Г. Л. МУРАВЬЕВ, В. И. ХВЕЩУК
БрГТУ (г. Брест, Беларусь)

СРЕДСТВА СПЕЦИФИКАЦИИ АЛГОРИТМОВ

Важной составляющей систем обучения разработке программ являются лингвистические средства – входной язык и средства поддержки, согласованные со спецификой действий, выполняемых при проектировании программ. Актуальным представляется использование исполнимых, моделируемых спецификаций алгоритмов, текстов программ и построение систем обучения в виде компьютерных сред, базирующихся на прототипировании программ, исполнимости спецификаций проектов (по аналогии с системами программирования) [1-3].

Спецификации алгоритмов должны быть инвариантными к языкам программирования высокого уровня, читаемыми, исполнимыми, легко контролируемыми вручную и автоматически в процессе разработки. Описания в итоге должны быть готовым документом для автоматического или ручного кодирования на подходящем языке программирования высокого уровня (ЯВУ). Соответственно входной язык (pseudo code) [1], должен напоминать естественный с возможностью описывать тексты по правилам, близким к правилам естественного языка, отвечать принципам структурного программирования, базироваться на минимальном наборе изобразительных средств, управляющих структур, отображающих соответствующие математические понятия [4]. Используется процедурный механизм построения спецификаций

<программа> ::= <основной_блок> [<подпрограмма>] { [<подпрограмма>] }
<основной_блок> ::= “ПРОГРАММА” [<идентификатор>]
<глобальные_объекты> “НАЧАЛО” <тело_программы>
“КОНЕЦ-ПРОГРАММЫ” [< идентификатор>] .

Для описания действий используются как исполнимые псевдокоманды (команды), соответствующие базовым управляющим структурам

<оператор> ::= <оператор_вызова> | <оператор_присваивания>
|<оператор_условный> |<оператор_цикла> | <оператор_ввода>
| <оператор_вывода> | <оператор_файловый> ,

так и команды-комментарии промежуточного типа. Первые имеют строгую синтаксическую форму, формируемую на базе ключевых слов, записываемых значимыми, прописными символами, например

<цикл_с_предусловием> ::= “ЦИКЛ-ПОКА” <выражение>
{“ПОВТОРЯТЬ” | “ВЫПОЛНЯТЬ” | “ДЕЛАТЬ” }
<набор_операторов>
“КОНЕЦ-ЦИКЛА” .

Вторые обладают упрощенным синтаксисом, записываются строчными символами, напоминают предложения естественного языка, предназначены для промежуточного использования в процессе разработки спецификаций, требуют дальнейшей детализации. Спецификация приобретает свойство исполнимости после трансформации всех команд промежуточного типа.

Для повышения читаемости текстов наряду с традиционными комментариями используются: “встраиваемые” комментарии, помещаемые в любое место текста, команду без выделения спецсимволами; префиксно-постфиксная система формирования идентификаторов переменных, названий функций, процедур, обеспечивающая склоняемость идентификаторов в зависимости от контекста; “подсветка”, выделение синтаксических единиц для удобства восприятия текстов при работе в редакторе. Названия подпрограмм формируются “активными”. Для этого имена процедур базируются на глаголе, означающем возложенное на модуль действие, за которым может следовать существительное либо составное существительное. Имена функций базируются на одиночном или составном существительном, означающем результат, вычисляемый, возвращаемый подпрограммой.

Склоняемость идентификаторов обеспечена применением префиксов и суффиксов, записываемых строчными символами и не обладающих значимостью при исполнении. При этом совместное использование указанных средств позволяет формулировать более осмысленные тексты по правилам, напоминающим правила естественного языка. Обобщенный идентификатор, составленный из значимой и опускаемой при исполнении частей, представлен ниже

<идентификатор> ::= [<приставка>] <корень> [<окончание>]
(["_" [<приставка>] <корень> [<окончание>]]) ,
<корень> ::= <прописная_буква> ([<прописная_буква> | "_" | <цифра>]) ,
<приставка> ::= <встраиваемый_комментарий> ,
<окончание> ::= <встраиваемый_комментарий> ,
<встраиваемый_комментарий> ::= <строчная_буква> ([<строчная_буква >]) .

Инструментальные средства поддержки языка спецификаций должны обеспечивать: редактирование спецификаций; исполнимость спецификаций, контроль корректности; автогенерацию по спецификациям программ на ЯВУ; анализ и структурирование ЯВУ-кодов. Исполнимость спецификаций обеспечивается формализованным построением языка и наличием соответствующей программной поддержки генерации загрузочных кодов либо интерпретации спецификаций алгоритмов [5].

Таким образом, в работе рассмотрен подход к построению языка спецификации алгоритмов, обеспечивающий соответствие принципам структурной разработки, "прозрачность" и читаемость текстов и потенциальную исполнимость. Приведено описание языка в нотации Бэкуса-Наура.

ЛИТЕРАТУРА

1. Одинцов, И. Профессиональное программирование. Системный подход / И. Одинцов. – СПб: БХВ-Петербург, 2002. – 512 с.

2. Лисков, Б. Использование абстракций и спецификаций при разработке программ / Б. Лисков, Дж. Гатэг. – М.: Мир, 1989. – 424 с.

3. Кузин, С.Г. Компьютерная технология обучения основам алгоритмизации / С.Г. Кузин, Р.О. Митин, И.С. Скрибловский // [Электронный ресурс]. – 2008. – Режим доступа: www.unn.ru/ vmk/graphmod.

4. Муравьев, Г.Л. Автоматизация обучения алгоритмизации / Г.Л. Муравьев, С.В. Мухов // Вести ИСЗ. – 2004. – № 3. – С. 24-29.

5. Муравьев, Г.Л. Информационные технологии для обучения конструированию программ / Г.Л. Муравьев, В.И. Хвещук // Инновационные технологии обучения физико-математическим дисциплинам: материалы III междунар. науч.-практ. конф., Мозырь, 2011. – С. 156-157.