4. Грани Агни Йоги. 1955 г. – Н.: Алгим, 2011. – 704 с. Режим доступа: http://www.roerich.com/zip3/grani_55.zip.
5. Гиндилис, Л.М. Сознание и его роль в мироздании: научно-философские и метанаучные аспекты [Текст] / Л.М. Гиндилис // Материалы X-ой междисциплинарной научной конференции «Этика и Наука Будущего» – Сознание как творящая сила Космоса. – М.: Дельфис, 2011. – С. 5-13.
6. Теория и практика дистанционного обучения. Учеб. пособие для студ. высш. пед. учеб. заведений [Текст] / Е.С. Полат, М.Ю. Бухаркина, М.В.Моисеева; под ред. Е.С. Полат. – М.: Издательский центр «Академия», 2004. – 416 с.
7. Блаватская, Е.П. Что есть Истина. Режим доступа: http://www.theosophy.ru/lib/hpb-ist.htm.
8. Гончаренко, С.У Педагогічні дослідження: Методологічні поради молодим науковцям. – Київ-Вінниця: ДОВ «Вінниця», 2008. – 278 с.
9. Вестник Орифламма. Режим доступа: http://www.roerich.com.
10. Шапиро, Д.И. Виртуальная реальность и проблемы нейрокомпьютинга [Текст] / Д.И. Шапиро – М.: РФК-Имидж Лаб, 2008. – С. 54-55.
11. Теслер, Г.С. Новая кибернетика [Текст] / Г.С. Теслер. – К.: Логос, 2006. – 2004. – с. 57-58.
12. Вершинина, Н.А. История педагогики как науковедческая дисциплина.Режим доступа:ftp://lib.herzen.spb.ru/text/vershinina_8_30_56_68.pdf.
13. Мельников, Л.Н. Виртуальная реальность и космическое сознание [Текст] / Л.Н. Мельников // Дельфис. - 2006. - №2(46).– С.111-112.
14. Джура, С.Г. К вопросу многомерности сознания [Текст] / С.Г. Джура // Материалы 10-ой междисциплинарной научной конференции «Этика и Наука Будущего» – Сознание как творящая сила Космоса. – М.: Дельфис, 2011. – С. 30-39.
15. Подласый, И.П. Энергоинфформационная педагогика. [Текст] / И.П. Подласый. – М.: Дата Сквер, 2010. – 424 с.
16. Романовский, А.Г. Духовная составляющая как основной фактор формирования национальной гуманитарно-технической элиты [Текст] / А.Г. Романовский // Теория и практика управления социальными системами: философия, психология, педагогика и социология, №2, 2009. – С. 15-20.

Пропонується методика застосування мови таблиць подій для специфікації і моделювання складних систем

Ключові слова: таблиця подій, мова таблиць подій, специфікація

Предлагается методика применения языка таблиц событий для спецификации и моделирования сложных систем

Ключевые слова: таблица событий, язык таблиц событий, спецификация

The methodology of the event tables language implementation for the specification and simulation of complex systems is presented

Keywords: event table, event tables language, specification

УДК 004.436.4

РАЗРАБОТКА МЕТОДОВ СПЕЦИФИКАЦИИ ИНФОРМАЦИОННЫХ МОДЕЛЕЙ СРЕДСТВАМИ ЯЗЫКА ТАБЛИЦ СОБЫТИЙ

О.Н. Кулешова
Аспирант
Кафедра кибернетики и вычислительной техники
Севастопольский национальный технический университет
ул. Университетская, 33, Севастополь, Украина, 99053
Контактный тел.: 067-652-43-41
E-mail: v_olgo4ka@inbox.ru

В настоящее время актуальной задачей становится развитие и внедрение в практику сложных систем разнообразных научных направлений, таких как информационные технологии, кибернетика, инженерия знаний, методы оптимизации, анализа и моделирования в различных областях.

Результатом этого является создание предпосылок для построения высокоэффективных систем по обработке и использованию знаний для решения широкого круга задач.

В условиях современного научно-технического прогресса сложные системы эксплуатируются в динамически изменяющейся среде, что сопровождается изменением условий, ограничений и даже целей реализуемых процессов. Это приводит к тому, что разработка адекватных и полных моделей отстаёт от потребностей в решении задач поставленных в рамках той или иной предметной области. Сложность описания объектов обуславливает необходимость разработки методов и средств спецификации и моделирования сложных си-

© О.Н. Кулешова, 2012

стем, основанных на объединении идей дискретного описания и событийного моделирования. Этот подход позволяет проводить моделирование, которое можно отнести к одной из разновидностей имитационного моделирования, и осуществлять на этой основе синтез сложных систем [1].

Создание современных сложных систем требует применения различных моделей представления знаний, обладающих сбалансированными характеристиками по выразительности их языка с одной стороны и приемлемой вычислительной сложностью с другой. К моделям представления знаний предъявляется широкий спектр требований, включающий возможности декларативного задания с использованием средств визуального конструирования, поддержку разнородных входных параметров, возможности верификации и оптимизации модели, и наличие эффективных алгоритмов моделирования и анализа.

Табличные модели являются удобной формой представления экспертных знаний. Язык таблиц решений относится к классу формальных языков, характеризующийся непроцедурной и наглядной формой описания процесса принятия решения, а так же автоматизации процессов проверки корректности (полноты, непротиворечивости, неизбыточности), оптимизации и трансляции табличной модели в программы поиска решений. Таблицы решений получили широкое распространение при автоматизации процессов принятия решений, диагностики и контроля в имитационном моделировании. В то же время, язык таблиц решений представляет собой не достаточно гибкое и эффективное средство представления сложной логики в решении задач проектирования, анализа и моделирования систем высокой сложности [2].

Используя как основу язык таблиц решений, предлагается разработка более универсального языка таблиц событий [3], средства которого позволят компактно и наглядно описывать процессы поведения сложных систем в различных предметных областях, а так же предоставят возможности эффективного моделирования и анализа.

Язык таблиц событий позволяет задать набор правил (событий), определяющих конкретные последовательности действий, при заданных условиях (правил) $R = \{R_p\}; p = \overline{1,n}$ на основе пропозициональной логики над множеством условий

$$E = \{E_i\}; i = \overline{1,m}$$

и действий

$$A = \{A_i\}; j = \overline{1,k} : (R_p) c_1^p \wedge c_2^p \wedge \ldots \wedge c_m^p \Rightarrow A_1^p, A_2^p, \ldots, A_k^p .$$

Множество условий и действий принимаются общими для всех правил в таблице событий. Такой набор правил может быть легко представлен в табличной форме.

Базовая спецификация ТС имеет следующий вид:

<ID,E, A, R, D, C, P, Cond, Act>

Здесь $E = \{E_i\}; i = \overline{1,m}$ - множество условий, описывающих параметры выбранной предметной области;

$A = \{A_i\}; j = \overline{1,k}$ - множество действий;

$SubT = \{SubT_i\}; j = \overline{1,k}$ - множество ссылок на таблицы решений, описывающих действия;

$R = \{R_p\}; p = \overline{1,n}$ - множество решающих правил, называемых правилами решений, определяющими конкретные действия, при заданных условиях;

$D = \{D_l\}; l = \overline{1,z}$ - множество решающих правил, определяющих действие «иначе»;

$C = \{C_q\}$, $q = \overline{1,n}$ - множество векторов условий, где $\{C_q\} = \{c_1^q, c_2^q, \ldots, c_m^q\}$;

$P = \{P_v\}, v = \overline{1,n}$ - множество действий, где :

$\{P_v\} = \{a_1^v, a_2^v, \ldots, a_k^v\}$;

$Cond = \|c_{ix}\|, Act = \|a_{jx}\|$ - матрицы взаимосвязей множеств векторов данных и векторов действий.

Вектор значения условий $S = \{s_i\}$, $i = \overline{1,m}$, называемый входным вектором (вектором состояния) описывает некоторое состояние предметной области. Пара матриц Cond и Act дают матричное соответствие между входными векторами состояний, описываемых как комбинация значений условий или так называемыми векторами условий и выполняемыми действиями – сценариями. Алфавит матрицы $Cond = \|c_{ix}\|$ фиксирован

$$c_{ix} = \begin{cases} \Omega T, \text{ если условие } e_i \text{ истинно;} \\ \Omega F, \text{если условие } e_i \text{ ложно;} \\ \times, \text{ если условие } e_i \text{ безразлично} \end{cases},$$

где ΩT и ΩF - парные истинностные значения (могут обозначаться как 0/1, Y/N, TRUE/FALSE, ДА/НЕТ, ИСТИНА/ЛОЖЬ и т.п.), а “x” является символом безразличия.

Действия в таблице событий, описывают атомарные сценарии в терминах предметной области, порядок выполнения которых определяется содержащимися в матрице $Act = \|a_{jx}\|$ целочисленными значениями

$$a_{jx} = \begin{cases} a \in (1,\ldots,k), \text{ порядок выполнения действий,} \\ \quad \text{если правило } R_x \text{ приводит к} \\ \quad \text{выбору решения } P_i; \\ 0 \text{ или пустое значение (NULL), в} \\ \text{противном случае.} \end{cases}$$

Для $\forall(a_{jx} \neq 0)$, значение a_{jx} интерпретируется как очерёдность (приоритет) выполнения действия при активации правил.

Правило $R_p =< C_p, P_p >$ называется простым, если вектор C_p не содержит символов безразличия “x”, в противном случае правило R_p называется обобщённым. Такие правила распознают несколько входных векторов и характеризуются одним или несколькими символами безразличия “x”, в матрице Cond, стоящих в позициях значений условий, несущественных для применимости данного правила. Применение обобщённых правил позволяет существенно сократить число правил в таблице событий (т.к. каждое обособленное правило представляет собой объединение нескольких простых правил), но образует дополнительные сложности при обработке таблицы событий.

Вектор $S=\{s_i\}$, $i=\overline{1,m}$ распознаётся таблицей событий, если $\exists p((C_p = S) \vee (C_p \subset S))$, при этом вектор S удовлетворят правилу R_p (или правило R_p применимо в состоянии S), и в записи это имеет следующий вид $S \to R_p$. Для реагирования на входные векторы, к которым не применимо ни одно правило $R=\{R_p\}; p=\overline{1,n}$, в таблицу событий вводится правило “Иначе”(“Else”), которое определяется правилом D, заданным как

$$(D =< \varnothing, P_{n+1} >) \vee (D =< (\{D_l\}, l=\overline{1,z}), P_{n+1} >)$$

или незаданными векторами условий и одним единственным сценарием P_{n+1}. Использование не пустого множества векторов условий в случае “Иначе” обусловлено большей наглядностью в процессе разработки таблицы событий, для наиболее полного и корректного описания предметной области с помощью средств языка таблиц событий.

Правило “Иначе” может рассматриваться как замыкание модели, то есть как средство пополнения и адаптации, обеспечивающее универсальную реакцию на неучтённые входные векторы.

Таблицы событий являются средством задания соответствия между значениями элементов некоторого конечного множества условий (событий), определяющих состояние предметной области, и последовательностями конечного множества действий (сценариями), определяющих реакцию на эти события.

Две таблицы событий

TE^1:<ID^1,E^1, A^1, R^1, D^1, C^1, P^1, $Cond^1$, Act^1 >

и

TE^2:<ID^2,E^2, A^2, R^2, D^2, C^2, P^2, $Cond^2$, Act^2 >

называются эквивалентными ($TE^1 \sim TE^2$), если они задают одинаковое отображение условий на действия.

$$\forall i,j(i \neq j \to ((C^2 = C^1_i \subset C^1_j) \vee (C^2 = C^1_i \cup C^1_j)),$$

$$P^2 = P^1_i = P^1_j)$$

К эквивалентным преобразованиям относятся: удаление/добавление избыточных правил; объединение правил с получением обобщённого правила; замена пересекающихся правил непересекающимися.

Условия могут быть разделены по типу на независимые и взаимоисключающие. Так же, условия можно разделить на сложные и простые:

– простое условие – это условие, которое сводится к проверке единственного элемента данных, т.е. проверка сводится к проверке равенства элемента данных 1 или 0;

– сложное условие – это условие, которое включает в себя функцию условия (логическую функцию), в результате которой будет получен результат в виде значения элемента данных равного 1 или 0.

Действия в таблице событий можно разделить на сложные и простые.

Сложные образуют переходы (связи) между таблицами событий в системе. Можно выделить следующие межтабличные связи:

– одиночный переход: выполнение сценария приведёт к единственному переходу к следующей таблице событий;

– переход возврат: выполнение сценария приведёт к повторному запуску той же таблице событий;

– разветвлённый переход: выполнение сценария приведёт к последовательным или параллельным переходам более чем к одной таблице событий.

– цикл: выполнение сценария приведёт к переходу к таблице событий, из которой был произведён переход к текущей таблице событий, напрямую или вследствие переходов через другие таблицы событий.

Простые действия представляют собой переход к внешним объектам, таким как функция или вывод сообщения, а так же прерывание и завершение.

При активации сценария таблицы событий производится один из возможных переходов с последующим возвратом к исходной таблице, пока не будут выполнены все действия, предусмотренные сценарием, затем будет произведён возврат к таблице событий, из которой был произведён переход к текущей таблице событий, за исключением прерывания, в этом случае будет произведён возврат к предыдущей таблице со сбросом всех текущих состояний. Если имеется необходимость разделения большой таблицы событий на несколько более мелких, то это может быть реализовано несколькими способами:

– разбиение по правилам: может быть применено с целью удовлетворения ограничения на количество правил таблицы событий; исходная таблица событий редуцируется на две таблицы событий, содержащие все условия и действия исходной таблицы событий, а правила распределяются между ними. Далее в правило “Иначе” первой таблицы событий заносится переход на вторую таблицу событий как единственный результат его выполнения, т.е. в первую таблицу событий включается дополнительное действие.

– разбиение по содержанию: условия и действия исходной таблицы событий распределяются по новым таблицам событий, связанных между собой переходами.

Язык таблиц событий включает следующую символику:

– идентификаторы таблиц событий, условий, действий, данных;

– имена функций условий, имена функций действий;

– символы типов переходов;

– символы матриц Cond и Act;

– идентификаторы векторов условий, действий, правил;

– символы описания функций и сообщений.

Идентификатор таблицы событий: TEнатуральное_число (Пример: TE1, TE20, TE184).

Идентификатор условия: eнатуральное_число (Пример: e1, e3, e12, e156).

Идентификатор действия: Aнатуральное_число (Пример: A2, A16, A287).

Идентификатор данных: Dtнатуральное_число (Пример: Dt3, Dt25, Dt678).

Имя функции условия: FCond.имя_функции, где имя_функции сформировано из букв латинского алфавита, цифр (только не на первой позиции имени функции), символов >, <, = (Пример: FCond.more, FCond.<).

Имя функции действия: FAct.имя_функции, где имя_функции сформировано из букв латинского алфавита, цифр (только не на первой позиции имени функции), символов +, -, /,*, % (Пример: FAct.sum, FCond.%).

Типы переходов:

– ! – переход к таблице событий;

– : – переход к функции;

– * – переход к выводу сообщение;

– . – переход на выполнение прерывания.

Типы условий:

– ? – сложное условие;

– # – простое условие;

Символы матрицы Cond: 1,0,x.

Символы матрицы Act: NULL(пустое значение), 0, натуральные числа.

Символы функций условий : [;]; (;); , .

Символы функций действий : [;]; {; }; , .

Символы сообщений: “ ; ” ; « ; » .

Все идентификаторы таблицы событий, условий, действий, данных; имена функций условий, имена функций действий; символы типов переходов; символы матриц Cond и Act; идентификаторы векторов условий, действий, правил; символы описания функций и сообщений должны быть записаны в соответствующие ячейки таблицы событий.

Общий вид заполнения таблицы событий представлен в табл. 1.

Функции условий имеют следующий синтаксис: [идентификатор_элемента_условия_1],[идентификатор_элемента_условия_2], ... ,(имя_логической_функции),[идентификатор_результата].

Функции действий имеют следующий синтаксис: [идентификатор_аргумента_1],[идентификатор_аргумента_2],..., {имя_функции},[идентификатор_результата].

Сообщение имеет следующий синтаксис: «текст_сообщения», или «текст_сообщения».

Таблица 1

Общий вид заполнения таблицы событий

Идентификатор ТС				R_1	...	R_n	D		
				C_1	...	C_n	D_1	...	D_l
E	Идентификатор условия	?/#	Функция условия/ пустое значение	1/0/x	...	1/0/x	1/0/x	...	1/0/x
	Идентификатор условия	?/#	Функция условия/ пустое значение	1/0/x	...	1/0/x	1/0/x	...	1/0/x
	Идентификатор условия	?/#	Функция условия/ пустое значение	1/0/x	...	1/0/x	1/0/x	...	1/0/x
				P_1	...	$P_т$	P_{n+1}		
A	Идентификатор действия	!/:/*/.	Функция действия/ Сообщение/ Идентификатор ТС/ Пустое значение	пустое значение/0/ натуральное число	...	пустое значение/0/ натуральное число	пустое значение/0/ нату-ральное число		
	Идентификатор действия	!/:/*/.	Функция действия /Сообщение/ Идентификатор ТС/ Пустое значение	пустое значение/0/ натуральное число	...	пустое значение/0/ натуральное число	пустое значение/0/ нату-ральное число		

Язык таблиц событий, представляя собой эффективное и наглядное средство проектирования, моделирования и анализа, даёт широкие возможности для автоматизации разработки и исследования сложных процессов в различных областях жизни, например, промышленном производстве, экономике, экологии, информационных технологиях.

В дальнейших исследованиях предполагается разработка графических расширений языка таблиц событий и интерфейсов для эффективного использования средств языка в специальной программной системе [3].

Литература

1. Мартынова, М.А. Информационные технологии для проведения ситуационного анализа деятельности кредитных организаций [Текст] / М.А. Мартынова, В.А. Фатуев, Э.Г. Годынский // Известия ТулГУ. Серия: Выч. Техника. Автоматика. Управление. - Вып. 7. - Тула: ТулГУ, 2001. - С 82-86.
2. Виноградов, О.В. Современные программные средства поддержки принятия решений на основе аппарата таблиц событий / О.В. Виноградов, А.П. Еремеев [Текст] / О.В. Виноградов, А.П. Еремеев // Интеллектуальные системы. Колл. Монография. Выпуск 3 / Под ред. В.М. Курейчика. - М: Физматлит, 2009 - С.140 - 155.
3. Кулешова О.Н. Спецификация распределенных систем на основе таблиц событий [Текст] / О.Н. Кулешова, Ю.К. Апраксин // Оптимизация производственных процессов: сб. науч. тр. Вып. 13. - Севастополь, 2011 - С. 52 - 57.