Программное обеспечение для миграции облачной инфраструктуры «Мигратор»

Описание процессов, обеспечивающих поддержание жизненного цикла разработки, в том числе устранение неисправностей и внесение совершенствований

ОГЛАВЛЕНИЕ

ОГЛАВЛЕНИЕ 2
ПЕРИОД ДЕЙСТВИЯ ДОКУМЕНТА 3
ОБЩИЕ ПОЛОЖЕНИЯ 4
ЭТАПЫ РАЗРАБОТКИ ПО 5
Определение требований 7
Оценка рисков проекта 7
Этап проектирования 8
Разработка архитектуры 8
Разработка ПО 9
СИСТЕМА УПРАВЛЕНИЯ ВЕРСИЯМИ ИСХОДНОГО КОДА 9
ПРОЦЕССЫ ПОДДЕРЖКИ ПО 11

ПЕРИОД ДЕЙСТВИЯ ДОКУМЕНТА

ООО «ИНТЕЛСОРС» периодически обновляет документацию на свои программные продукты.

Следовательно, если данный документ не был загружен в течение недавнего времени, в нем может отсутствовать последняя обновленная информация.

Актуальные версии документов размещены на веб-сайте компании https://intelsource.ru/

ОБЩИЕ ПОЛОЖЕНИЯ

Настоящий документ регулирует порядок выпуска **ООО «ИНТЕЛСОРС»** обновлений программного обеспечения "Мигратор" (далее – Продукт).

Обновления выпускаются в целях улучшения функциональных возможностей, предотвращения и устранения недостатков Продукта.

Право на получение обновлений Продукта имеет лицо, которое приобрело лицензию на Продукт по договору с ООО «ИНТЕЛСОРС» или партнером ООО «ИНТЕЛСОРС» (далее – Пользователь).

Обновления предоставляются в соответствии с лицензионным договором, заключенным между Пользователем и ООО «ИНТЕЛСОРС» или партнером ООО «ИНТЕЛСОРС».

ЭТАПЫ РАЗРАБОТКИ ПО

Процесс разработки программного обеспечения **ПО Мигратор** (далее ПО) включает в себя следующие основные этапы:

- разработка и согласование ТЗ проекта;
- согласование интерфейса оператора;
- согласование структуры ПО;
- разработка исходного кода ПО, компиляция и проведение тестирование программных модулей;
- разработка программной документации;
- проведение комплексного тестирования и устранения замечаний;
- проведение независимого тестирования и устранения замечаний;
- разработка инструкции по изготовлению носителя;
- проведение предварительных испытаний, коррекция КД и ПД по результатам испытаний;
- проведение приемочных испытаний, коррекция КД и ПД по результатам испытаний;
- присвоение литеры О1.

Для разрабатываемого на предприятии ПО в качестве модели жизненного цикла для большинства проектов выбирается спиральная модель, соответствующая масштабу и сложности проекта.

Спиральная модель схожа с инкрементной моделью, но с в ней уделяется больше внимания оценки и разрешения рисков. Спиральная модель подразделяет реализацию проекта на четыре этапа: планирование проекта, оценка рисков, **проектирование и разработка и проведение оценки**. Проект каждый раз заново проходит через эти четыре стадии при создании новой версии или фрагмента ПО (что соответствует одному витку спирали в данной модели). Базовый виток спирали, начинающийся на этапе

постановки задач, включает в себя определение требований и оценку рисков. Каждый последующий виток строится на основе базового.

Требования программному продукту и к особенностям разработки программного обеспечения определяются на этапе постановки задач. На этапе оценки и разрешения рисков применяется специальный процесс для определения рисков и нахождения разных решений по их разрешению. По окончании данного этапа создается прототип ПО.

Третий этап включает в себя непосредственно разработку ПО и его **тестирование** по окончании данного этапа. Этап проверки позволяет разработчику и заказчику оценить результат проекта на текущий момент, прежде чем начнется новый виток разработки.

Каждый этап жизненного цикла ПО подразделяется на различные процессы и операции.

На первом этапе определяется общая концепция разрабатываемого программного продукта («вид с высоты птичьего полета»), и на ее основе строится базовая структура проекта, оценивается его выполнимость и связанные с ним риски, описываются соответствующие подходы к конфигурационному управлению и технологиям.

Наиболее важная часть плана проекта – декомпозиция системы на совокупность составных частей в соответствии с требованиями высокого уровня системной иерархии. Все требования к компонентам ПО, устанавливаемые на этапе определения требований, следуют из одного или нескольких требований высокого уровня.

Результатами этапа проектирования являются план конфигурационного управления, план реализации качества ПО, план реализации проекта и календарный план, содержащий подробный список

запланированных работ грядущего этапа определения требований, а также предварительная оценка трудозатрат на последующих этапах.

Определение требований

В процессе определения требований в качестве исходных данных используются цели, поставленные в разделе плана проекта, описывающем требования высокого уровня.

Каждый программный компонент должен иметь собственную Спецификацию требований.

Требования к ПО определяют функционал программного компонента, производительность, точность, временные характеристики работы, затраты ресурсов используемого оборудования, работоспособность в нестандартных условиях и при перегрузках. В Спецификации требований к программному обеспечению описываются алгоритмы и математические методы.

Оценка рисков проекта

Риск – это любое событие, которое может помешать реализации проекта в соответствии с планом или его успешному завершению. Риски можно идентифицировать из разных источников. Некоторые из них могут быть довольно очевидными и будут выявлены до начала проекта. Другие будут идентифицированы в течение жизненного цикла проекта, и риск может быть идентифицирован любым участником проектом. Некоторые риски будут присущи самому проекту, в то время как другие будут результатом внешних воздействий, которые полностью неподконтрольны команде проекта.

Этап проектирования

На этапе проектирования в качестве исходных данных используются требования, определенные в принятой спецификации. По каждому требованию определяется элемент или набор элементов проектирования по результатам согласования с заказчиком, моделирования или работы над прототипами.

Элементы проектирования подробно описывают требуемый функционал ПО и, как правило, включают в себя схемы функциональной иерархии, схемы расположения элементов визуализации на экране, таблицы правил, схемы деловых процессов, псевдокод, а также схему всех потоков данных с полным словарем данных. Эти элементы проектирования предназначены для описания ПО в объеме, достаточном для того, чтобы квалифицированные специалисты могли разработать ПО с минимальной потребностью в дополнительных данных.

Разработка архитектуры

На данном этапе определяется архитектурный проект системы, в соответствии с которым выполняется идентификация элементов ПО и удовлетворяются заданные требования. При определении верхнего уровня архитектуры системы должны быть идентифицированы составные части технических средств, программных средств и ручных операций. Должно быть учтено что все системные требования распределяются между этими составными частями. Составные части конфигурации технических средств, программных средств и ручных операций должны последовательно идентифицироваться этими составными частями.

Разработка ПО

На этапе разработки ПО в качестве исходных данных используются элементы проектирования, описанные в принятом плане разработки. По каждому элементу определяется артефакт или набор артефактов ПО. Артефакты ПО включают в себя (но не ограничиваются ими) меню, диалоговые окна, формы для управления данными, форматы отчетных данных и специализированные процедуры, и функции.

СИСТЕМА УПРАВЛЕНИЯ ВЕРСИЯМИ ИСХОДНОГО КОДА

В процессе разработки ПО в качестве распределенной системы управления версиями исходного кода используется инструмент Git.

Git – это программное обеспечение, свободно распространяемое на условиях универсальной общедоступной лицензии GNU версии 2.

Каждый рабочий каталог в Git – это полноценный репозиторий, содержащий всю историю проекта с возможностью отслеживания версий, не зависящий от доступа к сети или центральному серверу.

Концепция Git возникла на основе опыта, полученного при управлении большим распределенным проектом разработки при работе над Linux, а также особенностей работы с файловыми системами, и острой потребности в создании жизнеспособной системы в кратчайшие сроки. Все это привело к следующим особенностям реализации:

Git позволяет быстро создавать и осуществлять слияние отдельных ветвей проекта и включает в себя специальные инструменты для визуализации и навигации по нелинейной истории разработки. Основным принципом в Git является предположение, что слияние изменения будет производиться чаще, чем его написание, так как оно распределяется для

оценки несколькими разработчиками. Ветви в Git занимают очень мало места: они являются лишь ссылками на отдельные коммиты. С помощью родительского коммита может быть построена полная структура ветвей.

Git предоставляет каждому разработчику локальную копию всей истории разработки, и изменения копируются из одного такого репозитория в другой. Такие изменения импортируются в виде дополнительных ветвей разработки, и их слияние может осуществляться так же, как и для ветви, разработанной локально.

Репозитории могут быть опубликованы с помощью HTTP, FTP, rsync или протокола Git через обычный сокет, ssh или HTTP. Git также имеет эмулятор сервера CVS (системы одновременных версий), который позволяет использовать существующие клиенты CVS и плагины IDE для доступа к репозиториям Git. Репозитории подверсий и SVK можно использовать напрямую с помощью git-svn.

История хранится в Git таким образом, что идентификатор конкретной версии («коммит» в терминологии Git) зависит от полной истории разработки, предшествовавшей данному коммиту. После его публикации невозможно изменить старые версии незаметно. Эта структура похожа на дерево хешей, но с наличием дополнительных данных в узлах и листовых вершинах.

ПО разрабатывается несколькими специалистами. Когда разработчик начинает внедрение нового функционала или отладку, он забирает последнюю версию проекта из системы Git. После выполнения задачи новая версия ПО хранится на его локальном хосте. Прежде чем залить новую версию в одну из рабочих ветвей, менеджер версий ПО отправляет измененное ПО на проверку двум другим разработчикам, выбранным произвольно. После одобрения кода ПО отправляется на слияние.

Менеджер версий ПО пытается совместить новые изменения с обновленными ветвями проекта. При неудачном слиянии новый код отправляется разработчикам ПО, чтобы они одобрили общее обновление. При успешном слиянии новая ветвь отправляется на тестирование. Тестировщики проверяют обновленное ПО разными методами, и при успешном прохождении тестов ветвь разработки подготавливается к релизу. Автоматическая компиляция версии релиза проекта осуществляется дважды в день. Также ПО проверяется в тестовой среде. Если все тесты и компиляция проходят успешно, версия считается готовой к релизу. В противном случае менеджер версий ПО находит проблему и возвращает ветвь разработчикам для исправления.

В процессе разработки было задействовано 2 ведущих инженера-программиста.

Разработка ПО проводится специалистами компании ООО «ИНТЕЛСОРС». Адрес компании: Республика Татарстан, Верхнеуслонский район, г. Иннополис, ул. Университетская д 7 этаж 5 помещ 503.

Контактный телефон для связи с группой разработки: +7 495 182 23 93, электронная почта: migratorsupport@ИНТЕЛСОРС.ru

ПРОЦЕССЫ ПОДДЕРЖКИ ПО

Цель процесса поддержки и решения проблем в Программе заключается в обеспечении гарантии качества оказанных услуг по контракту (или договору) и того, что все выявленные запросы идентифицируются, анализируются, контролируются для осуществления их решения.

В процессе поддержки и решения проблем в Программе:

a) проблемы регистрируются, идентифицируются и классифицируются в систему управления запросами (далее -- СУЗ);

b) запросы анализируются и оцениваются для определения приемлемого решения (решений);

c) выполняется решение запросов;

d) запросы отслеживаются вплоть до их закрытия;

e) известно текущее состояние всех зафиксированных запросов;

f) предоставляются регулярные версии Программы (в случае оказания услуг по сопровождению);

g) проводятся регламентные работы.

Режим работы службы поддержки - с 9:00 до 18:00 MSK.

В процессе сопровождения задействовано 2 ведущих инженера-программиста.

Фактический почтовый адрес, по которому осуществляется процесс сопровождения - 420500, Республика Татарстан, Верхнеуслонский район, г. Иннополис, ул. Университетская д 7 этаж 5 помещ 503.

В процессе модернизации задействовано 2 ведущих инженера-программиста.

Разработка ПО проводится специалистами компании ООО «ИНТЕЛСОРС». Адрес компании: Республика Татарстан, Верхнеуслонский район, г. Иннополис, ул. Университетская д 7 этаж 5 помещ 503.

В процессе гарантийного обслуживания задействовано 2 ведущих инженера-программиста.

Фактический почтовый адрес, по которому осуществляется процесс сопровождения - 420500, Республика Татарстан, Верхнеуслонский район, г. Иннополис, ул. Университетская д 7 этаж 5 помещ 503.

Совершенствование и модернизация программы происходит путем доработки интерфейса и внедрения новых функций. Увеличивается производительность программы, и минимизируется время на взаимодействия между органами управления интерфейса всего комплекса и пользователем. Фактическое улучшение всей программы отображается в виде изменения младших разрядов номера версии ПО с учетом доработок.

В процессе тестирования и эксплуатации программного обеспечения могут возникнуть сообщения о неисправности. В случае их возникновения необходимо осуществить процедуру передачи информации о характере ошибки в в техническую поддержку ООО «ИНТЕЛСОРС» по телефону : +7 495 182 23 93. Устранение неисправностей и техническое обслуживание может осуществлять только квалифицированный персонал, а именно сотрудники компании ООО «ИНТЕЛСОРС». Для оформления заявки на устранения неисправности необходимо оформить обращение через электронную почту на электронную почту: migratorsupport@ИНТЕЛСОРС.ru.

Так же заявку можно отправить по адресу: 420500, Республика Татарстан, Верхнеуслонский район, г. Иннополис, ул. Университетская д 7 этаж 5 помещ 503.