УДК 004.822:004.891.2

МОДЕЛЬ ИНТЕЛЛЕКТУАЛЬНОЙ HELP-СИСТЕМЫ ДЛЯ РАЗРАБОТЧИКОВ ПРОГРАММ, ОРИЕНТИРОВАННЫХ НА ОБРАБОТКУ БАЗ ЗНАНИЙ

О.В.ПИВОВАРЧИК

Барановичский государственный университет,
Войкова, 21, Барановичи, 225404, Беларусь

Поступила в редакцию 25 марта 2009

Объектом рассмотрения являются программы, входящие в состав баз знаний интеллектуальных систем, и описывающие способы решения различных задач по обработке баз знаний. В статье представлена модель интеллектуальной help-системы для разработчиков программ, ориентированных на обработку баз знаний. Описана формальная модель базы знаний, операции машины обработки знаний, методы проектирования пользовательского интерфейса help-системы.

Ключевые слова: help-система, язык программирования, база знаний, машина обработки знаний.

Введение

Расширение областей применения интеллектуальных программных систем привело к широкому использованию языков программирования, ориентированных на обработку знаний. Проектирование программ на языках данного класса требует высоких профессиональных навыков у разработчиков. Программист должен владеть принципами разработки баз знаний, знать языки представления знаний и языки обработки знаний. Кроме этого, для эффективного написания программных текстов он должен обладать большим опытом в области программирования. Поэтому в настоящее время появилась необходимость в разработке интеллектуальных help-систем по проектированию программного обеспечения, ориентированного на обработку баз знаний. Такие системы должны решать следующие задачи: осуществление помощи в разработке программ за короткие сроки; повышение качества разрабатываемых программ; уменьшение предъявляемых требований к квалификации программиста.

Help-системы для разработчиков программного обеспечения

Help-система представляет собой консультационную программную систему по технологии разработки программного обеспечения. Существующие help-системы для разработчиков программ можно разделить на классы:

- справочные гипертекстовые системы: WinHelp, Adobe Acrobat, HelpScribble, WebHelp, FlashHelp, HTMLHelp, NetHelp, WebHelp, JavaHelp, AnetHelp Tool, Help And Manual, Mif2GO, RoboHelp;
- браузерные системы: www.msdn.com, www.rsdn.ru, www.codeteacher.com, www.help-site.com, www.programmingtutorials.com, www.scientific-library.net и др.
- обучающие системы: Small Pascal System, Pascal ABC, Jeliot, GILD, BlueJ, Greenfoot.

К основным недостаткам существующих help-систем относятся:

- отсутствие должного уровня интерактивности и ориентация на специалистов;
- представление информации в виде баз или хранилищ данных;
- отсутствие подсистем управления изучением технологий программирования;
- отсутствие унифицированной модели представления знаний.

Перечисленные недостатки указывают на необходимость привлечения методов, технологий и средств искусственного интеллекта для разработки help-систем.
К интеллектуальным help-системам предъявляются следующие требования:

- представление и обработка знаний;
- унифицированная модель представления знаний;
- наличие универсальных поисковых средств;
- интеграция с инструментальным средством на уровне совместной разработки программ;
- мониторинг за действиями разработчика программ;
- наличие адаптивных средств обучения технологии программирования;
- управление разработкой программ.

Языки программирования, ориентированные на обработку знаний

За историю развития информационных технологий создано большое количество языков программирования, направленных на решение различных классов задач. Существует несколько различных подходов к классификации языков программирования [1-3]. Например, можно выделить классификацию по назначению, по моделям программирования, по подходам и приемам программирования [1], по ориентации на архитектуру вычислительных систем. Одной из основных классификаций языков программирования является поддержка парадигмы программирования. Парадигма программирования определяет совокупность подходов, методов, стратегий, идей и понятий, используемых разработчиком программ при создании программного обеспечения. Каждая парадигма имеет определенный класс успешно решаемых задач. В настоящее время насчитывается более 20 парадигм.

Возросший интерес к разработке интеллектуального программного обеспечения, ориентированного на обработку баз знаний, привел к бурному развитию языков представления знаний (ЯПЗ) и языков обработки знаний (языков программирования, ориентированных на обработку знаний).

ЯПЗ знаний используют определенные модели представления знаний, большинство из которых может быть сведено к следующим классам: продукционные модели, фреймы, формальные логические модели, представление знаний на основе исчисления предикатов; семантические сети. Правильный выбор модели представления знаний существенно упрощает разработку интеллектуальной системы. В настоящее время существуют десятки ЯПЗ для различных предметных областей, например:

- для задач планирования: STRIPS, ADL, PDDL;
- для онтологических систем: OWL, CycL, KL-ONE, KIF, OCML;
- для задач распознавания и генерации текстов: KRL, FRL.

Языки программирования, ориентированные на обработку знаний, существенно отличаются от языков, ориентированных на обработку традиционных структур данных. Особенностью таких языков является удовлетворение нижеследующим требованиям:

- наличие мощных языков или систем представления знаний;
- наличие гибких средств управления знаниями или базами знаний;
- возможность программной или аппаратной реализации.

Языки обработки знаний активно развиваются и внедряются в практику. В настоящее время обозначился ряд тенденций по разработке комплексов интеллектуальных систем с использованием языков данного класса. Для совместимости отдельных модулей этих систем необходимо использовать ЯПЗ, обеспечивающие семантическую унификацию представления информации. Для решения этой задачи наиболее перспективным является использование языков представления знаний в виде семантических сетей и языков программирования, ориентированных на обработку семантических сетей. SCP (Semantic Code Programming) [4] – это язык программирования, ориентированный на обработку однородных семантических сетей, имеющих базовую теоретико-множественную интерпретацию. Для представления знаний используется язык SC (Semantic Code) [5]. Однородность используемых семантических сетей существенно упрощает их обработку. Наличие базовой теоретико-множественной интерпретации обеспечивает унификацию представления знаний различного вида.

SCP является процедурным языком программирования, который учитывает особенности представления информации в виде графовых конструкций и позволяет их эффективно перерабатывать. Текст языка SCP (scp-программа) представляет собой sc-конструкцию, хранимую в базе знаний и описывающую некоторый процесс переработки других sc-конструкций. Данными scp-программ являются как отдельные sc-элементы (узлы и дуги), так и sc-конструкции. Структурно scp-программа представляет собой множество связанных между собой sc-узлов, каждый из которых обозначает scp-оператор некоторого класса. Обладая мощным набором операторов, SCP позволяет производить поиск, генерацию, удаление, обработку sc-элементов или sc-конструкций. SCP является подъязыком языка SC, поэтому scp-программы также являются знаниями, что обеспечивает согласованность программ с представлением знаний.

В данной работе представлена модель интеллектуальной help-системы на примере help-системы по технологии разработки программ на языке SCP.

Модель интеллектуальной help-системы для разработчиков программ, ориентированных на обработку баз знаний

Разрабатываемая help-система относится к классу sc-систем, т.е. она представляет собой интеллектуальную систему, основанную на знаниях, представленных в sc-коде. В соответствии с семантической технологией проектирования интеллектуальных help-систем, разрабатываемая help-система включает: базу знаний (БЗ), машину обработки знаний (МОЗ), пользовательский интерфейс (ПИ). К ней предъявляются следующие требования: интеграция с инструментальным средством разработки программ, взаимодействие с библиотекой IP-компонентов. Следовательно, разрабатываемая help-система должна содержать информацию не только об используемом языке программирования, но и об инструментальном средстве разработки программ, и об используемых в нем IP-компонентах.

Рис. 1. Архитектура интеллектуальной help-системы

БЗ интеллектуальной help-системы для разработчиков программ включает информацию о технологии программирования: описание языка программирования, языка представления знаний, операционной семантики языка программирования, библиотеки IP-компонентов для

данного языка, методики проектирования программ, инструментальных средств, методики использования инструментальных средств, методики использования IP-компонентов. МОЗ определяет операционную семантику help-системы, т.е. определяет множество правил преобразования информационных конструкций, а также множество правил построения новых информационных конструкций на базе существующих или уже построенных. ПИ обеспечивает обмен знаниями между пользователем и help-системой [6].

Интеллектуальная help-система по технологии разработки программ на языке SCP

На начальном этапе построения help-системы необходимо определить круг задач, которые она будет решать, четко определить назначение каждой из ее подсистем, а также масштаб представления предметной области. Один из способов определить круг задач и масштаб представления предметной области – специфицировать список вопросов, на которые будет отвечать система или каждая ее подсистема, и представить ответы на каждый из классов вопросов. На основании этой спецификации разрабатывается база знаний и машина обработки знаний help-системы. Задача разработки пользовательского интерфейса заключается в уточнении интерактивных, сервисных и оформительских параметров help-системы.

Справочная подсистема по языку SCP является консультантом-экспертом по программированию, который может ответить на любой вопрос новичка или опытного пользователя. К функциям справочной подсистемы относятся:

- поиск информации;
- отображение информации об обрабатываемых структурах данных и языке представления знаний;
- отображение информации об описываемом языке программирования с ориентацией на уровень подготовки разработчика программ;
- анализ программных текстов пользователя и внесение предложений по улучшению их эффективности;
- генерация ответов на запросы пользователя;
- генерация программ по запросу пользователя.

Основными классами вопросов, на которые отвечает данная подсистема, являются: вывод определений или пояснений ключевых понятий, вывод пояснений по выполнению действий, отображение классификаций понятий, определение принадлежности ключевого понятия к классу, выполнение аналитических действий (анализ понятий на схожесть и отличие; анализ программ на предмет синтаксической и семантической корректности, эффективности, оптимальности), поиск связи между понятиями, вывод синтаксиса построения языковых конструкций, выдача рекомендаций по выполнению определенных действий, генерация объектов (программ, фрагментов программ, алгоритмов). Для ответа на некоторые классы вопросов система производит поиск информации в базе знаний, а некоторые ответы генерирует на основании существующих утверждений. Например, для ответа на вопрос «Какие классы scp-операторов используют атрибутивные отношения then_ и else_ для задания последовательности исполнения» система на основании утверждения (рис. 2) выбирает класс «scp-оператор с условным переходом» и производит выбор scp-операторов, включаемых в данную классификационную группу. Для ответа на вопрос «Какие атрибутивные отношения необходимо устанавливать операндам scp-операторов, обрабатывающим одноэлементную структуру данных» система генерирует ответ на основании утверждения, отображенного на рис. 3.

На основании созданной спецификации разрабатывается формальная модель языка программирования SCP, которая является основой БЗ справочной подсистемы. БЗ включает общие понятия языка программирования SCP с последующей их конкретизацией. Например, к основным понятиям SCP относятся: scp-программа, scp-константа, scp-переменная, scp-параметр, scp-оператор, scp-операнд, scp-процесс, scp-метапеременная, scp-переменная, scp-константа. Более детальное описание предметной области представляется выделением подклассов понятий. Например, конкретизируя понятие scp-оператор, вводятся его подклассы: scp-операторы генерации sc-конструкций (gen), scp-операторы удаления sc-конструкций (erase), scp-операторы ассоциативного поиска sc-конструкций (search), scp-операторы выборки sc-элементов и sc-конструкций (select), scp-операторы проверки условий (if), scp-операторы управления содер-

жимым sc-элементов (change), scp-операторы управления scp-процессами (manage). Можно еще дальше конкретизировать класс scp-операторов генерации sc-конструкций: операторы генерации sc-элемента (genEl); операторы генерации sc-конструкции, состоящей из трех элементов (genElStr3); операторы генерации sc-конструкции, состоящей из пяти элементов (genElStr5). Далее таким же образом представляются все классы и подклассы основных понятий.

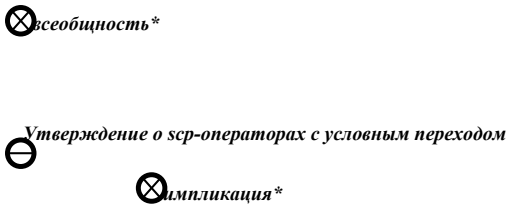

Рис. 2. Утверждение о scp-операторах с условным переходом

Для каждого понятия в базе знаний задаются его теоретико-множественные свойства:

- пояснение и определение понятия;
- разбиение на более частные понятия;
- включение в более общие понятия;
- синонимия понятия;
- пересечение и объединение с другими понятиями;
- основные утверждения о понятии;
- конкретные примеры понятия.

Для определения внутренней структуры понятий вводятся отношения. Например, каждый scp-оператор характеризуется количеством операндов, способом передачи управления, обрабатываемыми структурами данных и основным выполняемым действием. Язык программирования характеризуется классом задач, для решения которых эффективно его использовать. Для программы характерно: тип ее инициирования, интенсивность использования, вид обрабатываемых данных, выполняемые действия. Отдельным классом отношений в языке программирования SCP являются атрибутивные отношения, которые задают определенную семантику при вхождении конкретных экземпляров классов понятий в scp-программу. Например, атрибуты goto_, then_, else_, error_ указывают последовательность выполнения scp-операторов scp-программы. Атрибуты assign_, fixed_, const_, var_ уточняют тип значения scp-операнда, входящего в конкретный экземпляр scp-оператора.

Рис. 3. Утверждение о scp-операторе, обрабатывающем одноэлементную структуру данных

Кроме этого, в БЗ включаются утверждения и правила, описывающие способы проектирования программных текстов и задающие семантику языка. Например, можно выделить классы утверждений: утверждения о значениях scp-переменных и scp-констант; утверждения о scp-параметрах scp-программы; утверждения о порядке исполнения scp-операторов scp-программы; утверждения об обработке структур данных; утверждения о действиях scp-параметров; утверждения об обрабатываемых структурах данных; утверждения о согласовании атрибутов и др. А также БЗ включает множество ключевых узлов, которые имеют определенную семантику и предназначены для описания help-систем.

МОЗ справочной подсистемы включает базовый набор операций для интеллектуальных help-систем, а также дополняется scp-программами, обеспечивающими реализацию задач, характерных для разрабатываемого класса help-систем:

- трансляция информационных конструкций, являющихся запросами пользователя, и представление их в виде семантически эквивалентных sc-конструкций;
- анализ существующих информационных конструкций базы знаний (в том числе scp-программ) и генерация новых, являющихся ответом на запрос пользователя (например, вывод синтаксиса scp-оператора на основании утверждений, хранящихся в базе знаний);
- генерация scp-программ по постановке задачи пользователя или по постановке задачи, сгенерированной самой системой.

Т.к. предполагается интеграция разрабатываемой help-системы с инструментальной средой проектирования программ, то в МОЗ включаются задачи, которые взаимодействуют с операциями, реализованными МОЗ инструментальной среды:

- верификация scp-программ, разрабатываемых пользователем;
- анализ эффективности scp-программ;
- оптимизация scp-программ.

Каждая операция реализована на языке SCP [4] и представляет собой scp-программу, находящуюся в базе знаний. Операции являются демоническими процессами, которые активизируются действиями пользователя или происходящими событиями.

Разработка пользовательского интерфейса заключается в доработке базового интерфейса интеллектуальных help-систем дополнительными функциональными возможностями, в которых нуждаются разработчики программ.

Главной целью подсистемы мониторинга и анализа деятельности является сбор информации о разработчике программ. Основные классы задач, решаемых подсистемой:

- определение общей информации о разработчике программ;
- определение уровня знаний разработчика программ;
- определение и анализ ошибок;
- анализ действий.

Основой базы знаний является формальная модель разработчика программ, которая представляется в виде шаблона. Шаблон наполняется в процессе работы пользователя с системой и фиксируется для каждого конкретного пользователя. Модель включает следующие ключевые понятия: возраст, пол, место жительства, образование, уровень квалификации, степень владения языком программирования, тип характера, ошибки, мотивация, формы проведения тестирования. Некоторые ключевые понятия конкретизируются. Например, существуют формы проведения тестирования: тестово-опросная, тестирование в ходе слежения. Ошибки могут быть синтаксическими и семантическими, когнитивными и моторными. Также в базу знаний включаются утверждения, на основании которых система генерирует соответствующие заключения для формирования модели пользователя.

Основными операциями МОЗ подсистемы являются:

- анализ производимых пользователем действий;
- анализ ошибок, допущенных пользователем, и автоматическое исправление простейших опечаток;
- обработка данных и построение или обновление модели пользователя;
- применение модели пользователя для достижения эффекта адаптации пользовательского интерфейса.

Пользовательский интерфейс дополняется основными адаптационными функциями. В соответствии с моделью пользователя осуществляется подбор диалоговой структуры взаимодействия между разработчиком программ и help-системой, настройка оформительских параметров help-системы, определяется список персональных команд, форма представления информации.

Подсистема управления обучением несет организационную, обучающую и контролирующую функции, организует адаптивный диалог с пользователем. Основной задачей подсистемы является обеспечение пользователю помощи, соответствующей ситуации, и выдача рекомендаций с учетом истории его взаимодействия с help-системой, т.е. осуществление поддержки принятия решений при разработке программного обеспечения для конкретного разработчика программ. При начале работы help-системы используется общая модель пользователя, а затем подсистема управления настраивает эту модель на основании процесса из взаимодействия.

Основными операциями, производимыми подсистемой, являются:

- определение стратегии обучения для минимизации времени обучения,
- переход к новой стратегии;
- генерация тестовых заданий;
- проведение тестирования и интерпретации результатов,
- решение заданий и объяснение способов решений.

Пользовательский интерфейс отслеживает ситуацию, в которой находится разработчик программ, и, используя стратегии помощи и обучения, вмешивается в соответствующий момент или тогда, когда складывается впечатление, что пользователь испытывает трудности в работе [6].

Результаты

В работе представлена модель интеллектуальной help-системы для разработчиков программ, ориентированных на обработку баз знаний. Разрабатываемая система представляет собой консультанта-эксперта по технологии разработки программного обеспечения.

1. Использование help-системы позволит ускорить изучение языков программирования, ориентированных на обработку знаний, и, следовательно, существенно сократить сроки разработки интеллектуальных систем.

2. Разрабатываемая интеллектуальная help-система предоставляет возможность интеграции (объединения) с системами такого же класса в глобальные help-системы по технологиям программирования на языках, ориентированных на обработку знаний.

3. Предложенные принципы построения можно использовать для проектирования help-систем по технологиям программирования на традиционных языках.

Работа выполнена при поддержке БРФФИ – РФФИ (грант № Ф08Р-137).

INTELLECTUAL HELP-SYSTEM MODEL FOR SOFTWARE DEVELOPERS, DESIGNED TO PROCESS KNOWLEDGE BASE

O.V.PIVOVARCHYK

Abstract

The object of consideration is programs that make up the knowledge bases of intelligent systems, and describe ways to solve various problems in the processing of knowledge bases. A model of intellectual help-system for developers of programs for processing knowledge bases is presented. We describe the model of the knowledge base, machine operations processing knowledge, ways of designing the user interface help-system.

Литература

1. *Анурeев И.С., Бородин Е.В., Городняя Л.В. и др.* // Труды 11-й конференции по искусственному интеллекту с международным участием КИИ-2008, Дубна, 28 сент.–3 окт. 2008 г. Т. 3. 2008. С. 199–207

2. *Непейвода Н.Н., Скопин И.Н.* Основания программирования. Москва-Ижевск, 2003.

3. *Amnon H Eden.* // Minds and Machines. Vol. 17, No.2, 2007. P. 135–167.

4. *Голенков В.В., Осипов Г.С., Гулякина Н.А. и др.* Программирование в ассоциативных машинах / Под ред. В.В. Голенкова. Минск, 2001.

5. *Голенков В.В., Елисеева О.Е., Ивашенко В.П. и др.* Представление и обработка знаний в графодинамических ассоциативных машинах / Под ред. В.В. Голенкова. Минск, 2001.

6. *Голенков В.В., Тарасов В.Б., Елисеева О.Е. и др.* Интеллектуальные обучающие системы и виртуальные учебные организации / Под ред. В.В. Голенкова и В.Б.Тарасова. Минск, 2001