Управление вложениями в CM6-Base

# Термины

1. Документ - любой прикладной объект (ДО), для которого требуются вложения;
2. Версия документа - прикладной объект (ДО), сохраняющий (фиксирующий) прикладные данные ("реквизиты") документа по инициативе пользователя или СЭД (в соответствии с прикладной логикой);
3. Файл (файл вложения) - файл, в котором хранится содержание вложения (обычно в файловой системе сервера);
4. Вложение ("attachment") - ДО, необходимый для привязки файла к документу; типы ДО вложений могут расширяться, но все делаются по шаблонам на основе типа "Attachment" или его наследников.
5. Карточка вложения (КВ) - прикладная сущность, представляющая атрибуты вложения в документе.

# Основные требования

1. Вложения (присоединенные файлы) не имеют самостоятельного применения в СЭД, а только как часть "контента" определенного ДО, например, документа (далее под "документом" подразумевается любой прикладной объект, для которого требуются вложения).
2. У документа в СЭД может быть несколько "реквизитов", содержащих вложения, имеющих разное назначение. Например: "Содержание", "Приложения", "Образ после подписания", ...
3. Вложения некоторых других типов "относятся" не столько к реквизитам документа, сколько к другим вложениям (обычно m:1, т.е. несколько вложений такого типа к одному "обычному" вложению). Например:  
   - электронные подписи (ЭП);  
   - представления в других форматах ("rendition").
4. РичТекст (форматированный текст с таблицами, изображениями,...) скорее всего тоже будет сохраняться как текстовое вложение в формате MIME (с одной частью типа text/html или несколькими связанными multipart/related).
5. Возможность модификации файла вложения (например, в результате его редактирования) предусмотрена платформой, но не GUI.   
   Денис: "Что касается подмены ссылок, то этот функционал и сейчас есть в платформе - можно обновить содержимое файла, при этом обновив сам ДО вложения (но платформенный виджет так не делает, потому что в нём физически отсутствует функционал "замены" - есть только удаление и добавление нового вложения)"   
   1) надо будет подключить плагин для работы с файлами вложений от СМJ-Web 4.х (Uploader);  
   2) даже без плагина можно доработать виджет так, чтобы файл можно было загрузить в систему именно как модификацию уже имеющегося вложения.
6. Версии: при версионировании документа обычно версионируются и вложения. Т.е. у каждой версии документа в общем случае свой состав и содержание вложений. Нужен учет, что в какой версии добавлено, удалено, изменено. Очень желательно не хранить копии файлов, не модифицированных в очередной версии.
7. Включение одного и того же вложения в разные документы не требуется. Для СЭД таких требований нет, т.к. единицей документооборота является документ, а не файл.
8. Самостоятельное версионирование вложений (без создания версий документа, например, при каждом сохранении файла) пока не требуется.

# Решение

1. Прикладная сущность КВ реализуется как ДО Вложение одного из типов, сконфигурированных по правилам платформы специально для управления присоединенными файлами документов.
2. Если для конкретного типа КВ требуются дополнительные атрибуты, они добавляются с помощью шаблонного типа ДО, расширяющего системный тип "Attachment".
3. Возможность не копировать файл в версиях документа, если он не менялся от версии к версии, реализуется множественностью ДО вложений, указывающих на один и тот же файл. То есть, ДО Вложение ставим в отношение Мн:1 к файлу, и именно это обеспечивает Мн:Мн между версиями документов и файлами.
4. Права доступа к вложениям в большинстве случаев должны полностью наследоваться от родительского объекта - документа или версии. Получается, что один и тот же файл, полученный через вложение, прицепленное к документу, может редактироваться вместе с документом, а через вложение, привязанное к версии документа - нет. Это нормально, т.к. если файл из документа будет модифицирован, путь на него в соответствующем вложении будет изменен. То есть, в документе, можно сказать, будет уже другая "версия" вложения.
5. Отнесение определенных вложений документа к определенному его реквизиту можно обеспечить за счет разных типов ДО вложений, как это предусмотрено платформой, но возможно использование и дополнительных прикладных атрибутов вложений (подробнее см. ниже).
6. Вложение принадлежит (вложено в) одному родительскому объекту. Ссылка на "родителя" устанавливается только при создании вложения и более не меняется.
7. Как на ДО вообще, так и на вложения можно устанавливать ссылки. Возможна ссылка из любого ДО даже на "чужие" вложения, т.е. вложенные в другой объект.
8. Вложения некоторых типов не только "вложены" в какой-то прикладной ДО (документ или его версию), но еще и связываются с определенным другим вложением обычно того же родительского ДО, но это не обязательно. Для этого в конфигурацию типа вложения добавляется поле-ссылка. Её можно делать с произвольным типом (\*), чтобы не плодить дополнительных шаблонов для типов вложений с каждым конкретным типом такой ссылки.  
   Например, в структуре шаблона вложений для ЭП ссылка на источник - в поле Source:  
   <domain-object-type name=*"FileSignature"* template=*"true"* extends=*"Attachment"*>   
       <fields>   
           **<reference name=*"Source"* type=*"\*"* not-null=*"true"*/>**   
   Вместо таких ссылок можно было бы "вкладывать" одно вложение в другое, но конфигурация платформы не поддерживает вложенность шаблонов.
9. Расширение конфигурации по поводу вложений не требуется (всё задается существующими средствами). Пример конфига по части вложений представлен ниже.
10. Доработка платформы для поддержки нескольких записей "вложение" одного типа, ссылающихся на один и тот же файл в хранилище файлов. Подробнее см. [CMFIVE-4291](https://jira.inttrust.ru:8443/browse/CMFIVE-4291) Поддержка сервисом вложений размножения ДО Вложение без копирования файла (файл обрабатывается в режиме copy-on-write).   
    В качестве временного но быстрого решения (до доработки платформы) можно оставить копирование файлов, но следует сделать это так, чтобы потом было легко обновить. Например, в прикладном (уровня СМ6) сервисе вложений (см.ниже) можно реализовать особый метод копирования вложений - специально для копирования между документом и его версиями.

## Примерный сценарий обработки вложений при создании версии документа

Чтобы к версии документа приделать то же вложение, что уже есть в документе, надо:

1. Получить исходный ДО Вложение, привязанный к документу.
2. Создать новый ДО Вложение того же типа, но к версии -   
   методом createAttachmentDomainObjectFor(Id objectId, String attachmentType)
3. Заполнить его прикладные поля (если есть), копируя прикладные данные из исходного.
4. Если есть, обновить ссылку на другое вложение, к которому относится данное вложение (отсюда вытекает ограничение на порядок переноса вложений документа: сначала все без ссылок, потом со ссылками).
5. Вызвать новый метод (reuseAttachmentFrom - см. [CMFIVE-4291](https://jira.inttrust.ru:8443/browse/CMFIVE-4291) или его временную замену), передав ему идентификатор исходного ДО вложения и еще не сохраненный новый ДО вложение. Метод сохранит новое вложение и вернет его объект, после чего по его идентификатору на него тоже можно будет устанавливать ссылки.

## Учет разных типов вложений

Есть три способа "селекции" вложений (можно сказать три измерения), которые можно применять в приложениях:

1. Шаблон типа вложения: при добавлении типа вложения в конфигурацию типа родительского ДО указывается шаблон типа вложения. Например:   
   <attachment-types>   
       <attachment-type name=*"Doc\_Content"* template=*"Content"*/>   
       <attachment-type name=*"Doc\_FileSignature"* template=*"FileSignature"*/>   
   </attachment-types>   
   Более того, таких шаблонов может быть построена целая иерархия с расширением состава полей в производных шаблонах.   
   В прикладном коде для конкретного вложения легко определить и его тип, и шаблон типа, и всю его иерархию.   
   Именно этот уровень следует использовать для обобщения прикладного кода, обрабатывающего вложения одного назначения и одной структуры, но к разным типам родительских объектов, т.е. с разными типами вложений, но сделанными по общему шаблону.   
   Например, вложение Содержание в другом типе документа может быть сконфигурировано так:   
   <attachment-type name=*"****DocInput\_Content****"* **template=*"Content"***/>   
   Таким образом, если "бизнес-логика" работы с такими вложениями у этих типов документов общая, нужно использовать для них общий шаблон вложений, и все процедуры "затачивать" именно под него.
2. Тип вложения: тип ДО вложения, объявляемый элементом <attachment-type name=*"..."*/> внутри типа ДО-родителя.   
   По задумке платформы именно в соответствии с типом вложение относится к определенному реквизиту документа (и виджету на форме в GUI).
3. Прикладные атрибуты вложения позволяют различать уже конкретные вложения одного типа/шаблона. Например, это пригодится, если среди нескольких вложений в Содержании документа ( name=*"Doc\_Content"* **template=*"Content"***) надо будет выделить одно "основное".

## Вложения для разных реквизитов документа

Бывает, что в документе кроме, например, Содержания есть другие реквизиты, наполняемые с помощью вложений. Платформа по умолчанию обеспечивает это с помощью разных типов вложений в одном типе документа. Они могут иметь общий шаблон, если суть их различия - только в том, что они входят в разные реквизиты/виджеты. Например, в одном типе документа два реквизита Содержание и Приложения могли бы задаваться так:   
<attachment-type name=***"Doc\_Content"*** **template=*"Content"***/>   
<attachment-type name=*"****Doc\_Appendix****"* **template=*"Content"***/>

Однако, получается, что два этих типа отличаются только именем, но эти имена уникальны в пределах всей системы, а не только родительского типа документа. И поэтому программный код, различающий эти типы вложений, нельзя обобщить на разные типы документов. Поэтому здесь лучше пользоваться 1-м способом, сочетая его с "почкованием" дерева шаблонов, например:

От исходного шаблона ***"Content"***делаем наследников ***"Content\_Body"***и ***"Content\_Appendix"***и уже по ним делаем типы вложений в документах:   
<attachment-type name=*"Doc\_Content"* **template=*"Content\_Body"***/>   
<attachment-type name=*"Doc\_Appendix"* **template=*"Content\_Appendix"***/>

В программном коде для любого вложения можно вычислить не только тип его самого, но и "обобщенный/абстрактный" тип - по его шаблону согласно конфигурации, причем с учетом иерархии.

Наконец, сюда подходит и 3-й способ: для нескольких реквизитов/виджетов использовать вложения одного типа, но с дополнительным прикладным атрибутом, (например, "вид вложения", "назначение" и т.п.).   
Тогда по чести Содержания и Приложений в конфиге останется одна строка   
<attachment-type name=***"Doc\_Content"*** **template=*"Content"***/> , но в шаблон и в тип вложения "Doc\_Content" надо добавить атрибуты, по которым можно различать вложения для разных реквизитов. На данный момент можно сказать, рекомендуется именно этот способ. Тогда в обозримой перспективе понадобятся именно три шаблона: Content, FileSignature и Rendition.

## Правила/ограничения на типы вложений в документах

Исходя из предыдущего раздела. для состава типов вложений в типе документа имеет смысл установить следующие правила/ограничения:

1. В элементе <attachment-type .../> всегда задаем атрибут template с именем шаблона, отличным от дефолтного. Это нужно чтобы всегда можно было выделить именно такие вложения, даже если у них дефолтная структура.
2. В типе ДО с учетом наследования значение атрибута template у заданных типов вложений не должно повторяться. Тогда в прикладном коде по обобщенному/абстрактному типу вложения (т.е. по шаблону) всегда можно будет определить его конкретный тип в данном типе документа, и поэтому можно будет основываться не на конкретных типах вложений, а на обобщенных/абстрактных.

## Сервис вложений уровня приложения (CM6-Base)

Абстрактным типом вложения (АТВ) далее называем шаблон (имя шаблона?), по которому сделан "конкретный" тип вложения (КТВ). Возможна иерархия АТВ. Можно сказать, что АТВ определяет прикладной смысл таких вложений, а КТВ - детали внутренней реализации.

С учетом соображений предыдущего раздела можно и полезно реализовать несколько функций для работы с вложениями одного АТВ в разных типах документов.

Например:

1. Получить КТВ по АТВ (тип ДО-родителя; АТВ) - чтобы потом с КТВ можно было вызывать любые методы платформенного сервиса вложений;
2. Проверить вложение на соответствие АТВ (вложение, АТВ) - с учетом иерархии АТВ - для применения каких-либо правил к вложениям определенного АТВ;
3. Обёртки методов платформенного сервиса вложений, абстрагированные от КТВ - для тех процедур, которым параметром передается КТВ обрабатываемых вложений, например метод обертка метода createAttachmentDomainObjectFor будет получать в качестве attachmentType соответствующий АТВ, но вернет объект уже конкретного типа, подходящего родителю. Список методов:   
   DomainObject createAttachmentDomainObjectFor(Id objectId, String attachmentType);  
   List<DomainObject> findAttachmentDomainObjectsFor(Id domainObjectId, String attachmentType);  
   DomainObject copyAttachment(Id attachmentDomainObjectId, Id destinationDomainObjectId, String destinationAttachmentType);  
   List<DomainObject> copyAttachments(List<Id> attachmentDomainObjectIds, Id destinationDomainObjectId, String destinationAttachmentType);  
   List<DomainObject> copyAttachmentsFrom(Id sourceDomainObjectId, String sourceAttachmentType, Id destinationDomainObjectId, String destinationAttachmentType);  
   List<DomainObject> copyAllAttachmentsFrom(Id sourceDomainObjectId, Id destinationDomainObjectId, String destinationAttachmentType);  
   List<DomainObject> copyAllAttachmentsFrom(Id sourceDomainObjectId, Id destinationDomainObjectId, Map<String, String> attachmentTypeMap);
4. Метод копирования вложений специально между документом и его версиями: пока (до реализации CMFIVE-4291) - с обычным копированием, потом обновим на экономное копирование без размножения файлов.

Надо сказать что все или большинство методов копирования в абстрактной обёртке вообще не нуждаются в указании типа вложения, т.к. абстрактный тип при копировании вложений между разными документами обычно требуется сохранить.

# Ограничения принятого решения

1. ДО вложение с одним файлом не могут быть одновременно вложены в документы разных типов (ДО разных типов).
2. Системные атрибуты файла, хранимые в ДО вложения, будут дублироваться во всех вложениях с одним файлом. Это совсем небольшая денормализация в БД, к тому же модифицировать их приложениям всё-равно запрещено.
3. Прикладные атрибуты, которые будут добавляться во вложения, сейчас можно задавать только при создании вложения. Для возможности их изменения нужна будет небольшая доработка (см.ниже).

# Доработки платформы

1. [CMFIVE-4291](https://jira.inttrust.ru:8443/browse/CMFIVE-4291) Поддержка сервисом вложений размножения ДО Вложение без копирования файла (файл обрабатывается в режиме copy-on-write)
2. [CMFIVE-4295](https://jira.inttrust.ru:8443/browse/CMFIVE-4295#rpctoken=171941596) Обеспечить возможность модификации прикладных полей ДО Вложения (из приложений)
3. Возможно, потребуется доработка виджетов конфигурируемого GUI платформы с целью их дополнительной адаптации по месту использования в формах. В частности, чтобы одну конфигурацию формы можно было бы использовать для работы с вложениями в разных типах документов.

# Примеры конфигурации

На примере ЭП показана дополнительная ссылка из одного типа вложения на другой.   
Она там указана с произвольным типом (\*), но это не страшно, т.к. все типы вложений к одному типу документа всё-равно будут известны. Зато позволяет не плодить дополнительных шаблонов для типов вложений с каждым конкретным типом такой ссылки.

## Конфигурация ДО

<!-- Тип вложения реквизита "Содержание" -->

<domain-object-type name=*"Content"* template=*"true"* extends=*"Attachment"*>

<fields>

<!-- дополнительные поля по необходимости -->

</fields>

</domain-object-type>

<!-- Тип вложения для ЭП -->

<domain-object-type name=*"FileSignature"* template=*"true"* extends=*"Attachment"*>

<fields>

**<reference name=*"Source"* type=*"\*"* not-null=*"true"*/>**

<reference name=*"Signer"* type=*"Person"* not-null=*"true"*/>

<date-time name=*"SignTime"* not-null=*"true"*/>

</fields>

</domain-object-type>

<domain-object-type name=*"Document"*>

<fields>

<!-- ... -->

</fields>

<attachment-types>

<attachment-type name=*"Doc\_Content"* template=*"Content"*/>

<attachment-type name=*"Doc\_FileSignature"* template=*"FileSignature"*/>

</attachment-types>

</domain-object-type>

. . .

## Конфигурация прав доступа

Делается элементарно - с заимствованием ACL по ссылке из вложения на документ.

Поле со ссылкой на родителя в конфигурации типа вложения не задаётся, оно определяется **автоматически**!   
Его имя равно имени типа ДО родителя, ну и тип ссылки - тоже тип родителя.

**<domain-object-type name=*"Document"*>**   
        <!-- ... -->   
        <attachment-types>   
            <attachment-type name=*"Doc\_Content"* template=*"Content"*/>   
            <attachment-type name=*"Doc\_FileSignature"* template=*"FileSignature"*/>   
        </attachment-types>   
    </domain-object-type>

Матрица доступа к вложениям будет заимствоваться так:

<access-matrix type=*"Doc\_Content"* matrix-reference-field=*"****Document****"* />  
 <access-matrix type=*"Doc\_FileSignature"* matrix-reference-field=*"****Document****"* />

## Пример из проекта Атлас

Вложение типа **"attach"** автоматически получает поле-ссылку с именем **"Docm\_Attachments"** и таким же типом,   
и по этой ссылке наследуются права доступа.

Прикладной ДО:

<!-- ДО Вложение Документа -->  
 <domain-object-type name=**"Docm\_Attachments"** initial-status="Active" audit-log="true">  
 <fields>  
 <!-- поля -->  
 </fields>  
 <attachment-types>  
 <attachment-type name=**"attach"**/>  
 <!-- еще типы вложений -->  
 </attachment-types>  
 </domain-object-type>

Права доступа к вложению, заимствуемые от "родителя"

<access-matrix type=**"attach"** matrix-reference-field=**"Docm\_Attachments"** />