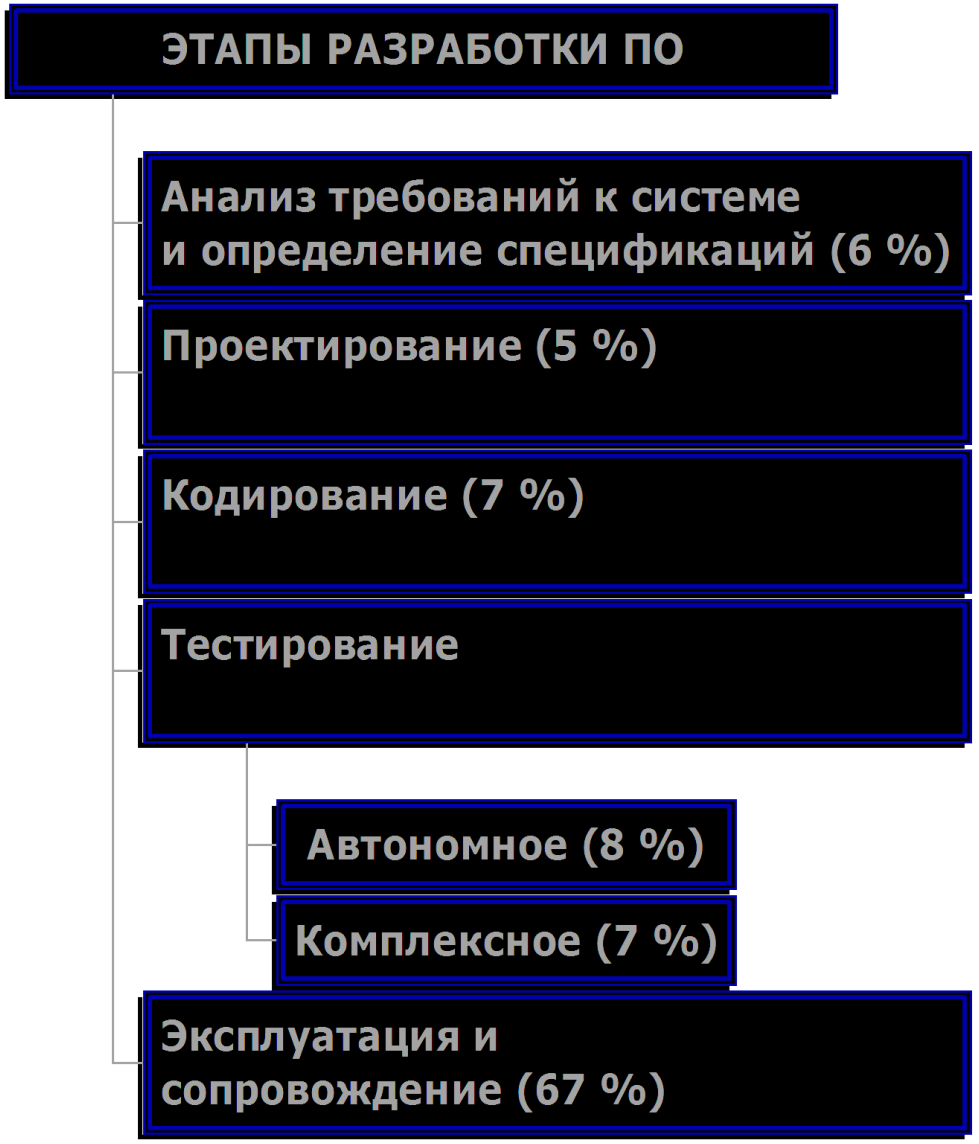
Глава 3. ЭТАПЫ РАЗРАБОТКИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ И ЯЗЫКИ ПРОГРАММИРОВАНИЯ

- Программное обеспечение
- Этапы разработки программного обеспечения
- Метод пошаговой детализации
- Языки программирования низкого уровня и высокого уровня
- Императивные, объектно-ориентированные, функциональные, логические языки программирования
- Классификация языков программирования
- Типы трансляторов
- Схема компилятора
- Языки веб-программирования

ЭТАПЫ РАЗРАБОТКИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Программное обеспечение (пакет программ) – группа взаимосвязанных и взаимодействующих программ, предназначенных для решения любой задачи из конкретной области (например, для моделирования технологических процессов, для выполнения графических работ). Программы какого-либо пакета рассчитаны на совместное использование в различных комбинациях друг с другом.



ЭТАПЫ РАЗРАБОТКИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

I. Анализ требований к системе и определение спецификаций

Первичный документ – постановка задачи. Результат (после нескольких итераций) – **техническое задание**, в котором, как правило, содержатся название системы, цели создания, характеристика области применения, требования к системе в целом (интерфейс, особые требования к отдельным модулям, безопасность, и т.д.), информационная база (структура базы данных, с которой будет взаимодействовать программная система), программное обеспечение (обоснование выбора языка программирования), техническое обеспечение, описание данных для тестирования.

На основании технического задания формируются **спецификации** – описание количества и режимов работы модулей, их взаимодействия.

ЭТАПЫ РАЗРАБОТКИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

II. Проектирование

На данном этапе разрабатываются **алгоритмы**, задаваемые спецификациями, и формируется общая структура будущей программы путем детальной проработки последовательности ее действий. Запись – с помощью блок-схем или псевдокода.

Применяются различные технологии – структурное программирование, объектно-ориентированное программирование и др. В рамках структурного программирования используется метод **пошаговой детализации**, при котором процесс преобразования исходных данных в результат вначале представляется в виде последовательности небольшого числа простых этапов (задач). На следующем шаге задачи разбиваются на последовательность подзадач нового уровня и т.д. Детализация заканчивается, когда каждый отдельный этап может быть простым способом записан на выбранном языке, или представляет собой известную задачу, для которой уже имеется готовая программа.

III. Кодирование

Кодирование представляет собой реализацию разработанных алгоритмов, составление по ним текстов программы с использованием конкретного языка программирования. Включает процесс трансляции – перевода программы в последовательность машинных команд (машинный код).

ЭТАПЫ РАЗРАБОТКИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

IV. Тестирование

При **автономном тестировании** каждый модуль проверяется отдельно. При этом программная среда модуля имитируется с помощью программы управления тестированием, содержащей фиктивные программы вместо реальных подпрограмм, к которым имеется обращение из данного модуля.
При **комплексном тестировании** производится совместная проверка групп программных компонентов.

В процессе тестирования происходит оптимизация системы (разгрузка участков повторяемости – циклов, замена сложных операций на более простые, экономия памяти и т.д.).

V. Эксплуатация и сопровождение

На данный этап приходится основная часть расходов, затрачиваемых в течение жизненного цикла системы.
Причины выпуска новых версий (модификаций) ПО:

- необходимость исправления ошибок, выявленных в процессе эксплуатации;
- необходимость совершенствования, например, улучшения интерфейса или расширения состава, выполняемых функций;
- изменение среды (появление новых технических средств и/или программ).

ЯЗЫКИ ПРОГРАММИРОВАНИЯ

Языки программирования – это тщательно составленные последовательности слов, букв, чисел и мнемонических сокращений, используемые для общения с компьютером.

Языки низкого уровня

Машинная команда состоит из кода операции и адресной части. Код операции указывает вид выполняемого действия (сложить, переместить и т.д.). Адресная часть содержит адреса (номера) ячеек памяти, в которых расположены операнды, и адрес ячейки, куда следует поместить результат.

Языки ассемблера отличаются от машинного кода тем, что коды операций заменены буквенными обозначениями (например, ADD – сложить, MOV – переслать данные) и вместо номеров ячеек используются символические адреса.

Ассемблер – программа, преобразующая мнемонику языка ассемблера непосредственно в двоичные представления машинных команд.

Языки ассемблера – машинно-зависимые языки низкого уровня, в которых одна команда соответствует одной машинной команде (возможны макрокоманды – вызовы макросов, объединяющих несколько машинных команд).

ЯЗЫКИ ВЫСОКОГО УРОВНЯ

Языки программирования, имитирующие естественные языки и способные на основании одного предложения строить несколько команд компьютера, принято считать **языками высокого уровня**.

Варианты языков – подмножества, расширения, диалекты.

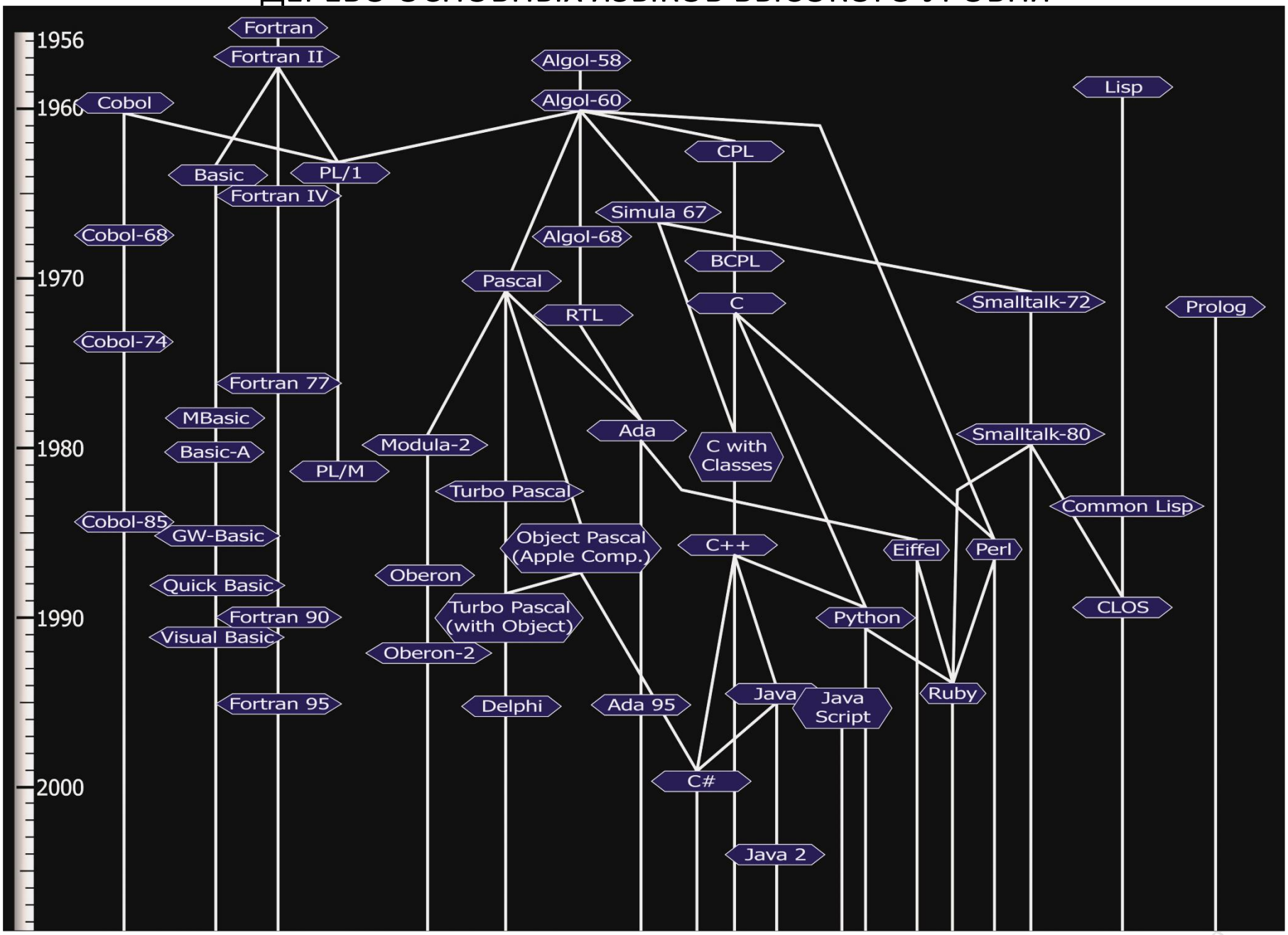
Критерии оценки (выбора) языка программирования:

- легкость чтения программ (особенно важна с точки зрения эксплуатации и сопровождения программ);
- легкость создания программ в выбранной области;
- надежность (т.е. насколько программа соответствует своему предназначению в любых условиях).

Основные характеристики языка:

- простота;
- структурированность;
- доступные типы и структуры данных;
- "естественность" синтаксиса;
- поддержка абстракции;
- выразительность;
- проверка совместимости типов;
- обработка исключительных ситуаций.

ДЕРЕВО ОСНОВНЫХ ЯЗЫКОВ ВЫСОКОГО УРОВНЯ



ИМПЕРАТИВНЫЕ ЯЗЫКИ

Основными элементами **императивных** языков программирования являются переменные (моделируют ячейки памяти), операторы присваивания (определяют и изменяют содержимое ячейки памяти), итеративная форма повторений (циклы). На использование этих языков ориентирована технология структурного программирования, базирующаяся на **процедурной декомпозиции**.

FORTRAN (FORmula TRANslator). Считается первым компилируемым языком высокого уровня. Ориентирован на создание программ, выполняющих естественно-научные и математические расчеты.

COBOL (COmmon Business Oriented Language). Структура и словарь близки к обычному английскому языку. Является основным языком в США для обработки данных в таких учреждениях как банки и страховые компании. Раздел данных – сильная сторона языка COBOL, тогда как раздел процедур – относительно слабая.

BASIC (Beginner's All-purpose Symbolic Instruction Code). Исходный BASIC был легок для изучения начинающими и не требователен к ресурсам компьютера. Популярный язык микрокомпьютеров в конце 70-х и начале 80-х годов. Важнейшим аспектом была его ориентация на использование удаленного доступа к компьютеру посредством терминала. Возрождение языка BASIC произошло в начале 90-х годов с выходом языка Visual BASIC (Microsoft, 1991), который является одним из основных языков в платформе Microsoft.NET.

ALGOL

ALGOL (ALGOrithmic Language). Язык ALGOL – результат попытки создания универсального языка. В этом языке была формализована концепция типов данных, реализована идея составных операторов. В диалекте ALGOL 60 была введена концепция блочной структуры, что позволяло программистам локализовать части программы, вводя разные области видимости. Имелась возможность передавать параметры подпрограммам по значению и по имени, а также создания рекурсивных процедур. Были доступны динамические массивы (ALGOL 68), т.е. массивы, диапазон индексов которых задавался переменной, которая могла менять свое значение во время работы программы. Свыше 20 лет ALGOL оставался единственным официальным средством представления алгоритмов в научной литературе.

ИМПЕРАТИВНЫЕ ЯЗЫКИ – ПОТОМКИ ЯЗЫКА ALGOL

Pascal. Обеспечивает возможность создания больших программ, поддерживая их строгую логическую структуру. Для коротких программ может оказаться слишком громоздким. Считается важнейшим инструментом для обучения методам структурного программирования.

C. Отличная замена ассемблеру для низкоуровневого программирования, с одной стороны, и применения принципов структурного программирования – с другой. Популярен благодаря многим решениям, сделавшим запись программы на С весьма компактной. В целом С – очень мощный и в то же время изящный язык. Одной из особенностей является отсутствие полной проверки типов, что порождает гибкость языка в сочетании с ненадежностью.

Ada. Происходит от Pascal, но заметно сложнее его. Содержит средства выделения в отдельные модули объектов данных, обеспечивает поддержку использования абстракции данных в структуре программы. Содержит средства обработки исключительных ситуаций. Программные блоки могут быть настраиваемыми. Обеспечивает параллельное выполнение программных блоков (заданий).

Modula-2. По сравнению с Pascal добавлена поддержка модулей. Имеются абстрактные типы данных, возможность использования процедур как типов, низкоуровневые средства системного программирования и сопрограммы.

ОБЪЕКТНО-ОРИЕНТИРОВАННЫЕ ЯЗЫКИ

Объектно-ориентированные языки программирования в наибольшей степени способны реализовать технологию объектно-ориентированного программирования, которая основана на **объектной декомпозиции**.

Smalltalk. Основные идеи происходят от первого ООЯ SIMULA 67. Программными модулями языка Smalltalk являются объекты – структуры, объединяющие локальные данные и набор операций (методы), которые доступны другим объектам. Метод определяет реакцию объекта на определенное сообщение, соответствующее данному методу. Абстракциями объектов являются классы. Экземпляры классов – объекты программы. Имеется иерархия классов. Подклассы наследуют функциональные возможности и переменные родительского класса, имея возможность добавлять новые функциональные возможности, а также изменять или скрывать унаследованные.

C++. Представляет собой надстройку над языком C, поддерживающую большинство возможностей, открытых языком Smalltalk (т.е. C++ – это объединение императивного и объектно-ориентированного языков). Поддерживается наследование и множественное наследование. Динамическое связывание обеспечивается функциями виртуального класса. Недостатки: объемность и сложность.

ОБЪЕКТНО-ОРИЕНТИРОВАННЫЕ ЯЗЫКИ

Смешанные языки, созданные путем введения в соответствующий императивный язык объектно-ориентированный поддержки: **Oberon, Delphi, Ada 95.**

Java. Является непосредственным наследником C++. Отличается от него отсутствием некоторых потенциально ненадежных механизмов, а также тем, что устранены любые, не относящиеся к объектно-ориентированному программированию, средства. Все объекты в этом языке – динамические. Особенность языка Java состоит также в использовании особого вида трансляции – динамической кодогенерации. Основные девизы Java – простота, объектная ориентированность, сетевые возможности, надежность, независимость от архитектуры, интерпретируемость.

C#. Разработан в Microsoft как часть новой технологии .NET. Устранены некоторые недостатки языка Java: присутствуют перечисления, статические структуры, многомерные массивы, передача параметров по ссылке. Предусмотрена единая среда выполнения программ (Common Language Runtime, CLR), написанных на разных языках. Трансляция основана на использовании промежуточного кода (Intermediate Language, IL). Компиляция увеличивает быстродействие.

ФУНКЦИОНАЛЬНЫЕ И ЛОГИЧЕСКИЕ ЯЗЫКИ

LISP (LISt Processing). Разрабатывался в связи с работами в области искусственного интеллекта. В чистом языке LISP (1958 г.) существовало только два типа структур данных: атомы и списки. Списки представляли собой совокупность узлов, каждый из которых представляет собой два указателя. Первый указывает на представление атома, т.е. на его символьное или числовое значение, а второй – на следующий элемент списка.
Все вычисления в этом языке, который разрабатывался как язык **функционального программирования**, производятся путем применения функций к аргументам. Итеративные процессы определяются с помощью рекурсивных функций.

Prolog (PROgramming LOGig) – язык **логического программирования**, основная идея которого состоит в использовании формальной логической записи для сообщения компьютеру вычислительных процессов. Программирование в нем является непроцедурным. Программы не устанавливают точно, как должен вычисляться результат, а только описывают его форму. Транслирующая система сама должна выбрать нужный порядок выполнения команд, который приведет к желаемому результату. Логическое программирование было использовано, главным образом, в системах управления реляционными базами данных, экспертных системах и обработке текстов на естественных языках.

КЛАССИФИКАЦИЯ ЯЗЫКОВ ПРОГРАММИРОВАНИЯ

По парадигмам (система идей и понятий, определяющих стиль программы) – императивные, функциональные, логические, объектно-ориентированные.

По факту создания процесса – языки, создающие процесс (после запуска программы создается отдельный процесс выполнения этой программы, C, C++, Pascal, Delphi и др.), сценарные языки (сценарий, или скрипт – программа, которую выполняет другая программа, PHP, Python, Ruby и др.)

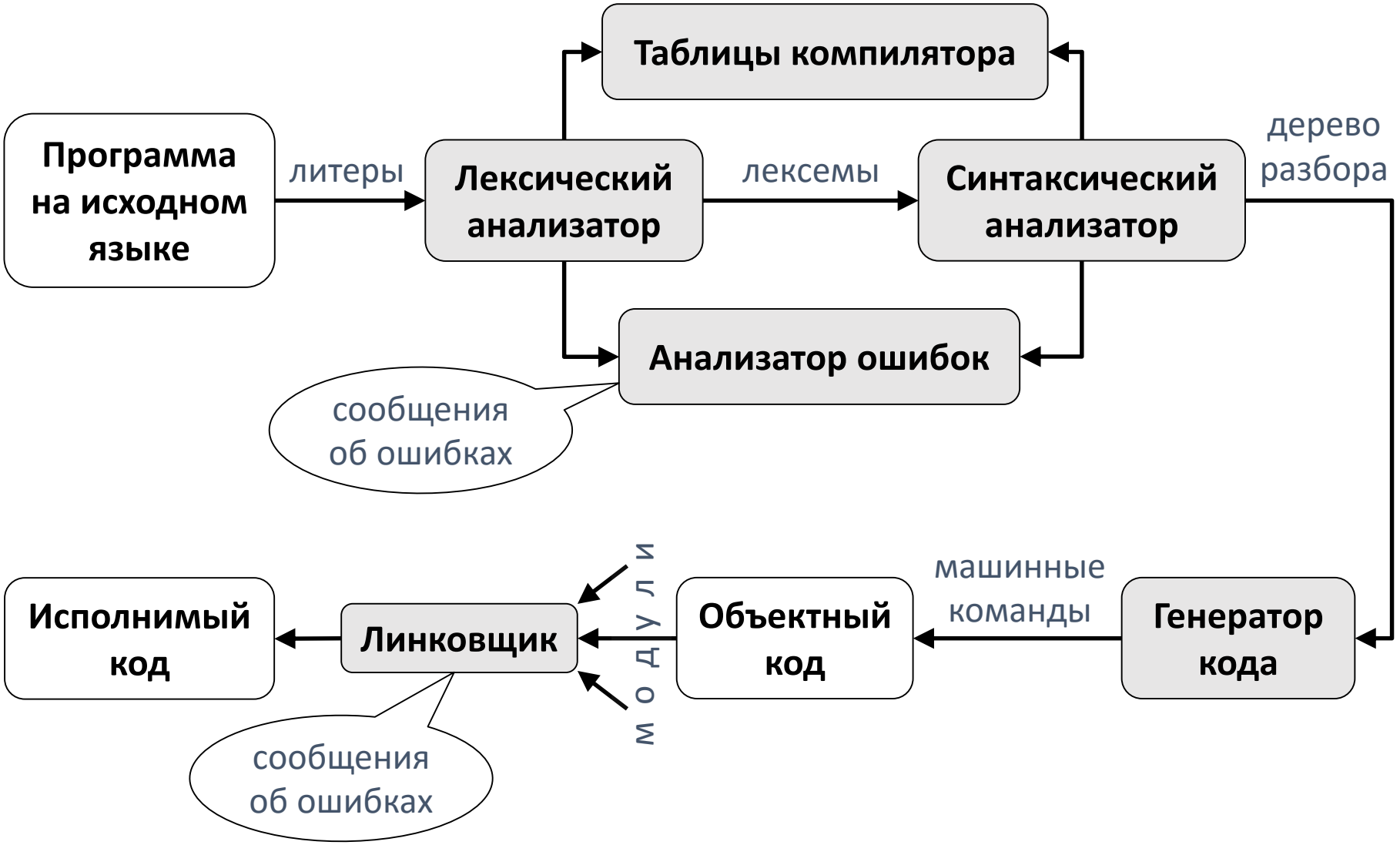
По поколениям (Generation Language, GL) –

- **1 GL** (время появления – 40-50 гг.). Языки машинных команд. Первый ассемблер (названия команд в символическом виде, десятичный и шестнадцатеричный формат чисел).
- **2GL** (конец 50-х – начало 60-х гг.). Символический ассемблер (включал понятие переменной, требовался компилятор).
- **3GL** (60-е годы). Универсальные языки высокого уровня (FORTRAN, COBOL, ALGOL, Pascal, C и др.) Современные системы программирования (Delphi, Visual BASIC, Java Development Kit и др.) включают поддержку объектно-ориентированной технологии и другие инструменты.
- **4 GL** (начало 70-х г.). Непроцедурные языки, ориентированные на конкретную область применения, в частности, на работу с базами данных (SQL, Structured Query Language и др.). Команды языков 4GL близки к обычному языку.
- **5 GL** (наст. время). По одной из точек зрения, декларативные языки.

ТРАНСЛЯТОРЫ

Трансляция – перевод программы, написанной на языке программирования, в последовательность машинных команд.

СХЕМА КОМПИЛЯТОРА

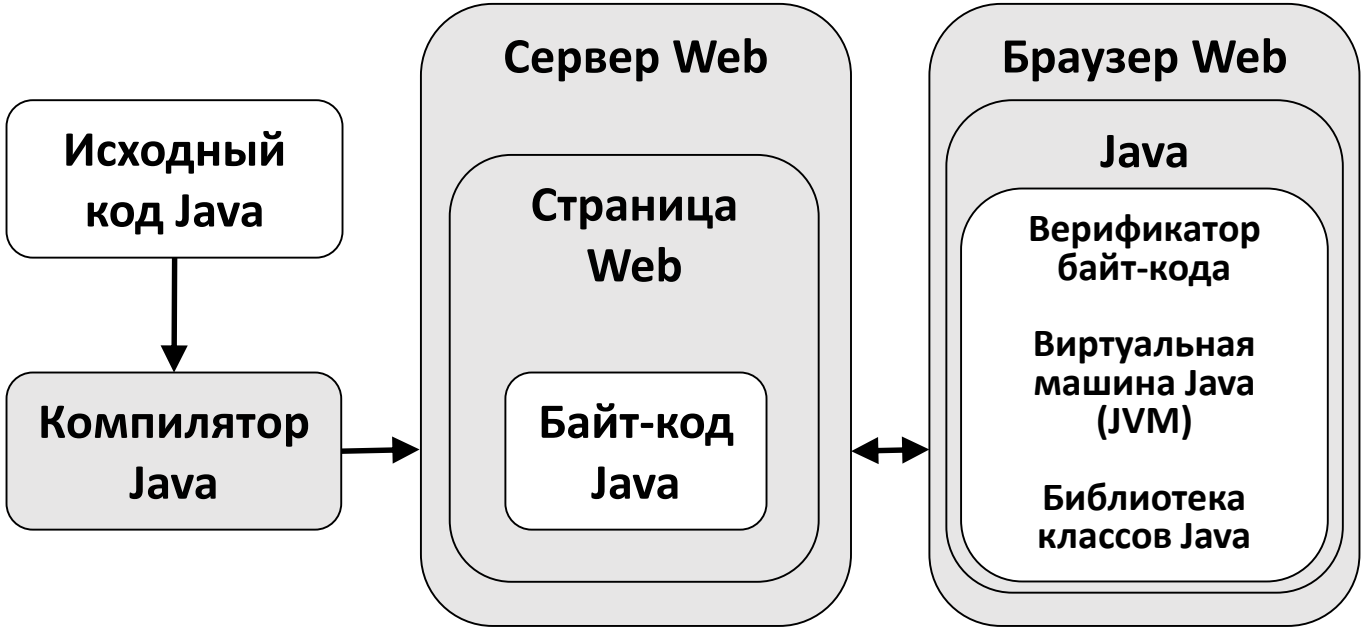


Лексема – минимальная значимая единица текста программы (идентификатор, специальный символ, константа, зарезервированное слово и др.).

ЯЗЫКИ ВЕБ-ПРОГРАММИРОВАНИЯ

Языки веб-программирования в основном предназначены для работы с интернет-технологиями. Делятся на две группы: клиентские (программы на клиентских языках обрабатываются на стороне пользователя, как правило их выполняет браузер – JavaScript, VBScript, Java и др.) и серверные (вызванная из браузера страница обрабатывается на сервере, то есть выполняются все программы, связанные со страницей, и потом возвращается к клиенту по сети в виде файла – PHP, Perl, Python, Ruby, любой .NET язык, Java и др.).

Выполнение Java-апплета



Java-апплет – прикладная программа, чаще всего написанная на языке программирования Java в форме байт-кода.

ЯЗЫКИ ВЕБ-ПРОГРАММИРОВАНИЯ

	Исполнение на сервере	*Исполнение у клиента*
Программа размещается внутри HTML-кода	Активные серверные страницы: **PHP**, **ASP**, **JSP**, …	Скрипты: **JavaScript**, **VB Script**, …
Программа размещается отдельно	CGI и сервлеты: **Perl**, **C**, **Pascal**, **Java**, …	Апплеты: **Java**, **Oberon-2**, …

Идея **WWW** (World Wide Web – всемирная паутина) базируется на языке описания гипертекста HTML (Hypertext Markup Language) и протоколе передачи гипертекста (HTTP). HTML следует считать языком внешнего представления Web-страниц (языком разметки), а не языком программирования.