ИПМ им.М.В.Келдыша РАН

Абрау-2016 • Труды конференции

А.И. Адамович, Анд.В. Климов

Об опыте использования среды метапрограммирования Eclipse/TMF для конструирования специализированных языков

Рекомендуемая форма библиографической ссылки

Адамович А.И., Климов Анд.В. Об опыте использования среды метапрограммирования Eclipse/TMF для конструирования специализированных языков // Научный сервис в сети Интернет: труды XVIII Всероссийской научной конференции (19-24 сентября 2016 г., г. Новороссийск). — М.: ИПМ им. М.В.Келдыша, 2016. — С. 3-8. — doi:10.20948/abrau-2016-45

Размещена также презентация к докладу

Об опыте использования среды метапрограммирования Eclipse/TMF для конструирования специализированных языков

А.И. Адамович[1], Анд.В. Климов[2]

[1] *Институт программных систем им. А.К. Айламазяна РАН*
[2] *Институт прикладной математики им. М.В. Келдыша РАН*

Аннотация. Авторы делятся опытом разработки специализированных Java-подобных языков программирования на современной платформе Eclipse Text Modelling Framework (Xtext). Обсуждается набор средств, составляющих эту систему метапрограммирования, и трудности их освоения.

Ключевые слова: метапрограммирование, проблемно-ориентированные языки, генераторы парсеров, интегрированные среды программирования, Eclipse, Xtext.

Инструменты для разработки и реализации языков, «компиляторы компиляторов», парсеры и проч., были одной из любимых тем разработчиков инструментов программирования с 1960-х годов. В последнее десятилетие в связи с бурным развитием мощных и удобных сред программирования (integrated development environments, IDE) стали появляться новые платформы для реализации всех требуемых компонентов систем программирования: текстовых редакторов с синтаксическим контролем и подсказками, средств запуска и отладки программ, справочников (help'ов), интеграции с существующими языками, в первую очередь с Java и другими языками на платформе JVM (Java Virtual Machine), и т.д. Такие средства заслужили название систем *метапрограммирования.*

С их помощью чаще всего разрабатываются языки с узкими областями применения и ограниченным числом пользователей, когда важно ускорить и удешевить разработку языка, чтобы автоматизация работы имеющихся

[1] Исследование выполнено в рамках НИР «Методы и средства разработки эффективного программного обеспечения, ориентированные на вычислительные системы, построенные на основе микропроцессоров с многоядерной архитектурой, и формальные основы высокоуровневых языков программирования для суперкомпьютеров с гибридной архитектурой» (№ г/р 01201455360) (госзадание ФАНО России).

[2] Работа поддержана грантом РФФИ № 16-01-00813-а.

пользователей была экономически выгодна. Такие языки получили название *проблемно-ориентированных* (domain-specific languages, DSL) или *специализированных* языков. Спектр DSL простирается от совсем простых табличных языков определения данных до достаточно развитых языков сценариев (scripting languages) и объектно-ориентированных языков типа Java.

В настоящее время среди программистов на языке Java и других языках на JVM наиболее популярны две среды программирования: Eclipse IDE [1] и JetBrains IntelliJ IDEA [3]. Каждая из них содержит средства метапрограммирования:

- Eclipse Text Modelling Framework (Eclipse/TMF), более известная под названием Xtext [2].
- JetBrains Metaprogramming System (MPS) [4].

В последние годы авторам довелось выполнять проект по разработке специализированного языка программирования, похожего на Java и Scala, и системы программирования для него. Мы выбрали для реализации Eclipse IDE [1], поскольку в тот момент предполагалось, что основными пользователями будут Java-программисты с опытом использования этой среды. В данной статье мы делимся опытом освоения Xtext и сопутствующих инструментов и разработки Java-подобного языка на платформе JVM.

Как и весь Eclipse,[3] Eclipse/TMF (Xtext) — открытая система, предоставляемая с исходными текстами.[4] Она содержит следующие инструменты и средства для разработки языков программирования:

- Генератор парсеров с входным языком Xtext, предназначенным для описания грамматик и управления построением внутреннего представления программы (того, что обычно называют *абстрактным синтаксическим деревом*, *abstract syntax tree*, AST). Генератор парсеров сам реализован в рамках Eclipse/TMF. Имеется описание грамматики языка Xtext на нем самом.[5] Это, в частности, позволяет определять свои обработчики текстов на Xtext, например, перевод в удобочитаемый вид в HTML'е для вставки грамматики в документацию (авторы реализовали для себя такой инструмент).
- Описание грамматики компилируется с языка Xtext во входной язык одного из самых мощных современных генераторов парсеров ANTLR [8], позволяющих задавать грамматику в форме LL(*)[6] — более гибкой, чем LALR(1)[7] популярного с 1970-х годов YACC'а.[8]

---

[3] https://github.com/eclipse

[4] https://github.com/eclipse/xtext

[5] https://github.com/eclipse/xtext-core/blob/master/org.eclipse.xtext/src/org/eclipse/xtext/Xtext.xtext

[6] https://en.wikipedia.org/wiki/LL_grammar

[7] https://en.wikipedia.org/wiki/LALR_parser

- Eclipse Modelling Framework (EMF) [9], разработанный фирмой IBM на заре существования платформы Eclipse. Она предоставляет базовые возможности представления *моделей* программ, восходящие к инструментам моделирования UML.[9] Xtext можно рассматривать как языковую надстройку над EMF. По исходному тексту на Xtext'е его компилятор строит *метамодель* данного языка, то есть описание в терминах EMF внутреннего представления, в которое отображается парсируемая программа и в котором происходит ее дальнейшая обработка средствами Eclipse/TMF и программами разработчиков специализированных языков. Eclipse содержит большой набор инструментов над EMF, которые могут использоваться совместно с Xtext.
- Библиотека инструментов и заготовок для программирования алгоритмов разрешения имен и типов реализуемого языка — как универсальные средства для описания любого пространства имен с вложенными областями видимости (scoping), так и специализированные средства представления Java-подобного пространства имен, состоящего из интерфейсов, классов и их членов — полей и методов, а также импорта имен, доступных в JVM. Имеется описание на Xtext'е синтаксиса Java-подобных типов — подъязыка Xtype[10] — и его интеграция со средствами разрешения типов. По нашему опыту, средства разрешения имен и типов — самая сложная для освоения часть. К сожалению, у нее нет достаточно подробного описания, а только учебные пособия (туториалы) по типовым случаям и поверхностное изложение в документации на сайте Xtext'а,[11] а также соответствующие главы в книге [7].
- Инструменты для проведения дополнительных проверок программы, подсказок и быстрого исправления по шаблонам (quick fixes), вызываемых из редактора текста в процессе набора исходной программы. С их помощью можно создавать весьма комфортные средства подготовки программ.
- Определение языка Xbase [2][12] — подъязыка Java-подобных выражений, которые авторы Xtext'а рекомендуют для использования в качестве подъязыка при определении Java-подобных языков. Язык Xbase реализован в рамках Eclipse/TMF с интеграцией с упомянутыми выше

---

[8] https://en.wikipedia.org/wiki/Yacc

[9] https://en.wikipedia.org/wiki/Unified_Modeling_Language

[10] https://github.com/eclipse/xtext-extras/blob/master/org.eclipse.xtext.xbase/src/org/eclipse/xtext/xbase/Xtype.xtext

[11] http://www.eclipse.org/Xtext/documentation/

[12] http://www.eclipse.org/Xtext/documentation/305_xbase.html#xbase-language-ref-introduction

средствами разрешения имен и типов. Имеется формальное описание его синтаксиса на Xtext'е.[13]

- Java-подобный язык программирования Xtend [6] (также реализованный средствами Eclipse/TMF и Xtext), который можно рассматривать как Java с синтаксическим сахаром. (Xbase входит в него как подъязык выражений.) Авторы Xtext'а рекомендуют его для программирования компонентов реализации языков в среде Eclipse/TMF. Xtend включает в себя специальное средство для описания генераторов выходного кода в другие языки — т.н. *богатые строки, rich strings*: строки, в которые могут входит выражения Xtend'а; они могут занимать несколько строк в исходном тексте; при компоновке выходного кода из фрагментов, изображенных в богатых строках, автоматически форматируются отступы по некоторому разумного алгоритму. Это позволяет порождать весьма читабельных код при компиляции со своего языка, например, в язык Java. Синтаксис языка Xtend также описан на Xtext'е.[14]
- Готовый кодогенератор в язык Java из внутреннего представления подъязыка выражений Xbase и элементов Java-программы: интерфейсов, классов, полей, методов. При реализации своего языка, компилируемого в Java, можно либо воспользоваться этими средствами и отображать свой язык в соответствующее внутреннее представление, воспользовавшись имеющимся кодогенератором, либо взять всю работу на себя, запрограммировав свой генератор выходного кода. Если же выходной язык — не Java, то подходит только второй вариант реализации.

Eclipse/TMF организует выполнение компонентов реализации языка. При каждом сохранении исходного файла в редакторе вызываются все фазы компилятора, включая кодогенератор. Пользователь может открыть окно со скомпилированной программой и наблюдать, как она меняется при каждом нажатии на Ctrl-S. (Если в момент сохранения в исходной программе есть ошибки, то выходной файл удаляется и окно с ним исчезает.)

Общее впечатление от среды Eclipse/TMF (Xtext) — положительное: система достойна освоения. Если язык реализован ее средствами, сразу получается весьма качественная реализация в рамках IDE Eclipse с автоматически сгенерированными типовыми средствами, которые можно потом расширять в процессе опытной эксплуатации и развития языка.

Язык Xtext и генератор парсеров, а также приемы программирования кодогенератора на Xtend'е осваиваются за разумное время (1–3 месяца в

[13] https://github.com/eclipse/xtext-extras/blob/master/org.eclipse.xtext.xbase/src/org/eclipse/xtext/xbase/Xbase.xtext

[14] https://github.com/eclipse/xtext-xtend/blob/master/org.eclipse.xtend.core/src/org/eclipse/xtend/core/Xtend.xtext

процессе реализации первого языка). Они достаточно хорошо документированы.

Язык Xtend также производит хорошее впечатление и им хочется пользоваться вместо Java. Документация адекватная.

EMF кажется переусложненным для данной задачи (внутреннего представления программ), но надо помнить, что это универсальное средство моделирования программ в Eclipse, имеющее широкую область применения для анализа и обработки программ помимо Xtext'а.

Наибольшую трудность вызывает освоение плохо документированных и малопонятных компонентов: средств разрешения имен и типов. Мы метались между двумя решениями: по максимуму воспользоваться имеющими средствами или все реализовать самим — но пока так и не можем уверенно порекомендовать выбор. Авторы Xtext'а и их основной пользователь за пределами команды разработчиков — Lorenzo Bettini (автор хорошей книги [7], которая закрывает часть пробелов документации), конечно, предпочитают пользоваться готовыми компонентами. Но достижение аналогичного уровня владения материалом требует года-двух работы над своим проектом.

Для самостоятельного определения системы типов и имен представляет интерес язык Xsemantics [8], являющийся языковой надстройкой над Eclipse/TMF. Однако, мы его еще не пробовали, и неясно, достаточен ли он для определения практических языков или еще находится в стадии красивой академической игрушки.

Конечно, выше мы упомянули не все компоненты Eclipse/TMF, а только основные. Eclipse как универсальная платформа также требует освоения. Мы ограничились обзором лишь основных средств метапрограммирования, то есть создания проблемно-ориентированных языков.

Для ознакомления с другими инструментами метапрограммирования можно порекомендовать книгу [10], излагающую методы и средства построения языков на примере и в сравнении трех систем: Eclipse/TMF [2], JetBrains MPS [4] и Spoofax [5].

Литература

1. *Eclipse IDE*. — URL: http://www.eclipse.org
2. *Eclipse Text Modelling Framework (Xtext)*. — URL: http://www.eclipse.org/Xtext
3. *JetBrains IntelliJ IDEA*. — URL: https://www.jetbrains.com/idea
4. *JetBrains Metaprogramming System MPS: DSL Development Environment*. — URL: https://www.jetbrains.com/mps/
5. *The Spoofax Language Workbench*. — URL: http://www.metaborg.org
6. *Xtend programming language*. — URL: http://www.eclipse.org/xtend
7. Lorenzo Bettini, *Implementing Domain-Specific Languages with Xtext and Xtend*. — Packt Publishing Limited, 2013. —

URL: https://www.packtpub.com/application-development/implementing-domain-specific-languages-xtext-and-xtend
8. Lorenzo Bettini, *Xsemantics - a DSL for writing rules for Xtext languages*. — URL: http://xsemantics.sourceforge.net
9. Terence Parr, *ANTLR (ANother Tool for Language Recognition)*. — URL: http://www.antlr.org
10. Dave Steinberg, Frank Budinsky, Marcelo Paternostro, Ed Merks, *EMF: Eclipse Modeling Framework* (2nd Edition). — Safari Books Online, 2008. — URL: https://www.safaribooksonline.com/library/view/emf-eclipse-modeling/9780321331885
11. Markus Voelter, *DSL Engineering: Designing, Implementing and Using Domain-Specific Languages*. — 2013. — URL: http://dslbook.org