Сопряжение языков программирования с предметными посредниками для решения научных задач ♣

© Вовченко А.Е.
ИПИ РАН
itsnein@gmail.com

Аннотация

В статье рассматривается проблема определения адекватного связывания процедурных языков программирования с декларативным языком используемым в предметных посредниках. В работе предложен набор характеристик для оценки различных существующих подходов по связыванию языков программирования и систем управления ресурсами. В статье показано какой набор характеристик должен быть выбран для решения проблемы несоответствия импеданса.

1 Введение

Диссертационные исследования, в рамках которых подготовлена настоящая статья, выполняются в соответствии с концепцией формулирования научных задач и создания информационных систем в среде предметных посредников [1], определяемых в терминах предметной области независимо от существующих информационных ресурсов (баз данных, программных сервисов, процессов). Релевантные задаче неоднородные, распределенные информационные ресурсы регистрируются в посреднике в виде взаимных отображений классов ресурсов и классов посредника, выражаемых декларативно при помощи GLAV взглядов [2]. Спецификации предметных посредников определяются средствами объектно-фреймового языка [3] и типизированного языка логики первого порядка для выражения формул, правил, утверждений и программ на правилах над классами посредника. Поддержка решения задач в среде посредников обеспечивается различными компонентами – посредниками, информационными ресурсами, программируемыми адаптерами, трансформациями, определяемыми отображениями классов ресурсов в классы посредников, прикладными программами над посредниками. Диссертационные исследования посвящены вопросам эффективной организации рассредоточенной поддержки решения задач такими компонентами. Одним из важных вопросов такой организации является определение архитектуры средств сопряжения процедурных языков программирования с декларативным языком предметных посредников.

Не смотря на многочисленные исследования в области интероперабельных архитектур, сопряжения ЯП с базами данных и существующие стандарты (стандарт OMG для сопряжения IDL с ЯП [4], стандарт ODMG 3.0 [5], сопряжения C++ с Oracle (OCCI) [6], сопряжение Java с БД (JDBC) [7], стандарт SQLJ для встраивания языка SQL в ЯП Java [8-9], стандарт Sun JDO[10], Microsoft LINQ[11], и др.), до сих пор не были систематически рассмотрены подходы проектирования и создания подобных сопряжений. Это затрудняет оценку сравнения качества различных подходов к сопряжению, а также создание сопряжения для новых языков и систем. Целью настоящей статьи является предложение системы характеристик (features), которыми можно было бы характеризовать и оценивать разнообразные средства сопряжения ЯП с системами управления различными информационными ресурсами. Основным мотивом разработки такой систематизации является необходимость обоснованного определения адекватной архитектуры сопряжения средств поддержки предметных посредников с языками программирования. При такой систематизации мы ограничимся классом объектно-ориентированных языков.

В следующем разделе представлено описание характеристик для оценки сопряжений ЯП с базами данных. Затем представлен краткая характеристика проблемы несоответствия импеданса при сопряжении языков запросов и языков программирования. Затем описан подход верификации отображений. После чего описывается подход реализуемый в работе.

2 Характеризация сопряжений ЯП с базами данных

Хотя предметом статьи является сопряжение средств поддержки предметных посредников с ЯП, в этом разделе будет рассматриваться и анализироваться сопряжение ЯП с базами данных в силу подавляющего числа исследований, стандартов

Труды 12ой Всероссийской научной конференции «Электронные библиотеки: перспективные методы и технологии, электронные коллекции» - RCDL'2010, Казань, Россия, 2010.

и реализаций в этой области. Общность рассуждений при этом не нарушается т.к. предметные посредники могут рассматриваться как системы управления виртуальными ресурсами.

Отображение типов ЯОД в типы ЯП, включая функции и инварианты типов, должно быть коммутативным [12]. При этом требуется сохранение отношений между типами (такими, как тип-подтип). Под коммутативностью отображения здесь понимается отображение, которое сохраняет операции и информацию исходной модели данных (модели информационного ресурса) при ее отображении в целевую (модель данных ЯП). Коммутативность достигается при условии, что диаграммы отображения ЯОД (схем) является коммутативной. В частности, при доказательстве коммутативности отображения типов необходимо устанавливать факт уточнения [16] операций типов исходной моделей операциями типов целевой модели [13].

Другим важным моментом сопряжения является отображение языка запросов в ЯП. Отображение должно обладать свойством включения (containment) [14]. Для поддержки статического контроля типов требуется расширение ЯП конструкциями языка запросов, в динамике подобного расширения не требуется и запрос можно воспринимать как строку.

Манипулирование может осуществляться как посредством изменения значений долговечных типов, так и посредством специальных операций ЯМД. Для поддержки статического контроля типов, ЯП должен быть расширен конструкциями ЯМД. В динамике этого не требуется, и операции ЯМД можно передавать СУБД в виде строки.

Важным моментом такого отображения является различение в ЯП долговечных и недолговечных типов (и их экземпляров). Поскольку в ЯП эти понятия не определены, ЯП требуют расширения. Отображение классов ЯОД в коллекции ЯП требует изменения семантики коллекций (долговечность) – фактически изменения ЯП. Важным моментом такого отображения является поддержка родовых (generic) коллекций. Для поддержки статического контроля типов требуется, чтобы результирующая коллекция была родовой (Set<Type>).

При таких предположениях множество возможных сопряжений ЯП и СУБД будем определять следующим набором ортогональных характеристик:

Представление конструкций ЯОД в ЯП (DDL Mapping):

Type2Type: Спецификация типа (класса) ЯОД представляется спецификацией типа (долговечной (persistent) коллекцией) ЯП

```
//Specification in DDL (SQL)
CREATE TYPE emp UNDER person AS (
    EMP_ID      INTEGER,
    SALARY      REAL ) INSTANIABLE NOT
FINAL REF ( EMP_ID )
INSTANCE METHOD GIVE_RAISE   (
    AMOUNT     REAL ) RETURNS REAL;
CREATE TABLE empls OF emp_type;

//Type specification in PL(Java)
Class Emp extends Person implements PersistentObject
{
    private int emp_id;
    private float salary;
    //getters and setters for emp_id and salart
    public float give_raise(float amount);
}
DBCollection<Emp>  empls  =  new
DBCollection<Emp>();
```

Type2TypePattern: Спецификация типа (класса) ЯОД представляется значением (объектом) в ЯП, изображающим соответствующую спецификацию типа в ЯП

```
//Specification in DDL (SQL)
CREATE TABLE customer (
    Name char(50),
    Birth_Date date)

//Type specification in PL(Java)
Class Attribute {String attName; String attType;}
Class DBTable {String tableName; List<Attribute>
atts;}
//PL Object
table = {“customer”, atts = {
        {“Name”,”char(50)”}, {“Birth_Date”,”date”}}}
```

PL-Type2DDLType: от типов ЯП к типам схемы базы данных

```
//Specification in C#
[Table(Name="People")]
public class Person {
 [Column(DbType="nvarchar(32) not null", Id=true)]
 public string Name;
 [Column]
 public int Age;
}

//In DataBase the following table is created:
create table People (
   Name nvarchar(32) primary key not null,
   Age int not null,
)
```

Представление конструкций ЯЗ в ЯП (Query Mapping):

Query2String: Запрос представляется строкой.

```
//QL – SQL, Programming Language - Java
OQLQuery query = impl.newOQLQuery();
query.create(
"select t.assists.taughtBy from t in TA where t.salary >
$1 and t in $2 ");
```

Query2QueryPattern: Запрос представляется значением (объектом) в ЯП.

```
//QL - Declarative JDOQL, Programming Language -
Java
Query q = pm.newQuery(org.jpox.Person.class,
"lastName == \"Jones\" && age < age_limit");
q.declareParameters("double age_limit");
```

```
List results = (List)q.execute(20.0);
```

Query2PL-Query: ЯП расширяется конструкциями языка запросов.

```
//QL – SQLJ, Programming language - Java
#sql ordIdIter = { SELECT OrderId FROM otn_deliverydetail };
while (ordIdIter.next()) {
        id = ordIdIter.orderid();
        gui.addToList(id);
}
```

Полнота отображения ЯЗ (Full Query Support):

ClassCompositionQuery: Язык запросов отображается полностью. Предоставляется возможность выразить любой запрос, допустимый в информационном ресурсе, средствами ЯП.

OneClasslQuery: Язык запросов сильно ограничен. Обеспечивается только извлечение объектов одного класса по условию.

Представление конструкций ЯМД в ЯП (Object Manipulation):

DMLoperator2String: Операции манипулирования объектами представляются строкой.

DMLoperator2PL-operator: ЯП расширяется конструкциями языка манипулирования данными.

Поддержка манипулирования долговечными данными в ЯП (Object Manipulation):

PersistentObjects: Долговечные данные поддерживаются, изменение объектов влечет за собой изменение данных в БД.

TransientObjects: Долговечные данные не поддерживаются, изменение объектов не влечет за собой изменения данных в БД.

Поддержка generic коллекций (Collections):

Generic-SetType: Коллекции Set<Type> поддерживаются.

Strict-SetType: Generic коллекции не поддерживаются.

Мы покажем полезность введенной характеризации сопряжений, применяя ее к определению набора характеристик сопряжения, достаточных для обеспечения статического или динамического контроля типов.

Для обеспечения статического контроля типов достаточно реализовать следующий набор решений (статический подход): Type2Type, Query2PL-Query, DMLoperator2PL-operator либо PersistentObjects, Generic-SetType. Для динамического контроля достаточно реализовать следующий набор решений (динамический подход): Type2TypePattern, Query2String, DMLoperator2String, Strict-SetTypet.

В Таблице 1 дана характеризация известных способов сопряжения по предложенным критериям. Подход, выбранный в настоящей работе, включен в таблицу как Synthesis.

Табл. 1. Характеризация известных способов сопряжения по предложенным критериям

	DDL Mapping	Query Mapping	Object Manipulation	Full Query Support	Collections
IDL OMG	Type2Type	No	No	No	No
ODMG 3.0	Type2Type	Query2String	TransientObjects	Composed ClassPlQuery	Strict-SetTypet
JDO	PL-Type2DDLType	Query2String, Query2Query Pattern	PersistentObjects	One ClassPlQuery	Generic-SetType
JDBC	Type2TypePattern	Query2String	DMLoperator2String, TransientObjects	Composed ClassPlQuery	Strict-SetTypet
SQLJ	Type2TypePattern	Query2PL-Query	DMLoperator2PL-operator, PersistentObjects	Composed ClassPlQuery	Generic-SetType
OCCI	Type2Type	Query2String	DMLoperator2String, Transient Objects	One ClassPlQuery	Strict-SetType
LINQ	PL-Type2DDLType	Query2PL-Query	DMLoperator2PL-operator, PersistentObjects, TransientObjects	Composed ClassPlQuery	Generic-SetType
Synthesis	Type2Type, Type2TypePattern	Query2String, Query2PL-Query	PersistentObjects, TransientObjects	Composed ClassPlQuery	Generic-SetType

3 Краткая характеристика проблемы несоответствия импеданса при сопряжении языков запросов и языков программирования

Проблемы несоответствия импеданса подробно описаны в статье Kazimierz Subieta «Impedance mismatch» [15]. Наиболее важные проблемы описаны ниже:

- **Syntax**: Специалист должен одновременно использовать два разных (синтаксически) языка. Одинаковые символы, могут означать разное в разных языках, например, в Java символ “=” означает присваивание, а в SQL – сравнение.
- **Typing**: Типы в ЯП и типы в ЯОД и в языке запросов могут сильно отличаться. Коммутативное отображение типов может окзаться невозможным или неэффективным. Эта проблема актуальна как для простых типов, так и для сложных, абстрактных типов данных.
- **Binding phases and mechanisms (Binding)**: Язык запросов основан на динамическом (времени выполнения) связывании имен в запросе, в то время как в языках программирования используется статическое (времени компиляции) связывание.
- **Name spaces and scope rules (Names)**: Пространства имен в языке запросов и языке программирования различаются. Например, мы не можем использовать переменные из запроса в языке программирования, и не можем использовать переменные ЯП в языках запросов. При этом важной остается задача параметризации запросов. Также важно наличие возможности использования результата запроса как переменной языка программирования.
- **Collections**: Коллекции в базах данных и в языках программирования семантически отличаются. В языках программирования возможности коллекции сильно ограничены. Коллекции, возвращаемые в качестве результата запросов, не имеют явного отображения в языках программирования, и должны обрабатываться специальными конструкциями со своим синтаксисом и семантикой.
- **Persistence**: Языки запросов оперируют долговечными данными, в то время как языки программирования обычно оперируют кратковременными данными (хранящимися в памяти). Объекты в языках программирования, получаемые из БД, имеют образ в базе данных. Таким образом, изменения в объектах должны быть отражены и в БД.
- **Queries and expressions (Queries):** Некоторые запросы и выражения синтаксически могут выглядеть идентично, и при этом иметь разную семантику. Например, в языках запросов 2+2 это запрос, в то время как в языках программирования это – выражение. Запрос не может быть параметром функции, в то время как выражение может.
- **References (Refs):** Для изменения данных в БД нужны ссылки на данные в базе. В языках запросов результат это таблица, а не ссылка. Таким образом в ЯМД, требуется поддержка ссылок на данные в БД, или же должны предоставляться иные возможности изменения данных (например Persistent Objects).
- **Refactoring:** При разработке сложных проектов важным моментом является возможность рефакторинга. В случае, когда запросы трактуются как строки, рефакторинг невозможен.

Табл. 2. Зависимость решения проблемы несоответствия импеданса, от характеристик

	Syntax	Typing	Binding	Names	Collections	Persistence	Queries	Refs	Refactoring
Type2Type		Yes	Yes(w)		Yes(w)				
Type2TypePattern									
PL-Type2DDLType									
Query2PL-Query	Yes		Yes(w)	Yes			Yes		Yes
Query2String									
Query2QueryPattern	Yes								Yes
DMLoperator2PL-operator								Yes	
DMLoperator2String								Yes	
Persistent Objects						Yes		Yes	
Transient Objects									
Generic-SetType									
Strict-SetType					Yes(w)				

Yes – означает что подход с данной характеристикой полностью решает проблему. Yes(w) означает что проблема решается только в сочетании с другой характеристикой. Из таблицы видно, что для решения всех проблем несоответствия импеданса подход к сопряжению должен обладать следующим набором характеристик: Type2Type, Query2PL-Query, Persistent Objects, Strict-SetType. Несложно заметить, что этот набор характеристик соответствует статическому подходу. Видно, что ни один из существующих подходов (Табл. 2) полностью не реализует статический подход. Кроме того из таблицы следует что предложенный набор характеристик полностью покрывает проблемы несоответствия импеданса, а следовательно набор характеристик достаточен.

4 Верификация отображений

Для обеспечения необходимых свойств отображения информационной модели предметных посредников в модель ЯП особое внимание уделяется проведению верификации коммутативности отображения типов. Для этого семантика информационных моделей ЯП и предметного посредника определяется формально в рамках логики первого порядка (нотация абстрактных машин [16]) и обеспечивается доказательство коммутативности отображений, демонстрируя уточнение типов посредника типами ЯП.

Технически отображение моделей реализуется в стиле MDA при использовании языка ATL и метамодели Ecore для представления абстрактного синтаксиса отображаемых моделей. Средства поддержки языка ATL реализованы в виде встраиваемого приложения платформы Eclipse, позволяющего редактировать и исполнять трансформации моделей. В качестве средства доказательства уточнения спецификаций используется инструментарий Atelier B, поддерживающий язык спецификаций AMN (Abstract Machine Notation [16]).

5 Заключение

В работе предложена система характеристик (features), которыми может характеризоваться и оцениваться способ сопряжения ЯП с системами управления информационными ресурсами. В терминах предложенной системы дана характеризация известных способов сопряжения ЯП с СУБД. Показано, как следует выбирать характеристики сопряжения для решения тех или иных проблем несоответствия импеданса. Осуществлен обоснованный выбор набора характеристик для развитого способа сопряжения процедурных ЯП с декларативным языком предметных посредников. Предлагается одновременно реализовать как статический подход, преодолевающий несоответствие импеданса, но накладывающий ряд ограничение на возможности посредников, так и динамический подход, никак не ограничивающий возможности посредников.
Рассмотренные в статье вопросы возникают в контексте общей проблемы рассредоточенной реализации информационных систем в среде предметных посредников.

Литература

[1] Briukhov D., Kalinichenko L., Martynov D., Skvortsov N., Stupnikov S., Vovchenko A., Zakharov V., Zhelenkova O. Application driven mediation middleware of the Russian virtual observatory for scientific problem solving over multiple heterogeneous distributed information resources. Scientific Information for Society -- from Today to the Future: Proc. of the 21st CODATA Conference. -- 2009. -- P. 80-85.

[2] Friedman M., Levy A., Millstein T. Navigational plans for data integration // National Conference on Artificial Intelligence (AAAI) Proceedings, 1999.

[3] Kalinichenko L.A., Stupnikov S.A., Martynov D.O. SYNTHESIS: a Language for Canonical Information Modeling and Mediator Definition for Problem Solving in Heterogeneous Information Resource Environments. Moscow: IPI RAN, 2007. - 171 p.

[4] OMG, IDL to Java Language Mapping, Version 1.3, January 2008

[5] R.G.G. Cattell, Douglas K. Barry, et. al. The Object Data Standard: ODMG 3.0. Morgan Kaufmann Publishers, San Francisco, California

[6] OCCI User Guide, http://download.oracle.com/docs/cd/B28359_01/appdev.111/b28390/toc.htm

[7] Seth White and Mark Hapner, JDBC 2.1 API, Novermber 30, 1999, http://www.informatik.uni-frankfurt.de/java/JDK/JDK_doku/jdk1.3.1/docs/guide/jdbc/spec2/jdbc2.1.frame.html

[8] Jim Melton, (ISO-ANSI Working Draft) Object Language Bindings (SQL/OLB), American National Standard, Information technology — Database languages — SQL — Part 10: Object Language Bindings (SQL/OLB), August 2003

[9] Andrew Eisenberg, Jim Melton, SQLJ-- Part 1: SQL Routines using the Java TM Programming Language, ACM SIGMOD Record, Vol. 28, No. 4, December 1999

[10] JDO documentation, http://java.sun.com/jdo/

[11] LINQ to SQL User Guide, http://msdn.microsoft.com/ru-ru/library/bb386976.aspx

[12] Kalinichenko L.A. Methods and tools for equivalent data model mapping construction. Proc. of the International Conference on Extending Database Technology EDBT'90. LNCS 416. -- Berlin-Heidelberg: Springer-Verlag, 1990. -- P. 92-119.

[13] Kalinichenko L.A., Stupnikov S.A. Constructing of Mappings of Heterogeneous Information Models into the Canonical Models of Integrated Information Systems. Advances in Databases and Information Systems (ADBIS): Proc. of the 12th East-European Conference. -- Pori: Tampere University of Technology, 2008. -- P. 106-122.
[14] Todd Millstein, Alon Halevy, Marc Friedman, Query containment for data integration systems, Journal of Computer and System Sciences, Volume 66, Issue 1, February 2003, Pages 20-39
[15] Kazimierz Subieta, Impedance mismatch, http://www.ipipan.waw.pl/~subieta/SBA_SBQL/Topics/ImpedanceMismatch.html
[16] Jean-Raymond Abrial, The B-book: assigning programs to meanings. Cambridge University Press, 1996

Binding of programming languages with subject mediators for scientific problem solving

© Vovchenko A.E.

Institute of informatics problems of RAS

Definition of an architecture of a procedural programming language (PL) binding with the declarative language used for specification of mediators is discussed in the paper. A set of features to be used for characterization and evaluation of different approaches of PL bindings to information resource management systems is proposed. It is shown how a set of supported features should be selected to solve impedance mismatch problems.

♣ Работа выполнена при частичной финансовой поддержке РФФИ (гранты 08-07-00157-а и 10-07-00342-а).