ИНФОРМАТИКА, ВЫЧИСЛИТЕЛЬНАЯ ТЕХНИКА И УПРАВЛЕНИЕ

DOI: 10.12737/article_5a27cb8d226d85.82154993

Ломакин В.В., канд. техн. наук, доц.,
Асадуллаев Р.Г., канд. техн. наук,
Зайцева Т.В., канд. техн. наук, доц.,
Путивцева Н.П., канд. техн. наук,
Белоконь Ю.Ю., аспирант
Белгородский государственный национальный исследовательский университет

ФОРМАЛЬНО-АЛГОРИТМИЧЕСКОЕ ПРЕДСТАВЛЕНИЕ СПЕЦИАЛИЗИРОВАННОГО ЯЗЫКА УПРАВЛЕНИЯ ДАННЫМИ*

lomakin@bsu.edu.ru

Авторами предложены новые инструментальные средства решения задачи построения формального описания специализированного языка управления данными и его реализации с применением макропроцессора. В работе на основе выявленных требований к реализуемому макропроцессору предложена структура универсального транслятора, ориентированного на обработку данных и знаний при решении широкого круга управленческих задач поддержки принятия решений. Рассмотрены возможности описания свойств языков программирования с помощью основных классов контекстно-свободных и контекстно-зависимых грамматик с учетом их характеристик, достоинств и недостатков. Проведенный анализ современных средств формального описания языков позволил синтезировать структуру описания специализированного инструментального языка, при этом контекстно-зависимые и неформально описываемые свойства языка предложено реализовать алгоритмически. Представлена структура формально-алгоритмического представления инструментального языка. Рассмотрены два практически значимых случая процесса разбора входной цепочки символов и генерирования текста на базовом языке.

Ключевые слова: *специализированные языки управления данными, формальное описание языков, формальные грамматики, сеть конечных автоматов.*

Введение. Под управлением данными в настоящее время понимают широкий круг процессов, связанных с их созданием, модификацией и удалением, а также организацией хранения и поиска информации на всех этапах жизненного цикла информационных систем [1, 2]. Для реализации унифицированных процедур управления данными при решении широкого круга задач используются различные программно-лингвистические средства [3–5]. При этом важное значение имеет обеспечение возможности описания семантики решаемых задач обработки данных в терминах, существенных для рассматриваемой предметной области, с помощью специализированных языков управления данными. В этом случае необходимо решить вопросы построения формальных средств, которые, с одной стороны, достаточно компактно опишут процедуры определения самого специализированного языка управления данными, а с другой – позволят описать процесс перехода от описания задач на языке, приближенном к естественному, к представлению процедур управления данными на традиционно используемых языках, для которых существуют надежные трансляторы. Примером такого языка может служить язык запросов *SQL*, применяемый, в том числе, в качестве встраиваемых конструкций в программах на языках высокого уровня, например, *PASCAL, C* и пр. [6, 7]. Все это приводит к необходимости получения структуры, состава и алгоритма описания специализированного языка управления данными совместно с процедурой получением транслятора с данного языка.

Основная часть. Для решения задачи построения формального описания специализированного языка (СЯ) необходимо выбрать подход, который обеспечивал бы совместное описание СЯ и процесса его перевода в базовый язык, в качестве которого выбирается наиболее подходящий с точки зрения специфики решаемых задач и надежности транслятора язык. Этим требованиям в полной мере удовлетворяет известный и достаточно давно применяемый подход, основанный на использовании макропроцессора, а также инструментальных языков, реализованных с его помощью (*DLIMP*) [8].

Данный подход позволяет разработать универсальный транслятор-обработчик знаний, не зависящий как от языка на входе, так и от базового языка. На рис. 1 представлена упрощенная структура универсального транслятора, отличительной особенностью которого является универсальность и ориентация на обработку данных и знаний для решения широкого круга управленческих задач поддержки принятия решений на различных уровнях управления предприятием. Генератор кода описывает правила развертывания конструкций исходного языка в конструкции на базовом. В файлах описания предметной области содержится информация, которая относительно часто меняется, и изменение которой осуществляется без вмешательства в текст транслятора или библиотеки, например, к такой информации можно отнести параметры рассматриваемого процесса или характеристики оборудования. Таблицы могут содержать информацию о лексемах и их описании, а также диапазонах значений переменных, такого рода информацию желательно изменять вне текста транслятора.

Определим требования к макропроцессору, вытекающие из специфики задач, связанных с обработкой данных на разных уровнях управления:

– независимость от входного и базового языков;

– доступность осуществления модификации выполняемых функций в зависимости от макробиблиотеки;

– облегчение проектирования за счет рационального построения структуры системы макропроцессор-макробиблиотека (СММ);

– мобильность, связанная с возможностью работы СММ в разных операционных средах;

– возможность повышения эффективности работы СММ на этапе эксплуатации;

– встраиваемость в другие программные продукты.

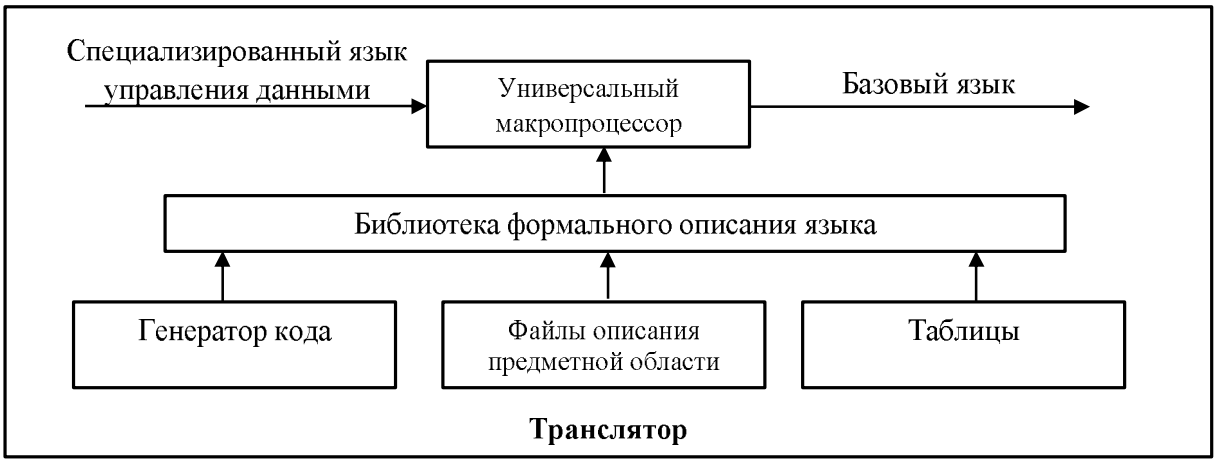

Рис. 1. Упрощенная структура универсального транслятора

В целом, возможность гибкой настройки и ориентированность на интеллектуальную поддержку принятия решений позволяют описывать на основе представленной методики информационные свойства и алгоритмы обработки объектов. Это обеспечивается универсальным характером предлагаемых программно-лингвистических средств, как инструмента реализации управления данными.

Построим формализованное представление объектов предметной области с точки зрения решаемых нами задач управления данными. С целью организации унифицированных процедур хранения и обработки данных рассмотрим методы формализации символьной информации, которая поступает на вход любой информационно-управляющей системы и затем интерпретируется ею как данные или знания [9, 10]. В зависимости от уровня управления используется многообразие современных инструментальных сред, характеризующихся направленностью на решение задач определенных предметных областей. Использование подобных средств позволяет провести формализацию решаемой задачи в терминах данной предметной области, а также получить рациональное решение поставленной задачи.

При использовании инструментальных языков возникает вопрос о разработке транслятора выбранного языка. В процессе построения транслятора возникают ситуации, когда необходима модификация инструментального языка и соответствующая адаптация библиотеки его описания. Практика показывает, что программные средства, разработанные для решения определенного круга задач, не всегда позволяют достаточно полно охватить конкретную предметную область. Трансляторы, построенные без возможности модификации структуры входного и базового языков и представляющие собой жесткую

структуру, не способны удовлетворять современным требованиям практического использования, так постоянно возникает необходимость модификации и доработки.

Для написания транслятора с языка необходимы специальные знания. Учитывая, что программисты, работающие с программным обеспечением, предназначенным для реализации задач управления и обработки данных, такими знаниями не обладают, появляется необходимость в разработке универсальных средств реализации специализированных языков управления для различных уровней иерархии управления данными. К разрабатываемой среде предъявляют высокий уровень требований, что, в свою очередь, приводит к необходимости использовать формальные методы анализа способов описания различных классов языков.

Синтаксис языков описывают с помощью формальных грамматик, которые фактически представляют собой математическую модель грамматики, описанную в рамках какой-то синтаксической теории. Для описания специализированных языков в основном применяются контекстно-свободные (КС) грамматики с некоторыми расширениями [11].

Целью разработки формальных грамматик является использование полученного языка для описания структуры и определения основных функций системы автоматизированного проектирования трансляторов. В рамках достижения этой цели рассмотрим возможности описания свойств языков программирования с помощью основных классов КС-грамматик.

Локальные и простые свойства языков описываются с использованием регулярных грамматик. Такие грамматики предполагают определение регулярных выражений и эквивалентны конечным автоматам. Главное достоинство регулярных грамматик – простота реализации, но при этом они могут описать лишь класс регулярных языков. В качестве аппарата распознавания регулярного языка используется конечный автомат.

LL(1)-грамматики относятся к детерминированным КС-грамматикам и позволяют получить детерминированный синтаксический анализатор, работающий по принципу нисходящего разбора. Преимущества LL(1)-анализа состоят в его естественности, данный метод является наглядным и удобным для создания основы для последующей компиляции языка программирования. Кроме того, его легко реализовать и убедиться в корректности его работы. Написанный код не только выполняет собственно синтаксический анализ, но может и проводить проверку соответствия типов и других проверок, а также реализовывать действия на этапе синтеза, такие как распределения памяти и генерация кода при включении в его содержание соответствующих функций.

В то же время можно определить и некоторые недостатки LL(1)-метода разбора, такие как неэффективность вызовов функций и необходимость преобразования грамматики, не обладая при этом информацией о том, существует ли подходящее преобразование. Проблема заключается не только в нахождении преобразования, но и в проверке корректности его применения. Таким образом, имеются веские причины использовать при преобразовании надежные инструментальные средства, а не зависеть от ручного подхода. Кроме того, при реализации преобразования могут получаться достаточно большие программы синтаксического анализа, и существует тенденция к появлению в теле одной функции операций, относящихся к разным фазам процесса компиляции.

LR(1)-грамматики предлагают более универсальный с точки зрения описания синтаксиса языка метод разбора, который позволяет значительно снизить неудобства, связанные с преобразованием грамматик. *LR*(1)-грамматики включают в себя дополнительные возможности по описанию синтаксических свойств конкретных языков. В качестве аппарата распознавания КС-языка используется эквивалентный конечному автомату МП-автомат, который включает в себя магазинную память.

Кроме описанных КС-грамматик существуют классы контекстно-зависимых грамматик, у которых левые и правые части всех продукций могут быть окружены терминальными и нетерминальными символами [11]. Их основное назначение – теоретическое описание свойств языков и возможностей реализации распознавателей. Но в описаниях синтаксиса реальных языков программирования или спецификаций они не используются.

Все классы грамматик представляют собой иерархию, в которой более простые грамматики характеризуются простотой описания и реализации, но при этом имеют ограничения касательно множества языков, описываемых с их помощью. Таким образом, возникает необходимость в использовании всей иерархии грамматик, начиная с регулярного класса, для реализации рационального и полного формального описания конкретного языка.

Те свойства языка, которые в практических случаях описываются формально в виде грамматик, называют синтаксическими. Причем на практике формальное описание свойств языка применяется только при осуществлении компактной записи таких свойств. Во всех остальных

случаях для описания свойств языка, которые не описаны формально, используют неформальные методы, соответственно при реализации трансляторов с языков возникает необходимость в описании неформальных свойств языков в виде алгоритмов.

Главной задачей при описании синтаксиса языка является решение задачи реализации транслятора рассматриваемого языка, при этом проблемы зацикливания анализатора или соблюдение строгости описания формальных свойств не имеют существенного значения, здесь важно только получение результата.

На практике постоянно требуется разработка новых программных продуктов из-за невозможности использования реализованных программ для построения трансляторов других языков, затруднений при встраивании таких программ в новые операционные среды [12, 13].

Так как возможно описание более широкого класса инструментальных языков при построении универсальной среды для реализации трансляторов, программ обработки и преобразования данных, то, учитывая, прежде всего, основную цель реализации трансляторов, появляется необходимость исследования вопросов формального описания инструментальных языков на основе предложенных ранее подходов для выработки структуры и главных функций лингвистической части разрабатываемых средств реализации трансляторов.

Структура описания синтаксических свойств языка основана на анализе методов формального описания специализированных языков и охватывает известные методы регулярного и КС-разбора [9]. Представленный анализ методов формального описания специализированных языков позволил определить структуру описания синтаксических свойств языка, как структуру, имеющую модульный характер в соответствии с используемыми методами регулярного и КС-разбора. Также, в связи с необходимостью описания контекстно-зависимых свойств языка и неформально описываемых свойств, предлагается использовать дополнительный аппарат, реализуемый алгоритмически.

Неформально определяемые свойства языка будем описывать в виде сети автоматов, где $S \to b\alpha$ – правила вывода, O – правила грамматики, отвечающие начальному состоянию, A, B, C,… – правила грамматики, описывающие одноименные состояния A, B, C, …

Опишем сеть автоматов формально следующим образом

$$Na = (S_1,..,S_n; P_1,..,P_e; q_1,..,q_m), \qquad (1)$$

где S_i – автоматы *(i=1..n)*, q_j – состояния сети автомата *(j=1..m)*, P_l – предикаты, описывающие переход к состояниям q_j *(l=1..e)*.

Пример фрагмента сети автоматов представлен на рис. 2.

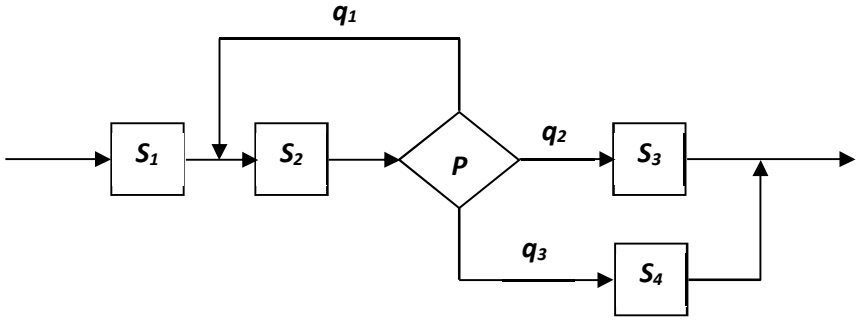

Рис. 2. Фрагмент сети автоматов

Окончание работы сети автоматов фиксируется в двух ситуациях:

– отсутствуют входные воздействия;

– не определены автоматы в конечном состоянии сети.

При этом переход сети автоматов в состояние q_j будет соответствовать контексту, а табличные данные, представленные в виде (T_k, A_k) (T_k – идентификационный признак объекта *k*, A_k – атрибуты объекта *k*), служат основанием для принятия решения о переходе.

Практическая разработка транслятора подразумевает выполнение нужных действий по мере необходимости их выполнения, и совершенно не требует соблюдения жесткой классификации этапов разбора [14]. При возникновении необходимости в разделении этапов разбора текста на лексический и синтаксический анализ программист продумывает рациональное распределение решаемых задач на данных этапах разбора.

Построение более эффективного компилятора возможно только при рациональном построении сети автоматов [15]. При этом эффективность реализации компилятора несущественна на

этапе его проектирования и отладки, а приобретает значение лишь на этапе эксплуатации. Преобразуем сеть автоматов в формально-алгоритмическое представление инструментального языка (рис. 3) в соответствии с предложенной ранее структурой описания языка следующим образом:

– правила регулярных грамматик определяют начальное состояние автомата *O*;

– при построении неконечных состояний q_j используется иерархический принцип от *LL*(1) до *LR*(1) грамматик;

– при отсутствии входной информации, а также, когда невозможно продолжить работу в случае ошибки, осуществляется переход в конечное состояние.

– неконечное состояние *e* соответствует обнаруженной в процессе работы ошибке, при этом можно задать процедуру обработки.

В результате получим формально-алгоритмическое описание специализированного инструментального языка управления данными (рис. 3).

Алгоритмическая часть полученной сети автоматов отвечает за семантическую обработку исходного текста. Основная часть работ по проектированию компиляторов посвящена теоретическому описанию процесса разбора исходной цепочки символов, а вопросы генерации кода рассмотрены сжато.

В полученном формально-алгоритмическом представлении инструментального языка генерация текста на базовом языке, осуществляется при выполнении специальных действия в процессе разбора входной цепочки символов. В данном случае нет жесткой привязки процесса преобразования текста к процессу разбора входного текста, т.е. все необходимые действия можно выполнить после полного анализа цепочки текста.

На основе структуры предлагаемого программного обеспечения (рис. 1) и формально-алгоритмического представления можно определить следующие требования к выбору базового языка, состоящие в соблюдении:

– соответствия структуры компилятора инструментальному языку; – в этом случае совместив процесс разбора исходного текста с процессом генерации кода, можно упростить последний.

– соответствия типов данных, – при наличии соответствия типов данных можно осуществлять обработку контекстно-зависимых свойств существующим компилятором базового языка;

– соответствия операций, – в этом случае отпадает необходимость обработки и преобразования операций исходного и базового языка;

– надежности компилятора базового языка (кроме случая языка машинных команд).

Анализ процесса разбора входной цепочки символов и генерация текста на базовом языке приводит к необходимости рассмотрения двух практически значимых случаев.

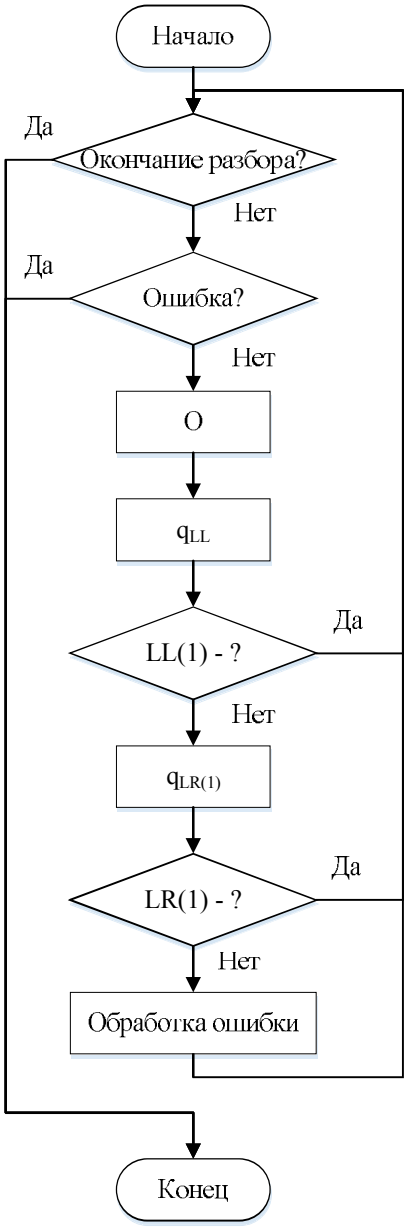

Рис. 3. Структура формально-алгоритмического представления инструментального языка

В первом случае процесс разбора охватывает вопросы принадлежности базовых конструкций к исходному языку и правильности употребления операций. Здесь не требуется сложных алгоритмов обработки при генерации текста, особенно в том случае, если языки имеют схожую структуру. Существенная роль в процессе анализа программы и диагностики ошибок отводится существующему компилятору базового языка. Структура сети автоматов упрощается и охватывает

только состояния O и $q_{LL(1)}$. При этом необходимо уделить внимание правильному заданию соответствия строк базового языка строкам входного посредством таблиц.

Во втором случае при осуществлении процесса разбора входной цепочки символов используются все возможные методы. Здесь можно определить все свойства специализированного языка и возможные ошибки в процессе разбора. При этом, несмотря на усложнение структуры сети автоматов и алгоритмов генерации текста, сеть автоматов может полностью описать процесс компиляции. На компилятор базового языка в этом случае возлагаются лишь технические функции преобразования текста в машинный код и дополнительный контроль правильности программы, т.к. все проверки выполнены на предыдущем этапе.

Для практической реализации транслятора можно на начальном этапе использовать вариант реализации для описанного первого случая. При возникновении необходимости обеспечения независимости от транслятора базового языка, можно перейти ко второму варианту, постепенно усложняя структуру сети.

С целью практической реализации сети автоматов необходимо осуществить преобразование предложенного формально-алгоритмического представления специализированного языка управления данными в лингвистические конструкции макробиблиотеки м реализовать программное обеспечение макропроцессора.

Выводы. Для формального описания и реализации специализированного языка управления данными предложено использование подхода, основанного на применении макропроцессора. Определены требования к реализуемому макропроцессору и предложена структура универсального транслятора. На основе анализа современных средств формального описания языков синтезирована структура описания языка, при этом неформальные свойства языка предложено описывать в виде сети автоматов. Получено формально-алгоритмическое описание специализированного инструментального языка управления данными, определены требования к выбору базового языка с точки зрения рациональной практической реализации. Рассмотрены практически значимые случаи, исходя из решения вопроса проектирования и разработки транслятора специализированного языка. Предложенные в статье формально-логические решения могут рассматриваться в качестве основы для построения и реализации программно-лингвистических средств управления данными.

Выполнено в рамках реализации комплексного проекта по созданию высокотехнологичного производства «Разработка методологии и инструментальных средств создания прикладных приложений, поддержки жизненного цикла информационно-технологического обеспечения и принятия решений для эффективного осуществления административно-управленческих процессов в рамках установленных полномочий», шифр «2017-218-09-187»; постановление Правительства Российской Федерации от 9 апреля 2010г. №218.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. ГОСТ Р ИСО/МЭК ТО 10032-2007: Эталонная модель управления данными

2. Кузовкин А.В., Цыганов А.А., Щукин Б.А. Управление данными: учебник для студ. высших учеб. заведений. М.: Издательский центр «Академия», 2010. 256 с.

3. Зайцев И.М., Зайцева Т.В., Лифиренко М.В., Ломакин В.В., Путивцева Н.П. Многокритериальный выбор корпоративной системы с применением инструментальных средств повышения степени согласованности матриц парных сравнений // Информационные системы и технологии. 2017. № 6 (104). С. 85–93.

4. Фаулер М. Языковой инструментарий: новая жизнь языков предметной области. URL: http://www.maxkir.com/sd/languageWorkbenches.html (дата обращения 21.02.2017)

5. Михелев В.М., Петров Д.В., Батищев Д.С., Кузнецов К.В., Ерошенко Я.Б. Методы и средства программирования для интернет. Технологии программирования на стороне web-сервера: учебно-практическое пособие для бакалавров, обучающихся по направлению подготовки 010200.62 "Математика и компьютерные науки". Белгород: ИД Белгород, 2016. 260 с.

6. Грофф Дж., Вайнберг П. SQL: полное руководство. К.: Издательская группа BHV, 2000. 608 с.

7. Горев А., Макашарипов С., Владимиров Ю. MicrosoftSQLServer6.5 для профессионалов. СПб.: Изд. Питер, 1998. 464 с.

8. Браун П. Макропроцессоры и мобильность программного обеспечения. М.: Изд-во Мир, 1977. 256 с.

9. Льюис Ф., Розенкранц Д., Стирнз Р. Теоретические основы проектирования компиляторов. М: Изд-во Мир, 1979. 654 с.

10. Nemerle. Programming language for "special forces" of developers. [Электронный ресурс] URL : http://nemerle.org/About (дата обращения 21.09.2017)

11. Хантер Р. Проектирование и конструирование компиляторов. М.: Изд. Финансы и статистика, 1984. 232 с.

12. Lomakin V.V., Afonin A.N. Formal and software language means for specialized language implementation // International Journal of Applied Engineering Research. 2015. Т. 10. № 24. С. 45239–45246.

13. Рязанов Ю.Д. Способ преобразования кс-грамматики в LL(1)-грамматику // Вестник Воронежского государственного университета. Серия: Системный анализ и информационные технологии. 2014. № 2. С. 113–116.

14. Муромцев В.В., Ломакин В.В., Цоцорина Н.В. Разработка специализированного языка для удаленного программирования микроконтроллеров // Научные ведомости Белгородского государственного университета. Серия: История. Политология. 2011. Т. 19. № 13-1 (108). С. 180–185.

15. Муромцев В.В., Ломакин В.В., Мишунин В.В. Подход к улучшению алгоритмов грамматического сжатия // Информационные системы и технологии. 2011. № 6 (68). С. 5–9.

Информация об авторах

Ломакин Владимир Васильевич, кандидат технических наук, доцент, заведующий кафедрой прикладной информатики и информационных технологий.
E-mail: lomakin@bsu.edu.ru.
Белгородский государственный национальный исследовательский университет,
Россия, 308015, Белгород, ул. Победы, д. 85.

Асадуллаев Рустам Геннадьевич, кандидат технических наук, доцент кафедры прикладной информатики и информационных технологий.
E-mail: asadullaev@bsu.edu.ru.
Белгородский государственный национальный исследовательский университет,
Россия, 308015, Белгород, ул. Победы, д. 85.

Зайцева Татьяна Валентиновна, кандидат технических наук, доцент, доцент кафедры прикладной информатики и информационных технологий.
E-mail: zaitseva@bsu.edu.ru.
Белгородский государственный национальный исследовательский университет,
Россия, 308015, Белгород, ул. Победы, д. 85.

Путивцева Наталья Павловна, кандидат технических наук, доцент кафедры прикладной информатики и информационных технологий.
E-mail: putivzeva@bsu.edu.ru.
Белгородский государственный национальный исследовательский университет,
Россия, 308015, Белгород, ул. Победы, д. 85.

Белоконь Юлия Юрьевна, аспирант кафедры прикладной информатики и информационных технологий.
E-mail: voitova@bsu.edu.ru.
Белгородский государственный национальный исследовательский университет,
Россия, 308015, Белгород, ул. Победы, д. 85.

Поступила в октябре 2017 г.

Lomakin V.V., Asadullaev R.G., Zaitseva T.V., Putivzeva N.P., Belokon Y.Y.

FORMAL-ALGORITHMIC REPRESENTATION OF THE SPECIALIZED LANGUAGE OF DATA CONTROL

The authors proposed new tools for solving the problem of constructing a formal description of a specialized data control language and its implementation with the use of a macroprocessor. In the paper structure of a universal translator which is oriented to the processing of data and knowledge in the process of solving a wide range of management tasks for decision making support on the basis of the identified requirements for the implemented macroprocessor is proposed. The possibilities of the description of the properties of programming languages with the help of the main classes of context-free and context-dependent grammars and considering their characteristics, merits and demerits are considered. The analysis of modern tools of formal description of languages made it possible to synthesize the structure of the description of a specialized instrumental language, while it was proposed to implement context-dependent and informally described properties

of the language algorithmically. The structure of the formal-algorithmic representation of the instrumental language is presented. Two practically significant cases of parsing the input string of symbols and text generation in the base language are considered.

***Keywords**: specialized languages for data control, formal description of languages, formal grammars, finite state machine network.*

Information about the authors
Lomakin Vladimir Vasil'yevich, PhD, Head of Department.
E-mail: lomakin@bsu.edu.ru.
Belgorod State National Research University,
Russia, 308015, Belgorod, st. Pobedy, 85.

Asadullaev Rustam Gennad'yevich, PhD, Assistant professor.
E-mail: asadullaev@bsu.edu.ru.
Belgorod State National Research University,
Russia, 308015, Belgorod, st. Pobedy, 85.

Zaitseva Tatiana Valentinovna, PhD, Assistant professor.
E-mail: zaitseva@bsu.edu.ru.
Belgorod State National Research University,
Russia, 308015, Belgorod, st. Pobedy, 85.

Putivzeva Natalia Pavlovna, PhD, Assistant professor.
E-mail: putivzeva@bsu.edu.ru.
Belgorod State National Research University,
Russia, 308015, Belgorod, st. Pobedy, 85.

Belokon Yuliya Yurevna, Research assistant.
E-mail: voitova@bsu.edu.ru.
Belgorod State National Research University,
Russia, 308015, Belgorod, st. Pobedy, 85.

Received in October 2017