1. **ОБЩИЕ СВЕДЕНИЯ ОБ КОМПАНИИ ИООО «ЭПАМ СИСТЕМЗ»**
   1. **О компании**

Компания ***EPAM*** – ведущий разработчик заказного программного обеспечения и один из ведущих игроков в области консалтинга в Центральной и Восточной Европе, на настоящий момент она имеет филиалы более чем в 20 странах мира и продолжает стабильный рост.

*EPAM Systems* - крупнейший поставщик услуг в области разработки проектного (заказного) программного обеспечения и решений в Центральной и Восточной Европе. Реализовав тысячи сложных и масштабных проектов по всему миру, *EPAM Systems* неизменно остается признанным лидером в таких областях, как: разработка, тестирование, сопровождение и поддержка заказного программного обеспечения и бизнес-приложения; интеграция приложений на базе продуктов *SAP*, *Oracle*, *IBM*, *Microsoft*; миграция приложений на новую интеграционную платформу; создание выделенных центров разработки (центров компетенции), центров тестирования и контроля качества программного обеспечения.

Преимуществами сотрудничества с *EPAM Systems* пользуются сотни компаний из различных секторов экономики, в том числе банки и финансовые компании, страхование, поставщики программного обеспечения, розничная торговля и потребительские товары, информационный и медиа-бизнес, индустрия путешествий, телекоммуникации, нефтегаз и энергетика, автобизнес, государственный сектор и др.

Сильная команда талантливых разработчиков, богатый опыт работы с заказчиками по всему миру, партнерство с ведущими разработчиками программного обеспечения, выстроенная методология и наличие современных инструментов проектного управления, глубокая отраслевая и технологическая экспертиза являются неотъемлемыми составляющими успеха ИТ-проектов, реализуемых *EPAM* *Systems*. Для того чтобы лучше понимать и эффективнее решать задачи клиентов, в компании формируются центры компетенции, где консолидируются лучшие мировые практики и отечественный опыт по различным отраслям и технологиям. В целях обеспечения прозрачности и эффективности управления проектами в компании применяются не имеющие аналогов на рынке инструменты и системы метрик собственной разработки. В зависимости от задач проекта и предпочтений заказчика *EPAM* предлагает различную методологию для разработки программного обеспечения.

* 1. **История развития компании**

Компания *EPAM* была основана в 1993 году в Принстоне (штат Нью-Джерси) двумя одноклассниками Аркадием Добкиным (он ранее эмигрировал в США и работал в компании SAP) и Леонидом Лознером. Название компании происходило от «*Effective Programming for America*». Первые офисы были открыты в США и Беларуси. Позже были открыты центральный североамериканский офис в Лоренсвилле, США, штат Нью-Джерси и центральный европейский офис в Будапеште, Венгрия, а также офисы по обслуживанию клиентов в Великобритании, Германии, России, Казахстане, Украине, Польше, Швеции, Швейцарии, Болгарии и Беларуси.

В марте 2004 года *EPAM* приобрела компанию *Fathom* *Technology* в Венгрии, а в сентябре 2006 *VDI* в России, образовав единую компанию под именем *EPAM* *Systems* со штатом сотрудников в 2200 человек.

В 2012 году компания совершает ряд приобретений на северо-американском рынке, в числе которых канадская компания *Thoughtcorp* и крупный поставщик услуг по разработке цифровых стратегий и организации мультиканального взаимодействия *Empathy* *Lab*.

В 2014 году *EPAM* приобрела китайскую ИТ-компанию *Jointech* (*Joint* *Technology* *Development* *Limited*), за счет чего, как следует из пресс-релиза *EPAM*, расширила свои возможности в Азиатско-Тихоокеанском регионе и купила американского поставщика услуг для здравоохранения и медико-биологического сектора *GGA* *Software* *Services*.

В 2015 *EPAM* *Systems* поглотил американские компании: *Navigation* *Arts*, специализирующуюся на цифровом консалтинге и дизайне, а также *Alliance* *Global* *Services*, которая специализируется на выпуске ПО и решений для автоматизированного тестирования. В связи с этим приобретением руководство *EPAM* *Systems* пересмотрело прогноз по выручке в сторону увеличения, ожидая её на уровне не ниже 905 млн долларов в 2015 году против 730 млн годом ранее.

В 2016 году *EPAM* поглотила китайскую компанию *Dextrys*, со штатом в 1400 сотрудников

**1.3 Заказчики компании и предоставляемые услуги**

*EPAM* работает с крупнейшими международными заказчиками, что позволяет приобрести опыт разработки и внедрения передовых решений для знаковых компаний глобального масштаба, получить уникальные знания об устройстве бизнеса ведущих представителей мировой экономики.

*EPAM* обладает обширным списком престижных клиентов по всему миру, среди которых многие члены списка *Fortune* 500 и ведущие компании-разработчики программного обеспечения. Основными заказчиками компании являются: *SAP*, *Microsoft*, *Oracle*, *Thomson* *Reuters*, *The* *Coca*-*Cola* *Company*, Лондонская фондовая биржа, ММВБ, *Viacom*/*MTV* *Networks*, *Wolters* *Kluwer*, «Ренессанс-Капитал», Американский Красный Крест, *Clarins* *Group* и многие другие.

К основным партнерам фирмы можно отнести *SAP*, *Microsoft*, *Oracle*, *Thomson* *Reuters*, *Salesforce*, *IBM*, *Datalex*, *Adobe*, *Dell*, *ADAM*, *Amazon* *Web* *Services*.

EPAM предоставляет широкий спектр услуг и продуктв: ИТ-аутсорсинг, разработка программных продуктов, разработка приложений, создание платформ для корпоративных приложений, тестирование приложений, поддержка и сопровождение приложений, *ASM* и инфраструктурные сервисы, *Business* *Intelligence*, *SAP*-технологии, электронная коммерция, порталы и контент-менеджмент, облачные технологии, мобильные решения, разработка цифровых стратегий, дизайн и проектирование, разработка технологических решений, бизнес-анализ, встроенные решения.

Сегодня *EPAM* объединяет более 13 200 высококвалифицированных ИТ-профессионалов: программистов, руководителей проектов и бизнес-аналитиков, специалистов по обеспечению качества программных продуктов, архитекторов программного обеспечения, переводчиков и дизайнеров.

Компания располагает всеми возможностями для предоставления интересной, перспективной и стабильной работы как опытным профессионалам, так и начинающим специалистам.

**1.4 Работа в компании**

Сегодня *EPAM* объединяет более 13 200 высококвалифицированных ИТ-профессионалов: программистов, руководителей проектов и бизнес-аналитиков, специалистов по обеспечению качества программных продуктов, архитекторов программного обеспечения, переводчиков и дизайнеров.

Компания располагает всеми возможностями для предоставления интересной, перспективной и стабильной работы как опытным профессионалам, так и начинающим специалистам.

В *EPAM* созданы условия для возможности карьерного роста в направлениях технической экспертизы, управления проектами, построения архитектуры сложных систем, консалтинга и управления командами и отделами разработки, тестирования и поддержки. В процессе работы практикуются командировки к крупным зарубежным заказчикам.

В компании успешно реализуется *Career* *Development* *Program* – последовательный процесс обучения молодых специалистов, позволяющий закрепить теорию практикой в предметной области. Участие в *CDP* создает возможность для ускоренного приобретения необходимого опыта и построения успешной карьеры.

## **1.5 Используемое оборудование**

Отдел ИТ оснащен современным оборудованием, а также всеми сопутствующими необходимыми аксессуарами. Типичное рабочее место инженера-программиста состоит из следующих частей:

1. персональный компьютер на базе процессора *AMD Athlon II X4*;
2. монитор 24” (по желанию работника могут быть установлены дополнительные мониторы);
3. клавиатура, мышь, наушники (выбирается работником исходя из его личных предпочтений).

Кроме этого, офис оснащён проводной и беспроводной (*WiFi*) сетью, объединяющей все компьютеры компании в рамках офиса. Данные сети, также, предоставляет доступ к сети Интернет.

## **1.6 Охрана труда на предприятии и техника безопасности на рабочем месте программиста.**

Охрана труда – система обеспечения безопасности жизни и здоровья работающих в процессе трудовой деятельности, включающая правовые, социально-экономические, организационные, технические, психофизиологические, санитарно-гигиенические, лечебно-профилактические, реабилитационные и иные мероприятия и средства [1].

Руководством завода уделяется большое внимание улучшению эргономики рабочих мест, обеспечению гигиены и совершенствованию организации труда, регламентации режимов труда и отдыха.

Вся деятельность в области охраны труда на ИООО «*EPAM Systems*» регламентирована действующим законодательством Республики Беларусь, санитарными нормами и правилами, гигиеническими нормативами, предписаниями надзорных органов.

К комплексу мероприятий в области охраны труда, осуществляемых на ИООО «*EPAM Systems*» относятся:

– проведение производственного лабораторного контроля за условиями труда на рабочих местах;

– разработка инструкций по охране труда и ознакомление с ними персонала;

– модернизация рабочих мест и технологического оборудования;

– создание безопасных условий труда.

Важным фактором создания безопасных условий труда является оптимизация организации рабочих мест. Рабочее место, хорошо приспособленное к трудовой деятельности инженера, правильно и целесообразно организованное, в отношении пространства, формы, размера обеспечивает ему удобное положение при работе и высокую производительность труда при наименьшем физическом и психическом напряжении.

При правильной организации рабочего места производительность труда инженера возрастает с 8 до 20 процентов.

Основным рабочим местом инженера-программиста является стол для выполнения машинописных работ. Основная поза при выполнении работы – вынужденная, сидячая.

Рабочее место для выполнения работ в положении сидя организуется в соответствии с ГОСТ 12.2.032-78 «Система стандартов безопасности труда. Рабочее место при выполнении работ сидя. Общие эргономические требования» [2].

Требования к организации работы при использовании персонального компьютера и организационных средств определяются СанПиН 9-131 РБ 2000 «Гигиенические требования к видеодисплейным терминалам, электронно-вычислительным машинам и организации работы».

Согласно ГОСТ 12.2.032-78 конструкция рабочего места и взаимное расположение всех его элементов должно соответствовать антропометрическим, физическим и психологическим требованиям. Большое значение имеет также характер работы. В частности, при организации рабочего места программиста должны быть соблюдены следующие основные условия:

* оптимальное размещение оборудования, входящего в состав рабочего места;
* достаточное рабочее пространство, позволяющее осуществлять все необходимые движения и перемещения;
* уровень акустического шума не должен превышать допустимого значения.

Рабочая поза сидя вызывает минимальное утомление программиста. Рациональная планировка рабочего места предусматривает четкий порядок и постоянство размещения предметов, средств труда и документации. То, что требуется для выполнения работ чаще, расположено в зоне легкой досягаемости рабочего пространства.

Помещения для работы программиста должны иметь естественное и искусст­венное освещение.

Площадь на одно рабочее место с видео-дисплейным терминалом (ВДТ) и ПЭВМ для взрослых поль­зователей должна составлять не менее 6,0 м2, а объем не менее 20,0 м3.

Искусственное освещение в помещениях эксплуатации ВДТ и ПЭВМ должно осуществляться системой общего равномерного освещения. В административно-общественных помещениях, в случаях преимуществен­ной работы с документами, допускается применение системы комбинированного освещения (к общему освещению дополнительно устанавливаются светильники местного освещения, предназначенные для освещения зоны расположения доку­ментов).

Освещенность на поверхности стола в зоне размещения рабочего докумен­та должна быть 300-500 лк. Местное освещение не должно создавать бликов на по­верхности экрана и увеличивать освещенность экрана более 300 лк.

В качестве источников света при искусственном освещении должны при­меняться преимущественно люминесцентные лампы.

Конструкция рабочего стола должна обеспечивать оптимальное размеще­ние на рабочей поверхности используемого оборудования с учетом его количества и конструктивных особенностей (размер ВДТ и ПЭВМ, клавиатуры и др.), характера выполняемой работы. При этом допускается использование рабочих столов различных конструкций, отвечающих современным требованиям эргономики.

Конструкция рабочего стула (кресла) должна обеспечивать поддержание рациональной рабочей позы при работе на ВДТ и ПЭВМ, позволять изменять позу с целью снижения статического напряжения мышц шейно-плечевой области истины для предупреждения развития утомления.

Тип рабочего стула (кресла) должен выбираться в зависимости от характера и продолжительности работы.

Рабочий стул (кресло) должен быть подъемно-поворотным и регулируе­мым по высоте и углам наклона сиденья и спинки, а также расстоянию спинки от переднего края сиденья, при этом регулировка каждого параметра должна быть не­зависимой, легко осуществляемой и иметь надежную фиксацию.

Экран видеомонитора должен находиться от глаз пользователя на опти­мальном расстоянии 600-700 мм, но не ближе 500 мм с учетом размеров алфавитно-цифровых знаков и символов.

Высота рабочей поверхности стола должна регулироваться в пределах 680-800 мм, при отсутствии такой возможности высота рабочей поверхности стола должна составлять 725 мм.

Рабочий стол должен иметь пространство для ног высотой не менее 600 мм, шириной – не менее 500 мм, глубиной на уровне колен - не менее 450 мм и на уровне вытянутых ног- не менее 650 мм.

Клавиатуру следует располагать на поверхности стола на расстоянии не менее чем 300 мм от края, обращенного к пользователю или на специальной, регу­лируемой по высоте рабочей поверхности, отделенной от основной столешницы.

Помимо требований к организации рабочего места СанПиН 9-131 РБ 2000 устанавливает требования к микроклимату рабочей зоны: влажности, температуре, скорости потока воздуха и пр.

Техника безопасности – это система организационных мероприятий и технических средств, предотвращающих воздействие на работающих опасных производственных факторов.

Для соблюдения техники безопасности в своей деятельности инженер-программист должен руководствоваться Инструкцией по охране труда для программиста при выполнении работ с применением ПЭВМ и ВДТ.

## **Должностная инструкция инженера-программиста.**

Инженер-программист относится к категории специалистов, принимается на работу и увольняется приказом начальника вычислительного центра (ВЦ) по представлению начальника производственного отдела ВЦ, начальника отдела разработок и внедрения ВЦ, руководителя иного структурного подразделения предприятия. При отсутствии вычислительного центра и иного специального подразделения, инженер-программист принимается на работу и увольняется приказом директора и подчиняется непосредственно ему.

На должность:

– инженера-программиста назначается лицо, имеющее высшее профессиональное (техническое или инженерно-экономическое) образование без предъявления требований к стажу работы или среднее профессиональное (техническое или инженерно-экономическое) образование и стаж работы в должности техника 1 категории не менее 3 лет, либо других должностях, замещаемых специалистами со средним специальным образованием, не менее 5 лет.

– инженера-программиста III категории назначается лицо, имеющее высшее профессиональное (техническое или инженерно-экономическое) образование и опыт работы по специальности, приобретенный в период обучения, или стаж работы на инженерно-технических должностях без квалификационной категории.

– инженера-программиста II категории – лицо, имеющее высшее профессиональное (техническое или инженерно-экономическое) образование и стаж работы в должности инженера-программиста III категории не менее 3 лет.

– инженера-программиста I категории – лицо, имеющее высшее профессиональное (техническое или инженерно-экономическое) образование и стаж работы в должности инженера-программиста II категории не менее 3 лет.

В своей деятельности инженер-программист руководствуется:

* постановлениями, распоряжениями, приказами, другими руководящими и нормативными документами, касающимися методов программирования и использования вычислительной техники при обработке информации;
* уставом предприятия;
* приказами и распоряжениями директора предприятия;
* должностной инструкцией.

Инженер-программист должен знать:

* руководящие и нормативные материалы, регламентирующие методы разработки алгоритмов и программ, и использования вычислительной техники при обработке информации;
* основные принципы структурного программирования;
* виды программного обеспечения;
* технико-эксплуатационные характеристики, конструктивные особенности, назначение и режимы работы ЭВМ, правила ее технической эксплуатации;
* технологию автоматической обработки и кодирования информации;
* формализованные языки программирования;
* действующие стандарты, системы счислений, шифров и кодов;
* порядок оформления технической документации;
* передовой отечественный и зарубежный опыт программирования и использования вычислительной техники;
* основы экономики, организации производства, труда и управления;
* основы трудового законодательства;
* правила внутреннего трудового распорядка;
* правила и нормы охраны труда.

На время отсутствия инженера-программиста (отпуск, болезнь, командировка, пр.) его обязанности исполняет лицо, назначенное в установленном порядке, которое несет ответственность за качественное исполнение возложенных на него обязанностей.

На инженера-программиста возлагаются следующие функции:

* разработка программ, обеспечивающих на основе анализа математических моделей и алгоритмов решения экономических и других задач возможность выполнения алгоритма, и, соответственно, поставленной задачи средствами вычислительной техники; проведение их тестирования и отладки;
* разработка технологии решения задач по всем этапам обработки информации.

Для выполнения возложенных на него функций инженер-программист обязан:

* определять информацию, подлежащую обработке средствами вычислительной техники, ее объемы, структуру, макеты и схемы ввода, обработки, хранения и вывода, методы ее контроля;
* осуществлять выбор языка программирования для описания алгоритмов и структур данных;
* выполнять работу по подготовке программ к отладке и проводить отладку;
* определять объем и содержание данных контрольных примеров, обеспечивающих наиболее полную проверку соответствия программ их функциональному значению;
* осуществлять запуск отлаженных программ и ввод исходных данных, определяемых условиями поставленных задач;
* проводить корректировку разработанной программы на основе анализа выходных данных;
* разрабатывать инструкции по работе с программами, оформлять необходимую техническую документацию;
* определять возможность использования готовых программных продуктов;
* осуществлять сопровождение внедрения программ и программных средств;
* разрабатывать и внедрять системы автоматической правильности проверки программ, типовые и стандартные программные средства, составлять технологию обработки информации;
* выполнять работу по унификации и типизации вычислительных процессов;
* принимать участие в создании каталогов и картотек стандартных программ, в разработке форм документов, подлежащих машинной обработке, в проектировании программ, позволяющих расширить область применения вычислительной техники.

Инженер-программист имеет право:

* знакомиться с проектами решений руководства предприятия, касающимися его деятельности;
* вносить на рассмотрение руководства предприятия предложения по совершенствованию работы связанной с обязанностями, предусмотренными настоящей инструкцией;
* запрашивать лично или по поручению непосредственного руководителя от руководителей структурных подразделений, специалистов информацию и документы, необходимые для выполнения своих должностных обязанностей;
* в пределах своей компетенции сообщать непосредственному руководителю о всех недостатках в деятельности предприятия (его структурных подразделениях) и вносить предложения по их устранению;
* привлекать специалистов всех (отдельных) структурных подразделений предприятия для решения возложенных на него обязанностей (если это предусмотрено положениями о структурных подразделениях, если нет – с разрешения их руководителей);
* требовать от своего непосредственного руководителя, руководства предприятия оказания содействия в исполнении им своих должностных обязанностей и прав.

Инженер-программист несет ответственность:

* за неисполнение (ненадлежащее исполнение) своих должностных обязанностей, предусмотренных настоящей должностной инструкцией, в пределах, определенных действующим трудовым законодательством Республики Беларусь;
* за совершение в процессе осуществления своей деятельности правонарушения в пределах, определенных действующим административным, уголовным и гражданским законодательством Республики Беларусь;
* за причинение материального ущерба – в пределах, определенных действующим трудовым, уголовным и гражданским законодательством Республики Беларусь.

**1.8 Используемые методологии разработки программного обеспечения**

Бизнес компаний часто зависит от качества работы информационных систем. Ошибки в ИТ-приложениях, некорректное выполнение функций могут привести к финансовым потерям. Для снижения подобных рисков *EPAM Systems* использует при разработке программного обеспечения методологию *Continuous Integration* (*CI*, непрерывная интеграция).

*Continuous Integration* – это полностью автоматизированная практика создания, сборки и тестирования программного кода. При ее применении компиляция и компоновка кода начинает проводиться как можно раньше и повторяется как можно чаще. Специальное программное обеспечение отслеживает процесс разработки: при наличии изменений в коде (например, добавилась новая часть) автоматически запускается процесс сборки и тестирования. Это позволяет найти дефекты и противоречия в компонентах системы еще на ранних стадиях ее создания. В результате – обеспечивается высокое качество программного обеспечения.

Преимущества *Continuous Integration*:

* сокращение ручных операций - этапы создания, сборки и тестирования программного обеспечения проводятся в автоматическом режиме;
* наличие рабочей ИТ-системы на всем протяжении процесса разработки – у проектной команды всегда есть свежая версия решения для демонстрации заказчику, получения обратной связи и быстрой доработки;
* качество программного обеспечения – в рамках *Continuous Integration* используются различные программные средства для контроля качества кода, что позволяет сократить количество ошибок;
* минимизация рисков – дефекты выявляются на ранних стадиях разработки информационной системы, что помогает избежать увеличения сроков и стоимости проекта;
* окупаемость инвестиций в ИТ – автоматизация процесса разработки обеспечивает высокую эффективность и надежность информационной системы.

Для разработки программного обеспечения в рамках *Continuous Integration* в *EPAM* используются различные программные и аппаратные средства. Выбор конкретных инструментов зависит от задач и специфики проекта.

* развертывание и подготовка к работе: *VMWare*, *Microsoft* *Hyper-V*, *Citrix Xen*, *Parallels*;
* средства разработки: *Eclipse*, *MS Visual Studio*, *Borland Delphi* и т.д;
* системы контроля версий: *StarTeam*, *Perforce*, *CVS*, *PVCS*, *VSS*, *Synergy*, *Subversion*, *GIT* и т.д;
* среды и системы автоматической сборки: *CruiseControl*, *CC.net*, *TeamCity*, *Build Forge*, *Team Foundation Server*, *Electric Commander*;
* компиляторы и средства сборки: *Compilers*, *Linkers*, *Ant*, *Make*, *NAnt* *MSBuild*, *Maven*;
* контроль качества: *Mercury Quality Center*, *LoadRunner*, *TestDirector*, *WinRunner*, *Xunit*, *Clover*, *IBM Functional*, *Performance* *& Manual Tester*.

Процесс *Continuous Integration* может быть полностью – от написания кода до запуска готового решения в эксплуатацию – выстроен как облачный сервис. Требуемое качество программного обеспечения и соблюдение сроков разработки и поддержки ИТ-систем обеспечиваются с помощью заключения *SLA* (*Service Level Agreement* – соглашение об уровне услуг) между заказчиком и *EPAM*.

**1.9 Используемые методы управления жизненным циклом проекта**

Для управления жизненным циклом разрабатываемых проектов в компании «*Epam* *Systems*» используется *Subversion*, который очень эффективно и хорошо реализуется на базе *BitBucket*.

*Subversion* – (также известная как «*SVN*») –свободная централизованная система управления версиями, официально выпущенная в 2004 году компанией *CollabNet* *Inc*.

Цель проекта – заменить собой распространенную на тот момент систему *Concurrent* *Versions* *System* (*CVS*), которая ныне считается устаревшей. *Subversion* реализует все основные функции *CVS* и свободна от ряда недостатков последней.

В настоящее время *Subversion* используется многими сообществами разработчиков открытого программного обеспечения (в том числе сообществами, ранее использовавшими *CVS*). В их числе такие известные проекты, как *Apache*, *GCC*, *FreePascal*, *Python*, *Ruby*, *FreeBSD*, *AROS*, *Blender*, *Boost*, *Tor*, *OGRE*. *Subversion* также широко используется в закрытых проектах и корпоративной сфере. Хостинг *Subversion*, в том числе для проектов с открытым кодом, также предоставляют популярные хостинг-проекты *SourceForge*.*net*, *Tigris*.*org*, *Google* *Code* и *BountySource*.

*BitBucket* («ведро битов») – веб-сервис для хостинга проектов и их совместной разработки, основанный на системе контроля версий *Mercurial* и *Git*. По назначению и предлагаемым функциям аналогичен *GitHub* (однако *GitHub* не предоставляет бесплатные «закрытые» репозитории, в отличие от *Bitbucket*), который поддерживает *Git* и *Subversion*.

*Git* – распределённая система управления версиями файлов. Проект был создан Линусом Торвальдсом для управления разработкой ядра *Linux*, первая версия выпущена 7 апреля 2005 года.

Ядро *Git* представляет собой набор утилит командной строки с параметрами. Все настройки хранятся в текстовых файлах конфигурации. Такая реализация делает *Git* легко портируемым на любую платформу и даёт возможность легко интегрировать *Git* в другие системы (в частности, создавать графические *git*-клиенты с любым желаемым интерфейсом).

Репозиторий *Git* представляет собой каталог файловой системы, в котором находятся файлы конфигурации репозитория, файлы журналов, хранящие операции, выполняемые над репозиторием, индекс, описывающий расположение файлов и хранилище, содержащее собственно файлы. Структура хранилища файлов не отражает реальную структуру хранящегося в репозитории файлового дерева, она ориентирована на повышение скорости выполнения операций с репозиторием. Когда ядро обрабатывает команду изменения (неважно, при локальных изменениях или при получении патча от другого узла), оно создаёт в хранилище новые файлы, соответствующие новым состояниям изменённых файлов. Существенно, что никакие операции не изменяют содержимого уже существующих в хранилище файлов.

По умолчанию репозиторий хранится в подкаталоге с названием «.*git*» в корневом каталоге рабочей копии дерева файлов, хранящегося в репозитории. Любое файловое дерево в системе можно превратить в репозиторий *git*, отдав команду создания репозитория из корневого каталога этого дерева (или указав корневой каталог в параметрах программы). Репозиторий может быть импортирован с другого узла, доступного по сети. При импорте нового репозитория автоматически создаётся рабочая копия, соответствующая последнему зафиксированному состоянию импортируемого репозитория (то есть не копируются изменения в рабочей копии исходного узла, для которых на том узле не была выполнена команда *commit*).

Репозиторий *git* бывает локальный и удаленный. Локальный репозиторий – это поддиректория .*git*, создается (в пустом виде) командой *git* *init* и (в непустом виде с немедленным копированием содержимого родительского удаленного репозитория и простановкой ссылки на родителя) командой *git* *clone*.

Практически все обычные операции с системой контроля версий, такие, как коммит и слияние, производятся только с локальным репозиторием. Удаленный репозиторий можно только синхронизировать с локальным как «вверх» (*push*), так и «вниз» (*pull*).

Наличие полностью всего репозитория проекта локально у каждого разработчика дает *git* ряд преимуществ перед *SVN*. Так, например, все операции, кроме *push* и *pull*, можно осуществлять без наличия Интернет-соединения.

Очень мощной возможностью *git* являются ветви, реализованные куда более полно, чем в *SVN*. Создать новую ветвь так же просто, как и совершить коммит. По сути, ветвь *git* есть не более чем читаемое человеком имя, «навешенное» на некий коммит в репозитории (используется поддиректория *refs*). Коммит без создания новой ветви всего лишь передвигает эту ссылку на себя, а коммит с созданием ветви – оставляет старую ссылку на месте, но создает новую на новый коммит, и объявляет её текущей. Заменить локальные девелоперские файлы на набор файлов из иной ветви, тем самым перейдя к работе с ней – также тривиально.