

Важные шаги на данном пути уже предприняты. Большое число современных исследований в области искусственного интеллекта посвящено моделям и технологиям создания интеллектуальных платформ, частью которых являются интеллектуальные редакторы и/или метаредакторы (средства «приема» БЗ от эксперта ПрО): графодинамические модели проекта OSTIS [2], двухуровневая орграфовая модель метаинформации и целевой информации предметно-независимой платформы IACPaaS [3-5], Web-портал знаний ИСИ СО РАН, в частности для слабоформализованных ПрО [6], интеллектуальный редактор «снизу-вверх» Web Protégé [7], оболочки, редакторы и метаредакторы И. Л. Артемьевой, основанные на логических моделях ПрО [8], и др. Многие платформы основаны на онтологическом подходе и в большей или меньшей степени содержат различные средства поддержки жизнеспособности интеллектуальных систем, подразумевающие адаптируемость и адаптивность (данным концепциям уделено значительное внимание в работе [4]) из-за изменчивости знаний и классов решаемых задач в процессе опытной эксплуатации интеллектуальной системы. Далее рассматриваются только БЗ, основанные на онтологии, где математическая абстракция *онтологии* есть $(\Sigma, Axiom_{\Sigma})$, где Σ – сигнатура, $Axiom_{\Sigma}$ – множество аксиом о свойствах терминов ПрО (данный подход подробно описан в [8-10]).

Несмотря на наличие упомянутых средств первичного приема БЗ и ее последующей коррекции, полнота и правильность БЗ зависят только от эксперта ПрО, а после исключения инженера знаний – формирование общей структуры БЗ, основанной на онтологии, промежуточных понятий, понятий различных уровней общности – также легли на плечи экспертов. Соответственно, необходим набор дополнительных инструментальных средств, позволяющих упростить процесс формирования систем понятий для эксперта. Возникает гипотеза о существовании возможности автоматизации самого процесса формирования систем понятий и БЗ для некоторых ПрО. В следующих разделах подтверждается гипотеза о существовании возможности автоматизации процесса формирования систем понятий и БЗ для класса ПрО, где объекты представляют собой системы сложной структуры. **Целью работы** является описание технологии автоматизации процесса формирования систем понятий и БЗ для ПрО с объектами сложной структуры. Автоматизация процесса получения системы понятий происходит за счёт разработанных процедуры исследования окрестности каждой вершины метаграфа системы и введения специального множества метапонятий. Подробности разработанной технологии изложены далее.

Глоссарий

Далее введены некоторые определения.

Структурные типы систем. Будем называть *элементарной* систему, не имеющую подсистем (предполагается, что процедура рекурсивного спуска вглубь устройства системы когда-то останавливается, а именно, когда информация об устройстве подсистемы не используется в решении практических задач, для которых разрабатывается интеллектуальная система). Элементарная система может иметь только один экземпляр (например, элементарная система «Электрон»). Экземпляры ее непосредственных (прямых) надсистем могут иметь лишь разное количество, ассоциированное с единственным экземпляром элементарной системы (например, экземпляр «Ядро атома дейтерия» системы «Ядро атома» имеет число 1, ассоциированное с единственным экземпляром элементарной системы «Нейтрон», а экземпляр

«Ядро атома трития» системы «Ядро атома» имеет число 2, ассоциированное с единственным экземпляром элементарной системы «Нейтрон»). Будем называть *дескриптивной* систему, не являющуюся элементарной, экземпляры которой имеют *элементами* экземпляры иных систем (подсистем), которые вместе образуют определенную целостность (например, дескриптивная система «Атом» имеет подсистемами системы «Ядро атома» и «Электронная оболочка атома»). Будем называть *системой-подмножеством* систему, не являющуюся элементарной, все элементы которой являются и элементами ее непосредственной (прямой) надсистемы (например, система «Функциональная группа», непосредственной (прямой) подсистемой для которой является система «Атом», является системой-подмножеством для системы «Химическое соединение»). Системы-подмножества могут быть отождествлены со своими непосредственными (прямыми) надсистемами. Так, экземпляр «Бензол» системы «Функциональная группа» может рассматриваться и как экземпляр системы «Химическое соединение», которая является непосредственной (прямой) надсистемой «Функциональной группы». Будем называть *процессом* систему, обозначающую совокупность взаимосвязанных или взаимодействующих видов деятельности, преобразующих входы в выходы (например, система-процесс «Химическая реакция») [11]. Процесс обязан иметь хотя бы одну непосредственную (прямую) подсистему – участника процесса (с указанием роли участника в процессе). Для экземпляров процессов должно быть задано максимальное количество шагов процесса.

Далее рассматриваются системы, структурный тип которых принадлежит одному из описанных типов. $System \in \{Elementary\ System\ (E),\ \ Descriptive\ Element\ System\ (D),\ Descriptive\ Subset\ System\ (S), Process\ (P)\}$.

Объект (система) сложной структуры (определение 1.1). Будем называть *объектами (системами) сложной структуры* объекты (системы) ПрО такого рода, в которых можно выделить подсистемы указанных выше типов.

База знаний о системе сложной структуры (определение 1.2). Будем называть *БЗ о системе сложной структуры* БЗ интеллектуальной системы, знания которой должны описывать следующие аспекты:

– внутреннюю структуру системы;

– критерии вхождения или отсутствия каждого из экземпляров подсистем в экземпляр системы (например, описание наличия функциональной группы в структуре химического соединения);

– свойства системы как таковой как отдельной сущности;

– свойства каждой из её подсистем как отдельных сущностей;

– свойства сочетаний каждой из её подсистем (например, совместные свойства протонов и нейтронов как подсистем ядра атома или совместные свойства фирм и домохозяйств как подсистем рынка);

– свойства системы при взаимодействии с внешней средой;

– свойства каждой из ее подсистем при взаимодействии с внешней средой;

– свойства сочетаний каждой из ее подсистем при взаимодействии с внешней средой.

Предметная область, объекты которой являются системами сложной структуры (определение 1.3). Будем называть *ПрО, объекты которой являются системами сложной структуры*, ПрО, в которой можно выделить подобласть, для которой выполняется:

– любой объект ПрО является системой сложной структуры или ее подсистемой (в смысле определения 1.1);

– БЗ данной ПрО является БЗ о системах сложной структуры (в смысле определения 1.2), где системы являются денотатами объектов ПрО;

– (*опционально*) при определении понятий и отношений между понятиями ПрО можно выделить множество «сквозных» понятий, участвующих в описании объективной реальности и условий возможности изменения окружающего мира, но не входящих во множество объектов ПрО (например, сквозными понятиями можно считать шаги процесса, моменты наблю-

дения процесса, давление, температуру, различные виды энергии и др.). Множество сквозных понятий может быть пусто.

Классом ПрО, объекты которых являются системами сложной структуры, называется совокупность ПрО в смысле определения 1.3.

Метаграф системы сложной структуры

Носители экспертных знаний в ПрО, объекты которых являются системами сложной структуры, имеют представление о системе сложной структуры в форме метаграфа, который неявно принимает участие в процессе формирования системы понятий ПрО. В данном разделе приведено формальное описание процесса формирования метаграфа системы. Далее для пояснения понятия метаграфа системы (в качестве напоминания) приведены некоторые известные математические определения, в частности понятие направленного ациклического графа, ставшее отправной точкой.

Направленным ациклическим графом (Directed Acyclic Graph, DAG) называется граф G с ориентированными ребрами, в котором нет циклов [12]. Ориентированным графом называется пара$(K,\ R \subseteq K \times K)$, где K – множество вершин, R – бинарное отношение на множестве вершин K, которое специфицирует ориентированное ребро из вершины n в другую вершину m, если $(n,m) \in R$. Условие ацикличности направленного ациклического графа $(K,\ R)$ обеспечивается требованием нерефлексивности транзитивного замыкания R^+ отношения R, то есть $\forall k \in K\colon (k,k) \notin R^+$. Далее K называется множеством вершин направленного ациклического графа G, а R – множеством дуг графа G. $K = \{k_i\}_{i=1}^{count\ i} = Terminal \cup Nonterminal = \{k_{it}\}_{t=1}^{count\ t} \cup \{k_{in}\}_{n=1}^{count\ i-count\ t}$ – множество вершин направленного ациклического графа G состоит из множества $Terminal$ – множества терминальных вершин и множества $Nonterminal$ – множества нетерминальных вершин. Если $|K| = 1$, то направленный ациклический граф G вырождается в граф, состоящий из единственной терминальной вершины. Направленный ациклический граф может считаться обобщением леса деревьев.

Метаграфом DAG называется пара (G, μ_G), где G – направленный ациклический граф, μ_G – разметка графа G. Разметка μ_G отображает множество вершин и дуг графа во множество имен систем $\Theta = SystemNames = \{SystemName_i\}_{i=1}^{count\ SystemNames} \cup \{$Являться прямой надсистемой (подсистемой)$\}$. Разметка μ_G задаётся следующим образом. $\mu_G\colon K \to \Theta$ $(\mu_G\colon Terminal \cup Nonterminal \to SystemNames)$; $\mu_G\colon R \to \{$Являться прямой надсистемой (подсистемой)$\}$.

$\forall k \in K$, вершина k представляет определенный концепт, обозначающий некую систему из множества систем, имена которых содержатся во множестве Θ, а бинарное отношение R обозначает отношение «Являться прямой надсистемой (подсистемой)». Так, $\forall (n,m) \in R$, ориентированное ребро из вершины n в другую вершину m обозначает то, что n является *прямой надсистемой* для системы m (m, в свою очередь, является *прямой подсистемой* для системы n).

Метаграфом DAG системы S сложной структуры называется *метаграф DAG*, в котором $\exists!\ s \in K$, где s – вершина, полустепень захода которой равна 0. Вершина s называется корнем. Вершина s представляет концепт, обозначающий систему *S* сложной структуры. $\forall k \in K \backslash \{s\}$ – вершины k представляют собой подсистемы системы *S*.

NB: Определение метаграфа DAG системы *S* сложной структуры отличается от определения ориентированного дерева тем, что не накладывает требования о том, что полустепени захода всех вершин ≤ 1.

Имена систем из множества Θ должны быть различны и не могут иметь имени «Являться прямой надсистемой (подсистемой)».

Впредь метаграф DAG системы *S* сложной структуры называется **метаграфом системы *S***. Пример метаграфа системы (система D) с размеченными вершинами и дугами приведен на рисунке 1. Множество названий систем $\Theta = \{A, B, K, C, D\}$.

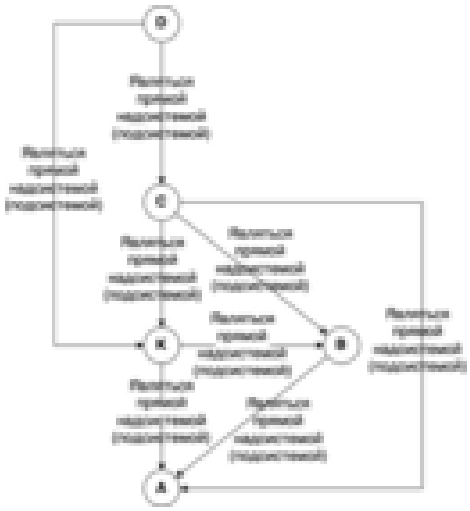

Рисунок 1 – Пример метаграфа системы D
Figure 1 – System D meta-graph example

Далее разметка дуг опускается, подразумеваясь существующей. Пример метаграфа системы сложной структуры «Химическая реакция» приведен на рисунке 2 (вместе со схемой автоматической генерации экземпляров шаблонов с метапонятиями, о которой говорится в следующих разделах). Концепты, обозначающие подсистемы, могут быть выделены произвольным образом. Необходимо учесть эту особенность в реализации.

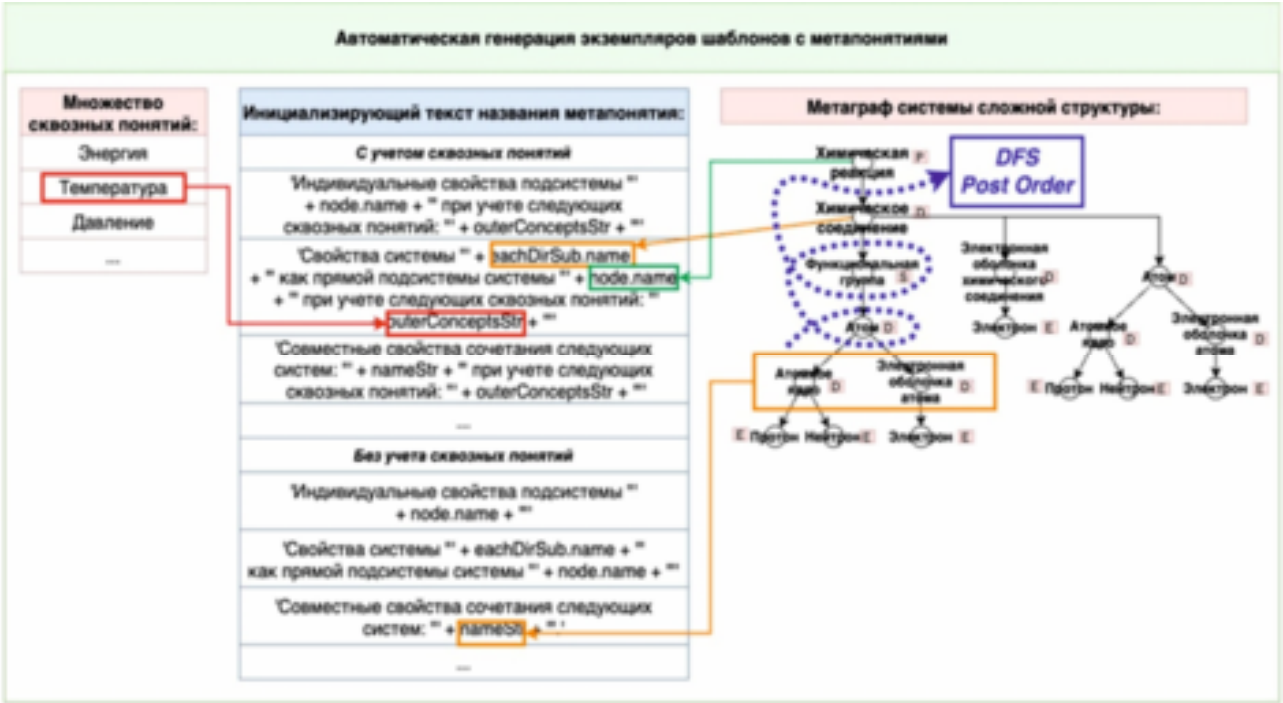

Рисунок 2 – Схема автоматической генерации экземпляров шаблонов с метапонятиями на основе метаграфа системы сложной структуры «Химческая реакция»
Figure 2 – The scheme of meta-concept template instance automatic generation based on "Chemical reaction" complex-structured system meta-graph

Процесс формирования системы понятий и базы знаний для предметной области с объектами сложной структуры и технология его автоматизации

На множестве систем (в соответствии с определением 1.1 системами могут быть объекты и процессы ПрО) может быть задан линейный порядок и определено множество максималь-

ных систем по высоте. Одна из максимальных систем выбирается в качестве образующей системы. Она становится корнем метаграфа. Системы, не являющиеся достижимыми из корня метаграфа, в процессе формирования системы понятий и БЗ участия не принимают.

Процесс получения системы понятий для системы [Название образующей системы] происходит за счет разработанных процедуры *исследования окрестности* каждой *вершины метаграфа системы* и *введения* специального *множества метапонятий*. Каждое метапонятие позволяет задать целый набор понятий, схема определения которых соответствует узлу конкретного метапонятия. Ключевые аспекты данных процедур представлены ниже.

Процедура исследования окрестности вершины метаграфа системы. Процедура исследования окрестности вершины метаграфа системы происходит в сознании человека неосознанно, но может быть формализована. Все достижимые из корня вершины метаграфа можно обойти одним из алгоритмов обхода деревьев. При этом окрестность каждой еще не исследованной вершины может быть исследована следующим образом.

Исследование окрестности вершины $node$ *метаграфа системы.* Процедура «ProcedureNodeNeighborhoodExamination», листинг которой представлен на рисунке 3 (совместно с вспомогательной функцией «outerConceptCombinList») с использованием Python-подобного синтаксиса, – это процедура обхода поддерева метаграфа из вершины $node$, $node \in K$, где K – множество вершин метаграфа. При исследовании окрестности вершины $node$ метаграфа системы, вершина $node$ становится корнем поддерева. Каждая отдельная вершина метаграфа представляет какую-то систему, обозначенную своим названием. Каждая система имеет высоту (по аналогии с высотой дерева) – натуральное число или 0 (если у системы нет прямых подсистем).

Корень поддерева – это та система, для которой строится часть системы понятий. Высота корня поддерева имеет важное значение для определения максимальной глубины обхода при исследовании окрестности этой вершины метаграфа:

1) если высота корня поддерева = 0, то максимальная глубина обхода = 0 (default);

2) если высота корня поддерева = 1, то максимальная глубина обхода = 1 («Medium»);

3) если высота корня поддерева ≥ 2, то максимальная глубина обхода ≥ 2 («Deep»).

Теоретически, если высота корня $node$ поддерева равна n, то максимальная глубина обхода при исследовании окрестности вершины $node$ равна n.

Образующая система (корень метаграфа) – это та система, для которой строится вся система понятий. Высота образующей системы имеет важное значение для определения максимальной глубины обхода при исследовании окрестностей любой из вершин метаграфа. Если высота образующей системы равна m, то максимальная глубина обхода при исследовании окрестности любой из вершин метаграфа $\leq m$.

Пусть p – выбранная экспертом ПрО глубина обхода любой вершины метаграфа системы сложной структуры. $p \in [0, m]$, где m – высота образующей системы. Тогда $\forall k \in K$: *Фактическая глубина обхода*$(k) = min(p, n)$, где K – множество вершин метаграфа, n – максимальная глубина обхода вершины k. В листинге рисунка 3 глубина обхода maxDepth выбирается пользователем-экспертом из следующих альтернатив: «Deep» соответствует 2, «Medium» – 1, дефолтный блок – 0 (операторы break не расставлены намеренно, предполагая генерацию метапонятий типов а) – е) до дефолтного блока включительно). При выборе пользователем глубины обхода «Deep», метапонятия для категорий ситуаций а) – е) рисунка 4 будут сгенерированы полностью. Типы метапонятий в листинге рисунка 3 обозначены следующим образом: «Metaconcept_type_d» – метапонятие типа d) без учета сквозных понятий, «Metaconcept_type_d_outerconcepts» – метапонятие типа d) с учетом сквозных понятий. Типы метапонятий соответствуют категориям ситуаций, возникающим при исследовании окрестности вершины метаграфа системы сложной структуры, пояснение которых приведено в следующем разделе.

```
def outerConceptCombinList(outerConcepts):
    """Возвращает список сочетаний сквозных понятий, включая 1-элементные
    Замечание: eachSubsysCombinList аналогична outerConceptCombinList"""
    import itertools
    resList = list()
    for i in range(1, (len(outerConcepts)+1)):
        resList.extend(list(itertools.combinations(outerConcepts, i)))
    return resList

def ProcedureNodeNeighborhoodExamination(maxDepth, node, outerConcepts=[]):
    """Процедура исследования окрестности вершины node метаграфа системы
    при заданной глубине обхода maxDepth"""
    switch(maxDepth):
        case 'Deep':
            for subsys in node.directSubsystems.all():
                for subSubsys in subsys.directSubsystems.all():
                    Gen(Metaconcept_type_d(subSubsys, subsys, node))
                    if outerConcepts:
                        for outerConceptCombin in outerConceptCombinList(outerConcepts):
                            Gen(Metaconcept_type_d_outerconcepts(subSubsys, subsys, node, outerConceptCombin))

                if len(set(subsys.directSubsystems.all())) >= 2:
                    for eachSubsysCombin in eachSubsysCombinList(set(subsys.directSubsystems.all())):
                        Gen(Metaconcept_type_e(eachSubsysCombin, subsys, node))
                        if outerConcepts:
                            for outerConceptCombin in outerConceptCombinList(outerConcepts):
                                Gen(Metaconcept_type_e_outerconcepts(eachSubsysCombin, subsys, node, outerConceptCombin))
        case 'Medium':
            for subsys in node.directSubsystems.all():
                Gen(Metaconcept_type_b(subsys, node))
                if outerConcepts:
                    for outerConceptCombin in outerConceptCombinList(outerConcepts):
                        Gen(Metaconcept_type_b_outerconcepts(subsys, node, outerConceptCombin))
        default:
            Gen(Metaconcept_type_a(node))
            if outerConcepts:
                for outerConceptCombin in outerConceptCombinList(outerConcepts):
                    Gen(Metaconcept_type_a_outerconcepts(node, outerConceptCombin))

            if len(set(node.directSubsystems.all())) >= 2:
                for eachSubsysCombin in eachSubsysCombinList(set(node.directSubsystems.all())):
                    Gen(Metaconcept_type_c(eachSubsysCombin))
                    if outerConcepts:
                        for outerConceptCombin in outerConceptCombinList(outerConcepts):
                            Gen(Metaconcept_type_c_outerconcepts(eachSubsysCombin, outerConceptCombin))
```

Рисунок 3 – Листинг процедуры исследования окрестности вершины метаграфа системы сложной структуры «ProcedureNodeNeighborhoodExamination»
Figure 3 – Listing of complex-structured system meta-graph node neighborhood examination procedure «ProcedureNodeNeighborhoodExamination»

Категории ситуаций, возникающие при исследовании окрестности вершины метаграфа системы сложной структуры. На рисунке 4 представлены категории ситуаций, возникающие при исследовании окрестности вершины метаграфа системы сложной структуры:

1) если фактическая глубина обхода вершины = 0 (default), то категории ситуаций, возникающие при исследовании окрестности данной вершины метаграфа системы сложной структуры: a) (см. рисунок 4);

2) если фактическая глубина обхода вершины = 1 («Medium»), то категории ситуаций, возникающие при исследовании окрестности данной вершины метаграфа системы сложной структуры: a), b), c) (см. рисунок 4);

3) если фактическая глубина обхода вершины = 2 («Deep»), то категории ситуаций, возникающие при исследовании окрестности данной вершины метаграфа системы сложной структуры: a) – i) (см. рисунок 4). При этом $\Delta height = 1$ в f) – i);

4) если фактическая глубина обхода вершины > 2, то категории ситуаций, возникающие при исследовании окрестности данной вершины метаграфа системы сложной структуры: a) – i) (см. рисунок 4). При этом в f) – i) при различных значениях $\Delta height$ количество категорий ситуаций может быть увеличено (решается в каждом случае целесообразностью введения дополнительных категорий).

Пояснение к категориям ситуаций, которые возникают при исследовании окрестности вершины метаграфа системы сложной структуры (см. рисунок 4):

– *категория a:* окрестность вершины метаграфа системы сложной структуры состоит только из нее самой;

– *категория b:* окрестность вершины метаграфа системы сложной структуры состоит из нее самой и одной ее прямой подсистемы;

– *категория c:* окрестность вершины метаграфа системы сложной структуры состоит из элементов всевозможных сочетаний ее прямых подсистем (за исключением наборов из одного элемента). За раз рассматривается одно из сочетаний;

– *категория d:* окрестность вершины метаграфа системы сложной структуры состоит из нее самой, одной из ее прямых подсистем и одной из прямых подсистем этой ее прямой подсистемы;

– *категория e:* окрестность вершины метаграфа системы сложной структуры состоит из нее самой, одной из ее прямых подсистем и из элементов всевозможных сочетаний прямых подсистем этой ее прямой подсистемы (за исключением наборов из одного элемента). За раз рассматривается одно из сочетаний;

– *категории f – g:* окрестность вершины метаграфа системы сложной структуры состоит из нее самой, элементов всевозможных сочетаний ее прямых подсистем (за исключением наборов из одного элемента), которые могут не иметь прямых подсистем, и элементов всевозможных сочетаний прямых подсистем-элементов выбранного сочетания с предыдущего уровня. Важно, что прямые подсистемы находятся на разных уровнях иерархии. За раз рассматривается одно из сочетаний;

– *категории h – i:* окрестность вершины метаграфа системы сложной структуры состоит из нее самой, элементов всевозможных сочетаний ее прямых подсистем (за исключением наборов из одного элемента), которые имеют прямые подсистемы, и элементов всевозможных сочетаний прямых подсистем, выбранных на предыдущем уровне. Прямые подсистемы должны находиться на одном уровне иерархии. За раз рассматривается одно из сочетаний.

Необходимо отметить, что категории ситуаций, возникающие при исследовании окрестности какой-либо вершины метаграфа системы сложной структуры, обозначают внутреннюю «онтологическую» структуру различных свойств системы, соответствующей исследуемой вершине метаграфа. Данная онтологическая структура группы свойств системы осознанно или не осознанно содержится в сознании носителей информации (в частности, носителей экспертных знаний) и часто имеет то или иное вербальное представление (зачастую, это неформальное словесное описание к названию самого свойства). Для того, чтобы дифференцировать категории ситуаций, возникающие при исследовании окрестности вершины метаграфа системы сложной структуры, необходимо рассмотреть их в связке с *множеством метапонятий метаграфа системы.* Обсуждение приводится далее.

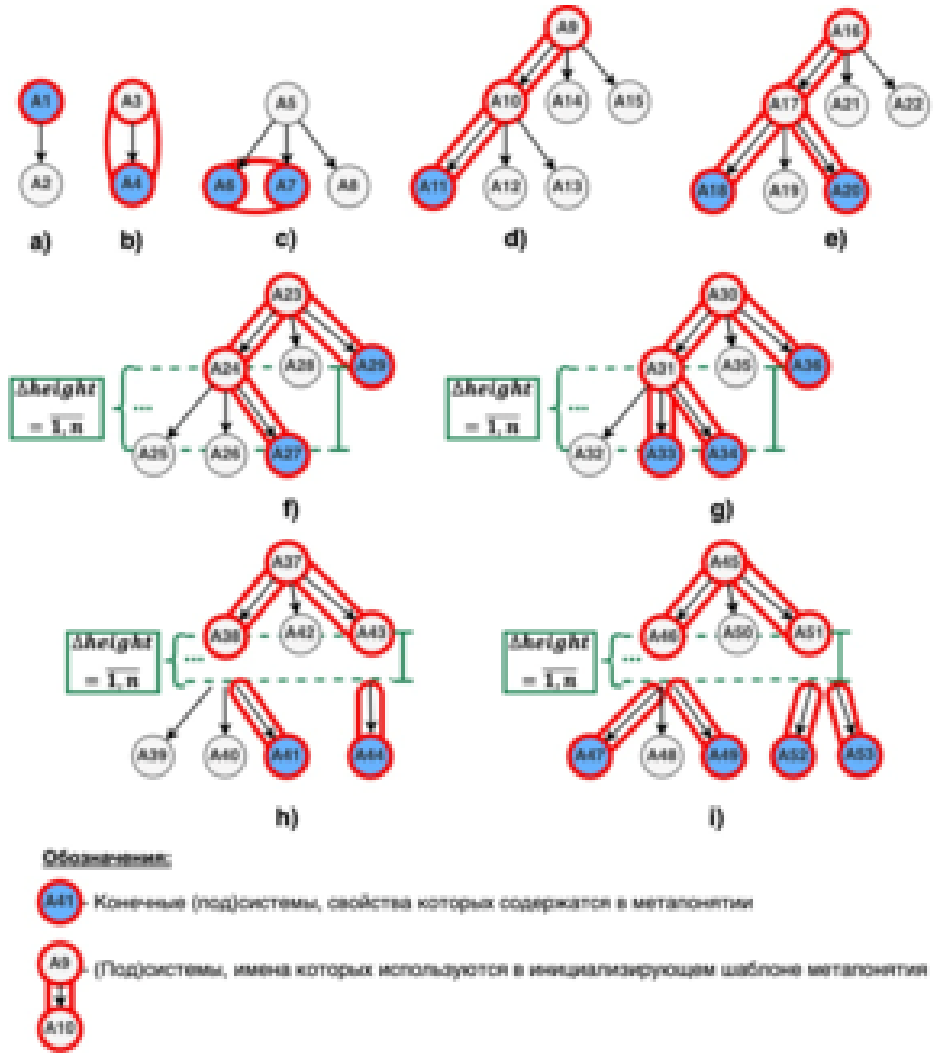

Рисунок 4 – Категории ситуаций, возникающие при исследовании окрестности вершины метаграфа системы сложной структуры
Figure 4 – Cases emerging from complex-structured system meta-graph node neighborhood examination procedure

Множество метапонятий метаграфа системы и их инициализирующие шаблоны. На рисунке 4 представлены категории ситуаций, возникающие при исследовании окрестности вершины метаграфа системы сложной структуры. Каждая из категорий а) – i) представляет «онтологию», или внутреннее устройство, группы свойств рассматриваемой системы (вершины-корня поддерева, из которого происходит обход поддерева).

Для категорий а) – е) введем множество метапонятий M и множество метапонятий с учётом сквозных понятий $MOut$, элементы которых репрезентированы инициализирующими шаблонами, которые представлены в таблице 1. Каждое метапонятие, принадлежащее $M \cup MOut$, помимо своего названия, характеризуется множеством систем узла и может характеризоваться множеством сквозных понятий (для метапонятий, принадлежащих $MOut$). Каждое метапонятие задает целый набор понятий, схема определения которых соответствует его узлу, определяя, таким образом, набор отношений экземпляров систем узла.

Для категорий ситуаций с) и е) важен факт того, что это сочетания. Количество метапонятий в одной из таких категорий (в категории е) при одной прямой надсистеме множества систем, участвующих в сочетаниях) равно $\sum_{k=2}^{n} \binom{n}{k}$, где n – число подсистем (например, на рисунках 4 с) и 4 е) представлены ситуации, при которых $n = 3$).

Для категорий ситуаций f) – i) действуют аналогичным образом. При этом, каждой категории ситуации f) – i) может соответствовать несколько метапонятий и соответствующих инициализирующих шаблонов при изменении параметра $\Delta height$ (см. рисунок 4).

Важно отметить, что метапонятия отражают внутреннюю структуру групп свойств системы, соответствующей исследуемой вершине метаграфа. Инициализирующий шаблон метапонятия представляет эту внутреннюю структуру. Однако, в сознании носителей информации он может иметь какую-то индивидуальную репрезентацию. Так, например, для категории d), если в метаграфе системы A9 = «Химическая реакция», A10 = «Химическое соединение», A11 = «Атом», то вместо инициализирующего шаблона метапонятия «Свойства системы Атом как прямой подсистемы системы Химическое соединение как прямой подсистемы системы Химическая реакция» носитель экспертных знаний может подразумевать группы свойств с названием «Свойства реакционного центра». Например, для категории c), если в метаграфе системы A6 = «Протон», A7 = «Нейтрон», носитель экспертных знаний может подразумевать группы свойств с названием «Свойства нуклонов». Пример сгенерированного экземпляра формы с метапонятием представлен на рисунке 5 (схема автоматической генерации – рисунок 2). Название метапонятия узла может быть переименовано.

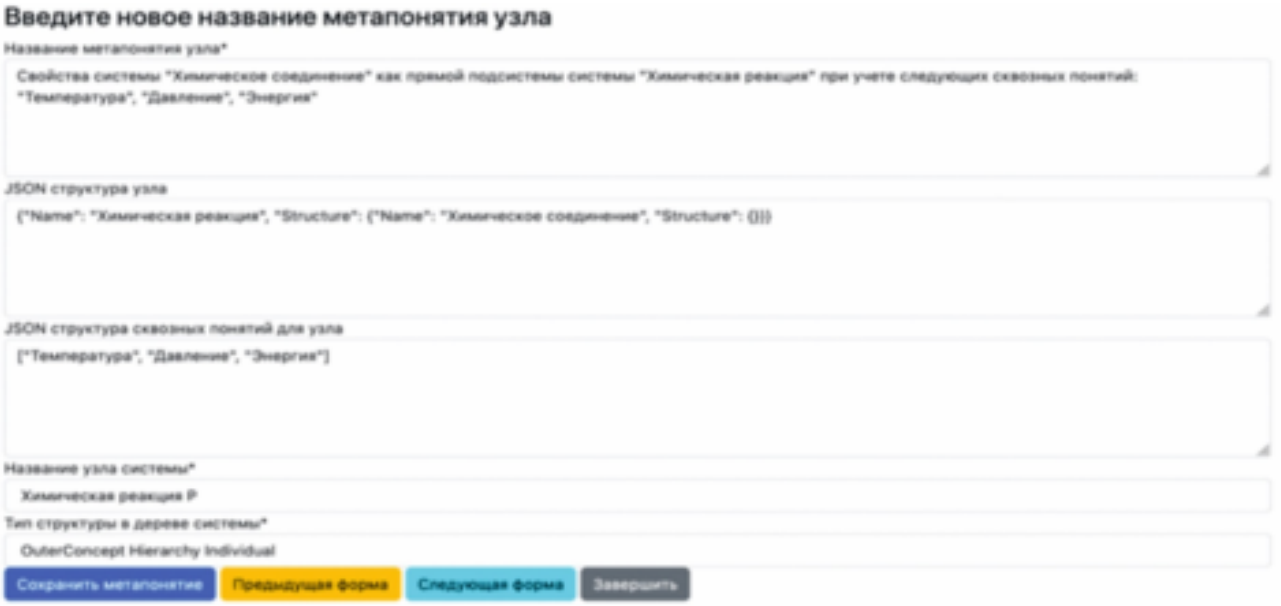

Рисунок 5 – Пример сгенерированного экземпляра формы с метапонятием
Figure 5 – Example of generated meta-concept form example

Функции и предикаты метапонятия. Метапонятие задает целый набор понятий, схема определения которых соответствует его узлу. Каждое метапонятие характеризуется названием, множеством систем узла (см. столбец 4 в таблице 1), а также, дополнительно, может характеризоваться множеством сквозных понятий. Каждое метапонятие определяет набор отношений экземпляров систем узла посредством задания набора функций и предикатов. Для каждого метапонятия также может быть задан набор “локальных” вспомогательных понятий, которые могут выступать формальными параметрами функций и предикатов только того метапонятия, для которого эти вспомогательные понятия заданы.

Таблица 1 – Метапонятия метаграфа системы и их инициализирующие шаблоны для различных категорий ситуаций, возникающих при исследовании окрестности вершины метаграфа системы сложной структуры

Table 1 – System meta-graph meta-concepts and their initialization templates for different cases emerging from complex-structured system meta-graph node neighborhood examination procedure

Название метапонятия				
Категория ситуации (рисунок 4)	Тип структуры в дереве (Названия типов структур выбраны произвольно и отражают, по мнению авторов, структуру обхода узла в дереве систем)	Конечные (под)системы, свойства которых содержатся в метапонятии	(Под)системы, имена которых используются в инициализирующем шаблоне метапонятия (множество систем узла)	Инициализирующий шаблон метапонятия M Инициализирующий шаблон метапонятия $MOut$
Метапонятия с индивидуальными свойствами (под)системы [без учета сквозных понятий | при учете сквозных понятий]				
a)	Individual	A1	A1	Индивидуальные свойства (под)системы A1
	OuterConcept Individual			Индивидуальные свойства (под)системы A1 при учете следующих сквозных понятий [Названия сквозных понятий]
Метапонятия с свойствами системы как прямой подсистемы системы [без учета сквозных понятий | при учете сквозных понятий]				
b)	Hierarchy Individual	A4	A4, A3	Свойства системы A4 как прямой подсистемы системы A3
	OuterConcept Hierarchy Individual			Свойства системы A4 как прямой подсистемы системы A3 при учете следующих сквозных понятий [Названия сквозных понятий]
Метапонятия с совместными свойствами сочетания систем [без учета сквозных понятий | при учете сквозных понятий]				
c)	Collective	A6, A7	A6, A7	Совместные свойства сочетания следующих систем A6, A7
	OuterConcept Collective			Совместные свойства сочетания следующих систем A6, A7 при учете следующих сквозных понятий [Названия сквозных понятий]
Метапонятия с свойствами системы как прямой подсистемы системы как прямой подсистемы системы [без учета сквозных понятий | при учете сквозных понятий]				
d)	Hierarchy Hierarchy	A11	A11, A10, A9	Свойства системы A11 как прямой подсистемы системы A10 как прямой подсистемы системы A9
	OuterConcept Hierarchy Hierarchy			Свойства системы A11 как прямой подсистемы системы A10 как прямой подсистемы системы A9 при учете следующих сквозных понятий [Названия сквозных понятий]
Метапонятия с совместными свойствами сочетания систем как прямых подсистем системы как прямой подсистемы системы [без учета сквозных понятий | при учете сквозных понятий]				
e)	Hierarchy Collective	A18, A20	A18, A20, A17, A16	Совместные свойства сочетания следующих систем A18, A20 – как прямых подсистем системы A17 как прямой подсистемы системы A16
	OuterConcept Hierarchy Collective			Совместные свойства сочетания следующих систем A18, A20 – как прямых подсистем системы A17 как прямой подсистемы системы A16 при учете следующих сквозных понятий [Названия сквозных понятий]

В связи с тем, что множество систем узла определяется метапонятием однозначно, часть формальных параметров данного набора функций и предикатов, а именно – формальные параметры, обозначающие системы узла метапонятия, – также определяются однозначно. Раз-

личие состоит лишь в количестве экземпляров каждой системы узла в качестве аргументов конкретной функции или предиката. Каждая система узла метапонятия может быть представлена ровно одним экземпляром в качестве фактических аргументов функции или предиката метапонятия, или количество экземпляров каждой из систем узла может быть натуральным числом, большим 1. Таким образом, в любой функции и предикате метапонятия часть обязательных аргументов – это экземпляры каждой из систем узла метапонятия (количество экземпляров каждой из систем узла обязано быть натуральным числом (здесь и далее натуральные числа – это $\mathbb{N}\backslash\{0\}$ (в отличие от ISO 80000-2)). Если для метапонятия задано множество сквозных понятий, то во всех функциях и предикатах метапонятия каждое сквозное понятие также является обязательным аргументом.

Необязательными аргументами функций и предикатов метапонятия могут быть локальные (заданные для данного метапонятия) и глобальные (заданные для всех метапонятий) вспомогательные понятия и стандартные значения. Стандартными значениями можно считать комплексные, действительные, целые значения, имена (скалярные значения) и логические значения. Количество экземпляров каждого из необязательных аргументов обязано быть натуральным числом. Типы и количества компонентов возвращаемого значения функции метапонятия задаются экспертом ПрО отдельно.

На рисунке 6 представлен пример пользовательского интерфейса при конструировании экспертом функции с названием «Продукты химической реакции при _температуре через _часов при _реагентах» на основе метапонятия «Свойства системы «Химическое соединение» как прямой подсистемы системы «Химическая реакция» при учете следующих сквозных понятий «Температура», «Время»» (переименование метапонятия не производилось). Типы аргументов, соответствующие сквозным понятиям (аргументы номер 1, 2), подставляются автоматически. У типов аргументов, соответствующих системам (аргументы номер 3, 4), эксперту необходимо выбрать, будет ли аргументом множество значений (аргумент номер 3) или одно значение (аргумент номер 4). В качестве возвращаемого значения функции экспертом выбран один компонент возвращаемого значения – множество значений, соответствующих химическим соединениям (продукты реакции).

После указания экспертом схемы определения функции на основе сгенерированного метапонятия, в БЗ будет сгенерирована форма, приведенная на рисунке 7, которая становится доступной для заполнения для того же эксперта, который выбрал схему определения, или для другого эксперта, который входит в группу проекта создания БЗ для данной ПрО.

Обсуждение полученных результатов

Данная технология предоставляет возможность автоматически сгенерировать исчерпывающий набор метапонятий (соответствующим промежуточным понятиям онтологии), которые позволяют эксперту ПрО систематическим образом исследовать набор отношений экземпляров систем узла, задавая при этом схемы определения функций и предикатов. Часть формальных параметров данных функций и предикатов генерируются автоматически по метапонятию, другая часть формальных параметров и типы значений компонентов возвращаемых значений функций – выбираются экспертом из конечного множества альтернатив. Автоматическая генерация метапонятий становится возможной благодаря построению метаграфа системы для ПрО выделенного класса.

Наиболее близкой технологией является моделирование сложно-структурированных ПрО с помощью необогащенных систем логических соотношений (НСЛС) O^m ($m \geq 2$), каждая из которых является модулем модели онтологии уровня m, и процедуры обогащения НСЛС, позволяющих строить многоуровневые системы понятий [8]. $O^m = (F^m, P^m, C^m)$, где F^m – множество определений понятий уровня m, ограничений на множества их значений и связи между значениями понятий, P^m – множество параметров уровня m, C^m – множество определений конструкторов уровня m. Множества P^m и C^m могут быть пустыми. Если $m = 2$, то $C^2 = \emptyset$. Переход от модели онтологии уровня m к модели онтологии (при $m \geq 3$)

или модели знаний (при $m = 2$) производится заданием обогащения системы O^m. Задание онтологии уровня $m - 1$ в терминах онтологии уровня m предполагает задание обогащения необогащенной системы логических соотношений O^m, которое является пятеркой вида $k^m = (\alpha_P^m, EN^m, ER^m, EC^m, EP^m)$, где α_P^m – интерпретация параметров уровня m, EN^m – множество определений понятий уровня $m - 1$, ER^m – множество связей между понятиями уровня $m - 1$, EC^m – множество определений конструкторов уровня $m - 1$, EP^m – множество параметров уровня $m - 1$. Множества EN^m, ER^m, EC^m, EP^m могут быть пустыми, но $EN^m \cup ER^m \cup EC^m \cup EP^m \neq \emptyset$.

Рисунок 6 – Пользовательский интерфейс при создании химиком-экспертом ПрО функции «Продукты химической реакции при _температуре через _часов при _реагентах»
Figure 6 – Intelligent system look during the function «Chemical reaction products at _temperature after _hours at _reactants» creation process performed by chemist-expert

Однако, предложенные в [8] сценарии диалога с экспертом предполагают только ручной выбор формальных параметров задаваемых функций и предикатов, промежуточные понятия требуют обязательного переименования и в большей степени зависят не от онтологии самого объекта (системы) сложной структуры, а от изначально хорошо подобранного экспертом набора типов сущностей, которые становятся аргументами конструкторов для создания с их помощью наборов промежуточных понятий онтологии (на разных уровнях). Так, НСЛС O^4 имеет максимальный уровень и определяет такие абстрактные термины, как, например, «Типы компонентов сущности», «Подмножества компонентов сущности» (как параметры четвертого уровня) и «Свойства подмножества компонентов указанного типа», «Свойства компонентов, задаваемых количеством», «Общие свойства процесса, участвующей в нем сущности и ее компонента» (как конструкторы четвертого уровня). Приводя некоторые примеры, можно упомянуть, что при задании обогащения O^4 для ПрО химии (для $O_{\text{ф}}^3$ – модели онтологии физической химии уровня 3 и $O_{\text{о}}^3$ – модели онтологии органической химии уровня 3) «Типами сущностей», созданными экспертом, становятся «Табличные значения температуры», «Табличные значения давления», «Химические реакции», «Химические вещества», «Химические элементы», что, в свою очередь, позволяет эксперту задать посредством ис-

пользования конструкторов такие промежуточные термины, как «Зависящие от температуры свойства простых веществ» (аргументами конструктора «Совместные свойства сущностей», выбранными экспертом, становятся сущности «Химические вещества» и «Табличные значения температуры»). В реализации данного сценария диалога при определении функций «Зависящие от температуры свойства простых веществ» эксперту заново приходится выбирать весь набор аргументов. При использовании разработанной технологии они были бы сгенерированы автоматически.

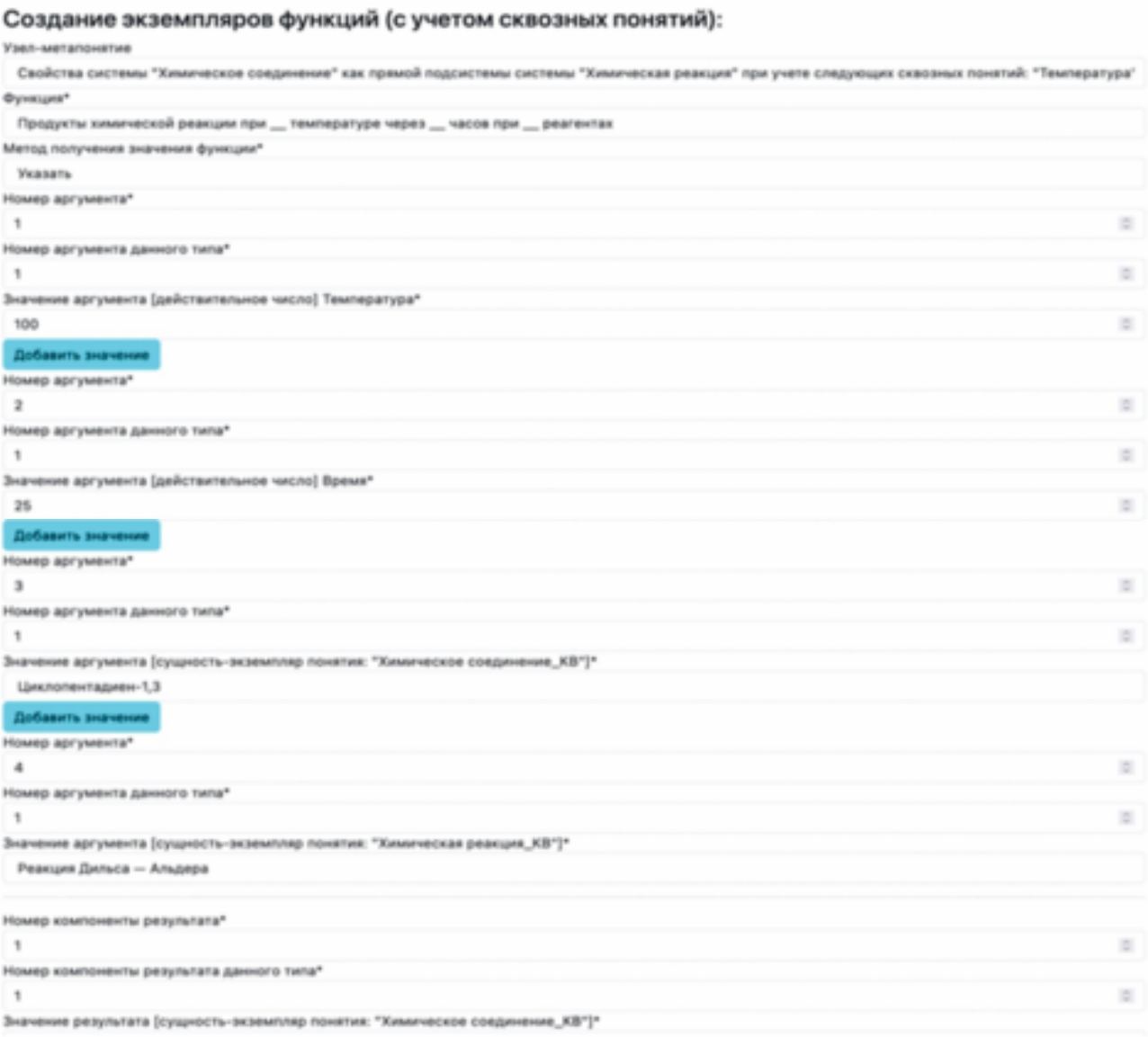

Рисунок 7 – Часть пользовательского интерфейса при создании химиком-экспертом ПрО экземпляров функции «Продукты химической реакции при _ температуре через _ часов при _ реагентах» в БЗ

Figure 7 – Part of the intelligent system look during the function «Reaction products at _temperature after _hours at _reactants» creation process performed by chemist-expert in the knowledge base

Перспективным решением для класса ПрО, объекты которых являются системами сложной структуры, представляется разработка гибридного редактора, основанного на метаонтологии и позволяющего автоматически генерировать метапонятия. Это позволит эксперту систематическим образом сформировать многоуровневую систему понятий для БЗ о системе сложной структуры.

Другой близкой технологией является двухуровневая модель метаинформации и целевой информации и предметно-независимая платформа IACPaaS для создания интеллектуальных

сервисов [3-5]. Однако, автоматическая генерация метапонятий (промежуточных понятий) не поддерживается на уровне редактора, а требует участия интеллектуальных агентов, которые бы по метаграфу системы сложной структуры генерировали участки подсети. Для имплементации данной технологии эксперту потребуется участие программиста (для написания на Java программного кода для шаблонов сообщений, агентов).

Заключение

В статье предложена технология автоматизации процесса формирования систем понятий и БЗ для ПрО с объектами сложной структуры. В связи с тем, что эксперты имеют представление о системе сложной структуры в форме метаграфа, который неявно принимает участие в процессе формирования системы понятий ПрО, данный процесс может быть формализован. Автоматизация процесса получения системы понятий происходит за счёт разработанных процедуры исследования окрестности каждой вершины метаграфа системы и введения специального множества метапонятий. Метапонятия отражают внутреннюю структуру групп свойств системы и позволяют эксперту ПрО исследовать данные свойства систематическим образом. Часть типов аргументов задаваемых экспертом функций и предикатов БЗ генерируется на основании метапонятия автоматически, а часть – выбирается из конечного множества альтернатив. Разработанная технология может быть использована в разработке редактора БЗ для ПрО с объектами сложной структуры.

Результаты, изложенные в данной статье, получены в рамках проекта РФФИ № 19-37-90137.

Библиографический список

1. **Колодько Д. А.** Меронимические отношения как проявление системности лексики // Вестник Самарского университета. История, педагогика, филология. 2016. № 3.2(22). С. 270-275.

2. **Голенков В. В., Гулякина Н. А.** Проект открытой семантической технологии компонентного проектирования интеллектуальных систем. Часть 1: Принципы создания // Онтология проектирования. 2014. №1(11). С. 42-64.

3. **Грибова В. В., Клещёв А. С., Шалфеева Е. А.** Управление интеллектуальными системами // Известия РАН. Теория и системы управления. 2010. № 6. С. 122-137.

4. **Грибова В. В., Клещёв А. С., Москаленко Ф. М., Тимченко В. А., Шалфеева Е. А., Федорищев Л. А.** Методы и средства разработки жизнеспособных интеллектуальных сервисов // Вестник ДВО РАН. Владивосток: Изд. «Дальнаука». 2016. № 4. С. 133-141.

5. **Клещёв А. С., Орлов В. А.** Компьютерные банки знаний. Универсальный подход к решению проблемы редактирования информации // Информационные технологии. 2006. № 5. С. 25-31.

6. **Загорулько Ю. А., Боровикова О. И.** Технология построения онтологий для порталов научных знаний. Вестник НГУ. Серия: Информационные технологии. 2007. Т. 5. № 2. С. 42-52.

7. **Musen M.** The Protégé Project: A Look Back and a Look Forward // AI Matters. 2015, vol. 1(4), pp. 4-12.

8. **Artemieva, I. L.** Ontology development for domains with complicated structures. In: Wolff, K. E., Palchunov, D. E., Zagoruiko, N. G., Andelfinger, U. (eds.) Knowledge Processing and Data Analysis. KPP 2007, KONT 2007. Lecture Notes in Computer Science. Springer, Berlin, Heidelberg. 2011, vol. 6581, pp. 759-767.

9. **Kleshchev A., Artemjeva I.** A mathematical apparatus for domain ontology simulation. An extendable language of applied logic // International Journal «Information Theories & Applications». 2005, no. 12 (2), pp.149-157.

10. **Клещёв А. С., Шалфеева Е. А.** Онтология задач интеллектуальной деятельности // Онтология проектирования. 2015. Т. 5, № 2. С. 179-205.

11. ГОСТ Р ИСО 9000-2001 Системы менеджмента качества. Основные положения и словарь. ИПК Издательство стандартов. 2001. 68 с.

12. **Fiore M., Campos M.D.** The Algebra of Directed Acyclic Graphs. Computer Laboratory. University of Cambridge. [Электронный ресурс]. Режим доступа: https://www.cl.cam.ac.uk/~mpf23/ papers/Algebra/dags.pdf. – (Дата обращения: 15.05.2023).

UDC 004.89

TECHNOLOGY OF AUTOMATED CONCEPT SYSTEM AND KNOWLEDGE BASE FORMATION PROCESS FOR APPLICATION DOMAINS WITH COMPLEX-STRUCTURED OBJECTS

K. A. Gulyaeva, lecturer, Department of Software Engineering and Artificial Intelligence, FEFU, Vladivostok, Russia;
orcid.org/0000-0003-0226-5072, e-mail: karinagulyaeva8@gmail.com
I. L. Artemieva, Dr. Sc. (Tech.), full professor, Department of Software Engineering and Artificial Intelligence, FEFU, Vladivostok, Russia;
orcid.org/0000-0003-2088-5259, e-mail: artemeva.il@dvfu.ru

The problem of knowledge base construction that are based on application domain ontologies is studied. ***The aim is*** *to create the technology of automated concept system and knowledge base formation process for application domains with complex-structured objects. The application domain class the objects of which are systems with complex structures is defined. The process of system metagraph creation is formalized. The automation of concept system elicitation process happens with the help of the developed system meta-graph vertex neighborhood examination procedure and meta-concept set introduction. Case categories emerging during system meta-graph vertex neighborhood examination procedure are derived. Initialization templates that depict inner structure of system property groups are defined. The technology allows the expert to generate intermediate ontology concepts in a systematic manner. Part of argument types of knowledge base functions and predicates created by expert is generated automatically based on metaconcept. Other argument types are chosen from a finite set of alternatives. The developed technology can be utilized at knowledge base editor creation phase for application domains with complex-structured objects. These design solutions help to obtain knowledge base schema design technologies of practical use.*

Keywords: *knowledge base, intelligent system, metagraph, ontology, application domain, complex-structured system, meronym system, theory.*

DOI: 10.21667/1995-4565-2023-85-65-81

References

1. **Kolodko D. A.** Meronimicheskie otnoshenija kak projavlenie sistemnosti leksiki. *Vestnik Samarskogo univer-siteta. Istorija, pedagogika, filologija.* 2016. no. 3.2(22). pp. 270-275 (in Russian).

2. **Golenkov V. V., Gulyakina N. A.** Proekt otkritoi semanticheskoi tekhnologii komponentno-go proektirovaniya intellektualnikh sistem. Chast 1: Printsipi sozdaniya. *Ontologiya proektirovaniya*, 2014, no. 1(11), pp. 42-64 (in Russian).

3. **Gribova V. V., Kleshchyov A. S., Shalfeeva Ye. A.** Upravlenie intellektualnimi sistemami. *Izvestiya RAN. Teoriya i sistemi upravleniya,* 2010, no. 6, pp. 122-137 (in Russian).

4. **Gribova V. V., Kleshchyov A. S., Moskalenko F. M., Timchenko V. A., Shalfeeva Ye. A., Fedorishchev L. A.** Metodi i sredstva razrabotki zhiznesposobnikh intellektualnikh servisov. *Vestnik DVO RAN* Vladivostok: Izd. «Dalnauka», 2016, no. 4, pp. 133-141 (in Russian).

5. **Kleshchyov A.S., Orlov V. A.** Kompyuternie banki znanii. Universalnii podkhod k resheniyu problemi redaktirovaniya informatsii. *Informatsionnie tekhnologii,* 2006, no. 5, pp. 25-31 (in Russian).

6. **Zagorulko Yu. A., Borovikova O. I.** Tekhnologiya postroeniya ontologii dlya portalov nauchnikh znanii. *Vestnik NGU. Seriya: Informatsionnie tekhnologii,* 2007, vol.5, no. 2, pp. 42-52 (in Russian).

7. **Musen M.** The Protégé Project: A Look Back and a Look Forward. *AI Matters,* 2015, vol. 1(4), pp.4-12.

8. **Artemieva, I. L.** Ontology development for domains with complicated structures. In: Wolff, K. E., Palchunov, D. E., Zagoruiko, N. G., Andelfinger, U. (eds.) *Knowledge Processing and Data Analysis. KPP 2007, KONT 2007. Lecture Notes in Computer Science.* Springer, Berlin, Heidelberg, 2011, vol. 6581, pp. 759-767.

9. **Kleshchev A., Artemjeva I.** A mathematical apparatus for domain ontology simulation. An extendable language of applied logic. *International Journal «Information Theories & Applications»*, 2005, no. 12 (2), pp. 149-157.

10. **Kleshchyov A. S., Shalfeeva Ye. A.** Ontologiya zadach intellektualnoi deyatelnosti. *Ontologiya proektirovaniya,* 2015, vol. 5, no. 2, pp. 179-205 (in Russian).

11. GOST R ISO 9000-2001 Sistemi menedzhmenta kachestva. Osnovnie polozheniya i slovar. IPK Izdatelstvo standartov, 2001. 68 p. (in Russian).

12. **Fiore M., Campos M. D.** The Algebra of Directed Acyclic Graphs. Computer Laboratory. University of Cambridge. URL: https://www.cl.cam.ac.uk/~mpf23/papers/Algebra/dags.pdf.