Министерство образования и науки Российской Федерации

САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ

РЕФЕРАТ

История развития популярных языков программирования.

Выполнили: Калакуцкий А.В., гр.: 4125
Мартьянов С.С., гр.: 4125
Толстиков Н.С. гр.: 4120

Санкт-Петербург
2012

1 Аннотация

Мы живем в очень интересное время. Время, когда у практически у каждого есть мобильный телефон в кармане, персональный компьютер на рабочем столе, телевизор на кухне и практически неограниченное «облачное» хранилище информации. Время, когда со всех сторон нас окружают машины. А что же превращает эти машины из груды железа в вещи, к которым мы так привыкли?
Эту важную роль на себя берет программное обеспечение. ПО играет роль души, которая делает разнообразные устройства так близки нам.
Для того что бы сделать программное обеспечение наиболее удачным, нужен хороший инструмент. Инструментом для создания ПО является язык программирования.
На сайте The Language List[1] сейчас представлено около 2500 языков, но сколько из них реально используется и почему только несколько их них получили широкое распространение в среде программистов?
В нашей работе мы поставили цель, выяснить какие факторы привели к появлению и распространению современных наиболее популярных и интересных нам языков программирования. Для достижения цели нужно ответить на вопросы: как развивались языки программирования, какие области программного обеспечения они охватывали, какие парадигмы программирования поддерживали и какие факторы повлияли на популяризацию данного языка программирования. Для ответов на данный вопрос нужно решить следующие задачи. Первое, какие языки имеют наибольший успех среди программистов. Второе, составить хронологическую модель развития этих языков, постараться определить факторы, из-за которых он получил свою репутацию.

2 Обзор источников

В настоящее время существует огромное количество источников информации по истории языков программирования, это и различные книги посвященыые истории ([1], [2]), так и просто книги о языках, различные справочники и самоучители, в которых приводся краткая история рассматриваемого интсрумента ([3], [4], [5]). Благодаря тому, что сам предмет достаточно молодой, то есть возможность лично беседовать с теми кто оставил свой след в истории программирования. Здесь можно отметить серию интервью с авторами языков, размещенную на сайте computerworld ([6], [7]).
Но никто, или почти никто, не делает попыток определить причины популярности того или иного языка, той или иной технологии. Можно встретить только отдельные записи. Например в статье [8], сделан анализ причин превосходства в популярности языка PHP над Perl, но более общие обзоры не встречаются.

3 Рейтинги популярности языков программирования

Первые десять страниц из результатов выдачи поисковой системы Google по запросу "рейтинг популярности языков программирования"прямо или косвенно ссылаются на следующие три рейтинга:

- Сравнение языков ohloh.net(ohloh.net Language Comparison) [9]

- Индекс программистского сообщества TIOBE (TIOBE Programming Community Index) [10]

[1]Киннерсли Б. “Language List” Официальный сайт Канзаского университета URL:http://people.ku.edu/ nkinners/LangList/Extras/langlist.htm Дата обращения: 18.09.2012

- Рейтинги языков программирования RedMonk (The RedMonk Programming Language Rankings) [11]

Эти рейтинги мы и взяли в своей работе для определения самых популярных языков программирования.

3.1 Индекс программистского сообщества TIOBE

Компания TIOBE занимается оценкой качества программных систем на основе оценки этих систем на соответствие своим стандартам программирования. [2]

Для попадания в рейтинге TIOBE язык программирования должен отвечать двум требованиям [3]

1 На википедии должна быть страничка посвященная этому языку.

2 Язык программирования должен быть полным по Тьюрингу[4]. Соответсвенно такие языки как HTML, XML, SQL не учитываются данным рейтингом, но различные расширения языка SQL, например PL/SQL, входят в рейтинг.

Рейтинг расчитывается на основе подсчета количества результатов в выдаче поисковых систем при запросе «называние языка> programming". Данное число домножается на специальный коэффициент, установленный для каждой используемой поисковой системы:

- Google: 30%
- Blogger: 30%
- Wikipedia: 15%
- YouTube: 9%
- Baidu: 6%
- Yahoo!: 3%
- Bing: 3%
- Amazon: 3%

Количество результатов в выдаче определяет рейтинг языка. Рейтинги нормализуются по сумме результатов для первых пятидесяти языков. Таким образом первые 50 языков в сумме будут иметь рейтинг 100%. После для каждого результата подсчитывается количество ложных совпадений, таких как выдачи по запросу "Basic programming"страницы "Improve your basic programming skills in Java". Подсчитывается процент таких промахов для первых ста страниц выдачи, дальше этот процент вычитается из общего количества результатов.
Так же вычисляются статусы языков на основе изменния их рейтинга, но нас это мало волнует.
Самые популярные языки программирования по версии TIOBE на декабрь 2012 года:

[2]Компания TIOBE, “TIOBE Company” Официальный сайт компании TIOBE URL:`http://www.tiobe.com/index.php/content/company/GeneralInfo.html` Дата обращения: 19.12.12

[3]Компания TIOBE, “TIOBE Programming Community Index Definition” Официальный сайт компании TIOBE URL:`http://www.tiobe.com/index.php/content/paperinfo/tpci/tpci_definition.htm` Дата обращения: 19.12.12

[4]Определение этого понятия дано в разделе 4.1

1 c

2 Java

3 Objective-c

4 c++

5 PHP

6 (Visual)Basic

7 Python

8 Perl

3.2 Сравнение языков ohloh.net

Ohloh — публичный ресурс посвященный свободному и открытому программному обеспечению. Он предоставляет возможности для общения разработчиков, аналитику и поиск по проектам [5].

На странице проекта можно вывести и просмотреть график отражающий популярность языков.[6]

График показывает результаты для языков выбранных пользователем. Значения на графике — это сумма все коммитов за месяц, которые содержат хотя бы одну строчку на запрашиваемом языке. Если в коммите несколько языков, то он защитывается в пользу всех. Информация собирается уже более 20 лет, и не включает текущий месяц, т.к. он еще не закончился.

Самые популярные языки программирования по версии ohloh.net на декабрь 2012 года:

1 c

2 java

3 c++

4 python

5 php

6 ruby

7 perl

8 objective-c

Мы специально исключили языки не рассматриваемые рейтингом TIOBE (html, sql, css), чтобы добиться соответствия результатам различных рейтингов, это относится и к следующему рассматриваемому нами рейтингу.

[5]Black Duck,"About Ohloh" Официальный сайт сообщества Ohloh URL:`http://meta.ohloh.net/us/` Дата обращения: 3.01.2012

[6]Black Duck, "Language Comparison Page" Официальный сайт сообщества Ohloh URL:`http://meta.ohloh.net/compare_languages/` Дата обращения: 19.12.12

3.3 Рейтинги языков программирования RedMonk

RedMonk — консалтингова фирма занимающаяся исследованиями рынка разработки программного обеспечения [7].

Их методика составления рейтинга весьма и весьма проста, впервые она была предложена Дрю Конвеем (Drew Conway) [12] в 2010 году. С тех пор уже второй год подряд компания RedMonk обновляет статистику.
Их рейтинг предствляет из себя график осями которого является: популярность языка на GitHub [8], одном из самых популярных сервисов для размещения программного кода, и колисчество вопросов имеющих отеношение к этим языкам на Stack Overflow [9], одном из самых популярных сервисов вопросов и ответов для программистов. Для расчета популярности на Stack Overflow ведется подсчет тэгов. А на GitHub есть свой собственная система подсчета количества проектов на каждом из языков.

Самые популярные языки программирования по версии Redmonk на декабрь 2012 года:

- Java
- C
- C++
- Perl
- Python
- Ruby
- Objective-C
- PHP

3.4 Выбор популярных языков

Можно заметить, что не смотря на выбор рейтинга популярные языки программирования везде остаются примерно одни и те же, лишь меняя своё место, лишь в первом из рейтингов Visual Basic занимает место Ruby. Так что самыми популярными языками можно назвать:

- C
- Java
- C++
- Objective-C
- Python
- Ruby
- Perl
- PHP

[7] RedMonk, "Who Are We?" Официальный сайт компании RedMonk URL:`http://redmonk.com/about/` Дата обращения: 19.12.12

[8] GitHub Inc "GitHub" Официальный сайт компании GitHub URL:`http://github.com/` Дата обращения: 3.01.2012

[9] stack exchange inc "StackOverflow" Официальный сайт сервиса StackOverflow URL:`http://stackoverflow.com/` Дата обращения: 3.01.2012

4 История языков программирования

4.1 Раняя история языков программирования.

Программирование появилось тогда, когда появились устройства, которые можно программировать. Первым таким устройством был не компьютер, а ткацкий станок. [13]
В 1801 году француз Жозеф Мари Жаккард (Joseph Marie Jacquard) изобрел первую программируемую машину, способную циклически выполнять заданные команды. Эта машина использовалась не для вычислений, а для создания узорчатых материй. Узор задавался при помощи набора перфокарт. Первым перфокарты для ткацких станков применил не Жаккард, а другой француз Жак де Вукансон (Jacques de Vaucanson) в 1745 году, но его машина требовала по-новой загружать карты после каждого прогона. Станок Жаккарда в свою очередь мог выполнять циклы из заранее установленных перфокарт, создавая повторяющийся узор. Его станок пользовался огромным успехом. В конечном итоге была создана колода из 24000 карт, описывающих различные узоры.
В 1822 году Чарльз Бэббидж (Charles Babbage) представил на обозрение Королевскому астрономическому обществу свою работу [14], в которой он описывал машину, способную быстро выполнять множество элементарных арифметических операций [15]. Он хотел автоматизировать работу "вычислителей— полуграмотных людей обученных только сложению и вычитанию, которые привлекались для расчетов логарифмических тригонометрических таблиц. Правительство Великобритании выделило Бэббиджу на создание Разностной машины в общей сложности 17000 фунтов стерлингов и затем прекратило финансирование, поэтому его машина так и не была закончена.
С 1847 по 1849 Бэббидж занимался разработкой Разностной машины №2 или аналитической машины, более общего инструмента, прообраза современного компьютера. Эта аналитическая машина имела арифметическое устройство, регистры памяти, устройство ввода и "принтер"для вывода на перфокарты. Бэббидж предусмотрел подсоединение к своей машине нескольких считывателей перфокарт. Причем некоторые из них могли контролировать работу других. Можно было как пропускать несколько карт, так и считывать карты в обратном порядке. Это позволяло пропускать или повторять участки программы, что уже соответсвовало трем основным типам выражений: последовательный, итеративыный и условный.
В 1842 году Августа Ада Кинг (Augusta Ada King), графина Лавлейс, дочь знаменитого английского поэта Джорджа Байрона, с детства увлекавшаяся математикой и науками и находившаяся в переписке с Чарльзом Бэббиджем, по порочению последнего занялась переводом работы итальянского инженера Луджи Менабреа (Luigi Menabrea). Эта работа представляла из себя конспект лекций самого Бэббиджа, с которыми он выступал в университете Турина. Ада перевела эти записи с французского и дополнила их комментариями, частично своими, частично самого Бэббиджа. Текст перевода был опубликован в 1953 году и сейчас с ним легко можно ознакомиться [10].
Содержание этих комментариев поражает, тогда в середине XIX века она смогла предвидеть появление цифровой музыки! Но самое главное, она включила в текст комментариев программу для вычисления чисел Бернулли (ур. 1) и ввела определения цикла. Это позволяет считать её первым программистом в истории.

$$B_n = \frac{-1}{n+1} \sum_{k=1}^{n} C_{n+1}^{k+1} B_{n-k}, \quad n \in \mathbb{N} \tag{1}$$

В 1991 год к двухсотлетию со дня рождения Чарльза Бэббиджа на основе его оригинальных работ в лондонском Музее Науки была собрана работающая копия аналитической машины. В 2000 году в том же музее заработал принтер для вывода информации на перфокарты. После устранения обнаруженных в старых чертежах небольших конструктивных неточностей,

[10] Charles Babbage "Sketch of The Analytical Engine" архив Джона Уолкера URL:`http://www.fourmilab.ch/babbage/sketch.html` Дата обращения: 3.01.2012

обе конструкции заработали безупречно [11] На этих машинах смогли воспроизвести программы предложенные графиней Лавлейс и все они оказались рабочими.
В плоть до сороковых годов двадцатого века больше не делалось попыток создать столь сложные устройства. Появлялись только различные калькуляторы, но ни о какой универсальности и речи не шло.
В 1936 году английский математик Алан Тьюринг формализовал понятие алгоритма и вычислимости на основе некой абстрактной машины, названной в последствии машиной тьюринга.
Это было огромной вехой в истории развития информатики и теории программирования. А полнотой по Тьюрингу или тьюринг-полнотой, стали называть свойство языка программирования, означающее его способность решать любые задачи. [16]
В 1942 году в Америке был создан ENIAC (Electronic Numerical Integrator And Computer — Электронный числовой интегратор и вычислитель) [17]. На смену механике пришли электронные сигналы. Правда процесс программирования такого компьютера заключался в его переконфигурировании. В течение нескольких дней специально обученные люди переподключали провода, и в результате из него получался компьютер специализированный для решения нужной задачи.
Хотя еще в годы Второй Мировой в Германии был разработан электромеханический Z4, программы которого хранились на перфокартах. Его автор, Конранд Цузе, придумал для него специальный язык программирования высокого уровня Планкалкюль. Но опубликованы работы с описаниями этого языка были довольно поздно, лишь в 1972 году, и уже не вляили на развитие языков программирования.
Американцы же учли свои ошибки. Уже работая над созданием ENIAC, они продумывали основы новой архитектуры, которая была впервые реализована в компьютере EDVAC. [?] Сейчас эта архитектура известна как архитектура фон Неймана. В её основе лежат принципы, которые и сейчас используются при создании ЭВМ.
С появлением EDVAC программирование стало заключаться в записи последовательностей процессорных команд: сложить два регистра, считать значение из памяти и так далее. Это называется низкоуровневым программированием. Сегодня оно мало отличается от того как оно выглядело в прошлом веке. Только люди давно перестали писать шестнадцатиричные коды команд, а используют ассемблер, программу, переводящую в коды команд мнемокоды, читаемые человеком.
В 1949 году Джоном В. Мочли создал первый высокоуровневый язык программирования для электронных вычислительных машин — Шорткод (Short Code - короткий код). Программа на этом языке уже не была набором машинных инструкций, а математических выражений. Программисту нужно было в ручную переписывать арифметические выражения в набор специальных символов, а затем в двоичный код, воспринимаемый интерпритатором [18], который был реализован В. Ф. Шмитом и запущен на UNIVAC I Serial 1 в 1950.

4.2 История С.

В 1969 году Язык С на заре Большой отпечаток в истории языков программирования наложил язык Си, являющийся очень популярным в среде разработчиков систем программного обеспечения (включая операционные системы). Си сочетает в себе черты как языка высокого уровня, так и машинно-ориентированного языка, допуская программиста ко всем машинным ресурсам, чего не обеспечивают такие языки, как Бейсик и Паскаль.

Язык программирования Си был разработан в лабораториях Bell Labs в период с 1969 по 1973 годы. Согласно Ритчи, самый активный период творчества пришёлся на 1972 год. Язык назвали «Си» (С — третья буква латинского алфавита), потому что многие его особенности берут начало от старого языка «Би» (В — вторая буква латинского алфавита). Существует

[11] Christine McGourty “Babbage printer finally runs” Официальный сайт BBC URL:`http://news.bbc.co.uk/2/hi/science/nature/710950.stm` Дата обращения: 3.01.2012

несколько различных версий происхождения названия языка Би. Кен Томпсон указывает на язык программирования BCPL, однако существует ещё и язык Bon, также созданный им, и названный так в честь его жены Бонни. Существует несколько легенд, касающихся причин разработки Си и его отношения к операционной системе UNIX, включая следующие:

- Разработка Си стала результатом того, что его будущие авторы любили компьютерную игру, подобную популярной игре Asteroids(Астероиды). Они уже давно играли в неё на главном сервере компании, который был недостаточно мощным и должен был обслуживать около ста пользователей. Томпсон и Ритчи посчитали, что им не хватает контроля над космическим кораблём для того, чтобы избегать столкновений с некоторыми камнями. Поэтому они решили перенести игру на свободный PDP-7, стоящий в офисе. Однако этот компьютер не имел операционной системы, что заставило их её написать. В конце концов, они решили перенести эту операционную систему ещё и на офисный PDP-11, что было очень тяжело, потому что её код был целиком написан на ассемблере. Было вынесено предложение использовать какой-нибудь высокоуровневый портируемый язык, чтобы можно было легко переносить ОС с одного компьютера на другой. Язык Би, который они хотели сначала задействовать для этого, оказался лишён функциональности, способной использовать новые возможности PDP-11. Поэтому они и остановились на разработке языка Си.

- Самый первый компьютер, для которого была первоначально написана UNIX, предназначался для создания системы автоматического заполнения документов. Первая версия UNIX была написана на ассемблере. Позднее для того, чтобы переписать эту операционную систему, был разработан язык Си.

К 1973 году язык Си стал достаточно силён, и большая часть ядра UNIX, первоначально написанная на ассемблере PDP-11/20, была переписана на Си. Это было одно из самых первых ядер операционных систем, написанное на языке, отличном от ассемблера.

В 1978 году Ритчи и Керниган опубликовали первую редакцию книги «Язык программирования Си» [5]. Эта книга, известная среди программистов как «K&R», служила многие годы неформальной спецификацией языка. Версию языка Си, описанную в ней, часто называют «K&R C». Вторая редакция этой книги посвящена более позднему стандарту ANSI C, описанному ниже.

K&R ввёл следующие особенности языка:

- структуры (тип данных struct);
- длинное целое (тип данных long int);
- целое без знака (тип данных unsigned int);
- оператор += и подобные ему (старые операторы =+ вводили анализатор лексики компилятора Си в заблуждение, например, при сравнении выражений i =+ 10 и i = +10).

K&R C часто считают самой главной частью языка, которую должен поддерживать компилятор Си. Многие годы даже после выхода ANSI C, он считался минимальным уровнем, которого следовало придерживаться программистам, желающим добиться от своих программ максимальной портативности, потому что не все компиляторы тогда поддерживали ANSI C, а хороший код на K&R C был верен и для ANSI C.

После публикации K&R C в язык было добавлено несколько возможностей:

- функции, не возвращающие значение (с типом void) и указатели, не имеющие типа (с типом void *);
- функции, возвращающие объединения и структуры;

- имена полей данных структур в разных пространствах имён для каждой структуры;
- присваивания структур;
- спецификатор констант (const);
- стандартная библиотека, реализующая большую часть функций, введённых различными производителями;
- перечислимый тип (enum);
- дробное число одинарной точности (float).

В конце 1970-х годов Си начал вытеснять Бейсик с позиции ведущего языка для программирования микрокомпьютеров. В 1980-х годах он был адаптирован для использования в IBM PC, что привело к резкому росту его популярности. В то же время Бьёрн Страуструп и другие в лабораториях Bell Labs начали работу по добавлению в Си возможностей объектно-ориентированного программирования. Язык, который они в итоге сделали, C++, оказал большое влияние на разработку ПО, но так и не смог сравняться по популярности с Си, особенно в UNIX-подобных системах.

В 1983 году Американский национальный институт стандартов (ANSI) сформировал комитет для разработки стандартной спецификации Си. По окончании этого долгого и сложного процесса в 1989 году он был наконец утверждён как «Язык программирования Си» ANSI X3.159-1989. Эту версию языка принято называть ANSI C или C89. В 1990 году стандарт ANSI C был принят с небольшими изменениямиМеждународной организацией по стандартизации (ISO) как ISO/IEC 9899:1990.

Одной из целей этого стандарта была разработка надмножества K&R C, включающего многие особенности языка, созданные позднее. Однако комитет по стандартизации также включил в него и несколько новых возможностей, таких как прототипы функций (заимствованные из C++) и более сложный препроцессор.

ANSI C сейчас поддерживают почти все существующие компиляторы. Почти весь код Си, написанный в последнее время, соответствует ANSI C. Любая программа, написанная только на стандартном Си, гарантированно будет правильно выполняться на любой платформе, имеющей соответствующую реализацию Си. Однако большинство программ написаны так, что они будут компилироваться и исполняться только на определённой платформе, потому, что:

- они используют нестандартные библиотеки, например, для графических дисплеев;
- они используют специфические платформо-зависимые средства;
- они рассчитаны на определённое значение размера некоторых типов данных или на определённый способ хранения этих данных в памяти для конкретной платформы.

После стандартизации в ANSI спецификация языка Си оставалась относительно неизменной в течение долгого времени, в то время как Си++ продолжал развиваться (в 1995 году в стандарт Си была внесена Первая нормативная поправка, но её почти никто не признавал). Однако в конце 1990-х годов стандарт подвергся пересмотру, что привело к публикации ISO 9899:1999 в 1999 году. Этот стандарт обычно называют «C99». В марте 2000 года он был принят и адаптирован ANSI. Интерес к поддержке новых особенностей C99 в настоящее время смешан. В то время как GCC, компилятор Си от Sun Microsystems и некоторые другие компиляторы в настоящее время поддерживают большую часть новых особенностей C99, компиляторы компаний Borland и Microsoft не делают этого, причём, похоже, что две эти компании и не думают их добавлять.

8 декабря 2011 опубликован новый стандарт для языка Си (ISO/IEC 9899:2011). Некоторые возможности нового стандарта уже поддерживаются компиляторами GCC и Clang. Основные изменения:

Успех Си в основном связан с тем, что на нём была написана значительная часть операционной системы UNIX, которая в итоге приобрела очень большую популярность. Если считать по количеству используемых на данный момент операционных систем, разработанных на базе UNIX, то она является самой распространённой системой в мире. В связи с её распространённостью, а также с тем, что на данный момент объём операционной системы измеряется в миллионах строк кода (для примера, в последних версиях Linux содержится более 10 000 000 строк кода), задача о переписывании UNIX на другой язык становиться практически невыполнимой (также следует учитывать тот факт, что при ручном переписывании неизбежно возникнут ошибки, что существенно снизит стабильность работы, а при переводе с использованием программных средств пострадает производительность кода). Кроме того, язык Си, будучи приближённым к аппаратной реализации компьютера позволяет выжать из него намного больше, чем многие другие языки программирования. Это обстоятельство показывает бессмысленность перевода UNIX на другой язык. Таким образом, если другие языки программирования могут исчезнуть с течением времени, уступив дорогу новым технологиям, то язык Си будет жить, пока живёт UNIX. То есть пока существуют компьютеры в том виде, в котором мы их себе представляем.

4.3 C++

Язык возник в начале 1980-х годов, когда сотрудник фирмы Bell Labs Бьёрн Страуструп придумал ряд усовершенствований к языку C под собственные нужды. До начала официальной стандартизации язык развивался в основном силами Страуструпа в ответ на запросы программистского сообщества. В 1998 году был ратифицирован международный стандарт языка C++: ISO/IEC 14882:1998 «Standard for the C++ Programming Language»; после принятия технических исправлений к стандарту в 2003 году — нынешняя версия этого стандарта — ISO/IEC 14882:2003. Ранние версии языка, известные под именем «C с классами», начали появляться с 1980 года.Идея создания нового языка берёт начало от опыта программирования Страуструпа для диссертации. Он обнаружил, что язык моделирования Симула имеет такие возможности, которые были бы очень полезны для разработки большого программного обеспечения, но работает слишком медленно. В то же время язык BCPL достаточно быстр, но слишком близок к языкам низкого уровня и не подходит для разработки большого программного обеспечения. Страуструп начал работать в Bell Labs над задачами теории очередей (в приложении к моделированию телефонных вызовов). Попытки применения существующих в то время языков моделирования оказались неэффективными. Вспоминая опыт своей диссертации, Страуструп решил дополнить язык C возможностями, имеющимися в языке Симула. Язык C, будучи базовым языком системы UNIX, на которой работали компьютеры Bell, является быстрым, многофункциональным и переносимым. Страуструп добавил к нему возможность работы с классами и объектами. В результате, практические задачи моделирования оказались доступными для решения как с точки зрения времени разработки (благодаря использованию Симула-подобных классов) так и с точки зрения времени вычислений (благодаря быстродействию C). Вначале в C были добавлены классы (с инкапсуляцией), производные классы, строгая проверка типов, inline-функции и аргументы по умолчанию. Разрабатывая C с классами (позднее C++), Страуструп также написал программу cfront — транслятор, перерабатывающий исходный код C с классами в исходный код простого C. Новый язык, неожиданно для автора, приобрёл большую популярность среди коллег и вскоре Страуструп уже не мог лично поддерживать его, отвечая на тысячи вопросов.

В 1983 году произошло переименование языка из "C с классами"в "C++". Кроме того, в него были добавлены новые возможности, такие как виртуальные функции, перегрузка функций и операторов, ссылки, константы, пользовательский контроль над управлением сво-

бодной памятью, улучшенная проверка типов и новый стиль комментариев (//). Его первый коммерческий выпуск состоялся в октябре 1985 года.

В 1985 году вышло первое издание «Языка программирования C++», обеспечивающее первое описание этого языка, что было чрезвычайно важно из-за отсутствия официального стандарта. В 1989 году состоялся выход C++ версии 2.0. Его новые возможности включали множественное наследование, абстрактные классы, статические функции-члены, функции-константы и защищённые члены. В 1990 году вышло «Комментированное справочное руководство по C++», положенное впоследствии в основу стандарта. Последние обновления включали шаблоны, исключения, пространства имён, новые способы приведения типов и булевский тип.

Стандартная библиотека C++ также развивалась вместе с ним. Первым добавлением к стандартной библиотеке C++ стали потоки ввода/вывода, обеспечивающие средства для замены традиционных функций C printf и scanf. Позднее самым значительным развитием стандартной библиотеки стало включение в неё Стандартной библиотеки шаблонов.

В 1998 году был опубликован стандарт языка ISO/IEC 14882:1998 (известный как C++98), разработанный комитетом по стандартизации C++ (ISO/IEC JTC1/SC22/WG21 working group). Стандарт C++ не описывает способы именования объектов, некоторые детали обработки исключений и другие возможности, связанные с деталями реализации, что делает несовместимым объектный код, созданный различными компиляторами. Однако для этого третьими лицами создано множество стандартов для конкретных архитектур иоперационных систем.

В 2003 году был опубликован стандарт языка ISO/IEC 14882:2003, где были исправлены выявленные ошибки и недочёты предыдущей версии стандарта.

В 2005 году был выпущен отчёт Library Technical Report 1 (кратко называемый TR1). Не являясь официально частью стандарта, отчёт описывает расширения стандартной библиотеки, которые, как ожидалось авторами, должны быть включены в следующую версию языка C++. Степень поддержки TR1 улучшается почти во всех поддерживаемых компиляторах языка C++.

С 2009 года велась работа по обновлению предыдущего стандарта, предварительной версией нового стандарта сперва был C++99, а спустя год C++0x, сегодня — C++11, куда были включены дополнения в ядро языка и расширение стандартной библиотеки, в том числе большую часть TR1.

C++ продолжает развиваться, чтобы отвечать современным требованиям. Одна из групп, разрабатывающих язык C++ и направляющих комитету по стандартизации C++ предложения по его улучшению — это Boost, которая занимается, в том числе, совершенствованием возможностей языка путём добавления в него особенностей метапрограммирования.

Никто не обладает правами на язык C++, он является свободным. Однако сам документ стандарта языка (за исключением черновиков) не доступен бесплатно.

4.4 Objective C

В начале 1980-х годов было популярно структурное программирование. Оно позволяло «разбивать» алгоритм на малые части, в основном чтобы выделить этапы алгоритма в отдельные блоки и работать с ними. Однако, с ростом сложности задач, структурное программирование приводило к снижению качества кода. Приходилось писать всё больше функций, которые очень редко могли использоваться в других программах. Многие увидели в объектно-ориентированном программировании потенциальное решение возникшей проблемы. С одной стороны, Smalltalk использовали почти все более-менее сложные системы. С другой — использование виртуальных машин сильно тормозило работу системы и требовало огромных ресурсов.

ObjC был создан Брэдом Коксом в начале 1980-х в его компании Stepstone. Он пытался решить проблему повторного использования кода.

Целью Кокса было создание языка, поддерживающего концепцию software IC. Под этой концепцией понимается возможность собирать программы из готовых компонентов (объектов), подобно тому как сложные электронные устройства могут быть легко собраны из набора готовых интегральных микросхем.

При этом такой язык должен быть достаточно простым и основанным на языке C, чтобы облегчить переход разработчиков на него.

Одной из целей было также создание модели, в которой сами классы также являются полноценными объектами, поддерживалась бы интроспекция и динамическая обработка сообщений.

Получившийся в результате язык Objective-C оказался крайне прост — его освоение у C-программиста займет всего несколько дней. Он является именно расширением языка C — в язык C просто добавлены новые возможности для объектно-ориентированного программирования. При этом любая программа на C является программой и на Objective-C.

Одной из отличительных черт Objective-C является его динамичность — целый ряд решений, обычно принимаемых на этапе компиляции, здесь откладывается непосредственно до этапа выполнения.

Ещё одной из особенностей языка является то, что он message-oriented в то время как C++ — function-oriented. Это значит, что в нём вызовы метода интерпретируются не как вызов функции (хотя к этому обычно все сводится), а именно как посылка сообщения (с именем и аргументами) объекту, подобно тому, как это происходит в Smalltalk-е.

Такой подход дает целый ряд плюсов — так, любому объекту можно послать любое сообщение. Объект может вместо обработки сообщения просто переслать его другому объекту для обработки (так называемое делегирование), в частности именно так можно легко реализовать распределенные объекты (то есть объекты, находящиеся в различных адресных пространствах и даже на разных компьютерах).

Привязка сообщения к соответствующей функции происходит непосредственно на этапе выполнения.

Язык Objective-C поддерживает нормальную работу с метаинформацией — так у объекта непосредственно на этапе выполнения можно спросить его класс, список методов (с типами передаваемых аргументов) и instance-переменных, проверить, является ли класс потомком заданного и поддерживает ли он заданный протокол и т. п.

В языке есть нормальная поддержка протоколов (то есть понятие интерфейса объекта и протокола четко разделены). Для объектов поддерживается наследование (не множественное), для протоколов поддерживается множественное наследование. Объект может быть унаследован от другого объекта и сразу нескольких протоколов (хотя это скорее не наследование протокола, а его поддержка).

На данный момент язык Objective-C поддерживается компиляторами Clang и GCC (под управлением Windows используется в составе MinGW или cygwin).

Довольно много в языке перенесено на runtime-библиотеку и сильно зависит от неё. Вместе с компилятором gcc поставляется минимальный вариант такой библиотеки. Также можно свободно скачать runtime-библиотеку компании Apple: Apple's Objective-C runtime.

Objective-C становится все более востребованным у разработчиков ПО. Всего за год Objective-C удалось не только продемонстрировать рекордный рост рыночной доли, но и войти в десятку самых распространенных языков программирования, что обусловлено повышенным потребительским спросом на продукты Apple и увеличением количества специалистов, создающих приложения для iPhone- и iPad-устройств.

4.5 Java

Один из самых популярных, используемых и создавших вокруг себя огромное сообщество программистов. Java, уже много лет подряд, не опускается ниже третей строчки в рейтингах самых популярных языков программирования. Сейчас технологии Java предоставляют

средства для разработки приложений во всех сферах компьютерной деятельности, начиная с обработки огромного количества данных в облачных хранилищах, заканчивая создания любимых вам игр на вашем смартфоне и даже для написания программ управления различной бытовой техники. Первоначально же язык Java корпорации Sun предназначался только для устройств бытовой электроники, и в некоторых из них еще может использоваться его ранний вариант, известный под названием Oak. Однако настоящей стартовой площадкой для стремительного взлета Java стал Internet. Разработка Java началась в 1990 году, первая официальная версия — Java 1.0, — была выпущена только в 1996 году. Главным вдохновителем проекта был Джемс Гослинг [12] , считающийся одним из одним из наиболее известных людей в истории UNIX и в общем современного программирования. Изначально Java, как язык планировался расширением языка C++, но вскоре стала понятно, что для достижения цели нужно что новое. Поэтому Java имеет похожий синтаксис, но совершенно другую внутреннюю составляющую и концепцию.

Программы на Java транслируются в байт-код, выполняемый виртуальной машиной Java(JVM) — программой, обрабатывающей байтовый код и передающей инструкции оборудованию как интерпретатор. Достоинство подобного способа выполнения программ — в полной независимости байт-кода от операционной системы и оборудования, что позволяет выполнять Java-приложения на любом устройстве, для которого существует соответствующая виртуальная машина. Другой важной особенностью технологии Java является гибкая система безопасности благодаря тому, что исполнение программы полностью контролируется виртуальной машиной. Любые операции, которые превышают установленные полномочия программы (например, попытка несанкционированного доступа к данным или соединения с другим компьютером) вызывают немедленное прерывание. Часто к недостаткам концепции виртуальной машины относят то, что исполнение байт-кода виртуальной машиной может снижать производительность программ и алгоритмов, реализованных на языке Java. В последнее время был внесен ряд усовершенствований, которые несколько увеличили скорость выполнения программ на Java.

Данные особенности подтолкнули разработчиков на использования Java в своих проектах, поэтому язык стал набирать популярность. На сегодняшний день можно выделить основные плюсы языка, за что его предпочитают использовать:

- Долгий цикл поддержки: все новые версии обратно совместимы вплоть до Java 1.0.2
- Полная кроссплатформенность: работает на Windows, Linux, BSD, Solaris, AIX, Irix, HP-UX, QNX, OS/2, а также существуют реализации для микроконтроллеров, например NanoVM — подо всё есть совместимая реализация JDK, причем правильно написанная программа будет под всем этим работать одинаково.
- Наличие макро-языков: NetRexx, Clojure, Groovy, Scala (функциональный язык широкого назначения), и таких библиотек, как Quercus, позволяющих повторно использовать код, написанный ранее на скриптовых языках.
- Безопасность: такие вещи, как SQL Injection, неплановые сбои при неправильной работе с памятью и ряд других мелочей — в Java отсутствуют.

Однако все эти бонусы не достались просто так – все они достигнуты с потерей производительности. Java начиная с версии 1.2 и далее разрабатывалась для написания и поддержки сложных сетевых приложений, давая выпускать готовый продукт достаточно быстро, но при этом тратя меньше времени на отладку и тестирование. Достигнуто это за счёт таких особенностей, как:

- Обязательное использование концепции ООП. ООП приводит к большей модульности кода с меньшим числом концептуально лишних зависимостей.

[12]И. Черных “История Java” Веб-сайт "История компьютера"URL:`http://chernykh.net/content/view/885/966/` Дата обращения: 3.01.2012

- Программы пишутся не под настоящий процессор и исполняются не на физической машине, а в виртуальной, которая оптимизирована на поддержку именно таких программ, что устраняет проблемы связанные с разработкой под разные платформы.

- Отсутствие указателей и адресной арифметики, а так же наличие сборщика мусора исключает как класс ошибки связанные с контролированием цикла жизни объектов и контролем за освобождением памяти.

Но в Java помимо плюсов существует так же и минусы. Основными претензиями в сторону языка есть следующие:

- Не слишком хорошо спланированные стандартные библиотеки, тянущиеся ещё с 1-й версии. На самом деле, это проблема всего ООП, где имеется строгая концепция написания кода.

- Все равно существуют такие сущности как глобальные переменные и ссылки на объект, что противоречит основным канонам ООП и приводит к затруднительной отладке.

- Отсутствие хорошей реализации метапрограммирования и функций высшего порядка. В версии 1.5 были добавлены джене?рики и аннотации, но они не решают всех проблем и являются просто синтаксическим сахаром, при этом с неочевидной алгоритмом работы.

- Необходимость тонкой оптимизации, — как кода, так и настроек JVM, — для получения прироста скорости работы. Но это достаточно нетривиальная область и только программисты с опытом способны на это.

Но, не смотря на свои минусы, Java начав с языка программирования для бытовой техники, смогла стать основным языком разработки во многих отраслях. Основными факторами позволившими сделать это стали:

- Строгое ООП

- Надежность

- Кроссплатформенность

4.6 Perl

В 1988 году Ларри Уолл (Larry Wall), лингвист по образованию, опубликовал в новостной конференции comp.sources.misc новость под заголовком ”Perl, a "replacement"for awk and sed“ (Перл, "заменитель"awk и sed) [13] . Сам язык появился был разработон Уоллом еще в 1986, когда он являлся системным администратором одного проекта на основе операционной системы UNIX, связанного с созданием многоуровневой безопасной сети, которая объединяла несколько компьютеров, разнесенных на большие расстояния. Работа была выполнена, но потребовалось создание отчетов на основе большого числа файлов с многочисленными перекрестными ссылками между ними. В то время существовало два основных инструмента для анализа и обработки текста, которые используются и до сих пор. Это программы sed и awk, "заменителем"которых и должен был явится язык Перл.
Sed (от англ. "строковый редактор") — потоковый текстовый редактор созданный Ли Макхэмоном (Lee E. McMahon) в 1973-1974 годах [14] . Он предназначен для редактирования текстов через применение к каждой строке последовательности операций. Операции можно задавать

[13]Ларри Уолл “Perl, a "replacement"for awk and sed” Архив новостных конференций Google `http://groups.google.com/group/comp.sources.unix/tree/browse_frm/month/1988-02?_done=/group/comp.sources.unix/browse_frm/month/1988-02?&` Дата обращения: 3.01.2012

[14]Сообщество sed “Frequently-Asked Questions about sed, the stream editor” Официальный сайт проекта sed URL:`http://sed.sourceforge.net/sedfaq2.html#s2.1` Дата обращения: 3.01.2012

в виде текстового скрипта, поэтому sed тоже можно считать языком программирования.
AWK — интерпретируемый язык для анализа текстовых данных и их обработки. AWK был создан в Bell Labs [15] тоже в начале 1970х и назван так по первым буквам фамилий своих авторов: Альфреда Ахо (Alfred Aho), Петера Вайнбергера (Peter Weinberger) и Брайана Кернигана (Brian Kernighan). Этот язык предназначался для построчного разбора и обработки входного потока (например, текстового файла) по заданным шаблонам. Он стал де-факто стандартом в решении подобных задач и до сих пор его новые версии NAWK и GAWK используются во всех UNIX-подобных системах [16] .
Ни один из этих инструментов не подходил Ларри, так, например, AWK не мог управлять открытием и закрытием большого числа файлов на основе содержащейся в них же самих информации об их расположении. Его первой мыслью было написать специальную системную утилиту, решающую поставленную задачу, но, вспомнив, что до этого ему уже пришлось написать несколько утилит для решения задач, не «берущихся» стандартными средствами UNIX, он принял кардинальное решение — разработать язык программирования, который сочетал бы в себе возможности обработки текстовых файлов (sed), генерации отчетов (awk), решения системных задач (shell) и низкоуровневое программирование, доступное на языке C. Результатом этого решения и явился язык Perl, интерпретатор для которого был написан на C. По утверждению самого Ларри Уолла, при создании языка Perl им двигала лень — в том смысле, что для решения стоявшей перед ним задачи следовало бы написать большое количество программ на разных языках, входящих в состав инструментальных средств UNIX, а это достаточно утомительное занятие. Новый язык программирования сочетал в себе возможности системного администрирования и обработки файлов — две основные задачи, решаемые обычно при программировании в системе UNIX. Причем следует отметить, что язык Perl появился из практических соображений, а не из-за желания создать еще одно «красивое» средство для работы в UNIX, поэтому-то он и получил широкое распространение среди системных администраторов, когда Лрри Уолл предоставил его широкому кругу пользователей. После создания языка Perl появилась возможность решать задачи с помощью одного инструмента и не тратить время на изучение нескольких языков среды программирования UNIX. Первоначально язык был очень примитивен, вся документация умещалась на 15 страницах, но Perl решал задачи быстрее, чем sed или awk, и скоро стал использоваться не только для решения задач системного администрирования.
В дальнейшем сам Ларри Уолл позаимствовал у Генри Спенсера (Henry Spencer) («бога регулярных выражений», как о нем написано в одной из книг) пакет для работы с регулярными выражениями и модифицировал его для использования в языке Perl. Другие функциональные возможности были разработаны не только Ларри Уоллом, но и его друзьями и коллегами, и также включены в состав языка. Опубликование в Интернете привело к появлению сообщества единомышленников, которые не только использовали для решения своих задач язык, но и развивали его.
Язык активно развивался, но до 1991 года, когда в свет вышла книга Программирование на Perl [3], не существовало приличной документации. Эта книга стала фактически стандартом языка. С её выходом текущую версию языка тут же назвали 4.0, чтобы было проще ссылаться на нее в книге [17] .
Язык начал активно развиваться и набирать популярность с выходом пятой версии. В 1995 году появилась CPAN (Comprehensive Perl Archive Network, Общая архивная сеть Perl) [18]

[15] Naomi Hamilton "The A-Z of Programming Languages: AWK" Официальный сайт журнала computerworld URL:http://www.computerworld.com.au/article/216844/a-z_programming_languages_awk/?fp=4194304&fpid=1 Дата обращения: 3.01.2012

[16] The Open Group "Utilities Interface Table" Домашняя страница проекта UNIX URL:http://www.unix.org/version3/apis/cu.html Дата обращения: 3.01.2012

[17] John Allen "perlhist" Perl Programming Documentation URL:http://perldoc.perl.org/perlhist.html Дата обращения: 3.01.2012

[18] Perl.org "CPAN" Comprehensive Perl Archive Network URL:http://www.cpan.org/ Дата обращения: 3.01.2012

, что позволило удобно расширять стандартные возможности при помощи модулей написаных другими программистами и обмениваться ими. Сейчас на серверах CPAN хранится около 170000 модулей.
Большие возможности языка в обработке текстовых данных сделали его очень удобным для создания серверных веб-приложений. Существует множество крупных веб-порталов написанных на языке PERL. Например известный онлайн-магазин Amazon [19]

Чтобы не писать всю серверную часть веб-приложений каждый раз при создании нового сайта, программисты стали создавать веб-фрэймворки, своеобразные заготовки для интернет-приложений, в которых уже реализован набор самых сложных и нужных функций, такой как работа с базой данных, обработка запросов, генерация страниц. В наше время таких продуктов существует огромное количество, вот некоторые из тех, что написаны на Perl:

- Catalyst [20]
- Dancer [21]
- Gantry [22]
- Mason[23]
- Maypole [24]
- Mojolicious [25]
- WebGUI [26]

На каждом из них создано множество сайтов.

4.7 PHP

В 1994 году Расмус Лердорф (Rasmus Lerdorf) написал небольшой набор серверных скриптов на языке C для отслеживания посещений своего веб-резюме и назвал его "Personal Homepages Tools"("Инструменты для персональных домашних страниц"), но более часто упоминалось название "PHP Tools"[27]. Со временем требовалось все больше улучшений функциональности, и Расмус переписал PHP Tools, создав более крупную и богатую реализацию. Эта новая реализация была способна взаимодействать с базами данных и многое другое, что создавало фреймворк, с помощью которого пользователи могли создавать простые динамические веб-приложения, такие как гостевые книги.

[19]Amazon.com“Amazon.com Interview: Larry Wall” Официальный сайт компании Amazon URL:`http://www.amazon.com/exec/obidos/tg/feature/-/7137/` Дата обращения: 3.01.2012

[20]Catalyst Foundation “Catalyst” Сайт проекта Catalyst URL:`http://www.catalystframework.org/` Дата обращения: 3.01.2012

[21]Alexis Sukrieh “PerlDancer” Сайт проекта Dancer URL:`http://www.perldancer.org/` Дата обращения: 3.01.2012

[22]RocketTheme, LLC “Gantry” Сайт проекта Gantry URL:`http://www.gantry-framework.org/` Дата обращения: 3.01.2012

[23]Jonathan Swartz “Masonhq” Сайт проекта Mason URL:`http://www.masonhq.com/` Дата обращения: 3.01.2012

[24]Aaron James Trevena “Maypole” Каталог CPAN URL:`http://search.cpan.org/~teejay/Maypole-2.13/lib/Maypole.pm` Дата обращения: 3.01.2012

[25]Mojolicious “Mojolicious” Сайт проекта Mojolicious URL:`http://mojolicio.us/` Дата обращения: 3.01.2012

[26]Plain Black Corporation “WebGUI” Сайт проекта WebGUI URL:`http://www.webgui.org/` Дата обращения: 3.01.2012

[27]The PHP Group “История PHP” Официальный сайт PHP URL:`http://php.net/manual/ru/history.php.php` Дата обращения: 3.01.2012

В июне 1995 года Расмус открыл исходный код PHP Tools общественности, что позволило разработчикам использовать его по своему усмотрению. Это также дало возможность пользователям исправлять ошибки в коде и улучшать его.

В сентябре того же года, Расмус расширил PHP и на короткое время убрал из названия PHP, назвав его FI (сокращение от "Интерпретатор Форм"). Новая реализация включала в себя все основные функциональные возможности современного PHP. Она имела Perl-подобные переменные, автоматическую интерпретацию форм и встраиваемый в HTML синтаксис. Синтаксис языка был похож на Perl, хотя и был гораздо более ограниченным, простым, и в некоторой степени противоречивым. Для того, чтобы вставлять код в HTML-файл, разработчикам пришлось использовать HTML комментарии. Хотя этот метод был не совсем хорошо принят, FI по-прежнему набирал популярность в качестве инструмента для создания серверной части сайтов, но все-таки не в качестве языка. Однако, перемены начались в следующем месяце, когда в октябре 1995 года Расмус выпустил полностью переписанный код, с вернувшимся именем PHP, но уже сокращенным от "Personal Home Page Construction Kit"("Инструменты для создания персональных домашних страниц"). Язык намеренно напоминал Си по структуре, что делал его легким для восприятия разработчиками, знакомыми с Си, Perl и подобными языками. Будучи все еще ограниченными UNIX и POSIX-совместимыми системами, был изучен вопрос для реализации языка в Windows NT.

Код получил еще одно существенное преобразование в апреле 1996 года. Объединив названия предыдущих версиий, Расмус представил PHP/FI. Реализации второго поколения начали по-настоящему развивать PHP из набора инструментов в самостоятельный язык программирования. PHP включал в себя встроенную поддержку для DBM, mSQL и Postgres95 баз данных, cookies, поддержку определяемых пользователем функций и многое другое. В июне PHP/FI была присвоена версия 2.0.

Несмотря на короткую историю, PHP/FI продолжал набирать популярность в молодом мире веб-разработки. В 1997 и 1998, PHP/FI стал культом для нескольких тысяч пользователей по всему миру. Исследования Netcraft в мае 1998 года показали, что почти 60 тысяч доменов передавали заголовки, содержащие "PHP". Это число равнялось примерно 1Несмотря на эти впечатляющие цифры, развитие PHP/FI было ограничено: несмотря на нескольких второстепенных участников, в целом он по-прежнему разработывался одним человеком.

В 1997 году Энди Гутманс и Зив Сураски из Тель-Авива, посчитав PHP/FI 2.0 все еще неэффективным и недостаточно функциональным для использования в коммерческих приложениях, разрабатываемых для их университетского проекта, начали еще раз заново переписывать парсер. Связавшись с Расмусом, они обсудили различные аспекты текущей реализации и их новой разработки PHP. Для улучшения движка и использования уже существующей базы пользователей PHP/FI, Энди, Расмус и Зив решили сотрудничать в развитии нового, независимого языка программирования. Теперь аббревиатура "PHP"означала "PHP: Hypertext Preprocessor".

Одной из сильнейших сторон PHP 3.0 была возможность расширения ядра. Кроме обеспечения пользователей надежной инфраструктурой из множества различных баз данных, протоколов и API, расширяемость PHP 3.0 привлекла к нему множество сторонних разработчиков, желающих добавить к языку свои расширения. Возможно, это и был главный ключ к успеху, но стоит добавить, что немаловажным шагом оказалась поддержка объектно-ориентированного синтаксиса и намного более мощного и последовательного синтаксиса самого языка.

В 1998 году, со множеством новых разработчиков со всего мира присоединившихся к проекту, PHP 3.0 был представлен новой командой разработчиков, как официальный преемник PHP/FI. Примерно через девять месяцев открытого публичного тестирования, при объявлении официального выпуска PHP 3.0, он уже был установлен на более чем 70000 доменах по всему миру, и уже не ограничивается POSIX-совместимыми операционными системами. Относительно небольшая доля доменов с установленным PHP была размещена на серверах под управлением Windows 95, 98 и NT, а также Macintosh. На пике своего развития, PHP

3.0 был установлен приблизительно на 10

В 1999 году свет увидело новое ядро "Zend Engine"(от имен создателей: Zeev и Andi), работа над которым началась сразу после официального выхода PHP 3.0. В задачи входило увеличение производительности сложных приложений и улучшение модульности кодовой базы PHP. PHP 3.0 дал возможность подобным приложениям успешно работать с набором баз данных и поддерживать большое количество различных API и протоколов, но PHP 3.0 не имел качественной поддержки модулей и приложения работали неэффективно.

PHP 4.0, основанный на этом движке и принесший с собой набор дополнительных функций, официально вышел в мае 2000 года, почти через два года после выхода своего предшественника. Кроме значительного улучшения производительности, PHP 4.0 имел еще несколько ключевых нововведений, таких как поддержка намного большего числа веб-серверов, поддержка HTTP сессий, буферизация вывода, более безопасные способы обработки вводимой пользователем информации и несколько новых языковых конструкций.

В 2004 году после долгой разработки был выпущен PHP 5. В основном он управляется ядром Zend Engine 2.0 с новой объектной моделью и множеством различных других нововведений.

Команда разработчиков PHP включает в себя десятки разработчиков, а также десятки других организаций, работающих над связанными с PHP и его поддержкой проектами, такими как PEAR [28]. , PECL [29]. и документацией, а также базовую инфраструктуру сети более чем из ста серверов на шести из семи континентах мира. Основываясь на статистике прошлых лет, можно с уверенностью предположить, что PHP теперь установлен на десятки или даже, возможно, сотни миллионов доменов по всему миру.

В области программирования для сети Интернет PHP — один из популярных скриптовых языков (наряду с JSP, Perl и языками, используемыми в ASP.NET) благодаря своей простоте, скорости выполнения, богатой функциональности, кроссплатформенности и распространению исходных кодов на основе свободной лицензии PHP.

Популярность в области построения веб-сайтов определяется наличием большого набора встроенных средств для разработки веб-приложений [30]. Основные из них:

- автоматическое извлечение POST и GET-параметров, а также переменных окружения веб-сервера в предопределённые массивы;
- взаимодействие с большим количеством различных систем управления базами данных (MySQL, MySQLi, SQLite, PostgreSQL, Oracle (OCI8), Oracle, Microsoft SQL Server, Sybase, ODBC, mSQL, IBM DB2, Cloudscape и Apache Derby, Informix, Ovrimos SQL, Lotus Notes, DB++, DBM, dBase, DBX, FrontBase, FilePro, Ingres II, SESAM, Firebird / InterBase, Paradox File Access, MaxDB, Интерфейс PDO);
- автоматизированная отправка HTTP-заголовков;
- работа с HTTP-авторизацией;
- работа с cookies и сессиями;
- работа с локальными и удалёнными файлами, сокетами;
- обработка файлов, загружаемых на сервер;
- работа с XForms.

[28]The PHP Group “Pear” Официальный сайт Pear URL:`http://pear.php.net/` Дата обращения: 3.01.2012

[29]The PHP Group “PECL” Официальный сайт PECL URL:`http://pecl.php.net/` Дата обращения: 3.01.2012

[30]The PHP Group “Отличительные особенности” Официальный сайт PHP URL:`http://docs.php.net/manual/ru/features.php` Дата обращения: 3.01.2012

В настоящее время PHP используется сотнями тысяч разработчиков. К крупнейшим сайтам, использующим PHP, относятся Facebook и Wikipedia.

Некоторые [31]. видят причины популярности PHP на фоне Perl в том, что PHP смог "застолбить"себе расширение файлов ".php". Каждая отдаваемая страничка с таким расширением служит своего рода рекламой, в то время как остальные обработчики, например те, что написаны на Perl, отдавали простые ".html"страницы.

4.8 Python

В 1989 году голландец Гвидо ван Россум (Guido van Rossum), у которого просто выдалось свободное время на зимних каникулах, создает язык Python. Язык назван так в честь нглийского юмористического шоу "Летающий цирк Монти Пайтона"("Monty Python's Flying Circus") [32]. Гвидо работал в Амстердамском университете над созданием новой операционной системы и заметил, что язык C весьма сложен, и его использование сильно замедляет процесс разработки. Он решил создать максимально простой язык. Язык на котором само написание кода занимало как можно меньше времени.
В 1987 году в том же Амстердамском университете появился язык ABC [33]. Его авторами являются Лео Гертс (Leo Geurts), Ламберт Меертенс (Lambert Meertens) и Стивен Пембертон (Steven Pemberton). При создании его они поставили для себя целью создать максимально простой в использовании язык, предназначенный для обучения программированию и для использования непрограммистами. Они упрощали синтаксис как могли, программы на языке ABC были легко читаемыми и были в пять раз короче аналогичных программ на C.
Гвидо ван Россум тоже участвовал в проекте по разработке языка ABC. Ему нравилась сама идея простоты, но он отметил для себя несколько важных недостатков, самым главным из которых по его мнению была "закрытость"языка, т.е. в языке нельзя было использовать сторонние модули.
В 1991 году ван Россум сделал исходный код своего языка открытым широкой общественности. Доступный изначально под собственной свободной лицензией, с выходом версии 2.0 в 2001 году Python стал распространяться под лицензией GNU GPL [34]. .
Активно развивающееся дружелюбное сообщество, можно считать главной причиной успеха Python. Люди со всего мира создавали множество модулей языка свободно доступных другим пользователям. Благодаря этим модулям, язык стал по-настоящему универсальным, он нашел применение во множестве областей среди которых:

- экспертные системы;
- веб-приложения;
- математические расчеты;
- разработка встравоемого программного обеспечения;
- прототипирование.

[31]Nick "Причины популярности PHP на фоне Perl Web Frameworks" Laziness, Impatience and Hubris URL:`http://laziness-impatience-hubris.blogspot.ru/2009/07/php-perl-web-frameworks.html` Дата обращения: 3.01.2012

[32]Naomi Hamilton "The A-Z of Programming Languages: Python" Официальный сайт журнала computerworld URL:`http://www.computerworld.com.au/article/255835/a-z_programming_languages_python/` Дата обращения: 3.01.2012

[33]Издательство IdHub "ABC programmers's handbook" Официальный сайт издательства IdHUB URL:`http://www.idhub.com/abc/` Дата обращения: 3.01.2012

[34]Python Software Foundation "Python 2.0.1 - a bugfix release for Python 2.0" Официальный сайт Python URL:`http://www.python.org/download/releases/2.0.1/` Дата обращения: 3.01.2012

В 2000 году появились PEP, сокращения от "Python Enhancement Proposal"("предложения по расширению языка Python") [35]. Это набор документов для сообщества программистов и разработчиков языка, в соответсвии с которыми происходит его развитие. В PEP предствлены как пожелания пользователей (эта документация открыта для редактирования всеми желающими), так и опросы пользователей языка о каких-либо его свойствах, но окончательные решения принимают авторы PEP во главе с самим Гвидо ван Россумом. Так же PEP может рассматриваться как описание языка, потому что из-за быстрого развития не сложилось стандарта Python. Существует несколько реализаций интерпритатора языка, одна из которых, CPython, считается канонической.

В 2004 году появляется PEP-20 [36]., в котором формулируется философия Python, так называемый "Дзен Python". Эта философия сформулирована в нескольких принципах:

- Красивое лучше, чем уродливое.
- Явное лучше, чем неявное.
- Простое лучше, чем сложное.
- Сложное лучше, чем запутанное.
- Плоское лучше, чем вложенное.
- Разреженное лучше, чем плотное.
- Читаемость имеет значение.
- Особые случаи не настолько особые, чтобы нарушать правила.
- При этом практичность важнее безупречности.
- Ошибки никогда не должны замалчиваться.
- Если не замалчиваются явно.
- Встретив двусмысленность, отбрось искушение угадать.
- Должен существовать один — и, желательно, только один — очевидный способ сделать это.
- Хотя он поначалу может быть и не очевиден, если вы не голландец[37].
- Сейчас лучше, чем никогда.
- Хотя никогда зачастую лучше, чем прямо сейчас.
- Если реализацию сложно объяснить — идея плоха.
- Если реализацию легко объяснить — идея, возможно, хороша.
- Пространства имён — отличная штука! Будем делать их побольше!

[35]Barry Warsaw “PEP Purpose and Guidelines” Официальный сайт PEP URL:`http://www.python.org/dev/peps/pep-0001/` Дата обращения: 3.01.2012

[36]Tim Peters “The Zen of Python ” Официальный сайт PEP URL:`http://www.python.org/dev/peps/pep-0020/` Дата обращения: 3.01.2012

[37]Намек на национальность Гвидо ван Россума

В дальнейшей работе над языком его разработчики стараются их придерживаться, и простым пользователям-программистам, тоже следует проникнуться этой философией, чтобы лучше понимать Python как инструмент.

В 2008 году вышла версия языка 3.0, являющаяся по-сути новым языком обратно не совместимым с предыдущими версиями [38]. Т.е. код написанный на второй версии Python уже может не работать в интерпритаторе третьей версии. Отказ от обратной совместимости позволил разрботчикам устронить многие "детские болезни"языка, например из главной библиотеки были удалены многие устаревшие или плохо работающие модули. Обе версии, третья и вторая, продолжают развиваться паралельно, потому что существует много кода написанного для старой версии, и переписать его практически нереальная задача.

4.9 Ruby

В 1994 году японец Юхиро Мацумото создает первую альфа-версию языка Ruby (рубин). Свое название он получил в честь драгоценного камня, чтобы подчеркнуть некоторую аналогию с Perl (Pearl — жемчужина). Позднее выяснилось, что жемчуг является камнем–покровителем людей родившихся в июне, а рубин — в июле, что еще раз подчеркнуло преемственность нового языка.
Мацумото говорил: "Мне нужен был скриптовый язык более мощный чем Perl и более объектно-ориентированный чем Python. Поэтому я решил сделать своя язык."[39] Основной упор при создании языка Мацумото, как фанат объектно-ориентированного программирования, сделал на реализацию поддержки этой парадигмы в своем языке. Кроме того он отталкивался от сформулированной им философии языка, которая звучит по-японски кратко: "Принцип наименьшего удивления". Мацумото хотел, чтобы программист чувствовал себя комфортно, чтобы язык делал именно то, что ожидает от него пользователь.

К 1996 году сформировалось сообщество пользователей, до этого момента Мацумото, выложивший все свои наработки в открытый доступ, работал один. Сообщество было небольшим, чисто японским. Только в 1997 году Мацумото перевел документацию на английский язык, и началось более активное продвижение языка за пределами Японии.

В 1998 году вышла первая стабильная версия языка, сразу за этим появилось первое англоязычное сообщество программистов на Ruby. А вышедшее в 2000 году первое издание книги "Programming Ruby"("Программирование на Ruby") [4] еще сильнее повлияло на распространение языка.

В 2005 году Давид Хейнемейер Ханссон (David Heinemeier Hansson) выпустил первую версию веб-фрэймворка Ruby-on-Rails. Это приложение стало ключевым [40] для расширения популярности языка Ruby, многие программисты начинали изучать язык, только для работы с этим фрэймворком, по всему миру[41] стали проводится постоянные конференции посвященные разработке при помощи Ruby-on-Rails.

В 2009 году у Ruby появился свой менеджер и хранилище библиотек RubyGems во многом аналогичный CPAN для Perl. Как и другие аналогичные события в истории других языков, это позволило развивать Ruby еще активнее.

Язык продолжает развиваться и сейчас, ведется разработка новой версии языка mruby [42] . Новая версия должна стать полностью обновленной, будет реализована новая виртуальная

[38]Python Software Foundation “Python 3.0” Официальный сайт Python URL:`http://www.python.org/download/releases/2.0.1/` Дата обращения: 3.01.2012

[39]Bruce Stewart “An Interview with the Creator of Ruby” Официальный сайт издательства O‘Reily URL:`http://linuxdevcenter.com/pub/a/linux/2001/11/29/ruby.html` Дата обращения: 3.01.2012

[40]Developer Shed, Inc “Ruby-on-Rails” Интернет сайт DevArticles URL:`http://www.devarticles.com/c/a/Ruby-on-Rails/Web-Development-Ruby-on-Rails/` Дата обращения: 3.01.2012

[41]Keavy “Events of interest to rubyists around the world” интернет сайт rubythere URL:`http://rubythere.com/` Дата обращения: 3.01.2012

[42]Matt Aimonetti “Mruby and MobiRuby” Персональный сайт Matt Aimonetti URL:`http://matt.aimonetti.net/posts/2012/04/20/mruby-and-mobiruby/` Дата обращения: 3.01.2012

машина, внесены несущественные изменения в синтаксис. По сути это будет новый язык, ориентированный на мобильные устройства.

5 Заключение.

Мы рассмотрели восемь языков, которые так или иначе превосходят своих конкурентов. И можно сказать что почти у каждого из них своя причина для популярности. Но основной конечно является популярность целевой платформы. Это особенно верно в отношении языков Perl, PHP, Ruby и Objective-C. Первые три из них используются для создания веб-приложений, а интернетом пользуется сейчас огромное количество людей. Так же самый быстрый рост популярности по данным рейтинга TIOBE обеспечивает себе Objective-C, за счет мирового повального увлечения продуктами компании Apple, и только за счет этого, потому что больше он практически нигде не используется.Python берет своей простотой и универсальнсотью, огромным количеством модулей. C, C++ — это "старички"использующиейся везде только потому, что они уже давно используются везде. Кроме этого стоит отметить, что это единственные два языка из всех рассмотренных, которые компилируются в машинный код, а не выполняются виртуальной машиной, программы написанные на этих языках способны работать быстрее чем их аналоги, например на Java. По ним существует много учебников, они хорошо описаны. Наличие документации, или по крайней мере большого числа доступных источников информации, сформировавшееся сообщество пользователей — это особенности всех предствленных здесь языков. С другой стороны именно популярность языка обеспечивает рост сообщества, а чем больше сообщество, тем популярнее становится язык и тем активнее он развивается.

Можно сделать вывод, что популярность конретного языка зависит от популярности платформы и задачи, для которой создан этот язык. Чем популярнее язык, тем больше программистов он привлекает, тем больше его сообщество, тем быстрее он развивается, благодря чему он становится еще популярней. Не стоит забывать и об удобствах, которые предоставляет язык программисту. Чем удобнее инструмент, тем проще и приятнее с ним работать.

Содержание

Список использованных источников

1. F. Biancuzzi. *Masterminds of Programming: Conversations with the Creators of Major Programming Languages*. Theory in Practice (o'Reilly) Series. O'Reilly Media, 2009.

2. Bjarne Stroustrup. A history of c++: 1979-1991. In *HOPL Preprints*, pages 271–297, 1993.

3. Л. Уолл, Т. Кристиансен, and Д. Орвант. *Программирование на Perl: [пер. с англ.]*. Символ-Плюс, 2008.

4. D. Flanagan and Y. Matsumoto. *The Ruby Programming Language*. Oreilly Series. O'Reilly Media, Incorporated, 2008.

5. Brian W. Kernighan. *The C Programming Language*. Prentice Hall Professional Technical Reference, 2nd edition, 1988.

6. Naomi Hamilton. The a-z of programming languages: Awk. Website, 2008. `http://www.computerworld.com.au/article/216844/a-z_programming_languages_awk/`.

7. Naomi Hamilton. The a-z of programming languages: Python. Website, 2008. `http://www.computerworld.com.au/article/255835/a-z_programming_languages_python/`.

8. Nick. Причины популярности php на фоне perl web frameworks. Website, 2009. `http://laziness-impatience-hubris.blogspot.ru/2009/07/php-perl-web-frameworks.html/`.

9. Black Duck. Compare languages. Website, 2012. `http://www.ohloh.net/languages/compare`.

10. TIOBE Software. Tiobe programming community index for january 2013. Website, 2012. `http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html`.

11. Stephen O'Grady. The redmonk programming language rankings: February 2012. Website, 2012. `http://redmonk.com/sogrady/2012/02/08/language-rankings-2-2012/`.

12. Drew Conway. Ranking the popularity of programming languages. Website, 2010. `http://www.dataists.com/2010/12/ranking-the-popularity-of-programming-langauges/`.

13. E.G. Swedin and D.L. Ferro. *Computers: The Life Story Of A Technology*. Greenwood Technographies. Greenwood Press, 2005.

14. Charles Babbage and Henry P. Babbage. Cambridge University Press, 2010.

15. John Fuegi and Jo Francis. Lovelace & babbage and the creation of the 1843 'notes'. *IEEE Ann. Hist. Comput.*, 25(4):16–26, October 2003.

16. Alan M. Turing. On computable numbers, with an application to the entscheidungsproblem. *Proceedings of the London Mathematical Society*, 42:230–265, 1936.

17. Юрий Полунов. Автора ! ! ! Website, 2006. `http://www.pcweek.ru/themes/detail.php?ID=72606&phrase_id=204737`.

18. William F. Schmitt. The univac short code. *Annals of the History of Computing*, 10(1):7 –18, jan.-march 1988.