ВЫБОР ИНСТРУМЕНТА КОДИРОВАНИЯ ПРИ РЕШЕНИИ ЗАДАЧ МАТЕМАТИЧЕСКОГО МОДЕЛИРОВАНИЯ

Л.В. Яблокова

Самарский государственный аэрокосмический университет имени академика С.П. Королёва (национальный исследовательский университет) (СГАУ), Самара, Россия

В статье рассматриваются вопросы, связанные с выбором инструмента кодирования в процессе научно-исследовательской работы на основе анализа идей, понятий и средств, предоставляемых соответствующими парадигмами программирования. Оценивается возможность применения нескольких парадигм программирования на примере исследования концепций, принципов и идиом присутствующих в современных мультипарадигменных языках. Целью статьи является формирование наиболее эффективной системы взглядов и образа мышления у современных специалистов занимающихся научными исследованиями и конструированием программного обеспечения.

Ключевые слова: парадигмы программирования, императивное программирование, процедурное программирование, объектно-ориентированное программирование, обобщённое программирование, анализ общности и изменчивости, мультипарадигменный подход.

Введение

Программирование – это искусство выражать решения задач так, чтобы компьютер мог их осуществить. С другой стороны программы нужно создавать для других людей, стараясь обеспечить при этом их высокое качество. Программирование – это нечто большее, нежели простое следование некоторым правилам, предписаниям и чтению справочника в поисках подходящей языковой инструкции. Чтобы программный код получился хорошим исследователю-программисту необходимо понимать основные идеи, принципы и методы, определяющие общие концепции и конструкции языков программирования. Как и математика, программирование представляет собой полезное интеллектуальное упражнение, оттачивающее мыслительные способности. Как и любая научно исследовательская деятельность программирование – это постоянное экспериментирование с данными, со способом их обработки, представления и интерпретации, проверка предположений и гипотез о возможности применения того или иного подхода в рамках решаемой научной задачи и используемого языка программирования. Формализация такого направления для научно-исследовательской деятельности возможна только с помощью чётких и систематизированных представлений об используемых инструментах, средствах и методиках позволяющих получить на выходе качественные решения научных проблем. Современные языки высокого уровня, в большинстве своем, поддерживают парадигму объектно-ориентированного программирования, использующую иерархии полиморфных типов данных, связанных отношением наследования. Это позволяет, применяя дополнительный уровень абстракции, предоставляемый парадигмой, ссылаться на значение полиморфного объекта, манипулировать им, не указывая точного типа, в рамках иерархии наследования. Правило для такого стиля программирования можно определить как: «Decide which classes you want; provide a full set of operations for each class; make commonality explicit by using inheritance». Обобщённое программирование, как развитие объектно-ориентированного направления, также определяет собственный уровень абстракции. Ключевая абстракция обобщённого программирования представляет собой набор аб-

страктных требований к типам, описываемых с помощью определения абстрактных видов, т.е. семейств абстрактных типов, моделирующих набор требований, который определяет интерфейс и семантическое поведение. Алгоритм, написанный в обобщённом стиле, может применяться для любых типов, удовлетворяющих синтаксическим и семантическим требованиям, которые он предъявляет к своим аргументам. Любой подобный алгоритм состоит из двух частей: конкретных инструкций, определяющих шаги исполнения (императивный и процедурно-операторный стиль), и набора абстрактных требований (определение абстрактных концепций), которым типы его аргументов должны точно соответствовать. Процедурно-операторный подход помогает нам создать конкретную специализацию алгоритма, делая акцент на обработке данных, т.е. на процессе необходимых вычислений. Основой для него служит предоставление механизмов передачи аргументов функциям и получение от них возвращаемого вычисленного значения. Правило для процедурного стиля программирования можно сформулировать так: «Decide which procedures you want; use the best algorithms you can find». Парадигма обобщённого программирования предполагает использование перечня определенных и систематизированных требований к абстрактному типу и представляет собой концепции, описывающие свойства, которыми должен обладать тип, чтобы удовлетворять предъявляемым требованиям. Правило для обобщённого стиля программирования звучит так: «Decide which algorithms you want; parameterize them so that they work for a variety of suitable types and data structures». В информатике – парадигма это комплекс концепций, принципов и абстракций, определяющих соответствующий стиль проектирования и реализации. Это совокупность идей и понятий, определяющих практические правила конструирования, принципы сцепления и связности структурных единиц и распределение обязанностей между ними.

Простые примеры

Парадигма процедурного программирования в объектно-ориентированном языке (Java).

Файл EntryPoint.java:

```
public class EntryPoint
{
	private static void printValue(Object arg)
	{
		System.out.print(arg);
	}
	public static void main(String[] args)
	{
		printValue("Hello! This is "
		+ "Procedural Programming.");
	}
```

```
};
```

Парадигма объектно-ориентированного программирования, с паттерном Dependency Injection (Java).

1. Файл IPrintHelperImpl.java:

```
public interface IPrintHelperImpl

{

	void printValue(Object arg);

};
```

2. Файл PrintHelper.java:

```
public class PrintHelper

{

	private IPrintHelperImpl impl_ = null;


	public void printValue(Object arg)

	{

		this.impl_.printValue(arg);

	}

	public PrintHelper(IPrintHelperImpl impl)

	{

		this.impl_ = impl;

	}

};
```

3. Файл PrintFormattedImpl.java:

```
public class PrintFormattedImpl

implements IPrintHelperImpl

{

	private String format_ = null;


	public void printValue(Object arg)

	{

		System.out.printf(this.format_,

			arg);

	}
```

```
	public PrintFormattedImpl(String format)
	{
		this.format_ = format;
	}
};
```

4. Файл EntryPoint.java:

```
public class EntryPoint
{
	public static void main(String[] args)
	{
		String format = "%s This is Java!";
		IPrintHelperImpl impl =
		new PrintFormattedImpl(format);

		PrintHelper printer =
		new PrintHelper(impl);
		printer.printValue("Hello!");
	}
};
```

Обобщённая парадигма программирования с использованием частичной специализации в мультипарадигменном языке (C++).

1. Файл print_helper.hpp:

```
#ifndef PRINT_HELPER_HPP_
#define PRINT_HELPER_HPP_
#include<stdio.h>
#include<iostream>
struct STDIO_TAG;
struct IOSTREAM_TAG;
template<typename PRINT_TAG, typename ARG>
class print_helper;
template<typename ARG>
class print_helper<STDIO_TAG, ARG>
```

```
{
private:
	const char *format_;
public:
	void print_value(ARG const &arg)
	{
		printf_s(format_, arg);
	}
	print_helper(const char *format) :
		format_(format){}
};
template<typename ARG>
class print_helper<IOSTREAM_TAG, ARG>
{
public:
	void print_value(ARG const &arg)
	{
		std::cout << arg << std::endl;
	}
};
#endif /* PRINT_HELPER_HPP_ */
```

2. Файл main.cpp:

```
#include "print_helper.hpp"
int main(int argc, char *argv[])
{
	print_helper<IOSTREAM_TAG, int> printer1;
	printer1.print_value(21);
	print_helper<STDIO_TAG, double>
	printer2("Real number: %.3f");
	printer2.print_value(21.235123);
	return 0;
}
```

Императивное программирование (ИП):

1. Процесс работы или план вычислений описывается в виде последовательности инструкций, изменяющих состояние программы.
2. Основное понятие – оператор.
3. Основной принцип – принцип последовательного изменения состояния вычислителя пошаговым образом.

Процедурное программирование (ПП):

1. Частный случай императивной парадигмы.
2. Задается последовательность процедур, каждая из которых есть последовательность элементарных действий или вызовов таких же процедур.
3. В качестве базовых элементов используются данные и процедуры.
4. Данные не зависят от процедур.

Объектно-ориентированное программирование (ООП):

1. Основные понятия – класс объектов, структура данных и тип значений.
2. Рассматривает процесс обработки информации как обработку объектов.
3. Позволяет выполнять модификации программы посредством объявления новых подклассов.
4. Полиморфное поведение упрощает логику управления и на уровне кода можно не заботиться о распознавании принадлежности определенного метода конкретному классу, находящемуся на соответствующем уровне иерархии.

Обобщённое программирование (ОП):

1. Суть в таком описании данных и алгоритмов, которое можно применять к различным типам данных и сигнатурам алгоритмов не меняя, при этом, само описание.
2. Поддержку обобщённого программирования обеспечивают шаблоны.
3. Генерирование всех необходимых экземпляров шаблонных элементов программы возлагается на компилятор.
4. Повторное использование кода для библиотек и алгоритмов.

Критерии оценки инструментов кодирования

Табл. 1. Парадигмы и языки программирования

	C	Fortran	Matlab	Java	C#	C++
ИП	±	+	+	±	±	±
ПП	+	+	+	+	+	+
ООП	–	+	+	+	+	+
ОП	–	–	–	±	±	+

Выводы

Каждая парадигма программирования имеет свой круг приверженцев и класс успешно решаемых задач. Для каждой парадигмы характерны ассоциированные с ней приоритеты

для оценки качества программирования, отличия в инструментах и методах работы и соответственно – стиле мышления, и используемых стереотипах. Но ключевым моментом при любом стиле программирования был и остается уровень абстракции, который позволяет рассматривать абстрактные сущности как способ описания конкретных объектов. Абстракция – это важнейшая часть инструментария при конструировании программного обеспечения. Разные парадигмы подразумевают собственные принципы абстракции, общие для множества методик разработки. Каждая методика – это свой способ группировки абстрактных представлений в соответствии с их общими свойствами, включая и закономерности в отличиях отдельных объектов. Результат определения общности должен указывать на повторяющиеся внешние свойства системы, характерные для выбранной предметной области. Но, с другой стороны, выявленная общность помогает упорядочить неявную структуру предметной области, обнаруживаемую при помощи анализа в повторяющихся решениях. Использование нескольких парадигм в научно-исследовательской работе подталкивает нас обращаться к обеим точкам зрения. Анализ общности и изменчивости – это основа мультипарадигменного подхода, поскольку именно таким образом человеческий разум может создавать абстракции. А хорошая техника абстрагирования, учитывая сложность программного обеспечения и реального мира, помогает упорядочивать знания и синтезировать более качественные результаты при решении научных задач.

Литература

1. Кун Т. Структура научных революций. С вводной статьей и дополнениями 1969 г. – М.: Прогресс, 1977. – 300 с.
2. Stroustrup, B. The C++ Programming Language / B. Stroustrup - U.S. Addisson-Wesley, 2013 – 1368 p.
3. Chivers, I. Introduction to Programming with Fortran / Ian Chivers, Jane Sleightholme - Switzerland: Springer, 2012 - 621 p. DOI 10.1007/978-0-85729-233-9.