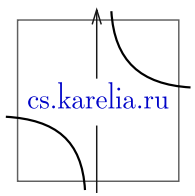
Петрозаводский государственный университет
Институт Математики и информационных технологий
Кафедра Информатики и математического обеспечения

Инструменты управления проектом.

Лектор:

к.т.н., доцент Богоявленский Ю. А.
ybgv@cs.karelia.ru



Разделы

- Система Управления проектами Jira
- Система Управления проектами Redmine

Система Управления проектами Jira. Содержание

Введение

При решении многосложных задач бывает момент, когда сотрудники и управляющие не могут видеть проект в целом, теряется из памяти необходимость сделать те или иные работы. Бывает и другая ситуация подобная этой — отдел, состоящий более чем из $15 - 20$ сотрудников работающих одновременно над одной задачей — сложная структура для управления, и без специальных средств проконтролировать их работу чрезвычайно сложно. Устаревший метод — бесконечные совещания и доклады — только усугубляет проблему, так как отвлекает от ведения главного дела.

Менеджеры всего мира ищут способы как не забыть что-то в потоке дел, при этом вовремя сообщить команде важные новости, поставить всем задачи, проследить за выполнением, отыскать и узкие места и принять меры, и, в конце концов, успешно завершить работу в срок. Для этого нужно интеллектуальное, удобное программное обеспечение, доступное всем, не отнимающее много усилий, простое и понятное и вместе с тем очень гибкое и легко настраиваемое.

JIRA. Общие сведения

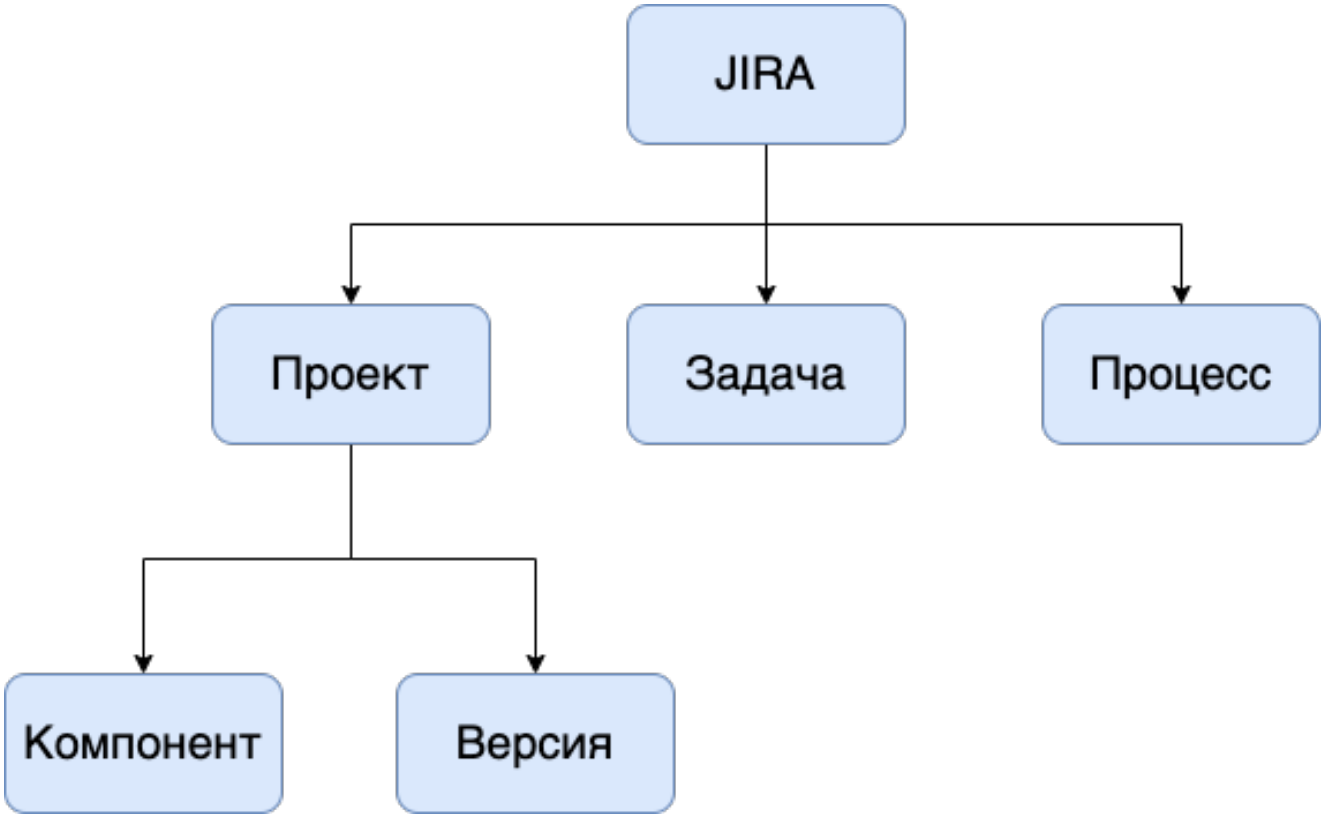
Одним из вариантов решения этой проблемы может стать **система управления проектами и задачами JIRA**.

JIRA — это средство отслеживания задач или управления проектами. Обычно она используется для отслеживания ошибок. JIRA была разработана австралийской компанией Atlassian. Первый выпуск JIRA произошёл в 2002 году.

Название «JIRA» выглядит как аббревиатура, но это не так. На самом деле, название системы получено путём усечения слова «Gojira» — японского имени монстра Годзилла, что, в свою очередь, является отсылкой к названию конкурирующего продукта — Bugzilla. JIRA создавалась в качестве замены Bugzilla и во многом повторяет её архитектуру. Система позволяет работать с несколькими проектами. Для каждого из проектов создаёт и ведёт схемы безопасности и схемы оповещения.

JIRA

Концептуальные понятия JIRA — **Задача**, **Проект** и **Процесс**. Это может быть проиллюстрировано следующим образом:



Задача JIRA

Фактически, все, что можно создать и отслеживать с помощью JIRA, можно назвать задачей

- **Ошибка** — задача посвященная устранению выявленных ошибок.
- **Улучшение** — улучшение существующих алгоритмов без изменения функциональности с точки зрения пользователя (задачи рефакторинга).
- **Новая функция** — тип предназначенный для регистрации задач по созданию новой функциональности.
- **Задача** — регистрация типовых задач, решаемых в ходе проекта.
- **Пользовательская задача** — тип предназначенный для регистрации работ, которые сотрудники планируют самостоятельно.

Задача JIRA

Атрибуты задачи

Задачи имеют следующие основные атрибуты:

Статус указывает ход выполнения проекта.

- **Открыта** — этот задача находится в начальном состоянии, готова для назначения на исполнителя.
- **В работе** — задача находится в реализации.
- **Решена**. Резолюция была определена или реализована, и эта задача ожидает проверки. Здесь проблемы либо “вновь открыты”, либо “закрыты”.
- **Переоткрыта** — эта задача была вновь переоткрыта после проверки (качество решения не соответствует требованиям).
- **Закрыта** — задача была проверена, задача выполнена с требуемым качеством, работы по задаче завершены.

Задача JIRA

Атрибуты задачи

Резолюции показывают, как можно закрыть задачу.

- **Исправлено** — было выполнено исправление продукта для решения этой задачи.
- **Не будет исправлено** — эта задача была закрыта поскольку больше не является актуальной (например, изменились требования).
- **Дубликат** — задача является дубликатом уже существующей задачи (в этом случае рекомендуется создать связь с дублируемой задачей).
- **Невыполнима** — недостаточно информации для решения этой задачи.
- **Невозможно воспроизвести** — условия описанные в задаче не могут быть воспроизведены в настоящее время (или недостаточно информации для их воспроизведения).

Задача JIRA

Атрибуты задачи

Приоритет определяет важность задачи по отношению к другим.

- **Наивысший** — наивысший приоритет. Задача является критически важной для проекта.
- **Высокий** — указывает, что эта задача вызывает проблемы и требует неотложного внимания.
- **Средний** — указывает, что эта задача имеет существенное значение.
- **Низкий** — указывает, что эта задача имеет относительно незначительное влияние на проект.
- **Самый низкий** — самый низкий приоритет.

JIRA процесс

Процесс (Workflow) JIRA — это набор статусов и переходов, через которые проходит ошибка/задача/проблема в течение её жизненного цикла. Фактически, он представляет внутренние процессы вашей организации.

Составные части процесса в JIRA

- **Статусы** определяют позиции задачи в текущем процессе.
- **Переходы** показывают переход задачи из одного статуса в другой.
- **Правопреемник** это лицо, ответственное за выполнение задачи.
- В **резолюциях** содержится объяснение перехода задачи из открытого состояния в закрытое.

JIRA процесс

Составные части процесса в JIRA

- **Условия** контролируют сотрудника, который может выполнить переход.
- **Валидаторы** проверяют, выполнен ли вход до перехода.
- **Свойства** используются для реализации ограничений на определенные шаги или переходы процесса.
- **Пост функции** позволяют выполнять любую дополнительную обработку, необходимую после выполнения перехода, например:
  - создание истории изменений задачи
  - создание события для отправки уведомлений по электронной почте
  - обновление полей задачи
  - добавление комментария к задаче

JIRA процесс

Процесс можно описать следующей схемой.

Проект JIRA

Проект JIRA — это набор задач. Основными атрибутами проекта являются:

- **Имя** выбирается администратором
- **Ключ** — это идентификатор, с которого начинаются все названия задач в рамках проекта
- **Компонент проекта** — это логическая группировка задач в рамках проекта
- Часто необходимо, чтобы конкретная проблема была связана с **версией** проекта. Существует три состояния версий:
  1. **выпущенные**
  2. **невыпущенные**
  3. **архивированные**

Отчеты и диаграммы в JIRA

JIRA позволяет создать карту проекта, просматривать загрузку каждого пользователя и делает многое другое для эффективного управления проектами. Также имеется целый ряд необходимых стандартных отчетов.

- нерешенные высокоприоритетные задачи
- количество задач созданных одним пользователем
- среднее время решения задачи
- отношение реального и заданного времени решения задач
- количество задач созданных в день, неделю, месяц, год
- задачи назначенные для указанной версии или этапа
- задачи имеющие определенные статус
- задачи имеющие определенный приоритет
- отчет по использованию времени
- отчет о затратах времени на выпуск определенной версии
- отчет о нагрузке на разработчиков

Отображение хода выполнения проектов в JIRA

JIRA позволяет управлять видом специальной стартовой страницы, называемой приборной панелью. На этой странице отображается ход выполнения проектов и имеются ссылки для быстрого доступа ко всем часто используемым функциям, отчетам и задачам:

- список задач назначенных вам
- список сохраненных фильтров
- статистика проекта
- статистика фильтра
- список проектов
- линейные графики
- объемные графики
- текстовые и HTML сообщения

Пакеты JIRA

В настоящее время основными пакетами JIRA являются:

- JIRA Core
- JIRA Software
- JIRA Service Desk

Основным пакетом является JIRA Core. Это оптимальное сочетание мощных возможностей для совместной работы с задачами в процессе или проекте.

Система Управления проектами Redmine. Содержание

- Введение.
- Функциональные возможности.
- Структура базы данных.
- Схема добавления и разработки нового проекта.
- Недостатки Redmine.
- Преимущества Redmine.

Введение

Redmine – открытое серверное веб-приложение для управления проектами и задачами (в том числе для отслеживания ошибок).

Проекты представляют из себя наборы задач. Сами проекты при этом могут иметь иерархическую структуру по аналогии с папками в компьютере: проект – подроект – подподпроект – задача. Смысловой единицей служит непосредственно задача. Проекты являются лишь способом группировки задач.

Для каждого проекта приложение позволяет создать свое описание, форум, контроль времени и несколько уровней доступа.

Redmine также включает в себя календарь и диаграмму Гантта, позволяющие визуализировать процесс разработки проектов и их дедлайны.

Также приложение включает в себя системы контроля версий и diff viewer (утилита, показывающая разницу между несколькими версиями одного и того же проекта).

Функциональные возможности

- ведение нескольких проектов;
- гибкая система доступа, основанная на ролях;
- система отслеживания ошибок;
- диаграммы Ганта и календарь;
- ведение новостей проекта, документов и управление файлами;
- оповещение об изменениях с помощью RSS-потоков и электронной почты;
- лёгкая интеграция с системами управления версиями (SVN, CVS, Git, Mercurial, Bazaar и Darcs);
- возможность самостоятельной регистрации новых пользователей;
- многоязычный интерфейс (в том числе русский);
- поддержка СУБД MySQL, Microsoft SQL Server[2], PostgreSQL, SQLite.

Структура базы данных

- Пользователи системы
  - Пользователи являются одним из центральных понятий предметной области. Модель пользователя является основой для идентификации и аутентификации работающего с системой персонала и клиентов, а также для авторизации их в разных ролях, проектах.
- Роли
  - Роли пользователей определяются гибкой моделью определения прав доступа пользователей. Роли включают в себя набор привилегий, позволяющих разграничивать доступ к различным функциям системы.
  - Пользователям назначается роль в каждом проекте, в котором он участвует, например, «менеджер в проекте по разработке сайта А», «разработчик в проекте по поддержанию интранета компании».

- Проекты
  - Проект является одним из основных понятий в предметной области систем управления проектами. Благодаря этой сущности возможно организовать совместную работу и планирование нескольких проектов одновременно с разграничением доступа различным пользователям.
- Трекеры
  - Трекеры являются основной классификацией, по которой сортируются задачи в проекте. Само по себе понятие «трекер» восходит к системам учёта ошибок, представлявшим каждая в отдельности один проект.

Роли и права доступа

Внутри системы существует возможность распределять права на использование функций системы для конкретных ролей. Роль – это просто именованный набор определенных прав доступа к системным функциям.

Пример

4 основные роли:
Участник, Исполнитель, Руководитель, Директор.

Суть этих ролей лишь в наборе прав внутри системы и они не являются полной аналогией фактических должностей.

Роли и права доступа

Права доступа	Исполнитель	Руководитель	Директор
Создание проекта			✓
Редактирование проектов			✓
Открывать проекты			✓
Выбор модулей проекта			✓
Управление участникам			✓
Управление версиям		✓	✓
Сохранение запросов	✓	✓	✓

Роли и права доступа

Таким образом, мы можем распределять внутри проектов возможности каждого пользователя путем присвоения ему соответствующей роли.

Если пользователь не включен в проект, то ему ни одна из задач внутри проекта не будет видна и он не сможет совершать никаких действий.

Если пользователь назначен Участником, то он будет видеть задачи, но набор его прав будет сильно ограничен. Суть этой роли в том, что любого пользователя системы, который включен в проект мы можем обозначить как наблюдателя при создании задачи. А система умеет рассылать все изменения в проекте или задаче по электронной почте либо создателю задачи, либо исполнителю, либо наблюдателю.

Задачи

- Задачи
  - Задачи являются центральным понятием всей системы, описывающим некую задачу, которую необходимо выполнить. У каждой задачи в обязательном порядке есть описание и автор, в обязательном порядке задача привязана к трекеру.
  - Каждая задача имеет статус. Статусы представляют собой отдельную сущность с возможностью определения прав на назначение статуса для различных ролей или определение актуальности задачи.
  - Для каждого проекта отдельно определяются набор этапов разработки и набор категорий задач. Среди других полей интересны также «оценённое время», служащее основой для построения управленческих диаграмм, а также поле выбора наблюдателей за задачей (см. «Получение уведомлений»). К задачам имеется возможность прикреплять файлы (имеется отдельная сущность «Приложение»).

Задачи

В системе задача являет собой набор полей для описания сути задачи (формализации). Есть системные поля и настраиваемые.

Системными полями являются такие поля, как

- трекер (список),
- исполнитель (список),
- тема (текстовое поле),
- описание задачи (текстовое поле),
- дата начала и дата окончания,
- родительская задача.

Также имеется возможность добавлять свои поля. Это могут быть как списки значений, текстовые поля, поле для ввода дат, список пользователей системы и т.п.

Задачи

- Отслеживание изменения параметров задач
  - За отслеживание изменений параметров задач пользователями в системе отвечают две сущности: «Запись журнала изменений» и «Изменённый параметр». Запись журнала отображает одно действие пользователя по редактированию параметров задачи и/или добавление комментария к ней. То есть служит одновременно инструментом ведения истории задачи и инструментом ведения диалога.
- Связи между задачами
  - Задачи могут быть взаимосвязаны: например, одна задача является подзадачей для другой или предшествовать ей. Эта информация может быть полезна в ходе планирования разработки программы, за её хранение в Redmine отвечает отдельная сущность.

- Учёт затраченного на проект времени
  - Система поддерживает учёт затраченного времени благодаря сущности «Затраченное время», связанной с пользователями и задачей. Сущность позволяет хранить затраченное время, вид деятельности пользователя (разработка, проектирование, поддержка) и краткий комментарий к работе. Эти данные могут быть использованы, например, для анализа вклада каждого участника в проект или для оценки фактической трудоемкости и стоимости разработки.
- Привязка репозиториев
  - Redmine предоставляет возможность интеграции с различными системами управления версиями (репозиториями). Интеграция заключается в отслеживании изменений во внешнем репозитории, их фиксации в базе данных, анализе изменений с целью их привязки к определённым задачам.
    В инфологической структуре системы за интеграцию с внешними репозиториями отвечают три сущности: Репозиторий, Редакция и Изменение.

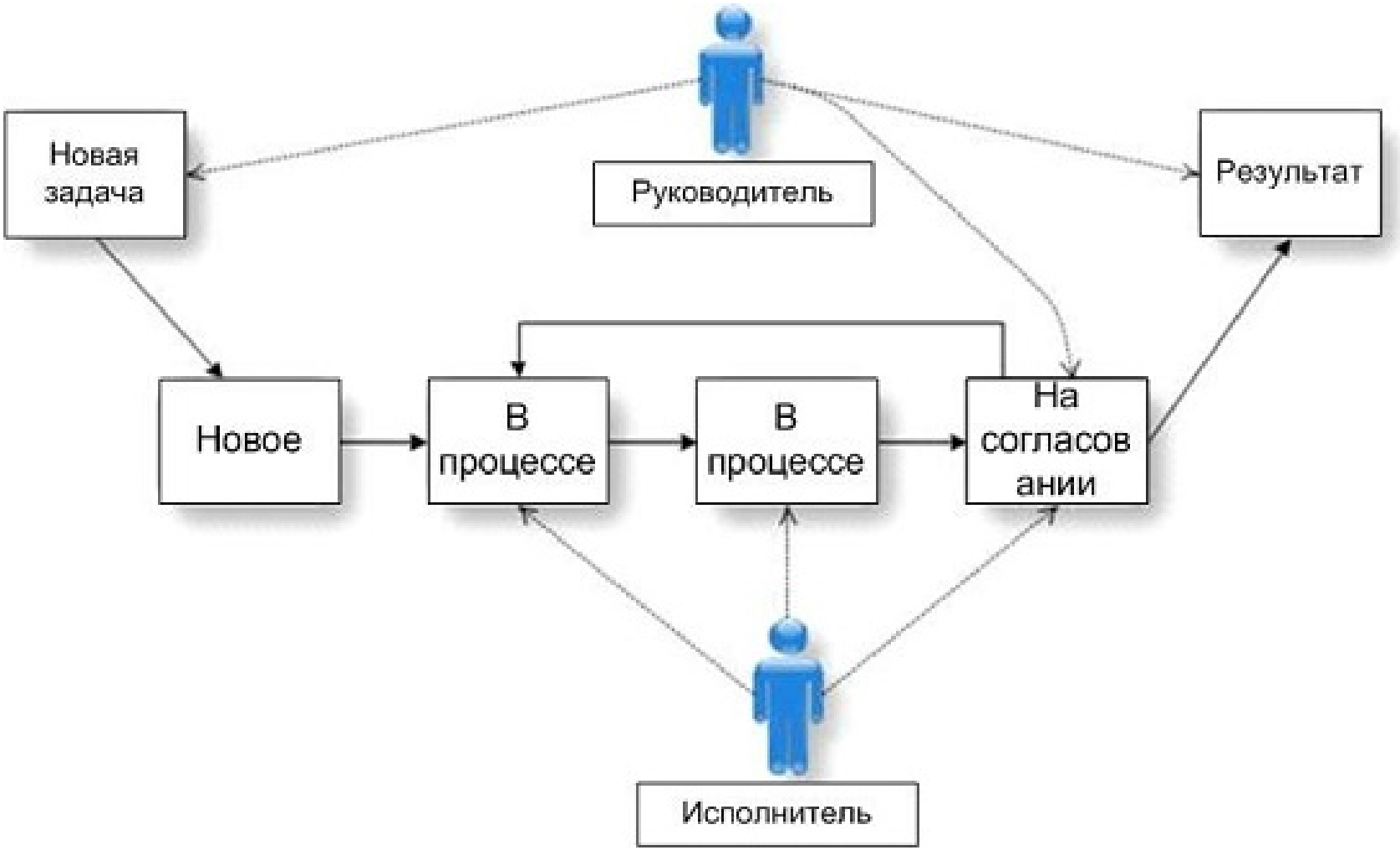
Схема добавления и разработки нового проекта

При разработке нового проекта руководитель проекта создает новую задачу в Redmine.

В ходе выполнении задачи у исполнителя могут возникнуть вопросы, которые он оформляет с помощью Redmine в виде примечаний. Если исполнитель считает, что задача завершена или требуется дополнительное обсуждение, он снова изменяет статус «Решено» или «На согласовании».

Далее, руководитель проекта принимает решение о том, удовлетворяет ли задача поставленным требованиям. Если удовлетворяет, задача остается в статусе «Решено», иначе после обсуждений, исполнитель продолжает работу над задачей, при этом исполнитель изменяет статус на «В процессе».

При достижении всех требуемых задач статуса «Решено» проект считается выполненным.



Таким образом, система управления проектами Redmine является центральным связующим звеном между руководителем проекта и командой исполнителей.

Маршрут задачи

Маршрут задачи – это последовательность назначения статусов конкретной задачи.

В логике системы, задача может иметь имеет два системных статуса: новая и закрытая. Имеется возможность называть эти статусы иначе, добавляя их количество по необходимости. Статусы назначаются в соответствии сутью конкретной задачи и служат для наглядного предоставления пользователям системы существующего положения дел по каждой из задач. В любом случае будет получен некий статус, присваиваемый вновь созданной задаче, промежуточные статусы и статус(ы), обозначающий факт завершения работы над задачей.

Трекер

Трекером управляются статусы задач по ролям. Кроме этого, трекеры позволяют указывать свой набор полей при создании задачи.

Трекер существует в системе самостоятельной сущностью, независимо от проектов и задач. Настроив трекер, достаточно указывать на его применимость в конкретном проекте. А при создании задачи внутри проекта можно выбрать конкретный трекер, привязанный к проекту. Таким образом, при создании в определенном проекте задачи, в зависимости от выбранного трекера, будет получен свой набор полей. Соответственно, и свои возможности по изменению статусов для каждой роли.

Основная логика работы в системе

Общий смысл работы в системе заключается в следующем:

1. Администратор создает задачу.
2. Путем изменения статуса с “Новая” – “Обсуждение” – “Утверждение” до статуса “Закрыта” или “Отклонена”, уведомляет всех заинтересованных лиц о происходящем с этой задачей.
3. Последние указанные два статуса закрывают задачу, то есть прекращают работу по ней.
   (Это не окончательный набор статусов и их можно настраивать как угодно для своих нужд.)
4. По мере прохождения этих статусов пользователи пишут комментарии, заполняют поля, вносят изменения, создают связанные задачи и подзадачи вплоть до завершения работы над задачей.

Такой способ взаимодействия позволяет сильно экономить время.

Основная логика работы в системе

Пример

Рассмотрим ситуацию на примере запроса закупки робы для нового сотрудника.

1. Непосредственный начальник участка создает служебную записку на имя руководителя, уведомляя о необходимости закупки робы для нового сотрудника.
   Задаче присваивается статус “Новая”.
2. На рабочем месте, войдя в систему и применив заранее приготовленный фильтр “Новые задачи”, руководитель подразделения увидит служебную записку.
   - Он меняет статус с “Новая” на “Обсуждение” и далее меняет на “Закрыта” в случае, если обсуждать нечего, тем самым информируя инициатора о том, что он ознакомился со служебной запиской и утвердил ее.

Основная логика работы в системе

Пример

2. - При наличии вопросов присваивается статус “Обсуждение” и в комментариях указываются вопросы, возникшие в процессе рассмотрения.
   - При полном несогласии задаче присваивается статус “Отклонена”.
3. Далее, руководитель подразделения создает подзадачу о необходимости закупки робы на имя начальника отдела снабжения.
   Задаче присваивается статус “Новая”.
   Задача “служебная записка” о закупке робы и задача непосредственно самой закупки обретают взаимосвязанный характер, тем самым позволяя проследить корни происхождения и пути следования, если в этом возникнет необходимость.

Основная логика работы в системе

Пример

4. Начальник снабжения, увидев новую задачу, присваивает ей статус “Обсуждение”, тем самым информируя о том, что задача принята в работу. Можно обязать начальника снабжения на этом этапе менять поле “Назначена”, определяя непосредственного исполнителя, отвечающего за закупку.
5. После того, как закупка будет осуществлена – начальник закроет эту задачу, назначив ей статус “Закрыта”.

Основная логика работы в системе

- При необходимости, можно легко проследить весь путь обсуждения запроса, начиная от служебной записки и заканчивая контролем на этапе работы службы снабжения.
- В системе контролировать можно не просто всё происходящее в виде обсуждения (комментариев), а также и скорость реагирования, сроки исполнения и так далее.
- Используя пользовательские поля можно легко проводить анализ процессов, отфильтровывая или группируя задачи по этим полям.
- В случае сбоев в производственных или управленческих процессах система также позволяет упростить анализ причин, благодаря тому, что каждый этап строго формализован. Кроме того, формализованы и функциональные возможности каждой кадровой единицы, участвующей в процессе.

Аналогия с электронной почтой

В электронной почте мы можем отслеживать общение по конкретному вопросу лишь по автору письма, теме, дате, держа в голове эти взаимосвязи.

Эта система предоставляет по сути те же самые возможности, как и электронная почта, только в немного измененном виде, не вынуждая помнить, кто и когда о чем писал, что поменялось, сроки исполнения и т.п., собирая необходимую информацию в одном месте и предоставляя в наглядном виде текущее состояние.

Как и в электронной почте, используя Redmine, мы пишем сообщения, комментарии, можем прикреплять файлы. Уведомляем иных интересантов о происходящем и тому подобное. Просто общий вид интерфейса и возможности системы позволяют упростить взаимодействие. Система позволяет как бы в одном месте и в наглядном виде показывать всё, что касается конкретного вопроса (в данном случае – задачи).

Некоторые недостатки Redmine

- Управление файлами и документами в Redmine сводится к их добавлению, удалению и редактированию. Правами доступа ни к файлам, ни к отдельным документам управлять нельзя;
- В Redmine нельзя управлять правами доступа на уровне отдельных полей задачи. Например, на данный момент от клиентов нельзя скрыть оценки времени работы над задачей. Но можно сделать дополнительные поля видимыми только пользователям с определёнными ролями;
- В Redmine в список задач не выводится общая трудоёмкость задач;

Некоторые недостатки Redmine

- Нет возможности дать пользователю роль во всей системе; например, «Руководитель проектного офиса» должен иметь доступ ко всем проектам в системе: для этого нужно добавить пользователя с этой ролью во все проекты;
- Подключить git репозиторий возможно только в случае, если и Redmine, и репозиторий находятся на одном сервере.

Преимущества системы управления проектами Redmine по сравнению с другими аналогичными программами

- Бесплатная лицензия по распространению продукта;
- Использование на любой ОС при помощи web-браузера;
- Возможность настройки большого числа параметров системы;
- Возможность интегрирования дополнительных программных специализированных модулей сторонней и своей разработки;
- Техническое сопровождение собственными силами;
- Инструмент для выполнения требований стандарта IRIS.

Спасибо за внимание.