Реферат

Дипломная работа содержит 67 страниц, 27 рисунков. Список использованной литературы включает в себя 7 источников.

ЧАТ, СИСТЕМА ОБМЕНА СООБЩЕНИЯМИ, КОЛЛЛЕКТИВНАЯ РАЗРАБОТКА, ПРОЕКТНАЯ ГРУППА, РЕПОЗИТОРИЙ, КЛИЕНТ-СЕРВЕР, МНОГОПОТОЧНОСТЬ, JAVA, SUBVERSION, THREAD.

Дипломная работа посвящена разработке многопользовательского многопоточного приложения на языке java для диалогового онлайн общения пользователей в сети.

Произведён краткий обзор популярных приложений для общения пользователей в сети из широко представленных на рынке программ. Представлено обоснование выбора платформы Javaдля проектирования.

Освещены вопросы коллективной разработки. Описан процесс разработки с использованием проектного хостинга и сервиса для разработчиков приложений с использованием системы контроля версий.

Программа чат представляет собой консольное приложение мгновенного обмена сообщениями, в основе программы лежит двухуровневая архитектура «Клиент-Сервер». Сервер обрабатывает запросы клиентов, журналирует и коммутирует их переписку. Используется циркуляр сообщений, который позволяет передавать сообщения всем пользователям, либо же вести приватную переписку между клиентами без отображения информации в консоли других пользователей чата. В чате представлена возможность выбора каналов, которые предустановлены на сервере и последующее общение пользователей в выбранном канале по интересам.

Разработанное приложение должно выполняться в средахWindows/Unix с использованием Java7.

Содержание

[Введение](#_Toc346006518) 4

[1. Исследовательская часть](#_Toc346006516) 7

1.[1 Обзор систем мнгновенного обмена сообщениями](#_Toc346006519) 7

1.[2Выбор технологии Java для разработки 2](#_Toc346006520)0

1.[3Коллективная разработка и управление проектом 3](#_Toc346006519)0

[2. Вычислительная часть](#_Toc346006516) 50

[2.1Основные сведения о программе чат и концепции программы](#_Toc346006521) 50

[2.2Архитектура приложения](#_Toc346006521) 52

[2.2.1Работа серверной части чата](#_Toc346006521) 52

[2.2.2Работа клиентской части чата](#_Toc346006521) 60

[3.Выводы](#_Toc346006567) 63

[Заключение **Ошибка! Закладка не определена.**](#_Toc346006568)65

[Список использованных источников **Ошибка! Закладка не определена.**](#_Toc346006569)66

## Введение

Чат, чаттер (chatter — болтать) — средство обмена сообщениями по компьютерной сети в режиме реального времени, а также [программное обеспечение](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B5_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D0%B5), позволяющее организовывать такое общение. Характерной особенностью является коммуникация именно в реальном времени или близкая к этому, что отличает чат от [форумов](https://ru.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%84%D0%BE%D1%80%D1%83%D0%BC) и других «медленных» средств.То есть, если на форуме можно написать вопрос и ждать, пока кто-нибудь посчитает нужным на него ответить (в то же время, можно получить и несколько ответов сразу от разных пользователей), то в чате общение происходит только с теми, кто присутствует в нём в настоящий момент, а результаты обмена сообщениями могут и не сохраняться.Под словом чат обычно понимается групповое общение, хотя к ним можно отнести и обмен текстом «один на один» посредством программ [мгновенного обмена сообщениями](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B0_%D0%BC%D0%B3%D0%BD%D0%BE%D0%B2%D0%B5%D0%BD%D0%BD%D0%BE%D0%B3%D0%BE_%D0%BE%D0%B1%D0%BC%D0%B5%D0%BD%D0%B0_%D1%81%D0%BE%D0%BE%D0%B1%D1%89%D0%B5%D0%BD%D0%B8%D1%8F%D0%BC%D0%B8), например, [XMPP](https://ru.wikipedia.org/wiki/XMPP), [ICQ](https://ru.wikipedia.org/wiki/ICQ) или дажеSMS. С развитием [информационных технологий](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D1%8B%D0%B5_%D1%82%D0%B5%D1%85%D0%BD%D0%BE%D0%BB%D0%BE%D0%B3%D0%B8%D0%B8) стали возможными ещё более глобальные коммуникации.

Историческим «докомпьютерным» предшественником чатов, несомненно, был [телефон](https://ru.wikipedia.org/wiki/%D0%A2%D0%B5%D0%BB%D0%B5%D1%84%D0%BE%D0%BD). Ни почта, ни телеграф не позволяли общаться в реальном времени и не были доступны в домашней обстановке. Изобретение и распространение телефона по планете вызвало настоящую революцию в средствах и способах общения. Возможность поговорить с собеседником на другой стороне Земли казалась настоящим чудом. Во второй половине [XX века](https://ru.wikipedia.org/wiki/XX_%D0%B2%D0%B5%D0%BA) начали бурно развиваться компьютеры. Однако долгое время они были большими и слишком дорогими, что препятствовало тому, чтобы расходовать драгоценное машинное время на забавы с обменом сообщениями вместо расчётов атомных бомб. К тому же до конца 60-х годов они не были связаны друг с другом. Предок Интернета, сеть [ARPANET](https://ru.wikipedia.org/wiki/ARPANET), в [1969 году](https://ru.wikipedia.org/wiki/1969_%D0%B3%D0%BE%D0%B4) насчитывала только четыре связанных друг с другом научных компьютера. Чуть позже, в [1971 году](https://ru.wikipedia.org/wiki/1971_%D0%B3%D0%BE%D0%B4), была придумана [электронная почта](https://ru.wikipedia.org/wiki/%D0%AD%D0%BB%D0%B5%D0%BA%D1%82%D1%80%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D0%BF%D0%BE%D1%87%D1%82%D0%B0), которая стала необычайно популярна ввиду своего удобства. Постепенно появились новые службы сообщений, такие, как [списки почтовой рассылки](https://ru.wikipedia.org/wiki/%D0%A0%D0%B0%D1%81%D1%81%D1%8B%D0%BB%D0%BA%D0%B8_%D1%8D%D0%BB%D0%B5%D0%BA%D1%82%D1%80%D0%BE%D0%BD%D0%BD%D0%BE%D0%B9_%D0%BF%D0%BE%D1%87%D1%82%D1%8B), [новостные группы](https://ru.wikipedia.org/wiki/%D0%93%D1%80%D1%83%D0%BF%D0%BF%D0%B0_%D0%BD%D0%BE%D0%B2%D0%BE%D1%81%D1%82%D0%B5%D0%B9) и[доски объявлений](https://ru.wikipedia.org/wiki/%D0%AD%D0%BB%D0%B5%D0%BA%D1%82%D1%80%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D0%B4%D0%BE%D1%81%D0%BA%D0%B0_%D0%BE%D0%B1%D1%8A%D1%8F%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D0%B9). Однако в то время сеть ARPANET ещё не могла легко взаимодействовать с другими сетями, построенными на других технических стандартах, что затрудняло её распространение. Но, тем не менее, эта проблема вскоре была решена после перехода сетей на протокол обмена данными [TCP/IP](https://ru.wikipedia.org/wiki/TCP/IP), который успешно применяется до сих пор. Именно в 1983 году термин «Интернет» закрепился за сетью ARPANET.

Программы для обмена текстовыми строками, несмотря на простоту самой идеи, появились не сразу. Примерно в 1974 году для [мэйнфрейма](https://ru.wikipedia.org/wiki/%D0%9C%D1%8D%D0%B9%D0%BD%D1%84%D1%80%D0%B5%D0%B9%D0%BC) [PLATO](https://ru.wikipedia.org/wiki/PLATO_(%D0%BA%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80%D0%BD%D0%B0%D1%8F_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0)) был разработана программа Talkomatic, потенциально позволявшая общаться между тысячей терминалов системы. В 1980-x появилась система Freelancing' Round table. Однако по-настоящему популярным стал разработанный в [1988 году](https://ru.wikipedia.org/wiki/1988_%D0%B3%D0%BE%D0%B4) протокол, названный [Internet Relay Chat](https://ru.wikipedia.org/wiki/Internet_Relay_Chat)(IRC), что примерно можно перевести как ретранслируемый интернет-разговор. Где-то в это же время появилось и распространилось само понятие «чат». Общение в IRC быстро стало популярным из-за простоты процесса и дружественности среды. В [1991 году](https://ru.wikipedia.org/wiki/1991_%D0%B3%D0%BE%D0%B4) во время операции «[Буря в пустыне](https://ru.wikipedia.org/wiki/%D0%92%D0%BE%D0%B9%D0%BD%D0%B0_%D0%B2_%D0%9F%D0%B5%D1%80%D1%81%D0%B8%D0%B4%D1%81%D0%BA%D0%BE%D0%BC_%D0%B7%D0%B0%D0%BB%D0%B8%D0%B2%D0%B5)» была организована IRC-трансляция новостей — сообщения со всего мира собирались в одном месте и в режиме реального времени передавались в IRC. Есть сведения, что подобным образом IRC использовался и во время [путча в СССР](https://ru.wikipedia.org/wiki/%D0%90%D0%B2%D0%B3%D1%83%D1%81%D1%82%D0%BE%D0%B2%D1%81%D0%BA%D0%B8%D0%B9_%D0%BF%D1%83%D1%82%D1%87), когда пользователи из Москвы моментально сообщали всему миру о происходящем на улицах

Разработчики IRC настолько хорошо продумали его архитектуру, что её с тех пор практически не требовалось изменять. Конечно, у него есть недостатки: короткие сообщения, проблема с кодировками, невозможность посмотреть историю сообщений при подключении. Однако он был и остаётся популярным средством для чата, хотя и в значительной мере потеснен со своих позиций. В частности, в [1998 году](https://ru.wikipedia.org/wiki/1998_%D0%B3%D0%BE%D0%B4) был придуман похожего назначения протокол [Jabber](https://ru.wikipedia.org/wiki/Jabber) — даже его название ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) jabber болтовня, трёп; тарабарщина) отсылало к слову chat. [Jabber](https://ru.wikipedia.org/wiki/Jabber) содержал в себе многие технические новшества и постепенно получил широкое распространение, а также стал основой многих сервисов. Были и другие протоколы, менее известные, например, [SIP](https://ru.wikipedia.org/wiki/SIP).

### 1. Исследовательская часть

### 1.1 Обзор систем мнгновенного обмена сообщениями

### Существует несколько разновидностей программной реализации чатов:

* [HTTP](https://ru.wikipedia.org/wiki/HTTP) или веб-чаты. Такой чат выглядит как обычная [веб-страница](https://ru.wikipedia.org/wiki/%D0%92%D0%B5%D0%B1-%D1%81%D1%82%D1%80%D0%B0%D0%BD%D0%B8%D1%86%D0%B0), где можно прочесть последние несколько десятков фраз, написанные участниками чата и [модераторами](https://ru.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B4%D0%B5%D1%80%D0%B0%D1%82%D0%BE%D1%80). Страница чата автоматически обновляется с заданной периодичностью.
* Чаты, использующие технологию [Adobe Flash](https://ru.wikipedia.org/wiki/Adobe_Flash). Вместо периодической перезагрузки страницы между клиентом и сервером открывается[сокет](https://ru.wikipedia.org/wiki/%D0%A1%D0%BE%D0%BA%D0%B5%D1%82_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D1%8B%D0%B9_%D0%B8%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81)), что позволяет моментально отправлять или получать сообщения, расходуя меньше [трафика](https://ru.wikipedia.org/wiki/%D0%A1%D0%B5%D1%82%D0%B5%D0%B2%D0%BE%D0%B9_%D1%82%D1%80%D0%B0%D1%84%D0%B8%D0%BA).
* [IRC](https://ru.wikipedia.org/wiki/IRC), специализированный [протокол](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%82%D0%BE%D0%BA%D0%BE%D0%BB_%D0%BF%D0%B5%D1%80%D0%B5%D0%B4%D0%B0%D1%87%D0%B8_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85) для чатов.
* Программы-чаты для общения в [локальных сетях](https://ru.wikipedia.org/wiki/%D0%9B%D0%BE%D0%BA%D0%B0%D0%BB%D1%8C%D0%BD%D0%B0%D1%8F_%D1%81%D0%B5%D1%82%D1%8C) (например, [Vypress Chat](https://ru.wikipedia.org/wiki/Vypress_Chat), [Intranet Chat](https://ru.wikipedia.org/wiki/Intranet_Chat), [Pichat](https://ru.wikipedia.org/w/index.php?title=Pichat&action=edit&redlink=1)). Часто есть возможность передачи файлов.
* Чаты, реализованные поверх сторонних протоколов (например, чат, использующий [ICQ](https://ru.wikipedia.org/wiki/ICQ)).
* Чаты, работающие по схеме [клиент-сервер](https://ru.wikipedia.org/wiki/%D0%9A%D0%BB%D0%B8%D0%B5%D0%BD%D1%82-%D1%81%D0%B5%D1%80%D0%B2%D0%B5%D1%80), это позволяет использовать их в сетях со сложной конфигурацией, а также управлять клиентскими приложениями (например, [Mychat](https://ru.wikipedia.org/wiki/Mychat), [Jabber](https://ru.wikipedia.org/wiki/Jabber)).

Веб-чаты базировались на технологиях [всемирной паутины](https://ru.wikipedia.org/wiki/%D0%92%D1%81%D0%B5%D0%BC%D0%B8%D1%80%D0%BD%D0%B0%D1%8F_%D0%BF%D0%B0%D1%83%D1%82%D0%B8%D0%BD%D0%B0), [HTTP](https://ru.wikipedia.org/wiki/HTTP) и [HTML](https://ru.wikipedia.org/wiki/HTML). Первоначально они представляли собой страницу с разговором и содержащую форму, посредством которой введённый текст отсылался на сервер. Сервер добавлял новые сообщения в текстовую область, удалял старые и обновлял файл. Важно заметить, что такой чат осуществлялся с заметной задержкой в десятки секунд, из-за того, что веб-средства не позволяли серверу сообщить клиенту об изменениях — клиент мог только запрашивать данные сам с некоторой периодичностью. Из-за этого за веб-чатами закрепилась репутация «тормозных», хотя позднее положение дел и было исправлено с применением технологий [AJAX](https://ru.wikipedia.org/wiki/AJAX) и [Flash](https://ru.wikipedia.org/wiki/Adobe_Flash). Другие системы сообщений не имели такого недостатка, поскольку разрабатывались без подобных технических ограничений. Также веб-чаты печально известны как мишени для атак на пользователей с применением уязвимостей в программном обеспечении (скриптах). Хотя к настоящему дню все крупные сервера уже в достаточной степени защитились от атак, чему немало способствовали их взломы в прошлом.Со временем, помимо обычных текстовых чатов, были придуманы видео, а также голосовые чаты. Видеочаты — это обмен текстовыми сообщениями плюс транслирование изображений с веб-камер. Поначалу это были фоточаты, из-за низкой пропускной способности каналов отправлялся не видеопоток, а картинка с некоторыми интервалами, что однако, давало возможность достаточно оперативно наблюдать смену эмоций у собеседника и было значительным прорывом. Позднее, конечно, стал транслироваться видеопоток, хотя и с низким разрешением. Веб-камеры являются простыми и дешёвыми, хотя обратная сторона этого — низкое разрешение видео и его плохое качество. Изображение получается с плохой цветопередачей, зашумлённое. Однако для целей общения такого качества более чем достаточно. Голосовые чаты тоже явились развитием идей обмена сообщениями. В настоящее время в компьютерных играх широко применяется система [TeamSpeak](https://ru.wikipedia.org/wiki/TeamSpeak), позволяющая общаться голосом между членами команды, не отвлекаясь от управления игрой. А общение по [Skype](https://ru.wikipedia.org/wiki/Skype) больше напоминает разговор по телефону, чем чат, хотя возможность отправки обычных текстовых сообщений в нём тоже присутствует.

Программы мгновенного обмена сообщениями ([англ.](https://ru.wikipedia.org/wiki/%D0%90%D0%BD%D0%B3%D0%BB%D0%B8%D0%B9%D1%81%D0%BA%D0%B8%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA) Instantmessenger, IM) используются для обмена сообщениями через [Интернет](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D1%80%D0%BD%D0%B5%D1%82) в реальном времени через службы мгновенных сообщений (Instant Messaging Service, IMS). Передаваться могут текстовые сообщения, звуковые сигналы, изображения, видео, а также производиться такие действия, как совместное рисование, игры и тому подобное.

Телечаты используются на телеканалах, таких, как [MTV](https://ru.wikipedia.org/wiki/MTV), [RU.TV](https://ru.wikipedia.org/wiki/RU.TV), [Bridge-TV](https://ru.wikipedia.org/wiki/Bridge-TV) и других. Сообщение передаётся путём отправки [SMS](https://ru.wikipedia.org/wiki/SMS) с мобильного. Чаще всего это объявления о знакомствах или поздравления с праздниками. Также на некоторых каналах ведётся общение с [диджеем](https://ru.wikipedia.org/wiki/%D0%94%D0%B8%D0%B4%D0%B6%D0%B5%D0%B9) или ведущим.

В начале 90-х годов самым популярным средством для онлайнового общения была служба IRC, которая предоставляла большой выбор тем для бесед с единомышленниками. Службой IRC пользовались многие специалисты, прежде всего компьютерные, что сформировало профессиональную среду, в которой можно было задать вопрос и получить консультацию. Позднее появились Web-чаты — этот сервис был проще, так как для него не нужен был собственный клиент, а достаточно было повсеместно распространенного браузера. Вследствие этого данную службу полюбили не столько специалисты, сколько обычные пользователи. Может быть, именно поэтому данный сервис у многих ассоциируется с пустыми беседами на отвлеченные темы.

Следующим шагом на пути совершенствования чата в Сети стала служба Instant Messaging, представляемая сегодня в первую очередь такими гигантами, как AOL, Microsoft и Yahoo. Среди основных преимуществ данной службы — более широкие возможности поиска собеседников по различным параметрам, возможность отслеживания состояния (оnline/оffline) собеседника из списка контактов и возможность «постучаться» к абоненту, вошедшему в Сеть. Основу каждой службы составляет не только технология, но и сообщество людей, или так называемое адресное пространство. Разные службы формируют его по-разному: либо самостоятельно (как это реализовано в службе ICQ), либо путем встраивания в существующее (например, Yahoo Messenger — в пространство членов Yahoo). Приведя краткий исторический экскурс развития служб онлайнового текстового общения, остановимся на каждом способе общения подробнее.

Ааббревиатура «IRC» расшифровывается как Internet Relay Chat и представляет собой многопользовательскую систему диалогового общения по Интернету, построенную по клиент-серверной архитектуре. IRC была разработана в Финляндии в конце 80-х годов и с тех пор распространилась по всему миру. Со временем появилось множество других сервисов, однако IRC по-прежнему остается популярным среди специалистов, в том числе программистов и Web-дизайнеров. Для доступа к IRC-дискуссии необходим IRC-клиент — программа, которая устанавливает соединение с ближайшим IRC-сервером и позволяет принимать и отправлять текстовые сообщения на IRC-сервер. IRC-сервер, в свою очередь, отвечает за то, чтобы все сообщения передавались участникам той или иной дискуссии. IRC-серверы объединены между собой в сети.

Первоначально существовала всего одна сеть IRC, но впоследствии она разделилась на несколько частей, и теперь существует несколько IRC-сетей (IRC Networks), не связанных друг с другом. Каждая сеть имеет собственное имя, например RusNet, IRCnet, Forestnet, UNDERnet, DALnet и т.д. Процесс появления новых сетей продолжается. Задержки при обмене сообщениями минимальны, и для обмена сообщениями с десятками абонентов из разных стран достаточно подключиться к ближайшему серверу.Внутри каждой IRC-сети существуют тематические области — каналы (channels), при подключении к которым можно видеть сообщения всех участников данного канала. Можно не только присоединиться к существующему каналу, но и создать собственный. Одни каналы функционируют постоянно, другие создаются временно и заканчивают свое существование после выхода последнего участника.Согласно общепринятому соглашению имена каналов начинаются с символа «решетка» (#), например: #russian, #russia, #RusTranslit, #russiancyrillic. Следует отметить, что каналы с одинаковыми именами в разных IRC-сетях являются разными каналами, поэтому вы не сможете связаться друг с другом, находясь в каналах с одинаковыми именами, но в разных IRC-сетях. IRC позволяет общаться не только по каналам, но и с помощью прямого соединения — Direct Communication Channel (DCC). Через DCC возможен как чат, так и обмен файлами. IRC-канал иногда служит для первоначальной стыковки собеседников, а дальнейшее общение осуществляется посредством DCC.

Существует несколько IRC-клиентов под Windows: mIRC, PIRCH, VIRC, XiRCON и др. Самым популярным среди них является mIRC.Скачать mIRC можно по адресу [http://www.mirc.com/get.html](http://www.compress.ru/article.aspx?id=10364&part=http%3a%2f%2fwww.mirc.com%2fget1ext1l). Далее необходимо заполнить форму mIRC Options, автоматически появляющуюся при запуске программы, где следует указать свое полное имя, адрес электронной почты и два варианта так называемого ника (nickname): основного и альтернативного. Ваш ник не является уникальным в канале, и если он окажется занятым, то система заменит его на альтернативный.Выбрать IRC-сервер можно в ниспадающем меню. Если нужного сервера в данном списке нет, его можно добавить путем нажатия кнопки Add.После того как выбран нужный сервер, остается лишь нажать первую слева кнопку Connect to IRC-сервер. В случае неудачного подключения следует повторить попытку или использовать другой сервер.При удачном подключении на экране появится панель выбора каналов mIRC Channels Folder. Для посещения любого из указанных каналов нужно выделить его в списке и нажать кнопку Join.Поскольку существует несколько кодировок русского языка, а сервисом IRC пользуются владельцы разных систем, принято применять транслитерацию. Обычно на каналах запрещается использовать цветные надписи, что связано с тем, что их по-разному кодируют различные IRC-клиенты.Как известно, не все участники чата соблюдают правила поведения, доставляя неудобства другим пользователям сети. Непечатные выражения, разного рода преследования и домогательства, реклама чужих каналов и Web-страниц — все это в IRC не приветствуется. Для наведения порядка в чате в IRC на уровне каналов имеются операторы каналов, которые обладают определенной властью на канале.Отличить оператора канала можно по наличию символа @ перед его ником. Оператор может выпроводить с канала нарушителей, заблокировать вход на канал и т.д. Статус оператора действителен только в пределах одного канала.В отсутствие операторов канала функции поддержания порядка могут выполнять специальные программы — боты. Такой бот может узнавать своих и давать им статус оператора. Бот также может распознавать ненормативную лексику и изгонять провинившихся с канала.Аналогично на уровне сети IRC действуют IRC-операторы (IRCOp), которые обладают неограниченной властью в IRC-сети, то есть могут выбрасывать любого пользователя из сети, запрещать доступ с определенных узлов и т.д. IRC-операторы — это, как правило, администраторы IRC-серверов, которые следят за работой всей IRC-сети, хотя и не вмешиваются в мелкие конфликты.

Web-чат — многопользовательская система диалогового общения, построенная на базе Web-серверов. Web-чат представляет собой службу, с помощью которой можно получать и отправлять текстовые сообщения в режиме реального времени на специально предназначенный для этой цели Web-сервер. Обычно Web-чат не требует отдельной программы и работает через обычный браузер.По сравнению с IRC данный сервис проще в работе, а также позволяет использовать более красочное оформление. В отличие от IRC-сетей, Web-чаты существуют независимо друг от друга, причем каждый из них имеет свои особенности.

Сравнивая Web-чаты и службу IRC, следует отметить, что Web-чаты обладают своими достоинствами и недостатками. Если IRC позволяет пользователю без ведома администрации создавать собственные каналы и через IRC пересылать друг другу файлы (DDC), то Web-чат не дает такой возможности. Обычно для участия в Web-чате необходимо зарегистрироваться: выбрать логин, пароль, ник (nickname) и цвет, которым будет отображаться вводимый текст. Войдя в тот или иной чат, можно вступить в диалог либо просто просматривать чужую переписку — такое поведение называется lurking, что означает пассивный просмотр сетевых новостей и конференций.Существуют тысячи Web-чатов на любые темы. Хотя аналога каналов, существующих в IRC, в Web-чатах нет, самих Web-серверов, на которых реализован чат, — огромное количество. Многие Web-сайты дополняют свой информационный сервис возможностью сразу обсудить какую-то тему.

ICQ — это персональное коммуникационное средство, которое поддерживает целый набор Интернет-приложений и служб. Наиболее популярным способом общения в ICQ является обмен мгновенными сообщениями (Instant Messaging), позволяющий отправлять сообщение, которое тут же возникает на экране адресата. Кроме чата, пользователи ICQ получают целый ряд дополнительных сервисов. Так, можно зарегистрировать бесплатный почтовый ящик электронной почты, за которым будет следить клиент ICQ и сообщать о поступлении новых писем. С помощью ICQ-клиента можно записать и послать по электронной почте звуковое сообщение, отправить SMS-сообщение на мобильный телефон и получить посланный с телефона ответ. Имея звуковую карту, микрофон и наушники, можно позвонить с одного компьютера на другой, если, конечно, на обоих запущены клиенты ICQ. За умеренную (по сравнению с обычными звонками за рубеж) плату можно позвонить на любой телефон. Используя мультипользовательский режим, групповые пользователи могут проводить конференции и играть в игры.

Первая версия программы была создана в 1996 году израильской фирмой Mirabilis, которую в 1998 году приобрел AOL. Сегодня ICQ пользуется более 120 млн. человек по всему миру, благодаря чему ICQ предоставляет широчайшие возможности поиска интересных собеседников и единомышленников. Подобно тому как поисковые машины позволяют вам разыскать ту или иную информацию в Web, служба ICQ помогает найти собеседников по разным критериям: увлечениям, интересам, вероисповеданию и т.д.

Служба построена по клиент-серверной архитектуре. ICQ-клиент можно бесплатно загрузить с серверов <http://www.icq.com/>. После установки ICQ необходима регистрация на сервере, который соединен с группой серверов, покрывающих сетью весь Интернет. В момент регистрации пользователь получает уникальный ICQ-номер. ICQ дает возможность внести и дополнительную персональную информацию. После регистрации можно составить список абонентов — так называемый контактный лист.Как только вы подключаетесь к Сети, ICQ-клиент определяет текущий IP-адрес вашего компьютера и отсылает его на сервер вместе с именами всех абонентов, зарегистрированных в вашем контактном листе.Сервер делает временную запись о данных вашего подключения и всех абонентах из вашего контактного листа. Если сервер обнаруживает, что кто-то из вашего листа контактов вошел в Сеть, он посылает сигнал на ваш ICQ-клиент с информацией о параметрах подключения этого пользователя. Аналогично ICQ-сервер посылает информацию о вашем подключении. После того как ваш ICQ-клиент получает информацию о подключении того или иного абонента, он меняет его статус на статус «В сети». При щелчке мышью по имени абонента из вашего контактного листа открывается окно, в котором можно вводить текст. Вы набираете текст сообщения, нажимаете кнопку «Отправить», и сообщение уходит вашему абоненту.Если во время сессии работы с ICQ вы добавите в ваш список контактов нового абонента, то ваш ICQ-клиент известит об этом сервер. Когда диалог заканчивается и вы выходите из сессии, ваш ICQ-клиент посылает сообщение на сервер, а сервер, в свою очередь, посылает сообщения всем абонентам из вашего контактного листа (которые находятся в онлайне) с извещением о том, что вы перешли в офлайновый режим. Сервер уничтожает временную запись, содержащую контактную информацию о вашем клиенте, а в клиентах ваших абонентов ваш статус меняется на «не в сети».

Программа MSN Messenger во многом повторяет функции ICQ, но в большей степени привязана к услугам Microsoft. Как и ICQ, MSN Messenger позволяет видеть, когда ваши собеседники находятся в Сети, и обмениваться с ними мгновенными сообщениями и звонками, посылать сообщения на пейджер, файлы и т.д. MSN Messenger дает возможность общаться не только с одним собеседником — для общения можно пригласить до 14 пользователей. Программа также позволяет позвонить в любую точку мира по низким тарифам, а наиболее подходящего поставщика телефонных услуг можно выбрать из списка.MSN Messenger дает пользователю возможность находить в каталоге пользователей сети MSN людей со схожими интересами, а также быть в курсе последних событий с помощью страницы MSN Today, где ежедневно помещается свежая информация из сети MSN, специально подготовленная для пользователей программы MSN Messenger, и откуда можно получить доступ к странице «Моя сеть MSN», к службе поиска MSN Search и службе Hotmail. Программа позволяет получать оповещения о новых сообщениях, поступивших через службу MSN Hotmail.

Skype — [бесплатное](https://ru.wikipedia.org/wiki/Freeware) [проприетарное программное обеспечение](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%BF%D1%80%D0%B8%D0%B5%D1%82%D0%B0%D1%80%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B5_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D0%B5) с [закрытым кодом](https://ru.wikipedia.org/wiki/%D0%97%D0%B0%D0%BA%D1%80%D1%8B%D1%82%D1%8B%D0%B9_%D0%B8%D1%81%D1%85%D0%BE%D0%B4%D0%BD%D1%8B%D0%B9_%D0%BA%D0%BE%D0%B4), обеспечивающее текстовую, голосовую связь и [видеосвязь](https://ru.wikipedia.org/wiki/%D0%92%D0%B8%D0%B4%D0%B5%D0%BE%D0%BA%D0%BE%D0%BD%D1%84%D0%B5%D1%80%D0%B5%D0%BD%D1%86%D0%B8%D1%8F) через [Интернет](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D1%80%D0%BD%D0%B5%D1%82) между [компьютерами](https://ru.wikipedia.org/wiki/%D0%9F%D0%B5%D1%80%D1%81%D0%BE%D0%BD%D0%B0%D0%BB%D1%8C%D0%BD%D1%8B%D0%B9_%D0%BA%D0%BE%D0%BC%D0%BF%D1%8C%D1%8E%D1%82%D0%B5%D1%80) ([IP-телефония](https://ru.wikipedia.org/wiki/IP-%D1%82%D0%B5%D0%BB%D0%B5%D1%84%D0%BE%D0%BD%D0%B8%D1%8F)), опционально используя технологии [пиринговых сетей](https://ru.wikipedia.org/wiki/%D0%9E%D0%B4%D0%BD%D0%BE%D1%80%D0%B0%D0%BD%D0%B3%D0%BE%D0%B2%D0%B0%D1%8F_%D1%81%D0%B5%D1%82%D1%8C), а также платные услуги для звонков на [мобильные](https://ru.wikipedia.org/wiki/%D0%9C%D0%BE%D0%B1%D0%B8%D0%BB%D1%8C%D0%BD%D1%8B%D0%B9_%D1%82%D0%B5%D0%BB%D0%B5%D1%84%D0%BE%D0%BD) и [стационарные](https://ru.wikipedia.org/wiki/%D0%A2%D0%B5%D0%BB%D0%B5%D1%84%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D0%B5%D1%82%D1%8C_%D0%BE%D0%B1%D1%89%D0%B5%D0%B3%D0%BE_%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F) [телефоны](https://ru.wikipedia.org/wiki/%D0%A2%D0%B5%D0%BB%D0%B5%D1%84%D0%BE%D0%BD). Skype имеет 663 миллиона пользователей по состоянию на конец 2010 года.Программа также позволяет совершать [конференц-звонки](https://ru.wikipedia.org/wiki/%D0%90%D1%83%D0%B4%D0%B8%D0%BE%D0%BA%D0%BE%D0%BD%D1%84%D0%B5%D1%80%D0%B5%D0%BD%D1%86%D0%B8%D1%8F) (до 25 голосовых абонентов, включая инициатора), [видеозвонки](https://ru.wikipedia.org/wiki/%D0%92%D0%B8%D0%B4%D0%B5%D0%BE%D0%BA%D0%BE%D0%BD%D1%84%D0%B5%D1%80%D0%B5%D0%BD%D1%86%D0%B8%D1%8F)(в том числе видеоконференции до 10 абонентов), а также обеспечивает [передачу текстовых сообщений](https://ru.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D0%BC%D0%B3%D0%BD%D0%BE%D0%B2%D0%B5%D0%BD%D0%BD%D0%BE%D0%B3%D0%BE_%D0%BE%D0%B1%D0%BC%D0%B5%D0%BD%D0%B0_%D1%81%D0%BE%D0%BE%D0%B1%D1%89%D0%B5%D0%BD%D0%B8%D1%8F%D0%BC%D0%B8) (чат) и передачу[файлов](https://ru.wikipedia.org/wiki/%D0%A4%D0%B0%D0%B9%D0%BB). Есть возможность вместо изображения с веб-камеры передавать изображение с экрана монитора, а также создавать и отправлять видеосообщения пользователям настольных версий программы.

Программные клиенты Skype выпущены для [Windows](https://ru.wikipedia.org/wiki/Windows), [Mac OS X](https://ru.wikipedia.org/wiki/Mac_OS_X), [Linux](https://ru.wikipedia.org/wiki/Linux), [iOS](https://ru.wikipedia.org/wiki/IOS), [Windows Phone](https://ru.wikipedia.org/wiki/Windows_Phone), [Open webOS](https://ru.wikipedia.org/wiki/Open_webOS), [Android](https://ru.wikipedia.org/wiki/Android), [PSP](https://ru.wikipedia.org/wiki/PlayStation_Portable),[Maemo](https://ru.wikipedia.org/wiki/Maemo), [Xbox 360](https://ru.wikipedia.org/wiki/Xbox_360), [PlayStation Vita](https://ru.wikipedia.org/wiki/PlayStation_Vita), [Symbian](https://ru.wikipedia.org/wiki/Symbian_OS), [Blackberry](https://ru.wikipedia.org/wiki/Blackberry). Также была выпущена версия для [Java](https://ru.wikipedia.org/wiki/Java).

Компания Skype Technologies была основана в 2003 году шведом Никласом Зеннстремом и датчанином Янусом Фриисом. Первые версии программы (0.97, 0.98) появились в сентябре-октябре [2003 года](https://ru.wikipedia.org/wiki/2003_%D0%B3%D0%BE%D0%B4) на сайте, декларирующем себя как шведский. Программа имела простой интерфейс, изначально адаптированный под голосовую связь, в отличие от мессенджеров вроде [ICQ](https://ru.wikipedia.org/wiki/ICQ) и [MSN Messenger](https://ru.wikipedia.org/wiki/MSN_Messenger) Skype при инсталляции сам выбирал язык локализации [Windows](https://ru.wikipedia.org/wiki/Windows) и имел более простую и быструю регистрацию логина, нежели в конкурирующих программах. Именно простота установки, освоения и использования программы быстро привлекли к ней внимание большого числа пользователей.В версии 1.2 впервые появился автоответчик ([Voicemail](https://ru.wikipedia.org/w/index.php?title=Voicemail&action=edit&redlink=1)), а начиная с версии 1.3 любой его владелец мог оставить сообщения другим пользователям, даже если они не использовали эту услугу. В версии 1.2 также появилась функция [SkypeIn](https://ru.wikipedia.org/w/index.php?title=SkypeIn&action=edit&redlink=1), позволяющая связать учётную запись Skype с [телефонным](https://ru.wikipedia.org/wiki/%D0%A2%D0%B5%D0%BB%D0%B5%D1%84%D0%BE%D0%BD) номером.Начиная с версии 1.4 появилась возможность перенаправления звонков на другие учётные записи Skype, а также на обычные телефоны.В версии 2.0 впервые появилась возможность видеосвязи, а в 2.5 — возможность отправки [SMS](https://ru.wikipedia.org/wiki/SMS), организации [SkypeCast](https://ru.wikipedia.org/wiki/SkypeCast) (эта технология официально отключена с[1 сентября](https://ru.wikipedia.org/wiki/1_%D1%81%D0%B5%D0%BD%D1%82%D1%8F%D0%B1%D1%80%D1%8F) [2008 года](https://ru.wikipedia.org/wiki/2008_%D0%B3%D0%BE%D0%B4)) и интеграции с [Microsoft Outlook](https://ru.wikipedia.org/wiki/Microsoft_Outlook).В версии 3 появился сервис [SkypeFind](https://ru.wikipedia.org/w/index.php?title=SkypeFind&action=edit&redlink=1), который позволяет пользователям создавать список фирм и давать к ним описание.

В отличие от многих других программ [IP-телефонии](https://ru.wikipedia.org/wiki/IP-%D1%82%D0%B5%D0%BB%D0%B5%D1%84%D0%BE%D0%BD%D0%B8%D1%8F), для передачи данных Skype изначально использовал [P2P](https://ru.wikipedia.org/wiki/P2P)-[архитектуру](https://ru.wikipedia.org/wiki/%D0%90%D1%80%D1%85%D0%B8%D1%82%D0%B5%D0%BA%D1%82%D1%83%D1%80%D0%B0_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B3%D0%BE_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D1%8F). Каталог пользователей Skype был распределён по компьютерам пользователей сети Skype, что позволяло сети легко масштабироваться до очень больших размеров (в данный момент более 100 миллионов пользователей, 15—25 миллионов онлайн) без дорогой инфраструктуры [централизованных](https://ru.wikipedia.org/wiki/%D0%A6%D0%B5%D0%BD%D1%82%D1%80%D0%B0%D0%BB%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F) [серверов](https://ru.wikipedia.org/wiki/%D0%A1%D0%B5%D1%80%D0%B2%D0%B5%D1%80_(%D0%B0%D0%BF%D0%BF%D0%B0%D1%80%D0%B0%D1%82%D0%BD%D0%BE%D0%B5_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D0%B5)).Skype может [маршрутизировать](https://ru.wikipedia.org/wiki/%D0%9C%D0%B0%D1%80%D1%88%D1%80%D1%83%D1%82%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F) звонки через компьютеры других пользователей. Это позволяет соединяться друг с другом пользователям, находящимся за[NAT](https://ru.wikipedia.org/wiki/NAT)или [брандмауэром](https://ru.wikipedia.org/wiki/%D0%9C%D0%B5%D0%B6%D1%81%D0%B5%D1%82%D0%B5%D0%B2%D0%BE%D0%B9_%D1%8D%D0%BA%D1%80%D0%B0%D0%BD), однако создаёт дополнительную нагрузку на компьютеры и [каналы](https://ru.wikipedia.org/wiki/%D0%9A%D0%B0%D0%BD%D0%B0%D0%BB_%D1%81%D0%B2%D1%8F%D0%B7%D0%B8) пользователей, подключённых к Интернету напрямую.Единственным центральным элементом для Skype являлся сервер идентификации, на котором хранятся [учётные записи](https://ru.wikipedia.org/wiki/%D0%A3%D1%87%D1%91%D1%82%D0%BD%D0%B0%D1%8F_%D0%B7%D0%B0%D0%BF%D0%B8%D1%81%D1%8C) пользователей и [резервные копии](https://ru.wikipedia.org/wiki/%D0%A0%D0%B5%D0%B7%D0%B5%D1%80%D0%B2%D0%BD%D0%B0%D1%8F_%D0%BA%D0%BE%D0%BF%D0%B8%D1%8F) их списков контактов. Центральный сервер нужен только для установки связи. После того как связь установлена, компьютеры могли пересылать голосовые данные напрямую друг другу (если между ними есть прямая связь) или через Skype-посредник (суперузел - ранее - компьютер, у которого есть внешний [IP-адрес](https://ru.wikipedia.org/wiki/IP-%D0%B0%D0%B4%D1%80%D0%B5%D1%81) и открыт [TCP-порт](https://ru.wikipedia.org/wiki/TCP-%D0%BF%D0%BE%D1%80%D1%82) для Skype, затем - сервера Microsoft). В частности, если два компьютера, находящиеся внутри одной [локальной сети](https://ru.wikipedia.org/wiki/%D0%9B%D0%BE%D0%BA%D0%B0%D0%BB%D1%8C%D0%BD%D0%B0%D1%8F_%D1%81%D0%B5%D1%82%D1%8C), установили между собой Skype-соединение, то связь с Интернетом можно было прервать, и разговор продолжался бы вплоть до его завершения пользователями или какого-либо сбоя связи внутри локальной сети.Благодаря используемым Skype [кодекам](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%B4%D0%B5%D0%BA) (алгоритмам сжатия данных) [SILK](https://ru.wikipedia.org/wiki/SILK) (8-24 [кГц](https://ru.wikipedia.org/wiki/%D0%9A%D0%93%D1%86)), [G.729](https://ru.wikipedia.org/wiki/G.729) (8 кГц) и [G.711](https://ru.wikipedia.org/wiki/G.711) (ранее использовались также [ILBC](https://ru.wikipedia.org/wiki/ILBC) и [ISAC](https://ru.wikipedia.org/w/index.php?title=ISAC&action=edit&redlink=1)) и при достаточной скорости интернет-соединения (30—60 кбит/с) в большинстве случаев качество звука сопоставимо с качеством [обычной телефонной связи](https://ru.wikipedia.org/wiki/%D0%A2%D0%B5%D0%BB%D0%B5%D1%84%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D0%B5%D1%82%D1%8C_%D0%BE%D0%B1%D1%89%D0%B5%D0%B3%D0%BE_%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F), а при хороших условиях соединения заметно лучше.

При установке соединения между ПК данные [шифруются](https://ru.wikipedia.org/wiki/%D0%A8%D0%B8%D1%84%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5) при помощи [AES](https://ru.wikipedia.org/wiki/Advanced_Encryption_Standard)-256, для передачи [ключа](https://ru.wikipedia.org/wiki/%D0%9A%D0%BB%D1%8E%D1%87_(%D0%BA%D1%80%D0%B8%D0%BF%D1%82%D0%BE%D0%B3%D1%80%D0%B0%D1%84%D0%B8%D1%8F)) которого, в свою очередь, используется 1024-[битный](https://ru.wikipedia.org/wiki/%D0%91%D0%B8%D1%82) ключ[RSA](https://ru.wikipedia.org/wiki/RSA). Открытые ключи пользователей сертифицируются центральным [сервером](https://ru.wikipedia.org/wiki/%D0%A1%D0%B5%D1%80%D0%B2%D0%B5%D1%80_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B5_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D0%B5)) Skype при входе в систему с использованием 1536- или 2048-битных [сертификатов](https://ru.wikipedia.org/wiki/%D0%A6%D0%B8%D1%84%D1%80%D0%BE%D0%B2%D0%BE%D0%B9_%D1%81%D0%B5%D1%80%D1%82%D0%B8%D1%84%D0%B8%D0%BA%D0%B0%D1%82)RSA, однако условия использования Skype предусматривают доступность расшифрованных данных владельцу сети (Microsoft), работникам Microsoft или аффилированных компаний, а также провайдерам сети Интернет. Также используется автоматическое сканирование пересылаемого текста и ссылок из него для борьбы со спамом и мошенничеством. Также допускается перехват и ручная обработка пересылаемых текстовых сообщений.

После приобретения компанией Microsoft, все суперузлы и каталог пользователей были перенесены на серверные мощности компании Microsoft. С их помощью были увеличены возможности по масштабированию сети, добавлена возможность антивирусной проверки пересылаемых по сети сообщений и ссылок, содержащихся в них.

[Протокол Skype](https://ru.wikipedia.org/w/index.php?title=%D0%9F%D1%80%D0%BE%D1%82%D0%BE%D0%BA%D0%BE%D0%BB_Skype&action=edit&redlink=1) ([англ.](https://en.wikipedia.org/wiki/Skype_protocol)) закрыт и используется только оригинальным программным обеспечением Skype. При помощи [API](https://ru.wikipedia.org/wiki/API) к его функциям могут получать доступ программы сторонних разработчиков.[3 июня](https://ru.wikipedia.org/wiki/3_%D0%B8%D1%8E%D0%BD%D1%8F) [2011](https://ru.wikipedia.org/wiki/2011) формат сообщений протокола Skype был взломан с помощью "обратной разработки", и исходный код программы, работающей по данному протоколу, оказался в сети [Интернет](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D1%80%D0%BD%D0%B5%D1%82)[[50]](https://ru.wikipedia.org/wiki/Skype#cite_note-50). Позже код был удален в соответствии с [DMCA](https://ru.wikipedia.org/wiki/DMCA) запросом.Официально подтверждённых разработчиком случаев расшифровки и/или перехвата данных в Skype было не зафиксировано до приблизительно 2008-2010 годов, и большинство [спецслужб](https://ru.wikipedia.org/wiki/%D0%A1%D0%BF%D0%B5%D1%86%D1%81%D0%BB%D1%83%D0%B6%D0%B1%D0%B0) выражали по этому поводу недовольств.

Для стабильного использования [видеосвязи](https://ru.wikipedia.org/wiki/%D0%92%D0%B8%D0%B4%D0%B5%D0%BE%D1%81%D0%B2%D1%8F%D0%B7%D1%8C) необходима скорость [интернет](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D1%80%D0%BD%D0%B5%D1%82)-соединения более 200 [кбит/с](https://ru.wikipedia.org/wiki/%D0%9A%D0%B1%D0%B8%D1%82/%D1%81) и желательна [тактовая частота](https://ru.wikipedia.org/wiki/%D0%A2%D0%B0%D0%BA%D1%82%D0%BE%D0%B2%D0%B0%D1%8F_%D1%87%D0%B0%D1%81%D1%82%D0%BE%D1%82%D0%B0) [процессора](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D1%86%D0%B5%D1%81%D1%81%D0%BE%D1%80) не менее 1 [ГГц](https://ru.wikipedia.org/wiki/%D0%93%D0%93%D1%86).Skype позволяет пользователям общаться не только с помощью голоса, но и более традиционным способом — с помощью [текстовых сообщений](https://ru.wikipedia.org/wiki/%D0%A1%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0_%D0%BC%D0%B3%D0%BD%D0%BE%D0%B2%D0%B5%D0%BD%D0%BD%D0%BE%D0%B3%D0%BE_%D0%BE%D0%B1%D0%BC%D0%B5%D0%BD%D0%B0_%D1%81%D0%BE%D0%BE%D0%B1%D1%89%D0%B5%D0%BD%D0%B8%D1%8F%D0%BC%D0%B8) (IM-чата). Голосовой чат позволяет разговаривать как с одним пользователем, так и устраивать [конференц-связь](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BD%D1%84%D0%B5%D1%80%D0%B5%D0%BD%D1%86-%D1%81%D0%B2%D1%8F%D0%B7%D1%8C). Он использует собственные [кодеки](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%B4%D0%B5%D0%BA).Skype-чат позволяет устраивать групповые чаты, посылать [смайлики](https://ru.wikipedia.org/wiki/%D0%A1%D0%BC%D0%B0%D0%B9%D0%BB%D0%B8%D0%BA), хранить историю. Также предоставляются обычные для [IM-чатов](https://ru.wikipedia.org/wiki/IM-%D1%87%D0%B0%D1%82) возможности — профили пользователя, индикаторы состояния (статус) и так далее. Кроме того, Skype предоставляет возможность [обмена файлами](https://ru.wikipedia.org/wiki/%D0%A1%D0%BE%D0%B2%D0%BC%D0%B5%D1%81%D1%82%D0%BD%D0%BE%D0%B5_%D0%B8%D1%81%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5_%D1%84%D0%B0%D0%B9%D0%BB%D0%BE%D0%B2) без ограничения размера и со стандартными опциями временной остановки пересылки и автоматического возобновления при подключении после потери связи или выключения программы Skype до конца передачи файла.Размер [трафика](https://ru.wikipedia.org/wiki/%D0%A1%D0%B5%D1%82%D0%B5%D0%B2%D0%BE%D0%B9_%D1%82%D1%80%D0%B0%D1%84%D0%B8%D0%BA), потребляемого Skype при разговоре, может варьироваться, в зависимости от некоторых факторов, таких, как скорость подключения к сети[интернет](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D1%80%D0%BD%D0%B5%D1%82) (и вызывающего и вызываемого), а также от насыщенности самого разговора.Количество трафика, потребляемого Skype, особенно если дело касается передачи [видео](https://ru.wikipedia.org/wiki/%D0%92%D0%B8%D0%B4%D0%B5%D0%BE), напрямую зависит от ширины (скорости) канала. Также Skype всегда подстраивается как под скорость соединения [Интернет](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D1%80%D0%BD%D0%B5%D1%82), так и под ресурсы системы. Таким образом, трафик при видеозвонке в разных условиях может очень сильно отличаться.

# Выбор технологии Java для разработки

Java - полностью объектно-ориентированный язык программирования. В Java отсутствует понятие процедур. С помощью Java решаются различные задачи и тот же самый круг проблем, что и на других языках программирования. Java может использоваться для создания приложений и апплетов. Приложение - программа, которая выполняется на компьютере, под его операционной системой. Приложения Java могут быть непосредственно выполнены, используя интерпретатор Java. Апплет - небольшая программа работающая с окнами, которые внедрены в страницу HTML. Чтобы выполнить Java апплеты, нужна поддержка Java Web-браузером.

Java - это интерпретируемый и компилированный язык программирования. Исходный текст (файлы с расширением a Java) откомпилирован со справкой компилятора Java (javac), который преобразовывает исходный текст в байт-код (файлы с расширением a.class). Цель Java выполнение программы всегда, в любое время на различных платформах.

Java - язык, который быстро изучаетсяи позволяет достаточно быстро перейти с другой платформы.

В Java всё может быть объектом. Так основное внимание уделяется свойствам и методам, которые оперируют данными в нашем приложении и нет концентрации только на процедурах. Свойства и методы вместе описывают состояние и поведение объекта. Java может использоваться для разработки приложений, которые работают на различных платформах, операционных системах и графических интерфейсов пользователя. Java предназначен также для поддержки сетевых приложений. Таким образом Java широко используется как инструмент разработки в среде подобной Internet, где существуют различные платформы.

Java - язык со строгим контролем типов, так что требуется явное объявление метода. Java проверяет код во время трансляции и во время интерпретации. Таким образом устраняются некоторые типы ошибок при программировании. Java не имеет указателей и соответственно арифметических операций над ними. Все данные массивов и строк проверяются во время выполнения, что исключает возможность выхода за границы дозволенного. Преобразование объектов с одного типа на другой также проверяется во время выполнения.

В Java, нет необходимости контролировать освобождение памяти. Это делается автоматически. Также обработка исключений упрощает задачу обработки ошибок и восстановления.

Java обеспечивает управляемую среду, в которой выполнена программа. Java никогда не предполагает, что код может быть безопасно выполнен. И так как Java - больше чем язык программирования, он обеспечивает несколько уровней контроля защиты.

Первый уровень - это безопасность, обеспеченная языком Java. Свойства и методы описываются в классе, и к ним можно обратиться только через интерфейс, обеспеченный классом. Java не позволяет никаких операций с указателями, таким образом запрещает прямой доступ к памяти. Избегается переполнение массивов. Проблемы, связанные с безопасностью и мобильностью, скрыты. На следующем уровне компилятор, прежде чем приступить к компиляции кода, проверяет безопасность кода и затем следует в соответствии с протоколами, установленными Java. Третий уровень - это безопасность, обеспеченная интерпретатором. Прежде, чем байт-код будет фактически выполнен, он является полностью укрытым верификатором. Четвертый уровень заботится о загрузке классов. Загрузчик класса гарантирует, что класс не нарушает ограничения доступа прежде, чем он загружен в систему.   
Мы способны выполнить код Java на множественных платформах. Нейтралитет достигается при смешении трансляции и интерпретации.

1. Программы Java оттранслированы в байт-код компилятором.

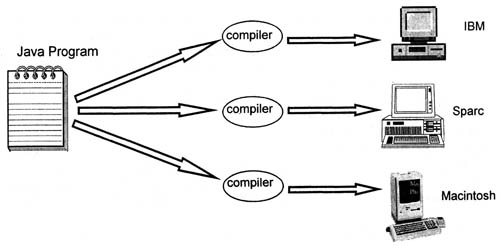
2. Байт-код выполняется интерпретатором (Виртуальная Машина Java).

Независимость от платформы означает лёгкость переноса программы с одного компьютера на другой компьютер без каких-либо трудностей. Также Java - платформа, независима на обоих уровнях, то есть на первичном (исходном) и на вторичном уровне.

Java - это язык со строгим контролем типов. И эти типы данных в Java одинаковы для всех платформ. Java имеет свои собственные библиотеки фундаментальных классов, которые облегчают запись кода для программиста, который может быть перемещен с одной машины на другую, без потребности перезаписи кода. Независимость от платформы на исходном уровне означает, что есть возможность перемещения исходного текста программы из одной системы в другую, компилируя код, и работая в системе.

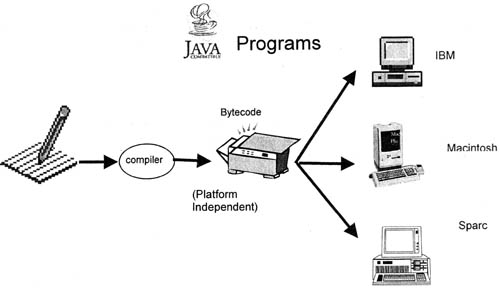
Платформа, независимая на вторичном уровне означает, что откомпилированный двоичный файл может быть выполнен на различных платформах, не перетранслируя код, если они имеют Виртуальную Машину Java, которая функционирует как интерпретатор.

Рис.1 Компилирование программ традиционным способом.



Для программ записанных в С и C++ или на любом другом языке, компилятор преобразует набор команды в машинный код или команды процессора. Эти команды являются специальными для нашего процессора. В результате, если мы хотим использовать этот код в некоторой другой системе, мы должны найти компилятор для этой системы, и мы должны компилировать код еще раз так, чтобы мы имели машинный код, определенный для этой машины. Программы Java (рисунок 1.2)могут быть выполнены на различных машинах.

Рис.2 Компилирование кода в Java.



Java, разделена на две части: Компилятор Java и интерпретатор Java.

В отличие от С и C++, компилятор Java преобразует исходный текст в байт-коды, которые являются машинно-независимыми. Байт-коды - это только части команд Java, разрезанные на байты, которые могут быть декодированы любым процессором.

Интерпретатор Java, также именуемый как JVM (Виртуальная Машина Java) или Java Runtime Interpreter выполняет байт-коды Java. Интерпретатор Java является частью среды разработки. Скоро будет время, когда каждая операционная система будет иметь в своём составе JVM.

Java - это интерпретируемый язык. Это означает, что каждая команда оттранслирована в машинный код во время выполнения, а не в течение трансляции.

Это качество позволяет реализовывать нейтралитет платформы:   
"WORA", или“Write Once Run Anywhere.”А также позволяет перезаписывать и изменять программу, во время её выполнения.Трансляция java и процедура выполнения включают следующее:

* Различные исходные файлы обрабатываются компилятором javac, для получения множество файлов класса. Эти файлы содержат байт-код, который не зависит от архитектуры и платформы исполняющей его.
* Файлы класса Java могут быть выполнены со справкой загрузчика (интерпретатора), утилиты по имени java, которая функционирует, чтобы транслировать универсальные байт-коды Java в машинные выполняемые коды. Никакой компоновщик при этом не требуется.

Из-за его интерпретирующей процедуры выполнения, Java имеет следующие преимущества:

* Файлы класса Java могут быть выполнены на любой платформе при условии, что данная платформа имеет надлежащую утилиту загрузчика java.
* Файлы класса Java делают более эффективное использование памяти, нежели отдельные (часто большие) выполняемые программы, потому что файлы класса могут быть связаны загрузчиком на основании управляемого запроса.

Java разработан, чтобы хорошо работать на центральных процессорах с очень низким энергопотреблением. Байт-код Java был тщательно продуман так, чтобы его можно сразу непосредственно транслировать в машинный код с высокой эффективностью, используя компилятор.

Сложно переоценить встроеннуюподдержкумногопоточности, снабжающую программистов Java мощным инструментом для распараллеливания программ посредством потоков. Потоки - это основной краеугольный камень Java. Библиотека Java обеспечивает класс потока, который содержит обширную коллекцию методов запуска, выполнения, и остановки потока, а также проверки его состояния.

Java - динамичный адаптированный язык программирования. Программы Java несут много информации во время выполнения, для проверки правильности обращения к объектам во время выполнения. Это свойство позволяет динамически безопасно связать код.

Виртуальная машина - это программное обеспечение, основанное на понятиях и идее относительно воображаемого компьютера, который имеет логический набор команд, и команд, определяющих операции этого компьютера. Это можно сказать, небольшая операционная система. Она формирует необходимый уровень абстракции, где достигается независимость от платформы, используемого оборудования.Компилятор конвертирует исходный текст в код, который основан на воображаемой системе команд компьютеров и не зависит от специфичности процессора. Интерпретатор -приложение, которое понимает эти потоки команд и преобразовывает эти команды для используемого оборудования, к которому относится интерпретатор. JVM создает систему поддержки выполнения внутренне, что помогает выполнению кода призагрузке файлов .class, управлению памятью и выполнении обработки исключений.

Из-за несогласованности аппаратных платформ виртуальная машина использует понятие стека, который содержит следующую информацию:

* Описатели состояния метода
* Операнды к байт-кодам
* Параметрыметодов
* Локальные переменные

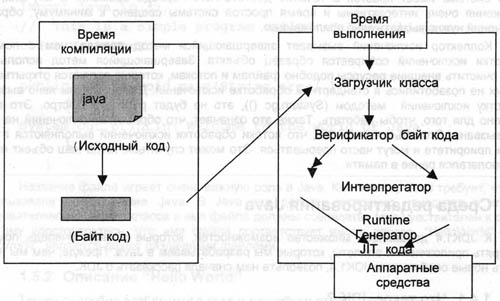
Когда код выполняется с помощью JVM, то существует один специальный регистр, который используется как счетчик, указывая выполняющиеся в настоящее время команды. Если необходимо, команды изменяют программу, изменяют поток выполнения, иначе поток последователен и переходит от одной команды к другой.

Другое понятие, которое становится популярным - это использование Just In Time (JIT) компилятора. Браузеры включают JIT компиляторы, которые увеличивают скорость выполнения кодов Java. Основная цель JIТ состоит в том, чтобы преобразовать систему команд байт-кода к машинным командам кода, целенаправленным для специфического микропроцессора. Эти команды сохраняются и используются всякий раз, когда запрос делается к этому специфическому методу.

JRE (Java Runtime Environment, среда выполнения Java) JVM взаимодействующего с аппаратными средствами на одной стороне и программе на другом. JRE выполняет код, откомпилированный для JVM:

* Загрузка .class файлов выполняется с помощью 'Загрузчика классов',загрузчик класса делает проверку защиты, если файлы используются в сети.
* Проверка байт-кода выполняется 'верификатором байт-кода', верификатор байт-кода проверяет формат кода, преобразования типов объектов и проверяет нарушение прав доступа.
* Выполнение кода, выполняется 'интерпретатором во время выполнения' Интерпретатор выполняет байт-коды и делает запросы на используемое оборудование.

Рис.3 Среда выполнения Java.



В С, C++ или Паскале, приходится использовать методы распределения и освобождения блоков памяти - динамическую память. Динамическая память –большая область памяти, которая обозначена в объёмевсей памяти.Динамическая память используется, как свободный блочный список и распределённый блочный список. Свободный список проверяет блок памяти всякий раз, когда делается запрос. Используется механизм распределения - "метод первого подходящего блока", посредством чего первый наименьший блок памяти распределяется в зависимости от запроса. Эта процедура распределяет и освобождает небольшие объёмы памяти различных размеров от динамической памяти, при этом фрагментация динамической памяти сводится к минимуму. Существует стадия, посредством которой выполняется запрос к памяти - для получения большего блока памяти, чем доступно. В таких случаях программа управления динамической областью должна создать больше памяти. Это процесс, посредством которого все свободные доступные блоки памяти объединяются вместе, перемещая свободную память одному из концов динамической памяти, таким образом, создавая один большой блок памяти, виртуальная машина Java использует две отдельные динамических памяти для статического и динамического распределения памяти.Динамическая память - не делает обработку исключений динамической памяти, которая сохраняет все свойства класса, постоянный пул и таблицы методов. Вторая динамическая память снова разделена на два раздела, которые могут быть расширены в противоположных направлениях когда потребуется. Один раздел используется, чтобы сохранять образцы объектов, а другой раздел используется, чтобы сохранять дескрипторы в эти образцы. Дескриптор - структура, которая состоит из двух указателей. Указываете на таблицу методов объекта и других пунктов к образцу того объекта. Это размещение в основном устраняет потребность сохранения путей, указывающих на объект при модифицировании указателей после уплотнения. Все, что мы должны сделать - это обновить значение указателя дескриптора.Алгоритм обработки исключений применяется к объектам, помещенным в динамическую память. Поскольку запрос о блоке памяти получен, программа управления динамической областью первые проверки свободный список и если программа управления динамической областью не может найти свободные блоки памяти, вызывается обработка исключений, как только система имеет простой в течение достаточного периода времени. В случаях, когда приложения очень интерактивны и время простоя системы сведено к минимуму, обработку исключений нужно вызвать явно приложением.Коллектор исключений вызывает завершающийся метод прежде, чем с помощью обработки исключений собирается образец объекта. Завершающийся метод используется чтобы очистить внешние ресурсы подобно файлам и потокам, которые являются открытыми и о которых не позаботились в стандартной обработке исключений. Даже если явно вызывается обработка исключений методом (System.gc ()), это не будет работать быстро. Это просто намечено для того, чтобы работать. Также это означает, что обработка исключений не может быть вызвана. Это объясняется тем, что потоки обработки исключений выполняются в очень низком приоритете и могут часто прерываться. Это может случиться, когда наш объект никогда не располагался ранее в памяти.

В целом целесообразно указать следующие преимущества Java определяющие выбор для разработки чата:

* переносимость, или кроссплатформенность ;
* объектная ориентированность, создана эффективная объектная модель;
* привычный синтаксис С/С++;
* встроенная и прозрачная модель безопасности;
* ориентация на Internet-задачи, сетевые распределенные приложения;
* динамичность, легкость развития и добавления новых возможностей;
* простота освоения.

Java – это полностью объектно-ориентированный язык, который в отношении синтаксиса многое унаследовал от С++. Конечно, преимущества Java далеко не исчерпываются межплатформенностью. Язык Java в синтаксическом отношении проще и логичнее, чем С++. Java как платформа предоставляет в распоряжение программистов большое количество библиотек (пакетов), в которых содержится большое количество описаний классов и интерфейсов на все случаи жизни. С их помощью можно создавать стопроцентные приложения Java с возможностью обращения к базам данных, поддержкой передачи почтовых сообщений, с клиентской частью, которой необходим web-браузер, или наоборот, с клиентской частью, обладающей изощренным интерфейсом.

Java – это очень элегантный и красивый язык. Однако при его использовании проблем также избежать не удастся. Одна из серьезных проблем заключается в том, что при создании сложного приложения на Java вам придется использовать только этот язык для создания всех частей этого приложения. В Java предусмотрено не так уж много средств для межъязыкового взаимодействия (что понятно ввиду предназначения Java быть единым многоцелевым языком программирования). В реальном мире существуют миллионы строк готового кода, которые хотелось бы интегрировать с новыми приложениями на Java. Однако это сделать очень трудно.

Java – это далеко не идеальный язык во многих ситуациях. Простой пример – если попытаться создать только на Java приложение, активно работающее с 3D – графикой, скорее всего, вы обнаружите, что работать такое приложение будет не очень быстро. Подумав, можно прийти к выводу, что для работы с 3D – графикой лучше использовать код, написанный на языке с более развитыми низкоуровневыми возможностями (например, на С++). Однако интегрировать такой код с кодом на Java очень сложно. Поскольку возможности для обращения к API компонентов, созданных на других языках, в Java очень ограничены, говорить о реальном межъязыковом взаимодействии на основе Java не приходится.

# 1.3 КОЛЛЕКТИВНАЯ РАЗРАБОТКА И УПРАВЛЕНИЕ ПРОЕКТОМ

Одной из главных составляющих, необходимых для производства программного продукта, являются люди. При организации работы программистов важно учитывать не только их технические навыки, но и взаимодействие между ними.

Допустим, нам необходимо разработать программный продукт объемом 50000 строк за 2 года, и средняя [производительность](javascript:function()%7breturn%20false%7d;) одного программиста – 5000 строк в год.

Кажется очевидным, что для разработки достаточно 5 программистов 5\*5000\*2=50000. Но этим программистам необходимо общаться между собой, работа в изоляции может привести к непониманию программистом предъявляемых к нему требований, а это приведет к снижению эффективности работы. Допустим, что каждый из путей взаимодействия уменьшает производительность программиста на 250 строк в год. Тогда реальная скорость каждого из них составит всего 4000 строк в год, и за два года они успеют написать всего 40000 строк. То есть проект будет закончен в два года. Для осуществления проекта потребуется 8 программистов со скоростью 3250 строк в год и руководитель так что на одного программиста придется 3000 строк в год. Добавление еще одного человека уменьшит скорость каждого разработчика на 250 строк, и мы получим10 ∙ 2750 = 27 500 строк.Таким образом, добавление еще одного человека не поможет закончить проект быстрее, последующее добавление людей будет сказываться даже хуже. Так, команда из 13 человек за год сможет написать только26 000 строк и будет менее продуктивной, чем команда из 9 человек. В команде из 13 человек больше половины времени уходит на разговоры, что приводит к снижению эффективности.

На [графике](javascript:function()%7breturn%20false%7d;) представлена приблизительная зависимость эффективности разработчика от количества людей, с которыми он взаимодействует.

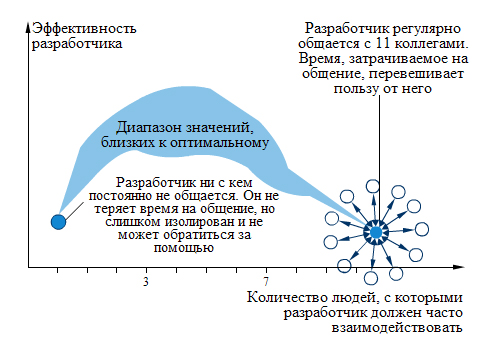


Рис.4 Эффективность разработки

Как следует из примера, рассмотренного на странице «Взаимодействие разработчиков» необходимость взаимодействия значительно снижает производительность труда программиста. Фирма IBM предложила один из путей преодоления этой проблемы – организацию бригад главного программиста. В этой модели организации проектной работы предполагается, что программисты имеют разные уровни компетентности, высококвалифицированные работники должны выполнять наиболее ответственную работу, а остальные – вспомогательную.

В бригаде главного программиста выделяется несколько ролей:

-Главный программист – превосходно подготовленный, творчески мыслящий, наделенный организационными способностями программист, является руководителем бригады. Его производительность может в 5 и более раз превышать производительность некоторых из членов бригады. Он является техническим руководителем проекта, осуществляет разработку системы и построение программных интерфейсов верхнего уровня для всех основных модулей.

-Администраторотвечает за распределение времени, подбор кадров, размещение исполнителей, финансирование, поддержку связи с высшим руководством. Часто несколько бригад программистов «делят» одного администратора. Эта роль появляется, если разрабатываемая система довольно большая.

- Старший программист поддерживает тесную связь с главным программистом и дорабатывает более мелкие вопросы, указанные последним. Старший программист занимает место главного в случае его ухода. При обсуждении проекта старший программист также может играть роль оппонента главному программисту.

-Младший программист. Он занимается разработкой модулей нижнего уровня, определенных главным программистом.

Пусть, как и в примере со страницы «Взаимодействие разработчиков» в команде 5 разработчиков; при этом имеется лишь 7 путей взаимодействия.

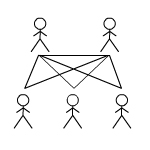


Рис.5 Схема взаимодействий в бригаде главного программиста

Таким образом, общение разработчиков «стоит» 1 750 строк/год, а не 5 000 строк/год, как в структуре на странице «Взаимодействие разработчиков».Кроме того, главный программист, являясь высококвалифицированным специалистом, может перевыполнить план в 5 000 строк/год. Таким образом, средняя скорость каждого разработчика может превысить 5 000 строк/год,тогда как в предыдущем примере составляла всего лишь 4 000 строк/год.Для успешной разработки программного обеспечения недостаточно просто распределить среди участников проектной команды роли и обязанности. Необходимо, чтобы все участники команды понимали и соблюдали следующие базовые принципы коллективной работы:

Единое, разделенное всеми участниками проектной команды понимание целей проекта является важнейшим фактором успеха разработки. На начальных этапах работы каждый участник команды имеет собственное мнение и представление о продукте. Необходимо обсудить частные мнения и по итогам обсуждения выработать концепцию проекта – единый документ, который содержит следующую информацию о продукте проекта и о путях реализации проекта:

* + описание возможностей продукта (что продукт позволяет сделать);
  + конкретизацию продукта (описание функциональных возможностей данной версии);
  + описание путей реализации проекта.

В группе разработчиков важна каждая роль. Только такой подход позволяет наладить свободный обмен информацией между проектными группами и отдельными разработчиками и сформировать общую ответственность за качество продукта проекта.Выпустить продукт в срок удастся только при постоянном формальном и неформальном обучении участников проектной группы. На разных этапах реализации проекта применяются следующие виды обучения:

* обучение на опыте других проектов;
* изучение методологии;
* изучение технологий.

Проектная группа должна эффективно взаимодействовать с внешними группами – с заказчиком, пользователями, другими разработчиками. За эффективность внешних взаимодействий отвечают менеджер проекта, менеджер продукта, специалист по внедрению и логистик. В их обязанности входят как внутренние, так и внешние контакты.

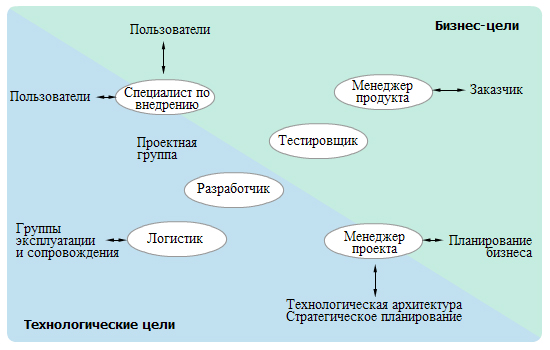


Рис.6 Взаимодействие проектной команды с внешними группами

При разработке коллективом программистов крупного программного проекта возникает проблема эффективной[синхронизации изменений](javascript:function()%7breturn%20false%7d;), параллельно вносимых в проект разными разработчиками и, как следствие, потребность в ПО контроля и объединения версий.Во время разработки любого проекта необходимо [регулярное сохранение](javascript:function()%7breturn%20false%7d;) предыдущих версий файлов (backup) с возможностью быстрого их восстановления. Более того, часто желательно сохранять каждую версию каждого файла, снабжая ее комментарием о том, кто и зачем изменил этот файл. Для этих целей используются системы контроля версий (Version Control Systems).

Также при одновременной разработке нескольких программ часто возникает необходимость использовать в разных проектах одни и те же компоненты исходного кода. При разработке долгосрочного проекта приходится выпускать [модифицированные версии](javascript:function()%7breturn%20false%7d;) программы или даже семейства таких версий – при этом часть исходных файлов, начиная с какого-то момента, изменяется (или наоборот не изменяется) независимо от основной их версии. Поддержка такого ветвления версий (branching) вручную является весьма сложной задачей.

Для решения этих задач применяются средства коллективной разработки (СКР) – специальное ПО, способное объединять различные версии текста программы и вести полную историю изменений сразу нескольких проектов с совместным использованием между ними общих компонентов (sharing) и ветвлением. СКР позволяют каждому программисту, работающему на проекте в условиях нечеткого разграничения ответственности между разработчиками, произвольно корректировать план своих действий во время работы.

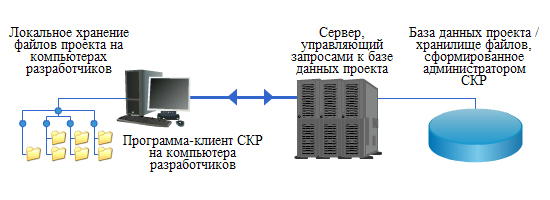


Рис.7 Основные элементы средства коллективной разработки

Работа средств коллективной разработки основана на выполнении двух базовых функций:

Ведение общей для всех разработчиков базы данных проекта, содержащей историю изменений в каждом файле проекта.

Автоматизация следующих операций:

* доступа к общей базе данных;
* обработки [конфликтующих версий](javascript:function()%7breturn%20false%7d;) файла;
* именования различных версий файла;
* ввода и сохранения комментариев к изменениям.

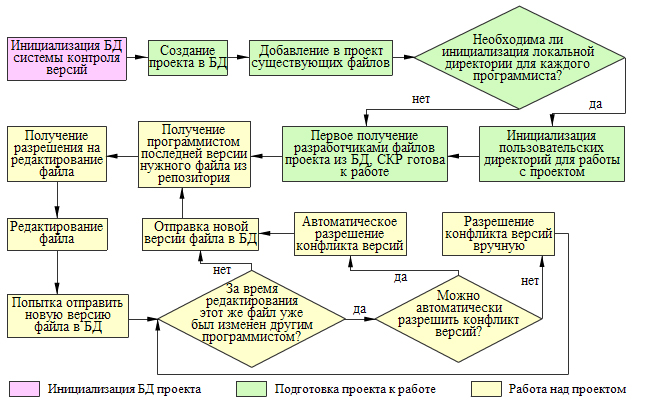


Рис.8 Общая схема работы СКР

Перед началом коллективной работы с проектом администратор проекта создает в СКР базу данных проекта и выдает участникам проекта права доступа к ней. После этого все участники проекта могут получать с сервера [необходимые файлы](javascript:function()%7breturn%20false%7d;) на свой локальный диск по мере надобности.

В процессе работы программист должен регулярно проверять (средствами СКР), какие изменения были внесены в исходный код его коллегами. Внеся в файлы нужные изменения, он должен отправить их обратно на сервер. На этом этапе разработки система коллективной разработки выполняет свои важнейшие функции: находит изменения в файле по сравнению с той его версией, до редактирования, проверяет историю изменений файла (обрабатывая, в случае необходимости, конфликт версий), и предлагает программисту добавить комментарий с описанием внесенных изменений.

Таким образом, проект по разработке ПО, реализуемый с использованием СКР, представляет собой набор директорий и файлов различных типов, находящихся в следующих локациях:

* на компьютерах участников проекта в [выбранных ими папках](javascript:function()%7breturn%20false%7d;);
* в базе данных проекта на сервере.

Типичный рабочий цикл с использованием subversion выглядит примерно так:

* Обновление рабочей копии
  + **svn update**
* Внесение изменений
  + **svn add**
  + **svn delete**
  + **svn copy**
  + **svn move**
* Анализ изменений
  + **svn status**
  + **svn diff**
  + **svn revert**
* Слияние изменений, выполненных другими, с вашей рабочей копией
  + **svn update**
  + **svn resolved**
* Фиксация изменений
  + **svn commit**

### Обновление рабочей копии

При командной работе над проектом обновление рабочей копии необходимо для получения любых изменений, внесенных другими разработчиками проекта с момента вашего последнего обновления. **Svn update** используется для синхронизации вашей рабочей копии с последней правкой в хранилище.

$ svn update

U foo.c

U bar.c

Updated to revision 2.

В данном случае, кто-то другой зафиксировал изменения в файлах foo.c и bar.c после последнего обновления, и Subversion обновила рабочую копию, включив эти изменения. Когда сервер отправляет изменения в рабочую копию, для каждого элемента выводится латинская буква — код, определяющий действие, выполненное Subversion для приведения рабочей копии в актуальное состояние:

U foo

Файл foo был Updated — обновлен (получил изменения с сервера).

A foo

Файл или каталог foo был Added — добавлен в рабочую копию.

D foo

Файл или каталог foo был Deleted — удален из рабочей копии.

R foo

Файл или каталог foo был Replaced — заменен в рабочей копии; это значит, что foo был удален, а новый элемент с таким же именем был добавлен. Несмотря на то, что они могут иметь одинаковое имя, хранилище рассматривает их как разные объекты с отдельной историей.

G foo

Файл foo получил новые изменения из хранилища, однако локальная копия в то же время содержит собственные изменения. Изменения, полученные из хранилища, либо не пересекаются, либо они точно такие же как локальные изменения, поэтому Subversion успешно выполнила merged — слияние изменений хранилища с файлом.

C foo

Файл foo получил от сервера Conflicting — конфликтующие изменения. Изменения с сервера пересекаются с вашими изменениями файла. Такое перекрытие нуждается в ручном разрешении.

### Внесение изменений в рабочую копию.

Как правило, целесообразнее всего работать отдельно по каждому частному изменению (или набору изменений), такому как реализация новой функциональной возможности, исправление ошибки и т. п. Здесь используются такие команды Subversion как **svn add**, **svn delete**,**svn copy** и **svn move**. Однако, если вы просто редактируете файлы, которые уже находятся под контролем Subversion, ни одна из этих команд не нужна. В рабочей копии можно делать следующие изменения:

Изменения файлов:

Subversion может автоматически определить измененные файлы.

Изменения в структуре:

Можнопопросить Subversion «отметить» файлы и каталоги для удаления, добавления, копирования или перемещения. Хотя эти изменения сразу же отразятся в рабочей копии, в хранилище не произойдет ни добавления, ни удаления до тех пор, пока не будет выполнена фиксация изменений.

Для внесения изменений в файлы используется текстовый редактор, текстовый процессор, графическая программа или любой другой инструмент. Subversion обрабатывает бинарные файлы так же как и текстовые. Целесообразно рассмотреть четыре подкоманды Subversion, которые чаще всего используются при внесении изменений в структуру.

### Анализ изменений.

После внесения изменений нужно зафиксировать их в хранилище, но перед этим надо посмотреть изменения. Проанализировав перед фиксацией свои изменения, можно составить более аккуратное лог-сообщение. Кроме того, можно обнаружить, что вы изменили файл непреднамеренно, что позволит еще до фиксации вернуть файл к предыдущему состоянию. Чтобы увидеть все сделанные изменения, можно воспользоваться**svn status**,**svn diff**и **svn revert**. Первые две команды используются для того, чтобы найти измененные файлы рабочей копии, а затем, при помощи третьей, можно отменить некоторые (или все) изменения.Все эти три команды (**svn status**, **svn diff**и **svn revert**) могут использоваться при полном отсутствии сетевого доступа. Это позволяет легко управлять рабочими изменениями, когда нет сетевого соединения. Subversion оптимизирована для решения такой задачи и способна выполнять множество действий без обращения к хранилищу. В частности, рабочая копия содержит в .svn-области скрытую кэшированую «нетронутую» копию каждого версионированного файла. За счет этого Subversion может быстро показать, как изменились рабочие файлы или даже предоставить, не связываясь с хранилищем, возможность откатить изменения.

#### **svn status**

При запуске **svn status** без параметров из корневого каталога рабочей копии будут найдены все сделанные изменения файлов и структуры. **Svn status** печатает пять колонок букв, затем несколько пробелов, затем имя файла или каталога. Первая колонка показывает статус файла или каталога и/или ее содержимого. При этом используются следующие коды:

A item

Файл, каталог или символьная ссылка item был запланирован для добавления в хранилище.

C item

Файл item находится в состоянии конфликта. Это означает, что изменения, полученные от сервера, при обновлении пересекаются с локальными изменениями, имеющимися в рабочей копии. Перед фиксацией изменений вам необходимо разрешить этот конфликт.

D item

Файл, каталог или символьная ссылка item запланирован для удаления из хранилища.

M item

Содержимое файла item было изменено.

R item

Файл, каталог или символьная ссылка запланирован для замены item в хранилище. Это значит, что сначала объект был удален, а затем другой объект с таким же именем был добавлен, все в одной правке.

X item

Каталог item не версионирован, но относится к внешним зависимостям Subversion.

? item

Файл, каталог или символьная ссылка не находится под контролем версий.

! item

Файл, каталог или символьная ссылка item находится под контролем версий, но отсутствует в рабочей копии или поврежден.

~ item

Файл, каталог или символьная ссылка item в хранилище является объектом одного типа, а то, что на самом деле находится в рабочей копии, является чем-то другим.

I item

Файл, каталог или символьная ссылка item находится под контролем версий, и Subversion настроена на его игнорирование при операциях **svn add**, **svn import** и **svn status**.

Вторая колонка показывает статус свойств файлов и каталогов, если во второй колонке показывается M, свойства были изменены. Если в этой колонке показывается C, то это означает, что свойства файла находятся в состоянии конфликта, который должен быть разрешен до фиксации изменений в хранилище. Во всех других случаях будет выведен пробел.

Третья колонка может содержать только пробел или L, это значит, что у каталога заблокирована рабочая область .svn.

Четвертая колонка может содержать только пробел или +, это означает, что элемент был запланирован для «добавления с историей». Это может быть файл или корень скопированного каталога. + означает, что элемент является частью поддерева, запланированного для «добавления с историей», т. е. один из родительских каталогов был скопирован, и этот элемент просто его часть. M  + означает, что элемент является частью поддерева, запланированного для «добавления с историей», и имеет локальные изменения. При выполнении фиксации вначале будет «добавлен с историей» родительский каталог, что означает автоматическое наличие файла в копии. После этого в копию будут загружены локальные изменения.

Пятая колонка может содержать только пробел или S. Это означает, что файл или каталог был переключен с пути остальной рабочей копии на ветку (используя **svn switch**).

Шестая колонка показывает информацию о блокировках.

Если указать конкретный путь для **svn status**, то будет получена информация только об этом элементе:

$ svn status stuff/fish.c

D stuff/fish.c

Кроме того, **svn status** имеет параметр --verbose (-v), который показывает статус каждого элемента в рабочей копии, даже если он не менялся:

$ svn status --verbose

M 44 23 sally README

44 30 sally INSTALL

M 44 20 harry bar.c

44 18 ira stuff

44 35 harry stuff/trout.c

D 44 19 ira stuff/fish.c

44 21 sally stuff/things

A 0 ? ? stuff/things/bloo.h

44 36 harry stuff/things/gloo.c

Первая колонка осталась та же самая, а вот вторая колонка показывает рабочую правку элемента. Третья и четвертая колонки показывают правку, в которой элемент последний раз изменялся и автора этих изменений.

Ни один из указанных выше вызовов **svn status** не обращается к хранилищу, они работают только локально, сравнивая метаданные каталога .svn с рабочей копией. Также есть параметр --show-updates (-u), указывающий на соединение с хранилищем и добавляющий информацию об устаревании элементов:

$ svn status --show-updates --verbose

M \* 44 23 sally README

M 44 20 harry bar.c

\* 44 35 harry stuff/trout.c

D 44 19 ira stuff/fish.c

A 0 ? ? stuff/things/bloo.h

Statusagainstrevision: 46

### Разрешение конфликтов (при слиянии с чужими изменениями)

Например, при запуске **svn update** отображается следущее сообщение:

$ svn update

U INSTALL

G README

C bar.c

Updated to revision 46.

Файлы, отмеченные C, имеют конфликт. Это значит, что изменения с сервера пересеклись с локальными, и теперь нужно вручную сделать между ними выбор.

Всякий раз, когда возникает конфликт, в его обнаружении и разрешении целесообразно учитывать следущее:

* Subversion печатает C во время обновления и запоминает, что файл в состоянии конфликта.
* Если Subversion считает, что тип файла допускает слияние изменений, она включает в него маркеры конфликта — специальные текстовые строки, отделяющие «стороны» конфликта — чтобы визуально показать пересекающиеся области.
* Для каждого конфликтного файла Subversion добавляет в рабочую копию до трех не версионированных дополнительных файлов:

filename.mine

Это ваш файл в том виде, в каком он присутствовал в рабочей копии до обновления — без маркеров конфликта. Этот файл содержит в себе только ваши изменения и ничего больше. (Если Subversion решает, что файл не пригоден для слияния изменений, то файл .mine не создается, так как он будет идентичным рабочему файлу.)

filename.rOLDREV

Это файл правки BASE, где BASE — правка, которая была до обновления рабочей копии. Иными словами, это файл, который был у вас до внесения изменений.

filename.rNEWREV

Это файл, который ваш Subversion-клиент получил с сервера при обновлении рабочей копии. Этот файл соответствует правке HEAD хранилища.

Здесь OLDREV — это номер правки файла в каталоге .svn, а NEWREV — номер правки HEAD хранилища.

Например, один пользователь внес изменения в файл sandwich.txt из хранилища. Одновременно второй пользователь изменил файл в своей рабочей копии и зафиксировал его. Первый пользователь обновляет свою рабочую копию перед фиксацией и получает конфликт:

$ svn update

C sandwich.txt

Updated to revision 2.

$ ls -1

sandwich.txt

sandwich.txt.mine

sandwich.txt.r1

sandwich.txt.r2

Теперь Subversion не позволит зафиксировать файл sandwich.txt, пока не будут удалены три временных файла.

$ svn commit --message "Add a few more things"

svn: Commit failed (details follow):

svn: Aborting commit: '/home/sally/svn-work/sandwich.txt' remains in conflict

Для разрешения конфликта есть три варианта:

* Объединить конфликтующий текст «вручную» (путем анализа и редактирования маркеров конфликта в файле).
* Скопировать один из временных файлов поверх своего рабочего файла.
* Выполнить **svn revert <filename>** для отказа от всех локальных изменений.

После разрешения конфликта вам нужно известить об этом Subversion, выполнив **svn resolved**. Эта команда удалит три временных файла, и Subversion больше не будет считать, что файл находится в состоянии конфликта.

$ svn resolved sandwich.txt

Resolved conflicted state of 'sandwich.txt'

#### Слияние конфликтов вручную

Например два разработчика независимо одновременно редактируют файл sandwich.txt. Первый зафиксировал свои изменения, и поэтому при обновлении своей рабочей копии второй получит конфликт, для разрешения которого ему необходимо отредактировать sandwich.txt.

$ cat sandwich.txt

Top piece of bread

Mayonnaise

Lettuce

Tomato

Provolone

<<<<<<< .mine

Salami

Mortadella

Prosciutto

=======

Sauerkraut

Grilled Chicken

>>>>>>> .r2

Creole Mustard

Bottom piece of bread

Строки, начинающиеся со знаком «меньше чем», «равно» и «больше чем», являются маркерами конфликта. Перед следующей фиксацией нужно будет убедиться, что они удалены из файла. Текст между первыми двумя маркерами состоит из ваших изменений в конфликтующей области:

<<<<<<< .mine

Salami

Mortadella

Prosciutto

=======

Текст между вторым и третьим маркером конфликта — это текст из фиксации первого разработчика:

=======

Sauerkraut

Grilled Chicken

>>>>>>> .r2

В данном случае придется в рабочем порядке разработчикам согласовывать редакцию изменений. После того, как общая редакция изменений будет согласована, нужно будет выполнить фиксацию. Для этого отредактируется файл и удаляются маркеры конфликта.

Toppieceofbread

Mayonnaise

Lettuce

Tomato

Provolone

Salami

Mortadella

Prosciutto

Creole Mustard

Bottom piece of bread

Теперь выполняется **svn resolved**, и фиксируются изменения:

$ svn resolved sandwich.txt

$ svn commit -m "Go ahead and use my sandwich, discarding Sally's edits."

#### Копирование файла поверх вашего рабочего файла

Если при получении конфликта решено отказаться от своих изменений, можно скопировать один из временных файлов, созданных Subversion, поверх файла в рабочей копии:

$ svn update

C sandwich.txt

Updated to revision 2.

$ ls sandwich.\*

sandwich.txt sandwich.txt.mine sandwich.txt.r2 sandwich.txt.r1

$ cp sandwich.txt.r2 sandwich.txt

$ svn resolved sandwich.txt

### Фиксацияизменений

Команда **svn commit** отправляет изменения в хранилище. При фиксации изменений необходимо описать ваши изменения в тексте лог-сообщения. Лог-сообщениебудетприсоединеноксозданнойправке.

$ svn commit --message "Corrected number of cheese slices."

Sending sandwich.txt

Transmitting file data .

Committed revision 3.

Можно попросить Subversion взять лог-сообщение из файла, передав имя этого файла в параметре --file:

$ svn commit --file logmsg

Sending sandwich.txt

Transmitting file data .

Committed revision 4.

В целом отмечается что в Subversion существует много других возможностей, которые можно применять для управления рабочей копией и хранилищем. Следует отметить что ничего полностью из хранилища не удаляется удаление касается только HEAD правки хранилища. Можновосстановить все, что удалено, создав рабочую копию или обновив существующую.

# Вычислительная часть.

# 2.1 Основные сведения о программе чат и концепции программы

В основе консольной программы Чат применена двухуровневая архитектура «Клиент-Сервер». Сервер обрабатывает запросы клиента и посылает ему текст ответа. Используется циркуляр сообщений, который позволяет передавать сообщения всем пользователям чата, или вести приватную переписку между клиентами без отображения информации в консоли других пользователей чата.

Клиент получает текст ответа и распечатывает его на консоль, предназначенную для вывода сообщений. Во вторую консоль клиента пользователь вводит текст сообщений.

Чат предусматривает регистрацию пользователя в сети чата, проверку уникальности выбранного им для регистрации ника, выбор пользователем тематического раздела чата для общения с другими пользователями по интересам, а также просмотра истории своей приватной переписки с другими пользователями.

Заказчиком были предъявлены следующие требования:

- Реализовать обработку сообщений пользователей, журналирование их переписки и сохранение в базе данных частной переписки пользователей.

-Разработать несколько тематических разделов, в которые пользователи могут входить по интересам для общения.

-Пользователь чата должен иметь возможность переписки с другими пользователями и отправки циркулярного сообщения всем подключенным пользователям.

-Пользователь должен иметь возможность просмотра своей прошлой частной переписки с другими пользователями.

-При разработке приложения применить автоматические тесты.

Приложение разработано на платформе Java 7 и предназначено для выполнения в средах Windows/Unix. Приложение разработано в open source интегрированной среде разработки модульных кроссплатформенных приложений Eclipse Juno с использованием системы контроля версий SVN репозитория Google Code–сервиса для разработчиков open source программного обеспечения. При разработке чата использовался фреймворк для автоматизации сборки Apache Maven, библиотека журналирования Java программ Apache Log4j 1.12.7, библиотека модульного тестирования Junit 4.10, фреймворк mock-тестирования mockito 1.8.2. В приложение встроена реляционная СУБД Apache Derby версии 10.9.1.0. Для оценки качества кода использована платформа SonarCube версии 3.6.2. Программа содержит 1348 строк кода, 8 пакетов, 27 классов, 91 метод.

# 2.2Архитектура приложения

**2.2.1 РАБОТА СЕРВЕРНОЙ ЧАСТИ ЧАТА.**

Двухуровневая архитектура «клиент-сервер» реализована за счет бесконечного цикла сервера ServerHandler.class в котором серверный сокет сервера слушает порт 2004 и при получении соединения создает и запускает в новом потоке клиента сервера - ServerThread.class передавая ему ссылку на себя.

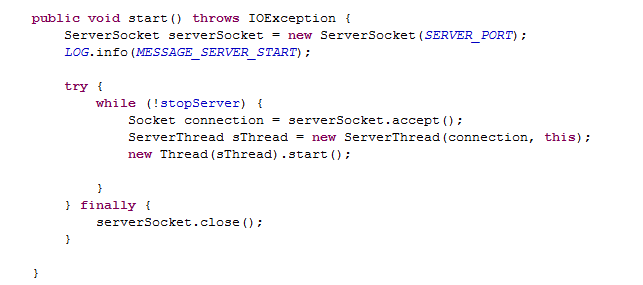


Рис.9ЦиклсервераServerHandler.class.

Данный класс выполняющийся для каждого клиента в отдельном потоке осуществляет все управление клиентским подключением, т.е. обрабатывает все действия клиента, такие как регистрация, проверка уникальности ника, получение, обработка и отсылка сообщений, журналирование сервера и другие. Данный поток выполняется до тех пор, пока клиент не направляет серверу сообщение «bye»- признак выхода из чата, при этом поток безопасно завершает свое выполнение и останавливается.

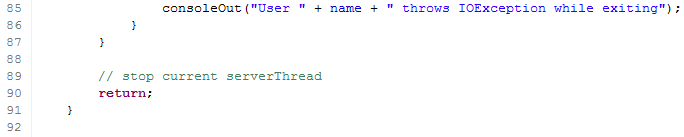


Рис.10 Метод run выполнения потока клиента.

Для управления подключениями клиентов и обеспечения подключения их к различным разделам чата сервер создает карту карт, т.е. коллекциюMap у которой ключом является Stringс названием раздела чата, например «Auto-room» а значением потокобезопасная карта ManualConcurrentHashMap.class в которой в свою очередь ключом является имя клиента т.е. проверенный на уникальность ник введенный им при регистрации в чате, а значением его текущее подключение- ServerThread.class т.е. запущенный при его подключении поток.

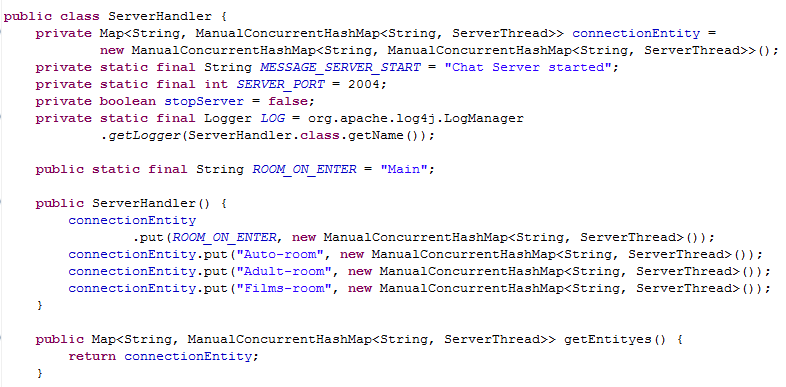


Рис.11Создание сервером карты карт подключений клиентов

Потокобезопасная карта ManualConcurrentHashMap.class унаследована от HashMap обеспечивает потокобезопасное управление потоками клиентов, такими как вставка в коллекцию, удаление из нее потоков, а также чтение коллекции за счет переопределения соответствующих методов родителя и применения механизма локов «lock» обеспечивающего целостность данных.

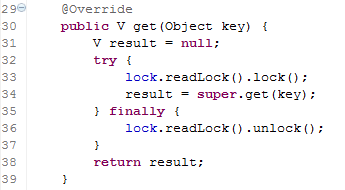


Рис.12 Переопределение метода get().

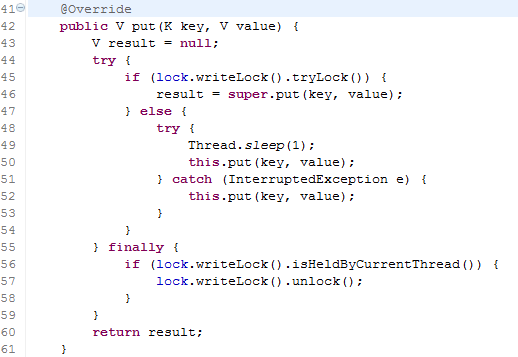


Рис.13 Переопределение метода put().

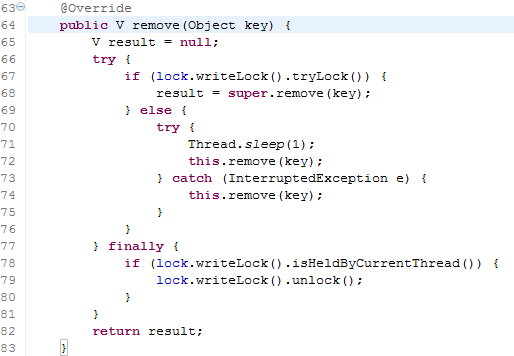


Рис.14 Переопределение метода remove().

Для реализации механизма выполнения запросов клиентов на перемещение по разделам чата, получения справки, просмотра истории переписки или получения списка доступных разделов чата в приложении применен класс CommandExecuter.class. Данный класс выполняет функцию «коммутатора» клиентских команд. Он реализует паттерн одиночка т.е. создается один раз первым вызвавшем его создание потоком клиента и паттерн команда для возвращения потоку сервера целевой команды пользователя. Паттерн команда реализован за чет возвращения серверу ссылки на соответствующий класс команды реализующего интерфейс ICommand. Соответствующий класс команды CommandExecuter.class находит в предварительно созданной коллекции HashMap, в которой ключом является команда String пользователя набранная им в консоли чата в соответствии с правилами чата, а значением предопределенно созданные классы команды выполняющие запросы по своей принадлежности.



Рис.15 Создание коллекции команд для обработки запросов клиентов

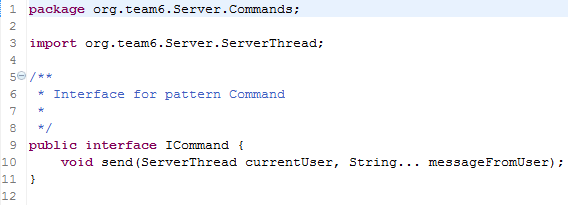


Рис.16 Интерфейс Command для реализации команды.

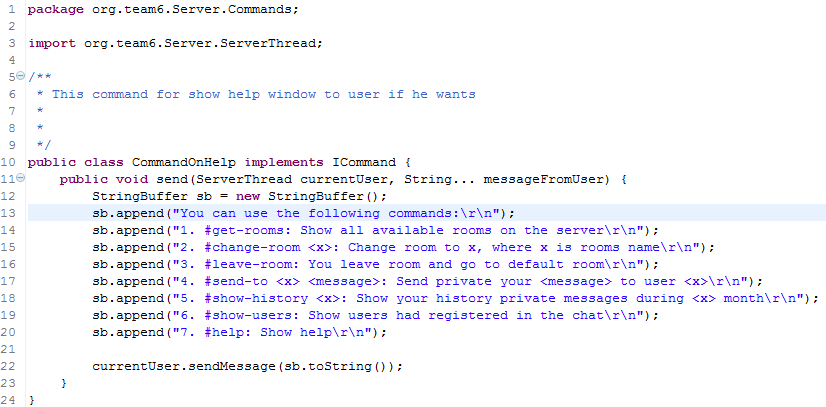


Рис.17 Пример класса команды

Для парсинга клиентского сообщения и выявления целевого значения перечисления CommandExecuter.class. создает одиночку CommandParser.class который сравнивает команды пользователя с предопределенными значениями String и возвращает значение перечисления.

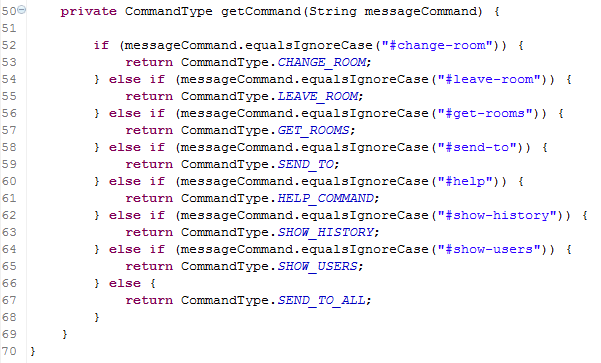


Рис.18 Работа CommandParser.class

Для выполнения требований заказчика о сохранении частной переписки пользователя в приложение встроена реляционная СУБД derby, подключение к базе данных осуществляет класс HistoryDB.class, через JDBC драйвер. База данных содержит две таблицы – таблицу пользователей и таблицу истории переписки каждого пользователя. СУБД запускается перед запуском сервера и прослушивает порт 1527.

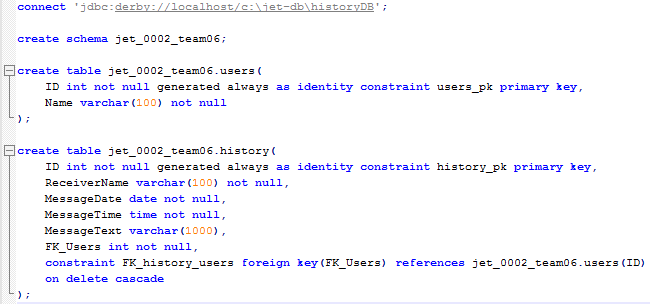


Рис.19 Скрипт создания схемы базы данных.

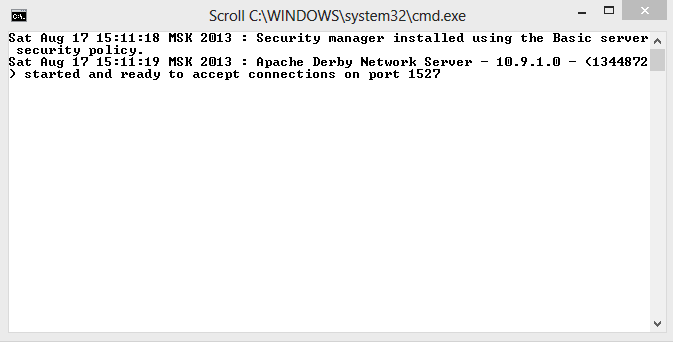


Рис.20 Работа СУБД derby.

**2.2.2 РАБОТА КЛИЕНТСКОЙ ЧАСТИ ЧАТА**

Основная задача клиентской части чата – это уверенно «снимать» с клиентского сокета поступающие с внешнего потока серверные сообщения и отображать их пользователю. А также обеспечить клиенту возможность писать и отправлять сообщения в соответствии с заданием заказчика. На стороне клиента ClientApp.class запускаются две консоли – одна для отправки сообщений и команд пользователя чата SenderApp.class прослушивающий сокет по порту сервера, вторая для получения сообщения с сервера (абонентских сообщений) ReceiverApp.class.

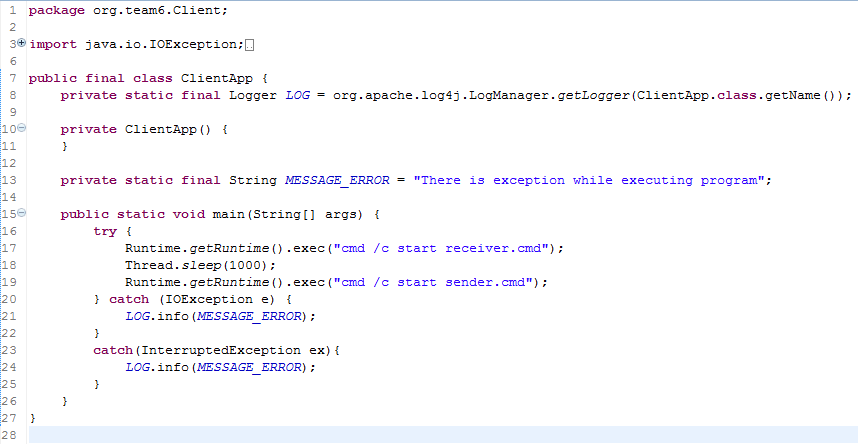


Рис.21 Запуск консолей на клиентской стороне.



Рис.22ЗапускReceiverApp.class.



Рис.23ЗапускSenderApp.class.

При этом работа указанных классов запуск соответствующих классов обработки входящих сообщений сервера и исходящих сообщений пользователя – MessagingReceiver.classи MessaingSender.class в отдельных потоках. SenderApp.class также запускает в отдельном потоке RedirectMessagingFromServer.class для перенаправления в соответствующую консоль входящих сообщений сервера по соответствующему порту.

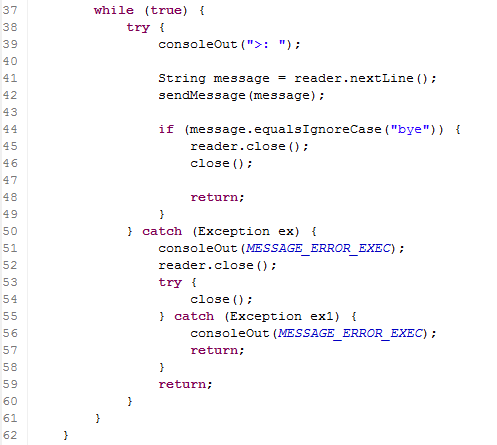


Рис.24 Работа MessagingSender.class



Рис.25 Работа MessagingReceiver.class



Рис.26 Работа RedirectMessagingFromServer.class

# Выводы

В целом по представленной реализации многопоточного пользовательского чата отмечается, что такое решение сервера и клиентов позволяет надежно управлять клиентскими подключениями, обрабатывать их команды и перенаправлять их переписку в соответствии с требованиями заказчика.

Преимуществами данной разработки представляются использование современных технологий системы коллективной разработки (репозиторий Google code с системой контроля версий SVN), технологии автоматической сборки проекта Apache Maven, применение востребованных сейчас библиотек логирования и тестирования Log4j, JUnit, Mockito, а также платформы SonarQube для оценки кода, работа которой приведена ниже. Данные организационно-технические мероприятия примененные в разработке позволили более эффективно выполнить требования заказчика, оптимизировать затраты на разработку и объективно оценить качество работы.

Целесообразно отметить эффективное архитектурное решение по реализации тематических разделов сайта, а именно за счет применения «хэш карты хэш карт», данное решение применено исходя из уникальности названия тематического раздела чата при том, что таких разделов не может быть сколько ни будь много (ключ String–название раздела) и уникальности ника клиента (ключ Stringв вложенной карте). Правда, такое решение потребовало реализации потокобезопасной хэш карты подключений. Зато позволило быстро и просто управлять перемещением клиентов по тематическим разделам чата, входом их в чат и выходом из чата путем удаления и вставки в соответствующую хэш карту нужного потока клиента и просмотра значений хэш карт по соответствующим ключам. Также представляется не лишним отметить эффективный механизм обработки команд пользователя с применением паттерна команды.

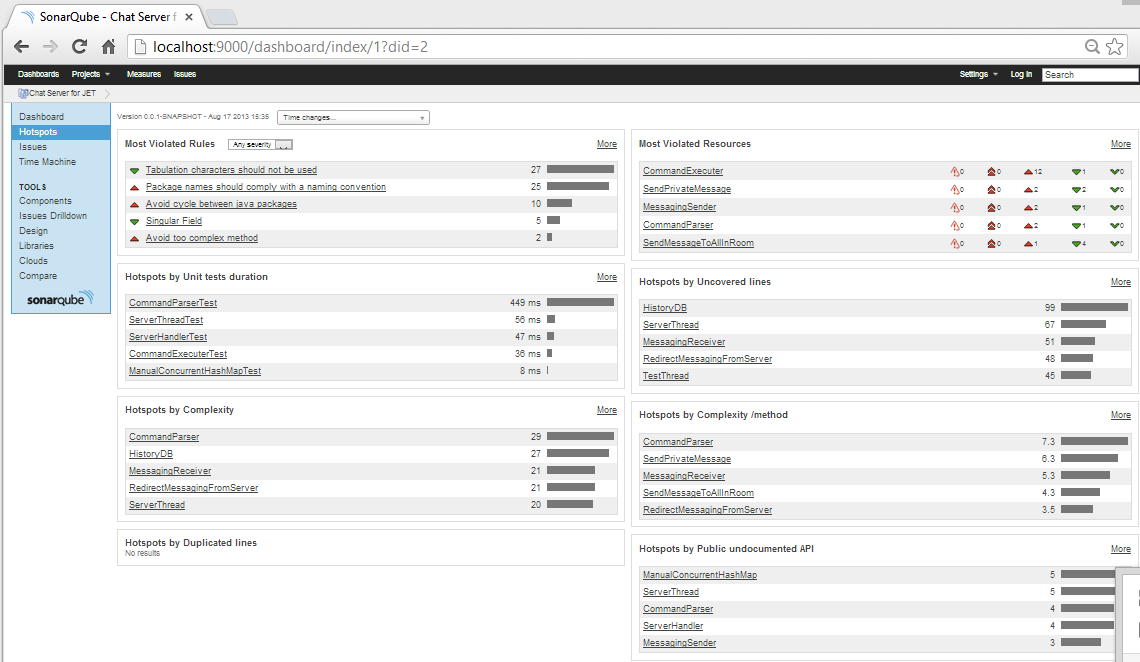
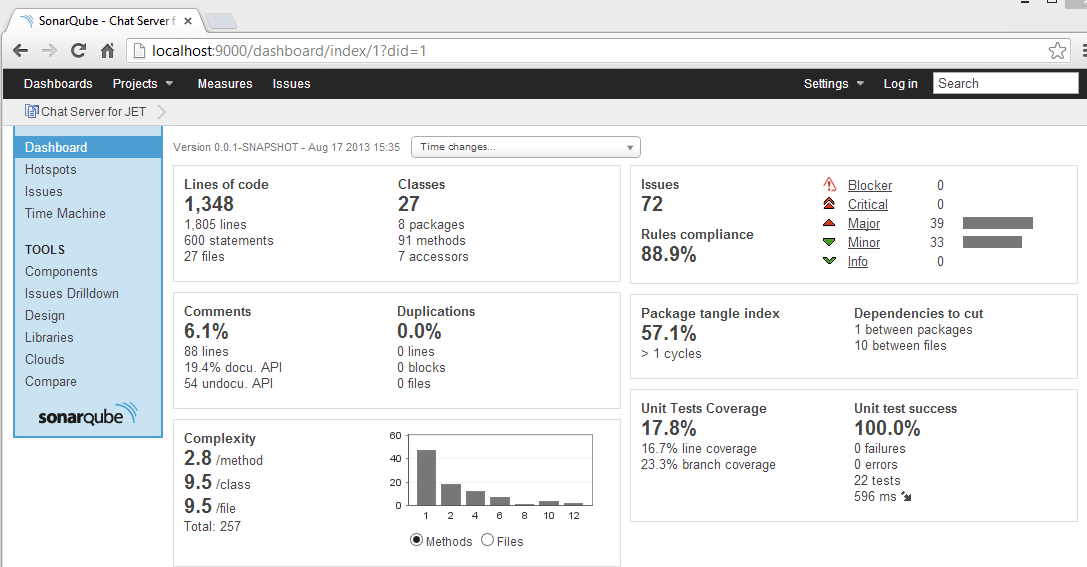


Рис.27 Оценка кода платформой SonarQube.

Вместе с тем необходимо указать, что данная архитектура не лишена недостатков, а именно при подключении значительного количества клиентов управление соответствующим числом потоков может быть сильно затруднено, а в крайнем случае сорвано. Однако для средних масштабов систем обработки сообщений, в рамках текущих требований заказчика решение представляется оптимальным.

**ЗАКЛЮЧЕНИЕ**

В данной работе посвященной исследованию систем обмена сообщениями, приведен их общий обзор, затронута история их создания и отмечены инструменты для их разработки. Приведены организационные и организационно-технические мероприятия коллективной разработки таких систем, что неминуемо, учитывая размер таких приложений и время, затрачиваемое на их разработку.

Непосредственно разработано приложение – многопоточный пользовательский чат на платформе Java. Процесс разработки от проектирования облика чата, продумывания и принятия архитектурных решений до непосредственно кодирования принятых решений безусловно показал преимущества выбранной платформы Java для создания таких приложений. Эти преимущества выражаются не только в продуманности платформы Java, ориентирования ее для сетевого программирования и наличия в ней большого количества удобных библиотек, но и присутствием в большом количестве сторонних технологий которые повсеместно применяются в разработках на Java. Такие, вышеуказанные в данной работе технологии применены и в работе над чатом. К тому же отмечается что данные технологии, включая среду разработки на которой программировался чат, являются open source, что важно, например для программиста «freelancer». Указанные преимущества работают на признание платформы Java как предпочтительной для использования в таких проектах.

Необходимо отметить, что есть и другие платформы, например NET от Microsoft, которые также позволяют работать над такими приложениями и у которых есть свои преимущества (их рассмотрение лежит вне рамок данной работы).

Тем не менее, в данной работе, предпочтение платформы Java другим инструментам оправдало заявленные ее разработчиками преимущества для разработки и обеспечило выполнение заявленных требований заказчика.

**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

1. Мартин Фаулер «Архитектура корпоративных программных приложений» Издательский дом «Вильямс» 2003г.
2. Анил Хемраджани «Гибкая разработка приложений на Java с помощью Spring, Hibernate и Eclipse. Издательский дом «Вильямс» 2008г.
3. Герберт Шилдт «Java. Полное руководство» 8-е издание. Издательский дом «Вильямс» 2012г.
4. Ильдар Хабибуллин «Создание распределенных приложений на Java2». Издательство «БХВ-Петербург» 2002г.
5. [http://www.java.com](http://www.java.com/) – официальный сайт Java.
6. [http://www.apache-maven.ru](http://www.apache-maven.ru/) – официальный сайт Apache Maven Project.
7. [http://www.sonarqube.org](http://www.sonarqube.org/) – официальный сайт технологии SonarQube.
8. [http://fcior.edu.ru](http://fcior.edu.ru/) - проект федерального центра информационно-образовательных ресурсов (ФЦИОР) Минобразования РФ.
9. [http://elis.dvo.ru](http://elis.dvo.ru/)- официальный сайт ДВО РАН.
10. <http://www.intuit.ru> – официальный сайт проекта национального исследовательского университета «Интуит».