УДК 004.514, 004.82
DOI 10.25205/1818-7900-2018-16-1-39-49

К. Г. Высоцкий [1, 2], **Ю. А. Родина** [2], **Е. А. Сидорова** [3]

[1] *ООО «ДатаВоркс»*
пр. Академика Лаврентьева, 2/2, Новосибирск, 630090, Россия

[2] *Новосибирский государственный университет*
ул. Пирогова, 1, Новосибирск, 630090, Россия

[3] *Институт систем информатики им. А. П. Ершова СО РАН*
пр. Академика Лаврентьева, 6, Новосибирск, 630090, Россия

kostya19494@gmail.com, julia.rodina97@gmail.com, lsidorova@iis.nsk.su

АТРИБУТНАЯ КОНВЕЙЕРНАЯ МОДЕЛЬ ДЛЯ ИНТЕРАКТИВНОЙ ВИЗУАЛИЗАЦИИ ОБЪЕКТНЫХ СВЯЗЕЙ *

Определена модель для визуализации связей между объектами и их атрибутами в различных процессах. На основании модели разработан универсальный абстрактный компонент графического пользовательского интерфейса и приведены примеры его программной реализации. Также проведена апробация компонента для решения прикладной задачи по извлечению информации из документов.

Ключевые слова: графическая нотация, графическая модель, диаграмма, визуализация процесса, объектная модель, объектные связи, схема фактов.

Введение

Одной из важнейших частей моделирования различных систем является формализация протекающих в них процессах. Процесс может быть смоделирован с помощью математических формул и словесных описаний, представляющих собой условные инструкции и последовательности действий. Для сложных систем представляется трудоемким не только составление формализованной модели, но и ее восприятие человеком, поэтому часто прибегают к помощи графических средств, чтобы упростить понимание материала. В их число входят диаграммы и графические нотации – стандартизируемые графические способы формализации, отображающие связи между компонентами системы или их взаимодействие в процессах. Например, технология структурного проектирования IDEF0 (ICAM Definition for Function Modeling, «определение функционального моделирования ICAM», ICAM – это сокращение от названия набора стандартов Integrated Computer Aided Manufacturing, «Интегрированное компьютеризированное производство») представляет анализируемый процесс в виде совокупности множества работ, взаимодействующих на основе определенных правил, с учетом потребляемых материальных или информационных ресурсов, имеющих вход и выход [1]. IDEF0 следует идее разбиения процесса на функциональные блоки с помощью гра-

* Работа выполнена при частичной финансовой поддержке Президиума СО РАН (Блок 36.1 Комплексной программы ФНИ СО РАН II.1) и РФФИ (грант № 17-07-01600).

Высоцкий К. Г., *Родина Ю. А.*, *Сидорова Е. А.* Атрибутная конвейерная модель для интерактивной визуализации объектных связей // Вестн. НГУ. Серия: Информационные технологии. 2018. Т. 16, № 1. С. 39–49.

фических объектов, где каждый блок преобразует ресурсы на входе в ресурсы на выходе, передавая их следующим блокам (рис. 1).

Еще одним популярным способом визуализации процессов является Process Flow Diagram [1] (PFD, «принципиальные схемы технологического процесса»). PFD предназначена для описания химических и инженерных процессов [2; 3] и фокусируется на демонстрации оборудования и веществ, которые оно преобразует (рис. 2). Важной составляющей PFD является перечень типичного оборудования, такого как емкости, нагреватели, клапаны и т. д. Это стандартный список, т. е. у всех инструментов и материалов имеется соответствующее стандартное графическое изображение.

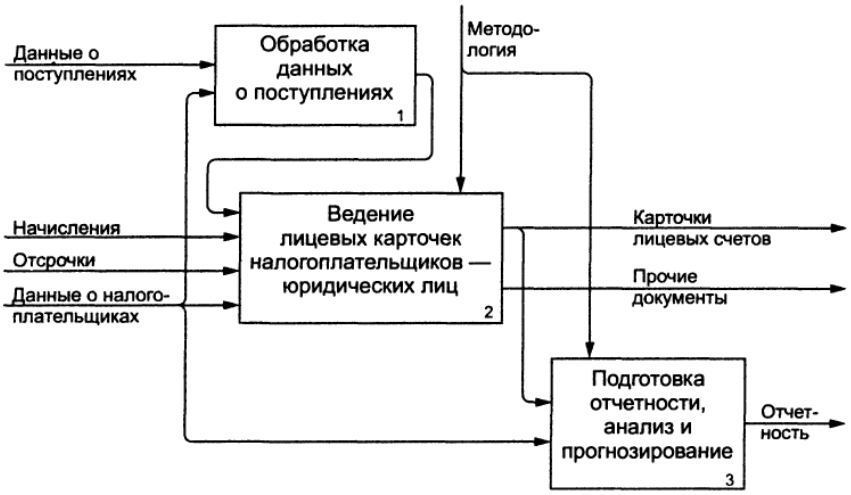

Рис. 1. Пример диаграммы IDEF0

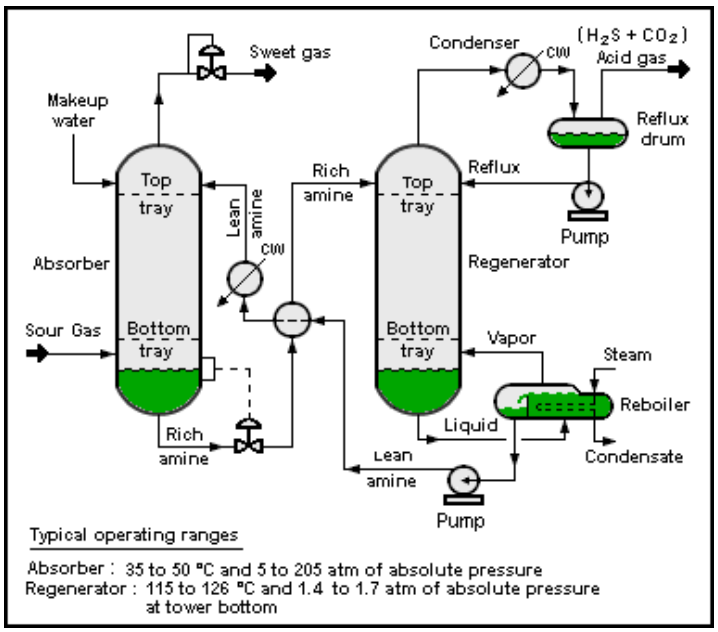

Рис. 2. Пример диаграммы PFD. Стрелки обозначают перемещение вещества, остальные графические элементы (круги, овалы и т. д.) – химическое оборудование. *Источник*: https://commons.wikimedia.org/wiki/File:AmineTreating.png

[1] Руководство по PFD. URL: http://www.klmtechgroup.com/PDF/EGD2/ENGINEERING_DESIGN_GUIDELINES_process_flow_sheet_rev_web.pdf (дата обращения 25.02.2018).

В этой статье мы обращаем внимание на еще одну интересную графическую нотацию. Она довольно широко используется в визуализации процессов, в которых важно взаимодействие не столько между самими объектами, сколько между их свойствами, в отличие от IDEF0 и PFD. Далее мы дадим ее формальное определение и приведем примеры использования в практическом приложении, для визуализации и моделирования процессов извлечения информации из документов.

Атрибутная конвейерная модель визуализации

Атрибутная конвейерная модель визуализации (АКМ) – это логическая модель данных, предназначенная для отображения связей между свойствами (или атрибутами) объектов, участвующих в одном процессе. АКМ составляют следующие графические элементы.

- Узел – сущность, предназначенная для визуализации объекта. Узел представляет собой прямоугольную панель с заголовком вверху. Под заголовком, сверху вниз, располагаются составляющие, соответствующие свойствам объекта.
- Панели свойств – это текстовые панели, соответствующие некоторым свойствам объекта, определяемым пользователем. Они располагаются внутри узла под заголовком в виде списка. Каждая панель свойства имеет один из двух типов: входной или выходной. Типизация устанавливает направленные отношения между свойствами: свойства выходного типа в процессе передают свои значения связанным с ними свойствам входного типа. В АКМ с одним объектным свойством сопоставляется лишь одна панель, однако разные свойства могут обозначаться одинаковым текстом. Например, при моделировании процесса «Сложение» мы можем создать узел для объекта «Сумматор» с двумя и более свойствами с именами «Слагаемое».
- Коннекторы (соединители) – это элементы, отображаемые в виде кружков слева или справа от панели свойства. Расположение коннектора задает тип свойства: свойство имеет входящий тип, если коннектор находится слева, и выходящий тип, если коннектор справа.
- Соединения – это линии между коннекторами, показывающие направленные связи между свойствами. Для обеспечения целостности модели на соединения накладываются следующие ограничения:
  - у коннектора свойства входного типа может быть только одно соединение;
  - у коннектора свойства выходного типа может быть сколько угодно соединений;
  - нельзя установить соединение между свойствами одного и того же объекта;
  - соединение можно установить только между свойствами разных типов.

Указанные ограничения, а также способ расположения коннекторов подталкивают пользователя к созданию диаграмм, где панели с определяемыми свойствами находятся правее панелей с определяющими свойствами (рис. 3). Такое расположение делает интуитивно более понятным то, в какой стадии процесса участвует тот или иной объект и его свойство.

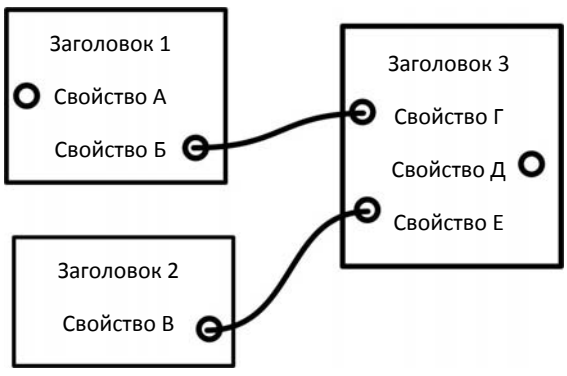

Рис. 3. Диаграмма АКМ

«Правильные» диаграммы АКМ должны обладать свойством конвейерности – описывать процессы, внутри которых не возникают циклы. Иначе говоря, в этих диаграммах нельзя составить такой путь по связям, который бы начинался и заканчивался в одном и том же объекте. Определенная нами модель частично заимствует идею конвейерного процесса из диаграмм модели SCA Assembly (модель сборки сервис-компонентной архитектуры) [4]. SCA была создана для того, чтобы описывать процессы взаимодействия служб с компонентами приложения (рис. 4): компоненты передают службам ссылки на себя, когда обращаются к службам (относительно компонентов эти связи представлены фиолетовыми указателями), и также выступают в роли обработчиков сигналов от служб, прини-

мая от них сигналы (эти связи представлены зелеными указателями). Диаграммы и модель SCA включают в себя также и другие элементы (параметры, производитель и т. д.), но в АКМ они не участвуют.

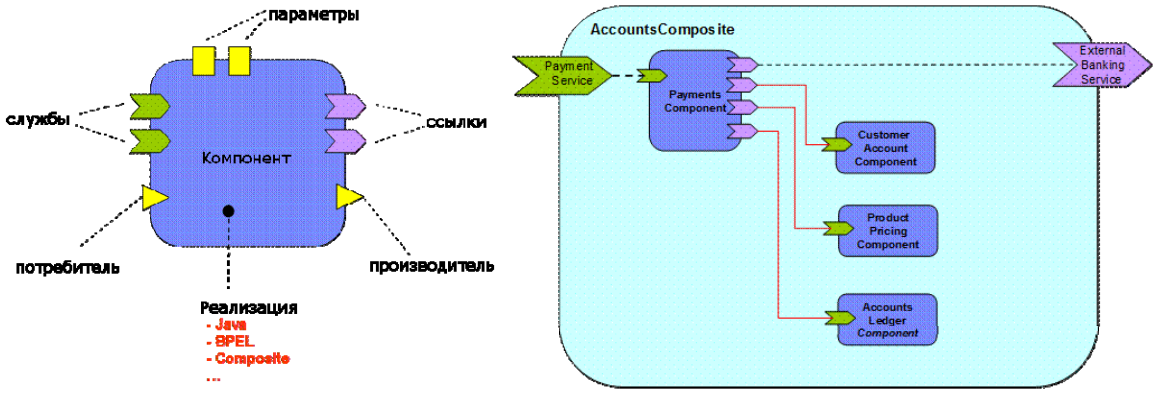

Рис. 4. Компонент диаграммы SCA Assembly (слева) и диаграмма системы, включающей множество компонентов (справа)

Проблемная область

Задача визуализации сформировалась во время работы над графическим редактором схем фактов, используемых для фактографического анализа текста. Схема фактов – это модель, которая проецирует определенный класс выражений на естественном языке на предметную область рассматриваемой системы [5]. Схема включает набор правил вывода и ограничений, по которым элементы текста порождают множество информационных объектов (или экземпляров предметной области). Для описания предметной области используется онтология. Онтология – это модель, позволяющая представить понятия предметной области в виде иерархии, где каждое понятие описывается классом и набором атрибутов и связей, которые наследуются вниз по иерархии. Таким образом, схемы фактов опираются на онтологию и при моделировании включают следующие основные компоненты.

- *Аргументы* – это набор объектов, отражающий структуру языкового высказывания, описывающего факт из выбранной предметной области. С каждым объектом сопоставляется фрагмент текста. Аргумент описывается с помощью семантических характеристик принадлежности объекта к определенному лексическому или онтологическому классу.
- *Комплексное условие схемы* – это множество ограничений, накладываемых на взаимоотношения нескольких аргументов, такие как морфологическая согласованность, близость или порядок следования.
- *Результат* обозначает объект, изменяемый или порождаемый при обработке схемы фактов анализатором. В соответствии со схемой атрибутам результирующего объекта присваиваются либо значения атрибутов *Аргументов*, либо значения по умолчанию, которые можно задать отдельно для каждого *Результата*. При порождении нового объекта для него указывается онтологический класс.

Так как схемы фактов отображают последовательные процессы создания информации (объектов онтологии), интенсивно используя при этом связи между свойствами объектов, мы решили воспользоваться подходом к визуализации на основе атрибутной конвейерной модели.

Рассмотрим некоторые проблемы, возникающие при графическом моделировании схем фактов с помощью АКМ.

Во-первых, для описания *Условия* взаимодействия *Аргументов* необходимо иметь возможность выбирать вид условия (морфологическая согласованность, условие на сегмент, взаиморасположение и т. д.) и указывать значение каких-либо параметров согласования (на-

пример, для морфологической согласованности – род, число, падеж и пр.). Другие элементы схемы фактов не влияют на эти параметры, поэтому при моделировании пользователь должен выбирать их сам (из списков возможных значений).

Во-вторых, специфика онтологического описания и схем фактов предполагает, что некоторые объекты могут сами выступать в качестве определяющих свойств. Рассмотрим эту проблему на примере схемы фактов (рис. 5), которая включает два аргумента arg1 и arg2 и одно условие на их взаиморасположение. Список свойств панели *Условия* состоит из двух аргументов, на которые накладывается ограничение определенного вида. Чтобы в АКМ можно было задать соединение между аргументом и его ограничением, нужно вынести этот аргумент в список его собственных свойств. Это кажется нелогичным и громоздким, тогда как наша задача – упростить создание схем фактов и представить их в компактном, удобном, читаемом виде. Более правильным представляется создание обособленного свойства, отвечающего объекту.

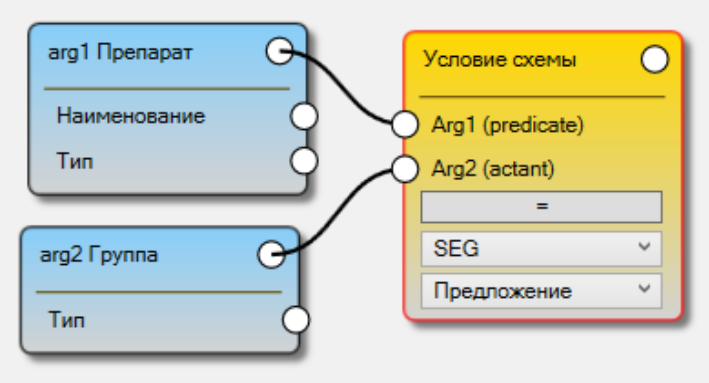

Рис. 5. Пример реализации расширенной АКМ

Расширение атрибутной конвейерной модели

На основании изложенных проблем мы сформировали следующие требования к графической библиотеке для визуализации схем фактов:

1) возможность изменять свойства объектов без обязательного использования связей;
2) возможность для объекта становиться свойством.

Выполнение этих требований предполагает не только необходимость расширения функционала отдельной графической библиотеки, но и необходимость расширения АКМ, поэтому далее мы расскажем, как изменения в библиотеке отразились на АКМ в целом.

Так как наша модель должна предоставлять пользователю более гибкий контроль над свойствами, менять их без участия связей, мы предлагаем абстрагироваться от понятия «свойство» как от текстовой панели. Чтобы свойства наиболее точно соответствовали нашему представлению об интерактивной визуализации объектной модели, мы разрешили пользователю передавать в качестве свойств любой компонент Windows Presentation Foundation. Это означает, что теперь интерактивность не ограничивается прикреплением и отделением соединений, но также появилась возможность изменять объект и изнутри его визуализирующей панели. В этом подходе мы ориентировались на систему трехмерного моделирования Blender [2], где с помощью аналогичной библиотеки программируется процесс создания изображения, его пред- и постобработки. На рис. 6 слева изображена панель настройки сдвига изображения по пространственной координате *Z*. Пользователь может изменить эту координату двумя способами: либо присоединив свойство к коннектору, которое передаст свое значение *Z*-координате, либо определив это значение вручную, без использования соединений. В Blender это реализовано через поля пользовательского ввода.

[2] См. документацию к инструменту: https://docs.blender.org/manual/en/dev/compositing/ (дата обращения 25.02.2018).

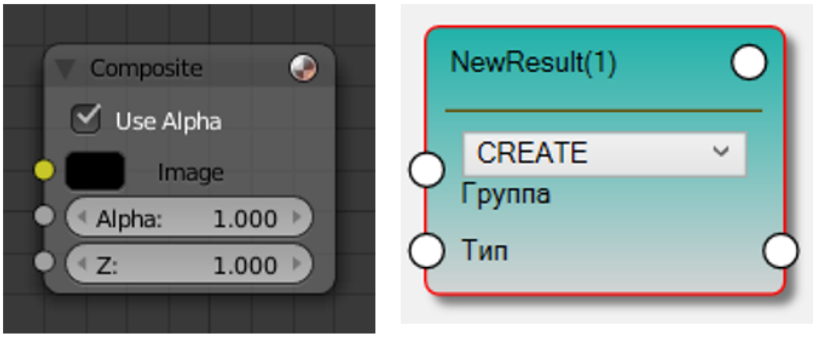

Рис. 6. Независимые свойства в Blender 2.79 (слева)
и в нашей реализации расширенной АКМ (справа)

Свойства, которые позволяют напрямую изменять объект, не используя соединения, мы будем называть независимыми. Возвращаясь к проблеме из примера со свойствами *Условия* схемы, мы можем заметить, что благодаря большему абстрагированию свойств модели проблема решается добавлением в панель свойств выпадающих меню с вариантами выбора вида и значения *Условия*.

Ограничения свойств конвейерной модели предполагают, что каждое свойство либо принимает значение другого свойства, либо задает его, т. е. у каждого свойства есть либо входной коннектор, либо выходной. Так как некоторые независимые свойства, такие как переключатели (см. рис. 6 слева на вершине списка свойств) или панель выбора вида ограничения в панели *Условия* схемы фактов, не предполагают изменения других свойств, мы решили изменить подход к типизации свойств. Теперь каждое свойство имеет один из трех типов:

1) входной – используется, если свойство принимает значения от другого свойства;
2) выходной – используется, если свойство передает свое значение другим свойствам;
3) без соединений – используется, если свойство независимое и не влияет на другие свойства.

Чтобы иметь возможность использовать объект в качестве свойства, мы ввели в графическую модель дополнительный коннектор и расположили справа от заголовка узла. Коннектор обладает теми же возможностями и ограничениями, что и коннектор свойства выходного типа.

Таким образом, мы определили расширенную атрибутную конвейерную модель (РАКМ), которая представляет собой АКМ со следующими отличиями:

- заголовок узла – это свойство выходного типа, которое представляется только текстовой панелью;
- свойства имеют один из вышеперечисленных типов;
- свойства представляются текстовыми панелями или определенными пользователем интерактивными панелями.

Отметим, что РАКМ предназначена в первую очередь для визуализации в графическом интерфейсе, а не для создания формализованных диаграмм, так как формальное описание каждого элемента, который пользователь желает внедрить в диаграмму, является достаточно трудоемкой задачей, тогда как программное описание пользовательских элементов формализма не требует.

Объектная модель РАКМ

Для визуализации схем фактов мы разработали компонент, который основывается на библиотеке NetworkView [3]. Несмотря на неоднозначное название, которое отсылает к сетевым

[3] Описание и реализация: https://www.codeproject.com/Articles/182683/NetworkView-A-WPF-custom-control-for-visualizing-a

моделям визуализации СУБД, она демонстрирует процесс реализации простой атрибутной конвейерной модели (рис. 7).

Как мы видим, библиотека NetworkView позволяет визуализировать графы, сети и модели протекания различных процессов. Она нетребовательна к ресурсам и с успехом может применяться в проектах на C#, построенных с использованием Windows Presentation Foundation или Windows Forms, если пользователю доступен компонент ElementHost, обеспечивающий поддержку элементов WPF в WinForms.

Несмотря на то что эта библиотека показалась нам в целом пригодной для наших целей, в дальнейшем выяснилось, что, для того чтобы обеспечить полную поддержку РАКМ, все же частично необходимо расширить ее функционал. Поэтому для разработки графического редактора схем фактов мы создали собственную версию NetworkView, изменив некоторые ее составляющие.

Библиотека NetworkViev предлагает клиентским приложениям обрабатывать события, посылаемые всеми графическими элементами, такими как добавление узла, выбор узла, перетаскивание курсора из коннектора, перемещение курсора, щелчки по элементам и т. д. Альтернативным решением является построение объектной модели для узла сети, инкапсулирующей события, с учетом того, что в нашем подходе используется более абстрактная конвейерная модель. Также передача в контейнер, содержащий узлы, объекта вместо разрозненного набора обработчиков событий интерфейса делает отладку приложения значительно удобнее, а структуру программы – понятнее и логичнее. Далее мы представим объектную модель узла.

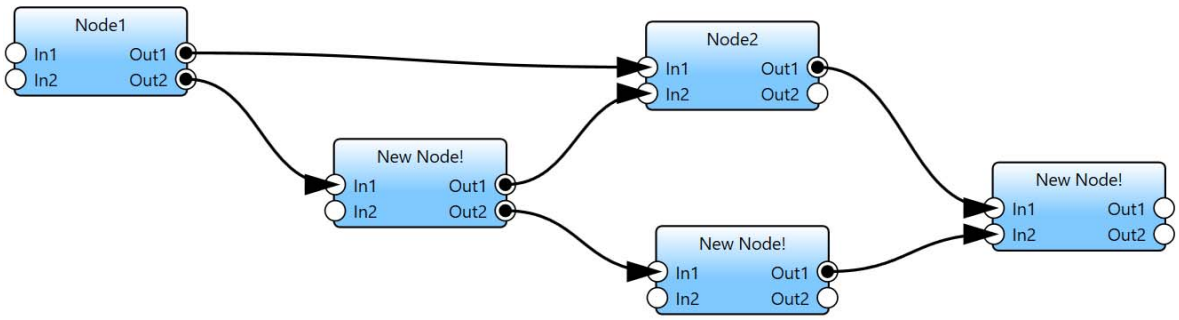

Рис. 7. Пример сети, реализованной с помощью NetworkView

NodeInfo – класс, определяющий узлы. В него входят следующие поля.

- Tag – это объект, который мы визуализируем. Он же является свойством, выходной коннектор которого располагается в верхнем углу узла.
- Name – заголовок узла. Изменение поля Name вызывает событие, для которого пользователь может задать обработчик, так клиентское приложение сможет откликаться на изменения заголовка.
- NameChangeable – флаг, который определяет, можно ли изменять заголовок узла.
- Sections – список свойств, отображаемых в узле.
- FillColor – цвет панели узла, для отличительной способности.

Свойства из списка Sections принадлежат к классу SectionInfo, объекты которого содержат следующую информацию:

- Input, Output – ссылки на входной и выходной коннектор соответственно;
- IsInput, IsOutput – комбинации этих полей определяют тип свойства (вход, выход, без соединений);
- InputValidation – ссылка на пользовательский валидатор. Эта функция вызывается всплывающим событием перед установкой соединения между двумя коннекторами. Если она

опускает флаг валидности события, то событие перестает обрабатываться, а соединение отменяется;

- InputAdded, OutputAdded – ссылка на пользовательский обработчик события, которое возникает при добавлении к коннектору входящего или исходящего соединения соответственно;
- UIPanel – визуальный элемент, отображающий свойство. Может быть и простым текстом, и сложным компонентом, делающим свойство независимым;
- Data – пользовательские данные, привязанные к свойству. Когда это поле изменяется, вызывается пользовательский обработчик события. Обычно изменения производятся внутри UIPanel.

Экземпляры класса NodeInfo описывают содержимое узла и собираются в клиентском приложении. В нашей библиотеке все узлы содержатся в контейнере, в котором происходит отрисовка соединительных линий и который занимается обработкой и переадресацией событий, срабатывающих на узлах и коннекторах. Внутри контейнера пользователь может перемещать узлы, также контейнер предусматривает смещение вида («сдвиг камеры») и масштабирование элементов для упрощения навигации.

Использование объектной модели

Графический редактор схем фактов предназначен для моделирования процессов извлечения информации из текстов на естественном языке. На рис. 8 приведен пример визуализации схемы фактов, предназначенной для извлечения информации из технической документации. Предполагается, что построение структурированной модели содержания технического описания позволит ускорить и упростить процесс анализа и верификации официальных документов для экспертов и разработчиков программного обеспечения [6].

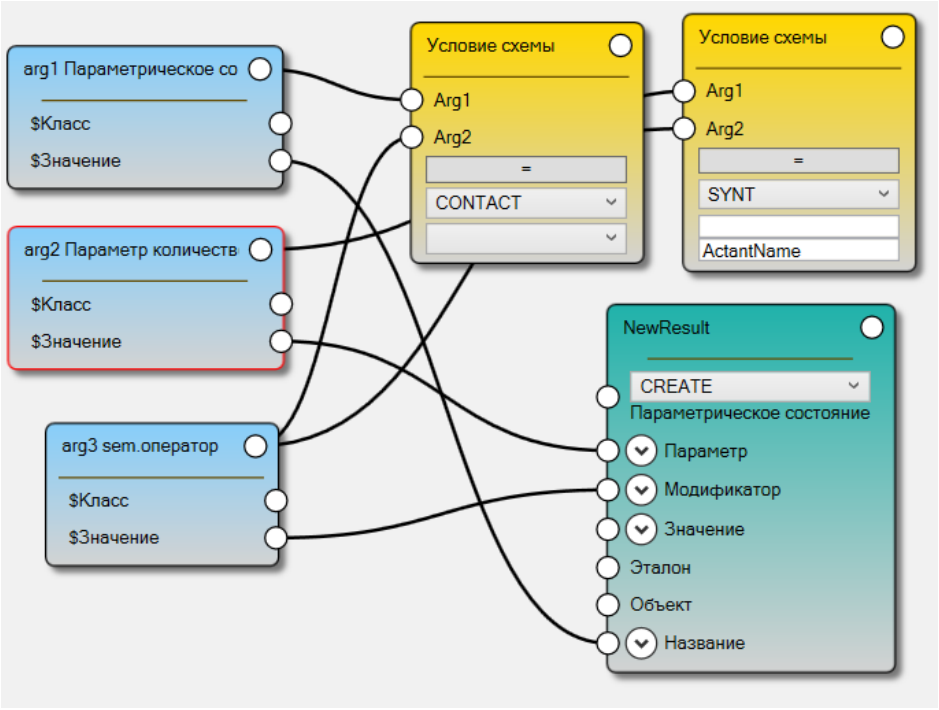

Рис. 8. Редактор схем фактов, реализующий РАКМ

Для апробации предложенного подхода были разработаны онтология предметной области «Автоматизированные системы управления техническими процессами (АСУ ТП)» и семантический словарь предметной лексики, которые загружаются в редактор схем фактов и зада-

ют множество начальных значений (имена понятий, списки атрибутов и т. п.), которыми может оперировать пользователь при моделирования процессов извлечения информации. Выделяются два основных типа схем фактов: схемы, служащие для начальной инициализации объектов, и схемы для выявления связей. Схемы фактов первой группы необходимы для начального формирования онтологических сущностей на основании словарных признаков. Схемы фактов второй группы моделируют процессы «обнаружения» фрагментов онтологии. В нашем примере к таким фрагментам относятся описания различных ситуаций, связанных с устройством АСУ ТП, наличие у объектов определенных состояний, управление одних объектов другими, контроль и измерение одних объектов другими и т. п. Так, с помощью схемы, приведенной на рис. 8, моделируется процесс извлечения информации о параметрических состояниях объектов. Для создания онтологического объекта класса *Параметрическое состояние* используется соответствующий признак словаря, а также признаки *Количественный параметр* для заполнения атрибута *Параметр* и *Оператор* для атрибута *Модификатор*. Данная схема включает два условия – на синтаксическую сочетаемость аргументов и их взаиморасположение.

Отвечая на вопрос, справляется ли разработанный на основе РАКМ инструмент со своей задачей – упростить написание схем фактов, сравним представление описанной выше схемы фактов в графическом виде с соответствующим ей XML-кодом:

```
<Scheme Name="ПарамСост" Segment="">
  <Argument Order="1" ObjectType="TERMIN" ClassName="Параметрическое состояние"
            TypeCompare="EQUAL" />
  <Argument Order="2" ObjectType="TERMIN" ClassName="Параметр количественный"
            TypeCompare="EQUAL" />
  <Argument Order="3" ObjectType="TERMIN" ClassName="sem.оператор"
            TypeCompare="EQUAL" />
  <Result Name="NewResult" ClassName="Параметрическое состояние" Type="CREATE">
      <Rule Type="ATTR" AttrName="Название" ResourceType="ARG" Resurce="1"
            FromAttrName="$Значение" Default="" />
      <Rule Type="ATTR" AttrName="Параметр" ResourceType="ARG" Resurce="2"
            FromAttrName="$Значение" Default="" />
      <Rule Type="ATTR" AttrName="Модификатор" ResourceType="ARG" Resurce="3"
            FromAttrName="$Значение" Default="" />
  </Result>
  <ConditionComplex Arg1="1" Arg2="3">
    <Condition ID="1" Type="CONTACT" Operation="EQ" Data="OBJ" />
  </ConditionComplex>
  <ConditionComplex Arg1="3" Arg2="2">
    <Condition ID="2" Type="SYNT" Operation="EQ" Data="" />
  </ConditionComplex>
</Scheme>
```

Если за показатель эффективности брать отношение числа графических элементов (узлов, соединений, свойств) к величине кода на XML, то мы видим, что для *Результата*, который обозначается как Result в исходном коде, эффективность сильно растет с числом определяемых атрибутов, так как на каждый связанный атрибут приходится по одной строчке Rule с набором параметров. Эффективность, однако, будет снижаться в случае увеличения числа *Условий*, так как в XML они сгруппированы по входящим в них аргументам. Эта проблема снимается, если разрешить добавление новых ограничений в существующие узлы с *Условиями*, чтобы не создавать новые.

Таким образом, мы можем сделать вывод, что мы получили эффективную и гибкую реализацию атрибутной конвейерной модели, которую можно использовать для решения задач моделирования различных процессов. Созданный на основе расширенной АКМ редактор схем фактов может активно использоваться для решения задач в области анализа текстов на естественном языке, так как за счет визуализации позволяет сделать процесс создания схем фактов более наглядным и удобным для лингвиста, а также помогает обеспечить целостность всего процесса извлечения информации (наличие входных данных для любой схемы, отсутствие циклов, корректность параметров и т. п.).

Заключение

В работе было дано формальное определение графической нотации модели конвейерного типа и предложено ее расширение, которое имеет большой потенциал для реализации в виде компонентов графического пользовательского интерфейса. Разработана программная библиотека, реализующая расширенную атрибутную конвейерную модель, которую можно легко встраивать в программные системы.

В дальнейшем планируется решить проблему полной формализации РАКМ: несмотря на то что эта модель позволяет создавать эффективные *интерактивные* решения за счет расширения пользовательскими элементами управления, на текущий момент нет удобного способа создания *статических*, иллюстрирующих формальных диаграмм, включающих неуправляемые панели свойств, так как пользователь должен строго определить все входящие в диаграмму новые элементы.

Также есть потенциал для дальнейшего развития нашей графической библиотеки и объектной модели РАКМ. Например, с текущим видом объектной модели трудно работать со свойствами и видеть результат обработки диаграмм в реальном времени, так как обратная связь от изменения свойств ограничивается только событиями подключения и удаления соединения. Планируется добавить новые сигналы, которые будут распространяться по соединениям до концов конвейера, когда создается новое соединение или изменяется независимое свойство.

Список литературы

1. *Черемных С. В., Семенов И. О., Ручкин В. С.* Моделирование и анализ систем. IDEF-технологии: практикум. М.: Финансы и статистика, 2006.

2. *Couper J. R. et al.* Chemical Process Equipment: Selection and Design. 2nd ed. Gulf Professional Publ., 2005.

3. *Turton R. et al.* Analysis, Synthesis and Design of Chemical Processes, 4th ed. Prentice Hall, 2012.

4. *Marino J., Rowley M.* Understanding SCA (Service Component Architecture). Addison-Wesley Professional, 2009.

5. *Сидорова Е. А.* Фактографический анализ текста в контексте интеллектуальных информационных систем // Тр. XVIII Байкальской Всерос. конф. «Информационные и математические технологии в науке и управлении» / Ин-т систем энергетики им Л. А. Мелентьева СО РАН. Иркутск,2013. Т. 3. С. 79–85.

6. *Ménard P. A., Ratté S.* Concept extraction from business documents for software engineering projects // Automated Software Engineering. Springer, 2016. Iss. 4.

Материал поступил в редколлегию 15.02.2018

K. G. Vysotski [1], **Yu. A. Rodina** [2], **E. A. Sidorova** [3]

[1] *DataWorks Ltd.*
2/2 Academician Lavrentiev Ave., *Novosibirsk*, *630090*, *Russian Federation*

[2] Novosibirsk State University
1 Pirogov Str., *Novosibirsk*, *630090*, *Russian Federation*

[3] *A. P. Ershov Institute of Information Systems SB RAS*
6 Academician Lavrentiev Ave., *Novosibirsk*, *630090*, *Russian Federation*

kostya19494@gmail.com, *julia.rodina97@gmail.com*, *lsidorova@iis.nsk.su*

ATTRIBUTE ASSEMBLY LINE MODEL FOR INTERACTIVE VISUALIZATION OF OBJECT INTERCONNECTIONS

In this work we define a model for visualization of interconnections of objects and their attributes in different processes. An abstract universal component of graphic user interface is developed

based on the defined model. The component is also proved to work on applied task of fact extraction from documents.

Keywords: graphic notation, graphic model, diagram, process visualization, object model, object interconnections, fact schemes

References

1. Cheremnykh S. V., Semyonov I. O., Ruchkin V. S. Modeling and analysis of systems. IDEF technologies: workshop. Moscow, Finances and Statistics, 2006. (In Russ.)

2. Couper J. R. et al. Chemical Process Equipment: Selection and Design. 2nd ed. Gulf Professional Publ., 2005.

3. Turton R. et al. Analysis, Synthesis and Design of Chemical Processes. 4th ed. Prentice Hall, 2012.

4. Marino J., Rowley M. Understanding SCA (Service Component Architecture). Addison-Wesley Professional, 2009.

5. Sidorova E. A. Factographic text analysis in context of intellectual information systems. XVIII Conference "Information and mathematic technologies in science and management". L. A. Melentyev Energy Systems Institute SB RAS. Irkutsk, 2013. (In Russ.)

6. Ménard P. A., Ratté S. Concept extraction from business documents for software engineering projects. *Automated Software Engineering*, Springer, 2016. 4th issue.

For citation:

Vysotski K. G., Rodina Yu. A., Sidorova E. A. Attribute Assembly Line Model for Interactive Visualization of Object Interconnections. *Vestnik NSU. Series*: *Information Technologies*, 2018, vol. 16, no. 1, p. 39–49. (In Russ.)