

## ЭВОЛЮЦИОННЫЙ ПРОЦЕСС РАЗВИТИЯ ЯЗЫКОВ ПРОГРАММИРОВАНИЯ

*Г.И. Шкатова, О.Г. Берестнева  
(г. Томск, Томский политехнический университет)*

### EVOLUTIONARY PROCESS OF PROGRAMMING LANGUAGES

*G.I. Shkatova, O.G. Berestneva  
(Tomsk, Tomsk Polytechnic University)*

As you know, programming languages are a means of knowledge representation for computer systems. The article presents an overview of programming languages, conceptual ideas which have left their mark on the development of programming and used in modern languages at the moment. The main characteristics and properties of programming languages. Conceptual ideas of programming languages, described how to implement them in the semantic structures, as well as properties that characterize the language contribute to the choice of the best in terms of solving a particular problem. Эволюционный процесс развития языков программирования

#### Эволюционный процесс развития языков программирования

В начале 60-х годов 20 века появилось много языков, часть из которых до сих пор представляет значительное явление в программировании [1, 4, 5].

Первый язык программирования высокого уровня FORTRAN создан в 1954 г. в корпорации IBM группой разработчиков во главе с Джоном Бэкусом.

FORTRAN включал самые элементарные средства, многие из которых непосредственно отражали возможности ЭВМ того времени. На нем написано огромное количество библиотек, как в области статистики, так и в области управления спутниками. Поэтому используется до сих пор, ведутся разработки новых стандартов: F2K — 2000г., HPF — HighPerformanceFortran для параллельных суперкомпьютеров со множеством процессоров.

Алгол-60 впервые предоставил для описания алгоритмов согласованное и единообразное множество конструкций, включающее явные операторы управления для представления последовательного выполнения операторов, выбора и итерации, а также достаточно мощный и удобный механизм подпрограмм. Стало возможным разрабатывать программы сверху вниз с помощью последовательного уточнения абстрактных действий.

Язык КОБОЛ заслуживает внимания за развитие такой важной части языков программирования как средства описания и обработки файлов.

Первым универсальным языком программирования был ПЛ/1, объединивший многие возможности и средства языков FORTRAN, Алгол-60, КОБОЛ.

Параллельно с развитием универсальных языков создавались и специализированные языки, в которых за счет сужения области их использования удавалось построить отдельные высокоуровневые механизмы для описания и данных и действий. Так, в языке Лисп допускается лишь один тип данных — список и базовый механизм выполнения — рекурсивные функции. Целью создания Лиспа было использование его в системах автоматического доказательства теорем. Поэтому он называется также языком искусственного интеллекта. К этой группе языков относятся также разработанные позже и вобравшие в себя многие идеи Лиспа языки Пролог, Плэнер. Еще одним специализированным языком, созданным в начале 60-х годов, является язык Снобол. Основным типом данных в Сноболе — строка символов. Он содержит высокоуровневые

средства для обработки строк, в основе реализации которых лежат нормальные алгоритмы А.А. Маркова. Поэтому он относится к марковским языкам. Области применения Снобола являются конструирование компиляторов, символьная математика, обработка текстов, перевод с естественных языков и др. И, наконец, язык АПЛ — мощный и высокоуровневый язык для решения научных задач. Основным типом данных является массив; имеется огромное множество разнообразных средств для описания действий с массивами. Этот язык интерактивный: он позволяет программисту вмешиваться в процесс трансляции и исполнения, внося на любой стадии исправления и изменения, в отличие от пакетных языков, которые не предоставляют таких возможностей.

Первым из значительных этапов в дальнейшем развитии средств описания данных был язык Pascal (Никлаус Вирт, 1970г.). Самым главным в Pascal был принцип, согласно которому данные должны представляться в программе непосредственно в той абстрактной форме, с которой работает программист. Программисту не требуется отображать абстрактные значения на небольшое число примитивных типов данных. Поэтому Pascal предоставляет возможность пользователю придумывать свои собственные типы данных с помощью богатого набора базовых форм, таких как перечислимые типы, записи, массивы и множества. Хотя в Pascal введены новые мощные механизмы построения типов данных, в нем не решены проблемы, как специфицировать множество операций, которые можно применять к этим типам.

Независимо от Pascal разработан язык Симула-67, который относится к языкам моделирования, т.е. используется как средство для имитационного моделирования сложных систем (к этому классу языков относятся также, например GPSS, Симскрипт, Слам). Симула-67 содержит в качестве подмножества Алгол-60. Но что более важно и что поставило этот язык впереди своего времени, он содержит также новую конструкцию — класс. Однако этот язык не располагал таким богатым набором типов данных, как Pascal, что не позволило воспользоваться всеми преимуществами понятия класса. Объединение этих двух важных идей было осуществлено в языке Параллельный Паскаль. Развитие идеи, заложенной в понятие класса, происходило и в рамках других языков. Например, язык SmallTalk основан на понятии объекта, содержащего понятийную и процедурную части. Объекты взаимодействуют между собой, посылая сообщения, представляющие собой требования выполнить те или иные процедуры над объектами-получателями сообщений. Этот язык принадлежит к классу объектно-ориентированных языков программирования. В языке Модуль введены конструкции «модуль», которая управляет доступом к объектам данных, ограничивая область вокруг объекта и операций над ним. Это существенное отличие от понятия класса стало основной идеей на последующих этапах развития языков программирования.

Параллельно с этим в различных языках программирования появились и другие важные идеи. Язык Евклид разработан с целью обеспечить существенную поддержку верификации программ с помощью внесенных в текст программ утверждений о ее выполнении. В языке LIS разработаны механизмы для отдельной компиляции модулей. Язык MESA позволил накопить опыт использования подпрограмм обработки исключений. Во многих языках вводились конструкции параллельного программирования.

Язык С разработан в 1972 году Керниганом и Ритчи и первоначально не рассматривался как массовый. Он планировался для замены Ассемблера, чтобы создавать компактные и эффективные программы, но не зависеть от типа процессора. Язык С также вобрал в себя многие возможности предшественников, но содержит ряд интересных идей, связанных с максимальной полнотой и эффективностью использования ЭВМ.

C++ — объектно-ориентированное расширение C. Создан Бьярном Страуструпом в 1980г.

Java(Ява). Язык создан компанией Sun в начале 90-х годов на основе C++. Он призван упростить разработку приложений C++ путем исключения из него низкоуровневых возможностей. Главная особенность языка — компиляция не в машинный код, а в платформно независимый. Очень популярен в настоящее время. Особое внимание развитие этого языка — поддержка всевозможных мобильных устройств, встраиваемых в бытовую технику и созданию платформно независимых модулей, способных работать на серверах в глобальных и локальных сетях с различными ОС. Разработаны и разрабатываются: языки моделирования (популярен язык графического моделирования UML); языки программирования баз данных; языки программирования для Интернета. Языки: HTML — для Web-страниц, Perl — для генерации текстовых отчетов и управления задачами, VRML — для организации виртуальных трехмерных интерфейсов в Интернете.

Язык Ада вобрал лучшие идеи языков-предшественников. Здесь две основные тенденции развития языков программирования достигли одновременной кульминации: и средства описания данных, и средства описания действий в Аде наиболее мощные, высокоуровневые, концептуально связанные в единое целое среди всех современных универсальных языков.

Развитие средств описания данных и действий продолжается в настоящее время. Однако с начала 80-х годов вышла на первый план новая тенденция развития языков программирования — развитие средств, обеспечивающих автоматическое (автоматизированное) доказательство правильности программ. Такая цель требует пересмотра семантики и ее выражения на языке многих широко используемых в настоящее время языковых возможностей. Кроме этого, необходимо наличие соответствующего программного обеспечения. Таким образом, можно уже говорить не о развитии языков, а о развитии языковых систем, включающих язык, транслятор, методы и процедуры автоматического (автоматизированного) доказательства правильности программ, верификатор и ряд других средств поддержки разработки программ. И раньше, конечно, говорили о системах программирования на том или ином языке. Однако из них вполне можно было выделить сам язык и разрабатывать, изучать его отдельно. При развитии указанной третьей тенденции делать это можно и нужно только в комплексе. Ряд языковых систем подобного рода разработаны, например системы IOTA, AFFIRM, CIP.

Разработаны и разрабатываются среды быстрого проектирования — визуальный подход, при котором все элементы оформления и управления создаются с помощью визуальных компонент, которые перетаскиваются в проектируемое окно, при этом исходный текст программы генерируется автоматически, что позволяет сосредоточиться на логике решаемой задачи.

В настоящее время, существуют программно-аппаратные комплексы, позволяющие организовать параллельное выполнение различных частей одного и того же вычислительного процесса. Для программирования таких систем в языки включаются средства параллельного программирования.

В последнее время в связи развитием Интернет-технологий, широким распространением высокопроизводительных компьютеров и рядом других факторов получили распространение так называемые скриптовые языки, такие как JavaScript (в компании Netscape Communications)- для описания сложного поведения веб-страниц.

Различные языки программирования поддерживают различные парадигмы программирования. Отчасти, искусство программирования состоит в том, чтобы выбрать один из языков, наиболее полно подходящий для решения имеющейся задачи. Разные языки

требуют от программиста различного уровня внимания к деталям при реализации алгоритма, результатом чего часто бывает компромисс между простотой и производительностью (или между временем программиста и временем пользователя).

Язык программирования не просто средство описания алгоритмов. На основе системы понятий языка человек может обдумывать свои задачи, а с помощью нотации может выразить свои соображения по поводу решения задачи. Любой язык, хороший или плохой, ограничивает программиста в выборе способов решения. В одних языках трудно или даже невозможно выразить некоторые понятия; другие языки предоставляют программисту такие возможности, которые облегчают разработку программ.

***Публикация подготовлена в рамках проекта 1957 Гос. задания «НАУКА»  
Министерства образования и науки РФ***

#### ЛИТЕРАТУРА

1. Богатырев Р., Природа и эволюция сценарных языков - МИР ПК – ДИСК. 2005. № 10
2. Головач В. Дизайн пользовательского интерфейса, 2002. – Режим доступа: [www.uibook.ru](http://www.uibook.ru)
3. Казакова А.Е. Методологические основания развития языков программирования. Диссертация по ВАК 09.00.08. – М, 2008. – Режим доступа: [dissertCat.com](http://dissertCat.com).
4. Неклюдова С.А., Балса А.Р. Парадигмы программирования как инструменты разработчика программных систем. Межвузовский сборник научных трудов: Информационные технологии и системы. Выпуск 1(12). Санкт-Петербург, 2014 год.
5. Теслер Г.С. Новая кибернетика, Киев: Логос, 2004 - [immsp.kiev.ua](http://immsp.kiev.ua)
6. Эккель Б. Философия C++. Введение в стандартный C++. 2-е изд. – СПб.: Питер, 2004.– 572 с.