**Описание Redmine**

Оглавление

[Общие сведения 2](#_Toc19286281)

[Основная логика работы в системе. 2](#_Toc19286282)

[Роли и права доступа 5](#_Toc19286283)

[Маршрут задачи 6](#_Toc19286284)

[Трекер 7](#_Toc19286285)

[Настраиваемые поля 9](#_Toc19286286)

[Фильтры. 11](#_Toc19286287)

[Уведомления 13](#_Toc19286288)

[Сторонние модули 13](#_Toc19286289)

[Ссылки на информацию в сети 13](#_Toc19286290)

# Общие сведения

Система управления Redmine служит для управления проектами.

Проекты представляют из себя наборы задач. Сами проекты, при этом, могут иметь иерархическую структуру по аналогии с папками в компьютере. То есть проект – подроект – подподпроект – задача.

Смысловой единицей служит непосредственно задача. Проекты являются лишь способом группировки задач. У них есть еще некоторые свойства технического характера, используемые при настройке системы, но о них в данном случае нет смысла вести речь.

Итак, мы подошли к задачам.

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

И так далее. Список не полный.

Кроме системных полей – мы имеем возможность добавлять свои поля. Это могут быть как списки значений, текстовые поля, поле для ввода дат, список пользователей системы и т.п.

В основном пользовательские поля служат для сокращения времени формализации задачи, а также для фильтрации в общем списке задач с целью удобства поиска необходимых.

Например, определив для проекта «Снабжение» пользовательское поле «Тип закупки» в виде списка, при создании задачи можно одним движением мышки определить тип закупки, без необходимости каждый раз формализовать это набором текста.

В дальнейшем мы можем фильтровать общий список задач и выбирать задачи с определенным типом закупки. Например, начальник производства может легко отобрать закупки для производственных нужд, находящиеся в статусе «Утверждено», т.е. одобренные руководством. Или, например, находящиеся в статусе «Отклонено».

# Основная логика работы в системе.

Общий смысл работы в системе заключается в следующем: мы создаем задачу и далее, путем изменения статуса с «Новая» - «Обсуждение» - «Утверждение» до статуса «Закрыта» или «Отклонена» , уведомляем всех заинтересованных лиц о происходящем с этой задачей. Последние указанные два статуса закрывают задачу, то есть прекращают работу по ней. Тут стоит заметить, что это не окончательный набор статусов и их можно настраивать как угодно для своих нужд.

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.

Рассмотрим ситуацию на примере запроса закупки робы для нового сотрудника.

Непосредственный начальник участка создает служебную записку на имя руководителя, уведомляя о необходимости закупки робы для нового сотрудника.

Задаче присваивается статус «Новая».

На рабочем месте, войдя в систему и применив заранее приготовленный фильтр «Новые задачи» - руководитель подразделения увидит служебную записку. Простым движением мышки 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. При наличии вопросов – присваивается статус «Обсуждение» и в комментариях указываются вопросы, возникшие в процессе рассмотрения. При полном несогласии – задаче присваивается статус «Отклонена».

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.

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.

После того, как закупка будет осуществлена – начальник закроет эту задачу, назначив ей статус «Закрыта».

Это показан процесс в общем виде. В описании процесса упущены такие нюансы как необходимость заполнения различных полей, права и обязательства по их заполнению на каждом этапе, переназначение исполнителей, создание иных связанных задач и так далее.

При необходимости, можно легко проследить весь путь обсуждения запроса, начиная от служебной записки и заканчивая контролем на этапе работы службы снабжения.
Благодаря системе – контролировать можно не просто всё происходящее в виде обсуждения (комментариев), а также и скорость реагирования, сроки исполнения и так далее.

Благодаря пользовательским полям 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.

Чтобы быстрее вникнуть в принципы работы системы я предлагаю рассматривать эту систему в качестве аналогии с электронной почтой. Только это более продвинутая система.

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.

Если по-началу будете ждать от системы чего-то «большого и высокого», чем просто «продвинутая почта», то у вас понимание наступит гораздо позже, нежели в случае взгляда на Redmine просто как на продвинутую электронную почту.

Судите сами, как и в электронной почте 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 нашем случае - задачи).

У нас есть задачи, в задачах есть поля с данными. Манипулируя полями и данными – каждый участник имеет возможность получать наглядное представление о текущем положении дел и реагировать на изменения. При этом эти поля сокращают время на формулирование вопроса, на предписания конкретному исполнителю, указание ответственных, назначение дат и тому подобного.

Да, тут нет связи с 1С, мы не можем осуществлять выборку оттуда или же автоматически вносить данные в 1С (технически возможность есть), мы не можем в этой системе вести учет договоров по той причине, что есть более удобные системы. Мы не можем вести конструкторские задачи или тонкие задачи по тендерным закупкам, где требуется глубокая интеграция в существующие информационные системы.

Мы просто можем при помощи этой системы формализовать определенные процессы, которые решались ранее путем электронной почты, скайпа, телефона, бумажных записок, устных распоряжений и тому подобного.

В системе мы можем разместить тексты всех приказов по предприятию, и сотрудникам можно будет быстро получать к ним доступ по мере возникновения необходимости. В ней можно хранить «Базу знаний», например с принципами проведения закупок или же о функционировании определенных внутренних систем, тем самым позволяя пользователям иметь простой способ получения необходимой информации.

Сейчас система практически полностью состоит из оригинальных (родных) модулей, обеспечивая базовый функционал. Существуют некоторые сторонние модули, упрощающие работу в системе или же расширяющие возможности (платные, $300-$1000). Но на данном этапе я не вижу смысла обращать на них пристальное внимание по причине того, что нам необходимо обеспечить базовый функционал и научиться им пользоваться.

Кроме этого, можно также привлечь специалистов с целью написания необходимого нам функционала.

В будущем, мы поймем чего нам нехватает и какой функционал необходим. И вот тогда уже мы займемся вопросом оптимизации работы в данной системе.

Эта система позволит нам без каких-либо затрат увидеть что есть информационная система для регулирования общих вопросов. Наличие этого понимания позволит нам с минимальными ошибками выбрать при необходимости иную систему или же приступить к написанию своей с более тесной интеграцией в иные информационные системы, существующие на предприятии.

Судя по опыту иных организаций, описание которого я встречал в сети, обычно внедрение информационной системы на предприятиях происходит по следующему сценарию:

1. Кто-то продвинутый на предприятии говорит: “*Нам нужна информационная система!*”.
2. Так как никто не в курсе что это такое и с чем ее едят, приглашаются сторонние специалисты, которые изучают нужды сотрудников путем опроса, составляют описание будущей системы и предоставляют счет на >500 т.р.
3. В описании они обычно ориентируются на базовый функционал подобных систем, лишь чуть-чуть обращая внимание на пожелания пользователей и специфику производственного процесса предприятия-заказчика. Но и пожеланий от опрашиваемых сотрудников обычно не так и много, так как любое нововведение заставляет пользователей менять привычные механизмы выполнения своих обязанностей, и на большинстве эшелонов структуры управления заказчика идет саботаж. Да и просто никто толком не знает, что должно получиться в результате, потому и считают, дескать, пусть об этом заботятся другие.
4. После согласования счета, который обычно исчисляется во многих сотнях тысяч рублей, начинается процесс разработки. Он длится от пары месяцев до полугода и в конце-концов система устанавливается на предприятии.
5. С этого момента начинается тяжелый процесс внедрения. Пользователи вынуждены «отвлекаться» на эту систему, наряду с выполнением обычных задач, продолжая действовать привычным способом, а пользование системой откладывается на второй план и лишь благодаря административному воздействию – с течением времени растет частота и эффективность пользования новой системой работниками.
6. Через пол года все устаканивается, и более-менее начинается полноценное использование новой информационной системы. Пользователи привыкают, запоминают куда щелкать мышкой, что вообще делать и какой в этом смысл.
7. Еще через пол года накапливается огромное количество претензий к несовершенству системы. А несовершенство происходит из того факта, что никто не сказал как оно должно быть и посторонние люди по своему усмотрению формализовали производственный процесс в виде кнопочек, полей и списков на экране монитора. Причем, обычно никто не заморачивается, если заказчик не сильно требователен, и дают типичный функционал, который адекватен имеющимся нуждам примерно как понятие «средняя температура больных по больнице»
8. Потихоньку, исправляя ошибки и меняя функционал, еще через пол года – год общие затраты на эту систему начинают приближаться к 200-300 %% от планируемых изначально (с учетом падения эффективности работы сотрудников, выраженную в человеко-часах в отношении с окладом).
9. Нехотя, руководство принимает решение отказаться от имеющегося «насоса средств» и инициирует написание своей системы. Тут дело идет более слаженно, так как пользователи нисколько не горят желанием опять жать миллион лишних кнопок и с б*о*льшим удовольствием участвуют в процессе описания необходимых взаимосвязей.
10. Вдумчиво, основываясь на опыте, происходит выбор поставщика услуг, вместе с ним формализуется довольно четкое техническое задание к системе, оплачивается еще пара миллионов рублей на написание и внедрение, плюс 10-30% на доработку в будущем и вот, через года три-четыре, с бюджетом под 3-5 миллионов (с учетом первого негативного опыта), на предприятии возникает наконец-то действующая система, упрощающая процесс управления и функционирования подразделений, вместо мучений и лишних телодвижений.

Мы не знаем что нам нужно. Мы не умеем формулировать в одном ключе на всех эшелонах структуры и в кооперации с нуждами всех подразделений. У нас нет понимания как оно может быть и как это изложить в виде конкретного техзадания. Следовательно, любые внедренцы будут внедрять по-своему, исходя из своих представлений. При этом они не заинтересованы в минимизации наших расходов, а наоборот!

Если мы внедрим свободно распространяемую систему Redmine, то через время у нас появится понимание как это должно быть, что можно сделать лучше, какой еще функционал необходим. И в будущем мы сможем сформулировать более точное техзадание ежели нам будет мало имеющегося функционала Redmine. А чем точнее техзадание – тем менее аморфны суммы затрат на разработку и меньше дополнительных вложений в будущем.

То есть, поработав в системе, даже если она нас перестанет удовлетворять, мы сможем в будущем не только сэкономить на внедрении иной системы, но еще и не тратить много времени на обучение сотрудников, так как у них уже будет понимание и опыт работы в информационной системе предприятия. Пусть даже и с отличающимся интерфейсом, но функционал и принципы будет схожи и пользователям будет гораздо легче освоиться.

Так как мы внедряем Redmine с нуля, следовательно новые задачи, решаемые при помощи этой системы будут добавляться постепенно. Этот факт упрощает время и затраты на обучение и привыкание существующих пользователей, чтобы они успешно могли работать в этой системе, повышая эффективность своего труда.

# 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.

Для начала я предлагаю использовать имеющиеся возможности следующим образом:

Определяем 4 основные роли: Участник, Исполнитель, Руководитель, Директор. Суть этих ролей лишь в наборе прав внутри системы и они не являются полной аналогией фактических должностей.

Вот, например, как можно распределить возможности внутри системы для каждой роли (упущены возможности установки прав для разделов «Новости», «Форумы», «Wiki», ввиду их малозначительности в данном контексте):

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Права доступа | Участник | Исполнитель | Руководитель | Директор |
| Создание проекта  |  |  |  |  |
| Редактирование проектов  |  |  |  |  |
| Закрывать / открывать проекты  |  |  |  |  |
| Выбор модулей проекта  |  |  |  |  |
| Управление участниками  |  |  |  |  |
| Управление версиями  |  |  |  |  |
| Создание подпроектов  |  |  |  |  |
| Управление общими запросами  |  |  |  |  |
| Сохранение запросов  |  |  |  |  |
|  Документы  | Участник  | Исполнитель  | Руководитель  | Директор  |
| Просмотр документов  |  |  |  |  |
| Добавить документы  |  |  |  |  |
| Редактировать документы  |  |  |  |  |
| Удалить документы  |  |  |  |  |
|  Файлы  | Участник  | Исполнитель  | Руководитель  | Директор  |
| Просмотр файлов  |  |  |  |  |
| Управление файлами  |  |  |  |  |
|  Задачи  | Участник  | Исполнитель  | Руководитель  | Директор  |
| Просмотр задач  |  |  |  |  |
| Добавление задач  |  |  |  |  |
| Редактирование задач  |  |  |  |  |
| Копирование задач  |  |  |  |  |
| Управление связыванием задач  |  |  |  |  |
| Управление подзадачами  |  |  |  |  |
| Установление видимости (общая/частная) для задач  |  |  |  |  |
| Установление видимости (общая/частная) для собственных задач  |  |  |  |  |
| Добавление примечаний  |  |  |  |  |
| Редактирование примечаний  |  |  |  |  |
| Редактирование собственных примечаний  |  |  |  |  |
| Просмотр приватных комментариев  |  |  |  |  |
| Размещение приватных комментариев  |  |  |  |  |
| Удаление задач  |  |  |  |  |
| Просмотр списка наблюдателей  |  |  |  |  |
| Добавление наблюдателей  |  |  |  |  |
| Удаление наблюдателей  |  |  |  |  |
| Импорт задач  |  |  |  |  |
| Управление категориями задач  |  |  |  |  |
| Просматривать чеклисты  |  |  |  |  |
| Выполнять чеклисты  |  |  |  |  |
| Редактировать чеклисты  |  |  |  |  |
|  Учёт затрат  | Участник  | Исполнитель  | Руководитель  | Директор  |
| Просмотр затрат  |  |  |  |  |
| Учёт затрат  |  |  |  |  |
| Редактирование затрат  |  |  |  |  |
| Редактирование собственных затрат  |  |  |  |  |
| Управление типами действий для проекта  |  |  |  |  |

Описанная выше система не окончательная, можно добавить иных ролей и выбрать для них новый набор прав доступа к системным функциям.

 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\задаче по электронной почте либо создателю задачи, либо исполнителю, либо наблюдателю. Таким образом, если нам необходимо чтобы кто-то просто был в курсе происходящего в определенном проекте или задаче, то включив пользователя в проект с ролью «Участник» и назначив в конкретной задаче этого пользователя как «Наблюдателя» - пользователь будет получать на почту все изменения по обозначенной задаче.

Как видно из таблицы – каждая следующая роль имеет всё более расширенные права. Это позволяет нам исключить некоторые возможные ошибки и злоупотребления, а так же упрощает интерфейс конкретного пользователя, ввиду отсутствия необходимости заполнения полей, несущественных для его выполнения его функций.

Кроме этих прав и ролей, система позволяет строить разрешения для маршрутов задачи.

# Маршрут задачи

Маршрут задачи – это последовательность назначения статусов конкретной задачи.

В логике системы, задача может иметь имеет два системных статуса: новая и закрытая. Мы имеем 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 у нас будет некий статус, присваиваемый вновь созданной задаче, промежуточные статусы и какой-то статус(ы), обозначающий факт завершения работы над задачей.

Вот наглядный пример



Как видно из рисунка, при использовании в задаче трекера «Снабжение», пользователь с ролью Руководитель может установить сразу статус задачи «Обсуждение», избегая статуса «Новая задача». По системным меркам, так сказать, статус «Обсуждение» и будет «новой задачей» (это используется в фильтрах). Назначение сразу некоторого промежуточного статуса уменьшает количество телодвижений в том случае, если создается задача, которая заранее уже утверждена. Другими словами для системы задача будет новой, хотя для пользователя будет выглядеть как уже дошедшая до статуса утвержденной.

По поводу прав назначения статусов (маршрутов): если у задачи будет статус «Обсуждение», то пользователь с ролью Руководитель сможет поменять статус на «Согласовано» или «Закрыто». На всякий случай указано, что Закрытую задачу Руководитель сможет перевести в статус «Согласовано» или «Обсуждение». Это необходимо на начальном этапе внедрения системы, так как возможны ошибки при назначении статусов, а так же я допускаю ошибки в принципах организации работы. Поэтому, данные переходы смогут позволить возобновить работу по ранее закрытой задаче. Не уверен, что подобная возможность переходов статусов будет необходима после того, как мы отладим функционал, распределим роли и разработаем оптимальные маршруты.

А вот пользователю с ролью Директор мы позволяем переводить задачу из любого статуса в любой в том же трекере «Снабжение»:



Повторю: это лишь варианты использования. Можно ввести сколько угодно статусов и позволить пользователям с соответствующими ролями менять статусы задач иными способами.

Помимо назначения статусов внутри трекера для определенной роли, система позволяет управлять также и полями, которые привязаны к трекеру.

Вот пример настройки прав роли Руководитель для трекера Снабжение:



Как можно увидеть, руководитель в случае назначения статуса Обсуждение обязан указать срок завершения и ожидаемую стоимость закупки. Кроме того, на этапе согласования он обязательно должен указать на кого назначена задача и на контроле у кого она стоит. В статусах Закрыта и Отклонена руководитель не может менять ни одно из полей.

Пришла пора пояснить что такое трекер. Надеюсь, у Вас уже образовалось общее понимание этой системы и можно попробовать пояснить суть инстанции под названием «трекер».

# Трекер

Трекер – это нечто вроде «тип задач». Как видно из рисунков выше – 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, мы просто указываем на его применимость в конкретном проекте. А при создании задачи внутри проекта – мы сможем выбрать конкретный трекер, привязанный к проекту. Таким образом, при создании в определенном проекте задачи – в зависимости от выбранного трекера мы будем иметь свой набор полей. Ну, соответственно, и свои возможности по изменению статусов для каждой роли.

Вот общий вид формы создания новой задачи для одного и того же проекта, но с разными трекерами:



Как можно увидеть, у нас различаются поля, которые необходимо заполнить пользователю при создании задачи. Прошу учитывать тот факт, что система еще не настроена более-менее оптимальным образом, поэтому существуют недоработки на предоставленных скриншотах.



# Настраиваемые поля

Система Redmine – это по сути набор принципов, позволяющих формализировать задачи, отслеживать текущее положение дел по задачам, вносить изменения и так далее. Для того, чтобы это было возможным, естественно нужны некоторые поля для заполнения. Изначально в системе этих полей имеется некий минимальный набор и этого набора может оказаться недостаточно для комфортного использования системы.

Разработчики системы ввели такой функционал, как настраиваемые поля. Если бы такой возможности небыло, то система была бы крайне ограниченна в функционале по той причине, что могло бы нехватить предопределенных полей для оптимального функционирования. А если бы разработчики сделали десятки предустановленных готовых полей, то это усложнило бы пользование системой. Поэтому, оставив возможность настройки пользовательских полей самим пользователям – разработчики сильно увеличили потенциал своей системы.

С их помощью мы можем упростить использование системы. Приведу пример процессов, происходящих на предприятии в настоящее время и как может выглядеть этот процесс в системе Redmine.

* Руководитель подразделения пишет служебную записку на определеного сотрудника отдела снабжения примерно следующего содержания*: «Прошу приобрести следующие комплектующие для обеспечения процесса производства»*. Указывается перечень приобретаемых предметов. Подпись, дата.
* Служебная записка сохраняется куда-то на диск, распечатывается, подписывается исполнителем и относится на подпись вышестоящему руководителю.
* Служебная записка утверждается, каким-то образом доставляется в отдел снабжения и начинается работа отдела снабжения по проработке этого заказа. Работник отдела где-то у себя в табличке заносит новую запись: Кто заказал, для каких нужд, дата и т.д.
* В случае необходимости уточнений – сотрудник отдела снабжения связывается с инициатором и начинает задавать вопросы. Эти вопросы он куда-то записывает отдельно. Общение идет по различным каналам связи: электронная почта, скайп, телефон, мессенджеры, личное общение. Может произойти так, что ответ невозможно получить сразу и оба сотрудника – снабженец и инициатор – должны помнить о том, что необходимо получить уточнения и обработать их и передать в соответствующие структуры.
* В случае возникновения проблем после закупки, разобраться кто кому чего писал, что спрашивал и отвечал – довольно трудоемко. А наглядно увидеть руководству как вообще происходят закупки и насколько эффективно идет работа – практически невозможно, так как трудоемкость сбора, систематизации и оформления выборки данных по закупкам отделом снабжения – очень высока, и подобные задачи крайне нерациональны. Особенно если учитывать, что резерва трудовых ресурсов у нас не наблюдается.
* Если на предприятии будет вводиться собственная служба внутренней безопасности, то подобное функционирование системы обеспечения закупок потребует координальных изменений, так как работникам СБ будет крайне сложно оперативно отслеживать происходящее в существующей системе.

В системе Redmine при создании задачи на закупку – пользователь просто заполняет необходимые поля, прикрепляет файлы и всё это сосредотачивается в едином довольно удобном интерфейсе. Любые изменения по задаче – видны моментально всем участникам. Данные вводятся единожды и используются всеми заинтересованными лицами. За счет наличия пользовательских полей существует возможность не только меньше писать текста, формализуя необходимость закупки (какой отдел, назначение, сроки и т.п.), но и позволяет в будущем легко проводить анализ ранее произведенных закупок в разрезе указанных полей.

При возникновении споров – можно легко отследить как происходила закупка и понять причины сбоев.

Все это позволит наглядно представлять происходящее в системе управления предприятием, проводить оптимизацию процессов, выявлять узкие места и в конечном счете это все приведет к уменьшению издержек и увеличению производительности труда. Соответственно, высвобожденные трудовые человеко-часы можно использовать для решения иных задач.

Пользовательские поля привязываются к трекерам. И в итоге получается так, что создавая новую задачу в каком-то проекте, через трекер эти поля появляются в формах создания новой задачи. Напомню, что трекеры привязываются к проектам.

Создание нового настраиваемого поля происходит в несколько этапов.

Первый этап: определение применимости поля.

Второй этап: определение типа поля

Третий этап: указание параметров и назначение прав и применимости данного поля.

|  |  |  |
| --- | --- | --- |
| **1** | **2** | **3** |
|  |  |  |

Последний этап для пользовательского поля для задач в целом выглядит следующим образом:



На первый взгляд всё довольно сложно. Но на самом деле эта глубокая интеграция позволяет настраивать функционал системы очень гибко с учетом многих специфических нужд конкретного типа задач.

Все эти подробности я привожу для того, чтобы объяснить, что я имел ввиду при указании на тот факт, что по умолчанию в системе мало готовых полей. И чтобы показать, насколько гибко может быть настроена система благодаря наличию пользовательских полей.

# Фильтры.

До этого момента повествование шло о внутренностях системы и о функционале ее в целом. Конечному пользователю было уделено мало внимания. Разве что в разделе о пользовательских полях мы коснулись функционала для конкретного пользователя. Пришло время устранить этот недостаток.

Итак, фильтры.

Фильтры являются простой и в то же время крайне важной функцией системы, предназначенной, прежде всего для обеспечения удобства работы пользователя. Причем, чем дольше будет функционировать и развиваться система, обеспечивая рабочий процесс предприятия, тем более важную роль будут обретать фильтры в сознании пользователей.

Как понятно из названия – фильтры служат для выборки информации определенного характера. В системе Redmine они играют крайне важную роль в обеспечении удобства пользования системой.

Если представить, что в системе отсутствует функционал фильтрации, то для отслеживания новых задач нам необходимо будет зайти в конкретный подпроект и в нем искать интересующую нас задачу. И уже в самой задаче смотреть, что же там поменялось и на каком этапе она находится.

Фильтры позволяют быстро получать необходимую информацию путем единоразового нажатия кнопки мыши.

Фильтры могут быть сохранены на уровне системы для всех пользователей и для пользователей с определенной ролью. Также пользователи имеют возможность применять и сохранять свои фильтры для личного пользования, которые недоступны иным участникам.

В фильтрах мы указываем значения полей, которые нас интересуют и после применения фильтра – система отберет из всех задач только соответствующие заданным условиям.

На скриншоте ниже указан функционал системы фильтров. Как видно, данный фильтр покажет нам задачи со статусом Открыто и относящиеся к трекеру Служебная записка.

Правее указан неполный раскрытый список иных способов фильтрации, которые можно комбинировать во всех возможных вариациях.

Кроме того, мы можем указать какие столбцы отображать в общей таблице результатов фильтрации.

По мере наполнения системы задачами мы будем обрастать и количеством фильтров. Как было упомянуто выше – часть из них мы будем использовать на уровне всей системы, а часть – пользователи будут делать сами для своего удобства.

Возможность создания фильтров на уровне системы позволяет избавиться от необходимости каждому пользователю создавать себе фильтры. Он просто будет пользоваться доступными.



# Уведомления

Система Redmine умеет оповещать о событиях внутри себя путем отправки уведомлений на email. Это избавляет пользователей от необходимости постоянно просматривать списки задач относящихся к ним.

Электронная почта используется для решения и иных функциональных обязанностей, поэтому возможность отправки на почту повышает удобство пользования системой, ведь электронной почтой сотрудники пользуются гораздо чаще и новые уведомления имеют шанс быть замеченными раньше, чем просто в случае ожидания, когда же пользователь соизволит зайти в интерфейс и посмотреть не обновилось ли чего-то.

Настройки данной функции следующие:



В принципе, этого функционала достаточно для типичных задач и он позволяет каждому пользователю иметь удовлетворяющий его уровень количества сообщений от системы.

На мой взгляд будет хорошей практикой обязать сотрудников начинать рабочий день с изучения положения текущих дел в системе Redmine. Постепенно они привыкнут периодически заходить в систему в течение рабочего дня в зависимости от наличия и объема текущих дел, даже не принимая во внимание наличие функции уведомлений на email.

# Сторонние модули

На сегодняшний день существует большое количество дополнительных модулей для Redmine. Бесплатные варианты чаще всего обеспечивают лишь небольшие улучшения частного характера. По мере развития интеграции системы в рабочий процесс на предприятии, мы будем принимать решения о необходимости улучшения функционала при помощи сторонних модулей.

Так же существуют и платные модули. Цена на них довольно высока, однако и функционал гораздо выше, чем в бесплатных аналогах.

На данный момент я могу обозначить три лидера по платным модулям:

<https://www.redmineup.com/pages/ru>

<http://rmplus.pro/ru/redmine/plugins/all>

<https://www.easyredmine.com/ru/>

Они весьма широко и качественно расширяют функционал системы, но на сегодня я не вижу острой необходимости в их приобретении. Полагаю, что нужно сначала завести систему на основе имеющихся функций и способов взаимодействия. И если система покажет себя как реально функциональная и у нас получится оптимизировать рабочий процесс с помощью этой системы, то тогда уже можно будет рассматривать необходимость использования дополнительных платных модулей.

# Ссылки на информацию в сети

[Преимущества Redmine](http://suos-ing.ru/stati/preimushhestva-redmine/)

[Оперативное планирование в Redmine](https://habr.com/ru/post/245065/)

[REDMINE - СИСТЕМА УПРАВЛЕНИЯ ПРОЕКТАМИ](https://www.dvbi.ru/articles/reading/Redmine-tasks-management-system)

[Redmine для управления ИТ: практический опыт обширного внедрения opensource-системы](https://infostart.ru/public/815261/)

[Redmine, который вы захотите попробовать](https://sohabr.net/habr/post/337884/) (допиленный плагинами)