

Инструментальная среда для анализа программных систем

А.М. Полоцев, гр. 63501/13
научный руководитель: к.т.н. доцент В.М. Ицыксон

Санкт-Петербургский Политехнический Университет

XLII Неделя науки СПбГПУ

- В различных методах анализа часто решаются похожие задачи:
 - Построение моделей программы (AST, CFG и т.п.)
 - Построение метрик
 - Реинжиниринг программного обеспечения (оптимизация, рефакторинг и т.п.)
 - Визуализация свойств системы

- В различных методах анализа часто решаются похожие задачи:
 - Построение моделей программы (AST, CFG и т.п.)
 - Построение метрик
 - Реинжиниринг программного обеспечения (оптимизация, рефакторинг и т.п.)
 - Визуализация свойств системы

Обычно эти задачи решаются вручную

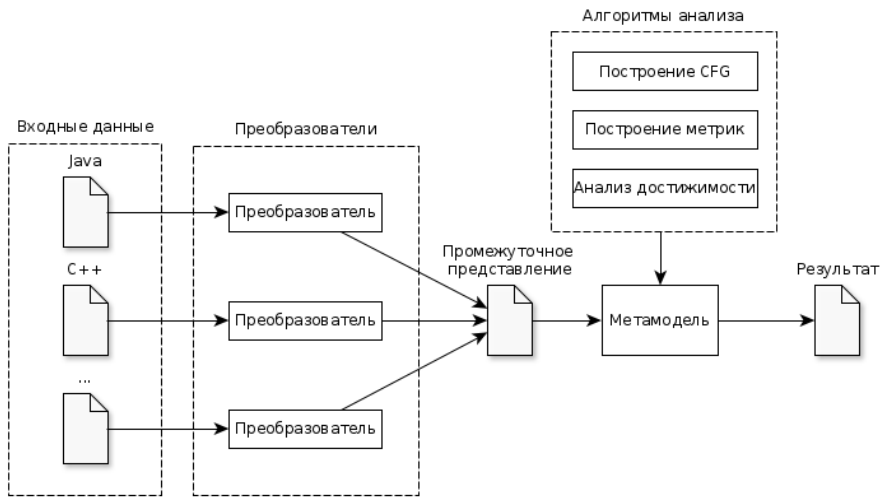
- Цели

- Автоматизация анализа
- Визуализация свойств

- Требования

- Поддержка анализа объектно-ориентированных языков
- Извлечение различных моделей анализируемой системы
- Встроенные средства отображения
- Модульная расширяемая структура
- Экспорт разных артефактов системы во внешние отчеты
- Предоставление API к разработанным методам анализа

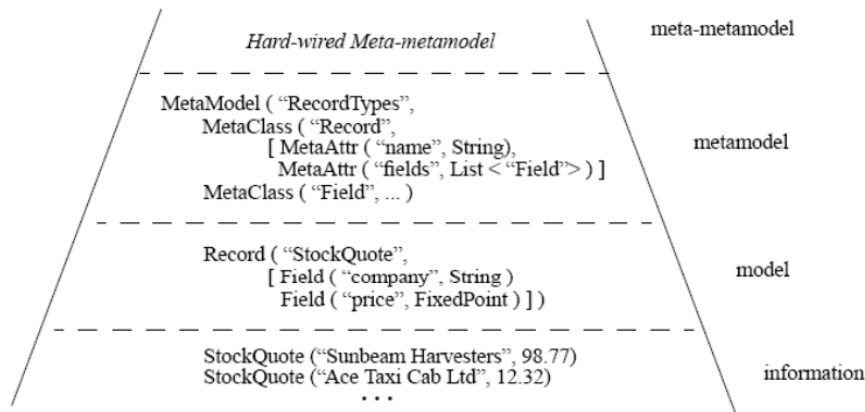
Архитектура инструментального средства



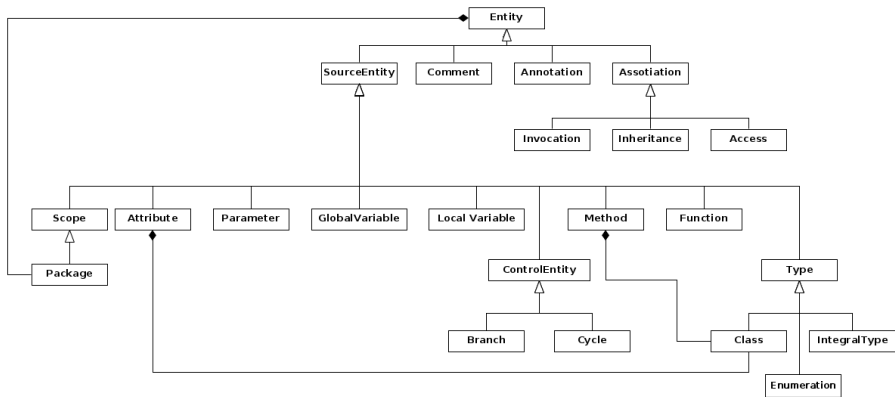
- Цели:
 - Унификация представления системы для проведения анализа и трансформаций
- Требования:
 - Независимость от языка описания анализируемой системы
 - Достаточная мощность для извлечения различных моделей
 - Расширяемость путем добавления новых метаданных

- Meta Object Facility (MOF)
- Стандарт, разработанный Object Management Group (OMG)
- Метаметамодель для описания метамodelей
- Поддержка объектно-ориентированной парадигмы
- Четырехуровневая архитектура
 - Уровень метаметамодели (M3)
 - Уровень метамodelей (M2)
 - Уровень моделей (M1)
 - Информационный уровень (M0)

Разработка метамодели. MOF

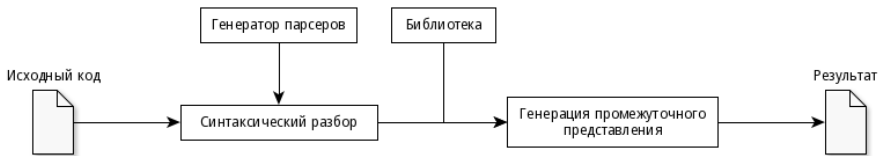


Разработка метамодели. Диаграмма классов



Преобразование исходного кода в метамодель

- Для каждого языка программирования необходимо написать специализированные преобразователи
- Задачи преобразователей - генерация промежуточного представления
- Для написания преобразователей предполагается разработать библиотеку для генерации промежуточного представления



- Требования:
 - Полнота описания
 - Расширяемость
- Формат XMI (XML Metadata Interchange):
 - Разработан OMG
 - Основан на XML
 - Предназначен для сериализации метамodelей

Выбор промежуточного представления. Пример.

```
<?xml version="1.0"?>
<XML xmi.version="1.2" xmlns:UML="org.omg/UML/1.4">
  <XML.header>
    <XML.documentation>
      <XML.exporter>ananas.org stylesheet </XML.exporter>
    </XML.documentation>
    <XML.metamodel xmi.name="UML" xmi.version="1.4"/>
  </XML.header>
  <XML.content>
    <UML:Model xmi.id="M.1" name="address" visibility="public"
      isSpecification="false" isRoot="false"
      isLeaf="false" isAbstract="false">
      <UML:Namespace.ownedElement>
        <UML:Class xmi.id="C.1" name="address" visibility="public"
          isSpecification="false" namespace="M.1" isRoot="true"
          isLeaf="true" isAbstract="false" isActive="false">
          <UML:Classifier.feature>
            <UML:Attribute xmi.id="A.1" name="name" visibility="private"
              isSpecification="false" ownerScope="instance"/>
            <UML:Attribute xmi.id="A.2" name="street" visibility="private"
              isSpecification="false" ownerScope="instance"/>
            <UML:Attribute xmi.id="A.3" name="zip" visibility="private"
              isSpecification="false" ownerScope="instance"/>
            <UML:Attribute xmi.id="A.4" name="region" visibility="private"
              isSpecification="false" ownerScope="instance"/>
            <UML:Attribute xmi.id="A.5" name="city" visibility="private"
              isSpecification="false" ownerScope="instance"/>
            <UML:Attribute xmi.id="A.6" name="country" visibility="private"
              isSpecification="false" ownerScope="instance"/>
          </UML:Classifier.feature>
        </UML:Class>
      </UML:Namespace.ownedElement>
    </UML:Model>
  </XML.content>
</XML>
```

Address
name
street
zip
region
city
country

- Сформулированы задачи анализа, требующие автоматизации
- Разработана архитектура инструментального средства
- Разработана метамодель
- Предложен формат промежуточного представления для экспорта метамодели

- Реализация преобразователя для языка Java
- Разработка методов трансформации и алгоритмов над метамodelью
- Разработка графического интерфейса
- Разработка библиотеки моделей

