УДК 004.434:004.94

О НОВОМ ЯЗЫКЕ СЦЕНАРИЕВ ДЛЯ ОБЪЕКТНОГО ИМИТАЦИОННОГО МОДЕЛИРОВАНИЯ

ПОПОВ А.С.

Описывается новый язык сценариев (scripting language) для имитационного моделирования поведения объектов, рассматриваемых как функциональные элементы проектируемой организационно-технической или информационной системы. Язык используется при создании программных модулей, предназначенных для количественного анализа моделируемых функциональных процессов, CASE-средств, автоматизирующих системологический анализ.

Возникновение и развитие объектно-ориентированной парадигмы разработки программного обеспечения и CASE-технологии увеличили ответственность, трудоемкость, сложность и значимость этапов анализа и проектирования программных систем. Это, в свою очередь, привело к необходимости повысить эффективность методов анализа и моделирования сложных динамических систем различной природы. Для решения последней задачи в настоящее время используются различные диаграммные техники, большинство из которых приняты в качестве государственных или международных стандартов: 3-View Modeling (DFD, ERD, STD); SADT; SSADM; Unified Modeling Language (UML).

Данные техники (методы) предназначены для моделирования разрабатываемых систем как в статике, так и в динамике. Однако большинство этих методов (3VM, SADT, SSADM) не согласованы с современной объектной парадигмой [1, 2]. Методы, разработанные в рамках самой объектной парадигмы, предназначены для специфицирования, визуализации и документирования различных статических и динамических характеристик разрабатываемых систем и пока не имеют собственных средств создания имитационных моделей [3].

Известно, что “анализ характеристик процессов функционирования больших систем с помощью только аналитических методов исследования наталкивается обычно на значительные трудности, приводящие к необходимости существенного упрощения моделей либо на этапе их построения, либо в процессе работы с моделью, что может привести к получению недостоверных результатов”. Кроме того, проведение “экспериментов с имитационными моделями больших систем позволяет не только анализировать их характеристики, но и решать задачи структурного, алгоритмического и параметрического синтеза таких систем” [4, с. 8-9].

Названные обстоятельства делают необходимым создание методов анализа и моделирования программных систем, согласованных с объектным подходом и обладающих средствами имитационного моделирования.

Подобная попытка предпринята в научно-учебной лаборатории приобретения знаний ХНУРЭ путем создания объектно-ориентированного метода системологического анализа и проектирования OMSAD (Object-oriented Method Systemology Analysis and Design). Заключительной фазой данного метода является создание имитационной модели проектируемой системы на основе предварительно построенной объектной модели. При этом, если предварительные фазы (построение концептуальной классификационной модели на основе базовой иерархии классов и построение объектной модели, объединяющей в себе object diagram, collaboration diagram и use case diagram UML) могут быть выполнены вручную, то для создания имитационной модели необходим соответствующий инструментарий. В настоящее время в НУЛ ПЗ разрабатывается очередная версия такого CASE-инструментария.

Задача имитации функционирования системы на ее объектной модели приводит к необходимости моделирования различных функциональных (аналитических, логических и т.д.) зависимостей. Для решения данной задачи в рамках проекта был разработан специальный язык сценариев (scripting language), а также компилятор и интерпретатор для него. Этот язык позволяет описывать (и визуализировать!) поведение объектов, рассматриваемых как функциональные элементы разрабатываемой системы, на этапе динамического (имитационного) моделирования.

Программа, написанная на данном языке, состоит из произвольного числа операторов, которые разделяются точкой с запятой. Они выполняются последовательно. Оператором является выражение, которое может содержать присваивания и вызовы функций. Кроме того, существуют условные операторы и составной оператор, который дает возможность использовать несколько операторов в том месте, где предполагается использование одного.

Язык позволяет описывать переменные и константы целого, вещественного и булевского типа. Данные целого и вещественного типа могут принимать значения от 1.7E-308 до 1.7E+308. Перед вычислением все операнды выражений преобразуются к вещественному типу. Если в выражении встречается присваивание, то оно преобразуется к типу левого операнда. Имя переменной может состоять из произвольного количества букв латинского алфавита и цифр. Первый символ имени не должен быть цифрой. Имя переменной не может быть идентично ключевому слову или имени функции. В языке зарезервировано пять констант: число “пи”, число “е”, “истина”, “ложь” и “бесконечность”.

Программа может содержать комментарии, которые могут быть вставлены между любыми двумя словами программы. Это произвольные последовательности символов, начинающиеся символами /* и оканчивающиеся */, либо начинающиеся // и оканчивающиеся символом конца строки.

Язык поддерживает все основные логические (И, ИЛИ, НЕ, исключающее ИЛИ) и арифметические (сложение, вычитание, умножение, деление, возведение в степень, инкремент, декремент, остаток от деления и др.) операции, а также операции отношения (больше, меньше, равно, не равно и др.). Список операций представлен в табл. 1.

Таблица 1. Операции

Название	Синтаксис
Мультипликативные операции:	
Умножение	Выражение * выражение
Деление	Выражение / выражение
Остаток от деления	Выражение % выражение
Возведение в степень	Выражение pow выражение
Квадрат числа	** выражение
Аддитивные операции:	
Сложение	Выражение + выражение
Вычитание	Выражение – выражение
Арифметическое отрицание	– выражение
Операции отношения:	
Меньше чем	Выражение < выражение
Больше чем	Выражение > выражение
Меньше или равно	Выражение <= выражение
Больше или равно	Выражение >= выражение
Операции равенства:	
Равно	Выражение == выражение
Не равно	Выражение != выражение
Логические операции:	
Логическое НЕ	! выражение
	not выражение
Логическое ИЛИ	Выражение | выражение
	Выражение or выражение
Логическое И	Выражение & выражение
	Выражение and выражение
Логическое исключающее ИЛИ	Выражение ^ выражение
	Выражение xor выражение
Остальные операции:	
Операции присваивания	= += -= *= /= %=
Операция запятая	Выражение, выражение
Инкремент	++ выражение
Декремент	--- выражение

Список поддерживаемых функций (табл. 2) включает тригонометрические, обратные тригонометрические, гиперболические, логарифмические функции, а также функцию извлечения квадратного корня и др.

Таблица 2. Функции

Название	Синтаксис
Тригонометрические функции:	
Синус	sin(выражение)
Косинус	cos(выражение)
Тангенс	tg(выражение)
Котангенс	ctg(выражение)
Обратные тригонометрические функции:	
Арксинус	arcsin(выражение)
Арккосинус	Arccos(выражение)
Арктангенс	arctg(выражение)
Арккотангенс	arcctg(выражение)
Гиперболические функции:	
Синус	sh(выражение)
Косинус	ch(выражение)
Тангенс	th(выражение)
Котангенс	cth(выражение)
Логарифмические функции:	
Логарифм	log(выражение, выражение)
Натуральный логарифм	ln(выражение)
Десятичный логарифм	lg(выражение)
Остальные:	
Извлечение корня	sqrt(выражение)
Минимум двух чисел	min(выражение)
Максимум двух чисел	max(выражение)
Случайное число	Random(выражение)

Компилятор и интерпретатор представляют собой библиотеку классов на языке C++, которая может быть скомпилирована на любом компиляторе, совместимом со стандартом ANSI C++. Таким образом, компилятор и интерпретатор могут использоваться на большом количестве платформ. В настоящий момент существует реализация языка в виде динамически загружаемой библиотеки (DLL) под Windows.

Компилятор оптимизирует и преобразует текст программы в двоичные коды, что позволяет получить существенный выигрыш в производительности во время ее выполнения. Программа в двоичных кодах может быть выполнена на любой платформе без перекомпиляции.

Планируется возможность добавления работы с динамическими массивами, связанными списками, стеками и очередями, а также возможность определения пользовательских функций. Эти возможности позволят расширить круг решаемых задач и упростить моделирование поведения сложных объектов.

Пример программы:

```
/*
    Программа решения квадратного уравнения
*/
real a = 1, // Объявление и инициализация трех
      b = 5, // переменных вещественного типа -
      c = 6; // коэффициентов уравнения


real d = b*b - 4*a*c; // Вычисление дискриминанта


if (d==0)              // Если дискриминант равен нулю
real x = -b/(2*a);     // уравнение имеет два
// равных корня
else if (d>0)          // в противном случае
{
    // вычислить корни
    real x1 = (-b-sqrt(d)) / (2*a);
    real x2 = (-b+sqrt(d)) / (2*a);
}
```

Предлагаемый язык моделирования (FAscript) позволяет моделировать как непрерывные, так и дискретные процессы, а также преодолевать существующее противопоставление языков собственно имитационного моделирования и универсальных языков программирования (общего назначения). Данное обстоятельство объясняется тем, что предложенный вариант построения языка обеспечивает эффективную машинную реализацию программы с одновременной концептуальной направленностью языка на функциональные процессы, подлежащие моделированию. Практика моделирования систем показывает, что именно такое сочетание свойств языка определяет "успех имитации как метода экспериментального исследования сложных реальных объектов" [4, с. 137].

FAscript (совместно со средством визуализации функциональных зависимостей) используется при разработке программного модуля имитации функционирования системных компонент объектной модели. Это значительно повышает возможности CASE-инструментария, автоматизирующего новый метод системологического анализа, так как позволяет анализировать и проектировать структуру и состав системы с учетом количественных критериев.

Дальнейшее развитие данного языка обусловлено общей тенденцией развития объектной парадигмы разработки программного обеспечения и направлено на повышение концептуального уровня прикладного программирования. В конце концов данная тенденция должна привести к тому, что создание прикладной программы будет осуществляться конечным пользователем самостоятельно путем концептуального объектного моделирования без применения какого-либо языка программирования и без привлечения программиста. Одним из путей превращения данной тенденции в реальность является объединение особенностей и преимуществ метода OMSAD, CASE-технологии и FAscript.

Литература: 1. *Буч Г.* Объектно-ориентированный анализ и проектирование с примерами приложений на C++, 2-е изд.: Пер. с англ. М.: "Издательство Бином", СПб.: "Невский диалект", 1999. 560с. **2**. *Йордан Э., Аргила К.* Структурные модели в объектно-ориентированном анализе и проектировании. / Пер. с англ. М.: "ЛОРИ", 1999. 264с. **3.** *Буч Г., Рамбо Д., Джекобсон А.* Язык UML. Руководство пользователя / Пер. с англ. М.: ДМК, 2000. 432с. **4.** *Советов Б.Я., Яковлев С.А.* Моделирование систем. М.: Высш. шк., 1998. 319с.

Поступила в редколлегию 16.05.2001

Рецензент: д-р техн. наук, проф. Соловьева Е.А.

Попов Андрей Сергеевич, студент гр. ПОАС 00-1, сотрудник научно-учебной лаборатории Приобретения знаний ХНУРЭ. Адрес: Украина, 61022, Харьков, ул. Сумская, 59, тел. 40-95-91, 43-49-95.

УДК 681. 5

ИНВАРИАНТНАЯ АЛГЕБРА РАНГОВЫХ ПРЕДИКАТОВ ДЛЯ МОДЕЛИРОВАНИЯ И СИНТЕЗА НЕПАРАМЕТРИЧЕСКИХ СИСТЕМ

ПОЛОНСКИЙ А.Д.

На основе совместного использования скалярного произведения векторов и процедуры ранжирования строится инвариантная алгебра ранговых предикатов. В рамках теории ИАРП получены математические модели непараметрических систем с кодированием номера канала рангом. Предлагается метод их электрического синтеза в элементном базисе ИАРП – рангерах.

Введение

До настоящего времени остается актуальной проблема обработки измеряемых случайных процессов (ИСП). Суть этой проблемы состоит в том, что полного объема априорных сведений о свойствах ИСП, требуемого известными методами обработки информации [1], как правило, нет. В связи с этим необходимо создание непараметрических систем (НС), обладающих свойством инвариантности к распределению ИСП. Решение этой задачи может быть получено путем разработки математического аппарата для моделирования и синтеза инвариантных НС. Один из таких математических аппаратов, который называется инвариантная алгебра ранговых предикатов (ИАРП), предлагается в настоящей работе.