



на производительность схемы шифрования и ее устойчивость к основным видам криптографических атак. Подход к анализу криптостойкости ал-

горитма шифрования, которого придерживаются авторы, применим для анализа криптостойкости всех алгоритмов данного класса.

СПИСОК ЛИТЕРАТУРЫ

1. **Богач, Н.В.** Проблема малого количества ключей в алгоритме шифрования двумерных данных на основе tent-отображения [Текст] / Н.В. Богач, В.А. Чупров, В.Н. Шашихин // Научно-технические ведомости СПбГПУ. Сер. Информатика. Телекоммуникации. Управление. –2012. –№ 2.
2. **Кузнецов, С.П.** Динамический хаос [Текст] / С.П. Кузнецов. –М.: Физматлит, 2006. –356 с.
3. **Алферов, А.П.** Основы криптографии: Учеб. пособие [Текст] / А.П. Алферов, А.Ю. Зубов, А.С. Кузьмин [и др.]. –М.: Гелиос АРВ, 2002. –480 с.
4. **Птицын, Н.** Приложение теории детерминированного хаоса в криптографии [Текст] / Н. Птицын. –Изд-во МГТУ им. Баумана, 2002. –80 с.
5. **Андреев, Ю.В.** Использование динамического хаоса в коммуникационных системах и компьютерных сетях [Текст] / Ю.В. Андреев, А.М. Балабин, А.А. Дмитриев [и др.]. –М.: Препринт ИРЭ РАН, 2000. –№ 2 (626). –76 с.
6. **Alvarez, G.** Some basic cryptographic requirements for chaos-based cryptosystems [Text] / G. Alvarez, S. Li // International J. of Bifurcation and Chaos. –2006. –Vol. 16 (8). –P. 2129–2151.
7. **Habutsu, T.** A Secret Key Cryptosystem by Iterating a Chaotic Map [Text] / T. Habutsu, Y. Nishio, I. Sasase [et al.] // Advances in Cryptology. EUROCRYPT'91; ed. D.W. Davies. –LNCS 547. –Berlin: Springer-Verlag, 1991. –127 p.
8. **Biham, E.** Cryptanalysis of the Chaotic-Map Cryptosystem Suggested at EUROCRYPT'91 [Text] / E. Biham // Advances in Cryptology. EUROCRYPT'91; ed. D.W. Davies. –LNCS 547. – Berlin: Springer-Verlag, 1991. –532 p.
9. **Deffeyes, K.S.** Encryption System and Method [Text] / K.S. Deffeyes. –US Patent № 5,001,754. –Mar. 19. 1991.
10. **Bernstein, G.M.** Method and Apparatus for Generating Secure Random Numbers Using Chaos [Text] / G.M. Bernstein, M.A. Lieberman. –US Parent № 5,007,087. –Apr. 9. 1991.
11. **Fridrich, J.** Symmetric Ciphers Based on Two-dimensional Chaotic Maps [Text] / J. Fridrich // Int. J. Bifurcation and Chaos. –1998. –Vol. 8. –№ 6. –1259 p.
12. **Дмитриев, А.С.** Динамический хаос: новые носители информации для систем связи [Текст] / А.С. Дмитриев, А.И. Панас. –М.: Физматлит, 2002. –252 с.

УДК 007:681.512.2

Л.Ю. Григорьев, А.А. Заблоцкий, Д.В. Кудрявцев

ТЕХНОЛОГИЯ НАПОЛНЕНИЯ БАЗ ЗНАНИЙ ОНТОЛОГИЧЕСКОГО ТИПА

В качестве основы баз знаний в настоящее время выступают онтологии [1–3]. Для представления онтологий в базах знаний используются специальные языки: RDFS [4], OWL [5]. Указанные языки эффективны для решения задач автоматизированной интеграции информации, обмена знаниями и выполнения логического вывода. Проблема существующих языков и методов онтологического инжиниринга заключается в том, что основной акцент в них сделан на задаче проектирования онтологии, а важность формата представления знаний для наполнения онтологии экземплярами недооценивается и отдается на откуп разработчикам редакторов онтологий и баз знаний. В результате этого при разработке масштабных баз

знаний (более 1000 элементов) в существующих инструментах возникают сложности при наполнении онтологий: экспертам в предметной области, не являющимся специалистами в семантических технологиях, сложно вводить информацию в базу знаний, а также сопровождать ее.

В статье предлагается язык классификаторов и проекций для формирования баз знаний онтологического типа, в котором в явном виде задается формат для визуальной работы с экземплярами онтологии, их атрибутами и отношениями.

Обзор методов и программных средств для ручного наполнения онтологий

Для ручного наполнения онтологии могут

использоваться RDF-редакторы, универсальные и специализированные редакторы онтологий, предметно-ориентированный подход к моделированию (DSM), а также семантические вики.

RDF-редакторы. Инструменты, называемые «RDF-редакторами», имеют довольно разнообразную функциональность. Некоторые просто предоставляют пользователям экранные формы для создания RDF-файла – они освобождают пользователя от необходимости вводить синтаксические конструкции и предоставляют несложные возможности автозаполнения (например, [6]). Некоторые инструменты (например, [7]) представляют визуальный интерфейс, позволяющий пользователю увидеть в развернутом виде соответствующий RDF-граф. Система Tabulator [8] является уникальной в данном классе тем, что работает в распределенном режиме. Вместо создания одного локального RDF-файла, Tabulator позволяет пользователям видеть и редактировать распределенную семантическую паутину прямо на рабочем месте. RDF-утверждения, полученные из нескольких источников, представляются пользователю так, как будто они составляют единый согласованный граф. Редактирование такого графа приводит к тому, что Tabulator вносит изменения в более чем одно RDF-хранилище.

Универсальные редакторы онтологий. Приложения, такие, как Protege (<http://protege.stanford.edu/>) and NeOn (<http://neon-toolkit.org/>) также позволяют пользователям совместно создавать знания, однако акцент в данных средствах сделан на разработке схемы, а не на вводе фактов/данных: онтология вместо ее экземпляров. Они предоставляют понятный инструментарий для установления сложных взаимосвязей и ограничений между классами. Недостатком универсальных редакторов в контексте задачи ввода экземпляров онтологии является их сложность для неспециалистов в области семантического веба (экспертов в предметной области). Кроме того, сам механизм ввода экземпляров онтологии, основанный на использовании форм, не всегда эффективен.

Предметно-ориентированный подход к моделированию (Domain-Specific Modeling, DSM). При визуализации и наполнении онтологий можно использовать DSM-подход, широко распространенный при разработке средств программирования [9]. Подход подразумевает быстрое создание визуального языка и соответствующего графического редактора. Обзор сред быстрой раз-

работки графических редакторов можно найти здесь [10, 11]. Достоинство этого подхода в том, что можно создать свой собственный визуальный язык, понятный и удобный для неспециалистов в области семантических технологий. Ограничением данного подхода можно считать сложность работы с масштабными моделями.

Специализированные редакторы. Специализированные приложения для наполнения онтологий обеспечивают максимальные возможности по адаптации инструмента под заданную онтологию и требования пользователя. Основное преимущество использования специализированных редакторов – корректность вводимых данных, поскольку разработчики данных систем могут предоставить конечным пользователям интуитивно понятный минимальный необходимый и достаточный функционал для решения требуемых задач, снижающий вероятность ошибок ввода данных.

Недостатком данного подхода являются затраты, связанные с разработкой и сопровождением соответствующих специализированных приложений. Также данный подход накладывает временные ограничения (разработка редактора не может начаться, пока не будет готова онтология) и предполагает необходимость обновления системы в случае изменения онтологии.

Семантические вики. Семантические вики объединяют свойства вики-систем с семантическими утверждениями. В процессе редактирования веб-страницы пользователь может добавить семантическое описание содержимого с помощью RDF-совместимых утверждений. Это может быть реализовано либо через расширение разметки [12], либо через отдельный интерфейсный механизм [13]. Использование семантических вики основано на предпосылке о том, что пользователь, в основном, работает с естественным языком и хочет дополнить неструктурированный текст семантическими данными.

Большинство из этих подходов предоставляют визуальный пользовательский интерфейс для ввода и редактирования экземпляров классов и их свойств. Однако до сих пор мало внимания уделяется удобству предлагаемого интерфейса. Наиболее распространенный подход – использование форм для ввода данных – не всегда удобен: за множеством форм, позволяющих представить свойства отдельной моделируемой сущности, сложно видеть целостную систему связей в онтологической модели (базе знаний). Хотя существу-



Схема соответствия элементов языка классификаторов и проекций компонентам языка OWL

| Элементы предложенного языка | Элементы OWL |
|---|--|
| TYPES | |
| e-type _i + sign | owl:Class + rdfs:label(пиктограмма) |
| pt-type _k + sign | owl:DatatypeProperty + rdfs:label(пиктограмма) |
| rel-type _j + sign | owl:ObjectProperty + rdfs:label(пиктограмма) |
| («является», e-type _{i1} , e-type _{i2}) | rdfs:subClassOf (owl:Class, owl:Class) rdfs:subPropertyOf |
| CL | |
| Onto-CL | |
| type _{Cl_i} | owl:Class ИЛИ owl:DatatypeProperty ИЛИ owl:ObjectProperty |
| {(type _{Cl_i} , pt-type _k , pt-type ^{VAL} _k })} | owl:DatatypeProperty (rdfs:domain, rdfs:range) |
| {(rel-type _j , type _{Cl_{i1}} , type _{Cl_{i2}} }) | owl:ObjectProperty (rdfs:domain, rdfs:range) граф, создаваемый на основе указанной связи должен быть деревом |
| Base-CL | |
| {E _{Cl_i} } – элементы классификатора | rdf:ID (Индивиды) ИЛИ owl:DatatypeProperty ИЛИ owl:ObjectProperty |
| («имеет тип», E _{Cl_i} , e-type _i) | rdf:type(E _i , C _{0j}) |
| (rel-type _j , E _{Cl_{i1}} , E _{Cl_{i2}}) | owl:ObjectProperty (E ₁ , E ₂) получаемый граф, должен быть деревом |
| PR | |
| Onto-PR | |
| (rel-type _j , type _{Cl_{i1}} , type _{Cl_{i2}}) | owl:ObjectProperty (rdfs:domain, rdfs:range) Иногда: owl:DatatypeProperty (rdfs:domain, rdfs:range) |
| Base-PR | |
| (rel-type _j , E _{Cl_{i1}} , E _{Cl_{i2}}) | owl:ObjectProperty (E ₁ , E ₂) Иногда: owl:DatatypeProperty (E ₁ , E ₂) |

ют некоторые решения, частично устраниющие данную проблему (например, InstanceTree плагин¹ для Protégé обеспечивает древовидную навигацию по фреймам, на которые прямо или опосредованно ссылается определенный экземпляр класса), они требуют более развитых средств для работы с неиерархическими связями.

Описание языка классификаторов и проекций для наполнения баз знаний онтологического типа

Язык классификаторов и проекций предназначен, в первую очередь, для наполнения онтологической базы знаний [14]. Он интегрирует иерархические списки с табличными представлениями (связи между несколькими иерархически-

ми списками) и визуальными (DSM) методами (экземпляры разных классов онтологии имеют уникальные пиктограммы).

Язык классификаторов и проекций содержит следующие элементы:

L ::= <TYPES, CL, PR, TASKS, SPEC>, где

TYPES – описание типов, позволяющее представить таксономию верхнеуровневых классов, типы связей между классами, значения свойств классов. Каждому типу ставится в соответствие пиктограмма (графический знак);

CL – классификатор, формат ввода нижнеуровневых классов и экземпляров онтологии, их свойств и иерархических связей между ними (основные типы связей: «класс-подкласс», «часть-целое», «подчиняется»; CL= {CL_i^{name}, Onto-CL_i, Base-CL_i} – классификатор, где CL_i^{name} – имя классификатора; Onto-CL_i – свойства класси-

¹ <http://protegewiki.stanford.edu/wiki/InstanceTree>

фикатора – часть классификатора, задающая состав типов элементов (в т. ч. визуально), перечисляемых в классификаторе, перечень их свойств с областями допустимых значений и типы иерархических связей между элементами; Base-CL_i – содержание классификатора – часть классификатора, в которой перечисляются и типизируются моделируемые нижнеуровневые классы, значения их свойств и задается древовидная система связей между классами);

PR – проекция, формат ввода связей между экземплярами онтологии, перечисленными в классификаторах (примеры типов связей: «выполняет», «обеспечивает достижение», «отвечает за»; PR_j = {PR_j^{name}, Onto-PR_j, Base-PR_j} – проекция, где PR_j^{name} – наименование проекции; Onto-PR_j – свойства проекции – часть проекции, определяющая состав связей между типами, перечисляемых в классификаторе элементов; Base-PR_j – содержание проекции – часть проекции, содержащая множество связей между элементами, связываемых классификаторов);

TASKS – формат спецификации задач по разработке и использованию онтологии (не рассматривается в данной статье);

SPEC – формат спецификации запросов к онтологической модели (не рассматривается в настоящей статье).

Данный язык используется для структурирования и представления знаний, а также для формирования запросов к онтологической модели.

В таблице представлена схема соответствия элементов языка классификаторов и проекций компонентам языка OWL.

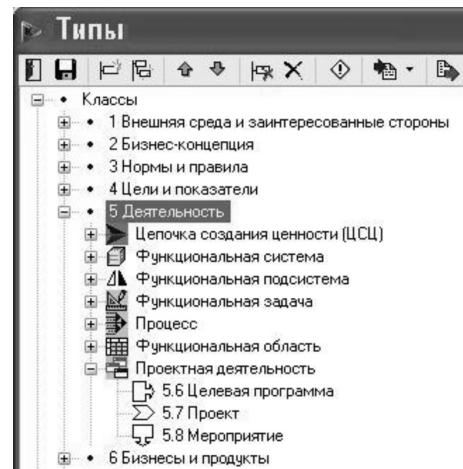


Рис. 1. Типы (TYPES) для разработки и визуализации верхнеуровневых классов

Программная реализация языка классификаторов и проекций в системе ОРГ-Мастер

Язык визуального проектирования и наполнения баз знаний реализован в системе ОРГ-Мастер [15]. На рис. 1 представлен фрагмент описания типов моделируемых в системе объектов.

Источником типов может быть готовая онтология, созданная в каком-либо стандартном редакторе онтологий.

При создании нижнеуровневой онтологии (левое окно на рис. 2) происходит типизация элементов с помощью верхнеуровневой онтологии (правое окно). С точки зрения механизма визуализации, устанавливается связь между элементами двух иерархических списков.

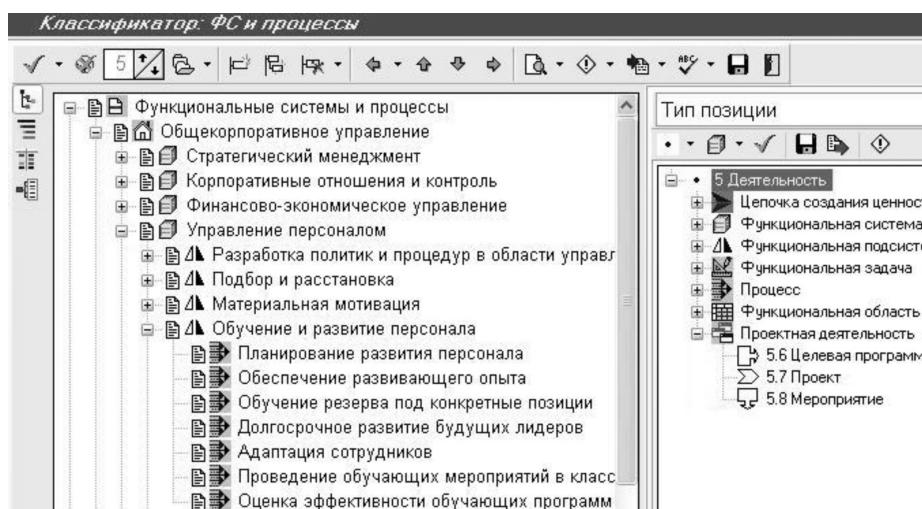


Рис. 2. Пример классификатора (CL) бизнес-процессов предприятия

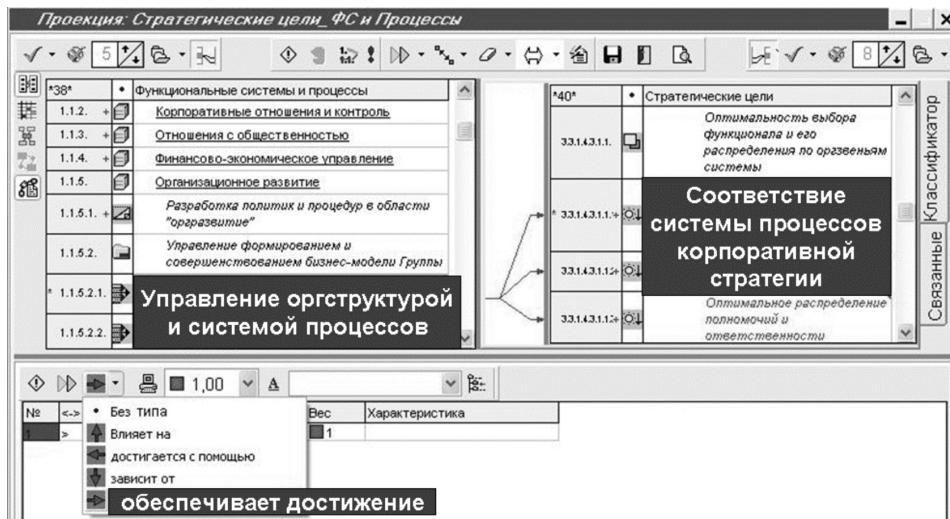


Рис. 3. Пример проекции (PR), связывающей бизнес-процессы предприятия с целями

Для визуализации перекрестных связей между классами используются проекции (рис. 3).

Связи в проекции могут быть направленными и ненаправленными, типизированными и нетипизированными.

Предлагаемый язык для наполнения онтологий и его программная реализация основаны на использовании графически размеченных иерархических списков [16] и связей между ними. Иерархические списки используются в первую очередь для ввода и редактирования экземпляров онтологии, хотя в частном случае они могут использоваться и для ввода классов (если тип связи в классификаторе «класс-подкласс»). Основные преимущества визуализации иерархическим списком – простота реализации и привычность для пользователя, поскольку такой же подход используется в многочисленных файловых браузерах (Проводник Microsoft Windows, Total Commander...). Данный метод дает четкое представление имен объектов и их иерархии. В отличие от других методов визуализации, в иерархическом списке имена классов не перекрывают друг друга, и чтобы их увидеть не нужно наводить курсор на соответствующий объект. Основная проблема визуализации иерархическим списком – представление только древовидных структур, но не сетевых. Однако данная проблема устранена в предложенном языке и программном инструментарием с помощью специального конструкта – проекции. Таким образом, предлагаемый способ визуализации онтологии позволяет устранить главный недостаток использования иерархических списков, сохранив его преимущества.

По простоте и наглядности разработанный инструмент близок средствам когнитивного моделирования (mindmappers, concept mappers), а по объему явно представленной семантики приближается к редакторам онтологий.

Настройка интерфейса пользователя-разработчика базы знаний

Для настройки интерфейсных форм языка классификаторов и проекций с помощью языка OWL и стандартных редакторов онтологий предлагается использовать элемент owl:Annotation Property, позволяющий производить аннотирование классов и свойств онтологии [17]. В аннотации предлагается указывать, в каком классификаторе или проекции будут редактироваться экземпляры аннотируемого класса или свойства.

Для реализации данного подхода онтология предметной области дополняется онтологической моделью интерфейса пользователя, содержащей следующее:

- классы «Классификаторы» и «Проекция»;
- экземпляры классов «Классификаторы» и «Проекция» – наименования конкретных классификаторов и проекций для наполнения онтологии;
- свойства-аннотации (owl:AnnotationProperty) «редактируется в классификаторе» и «редактируется в проекции».

После этого все классы и свойства онтологии предметной области, которую нужно наполнять, аннотируются с помощью заданных свойств-аннотаций и перечня классификаторов и проекций (рис. 4, 5).

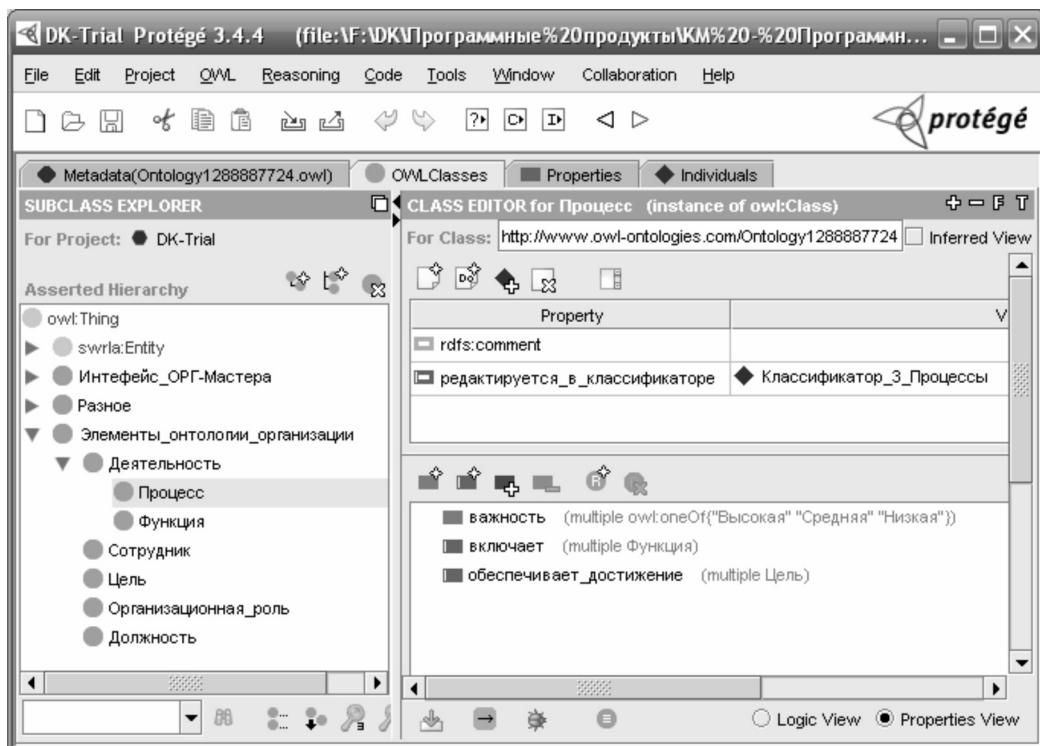


Рис. 4. Настройка классификатора: элементы класса Процессы редактируются в Классификаторе «Процессы»

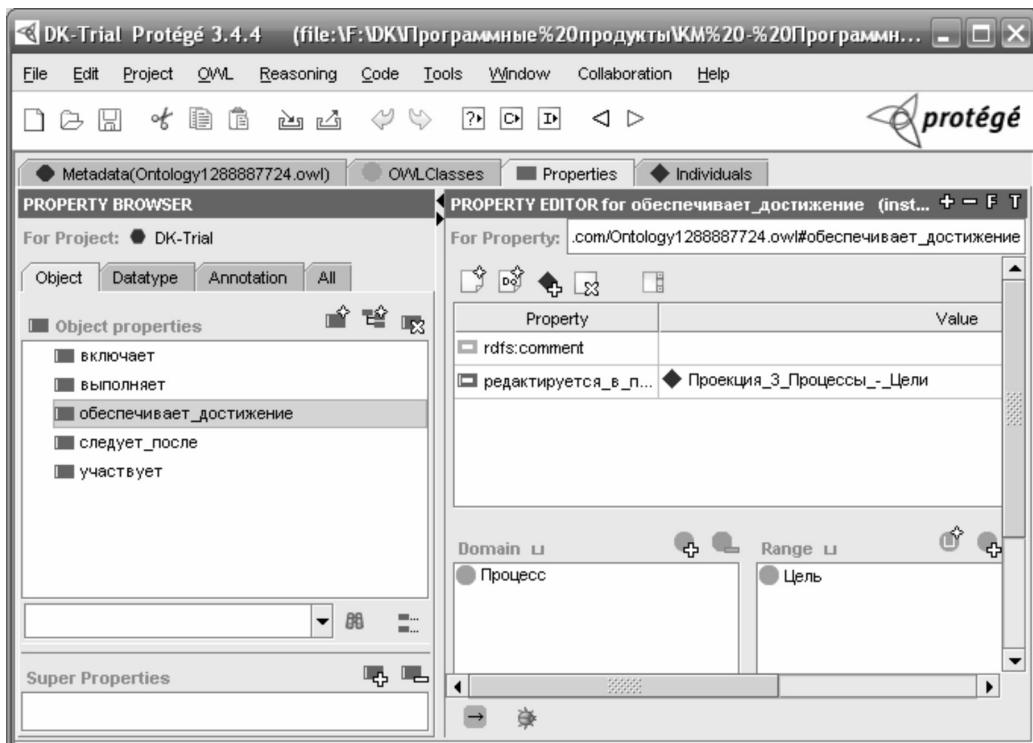


Рис. 5. Настройка проекции: свойства, связывающие Процессы с Целями, редактируются в проекции «Процессы – Цели»



Фрагмент полученной онтологии в формате OWL:

```

<owl:Class rdf:ID="Процесс">
  <редактируется_в_классификаторе>
    <Классификатор rdf:ID="Классификатор_З_Процессы"/>
  </редактируется_в_классификаторе>
  <rdfs:subClassOf rdf:resource="#Элементы_онтологии_организации"/>
</owl:Class>
  <owl:ObjectProperty rdf:ID="обеспечивает_достижение">
    <rdfs:domain rdf:resource="#Процесс"/>
    <rdfs:range rdf:resource="#Цель"/>
  <редактируется_в_проекции>
    <Проекция rdf:ID="Проекция_З_Процессы_—_Цели"/>
  </редактируется_в_проекции>
</owl:ObjectProperty>

```

Полученная онтология с настройками форм ввода и редактирования может быть передана в редактор базы знаний, поддерживающий язык классификаторов и проекций. В редакторе на основе полученной онтологии будет сформирована структура модели, в которой останется только уточнить пиктограммы для классов и свойств, и можно переходить к наполнению. Наполненная в редакторе база знаний должна использоваться, для этого из нее формируются необходимые отчеты, и к ней обеспечивается доступ, например, через веб-портал (рис. 6).

Внедрение

Предложенная технология используется для формирования баз знаний о деятельности коммерческих предприятий и органов государственной власти в задачах проектирования организа-

ционной структуры, оптимизации бизнес- или административных процессов и формирования организационно-нормативной документации (регламенты деятельности, положения о подразделениях, должностные инструкции) [15].

С использованием системы разработано более 10 административных регламентов для Федеральной миграционной службы, Федеральной регистрационной службы, Федерального агентства по физической культуре и спорту, Федеральной службы по труду и занятости и других организаций. Коммерческими организациями-пользователями системы являются: Бизнес-Инжиниринг Групп, группа предприятий ГОТЭК, «Группа «Илим», Киришская ГРЭС, ЗАО «Евросиб», ОАО «Иркутскэнерго», ОАО «Газаппарат», «АСТРА Холдинг», Торговый дом «Петровский», Холдинг ПЕКАР и др. Разработкой и сопровождением



Рис. 6. Комплексная система для разработки и использования баз знаний онтологического типа



Рис. 7. Фрагмент системы классов онтологии организации

модели предприятия (базы знаний) занимаются пользователями бизнес-аналитики, не владеющие специальными знаниями в области представления знаний. Масштаб создаваемых моделей достигает десятков тысяч элементов, например, классификатор функций на одном из предприятий включал более 20 000 позиций.

Оценка эффективности предложенной технологии

Для оценки эффективности предложенной технологии сравним ее с наиболее распространенным средством наполнения баз знаний – через формы, см. редактор экземпляров (The instance editor) в редакторе онтологий Protégé² или семантические формы (semantic forms) в системе Semantic MediaWiki³.

Для проведения эксперимента возьмем задачу установления связей между наборами экземпляров двух классов, в частности, между Бизнес-процессами и Целями, достижение которых они обеспечивают (связь многие ко многим). Данные понятия и связи между ними входят в онтологию организации. Установление указанных связей между экземплярами понятий является типовой задачей при моделировании организаций (рис. 7).

Для установления связей между «Процессом нижнего уровня 1.1.2» и «Целью 1.2» и «Целью 1.4» в предлагаемой системе ОРГ-Мастер (рис. 8) требуется пять действий (нажатий кнопки мыши):

- 1) выбор «Процесса нижнего уровня 1.1.2» в левом классификаторе;
- 2) выбор «Цели 1.2» в правом классификаторе;
- 3) установление связи между выбранными объектами путем нажатия на значок «Назначить связь»;

4) выбор «Цели 1.4» в правом классификаторе;

5) установление связи между выбранными объектами путем нажатия на значок «Назначить связь».

Рассмотрим теперь решение этой задачи с помощью форм в стандартном редакторе онтологий (рис. 9). Для установления связей между «Процессом нижнего уровня 1.1.2» и «Целью 1.2» и «Целью 1.4» в редакторе экземпляров системы Protégé требуется семь действий (нажатий кнопки мыши):

- 1) выбор «Процесса нижнего уровня 1.1.2» в навигаторе экземпляров (Instance browser);
- 2) нажатие кнопки «Добавить из имеющихся» («Add ...»);
- 3) выбор «Цели 1.2» из списка экземпляров класса «Цель»;
- 4) установление связи – нажатие кнопки «OK» в окне выбора;
- 5) нажатие кнопки «Добавить из имеющихся» («Add ...»);
- 6) выбор «Цели 1.4» из списка экземпляров класса «Цель»;
- 7) установление связи – нажатие кнопки «OK» в окне выбора.

В результате предлагаемая технология позволяет решать представленную типовую задачу на 28 % быстрее.

Кроме количественных можно отметить качественные преимущества предложенной технологии. В случае увеличения масштаба базы знаний (например, при моделировании организаций количество процессов исчисляется сотнями) при использовании форм усложняется поиск нужного экземпляра класса в линейном списке. Хотя редакторы экземпляров обычно предоставляют возможность поиска по ключевым словам, это часто не помогает – поиск по ключевым словам полезен тогда, когда пользователь точно знает, что он ищет, а когда имеет приблизительные представления, лучше работает поиск по каталогу (навигация) [18, 19]. В предлагаемой технологии вы-

² http://protegewiki.stanford.edu/wiki/PrF_UG_instance_editor

³ http://semantic-mediawiki.org/wiki/Semantic_Forms

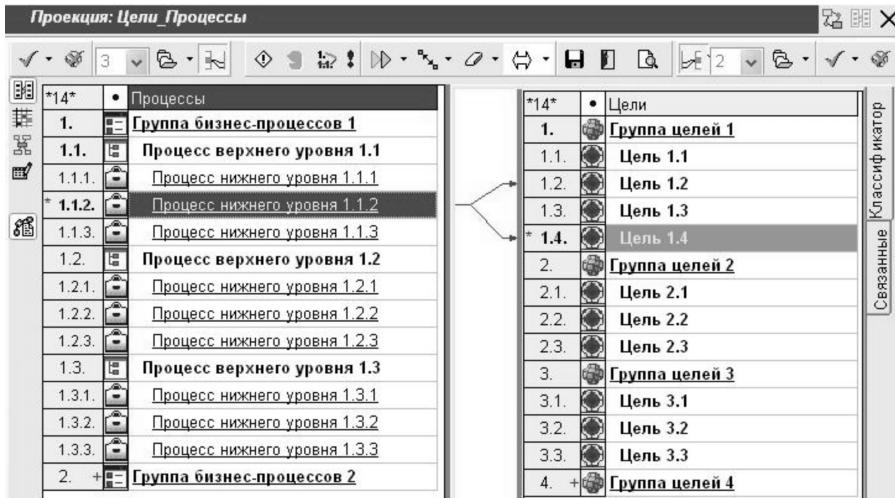


Рис. 8. Наполнение онтологии (установление связей)
в предлагаемой системе наполнения баз знаний онтологического типа

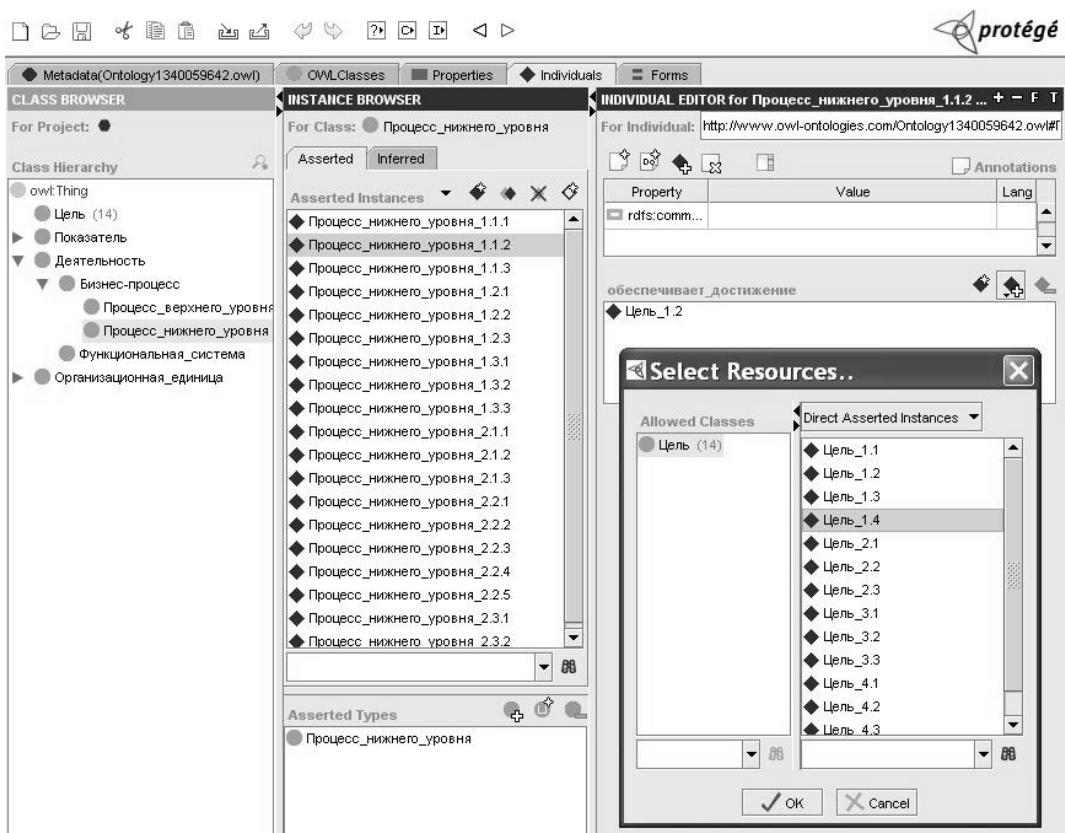


Рис. 9. Наполнение онтологии (установление связей) в стандартном редакторе экземпляров

бор экземпляров классов для установления связей осуществляется путем навигации по иерархическим спискам (классификаторам), а не только (и не столько) поиском по ключевым словам. Такая возможность делает редактор средством рассуждения для пользователя. Кроме того, выбор объ-

ектов путем навигации по иерархическим спискам (классификаторам) обеспечивает целостное восприятие предметной области [20].

В статье предложены язык классификаторов и проекций для наполнения баз знаний онтологиче-

ского типа и его программная поддержка. Показан механизм настройки интерфейса пользователя-разработчика базы знаний, основанный на стандартном языке представления онтологий OWL, который может быть реализован в стандартных редакторах онтологий. Кроме того, предложен подход к созданию комплексной системы для разработки и использования баз знаний онтологического типа, объединяющий разработанный редактор базы знаний со стандартными инструментами разработки онтологий и средствами предоставле-

ния доступа к знаниям (запросный поиск, навигация, частные представления знаний, порталы). Новизна работы, в первую очередь, заключается в уникальном языке классификаторов и проекций для формирования баз знаний онтологического типа, объединяющем возможности современных средств визуализации онтологий с подходами к ручному наполнению онтологий. Результаты работы апробированы и внедрены в практику. Проведена сравнительная оценка предложенной технологии.

СПИСОК ЛИТЕРАТУРЫ

1. Гаврилова, Т.А. Онтологический подход к управлению знаниями при разработке корпоративных информационных систем [Текст] / Т.А. Гаврилова. –Новости искусственного интеллекта. –2003. –№ 2. –С. 24.
2. Гаврилова, Т.А. Модели и методы формирования онтологий [Текст] / Т.А. Гаврилова, Д.В. Кудрявцев, В.А. Горовой // Научно-технические ведомости Санкт-Петербургского государственного политехнического университета. –2006. –№ 46. –С. 21–28.
3. Davies, J. Towards the Semantic Web: Ontology-driven Knowledge Management [Text] / Eds. J. Davies, D. Fensel, F. van Harmelen, – England: John Wiley and sons Ltd, 2003.
4. Brickley, D. RDF Vocabulary Description Language 1.0: RDF Schema W3C Recommendation [Электронный ресурс] / D. Brickley, R.V. Guha. –10.02.2004. –Режим доступа: <http://www.w3.org/TR/rdf-schema/>
5. Dean, M. OWL Web Ontology Language Reference [Электронный ресурс] / Eds M. Dean, G. Schreiber, F. van Harmelen [et al.]. –Режим доступа: <http://www.w3.org/TR/owl-ref/>
6. Grove, M. RDF Instance Creator (RIC) [Электронный ресурс] / M. Grove. –2009. –Режим доступа: <http://www.mindswap.org/~mhgrove/RIC/RIC.shtml>
7. Palmer, M. Conzilla - a Conceptual Interface to the Semantic Web [Text] / M. Palmer, A. Naeve // Lecture notes in computer science 3596, 2005. –Vol. 136.
8. Berners-Lee, T. Tabulator Redux: Writing Into the Semantic Web [Text] / T. Berners-Lee [et al.] // Technical Report. –University of Southampton Electronics and Computer Science, 2007.
9. Kelly, S. Domain-Specific Modelling: Enabling Full Code Generation [Text] / S. Kelly, J. Tolvanen // Wiley-IEEE Computer Society Press. –2008. –448 р.
10. Павлинов, А.А. О средствах разработки проблемно-ориентированных визуальных языков [Текст] / А.А. Павлинов, Д.В. Кознов, А.Ф. Перегудов [и др.] // Системное программирование. –2006. –Т. 2. –№ 1. –С. 116–141.
11. Сорокин, А.В. Обзор eclipse modelling project [Текст] / А.В. Сорокин, Д.В. Кознов // Системное программирование. –2010. –Т. 5. –№ 1. –С. 6–32.
12. Krotzsch, M. Semantic MediaWiki [Текст] / M. Krotzsch, D. Vrandecic, M. Volkel // Proc. of the V International Semantic Web Conf. (ISWC06). –2006. –Р. 935–942.
13. Schaffert, S. IkeWiki: A Semantic Wiki for Collaborative Knowledge Management [Текст] / S. Schaffert // Proc. of the I International Workshop on Semantic Technologies in Collaborative Applications (STICA 6). –2006.
14. Григорьев, Л.Ю. Визуальный язык классификаторов и проекций для разработки баз знаний [Текст] / Л.Ю. Григорьев, Д.В. Кудрявцев // Труды XII Национальной конф. по искусственному интеллекту (КИИ-2010). –Тверь, 2010. –Т. 2.
15. Григорьев Л.Ю. Организационное проектирование на основе онтологий: методология и система ОРГ-Мастер [Текст] / Л.Ю. Григорьев, Д.В. Кудрявцев // Научно-технические ведомости Санкт-Петербургского государственного политехнического университета. Сер. Информатика. Телекоммуникации. Управление. –2012. –№1. –С. 21–29.
16. Katifori A. Ontology visualization methods - a survey [Текст] / A. Katifori, C. Halatsis, G. Lepouras [et al.] // ACM Comput. Survey. –2007. –Vol. 39(4).
17. Кудрявцев, Д.В. Настройка форм для визуального наполнения онтологий [Текст] / Д.В. Кудрявцев // Сб. трудов конф. Управление знаниями и технологиями семантического веба - 2010. –СПб.: Изд-во СПбГУИТМО, 2010. –С. 135–142.
18. Spencer, D. Four Modes of Seeking Information and How to Design for Them: [Электронный ресурс] / D. Spencer 19.06.122006/03/14. –Режим доступа: http://www.boxesandarrows.com/view/four_modes_of_seeking_information_and_how_to_design_for_them
19. Кудрявцев, Д.В. Системы управления знаниями и применение онтологий: Учеб. пособие [Текст] / Д.В. Кудрявцев. –СПб.: Изд-во Политехн. ун-та, 2010. –343 с.
20. Гаврилова, Т.А. Когнитивный подход к созданию онтологии [Текст] / Т.А. Гаврилова, А.В. Воинов // Научно-техническая информация. Сер. 2 Информационные процессы и системы. –2007. –№ 3. –С. 19–23.