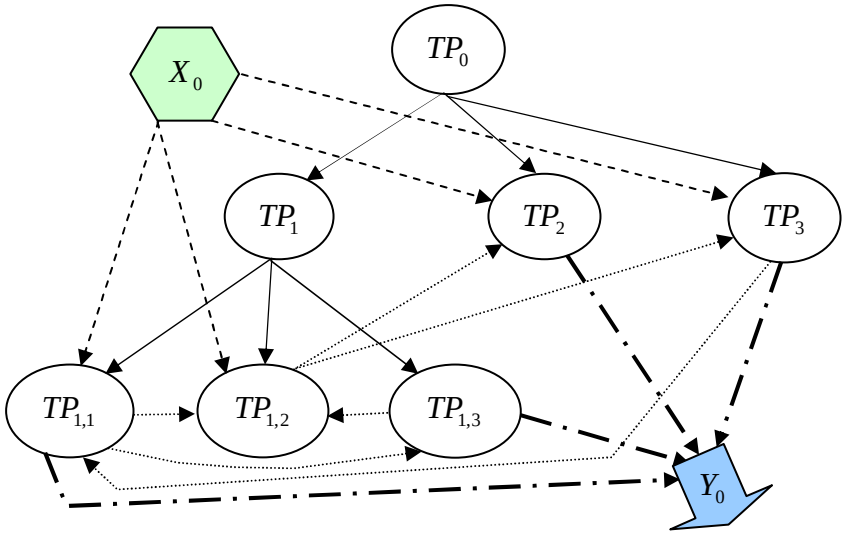

Рис.3. Иллюстрационный пример онтологии технологии

Разработанный метод обеспечивает возможность разработки эффективных программных средств машинного оперирования технологическими знаниями в целях решения широкого круга задач. Это в первую очередь задачи:

- построения структурированных хранилищ описания технологий;
- осуществления поиска (подбора) наиболее подходящих технологий в широком диапазоне возможных ситуаций;
- проведения экспертных исследований разрабатываемых технологий;
- разработки учебно-методических и тренажерных комплексов.

Представляется, что использование онтологических моделей описания технологий будет способствовать разрешению актуальных проблем автоматизированного измерения и сравнения существующих, а также синтеза перспективных технологий.

ЛИТЕРАТУРА

1. *Пименов, В.И.* Формализация технологических процессов при построении обучающих систем // Дизайн. Материалы. Технология. – 2007. – №1(2). – С. 95–99.

2. *Кудряшова, Т.Б.* Техническая теория. Специфика технического и технологического знания. /Электронный ресурс Кудряшовой Т.Б. Режим доступа: http://kudr-phil.narod.ru/aspirant/6texhist.html (дата обращения 14.05.2011).

3. *Воронов, М.В.* Система формализации технологических знаний //Моделирование и анализ данных: Труды факультета информационных технологий МППГУ. – М.: РУСАВИА, 2009. – Вып.4. – С. 4-18.

4. *Антонов, И.В.* Метод построения онтологии предметной области. /Антонов И.В., Воронов М.В. // Вестник СПГУТД.- 20010.- №2. - С.28-32.

РАЗРАБОТКА СИСТЕМЫ ИССЛЕДОВАНИЯ ДИНАМИКИ ДАННЫХ И МЕТАДАННЫХ ДЛЯ МОДЕЛИ РЕЛЯЦИОННОГО ТИПА

Л.Д. Шумский
Национальный исследовательский ядерный университет «МИФИ»
Москва

ВВЕДЕНИЕ

При сохранении скорости увеличения объемов информации, хранимой в цифровом виде, необходимость анализа и контроля этих данных также будет расти. Большая часть данных и метаданных хранится, на данный момент, в реляционных базах данных, основанных на модели Э. Кодда [1].

Серьезных оснований полагать, что сложившаяся ситуация может радикальным образом измениться в ближайшем будущем нет, следовательно ориентироваться необходимо именно на этот тип баз данных. Для современных, промышленных СУБД разработаны достаточно мощные средства мониторинга такие как: Primos [13], Ignite [14], Zero Impact Enterprise и Performance Monitor [15].

К сожалению, современные средства обладают рядом существенных недостатков. Во-первых, хотя эти системы позволяют находить тренды запросов, определять аномальности и предпринимать некоторые проактивные действия, они лишены возможности делать какие-либо логические выводы. Весь анализ необходимо производить администратору базы данных, что в условиях сверхбольших объемов данных не всегда возможно. Во-вторых, количество метаданных увеличивается даже быстрее, чем количество самих данных, а большинство современных систем либо вовсе игнорируют метаданные, либо работают лишь с крайне незначительной их частью. Наконец, данные системы достаточно жестки, они не поддаются гибкой настройке, которая может потребоваться для детального исследования.

Отдельно следует отметить, что все существующие системы связываются не с концептуальной схемой базы данных, а с физической. Это также является минусом т.к. делает несравнимыми результаты для различных физических схем, отражающих одно концептуальное представление. Кроме того, это отдаляет изучение изменения базы данных от изучения изменений в предметной области.

Для избавления от перечисленных недостатков предлагается разработать систему исследования динамики данных обладающую следующими свойствами. Во-первых, данная система должна быть привязана к концептуальной модели предметной области, а не к конкретной базе данных. Во-вторых, такая система должна иметь возможность описывать и обрабатывать утверждения, касающиеся динамики.

В данной работе предлагается следующий путь построения подобной системы – разрабатываются модели описания динамики и концептуальной схемы предметной области. Разработанные модели связываются между собой и с реляционной моделью Кодда. За счет такого подхода система позволит формулировать правила на внутренних языках описания концептов и динамики. Эти правила относятся к предметной области, а значит, сразу могут быть сформулированы аналитиками. За счет связи с реляционной моделью, эти правила могут быть транслированы в вид, понятный для реляционной базы данных.

В последующих разделах будут представлены основные положения, касающиеся моделей описания динамики и концептуальной модели и их связей. Отдельно будут рассмотрены логический вывод в модели и сценарии ее функционирования.

МОДЕЛЬ ОПИСАНИЯ ДИНАМИКИ

Основным назначением данной модели в разрабатываемой системе является формальное описание динамических процессов, т.е. процессов имеющих характер протяженных или возникающих во времени.

За основу при построении нашей модели мы возьмем модель, описанную в [2], однако нам придется ее несколько изменить, с учетом специфики задачи.

Для того чтобы строить логический вывод и формулировать правила о временных изменениях необходимо определить модель, с помощью которой мы будем описывать время. Время мы будем описывать парой $\mathcal{T} = (T, <)$, где элементы T – это временные точки (состояния или моменты времени), а ‘<’ – это отношение предшествования. Отношение ‘<’ – иррефлексивно и транзитивно [3]. Задав время описанным таким образом, мы обеспечили его ацикличность. Следующие важные вопросы, касательно времени, которые необходимо обговорить – это, непрерывность и определенность времени.

Первый вопрос относится к тому, существует ли интервал конечный длины между любой парой событий. В задачах статистики и оценки, к которым относится и рассматриваемая, возможно применить модель дискретного времени, т.е. модель в которой выполняется (3) для любых s_i – моментов времени

$$\forall s_1 s_2 (s_1 < s_2 \rightarrow \exists s_3((s_1 < s_3 \leq s_2) \,\&\, \forall s_4((\, s_4 \leq s_1 < s_3 \leq s_2) \vee (s_1 < s_3 \leq s_4 \leq s_2)) \,\&\, (d(s_3) - d(s_1) = \mathcal{d}))). \quad (1)$$

Величину $\mathcal{d}$ мы будем называть интервалом дискретности.

Второй вопрос относится к определенности времени или каких-либо его отрезков. В данном вопросе мы будем придерживаться точки зрения "открытости будущего" – в следующий момент времени, может произойти более одного события. Т.е. мы будем считать, что прошлое текущего момента единственно, хотя может быть и неизвестно, а будущее не определено и, вообще говоря, может быть любым. Данная концепция называется "ветвящееся будущее".

Изменения, происходящие в модели и ее состояния во времени будут отслеживаться с помощью механизма событий и фактов. Подобный механизм позволяет уйти от необходимости вводить специальную функцию означивания, для получения значения концепта в конкретный момент времени.

Подобная функция присутствует только неявно в определении элементарных фактов и событий в виде специальных конструкторов.

Фактом мы будем считать некоторое фиксированное состояние системы, с которым мы можем сравнивать текущее состояние в каждый момент времени. Факт может быть представлен экстенсионально – интервалами истинности, в таком случае производится автоматическое выучивание факта [4], или интенсионально – набором свойств системы, которые определяют данный факт. Для фактов определен набор элементарных конструкторов, для формулирования фактов о свойствах концептуальной модели, и сложных конструкторов для формулирования высокоуровневых фактов [5]. Минимальное время истинности факта соответствует интервалу дискретности и встречается тогда, когда факт истинен в момент s и ложен в момент s + d.

Под элементарным событием мы будем понимать некоторое изменение характеристик наблюдаемой системы. Под составным событием мы будем понимать событие, связанное с прочими событиями или фактами.

КОНЦЕПТУАЛЬНАЯ МОДЕЛЬ

Модель «состояние-контекст-свойство», которая будет использоваться для описания, основана на модели, описанной в [6]. В данной модели концепт C представляется следующим образом:

$C = (\sum, \mathcal{M}, \mathcal{L}, \mu, \nu)$

Здесь экстенсионал концепта представляется с помощью $\sum$ – множество состояний концепта и $\mathcal{M}$ – множество контекстов концепта. Интенсионал концепта – это $\mathcal{L}$ – множество свойств концепта и две функции μ и ν. Под состояниями концепта мы будем понимать его конкретные примеры, под контекстами некоторые дополнительные, внешние, условия, которые могут изменить текущее состояние концепта. Под свойствами концепта мы будем понимать некоторые конкретные значения соотнесенные с тем или иным состояние концепта. Необходимо отметить, что свойство применимо не для каждого состояния концепта, а только для его некоторого подмножества. Функция $\mu: \mathcal{P}(\sum) \rightarrow \mathcal{M} \rightarrow \mathcal{P}(\sum)$ определяет состояние концепта в контексте в зависимости от текущего состояния. Функция $\nu: \sum \times \mathcal{L} \rightarrow \{0, 1\}$ определяет применимость свойства к некоторому состоянию. Пустое значение и неприменимость свойства не являются в реляционной базе данных одним и тем же [7].

Выбор данной модели как основной обусловлен следующими соображениями. Основное преимущество модели подобного типа перед остальными, например дескрипционной логикой [8] или интенсиональными логиками [9, 10], заключается в том, что в ней изначально присутствуют элементы динамики, в виде смены состояний концепта, последовательных смен контекстов и изменения функций весов свойств и вероятности состояний. Во-вторых, модель подобного типа сравнительно проста – она может быть создана на основании уже существующей базы данных и не требует детального изучения предметной области, т.к. легко корректируется. В-третьих, подобная модель достаточно гибка – есть возможность изменять и усложнять функции μ и ν, которые изначально могут быть тривиальными, а также расширять множества контекстов и свойств, тем самым точнее и полнее описывать предметную область. Кроме того, модель данного типа позволяет сравнительно легко создавать концепты «на ходу», т.к. для этого, фактически, достаточно только определить множество состояний.

Концептам данной модели соответствую сущности предметной области и существенные отношения. Атрибутам сущностей соответствуют свойства концепта, причем простым атрибутам соответствуют простые свойства, составным атрибутам – списковые свойства. Связь концептов между собой осуществляется с помощью свойств концептуальной связи. Значением такого свойства для состояния концепта *A* является состояние концепта *B*.

Все концепты модели входят в метаконцепт «Система», описывающий систему в общем. Состояниями данного концепта является набор всех концептов в текущем состоянии, свойствами – глобальные свойства системы.

Связь концептуальной модели с моделью описания динамики производится с помощью набора конструкторов элементарных фактов и событий.

Связь с реляционной моделью производится с помощью правил выбора состояний концепта, а также с помощью особого задания контекстов и свойств. Концептам могут быть поставлены в соответствие те таблицы или представления базы данных, которые соответствуют сущностям или идентифицируемым отношениям предметной области. Тогда состояниями концепта будут соответствующие строки таблицы или представления. Для задания нового концепта достаточно указать те критерии, которым должны удовлетворять строки, т.е. сформулировать запрос определенного вида.

ЛОГИЧЕСКИЙ ВЫВОД В СИСТЕМЕ

Система должна поддерживать два вида логического вывода. Во-первых, вывод о свойствах динамической составляющей модели – причинно-следственные связи между событиями и фактами. Во-вторых, вывод свойств концептуальной структуры – вывод с помощью правил Хорна.

Мы будем рассматривать два вида причинно-следственной связи: событие-событие и событие-факт. . Сначала опишем более простой вид причинно-следственной связи – связь между парой событий.

$$\begin{aligned}&ecause(p, e1, e2, rf, i) \rightarrow (Occ(s1, s2, e1)\rightarrow\\ &\quad(\forall ch(s2 \in ch \rightarrow \exists s3((s3 \in ch) \,\&\\ &\qquad(within\text{-}delay(s3, rf, i, s1, s2)) \,\&\\ &\qquad(\neg(TT(s2, s3, p)) \text{ or } \exists s4((s4 \in ch) \,\&\\ &\qquad\quad(Occ(s3, s4, e2))))))))\end{aligned}$$

Здесь и далее под within-delay будем понимать следующее – момент s наступает после моментов s1 и s2 с задержкой из интервала i, начиная с rf, под Occ – событие e происходит между двумя моментами времени, под TT – что факт p становится истинным хотя бы в один момент между двумя указанными. Все вместе выражение для ecause следует читать следующим образом – за событием e1 всегда следует событие e2 с задержкой из интервала i, начиная с момента rf, если только факт p не перестанет быть ложным.

Следующий вид причинно-следственной связи – связь между событием и фактом. Выражение причинно-следственной связи (pcause):

$$\begin{aligned}&pcause(p, e, q, rf, i, r1) \rightarrow (Occ(s1, s2, e)) \rightarrow\\ &\quad\forall ch((s2 \in ch) \rightarrow \exists s3((s3\in ch) \,\&\\ &\qquad(within\text{-}delay(s3, rf, i, s1, s2)) \,\&\\ &\qquad(\neg T(s3, p) \text{ or } (persist(s3, q, r1))))))\end{aligned}$$

Записанное выражение читается следующим образом – событие e всегда влечет за собой продолжение факта q на r1, с задержкой из интервала i, начиная с момента rf, если только факт p не перестанет быть ложным. Если факт находится в состоянии продолжения, ситуация, когда факт становится ложным, является исключительной и заслуживает отдельного отношения.

Правила причинной связи задаются таким образом, что могут нарушаться, как вследствие изначально неполного списка условий, так и из-за изменения предметной области, и это нормально в рамках данной модели. Когда возникает необходимость, мы можем отредактировать наши правила. Таким образом, модель оказывается «приблизительно» истинной, приближаясь к истине с каждым редактированием.

Второй вид логического вывода в модели – это вывод о свойствах концептов системы. Необходимо отметить, что концептуальная схема подчиняется правилу ограничения [11, 12], т.е. не существует никаких концептов (состояний, контекстов), существование которых не следует явно из существующих концептов (состояний контекстов). Логический вывод осуществляется с помощью правил Хорна, т.е. правил вида $a1 \wedge a2 \wedge \ldots \wedge an \rightarrow b$. В качестве рабочей памяти для правил Хорна будут использоваться значения свойств (выводимых или нет) специального «системного» концепта.

СЦЕНАРИИ РАБОТЫ СИСТЕМЫ

Правила в системе делятся на три вида:

1. Правила, касающиеся концептуальной схемы. В данную категорию входят утверждения о составе концептов, а именно об их состояниях, контекстах и свойствах;
2. Правила касающиеся динамических свойств модели. К этой категории относятся правила причинно следственной связи, рассмотренные в предыдущем разделе, а также записи составных событий и фактов;
3. Пользовательские правила (сценарии).

Функционирование системы определяется сценариями. В системе предусмотрено два вида сценариев – системный, глобальный, сценарий и пользовательские, локальные сценарии. Задачей локальных сценариев является дополнение и расширение глобального.

Системный, глобальный сценарий описывается следующей последовательностью действий:

1. Наступает момент времени, кратный d;
2. Выполняется логический вывод для заполнения всех выводимых свойств модели;

3. Выполняются все пользовательские сценарии в определенном порядке. Происходит заполнение выводимых переменных, проверка пользовательских фактов, установка текущих контекстов и вычисления значений пользовательских переменных. Выполнения действий на этом шаге не производится;
4. Проверяются все события, регистрируются все события, которые произошли;
5. Проверяется истинность всех фактов;
6. Обрабатываются правила причинно-следственной связи;
7. Выполняются все пользовательские сценарии в определенном порядке. Выполнение действий;
8. Регистрируется финальное состояние системы.

Этот сценарий является жестко заданным и последовательность и содержание шагов в нем не может быть изменено. Последовательность шагов выбрана в соответствии с внутренними зависимостями. Пользовательские сценарии представляют собой набор правил Хорна, антецедентом которых является набор предикатов (предикатных свойств системного концепта), а консеквентом набор новых значений свойств системного концепта или никоторое внешнее действие, называемое выходом сценария. Под выполнением пользовательского сценария понимается последовательное выполнение правил, в него входящих, антецедент которых истинен. Пользовательский сценарий завершает свое выполнение, когда ни у одного правила антецедент не истинен (ложен или не может быть вычислен) либо когда выполняется любой из выходов данного сценария.

ЗАКЛЮЧЕНИЕ

Задачей работы являлось создание системы исследования изменения данных и метаданных. Для достижения данной задачи были выполнены следующие шаги – были разработаны модели представления концептов и концептуальных зависимостей, а также модель описания динамики. Построенные модели были связаны между собой и с реляционной моделью Кодда. Были разработаны правила логического вывода в динамической модели и концептуальной структуре. Также были разработаны правила функционирования системы.

Для развития системы предполагается осуществить следующие шаги. Усилить логический вывод на динамической модели – дать системе возможность автоматически определять причинно-следственные связи между событиями и фактами. Повысить выразительную мощность концептуальной схемы – включить в нее под концепты, увеличить способы описания концептуальных зависимостей, расширить возможности логического вывода на концептуальной структуре. Отдельной, связанной, задачей является разработка языка описания концептов, правил и сценариев.

ЛИТЕРАТУРА

1. *Codd, E.F.* Further normalization of the data base relational model // IBM Research Laboratory. 31 August 1971.
2. *McDermont, D.* A Temporal Logic for Reasoning About Processes and Plans // COGNITIVE SCIENCE. 1982. №6. С.101-155.
3. *Venema, Y.* Temporal Logic // Lou ed Goble. The Blackwell Guide to Philosophical Logic. Blackwell Publishers. 2001.
4. *Valiant, L. G.* A Theory of the Learnable // Artificial Intelligence and Language Processing. 11 November 1984. №27. С. 436-445.
5. *Benthem, J. van.* Temporal Logic // Handbook of Logic in Artificial Intelligence and Logic Programming / Ch. Hogger & J. Robinson, eds. D. Gabbay. Oxford University Press. 1991.
6. *Aerts, D.* A theory of concepts and their combinations I: The structure of the sets of contexts and properties / Aerts D., Gabora L. M. // Kybernetes. 2005 №34, С. 151-175.
7. *Codd, E. F.* Does Your DBMS Run By the Rules? // ComputerWorld. 1985.
8. *Baader, F.* The Description Logic Handbook: Theory, Implementation, and Applications / Baader F, Calvanes D and McGuiness D L. Cambridge. Cambridge University Press. 2003.
9. *Fox, Ch.* Intensional first-order logic with types / Chris Fox, Shalom Lappin, Carl Pollard. 2003.
10. *Ronald, J. B.* Knowledge Representation and Reasoning / Ronald J. Brachman, Hector J. Levesque. Elsevier Inc. 2004.
11. *McCarthy, J.* Applications of Circumscription to Formalizing Common Sense Knowledge. Stanford: Computer Science Department Stanford University, 1986.
12. *McCarthy, J.* Circumscription - a Form of Nonmonotonic Reasoning. Stanford: Computer Science Department Stanford University, 1986.

13. *Официальный* сайт проекта Primos. [Электронный ресурс]. Режим доступа: http://www.primos.nl/. — Загл. с экрана.
14. *Официальный* сайт проекта Ignite. [Электронный ресурс]. Режим доступа: http://www.confio.com/. — Загл. с экрана.
15. *Официальный* сайт проектов Zero Impact Enterprise и Performance Monitor. [Электронный ресурс] Режим доступа: http://www.sqlpower.com/. — Загл. с экрана.

ТЕКСТ И МУЛЬТИМЕДИА В ЭЛЕКТРОННОЙ СЕМАНТИЧЕСКОЙ БИБЛИОТЕКЕ

М.В. Яковлева, А.К. Тен, В.М. Куглер
Свердловская областная универсальная научная библиотека им. В.Г. Белинского
Екатеринбург

ВВЕДЕНИЕ

Современные научные исследования и разработки предполагают наличие развитой системы справочников. Объектный подход и движение в направлении электронных семантических библиотек позволяют улучшить индексацию и поиск материалов.

ЧАСТЬ I. НА ПУТИ К ЭЛЕКТРОННОЙ СЕМАНТИЧЕСКОЙ БИБЛИОТЕКЕ

Признаки электронной семантической библиотеки (ЭСБ)

По мнению авторов, существенные признаки ЭСБ лежат в области модели мира, способов работы с ней и удобных интерфейсах пользователей:

А) Это хранилище объектов с их свойствами;

Б) Между объектами установлены связи и отношения;

В) Объекты можно искать и просматривать;

Г) К объектам присоединены традиционные формы: тексты, картинки, видео, аудио;

Д) Экранные интерфейсы пользователя понятны, не содержат специальных терминов и удобны;

Е) Универсальность: обеспечено размещение объектов любой сферы деятельности (области знаний).

Ж) ЭСБ может содержать ограничения на объекты, связи и прочее, дополняющие модель объектов;

З) Отдельные свойства, связи объектов могут вычисляться (выводиться) из других свойств и связей.

Представление объекта в ЭСБ определяется пунктами (А)-(З). Объекты характеризуются свойствами. Например, у пещер есть свойства «Доступность» и «Высотная отметка входа», а у людей – «Дата рождения». Возможности формировать у объекта требуемый набор свойств – одна из характеристик универсальности ЭСБ. Свойства (атрибуты) используются, чтобы изложить информацию, легко структурируемую. Обычно свойство, например, «Протяженность», используется как характеристика для многих объектов.

Между объектами могут существовать связи. Например, объект Иванов Иван Иванович может быть связан с объектом Фабрика «Одежда» свойством-ссылкой Директор, а объект Статья «История Москвы» с объектом Красная площадь свойством «О чем». В целом, объекты, представленные в ЭСБ, составляют сеть, и ознакомление с материалом может происходить переходами по именованным ссылкам от объекта к объекту. В узлах сети находятся не только документы, но большей частью объекты действительности – галактики, события, изобретения, люди и т.д.

Поиск объекта реализуется на основе его наименования, класса, к которому он относится, условий, накладываемых на значения свойств.

Экранные интерфейсы, то есть то, как пользователь видит представление объекта, позволяют передать логические связи объекта с другими объектами, значения его свойств, а также прикрепленные к нему традиционные информационные ресурсы. Пользователи с большим энтузиазмом встретят оформление объектов, похожее на представление страниц в традиционном вебе.

Универсальность определяется свойствами программного обеспечения. Например, в системах, основанных на Dublin Core [1], все сосредоточено на объектах типа документ: книгах, статьях и т.д. Чтобы индексировать содержание документов, необходима универсальная онтология [2].