# Регламент работы с Jira в проекте ELMA3

Все запросы по разработке ELMA3 и заказной разработке на базе ELMA3 ведутся в системе Jira4 (<http://jira4.elewise.com>). Это система написана программистами ТОЛЬКО для программистов – обычные смертные ее не понимают.

Данный документ – это «Jira для чайников» (к которым я отношу и себя) и документ описывает правила игры.

**Наша цель** – всегда уметь поставить запрос и получить на него своевременный ответ без участия Директора в этом замечательном и интересном процессе ☺ По сути – Jira – это система управления бизнес-процессами **разработки софта**, да еще и с учетом ресурсов сразу. Красота, давайте пользоваться!

## Типы запросов

Вот перечень типов запросов в проекте ELMA3.

1. Фича
2. Обратная связь
3. Задача
4. Заказная разработка
5. Залет
6. Ошибка QA

## Кейзы

// todo – сделать перечень типовых ситуаций и рекомендацией по этим типовым ситуациям. Сделать это надо на обучении вместе с сейлами и внедренцами.

## Резолюции (Resolutions)

В таблице описаны резолюции в проекте ELMA3. Резолюция – это то, как вопрос разрешен. Запрос может быть отклонен, либо принят. По сути – если Fixed, то это значит сделано – все остальное – это разные типы отмаза почему не сделано. Смотрим сюда внимательно. Т.е. если запрос закрыт, это совершенно не значит, что он исправлен – он может быть под разным предлогом быть просто отфутболен.

|  |  |  |  |
| --- | --- | --- | --- |
| **Резолюция** | **По-русски (в письмах)** | **Английский коммент** | **Чёй то (по русски) – перевод с программистского на русский** |
| Fixed | Решен | A fix for this issue is checked into the tree and tested. | Программеры сделали.  Если **Fixed** и при этом **Resolved** (статус) – это значит, что Вы можете посмотреть это на dev-стенде  Если **Fixed** и при этом **Closed** (статус) – это значит что Вы можете это исправление уже видеть в релизном бинарнике или в кастомном модуле (в зависимости от того, что делаете) |
| Won't Fix | Не решен | The problem described is an issue which will never be fixed. | Не решен и не будет решен.  Почему – ХЗ. Если Вам реально надо и непонятно почему нельзя – переоткрывайте запрос (Reopen).  Если опять вернули, а вам ну очень надо – ставьте на Директора (Reopen). |
| Duplicate | Дубликат | The problem is a duplicate of an existing issue. | Вам говорят, что это уже было и давайте что бы не плодить - закроем… Блин, бывает. Только вот чего – обязательно при таком ответе Вам должно быть написан номер Issue где смотреть подобный запрос.  Сделайте простую вещь – скажите Watch на тот запрос. И когда его решат – вы получите всю информацию. |
| Incomplete | Не описана полностью | The problem is not completely described. | Обычно значит что мне некогда и я не понял. Подходим, объясняем. И просим не закрывать в след. раз, а задавать вопрос если что-то непонятно.  Если крики «вы не умеете писать» - спрашиваем «дайте нам шаблон». Криков сразу стает меньше, ибо написать шаблон – непосильная задача ☺ |
| Cannot Reproduce | Нельзя воспроизвести | All attempts at reproducing this issue failed, or not enough information was available to reproduce the issue. Reading the code produces no clues as to why this behavior would occur. If more information appears later, please reopen the issue. | А вот здесь внимательно! Программист не должен тратить свое время на воспроизводство ошибки. Это Ваша задача.  Поэтому если обоснованно – виноваты сами. Воспроизводим и отдаем в нормальном виде. |
|  |  |  |  |

## Типы запросов в ELMA3

Ниже перечислены типы запросов, которые есть в проекте ELMA3. Других запросов нет – пользуйтесь ими.

Для «нетехнарей»: читайте про типы «Обратная связь» и «Залеты». Остальное не обязательно для вас.

|  |  |  |
| --- | --- | --- |
| **Что это** | **На кого ставится** | **Кто и когда может ставить** |
| **Фича**  Функция в минорной версии. Групповая задача – объединяет в себе **Задачи** и **Ошибки R&D**.  Dashboard: ELMA3 - План | Исполнитель по этой фиче (отдел разработки) | **Технический директор ELMA**  **Директор ELMA**  Данные функции ставятся во время планирования. Финальный список фич на версию определяется 15-го числа. |
| Операции |  |  |
| **Создание** | 1. **Обязательно**: должно быть ссылка на ТЗ 2. Если фича растянута на несколько месяцев    1. В случае если мы решили выпускать итерации, то делаем следующим образом:       1. Атрибуты:          1. Тема = ФИЧА\_ИМЯ (Итерация 1).             1. Если данная итерация не пойдет в релиз, то добавляем:             2. Тема = [NR] ФИЧА\_ИМЯ (Итерация 1) - это значит Not Release          2. Версия = все версии, начиная с текущей Dev-версии до версии, в которой выходит Итерация 1          3. В ТЗ четко описано что в Итерации 1 должно быть по этой фиче       2. После выхода версии мы редактируем запрос:          1. Тема = ФИЧА\_ИМЯ (Итерация 2). Если фича выходит в этой версии, то убираем префикс [NR]          2. Версия = меняем на версию, где выходит итерация    2. В случае, если итераций не будет – а будет все и сразу и потом, то:       1. Атрибуты:          1. Тема = [NR] ФИЧА\_ИМЯ – это значит Not Release          2. Версии = все версии, где будет производится работа до версии где эта фича выходит 3. Другие атрибуты:    1. Постановщик задачи – заполнить, когда будет понятно    2. Внутренний заказчик – заполнить, когда будет понятно    3. Куратор – заполнить, когда будет понятно    4. Разработчик – заполнить, когда будет понятно    5. Тестировщик – заполнить, когда будет понятно    6. Внешний заказчик – заполнить, если фича инициирована внешним заказчиком 4. Watch лист: Добавляйте сюда всех, кто имеет отношение к данной фиче (роли выше уже будут получать оповещения). В этом случае фича превратится у вас в мини-форум по задаче и все будет в одном месте – что крайне удобно ! | |
| **Релиз итерации по фиче** | Если в данной минорной версии выходит итерация Фичи, то с одной стороны – надо продолжить данную фичу и сохранить накопленную ранее информацию, с другой стороны надо ведь как-то сказать что итерация 1 выпущена !  Для этого надо использовать операцию Clone. Регламент такой:   1. Делаем Clone – реально переносится все, только почему-то не переносятся комменты… 2. Закрываем клон (Resolve Issue Fixed, Close Issue) 3. Оригинал переименовываем и продолжаем с ним работать   Такой ход дает нам возможность в списке закрытых запросов видеть эту фичу (реально клон фичи) и продолжать работать по разработке фичи с накопленной информацией в следующей итерации. | |
|  |  |  |
|  |  |  |
| **Обратная связь**  Предложение по улучшению текущей версии  Dashboard: ELMA3 - Обратка | **Начальник отдела поддержки и заказной разработки** (Пономарев Александр)  Саша может изменить приоритет запроса. После определения приоритета запроса и целесообразности принимает решение и ставит на исполнителя в соответствии с регламентом Технической поддержки.  Регламент: 2 рабочих дня на принятие решения. | **Любой сотрудник компании**  Сотрудник отслеживает ответ Начальника отдела поддержки и заказной разработки, отслеживает на какую версию был поставлен его запрос. |
| Блокер | Данный тип запроса создается для **Критических** запросов (см. регламент Технической поддержки)  Примечание: Это может быть запрос типа – сделайте и мы точно купим в этом месяце! Но желательно в этом случае все-таки договариваться на выход в конце месяца и ставить тип запроса **Высокий** (см. следующий пункт). | Исправляется в Релизной версии (выходит быстро) |
| Высокий | Данный тип запроса создается для запросов с приоритетом **Высокий** (см. регламент Технической поддержки)  Примечание: Это может быть запрос типа – сделайте в этом месяце и мы точно купим. | Исправляется в Dev-версии (выходит в конце текущего месяца) |
| Средний | Данный тип запроса создается для запросов с приоритетом **Нормальный** (см. регламент Технической поддержки)  Примечание: Это приоритет по умолчанию. Этот приоритет нужен для сбора информации в течение месяца для планирования и улучшений на следующие периоды. | Подобные запросы выносятся на совещание 5-го числа на обсуждение.  **5-го числа**  Мы делаем одну из 2-ух вещей со ВСЕМИ поступившими запросами:   1. Закрываем запрос 2. Меняем статус запроса. Ставим статус «На рассмотрении».   Результат 5-го числа: Список входящих запросов пуст.  **15-го числа**  ВСЕ запросы в статусе «На рассмотрении»   * Либо закрываем с аргументацией * Либо планируем на версию (с указанием версии) и меняем статус на статус «В работе»   Результат 15-го числа: список «На рассмотрении» пуст |
| Операции |  |  |
| **Создание** | Если обратка пришла из Support, то **обязательно** в теле запроса поставить ссылку на запрос в Support’е в поле **Окружение**.  Основное замечание: я прошу - пишите так, что бы Ваш сосед понял это через 2 недели. Поменьше эпитетов, побольше дела.  Если хотите что бы кто-то глянул – скиньте ему в скайпе ссылку на ваш запрос, либо в канале забросьте если это гипер-важно. Если хотите протолкнуть 15-го числа в план – надо начинать обсуждение этой обратки в течение месяца.  На совещании очень сложно на самом деле быстро понять целесообразность - поэтому лучше растягивать во времени удовольствие. | |
| **Осуждение** | Если человеку это интересно – он может сделать 2 вещи:   1. Сказать Vote 2. Подписать а получение всех комментариев и действий по этой задаче (Watch) 3. Обмен мыслями как и в ELMA идет через комментарии   Каждый подписывается на то, что интересно ему. Соответственно, чем больше у вашего запроса Watche’ров – тем больше вероятность, что 15-го протащим ваше предложение. | |
|  |  |  |
|  |  |  |
| **Задача**  Задача на изменение функционала ELMA или по созданию продукта ELMA (Справка, инсталлятор и т.д.) либо **вопрос**. Ставится сотруднику отдела разработки ELMA.  Dashboard: не надо. «Непрофессиональные» пользователи джира будут пользоваться страничкой «Мои задачи». | **Исполнитель по этой задаче**  Исполнитель получает задачу, выполняет задачу. | **Сотрудник отдела разработки**  **Менеджер проекта**  **Сотрудник отдела внедрения**  **Директор** |
|  |  |  |
| Операции |  |  |
| Создание | Если вы просто хотите задать вопрос, **то обязательно** тема должна выглядеть следующим образом:   * Тема = [Вопрос] СУТЬ\_ВОПРОСА   Для вопросов еще полезно использовать **Под-задачи.** Это такие «обрезанные» задачи которые живут только внутри родительской задачи. | |
|  |  |  |
|  |  |  |
| **Заказная разработка**  Групповая задача по заказной разработке  // todo apon сделать dashboard для себя и внутренних заказчиков | Исполнитель, если определен  Саша Пономарев – если не определен | Внутренний заказчик по этой заказной разработке |
|  |  |  |
| Операции |  |  |
| Создание | Особенности:   1. **Обязательно**: должно быть ТЗ, в теле запроса – ссылка на ТЗ. 2. Тема = [НАЗВАНИЕ\_ПРОЕКТА] ЧТО\_НАДО\_СДЕЛАТЬ 3. Другие атрибуты:    1. Постановщик задачи – кто писал ТЗ    2. Внутренний заказчик – менеджер проекта    3. Куратор – Сейл, который продал проект    4. Разработчик – кто делает    5. Тестировщик – заполнить, когда будет понятно. ОЧЕНЬ рекомендую сразу просить выделить человека и заранее прописать его сюда.    6. Внешний заказчик – Название компании-заказчика 4. Watch лист: Добавляйте сюда всех, кто имеет отношение к данному запросу (роли выше уже будут получать оповещения). В этом случае запрос превратится у вас в мини-форум по задаче и все будет в одном месте – что крайне удобно! 5. Ставим Компонент = Заказная разработка 6. Версия = пусто (ничего не указываем) | |
| Создание задачи по Заказной разработке | Действия:   1. Создаем обычную задачу 2. Ставим Компонент = Заказная разработка 3. Версия = пусто 4. Линуем с задачей типа «Заказная разработка» | |
|  |  |  |
|  |  |  |
| **Залет**  Ошибки, найденные не во время тестирования R&D – как правило это ошибки, которые находятся клиентами, партнерами и нами самими в процессе работы. Это «залеты» отдела QA и разработки - точнее не сказать.  Dashboard: ELMA3 – Залеты | **Начальник отдела поддержки и заказной разработки** (Пономарев Александр)  Может изменить приоритет запроса. После определения приоритета запроса и целесообразности принимает решение и ставит на исполнителя в соответствии с регламентом Технической поддержки.  Регламент: 2 рабочих дня на принятие решения. | **Любой сотрудник компании**  Сотрудник отслеживает ответ Начальника отдела поддержки и заказной разработки, отслеживает на какую версию был поставлен его запрос. |
| Блокер | Данный тип ошибки создается для **Критических** запросов (см. регламент Технической поддержки) | Исправляется в Релизной версии (выходит быстро) |
| Высокий | Данный тип ошибки создается для запросов с приоритетом **Высокий** (см. регламент Технической поддержки) | Исправляется в Dev-версии (выходит в конце текущего месяца) |
| Средний | Данный тип ошибки создается для запросов с приоритетом **Нормальный** (см. регламент Технической поддержки) | Исправляется либо в Dev-(выходит в конце текущего месяца) либо в версии DevNext (выходит в конце следующего месяца) |
|  |  |  |
| Операции |  |  |
| Создание | Если залет пришел из support’а, то **обязательно** проставьте ссылку на запрос в support’е. | |
|  |  |  |
|  |  |  |
| **Ошибка QA**  Все ошибки, которые были найдены на этапе тестирования R&D или отделом QA.  Отличие от **Ошибки** в том, что это замечания, которые мы нашли  Dashboard: ELMA3 - QA | Исполнитель по тестируемой задаче. Если это регрессионное тестирование – то ставим пользователя по умолчанию | Все время:   * Отдел QA * Заказчик Фичи   C 20-ое по 0-ое:   * Отдел R&D * Директор * Заказчик фичи * Внедренцы   Обязательно проставляется поле «Проявляется в версиях» - ставится именно та минорная версия, которая тестируется.  Желательно **Ошибку QA** привязывать к **Задаче** или **Фиче** или **Обратной связи**, которая тестируется. |
| Регламент расстановки приоритетов | Запрос имеет уровень приоритета, который показывает его важность. Определенные в данный момент приоритеты представлены ниже.  Приоритет выставляет инициатором. В дальнейшем может уточняться:   1. Начальником отдела QA 2. Директором 3. Начальником отдела разработки 4. Начальником отдела технической поддержки  |  |  | | --- | --- | | **Приоритет** | **Описание** | | http://jira.teamlead.ru/images/icons/priority_blocker.gif Блокер | Блокирующая ошибка, приводящая приложение в нерабочее состояние, в результате которого дальнейшая работа с тестируемой системой или ее ключевыми функциями становится невозможна. Решение проблемы необходимо для дальнейшего функционирования системы. | | http://jira.teamlead.ru/images/icons/priority_critical.gif Высокий | Важная ошибка, неправильно работающая ключевая бизнес логика, дыра в системе безопасности, проблема, приведшая к временному падению сервера или приводящая в нерабочее состояние некоторую часть системы, без возможности решения проблемы, используя другие входные точки. Решение проблемы необходимо для дальнейшей работы с ключевыми функциями тестируемой системой. | | http://jira.teamlead.ru/images/icons/priority_major.gif Средний | Ошибка связана с важным функционалом приложения. | | http://jira.teamlead.ru/images/icons/priority_minor.gif Низкий | Незначительная ошибка, не нарушающая бизнес логику тестируемой части приложения, очевидная проблема пользовательского интерфейса | | http://jira.teamlead.ru/images/icons/priority_trivial.gif Тривиальный | Тривиальная ошибка, не касающаяся бизнес логики приложения, плохо воспроизводимая проблема, малозаметная по средствам пользовательского интерфейса, проблема сторонних библиотек или сервисов, проблема, не оказывающая никакого влияния на общее качество продукта.  Опечатки. | | |
| Операции |  |  |
| **Создание** | Особенности:   1. Тема:    1. Если это не ошибка, а улучшение функционала – ставим тему следующим образом: [Улучшение] \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_    2. Если это ошибка по заказной разработке, то ставим тему следующим образом: [НАЗВАНИЕ\_ПРОЕКТА] \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_    3. Если ошибка – то ставим как обычно, без префикса 2. Линкуем запрос к запросу, который тестируем в данный момент. Если ведется регрессионное тестирование, то не линкуем. 3. Проставляем поле «Проявляется в версиях».    1. Если это регрессионное тестирование и мы тестируем версию Release – то и ставим эту версию (которая уже выпущена).    2. Если это R&D тестирование:       1. Ставим ту версию, на которую запланирован выход задачи (это поле «Исправить в версиях». Если здесь несколько версий – то ставим последнюю. Цель – быстро получить список багов в том, что мы собираемся выпускать.    3. Если это заказная разработка – то ставим версию «Custom Development» | |
|  |  | |

## Диаграмма состояний

Запрос каждый момент времени находится в каком-то состоянии. Диаграмма состояний показывает, какие переходы между состояниями возможны и кто может осуществлять данные переходы. К сожалению, в Jira4 нет нормальных бизнес-процессов, поэтому приходится регламентировать переходы.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Статус** | **Переход** | **Конечный статус** | **Кого надо ставить исполнителем** | **Кто делает переход** | **Комментарий к действию (переходу)** |
| Open (**Открыт)** | Начальный статус для запроса. Это значит, он просто есть и активен. | | | | |
| Test Issue | Testing (**На тестировании**) | Сотрудник QA или Вика (если поле Тестировщик явно не определено) | Исполнитель | Отправка запроса на тестирование.  Если запрос падет на Вику и она переназначает на тестировщика, то Вика руками проставляет поле «Тестировщик».  Внимание: читаем Примечание 1 внизу таблицы. |
| *Review Issue* | *-* | *-* |  | *-* |
| Start Progress | In Progress (**В работе**) | Исполнитель | Исполнитель | Если не лень |
| Resolve Issue | Resolved (**Разрешен**) | Автоматом ставится на того, кто выполняет операцию Resolve Issue | Team Lead или Технический директор | Что должно быть:   1. Должна быть установлена резолюция 2. Если Fixed, то должны быть проставлены версии, в которых этот фикс появится. Это могут быть текущая релизная, текущая dev-версия, либо Custom, если ошибка в нем.   Данное действие делается либо тим лидом, либо техническим директором (на кого пришло с тестирования). Цель: тим лид должен четко понимать что закрывает в своей зоне ответсвенности. |
| Close Issue | Closed (**Закрыт**) | Автоматом ставится на того, кто выполняет операцию Close Issue | Технический директор или Начальник отдела поддержки | Данную операцию может выполнить только **Технический директор** или **Начальник отдела поддержки**.  Должна быть установлена резолюция.  Если резолюция = Fixed, то в данную стадию попасть нельзя – только через статус Resolved (Closed).  Т.е. данный переход только для отмазок и только для некоторых. |
|  | Check Online | Checking Online (**Проверяем на коленке**) | Внутренний заказчик запроса | Разработчик по запросу | Отправка на проверку на коленке. См. Приложение 1. |
|  |  |  |  |  |  |
| In Progress (**В работе**) | Данный статус показывает, что сейчас ведется работа над запросом. Реально работа может вестись и в статусе Open (**Открыт**) | | | | |
|  | Stop Progress | Open (**Открыт**) | Исполнитель | Исполнитель | Если вдруг зачем-то прекратил работу |
|  | *Review Issue* | *-* | *-* | *-* | *-* |
|  | Test Issue | Testing (**На тестировании**) | -/- | -/- | То же что и для Open (**Открыт**) |
|  | Resolve Issue | Resolved (**Разрешен**) | -/- | -/- | То же что и для Open (**Открыт**) |
|  | Close Issue | Closed (**Закрыт**) | -/- | -/- | То же что и для Open (**Открыт**) |
|  |  |  |  |  |  |
| Resolved (**Разрешен**) | **Вопрос разрешен**. Это значит, что он протестирован отделом QA и выложен на dev-стенд. Т.е. его можно глянуть и «пощупать». Если при «ощупывании» видите что лажа – бейте тревогу через операцию Reopen ! | | | | |
|  | Close Issue | Closed (**Закрыт**) | Автоматом ставится на того, кто выполняет операцию закрытия | Технический директор | Данная операция осуществляется **Техническим директором** 0-го числа месяца. С помощью данной операции формируется список запросов, которые решены в выпускаемой версии.  Итак, каждый месяц надо делать следующее:   1. Выходит бинарник 2. Антон переводит запросы в состояние Closed 3. На странице Release становится доступен список того, что было закрыто в   Выходит бинарник, закрываются запросы.  Таким образом, мы формируем на 100% верный список того, что бы в версии выпущено. Критерии:   1. Closed 2. Fixed 3. Исправлено в версии |
|  | Reopen Issue | Reopened (**Переоткрыт**) | Исполнитель  Так как запрос на том, кто сделал операцию Resolve, то и переоткрытие пойдет на того, кто зарезолвил | Внутренний заказчик | Resolved – это значит выложено. Значит можно смотреть. Смотрит, как правило, на этом этапе Внутренний заказчик. Замечает косяк. Открывает заново. |
|  | Check | Checking (**На проверке**) | Внутренний заказчик запроса | Разработчик по запросу | Когда все выложено – отправляем на проверку концепции |
|  |  |  |  |  |  |
| Reopened (**Переоткрыт**) | По какой-то причине запрос переоткрыли. Все переходы работают точно также как для состояния Open (**Открыт**). | | | | |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
| Closed (**Закрыт**) | Запрос закрыт, выпущен в релиз, в бинарниках уже давно. | | | | |
|  | Reopen Issue | Reopened (**Переоткрыт**) | Исполнитель  Тот, кто закрыл запрос | Только Технический директор | Это делает в Критическом случае. И только техническим директором.  Если что-то «отвалилось» - создайте новый запрос и сошлитесь на старый. |
|  |  |  |  |  |  |
| Testing (**На тестировании**) | Запрос находится в отделе QA. У кого – смотри текущего исполнителя. | | | | |
|  | Test Fail | Test Failed (**Тест провален**) | Разработчик по запросу | Отдел QA |  |
|  | Test Done | Test Done (**Тест пройден**) | Разработчик по запросу | Отдел QA |  |
|  |  |  |  |  |  |
| Test failed (**Тест провален**) | Что-то пошло не так | | | | |
|  | Reopen Issue | Reopened (Переоткрыт) | Исполнитель | Исполнитель | Это делает сам разработчик |
|  |  |  |  |  |  |
| Test done (**Тест пройден**) | Тестеры ничего не нашли. Оно там конечно есть, просто не нашли ☺ | | | | |
|  | Resolve Issue | Resolved (Разрешен) | Автоматом ставится на того, кто выполняет операцию Resolve Issue | Тим лид, технический директор | Сам разработчик не может закрыть запрос.  Он должен переназначить запрос на тим лида или технического директора (предпочтительнее тим лид) и уже тот Разрешает запрос. |
|  | Check | Checking (**На проверке**) | Внутренний заказчик запроса | Разработчик по запросу | Отправляем запрос на проверку концепции Внутреннему заказчику сразу после QA. |
|  |  |  |  |  |  |
| Checking Online (**Проверяем на коленке**) | Если запрос в этой стадии – значит Внутреннему заказчику надо пощелкать и понять так или не так сделано.  Внутренний заказчик должны проверить «на коленке» – правильно ли разработчик вас понял и правильно ли он реализовал, что он хотел.  Варианта 2:   1. Лучше всего подойти к разработчику и проверить на его компе то ли он сделал, что Вы хотели 2. Пощелкать на стенде, который он предоставит.    1. Вариант 1: Разработчик разворачивает на своем компе, дает Вам ссылку    2. Вариант 2: Разработчик выкладывает на сервер своего тима, дает Вам ссылку   На данном этапе не стоит смотреть на мелочи – запрос пойдет еще в отдел качества на проверку. Надо смотреть правильно ли он Вас концептуально понял и правильно ли концептуально реализовал. | | | | |
|  | Check Fail | Check Failed (**Тест провален**) | Разработчик запроса | Внутренний заказчик | Совсем не то |
|  | Check Done | Check Done (**Тест пройден**) | Разработчик запроса | Внутренний заказчик | Примерно то что надо |
|  |  |  |  |  |  |
| Checking (**На проверке**) | Запрос находится у внутреннего заказчика на проверке правильности реализации требований. Сюда запрос попадет через стадию Resolve – т.е. запрос реализован и выложен на dev-стенд.  Вы должны на dev-стенде пощелкать и проверить, правильно ли реализовано Ваше требование (запрос). Данное действие имеет смысл до выхода релиза. Потом уже поздно – релиз уже выйдет.  Во время выхода релиза все требования со статусом Checking будут принудительно закрыты.  Если запрос переведен в это состояние, это значит простую вещь:   1. Запрос сделан 2. Он протестирован разработчиком 3. Он протестирован сотрудником отдела QA 4. Выложен на dev-стенд   Вы можете гонять в хвост и в гриву. Тут уже не должно быть ошибок. Находим ошибку:   1. Фиксируем запрос Ошибка QA. Понятно, что фиксировать стоит не одну ошибку, а серий – скажите все, что думаете по этому поводу! 2. Привязываем ошибку к этому запросу 3. Возвращаем запрос со статусом Check Fail тому, кто перевел запрос в состояние Resolved (Смотрим историю) | | | | |
|  | Check Fail | Check Failed (**Тест провален**) | Тому, кто перевел запрос в состояние Resolved | Внутренний заказчик |  |
|  | Check Done | Check Done (**Тест пройден**) | Тому, кто перевел запрос в состояние Resolved либо на Технического директора | Внутренний заказчик |  |
|  |  |  |  |  |  |
| Check failed (**Проверка провалена**) | Заказывал качели, а сделали только веревочки от качелей | | | | |
|  | Reopen Issue | Reopened (Переоткрыт) | Исполнитель | Исполнитель | Это делает сам разработчик |
|  |  |  |  |  |  |
| Check done (**Проверка пройдена**) | Как-то еле-еле сделано | | | | |
|  | Продолжить | Open (**Открыт**) | Исполнитель (отдел разработки) | Внутренний заказчик | Только для Checking Online |
|  | Resolve Issue | Resolved (Разрешен) | Автоматом ставится на того, кто выполняет операцию Resolve Issue | Тим лид, технический директор | Сам разработчик не может закрыть запрос.  Он должен переназначить запрос на тим лида или технического директора (предпочтительнее тим лид) и уже тот Разрешает запрос.  Resolve делается для Проверок, которые идут до Resolve сразу после QA. |
|  |  |  |  |  |  |
| In Progress (**На рассмотрении**) | Только для запросов типа «**Обратная связь**». 5-го числа на совещании принято решение, что этот запрос стоит рассмотреть подробнее и принять решение 15-го числа | | | | |
|  | Move | Open (**Открыт**) | Исполнитель (отдел разработки) | Технический директор, Директор | 15-го числа совещание решает что, это гут. ТЗ сформировано – можно ставить в работу и в план.  Делается операция перемещения запроса, т.е. он превращается либо в **Фичу** либо в **Задачу** либо в **Ошибку**.  Указывается версия, исполнитель и понеслось. |
|  |  |  |  |  |  |

**Примечание 1**:

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.

## Dashboards

Мои задачи - <http://jira4.elewise.com/secure/Dashboard.jspa?selectPageId=10600>

ELMA3 - План - <http://jira4.elewise.com/secure/Dashboard.jspa?selectPageId=10500>

ELMA3 - Обратка - <http://jira4.elewise.com/secure/Dashboard.jspa?selectPageId=10700>

ELMA3 - Залеты - <http://jira4.elewise.com/secure/Dashboard.jspa?selectPageId=10701>

ELMA3 - QA - <http://jira4.elewise.com/secure/Dashboard.jspa?selectPageId=10702>

ELMA3 - Dev (3.0.2) - <http://jira4.elewise.com/secure/Dashboard.jspa?selectPageId=10703>

ELMA3 - Release (3.0.1) - <http://jira4.elewise.com/secure/Dashboard.jspa?selectPageId=10602>

Щелкните на ссылке и скажите «Add to favorites» - и эти вкладки появятся у вас слева и останутся там

## TODO

**Дальше читать не надо**

// todo – написать регламент работы с QA для типов запросов, минимизировать движения опой и мышкой.

// todo обсудить с Антоном - **Залет**, **Ошибка QA, Фича, Задача, Обратная связь** – они должны закрываться только при выпуске версии – так должны висеть в каком-то статусе типа исправлено. Но пока нет артефакта где это можно забрать – ошибка не закрыта.

// todo Ввести атрибуты Test passed и Check Passed что бы однозначно понимать в таблице что данные запросы прошли проверки. Это для «беглого» анализа очень хорошо.

// todo Итерация 2: Ввести атрибут Help update: Пусто, Quick Help (на итерации 20-0 положена быстрая глава справки), Да, Нет (Не требуется). По уму надо сделать фильтр по зарелезинным вещам, по которым ничего не проставлено. Суть – Ира по релизному стакану смотрит что изменилось и на каждый запрос ставит отметку что принято или не принято данное изменение во внимание в справке. При этом она указывает минорную версию, в которой это обновление справки выйдет. Т.е. надо сделать поле Helped At.

## ТЗ на Dashboards

В таблице внизу перечислены главные страницы, которые должны быть сделаны

|  |  |
| --- | --- |
| ELMA3 – План | Как есть, сделано путево |
| ELMA3 – ReleaseNotes | Надо превратить в ELMA3-Release (3.0.0) – см. требования ниже. |
| ELMA3 - Обратка | Цели и потребитель:   1. 5-го числа садимся и смотрим на одном экране все 2. 15-го финализируем 3. В течение месяца сюда заглядываем что бы понять что накопилось 4. Отслеживаем блокеры и критичные вещи – это все внутренние заказчики в течение месяца   Портлеты:   1. Приоритет «Средний». Данный тип запроса 5-го числа рассматривается    1. Все открытые запросы по ELMA (Входящий стакан)       1. Тип: Обратная связь       2. Приоритет: Средний       3. Статус: Активен    2. Запросы «На рассмотрении» (с 5 по 15-ое здесь живут запросы)       1. Тип: Обратная связь       2. Приоритет: Средний       3. Статус: Активен       4. Статус запроса: в работе 2. Блокеры    1. Блокеры на рассмотрении – это у Саши на рассмотрении висят    2. Блокеры в работе – это в работе у программеров в release-версии находятся. Супер если будет видно когда завершена будет работа. 3. Высокие    1. Высокие на рассмотрении – это у Саши на рассмотрении висят    2. Высокие в работе – это у программеров в dev-версии в работе находятся |
| ELMA3 – Залеты | Цели и потребитель:   1. Потребитель: Директор 2. Цель: отслеживать качество работы отдела QA   Портлеты:   1. Блокер. Надо показывать в какой версии залет найден.    1. Активные Блокеры – Список активных блокеров в хронологическом порядке (последний наверху)       1. Дата создания       2. К какой версии       3. Кто исполнитель       4. Внешний заказчик    2. Все блокеры – Список блокеров в хронологическом порядке 2. Высокий. Надо показывать в какой версии залет найден.    1. Активные Высокие – Список активных высоких в хронологическом порядке (последний наверху)       1. Дата создания       2. К какой версии       3. Кто исполнитель       4. Внешний заказчик    2. Все Высокие – Список высоких в хронологическом порядке 3. Средний. Надо показывать в какой версии залет найден. Гут если удастся показать на какую версию запланировано исправление.    1. Активные Средние – Список активных средних в хронологическом порядке (последний наверху)       1. Дата создания       2. К какой версии       3. Версия в которой будет исправлено       4. Кто исполнитель       5. Внешний заказчик    2. Все Средние – Список средних в хронологическом порядке |
| ELMA3 - QA | Цели и потребитель:   1. Потребитель: Директор, разработка 2. Цель: отслеживать динамику работы отдела QA   Портлеты:   1. R&D тестирование    1. Активные Ошибки в Dev-версии    2. Все ошибки в Dev-версии 2. Регрессионное тестирование    1. Активные ошибки в Release-версии    2. Все ошибки в Release-версии 3. Custom Development    1. Активные ошибки в заказной разработке    2. Все ошибки в заказной разработке |
| Мои задачи | Цели и потребитель:   1. Потребитель: Все «нетехнари» 2. Цель: получить привычный интерфейс для работы с Jira4   Портлеты:   1. Задачи мне – текущие (слева, как в эльме - синим) 2. Задачи от меня – текущие (справа, как в эльме – зеленым)    1. Указать версию 3. Все задачи от меня – архив (справа внизу, как в эльме – темно-зеленым) |
| ELMA3 – Dev (3.0.1) | Это по версии 3.0.1, которая сейчас находится в разработке  Цели и потребитель:   1. Потребитель: Директор, технической директор, разработчики, консультанты. 2. Цель: быстро понять, что сделано в текущей Dev-версии, понять динамику. Что исправлено, что будет исправлено.   Портлеты:   1. Сделано в 3.0.1 – здесь должно быть все, что сделано и прошло выше статуса Resolved 2. Фичи сделанные в 3.0.1 – здесь должно быть все, что сделано и прошло QA 3. Текущие известные ошибки по 3.0.1 |
| ELMA3 – Release (3.0.0) | Это по версии 3.0.0 – по текущей релизной версии. По сути это единая карта – что выпущено, что в осталось в ошибках при выпуске версии 3.0.0  Данный DashBoard должен обновляться при выходе релиза.  Портлеты:   1. Выпущено в в 3.0.0    1. Все, что Closed+Fixed 2. Фичи выпущенные в 3.0.0    1. Все, что Closed+Fixed 3. Known bugs 3.0.0. Здесь надо показывать все не Closed ошибки, которые найдены в версиях 3.0.0 и младше. Как это сказать – ХЗ, надо подумать. Реально ошибки могут «тянуться» несколько версий, достаточно долго… |

## Custom fields

Необходимо ввести custom fields для более формализованного хранения информации. Поля для добавления помечены красным.

R – обязательное поле на форме создания

+ – должно быть на форме создания и редактирования

Если пусто – то поле необязательно должно быть на форме создания и редактирования

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Тип** |  | **Фича** | **Обратная связь** | **Задача** | **Заказная работка** | **Залет** | **Ошибка QA** | **Вопрос** |
| Автор | Пользователь, создавший запрос | R | R | R | R | R | R | R |
| Текущий исполнитель | Пользователь, на котором в текущий момент находится работа по запросу | R | R | R | R | R | R | R |
| Постановщик задачи | Пользователь, который пишет ТЗ по задаче  Тип: User | + |  |  | + |  |  |  |
| Внутренний заказчик | Пользователь, который полностью отвечает за реализацию данного запроса.  Тип: User | + |  |  | + |  |  |  |
| Куратор | Пользователь, который курьирует данный запрос. В случае, если внутренний заказчик недоступен (уехал) – выполняет его роль.  Тип: User | + |  |  | + |  |  |  |
| Разработчик | Ответственный разработчик за эту функцию  Тип: User | + |  |  | + |  |  |  |
| Тестировщик | Ответственный за тестирование данного запроса. Сотрудник отдела качества, который отвечал за тестирование данного запроса  Тип: User | + |  |  | + |  |  |  |
|  |  |  |  |  |  |  |  |  |
| Внешний заказчик | Сюда пишем внешнего заказчика  Тип: Строка |  | + |  | + | + |  |  |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |

## Компоненты

// todo – я предлагаю перейти к стандартной компонентной модели – по приложениям и дополнительным модулям. Если хотите в портале выделить отдельные части – ок, давайте. Но за основу взять модель приложений.

// todo сделать компонент Custom development