Регламент выпуска версий системы ELMA 3 и работы с ветками SVN.

Адаптация для отдела разработки и тестирования.

Оглавление

[Принятые обозначения версий 2](#_Toc349030704)

[Работа с ветками репозитория 2](#_Toc349030705)

[Ежедневная работа отдела разработки 3](#_Toc349030706)

[Регламент обновления веток разработки 4](#_Toc349030707)

[Правило именования новой ветки для длительной разработки 5](#_Toc349030708)

[Работа с веткой для продуктивного сервера EleWise 5](#_Toc349030709)

[Отчет об изменениях в версии 6](#_Toc349030710)

[Процедура выкладывания версии на тестовые и демонстрационные стенды 7](#_Toc349030711)

[Cтенд для выкладывания текущей версии тестирования с ветки Dev или Dev20 7](#_Toc349030712)

[Cтенд для выкладывания текущей релизной версии с ветки Release 9](#_Toc349030713)

[Cтенд для выкладывания онлайн-демо (demo.elma-bpm.ru) с ветки Release 10](#_Toc349030714)

[Работа с запросами в системе JIRA 11](#_Toc349030715)

В соответствии с общим регламентом выпуск версии происходит 1-го числа каждого месяца, т.е. в этот день уже выкладываются пакеты на сервера для скачивания.

## Принятые обозначения версий

|  |  |
| --- | --- |
| **Release** | Текущая релизная минорная версия |
| **Dev** | Текущая версия, которая находится в разработке (3.0.2). Сюда скидывается все, что сделали программисты и сами протестировали. На ней ведется тестирование. |
| **Dev20** | Версия получается копированием версии **Dev** 20-го числа. Живет 10 дней. На ней идет усиленное тестирование и туда не заливаются новые функции. |
| **~~Stable~~** | ~~Данная версия появляется 1-го числа месяца вместе с релизом и потом 30-го числа превращается в следующий релиз. Сюда попадают функции из ветки~~ **~~Dev~~**~~, которые были протестированы отделом QA. Т.е. это «более-менее» стабильная версия.~~  **На данный момент не используется из-за чрезмерной сложности при непоследовательном слиянии изменений.** |
| **DevNext** | Следующая минорная версия для планирования (3.0.3) |
| **DevFuture** | Все следующие версии для планирования (3.0.4, 3.0.5, … FutureRelease) |

## Работа с ветками репозитория

Для обеспечения качества продукта вводится следующий регламент на работу с ветками.

|  |  |
| --- | --- |
| **Branches** | Для всех доработок функционала и исправлений ошибок, требующих **больше 2-х дней работы** необходимо создавать ветки в папке **branches\**  После завершения работы изменения сливаются в ветку **Trunk**. |
| **Trunk** | Основная ветка, служит для сбора и проверки всего функционала. Тут делаются доработки и ведутся текущие исправления ошибок, требующие **меньше 2-х дней работы**. |
| **Dev** | Тестовая ветка, необходима для обеспечения буфера качества. Сюда сливаются изменения из ветки **Trunk** после первоначальной проверки разработчиком. Сюда уже попадает **только готовый функционал**.  В эту ветку также может происходить **слияние критичных исправлений** из ветки **Release**. |
| **Dev20** | Тестовая ветка текущей версии с **20 числа**. Эта ветка создается 20 числа, сюда идет слияние только исправлений по версии **Dev** (3.0.2). После выпуска версии эта ветка удаляется до следующего 20 числа. |
| **Stable** | Текущая стабильная ветка разработки R&D. (т.е. после выпуска версии 3.0.1 тут лежит ветка 3.0.2). Сюда сливаются **только протестированные и подтвержденные изменения из ветки Dev.** |
| **Release** | Ветка с текущим релизом (3.0.1). В этой ветке **можно производить только исправления критических ошибок** по линии VIP техподдержки. |
| **EleWise** | Ветка для **продуктивного сервера EleWise.** В этой ветке сливаются изменения готовые для тестирования внутренним заказчиком на рабочей среде. Слияние изменений в эту ветку **обязательно согласуется** с самим заказчиком и начальником отдела разработки. |

Общий график использования веток в цикле разработки (от даты выпуска версии 3.0.1 до релиза версии 3.0.3)

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 |  |  |  | 5 |  |  |  |  |  |  |  |  |  | 15 |  |  |  |  | 20 |  |  |  |  |  |  |  |  |  | 30 |
|  | #1 **Release** (3.0.1) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|  | #2 **Dev** (3.0.2) | | | | | | | | | | | | | | | | | | | | #3 **Dev** (3.0.3 - **DevNext**) | | | | | | | | | |
|  |  | | | | | | | | | | | | | | | | | | | | #2 **Dev20** (3.0.2 - **Dev**) | | | | | | | | | |
|  | ~~#3~~ **~~Stable~~** ~~(3.0.2)~~ | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| **Выпуск версии 1-го числа месяца, начинается новый месяц** | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|  | #1 **Release** (3.0.2) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|  | #2 **Dev** (3.0.3 - **DevNext**) | | | | | | | | | | | | | | | | | | | | #3 **Dev** (3.0.4 - **DevFuture**) | | | | | | | | | |
|  |  | | | | | | | | | | | | | | | | | | | | #2 **Dev20** (3.0.3 - **DevNext**) | | | | | | | | | |
|  | ~~#3~~ **~~Stable~~** ~~(3.0.3)~~ | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |

Принимается следующая терминология обозначения ключевых чисел в рамках месяца:

|  |  |  |
| --- | --- | --- |
| **Дата** | **Обозначение** |  |
| **0 (1) число** | **0 число \ Выпуск версии** | День в месяце, когда делается выкладка готовых версий на публичный сервер. Этот день определяется при планировании в предыдущем месяце в соответствии с рабочим календарем и графиком работ. В этот день осуществляются следующие движения:   * переход на версию разработки **Dev** (3.0.2) * обновление веток разработки в соответствии с регламентом * оповещение всех заинтересованных лиц о выходе версии * отчетное совещание, выставление процентов эффективности |
| **5 число** | **Общее собрание** | В этот день проводится ряд совещаний и общее собрание с партнерами. В этот же день окончательно утверждается объем работ на версию **DevNext** (3.0.3). Производится сбор и фиксация требования на разработку.  Регламент этого дня готовится отдельно (//**TODO** Будин) |
| **15 число** | **День планирования** | В этот день проводится сводка и планирование ресурсов на версию **DevFuture** (3.0.4). Планирование версии осуществляется на основе отзывов и накопленных ошибок. Также необходимо учитывать готовность тех. заданий по отдельным требованиям для версии **DevNext** (3.0.3).  В этот же день производится финализация плана на версию **DevNext** (3.0.3). |
| **20 число** | **Заморозка версии** | В этот день производится заморозка ветки разработки версии **Dev** (3.0.2), создается ветка **Dev20** и начинается активный процесс тестирования и исправления в данной ветке.  В этот же день происходит старт разработки версии **DevNext** (3.0.3), весь незавершенный функционал переходит в план версии **DevNext** (3.0.3).  В этот же день при необходимости производится слияние в ветку EleWise и выкладывание на продуктивный сервер. |
| **30 число** | **Релиз версии** | В этот день происходит финальная сборка всех артефактов (инсталляционные пакеты, справка). Все пакеты готовятся для выкладывания на публичных серверах (помечаются сборки на сервере сборок). |

### Ежедневная работа отдела разработки

|  |  |
| --- | --- |
| **С 0 по 20** | **Активная фаза разработки**  В это время выделяется **80% времени на разработку нового функционала, 20% - на текущие критичные исправления**, техподдержку и другие работы по улучшению качества продукта.  В эти дни необходимо работать максимально на результат, т.е. **внедрение нового функционала в систему**. |
| **Разработка запланирована на 2 дня или меньше** | Цикл для разработчика по веткам:  **Trunk** (собственная отладка и тестирование)   * **Dev** (отправка на проверку в отдел тестирования)   + Тестирование не прошло: правка в **Dev,** слияние исправлений в **Trunk** * **Stable**(слияние в стабильную ветку и тестирование) |
| **Разработка запланирована дольше 2-х дней** | Цикл для разработчика по веткам:  **Branch** (создается ветка, собственная отладка и тестирование, периодическое слияние изменений из **Trunk**)   * **Trunk** (разработка полностью завершена, пройдено первоначальное тестирование разработчиком) * **Dev** (отправка на проверку в отдел тестирования)   + Тестирование не прошло: правка в **Dev,** слияние исправлений в **Trunk** * **Stable**(слияние в стабильную ветку и тестирование) |
| **C 20 по 0** | **Активная фаза тестирования и исправления**  В это время выделяется **80% времени на исправления в разработанном функционале, запланированные исправления на версию в джире, 20% - на старт новой фазы разработки версии DevNext** (3.0.3), на продолжение длительных разработок и планирование новых.  В эти дни необходимо работать максимально на результат, т.е. **выпуск качественного продукта в срок**. |
| **Исправление ошибки, 2 дня или меньше** | Цикл для разработчика по веткам:  **Dev20** (собственное исправление и тестирование, отправка на тестирование)   * Тестирование не прошло: правка в **Dev20** * **Dev**, **Trunk** (слияние исправлений, либо их повторение в ветках) * **Stable** (слияние в стабильную ветку и тестирование) |
| **Исправление ошибки дольше, чем 2 дня** | Цикл для разработчика по веткам:  **Branch** (делается ветка из текущей **Dev20**, собственное исправление и тестирование, отправка на тестирование)   * **Dev20** (слияние полностью готового исправления)   + Тестирование не прошло: правка в **Dev20** * **Dev**, **Trunk** (слияние исправлений, либо их повторение в ветках) * **Stable** (слияние в стабильную ветку и тестирование) |
| **Исправление ошибки по линии VIP техподдержки** | При поступлении запроса на исправление по линии VIP техподдержки необходимо **обеспечить качество и скорость исправления, также нельзя ломать текущую публичную версию**.  Цикл для разработчика по веткам:  **Branch**(всегда делается ветка из **Release**, исправления и тестирование с подключением отдела тестирования)   * **Release**(слияние исправлений, отправка на тестирование, уведомление сотрудника техподдержки)   + Тестирование не прошло: правка в **Release**, уведомление ответственного за запрос. * **Dev, Trunk, Dev20, Stable**(слияние исправлений в ветки, либо повторение исправлений, в ветку **Dev20**только с 20 по 0) |

## Регламент обновления веток разработки

|  |  |
| --- | --- |
| **30 число / Релиз версии** | В этот день полностью должны все залить финальные исправления в ветку **Stable**. После этого необходимо проделать следующее:   1. Перенести ветку **Release** в **Tags\Release\_3\_x\_y** 2. ~~Скопировать ветку~~ **~~Stable~~** ~~в~~ **~~Release~~** 3. Скопировать ветку **Dev20** в **Release** 4. Удалить ветку разработки **Dev20**   Таким образом, в ветке **Release** должна оказаться версия для сборки релиза на следующий день (День выпуска версии). |
|  |  |
| **20 число / Заморозка версии** | В этот день фиксируется функционал версии, заливка новых функций в ветку **Dev** прекращается. Необходимо проделать следующее:   1. Скопировать ветку **Dev** в новую ветку **Dev20**   Таким образом, мы получаем ветку **Dev20,** в которой идет только исправление ошибок и которая живет до Релиза версии. |

## Правило именования новой ветки для длительной разработки

Ветки нужны для того, чтобы в случае непредвиденных обстоятельств работа не потерялась, или над задачей могли продолжать работу более одного человека.

Правило именования очень простое:

1. Работа в ветке может вестись **ТОЛЬКО** по запросу в джире (Фича, Заказная разработка, Ошибка, Залет), если запроса нет – необходимо его создать в обязательном порядке.
2. Создаем ветку **branches\Dev-ELMA-XXX-IssueDesc**, где
   1. ELMA-XXX – код задачи в джире
   2. IssueDesc – очень краткое описание запроса на английском (например, для фичи интеграции с LDAP – это будет *LDAP* или для замещений - *Replacement*). В случае, если исправление идет по ошибке и описать ее одним словом не представляется возможным, то вместо IssueDesc просто подставляете *Bug*.

Пример именования веток:

* ELMA-2062 Формы добавления / редактирования / просмотра кастомных объектов  
  ветка: branches\Dev-ELMA-2062-CustomForms
* ELMA-2396 [Проекты] Исключение при импорте контрольной точки с подзадачей  
  ветка: branches\Dev-ELMA-2396-Bug

## Работа с веткой для продуктивного сервера EleWise

Для обеспечения стабильности и качества разработки вводится специальная ветка **EleWise.** Процесс работы с этой веткой выстраивается следующим образом:

1. При заморозке или выпуске версии
   * Сделать копию текущей ветки (**Dev** или **Release** соответственно) в ветку **EleWise**
   * Слить все изменения из веток длительных разработок (список веток предварительно собрать через джиру и разработчиков)
   * Отправить сборку на предварительное тестирование (не более 1 дня)
2. После выпуска версии (обновление по необходимости)
   * Из ветки разработки делается слияние недостающего функционала в ветку **EleWise**
   * При необходимости (уточнять у разработчика функционала) сборка отправляется на тестирование (не более 1 дня)

Основное предназначение данной ветки – это чтобы непроверенный заказчиком (Будиным) функционал не попадал в общую ветку разработки и как следствие не проходил в релиз. По окончании проверки на боевом сервере функционал в общем порядке сливается в текущие ветки (**Dev**, **Dev20**, **Trunk**).

## Отчет об изменениях в версии

Для обеспечения стабильной поддержки пользователей между версиями, необходимо предоставлять отчет об изменениях в версии. В таком отчете **должны быть**:

* Изменения публичной части кода (API), в том числе модели данных
* Изменения пользовательского интерфейса, касающегося настроек или форм ввода

Описывать надо только изменения уже существующего функционала, пример описания изменений можно увидеть в документе <http://elma.elewise.com/Documents/Document/View/23785>.

Такой документ необходимо в обязательном порядке предоставлять перед выпуском версии **30 числа**. Для этого в ЭДО в папке <http://elma.elewise.com/Documents/Folder/View/23782> в папку с номером версии складываете документ с именем в формате «Список исправлений к версии (Фамилия И.О.)».

## Процедура выкладывания версии на тестовые и демонстрационные стенды

### Cтенд для выкладывания текущей версии тестирования с ветки Dev или Dev20

**Адрес сервера**: dev-elma3.prj.elewise.com

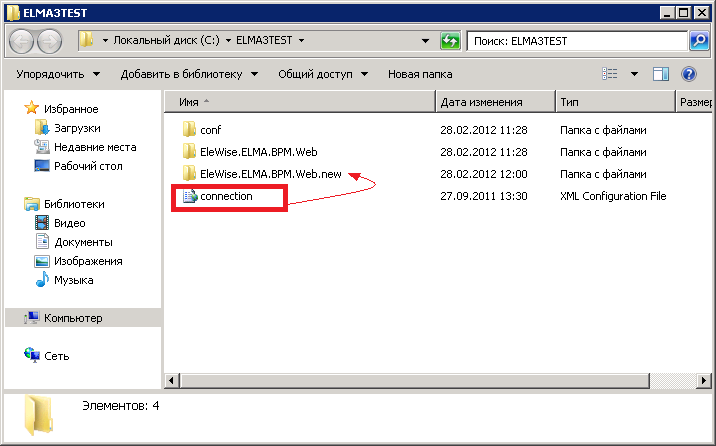
**Периодичность выкладывания**: по необходимости \ по готовности новой сборки

**Тестовые веб-сайты, запущенные на IIS:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Имя сайта в IIS** | **Порт** | **СУБД** | **Физический путь до веб-сайта** |
| ELMA3TEST | 80 | MS SQL | C:\ELMA3TEST\EleWise.ELMA.BPM.Web |
| ELMA3TEST\_FB | 8090 | Firebird | C:\ELMA3TEST\_FB\EleWise.ELMA.BPM.Web |

Для обновления нужно (рассмотрим на примере обновления сайта ELMA3TEST):

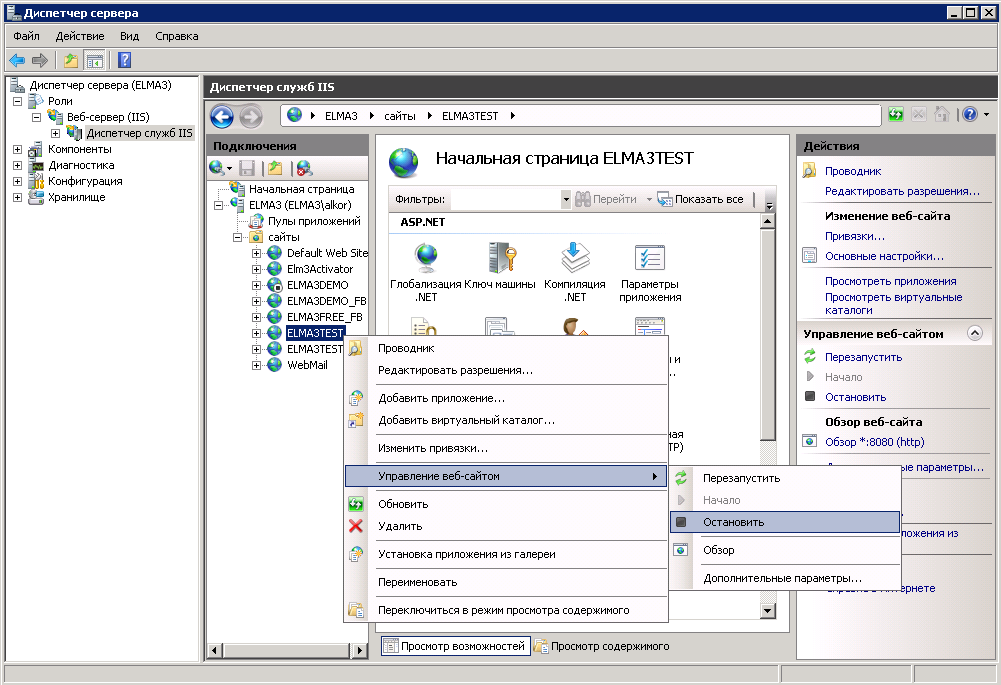
1. Войти через терминал-клиент на сервер **dev-elma3.prj.elewise.com**. Входить нужно под своим доменным пользователем (как в почту или джиру)
2. Скачать с сервера сборок (**build.elewise.com**) архив с веб-частью нужной сборки: <http://build.elewise.com/viewType.html?buildTypeId=bt126&tab=buildTypeStatusDiv>.
3. Распаковать содержимое архива в какую-либо папку рядом с установленным веб-сайтом (например, **C:\ELMA3TEST\EleWise.ELMA.BPM.Web.NEW**).
4. Скопировать файл **connection.config** из родительской папки веб-сайта (в данном случае C:\ELMA3TEST\connection.config) в папку из пункта 3 (такой файл уже будет существовать в этой папке, его нужно заменить).



1. Открыть консоль управления IIS (можно через «Диспетчер сервера»).



1. Остановить веб-сайт.



1. Удалить папку со старым веб-сайтом **C:\ELMA3TEST\EleWise.ELMA.BPM.Web**, а затем переименовать **C:\ELMA3TEST\EleWise.ELMA.BPM.Web.NEW** в **C:\ELMA3TEST\EleWise.ELMA.BPM.Web**.
2. Запустить веб-сайт (см. скриншот из п. 6 - нажать на «Начало»).

### Cтенд для выкладывания текущей релизной версии с ветки Release

**Адрес сервера:** release-elma3.prj.elewise.com

**Периодичность выкладывания**: после объявления о выходе новой \ исправленной релизной версии

**Тестовые веб-сайты, запущенные на IIS:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Имя сайта в IIS** | **Порт** | **СУБД** | **Физический путь** |
| ELMA3-Standart  Учетная запись:  LocalSystem | 80  7100 | MS SQL  База:  ELMA3 | C:\ELMA3-Standart\ |
| ELMA3-Express  Учетная запись:  LocalSystem | 90  7190 | FireBird  Порт:  3056 | C:\ELMA3-Express\ |

Для обновления необходимо проделать следующее:

1. Войти через терминал-клиент на сервер **release-elma3.prj.elewise.com**. Входить нужно под своим доменным пользователем (как в почту или джиру)
2. Скачать с сервера сборок (**build.elewise.com**) инсталлятор последней релизной версии системы: <http://build.elewise.com/searchResults.html?query=t%3ARelease%20t%3ALast%20p%3A(ELMA%203%20Release)%20c%3A(ELMA%203%20Release%20Setup)&buildTypeId=&byTime=true>
3. Запустить инсталлятор и следовать инструкциям, параметры установки смотреть в табличке выше
4. Проверить обновленный сайт, дать комментарий в инфо канал с сообщением о выходе версии.

//TODO: выкладывать должен **один** ответственный с отдела тестирования

### Cтенд для выкладывания онлайн-демо (demo.elma-bpm.ru) с ветки Release

**Адрес сервера**: demo.elma-bpm.ru; авторизация доменная (домен EleWise)

**Периодичность выкладывания**: после объявления о выходе новой \ исправленной релизной версии

**Веб-сайт, запущенный на IIS:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Имя сайта в IIS** | **Порт** | **СУБД** | **Физический путь до веб-сайта** |
| Elma3Demo | 80 | Firebird | C:\Elma3-Demo\Web |

Процедура обновления аналогична тестовым серверам. Для обновления нужно:

1. Войти через терминал-клиент на сервер **demo.elma-bpm.ru**. Логин, пароль – доменные (EleWise\логин, пароль).
2. Скачать с сервера сборок (**build.elewise.com**) архив с веб-частью (Release/ ELMA 3 DemoOnline): <http://build.elewise.com/viewType.html?buildTypeId=bt149&tab=buildTypeStatusDiv>.
3. Распаковать содержимое архива в какую-либо папку рядом с установленным веб-сайтом (например, **C:\Elma3-Demo\Web.NEW**). **Распаковывать нужно через Far Manager** (не через Проводник).
4. Скопировать файл **connection.config** из родительской папки веб-сайта (C:\Elma3-Demo\connection.config) в папку из пункта 3 (такой файл уже будет существовать в этой папке, его нужно заменить).
5. Открыть консоль управления IIS.
6. Остановить веб-сайт и соответствующий ему пул приложений (**Elma3Demo**).
7. Удалить папку со старым веб-сайтом **C:\Elma3-Demo\Web**, а затем переименовать **C:\Elma3-Demo\Web.NEW** в **C:\Elma3-Demo\Web**.
8. Обновить шаблон чистой базы (**пункт не обязателен при обновлении на версию с исправлениями, не касающимися базы демо-версии**). Для этого нужно:

* Установить на любую тестовую машину демо-версию из инсталлятора (<http://build.elewise.com/viewType.html?buildTypeId=bt144&tab=buildTypeStatusDiv>);
* После установки запустить сервер и дождаться завершения запуска (т.е. чтобы в браузере отобразилась главная страница);
* Остановить сервер демо-версии и скопировать базу данных (по умолчанию C:\ELMA3-FreeDemo\UserConfigs\Demo\BASE.FDB) на сервер под именем «**C:\Elma3-Demo\Conf\base\_clean.FDB**».

1. Запустить файл **C:\Elma3-Demo\update.demo.bat от имени Администратора** (нажать в Проводнике на файле правой кнопкой мыши и нажать на «Запустить от имени администратора»). Этот файл выполнит сброс базы и запустит пул приложений.
2. Запустить веб-сайт.
3. Проверить работоспособность сайта <http://demo.elma-bpm.ru>, дать комментарий в инфо канал с сообщением об обновлении демо-онлайн.

//TODO: выкладывать должен **один** ответственный с отдела тестирования

## Работа с запросами в системе JIRA

Основной документ регламента лежит в Элме по ссылке: <http://elma.elewise.com/Documents/Document/View/17317>

**Далее идет краткая выдержка того, что обязательно нужно понимать для выпуска версии.**

Вот перечень типов запросов в проекте ELMA3:

|  |  |
| --- | --- |
| New Feature Фича | Функция в минорной версии. Групповая задача – объединяет в себе **Задачи** и **Ошибки R&D** |
| Feedback Обратная связь | Предложение по улучшению текущей версии |
| Task Задача | Задача на изменение функционала ELMA или по созданию продукта ELMA (Справка, инсталлятор и т.д.) либо **вопрос**. |
| Order Заказная разработка | Групповая задача по заказной разработке |
| Fail Залет | Ошибки, найденные не во время тестирования R&D – как правило, это ошибки, которые находятся клиентами, партнерами и нами самими в процессе работы. |
| Bug Ошибка QA | Все ошибки, которые были найдены на этапе тестирования R&D или отделом QA. |

**Дополнительные поля:**

Для всех типов запросов введено дополнительное поле «**Результат**» типа текст. В это поле при отправке на тестирование, решении запроса или закрытии необходимо обязательно писать, что было сделано по итогам запроса. Эта информация дальше передается партнерам для понимания ситуации с обновлением.

У Фичи и Заказной разработки стоит обращать внимание на дополнительные поля (эти поля есть и у остальных типов, но там они менее важны):

1. Ссылка на ТЗ – ссылка на документ с ТЗ
2. Постановщик задачи – кто писал ТЗ
3. Внутренний заказчик – менеджер проекта
4. Куратор – Сейл, который продал проект
5. Разработчик – кто делает
6. Тестировщик – кто проверяет.
7. Внешний заказчик – название внешней компании-заказчика

Добавлено 2 перехода в цикл запроса (использовать для Фичи, Заказной разработки и сложных запросов длительностью более 2-х дней):

* 1. **Проверить на стенде** – данный переход означает, что необходимо проверить реализованные требования (именно требования \ концепцию), при этом необходимо предварительно развернуть готовую версию на соответствующем стенде. Проверку осуществляет внутренний заказчик или куратор.
  2. **Быстрая проверка** – данный переход означает, что необходимо проверить реализованные требования (именно требования \ концепцию), при этом проверять нужно прямо у разработчика. Обычно это полезно делать на ранних этапах, чтобы быстро получить коррекцию требований. Проверку осуществляет внутренний заказчик или куратор.

**Примечание по тестированию**:

Не надо бездумно отправлять в отдел QA. Над задачей работают 3 человека как минимум (за исключением, пожалуй, ошибок):

* Внутренний заказчик
* Исполнитель (разработчик)
* Специалист отдела QA

Так вот QA отвечает за функционирование того, что уже написано. Но реально хорошее качество скорее заложено в правильной постановке задачи и в правильном понимании задачи исполнителем. Поэтому зачастую гораздо полезнее сгонять задачу на Проверку требований (Check requirements), чем впустую заставлять молотить отдел тестирования. Они в этот период времени могут заняться чем-то более для качества полезным.

После прохождения запросом всех этапов тестирования он должен вернуться к разработчику со статусом Тест пройден (Test done), разработчик в свою очередь принимает решение о слиянии изменений в другие ветки и отмечает задачу как Решенную (Resolved).

**Дашборд в джире:**

В джиру добавлен дашборд ELMA3 – План: <http://jira4.elewise.com/secure/Dashboard.jspa?selectPageId=10500>

На этом дашборде собрана информация по текущей версии в разработке – **Dev,** и по планам на следующую версию – **DevNext.** Добавьте дашборд к себе в избранное и отслеживайте свои запросы по версии.

**Выпуск версии и статусы запросов:**

**Важно помнить**, что **в момент выпуска у нас** **не должно оставаться открытых запросов**, т.е. могут быть только «На тестировании» и «Тест пройден». Для этого необходимо заранее отслеживать свою загрузку и уведомлять своего тимлида, если есть вероятность выбиться из графика.

Если вам в последний момент возвращают что-то со статусом «Тест провален», при этом вы не видите объективно быстрого пути все исправить – также обращайтесь к тимлиду для принятия решения о переносе на следующую версию.