**Описание функциональных характеристик, информация для установки и эксплуатации программного обеспечения**

**«Модуль IT-Finance «Документооборот»**

Оглавление

[**Функциональные характеристики программного обеспечения** 1](#_Toc129790330)

[**Установка программного обеспечения** 2](#_Toc129790331)

[**Требования к рабочему окружению** 2](#_Toc129790332)

[**Требуемые программные продукты в составе клиентской части** 2](#_Toc129790333)

[**Эксплуатация программного обеспечения** 2](#_Toc129790334)

[Руководство пользователей 2](#_Toc129790335)

[1. Создание пакета документов 2](#_Toc129790336)

[1.1 Редактирование пакета документов 3](#_Toc129790337)

[2. Сценарии, алгоритмы 4](#_Toc129790338)

[2.1. Алгоритм автоматического назначения ответственного сотрудника в заявку (change\_assign\_user) 4](#_Toc129790339)

[2.2. Алгоритм сравнения сумм для правил (get\_lookup\_use\_summ\_rule) 8](#_Toc129790340)

[2.3. Алгоритм отвязки ответственного сотрудника 9](#_Toc129790341)

[2.4. Алгоритм привязки ответственного сотрудника (fasten\_request) 9](#_Toc129790342)

[2.5. Алгоритм закрепления сотрудника за компанией по ИНН (fasten\_bidder) 9](#_Toc129790343)

[3. Сценарии, алгоритмы 10](#_Toc129790344)

[3.1. Алгоритм генерации превью документов (preview-generator) 10](#_Toc129790345)

# **Функциональные характеристики программного обеспечения**

**Программное обеспечение Модуль IT-Finance «Документооборот» (далее по тексту также программное обеспечение/ ПО/ Cистема)**

Программа обеспечивает автоматизацию управления документами и бизнес-процессами.

Область применения: для организаций, использующих программу IT-Finance (свидетельство № 2018615047 от 23.04.2018).

Функциональные возможности:

Автоматическое заполнения справочных данных из различных сервисов;

Обработка и сохранение электронных копий документов клиента, прикрепление к кредитной заявке; Поддержка версионности и сроков хранения документов; Маршрутизация и согласование заявки; Поддержка юридически значимого документооборота; Создание шаблонов документов; Настройка маршрутов согласования.

.

# **Установка программного обеспечения**

ПО распространяется в виде интернет-сервиса, специальные действия по установке ПО на стороне пользователя не требуются

# **Требования к рабочему окружению**

# **Требуемые программные продукты в составе клиентской части**

|  |  |  |
| --- | --- | --- |
| **Наименование** | **Версия** | **Официальный сайт продукта** |
| КриптоПро CSP | Действующие сертифицированные | <http://www.cryptopro.ru/> |
| КриптоПро ЭЦП Browser plug-in | <http://www.cryptopro.ru/> |
| **Браузеры (любой из):** |  |  |
| Firefox | Три последние официальные стабильные версии (вышедшие не позднее года на момент проверки версионности) | <https://www.mozilla.org/> |
| Google Chrome | <https://www.google.ru/chrome/> |

# **Эксплуатация программного обеспечения**

# Руководство пользователей

### Создание пакета документов

Чтобы создать новый пакет документов, нажмите кнопку *"Создать набор документов"*в окне со списком пакетов (см. рис. 13.1.2). Откроется оно для создания нового пакета документов (см. рис. 13.2.1).



Рисунок 13.2.1 - Окно создания пакета документов

Для создания пакета необходимо:

* Заполнить поле *"Название пакета" (1)*
* Выбрать список ограничений (в качестве вариантов выбора используется содержимое справочника *"Ограничения"*), с которыми может быть применен данный пакет (2)
* Добавить хотя бы один документ в пакет. Добавление осуществляется путем выбора из списка доступных документов в поле *"Типы документов" (3).*а
* Чтобы требовать обязательного прикрепления документа при использовании пакета - установите флажок "*Обязательный" (4)* рядом с соответствующим документом
* Чтобы расширить количество документов в пакете нажмите кнопку *"Добавить документ" (5)*
* Для использования пакета документов в составе заявок необходимо активировать флажок *"Пакет активен" (6)*
* Чтобы удалить лишний документ нажмите кнопку *"Удаление" |*
* Для сохранения пакета нажмите кнопку *"Сохранить" (7)*

### 1.1 Редактирование пакета документов

Для редактирования существующего пакета документов нажмите кнопку "*Редактирование*"рядом с соответствующим пакетом документов в окне со списком доступных пакетов. Окно редактирование аналогично окну создания пакета (см. рис. 2.3.1). Для редактирования доступны все поля. Чтобы сохранить изменения, нажмите кнопку *"Сохранить".*



Рисунок 2.3.1 - Окно редактирования пакета документов

## 2. Сценарии, алгоритмы

### 2.1. Алгоритм автоматического назначения ответственного сотрудника в заявку (change\_assign\_user)

#### 2.1.1. Основной сценарий

1) Сначала смотрим, кого должны назначить для повторной заявки:

По истории назначения ОС по заявкам (ebb\_historyrequestsassigneduser) ищем самую свежую запись (по assigned\_date), удовлетворяющую одновременно следующим условиям, как в текущей заявке:

* + Если указан тендер (сопоставляем по request\_id.tender\_id и req.tender\_id):
		- Номером тендера (number)
		- Номером лота (numberLot)
		- ИНН заказчика (customerInn)
	+ ИНН соискателя (request\_id → inn=req.inn)
	+ Статусом назначения (assigned\_status\_id IN req.status\_id)

И в которой указан пользователь (user\_id IS NOT NULL), для которого существует правило назначения (ebb\_assigneduserrules.assigned\_user\_id=user\_id), удовлетворяющее следующим условиям:

* + Активное (active=True)
	+ Стоит признак 'Назначать ответственным в повторной заявке' (assign\_when\_repeat=True)
	+ Без указания типов банковских продуктов (product\_types\_id IS NULL) ИЛИ с указанием банковского продукта текущей заявки (product\_types\_id IN req.product\_id)
	+ Без признака 'Проверять по сумме БГ для повторной заявки' (assign\_when\_repeat\_check\_sum\_bg=False) ИЛИ с признаком (assign\_when\_repeat\_check\_sum\_bg=True) И (подходящей суммой (см. ниже [Алгоритм сравнения сумм для правил (get\_lookup\_use\_summ\_rule)](#scroll-bookmark-33)) ИЛИ с нулевыми суммами (min\_sum\_bg=0 AND max\_sum\_bg=0)).

Ну и берем самое первое активное правило для этого пользователя с признаком 'Назначать ответственным в повторной заявке' (rule = ebb\_assigneduserrules(assigned\_user\_id=user\_id, assign\_when\_repeat=True, active=True).

2) Если мы не нашли такое правило

2.1) Смотрим, может не стоит менять ответственного сотрудника и подходит уже назначенный:

Смотрим, существует ли в правилах назначения правило, одновременно удовлетворяющее следующим условиям:

* + - Активное (active=True)
		- Принадлежит уже назначенному ОС (assigned\_user\_id=req.bidders\_requestsadditionaldata.assigned\_user\_id)
		- Сам сотрудник не удален (ebb\_extendedusers(id = assigned\_user\_id).wasRemoved=False)
		- Заявка может назначаться на ОС в этом статусе (statuses\_sum\_bg IN req.status\_id)
		- Без указания типов банковских продуктов (product\_types\_id IS NULL) ИЛИ с указанием банковского продукта текущей заявки (product\_types\_id IN req.product\_id)

Если существует, значит менять ничего не надо и выходим из алгоритма.

2.2) Иначе, пытаемся найти правила для назначения, одновременно удовлетворяющие следующим условиям:

* + - Активное (active=True)
		- Сам сотрудник не удален (ebb\_extendedusers(id = assigned\_user\_id).wasRemoved=False)
		- Заявка может назначаться на ОС в этом статусе (statuses\_sum\_bg IN req.status\_id)
		- Без указания типов банковских продуктов (product\_types\_id IS NULL) ИЛИ с указанием банковского продукта текущей заявки (product\_types\_id IN req.product\_id)
		- Подходящее по сумме (см. ниже [Алгоритм сравнения сумм для правил (get\_lookup\_use\_summ\_rule)](#scroll-bookmark-33) ИЛИ с нулевыми суммами (min\_sum\_bg=0 AND max\_sum\_bg=0))
		- Если выполняем данную проверку первый раз (в рамках текущего алгоритма возможен переход к этому пункту из п.3, т.е. если мы выполняемся при переходе из него, то этот фильтр не нужен):
		Подходящее по скорингу заявки (scorings\_id IN req.bidders\_requestsadditionaldata.used\_scoring\_id)

Для каждого такого правила вычисляем:

* + - Приоритет роли из настроек (priority\_max) = MAX(priority из ebb\_assignedusersettings где role\_id есть в списке ролей пользователя правила (ebb\_userroles(user\_id=assigned\_user\_id)))
		- Приоритет суммы или привязки сотрудника к компании по ИНН из настроек распределения для ролей (filter\_by\_summ) = MAX(use\_sum\_bg\_or\_assigned\_bidder\_priority из ebb\_assignedusersettings где role\_id есть в списке ролей пользователя правила (ebb\_userroles(user\_id=assigned\_user\_id)))
		- Собственно, определяем признак - есть привязка компании к сотруднику по ИНН или нет (filter\_by\_bidder) = 1, если есть привязка (EXISTS ebb\_extendedusersassignedinnkpp(user\_id=assigned\_user\_id, inn = req.inn)), иначе 0

и дополнительно смотрим, если в заявке указана годовая выручка (берем самый свежий годовой показатель 2110 из ФП → Форма 2 (в сис.переменных это было бы 2110^y), то берем его, умножаем на 1000 и проверяем, что правило соответствует условию

COALSECE('Годовая выручка от (руб.)' (min\_fp\_2110\_year), -INF) <= 2110^y \* 1000 <= COALESCE('Годовая выручка до (руб.)' (max\_fp\_2110\_year), INF)

Далее, для более равномерного распределения по сотрудникам нам необходимо посчитать, кому сколько заявок уже назначено по настройкам правил.

Поэтому для каждого правила считаем кол-во заявок, назначенных сотруднику (count\_req), при этом вообще исключаем правила, если они относяться к уже назначенному сотрудинку в заявке (assigned\_user\_id=req.bidders\_requestsadditionaldata.assigned\_user\_id), т.к. проверка по нему уже была в предыдущем пункте и значит тут нам он уже не подходит:

count\_req = Выбираем кол-во заявок, исключая текущую, где назначен сотрудник из правила (bidders\_requests ← bidders\_requestsadditionaldata.assigned\_user\_id=assigned\_user\_id) и одновременно выполняются следующие условия

* + - * Заявка в статусе 'Статусы для определения количества назначенных заявок' (bidders\_requests.status\_id IN max\_current\_assigned\_requests\_statuses\_id)
			* Продукт заявки в списке 'Типы банковских продуктов для определения количества назначенных заявок' (bidders\_requests.product\_id IN current\_assigned\_requests\_product\_types\_id), если список указан, иначе все подходят

И оставляем только правила, подходящие под условие

* + - Стоит признак 'Учитывать количество назначенных заявок' (use\_max\_current\_assigned\_requests\_count) И 'Максимальное количество назначенных заявок' (max\_current\_assigned\_requests\_count) > count\_req

ИЛИ

* + - Не стоит признак 'Учитывать количество назначенных заявок'

ИЛИ

* + - 'Максимальное количество назначенных заявок'=0

Далее берем первое подходящее правило с учетом сортировки (если какое-то из полей сортировки NULL, то считаем, что оно = 0)

* + - priority\_max DESC
		- filter\_by\_summ DESC
		- filter\_by\_bidder DESC
		- priority DESC
		- count\_req ASC
		- На всякий случай, добавим элемент рандомности ) RANDOM ASC

3) Если на предыдущих этапах мы так и не нашли нужное правило, то всё равно упорно стараемся найти ОС, поэтому ищем правило, у которого стоит признак 'Назначать, если не найдено других сотрудников', т.е. выбираем правила, которые соответствуют следующим условиям одновременно

* + Активное (active=True)
	+ Сам сотрудник не удален (ebb\_extendedusers(id = assigned\_user\_id).wasRemoved=False)
	+ Заявка может назначаться на ОС в этом статусе (statuses\_sum\_bg IN req.status\_id)
	+ Стоит признак 'Назначать, если не найдено других сотрудников' (assign\_request\_if\_no\_any\_user)
	+ Без указания типов банковских продуктов (product\_types\_id IS NULL) ИЛИ с указанием банковского продукта текущей заявки (product\_types\_id IN req.product\_id)
	+ Если выполняем данную проверку первый раз (в рамках текущего алгоритма возможен ещё раз переход к данному пункту, т.е. если мы выполняемся повторно, то этот фильтр не нужен):
	Подходящее по скорингу заявки (scorings\_id IN req.bidders\_requestsadditionaldata.used\_scoring\_id)

Далее, определяем для каждого правила приоритет по роли

priority\_max = MAX(priority из ebb\_assignedusersettings где role\_id есть в списке ролей пользователя правила (ebb\_userroles(user\_id=assigned\_user\_id)))

И берем первое правило с учетом сортировки

* + priority\_max DESC
	+ priority DESC

Если правило найдено - УРА!!! идем дальше, иначе пытаемся ещё раз найти сотрудника, но уже без учета привязки к скорингу, т.е. переходим к п. 2.2), а если уже пытались, то УВЫ, бесполезно искать и выходим.

4) Если в результате всех мытарств на предыдущих шагах правило найдено, то начинаем следующие манипуляции:

* + Отвязываем ОС (см. ниже [Алгоритм отвязки ответственного сотрудника (unfasten\_request)](#scroll-bookmark-34))
	+ Записываем в историю по заявке событие 'Открепление ответственного сотрудника' (action = ebb\_lovs(lovName\_id=41, pid=31)
	+ Привязываем нового ОС (см. ниже [Алгоритм привязки ответственного сотрудника (fasten\_request)](#scroll-bookmark-35))
	+ Записываем в историю по заявке событие 'Назначение ответственного сотрудника' (action = ebb\_lovs(lovName\_id=41, pid=31)

Ну и, если у соискателя по ИНН из заявки нет закрепленного сотрудника (NOT EXISTS ebb\_extendedusersassignedinnkpp(inn=req.inn)) И в правиле стоит признак 'Привязывать к Принципалу без ответственного' (can\_assign\_for\_free\_bidder), то привязываем к компании сотрудника (см. ниже [Алгоритм закрепления сотрудника за компанией по ИНН (fasten\_bidder)](#scroll-bookmark-36)).

### 2.2. Алгоритм сравнения сумм для правил (get\_lookup\_use\_summ\_rule)

#### 2.2.1. Предисловие

Для продуктов (req.product\_id → pid)

* 8, Расчетно-кассовое обслуживание
* 11, РКО (открыть по видеосвязи с документами)

возвращаем NULL.

#### 2.2.2. Основной сценарий

В зависимости от правила 'Правило выбора суммы' (use\_summ\_rule\_id → pid) выбираем сумму по текущей заявке (**TO\_DO - правила выбора сумм используются во многих местах, но пока не описаны**)

* 1: product\_liabiliti\_summ
* 2: get\_product\_summ\_bg
* 3: summ\_bg\_from\_judgment\_or\_request
* 4: summBg

и возвращаем результат сравнения

'Сумма БГ от (руб.)' (min\_sum\_bg) <= полученная сумма <= 'Сумма БГ до (руб.)' (max\_sum\_bg)

### 2.3. Алгоритм отвязки ответственного сотрудника

#### 2.3.1. Основной сценарий

1) Запоминаем текущего ОС из заявки (old\_assigned\_user\_id = req.bidders\_requestsadditionaldata.assigned\_user\_id)

2) Убираем привязку к ОС в заявке (req.bidders\_requestsadditionaldata.assigned\_user\_id = NULL)

3) Берём последнюю запись по id из истории привязок бывшего ОС к заявке (ebb\_historyrequestsassigneduser(user\_id=old\_assigned\_user\_id , request\_id = req.id))

4) Обновляем эту запись

* 'Дата снятия' (removal\_date) = CURRENT\_TIMESTAMP
* 'Статус снятия' (removal\_status\_id) = текущий статус заявки

### 2.4. Алгоритм привязки ответственного сотрудника (fasten\_request)

#### 2.4.1. Основной сценарий

1) Запоминаем текущего ОС из заявки (old\_assigned\_user\_id = req.bidders\_requestsadditionaldata.assigned\_user\_id)

2) Заменяем привязку к ОС в заявке (req.bidders\_requestsadditionaldata.assigned\_user\_id = id нового сотрудника)

3) Берём последнюю запись по id из истории привязок бывшего ОС к заявке (ebb\_historyrequestsassigneduser(user\_id=old\_assigned\_user\_id , request\_id = req.id))

4) Обновляем эту запись

* 'Дата снятия' (removal\_date) = CURRENT\_TIMESTAMP
* 'Статус снятия' (removal\_status\_id) = текущий статус заявки

5) Добавляем в историю привязок ОС по заявке  (ebb\_historyrequestsassigneduser) запись

* 'Ответственный сотрудник' (user\_id) = id нового сотрудника
* 'Дата назначения' (assigned\_date) = CURRENT\_TIMESTAMP
* 'Статус назначения' (assigned\_status) = текущий статус заявки

### 2.5. Алгоритм закрепления сотрудника за компанией по ИНН (fasten\_bidder)

#### 2.5.1. Основной сценарий

1) Ищем старую привязку к сотруднику (ebb\_extendedusersassignedinnkpp(inn=req.inn))

2) Запоминаем старого сотрудника (old\_assigned\_user\_id)

3) Если привязка есть, то:

3.1) Обновляем в ней

* + 'Ответственный сотрудник' (user\_id) = id нового сотрудника

3.2) Берём последнюю запись по id из истории закрепленных сотрудников по ИНН (ebb\_historyextendedusersassignedinnkpp(inn=inn, user\_id=old\_assigned\_user\_id)) и обновляем в ней

* + 'Дата снятия' (removal\_date) = CURRENT\_TIMESTAMP
	+ 'Статус снятия' (removal\_status\_id) = текущий статус заявки
	+ 'Открепляющий сотрудник' (unfasten\_user\_id) = id текущего пользователя

4) Если привязки нет, то создаём её (ebb\_extendedusersassignedinnkpp)

* + 'ИНН' (inn) = ИНН
	+ 'Ответственный сотрудник' (user\_id) = id нового сотрудника

5) Создаём в истории закрепленных сотрудников по ИНН (ebb\_historyextendedusersassignedinnkpp) запись

* 'ИНН' (inn) = ИНН
* 'Ответственный сотрудник' (user\_id) =  id нового сотрудника
* 'Дата назначения' (assigned\_date) = CURRENT\_TIMESTAMP
* 'Статус назначения' (assigned\_status) = текущий статус заявки

## 3. Сценарии, алгоритмы

### 3.1. Алгоритм генерации превью документов (preview-generator)

#### 3.1.1. Основной сценарий

Принимаем какой-либо из следующих параметров:

1. Только высоту
2. Только ширину
3. Высоту и ширину одновременно

Тело исходника получаем в body  (обязательное).

1. Если получена только высота или ширина вычисляем второй размер.
2. Если переданы оба параметра, изображение должно быть помещено по центру полотна этого размера, и выполнена подрезка сторон без искажений.

На выходе получаем файл с заданным расширением и код 200.
Или устанавливаем код ошибки 400, 500 и в detail пишем исключение.