ФЕДЕРАЛЬНОЕ АГЕНТСТВО

ПО ТЕХНИЧЕСКОМУ РЕГУЛИРОВАНИЮ И МЕТРОЛОГИИ

НАЦИОНАЛЬНЫЙ СТАНДАРТ РОССИЙСКОЙ ФЕДЕРАЦИИ

ГОСТ Р ИСО/МЭК 17826— 2015

Информационные технологии

ИНТЕРФЕЙС УПРАВЛЕНИЯ ОБЛАЧНЫМИ ДАННЫМИ (CDMI)

ISO/IEC 17826:2012
Information technology
Cloud Data Management Interface (CDMI)
(IDT)

Издание официальное

Москва
Стандартинформ
2015

Предисловие

1 ПОДГОТОВЛЕН Обществом с ограниченной ответственностью «Информационно-аналитический вычислительный центр» (ООО «ИАВЦ») на основе собственного аутентичного перевода на русский язык международного стандарта, указанного в пункте 4

2 ВНЕСЕН Техническим комитетом по стандартизации ТК 22 «Информационные технологии»

3 УТВЕРЖДЕН И ВВЕДЕН В ДЕЙСТВИЕ Приказом Федерального агентства по техническому регулированию и метрологии от 29 мая 2015 г. № 467-ст

4 Настоящий стандарт идентичен международному стандарту ИСО/МЭК 17826:2012 «Информационные технологии. Интерфейс управления облачными данными (CDMI)» (ISO/IEC 17826:2012 «Information technology – Cloud Data Management Interface (CDMI)»).

При применении настоящего стандарта рекомендуется использовать вместо ссылочных международных стандартов соответствующие им национальные стандарты Российской Федерации, сведения о которых приведены в дополнительном приложении ДА

5 ВВЕДЕН ВПЕРВЫЕ

Правила применения настоящего стандарта установлены в ГОСТ Р 1.0–2012 (раздел 8). Информация об изменениях к настоящему стандарту публикуется в ежегодном (по состоянию на 1 января текущего года) информационном указателе «Национальные стандарты», а официальный текст изменений и поправок – в ежемесячном информационном указателе «Национальные стандарты». В случае пересмотра (замены) или отмены настоящего стандарта соответствующее уведомление будет опубликовано в ближайшем выпуске ежемесячного информационного указателя «Национальные стандарты». Соответствующая информация, уведомления и тексты размещаются также в информационной системе общего пользования – на официальном сайте Федерального агентства по техническому регулированию и метрологии в сети Интернет (www.gost.ru)

Содержание

Введение

Настоящий стандарт CDMI™ предназначен для использования разработчиками приложений, использующих облачное хранение данных. Он документирует доступ к облачному хранилищу и управление данными, хранящимися там.

Настоящий стандарт состоит из следующих разделов:

1 – Область применения	Определяет основные задачи данного документа
2 – Нормативные ссылки	Перечисляет ссылки на нормативные документы, связанные с данным документом
3 – Термины и определения	Устанавливает терминологию, используемую в данном документе
4 – Соглашения	Определяет формат описания интерфейсов и типографические соглашения, принятые в данном документе
5 – Обзор облачного хранения	Дает краткое описание облачного хранения и основные предпосылки, лежащие в основе настоящего стандарта как модели операций
6 – Обычные операции	Дает пример ресурсов, к которым может быть предоставлен доступ, и представления, используемые для их изменения
7 – Стандарт интерфейса	Описывает коды состояния HTTP, типы объектов интерфейса управления облачными данными (CDMI), ссылки на объекты и операции с объектами
8 – Операции с ресурсами объекта данных	Определяет стандартные операции с ресурсами объектов данных
9 – Операции с ресурсами вида объект – контейнер	Определяет стандартные операции с ресурсами объектов–контейнеров
10 – Операции с ресурсами объекта-домена	Определяет стандартные операции с ресурсами объектов-доменов
11 – Операции с ресурсами объекта передачи	Определяет стандартные операции с ресурсами объектов очередей
12 – Операции с ресурсами объекта-опции	Определяет стандартные операции с ресурсами объектов-опций
13 – Экспортируемые протоколы	Обсуждает возможные сценарии использования виртуальными машинами облачных вычислительных сред протоколов, экспортированных контейнерами CDMI.
14 – Снимки состояния	Обсуждает доступ к снимкам состояния через контейнеры CDMI
15 – Сериализация/десериализация	Обсуждает сериализацию и десериализацию, включая импорт и экспорт сериализованных данных в рамках CDMI
16 – Метаданные	Определяет стандартные метаданные, используемые в интерфейсе
17 – Управление отложенным удалением и удержанием	Описывает необязательные процедуры управления отложенным удалением, реализуемые в функциях системы управления
18 – Спецификация условий запросов	Описывает структуру спецификации условий запросов в нотации JSON
19 – Спецификация результатов	Описывает стандартизированные механизмы определения подмножеств содержимого объекта CDMI
20 – Ведение журналов событий	Описывает функции ведения журналов событий CDMI для функций объектов, событий безопасности, событий управления данными и очередей
21 – Очереди сообщений	Описывает, как клиенты CDMI могут эффективно обнаружить изменения системы
22 – Очереди запросов	Описывает, как клиенты CDMI могут эффективно обнаружить, какое содержание соответствует заданному набору критериев запроса метаданных или критерию поиска по всем полям метаданных
Приложение А – (обязательное) Безопасность передачи	Определяет требования к обеспечению безопасной передачи данных по протоколу HTTP для передачи CDMI сообщений
Приложение ДА – (справочное)	Приложение ДА (справочное) Сведения о соответствии ссылочных международных стандартов национальным стандартам Российской Федерации
Библиография	

НАЦИОНАЛЬНЫЙ СТАНДАРТ РОССИЙСКОЙ ФЕДЕРАЦИИ

Информационные технологии

ИНТЕРФЕЙС УПРАВЛЕНИЯ ОБЛАЧНЫМИ ДАННЫМИ (CDMI)

Information technology. Cloud Data Management Interface (CDMI)

Дата введения — 2016—06—01

1 Область применения

Настоящий стандарт CDMI™ описывает интерфейс доступа к облачному хранилищу и управления хранимой там информацией. Настоящий стандарт предназначен для разработчиков, использующих или внедряющих облачное хранение данных.

2 Нормативные ссылки

В настоящем стандарте использованы нормативные ссылки на следующие стандарты. Предоставленные спецификации, не относящиеся к документам ИСО/МЭК, МЭК, ИСО и ITU, применимы лишь в контексте настоящего стандарта. Ссылка на эти документы не предоставляет им нового статуса в системе ИСО/МЭК. В частности, это не дает цитированным ссылкам статуса международного стандарта.

ИСО 3166 Коды для представления названий стран и единиц их административно-территориального деления (части 1, 2 и 3) (ISO 3166, Codes for the representation of names of countries and their subdivisions)

ИСО 4217:2008 Коды для представления валют и фондов (ISO 4217:2008, Codes for the representation of currencies and funds)

ИСО 8601:2004 Элементы данных и форматы для обмена информацией. Обмен информацией. Представление дат и времени (ISO 8601:2004, Data elements and interchange formats – Information interchange – Representation of dates and times)

ИСО/МЭК 9594-8:2008 Информационные технологии. Взаимосвязь открытых систем. Директория. Часть 8. Структура сертификата на открытый ключ и атрибуты (ISO/IEC 9594-8:2008, Information technology – Open Systems Interconnection – The Directory: Publickey and attribute certificate frameworks)

ИСО/МЭК 14776-414 Информационные технологии. Интерфейс для малых вычислительных систем (SCSI). Часть 414. Структурная модель 4 (SAM-4) (ISO/IEC 14776-414, SCSI Architecture Model - 4 (SAM-4))

IEEE Std 1003.1, 2004, POSIX ERE Открытая группа, основные спецификации, Вып. 6 (IEEE Std 1003.1, 2004, POSIX ERE, The Open Group, Base Specifications Issue 6) - http://www.unix.org/version3/ieee_std.html

RFC 2045 Многоцелевые расширения почты для интернета (MIME) Часть 1. Формат тела письма в интернете (RFC 2045, Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies) - http://www.ietf.org/rfc/rfc2045.txt

RFC 2119 Ключевые слова для обозначения уровня требований в RFC (RFC 2119, Key Words for Use in RFCs to Indicate Requirement Levels) - http://tools.ietf.org/html/rfc2119

RFC 2246 Протокол TLS версии 1.0 (RFC 2246, The TLS Protocol Version 1.0) - http://www.ietf.org/rfc/rfc2246.txt

RFC 2578 Структура управляющей информации, версия 2 (RFC 2578, Structure of Management Information Version 2, SMIv2) - http://www.ietf.org/rfc/rfc2578.txt

RFC 2616 Протокол передачи гипертекста 1.1 (RFC 2616, Hypertext Transfer Protocol -- HTTP/1.1) - http://www.ietf.org/rfc/rfc2616.txt

RFC 2617 Аутентификация HTTP: Базовая аутентификация доступа и аутентификация доступа с использованием дайджеста (RFC 2617, HTTP Authentication: Basic and Digest Access Authentication) - http://www.ietf.org/rfc/rfc2617.txt

Издание официальное

RFC 3280 Профиль инфраструктуры открытых ключей X.509 Интернета: сертификат и список отозванных сертификатов (RFC 3280, Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile) - http://www.ietf.org/rfc/rfc3280.txt

RFC 3530 Протокол сетевой файловой системы версии 4 (RFC 3530, Network File System (NFS) Version 4 Protocol) - http://www.ietf.org/rfc/rfc3530.txt

RFC 3720 Интерфейс для малых вычислительных систем в интернете (RFC 3720, Internet Small Computer Systems Interface, iSCSI) - http://www.ietf.org/rfc/rfc3720.txt

RFC 3986 Универсальный идентификатор ресурса: общий синтаксис (RFC 3986, Uniform Resource Identifier (URI): Generic Syntax) - http://www.ietf.org/rfc/rfc3986.txt

RFC 4346 Протокол безопасности транспортного уровня, версия 1.1 (RFC 4346, The Transport Layer Security (TLS) Protocol Version 1.1) - http://www.ietf.org/rfc/rfc4346.txt

RFC 4627 Тип медиа application/JSON для объектной нотации JavaScript (RFC 4627, The Application/JSON Media Type for JavaScript Object Notation (JSON)) - http://www.ietf.org/rfc/rfc4627.txt

RFC 4648 Кодировки данных Base16, Base32, и Base64 (RFC 4648, The Base16, Base32, and Base64 Data Encodings) http://www.ietf.org/rfc/rfc4648.txt

RFC 4918 Расширения HTTP для авторства и версионирования в сети Интернет (RFC 4918, HTTP Extensions for Web Distributed Authoring and Versioning (WebDAV)) - http://www.ietf.org/rfc/rfc4918.txt

RFC 5246 Протокол безопасности транспортного уровня, версия 1.2 (RFC 5246, The Transport Layer Security (TLS) Protocol Version 1.2) - http://www.ietf.org/rfc/rfc5246.txt

RFC 6208 Типы медиа для интерфейса управления облачными данными (RFC 6208, Cloud Data Management Interface (CDMI) Media Types) - http://www.ietf.org/rfc/rfc6208.txt

3 Термины и определения

В настоящем стандарте применены следующие термины с соответствующими определениями:

3.1 **список управления доступом** (Access Control List) ACL. Постоянный список, состоящий из записей управления доступом (Access Control Entries, ACE), который перечисляет права доступа принципалов (пользователей и групп) к ресурсам.

3.2 **CDMI™**: Интерфейс управления облачными данными.

3.3 **CIFS**: Единая файловая система для Интернета.

3.4 **облачное хранение** (cloud storage): См. «Хранение данных как служба».

3.5 **CRC**: Циклический избыточный код.

3.6 **CRUD**: Создать, получить, изменить, удалить (create, retrieve, update, delete).

3.7 **хранение данных как служба** (data storage as a Service) DaaS: Организация сетевых служб виртуализированного хранения и доступа к данным, основанная на требовании заданного уровня службы, что снимает границы масштабируемости; является самообеспечивающимся или не требующем обеспечения и оплачивается в зависимости потребления.

3.8 **домен** (domain): Совместная база данных прав пользователей, содержащая пользователей, группы, их политики безопасности и связанную информацию об учетных записях.

Примечание — Каждый объект CDMI принадлежит к единственному домену, и каждый домен содержит списки сопостовления имен пользователей и информацию об учетных записях.

3.9 **целостность в конечном итоге** (eventual consistency): Поведение транзактной системы, не предоставляющее гарантию связности в каждый момент времени, что улучшает доступность системы и ее устойчивость к распадению сети.

3.10 **HTTP**: Протокол передачи гипертекста.

3.11 **инфраструктура как служба** (Infrastructure as a Service) IaaS: Предоставление по сети соответствующим образом сконфигурированного виртуального вычислительного окружения, основанное на запрошенном уровне услуг.

Примечание — Обычно IaaS является самообеспечивающейся или не требующей обеспечения и оплачивается в зависимости потребления.

3.12 **iSCSI**: Интерфейс для малых вычислительных систем в интернете (см. RFC 3720).

3.13 **LUN**: Номер логического устройства (см. ИСО/МЭК 14776-414).

3.14 **MIME**: Многоцелевые расширения для почты в интернете (см. RFC 2045).

3.15 **NFS**: Сетевая файловая система (см. RFC 3530).

3.16 **объект** (object): Сущность, имеющая идентификатор объекта (ID), уникальный URI, и обладающая состоянием.

Примечание — Существуют следующие типы объектов CDMI: объекты данных, контейнеры, опции, домены, очереди.

3.17 **идентификатор объекта** (object identifier): Глобально-уникальное значение, соотнесенное с объектом в момент его создания.

3.18 **OCCI**: Интерфейс открытых облачных вычислений (Open Cloud Computing Interface) (см. спецификацию OCCI).

3.19 **платформа как служба** PaaS (Platform as a Service): Предоставление по сети виртуализированной среды программирования, состоящей из развернутого стека приложения, основанного на виртуальной вычислительной среде.

Примечание — Обычно PaaS основана на IaaS, является самообеспечивающейся или не требующей обеспечения и оплачивается в зависимости фактического использования.

3.20 **POSIX**: Интерфейс переносимой операционной системы (Portable Operating System Interface) (см. IEEE Std 1003.1).

3.21 **частное облако** (private cloud): Предоставление SaaS, PaaS, IaaS, и/или DaaS ограниченному числу пользователей, обычно принадлежащих к одной и той же организации.

Примечание — Частные облака создаются в целях безопасности.

3.22 **общественное облако** (public cloud): Предоставление SaaS, PaaS, IaaS, и/или DaaS потенциально неограниченному количеству пользователей.

3.23 **передача репрезентативного состояния** (Representational State Transfer) REST: Особый набор принципов определения, адресации и взаимодействия с ресурсами, адресуемыми посредством URI (см. [REST]).

3.24 **RPO**: Целевая точка восстановления (recovery point objective).

3.25 **RTO**: Целевое время восстановления (recovery time objective).

3.26 **уровень службы** (service level): Целевые показатели производительности службы.

3.27 **программы как служба** (Software as a Service) SaaS: Предоставление по сети доступ к приложению по запросу.

3.28 **тонкое резервирование** (thin provisioning): Технология распределения физической емкости тома или файловой системы по мере записи данных приложением вместо предварительного резервирования.

3.29 **универсальный идентификатор ресурса** (Uniform Resource Identifier) URI: Короткая последовательность символов, идентифицирующая абстрактный или физический ресурс (см. RFC 3986).

3.30 **виртуализация** (virtualization): Представление ресурсов, как если бы они были физическими, тогда как на самом деле они отделены от базовых физических ресурсов.

3.31 **WebDAV**: Набор расширений и дополнений к протоколу HTTP, поддерживающих совместную работу пользователей над редактированием файлов и управление файлами на удаленных веб-серверах (см. RFC 4918).

3.32 **XAM**: Расширяемый метод доступа (eXtensible Access Method) (см. INCITS 464-2010).

4 Соглашения

4.1 Формат интерфейса

Каждое описание интерфейса состоит из девяти компонентов, описанных в таблице 1.

Таблица 1 – Формат интерфейса

Компонент	Описание
Краткий обзор	Семантика GET, PUT, POST и DELETE
Отсроченное завершение создания	В случае длительных операций описывает поведение, если операция не завершается немедленно
Опции	Описание поддерживаемых операций
Заголовки запроса	Заголовки запроса, например, Accept, Authorization, Content-Length, ContentType, X-CDMI-Specification-Version
Тело сообщения-запроса	Описание содержимого тела сообщения-запроса

Окончание таблицы 1

Компонент	Описание
Заголовки ответа	Заголовки ответа, например, Content-Length, Content-Type
Тело сообщения-ответа	Описание содержимого тела сообщения-ответа
Статус ответа	Список кодов состояния HTTP
Пример	Пример операции

4.2 Типографские соглашения

Все исходные тексты программ и сообщений показаны следующим шрифтом:

Пример –
PUT /MyContainer/MyDataObject.txt HTTP/1.1
Host: cloud.example.com
Accept: application/cdmi-object
Content-Type: application/cdmi-object
X-CDMI-Specification-Version: 1.0.2
{
«mimetype» : «text/plain»,
«metadata» : {
},
«value» : «This is the Value of this Data Object»
}

4.3 Требования к телу сообщений запросов и ответов

В таблицах, представляющих тела запросов и ответов, колонка «Требования» содержит одно из трех значений:

- обязательное. Значение, описанное в данной строке, должно быть указано.
- условное. Если выполнены условия, указанные в ячейке «Описание» данной строки (слева от ячейки «Требования») то значение, описанное в данной строке, должно быть указано. В противном случае, оно может быть указано, если это не запрещено в Описании (в этом случае оно должно отсутствовать);
- необязательное. Значение, описанное в данной строке, может быть указано.

4.4 Требования к ключевым словам

В данном международном стандарте, ключевые слова, указанные в табл. 2, должны интерпретироваться как указано в RFC 2119.

Таблица 2 – Требования к ключевым словам

Ключевые слова	Описание
Должно, требуется (shell, must, required)	Действие, описанное одним из этих слов, безоговорочно требуется
Не должно, не может (shall not, must not)	Действие, описанное одним из этих слов, безоговорочно запрещено
Следует, рекомендуется (should, recommended)	В некоторых обстоятельствах возможно игнорировать действие, описанное одним из этих ключевых слов, но основания для этого должны быть полностью взвешены и осознанны
Не следует, не рекомендуется (should not, not recommended)	В некоторых обстоятельствах возможно выполнить действие, описанное одним из этих ключевых слов, но основания для этого должны быть полностью взвешены и осознанны
Может, опционально (may, optional)	Действие, описанное одним из этих ключевых слов, полностью опционально. Одни разработчики могут включать данное действие, потому что оно требуется определенному сегменту пользователей или может расширить возможности продукта. Другие разработчики могут опустить ту же опцию. Реализация, которая не включает некоторую опцию, должна иметь возможность взаимодействовать с реализацией, включающей эту опцию. Аналогично, реализация, включающая некоторую опцию, должна иметь возможность взаимодействовать с реализацией, не включающей данную опцию (за исключением, разумеется, функциональности данной опции)

5 Обзор облачного хранения

5.1 Введение

При обсуждении облачного хранения и стандартов важно различать ресурсы, предлагаемые как службы. Эти ресурсы открыты пользователям как функциональные интерфейсы (например, пути к данным) и управляются интерфейсами управления (например, пути управления). Настоящий стандарт представляет различные типы интерфейсов, которые входят в состав существующих реализаций, и указывает, как они взаимосвязаны. Настоящий стандарт определяет модель интерфейсов, которая может быть отражена на различные реализации, и модель, которая послужит основой для интерфейсов облачного хранения в будущем.

Другая важная концепция, заложенная в данном международном стандарте – метаданные. При управлении большими объемами данных с различными требованиями, метаданные – удобный механизм для выражения этих требований таким образом, что нижележащие службы данных могут обращаться с данными в соответствии с требованиями.

Привлекательность использования облачного хранения определяется рядом характеристик, общих с другими облачными службами: оплата по факту использования, кажущаяся неограниченной емкость (эластичность) и простота использования/управления. Поэтому важно, чтобы любой интерфейс облачного хранения поддерживал данные характеристики, тем самым обеспечивая реализацию множества сценариев использования.

5.2 Что такое облачное хранение?

Использование термина «облако» связано с тем, что эти новые модели произошли от графического представления архитектур, использующих облако как символ сети. Облако абстрактно обозначает связь любого компьютера сети с любым другим компьютером. В этой абстракции сетевое соединение в облаке не определяет, как именно оно устанавливается.

Облако абстрагирует сложность, предоставляя простую основу для построения новых функций. Обобщенная облачная модель расширяет эту основу посредством добавления набора ресурсов. Важная часть облачной модели – концепция пула ресурсов, который при необходимости создается из небольших фрагментов. Это стало возможным благодаря сравнительно недавней инновации – виртуализации.

Таким образом, облачное хранение есть предоставление по запросу виртуализированного хранилища. Для обозначения этого используется формальный термин «Хранение данных как служба» (Data storage as a Service, DaaS).

5.3 Хранение данных как служба

Абстрагирование хранения данных от набора интерфейсов служб и предоставление его по запросу допускает большое количество конкретных реализаций и сценариев работы. Единственный тип хранения, который исключается из данного определения – тот, который предоставляется на основе фиксированных приращений, а не основанный на запросах.

Важная часть реализации любой системы DaaS – поддержка унаследованных клиентов. Эта поддержка включена в действующие стандартные протоколы, такие как iSCSI (и другие) для блочного и CIFS/NFS или WebDAV для файлового и сетевого хранения, как показано на рисунке 1.

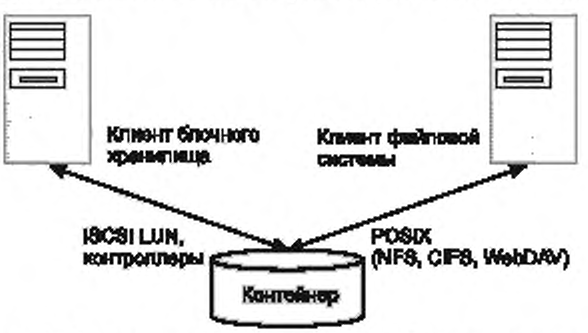

Рисунок 1 – Существующие стандарты интерфейсов хранилищ данных

Различие между приобретением обычного устройства и облачного хранилища состоит не в функциональных интерфейсах, а в том, что в последнем случае хранилище предоставляется по запросу. Пользователь платит либо за то, чем фактически пользуется, либо за то, что запросил для использования. В случае блочного хранилища размер запрошенного объема измеряется в виртуальных томах или номерах логического устройства (LUN). В случае файловых протоколов, единицей измерения является файл. В любом случае, действительный объем хранилища может быть тонко зарезервирован и оплачен исходя из действительного использования. Для дальнейшего снижения фактически используемого объема могут использоваться такие службы данных, такие как сжатие или удаление дубликатов.

Управление таким хранилищем в случае стандартных интерфейсов хранения данных обычно осуществляется через API или (чаще) через пользовательский интерфейс администратора в браузере по выделенному каналу связи. Этот интерфейс по выделенному каналу связи может использоваться для запуска других служб данных (например, создания моментального снимка состояния или клонирования).

В рамках такой модели фактическое хранилище, доступ к которому реализуется посредством интерфейса по выделенному каналу связи, абстрагируется концепцией контейнера. Контейнер – не только полезная абстракция места хранения, он также служит для группировки хранимых данных и точкой управления для применения служб данных к такой группе.

Каждый объект данных создается, получается, обновляется и уничтожается как отдельный ресурс. В этом интерфейсе контейнер, если он используется, служит для удобства, группируя объекты данных. Ничто не препятствует создавать иерархические контейнеры, хотя каждая конкретная реализация может поддерживать лишь единственный уровень (см. рисунок 2).

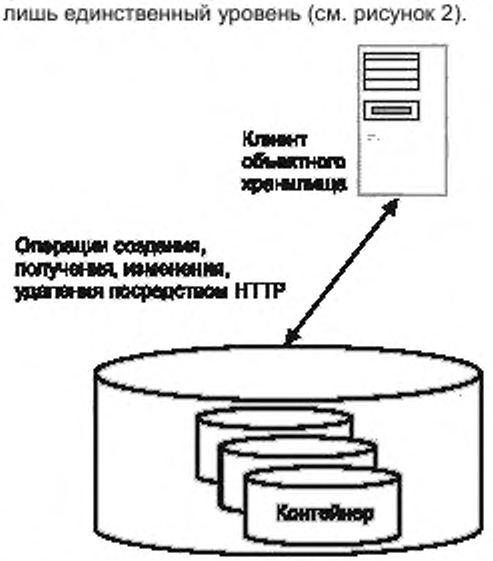

Рисунок 2 – Интерфейсы хранилища для данных клиента объектного хранилища

5.4 Управление данными для облачного хранения

Многие из первых реализаций облачного хранилища фокусировались на хранении данных с минимальными гарантиями качества хранения и игнорировали большинство других типов служб данных. Однако, учитывая нужды корпоративных приложений, использующих облачное хранение, существует необходимость совершенствовать качество предоставления услуг и внедрять дополнительные службы данных.

Облачное хранение может потерять свои преимущества абстрактности и простоты, если появятся новые службы данных, требующие сложного управления. Клиенты облачных хранилищ, вероятно, будут не согласны тратить время на дополнительные операции (например, настраивать через специализированные интерфейсы резервное копирование по расписанию, разворачивать службу данных индивидуально для каждого элемента данных).

Поддержка метаданных в интерфейсе облачного хранилища и указание, как система хранения и ее метаданные интерпретируются для соблюдения требований к данным, могут сохранить простоту облачного хранения и при этом реализовать необходимые для корпоративных приложений функции.

Пользовательские метаданные сохраняются в облаке и могут использоваться для поиска объектов данных и контейнеров с использованием поисковых запросов по значениям метаданных. Схема метаданных может определяться конкретным приложением, доменом или пользователем. Подробнее о поддержке пользовательских метаданных см. 16.2.

Метаданные системы хранения формируются/интерпретируются реализацией облака и низкоуровневыми функциями хранилища (например, статистика изменения и доступа, управление доступом). Подробнее о поддержке метаданных системы хранения см. 16.3.

Метаданные системы данных интерпретируются реализацией облака как требования к данным, которые управляют операциями соответствующих служб данных. Это может относиться к объединению данных в контейнере или к индивидуальным объектам данных, если реализация поддерживает такую детализацию. Подробнее о поддержке метаданных системы данных см. 16.4.

5.5 Управление данными и контейнерами

Нет причины, по которой управление данными и контейнерами должно осуществляться различными интерфейсами.

Поэтому использование метаданных расширяется от применения к индивидуальным элементам данных на контейнеры данных. Таким образом, любые данные, помещенные в контейнер, наследуют метаданные системы данных контейнера, в который они помещаются. При создании нового контейнера внутри существующего контейнера, новый контейнер наследует настройки метаданных родительского контейнера. После создания элемента данных, отдельные метаданные системы данных могут быть перекрыты метаданными, заданными на уровне контейнера или индивидуального элемента данных.

Даже если предоставляемый интерфейс не поддерживает установку метаданных для индивидуальных элементов, метаданные могут быть настроены для контейнеров. В этом случае, интерфейс не предоставляет механизма для перекрывания метаданных, которые индивидуальный элемент наследует от родительского контейнера. Для интерфейсов, основанных на файлах, поддерживающих расширенные атрибуты (например, CIFS, NFSv4), эти атрибуты могут быть использованы для указания на то, что метаданные системы данных должны перекрыть метаданные контейнера.

5.6 Эталонная модель интерфейсов облачного хранения

Эталонная модель облачного хранилища показана на рисунке 3.

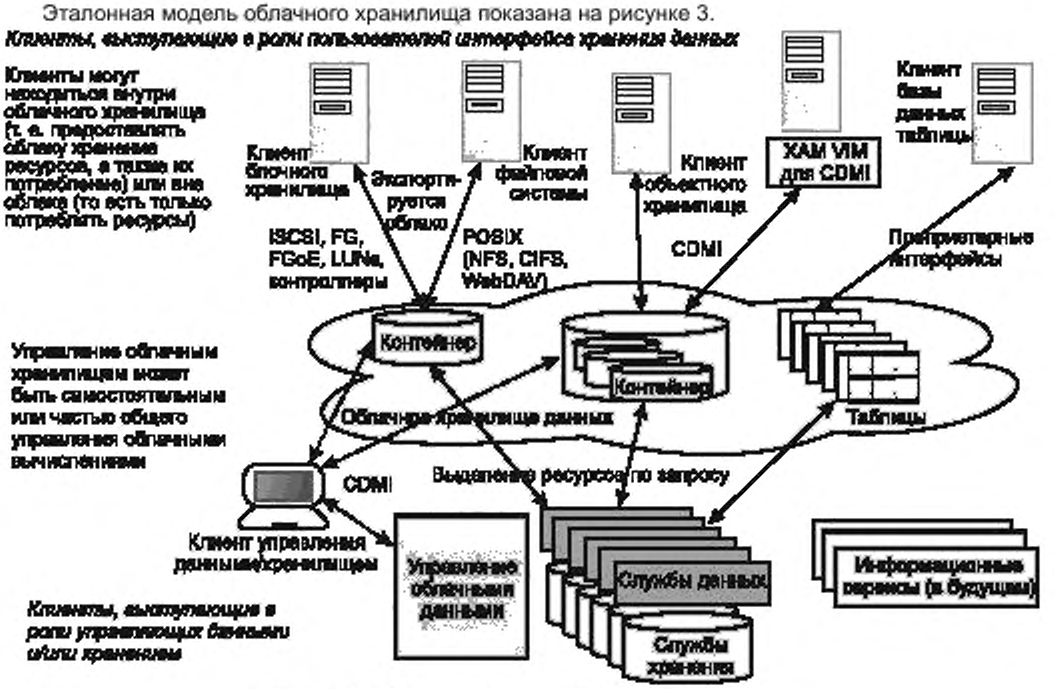

Рисунок 3 – Эталонная модель облачного хранилища

Эта модель показывает множество типов интерфейсов облачного хранения, способных поддерживать как существующие, так и новые приложения. Все эти интерфейсы позволяют предоставлять хранилище данных из пула ресурсов по запросу. Емкость хранилища определяется емкостями пулов хранилищ, предоставляемых службами хранения.

Службы данных применяются к индивидуальным элементам данных в соответствии с метаданными системы данных. Метаданные определяют требования к данным на основании индивидуальных элементов или групп элементов (контейнеров).

5.7 Интерфейс управления облачными данными

Интерфейс управления облачными данными (CDMI™), показанный на рисунке 3, может использоваться для создания, доступа, изменения или удаления объектов в облаке. Функции CDMI позволяют:

- клиентам определить набор опций, поддерживаемых реализацией облака;
- управлять контейнерами и данными в них;
- устанавливать метаданные для контейнеров и объектов данных в них.

Настоящий стандарт разделяет стандартные операции на два типа: использующие тип содержимого CDMI в теле HTTP и не использующие их. Несмотря на то, что многие данные доступны посредством операций обоих типов, предоставление и тех, и других операций позволяет взаимодействовать с провайдером CDMI как клиентам, поддерживающим CDMI, так и не поддерживающим CDMI.

CDMI может также использоваться для административных или управляющих операций для работы с контейнерами, доменами, безопасным доступом и информации мониторинга/оплаты, даже для хранилищ, доступных через унаследованные или проприетарные протоколы. Опции соответствующих служб данных и хранения предоставляются для того, чтобы клиенты могли понимать реализацию.

Совместимые облачные реализации могут поддерживать подмножество CDMI, если они предоставляют через интерфейс сведения об ограничениях опций.

Настоящий стандарт использует принципы REST в разработке интерфейса, насколько это возможно (см. [REST]).

CDMI определяет как средства для управления данными, так и средства для сохранения и получения данных. Средства, которыми осуществляется сохранение и получение данных, обозначаются как путь к данным. Средства управления данными обозначаются как путь к управлению. CDMI описывает как путь к данным, так и путь к управлению.

Не требуется, чтобы только CDMI использовался для путей к данным. CDMI может управлять свойствами облачного хранилища для произвольного интерфейса пути к данным (например, стандартизованного или проприетарного некоторого поставщика).

Метаданные контейнера используются для конфигурирования требований к данным хранилища, предъявляемых контейнером посредством экспортируемых протоколы (например, блочный или файловый протокол). Для реализаций, основанных на файловой системе для блочного хранения данных (например, iSCSI), контейнер CDMI предоставляет полезную абстракцию для представления метаданных системы данных для данных и структур, управляющих экспортированными протоколами.

Реализация облака может также поддерживать домены, позволяющие сопоставлять административные права владения хранимым объектам. Домены (наряду с другими возможностями) позволяют стандарту:

- определять соответствие пользовательских полномочий принципалами из списков контроля доступа (ACL);
- выдавать специальные привилегии, связанные с облаком;
- делегировать авторизацию пользователей внешней системе авторизации пользователей (например, LDAP или Active Directory).

Домены также могут быть иерархическими, позволяя создавать корпоративные домены с многочисленными дочерними доменами для отделов или индивидуальных пользователей. Концепция домена может использоваться для агрегирования данных пользователей для оплаты или мониторинга использования облака.

Опции (capabilities) позволяют клиентам определить состав функций CDMI, поддерживаемых реализацией. В рамках данного стандарта требования должны восприниматься в контексте опций CDMI. Обязательные требования к функциональности, определяемой некоторой опцией CDMI, не должны пониматься, как требование всем реализациям поддерживать данную опцию, а как относящиеся к реализациям, включающим данную опцию.

Например, в 5.10, настоящий стандарт указывает, что «каждая облачная система хранения должна обеспечивать доступ к хранимым объектам по ID». Данное требование должно пониматься в контексте, что у функции доступа к ID объекта есть предусловие – наличие опции cdmi_object_access_by_ID.

5.8 Объектная модель для CDMI

Модель для CDMI показана на рисунке 4

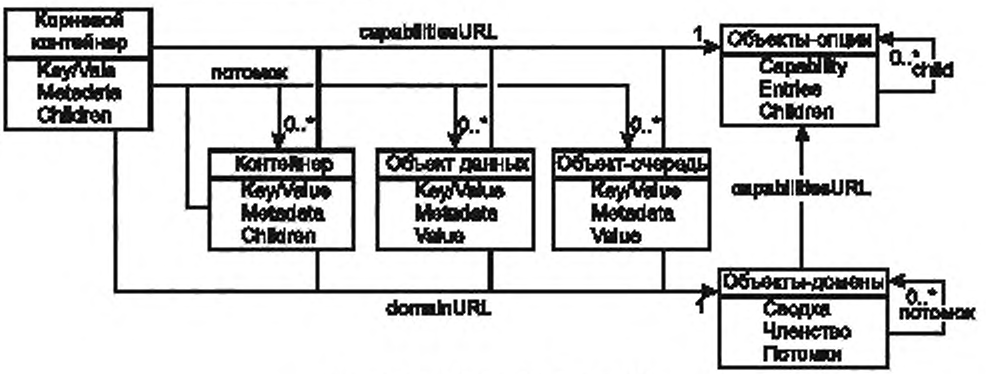

Рисунок 4 – Модель объекта CDMI

В таблице 3 перечислены пять типов ресурсов. Содержание каждой конкретной операции зависит от типа ресурса.

Таблица 3 – Типы ресурсов в модели

Тип ресурса	Описание	Ссылка
Объекты данных	Объекты данных служат для хранения значений и предоставляют функциональность аналогичную файлам в файловой системе.	См. раздел. 8
Объекты-контейнеры	Контейнер имеет неотрицательное количество дочерних объектов, но не хранит значений. Их функциональность аналогична папкам файловой системы.	См. раздел 9
Объекты-домены	Домены выражают административные группы пользователей для их аутентификации и учета использования ресурсов.	См. раздел 10
Объекты-очереди	Очереди хранят неотрицательное число значений, доступ к которым осуществляется по принципу "первый пришел – первый ушел".	См. раздел 11
Объекты опций	Объекты опций описывают функциональность, реализованную сервером CDMI и используются клиентом для обнаружения поддерживаемой функциональности	См. раздел 12

Для операций хранения данных клиенту интерфейса достаточно информации о контейнерах и объектах данных. Реализации путей к данным должны поддерживать хотя бы один уровень контейнеров (см. 5.5). С использованием объектной модели CDMI (см. рисунок 4) клиент может отослать запрос PUT через CDMI (см. 5.6) с URI нового контейнера и создать новый контейнер с определенным именем. Метаданные контейнера опциональны и выражаются набором пар имя–значение. После создания контейнера клиент может отправить запрос PUT для создания объекта данных внутри нового контейнера. Последующий запрос GET выбирает объект данных, включая поле значения.

Определены также объекты очереди (см. рисунок 4) со специальными свойствами для упорядоченного, в дисциплине FIFO, создания и удаления значений. Подробнее см. в разделе 11.

CDMI определяет два пространства имен, которые могут использоваться для доступа к хранимым объектам: плоское пространство идентификаторов объектов и иерархическое пространство имен-путей. Поддержка доступа к объекту по ID выражается глобальной опцией cdmi_object_access_by_ID, а поддержка иерархических путей определяется опцией контейнера cdmi_create_dataobject, находящейся в корневом или вложенном контейнере.

Объекты создаются по ID выполнением команды HTTP POST с определенным URI, обозначаемым как /cdmi_objectid/ (см. 9.8). После создания клиенты могут объект командой PUT с указанием ID объекта, присвоенный сервером CDMI, используя /cdmi_objectid/ URI (см. 8.6). Этот URI используется также для извлечения и удаления объектов по ID.

Объекты создаются по имени выполнением HTTP PUT с указанием URI пути (см. 8.2). После создания, объекты можно модифицировать, выполняя команды PUT с указанием пути к объекту, заданному клиентом (см. 8.6). То же самый URI используется для извлечения и удаления объектов.

CDMI определяет механизмы, позволяющие сопоставить путь в иерархическом пространстве объекту, имеющему лишь ID, а также позволяющие удалить путь, оставив только ID, у объекта, имеющего и путь, и ID. Это осуществляется с использованием модификатора перемещения для операций PUT или POST, как показано на рисунок 5.

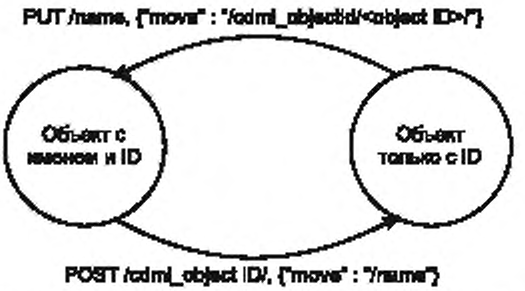

Рисунок 5 – Диаграмма переходов объекта: добавление имени и удаление имени

5.9 Метаданные CDMI

CDMI использует много видов метаданных, включая метаданные HTTP, метаданные системы данных, пользовательские метаданные и метаданные системы хранения.

Метаданные HTTP – это метаданные, связанные с использованием протокола HTTP (например, Content-Length, Content-Type и др.). Метаданные HTTP не связаны с данным международным стандартом, но их необходимо обсудить для объяснения, как CDMI использует стандарт HTTP.

Метаданные системы данных CDMI, пользовательские метаданные и метаданные системы хранения определятся в форме пар имя–значение. Метаданные системы данных, определенные производителем, должны начинаться с обращенного доменного имени производителя.

Метаданные системы данных – это метаданные, определяемые клиентом CDMI, они входят в состав объекта. Метаданные системы данных абстрактно определяют требования к данным, связанные со службами данных, предоставляемыми облачным хранилищем.

Пользовательские метаданные состоят из определенных клиентом строк, массивов и объектов в нотации JSON, сохраненных в поле метаданных. Пространство имен для пользовательских метаданных является самоуправляемым (например, использует обращенное имя домена), причем имена пользовательских метаданных не должны начинаться с префикса «cdmi_».

Метаданные системы хранения – это метаданные, сгенерированные службами хранения системы (например, время создания или размер) для предоставления необходимой информации клиенту CDMI.

Матрица создания и использования метаданных системы хранения приведена в табл 4.

Таблица 4 – Создание–использование метаданных системы хранения

Тип метаданных	Созданные пользователем	Созданные системой
Используемые пользователем	Пользовательские метаданные	Метаданные системы хранения
Используемые системой	Метаданные системы данных	N/A

5.10 ID объекта

Каждому объекту, хранимому в CDMI-совместимой системе, в момент создания ставится в соответствие глобально-уникальный идентификатор объекта (ID). ID объекта CDMI – это строка, на которую накладываются требования, обеспечивающие ее правильное создание и уникальность. Каждая реализация CDMI способна создавать уникальные идентификаторы, не конфликтуя с другими реализациями.

Каждая облачная система хранения должна предоставлять доступ к хранимым объектам по ID посредством добавления ID объекта к URI корневого контейнера. Если объект данных «MyDataObject.txt» имеет ID «00006FFD001001CCE3B2B4F602032653», следующая пара URI дает доступ к одному и тому же объекту данных:

http://cloud.example.com/root/MyDataObject.txt

http://cloud.example.com/root/cdmi_objectid/00006FFD001001CCE3B2B4F602032653

Если поддерживается работа с контейнерами, они также должны быть доступными по ID объекта. Если контейнер «MyContainer» имеет ID объекта «00006FFD0010AA33D8CEF9711E0835CA», следующие пары URI дают доступ к одному и тому же объекту данных:

http://cloud.example.com/MyContainer/
http://cloud.example.com/cdmi_objectid/00006FFD0010AA33D8CEF9711E0835CA/
http://cloud.example.com/MyContainer/MyDataObject.txt
http://cloud.example.com/cdmi_objectid/00006FFD0010AA33D8CEF9711E0835CA/MyDataObject.txt

5.11 Формат идентификатора объекта CDMI

Каждая реализация должна создавать ID объекта, однозначно идентифицирующий объект. ID объекта должен быть глобально-уникальным и должен отвечать формату, определенному на рисунке 6. Исходный формат ID объекта – строка байт переменной длины (максимум 40 байт). Приложение обращается с объектом ID как с непрозрачной байтовой строкой. Тем не менее, формат ID объекта определен так, что его целостность может быть проверена и независимые реализации могут независимо создавать уникальные ID Объектов.

0	1	2	3	4	5	6	7	8	9	10	...	38	39
Зарезервировано (ноль)	Номер организации			Зарезервировано (ноль)	Длина	CRC		Непрозрачные данные					

Рисунок 6 – Формат ID объекта

Поля, показанные на рисунке 6, определены следующим образом:

- Зарезервированные байты должны быть нулевыми.

Поле номера организации – это код, присвоенный организации, чья реализация создает ID объекта, в спецификации SNMP, в сетевом порядке байтов. См. RFC 2578 и http://www.iana.org/assignments/enterprise-numbers. Код 0 зарезервирован.

Байт со смещением 5 должен содержать длину ID в байтах.

Поле CRC должно содержать двухбайтовый (16-битный) CRC в сетевом порядке байтов. Это поле позволяет контролировать целостность ID объекта. Поле CRC должно заполняться выполнением алгоритма (см. [CRC]) по всем байтам ID объекта, определенным полем Длина, при обнуленном поле CRC. Вычисление CRC определяется набором параметров:

- Name : «CRC-16»,
- Width : 16,
- Poly : 0x8005,
- Init : 0x0000,
- RefIn : True,
- RefOut : True,
- XorOut : 0x0000, и
- Check : 0xBB3D.

Эта функция возвращает 16-битный CRC по полиному 0x8005, обращенным входом и обращенным результатом. Алгоритм вычисления CRC-16 определен в [CRC].

- Непрозрачные данные ID каждого объекта должны быть уникальными в пределах одного номера организации.

Исходный формат объекта ID – бинарный. При необходимости, например при включении в URI или строки JSON, текстовое представление ID объекта должно кодироваться на основе правил base 16, описанных в [RFC 4648] и не должно учитывать регистр.

5.12 Безопасность

В контексте CDMI безопасность относится к защитным мерам, применяемым при управлении и доступе к данным и хранилищу. В частности, система безопасности должна решать следующие задачи:

- предоставление механизма, обеспечивающего невозможность прочтения третьей стороной обмен данными между CDMI клиентом и сервером;
- предоставление механизма, позволяющего CDMI клиентам и серверам доказывать свою идентичность;
- предоставление механизма, позволяющего управлять разрешенными действиями CDMI клиента на CDMI сервере;

- предоставление механизма для записи действий, выполненных CDMI клиентом на CDMI сервере;
- предоставление механизмов защиты данных в фоновом режиме;
- предоставление механизмов контролируемого удаления данных;
- предоставление механизма для определения набора опций безопасности конкретной реализации.

Меры безопасности, включенные в CDMI, можно описать как:

- безопасная передача;
- аутентификация пользователей и сущностей;
- авторизация и контроль доступа;
- целостность данных;
- очистка данных и среды;
- удерживание данных;
- защита от вредоносных действий;
- шифрование данных в фоновом режиме;
- опции безопасности.

За исключением безопасности передачи и опций безопасности, которые обязательны для реализации, остальные меры безопасности существенно зависят от конкретной реализации.

Если безопасность является важной задачей, CDMI клиент должен начинать работу с серии поисков опций безопасности (см. подпункт 12.1.1) для определения точной природы возможных мер безопасности. Основываясь на значениях опций и оценке рисков, принимается решение о том, может ли использоваться данный сервер CDMI. Это особенно важно, если в облаке будет храниться ценные данные или данные ограниченного доступа, и требуется определенная защита (шифрование) или особая обработка данных (например, опция записи и просмотра всех выполняемых действий).

HTTP является обязательным механизмом передачи данных, а HTTP над TLS (то есть, HTTPS) является механизмом, используемым для безопасного взаимодействия между клиентами и сервером CDMI. Для обеспечения безопасности и совместимости, все реализации CDMI должны поддерживать протокол безопасности транспортного уровня (Transport Layer Security, TLS) как описано в приложении А, но его использование клиентами и сервером CDMI опционально.

5.13 Необходимая поддержка HTTP

5.13.1 Требования поддержки RFC 2616

Совместимая реализация CDMI должна также включать реализацию RFC2616 (см. [RFC 2616]) (т.е., HTTP 1.1). Перечисленные ниже примеры описывают некоторые части RFC 2616, которые должны поддерживаться, но этот список не является исчерпывающим.

5.13.2 Согласование типа содержимого

Для операций CDMI используются типы медиа CDMI объектов, определенные в [RFC 6208].

Клиент может опционально предоставлять заголовок HTTP Accept согласно 14.1 RFC 2616. Если клиент требует, чтобы в ответе содержался определенный тип медиа CDMI, соответствующий тип должен быть указан в заголовке Accept. Иначе, заголовок может содержать «*/*» или список типов, или быть опущен.

Если в сообщении-запросе присутствует тело, клиент должен включать заголовок Content-Type согласно пункту 14.17 RFC 2616. Если в таком случае клиент не предоставляет заголовок Content-Type содержимого или тип медиа в заголовке Content-Type не соответствует существующему типу ресурса, сервер должен вернуть код состояния HTTP 400 Bad Request.

Если в сообщении-ответе присутствует тело, сервер должен предоставить заголовок Content-Type.

Настоящий стандарт допускает дальнейшие согласования типа содержимого (например, в 9.3 отсутствие заголовка Content-Type несет особую информацию).

5.13.3 Поддержка диапазона

Сервер должен поддерживать заголовки HTTP Range и ответы с частичным содержимым (см. 14.16 RFC 2616).

5.13.4 Экранирование в URI

Ко всем строкам, использованным в URI, должно применяться экранирование зарезервированных символов знаком процента (%), определенное в RFC 3986. Это касается имен полей, предоставленных пользователем, имен метаданных, имен объектов, имен контейнеров и имен доменов, использованных в URI.

Имена и значения полей не должны экранироваться в телах сообщений запросов и ответов.

Пример – Клиент, получающий метаданные объекта «@user» из контейнера «@MyContainer», должен выполнить следующий запрос:

```
GET /%40MyContainer/?objectName;metadata:%40user HTTP/1.1
Host: cloud.example.com
Accept: application/cdmi-container
X-CDMI-Specification-Version: 1.0.2
В ответ должно быть получено:
HTTP/1.1 200 OK
Content-Type: application/cdmi-container
X-CDMI-Specification-Version: 1.0.2
{
        «objectName»: «@MyContainer»,
«metadata»: {
        «@user»: «test»
}
}
```

5.13.5 Использование URI

Формат и синтаксис URI определяются RFC 3986.

Каждый клиент CDMI должен поддерживать один или несколько корневых URI, чтобы каждый из них соответствовал корневому контейнеру сервера CDMI. Так как все URI контейнеров CDMI завершаются наклонной чертой, все корневые URI также завершаются наклонной чертой.

Все URI в данном международном являются относительными (relative) URI, заданными относительно корневых URI, если не сказано иное. Таким образом, в качестве алгоритма разрешения URI используется алгоритм из пункта 5.2 RFC 3986.

В таблице 5 приведено разрешение относительного URI по корневому URI.

Таблица 5 – Разрешение относительного URI по корневому URI

Корневой URI	+ Относительный URI	=> Разрешенный URI
http://cloud.example.com/	cdmi_object/testObject	http://cloud.example.com/cdmi_object/testObject
http://cloud.example.com/	/cdmi_object/testObject	http://cloud.example.com/cdmi_object/testObject
http://cloud.example.com/p1/	cdmi_object/testObject	http://cloud.example.com/p1/cdmi_object/testObject
http://cloud.example.com/p1/	/cdmi_object/testObject	http://cloud.example.com/cdmi_object/testObject
http://cloud.example.com/p1/p2/	cdmi_object/testObject	http://cloud.example.com/p1/p2/cdmi_object/ testObject
http://cloud.example.com/p1/p2/	/cdmi_object/testObject	http://cloud.example.com/cdmi_object/testObject

Настоящий стандарт не накладывает ограничений на корневые и относительные URI. Все примеры, приведенные в таблицу 5, допустимы, используя корневой URI http://cloud.example.com/ и возвращая ссылки на абсолютные пути, как показано во второй строке таблицу 5.

5.13.6 Зарезервированные символы

Имена объектов данных CDMI, контейнеров, очередей, доменов и объектов опций не могут содержать символы «/» и «?», так как они зарезервированы как разделители.

5.14 Представление времени

Если не указано иное, все значения даты/времени даются согласно расширенному представлению ИСО 8601:2004 (YYYY-MM-DDThh:mm:ss.ssssssZ). Должна быть указана полная точность, разделитель долей секунды «.» (точка), должен быть включен индикатор пояса UTC Z UTC, и все отметки времени должны быть в часовом поясе UTC. Представление YYYY-MM-DDT24:00:00.000000Z не следует использовать, вместо этого следует использовать представление YYYY-MM-DDT00:00:00.000000Z.

Если не указано иное, все интервалы даты/времени должны быть в формате начальная дата/конечная дата в соответствии со стандартом ISO 8601:2004 (YYYY-MM-DDThh:mm:ss.ssssssZ/YYYY-MM-DDThh:mm:ss.ssssssZ). Конечная дата должна быть не раньше начальной даты. Должна быть указана полная точность, разделитель долей секунды «.» (точка), должен быть включен индикатор пояса UTC Z UTC, и все отметки времени должны быть в часовом поясе UTC. Представление YYYY-MM-DDT24:00:00.000000Z не следует использовать, вместо этого следует использовать представление YYYY-MM-DDT00:00:00.000000Z.

5.15 Обратная совместимость

5.15.1 Кодировка значений для передачи

Стандарт CDMI версии 1.0.1 ввел концепцию кодировки значений для передачи, чтобы обеспечить опции хранения и получения произвольных бинарных данных через операции с типом содержимого CDMI. Объекты данных, созданные клиентами CDMI 1.0 посредством операций с типом содержимого CDMI, должны использовать кодировку «utf-8», а объекты, созданные другими операциями, должны использовать кодировку «base64».

Значения полей объектов данных в кодировке base64 не должны быть доступны клиентам CDMI 1.0 через операции с типом содержимого CDMI. Попытка чтения значения таких объектов должна возвращать таким клиентам пустой результат («»). Клиенты CDMI 1.0 могут выявить такую ситуацию: в этом случае метаданные cdmi_size не равно 0, а значение поля пустое.

5.15.2 Опции экспорта контейнера

CDMI версии 1.0.2 упорядочивает имена опций, которые использует клиентом для определения, может ли контейнер экспортироваться через различные протоколы. Следующие имена опций экспорта контейнера более недействительны:

- cdmi_cifs_export,
- cdmi_nfs_export,
- cdmi_iscsi_export,
- cdmi_occi_export.

6 Обычные операции

6.1 Обзор

Все примеры, приведенные в настоящем стандарте, не являются нормативными.

Данный раздел включает примеры следующих типизованных операций CDMI:

- определение опций провайдера облачного хранилища (см. 6.2),
- создание нового контейнера (см. 6.3),
- создание нового объекта данных (см. 6.4),
- получение списка содержимого контейнера (см. 6.5),
- чтение содержимого объекта данных (см. 6.6),
- чтение только значения объекта данных (см. 6.7),
- удаление объекта данных (см. 6.8).

6.2 Определение опций провайдера облачного хранения

Пример – Выполнение GET к URI, по которому сервер публикует перечень опций:

```
GET /cdmi_capabilities/ HTTP/1.1
Host: cloud.example.com
Accept: application/cdmi-capability
X-CDMI-Specification-Version: 1.0.2
Будет получен следующий ответ.
HTTP/1.1 200 OK
Content-Type: application/cdmi-capability
X-CDMI-Specification-Version: 1.0.2
{
«objectType» : «application/cdmi-capability»,
«objectID» : «00007E7F0010CEC234AD9E3EBFE9531D»,
«objectName» : «cdmi_capabilities/»,
«parentURI» : «/»,
«parentID» : «00007E7F0010DCECC805FB6D195DDBCB»,
«capabilities» : {
«cdmi_domains» : «true»,
«cdmi_export_nfs» : «true»,
«cdmi_export_webdav» : «true»,
«cdmi_export_iscsi» : «true»,
«cdmi_queues» : «true»,
```

«cdmi_notification» : «true»,
«cdmi_query» : «true»,
«cdmi_metadata_maxsize» : «4096»,
«cdmi_metadata_maxitems» : «1024»,
«cdmi_size» : «true»,
«cdmi_list_children» : «true»,
«cdmi_read_metadata» : «true»,
«cdmi_modify_metadata» : «true»,
«cdmi_create_container» : «true»,
«cdmi_delete_container» : «true»
},
«childrenrange» : «0-3»,
«children» : [
«domain/»,
«container/»,
«dataobject/»,
«queue/»
]
}

6.3 Создание нового контейнера

Пример – Выполнение PUT с URI нового контейнера:
PUT /MyContainer/HTTP/1.1
Host: cloud.example.com
Accept: application/cdmi-container
Content-Type: application/cdmi-container
X-CDMI-Specification-Version: 1.0.2
{
«metadata» : {
}
}
Будет получен следующий ответ.
HTTP/1.1 201 Created
Content-Type: application/cdmi-container
X-CDMI-Specification-Version: 1.0.2
{
«objectType» : «application/cdmi-container»,
«objectID» : «00007E7F00102E230ED82694DAA975D2»,
«objectName» : «MyContainer/»,
«parentURI» : «/»,
«parentID» : «00007E7F0010128E42D87EE34F5A6560»,
«domainURI» : «/cdmi_domains/MyDomain/»,
«capabilitiesURI» : «/cdmi_capabilities/container/»,
«completionStatus» : «Complete»,
«metadata» : {
«cdmi_size» : «0»
},
«childrenrange» : «»,
«children» : [

]
}

6.4 Создание объекта данных в контейнере

Пример – Выполнение PUT с URI нового объекта данных:
PUT /MyContainer/MyDataObject.txt HTTP/1.1
Host: cloud.example.com
Accept: application/cdmi-object
Content-Type: application/cdmi-object
X-CDMI-Specification-Version: 1.0.2
{

«mimetype» : «text/plain»,
«metadata» : {
},
«value» : «Hello CDMI World!»
}
Будет получен следующий ответ.
HTTP/1.1 201 Created
Content-Type: application/cdmi-object
X-CDMI-Specification-Version: 1.0.2
{
«objectType» : «application/cdmi-object»,
«objectID» : «00007E7F0010BD1CB8FF1823CF05BEE4»,
«objectName» : «MyDataObject.txt»,
«parentURI» : «/MyContainer/»,
«parentID» : «00007E7F00102E230ED82694DAA975D2»,
«domainURI» : «/cdmi_domains/MyDomain/»,
«capabilitiesURI» : «/cdmi_capabilities/dataobject/»,
«completionStatus» : «Complete»,
«mimetype» : «text/plain»,
«metadata» : {
«cdmi_size» : «17»
}
}

6.5 Список содержимого контейнера

Пример – Выполнение GET с URI контейнера:
GET /MyContainer/ HTTP/1.1
Host: cloud.example.com
*Accept: */**
X-CDMI-Specification-Version: 1.0.2
Будет получен следующий ответ.
HTTP/1.1 200 OK
Content-Type: application/cdmi-container
X-CDMI-Specification-Version: 1.0.2
{
«objectType» : «application/cdmi-container»,
«objectID» : «00007E7F00102E230ED82694DAA975D2»,
«objectName» : «MyContainer/»,
«parentURI» : «/»,
«parentID» : «00007E7F0010128E42D87EE34F5A6560»,
«domainURI» : «/cdmi_domains/MyDomain/»,
«capabilitiesURI» : «/cdmi_capabilities/container/»,
«completionStatus» : «Complete»,
«metadata» : {
«cdmi_size» : «83»
},
«childrenrange» : «0-0»,
«children» : [
«MyDataObject.txt»
]
}

6.6 Чтение содержимого объекта данных

Пример – GET по URI объекта данных:
GET /MyContainer/MyDataObject.txt HTTP/1.1
Host: cloud.example.com
Accept: application/cdmi-object
X-CDMI-Specification-Version: 1.0.2
Будет получен следующий ответ.
HTTP/1.1 200 OK
Content-Type: application/cdmi-object

X-CDMI-Specification-Version: 1.0.2
{
«objectType»: «application/cdmi-object»,
«objectID»: «00007E7F0010BD1CB8FF1823CF05BEE4»,
«objectName»: «MyDataObject.txt»,
«parentURI»: «/MyContainer/»,
«parentID» : «00007E7F00102E230ED82694DAA975D2»,
«domainURI»: «/cdmi_domains/MyDomain/»,
«capabilitiesURI»: «/cdmi_capabilities/dataobject/»,
«completionStatus»: «Complete»,
«mimetype»: «text/plain»,
«metadata»: {
«cdmi_size»: «17»
},
«valuetransferencoding»: «utf-8»,
«valuerange»: «0-16»,
«value»: «Hello CDMI World!»
}

6.7 Чтение только значения объекта данных

Пример – Выполнение GET с URI объекта данных:
GET /MyContainer/MyDataObject.txt HTTP/1.1
Host: cloud.example.com
Будет получен следующий ответ.
HTTP/1.1 200 OK
Content-Type: text/plain
Hello CDMI World!

6.8 Удаление объекта данных

Пример – Выполнение DELETE с URI объекта данных:
DELETE /MyContainer/MyDataObject.txt HTTP/1.1
Host: cloud.example.com
X-CDMI-Specification-Version: 1.0.2
Будет получен следующий ответ.
HTTP/1.1 204 No Content

7 Стандарт интерфейса

7.1 Коды состояния HTTP

Коды состояния HTTP (см. таблицу 6) используются для передачи результата операций REST и выполнения базовой семантики HTTP с минимальной перегрузкой. Другие коды состояния HTTP не являются частью настоящего стандарта и сохраняют оригинальную семантику HTTP 1.1.

Таблица 6 – Коды состояния HTTP

Код состояния	Имя HTTP	Описание
200	OK	Запрос завершился успешно.
201	Created	Ресурс успешно создан.
202	Accepted	Длительная операция принята на выполнение.
204	No Content	Операция завершилась успешно, никаких данных не возвращено.
302	Found	Ресурс является ссылкой на другой ресурс.
400	Bad Request	Содержание запроса неверное или отсутствует.
401	Unauthorized	Неверные данные аутентификации/авторизации.
403	Forbidden	У клиента недостаточно прав для выполнения запроса.
404	Not Found	Ресурс не найден по указанному URI.
406	Not Acceptable	На данном URI не может быть создано содержание, соответствующее запросу.
409	Conflict	Операция конфликтует блокировкой, установленной протоколом доступа, отличным от CDMI™, или может вызвать ошибку изменения состояния сервера.

7.2 Ссылки на объект

Ссылки на объект – это URI в пространстве имен облачного хранилища, переадресующие на другой URI в том же или другом пространстве имен облачного хранилища. Ссылки подобны символическим ссылкам файловой системы. Облако не гарантирует, что URI, на который дана ссылка, будет корректен после создания ссылки.

Ссылки отображаются как дочерние элементы контейнера и отличаются от других объектов завершающим символом «?» в имени. Выполнение операции (за исключением создания или удаления) по URI ссылки приведет к переадресации HTTP 302 Found; поле заголовка Location будет содержать URI назначения, указанный в момент создания ссылки. URI назначения ссылки не должен меняться после создания ссылки.

Для продолжения операции после получении переадресации 302 Found клиент CDMI должен повторить запрос с использованием URI, содержащегося в заголовке Location.

Операция удаления по URI ссылки должна удалить ссылку. Ссылки не могут обновляться. Для обновления адреса назначения клиент должен вначале удалить имеющуюся ссылку, а затем создать новую с желаемым адресом назначения переадресации.

Пример – Применение GET к URI, где URI – ссылка:
GET /MyContainer/MyDataObject.txt HTTP/1.1
Host: cloud.example.com
Accept: application/cdmi-object
X-CDMI-Specification-Version: 1.0.2
Будет получен следующий ответ.
HTTP/1.1 302 Found
Location: http://cloud.example.com/MyContainer/MyOtherDataObject.txt

Ссылки на ID объекта должны всегда переадресовывать на URI, завершающийся тем же ID, что и URI запроса.

Пример – Применение GET к URI объекта по ID, где URI – ссылка:
GET /cdmi_objectid/00006FFD0010AA33D8CEF9711E0835CA HTTP/1.1
Host: cloud.example.com
Accept: application/cdmi-object
X-CDMI-Specification-Version: 1.0.2
Будет получен следующий ответ.
HTTP/1.1 302 Found
Location: http://archive.example.com/cdmi_objectid/00006FFD0010AA33D8CEF9711E0835CA

8 Операции с ресурсами объекта данных

8.1 Обзор

Объекты данных – основной компонент хранения в системе CDMI™, аналогичный файлам в файловой системе. Каждый объект данных состоит из набора определенных полей, включающих:

- единственное значение (value)
- опциональные метаданные, генерируемые системой облачного хранения и/или пользователем системы.

Объекты данных в CDMI могут адресоваться двумя способами:

- по имени (например, http://cloud.example.com/dataobject);
- по ID объекта (например, http://cloud.example.com/cdmi_objectid/0000706D0010B84FAD185C425D8B537E).

Каждый объект данных включает единственный, глобально-уникальный идентификатор объекта (ID), который остается неизменным на протяжении времени жизни объекта. Каждый объект данных имеет один или несколько адресов URI, позволяющих обращаться к объекту.

Каждый объект данных имеет родительский объект, от которого он наследует метаданные, которые не заданы явным образом для рассматриваемого объекта. Так, объект «budget.xls», хранящийся по следующему URI, наследует метаданные системы данных своего родительского контейнера «finance»:

http://cloud.example.com/finance/budget.xls

Для доступа к отдельным полям объекта данных необходимо указать имя поля после символа «?», добавляемого к URI объекта данных. Например, следующий URI возвращает значение поля value в сообщении-ответе:

http://cloud.example.com/dataobject?value

Кодировка данных, передаваемых из поля value объекта данных, определяется значением поля valuetransferencoding этого объекта.

Если кодировка значения установлена в «utf-8», данные в поле value объекта должны быть корректной строкой UTF-8, и должны передаваться в поле value как строка UTF-8.

Если кодировка значения установлена в «base64», данные в поле value объекта могут быть произвольной бинарной последовательностью и должны передаваться как строка в кодировке base 64.

Отдельный диапазон значения объекта может быть получен указанием диапазона байтов после имени поля. Например, следующий URI возвращает первые 1000 байт поля value:

http://cloud.example.com/dataobject?value:0-999

Так как диапазон байтов строки UTF-8 не всегда является корректной строкой UTF-8, ответ на запрос с обозначенными границами диапазона передается как строка в кодировке base 64. Аналогично, при изменении диапазона байтов значения поля объекта, содержимое поля должно передаваться как строка в кодировке base 64.

Диапазон байтов включает границы диапазона, как указано в п. 14.35.1 RFC 2616.

Допускается указание списка уникальных полей, разделенных символом «;», что позволяет обратиться к нескольким полям в одном запросе. Например, следующий URI возвращает поля value и metadata в сообщении-ответе:

http://cloud.example.com/dataobject?value;metadata

Если клиент создает поля, не определенные в настоящем стандарте, или десериализует объект, содержащий поля, не описанные в данном международном стандарте, эти поля должны храниться как часть объекта, но не должны интерпретироваться.

8.1.1 Метаданные объекта данных

Метаданные объекта данных могу также включать произвольные метаданные, определенные пользователем, и метаданные системы данных, см. гл. 16. Метаданные следует хранить как корректную строку UTF-8. Бинарные данные, сохраненные в пользовательских метаданных, должны быть вначале перекодированы таким образом, чтобы их можно было включить в корректную строку UTF-8; рекомендуется использовать кодировку base 64.

8.1.2 Согласованность объекта данных

Запись в объект данных является атомарной операцией.

Если клиент читает данные из объекта одновременно с операцией записи данных в тот же объект, чтение должно возвращать либо старое значение, либо новое, но не их смесь.

Если запись завершается с ошибкой, содержимое объекта должно остаться таким, как если бы запись не производилась (другими словами, запись является атомарной операцией по отношению к ошибкам).

Метка времени, возвращенная в ответ на запись, показывает, была ли запись последней операцией (то есть, запись, результат которой будет возвращен при последующем чтении, пока не произойдет новая запись). Если метка времени, возвращаемая при записи, показывает более позднее время, чем метка времени для другой записи, тогда запись с более поздней меткой становится той, чьи данные находятся в объекте в настоящее время, если записи происходят одновременно.

Записи в диапазон могут привести к промежутку в значении объекта, не содержащему записанных данных. Чтение из промежутка с отсутствующими данными должно выдавать нуль в каждом прочитанном байте.

Реализации настоящего стандарта должны обеспечивать атомарность операций, описанных в данном разделе, если объекты данных обрабатываются через CDMI. Атомарность свойств объектов, обрабатываемых по другим протоколам, выходит за рамки настоящего стандарта.

8.1.3 Представления объекта данных

В данном разделе используется нотация JSON. И клиенты, и серверы должны поддерживать представление UTF-8 JSON. В теле сообщения-запроса и соощения-ответа поля объекта JSON могут обозначаться или возвращаться в любом порядке, за исключением (в случае их наличия) полей valuerange и value объекта данных, которые должны появляться последними и в указанном порядке.

8.2 Создание объекта данных с использованием типа содержимого CDMI

8.2.1 Обзор

Для создания нового объекта данных следует выполнить запрос:

PUT <root URI>/<ContainerName>/<DataObjectName>

Создание нового объекта по ID, см. 9.9.

где:

- <root URI> путь к облаку CDMI.
- <ContainerName> неотрицательное число промежуточных уже существующих контейнеров, с наклонной чертой «/», разделяющей каждую пару контейнеров.
- <DataObjectName> имя создаваемого объекта данных.

После создания к объекту можно обращаться как <root URI>/cdmi_objectid/<objectID>.

8.2.2 Отложенное завершение создания

В ответ на операцию создания объекта сервер может вернуть код 202 Accepted, указывающий на то, что объект находится в стадии создания. Этот ответ полезен в случае длительных операций (например, копирование большого объекта данных из URI источника). Этот ответ имеет следующие последствия:

- сервер вернет заголовок Location с URI создаваемого объекта вместе с кодом HTTP 202 Accepted;
- код 202 Accepted от сервера означает, что были пройдены следующие проверки:
- пользователь авторизован создавать объект;
- пользователь авторизован считывать любые источники, которые должны быть скопированы, перемещены, сериализованы или десериализованы
- доступно достаточно места для создания объекта, или, по крайней мере, достаточно места, чтобы создать URI и сообщение об ошибке.
- Клиент, возможно, не сможет немедленно обратиться к создаваемому объекту, например, из-за задержек вследствие использования в данной реализации целостности в конечном итоге.

Клиент выполняет операции GET по указанному URI для отслеживания хода выполнения операции. В ответ сервер возвращает два поля в теле сообщения-ответа, указывающие на состояние процесса:

- обязательное текстовое поле completionStatus содержит «Processing», «Complete», либо сообщение об ошибке, начинающееся с «Error»;
- опциональное поле percentComplete, содержащее процент выполнения операции (от 0 до 100).

GET не должен возвращать никакого значения в объект, если его статус не completionStatus = «Complete». Если конечный результат операции создания – ошибка, то создается URI с полем completionStatus, содержащим сообщение об ошибке. Удаление URI после обнаружение ошибки – обязанность клиента.

8.2.3 Опции

Следующие опции перечисляют операции, которые могут выполняться при создании нового объекта данных:

- поддержка функции создания нового объекта данных обозначена наличием в родительском контейнере опции cdmi_create_dataobject;
- если создаваемый объект – ссылка в родительском контейнере, поддержка этой функции обозначена наличием в родительском контейнере опции cdmi_create_reference;
- если новый объект должен быть копией существующего объекта, поддержка этой функции обозначена наличием в родительском контейнере опции cdmi_copy_dataobject;
- если создаваемый объект – результат перемещения существующего объекта, поддержка этой функции обозначена наличием в родительском контейнере опции cdmi_move_dataobject;
- если создаваемый объект – результат операции десериализации исходного объекта, поддержка этой функции обозначена наличием в родительском контейнере опции cdmi_deserialize_dataobject;
- если новый объект – результат операции сериализации, поддержка этой функции обозначена присутствием в родительском контейнере опций cdmi_serialize_dataobject, cdmi_serialize_container, cdmi_serialize_domain или cdmi_serialize_queue.

8.2.4 Заголовки запроса

Заголовки запроса HTTP для создания объекта CDMI с типом содержимого CDMI приведены в таблице 7.

Таблица 7 – Заголовки запроса для создания объекта данных CDMI типом содержимого CDMI

Заголовок	Тип	Описание	Требование
Accept	Строка заголовка	"application/cdmi-object" или допустимое значение в соответствии с 5.13.2	Опционально
Content-Type	Строка заголовка	"application/cdmi-object"	Обязательно
X-CDMI-SpecificationVersion	Строка заголовка	Разделенный запятыми список версий, поддерживаемых клиентом, например, "1.0.2, 1.5, 2.0"	Обязательно
X-CDMI-Partial	Строка заголовка	"true" указывает на то, что новый объект является частью набора операций записи, и что его значение еще заполнено не полностью. Если присутствует X-CDMI-Partial, поле completionStatus в теле ответа будет выставлено в "Processing".	Опционально

8.2.5 Тело сообщения-запроса

Поля запроса на создание объекта с использованием типа содержимого CDMI приведено в таблице 8.

Таблица 8 – Тело сообщения-запроса – создание объекта данных с использованием типа содержимого CDMI

Имя поля	Тип	Описание	Требование
mimetype	Строка JSON	Тип данных MIME, содержащихся в поле value объекта данных Данное поле может добавляться при создании по значению, также при сериализации, десериализации, копировании или перемещении объекта данных. Данное поле должно храниться как часть объекта. Если данное поле не указано, ему должно быть присвоено значение "text/ plain". Это поле не должно добавляться при создании ссылки. Перед сохранением значение типа MIME должно быть преобразовано в нижний регистр.	Опционально
metadata	Объект JSON	Метаданные объекта данных Если данное поле включено при десериализации, сериализации, копировании или перемещении объекта данных, его значение заменяет метаданные из URI источника. Если данное поле не включается при десериализации, сериализации, копировании или перемещении объекта данных, должны использоваться метаданные URI источника. Если данное поле включается при создании нового объекта данных по передаваемому значению, значение данного поля должно быть использовано как метаданные. Если данное поле не включается при создании нового объекта данных по значению, данному полю должен быть присвоен пустой объект JSON (т.е. "{}"). Данное поле не должно включаться при создании ссылки.	Опционально
domainURI	Строка JSON	URI домена-владельца Если домен-владелец не совпадает с родительским доменом, пользователь должен иметь права cross_domain privilege (см. cdmi_member_privileges в таблице 64). IЕсли не указано иное, должен использоваться домен родительского контейнера.	Опционально
deserialize	Строка JSON	URI сериализованного объекта данных CDMI, который должен быть десериализован в создаваемый объект данных	Опционально[a]
serialize	Строка JSON	URI объекта CDMI, который должен быть сериализован в создаваемый объект данных	Опционально[a]
copy	Строка JSON	URI объекта данных или очереди CDMI, которые должны быть скопированы в создаваемый объект данных	Опционально[a]

Окончание таблицы 8

Имя поля	Тип	Описание	Требование
move	Строка JSON	URI существующего локального или удаленного объекта данных CDMI (URI источника), который должен быть перенесен в URI, указанного в команде PUT. Содержимое объекта, включая ID, должно при перемещении сохраняться, а объект по URI источника должен быть удален после успешного создания нового объекта. Если недостаточно прав для чтения объекта данных по URI источника, записи объекта по URI назначения или удаления объекта данных по URI источника, или если любая из этих операций не завершена успешно, операция перемещения должна вернуть состояние ошибки 400 Bad Request, а исходный и новый объект должны остаться неизменными.	Опционально[a]
reference	Строка JSON	URI объекта данных CDMI, на который должна быть создана ссылка. Если при создании ссылки предоставляются какие-либо другие поля, сервер должен вернуть сообщение об ошибке HTTP 400 Bad Request.	Опционально[a]
deserialize-value	Строка JSON	Объект данных, сериализованный в соответствии с гл. 15 и кодированный с использованием правил base 64 (см. RFC 4648).	Опционально[a]
valuetransfer-encoding	Массив JSON строк JSON	Кодировка, использованная значения объекта данных. Определены два значения кодировки. "utf-8" указывает на то, что объект данных содержит корректную строку UTF-8, и должен передаваться как строка UTF-8 в поле value. "base64" указывает на то, что объект данных содержит произвольную бинарную последовательность и должен передаваться в поле value как строка в кодировке base 64. Задание содержимого поля value объекта данных иное, чем корректная строка в кодировке base 64, должно возвращать клиенту ошибку 400 Bad Request. Данное поле должно включаться лишь при создании объекта по значению. Если иное не указано клиентом, сервер должен установить значение поля valuetransferencoding равным "utf-8". Данное поле должно храниться как часть объекта.	Опционально
value	Строка JSON	Значение объекта данных Если данное поле не включено, ему должна быть присвоена пустая строка (т.е., ""). Если поле valuetransferencoding указывает на кодировку UTF-8, значение должно быть строкой UTF-8, сформированной по правилам JSON (RFC 4627). Если поле valuetransferencoding указывает на кодировку base 64, значение должно быть вначале кодировано в base 64 (RFC 4648).	Опционально[a]
[a] В каждой отдельной операции должно указываться лишь одно из этих полей. За исключением value, эти поля не должны храниться. Если имеется более одного такого поля, сервер должен вернуть сообщение об ошибке 400 Bad Request.			

8.2.6 Заголовки ответа

Заголовки HTTP ответов на создание объекта с использованием типа содержимого CDMI указаны в таблице 9.

Таблица 9 – Заголовки ответа – создание объекта данных с использованием типа содержимого CDMI

Заголовок	Тип	Описание	Требование
Content-Type	Строка заголовка	"application/cdmi-object"	Обязательно
X-CDMI-SpecificationVersion	Строка заголовка	Сервер должен возвращать наибольший номер версии, поддерживаемой и клиентом, и сервером, например "1.0.2". Если сервер не поддерживает ни одну из версий, поддерживаемых клиентом, сервер должен вернуть код состояния 400 Bad Request.	Обязательно

8.2.7 Тело сообщения-ответа

Поля тела сообщения-ответа на создание объекта данных с использованием типа содержимого CDMI приведены в таблице 10.

Таблица 10 – Тело сообщения-ответа – создание объекта данных с использованием типа содержимого CDMI

Имя поля	Тип	Описание	Требование
objectType	JSON String	"application/cdmi-object"	Обязательно
objectID	JSON String	ID объекта	Обязательно
objectName	JSON String	Имя объекта	Обязательно
parentURI	JSON String	URI родительского объекта Присоединение objectName к parentURI должно всегда давать корректный URI объекта.	Обязательно
parentID	JSON String	ID родительского контейнера	Обязательно
domainURI	JSON String	URI домена-владельца	Обязательно
capabilitiesURI	JSON String	URI опций для объекта	Обязательно
completionStatus	JSON String	Строка, показывающая статус создания объекта данных. Принимает одно из следующих значений "Processing" указывает на то, что объект находится в процессе создания. "Completed" указывает на успешное создание объекта данных. Строка, начинающаяся с "Error" указывает на то, что при создании объекта возникла ошибка.	Обязательно
percentComplete	JSON String	Если значение completionStatus равно "Processing", данное поле, при наличии, должно показывать процент выполнения операции создания объекта числовым значением от 0 до 100. Если значение completionStatus равно "Complete", данное поле, при наличии, должно иметь значение "100". Если значение completionStatus начинается с "Error", данное поле, при наличии, может содержать любое целое число от 0 до 100.	Опционально
mimetype	JSON String	Тип MIME значения объекта данных	Обязательно
metadata	JSON Object	Метаданные объекта данных. Данное поле включает любые пользовательские метаданные или метаданные системы данных, указанные в соответствующем поле тела запроса на создание, наряду с метаданными системы хранения, созданными облачным хранилищем. См. детальное описание метаданных в разделе 16.	Обязательно

8.2.8 Статус ответа

Коды состояния HTTP, возникающего при создании объекта данных с использованием типа содержимого CDMI, перечислены в таблице 11.

Таблица 11 – Коды состояния HTTP – создание объекта данных с использованием типа содержимого CDMI

Состояние HTTP	Описание
201 Created	Новый объект данных был создан
202 Accepted	Объект данных в процессе создания. Клиент CDMI должен отслеживать поля completionStatus и percentComplete для определения состояния операции
400 Bad Request	Запрос содержит неверные параметры или имена полей
401 Unauthorized	Неверные данные аутентификации/авторизации
403 Forbidden	Клиент не обладает правами для выполнения данного запроса
404 Not Found	Ресурс не найден по указанному URI
409 Conflict	Операция конфликтует с блокировкой не-CDMI протокола доступа или может вызвать ошибку передачи на сервер

8.2.9 Примеры

Примеры

1 Применение PUT к URI контейнера: имя объекта данных и текстовое содержимое:

```
PUT /MyContainer/MyDataObject.txt HTTP/1.1
Host: cloud.example.com
Accept: application/cdmi-object
Content-Type: application/cdmi-object
X-CDMI-Specification-Version: 1.0.2
{
«mimetype» : «text/plain»,
«metadata» : {
},
«value» : «This is the Value of this Data Object»
}
```

Будет получен следующий ответ.

```
HTTP/1.1 201 Created
Content-Type: application/cdmi-object
X-CDMI-Specification-Version: 1.0.2
{
«objectType» : «application/cdmi-object»,
«objectID» : «0000706D0010B84FAD185C425D8B537E»,
«objectName» : «MyDataObject.txt»,
«parentURI» : «/MyContainer/»,
«parentID» : «00007E7F00102E230ED82694DAA975D2»,
«domainURI» : «/cdmi_domains/MyDomain/»,
«capabilitiesURI» : «/cdmi_capabilities/dataobject/»,
«completionStatus» : «Complete»,
«mimetype» : «text/plain»,
«metadata» : {
        «cdmi_size» : «37»
}
}
```

2 Применение PUT к URI контейнера: имя объекта данных и бинарное содержимое:

```
PUT /MyContainer/MyDataObject.txt HTTP/1.1
Host: cloud.example.com
Accept: application/cdmi-object
Content-Type: application/cdmi-object
X-CDMI-Specification-Version: 1.0.2
{
«mimetype» : «text/plain»,
«metadata» : { },
«valuetransferencoding» : «base64»,
«value» : «VGhpcyBpcyB0aGUgVmFsdWUgb2YgdGhpcyBEYXRhIE9iamVjdA==»
}
```

Будет получен следующий ответ.

```
HTTP/1.1 201 Created
Content-Type: application/cdmi-object
X-CDMI-Specification-Version: 1.0.2
{
«objectType»: «application/cdmi-object»,
«objectID»: «0000706D0010374085EF1A5C7018D774»,
«objectName»: «MyDataObject.txt»,
«parentURI»: «/MyContainer/»,
«parentID» : «00007E7F00102E230ED82694DAA975D2»,
«domainURI»: «/cdmi_domains/MyDomain/»,
«capabilitiesURI»: «/cdmi_capabilities/dataobject/»,
«completionStatus»: «Complete»,
«mimetype»: «text/plain»,
«metadata»: {
        «cdmi_size»: «37»
}
}
```

8.3 Создание объекта данных с использованием типа содержимого, отличного от CDMI

8.3.1 Обзор

Для создания объекта данных следует выполнить следующий запрос:

PUT <root URI>/<ContainerName>/<DataObjectName> 2

где:

- <root URI> путь к облаку CDMI;
- <ContainerName> неотрицательное число уже существующих промежуточных контейнеров, разделенных наклонной чертой (т.е., «/»);
- <DataObjectName> имя создаваемого объекта данных.

После создания к объекту данных можно обращаться как <root URI>/cdmi_objectid/<objectID>.

8.3.2 Опции

Следующие опции описывают поддерживаемые операции, которые можно выполнять при создании нового объекта данных:

- поддержка возможности создания нового объекта данных обозначается присутствием опции cdmi_create_dataobject в родительском контейнере.

8.3.3 Заголовки запроса

Заголовки HTTP запроса на создание объекта данных CDMI с использованием типа содержимого, отличного от CDMI, приведены в таблице 12.

Таблица 12 – Заголовки запроса – создание объекта данных CDMI с использованием типа содержимого, отличного от CDMI

Заголовок	Тип	Описание	Требование
Content-Type	Строка заголовка	Тип содержимого данных, которые должны храниться в объекте данных. Указанное здесь значение должно использоваться как поле mimetype объекта данных CDMI. Если тип содержимого включает параметр charset (RFC 2046), равный "utf-8" (например, ";charset=utf-8"), поле valuetransferencoding объекта данных CDMI должно быть установлено в "utf-8". В ином случае, поле valuetransferencoding объекта данных CDMI должно быть установлено в "base64".	Обязательно
X-CDMI-Partial	Строка заголовка	"true" указывает на то, что новый объект является частью набора операций записи, и что его значение еще заполнено не полностью. Поле completionStatus будет установлено в "Processing".	Опционально

8.3.4 Тело сообщения-запроса

Тело сообщения-запроса содержит данные, которые должны храниться как значение объекта данных.

8.3.5 Заголовки ответа

Заголовки ответа не определены.

8.3.6 Тело сообщения-ответа

Поля сообщения-ответа не определены.

8.3.7 Статус ответа

Коды состояний HTTP, возникающих при создании объекта данных с использованием содержимого, отличного от CDMI, описаны в таблице 13.

Таблица 13 – Коды состояния HTTP – создание объекта данных CDMI с использованием типа содержимого, отличного от CDMI

Статус HTTP	Описание
201 Created	Новый объект данных был создан.
400 Bad Request	Запрос содержит неверные параметры или имена полей.
401 Unauthorized	Неверные данные аутентификации/авторизации.
403 Forbidden	Клиент не обладает правами для выполнения данного запроса.

Окончание таблицы 13

Статус HTTP	Описание
404 Not Found	Ресурс не найден по указанному URI.
409 Conflict	Операция конфликтует с блокировкой не-CDMI протокола доступа или может вызвать ошибку передачи на сервер.

8.3.8 Пример

Пример – Применение PUT к URI контейнера: имя объекта данных и содержимое:
PUT /MyContainer/MyDataObject.txt HTTP/1.1
Host: cloud.example.com
Content-Type: text/plain;charset=utf-8
Content-Length: 37
This is the Value of this Data Object
Будет получен следующий ответ.
HTTP/1.1 201 Created

8.4 Чтение объекта данных с использованием типа содержимого CDMI

8.4.1 Обзор

Для чтения всех полей существующего объекта данных необходимо выполнить следующий запрос:

GET <root URI>/<ContainerName>/<DataObjectName>

Для чтения одного или нескольких полей существующего объекта данных необходимо выполнить один из следующих запросов:

GET <root URI>/<ContainerName>/<DataObjectName>?<fieldname>;<fieldname>;...

GET <root URI>/<ContainerName>/<DataObjectName>?value:<range>;...

GET <root URI>/<ContainerName>/<DataObjectName>?metadata:<prefix>;...

где:

- <root URI> путь к облаку CDMI.
- <ContainerName> неотрицательное число промежуточных контейнеров.
- <DataObjectName> имя объекта данных, из которого необходимо произвести чтение.
- <fieldname> имя поля.
- <range> диапазон байтов значения объекта данных, которые необходимо вернуть в поле value.
- <prefix> шаблон префикса: возвращаются все метаданные, начинающиеся с данного префикса.

К объекту можно также обратиться как <root URI>/cdmi_objectid/<objectID>.

8.4.2 Опции

Следующие опции описывают поддерживаемые операции, которые можно выполнять при чтении существующего объекта данных:

- поддержка возможности чтения метаданных существующего объекта данных обозначена наличием опции cdmi_read_metadata у объекта;
- поддержка возможности чтения данных существующего объекта обозначена наличием опции cdmi_read_value у объекта;
- поддержка возможности чтения определенного диапазона байт значения объекта обозначена наличием опции cdmi_read_value_range у объекта.

8.4.3 Заголовки запроса

Заголовки HTTP запросов на чтение объекта данных CDMI, использующего тип содержимого CDMI, приведены в таблице 14.

Таблица 14 – Заголовки запроса – чтение объекта данных CDMI с использованием типа содержимого CDMI

Заголовок	Тип	Описание	Требование
Accept	Header String	"application/cdmi-object" или допустимой значение, см. 5.13.2	Опционально
X-CDMI-SpecificationVersion	Header String	Список версий, поддерживаемых клиентом, разделенных запятыми, например, "1.0.2, 1.5, 2.0"	Обязательно

8.4.4 Тело сообщения-запроса

Тело сообщения-запроса должно отсутствовать.

8.4.5 Заголовки ответа

Заголовки HTTP ответа на чтение объекта данных с использованием типа содержимого CDMI приведены в таблице 15.

Таблица 15– Заголовки ответа – чтение объекта данных CDMI с использованием типа содержимого CDMI

Заголовок	Тип	Описание	Требование
X-CDMI-SpecificationVersion	Строка заголовка	Сервер должен вернуть наибольший номер версии, поддерживаемой и клиентом, и сервером, например, "1.0.2". Если сервер не поддерживает ни одну из версий, поддерживаемых клиентом, сервер должен вернуть сообщение об ошибке 400 Bad Request.	Обязательно
Content-Type	Строка заголовка	"application/cdmi-object"	Обязательно
Location	Строка заголовка	Сервер должен вернуть URI, на который указывает ссылка, если объект является ссылкой.	Условно

8.4.6 Тело сообщения-ответа

Поля тела сообщения-ответа на запрос чтения объекта данных с использованием типа содержимого CDMI приведены в таблице 16.

Таблица 16 – Тело сообщения-ответа – чтение объекта данных CDMI с использованием типа содержимого CDMI

Имя поля	Тип	Описание	Требование
objectType	Строка JSON	"application/cdmi-object"	Обязательно
objectID	Строка JSON	ID объекта	Обязательно
objectName	Строка JSON	Имя объекта: для объектов в контейнере должно быть возвращено поле objectName; для объектов не в контейнере (доступных только по ID), поле objectName не существует и не должно возвращаться.	Условно
parentURI	Строка JSON	URI родительского объекта: для объектов в контейнере должно возвращаться поле parentURI; для объектов не в контейнере (доступных только по ID), поле parentURI не существовует и не должно возвращаться. Добавление objectName к parentURI должно всегда давать корректный URI объекта.	Условно
parentID	Строка JSON	ID родительского контейнера: для объектов в контейнере должно возвращаться поле parentID; для объектов не в контейнере (доступных только по ID), поле parentID не существует и не должно возвращаться.	Условно
domainURI	Строка JSON	URI домена-владельца	Обязательно
capabilitiesURI	Строка JSON	URI опций для объекта	Обязательно
completion-Status	Строка JSON	Строка, указывающая, находится ли объект в процессе создания (а после создания – был ли он создан успешно или с ошибкой). Значение должно быть строкой "Processing" или "Complete" или строкой, начинающейся с "Error".	Обязательно

Окончание таблицы 16

Имя поля	Тип	Описание	Требование
percent-Complete	Строка JSON	Если значение completionStatus равно "Processing", данное поле, при наличии, должно показывать процент выполнения операции создания объекта, числовым значением от 0 до 100. Если значение completionStatus равно "Complete", данное поле, при наличии, должно иметь значение "100". Если значение completionStatus начинается с "Error", данное поле, при наличии, может содержать любое целое число от 0 до 100.	Опционально
mimetype	Строка JSON	Тип MIME значения объекта данных	Обязательно
metadata	Объект JSON	Метаданные объекта данных. Данное поле включает любые пользовательские метаданные или метаданные системы данных, указанные в соответствующем поле тела запроса при создании, наряду с метаданными системы хранения, созданными облачным хранилищем. См. детальное описание метаданных в разделе 16.	Обязательно
valuerange	Строка JSON	Диапазон байт объекта, которые следует вернуть в поле value Если запрошен определенный диапазон, значение поля должно соответствовать запрошенным байтам. Если запрос выходит за границы данных, в поле диапазона вывода должно быть указано, что возвращено меньшее число байтов. Если значение объекта содержит промежутки (из-за несмежных диапазонов PUT), значение диапазона должно указывать на интервал до первого промежутка в значении объекта. Метаданные хранилища cdmi_size должны всегда хранить полный размер объекта, с учетом пропусков, заполненных нулями.	Обязательно
valuetransfer-encoding	Массив JSON строк JSON	Кодировка, использованная при передаче объекта данных. Определены два значения кодировки. - "utf-8" указывает на то, что объект данных содержит корректную строку UTF-8, и должен передаваться в поле value как строка UTF-8. - "base64" указывает на то, что объект данных содержит произвольную бинарную последовательность и должен передаваться в поле value как строка в кодировке base 64.	Обязательно
value	Строка JSON	Значение объекта данных - Если поле valuetransferencoding указывает на кодировку UTF-8, поле value должно содержать строку UTF-8 согласно правилам JSON, описанным в RFC 4627. - IЕсли поле valuetransferencoding указывает на кодировку base 64, поле value должно содержать строку, закодированную base 64 (RFC 4648). - Поле value должно предоставляться только если поле completionStatus содержит "Complete". - При чтении значения, для промежутков данных возвращаются нули.	Условно

Если в запрос GET указаны отдельные поля, только эти поля возвращаются в теле ответа. Запрошенные опциональные поля, отсутствующие в объекте, в ответе опускаются.

8.4.7 Статус ответа

Коды состояний HTTP, возникающих при чтении из объекта данных с использованием типа содержимого CDMI, описаны в таблице 17.

Таблица 17 – Коды состояния HTTP – чтение объекта данных CDMI с использованием типа содержимого CDMI

Статус HTTP	Описание
200 OK	Содержимое объекта данных возвращено в ответе.
202 Accepted	Объект данных в процессе создания. Клиент CDMI должен отслеживать поля completionStatus и percentComplete для определения состояния операции.
302 Found	URI является ссылкой на другой URI.
400 Bad Request	Запрос содержит неверные параметры или имена полей.

Окончание таблицы 17

Статус HTTP	Описание
401 Unauthorized	Неверные данные аутентификации/авторизации.
403 Forbidden	Клиент не обладает правами для выполнения данного запроса.
404 Not Found	Ресурс не найден по указанному URI.
406 Not Acceptable	Сервер не может передать объект, используя тип, указанный в заголовке Accept.

8.4.8 Примеры

Примеры

1 Применение GET к URI объекта данных для чтения всех полей объекта:

```
GET /MyContainer/MyDataObject.txt HTTP/1.1
Host: cloud.example.com
Accept: application/cdmi-object
X-CDMI-Specification-Version: 1.0.2
```

Будет получен следующий ответ.

```
HTTP/1.1 200 OK
X-CDMI-Specification-Version: 1.0.2
Content-Type: application/cdmi-object
{
«objectType» : «application/cdmi-object»,
«objectID» : «0000706D0010B84FAD185C425D8B537E»,
«objectName» : «MyDataObject.txt»,
«parentURI» : «/MyContainer/»,
«parentID» : «00007E7F00102E230ED82694DAA975D2»,
«domainURI» : «/cdmi_domains/MyDomain/»,
«capabilitiesURI» : «/cdmi_capabilities/dataobject/»,
«completionStatus» : «Complete»,
«mimetype» : «text/plain»,
«metadata» : {
        «cdmi_size» : «37»
},
«valuerange» : «0-36»,
«valuetransferencoding» : «utf-8»,
«value» : «This is the Value of this Data Object»
}
```

2 Применение GET к URI идентификатора объекта данных для чтения всех полей объекта по ID:

```
GET /cdmi_objectid/0000706D0010B84FAD185C425D8B537E
HTTP/1.1 Host: cloud.example.com
Accept: application/cdmi-object
X-CDMI-Specification-Version: 1.0.2
```

Будет получен следующий ответ.

```
HTTP/1.1 200 OK
Content-Type: application/cdmi-object
X-CDMI-Specification-Version: 1.0.2
{
«objectType» : «application/cdmi-object»,
«objectID» : «0000706D0010B84FAD185C425D8B537E»,
«objectName» : «MyDataObject.txt»,
«parentURI» : «/MyContainer/»,
«parentID» : «00007E7F00102E230ED82694DAA975D2»,
«domainURI» : «/cdmi_domains/MyDomain/»,
«capabilitiesURI» : «/cdmi_capabilities/dataobject/»,
«completionStatus» : «Complete»,
«mimetype» : «text/plain»,
«metadata» : {
        «cdmi_size» : «37»
},
```

«valuetransferencoding» : «utf-8»,
«valuerange» : «0-36»,
«value» : «This is the Value of this Data Object»
}

3 Применение GET к URI объекта данных для чтения полей value и mimetype:
GET /MyContainer/MyDataObject.txt?value;mimetype HTTP/1.1
Host: cloud.example.com
Accept: application/cdmi-object
X-CDMI-Specification-Version: 1.0.2
Будет получен следующий ответ.
HTTP/1.1 200 OK
Content-Type: application/cdmi-object
X-CDMI-Specification-Version: 1.0.2
{
«value» : «This is the Value of this Data Object»,
«mimetype» : «text/plain»
}

4 Применение GET к URI объекта данных для чтения первых 11 байт значения объекта:
GET /MyContainer/MyDataObject.txt?valuerange;value:0-10 HTTP/1.1
Host: cloud.example.com
Accept: application/cdmi-object
X-CDMI-Specification-Version: 1.0.2
Будет получен следующий ответ.
HTTP/1.1 200 OK
Content-Type: application/cdmi-object
X-CDMI-Specification-Version: 1.0.2
{
«valuerange» : «0-10»,
«value» : «VGhpcyBpcyB0aGU=»
}

8.5 Чтение объекта данных с использованием типа содержимого, отличного от CDMI

8.5.1 Обзор

Для чтения значения существующего объекта данных, следует выполнить запрос:
GET <root URI>/<ContainerName>/<DataObjectName>

где:

- <root URI> путь к облаку CDMI;
- <ContainerName> неотрицательное число промежуточных контейнеров;
- <DataObjectName> имя объекта данных для чтения.

К объекту возможно обратиться как <root URI>/cdmi_objectid/<objectID>.

8.5.2 Опции

Следующие опции описывают поддерживаемые операции, которые можно выполнять при чтении из существующего объекта данных:

- поддержка возможности чтения значения из существующего объекта данных обозначена наличием опции cdmi_read_value в объекте. При чтении из байта, в который не было записано значение при создании или изменении объекта, должно возвращать нулевой байт.
- поддержка возможности чтения обозначенного диапазона байтов значения из существующего объекта данных обозначена наличием опции cdmi_read_value_range в объекте. При чтении из байта, в который не было записано значение при создании или изменении объекта, должно возвращать нулевой байт.

8.5.3 Заголовок запроса

Заголовок запроса HTTP на чтение объекта данных CDMI с использованием типа содержимого, отличного от CDMI, показан в таблице 18.

Таблица 18 – Заголовок запроса – чтение объекта данных CDMI с использованием типа содержимого, отличного от CDMI

Заголовок	Тип	Описание	Требование
Range	Строка заголовка	Корректное описание диапазона (см. RFC 2616, п. 14.35.1)	Опционально

8.5.4 Тело сообщения-запроса

Тело в сообщении должно отсутствовать.

8.5.5 Заголовки ответа

Заголовки HTTP ответа на чтение объекта данных CDMI с использованием типа содержимого, отличного от CDMI, показаны в таблице 19.

Таблица 19– Заголовки ответа – чтение объекта данных CDMI с использованием типа содержимого, отличного от CDMI

Заголовок	Тип	Описание	Требование
Content-Type	Строка заголовка	Тип возвращаемого содержимого должен быть равен полю mimetype объекта данных.	Обязательно
Location	Строка заголовка	Сервер должен вернуть URI, на который указывает ссылка, если объект является ссылкой.	Опционально

8.5.6 Тело сообщения-ответа

К чтению объекта данных с использованием типа содержимого, отличного от CDMI, предъявляются следующие требования:

- тело сообщения-ответа является содержимым поля value объекта;
- в промежутках, возникающих из-за разрывов при записи, должны возвращаться нули.

8.5.7 Статус ответа

Коды состояний HTTP, возникающих при чтении из объекта данных с использованием типа содержимого, отличного от CDMI, описаны в таблице 20.

Таблица 20 – Коды состояний HTTP – чтение объекта данных CDMI с использованием типа содержимого, отличного от CDMI

Статус HTTP	Описание
200 OK	Содержимое объекта данных возвращено в ответе
206 Partial Content	Запрошенный диапазон данных содержимого объекта возвращен в ответе
302 Found	URI является ссылкой на другой URI
400 Bad Request	Запрос содержит неверные параметры или имена полей
401 Unauthorized	Неверные данные аутентификации/авторизации
403 Forbidden	Клиент не обладает правами для выполнения данного запроса
404 Not Found	Ресурс не найден по указанному URI

8.5.8 Примеры

Примеры –
1 Применение GET к URI объекта данных для чтения значения объекта:
GET /MyContainer/MyDataObject.txt HTTP/1.1
Host: cloud.example.com
Будет получен следующий ответ.
HTTP/1.1 200 OK
Content-Type: text/plain
Content-Length: 37
This is the Value of this Data Object

2 Применение GET к URI объекта данных для чтения первых 11 байт значения объекта данных:
GET /MyContainer/MyDataObject.txt HTTP/1.1
Host: cloud.example.com
Range: bytes=0-10

Будет получен следующий ответ.
HTTP/1.1 206 Partial Content
Content-Type: text/plain
Content-Range: bytes 0-10/37
Content-Length: 11
This is the

8.6 Изменение объекта данных с использованием типа содержимого CDMI

8.6.1 Обзор

Для изменения некоторых или всех полей существующего объекта данных следует выполнить запрос:

PUT <root URI>/<ContainerName>/<DataObjectName>

Для изменения значения существующего объекта данных следует выполнить запрос:

PUT <root URI>/<ContainerName>/<DataObjectName>?value:<range>

Для добавления, изменения или удаления определенных метаданных существующего объекта данных следует выполнить запрос:

PUT <root URI>/<ContainerName>/<DataObjectName>?metadata:<metadataname>;...

где:

- <root URI> путь к облаку CDMI;
- <ContainerName> неотрицательное число промежуточных контейнеров;
- <DataObjectName> имя объекта данных для изменения;
- <range> диапазон байт объекта данных для изменения.

К объекту данных возможно обратиться также как <root URI>/cdmi_objectid/<objectID>, изменение объекта не должно изменять его ID.

8.6.2 Опции

Следующие опции описывают поддерживаемые операции, которые можно выполнять при изменении существующего объекта данных:

- поддержка возможности изменения метаданных существующего объекта данных обозначена наличием опции cdmi_modify_metadata у объекта;
- поддержка возможности изменения значения и/или типа MIME существующего объекта данных обозначена наличием опции cdmi_modify_value у объекта;
- поддержка возможности изменения определенного диапазона байт значения существующего объекта данных обозначена наличием опции cdmi_modify_value_range у объекта.

8.6.3 Заголовки запроса

Заголовки HTTP запроса для изменения объекта данных CDMI с использованием типа содержимого CDMI перечислены в таблице 21.

Таблица 21 – Заголовки запроса – изменение CDMI объекта данных с использованием типа содержимого CDMI

Заголовок	Тип	Описание	Требование
Content-Type	Строка заголовка	"application/cdmi-object"	Обязательно
X-CDMI-SpecificationVersion	Строка заголовка	Список версий, поддерживаемых клиентом, разделенных запятыми, например "1.0.2, 1.5, 2.0"	Обязательно
X-CDMI-Partial	Строка заголовка	"true". Указывает на то, что объект находится в процессе изменения, и еще не был изменен полностью. При этом значение поля completionStatus должно быть установлено в "Processing". Если поле the completionStatus было ранее установлено в "Processing" включением данного заголовка при создании или изменении, при следующем изменении без данного заголовка поле completionStatus снова примет значение "Complete".	Опционально

8.6.4 Тело сообщения-запроса

Поля тела сообщения-запроса для изменения объекта данных CDMI с использованием типа содержимого CDMI перечислены в таблице 22.

Таблица 22 – Тело сообщения-запроса – изменение CDMI объекта данных с использованием типа содержимого CDMI

Имя поля	Тип	Описание	Требование
mimetype	Строка JSON	Тип MIME данных, содержащихся в поле value объекта данных. При наличии, заменяет существующий тип MIME. Данное поле может быть включено при изменении по значению, десериализации или копировании объекта данных. Данное поле должно храниться как часть объекта. Если данное поле не указано, имеющееся значение поля mimetype должно остаться неизменным. Данное поле не должно включаться при создании ссылки. Значение mimetype должно быть преобразовано к нижнему регистру перед сохранением.	Опционально
metadata	Объект JSON	Метаданные объекта данных. При наличии, новые метаданные заменяют имеющиеся. Если в URI указан отдельный пункт метаданных, остальные метаданные сохраняются. Детальное описание метаданных см. в разделе 16.	Опционально
domainURI	Строка JSON	URI домена-владельца В случае отличия от домена-родителя, пользователь должен иметь права cross_domain" (см. cdmi_member_privileges в таблице 64). Если не указано, сохраняется существующий домен.	Опционально
deserialize	Строка JSON	URI сериализованного объекта данных CDMI, который должен быть десериализован для изменения существующего объекта данных. ID сериализованного объекта должен соответствовать ID конечного объекта.	Опционально[a]
copy	Строка JSON	URI объекта данных или очереди CDMI, которые должны быть скопированы в существующий объект данных.	Опционально[a]
deserialize-value	Строка JSON	Сериализованный объект данных (см. гл. 15), кодированный по правилам base 64 (RFC 4648). ID сериализованного объекта данных должен соответствовать ID изменяемого объекта.	Опционально[a]
valuetransfer-encoding	Массив JSON строк JSON	Кодировка, использованная при передаче объекта данных. Определены два значения кодировки. - "utf-8" указывает на то, что объект данных содержит корректную строку UTF-8, и должен передаваться в поле value как строка UTF-8. - "base64" указывает на то, что объект данных содержит произвольную бинарную последовательность и должен передаваться в поле поля value строкой в кодировке base 64. Задание содержимого поля value объекта данных иное, чем корректная строка в кодировке base 64, должно возвращать клиенту ошибку 400 Bad Request. Данное поле должно включаться лишь при изменении объекта по значению. Если иное не указано клиентом, сервер должен установить значение поля valuetransferencoding равным "utf-8". Данное поле должно храниться как часть объекта.	Опционально
value	Строка JSON	Новые данные объекта. При наличии, значение поля заменяет существующее значение. Если поле valuetransferencoding указывает на кодировку UTF-8, значение должно быть строкой UTF-8, сформированной по правилам JSON (RFC 4627). Если поле valuetransferencoding указывает на кодировку base 64, значение должно быть вначале кодировано в base 64 (RFC 4648). Если в запросе указан диапазон, новые данные должны быть вставлены в указанный диапазон. Если при этом возникнут промежутки, они считаются заполненными нулями и должны учитываться при вычислении размера значения. При сохранении диапазона value должно содержать строку в кодировке base 64, и значение поля valuetransferencoding должно быть установлено в "base64".	Опционально

[a] В каждой данной операции должно указываться лишь одно из этих полей. За исключением value, эти поля не должны храниться. Если имеется более одного такого поля, сервер должен вернуть сообщение об ошибке 400 Bad Request.

8.6.5 Заголовок ответа

Заголовок HTTP ответа на запрос изменения объекта данных CDMI с использованием типа содержимого CDMI указан в таблице 23.

Таблица 23 – Заголовок ответа – изменение CDMI объекта данных с использованием типа содержимого CDMI

Заголовок	Тип	Описание	Требование
Location	Строка заголовка	Сервер должен вернуть URI, на который идет переадресация, если объект является ссылкой.	Условно

8.6.6 Тело сообщения-ответа

Сообщение-ответ может содержать тело, соответствующее RFC 2616.

8.6.7 Статус ответа

Коды состояния HTTP возникающие при изменении объекта данных с использованием типа содержимого CDMI, перечислены в таблице 24.

Таблица 24 – Коды состояний HTTP – изменение CDMI объекта данных с использованием типа содержимого CDMI (лист 1 из 2)

Статус HTTP	Описание
204 No Content	Операция успешна; никаких данных не возвращено.
302 Found	URI является ссылкой на другой URI.
400 Bad Request	Запрос содержит неверные параметры или имена полей.
401 Unauthorized	Неверные данные аутентификации/авторизации.
403 Forbidden	Клиент не обладает правами для выполнения данного запроса.
404 Not Found	Ресурс не найден по указанному URI.
409 Conflict	Операция конфликтует с блокировкой не-CDMI протокола доступа или может вызвать ошибку передачи на сервер.

8.6.8 Примеры

Примеры

1 Применение PUT к URI объекта данных для установления новых значений полей:

```
PUT /MyContainer/MyDataObject.txt HTTP/1.1
Host: cloud.example.com
Content-Type: application/cdmi-object
X-CDMI-Specification-Version: 1.0.2
{
«mimetype» : «text/plain»,
«metadata» : {
«colour» : «blue»,
«length» : «10»
},
«value» : «This is the Value of this Data Object»
}
```

Будет получен следующий ответ.

```
HTTP/1.1 204 No Content
```

2 Применение PUT к URI объекта данных для изменения типа MIME:

```
PUT /MyContainer/MyDataObject.txt?mimetype HTTP/1.1
Host: cloud.example.com
Content-Type: application/cdmi-object
X-CDMI-Specification-Version: 1.0.2
{
«mimetype» : «text/plain»
}
```

Будет получен следующий ответ.

```
HTTP/1.1 204 No Content
```

3 Применение PUT к URI объекта данных для изменения диапазона значения:
PUT /MyContainer/MyDataObject.txt?value:21-24 HTTP/1.1
Host: cloud.example.com
Content-Type: application/cdmi-object
X-CDMI-Specification-Version: 1.0.2
{
«value» : «dGhhdA==»
}
Будет получен следующий ответ.
HTTP/1.1 204 No Content

При изменении значения без указания кодировки передачи, клиент должен учитывать текущую кодировку передачи объекта. Если клиент посылает значение строкой UTF-8 для изменения существующего объекта со значением valuetransferencoding равным «base64», должна быть возвращена ошибка. Если клиент отсылает строку base 64 для изменения объекта с valuetransferencoding равным «utf-8», это не должно приводить к ошибке, но приводить к сохранению символьной последовательности без изменений как строки base 64 вместо перекодирования ожидаемых данных в кодировку base 64.

4 Применение PUT к URI объекта данных для замены метаданных:
PUT /MyContainer/MyDataObject.txt?metadata HTTP/1.1
Host: cloud.example.com
Content-Type: application/cdmi-object
X-CDMI-Specification-Version: 1.0.2
{
«metadata» : {
«colour» : «red»,
«number» : «7»
}
}
Будет получен следующий ответ.
HTTP/1.1 204 No Content

5 Применение PUT к URI объекта данных для добавления новых метаданных и сохранения существующих:
PUT /MyContainer/MyDataObject.txt?metadata:shape HTTP/1.1
Host: cloud.example.com
Content-Type: application/cdmi-object
X-CDMI-Specification-Version: 1.0.2
{
«metadata» : {
«shape» : «round»
}
}
Будет получен следующий ответ.
HTTP/1.1 204 No Content

6 Применение PUT к URI объекта данных для замены лишь одного элемента метаданных новым значением:
PUT /MyContainer/MyDataObject.txt?metadata:colour HTTP/1.1
Host: cloud.example.com
Content-Type: application/cdmi-object
X-CDMI-Specification-Version: 1.0.2
{
«metadata» : {
«colour» : «green»
}
}
Будет получен следующий ответ.
HTTP/1.1 204 No Content

8.7 Изменение объекта данных с использованием типа содержимого, отличного от CDMI

8.7.1 Обзор

Для изменения значения существующего объекта данных следует выполнить запрос:
PUT <root URI>/<ContainerName>/<DataObjectName>

где:

- <root URI> путь к облаку CDMI.
- <ContainerName> неотрицательное число промежуточных контейнеров.
- <DataObjectName> имя изменяемого объекта данных.

К объекту можно обратиться также как <root URI>/cdmi_objectid/<objectID>. Изменение не должно изменять ID объекта.

8.7.2 Опции

Следующие опции описывают поддерживаемые операции, которые можно выполнять при изменении существующего объекта данных:

- поддержка возможности изменения значения существующего объекта и/или типа MIME обозначается присутствием опции cdmi_modify_value у объекта;
- поддержка возможности изменения выбранного диапазона байт значения объекта обозначается наличием cdmi_modify_value_range у объекта.

8.7.3 Заголовки запроса

Заголовки HTTP запроса на изменение объекта данных CDMI с использованием типа содержимого, отличного от CDMI, приведены в таблице 25.

Таблица 25 – Заголовки запроса – изменение объекта данных CDMI с использованием типа содержимого, отличного от CDMI

Заголовок	Тип	Описание	Требование
Content-Type	Строка заголовка	Тип данных, сохраняемых в объект. Указанное здесь значение должно быть использовано в поле mimetype объекта данных CDMI.	Обязательно
Content-Range	Строка заголовка	Корректное обозначение диапазона (см. RFC 2616, гл. 14.35.1)	Опционально
X-CDMI-Partial	Строка заголовка	"true". Указывает на то, что объект находится в процессе изменения, и еще не был изменен полностью. При этом значение поля completionStatus должно быть установлено в "Processing". Если поле completionStatus было ранее установлено в "Processing" включением данного заголовка при создании или изменении, следующее изменение без данного поля должно установить значения поле completionStatus обратно на "Complete".	Опционально

8.7.4 Тело сообщения-запроса

Тело сообщения-запроса на изменение данных содержит данные для сохранения в значение объекта.

8.7.5 Заголовок ответа

Заголовок ответа HTTP на изменение объекта данных CDMI с использованием типа содержимого, отличного от CDMI, приведен в таблице 26.

Таблица 26 – Заголовок ответа – изменение объекта данных CDMI с использованием типа содержимого, отличного от CDMI

Заголовок	Тип	Описание	Требование
Location	Строка заголовка	Сервер должен вернуть URI, на который происходит переадресация, если объект является ссылкой.	Условно

8.7.6 Тело сообщения-ответа

Сообщение-ответ может содержать тело, соответствующее RFC 2616.

8.7.7 Статус запроса

Коды состояний HTTP, возникающих в ответ на изменение объекта данных CDMI с использованием типа содержимого, отличного от CDMI, приведены в таблице 27.

Таблица 27 – Коды состояния HTTP – изменение объекта данных CDMI с использованием типа содержимого, отличного от CDMI

Статус HTTP	Описание
204 No Content	Операция успешна; никаких данных не возвращено
302 Found	URI является ссылкой на другой URI
400 Bad Request	Запрос содержит неверные параметры или имена полей
401 Unauthorized	Неверные данные аутентификации/авторизации
403 Forbidden	Клиент не обладает правами для выполнения данного запроса
404 Not Found	Ресурс не найден по указанному URI
409 Conflict	Операция конфликтует с блокировкой не-CDMI протокола доступа или может вызвать ошибку передачи на сервер

8.7.8 Примеры

Примеры

1 Применение PUT к URI объекта данных для изменения значения объекта данных:
PUT /MyContainer/MyDataObject.txt HTTP/1.1
Host: cloud.example.com
Content-Type: text/plain
Content-Length: 37
This is the value of this data object
Будет получен следующий ответ.
HTTP/1.1 204 No Content

2 Применение PUT к URI объекта данных для изменения четырех байтов в значении объекта данных:
PUT /MyContainer/MyDataObject.txt HTTP/1.1
Host: cloud.example.com
Content-Range: bytes 21-24/37
Content-Type: text/plain
Content-Length: 4
that
Будет получен следующий ответ.
HTTP/1.1 204 No Content

8.8 Удаление объекта данных с использованием типа содержимого CDMI

8.8.1 Обзор

Для удаления существующего объекта данных следует выполнить запрос:
DELETE <root URI>/<ContainerName>/<DataObjectName>

где:

- <root URI> путь к облаку CDMI.
- <ContainerName> неотрицательное число промежуточных контейнеров.
- <DataObjectName> имя удаляемого объекта данных.

К объекту можно также обратиться как <root URI>/cdmi_objectid/<objectID>.

8.8.2 Опции

Следующие опции описывают поддерживаемые операции, которые можно выполнять при удалении существующего объекта данных:

- поддержка возможности удаления существующего объекта данных обозначается наличием опции cdmi_delete_dataobject у объекта.

8.8.3 Заголовок запроса

Заголовок HTTP запроса на удаление CDMI объекта с использованием типа содержимого CDMI приведен в таблице 28.

Таблица 28 – Заголовок запроса – удаление CDMI объекта с использованием типа содержимого CDMI

Заголовок	Тип	Описание	Требование
X-CDMI-SpecificationVersion	Строка заголовка	Список версий, поддерживаемых клиентом, разделенных запятыми, например "1.0.2, 1.5, 2.0"	Обязательно

8.8.4 Тело сообщения-запроса

Сообщение-запрос может содержать тело, соответствующее RFC 2616.

8.8.5 Заголовки ответа

Сообщение-ответ может содержать заголовки, соответствующие RFC 2616.

8.8.6 Тело сообщения-ответа

Сообщение-ответ может содержать тело, соответствующее RFC 2616.

8.8.7 Статус запроса

В таблице 29 приведены коды состояний HTTP, возникающих при удалении CDMI объекта с использованием типа содержимого CDMI.

Таблица 29 – Коды состояния HTTP Status Codes – удаление CDMI объекта с использованием типа содержимого CDMI

Статус HTTP	Описание
204 No Content	Объект данных успешно удален.
400 Bad Request	Запрос содержит неверные параметры или имена полей.
401 Unauthorized	Неверные данные аутентификации/авторизации.
403 Forbidden	Клиент не обладает правами для выполнения данного запроса.
404 Not Found	Ресурс не найден по указанному URI.
409 Conflict	Операция конфликтует с блокировкой не-CDMI протокола доступа или может вызвать ошибку передачи на сервер.

8.8.8 Пример

Пример – Применение DELETE к URI объекта данных:
DELETE /MyContainer/MyDataObject.txt HTTP/1.1
Host: cloud.example.com
X-CDMI-Specification-Version: 1.0.2
Будет получен следующий ответ.
HTTP/1.1 204 No Content

8.9 Удаление объекта данных с использованием типа содержимого, отличного от CDMI

8.9.1 Обзор

Для удаления существующего объекта данных следует выполнить запрос:
DELETE <root URI>/<ContainerName>/<DataObjectName> Где:
- <root URI> путь к облаку CDMI.
- <ContainerName> неотрицательное число промежуточных контейнеров.
- <DataObjectName> имя удаляемого объекта.

К объекту можно также обратиться как <root URI>/cdmi_objectid/<objectID>.

8.9.2 Опции

Следующие опции описывают поддерживаемые операции, которые можно выполнять при удалении существующего объекта данных:

- поддержка возможности удаления существующего объекта данных обозначена наличием способности cdmi_delete_dataobject у объекта.

8.9.3 Заголовки запроса

Сообщение-ответ может содержать заголовки, соответствующие RFC 2616.

8.9.4 Тело сообщения-запроса

Сообщение-запрос может содержать тело, соответствующее RFC 2616.

8.9.5 Заголовки ответа

Сообщение-ответ может содержать заголовки, соответствующие RFC 2616.

8.9.6 Тело сообщения-ответа

Сообщение-ответ может содержать тело, соответствующее RFC 2616.

8.9.7 Статус запроса

Таблица 30 описывает коды состояний HTTP, возникающих при удалении CDMI объекта с использованием типа содержимого, отличного от CDMI

Таблица 30– Коды состояний HTTP – удаление CDMI объекта с использованием типа содержимого, отличного от CDMI

HTTP Статус	Описание
204 No Content	Объект данных успешно удален.
400 Bad Request	Запрос содержит неверные параметры или имена полей.
401 Unauthorized	Неверные данные аутентификации/авторизации.
403 Forbidden	Клиент не обладает правами для выполнения данного запроса.
404 Not Found	Ресурс не найден по указанному URI.
409 Conflict	Операция конфликтует с блокировкой не-CDMI протокола доступа или может вызвать ошибку передачи на сервер.

8.9.8 Пример

Пример – Применение DELETE к URI объекта данных:
DELETE /MyContainer/MyDataObject.txt HTTP/1.1 Host: cloud.example.com
Будет получен следующий ответ.
HTTP/1.1 204 No Content

9 Операции с ресурсами вида объект-контейнер

9.1 Обзор

Объекты-контейнеры – основное средство группировки в CDMI™, аналогич-ные папкам файловой системы. Каждый объект-контейнер имеет неотрицательное число дочерних объектов и набор полей, включающих стандартизованные и произ-вольные метаданные. Эти метаданные генерируются облачным хранилищем или задаются пользователем. В модели CDMI контейнеры адресуются двумя способами:

- по имени (например, http://cloud.example.com/container/);
- по ID объекта (например, http://cloud.example.com/cdmi_objectid/0000706D0010B84FAD185C425D8B537E.

Каждый контейнер имеет единственный глобально-уникальный ID, который остается неизменным на протяжении жизненного цикла объекта. Каждый контейнер также может иметь один или несколько адресов URI, что позволяет адресовать кон-тейнеры. Следуя соглашениям об иерархическом пути, URI контейнера должны со-стоять из одного или нескольких имен контейнеров, разделенных наклонной чертой («/») и заканчиваться наклонной чертой («/»).

При запросе к ресурсу существующего контейнера, если завершающая наклонная черта в его URI опущена, сервер должен вернуть код состояния HTTP 301 Moved Permanently с заголовком Location, содержащим URI с завершающей наклон-ной чертой.

При выполнении запроса CDMI на создание нового ресурса контейнера, если опущена завершающая наклонная черта в URI, сервер должен вернуть код состоя-ния HTTP 400 Bad Request.

Запросы вне модели CDMI на создание ресурса контейнера должны включать завершающую URI наклонную черту; в противном случае запрос будет распознан как запрос на создание объекта.

Контейнеры могут быть вложенными.

Пример – Следующий URI указывает на вложенный контейнер:
http://cloud.example.com/container/subcontainer/

Вложенный контейнер имеет родительский объект-контейнер, должен быть включен в поле children контейнера-родителя и наследовать метаданные системы данных и список прав доступа родительского контейнера.

Данная модель позволяет устанавливать прямое соответствие между управ-ляемым CDMI облачным хранилищем и файловыми системами (например, NFSv4 или WebDAV). Если объект-контейнер CDMI экспортируется в файловую систему, файловая система может предоставлять особые механизмы доступа к метаданным CDMI. При создании файлов и папок в файловой системе они становятся видимыми через интерфейс CDMI, служащий путем к данным. Соответствие между конструкциями файловой системы и объектами данных, контейнерами и метаданными CDMI находится за рамками настоящего стандарта.

Для получения отдельного поля объекта контейнера необходимо указать имя поля после знака «?» на конце URI объекта.

Пример – Следующий URI возвращает только поле children в теле сообщения-ответа:
http://cloud.example.com/container/?children

Указанием диапазона после названия поля children можно осуществлять до-ступ к диапазону поля children.

Пример – Следующий URI возвращает имена первых трех потомков из поля children:
http://cloud.example.com/container/?children:0-2

Диапазоны потомков определяются аналогично диапазону байт, согласно гл. 14.35.1 в RFC 2616. Клиент может определить количество потомков запросом поля childrenrange без запроса диапазона потомков.

Возможно указать список полей, разделенных «;», что позволяет обращаться к нескольким полям в одном запросе.

Пример – Следующий URI вернет поля children и metadata в теле сообщения-ответа:
http://cloud.example.com/container/?children;metadata

Если клиент поддерживает или использует поля десериализации, которые не определены в настоящем стандарте, эти поля должны храниться как часть объекта.

9.1.1 Метаданные контейнера

Могут быть предоставлены следующие опциональные метаданные системы данных (см. таблице 31).

Таблица 31 – Метаданные контейнера

Имя метаданных	Тип	Описание	Требование
cdmi_assignedsize	Строка JSON	Число байт, которые передаются через внешние протоколы (например, устройство может использовать тонкую инициализацию). Это число может ограничивать cdmi_size.	Опционально

Метаданные контейнера могут также включать произвольные пользователь-ские метаданные и метаданные системы данных как описано в разделе 16.

9.1.2 Зарезервированные имена метаданных контейнера

Настоящий стандарт определяет зарезервированные имена контейнеров, которые не должны использоваться при создании новых контейнеров. При попытке создать или контейнер с одним из этих имен сервер должен вернуть клиенту код HTTP 400 Bad Request.

Зарезервированы следующие имена контейнеров:

- cdmi_objectid;
- cdmi_domains;
- cdmi_capabilities;
- cdmi_snapshots;
- cdmi_versions.

В последующих версиях настоящего стандарта могут быть добавлены новые зарезервированные имена, поэтому реализации сервера не должны допускать создания пользовательских контейнеров с именами, начинающимися с «cdmi_».

9.1.3 Адресация объекта-контейнера

Каждый контейнер может адресоваться одним или несколькими уникальными URI, и все операции должны выполняться через эти URI. Возможны случаи, когда объект-контейнер может доступен через несколько виртуальных путей. Например, http://cloud.example.com/users/snia/cdmi/ также доступен через http://snia.example.com/cdmi/. Конфликт записи через разные пути должен регулиро-ваться аналогично обработке конфликта записи через один и тот же путь, по прин-ципу возможной связности (см. 9.2).

9.1.4 Представления объекта-контейнера

Представления в данном разделе показаны с использованием нотации JSON. И клиенты, и серверы должны поддерживать UTF-8 представление JSON. В телах сообщений-запросов и ответов, поля JSON могут указываться и возвращаться в лю-бом порядке, за исключением полей childrenrange и children, которые указываются последними и в данном порядке.

9.2 Создание объекта-контейнера с использованием типа содержимого CDMI

9.2.1 Обзор

Для создания нового объекта-контейнера следует выполнить запрос:
PUT <root URI>/<ContainerName>/<NewContainerName>/

где:

- <root URI> путь к облаку CDMI;
- <ContainerName> неотрицательное число уже существующих объектов-контейнеров, разделенных одной наклонной чертой (т.е., «/»);
- <NewContainerName> имя нового создаваемого объекта-контейнера.

После создания, к контейнеру можно обращаться как <root URI>/cdmi_objectid/<objectID>/.

9.2.2 Отсроченное завершение создания

В ответ на запрос создания объекта-контейнера, сервер может вернуть код 202 Accepted, что указывает на то, что объект находится в процессе создания. Это полезно в случае длительных операций (например, десериализации исходного объекта данных для создания сложной иерархии объектов). Такой ответ означает, что:

- сервер должен вернуть заголовок Location, содержащий URI к создавае-мому объекту со статусом HTTP 202 Accepted;
- статус 202 Accepted со стороны сервера удостоверяет, что были пройдены несколько проверок:
- пользователь авторизован для создания нового объекта-контейнера;
- пользователь авторизован для чтения любого исходного объекта, ко-торый необходимо переместить, скопировать, сериализовать или десериализовать;
- достаточно места для создания объекта-контейнера или, по крайней мере, достаточно места для создания URI к сообщению об ошибке.

Клиент может не иметь опции немедленно обратиться к созданному объекту, например, из-за задержек, вызванных использованием реализацией целостности в конечном итоге.

Для отслеживания процесса создания клиент выполняет операции GET к URI. В ответ сервер возвращает два поля в теле сообщения-ответа, которые описывают текущее состояние:

- обязательное текстовое поле completionStatus содержит «Processing», «Complete», либо строку сообщения об ошибке, начинающуюся с «Error»;
- опциональное поле percentComplete содержит процент выполнения приня-того запроса PUT (от 0 ло 100). GET не возвращает объектов-потомков контейнера, если completionStatus не равно «Complete».

Если создание объекта завершается с ошибкой, создается URI, а поле comple-tionStatus устанавливается равным сообщению об ошибке. Удаление URI после об-работки ошибки возлагается на клиента.

9.2.3 Опции

Следующие опции описывают поддерживаемые операции при создании но-вого объекта:

- поддержка возможности создания нового объекта-контейнера обо-значается наличием опции cdmi_create_container в родительском контейнере;
- если объект, создаваемый в родительском контейнере, является ссылкой, поддержка этой операции обозначается наличием опции cdmi_create_reference в родительском контейнере;
- если новый объект является копией существующего, поддержка копирова-ния обозначается наличием опции cdmi_copy_container в родительском контейнере;
- если новый контейнер создается в результате перемещения, поддержка этой операции обозначается наличием опции cdmi_move_container у родительского контейнера;
- если новый объект создается в результате операции десериализации, поддержка этой операции обозначается наличием опции cdmi_deserialize_container в родительском контейнере.

9.2.4 Заголовки запроса

Заголовки HTTP запросов на создание объекта-контейнера CDMI с использованием типа содержимого CDMI приведены в таблице 32.

Таблица 32– Заголовки запроса – создание объекта-контейнера с использованием типа содержимого CDMI

Заголовок	Тип	Описание	Требование
Accept	Строка заголовка	"application/cdmi-container" или совместимое значение со-гласно 5.13.2	Опционально
Content-Type	Строка заголовка	"application/cdmi-container"	Обязательно
X-CDMI-SpecificationVersion	Строка заголовка	Список версий, поддерживаемых клиентом, разделенных запятыми, например, "1.0.2, 1.5, 2.0"	Обязательно

9.2.5 Тело сообщения-запроса

Поля тела сообщения-запроса на создание объекта-контейнера с использованием типа содержимого CDMI приведены в таблице 33.

Таблица 33 – Тело сообщения-запроса – создание объекта-контейнера с использованием типа содержимого CDMI (лист 1 из 2)

Имя поля	Тип	Описание	Требование
metadata	Объект JSON	Метаданные объекта-контейнера Если данное поле включено при десериализации, сериализации, копировании или перемещении объекта-контейнера, то значение этого поля должно заменять метаданные URI источника. Если данное поле не включено при десериализации, сериализации, копировании или перемещении объекта-контейнера, то следует использовать метаданные URI источника. Если данное поле включено при создании нового объекта по значению, это поле должно быть использовано как метаданные. Если данное поле не включено при создании нового контейнера по значению, этому полю следует поставить в соответствие пустой объект JSON (т.е., "{}"). Данное поле не должно указываться при создании ссылки на объект-контейнер	Опционально
DomainURI	Строка JSON	URI домена-владельца В случае отличия от родительского домена, пользователь должен обладать правами cross_domain (см. cdmi_member_privileges в таблице 64). Если не указано, должен использоваться родительский домен	Опционально
exports	Объект JSON	Структура для каждого из возможных протоколов для данного объекта-контейнера (см. гл. 13). Данное поле не должно указываться при создании ссылки	Опционально

Окончание таблицы 33

Имя поля	Тип	Описание	Требование
Deserialize	Строка JSON	URI сериализованного объекта данных CDMI, который должен быть десериализован при создании нового контейнера, включая всех потомков внутри исходного сериализованного объекта (см. гл. 15). При десериализации контейрера, ни один из экспортированных протоколов исходного сериализованного объекта не должен применяться к вновь создаваемому объекту(ам)	Опционально[a]
copy	Строка JSON	URI объекта-контейнера CDMI, который должен быть скопирован в новый объект-контейнер, включая всех потомков исходного объекта. При копировании объекта-контейнера, экспортированные протоколы не сохраняются в копии	Опционально[a]
move	Строка JSON	URI существующего локального или удаленного объекта-контейнера CDMI (URI источника), который должен быть перемещен, включая все объекты-потомки, в URI, указанный в команде PUT. Содержимое объекта-контейнера и всех потомков, включая ID объекта, должны при перемещении сохраняться (после успешного копирования). Если недостаточно прав для чтения объекта по URI источника, записи объекта по URI нового объекта, удаления объектов по URI источника или одна из этих операций завершается с ошибкой, сервер должен вернуть код результата 400 Bad Request, причем исходный и конечный объекты должны сохраниться неизменными	Опционально[a]
reference	Строка JSON	URI объекта-контейнера CDMI, на который должна быть сделана переадресация. Если при создании ссылки указаны другие поля, сервер должен вернуть код ошибки 400 Bad Request	Опционально[a]
deserialize-value	Строка JSON	Сериализованный объект-контейнер (см. гл. 15), кодированный с помощью base 64 (RFC 4648). ID сериализованного контейнера должен соответствовать ID контейнера-назначения	Опционально[a]

[a] Лишь одно из этих полей должно быть указано в любой из операций. За исключением поля value, данные поля не должны сохраняться. Если указано более чем одно из этих полей, сервер должен вернуть сообщение об ошибке 400 Bad Request

9.2.6 Заголовки ответа

Заголовки HTTP ответа на создание объекта-контейнера CDMI с использова-нием типа содержимого CDMI указаны в таблице 34.

Таблица 34– Заголовки ответа – создание объекта-контейнера с использованием типа содержимого CDMI

Заголовок	Тип	Описание	Требование
Content-Type	Строка заголовка	"application/cdmi-container"	Обязательно
X-CDMI-SpecificationVersion	Строка заголовка	Сервер должен вернуть наибольший номер версии, поддерживаемой и сервером, и клиентом, например, "1.0.2". Если сервер не поддерживает ни одной версии, поддерживаемой клиентом, сервер должен вернуть код состояния 400 Bad Request	Обязательно

9.2.7 Тело сообщения-ответа

Поля тела сообщения-ответа на создание объекта-контейнера CDMI с исполь-зованием типа содержимого CDMI приведены в таблице 35.

Таблица 35 – Тело сообщения-ответа – создание объекта-контейнера с использованием типа содержимого CDMI

Имя поля	Тип	Описание	Требование
objectType	Строка JSON	"application/cdmi-container"	Обязательно
objectID	Строка JSON	ID объекта	Обязательно

Окончание таблицы 35

Имя поля	Тип	Описание	Требование
objectName	Строка JSON	Имя объекта	Обязательно
parentURI	Строка JSON	URI родительского объекта Добавление objectName к parentURI должно всегда давать корректный URI объекта.	Обязательно
parentID	Строка JSON	ID родительского объекта-контейнера	Обязательно
domainURI	Строка JSON	URI домена-владельца	Обязательно
capabilitiesURI	Строка JSON	URI опций объекта	Обязательно
Completion-Status	Строка JSON	Строка, указывающая, находится ли объект в процессе создания, а после завершения операции – был ли он создан успешно или с ошибкой. Значение должно быть строкой "Processing", "Complete" или строкой, начинающейся с "Error".	Обязательно
percent-Complete	Строка JSON	Если значение completionStatus равно "Processing", данное поле, при наличии, должно показывать процент выполнения операции создания объекта, числовым значением от 0 до 100. Если значение completionStatus равно "Complete", данное поле, при наличии, должно иметь значение "100". Если значение completionStatus начинается с "Error", данное поле, при наличии, может содержать любое целое число от 0 до 100.	Опционально
metadata	Объект JSON	Метаданные объекта-контейнера. Могут включать пользовательские метаданные и метаданные системы данных, указанные в теле запроса на создание (поле metadata) вместе с метаданными системы хранения, создаваемыми облачным хранилищем. Подробнее о метаданных см. 16.	Обязательно
exports	Объект JSON	Структура для каждого протокола, разрешенного для данного объекта-контейнера. См. гл. 13.	Опционально[a]
snapshots	Массив JSON	Один или несколько URI снимков состояния объектов-контейнеров. См. гл. 14.	Опционально[a]
childrenrange	Строка JSON	Потомки контейнера в виде диапазона. При запросе диапазона потомков выдается содержимое этого поля в виде диапазона.	Обязательно
children	Массив JSON	Имена объекто-потомков контейнера. Контейнеры-потомки начинаются символом "/".	Обязательно
[a] Возвращается только при наличии			

9.2.8 Статус запроса

В таблице 36 приведены коды состояния HTTP, возникающие при создании объекта-контейнера с использованием типа содержимого CDMI.

Таблица 36 – Коды состояния HTTP – создание объекта-контейнера с использованием типа содержимого CDMI

Статус HTTP	Описание
201 Created	Новый контейнер был создан.
202 Accepted	Новый контейнер в процессе создания. Клиент CDMI должен отслеживать значения полей completionStatus и percentComplete для определения текущего статуса операции.
400 Bad Request	Запрос содержит неверные параметры или имена полей
401 Unauthorized	Неверные данные аутентификации/авторизации.
403 Forbidden	Клиент не обладает правами для выполнения данного запроса.
404 Not Found	Ресурс не найден по указанному URI.
409 Conflict	Контейнер с таким именем уже существует.

9.2.9 Пример

Пример – Применение PUT к URI объекта-контейнера: имя и метаданные:
PUT /MyContainer/HTTP/1.1

Host: cloud.example.com
Accept: application/cdmi-container
Content-Type: application/cdmi-container
X-CDMI-Specification-Version: 1.0.2
{
«metadata» : {
},
«exports» : {
«OCCI/iSCSI»: {
«identifier»: «00007E7F00104BE66AB53A9572F9F51E»,
«permissions»: [
«http://example.com/compute/0/»,
«http://example.com/compute/1/»
]
},
«Network/NFSv4» : {
«identifier» : «/users»,
«permissions» : «domain»
}
}
}
Будет получен следующий ответ.
HTTP/1.1 201 Created
Content-Type: application/cdmi-container
X-CDMI-Specification-Version: 1.0.2
{
«objectType» : «application/cdmi-container»,
«objectID» : «0000706D0010B84FAD185C425D8B537E»,
«objectName» : «MyContainer/»,
«parentURI» : «/»,
«parentID» : «00007E7F0010128E42D87EE34F5A6560»,
«domainURI» : «/cdmi_domains/MyDomain/»,
«capabilitiesURI» : «/cdmi_capabilities/container/»,
«completionStatus» : «Complete»,
«metadata» : {

},
«exports» : {
«OCCI/iSCSI» : {
«identifier» : «0000706D0010B84FAD185C425D8B537E»,
«permissions» : «00007E7F00104EB781F900791C70106C»
},
«Network/NFSv4» : {
«identifier» : «/users»,
«permissions» : «domain»
}
},
«childrenrange» : «»,
«children» : [

]
}

9.3 Создание объекта-контейнера с использованием типа содержимого, отличного от CDMI

9.3.1 Обзор

Для создания нового объекта-контейнера следует выполнить запрос:
PUT <root URI>/<ContainerName>/<NewContainerName>/

где:

- <root URI> путь к облаку CDMI;

– <ContainerName> неотрицательное число уже существующих объектов-контейнеров, имена которых разделены символами наклонной черты (т.е., «/»);

– <NewContainerName> имя создаваемого контейнера.

После создания к контейнеру можно обращаться как <root URI>/cdmi_objectid/<objectID>/.

Наличие завершающей наклонной черты в URI, по которому выполняется операция PUT, указывает на создание объекта-контейнера, в отличие от создания объекта данных.

9.3.2 Опции

Следующие опции описывают поддерживаемые операции при создании но-вого объекта:

– поддержка возможности создания нового объекта-контейнера обознача-ется наличием опции cdmi_create_container в родительском контейнере.

9.3.3 Заголовки запроса

Сообщение-запрос может содержать заголовки, соответствующие RFC 2616.

9.3.4 Тело сообщения-запроса

Тело запроса должно отсутствовать.

9.3.5 Заголовки ответа

Сообщение-ответ может содержать заголовки, соответствующие RFC 2616.

9.3.6 Тело сообщения-ответа

Сообщение-ответ может содержать тело, соответствующее RFC 2616.

9.3.7 Статус запроса

В таблице 37 приведены коды состояний HTTP, возникающих при создании объекта-контейнера с использованием типа содержимого, отличного от CDMI.

Таблица 37 – Коды состояний HTTP – создание объекта-контейнера с использованием типа содержимого, отличного от CDMI

Статус HTTP	Описание
201 Created	Новый объект-контейнер был создан
400 Bad Request	Запрос содержит неверные параметры или имена полей
401 Unauthorized	Неверные данные аутентификации/авторизации.
403 Forbidden	Клиент не обладает правами для выполнения данного запроса.
409 Conflict	Контейнер с таким именем уже существует.

9.3.8 Пример

Пример – Применение PUT к URI имени объекта-контейнера:
PUT /MyContainer/ HTTP/1.1
Host: cloud.example.com
Будет получен следующий ответ.
HTTP/1.1 201 Created

9.4 Чтение объекта-контейнера с использованием типа содержимого CDMI

9.4.1 Обзор

Для чтения всех полей существующего объекта-контейнера следует выпол-нить запрос:

GET <root URI>/<ContainerName>/<TheContainerName>/

Для чтения одного или нескольких определенных полей существующего объ-екта-контейнера следует выполнить один из следующих запросов:

GET<root URI>/<ContainerName>/<TheContainerName>/?<fieldname>;<fieldname>;...

GET <root URI>/<ContainerName>/<TheContainerName>/?children:<range>;...

GET <root URI>/<ContainerName>/<TheContainerName>/?metadata:<prefix>;...

где:

– <root URI> путь к облаку CDMI;

– <ContainerName> неотрицательное число промежуточных объектов-контейнеров;

- <TheContainerName> имя контейнера для чтения;
- <fieldname> имя поля;
- <range> числовой диапазон в списке потомков;
- <prefix> строковый префикс, будут возвращены все метаданные, начинающиеся с данного префикса.

К контейнеру можно также обратиться как <root URI>/cdmi_objectid/<objectID>/.

9.4.2 Опции

Следующие опции описывают поддерживаемые операции при чтении су-ществующего объекта-контейнера:

- поддержка возможности чтения метаданных существующего объекта-контейнера обозначается наличием опции cdmi_read_metadata в контейнере;
- поддержка возможности перечисление потомков существующего объекта-контейнера обозначается наличием опции cdmi_list_children в контейнере;
- поддержка возможности перечисления диапазона потомков существую-щего объекта-контейнера обозначается наличием опции cdmi_list_children_range в контейнере.

9.4.3 Заголовки запроса

Заголовки HTTP запроса на чтение из объекта-контейнера CDMI с использо-ванием типа содержимого CDMI приведены в таблице 38.

Таблица 38 – Заголовки запроса – чтение из объекта-контейнера с использованием типа содержимого CDMI

Заголовок	Тип	Описание	Требование
Accept	Строка заголовка	"application/cdmi-container" или совместимое значение со-гласно 5.13.2	Опционально
X-CDMI-SpecificationVersion	Строка заголовка	Список версий, поддерживаемых клиентом, разделенных запятыми, например, "1.0.2, 1.5, 2.0"	Обязательно

9.4.4 Тело сообщения-запроса

В запросе должно отсутствовать тело.

9.4.5 Заголовки ответа

Заголовки HTTP ответа для операции чтения объекта-контейнера CDMI с ис-пользованием типа содержимого CDMI приведены в таблице 39.

Таблица 39 – Заголовки ответа – чтение из объекта-контейнера с использованием типа содержимого CDMI

Заголовок	Тип	Описание	Требование
X-CDMI-SpecificationVersion	Строка заголовка	Сервер должен вернуть наибольший номер версии, под-держиваемой и сервером, и клиентом, например, "1.0.2". Если сервер не поддерживает ни одной версии, поддержи-ваемой клиентом, сервер должен вернуть код состояния 400 Bad Request	Обязательно
Content-Type	Строка заголовка	"application/cdmi-container"	Обязательно
Location	Строка заголовка	Сервер должен вернуть URI, на который осуществляется переадресация, если объект является ссылкой	Условно

9.4.6 Тело сообщения-ответа

Поля тела сообщения-ответа для операции чтения объекта-контейнера CDMI с использованием типа содержимого CDMI приведены в таблице 40.

Таблица 40 – Тело сообщения-ответа – чтение объекта-контейнера с использованием типа содержимого CDMI (лист 1 из 2)

Имя поля	Тип	Описание	Требование
objectType	Строка JSON	"application/cdmi-container"	Обязательно

Окончание таблицы 40

Имя поля	Тип	Описание	Требование
objectID	Строка JSON	ID объекта	Обязательно
objectName	Строка JSON	Имя объекта Для объектов в контейнере следует вернуть поле objectName. Для объектов не в контейнере (доступных только по ID), поле objectName не существует и не должно передаваться в ответе	Условно
parentURI	Строка JSON	URI родительского объекта Для объектов в контейнере следует вернуть поле parentURI. Для объектов не в контейнере (доступных только по ID), поле parentURI не существует и не должно передаваться в ответе. Добавление objectName к parentURI должно всегда возвращать правильный URI объекта.	Условно
parentID	Строка JSON	ID объекта родительского контейнера Для объектов в контейнере следует вернуть поле parentID. Для объектов не в контейнере (доступных только по ID), поле parentID не существует и не должно передаваться в ответе	Условно
domainURI	Строка JSON	URI домена-владельца	Обязательно
capabilitiesURI	Строка JSON	URI опций объекта	Обязательно
completionStatus	Строка JSON	Строка, указывающая, находится ли объект в процессе создания, а после завершения операции – был ли он создан успешно или с ошибкой. Значение должно быть строкой "Processing", "Complete" или строкой, начинающейся с "Error"	Обязательно
percent-Complete	Строка JSON	Если значение completionStatus равно "Processing", данное поле, при наличии, должно показывать процент выполнения операции создания объекта, числовым значением от 0 до 100. Если значение completionStatus равно "Complete", данное поле, при наличии, должно иметь значение "100". Если значение completionStatus начинается с "Error", данное поле, при наличии, может содержать любое целое число от 0 до 100	Опционально
metadata	Объект JSON	Метаданные объекта-контейнера. Могут включать пользовательские метаданные и метаданные системы данных, указанные в теле запроса на создание (поле metadata) вместе с метаданными системы хранения, создаваемыми облачным хранилищем. Подробнее о метаданных см. гл. 16	Обязательно
exports	Объект JSON	Структура каждого протокола, возможного для данного контейнера (см. гл. 13)	Опционально[a]
snapshots	Массив JSON	Массив URI снимков состояния объекта-контейнера	Опционально[a]
childrenrange	Строка JSON	Объекты-потомки контейнера в виде диапазона. При запросе диапазона потомков это поле показывает объекты-потомки в форме диапазона.	Обязательно
children	Массив JSON	Имена дочерних объектов контейнера. Имена дочерних объектов не должны содержать зарезервированных символов (см. RFC 3986), например, "%" в имени будет представлен как "%25". У дочерних объектов, являющихся контейнерами, имя должно заканчиваться символом "/". У дочерних объектов, являющихся ссылками, имя должно заканчиваться символом "?"	Обязательно
[a]Возвращается только при наличии			

Если в запросе GET указаны отдельные поля, в ответе должны быть возвращены только эти поля. Опциональные запрошенные поля, отсутствующие в объекте, опускаются в ответе.

9.4.7 Статус запроса

Таблица 41 описывает коды состояний HTTP, возникающих при чтении из кон-тейнера с использованием данных типа CDMI.

Таблица 41 – Коды состояний HTTP – чтение объекта-контейнера с использованием типа содержимого CDMI

Статус HTTP	Описание
200 OK	Метаданные объекта-контейнера включены в сообщение-ответ.
302 Found	URI ссылается на другой URI.
400 Bad Request	Запрос содержит неверные параметры или имена полей
401 Unauthorized	Неверные данные аутентификации/авторизации.
403 Forbidden	Клиент не обладает правами для выполнения данного запроса.
404 Not Found	Ресурс не найден по указанному URI.
406 Not Acceptable	Сервер не может предоставить объект, типизованный как обозначено в заголовке Accept.

9.4.8 Примеры

Примеры
1 Применение GET к URI объекта-контейнера для чтения всех его полей:
GET /MyContainer/HTTP/1.1
Host: cloud.example.com
Accept: application/cdmi-container
X-CDMI-Specification-Version: 1.0.2
Будет получен следующий ответ.
HTTP/1.1 200 OK
Content-Type: application/cdmi-container
X-CDMI-Specification-Version: 1.0.2
{
«objectType» : «application/cdmi-container»,
«objectID» : «0000706D0010B84FAD185C425D8B537E»,
«objectName» : «MyContainer/»,
«parentURI» : «/»,
«parentID» : «00007E7F0010128E42D87EE34F5A6560»,
«domainURI» : «/cdmi_domains/MyDomain/»,
«capabilitiesURI» : «/cdmi_capabilities/container/»,
«completionStatus» : «Complete»,
«metadata» : {

},
«exports» : {
«OCCI/iSCSI»: {
«identifier»: «00007E7F00104BE66AB53A9572F9F51E»,
«permissions»: [
«http://example.com/compute/0/»,
«http://example.com/compute/1/»
]
},
«Network/NFSv4» : {
«identifier» : «/users»,
«permissions» : «domain»
}
},
«childrenrange» : «0-4»,
«children» : [
«red»,
«green»,
«yellow»,

«orange/»,
«purple/»
]
}

2 Применение GET к URI объекта-контейнера для чтения parentURI и списка потомков:
GET /MyContainer/?parentURI;children HTTP/1.1
Host: cloud.example.com
Accept: application/cdmi-container
X-CDMI-Specification-Version: 1.0.2
Будет получен следующий ответ.
HTTP/1.1 200 OK
Content-Type: application/cdmi-container
X-CDMI-Specification-Version: 1.0.2
{
«parentURI» : «/»,
«children» : [
«red»,
«green»,
«yellow»,
«orange/»,
«purple/»
]
}

3 Применение GET к URI объекта-контейнера для чтения по-томков 0..2 и диапазона потомуов контейнера:
GET /MyContainer/?childrenrange;children:0-2 HTTP/1.1
Host: cloud.example.com
Accept: application/cdmi-container
X-CDMI-Specification-Version: 1.0.2
Будет получен следующий ответ.
HTTP/1.1 200 OK
Content-Type: application/cdmi-container
X-CDMI-Specification-Version: 1.0.2
{
«childrenrange» : «0-2»,
«children» : [
«red»,
«green»,
«yellow»
]
}

9.5 Изменение объекта-контейнера с использованием типа содержимого CDMI

9.5.1 Обзор

Для изменения некоторых или всех полей существующего объекта-контейнера следует выполнить запрос:

PUT <root URI>/<ContainerName>/<TheContainerName>/

Для добавления, обновления или удаления определенных элементов мета-данных существующего объекта-контейнера следует выполнить запрос:

PUT <root URI>/<ContainerName>/<TheContainerName>/?metadata: <metadataname>;...

где:

- <root URI> путь к облаку CDMI;
- <ContainerName> неотрицательное количество промежуточных контейнеров;
- <TheContainerName> имя изменяемого контейнера.

Объект-контейнер также доступен как <root URI>/cdmi_objectid/<objectID>/. Об-новление объекта не должно изменять его ID.

9.5.2 Отсроченное завершение снимка состояния

Если запрашивается создание снимка состояния (включением поля snap-shot в запрос, см. гл. 14), сервер может вернуть код состояния 202 Accepted. Этот ответ предполагает, что:

• Сервер удостоверяет, что пройдены определенные проверки:

– пользователь авторизован для создания снимка,

– пользователь авторизован для чтения объекта-контейнера,

– достаточно места для создания снимка или, по крайней мере, достаточно места для создания URI сообщения об ошибке.

• Клиент может не получить немедленного доступа к снимку из-за задержек, связанных с использованием в реализации целостности в конечном итоге.

Клиент выполняет запросы GET к URI снимка для отслеживания состояния операции. В частности, сервер возвращает два поля в теле сообщения-ответа:

– текстовое поле completionStatus, содержащее «Processing», «Complete», или сообщение об ошибке, начинающееся с «Error»;

– опциональное поле percentComplete, представляющее степень выполнения принятого запроса PUT числом от 0 до 100. Запрос GET не возвращает значе-ния объекта, если completionStatus отлично от «Complete».

Если операция создания снимка завершается ошибкой, создается URI снимка с полем completionStatus, содержащим сообщение об ошибке. Удаление этого URI после обработки ошибки возложено на клиента.

9.5.3 Заголовки запроса

Следующие опции описывают поддерживаемые операции при изменении существующего объекта-контейнера:

– поддержка возможности изменения метаданных существующего объекта-контейнера обозначается наличием опции cdmi_modify_metadata в контейнере;

– поддержка возможности сохранения снимка состояния содержимого су-ществующего объекта-контейнера обозначается наличием опции cdmi_snapshot ca-pability в контейнере;

– поддержка возможности добавления экспортируемого протокола к суще-ствующему объекту-контейнеру обозначается наличием опции cdmi_export_<protocol> в контейнере.

9.5.4 Заголовки запроса

Заголовки запроса HTTP для изменения объекта-контейнера CDMI с использованием типа содержимого CDMI перечислены в таблице 42.

Таблица 42 – Заголовки запроса – изменение объекта-контейнера с использованием типа содержимого CDMI

Заголовок	Тип	Описание	Требование
Content-Type	Строка заголовка	"application/cdmi-container"	Обязательно
X-CDMI-SpecificationVersion	Строка заголовка	Список версий, поддерживаемых клиентом, разделенных запятыми, например "1.0.2, 1.5, 2.0"	Обязательно

9.5.5 Тело сообщения-запроса

Поля тела запроса операции изменения объекта-контейнера с использова-нием типа содержимого CDMI перечислены в таблице 43.

Таблица 43 – Тело сообщения-запроса – изменение объекта-контейнера с использованием типа содержимого CDMI

Имя поля	Тип	Описание	Требование
metadata	Объект JSON	Метаданные для объекта-контейнера. Если присутствуют, указанные метаданные замещают существующие метаданные объекта. Если URI указывает на конкретные элементы метаданных, остальные элементы не должны меняться. Подробнее о метаданных см. раздел 16.	Опционально

Окончание таблицы 43

Имя поля	Тип	Описание	Требование
domainURI	Строка JSON	URI домена-владельца В случае отличия от родительского домена, у пользователя должны быть права cross_domain (см. cdmi_member_privileges в таблице 64). Если не указано, следует использовать родительский домен.	Опционально
snapshot	Строка JSON	Имя создаваемого снимка состояния. Это не URL, а конечный компонент абсолютного URL, где будет находиться снимок после успешного завершения операции создания снимка состояния. Если снимок добавляется или изменяется, операция PUT возвращает результат лишь после того как снимок добавлен в список снимков. После создания к снимкам можно обращаться как к объектам-потомкам контейнера cdmi_snapshots, вложенного в контейнер, чье состояние снимается. При создании снимка с тем же именем, что у уже имеющегося снимка, существующий снимок будет заменен.	Опционально
deserialize	Строка JSON	URI сериализованного объекта-контейнера CDMI, который должен быть десериализован для изменения существующего объекта-контейнера. ID сериализованного объекта-контейнера должен совпадать с ID объекта-назначения. Если сериализованный объект-контейнер не имеет потомков, изменение применяется только к объекту-контейнеру, а любые имеющиеся потомки остаются как есть. Если сериализованный объект имеет потомков, тогда операции создания, добавления, изменения и удаления рекурсивно применяются к каждому потомку, в зависимости от различия между представленным сериализованным состоянием и текущим состоянием потомков.	Опционально[a]
copy	Строка JSON	URI объекта-контейнера CDMI, который должен быть скопирован в имеющийся объект-контейнер. Будет скопировано лишь непосредственное содержимое объекта, но не его потомков.	Опционально[a]
deserialize-value	Строка JSON	Сериализованный объект-контейнер (см. гл. 15), кодированный с помощью base 64 как описано в RFC 4648. ID сериализованного объекта-контейнера должно соответствовать ID контейнера-цели. В противном случае сервер должен вернуть состояние 400 Bad Request. Если сериализованный объект-контейнер не имеет потомков, изменение применяется только к объекту-контейнеру, а любые имеющиеся потомки остаются как есть. Если сериализованный объект имеет потомков, тогда операции создания, добавления, изменения и удаления рекурсивно применяются к каждому потомку, в зависимости от различия между представленным сериализованным состоянием и текущим состоянием потомков.	Опционально[a]
exports	JSON Object	Структура для каждого протокола, разрешенного для данного объекта-контейнера (см. гл. 13). Если экспортированный протокол добавляется или изменяется, операция PUT возвращается только после завершения операции экспорта.	Опционально

[a] Лишь одно из этих полей должно быть указано в любой из операций. За исключением поля value, данные поля не должны сохраняться.

9.5.6 Заголовок ответа

Заголовок ответа HTTP на изменение объекта-контейнера CDMI с использо-ванием типа содержимого CDMI приведен в таблице 44.

Таблица 44 – Заголовок ответа – изменение объекта-контейнера с использованием типа содержимого CDMI

Заголовок	Тип	Описание	Требование
Location	Строка заголовка	Сервер должен вернуть URI, на который происходит переадресация, если объект является ссылкой.	Условно

9.5.7 Тело сообщения-ответа

Сообщение-ответ может содержать тело, соответствующее RFC 2616.

9.5.8 Статус запроса

В таблица 45 приведены коды состояний HTTP, возникающих при изменении объекта-контейнера с использованием типа содержимого CDMI.

Таблица 45 – Коды состояний HTTP – изменение объекта-контейнера с использованием типа содержимого CDMI (лист 1 из 2)

Статус HTTP	Описание
204 No Content	Операция завершилась успешно, никакие данные не возвращены.
202 Accepted	Контейнер или снимок состояния (вложенный контейнер) находится в процессе создания. Клиент CDMI должен отслеживать поля completionStatus и percentComplete для определения текущего статуса операции.
302 Found	URI ссылается на другой URI.
400 Bad Request	Запрос содержит неверные параметры или имена полей
401 Unauthorized	Неверные данные аутентификации/авторизации.
403 Forbidden	Клиент не обладает правами для выполнения данного запроса.
404 Not Found	Ресурс не найден по указанному URI.
409 Conflict	Операция конфликтует с блокировкой не-CDMI протокола доступа или может вызвать ошибку передачи на сервер.

9.5.9 Примеры

Примеры

1 Применение PUT к URI объекта-контейнера для задания новых значений полей:

```
PUT /MyContainer/ HTTP/1.1
Host: cloud.example.com
Content-Type: application/cdmi-container
X-CDMI-Specification-Version: 1.0.2
{
«metadata» : {

} ,
«exports» : {
«OCCI/iSCSI»: {
«identifier»: «00007E7F00104BE66AB53A9572F9F51E»,
«permissions»: [
«http://example.com/compute/0/»,
«http://example.com/compute/1/»
]
},
«Network/NFSv4» : {
«identifier» : «/users»,
«permissions» : «domain»
}
}432        }
```

Будет получен следующий ответ.

```
HTTP/1.1 204 No Content
```

2 Применение PUT к URI объекта-контейнера для установки нового значения экспортируемого протокола:

```
PUT /MyContainer/?exports HTTP/1.1
Host: cloud.example.com
Content-Type: application/cdmi-container
X-CDMI-Specification-Version: 1.0.2
{
«exports» : {
«OCCI/iSCSI» : {
```

«identifier» : «0000706D0010B84FAD185C425D8B537E»,
«permissions» : «00007E7F00104EB781F900791C70106C»
} ,
«Network/NFSv4» : {
«identifier» : «/users»,
«permissions» : «domain»
}
}
}
Будет получен следующий ответ.
HTTP/1.1 204 No Content

9.6 Удаление объекта-контейнера с использованием типа содержимого CDMI

9.6.1 Обзор

Для удаления существующего объекта-контейнера, включая все потомки и снимки состояния, следует выполнить запрос:

DELETE <root URI>/<ContainerName>/<TheContainerName>/

где:

- <root URI> путь к облаку CDMI;
- <ContainerName> неотрицательное число промежуточных контейнеров;
- <TheContainerName> имя уничтожаемого объекта-контейнера.

К объекту можно также обратиться как <root URI>/cdmi_objectid/<objectID>/.

9.6.2 Опции

Следующие опции описывают поддерживаемые операции при удалении существующего объекта-контейнера:

- поддержка возможности удаления существующего объекта-контейнера обозначается наличием опции cdmi_delete_container в контейнере.

9.6.3 Заголовок запроса

Заголовок запроса HTTP для удаления объекта-контейнера CDMI с использованием типа содержимого CDMI приведен в таблице 46.

Таблица 46 – Заголовок запроса – удаление объекта-контейнера с использованием типа содержимого CDMI

Заголовок	Тип	Описание	Требование
X-CDMI-SpecificationVersion	Строка заголовка	Список версий, поддерживаемых клиентом, разделенных запятыми, например, "1.0.2, 1.5, 2.0"	Обязательно

9.6.4 Тело сообщения-запроса

Сообщение-запрос может содержать тело, соответствующее RFC 2616.

9.6.5 Заголовки ответа

Заголовки ответа могут быть указаны в соответствии с RFC 2616.

9.6.6 Тело сообщения-ответа

Сообщение-ответ может содержать тело, соответствующее RFC 2616.

9.6.7 Статус запроса

В таблице 47 приведены коды состояний HTTP, которые могут возникнуть при удалении объекта-контейнера с использованием типа содержимого CDMI.

Таблица 47 – Коды состояний HTTP – удаление объекта-контейнера с использованием типа содержимого CDMI

Статус HTTP	Описание
204 No Content	Объект-контейнер успешно удален.
400 Bad Request	Запрос содержит неверные параметры или имена полей
401 Unauthorized	Неверные данные аутентификации/авторизации.

Окончание таблицы 47

Статус HTTP	Описание
403 Forbidden	Клиент не обладает правами для выполнения данного запроса.
404 Not Found	Ресурс не найден по указанному URI.
409 Conflict	Объект-контейнер может не быть удален.

9.6.8 Пример

Пример – Применение DELETE к URI объекта-контейнера:
DELETE /MyContainer/ HTTP/1.1
Host: cloud.example.com
X-CDMI-Specification-Version: 1.0.2
Будет получен следующий ответ.
HTTP/1.1 204 No Content

9.7 Удаление объекта-контейнера с использованием типа содержимого, отличного от CDMI

9.7.1 Обзор

Для удаления существующего объекта-контейнера, включая всех потомков и снимки состояния, следует выполнить запрос:

DELETE <root URI>/<ContainerName>/<TheContainerName>/

где:

- <root URI> путь к облаку CDMI;
- <ContainerName> неотрицательное количество промежуточных контейнеров;
- <TheContainerName> имя удаляемого объекта-контейнера.

К объекту можно также обратиться как <root URI>/cdmi_objectid/<objectID>/.

9.7.2 Опции

Следующие опции описывают поддерживаемые операции при удалении существующего объекта-контейнера:

- поддержка возможности удаления существующего объекта-контейнера обозначается наличием опции cdmi_delete_container в контейнере.

9.7.3 Заголовки запроса

Сообщение-запрос может содержать заголовки, соответствующие RFC 2616.

9.7.4 Тело сообщения-запроса

Сообщение-запрос может содержать тело, соответствующее RFC 2616.

9.7.5 Заголовки ответа

Сообщение-ответ может содержать заголовки, соответствующие RFC 2616.

9.7.6 Тело сообщения-ответа

Сообщение-ответ может содержать тело, соответствующее RFC 2616.

9.7.7 Статус запроса

В таблице 48 приведены коды состояний HTTP, которые могут возникнуть при удалении объекта-контейнера с использованием типа содержимого, отличного от CDMI.

Таблица 48 – Коды состояний HTTP – удаление объекта-контейнера с использованием типа содержимого, отличного от CDMI

Статус HTTP	Описание
204 No Content	Объект-контейнер успешно удален.
400 Bad Request	Запрос содержит неверные параметры или имена полей
401 Unauthorized	Неверные данные аутентификации/авторизации.
403 Forbidden	Клиент не обладает правами для выполнения данного запроса.
404 Not Found	Ресурс не найден по указанному URI.
409 Conflict	Объект-контейнер не может быть удален.

9.7.8 Пример

Пример – Применение DELETE к URI объекта-контейнера:
DELETE /MyContainer/ HTTP/1.1
Host: cloud.example.com
Будет получен следующий ответ.
HTTP/1.1 204 No Content

9.8 Создание (POST) нового объекта данных с использованием типа содержимого CDMI

9.8.1 Обзор

Для создания нового объекта данных в заданном контейнере, где имя объекта – идентификатор объекта, присвоенный сервером, следует выполнить запрос:

POST <root URI>/<ContainerName>/

Для создания нового объекта данных в случае, когда объект не принадлежит контейнеру и доступен только по ID (см. 5.8), следует выполнить запрос:

POST <root URI>/cdmi_objectid/

где:

– <root URI> путь к облаку CDMI;

– <ContainerName> неотрицательное количество существующих промежуточных контейнеров, имена которых разделены одиночными наклонными чертами (т.е., «/»).

После создания в «/cdmi_objectid/», к объекту данных можно обращаться как <root URI>/cdmi_objectid/<objectID>.

После создания в контейнере, объект данных доступен как потомок контейнера с именем, присвоенным сервером; к нему также можно обратиться как <root URI>/cdmi_objectid/<objectID>.

9.8.2 Отсроченное завершение создания

В ответ на запрос создания объекта данных сервер может вернуть код 202 Accepted, что указывает на то, что объект находится в процессе создания. Это полезно в случае длительных операций (например, копирования большого объема данных от URI источника). Такой ответ означает, что:

• Сервер должен вернуть заголовок Location, содержащий URI к создаваемому объекту вместе со статусом HTTP 202 Accepted.

• Статус 202 Accepted со стороны сервера удостоверяет, что были пройдены несколько проверок:

– пользователь авторизован для создания нового объекта данных;

– пользователь авторизован для чтения любого исходного объекта, который необходимо переместить, скопировать, сериализовать или десериализовать;

– достаточно места для создания объекта-контейнера или, по крайней мере, достаточно места для создания URI к сообщению об ошибке.

• Клиент может не иметь опции немедленно обратиться к созданному объекту, например, из-за задержек, вызванных использованием в реализации целостности в конечном итоге.

Клиент выполняет операции GET к URI для отслеживания процесса создания. В ответ сервер возвращает два поля в теле сообщения-ответа, которые описывают текущее состояние операции:

– обязательное текстовое поле completionStatus содержит «Processing», «Complete», либо строку сообщения об ошибке, начинающуюся с «Error»;

– опциональное поле percentComplete содержит процент выполнения принятого запроса POST (от 0 до 100).

GET не возвращает значение объекта, если completionStatus не равно «Complete». Если создание объекта завершается с ошибкой, создается URI, а поле completionStatus устанавливается равным сообщению об ошибке. Удаление URI после обработки ошибки возлагается на клиента.

9.8.3 Опции

Следующие опции описывают поддерживаемые операции при создании нового объекта данных по ID в «/cdmi_objectid/»:

– поддержка возможности создания нового объекта данных посредством этой операции обозначается наличием опции cdmi_post_dataobject_by_ID в системе;

– если объект, создаваемый в «/cdmi_objectid/», является ссылкой, поддержка этой операции обозначается присутствием опции cdmi_create_reference_by_ID в системе;

– если объект, создаваемый в «/cdmi_objectid/», является результатом операции копирования существующего объекта, поддержка этой операции обозначается присутствием опции cdmi_copy_dataobject_by_ID в системе;

– если объект, создаваемый в «/cdmi_objectid/», является результатом операции перемещения, поддержка этой операции обозначается присутствием опции cdmi_object_move_to_ID в системе;

– если объект, создаваемый в «/cdmi_objectid/», является результатом операции десериализации, поддержка этой операции обозначается присутствием опции cdmi_deserialize_dataobject_by_ID в системе;

– если объект, создаваемый в «/cdmi_objectid/», является результатом выполнения операции сериализации, поддержка этой операции обозначается присутствием следующих опций в системе:

—cdmi_serialize_dataobject_to_ID;

—cdmi_serialize_container_to_ID;

—cdmi_serialize_domain_to_ID, или

—cdmi_serialize_queue_to_ID.

Следующие опции описывают поддерживаемые операции при создании нового объекта данных по ID в контейнере:

– поддержка возможности создания нового объекта данных посредством этой операции обозначается наличием опций cdmi_post_dataobject в системе и cdmi_create_dataobject в контейнере;

– если объект, создаваемый таким образом, является ссылкой, поддержка этой операции обозначается присутствием опции cdmi_create_reference в родительском контейнере;

– если объект, создаваемый таким образом, является копией существующего объекта, поддержка этой операции обозначается присутствием опции cdmi_copy_dataobject в родительском контейнере;

– если объект, создаваемый таким образом, является результатом выполнения операции перемещения, поддержка этой операции обозначается присутствием опции cdmi_move_dataobject в родительском контейнере;

– если объект, создаваемый таким образом, является результатом выполнения операции десериализации, поддержка этой операции обозначается присутствием опции cdmi_deserialize_dataobject в родительском контейнере;

– если объект, создаваемый таким образом, является результатом выполнения операции сериализации, поддержка этой операции обозначается присутствием следующих опций в родительском контейнере: «cdmi_serialize_dataobject», «cdmi_serialize_container», «cdmi_serialize_domain» или «cdmi_serialize_queue».

9.8.4 Заголовки запроса

Заголовки запроса HTTP на создание нового объекта CDMI с использованием типа содержимого CDMI приведены в таблице 49.

Таблица 49 – Заголовки запроса – создание нового объекта данных с использованием типа содержимого CDMI

Заголовок	Тип	Описание	Требование
Accept	Строка заголовка	"application/cdmi-container" или совместимое значение согласно 5.13.2	Опционально
Content-Type	Строка заголовка	"application/cdmi-object"	Обязательно
X-CDMI-SpecificationVersion	Строка заголовка	Список версий, поддерживаемых клиентом, разделенных запятыми, например, "1.0.2, 1.5, 2.0"	Обязательно

9.8.5 Тело сообщения-запроса

Поля тела сообщения-запроса на создание нового объекта данных с использованием типа содержимого CDMI перечислены в таблице 50.

Таблица 50 – Тело сообщения-запроса – создание нового объекта данных с использованием типа содержимого CDMI

Имя поля	Тип	Описание	Требование
mimetype	Строка JSON	Тип MIME данных, содержащихся в поле value объекта данных Данное поле может добавляться при создании по значению или десериализации, копировании или перемещении объекта данных. Данное поле должно храниться как часть объекта. Если данное поле не указано, ему должно быть присвоено значение "text/ plain". Это поле не должно добавляться при создании ссылки. Перед сохранением значение типа MIME должно быть преобразовано в нижний регистр.	Опционально
metadata	Объект JSON	Метаданные объекта данных Если данное поле включено при десериализации, сериализации, копировании или перемещении объекта данных, его значение заменяет метаданные из URI источника. Если данное поле не включается при десериализации, сериализации, копировании или перемещении объекта данных, должны использоваться метаданные URI источника. Если данное поле включается при создании нового объекта данных по передаваемому значению, значение этого поля должно быть использовано как метаданные. Если данное поле не включается при создании нового объекта данных по значению, данному полю должен быть присвоен пустой объект JSON (т.е. "{}"). Данное поле не должно включаться при ссылке на объект данных.	Опционально
domainURI	Строка JSON	URI домена-владельца Если домен-владелец не совпадает с родительским доменом, пользователь должен иметь права cross_domain privilege (см. cdmi_member_privileges в таблице 64). IЕсли не указано иное, должен использоваться корневой домен "/ cdmi_domains/".	Опционально
deserialize	Строка JSON	URI сериализованного объекта данных CDMI, который должен быть десериализован при создании нового объекта данных	Опционально[a]
serialize	Строка JSON	URI объекта CDMI, который должен быть сериализован в новый объект данных	Опционально[a]
copy	Строка JSON	URI объекта данных или очереди CDMI, которые должны быть скопированы в новый объект данных	Опционально[a]
move	Строка JSON	URI объекта данных или очереди CDMI, которые должны быть скопированы в новый объект данных. Объект по URI источника должен быть удален после успешного завершения копирования.	Опционально[a]
reference	Строка JSON	URI объекта данных CDMI, который должен быть переадресован по ссылке. Если при создании ссылки предоставляются какие-либо другие поля, сервер должен вернуть сообщение об ошибке HTTP 400 Bad Request.	Опционально[a]
deserialize-value	Строка JSON	Объект данных, сериализованный как описано в гл. 15 и кодированный с использованием правил base 64 как описано в RFC 4648.	Опционально[a]
value-transfer-encoding	Массив JSON строк JSON	Кодировка, использованная при передаче объекта данных. Определены два значения кодировки. - "utf-8" указывает на то, что объект данных содержит корректную строку UTF-8, и должен передаваться как строка UTF-8 в поле value. - "base64" указывает на то, что объект данных содержит произвольную бинарную последовательность и должен передаваться в поле value как строка в кодировке base 64. Передача в поле value объекта данных значения, не являющегося корректной строкой в кодировке base 64, должно возвращать клиенту ошибку 400 Bad Request. Данное поле должно включаться лишь при создании объекта по значению. Если иное не указано клиентом, сервер должен установить значение поля valuetransferencoding равным "utf-8". Данное поле должно храниться как часть объекта.	Опционально

Окончание таблицы 50

Имя поля	Тип	Описание	Требование
value	Строка JSON	JSON-кодированное значение объекта данных Если данное поле не включено, ему должна быть присвоена пустая строка JSON (т.е., ""). Если поле valuetransferencoding указывает на кодировку UTF-8, значение должно быть строкой UTF-8, сформированной по правилам JSON как описано в RFC 4627. Если поле valuetransferencoding указывает на кодировку base 64, значение должно быть вначале кодировано в base 64 по правилам, описанным в RFC 4648.	Опционально[a]

[a] Лишь одно из этих полей должно быть указано в любой из операций. За исключением поля value, данные поля не должны сохраняться. Если указано более чем одно из этих полей, сервер должен вернуть сообщение об ошибке 400 Bad Request.

9.8.6 Заголовки ответа

Заголовки HTTP ответов на создание объекта с использованием типа содержимого CDMI указаны в таблице 51.

Таблица 51 – Заголовки ответа – создание нового объекта данных с использованием типа содержимого CDMI

Заголовок	Тип	Описание	Требование
Content-Type	Строка заголовка	"application/cdmi-object"	Обязательно
X-CDMI-SpecificationVersion	Строка заголовка	Сервер должен вернуть наибольший номер версии, поддерживаемой и сервером, и клиентом, например, "1.0.2". Если сервер не поддерживает ни одной версии, поддерживаемой клиентом, сервер должен вернуть код состояния 400 Bad Request.	Обязательно
Location	Строка заголовка	Уникальный URI нового объекта данных, присвоенный системой. В отсутствие имени файла от клиента, система должна присваивать URI в форме: <root URI>/<ContainerName>/<Object ID>.	Обязательно

9.8.7 Тело сообщения-ответа

Поля тела сообщения-ответа на создание нового объекта данных CDMI с использованием типа содержимого CDMI приведены в таблице 52.

Таблица 52 – Тело сообщения-ответа – создание объекта данных с использованием типа содержимого CDMI

Имя поля	Тип	Описание	Требование
objectType	Строка JSON	"application/cdmi-object"	Обязательно
objectID	Строка JSON	ID объекта	Обязательно
objectName	Строка JSON	Имя объекта Для объектов в контейнере должно быть возвращено поле objectName. Для объектов не в контейнере (доступных только по ID), поле objectName не существует и не должно возвращаться.	Условно
parentURI	Строка JSON	URI родительского объекта Для объектов в контейнере должно быть возвращено поле parentURI. Для объектов не в контейнере (доступных только по ID), поле parentURI не существует и не должно передаваться в ответе. Присоединение objectName к parentURI должно всегда давать корректный URI объекта.	Условно

Окончание таблицы 52

Имя поля	Тип	Описание	Требование
parentID	Строка JSON	ID родительского контейнера Для объектов в контейнере должно быть возвращено поле parentID. Для объектов не в контейнере (доступных только по ID), поле parentID не существует и не должно возвращаться.	Условно
domainURI	Строка JSON	URI домена-владельца	Обязательно
capabilitiesURI	Строка JSON	URI опций для объекта	Обязательно
completion-Status	Строка JSON	Строка, показывающая статус создания объекта данных, принимающая одно из следующих значений - "Processing" указывает на то, что объект находится в процессе создания. - "Completed" указывает на успешное создание объекта данных. - Строка, начинающаяся с "Error" указывает на то, что при создании объекта возникла ошибка.	Обязательно
percent-Complete	Строка JSON	Если значение completionStatus равно "Processing", данное поле, при наличии, должно показывать процент выполнения операции создания объекта как числовое значение от 0 до 100. Если значение completionStatus равно "Complete", данное поле, при наличии, должно иметь значение "100". Если значение completionStatus начинается с "Error", данное поле, при наличии, может содержать любое целое число от 0 до 100.	Опционально
mimetype	Строка JSON	Тип MIME значения объекта данных	Обязательно
metadata	JSON Object	Метаданные объекта данных. Данное поле включает любые пользовательские метаданные или метаданные системы данных, указанные в соответствующем поле тела запроса на создание, наряду с метаданными системы хранения, созданными облачным хранилищем. См. детальное описание метаданных в гл. 16.	Обязательно

9.8.8 Статус запроса

В таблице 53 приведены коды состояний HTTP, которые могут возникать при создании нового объекта с использованием типа содержимого CDMI.

Таблица 53 – Коды состояний HTTP – создание объекта данных с использованием типа содержимого CDMI

Статус HTTP	Описание
201 Created	Новый объект данных был создан.
202 Accepted	Объект данных в процессе создания. Клиент CDMI должен отслеживать поля completionStatus и percentComplete для определения текущего статуса операции.
400 Bad Request	Запрос содержит неверные параметры или имена полей
401 Unauthorized	Неверные данные аутентификации/авторизации.
403 Forbidden	Клиент не обладает правами для выполнения данного запроса.
404 Not Found	Ресурс не найден по указанному URI.
409 Conflict	Операция конфликтует с блокировкой не-CDMI протокола доступа или может вызвать ошибку передачи на сервер.

9.8.9 Примеры

Примеры

1 Применение POST к URI объекта-контейнера: добавление содержимого объекта данных:

POST /MyContainer/ HTTP/1.1

Host: cloud.example.com

Accept: application/cdmi-object

Content-Type: application/cdmi-object

X-CDMI-Specification-Version: 1.0.2

{
«mimetype» : «text/plain»,
«metadata» : {

},
«value» : «This is the Value of this Data Object»
}
Будет получен следующий ответ.
HTTP/1.1 201 Created
Content-Type: application/cdmi-object
X-CDMI-Specification-Version: 1.0.2
Location: http://cloud.example.com/MyContainer/0000706D0010B84FAD185C425D8B537E
{
«objectType» : «application/cdmi-object»,
«objectID» : «0000706D0010B84FAD185C425D8B537E»,
«objectName» : «0000706D0010B84FAD185C425D8B537E»,
«parentURI» : «/MyContainer/»,
«parentID» : «0000706D0010B84FAD185C425D8B537E»,
«domainURI» : «/cdmi_domains/MyDomain/»,
«capabilitiesURI» : «/cdmi_capabilities/dataobject/»,
«completionStatus» : «Complete»,
«mimetype» : «text/plain»,
«metadata» : {
}
}

2 Применение POST к URI ID объекта: добавление содержимого объекта данных:
POST /cdmi_objectid/ HTTP/1.1
Host: cloud.example.com
Accept: application/cdmi-object
Content-Type: application/cdmi-object
X-CDMI-Specification-Version: 1.0.2
{
«mimetype»: «text/plain»,
«domainURI»: «/cdmi_domains/MyDomain/»,
«value»: «This is the Value of this Data Object»
}
Будет получен следующий ответ.
HTTP/1.1 201 Created
Location: http://cloud.example.com/cdmi_objectid/0000706D0010B84FAD185C425D8B537E
Content-Type: application/cdmi-object
X-CDMI-Specification-Version: 1.0.2
{
«objectType»: «application/cdmi-object»,
«objectID»: «0000706D0010B84FAD185C425D8B537E»,
«domainURI»: «/cdmi_domains/MyDomain/»,
«capabilitiesURI»: «/cdmi_capabilities/dataobject/»,
«completionStatus»: «Complete»,
«mimetype»: «text/plain»,
«metadata»: {
«cdmi_acl»: [
{
«acetype»: «ALLOW»,
«identifier»: «OWNER@»,
«aceflags»: «NO_FLAGS»,
«acemask»: «ALL_PERMS»
}
]
}
}

9.9 Создание (POST) нового объекта данных с использованием типа содержимого, отличного от CDMI

9.9.1 Обзор

Для создания нового объекта данных в заданном контейнере, где имя объекта – идентификатор объекта, присвоенный сервером, следует выполнить запрос:

POST <root URI>/<ContainerName>/

где:

- <root URI> путь к облаку CDMI;
- <ContainerName> неотрицательное количество существующих промежуточных контейнеров, имена которых разделены одиночными наклонными чертами (т.е., «/»).

После создания в контейнере объект данных доступен как потомок контейнера с именем, присвоенным сервером; к нему также можно обратиться как <root URI>/cdmi_objectid/<objectID>.

9.9.2 Опции

Следующие опции описывают поддерживаемые операции при создании нового объекта данных:

- поддержка возможности создания новых объектов данных посредством данной операции обозначается наличием опций cdmi_post_dataobject и cdmi_create_dataobject в контейнере.

9.9.3 Заголовок запроса

Заголовок запроса HTTP на создание нового объекта CDMI с использованием типа содержимого, отличного от CDMI, приведен в таблице 54.

Таблица 54 – Заголовок запроса – создание нового объекта данных с использованием типа содержимого, отличного от CDMI

Заголовок	Тип	Описание	Требование
Content-Type	Строка заголовка	Тип содержимого, которое необходимо сохранить в объект данных. Указанное здесь значение должно быть преобразовано в нижний регистр и сохранено поле mimetype объекта данных CDMI. Если тип данных содержит строковый параметр (см. RFC 2246), равный "utf-8", (например, ";charset=utf-8"), поле valuetransferencoding объекта данных CDMI должно быть установлено в "utf-8". В противном случае поле valuetransferencoding объекта данных CDMI должно быть установлено в "base64".	Обязательно

9.9.4 Тело сообщения-запроса

Тело сообщения-запроса на создание объекта данных содержит данные, которые необходимо сохранить в создаваемом объекте.

9.9.5 Заголовок ответа

Заголовки HTTP ответов на создание объекта с использованием типа содержимого, отличного от CDMI, указаны в таблице 55.

Таблица 55– Заголовок ответа – создание объекта данных с использованием типа содержимого, отличного от CDMI

Заголовок	Тип	Описание	Требование
Location	Строка заголовка	Уникальный URI нового объекта данных, присвоенный системой. При отсутствии имени файла от клиента, система должна присваивать URI в форме <root URI>/<ContainerName>/<ObjectID>.	Обязательно

9.9.6 Тело сообщения-ответа

Сообщение-ответ может содержать тело, соответствующее RFC 2616.

9.9.7 Статус запроса

В таблице 56 приведены коды состояний HTTP, которые возникают при создании нового объекта данных с использованием содержимого, отличного от типа CDMI.

Таблица 56 – Коды состояний HTTP – создание нового объекта данных с использованием типа содержимого, отличного от CDMI

Статус HTTP	Описание
201 Created	Новый объект данных был создан.
400 Bad Request	Запрос содержит неверные параметры или имена полей
401 Unauthorized	Неверные данные аутентификации/авторизации.
403 Forbidden	Клиент не обладает правами для выполнения данного запроса.
404 Not Found	Ресурс не найден по указанному URI.

9.9.8 Примеры

Примеры
1 Применение POST к URI объекта-контейнера: добавьте данные в новый объект:
POST /MyContainer/ HTTP/1.1
Host: cloud.example.com
Content-Type: text/plain;charset=utf-8
<object contents>
Будет получен следующий ответ.
HTTP/1.1 201 Created
Location: http://cloud.example.com/MyContainer/0000706D0010B84FAD185C425D8B537E

2 Применение POST URI ID объекта: добавьте данные в новый объект:
POST /cdmi_objectid/ HTTP/1.1
Host: cloud.example.com
Content-Type: text/plain;charset=utf-8
<object contents>
Будет получен следующий ответ.
HTTP/1.1 201 Created
Location: http://cloud.example.com/cdmi_objectid/0000706D0010B84FAD185C425D8B537E

9.10 Создание (POST) нового объекта-очереди с использованием типа содержимого CDMI

9.10.1 Обзор

Для создания нового объекта-очереди (см. раздел 11) в определенном контейнере, где имя очереди – идентификатор объекта, присвоенный сервером, следует выполнить запрос:

POST <root URI>/<ContainerName>/

Для создания нового объекта-очереди, где объект не принадлежит контейнеру и доступен только по ID (см. 5.8), следует выполнить запрос:

POST <root URI>/cdmi_objectid/

Где:

- <root URI> путь к облаку CDMI;
- <ContainerName> неотрицательное количество существующих промежуточных контейнеров, имена которых разделены одиночными наклонными чертами (т.е., «/»).

В случае создания в «/cdmi_objectid/» к объекту-очереди можно обращаться как <root URI>/cdmi_objectid/<objectID>.

После создания в контейнере объект-очередь доступен как потомок контейнера с именем, присвоенным сервером; к нему также можно обратиться как <root URI>/cdmi_objectid/<objectID>.

9.10.2 Отсроченное завершение создания

В ответ на запрос создания объекта-очереди сервер может вернуть код 202 Accepted, что указывает на то, что объект находится в процессе создания. Это полезно в случае длительных операций (например, копирования большого числа элементов очереди из URI источника). Такой ответ означает, что:

- сервер должен вернуть заголовок Location, содержащий URI к создаваемому объекту вместе со статусом HTTP 202 Accepted;
- статус 202 Accepted со стороны сервера удостоверяет, что были пройдены несколько проверок:

—пользователь авторизован для создания нового объекта;

—пользователь авторизован для чтения любого исходного объекта, который необходимо переместить, скопировать, сериализовать или десериализовать;

—достаточно места для создания объекта или, по крайней мере, достаточно места для создания URI к сообщению об ошибке.

Клиент может не иметь опции немедленно обратиться к созданному объекту, например, из-за задержек, вызванных использованием в реализации связности в конечном итоге;

Клиент выполняет операции GET к URI для отслеживания процесса создания. В ответ сервер возвращает два поля в теле сообщения-ответа, которые описывают состояние выполнения операции:

- обязательное текстовое поле completionStatus содержит «Processing», «Complete», либо строку сообщения об ошибке, начинающуюся с «Error»;
- опциональное поле percentComplete содержит процент выполнения принятого запроса PUT (от 0 до 100).

GET не возвращает значение объекта, если completionStatus не равно «Complete». Если создание объекта завершается с ошибкой, создается URI, а поле completionStatus устанавливается равным сообщению об ошибке. Удаление URI после обработки ошибки возлагается на клиента.

9.10.3 Опции

Следующие опции описывают поддерживаемые операции при создании нового объекта-очереди по ID в «/cdmi_objectid/»:

- поддержка возможности создания нового объекта-очереди посредством данной операции обозначается наличием опции cdmi_post_queue_by_ID в системе;
- если объект, создаваемый в «/cdmi_objectid/», является ссылкой, поддержка этой операции обозначается присутствием опции cdmi_create_reference_by_ID в системе;
- если очередь, создаваемая в «/cdmi_objectid/», является копией существующего объекта-очереди, поддержка этой операции обозначается присутствием опции cdmi_copy_queue_by_ID» в системе;
- если очередь, создаваемая в «/cdmi_objectid/», является результатом выполнения операции перемещения, поддержка этой операции обозначается присутствием опции cdmi_object_move_to_ID в системе,
- если очередь, создаваемая в «/cdmi_objectid/», является результатом выполнения операции десериализации, поддержка этой операции обозначается присутствием опции cdmi_deserialize_queue_by_ID в системе.

Следующие опции описывают поддерживаемые операции при создании нового объекта-очереди по ID в контейнере:

- поддержка возможности создания нового объекта-очереди посредством данной операции обозначается наличием опций the cdmi_post_queue и cdmi_create_queue у контейнера;
- если объект, создаваемый в родительском контейнере, является ссылкой, поддержка этой операции обозначается присутствием опции cdmi_create_reference в родительском контейнере;
- если новая очередь является копией существующей очереди, поддержка этой операции обозначается присутствием опции cdmi_copy_queue в родительском контейнере;
- если новая очередь является результатом выполнения операции перемещения, поддержка этой операции обозначается присутствием опции cdmi_move_queue в родительском контейнере;
- если новая очередь является результатом выполнения операции десериализации, поддержка этой операции обозначается присутствием опции cdmi_deserialize_queue в родительском контейнере.

9.10.4 Заголовки запроса

Заголовки запроса HTTP на создание новой очереди CDMI с использованием типа содержимого CDMI приведены в таблице 57.

Таблица 57– Заголовки запроса – создание новой очереди с использованием типа содержимого CDMI

Заголовок	Тип	Описание	Требование
Accept	Строка заголовка	"application/cdmi-container" или совместимое значение согласно 5.13.2	Опционально
Content-Type	Строка заголовка	"application/cdmi-queue"	Обязательно
X-CDMI-Specification-Version	Строка заголовка	Список версий, поддерживаемых клиентом, разделенных запятыми, например "1.0.2, 1.5, 2.0"	Обязательно

9.10.5 Тело сообщения-запроса

Поля тела сообщения-запроса на создание новой очереди с использованием типа содержимого CDMI перечислены в таблице 58.

Таблица 58 – Тело сообщения-запроса – создание новой очереди с использованием типа содержимого CDMI

Имя поля	Тип	Описание	Требование
metadata	Объект JSON	Метаданные объекта-очереди Если данное поле включено при десериализации, сериализации, копировании или перемещении объекта-очереди, его значение заменяет метаданные из URI источника. Если данное поле не включается при десериализации, сериализации, копировании или перемещении объекта-очереди, должны использоваться метаданные URI источника. Если данное поле включается при создании нового объекта-очереди по передаваемому значению, значение этого поля должно быть использовано как метаданные. Если данное поле не включается при создании нового объекта-очереди по значению, данному полю должен быть присвоен пустой объект JSON (т.е. "{}"). Данное поле не должно включаться при ссылке на объект-очередь.	Опционально
domainURI	Строка JSON	URI домена-владельца Если домен-владелец не совпадает с родительским доменом, пользователь должен иметь права cross_domain privilege (см. cdmi_member_privileges в табл. 64). Если не указано иное, должен использоваться корневой домен "/cdmi_domains/".	Опционально
deserialize	Строка JSON	URI сериализованного объекта-очереди CDMI, который должен быть десериализован при создании нового объекта-очереди	Опционально[a]
copy	Строка JSON	URI объекта-очереди CDMI, который должен быть скопирован в новый объект-очередь	Опционально[a]
move	Строка JSON	URI объекта-очереди CDMI, который должен быть перенесен в URI, указанного в команде PUT. Объект по URI источника должен быть удален после успешного создания новой очереди.	Опционально[a]
reference	Строка JSON	URI объекта-очереди CDMI, который должен быть переадресован по ссылке. Если при создании ссылки предоставляются какие-либо другие поля, сервер должен вернуть сообщение об ошибке HTTP 400 Bad Request.	Опционально[a]
deserialize-value	Строка JSON	Объект-очередь, сериализованный как описано в гл. 15 и кодированный с использованием правил base 64 как описано в RFC 4648.	Опционально[a]

[a] Лишь одно из этих полей должно быть указано в любой из операций. За исключением поля value, данные поля не должны сохраняться. Если указано более чем одно из этих полей, сервер должен вернуть сообщение об ошибке 400 Bad Request.

9.10.6 Заголовки ответа

Заголовки HTTP ответов на создание объекта-очереди с использованием типа содержимого CDMI указаны в таблице 59.

Таблица 59 – Заголовки ответа – создание нового объекта-очереди CDMI с использованием типа содержимого CDMI

Заголовок	Тип	Описание	Требование
Content-Type	Строка заголовка	"application/cdmi-queue"	Обязательно
X-CDMI-Specification-Version	Строка заголовка	Сервер должен вернуть наибольший номер версии, поддерживаемой и сервером, и клиентом, например, "1.0.2". Если сервер не поддерживает ни одной версии, поддерживаемой клиентом, сервер должен вернуть код состояния 400 Bad Request.	Обязательно
Location	Строка заголовка	Уникальный URI нового объекта-очереди, присвоенный системой. В отсутствие имени файла от клиента, система должна присваивать URI в форме: <root URI>/<ContainerName>/<ObjectID>.	Обязательно

9.10.7 Тело сообщения-ответа

Поля тела сообщения-ответа на создание нового объекта-очереди CDMI с использованием типа содержимого CDMI приведены таблице 60.

Таблица 60 – Тело сообщения-ответа – создание объекта-очереди с использованием типа содержимого CDMI (лист 1 из 2)

Имя поля	Тип	Описание	Требование
objectType	Строка JSON	"application/cdmi-queue"	Обязательно
objectID	Строка JSON	ID объекта	Обязательно
objectName	Строка JSON	Имя объекта Для объектов в контейнере должно быть возвращено поле objectName. Для объектов не в контейнере (доступных только по ID), поле objectName не существует и не должно возвращаться.	Условно
parentURI	Строка JSON	URI родительского объекта Для объектов в контейнере должно быть возвращено поле parentURI. Для объектов не в контейнере (доступных только по ID), поле parentURI не существует и не должно передаваться в ответе. Присоединение objectName к parentURI должно всегда давать корректный URI объекта.	Условно
parentID	Строка JSON	ID родительского контейнера Для объектов в контейнере должно быть возвращено поле parentID. Для объектов не в контейнере (доступных только по ID), поле parentID не существует и не должно возвращаться.	Условно
domainURI	Строка JSON	URI домена-владельца	Обязательно
capabilities-URI	Строка JSON	URI опций для объекта	Обязательно
completion-Status	Строка JSON	Строка, указывающая, находится ли объект в процессе создания (а после создания – был ли он создан успешно или с ошибкой). Значение должно быть строкой "Processing", "Complete" или строкой, начинающейся с "Error".	Обязательно
percent-Complete	Строка JSON	Если значение completionStatus равно "Processing", данное поле, при наличии, должно показывать процент выполнения операции создания объекта как числовое значение от 0 до 100. Если значение completionStatus равно "Complete", данное поле, при наличии, должно иметь значение "100". Если значение completionStatus начинается с "Error", данное поле, при наличии, может содержать любое целое число от 0 до 100.	Опционально
metadata	Объект JSON	Метаданные объекта-очереди. Данное поле включает любые пользовательские метаданные или метаданные системы данных, указанные в соответствующем поле тела запроса на создание, наряду с метаданными системы хранения, созданными облачным хранилищем. См. детальное описание метаданных в разделе 16.	Обязательно
queueValues	Строка JSON	Диапазон индексов значений в очереди. Каждое значение в очереди должно иметь уникальный монотонно возрастающий целый положительный индекс, начиная с 0. Если очередь пустая, должна возвращаться пустая строка. Если очередь не пустая, должны быть возвращены минимальный и максимальный (в таком порядке) индекс значений в очереди, разделенные дефисом ("-")	Обязательно

9.10.8 Статус запроса

В таблице 61 приведены коды состояний HTTP, возникающие при создании нового объекта-очереди с помощью типа содержимого CDMI.

Таблица 61 – Коды состояний HTTP – создание нового объекта-очереди CDMI с использованием типа содержимого CDMI

Статус HTTP	Описание
201 Created	Новая очередь успешно создана.
202 Accepted	Объект-очередь в процессе создания. Клиент CDMI должен отслеживать поля completionStatus и percentComplete для определения текущего статуса операции.
400 Bad Request	Запрос содержит неверные параметры или имена полей
401 Unauthorized	Неверные данные аутентификации/авторизации.
403 Forbidden	Клиент не обладает правами для выполнения данного запроса.
404 Not Found	Ресурс не найден по указанному URI.
409 Conflict	Операция конфликтует с блокировкой не-CDMI протокола доступа или может вызвать ошибку передачи на сервер.

9.10.9 Пример

Пример – Применение POST к URI контейнера: добавление объекта-очереди:

```
POST /MyContainer/ HTTP/1.1
Host: cloud.example.com
Content-Type: application/cdmi-queue
Accept: application/cdmi-queue
X-CDMI-Specification-Version: 1.0.2
{
}
```

Будет получен следующий ответ.

```
HTTP/1.1 201 Created
Content-Type: application/cdmi-queue
X-CDMI-Specification-Version: 1.0.2
Location: http://cloud.example.com/MyContainer/0000706D0010B84FAD185C425D8B537E
{
«objectType» : «application/cdmi-queue»,
«objectID» : «0000706D0010B84FAD185C425D8B537E»,
«objectName» : «0000706D0010B84FAD185C425D8B537E»,
«parentURI» : «/MyContainer/»,
«parentID» : «0000706D0010B84FAD185C425D8B537E»,
«domainURI» : «/cdmi_domains/MyDomain/»,
«capabilitiesURI» : «/cdmi_capabilities/queue/»,
«completionStatus» : «Complete»,
«metadata» : {
},
«queueValues» : «»
}
```

10 Операции с ресурсами объекта-домена

10.1 Обзор

Объекты-домены отражают концепцию административного владения сохраненными данными в рамках системы хранения CDMI™. CDMI может включать иерархию доменов, которые предоставляют доступ к связанной с доменами информации в системе CDMI. Иерархия доменов представляет собой набор объектов CDMI, соответствующих доменам-родителям и доменам-потомкам, причем каждый домен соответствует логически организованной группе объектов, которые должны управляться совместно. В соответствии с облачной моделью хранения домен интегрирует информацию о принадлежащих к нему объектах и передает ее родительскому домену, упрощая операции учета и управления.

Реализация CDMI может опционально включать URI доменов в следующей форме:

http://example.com/cdmi_domains/parent_domain/child_domain/

Объекты-домены создаются в контейнере the cdmi_domains, находящемся в корневом URI облачного хранилища. Если опция cdmi_create_domain присутствует в URI некоторого домена, облачное хранилище поддерживает создание доменов-потомков по этому URI. Если облачное хранилище поддерживает домены, должен присутствовать контейнер cdmi_domains.

Если клиент включает или передает поля десериализации, не описанные в настоящем стандарте, эти поля должны передаваться как часть объекта.

10.1.1 Метаданные объекта-домена

Для каждого домена должны присутствовать следующие поля (см. таблицу 62).

Таблица 62 – Обязательные метаданные объекта-домена

Имя метаданных	Тип	Описание	Требование
cdmi_domain_enabled	Строка JSON	Показывает, поддерживается ли домен и определен ли он в момент создания. Принимает значения "true" или "false". Если домен не поддерживается, облачное хранилище не должно позволять никаких операций с любым URI, управляемым этим доменом. Если данный элемент метаданных отсутствует в момент создания домена, его значение должно быть выставлено в "false".	Обязательно
cdmi_domain_delete_reassign	Строка JSON	Указывает, каком домену следует переподчинить объекты данного домена при его удалении. Данный элемент метаданных должен присутствовать, чтобы операция удаления домена, содержащего объекты, была допустимой. Если данный элемент метаданных отсутствует либо не содержит корректного URI домена, отличного от удаляемого домена, попытка удалить домен должна возвращать код HTTP 409 Conflict.	Условно

10.1.2 Сводка объекта-домена

Сводки объекта-домена содержат обобщенные метрики об использовании домена и оплаты. В случае поддержки данной функций, каждый домен должен содержать контейнер «cdmi_domain_summary». Подобно любому контейнеру, вложенный контейнер сводки домена может содержать список контроля доступа (Access Control List, ACL) (см. 16.1), который ограничивает доступ к этой информации.

Внутри контейнера-сводки каждого домена содержится набор объектов данных сводки, которые создаются системой облачного хранения. Контейнеры «yearly», «monthly» и «daily» этих объектов данных содержат объекты сводки, относящиеся к каждому году, месяцу и дате, соответственно. Эти контейнеры организованы в следующие структуры:

http://example.com/cdmi_domains/domain/
http://example.com/cdmi_domains/domain/cdmi_domain_summary/
http://example.com/cdmi_domains/domain/cdmi_domain_summary/cumulative
http://example.com/cdmi_domains/domain/cdmi_domain_summary/daily/
http://example.com/cdmi_domains/domain/cdmi_domain_summary/daily/2009-07-01
http://example.com/cdmi_domains/domain/cdmi_domain_summary/daily/2009-07-02
http://example.com/cdmi_domains/domain/cdmi_domain_summary/daily/2009-07-03
ttp://example.com/cdmi_domains/domain/cdmi_domain_summary/monthly/
http://example.com/cdmi_domains/domain/cdmi_domain_summary/monthly/2009-07
http://example.com/cdmi_domains/domain/cdmi_domain_summary/monthly/2009-08
http://example.com/cdmi_domains/domain/cdmi_domain_summary/monthly/2009-10
http://example.com/cdmi_domains/domain/cdmi_domain_summary/yearly/
http://example.com/cdmi_domains/domain/cdmi_domain_summary/yearly/2009
http://example.com/cdmi_domains/domain/cdmi_domain_summary/yearly/2010

Объект сводки «cumulative» покрывает весь промежуток времени, от создания домена до момента доступа к сводке «cumulative». Каждый объект данных ежедневного, ежемесячного и ежегодного уровня содержит сводки информации о домене, относящейся к соответствующему временному промежутку, ограниченного временем создания домена и временем доступа к сводке.

Если промежуток времени начинается раньше, чем был создан домен, информация сводки включает время, начиная со времени создания домена и заканчиваясь запрошенной датой окончания промежутка.

Пример – Если домен был создан в полдень 4 июля 2009 г., ежедневная сводка «2009-07-04» содержит информацию до полуночи этого дня, ежемесячная сводка «2009-07» содержит информацию от полудня 4 июля 2009 г. до полуночи 31 июля, а ежегодная сводка «2009» содержит информацию от полудня 4 июля до 31 декабря.

Если временной промежуток начинается позже времени создания домена и заканчивается раньше момента доступа сводке, объект сводки содержит полную информацию за данный промежуток времени.

Пример – Если домен был создан 4 июля 2009 г., то 10 июля ежедневная сводка «2009-07-06» объекта содержит информацию за полный день.

Если временной промежуток заканчивается позже текущего времени доступа, объект сводки данных о домене содержит частичную информацию от начала указанного периода времени (или времени создания домена) до момента доступа.

Пример – Если домен создан 4 июля 2009 г., то в полдень 10 июля ежедневная сводка «2009-07-10» содержит информацию от начала суток до полудня.

Информация, приведенная в таблице 63, должна присутствовать в каждом объекте сводки о домене, в представлении JSON.

Таблица 63 – Содержание объекта сводки о домене

Имя метаданных	Тип	Описание	Требование
cdmi_domainURI	Строка JSON	Имя домена, для которого приводится сводка	Обязательно
cdmi_summary_start	Строка JSON	Время в формате ISO-8601, обозначающее начало временного промежутка, за который приведена сводка	Обязательно
cdmi_summary_end	Строка JSON	Время в формате ISO-8601, обозначающее окончание временного промежутка, за который приведена сводка	Обязательно
cdmi_summary_ objecthours	Строка JSON	Суммарное время существования каждого объекта, принадлежащего домену (за промежуток, для которого приведена сводка)	Опционально
cdmi_summary_ objectsmin	Строка JSON	Минимальное количество объектов, принадлежащих домену (за промежуток, для которого приведена сводка)	Опционально
cdmi_summary_ objectsmax	Строка JSON	Максимальное количество объектов, принадлежащих домену (за промежуток, для которого приведена сводка)	Опционально
cdmi_summary_ objectsaverage	Строка JSON	Среднее количество объектов, принадлежащих домену (за промежуток, для которого приведена сводка)	Опционально
cdmi_summary_puts	Строка JSON	Количество объектов, записанных в домен	Опционально
cdmi_summary_gets	Строка JSON	Количество объектов, прочитанных из домена	Опционально
cdmi_summary_bytehours	Строка JSON	Суммарное время существования каждого байта, принадлежащего домену (за промежуток, для которого приведена сводка)	Опционально
cdmi_summary_bytesmin	Строка JSON	Минимальное количество байтов, принадлежащих домену (за промежуток, для которого приведена сводка)	Опционально
cdmi_summary_bytesmax	Строка JSON	Максимальное количество байтов, принадлежащих домену (за промежуток, для которого приведена сводка)	Опционально
cdmi_summary_ bytesaverage	Строка JSON	Среднее количество байтов, принадлежащих домену (за промежуток, для которого приведена сводка)	Опционально
cdmi_summary_writes	Строка JSON	Количество байтов, записанных в домен	Опционально
cdmi_summary_reads	Строка JSON	Количество байтов, прочитанных из домена	Опционально

Окончание таблицы 63

Имя метаданных	Тип	Описание	Требование
cdmi_summary_charge	Строка JSON	Код валюты по ISO 4217 (см. ISO 4217:2008), который предшествует или следует за числовым значением (отделяясь пробелом), где числовое значение показывает стоимость услуг, связанных с данным доменом за временной промежуток, для которого приведена сводка	Опционально
cdmi_summary_kwhours	Строка JSON	Суммарное количество энергии (в киловатт-часах), затраченное доменом за сводный промежуток времени	Опционально
cdmi_summary_kwmin	Строка JSON	Минимальная скорость расходования энергии (в киловаттах), затраченной доменом за сводный промежуток времени	Опционально
cdmi_summary_kwmax	Строка JSON	Максимальная скорость расходования энергии (в киловаттах), затраченная доменом за сводный промежуток времени	Опционально
cdmi_summary_ kwaverage	Строка JSON	Средняя скорость расходования энергии (в киловаттах), затраченная доменом за сводный промежуток времени	Опционально

Пример – Пример ежедневной сводки о домене:

```
{
«cdmi_domainURI» : «/cdmi_domains/MyDomain/»,
«cdmi_summary_start» : «2009-12-10T00:00:00»,
«cdmi_summary_end» : «2009-12-10T23:59:59»,
«cdmi_summary_objecthours» : «382239734»,
«cdmi_summary_puts» : «234234»,
«cdmi_summary_gets» : «489432»,
«cdmi_summary_bytehours» : «334895798347»,
«cdmi_summary_writes» : «7218368343»,
«cdmi_summary_reads» : «11283974933»,
«cdmi_summary_charge» : «4289.23 USD»
}
```

Если указана стоимость услуг, она соответствует эксплуатационным расходам (не включая фиксированные тарифы) на выполненные операции и уже потраченные объемы хранения и передачи. Стоимость услуг указывается отдельно.

Сводка о домена может быть расширена производителями, включая дополнительные по сравнению с описанными в настоящем стандарте элементы метаданных или отчеты о домене. Имена дополнительных полей не должны начинаться с «cdmi_».

10.1.3 Пользователи объекта-домена

В окружении облачного хранилища, подобно тому, как программно создаются домены, с использованием таких интерфейсов могут заполняться учетные данные пользователей объекта-домена. Наличие доступа к управлению пользователями позволяет самостоятельную регистрацию, автоматическую подготовку к работе и другие продвинутые средства самообслуживания, либо напрямую через CDMI, либо через программные средства, интерфейс которых использует CDMI.

Опция членства в домене дает информацию (и позволяет описывать) о конечных пользователях и группах пользователей, которым разрешен доступ к домену через CDMI и другие протоколы доступа. Концепция членства в домене не предназначена для замены или вытеснения ACL (см. 16.1), но предоставляет единое унифицированное пространство для хранения информации о пользователях и правах доступа, которые используются ACL в контексте домена (см. описание модели в 10.1.4). Кроме того, это пространство также позволяет хранить схемы аутентификации для внешних провайдеров аутентификации, таких как LDAP и AD.

Если данная функциональность поддерживается реализацией, в каждом домене должен присутствовать контейнер с именем «cdmi_domain_members». Как другие домены, он может включать список контроля доступа (Access Control List, см. 16.1), который ограничивает доступ к этой информации.

В каждом таком контейнере присутствует набор объектов-пользователей, которые определяются через CDMI. Эти объекты организованы в следующую структуру:

http://example.com/cdmi_domains/domain/
http://example.com/cdmi_domains/domain/cdmi_domain_members/
http://example.com/cdmi_domains/domain/cdmi_domain_members/john_doe
http://example.com/cdmi_domains/domain/cdmi_domain_members/john_smith

Контейнер пользователей домена также может включать вложенные контейнеры, содержащие объекты данных. Эти объекты данных обрабатываются таким же образом, что и объекты данных в контейнере пользователей домена, и имя вложенных контейнеров не несет специального значения. Это позволяет определять различные отношения между правами безопасности различных групп и объектов и позволяет выделять группы с общими правами.

В таблице 64 приведены настройки домена, которые должны присутствовать для каждого объекта-пользователя домена.

Таблица 64 – Необходимые настройки для объектов-пользователей домена

Имя метаданных	Тип	Описание	Требование
cdmi_member_enabled	Строка JSON	Если истинно, то запросы данного пользователя домена допустимы. Если ложно, запросы данного пользователя домена должны возвращать код состояние HTTP 403 Forbidden.	Обязательно
cdmi_member_type	Строка JSON	Данное поле показывает тип записи пользователя; допустимые значения: "user", "group" и "delegation".	Обязательно
cdmi_member_name	Строка JSON	Данное поле содержит имя пользователя или группы, предоставленное клиентом. Обычно это полное имя пользователя.	Обязательно
cdmi_member_ credentials	Строка JSON	Данное поле содержит данные аутентификации. Они используются при сравнении данных, предоставленных клиентом при аутентификации. Если данное поле отсутствует, должны быть одно или несколько делегирований для разрешения данных аутентификации. Так как невозможно подключиться к системе от имени группы (а лишь от имени участника группы), записи типа "group" не должны обладать данными аутентификации.	Опционально
cdmi_member_principal	Строка JSON	Данное поле указывает, какому имени принципала (используемому в ACL) соответствует пользователь или группа. Если данное поле отсутствует, должны присутствовать одно или несколько делегирований для установки соответствия имен принципала.	Опционально
cdmi_member_privileges	Массив JSON строк JSON	Данное поле содержит список в нотации JSON, перечисляющий права пользователя или группы. Определены следующие права: - "administrator". Все проверки доступа ACL успешны. - "backup_operator". Все проверки доступа на чтение ACL успешны. - "cross_domain". Допустимы операции с указанием домена, отличного от домена родительского объекта. Если данное право не предоставляется записью пользователя или группы (возможно, вложенной), к которой принадлежит пользователь, все попытки изменить домен объекта на иной, чем родительский домен, должны быть неуспешными.	Обязательно
cdmi_member_groups	Массив JSON строк JSON	Данное поле содержит список в нотации JSON, перечисляющий группы, к которым принадлежит пользователь.	Опционально

В таблице 65 приведены настройки домена, которые должны присутствовать в каждом объекте делегирования пользователя домена.

Таблица 65 – Необходимые настройки объектов делегирования пользователя домена

Имя метаданных	Тип	Описание	Требование
cdmi_member_enabled	Строка JSON	Если истинно, запросы, связанные с данным пользователем домена допустимы. Если ложно, все запросы, выполненные данным членом домена, должны возвращать HTTP статус 403 Forbidden.	Обязательно
cdmi_member_type	Строка JSON	Данное поле показывает тип записи пользователя: "user" или "delegation".	Обязательно
cdmi_delegation_URI	Строка JSON	Данной поле содержит URI внешнего провайдера идентификации пользователей (такому как LDAP или Active Directory) или URI к контейнеру пользователей домена.Внешние делегирования представляются в форме ldap:// или ad://.	Обязательно

Примеры

1 Пример объекта пользователя домена:

```
{
«cdmi_member_enabled» : «true»,
«cdmi_member_type» : «user»,
«cdmi_member_name» : «John Doe»,
«cdmi_member_credentials» : «p+5/oX1cmExfOlrUxhX1lw==»,
«cdmi_member_groups» : [
«users»
],
«cdmi_member_principal» : «jdoe»,
«cdmi_privileges» : [
«administrator»,
«cross_domain»
]
}
```

2 Пример объекта делегирования пользователя домена:

```
{
«cdmi_member_enabled» : «true»,
«cdmi_member_type» : «delegation»,
«cdmi_delegation_URI» : «/cdmi_accounts/MyAccount/»,

128}
```

10.1.4 Использование домена в управлении доступом

При выполнении транзакции с объектом CDMI связанный объект-домен (т.е., объект-домен, указанный в поле domainURI) определяет контекст аутентификации. Личность пользователя и его права, представленные как часть транзакции, сравниваются со списком пользователей домена для определения, авторизован ли пользователь в домене, и установления имени принципала. Если имя принципала установлено, оно сравнивается с ACL объекта для определения допустимости транзакции.

При обработке пользователей домена в первую очередь обрабатываются делегирования (в произвольном порядке), а затем записи пользователей (в произвольном порядке). Если имеется хотя бы одна подходящая запись и не найдено ни одной записи, которая блокирует пользователя, он считается членом домена.

Если изначально создается вложенный домен, контейнер членства содержит одну запись пользователя, которая является делегированием, устанавливающей URI делегирования равным URI родительского домена.

10.1.5 Представления объекта-домена

Представления в данном разделе показаны с использованием нотации JSON. И клиенты, и серверы должны поддерживать представление JSON в кодировке UTF-8. В телах сообщений-запросов и ответов поля JSON могут указываться и возвращаться в любом порядке, за исключением (при наличии) полей объекта-домена childrenrange и children, которые указываются последними и в данном порядке.

10.2 Создание объекта-домена с использованием типа содержимого CDMI

10.2.1 Обзор

Для создания нового объекта-домена следует выполнить запрос:
PUT <root URI>/cdmi_domains/<DomainName>/<NewDomainName>/
Где:
- <root URI> путь к облаку CDMI;
- <DomainName> неотрицательное число уже существующих доменов;
- <NewDomainName> имя создаваемого домена.

После создания к домену можно обращаться как <root URI>/cdmi_objectid/<objectID>/.

10.2.2 Опции

Следующие опции описывают поддерживаемые операции при создании нового домена:
- поддержка возможности создания нового объекта-домена обозначается наличием опции cdmi_create_domain capability в родительском домене;
- если новый объект-домен является копией существующего объекта- домена, поддержка этой операции обозначается наличием опции cdmi_copy_domain в домене-источнике;
- если новый домен получается в результате операции десериализации, поддержка этой операции обозначается наличием опции cdmi_deserialize_domain в родительском домене.

10.2.3 Заголовки запроса

Заголовки HTTP запросов на создание объекта-домена CDMI с использованием типа содержимого CDMI приведены в таблице 66.

Таблица 66 – Заголовки запроса – создание объекта-домена с использованием типа содержимого CDMI

Заголовок	Тип	Описание	Требование
Accept	Строка заголовка	"application/cdmi-container" или совместимое значение согласно 5.13.2	Опционально
Content-Type	Строка заголовка	"application/cdmi-domain"	Обязательно
X-CDMI-SpecificationVersion	Строка заголовка	Список версий, поддерживаемых клиентом, разделенных запятыми, например "1.0.2, 1.5, 2.0"	Обязательно

10.2.4 Тело сообщения-запроса

Поля тела сообщения-запроса на создание объекта-домена с использованием типа содержимого CDMI приведены в таблице 67.

Таблица 67 – Тело сообщения-запроса – создание объекта-домена с использованием типа содержимого CDMI

Имя поля	Тип	Описание	Требование
metadata	Объект JSON	Метаданные объекта-домена Если данное поле включено при десериализации, сериализации, копировании или перемещении объекта-домена, то значение этого поля должно заменять метаданные URI источника. Если данное поле не включено при десериализации, сериализации, копировании или перемещении объекта-домена то следует использовать метаданные URI источника. Если данное поле включено при создании нового объекта по значению, это поле должно быть использовано как метаданные. Если данное поле не включено при создании нового домена по значению, этому полю следует поставить в соответствие пустой объект JSON (т.е., "{}").	Опционально
copy	Строка JSON	URI объекта-домена CDMI, который должен быть скопирован в новый объект-домен, включая всех потомков и списки пользователей исходного домена	Опционально[a]

Окончание таблицы 67

Имя поля	Тип	Описание	Требование
move	Строка JSON	URI существующего локального или удаленного объекта-домена CDMI (URI источника), который должен быть перемещен, включая все дочерние домены, в URI, указанный в команде PUT. Содержимое объекта-домена и всех поддоменов, включая ID объекта, должны при перемещении сохраняться. После успешного создания всех необходимых объектов, перемещаемый домен и вложенные домены по исходному URI должны быть удалены. Если недостаточно прав для чтения объектов по URI источника, записи объектов по URI нового объекта, удаления объектов по URI источника или одна из этих операций завершается с ошибкой, сервер должен вернуть код результата 400 Bad Request, причем исходный и конечный объекты должны сохраниться неизменными.	Опционально[a]
deserialize	Строка JSON	URI сериализованного объекта данных CDMI, который должен быть десериализован для создания нового домена, включая всех дочерних объектов исходного сериализованного объекта данных	Опционально[a]
deserialize-value	Строка JSON	Сериализованный объект-домен (см. гл. 15), кодированный с помощью base 64 как описано в RFC 4648.	Опционально[a]

[a] Лишь одно из этих полей должно быть указано в любой из операций. За исключением поля value, данные поля не должны сохраняться. Если указано более чем одно из этих полей, сервер должен вернуть сообщение об ошибке 400 Bad Request.

10.2.5 Заголовки ответа

Заголовки HTTP ответа на создание объекта-домена CDMI с использованием типа содержимого CDMI указаны в таблице 68.

Таблица 68 – Заголовки ответа – создание объекта-домена с использованием типа содержимого CDMI

Заголовок	Тип	Описание	Требование
Content-Type	Строка заголовка	"application/cdmi-domain"	Обязательно
X-CDMI-Specification-Version	Строка заголовка	Сервер должен вернуть наибольший номер версии, поддерживаемой и сервером, и клиентом, например, "1.0.2". Если сервер не поддерживает ни одной версии, поддерживаемой клиентом, сервер должен вернуть код состояния 400 Bad Request.	Обязательно

10.2.6 Тело сообщения-ответа

Поля тела сообщения-ответа на создание объекта-домена CDMI с использованием типа содержимого CDMI приведены в таблице 69.

Таблица 69 – Тело сообщения-ответа – создание объекта-домена с использованием типа содержимого CDMI

Имя поля	Тип	Описание	Требование
objectType	Строка JSON	"application/cdmi-domain"	Обязательно
objectID	Строка JSON	ID домена	Обязательно
objectName	Строка JSON	Имя объекта	Обязательно
parentURI	Строка JSON	URI родительского объектаДобавление objectName к parentURI должно всегда давать корректный URI объекта.	Обязательно
parentID	Строка JSON	ID родительского объекта-контейнера	Обязательно
domainURI	Строка JSON	URI домена-владельца. Владелец объекта-домена – всегда сам домен.	Обязательно
capabilitiesURI	Строка JSON	URI опций объекта	Обязательно

Окончание таблицы 69

Имя поля	Тип	Описание	Требование
metadata	JSON Object	Метаданные объекта-домена. Могут включать пользовательские метаданные и метаданные системы данных, указанные в теле запроса на создание (поле metadata) вместе с метаданными системы хранения, создаваемыми облачным хранилищем. Подробнее о метаданных см. гл. 16.	Обязательно
childrenrange	Строка JSON	Вложенные домены, представленные как диапазон. При запросе диапазона вложенных доменов это поле показывает потомков, выраженных диапазоном.	Обязательно
children	Массив JSON	Имена доменов, вложенных в данный домен. Дочерние контейнеры завершаются символом "/".	Обязательно

10.2.7 Статус запроса

В таблице 70 приведены коды состояний HTTP, возникающие при создании объекта-домена с использованием типа содержимого CDMI.

Таблица 70 – Коды состояний HTTP – создание объекта-домена с использованием типа содержимого CDMI

Статус HTTP	Описание
201 Created	Новый объект-домен успешно создан.
400 Bad Request	Запрос содержит неверные параметры или имена полей
401 Unauthorized	Неверные данные аутентификации/авторизации.
403 Forbidden	Клиент не обладает правами для выполнения данного запроса.
404 Not Found	Ресурс не найден по указанному URI.
409 Conflict	Домен с таким именем уже существует.

10.2.8 Пример

Пример – Применение PUT к URI домена: добавление имени домена и метаданных:
PUT /cdmi_domains/MyDomain/ HTTP/1.1
Host: cloud.example.com
Accept: application/cdmi-domain
Content-Type: application/cdmi-domain X-CDMI-Specification-Version: 1.0.2
{
«metadata» : {

}
}
Будет получен следующий ответ.
HTTP/1.1 201 Created
Content-Type: application/cdmi-domain X-CDMI-Specification-Version: 1.0.2
{
«objectType» : «application/cdmi-domain»,
«objectID» : «00007E7F00104BE66AB53A9572F9F51E»,
«objectName» : «MyDomain/»,
«parentURI» : «/cdmi_domains/»,
«parentID» : «00007E7F0010C058374D08B0AC7B3550»,
«domainURI» : «/cdmi_domains/MyDomain/»,
«capabilitiesURI» : «/cdmi_capabilities/domain/»,
«metadata» : {

},
«childrenrange» : «0-1»,
«children» : [
«cdmi_domain_summary/»,
«cdmi_domain_members/»
]
}

10.3 Чтение объекта-домена с использованием типа содержимого CDMI

10.3.1 Обзор

Для чтения всех полей существующего объекта-домена следует выполнить запрос:

GET <root URI>/cdmi_domains/<DomainName>/<TheDomainName>/

Для чтения одного или нескольких выбранных полей из существующего объекта-домена следует выполнить один из следущих запросов:

- GET <root URI>/cdmi_domains/<DomainName>/<TheDomainName>/?<fieldname>;<fieldname>;...
- GET <root URI>/cdmi_domains/<DomainName>/<TheDomainName>/?children:<range>;...
- GET <root URI>/cdmi_domains/<DomainName>/<TheDomainName>/?metadata:<prefix>;...

где:

- <root URI> путь к облаку CDMI;
- <DomainName> неотрицательное число родительских доменов;
- <TheDomainName> имя домена, из которого необходимо провести чтение;
- <fieldname> имя поля;
- <range> числовой диапазон в списке потомков;
- <prefix> соответствующий префикс, возвращающий все метаданные, начинающиеся с данного префикса..

К объекту можно также обратиться как <root URI>/cdmi_objectid/<objectID>/.

10.3.2 Опции

Следующие опции описывают поддерживаемые операции при чтении существующего домена:

- поддержка возможности чтения метаданных существующего объекта-домена обозначается наличием опции cdmi_read_metadata в домене.
- поддержка возможности перечисления потомков существующего объекта-домена обозначается наличием опции cdmi_list_children в домене.

10.3.3 Заголовки запроса

Заголовки HTTP запроса на чтение из объекта-домена CDMI с использованием типа содержимого CDMI приведены в таблице 71.

Таблица 71 – Заголовки запроса – чтение из объекта-домена с использованием типа содержимого CDMI

Заголовок	Тип	Описание	Требование
Accept	Строка заголовка	"application/cdmi-container" или совместимое значение согласно 5.13.2	Опционально
X-CDMI-SpecificationVersion	Строка заголовка	Список версий, поддерживаемых клиентом, разделенных запятыми, например, "1.0.2, 1.5, 2.0"	Обязательно

10.3.4 Тело сообщения-запроса

Тело сообщения должно отсутствовать в запросе.

10.3.5 Заголовки ответа

Заголовки HTTP ответа для чтения объекта-домена CDMI с использованием типа содержимого CDMI приведены в таблице 72.

Таблица 72 – Заголовки ответа – чтение из объекта-домена с использованием типа содержимого CDMI

Заголовок	Тип	Описание	Требование
X-CDMI-SpecificationVersion	Строка заголовка	Сервер должен вернуть наибольший номер версии, поддерживаемой и сервером, и клиентом, например, "1.0.2". Если сервер не поддерживает ни одной версии, поддерживаемой клиентом, сервер должен вернуть код состояния 400 Bad Request.	Обязательно
Content-Type	Строка заголовка	"application/cdmi-domain"	Обязательно
Location	Строка заголовка	Сервер должен вернуть URI, на который происходит переадресация, если объект является ссылкой.	Условно

10.3.6 Тело сообщения-ответа

Поля тела сообщения-ответа для чтения объекта-домена CDMI с использованием типа содержимого CDMI приведены в таблице 73.

Таблица 73 – Тело сообщения-ответа – чтение из объекта-домена с использованием типа содержимого CDMI

Имя поля	Тип	Описание	Требование
objectType	Строка JSON	"application/cdmi-domain"	Обязательно
objectID	Строка JSON	ID объекта	Обязательно
objectName	Строка JSON	Имя объекта	Обязательно
parentURI	Строка JSON	URI родительского объекта	Обязательно
parentID	Строка JSON	ID объекта родительского контейнера	Обязательно
domainURI	Строка JSON	URI домена-владельца. Владелец объекта-домена – всегда сам домен.	Обязательно
capabilitiesURI	Строка JSON	URI опций объекта	Обязательно
metadata	Объект JSON	Метаданные объекта-домена. Могут включать пользовательские метаданные и метаданные системы данных, указанные в теле запроса на создание (поле metadata) вместе с метаданными системы хранения, создаваемыми облачным хранилищем. Подробнее о метаданных см. гл. 16.	Обязательно
childrenrange	Строка JSON	Вложенные домены, представленные как диапазон. При запросе диапазона вложенных доменов это поле показывает потомков, выраженных диапазоном.	Обязательно
children	Массив JSON	Имена доменов, вложенных в данный домен. Дочерние контейнеры завершаются символом "/".	Обязательно

Если в запросе GET указаны отдельные поля, в ответе следует возвращать только эти поля. Опциональные запрошенные поля, отсутствующие в объекте, опускаются в ответе.

10.3.7 Статус запроса

В таблице 74 приведены коды состояний HTTP, которые могут возникать при чтении из объекта-домена с использованием типа содержимого CDMI.

Таблица 74 – Коды состояний HTTP – чтение из объекта-домена с использованием типа содержимого CDMI

Статус HTTP	Описание
200 OK	Содержимое объекта-домена возвращено в ответе.
302 Found	URI ссылается на другой URI.
400 Bad Request	Запрос содержит неверные параметры или имена полей
401 Unauthorized	Неверные данные аутентификации/авторизации.
403 Forbidden	Клиент не обладает правами для выполнения данного запроса.
404 Not Found	Ресурс не найден по указанному URI.
406 Not Acceptable	Сервер не способен предоставить объект с содержимым типа, указанного в заголовке Accept.

10.3.8 Примеры

Примеры

1 Применение GET к URI домена для чтения всех полей объекта:

GET /cdmi_domains/MyDomain/ HTTP/1.1

Host: cloud.example.com

Accept: application/cdmi-domain X-CDMI-Specification-Version: 1.0.2

Будет получен следующий ответ.

HTTP/1.1 200 OK

Content-Type: application/cdmi-domain X-CDMI-Specification-Version: 1.0.2

{
«objectType» : «application/cdmi-domain»,
«objectID» : «00007E7F00104BE66AB53A9572F9F51E»,
«objectName» : «MyDomain/»,
«parentURI» : «/cdmi_domains/»,
«parentID» : «00007E7F0010C058374D08B0AC7B3550»,
«domainURI» : «/cdmi_domains/MyDomain/»,
«capabilitiesURI» : «/cdmi_capabilities/domain/»,
«metadata» : {

},
«childrenrange» : «0-1»,
«children» : [
«cdmi_domain_summary/»,
«cdmi_domain_members/»
]
}

2 Применение GET к URI объекта для чтения полей parentURI и children домена:
GET /MyDomain/?parentURI;children HTTP/1.1
Host: cloud.example.com
Accept: application/cdmi-domain
X-CDMI-Specification-Version: 1.0.2
Будет получен следующий ответ.
HTTP/1.1 200 OK
Content-Type: application/cdmi-domain X-CDMI-Specification-Version: 1.0.2
{
«parentURI» : «/cdmi_domains/»,
«children» : [
«cdmi_domain_summary/»,
«cdmi_domain_members/»
]
}

3 Применение GET к URI домена для чтения первых двух потомков домена:
GET /MyDomain/?childrenrange;children:0-1 HTTP/1.1
Host: cloud.example.com
Accept: application/cdmi-domain X-CDMI-Specification-Version: 1.0.2
Будет получен следующий ответ.
HTTP/1.1 200 OK
Content-Type: application/cdmi-domain X-CDMI-Specification-Version: 1.0.2
{
«childrenrange» : «0-1»,
«children» : [
«cdmi_domain_summary/»,
«cdmi_domain_members/»
]
}

10.4 Изменение объекта-домена с использованием типа содержимого CDMI

10.4.1 Обзор

Для изменения некоторых или всех полей существующего объекта-домена следует выполнить запрос:

PUT <root URI>/cdmi_domains/<DomainName>/<TheDomainName>/

Для добавления, обновления или удаления определенных элементов метаданных существующего объекта-домена следует выполнить запрос:

PUT <root URI>/cdmi_domains/<DomainName>/<TheDomainName>/ ?metadata:<metadataname>,...

где:

– <root URI> путь к облаку CDMI;

– <DomainName> неотрицательное количество родительских доменов;
– <TheDomainName> имя изменяемого домена.

Объект-контейнер также доступен как <root URI>/cdmi_objectid/<objectID>/. Обновление объекта не должно изменять его ID.

10.4.2 Опции

Следующие опции описывают поддерживаемые операции при изменении существующего домена:
– поддержка возможности изменения метаданных существующего объекта-домена обозначается наличием опции cdmi_modify_metadata в домене.

10.4.3 Заголовки запроса

Заголовки запроса HTTP для изменения объекта-домена CDMI с использованием типа содержимого CDMI перечислены в таблице 75.

Таблица 75 – Заголовки запроса – изменение объекта-домена с использованием типа содержимого CDMI

Заголовок	Тип	Описание	Требование
Content-Type	Строка заголовка	"application/cdmi-domain"	Обязательно
X-CDMI-SpecificationVersion	Строка заголовка	Список версий, поддерживаемых клиентом, разделенных запятыми, например "1.0.2, 1.5, 2.0"	Обязательно

10.4.4 Тело сообщения-запроса

Поля тела запроса на изменение объекта-домена с использованием типа содержимого CDMI перечислены в таблице 76.

Таблица 76 – Тело сообщения-запроса – изменение объекта-домена с использованием типа содержимого CDMI (лист 1 из 2)

Имя поля	Тип	Описание	Требование
metadata	Объект JSON	Метаданные для объекта-домена. Если присутствуют, указанные метаданные замещают существующие метаданные объекта. Если URI указывает на отдельные элементы метаданных, остальные элементы не должны меняться. Подробнее о метаданных см. гл. 16.	Опционально
copy	Строка JSON	URI объекта-домена CDMI, который должен быть скопирован в существующий объект-домен. Должны быть скопированы лишь метаданные и списки пользователей домена, но не дочерних доменов.	Опционально[a]
deserialize	Строка JSON	URI сериализованного объекта данных CDMI, который должен быть десериализован для изменения существующего домена. ID сериализованного домена должен совпадать с ID изменяемого домена. Если сериализованный домен не имеет потомков, изменение применяется только к объекту-домену и не затрагивает существующих потомков. Если сериализованный объект-домен имеет потомков, то операции создания, изменения и удаления должны рекурсивно применяться к каждому потомку, в зависимости от различий между предоставленным сериализованным состоянием и текущим состоянием потомков.	Опционально[a]
deserializevalue	Строка JSON	Сериализованный объект-домен (см. гл. 15), кодированный с помощью base 64 как описано в RFC 4648. ID сериализованного домена должен совпадать с ID домена-назначения. Если сериализованный домен не имеет потомков, изменение применяется только к объекту-домену и не затрагивает существующих потомков. Если сериализованный лбъект-домен имеет потомков, то операции создания, изменения и удаления должны рекурсивно применяться к каждому потомку, в зависимости от различий между предоставленным сериализованным состоянием и текущим состоянием потомков.	Опционально[a]

[a] Лишь одно из этих полей должно быть указано в любой из операций. За исключением поля value, данные поля не должны сохраняться.

10.4.5 Заголовок ответа

Заголовок HTTP ответа на изменение объекта-домена CDMI с использованием типа содержимого CDMI указан в таблице 77.

Таблица 77 – Заголовок ответа – изменение объекта-домена с использованием типа содержимого CDMI

Заголовок	Тип	Описание	Требование
Location	Строка заголовка	Сервер должен вернуть URI, на который происходит переадресация, если объект является ссылкой.	Условно

10.4.6 Тело сообщения-ответа

Сообщение-ответ может содержать тело, соответствующее RFC 2616.

10.4.7 Статус запроса

В таблице 78 приведены коды состояний HTTP, которые возникают при изменении объекта-домена с использованием типа содержимого CDMI.

Таблица 78 – Коды состояний HTTP – изменение объекта-домена с использованием типа содержимого CDMI

Статус HTTP	Описание
204 No Content	Операция успешна, данные не возвращены.
302 Found	URI ссылается на другой URI.
400 Bad Request	Запрос содержит неверные параметры или имена полей
401 Unauthorized	Неверные данные аутентификации/авторизации.
403 Forbidden	Клиент не обладает правами для выполнения данного запроса.
404 Not Found	Ресурс не найден по указанному URI.
409 Conflict	Операция конфликтует с блокировкой не-CDMI протокола доступа или может вызвать ошибку передачи на сервер.

10.4.8 Пример

Пример – Применение PUT к URI домена для установки новых значений полей:
PUT /cdmi_domains/myDomain/ HTTP/1.1
Host: cloud.example.com
Content-Type: application/cdmi-domain X-CDMI-Specification-Version: 1.0.2
{
«metadata» : {
«test» : «value»
}
}
Будет получен следующий ответ.
HTTP/1.1 204 No Content

10.5 Удаление объекта-домена с использованием типа содержимого CDMI

10.5.1 Обзор

Для удаления существующего домена и переноса всех связанных с ним объектов в другой домен (для сохранения доступа) следует выполнить запрос:

DELETE <root URI>/cdmi_domains/<DomainName>/<TheDomainName>/

где:

- <root URI> путь к облаку CDMI.
- <DomainName> неотрицательное количество родительских доменов.
- <TheDomainName> имя удаляемого домена.

К объекту можно также обратиться как <root URI>/cdmi_objectid/<objectID>/.

10.5.2 Опции

Следующие опции описывают поддерживаемые операции при удалении существующего домена:

- поддержка возможности удаления существующего объекта-домена обозначается наличием опции cdmi_delete_domain в домене.

10.5.3 Заголовки запроса

Заголовки запроса HTTP на удаление объекта-домена CDMI с использованием типа содержимого CDMI приведены в таблице 79.

Таблица 79 – Заголовки запроса – удаление объекта-домена с использованием типа содержимого CDMI

Заголовок	Тип	Описание	Требование
X-CDMI-SpecificationVersion	Строка заголовка	Список версий, поддерживаемых клиентом, разделенных запятыми, например "1.0.2, 1.5, 2.0"	Обязательно

10.5.4 Тело сообщения-запроса

Сообщение-запрос может содержать тело, соответствующее RFC 2616.

10.5.5 Заголовки ответа

Сообщение-ответ может содержать заголовки, соответствующие RFC 2616.

10.5.6 Тело сообщения-ответа

Сообщение-ответ может содержать тело, соответствующее RFC 2616.

10.5.7 Статус запроса

В таблице 80 приведены коды состояний HTTP, которые могут возникнуть при удалении объекта-домена с использованием типа содержимого CDMI.

Таблица 80 – Коды состояний HTTP – удаление объекта-домена с использованием типа содержимого CDMI

Статус HTTP	Описание
204 No Content	Домен был успешно удален.
400 Bad Request	Запрос содержит неверные параметры или имена полей
401 Unauthorized	Неверные данные аутентификации/авторизации.
403 Forbidden	Клиент не обладает правами для выполнения данного запроса.
404 Not Found	Ресурс не найден по указанному URI.
409 Conflict	Домен не может быть удален, потому что имеются объекты, принадлежащие к данному домену, и поле cdmi_domain_delete_reassign отсутствует, неверное или не может быть использовано.

10.5.8 Пример

Пример – Применение DELETE к URI домена:
DELETE /cdmi_domains/MyDomain/ HTTP/1.1
Host: cloud.example.com
X-CDMI-Specification-Version: 1.0.2
Будет получен следующий ответ.
HTTP/1.1 204 No Content

11 Операции с ресурсами объекта-очереди

11.1 Обзор

Объекты-очереди предоставляют доступ в порядке поступления (FIFO, «первый пришел – первый вышел») при сохранении и получении данных. Для добавления элемента в объект-очередь поставщик данных выполняется запрос POST, а получатель данных выполняет запрос GET для получения элементов из очереди, при этом одновременно полученные элементы удаляются из очереди. Очереди предоставляют простой механизм, позволяющий нескольким поставщикам данных надежно передавать данные одному получателю. Если данный механизм поддерживается облачным хранилищем, клиенты создают объекты очереди как описано в 9.10 и в данном разделе.

В модели CDMI™ объекты-очереди могут адресоваться двумя способами:

- по имени (например, http://cloud.example.com/queueobject): and
- по ID объекта (например, http://cloud.example.com/cdmi_objectid/0000706D0010B84FAD185C425D8B537E).

Каждая очередь имеет единственный глобально-уникальный ID, который остается неизменным на протяжении жизненного цикла объекта. Каждый объект-очередь должен иметь один или несколько адресов URI для обеспечения доступа к объекту.

Очередь может иметь родительский объект. В этом случае, объект-очередь наследует те элементы метаданных системы данных, которые не заданы явным образом в самом объекте-очереди.

Примеры

1 Объект-очередь «receipts.queue», сохраненный по нижеуказанному URI, наследует метаданные системы данных от родительского контейнера «finance»:

http://cloud.example.com/finance/receipts.queue

Индивидуальные поля объекта-очереди могут быть доступны указанием имени поля после символа «?», следующего за URI объекта.

2 Следующий URI возвращает в теле сообщения-ответа в поле value элемент из головы очереди (элемент, дольше других находящийся в очереди):

http://cloud.example.com/queueobject?value

Кодировка данных, передаваемых в поле value объекта-очереди, определяется значением поля valuetransferencoding объекта-очереди:

- если valuetransferencoding равно «utf-8», данные, сохраняемые в очереди, должны быть корректной строкой UTF-8 и должны передаваться в поле value как строка UTF-8;
- если valuetransferencoding равно «base64», данные, сохраняемые в очереди, могут содержать произвольные бинарные последовательности и должны передаваться в поле value как строки в кодировке base 64.

Указание диапазона в запросе к полю value позволяет получить доступ к отдельным байтам объекта в очереди. Так, следующий URI возвращает первую тысячу байт элемента в голове очереди (элемента, дольше других находящегося в очереди):

http://cloud.example.com/queueobject?value:0-999

Так как диапазон байтов строки UTF-8 часто не является корректной строкой UTF-8, ответ на запрос диапазона должен всегда передаваться как строка в кодировке base 64.

Диапазоны байтов определяются как целые диапазоны, включающие границы (пункт 14.35.1 в RFC 2616).

Если клиент поддерживает или включает поля десериализации, которые не определены в настоящем стандарте, эти поля должны храниться как часть объекта.

11.1.1 Метаданные объекта-очереди

Метаданные объекта-очереди могут также включать произвольные элементы, созданные пользователем, и метаданные системы данных, как описано в разделе 16.

11.1.2 Адресация объекта-очереди

Каждый объект-очередь может адресоваться одним или несколькими URI, и все доступные операции должны выполняться через эти URI.

11.1.3 Представления объекта-очереди

Представления в данном разделе показаны с использованием нотации JSON. И клиенты, и серверы должны поддерживать представление JSON в кодировке UTF-8. В телах сообщений-запросов и ответов поля JSON могут указываться и возвращаться в любом порядке, за исключением (при наличии) полей valuerange и value, которые указываются последними и в данном порядке.

11.2 Создание объекта-очереди с использованием типа содержимого CDMI

11.2.1 Обзор

Для создания нового объекта-очереди следует выполнить запрос:

PUT <root URI>/<ContainerName>/<QueueName>

О создании объекта-очереди по ID, см. 9.10.

где:

- <root URI> путь к облаку CDMI;
- <ContainerName> неотрицательное число уже существующих промежуточных контейнеров, имена которых разделены одинарной наклонной чертой (т.е., «/»);

– <QueueName> Имя создаваемого объекта-очереди.

После создания, к объекту можно также обратиться как <root URI>/cdmi_objectid/<objectID>.

Новые объекты-очереди должны быть пустыми, за исключением случаев, когда они создаются посредством копирования или перемещения существующей непустой очереди, либо в результате десериализации непустой сериализованной очереди.

11.2.2 Отсроченное завершение создания:

В ответ на запрос создания объекта-очереди, сервер может вернуть код 202 Accepted, что указывает на то, что объект находится в процессе создания. Это полезно в случае длительных операций (например, копировании объекта-очереди с большим количеством значений). Такой ответ означает, что:

– сервер должен вернуть заголовок Location, содержащий URI к создаваемому объекту вместе со статусом HTTP 202 Accepted;

– статус 202 Accepted со стороны сервера удостоверяет, что были пройдены несколько проверок:

– пользователь авторизован для создания нового объекта-контейнера;

– пользователь авторизован для чтения любого исходного объекта, который необходимо переместить, скопировать, сериализовать или десериализовать;

– достаточно места для создания объекта-контейнера или, по крайней мере, достаточно места для создания URI к сообщению об ошибке.

– клиент может не иметь опции немедленно обратиться к созданному объекту, например, из-за задержек, вызванных использованием в реализации связности в конечном итоге.

Клиент выполняет операции GET к URI для отслеживания процесса создания. В ответ сервер возвращает два поля в теле сообщения-ответа, которые описывают состояние выполнения операции:

– обязательное текстовое поле completionStatus содержит «Processing», «Complete», либо строку сообщения об ошибке, начинающуюся с «Error»;

– опциональное поле percentComplete содержит процент выполнения принятого запроса PUT (от 0 до 100).

GET не возвращает объекты из очереди, если completionStatus не равно «Complete». Если создание объекта завершается с ошибкой, создается URI, а поле completionStatus устанавливается равным сообщению об ошибке. Удаление URI после обработки ошибки возлагается на клиента.

11.2.3 Опции

Следующие опции описывают поддерживаемые операции при создании нового объекта-очереди:

– поддержка возможности создания нового объекта-очереди обозначается наличием опции cdmi_create_queue в родительском контейнере;

– если очередь, создаваемая в родительском контейнере, является ссылкой, поддержка этой операции обозначается наличием опции cdmi_create_reference в родительском контейнере;

– если новая очередь является копией существующей, поддержка копирования обозначается наличием опции cdmi_copy_queue в родительском контейнере;

– если новая очередь создается в результате перемещения существующей очереди, поддержка этой операции обозначается наличием опции cdmi_move_queue в родительском контейнере;

– если новая очередь создается в результате операции десериализации, поддержка этой операции обозначается наличием опции cdmi_deserialize_queue в родительском контейнере.

11.2.4 Заголовки запроса

Заголовки HTTP запросов на создание объекта-очереди CDMI с использованием типа содержимого CDMI приведены в таблице 81.

Таблица 81 – Заголовки запроса – создание объекта-очереди с использованием типа содержимого CDMI

Заголовок	Тип	Описание	Требование
Accept	Строка заголовка	"application/cdmi-queue"	Обязательно
Content-Type	Строка заголовка	"application/cdmi-queue"	Обязательно
X-CDMI-SpecificationVersion	Строка заголовка	Список версий, поддерживаемых клиентом, разделенных запятыми, например, "1.0.2, 1.5, 2.0"	Обязательно

11.2.5 Тело сообщения-запроса

Поля тела сообщения-запроса на создание объекта-очереди с использованием типа содержимого CDMI приведены в таблице 82.

Таблица 82 – Тело сообщения-запроса – создание объекта-очереди с использованием типа содержимого CDMI

Имя поля	Тип	Описание	Требование
metadata	Объект JSON	Метаданные объекта-очереди Если данное поле включено при десериализации, сериализации, копировании или перемещении объекта-очереди, то значение этого поля должно заменять метаданные URI источника. Если данное поле не включено при десериализации, сериализации, копировании или перемещении объекта-очереди, то следует использовать метаданные URI источника. Если данное поле включено при создании нового объекта по значению, это поле должно быть использовано как метаданные. Если данное поле не включено при создании нового объекта по значению, этому полю следует поставить в соответствие пустой объект JSON (т.е., "{}"). Данное поле не должно быть задано при создании ссылки на объект-очередь.	Опционально
domainURI	Строка JSON	URI домена-владельца В случае отличия от родительского домена, пользователь должен обладать правами cross_domain (см. cdmi_member_privileges в таблице 64). Если не указано, должен использоваться родительский домен.	Опционально
deserialize	Строка JSON	URI сериализованного объекта данных CDMI, который должен быть десериализован при создании новой очереди	Опционально[a]
copy	Строка JSON	URI объекта CDMI, который должен быть скопирован в новый объект-очередь	Опционально[a]
move	Строка JSON	URI существующего локального или удаленного объекта-очереди CDMI (URI источника), который должен быть перемещен в URI, указанный в команде PUT. Содержимое объекта-очереди, включая ID объекта, должны при перемещении сохраняться, а исходный объект должен быть уничтожен после успешного копирования. Если недостаточно прав для чтения объекта по URI источника, записи объекта по URI нового объекта, удаления объектов по URI источника или одна из этих операций завершается с ошибкой, сервер должен вернуть код результата 400 Bad Request, причем исходный и конечный объекты должны сохраниться неизменными.	Опционально[a]
reference	Строка JSON	URI объекта-очереди CDMI, на который должна быть сделана переадресация. Если при создании ссылки указаны другие поля, сервер должен вернуть код ошибки 400 Bad Request.	Опционально[a]
deserialize-value	Строка JSON	Сериализованный объект-очередь (см. гл. 15), кодированный с помощью base 64 как описано в RFC 4648.	Опционально[a]

[a] Лишь одно из этих полей должно быть указано в любой из операций. За исключением поля value, данные поля не должны сохраняться. Если указано более чем одно из этих полей, сервер должен вернуть сообщение об ошибке 400 Bad Request.

11.2.6 Заголовки ответа

Заголовки HTTP ответа на создание объекта-очереди CDMI с использованием типа содержимого CDMI указаны в таблице 83.

Таблица 83 – Заголовки ответа – создание объекта-очереди с использованием типа содержимого CDMI

Заголовок	Тип	Описание	Требование
Content-Type	Строка заголовка	"application/cdmi-queue"	Обязательно
X-CDMI-Specification-Version	Строка заголовка	Сервер должен вернуть наибольший номер версии, поддерживаемой и сервером, и клиентом, например "1.0.2". Если сервер не поддерживает ни одной версии, поддерживаемой клиентом, сервер должен вернуть код состояния 400 Bad Request.	Обязательно

11.2.7 Тело сообщения-ответа

Поля тела сообщения-ответа на создание объекта-очереди CDMI с использованием типа содержимого CDMI приведены в таблице 84.

Таблица 84 – Тело сообщения-ответа – создание объекта-очереди с использованием типа содержимого CDMI

Имя поля	Тип	Описание	Требование
objectType	Строка JSON	"application/cdmi-queue"	Обязательно
objectID	Строка JSON	ID объекта	Обязательно
objectName	Строка JSON	Имя объекта	Обязательно
parentURI	Строка JSON	URI родительского объекта Добавление objectName к parentURI должно всегда давать корректный URI объекта.	Обязательно
parentID	Строка JSON	ID родительского объекта-контейнера	Обязательно
Domain URI	Строка JSON	URI домена-владельца	Обязательно
Capabilities-URI	Строка JSON	URI опций объекта	Обязательно
completion-Status	Строка JSON	Строка, указывающая, находится ли объект в процессе создания, а после создания – был ли он создан успешно или с ошибкой. Значение должно быть строкой "Processing", "Complete" или строкой, начинающейся с "Error".	Обязательно
percent-Complete	Строка JSON	Если значение completionStatus равно "Processing", данное поле, при наличии, должно показывать процент выполнения операции создания объекта, числовым значением от 0 до 100. Если значение completionStatus равно "Complete", данное поле, при наличии, должно иметь значение "100". Если значение completionStatus начинается с "Error", данное поле, при наличии, может содержать любое целое число от 0 до 100.	Опционально
metadata	Объект JSON	Метаданные объекта-очереди. Могут включать пользовательские метаданные и метаданные системы данных, указанные в теле запроса на создание (поле metadata) вместе с метаданными системы хранения, создаваемыми облачным хранилищем. Подробнее о метаданных см. раздел 16.	Обязательно
queue-Values	Строка JSON	Диапазон индексов элементов в очереди. Каждый элемент в очереди должен иметь уникальный монотонно возрастающий целый положительный индекс, начиная с 0. Если очередь пустая, должна возвращаться пустая строка. Если очередь не пустая, должны быть возвращены минимальный и максимальный (в таком порядке) индекс значений в очереди, разделенные дефисом ("-")	Обязательно

11.2.8 Статус запроса

В таблице 85 приведены коды состояний HTTP, которые возникают при создании объекта-очереди с использованием типа содержимого CDMI.

Таблица 85 – Коды состояний HTTP – создание объекта-очереди с использованием типа содержимого CDMI

Статус HTTP	Описание
201 Created	Новый объект-очередь был создан.
202 Accepted	Новый объект-очередь в процессе создания. Клиент CDMI должен отслеживать поля completionStatus и percentComplete для определения текущего статуса операции.
400 Bad Request	Запрос содержит неверные параметры или имена полей
401 Unauthorized	Неверные данные аутентификации/авторизации.
403 Forbidden	Клиент не обладает правами для выполнения данного запроса.
404 Not Found	Ресурс не найден по указанному URI.
409 Conflict	Операция конфликтует с блокировкой не-CDMI протокола доступа или может вызвать ошибку передачи на сервер.

11.2.9 Пример

Пример – Применение PUT к URI объекта-очереди: задание имени очереди и содержимого:

```
PUT /MyContainer/MyQueue HTTP/1.1
Host: cloud.example.com
Accept: application/cdmi-queue
Content-Type: application/cdmi-queue X-CDMI-Specification-Version: 1.0.2
{
«metadata» : {

}
}
```

Будет получен следующий ответ.

```
HTTP/1.1 201 Created
Content-Type: application/cdmi-queue X-CDMI-Specification-Version: 1.0.2
{
«objectType» : «application/cdmi-queue»,
«objectID» : «00007E7F00104BE66AB53A9572F9F51E»,
«objectName» : «MyQueue»,
«parentURI « : «/MyContainer/»,
«parentID» : «0000706D0010B84FAD185C425D8B537E»,
«domainURI» : «/cdmi_domains/MyDomain/»,
«capabilitiesURI» : «/cdmi_capabilities/queue/»,
«completionStatus» : «Complete»,
«metadata» : {

},
«queueValues» : «»
}
```

11.3 Чтение объекта-очереди с использованием типа содержимого CDMI

11.3.1 Обзор

Для чтения всех полей существующего объекта-очереди следует выполнить запрос:

GET <root URI>/<ContainerName>/<QueueName>

Для чтения одного или нескольких определенных полей существующего объекта-контейнера следует выполнить один из следующих запросов:

GET <root URI>/<ContainerName>/<QueueName>?<fieldname>;<fieldname>;...

GET <root URI>/<ContainerName>/<QueueName>?value:<range>;...

GET <root URI>/<ContainerName>/<QueueName>?metadata:<prefix>;...

Для чтения одного или нескольких элементов существующего объекта-очереди следует выполнить запрос:

GET <root URI>/<ContainerName>/<QueueName>?values:<count>

где:

- <root URI> путь к облаку CDMI;
- <ContainerName> неотрицательное число промежуточных контейнеров;
- <QueueName> имя объекта-очереди для чтения;
- <fieldname> имя поля;
- <range> диапазон байтов значения объекта-очереди, которые должны быть возвращены в поле value. При указании диапазона байтов полученное значение должно быть возвращено из элемента в голове очереди;
- <prefix> соответствующий префикс, возвращающий все метаданные, начинающиеся с данного префикса;
- <count> число элементов, которые должны быть извлечены из очереди. Если запрашивается больше значений, чем имеется в очереди, запрос должен обрабатываться так, как будто count равно числу элементов в очереди.

К объекту возможно также обратиться как <root URI>/cdmi_objectid/<objectID>.

По умолчанию, чтение из объекта-очереди должно возвращать элемент из головы очереди при условии, что диапазон queueValues не пуст.

11.3.2 Опции

Следующие опции описывают поддерживаемые операции при чтении существующего объекта-очереди:

- поддержка возможности чтения метаданных существующего объекта-очереди обозначается наличием опции cdmi_read_metadata в очереди;
- поддержка возможности чтения значения в существующем объекте-очереди обозначается наличием опции cdmi_read_value в очереди.

11.3.3 Заголовки запроса

Заголовки HTTP запроса на чтение из объекта-очереди CDMI с использованием типа содержимого CDMI приведены в таблице 86.

Таблица 86 – Заголовки запроса – чтение из объекта-очереди с использоеанием типа содержимого CDMI

Заголовок	Тип	Описание	Требование
Accept	Строка заголовка	"application/cdmi-container" или совместимое значение согласно 5.13.2	Опционально
X-CDMI-SpecificationVersion	Строка заголовка	Список версий, поддерживаемых клиентом, разделенных запятыми, например "1.0.2, 1.5, 2.0"	Обязательно

11.3.4 Тело сообщения-запроса

Не следует формировать тело сообщения-запроса.

11.3.5 Заголовки ответа

Заголовки HTTP ответа на чтение из объекта-очереди CDMI с использованием типа содержимого CDMI приведены в таблице 87.

Таблица 87 – Заголовки ответа – чтение из объекта-очереди с использованием типа содержимого CDMI

Заголовок	Тип	Описание	Требование
X-CDMI-SpecificationVersion	Строка заголовка	Сервер должен вернуть наибольший номер версии, поддерживаемой и сервером, и клиентом, например, "1.0.2". Если сервер не поддерживает ни одной версии, поддерживаемой клиентом, сервер должен вернуть код состояния 400 Bad Request.	Обязательно
Content-Type	Строка заголовка	"application/cdmi-queue"	Обязательно
Location	Строка заголовка	Сервер должен вернуть URI, на который происходит переадресация, если объект является ссылкой.	Условно

11.3.6 Тело сообщения-ответа

Поля тела сообщения-ответа для чтения объекта-очереди CDMI с использованием типа содержимого CDMI приведены в таблице 88.

Таблица 88 – Тело сообщения-ответа – чтение из объекта-очереди с использованием типа содержимого CDMI

Имя поля	Тип	Описание	Требование
objectType	Строка JSON	"application/cdmi-queue"	Обязательно
objectID	Строка JSON	ID объекта	Обязательно
objectName	Строка JSON	Имя объекта Для объектов в контейнере следует вернуть поле objectName. Для объектов не в контейнере (доступных только по ID), поле objectName не существует и не должно передаваться в ответе	Условно

Продолжение таблицы 88

Имя поля	Тип	Описание	Требование
parentURI	Строка JSON	URI родительского объекта Для объектов в контейнере следует вернуть поле parentURI. Для объектов не в контейнере (доступных только по ID), поле parentURI не существует и не должно передаваться в ответе. Добавление objectName к parentURI должно всегда возвращать правильный URI объекта	Условно
parentID	Строка JSON	ID объекта родительского контейнера Для объектов в контейнере следует вернуть поле parentID. Для объектов не в контейнере (доступных только по ID), поле parentID не существует и не должно быть возвращено	Условно
domainURI	Строка JSON	URI домена-владельца	Обязательно
capabilitiesURI	Строка JSON	URI опций объекта	Обязательно
completion-Status	Строка JSON	Строка, указывающая, находится ли объект в процессе создания, а после создания – был ли он создан успешно или с ошибкой. Значение должно быть строкой "Processing", "Complete" или строкой, начинающейся с "Error"	Обязательно
percent-Complete	Строка JSON	Если значение completionStatus равно "Processing", данное поле, при наличии, должно показывать процент выполнения операции создания объекта, числовым значением от 0 до 100. Если значение completionStatus равно "Complete", данное поле, при наличии, должно иметь значение "100". Если значение completionStatus начинается с "Error", данное поле, при наличии, может содержать любое целое число от 0 до 100	Опционально
metadata	Объект JSON	Метаданные объекта-очереди. Могут включать пользовательские метаданные и метаданные системы данных, указанные в теле запроса на создание (поле metadata) вместе с метаданными системы хранения, создаваемыми облачным хранилищем. Подробнее о метаданных см. разделе 16	Обязательно
queueValues	Строка JSON	Диапазон индексов элементов в очереди. Каждый элемент в очереди должен иметь уникальный монотонно возрастающий целый положительный индекс, начиная с 0. Если очередь пустая, должна возвращаться пустая строка. Если очередь не пустая, должны быть возвращены минимальный и максимальный (в таком порядке) индекс значений в очереди, разделенные дефисом ("-")	Обязательно
mimetype	Массив JSON строк JSON	Типы MIME для каждого из элементов очереди Возвращаются типы MIME элементов, каждый на соответствующей позиции массива JSON. Это поле должно возвращаться только если completionStatus равно"Complete" и в очереди находятся одно или несколько значений	Опционально
valuerange	Массив JSON строк JSON	Диапазон байт элементов объекта-очереди, которые должен быть возвращены в поле value Возвращаются диапазоны значений элементов, каждый на соответствующей позиции в массиве JSON. При запросе определенного диапазона, запись в поле диапазона значений должна соответствовать запрошенным байтам Если запрос выходит за пределы значения, поле valuerange должно указывать на возвращенный меньший диапазон. Это поле должно возвращаться только если completionStatus равно"Complete", и в очереди находятся один или несколько значений	Опционально

Окончание таблицы 88

Имя поля	Тип	Описание	Требование
valuetransfer-encoding	Массив JSON строк JSON	Кодировки, использованные для отдельных объектов в очереди. Определены два значения кодировки. - "utf-8" указывает на то, что объект данных содержит корректную строку UTF-8, и должен передаваться как строка UTF-8 в поле value. - "base64" указывает на то, что объект данных содержит произвольную бинарную последовательность и должен передаваться как строка поля значения в кодировке base 64 Возвращаются кодировки каждое на соответствующей позиции в массиве JSON. Это поле должно возвращаться только если completionStatus равно"Complete"	Опционально
value	Массив JSON строк JSON	Значения объектов в голове очереди Значения возвращаются в массиве JSON в порядке от старых к новым. Если поле valuetransferencoding указывает на кодировку UTF-8, значение должно быть строкой UTF-8, сформированной по правилам JSON как описано в RFC 4627. Если поле valuetransferencoding указывает на кодировку base 64, значение должно быть вначале кодировано в base 64 по правилам, описанным в RFC 4648. Это поле должно возвращаться только если completionStatus равно"Complete"	Условно

Если в запросе GET указаны отдельные поля, в теле ответа должны возвращаться только эти поля.

Опциональные запрошенные, но несуществующие поля опускаются в теле ответа.

11.3.7 Статус запроса

В таблице 89 приведены коды состояний HTTP, возникающих при чтении из объекта-очереди с использованием типа содержимого CDMI.

Таблица 89 – Коды состояний HTTP – чтение из объекта-очереди с использованием типа содержимого CDMI

Статус HTTP	Описание
200 OK	Содержимое объекта-очереди возвращено в сообщении-ответе.
302 Found	URI ссылается на другой URI.
400 Bad Request	Запрос содержит неверные параметры или имена полей
401 Unauthorized	Неверные данные аутентификации/авторизации.
403 Forbidden	Клиент не обладает правами для выполнения данного запроса.
404 Not Found	Ресурс не найден по указанному URI.
406 Not Acceptable	Сервер не может предоставить объект, типизованный как обозначено в заголовке Accept.

11.3.8 Примеры

Примеры

1 Применение GET к URI объекта-очереди для чтения всех полей объекта-очереди:

```
GET /MyContainer/MyQueue HTTP/1.1
Host: cloud.example.com
Accept: application/cdmi-queue
X-CDMI-Specification-Version: 1.0.2
Будет получен следующий ответ.
HTTP/1.1 200 OK
Content-Type: application/cdmi-queue
X-CDMI-Specification-Version: 1.0.2
{
«objectType»: «application/cdmi-queue»,
«objectID»: «00007E7F00104BE66AB53A9572F9F51E»,
«objectName»: «MyQueue»,
```

«parentURI»: «/MyContainer/»,
«parentID» : «0000706D0010B84FAD185C425D8B537E»,
«domainURI»: «/cdmi_domains/MyDomain/»,
«capabilitiesURI»: «/cdmi_capabilities/queue/»,
«completionStatus»: «Complete»,
«metadata»: {},
«queueValues»: «1-2»,
«mimetype»: [
«text/plain»
],
«valuerange»: [
«0-19»
],
«valuetransferencoding»: [
«utf-8»
],
«value»: [
«First Enqueued Value»
]229 }

2 Применение GET к URI объекта-очереди для чтения полей value и queueValues объекта-очереди:
GET /MyContainer/MyQueue?value;queueValues HTTP/1.1
Host: cloud.example.com
Accept: application/cdmi-queue
X-CDMI-Specification-Version: 1.0.2
Будет получен следующий ответ.
HTTP/1.1 200 OK
Content-Type: application/cdmi-queue
X-CDMI-Specification-Version: 1.0.2
{
«queueValues» : «1-2»,
«value» : [
«First Enqueued Value»
] 244 }

3 Применение GET к URI объекта-очереди для чтения первых пяти байт значения объекта-очереди:
GET /MyContainer/MyQueue?value:0-5 HTTP/1.1
Host: cloud.example.com
Accept: application/cdmi-queue
X-CDMI-Specification-Version: 1.0.2
Будет получен следующий ответ:
HTTP/1.1 200 OK
Content-Type: application/cdmi-queue
X-CDMI-Specification-Version: 1.0.2
{
«value» : [
«First»
] 258 }

4 Применение GET к URI объекта-очереди для чтения двух значений объекта-очереди:
GET /MyContainer/MyQueue?mimetype;valuerange;values:2 HTTP/1.1
Host: cloud.example.com
Accept: application/cdmi-queue
X-CDMI-Specification-Version: 1.0.2
Будет получен следующий ответ.
HTTP/1.1 200 OK
Content-Type: application/cdmi-queue
X-CDMI-Specification-Version: 1.0.2
{
«mimetype» : [
«text/plain»,
«text/plain»

```
    ],
    «valuerange» : [
    «0-19»,
    «0-20»
    ],
    «value» : [
    «First Enqueued Value»,
    «Second Enqueued Value»
    ]
}
```

11.4 Изменение объекта-очереди с использованием типа содержимого CDMI

11.4.1 Обзор

Для изменения некоторых или всех полей существующего объекта-очереди (за исключением постановки в очередь) следует выполнить запрос:

PUT <root URI>/<ContainerName>/<QueueName>

Для добавления, обновления или удаления определенных элементов метаданных существующего объекта-очереди следует выполнить запрос:

PUT <root URI>/<ContainerName>/<QueueName>?metadata:<metadataname>;...

где:

- <root URI> путь к облаку CDMI;
- <ContainerName> неотрицательное число промежуточных контейнеров;
- <QueueName> имя изменяемого объекта-очереди.

Объект-очередь также доступен как <root URI>/cdmi_objectid/<objectID>/. Обновление объекта не должно изменять его ID.

11.4.2 Опции

Следующие опции описывают поддерживаемые операции при изменении существующего объекта-очереди:

- поддержка возможности изменения метаданных существующего объекта-очереди обозначается наличием опции cdmi_modify_metadata в очереди.

11.4.3 Заголовки запроса

Заголовки запроса HTTP для изменения объекта-очереди CDMI с использованием типа содержимого CDMI перечислены в таблице 90.

Таблица 90 – Заголовки запроса – изменение объекта-очереди с использованием типа содержимого CDMI

Заголовок	Тип	Описание	Требование
Content-Type	Строка заголовка	"application/cdmi-queue "	Обязательно
X-CDMI-SpecificationVersion	Строка заголовка	Список версий, поддерживаемых клиентом, разделенных запятыми, например "1.0.2, 1.5, 2.0"	Обязательно

11.4.4 Тело сообщения-запроса

Поля тела запроса на изменение объекта-очереди с использованием типа содержимого CDMI перечислены в таблице 91.

Таблица 91 – Тело сообщения-запроса – изменение объекта-очереди с использованием типа содержимого CDMI

Имя поля	Тип	Описание	Требование
metadata	Объект JSON	Метаданные для объекта-очереди. Если присутствуют, указанные метаданные замещают существующие метаданные объекта. Если URI указывает на отдельные элементы метаданных, остальные элементы не должны меняться. Подробнее о метаданных см. раздел 16.	Опционально

Окончание таблицы 91

domainURI	Строка JSON	URI домена-владельца - В случае отличия от родительского домена, пользователь должен обладать правами cross_domain (см. cdmi_member_privileges в табл. 64). - Если не указано, должен использоваться родительский домен.	Опционально
Deserialize	Строка JSON	URI сериализованного объекта-очереди CDMI, который должен быть десериализован для изменения существующего объекта-очереди. ID сериализованного объекта-очереди должен совпадать с ID объекта-назначения. Все элементы сериализованной очереди должны быть добавлены в изменяемую очередь.	Опционально[a]
copy	Строка JSON	URI объекта-очереди CDMI, который должен быть скопирован в имеющийся объект-очередь. Элементы очереди не копируются. Копирование элементов в очереди см. 11.6.	Опционально[a]
deserializevalue	Строка JSON	Сериализованный объект-очередь (см. раздел 15), кодированный с помощью base 64 как описано в RFC 4648. ID сериализованного объекта-очереди должно соответствовать ID очереди-цели. Все элементы сериализованной очереди должны быть добавлены в изменяемую очередь.	Опционально[a]
[a] Лишь одно из этих полей должно быть указано в любой из операций. За исключением поля value, данные поля не должны сохраняться.			

11.4.5 Заголовок ответа

Заголовок ответа HTTP на изменение объекта-очереди CDMI с использованием типа содержимого CDMI приведен в таблице 92.

Таблица 92 – Заголовок ответа – изменение объекта-очереди с использованием типа содержимого CDMI

Заголовок	Тип	Описание	Требование
Location	Строка заголовка	Сервер должен вернуть URI, на который происходит переадресация, если объект является ссылкой.	Условно

11.4.6 Тело сообщения-ответа

Сообщение-ответ может содержать тело, соответствующее RFC 2616.

11.4.7 Статус запроса

В таблице 93 приведены коды состояний HTTP, которые возникают при изменении объекта-очереди с использованием типа содержимого CDMI.

Таблица 93 – Коды состояний HTTP – изменение объекта-очереди с использованием типа содержимого CDMI

Статус HTTP	Описание
204 No Content	Операция успешна, данные не возвращены.
302 Found	URI ссылается на другой URI.
400 Bad Request	Запрос содержит неверные параметры или имена полей
401 Unauthorized	Неверные данные аутентификации/авторизации.
403 Forbidden	Клиент не обладает правами для выполнения данного запроса.
404 Not Found	Ресурс не найден по указанному URI.
409 Conflict	Операция конфликтует с блокировкой не-CDMI протокола доступа или может вызвать ошибку передачи на сервер.

11.4.8 Пример

Пример – Применение PUT к URI объекта-очереди для установки новых метаданных:

```
PUT /MyContainer/MyQueue HTTP/1.1
Host: cloud.example.com
Content-Type: application/cdmi-queue X-CDMI-Specification-Version: 1.0.2
```

{
«metadata» : {

} 325 }
Будет получен следующий ответ.
HTTP/1.1 204 No Content

11.5 Удаление объекта-очереди с использованием типа содержимого CDMI

11.5.1 Обзор

Для удаления существующего объекта-очереди, включая все элементы очереди, следует выполнить запрос:

DELETE <root URI>/<ContainerName>/<QueueName>

где:

- <root URI> путь к облаку CDMI;
- <ContainerName> неотрицательное число промежуточных контейнеров;
- <QueueName> имя уничтожаемого объекта-очереди.

К объекту можно также обратиться как <root URI>/cdmi_objectid/<objectID>.

11.5.2 Опции

Следующие опции описывают поддерживаемые операции при удалении существующего объекта-очереди:

- поддержка возможности удаления существующего объекта-очереди обозначается наличием опции cdmi_delete_queue в очереди.

11.5.3 Заголовок запроса

Заголовок запроса HTTP для удаления объекта-очереди CDMI с использованием типа содержимого CDMI показан в таблице 94.

Таблица 94 – Заголовок запроса – удаление объекта-очереди с использованием типа содержимого CDMI

Заголовок	Тип	Описание	Требование
X-CDMI-SpecificationVersion	Строка заголовка	Список версий, поддерживаемых клиентом, разделенных запятыми, например "1.0.2, 1.5, 2.0"	Обязательно

11.5.4 Тело сообщения-запроса

Сообщение-запрос может содержать тело, соответствующее RFC 2616.

11.5.5 Заголовки ответа

Сообщение-ответ может содержать заголовки, соответствующие RFC 2616.

11.5.6 Тело сообщения-ответа

Сообщение-ответ может содержать тело, соответствующее RFC 2616.

11.5.7 Статус запроса

В таблице 95 приведены коды состояний HTTP, возникающих при удалении объекта очереди с использованием типа содержимого CDMI.

Таблица 95 – Коды состояний HTTP – удаление объекта-очереди с использованием типа содержимого CDMI

Статус HTTP	Описание
204 No Content	Объект-очередь успешно удален.
400 Bad Request	Запрос содержит неверные параметры или имена полей
401 Unauthorized	Неверные данные аутентификации/авторизации.
403 Forbidden	Клиент не обладает правами для выполнения данного запроса.
404 Not Found	Ресурс не найден по указанному URI.
409 Conflict	Объект-очередь не может быть удален (может быть постоянным).

11.5.8 Пример

Пример – Применение DELETE к URI объекта-очереди:
DELETE /MyContainer/MyQueue HTTP/1.1
Host: cloud.example.com
X-CDMI-Specification-Version: 1.0.2
Будет получен следующий ответ.
HTTP/1.1 204 No Content

11.6 Добавление элемента в очередь с использованием типа содержимого CDMI

11.6.1 Обзор

Для постановки в очередь одного или нескольких значений следует выполнить запрос:
POST <root URI>/<ContainerName>/<QueueName>

где:

– <root URI> путь к облаку CDMI;

– <ContainerName> неотрицательное число существующих промежуточных контейнеров, имена которых разделены одинарной наклонной чертой («/»);

– <QueueName> объект-очередь для пополнения.

К объекту можно также адресоваться как <root URI>/cdmi_objectid/<objectID>.

11.6.2 Опции

Следующие опции описывают поддерживаемые операции при добавлении нового элемента в существующую очередь:

– поддержка возможности изменения значения в существующем объекте-очереди обозначается наличием опции cdmi_modify_value в очереди.

11.6.3 Заголовки запроса

Заголовки HTTP запросов на добавление элементов в объект-очередь CDMI с использованием типа содержимого CDMI приведены в таблице 96.

Таблица 96 – Заголовки запроса – добавление нового элемента в очередь с использованием типа содержимого CDMI

Заголовок	Тип	Описание	Требование
Content-Type	Строка заголовка	"application/cdmi-queue"	Обязательно
X-CDMI-SpecificationVersion	Строка заголовка	Список версий, поддерживаемых клиентом, разделенных запятыми, например "1.0.2, 1.5, 2.0"	Обязательно

11.6.4 Тело сообщения-запроса

Поля тела сообщения-запроса добавление элементов в объект-очередь CDMI с использованием типа содержимого CDMI приведены в таблице 97.

Таблица 97 – Тело сообщения-запроса – добавление элементов в очередь с использованием типа содержимого CDMI (лист 1 из 2)

Имя поля	Тип	Описание	Требование
mimetype	Массив JSON строк JSON	Тип(ы) MIME для элементов, добавляемых в очередь. Данное поле должно храниться как часть объекта. Если это поле не указано, ему должно быть присвоено значение "text/plain". Должны быть представлены столько же элементов массива, сколько имеется элементов в поле value, причем типы должны соответствовать элементам на соответствующих позициях. Значение этого поля должно быть перед сохранением преобразовано в нижний регистр.	Опционально

Окончание таблицы 97

Имя поля	Тип	Описание	Требование
copy	Строка JSON	URI объекта-очереди или данных CDMI, который необходимо добавить в объект-очередь Если приведен URI объекта данных, то поля value, mimetype и valuetransferencoding объекта данных используются для создания нового элемента очереди, а объект данных атомарно уничтожается. Если задан URI объекта-очереди, то поля value, mimetype и valuetransferencoding указанного количества добавляемых элементов переносятся в очередь-назначение и атомарно удаляются из очереди-источника.	Опционально[a]
move	Строка JSON	URI объекта данных или очереди CDMI, из которых необходимо взять элементы для постановки в очередь. Исходные элементы удаляются после успешного добавления в очередь	Опционально[a]
valuetransferencoding	Массив JSON строк JSON	Кодировка(и), использованная(ые) при передаче элемента(ов) объекта-очереди. Определены два значения кодировки. - "utf-8" указывает на то, что объект данных содержит корректную строку UTF-8, и должен передаваться как строка UTF-8 в поле value. - "base64" указывает на то, что объект данных содержит произвольную бинарную последовательность и должен передаваться как строка поля значения в кодировке base 64. Попытка установить значение элемента иное, чем корректная строка в кодировке base 64, должно приводить к ошибке 400 Bad Request. Если значение поля не указано, оно должно быть установлено в "utf-8". Данное поле должно храниться как часть объекта.	Опционально
value	Массив JSON строк JSON	Элемент(ы) для добавления в очередь. Если соответствующее поле valuetransferencoding указывает на кодировку UTF-8, значение должно быть строкой UTF8 согласно правилам JSON, описанным в RFC 4627. Если соответствующее поле valuetransferencoding указывает на кодировку base 64 encoding, значение должно быть вначале перекодировано в base 64 как описано в RFC 4648.	Опционально[a]
[a] Лишь одно из этих полей должно быть указано в любой из операций. За исключением поля value, данные поля не должны сохраняться. Если указано более чем одно из этих полей, сервер должен вернуть сообщение об ошибке 400 Bad Request.			

11.6.5 Заголовки ответа

Сообщение-ответ может содержать заголовки, соответствующие RFC 2616.

11.6.6 Тело сообщения-ответа

Сообщение-ответ может содержать тело, соответствующее RFC 2616.

11.6.7 Статус запроса

В таблице 98 приведены коды состояний HTTP, которые могут возникнуть при добавлении элементов в очередь с использованием типа содержимого CDMI.

Таблица 98 – Коды состояний HTTP – добавление в очередь нового элемента с использованием типа содержимого CDMI

Статус HTTP	Описание
204 No Content	Новые элементы добавлены в очередь.
400 Bad Request	Запрос содержит неверные параметры или имена полей
401 Unauthorized	Неверные данные аутентификации/авторизации.
403 Forbidden	Клиент не обладает правами для выполнения данного запроса.
404 Not Found	Ресурс не найден по указанному URI.
409 Conflict	Операция конфликтует с блокировкой не-CDMI протокола доступа или может вызвать ошибку передачи на сервер.

11.6.8 Примеры

Примеры

1 Применение POST к URI объекта-очереди для добавления нового элемента:

```
POST /MyContainer/MyQueue HTTP/1.1
Host: cloud.example.com
Content-Type: application/cdmi-queue X-CDMI-Specification-Version: 1.0.2
{
«mimetype» : [
«text/plain»
],
«value» : [
«Value to Enqueue»
] 403     }
```

Будет получен следующий ответ.

```
HTTP/1.1 204 No Content
```

2 Применение POST к URI объекта-очереди для копирования существующего элемента:

```
POST /MyContainer/MyQueue HTTP/1.1
Host: cloud.example.com
Content-Type: application/cdmi-object
X-CDMI-Specification-Version: 1.0.2
{
«copy» : «/MyContainer/MyDataObject»
}
```

Будет получен следующий ответ.

```
HTTP/1.1 204 No Content
```

3 Применение POST к URI объекта-очереди для переноса 20 значений из другого объекта-очереди:

```
POST /MyContainer/MyQueue HTTP/1.1
Host: cloud.example.com
Content-Type: application/cdmi-object
X-CDMI-Specification-Version: 1.0.2
{
«move» : «/MyContainer/FirstQueue?values:20»
}
```

Будет получен следующий ответ.

```
HTTP/1.1 204 No Content
```

4 Применение POST к URI объекта-очереди для добавления двух новых элементов:

```
POST /MyContainer/MyQueue
HTTP/1.1 Host: cloud.example.com
Content-Type: application/cdmi-object
X-CDMI-Specification-Version: 1.0.2
{
«mimetype» : [
«text/plain»,
«text/plain»
],
«value» : [
«First»,
«Second»
] 440     }
```

Будет получен следующий ответ.

```
HTTP/1.1 204 No Content
```

5 Применение POST к URI объекта-очереди для добавления двух новых значений, одного в кодировке base 64 и другого в кодировке utf-8:

```
POST /MyContainer/MyQueue HTTP/1.1
Host: cloud.example.com
Content-Type: application/cdmi-object
X-CDMI-Specification-Version: 1.0.2
```

{
«mimetype»: [
«text/plain»,
«text/plain»
],
«valuetransferencoding»: [
«utf-8»,
«base64»
],
«value»: [
«First»,
«U2Vjb25k»
]463 }
Будет получен следующий ответ.
HTTP/1.1 204 No Content

11.7 Удаление значения объекта-очереди с использованием типа содержимого CDMI

11.7.1 Обзор

Для удаления одного или нескольких элементов из головы очереди следует выполнить запрос:

DELETE <root URI>/<ContainerName>/<QueueName>?value

DELETE <root URI>/<ContainerName>/<QueueName>?values:<count>

где:

- <root URI> путь к облаку CDMI;
- <ContainerName> неотрицательное число промежуточных контейнеров;
- <QueueName> имя очереди, из которой удаляются элементы;
- <count> число элементов, начиная с головы очереди, которые следует удалить из объекта-очереди. Если запрошено удаление большего количества элементов, чем есть в очереди, данное значение должно быть установлено равным количеству элементов в объекте-очереди.

К объекту можно также обратиться как <root URI>/cdmi_objectid/<objectID>.

Суффикс «?value» на конце URI объекта-очереди должен быть добавлен, чтобы различать удаление элементов очереди от удаления объекта очереди (см. 11.5).

11.7.2 Опции

Следующие опции описывают поддерживаемые операции при удалении существующего значения объекта из очереди:

- поддержка возможности удаления значения из существующего объекта-очереди обозначается наличием опции cdmi_modify_value в очереди.

11.7.3 Заголовок запроса

Заголовок HTTP запросов на удаление элементов из объекта-очереди CDMI с использованием типа содержимого CDMI приведен в таблице 99.

Таблица 99 – Заголовок запроса – удаление элемента из объекта-очереди с использованием типа содержимого CDMI

Заголовок	Тип	Описание	Требование
X-CDMI-SpecificationVersion	Строка заголовка	Список версий, поддерживаемых клиентом, разделенных запятыми, например "1.0.2, 1.5, 2.0"	Обязательно

11.7.4 Тело сообщения-запроса

Сообщение-запрос может содержать тело, соответствующее RFC 2616.

11.7.5 Заголовки ответа

Сообщение-ответ может содержать заголовки, соответствующие RFC 2616.

11.7.6 Тело сообщения-ответа

Сообщение-ответ может содержать тело, соответствующее RFC 2616.

11.7.7 Статус запроса

В таблице 100 приведены коды состояний HTTP, которые могут возникнуть при удалении элементов из объекта-очереди с использованием содержимого типа CDMI.

Таблица 100 – Коды состояний HTTP – удаление элемента из объекта-очереди с использованием типа содержимого CDMI

Статус HTTP	Описание
204 No Content	Элемент очереди успешно удален.
400 Bad Request	Запрос содержит неверные параметры или имена полей
401 Unauthorized	Неверные данные аутентификации/авторизации.
403 Forbidden	Клиент не обладает правами для выполнения данного запроса.
404 Not Found	Ресурс не найден по указанному URI.
409 Conflict	Не удалось удалить элемент очереди (может быть постоянным).

11.7.8 Пример

Пример – Применение DELETE к URI объекта-очереди: удаление элемента для доступа к следующему элементу:

```
DELETE /MyContainer/MyQueue?value HTTP/1.1
Host: cloud.example.com
X-CDMI-Specification-Version: 1.0.2
```

Будет получен следующий ответ.

```
HTTP/1.1 204 No Content
```

12 Операции с ресурсами объекта-опции

12.1 Обзор

Объекты-опции позволяют клиенту CDMI™ определять, какое подмножество настоящего стандарта реализуется провайдером CDMI.

Для каждого URI в системе облачного хранения набор взаимодействий, которые возможно осуществить для данного URI, описывается наличием именованных «опций». Каждая опция, присутствующая для данного URI показывает, какие функции предоставляет система по отношению к данному URI. Опции всегда статически заданы.

Опции могут отличаться от операций, допустимых согласно списку контроля доступа (Access Control List, ACL) (см. 16.1), соответствующего данному URI. Например, облако, реализующее только чтение, может не допускать доступ для записи к контейнеру или объекту, несмотря на разрешение данных операция в ACL.

Облачные клиенты могут использовать опции для определения, какие операции поддерживаются. Если по отношению к объекту CDMI запрашивается выполнение операции, для которой отсутствует соответствующая опция, должен быть возвращен код HTTP 400. Все CDMI-совместимые системы облачного хранения должны реализовывать чтение опций, но поддержка функциональности, обозначаемой этими опциями, не обязательна.

Каждый объект данных, контейнер, домен и очередь системы CDMI должны иметь поле capabilitiesURI, которое содержит корректный URI объекта-опции. В пределах объекта-опции имя каждой опции имеет определенное значение, согласованно понимаемое провайдером облачного хранения и потребителем услуги.

Опции, определенные в рамках настоящего стандарта, описаны начиная с 12.1.1. Опции, определенные поставщиком услуг, не описаны в данном стандарте; их имена не должны начинаться с «cdmi_».

Рисунок 7 показывает иерархию опций реализации и то, как capabilitiesURI связывают объекты данных и контейнеры в дерево опций.

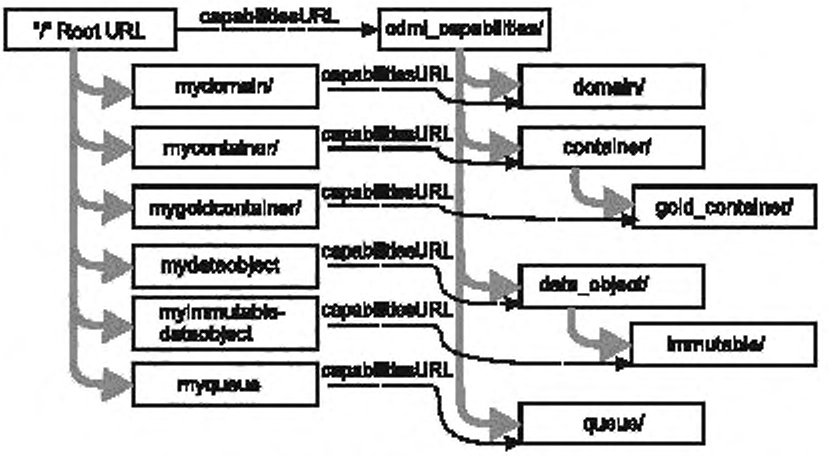

Рисунок 7 – Иерархия опций

Контейнер опций в дереве опций, с которым связан объект, определяется типом объекта и полей метаданных системы данных, присутствующих в данном объекте.

Пример – Опции контейнера без полей метаданных системы данных определяются набором опций «container».

Реализация CDMI может опционально связывать контейнер с набором опций «gold_container», если присутствует поле метаданных системы данных, и если оно установлено в определенное значение, например, если поле cdmi_data_redundancy равно «4». Это позволяет облачному провайдеру создавать профили полей метаданных системы данных и их значений.

Опции не имеют поля метаданных CDMI.

12.1.1 Общесистемные опции облачного хранения

В таблице 101 приведены общесистемные опции облачного хранилища. Эти опции содержатся в объекте опций, доступном через корневой URI (корневые опции).

Таблица 101 – Общесистемные опции

Имя опции	Тип	Определение
cdmi_domains	Строка JSON	Если существует и равна "true", означает, что облачная система хранения поддерживает домены. Если отсутствует, поле domainURI не должно присутствовать в теле сообщений-ответов; не должно быть URI cdmi_domains.
cdmi_export_cifs	Строка JSON	Если существует и равна "true", данная опция означает, что облачная система хранения поддерживает экспорты CIFS.
cdmi_dataobjects	Строка JSON	Если существует и равна "true", данная опция означает, что облачная система хранения поддерживает объекты данных.
cdmi_export_iscsi	Строка JSON	Если существует и равна "true", данная опция означает, что облачная система хранения поддерживает экспорты iSCSI.
cdmi_export_nfs	Строка JSON	Если существует и равна "true", данная опция означает, что облачная система хранения поддерживает экспорты по протоколу NFS.
cdmi_export_occi_iscsi	Строка JSON	Если существует и равна "true", данная опция означает, что облачная система хранения поддерживает экспорты OCCI/iSCSI.
cdmi_export_webdav	Строка JSON	Если существует и равна "true", данная опция означает, что облачная система хранения поддерживает экспорты WebDAV.
cdmi_metadata_maxitems	Строка JSON	Если присутствует, данная опция указывает максимальное количество определенных пользователем элементов метаданных. Если отсутствует, ограничений на количество элементов пользовательских метаданных нет.

Продолжение таблицы 101

Имя опции	Тип	Определение
cdmi_metadata_maxsize	Строка JSON	Если присутствует, данная опция указывает максимальный размер (в байтах) каждого элемента пользовательских метаданных на один объект. Если отсутствует, таких ограничений нет.
cdmi_metadata_maxtotalsize	Строка JSON	Если присутствует, данная опция указывает максимальный размер (в байтах) определенных пользователем метаданных, поддерживаемых облачной системой хранения. Если отсутствует, таких ограничений нет.
cdmi_notification	Строка JSON	Если существует и равна "true", данная опция означает, что облачная система хранения поддерживает очереди уведомлений.
cdmi_logging	Строка JSON	Если существует и равна "true", данная опция означает, что облачная система хранения поддерживает очереди журналов событий.
cdmi_query	Строка JSON	Если существует и равна "true", данная опция означает, что облачная система хранения поддерживает очереди запросов.
cdmi_query_regex	Строка JSON	Если существует и равна "true", данная опция означает, что облачная система хранения поддерживает запросы с регулярными выражениями.
cdmi_query_contains	Строка JSON	Если существует и равна "true", данная опция означает, что облачная система хранения поддерживает запросы с выражениями "contains".
cdmi_query_tags	Строка JSON	Если существует и равна "true", данная опция означает, что облачная система хранения поддерживает запросы с выражениями проверки на соответствие метке.
cdmi_query_value	Строка JSON	Если существует и равна "true", данная опция означает, что облачная система хранения поддерживает запросы поля значения.
cdmi_queues	Строка JSON	Если существует и равна "true", данная опция означает, что облачная система хранения поддерживает объекты-очереди.
cdmi_security_access_control	Строка JSON	Если существует и равна "true", данная опция означает, что облачная система хранения поддерживает ACL. Подробнее см. 12.1.3.
cdmi_security_audit	Строка JSON	Если существует и равна "true", данная опция означает, что облачная система хранения поддерживает журнал событий безопасности. Подробнее см. 20.3.
cdmi_security_data_integrity	Строка JSON	Если существует и равна "true", данная опция означает, что облачная система хранения поддерживает проверки целостности/подлинности данных. Подробнее см. 12.1.3.
cdmi_security_encryption	Строка JSON	Если существует и равна "true", данная опция означает, что облачная система хранения поддерживает фоновое шифрование данных. Подробнее см. 12.1.3.
cdmi_security_immutability	Строка JSON	Если существует и равна "true", данная опция означает, что облачная система хранения поддерживает неизменяемые данные и удержание данных. Подробнее см. 12.1.3.
cdmi_security_sanitization	Строка JSON	Если существует и равна "true", данная опция означает, что облачная система хранения поддерживает очистку данных/носителя. Подробнее см. 12.1.3.
cdmi_serialization_json	Строка JSON	Если существует и равна "true", данная опция означает, что облачная система хранения поддерживает JSON как формат сериализации.
cdmi_snapshots	Строка JSON	Если существует и равна "true", данная опция означает, что облачная система хранения поддерживает снимки состояния.
cdmi_references	Строка JSON	Если существует и равна "true", опция означает, что облачная система хранения поддерживает ссылки.
cdmi_object_move_from_local	Строка JSON	Если существует и равна "true", данная опция означает, что облачная система хранения поддерживает перемещение CDMI объектов в пределах той же системы хранения.
cdmi_object_move_from_ remote	Строка JSON	Если существует и равна "true", данная опция означает, что облачная система хранения поддерживает перемещение CDMI объектов из других систем хранения CDMI.

Окончание таблицы 101

Имя опции	Тип	Определение
cdmi_object_move_from_ID	Строка JSON	Если существует и равна "true", данная опция означает, что облачная система хранения поддерживает перемещение CDMI объектов без указания пути, по URI из /cdmi_objectid/ в пределах той же системы хранения. Это добавляет путь, позволяя доступ к объекту и по ID, и по пути.
cdmi_object_move_to_ID	Строка JSON	Если существует и равна "true", данная опция означает, что облачная система хранения поддерживает перемещение CDMI объектов, для которых задан путь, в URI из /cdmi_objectid/ в пределах той же системы хранения. Это удаляет путь, позволяя обращаться к объекту лишь по ID.
cdmi_object_copy_from_local	Строка JSON	Если существует и равна "true", данная опция означает, что облачная система хранения поддерживает копирование CDMI объектов в пределах той же системы хранения.
cdmi_object_copy_from_remote	Строка JSON	Если существует и равна "true", данная опция означает, что облачная система хранения поддерживает копирование CDMI объектов из других систем хранения CDMI.
cdmi_object_access_by_ID	Строка JSON	Если существует и равна "true", данная опция означает, что объекты могут быть доступны для чтения, изменения и удаления через "/cdmi_objectid/".
cdmi_post_dataobject_by_ID	Строка JSON	Если существует и равна "true", данная опция означает, что система позволяет добавлять новый объект данных по ID посредством POST в "/cdmi_objectid/".
cdmi_post_queue_by_ID	Строка JSON	Если существует и равна "true", данная опция означает, что система позволяет добавлять новый объект-очередь по ID посредством POST в "/cdmi_objectid/".
cdmi_deserialize_dataobject_by_ID	Строка JSON	Если существует и равна "true", данная опция означает, что система позволяет десериализацию сериализованных данных при создании нового объекта данных по ID посредством POST в "/cdmi_objectid/".
cdmi_deserialize_queue_by_ID	Строка JSON	Если существует и равна "true", данная опция означает, что система позволяет десериализацию сериализованной очереди при создании нового объекта-очереди по ID посредством POST в "/cdmi_objectid/".
cdmi_serialize_dataobject_to_ID	Строка JSON	Если существует и равна "true", данная опция означает, что система позволяет сериализовать объекты данных при создании нового объекта данных по ID посредством POST в "/cdmi_objectid/".
cdmi_serialize_domain_to_ID	Строка JSON	Если существует и равна "true", данная опция означает, что система позволяет сериализовать объекты-домены при создании нового объекта данных по ID посредством POST в "/cdmi_objectid/".
cdmi_serialize_container_to_ID	Строка JSON	Если существует и равна "true", данная опция означает, что система позволяет the сериализовать объекты-контейнеры при создании нового объекта данных по ID посредством POST в "/cdmi_objectid/".
cdmi_serialize_queue_to_ID	Строка JSON	Если существует и равна "true", данная опция означает, что система позволяет сериализовать объекты-очереди при создании нового объекта данных по ID посредством POST в "/cdmi_objectid/".
cdmi_copy_dataobject_by_ID	Строка JSON	Если существует и равна "true", данная опция означает, что система позволяет копировать существующий объект данных при создании нового объекта данных по ID посредством POST в "/cdmi_objectid/".
cdmi_copy_queue_by_ID	Строка JSON	Если существует и равна "true", данная опция означает, что система позволяет копировать существующий объект-очередь при создании нового объекта-очереди по ID посредством POST в "/cdmi_objectid/".
cdmi_create_reference_by_ID	Строка JSON	Если существует и равна "true", данная опция означает, что система позволяет создавать новую ссылку по ID: ссылка-потомок может быть создана посредством POST в "/cdmi_objectid/".

12.1.2 Опции метаданных системы хранения

В таблице 102 приведены опции метаданных системы хранения в облачном хранилище. Эти опции могут быть найдены среди опций для объектов-доменов, объектов-контейнеров, объектов-очередей. Подробное описание элементов метаданных системы хранения см. в 16.3.

Таблица 102 – Опции метаданных системы хранения

Имя опции	Тип	Определение
cdmi_acl	Строка JSON	Если существует и равна "true", данная опция означает, что облачная система хранения поддерживает ACL. Если реализация CDMI поддерживает ACL для управления доступом, общесистемная опция cdmi_security_access_control (см. таблицу 102 гл. 12.1.1) должна быть установлена в "true". В противном случае данная должна отсутствовать, что обозначет отсутствие поддержки управления доступом.
cdmi_size	Строка JSON	Если существует и равна "true", данная опция означает, что облачная система хранения должна создавать метаданные cdmi_size для каждого хранимого объекта.
cdmi_ctime	Строка JSON	Если существует и равна "true", данная опция означает, что облачная система хранения должна создавать метаданные cdmi_ctime для каждого хранимого объекта.
cdmi_atime	Строка JSON	Если существует и равна "true", данная опция означает, что облачная система хранения должна создавать метаданные cdmi_atime для каждого хранимого объекта.
cdmi_mtime	Строка JSON	Если существует и равна "true", данная опция означает, что облачная система хранения должна создавать метаданные cdmi_mtime для каждого хранимого объекта.
cdmi_acount	Строка JSON	Если существует и равна "true", данная опция означает, что облачная система хранения должна создавать метаданные cdmi_acount для каждого хранимого объекта.
cdmi_mcount	Строка JSON	Если существует и равна "true", данная опция означает, что облачная система хранения должна создавать метаданные cdmi_mcount для каждого хранимого объекта.

12.1.3 Опции метаданных системы данных

Таблица 103 определеяет опции, указывающие, какие элементы метаданных системы данных поддерживаются для объектов в облачном хранилище. Эти опции касаются объектов-доменов, объектов данных, объектов-контенеров и объектов-очередей. Описание элементов метаданных системы данных см. в 16.4 (таблица 117).

Таблица 103 – Опции метаданных системы данных

Имя опции	Тип	Определение
cdmi_ assignedsize	Строка JSON	Если облачная система хранения поддерживает метаданные системы данных cdmi_assignedsize (см. 16.4), должна присутствовать опция cdmi_assignedsize, установленная в строковое значение "true". Если эта опция отсутствует или присутствует, но установлена в "false", метаданные системы данных cdmi_assignedsize не должны использоваться.
cdmi_data_ redundancy	Строка JSON	Если облачная система хранения поддерживает метаданные систем данных cdmi_ data_redundancy (см. 16.4), опция cdmi_data_redundancy должна присутствовать и быть установлена в положительное числовое значение, отражающее максимальное значение, поддерживаемое сервером. Если эта опция отсутствует или присутствует, но равна пустой строке "", метаданные системы данных cdmi_data_redundancy не должны использоваться.
cdmi_data_ dispersion	Строка JSON	Если облачная система хранения поддерживает метаданные систем данных cdmi_ data_dispersion (см. 16.4), должна присутствовать опция cdmi_data_dispersion, уставновленная в "true". Если эта опция отсуствует или присутствует, но установлена в "false", метаданные системы данных cdmi_data_dispersion не должны использоваться.
cdmi_data_ retention	Строка JSON	Если облачная система хранения поддерживает метаданные систем данных cdmi_ retention_id и cdmi_retention_period (см. 16.4), опция cdmi_data_retention должна присутствовать и быть установлена в "true". Если эта опция отсутствует или присутствует, но установлена в "false", метаданные системы данных cdmi_retention_id и cdmi_retention_period не должны использоваться.
cdmi_data_ autodelete	Строка JSON	Если облачная система хранения поддерживает метаданные систем данных cdmi_data_autodelete (см. 16.4), опция cdmi_data_autodelete должна присутствовать и быть установлена в "true". Если эта опция отсутствует или присутствует, но установлена в "false", метаданные системы данных cdmi_data_autodelete не должны использоваться.

Продолжение таблицы 103

Имя опции	Тип	Определение
cdmi_data_holds	Строка JSON	Если облачная система хранения поддерживает метаданные систем данных cdmi_hold_id (см. 16.4), опция cdmi_data_holds должна присутствовать и быть установлена в "true". Если эта опция отсутствует или присутствует, но установлена в "false", метаданные системы данных, cdmi_data_holds не должны использоваться. Если а облачная система хранения поддерживает удержание для обеспечения неизменямости данных, общесистемная опция cdmi_security_immutability (см. таблицу 101 в 12.1.1) должна присутствовать и быть установлена в "true".
cdmi_encryption	Массив JSON	Если облачная система хранения поддерживает метаданные систем данных cdmi_encryption (см. 16.4), опция cdmi_encryption должна присутствовать и быть установлена в одно из значений, описанных в метаданных системы данных cdmi_encryption (см. 16.4). Если эта опция отсуствует или присутствует, но установлена равной пустому массиву JSON, метаданные системы данных cdmi_encryption не должны использоваться. Если облачная система хранения поддерживает фоновое шифрование, системная опция cdmi_security_encryption (см. таблицу 101 в 12.1.1) должна присутствовать и быть установлена в "true".
cdmi_geographic_placement	Строка JSON	Если облачная система хранения поддерживает метаданные систем данных cdmi_geographic_placement (см. 16.4), опция cdmi_geographic_placement должна присутствовать и быть установлена в "true". Если эта опция отсутствует или присутствует, но установлена в "false", метаданные системы данных cdmi_geographic_placement не должны использоваться.
cdmi_immediate_redundancy	Строка JSON	Если облачная система хранения поддерживает метаданные систем данных cdmi_immediate_redundancy (см. 16.4), опция cdmi_immediate_redundancy должна присутствовать и быть установлена в положительную числовую строку, отражающу максимальное значение, поддерживаемое сервером. Если эта опция отсутствует или присутствует, но установлена в "", метаданные системы данных cdmi_immediate_redundancy не должны использоваться.
cdmi_infrastructure_redundancy	Строка JSON	Если облачная система хранения поддерживает метаданные систем данных cdmi_infrastructure_redundancy (см. 16.4), опция cdmi_infrastructure_redundancy должна присутствовать и быть установлена в положительное числовое значение, отражающее максимальное значение, поддерживаемое сервером. Если эта опция отсутствует или присутствует, но равна пустой строке "", метаданные системы данных cdmi_infrastructure_redundancy не должны использоваться.
cdmi_latency	Строка JSON	Если облачная система хранения поддерживает метаданные систем данных cdmi_latency (см. 16.4), опция cdmi_latency должна присутствовать и быть установлена в "true". Если эта опция отсутствует или присутствует, но установлена в "false", метаданные системы данных cdmi_latency не должны использоваться..
cdmi_RPO	Строка JSON	Если облачная система хранения поддерживает метаданные систем данных cdmi_RPO (см. 16.4), опция cdmi_RPO должна присутствовать и быть установлена в "true". Если эта опция отсутствует или присутствует, но установлена в "false", метаданные системы данных cdmi_RPO не должны использоваться.
cdmi_RTO	Строка JSON	Если облачная система хранения поддерживает метаданные систем данных cdmi_RTO (см. 16.4), опция cdmi_RTO должна присутствовать и быть установлена в "true". Если эта опция отсутствует или присутствует, но установлена в "false", метаданные системы данных cdmi_RTO не должны использоваться.
cdmi_sanitization_method	Массив JSON	Если облачная система хранения поддерживает метаданные систем данных cdmi_sanitization_method (см. 16.4), опция cdmi_sanitization_method должна присутствовать и быть установлена в одно из значений, описанных в метаданных системы данных cdmi_sanitization_method (см. 16.4). Если эта опция отсуствует или присутствует, но установлена равной пустому массиву JSON, метаданные системы данных cdmi_sanitization_method не должны использоваться. Если облачная система хранения поддерживает очистку данных, общесистемная опция cdmi_security_sanitization (см. таблицу 101 в 12.1.1) должна присутствовать и быть установлена в "true".
cdmi_throughput	Строка JSON	Если облачная система хранения поддерживает метаданные систем данных cdmi_throughput (см. 16.4), опция cdmi_throughput capability должна присутствовать и быть установлена в "true". Если эта опция отсутствует или присутствует, но установлена в "false", метаданные систем данных cdmi_throughput не должны использоваться.

Окончание таблицы 103

Имя опции	Тип	Определение
cdmi_value_hash	Массив JSON	Если облачная система хранения поддерживает метаданные систем данных cdmi_value_hash (см. 16.4), опция cdmi_value_hash должна присутствовать и быть установлена в одно из значений, описанных в метаданных системы данных cdmi_value_hash (см. 16.4). Если эта опция отсутствует или присутствует, но установлена равной пустому массиву JSON, метаданные системы данных cdmi_value_hash не должны использоваться. Если облачная система хранения поддерживает хеширование значений, общесистемная опция cdmi_security_data_integrity (см. таблицу 101 в 12.1.1) должна присутствовать и быть установлена в "true".

12.1.4 Опции объекта данных

В таблице 104 приведены опции объекта данных в системе облачного хранения.

Таблица 104 – Опции объекта данных

Имя опции	Тип	Определение
cdmi_read_value	Строка JSON	Если существует и равна "true", данная опция означает, что значение объекта может быть прочитано.
cdmi_read_value_range	Строка JSON	Если существует и равна "true", данная опция означает, что может быть прочитан диапазон байтов значения объекта.
cdmi_read_metadata	Строка JSON	Если существует и равна "true", данная опция означает, что метаданные объекта могут быть прочитаны.
cdmi_modify_value	Строка JSON	Если существует и равна "true", данная опция означает, что значение объекта может быть изменено.
cdmi_modify_value_range	Строка JSON	Если существует и равна "true", данная опция означает, что может быть изменен диапазон байтов значения объекта.
cdmi_modify_metadata	Строка JSON	Если существует и равна "true", данная опция означает, что метаданные объекта могут быть изменены.
cdmi_modify_deserialize_dataobject	Строка JSON	Если существует и равна "true", данная опция означает, что при изменении объекта допустима десериализация сериализованного объекта данных.
cdmi_delete_dataobject	Строка JSON	Если существует и равна "true", данная опция означает, что объект может быть удален.

12.1.5 Опции контейнеров

В таблице 105 приведены опции объекта-контейнера в системе облачного хранения.

Таблица 105 – Опции объекта-контейнера

Имя опции	Тип	Определение
cdmi_list_children	Строка JSON	Если существует и равна "true", данная опция означает, что может быть получен список потомков контейнера.
cdmi_list_children_range	Строка JSON	Если существует и равна "true", данная опция означает, что может быть получен список потомков контейнера как диапазон.
cdmi_read_metadata	Строка JSON	Если существует и равна "true", данная опция означает, что метаданные контейнера могут быть прочитаны.
cdmi_modify_metadata	Строка JSON	Если существует и равна "true", данная опция означает, что метаданные контейнера могут быть изменены.
cdmi_modify_deserialize_container	Строка JSON	Если существует и равна "true", данная опция означает, что при изменении контейнера допускается десериализация сериализованного контейнера.
cdmi_snapshot	Строка JSON	Если существует и равна "true", данная опция означает, что контейнер допускает создание нового снимка состояния.
cdmi_serialize_dataobject	Строка JSON	Если существует и равна "true", данная опция означает, что объект может быть сериализован.
cdmi_serialize_container	Строка JSON	Если существует и равна "true", данная опция означает, что содержимое контейнера и его потомков может быть сериализовано.

Окончание таблицы 105

Имя опции	Тип	Определение
cdmi_serialize_queue	Строка JSON	Если существует и равна "true", данная опция означает, что очередь может быть сериализована.
cdmi_serialize_domain	Строка JSON	Если существует и равна "true", данная опция означает, что домен и его потомки могут быть сериализованы.
cdmi_deserialize_container	Строка JSON	Если существует и равна "true", данная опция означает, что контейнер допускает десериализацию сериализованных контейнеров и связанных сериализованных потомков в данный контейнер
cdmi_deserialize_queue	Строка JSON	Если существует и равна "true", данная опция означает, что контейнер допускает десериализацию сериализованной очереди в контейнер.
cdmi_deserialize_dataobject	Строка JSON	Если существует и равна "true", данная опция означает, что контейнер допускает десериализацию сериализованных объектов данных в контейнер.
cdmi_create_dataobject	Строка JSON	Если существует и равна "true", данная опция означает, что контейнер допускает добавление нового объекта.
cdmi_post_dataobject	Строка JSON	Если существует и равна "true", данная опция означает, что контейнер допускает добавление нового объекта посредством POST.
cdmi_post_queue	Строка JSON	Если существует и равна "true", данная опция означает, что контейнер допускает добавление новой очереди посредством POST.
cdmi_create_container	Строка JSON	Если существует и равна "true", данная опция означает, что контейнер допускает создание нового контейнера посредством PUT.
cdmi_create_queue	Строка JSON	Если существует и равна "true", данная опция означает, что контейнер допускает создание новых очередей.
cdmi_create_reference	Строка JSON	Если существует и равна "true", данная опция означает, что контейнер допускает создание новых ссылок на потомка посредством PUT.
cdmi_export_container_cifs	Строка JSON	Если существует и равна "true", данная опция означает, что контейнер может быть экспортирован как файловая система через CIFS.
cdmi_export_container_nfs	Строка JSON	Если существует и равна "true", данная опция означает, что контейнер может быть экспортирован как файловая система через NFS.
cdmi_export_container_iscsi	Строка JSON	Если существует и равна "true", данная опция означает, что контейнер может быть экспортирован как файловая система через iSCSI.
cdmi_export_container_occi	Строка JSON	Если существует и равна "true", данная опция означает, что контейнер может быть экспортирован как файловая система через OCCI.
cdmi_export_container_webdav	Строка JSON	Если существует и равна "true", данная опция означает, что контейнер может быть экспортирован как файловая система через WebDAV.
cdmi_delete_container	Строка JSON	Если существует и равна "true", данная опция означает, что контейнер может быть удален.
cdmi_move_container	Строка JSON	Если существует и равна "true", данная опция означает, что контейнер может быть перемещен в другой контейнер.
cdmi_copy_container	Строка JSON	Если существует и равна "true", данная опция означает, что контейнер может быть скопирован в другой контейнер.
cdmi_move_dataobject	Строка JSON	Если существует и равна "true", данная опция означает, что объект данных может быть перемещен в контейнер.
cdmi_copy_dataobject	Строка JSON	Если существует и равна "true", данная опция означает, что объект данных может быть скопирован в контейнер.

12.1.6 Опции объекта-домена

В таблице 106 приведены опции объекта-домена в системе облачного хранения. (Все опции определяют действия, выполняемые посредством операций с типом содержимого CDMI).

Таблица 106 – Опции для объектов-доменов

Имя опции	Тип	Определение
cdmi_create_domain	Строка JSON	Если существует и равна "true", данная опция означает, что домен поддерживает создание вложенных доменов.
cdmi_delete_domain	Строка JSON	Если существует и равна "true", данная опция означает, что домен может быть удален.

Окончание таблицы 106

Имя опции	Тип	Определение
cdmi_domain_ summary	Строка JSON	Если существует и равна "true", данная опция означает, что домен поддерживает сводки.
cdmi_domain_ members	Строка JSON	Если существует и равна "true", данная опция означает, что домен поддерживает доменное управление пользователями.
cdmi_list_children	Строка JSON	Если существует и равна "true", данная опция означает, что возможно получить список потомков домена.
cdmi_read_ metadata	Строка JSON	Если существует и равна "true", данная опция означает, что метаданные домена могут быть прочитаны.
cdmi_modify_ metadata	Строка JSON	Если существует и равна "true", данная опция означает, что метаданные домена могут быть изменены.
cdmi_modify_ deserialize_domain	Строка JSON	Если существует и равна "true", данная опция означает, что домен допускает десериализацию сериализованного объекта-домена при изменении данного домена.
cdmi_copy_domain	Строка JSON	Если существует и равна "true", данная опция означает, что домен может быть скопирован (через PUT) по-другому URI.
cdmi_deserialize_ domain	Строка JSON	Если существует и равна "true", данная опция означает, что домен допускает десериализацию сериализованных доменов и связанных сериализованных вложенных доменов в данный домен.

12.1.7 Опции объекта-очереди

В таблице 107 приведены опции объекта-очереди в системе облачного хранения.

Таблица 107 – Опции для объектов-очередей

Имя опции	Тип	Определение
cdmi_read_value	Строка JSON	Если существует и равна "true", данная опция означает, что значение очереди может быть прочитано.
cdmi_read_metadata	Строка JSON	Если существует и равна "true", данная опция означает, что метаданные очереди могут быть прочитаны.
cdmi_modify_value	Строка JSON	Если существует и равна "true", данная опция означает, что значение очереди может быть изменено.
cdmi_modify_metadata	Строка JSON	Если существует и равна "true", данная опция означает, что метаданные очереди могут быть изменены.
cdmi_modify_deserialize_queue	Строка JSON	Если существует и равна "true", данная опция означает, что очередь допускает десериализацию сериализованной очереди при изменении данной очереди.
cdmi_delete_queue	Строка JSON	Если существует и равна "true", данная опция означает, что очередь может быть удалена.
cdmi_move_queue	Строка JSON	Если существует и равна "true", данная опция означает, что очередь может быть перемещена по-другому URI.
cdmi_copy_queue	Строка JSON	Если существует и равна "true", данная опция означает, что очередь может быть скопирована по-другому URI.
cdmi_reference_queue	Строка JSON	Если существует и равна "true", данная опция означает, что на очередь может ссылаться другая очередь.

12.1.8 Представления опций объектов

Представления в данном разделе показаны с использованием нотации JSON. И клиенты, и серверы должны поддерживать UTF-8 представление JSON. В телах сообщений-запросов и ответов поля JSON могут указываться и возвращаться в любом порядке за исключением (при наличии) полей childrenrange и children, которые указываются последними и в данном порядке.

12.2 Чтение опций объекта с использованием типа содержимого CDMI

12.2.1 Обзор

Для чтения всех полей существующего объекта-опции следует выполнить запрос:

GET <root URI>/cdmi_capabilities/<Capability>/<TheCapability>/

Для чтения одного или нескольких выбранных полей существующего объекта-опции следует выполнить один из запросов:

GET <root URI>/cdmi_capabilities/<Capability>/<TheCapability>/?<fieldname>;<fieldname>

GET <root URI>/cdmi_capabilities/<Capability>/<TheCapability>/?children:<range>

где:

- <root URI> путь к облаку CDMI;
- <Capability> неотрицательное количество промежуточных контейнеров опций;
- <TheCapability> имя объекта для чтения;
- <fieldname> имя поля;
- <range> числовой диапазон в списке потомков.

Объект также должен быть доступен как <root URI>/cdmi_objectid/<objectID>/.

12.2.2 Опции

Следующие опции описывают поддерживаемые операции при чтении существующего объекта-опции:

- все реализации CDMI должны позволять клиентам чтение всех полей объектов-опций.

12.2.3 Заголовки запроса

Заголовки HTTP запроса на чтение объекта-опции CDMI с использованием типа содержимого CDMI приведены в таблице 108.

Таблица 108 – Заголовки запроса – чтение объекта-опции с использованием типа содержимого CDMI

Заголовок	Тип	Описание	Требование
Accept	Строка заголовка	"application/cdmi-container" или совместимое значение согласно 5.13.2	Опционально
X-CDMI-SpecificationVersion	String Array	Список версий, поддерживаемых клиентом, разделенных запятыми, например "1.0.2, 1.5, 2.0"	Обязательно

12.2.4 Тело сообщения-запроса

Тело в сообщении-запросе должно отсутствовать.

12.2.5 Заголовки ответа

Заголовки HTTP ответа для чтения объекта-опции CDMI с использованием типа содержимого CDMI приведены в таблице 109.

Таблица 109 – Заголовки ответа – чтение объекта-опции с использованием типа содержимого CDMI

Заголовок	Тип	Описание	Требование
X-CDMI-Specification-Version	Строка заголовка	Сервер должен вернуть наибольший номер версии, поддерживаемой и сервером, и клиентом, например, "1.0.2". Если сервер не поддерживает ни одной версии, поддерживаемой клиентом, сервер должен вернуть код состояния 400 Bad Request.	Обязательно
Content-Type	Строка заголовка	"application/cdmi-capability"	Обязательно

12.2.6 Тело сообщения-ответа

Поля тела сообщения-ответа для чтения объекта-опции CDMI с использованием типа содержимого CDMI приведены в таблице 110.

Таблица 110 – Тело сообщения-ответа – чтение объекта-опции с использованием типа содержимого CDMI

Имя поля	Тип	Описание	Требование
objectType	Строка JSON	"application/cdmi-capability"	Обязательно
objectID	Строка JSON	ID объекта	Обязательно

Окончание таблицы 110

objectName	Строка JSON	Имя объекта	Обязательно
parentURI	Строка JSON	URI родительского объекта	Обязательно
parentID	Строка JSON	ID объекта родительского контейнера	Обязательно
capabilities	Объект JSON	Опции, которые поддерживает соответствующий объект. Опции объекта "/cdmi_capabilities/" – всесистемные опции. Опции в дочерних объектах корневого контейнера "/cdmi_capabilities/" соответствуют опциям определенного подмножества объектов. Каждая опция выражается строкой JSON.	Обязательно
childrenrange	Строка JSON	Опции-потомки, выраженные диапазоном. Если запрошен диапазон опций-потомков, данное поле указывает на потомков, возвращенных как диапазон.	Обязательно
children	Массив JSON	Имена дочерних объектов-опций. Для опций корневого контейнера они включают "domain/", "container/", "dataobject/" и "queue/". Внутри каждого из этих объектов-опций имеются более специальные профили опций, определенные облачной системой хранения.	Обязательно

Если в запросе GET указаны отдельные поля, в ответе следует возвращать только эти поля. Опциональные запрошенные поля, отсутствующие в объекте, опускаются в ответе.

12.2.7 Статус запроса

В таблице 111 приведены коды состояний HTTP, возникающие при чтении объектов-опций с использованием типа содержимого CDMI.

Таблица 111– Коды состояний HTTP – чтение объекта-опции с использованием типа содержимого CDMI

Статус HTTP	Описание
200 OK	Содержимое объекта-опции возвращено в ответе.
400 Bad Request	Запрос содержит неверные параметры или имена полей
401 Unauthorized	Неверные данные аутентификации/авторизации.
403 Forbidden	Клиент не обладает правами для выполнения данного запроса.
404 Not Found	Ресурс не найден по указанному URI.
406 Not Acceptable	Сервер не может предоставить объект, типизованный как обозначено в заголовке Accept.

12.2.8 Примеры

Примеры

1 Применение GET к URI корневого контейнера опций для чтения всех полей контейнера:

```
GET /cdmi_capabilities/ HTTP/1.1
Host: cloud.example.com
Accept: application/cdmi-capability
X-CDMI-Specification-Version: 1.0.2
```

Будет получен следующий ответ.

```
HTTP/1.1 200 OK
Content-Type: application/cdmi-capability X-CDMI-Specification-Version: 1.0.2
{
«objectType» : «application/cdmi-capability»,
«objectID» : «00007E7F00104BE66AB53A9572F9F51E»,
«objectName» : «cdmi_capabilities/»,
«parentURI» : «/»,
«parentID» : «00007E7F0010128E42D87EE34F5A6560»,
«capabilities» : {
«cdmi_domains» : «true»,
«cdmi_export_nfs» : «true»,
«cdmi_export_iscsi» : «true»,
«cdmi_queues» : «true»,
«cdmi_notification» : «true»,
«cdmi_query» : «true»,
```

«cdmi_metadata_maxsize» : «4096»,
«cdmi_metadata_maxitems» : «1024»
},
«childrenrange» : «0-3»,
«children» : [
«domain/»,
«container/»,
«dataobject/»,
«queue/»
]129 }

2 Применение GET к URI корневого контейнера опций для чтения содержимого и потомков контейнера:

GET /cdmi_capabilities/?capabilities;children HTTP/1.1
Host: cloud.example.com
Accept: application/cdmi-capability
X-CDMI-Specification-Version: 1.0.2
Будет получен следующий ответ.
HTTP/1.1 200 OK
Content-Type: application/cdmi-capability X-CDMI-Specification-Version: 1.0.2
{
«capabilities» : {
«cdmi_domains» : «true»,
«cdmi_export_nfs» : «true»,
«cdmi_export_iscsi» : «true»,
«cdmi_queues» : «true»,
«cdmi_notification» : «true»,
«cdmi_query» : «true»,
«cdmi_metadata_maxsize» : «4096»,
«cdmi_metadata_maxitems» : «1024»
},
«children» : [
«domain/»,
«container/»,
«dataobject/»,
«queue/»
]156 }

3 Применение GET к URI корневого контейнера опций для чтения первых двух потомков контейнера:

GET /cdmi_capabilities/?childrenrange;children:0-1 HTTP/1.1
Host: cloud.example.com
Accept: application/cdmi-capability
X-CDMI-Specification-Version: 1.0.2
Будет получен следующий ответ.
HTTP/1.1 200 OK
Content-Type: application/cdmi-capability X-CDMI-Specification-Version: 1.0.2
{
«childrenrange» : «0-1»,
«children» : [
«domain/»,
«container/»
]
}

13 Экспортируемые протоколы

13.1 Обзор

Контейнеры CDMI™ доступны не только через путь к данным, но также и через другие протоколы. Такой доступ особенно полезен для использования CDMI в качестве интерфейса хранения для облачных вычислительных сред (см. рисунок 8).

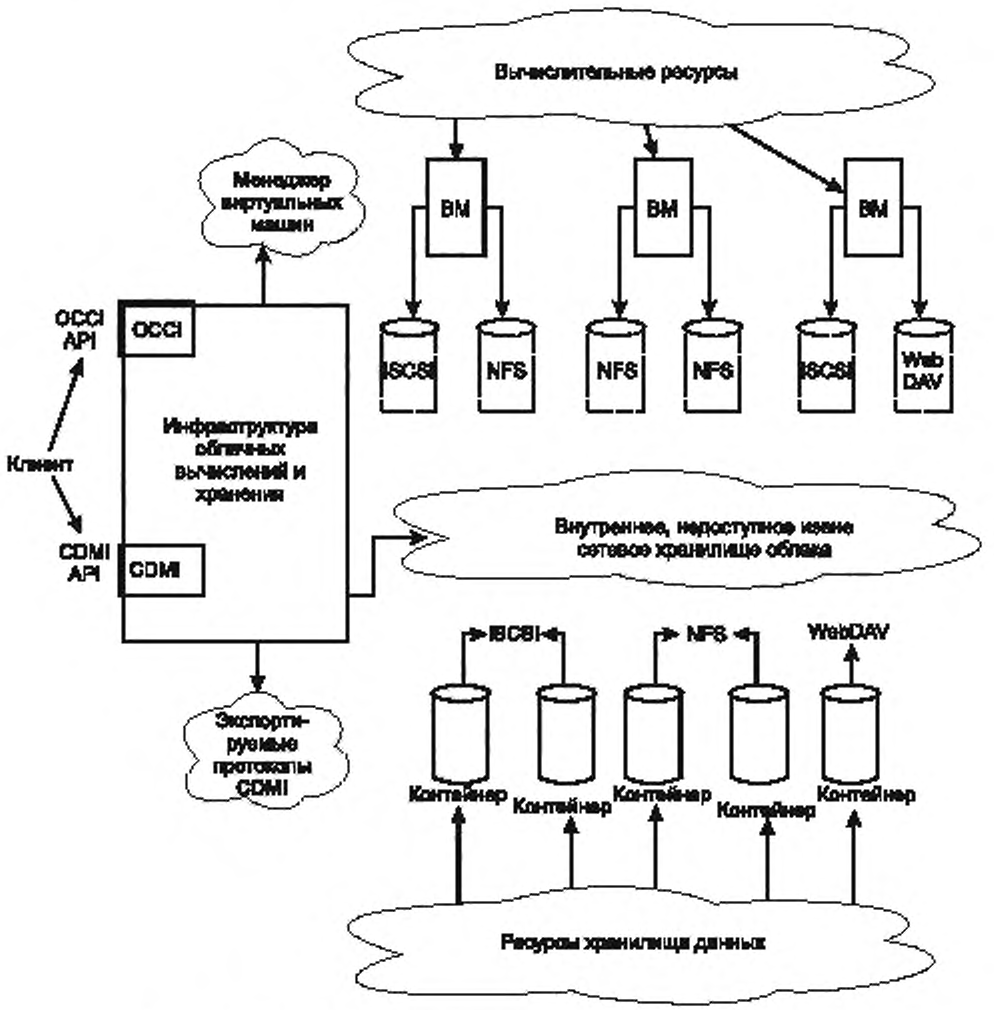

Рисунок 8 – CDMI и OCCI в интегрированной облачной вычислительной среде

Контейнеры CDMI, представленные различными протоколами, могут использоваться виртуальными машинами облачной вычислительной среды как виртуальные диски на каждом госте. Показанная на рисунке инфраструктура облачных вычислений реализует как интерфейс открытого облачного компьютера (Open Cloud Computer Interface, OCCI), так и интерфейс CDMI. Вместе с внутренним знанием сети и соответствия дисков, установленного менеджером виртуальных машин, эта инфраструктура может ассоциировать контейнеры CDMI с гостевыми ОС посредством подходящего экспортируемого протокола.

Для поддержки экспортируемых протоколов и улучшения их функциональной совместимости с CDMI, модель CDMI предоставляет тип протокола экспорта, который содержит информацию, полученную через интерфейс OCCI. Кроме того, OCCI предоставляет тип хранения, который соответствует контейнеру CDMI, экспортируемому по особому протоколу OCCI. Клиент обоих интерфейсов выполняет операции, совмещающие обе архитектуры, включая следующие:

– Клиент создает контейнер CDMI через интерфейс CDMI и экспортирует его по протоколу OCCI. В результате возвращается objectID контейнера CDMI;

– Клиент создает виртуальную машину через интерфейс OCCI и присоединяет том хранения типа CDMI, используя objectID и тип протокола. В результате возвращается ID виртуальной машины OCCI;

– Клиент изменяет структуру протокола экспорта CDMI контейнера, включая в него ID виртуальной машины OCCI, что предоставляет виртуальной машине доступ к контейнеру;

– Клиент запускает виртуальную машину через интерфейс OCCI.

13.2 Структура экспортируемого протокола

Экспорт контейнера через протокол пути к данным, отличный от CDMI, осуществляется созданием и обновлением контейнера и предоставлением одной или нескольких структур экспортируемого протокола, для каждого такого протокола. В данном международном стандарте все такие протоколы называются «сторонними» протоколами. Реализация сторонних протоколов должна быть обозначена значениями «true» общесистемных опций, см. 12.1.1, имена которых должны всегда начинаться с «cdmi_export_».

Элементы структуры протокола экспорта включают:

– используемый протокол;

– идентификатор контейнера в соответствии со стандартом протокола;

– интернет-домен сервера имен протокола для обслуживаемого клиента;

– список тех, кто может монтировать этот контейнер по данному протоколу, определенный в соответствии со стандартом протокола или (опционально) посредством протокола установления соответствия имен (см. 13.2.1) с указанием пользователей или названий групп CDMI;

– обязательные параметры экспорта протокола;

– опциональные параметры экспорта протокола;

– параметры управления экспортом.

Настоящий стандарт определяет структуры экспорта в нотации JSON для нескольких широко известных сторонних протоколов. Доступ к контейнеру по нескольким протоколам одновременно зависит от функции установления соответствия имен пользователей и групп. Тем не менее, установление соответствия имен не является необходимым, если CDMI используется только для подготовки контейнера, с которым работает исключительно сторонний протокол.

Реализации, которые поддерживают аутентификацию и авторизацию доступа к объектам CDMI как через CDMI, так и через сторонние протоколы, нуждается в поддержке функции безопасности для объектов. Многочисленные соответствующие методы включают:

– определения и принятие схемы безопасности и отображения всех запросов в эту схему. Реализации CDMI, принимающие такую схему, должны использовать процедуру установления соответствия имен, так как:

a) оно проще для управления администраторами, чем прямое установление соответствия идентификаторов;

b) кроме того, желательно, чтобы различные реализации CDMI вели себя схожим образом в этом отношении.

Это означает, что для имени принципала входящего запроса устанавливается соответствующее имя принципала в домене безопасности, а затем полученный идентификаторов принципала используется для авторизации;

– каждый протокол может устанавливать собственную систему безопасности; это означает, что объект может быть доступен конкретному пользователю только через один из протоколов, но не через любой;

– использование схемы безопасности последнего протокола, который использовался для настройки разрешения доступа к объекту. Этот метод также требует устанвления соответствия между именем принципала входящего запроса и именем принципала из домена безопасности объекта. Как и в первом случае, сервер должен использовать процедуру установления соответствия имен для авторизации пользователя согласно списку ACL объекта.

CDMI не определяет, какой метод будет использован. Он, однако, определяет, как пользователи и группы должны быть отображаться между протоколами.

13.2.1 Установка соответствия имен между CDMI и другим протоколом

Если клиенту необходимо ограничить экспорт через сторонние протоколы, разрешив подключение лишь отдельным пользователям или группам, может быть необходимо передать информацию об установлении соответствия имен групп и пользователей на сервер. Такая информация также необходима для доступа к контейнеру по нескольким протоколам (например, CDMI, и NFS). Соответствие устанавливается следующим образом.

1 При создании общего доступа к контейнеру CDMI сервер должен использовать подходящий механизм, например вызов WmiClass.Create( ) оболочки Powershell в среде Windows или /etc/exports в среде Unix, для ограничения разрешенных подключений от других серверов в соответствии с инструкциями в строке «hosts» свойства «exports». Синтаксис строки hosts соответствует синтаксису / etc/exports в системе Linux, причем строка кодируется в строку JSON. Если сервер CDMI неспособен ограничить подключения согласно указаниям в строке hosts, должна возникать ошибка, но успех или отказ операции зависит от реализации.

2 При поступлении через сторонний протокол запроса, требующего использования имени принципала CDMI, необходимо запросить контроллер стороннего домена, к которому принадлежит сторонний сервер, какому имени принципала соответствует идентификатор пользователя, пришедший в запросе. Отказ в предоставлении имени принципала приведет к отказу в обработке запроса.

3 В списке соответствий имен пользователей для этого протокола необхдимо найти запись с именем приниципала, полученного от контроллера стороннего домена (подробнее об операции поиска см. 13.2.3). Если запись не найдена, запрос должен быть отклонен. Результат поиска может храниться в той же записи кеша, что и информация с предыдущего шага.

4 Имя принципала CDMI из первой подходящей записи из списка соответствия пользователей протокола используется для авторизации пользовательского запроса посредством механизма безопасности протокола, который отвечает за доступ к объекту.

13.2.1.1 Опции

Следующие опции описывают допустимые операции над существующим контейнером.

– общесистемная опция экспорта через некоторый протокол обозначается наличием опции cdmi_<protocol>_export в метаданных системного уровня (например, если «cdmi_nfs_export» равно «true», возможен экспорт контейнера через NFS). Если соответствующее значение равно “false” или не установлено, попытки экспорта контейнера через данный протокол должны быть неуспешными;

– поддержка возможности экспорта существующего объекта-контейнера через некоторый сторонний протокол обозначается присутствием опции cdmi_<protocol>_export у данного контейнера. Значение опции по умолчанию (если не установлено) равно «true».

13.2.1.2 Домены

Доменное имя, соответствующее каждому экспорту, должно быть задано как строка JSON в дочернем элементе «domain» спецификации протокола экспорта. Если этот элемент отсутствует, доменное имя считается совпадающим с именем сервера, на котором функционирует реализация CDMI.

13.2.1.3 Кеширование

Обращение к контроллеру стороннего домена может быть довольно дорогим, особенно в случае протоколов без запоминания состояния (например, NFS v3), когда поиск теоретически необходим почти для каждой операции. Должна быть возможность кешировать результаты поиска. Рекомендуемое время жизни записи в кеша пользователей 30 мин. В реализациях следует использовать такое время или меньшее, если это возможно. Сервер должен сбрасывать этот кеш в случае изменения в метаданных экспорта, относящихся к данному протоколу. Клиент может запросить сброс кеша посредством чтения данных о соответствии имен пользователей для одного или нескольких протоколов и последующей записи без изменения. Сервер должен сбросить кеши соответствия имен пользователей при обработке запроса на перезапись (для всех протоколов, которых касается перезаписываемая информация).

Для авторизации от имени группы для операций посредством стороннего протокола должен выполняться аналогичный поиск. Для нахождения всех групп, к которым принадлежит данный пользователь, могут потребоваться более одного запроса (например, обычная ситуация, когда пользователь NFS является членом полудесятка групп). Кеш имен групп может использоваться для снижения стоимости таких поисков. Рекомендуемое время жизни кеша имен групп составляет 12 ч; в конкретных реализациях это время должно быть таким или менее, если возможно. Клиент может запросить сброс кеша посредством чтения данных о соответствии имен групп для одного или нескольких протоколов и последующей записи без изменения. Сервер должен сбросить кеши соответствия групп пользователей при обработке запроса на перезапись (для всех протоколов, которых касается перезаписываемая информация).

13.2.1.4 Группы

Установка соответствия имен групп для каждого стороннего протокола должна быть описана в поле groupname спецификации протокола экспорта; его синтаксис аналогичен синтаксису поля username.[1]

[1] Данная информация требуется только при экспорте контейнера.

13.2.1.5 Пример

Пример – PUT /MyContainer HTTP/1.1
Host: cloud.example.com
Accept: application/cdmi-container
Content-Type: application/cdmi-container
X-CDMI-Specification-Version: 1.0
{
«exports» : { «nfs» : {
«hosts» : { «.mycollege.edu», «derf.cs.myuni.edu» },*
«domain» : «lab.mycollege.edu»,
«usermap» : {
{ <cdminame>, <map>, <nfsname> },
{ «jimsmith», «<-->», «jims» },
{ [ordered list of CDMIname/operator/NFSname triples] },
{ «», «<-->», «*» }*
}
«groupmap» : {
{ «admins», «<-», «wheel» },
{ «everyone», «<-», «» }*
}
}
«cifs» : {
«hosts» : «»,*
«domain» : «lab.mycollege.edu»,
«usermap» : {
{ «jimsmith», «<-->», «james.smith» }
{ [ordered list of CDMIname/operator/NFSname triples] },
{ «», «<-->», «*» }*
}
«groupmap» : {
{ «admins», «<-», «Administrators» },
{ «everyone», «<-», «» }*
}
}
}155 }
Будет получен следующий ответ.
HTTP/1.1 200 OK
Content-Type: application/cdmi-container
X-CDMI-Specification-Version: 1.0
{
«objectURI» : «/Containers/MyContainer/»,
«objectID» : «00007E7F00100C435125A61B4C289455»,
«objectName» : «MyContainer/»,
«parentURI» : «/Containers/»,
«parentID» : «00007E7F0010D538DEEE8E38399E2815»,
«domainURI» : «/cdmi_domains/MyDomain/»,
«capabilitiesURI» : «/cdmi_capabilities/container/»,
«completionStatus» : «complete»,
«metadata» : { ... },
«exports» : { <exports as listed in request> }
}

13.2.2 Пользователи-администраторы

По умолчанию, следующие пользователи эквивалентны и должны рассматриваться как суперпользователи или администраторы:

- root (Unix/NFS/LDAP);
- Администратор/Administrator (Windows/AD/CIFS);
- владелец домена (CDMI).

Серверы должны автоматически ставить этих пользователей в соответствие суперпользователю (root) соответствующего протокола, если иное не указано в списках соответствия имен пользователей.

Так как автоматическое установление имен не соответствует строгим стандартам безопасности, сервер должен перекрыть эти встроенные записи, соответствующие одному или нескольким суперпользователям.

Пример – Суперпользователь отображается в nobody, а всем прочим пользователям в домене NFS соответствует то же имя, что и в домене CDMI.

```
PUT /MyContainer HTTP/1.1
Host: cloud.example.com
Accept: application/vnd.org.snia.cdmi.container+json
Content-Type: application/vnd.org.snia.cdmi.container+json X-CDMI-Specification-Version: 1.0
{
«exports» : {      «nfs» : {
«usermap» : {
{ «nobody», «<-», «root» },
{ «*», «<-->», «*» }
}
}
}198     }
```

Соответствие разрешений

К сожалению, установки разрешений файловых протоколов, невозможно отобразить напрямую. Например, и NFSv4 ACL, и Windows ACL, и POSIX ACL, и параметры NFSv3 позволяют выражать условия безопасности, которые невозможно представить в трех других механизмах, за исключением NFSv3, в котором набор параметров безопасности наименьший. Таким образом, главная задача состоит в построении соответствия между CDMI ACL, где можно указать самые разные разрешения, и более ограниченными в этом смысле системами, например, NFSv3, для отображения пользователям.

Так как существует несколько различных возможностей установить соответствия между разрешениями/ACL и CDMI ACL, настоящий стандарт не устанавливает единой рекомендации. Тем не менее, конкретные способы установления соответствия имен пользователей и групп между доменами должны следовать механизму, описанному в 13.2.3.

13.2.3 Правила синтаксиса и проверки соответствия имен и групповых имен

Грамматика BNF для установления соответствия имен выглядит следующим образом:

name_mapping_list = protocol protocol mapping_list
protocol = «cdmi» | «nfs» | «cifs» | «ldap»
mapping_list = name mapping_operator name
name = pattern | utf8_name | quoted_utf8_name
quoted_utf8_name = « utf8_name «
utf8_name = <допустимая последовательность utf8 символов, не включающая «,',\,/,:,*,?>
pattern = <utf8_name> * | *
mapping_operator = «<--» | «<-->» | «-->»

Запись соответствия имен состоит из пары имен, разделенных оператором направления. Так как большинство сред используют одинаковые имена пользователей и групп в административных доменах, наиболее распространен картирующий оператор « * <--> * «, который устанавливает соответствие каждого имени такому же имени стороннего протокола, и наоборот. Рекомендуется делать эту запись единственной записью списка соответствия по умолчанию и последней записью сложных списков соответствия.

CDMI специфицирует соответствие по шаблонам имен, но требуется лишь совпадение префикса. Символ « * » на конце символьной строки замещает произвольное число любых непробельных символов.

Проверка соответствия имен должна проводиться по порядку; если найдено соответствие, возвращается найденный результат, и другие проверки не проводятся.

Если соответствия не найдено, результат зависит от системы. Тем не менее, рекомендуется, чтобы серверы либо запрещали доступ, либо относили бы пользователя к «анонимному» («anonymous») или его эквиваленту в целевом протоколе. Кроме того, рекомендуется создавать запись для специального пользователя «EVERYONE@».

13.3 Обнаружение и подключение контейнеров через сторонние протоколы

Клиентам требуется механизм обнаружения экспортированных контейнеров, которые можно использовать для подключения (mount). Это проводится посредством выполнения операции GET к дочернему элементу «exports» контейнера.

Обзор:

Для чтения всех экспортов для существующего объекта-контейнера следует выполнить запрос:

GET <root URI>/<ContainerName>/<TheContainerName>/?exports

Для чтения отдельных экспортов для существующего объекта-контейнера следует выполнить запрос:

GET<rootURI>/<ContainerName>/<TheContainerName>/?exports:protocol=<protocol>,user=<user>,verbose=»false»

где:

- <root URI> путь к облаку CDMI;
- <ContainerName> неотрицательное число промежуточных контейнеров;
- <TheContainerName> имя указанного контейнера наивысшего уровня, для которого возможен экспорт,
- <protocol> имя протокола, которым ограничен экспорт. Этот параметр необязателен; если он опущен, он принимается равным «all», тогда возвращается информация обо всех протоколах для дальнейшей фильтрации;
- <user> имя пользователя CDMI, который подключает ресурс. Этот параметр необязателен, и по умолчанию равен имени владельца контейнера. Если параметр не пустой, серверы должны фильтровать список экспорта, чтобы включить только экспорты, которые могут быть выполнены пользователем с учетом ограничений протокола экспорта;
- <verbose> опциональный параметр, указывающий на желание получить максимальную информацию об экспортах. Если присутствует, может иметь значения «true» или «false» (по умолчанию - «false»). Если равен «true», сервер должен вернуть дополнительную информацию о контейнере, которая содержится в его элементе «exports». Количество возвращаемой информации зависит от реализации, так как разработчики серверных решений вынуждены искать баланс между потребностями клиентов и требованиями безопасности.

13.4 Экспортный протокол NFS

Чтобы экспортировать контейнер через NFS, необходима информация о том, какая серверная реализация будет выполнять экспорт. Обычно эта информация содержится в файле /etc/exports или его эквиваленте на сервере. Администраторы должны иметь в виду, что в этот файл могут автоматически добавляться строки для каждого экспортируемого CDMI контейнера.

Необходимые элементы структуры экспорты NFS включают:

- «protocol». Запрошенный протокол может быть «NFSv3», «NFSv4», «NFSv4.1» или более поздняя версия NFS, описанная в основном стандарте IETF RFC. Версия 2 протокола NFS не поддерживается в модели CDMI;
- «exportpath». Путь, куда следует направить экспорт. Это должна быть строка UTF8 в форме [<server>]:/<path>, где <server> необязательный параметр, (например, «eeserver:/lessons/number1»). Параметр <server> должен быть получен от администратора сервиса, запускающего реализацию CDMI;
- «exportdomain». Доменное имя сервера имен для обслуживаемого клиента. Обычно это домен LDAP организации, например, «iti.edu». Значение «.» следует интерпретировать как имя DNS домена, занятого сервером CDMI;
- «mode». Может принимать одно из значений «ro», «rw», «root» или «rpc_gsssec». Указанный режим становится режимом экспорта по умолчанию. Узлы, требующие различного доступа, должны быть указаны в опциональных элементах структуры «rw_mode», «ro_mode» и «root_mode». Однако, режим «rpc_gsssec» перекрывает все другие режимы, и другие элементы, связанные с режимами доступа, и их содержимое будут в этом случае игнорироваться;
- «control». Управление экспортом контейнера, может принимать значения «immediate», «off», «on», или <n> (число). Серверы могут устанавливать значение в «on», а клиенты нет. Числовое значение (<n>) указывает, что экспорт должен быть отключен через <n> секунд, возможно, после отсылки сообщения клиентам, подключенным для экспорта. Если клиент указывает значение <n>, но сервер не поддерживает отложенное отключение экспортов, данный параметр должен обрабатываться как равный «off».

Опциональные параметры для экспорта по протоколу NFS включают:

– «domain_servers». Список имен серверов или IP адресов, которые функционируют как сервера имен для домена, указанного в «domain». Если приведено, это значение перекрывает имена, полученные от сервера CDMI другими программными средствами;

– «mount_name». Имя точки подключения экспорта. Этот параметр замещает последний сегмент имени в строке пути (например, при подключении «eeserver:/lessons/number1» с mount_name равно «1» в папку /somepath/lessons/num1 должно дать папку /somepath/lessons/1 клиента);

– «hosts». Список узлов, которые могут получить доступ к контейнеру в режиме, указанном в «mode». По умолчанию равно «*»; другие значения ограничивают возможности;

– «root_hosts». Список узлов, которые могут получить доступ к контейнеру в режиме суперпользователя. По умолчанию список пуст;

– «rw_hosts». Список узлов, которые могут получить доступ к контейнеру в режиме r/w. По умолчанию список пуст;

– «ro_hosts». Список узлов, которые могут получить доступ к контейнеру в режиме лишь r/o. По умолчанию список пуст;

– «mount_type». Одна из двух строк «hard» или «soft». Если сервер перестает отвечать, клиент с подключением "hard" зависает. Клиент с подключением "soft" при в этой систуации генерирует сообщения об ошибке. Значение по умолчанию зависит от реализации,

– «recurse». Может быть «true» или «false», по умолчанию «true». Если «true», параметр указывает, клиент может переходить по подключениям (mount) в пределах структуры директорий CDMI (предположительно, созданным другими NFS операциями), и подключенная директория должна быть показана, как если бы она была частью экспортируемого CDMI контейнера. Параметр эквивалентен параметру «crossmnt» системы Linux.

Другие параметры NFS экспорта не описаны в протоколе CDMI, но могут включаться в структуру экспорта. Например, такими параметрами могут быть аналоги параметров из системы Linux, такие как «sync», «no_wdelay», «insecure_locks» и «no_acl», а также любые другие параметры, использованные данной серверной операционной системой. В таких случаях, параметры должны быть указаны в нотации JSON, в которой «true» и «false» используются как бинарные флаги, а для параметров другого типа используются строки или списки.

Пример –

```
{ «exports»
        { «nfs»
                {...
                        {«no_wdelay», «true» },
                        {«refer», «otherserver://path/leaf»},
                ...
                }
        }
}
```

Управление экспортом

Управление экспортом осуществляется с участием единственного элемента с именем «control»:

– значение «immediate» указывает серверу на то, что экспорт необходимо осуществить немедленно, перед завершением операции PUT. Серверы должны заменять значение в «on» и помещать это в ответ;

– значение «off» указывает серверу, что не следует начинать новый экспорт, а существующий экспорт должен быть отменен, с разрывом всех клиентских соединений;

– числовое значение <n> указывает, что сервер должен выждать <n> секунд перед принудительным сбросом экспорта и разрывом клиентских соединений. Рекомендуется, чтобы сервер посылал предупредительное сообщение, позволяя клиентам выйти из соединения обычным путем, но зависит от реализации. Если экспорт был отменен, сервер должен также изменить значение параметра «control» на «off» в структуре экспорта.

Серверы должны поддерживать символы подстановки «*» и «?» в списках узлов (это стандартное поведение), так что **.cs.uscs.edu» соответствует всем серверам факультета cs.ucsc.edu.

Серверы могут поддерживать сетевые групповые имена в различных списках узлов. Эти списки, если поддерживаются, должны разрешаться в обычные списки имен узлов посредством запросов к серверу имен домена.

Серверы также могут поддерживать диапазоны IP адресов в различных списках узлов. Это должны быть IP адреса, построенные с использованием обычных подстановочных символов (например, «192.168.1.*» экспортирует на все машины домашней сети NetGear по умолчанию). Разработчики клиентской части должны иметь в виду, что «экспортирование» означает единственно то, что контейнер может быть экспортирован. Клиент должен также подключить экспортированный контейнер перед установкой соединения с сервером.

Пользователи, желающие использовать опциональные и зависящие от реализации настройки сами определяют от поставщика CDMI продукта, какие настройки допустимы и в каком формате. Серверы могут возвращать код 400 (Bad Request) если настройки экспорта не соответствуют допустимым настройкам сервера.

13.5 Экспортный протокол CIFS

Для экспорта контейнера посредством CIFS, необходима информация о том, какая серверная реализация будет выполнять экспорт. Местонахождение этой информации зависит от реализации. Сервер может автоматически добавлять или удалять строки из файла для каждого контейнера CDMI, который экспортируется или деэкспортируется.

Обязательные элементы протокола структуры для экспорта CIFS включают:

- «share_name». Имя, по которому экспортированный элемент обнаруживается в CIFS;
- «exportdomain». Домен сервера имен протокола для обслуживаемого клиента. Обычно это домен LDAP организации, например «iti.edu». Значение «.» следует интерпретировать как DNS имя домена, занятого сервером CDMI;
- «mode». может быть «ro» или «rw»;
- «control». Управление экспортом для контейнера. Может принимать значения «immediate», «off», или <n> (число). Серверы могут устанавливать значение в «on», но клиенты не могут. Семантика и требования к параметру в точности соответствуют таковым для NFS, см. «Управление экспортом», 13.4).

Поле "protocol" не специфицируется; модель CDMI предполагает, что осуществляется обычное согласование версии по протоколу SMB.

Опциональный параметр экспорта -- «comment», который часто используется для задания удобочитаемого имени экспортируемого ресурса для клиента.

Другие параметры CIFS экспорта не описаны в протоколе CDMI, но могут включаться в структуру экспорта. Например, такими параметрами могут быть параметры производителя «forcegroup», «umask», «caching», and «oplocks», а также любые другие параметры, использованные данной серверной операционной системой. В таких случаях, параметры должны быть указаны в нотации JSON, в которой «true» и «false» используются как бинарные флаги, а для параметров другого типа используются строки или списки.

Пример –

```
{ «exports»
        { «cifs»
                {...
                        {«caching», [{ «manual», «document», «program» }],
                        {«oplocks», «true»},
...
}
} 379   }
```

Пользователи, желающие использовать опциональные и зависящие от реализации настройки, сами определяют от поставщика CDMI продукта, какие настройки допустимы и в каком формате. Серверы могут возвращать код 400 (Bad Request) если настройки экспорта не соответствуют допустимым настройкам сервера.

13.6 Экспортный протокол OCCI

CDMI определяет структуру протокола экспорта для стандарта OGF: Open Cloud Computing Interface (OCCI) следующим образом:

- Протокол: «OCCI/<стандарт протокола>» (например, OCCI/NFSv4).
- Идентификатор – ID объекта CDMI.

– Массив JSON из URI к вычислительным ресурсам OCCI должны иметь доступ к экспортированному контейнеру.

Пример – Структура протокола экспорта OCCI в JSON:

```
«OCCI/iSCSI»: {
        «identifier»: «00007E7F00104BE66AB53A9572F9F51E»,
        «permissions»: [
                «http://example.com/compute/0/»,
                «http://example.com/compute/1/»
        ]
}
```

Подробное описание структуры протокола экспорта OCCI приведено в 13.1. Так как в действительности управление сетями и доступом осуществляется невидимой для клиентов общей инфраструктурой, включающей и OCCI, и CDMI, обычная структура доступа не должна предоставляться.

13.7 Модификация экспорта iSCSI

CDMI определяет экспорт контейнера по протоколу iSCSI (см. RFC 3720). Каждый контейнер экспортируется как отдельное логическое устройство (Logical Unit) SCSI с номером логического устройства (Logical Unit Number, LUN). Один или несколько инициаторов iSCSI импортируют LUN из контроллера iSCSI, используя один или несколько сетевых порталов iSCSI (IP адресов).

Такой экспорт описывается наличием в контейнере структуры экспортного поля, которое указывает

– протокол экспорта («Network/iSCSI»);

– информацию о контроллере iSCSI (IP адрес или полное доменное имя, идентификатор контроллера и LUN);

– имя логического устройства;

– iSCSI инициаторы, которым предоставлен доступ.

Идентификатор контроллера может указываться в формате iqn, naa или eui, и должен иметь шестнадцатиричную метку группы портала.

13.7.1 Чтение контейнера

Возврат полной информации о структуре экспорта:

```
«exports» :
{
        «Network/iSCSI»: {
                «portals»: [
                        «192.168.1.101»,
                        «192.168.1.102»
                ],
                «target_identifier». «iqn.2010-01.com.cloudprovider:acmeroot.container1,t,0x0001»,
                «logical_unit_number»: «3»,
                «logical_unit_name»: «0x60012340000000000000000000000001»,
                «permissions»: [
                        «iqn.2010-01.com.acme:host1»,
                        «iqn.2010-01.com.acme:host2»
                ]
        }
}
```

13.7.2 Создание и изменение контейнеров

Следующий код создает контейнер с iSCSI экспортом и изменяет существующий контейнер, добавляя новый iSCSI экспорт. Поддержка любой из этих операций обозначается наличием опции cdmi_export_iscsi в родительском или существующем контейнере.

```
«exports» :
{
```

```
        «Network/iSCSI»: {
                «permissions»: [
                        «iqn.2010-01.com.acme:host1»,
                        «iqn.2010-01.com.acme:host2»
                ]
        }
}
```

Для этих экспортных операций реализация CDMI выбирает IP порталы, iSCSI цель, номер логического устройства и его имя; эти параметры не выдаются. Указывается только список инициаторов, которые должны иметь доступ к экспортируемому контейнеру.

13.7.3 Изменение экспорта

Следующий код изменяет экспорт существующего контейнера. Поддержка этой операции обозначается наличием опции cdmi_export_iscsi в родительском контейнере существующего контейнера. Для данной операции указывается только список инициаторов, которые должны иметь доступ к экспортируемому контейнеру.

```
«exports» :
{
«Network/iSCSI»: {
«permissions»: [
«iqn.2010-01.com.acme:host2»
]
}
}
```

13.8 Экспортный протокол WebDAV

CDMI определяет экспорт контейнера по стандарту WebDAV следующим образом (см. RFC 4918):

- Протокол равен «Network/WebDAV».
- Путь точки подключения WebDAV в том виде, как передается клиентам (включая имя узла сервера).
- Список тех, кто имеет доступ к разделяемому объекту, определяется стандартными ACL CDMI для каждого ресурса, экспортированного через WebDAV.

Пример – Структура протокола экспорта WebDAV в JSON:

```
«Network/WebDAV» :
{
        «identifier»: «/users»,
        «permissions»: «domain»
}
```

В этом примере значение «domain» в поле разрешений указывает, что имена пользователей должны ставиться в соответствие через членство в домене экспортируемого CDMI контейнера.

WebDAV поддерживает блокировку, но поддерживается ли блокировка доступа через CDMI, определяется реализацией, поэтому взаимодействие двух протоколов намеренно не описывается в настоящем стандарте.

14 Снимки состояния

Снимок состояния – моментальная копия контейнера и его содержимого, включая вложенные контейнеры, объекты данных и объекты-очереди. Клиент именует снимок в момент запроса снимка. Операция создания снимка создает новый контейнер, содержащий моментальную копию. Первое создание снимка добавляет вложенный контейнер cdmi_snapshots в исходный контейнер. Создаваемые затем моментальные снимки добавляются как потомки контейнера cdmi_snapshots. Снимок на рисунке 9 не содержит контейнер cdmi_snapshots и его содержимое.

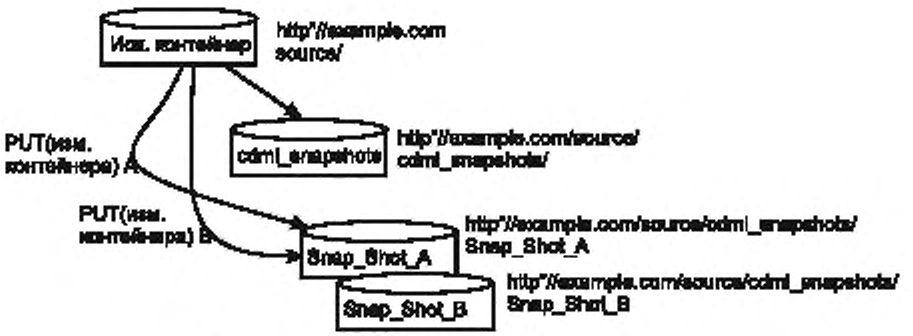

Рисунок 9 – Структура снимка состояния контейнера

Операция создания снимка состояния запрашивается посредством операции обновлении контейнера (см. 9.5), в которой поле snapshot указывает на запрошенное имя снимка.

Доступ к снимку состояния осуществляется аналогично другим объектам CDMI™. Важное применение снимков состояния – откат состояния исходного контейнера к предыдущему состояние с использованием операции копирования CDMI объектов.

15 Сериализация/ десериализация

15.1 Обзор

Зачастую требуется переместить большой объем данных между облаками, в облако или из него. Операции облачной сериализации, предоставляющие механизм нормализации данных в канонический, самоописывающий формат, включают:

- перемещение данных между облаками;
- перемещение данных во время обновления или замены облачных реализаций;
- простое резервное копирование.

Канонический формат сериализованных данных описывает, как данные должны представляться в виде потоке байтов. Если этот поток не изменяется во время передачи, данные могут быть восстановлены в системе назначения.

15.2 Экспорт сериализованных данных

Каноническая форма данных получается при создании нового объекта и указании, что источник данных – сериализация заданного объекта данных, контейнера или очереди CDMI™. Результат успешной сериализации – объект данных, содержащи сериализованные данные в поле value. Если контейнер включает блочный протокол экспорта, сериализованные данные могут включать содержимое контейнера поблочно, а также его метаданные.

Полученный в результате объект данных есть каноническое представление выбранного объекта данных, контейнера и потомков или очереди.

- Если в качестве источника указан объект данных, канонический формат должен включать все поля объекта данных, в том числе value, valuetransferencoding и поля метаданных.
- Если в качестве источника указана очередь, канонический формат должен включать все поля очереди, включая поля value и valuetransferencoding элементов очереди, вместе с метаданными самой очереди.
- Если в качестве источника указан контейнер, канонический формат должен рекурсивно включать все поля контейнера и его потомков. Если пользователь пытается сериализовать контейнер, и к отдельным потомкам данного контейнера у него нет разрешения на доступ, эти элементы не должны включаться в конечный сериализованный объект.

При выполнении сериализации, в результат должны включаться лишь те объекты, для которых у пользователя, инициирующего сериализацию, есть права на чтение.

15.3 Импорт сериализованных данных

Канонические данные могут быть десериализованы обратно в облако, создавая новый объект данных, контейнер или очередь; для этого следует указать, что источником данных для создания является десериализация заданного объекта, либо поместить сериализованные данные в кодировке base 64 в поле deserializevalue.

Объект назначения может как существовать заранее, так и создаваться заново. Если контейнер уже существует, операция обновления с сериализованным потомком должна обновить контейнер и всех его потомков. Если сериализованный контейнер не содержит потомков, только объект-контейнер будет обновлен. Объекты данных воссоздаются в соответствии с каноническим форматом, включая все метаданные и ID объекта.

- Если пользователь, который выполняет десериализацию сериализованных данных, имеет права «cross_domain» и не указал domainURI как часть операции десериализации, будет использовано исходное поле domainURI сериализованного объекта. Если некоторые из указанных domainURI некорректны в контексте системы хранения, в которой выполняется десериализация, вся операция десериализации должна быть отменена.
- Если пользователь, который выполняет десериализацию сериализованного объекта, указывает domainURI как часть операции десериализации, domainURI каждого десериализуемого объекта устанавливается в указанное значение. Чтобы указать domainURI, отличный от domainURI родителя, пользователь должен иметь права cross_domain. Если у пользователя нет таких прав, и указан domainURI, отличный от domainURI родителя, должен быть возвращен ответ 400 Bad Request.
- Если пользователь, который выполняет десериализацию сериализованного объекта, не указывает domainURI как часть операции десериализации и не имеет прав «cross_domain», десериализация должна быть успешной только если у всех объектов тот же domainURI, что у родительского объекта, на котором выполняется десериализация.

Операции десериализации должны восстанавливать все метаданные из указанного источника. Если в исходном сериализованном объекте были использованы расширения производителя через произвольные ключи и значения метаданных, эти произвольные требования должны быть восстановлены при десериализации. Однако произвольные метаданные (ключи и значения) могут быть интерпретированы как пользовательские метаданные (сохраняться, но не обрабатываться).

Такое сохранение позволяет перемещать между облаками данные произвольной конфигурации.

15.3.1 Канонический формат

Канонический формат должен представлять указанные объекты данных и контейнеры как если бы они существовали в системе хранения. Каждый объект должен представляться метаданными объекта, идентификаторами и потоком данных объекта данных. Поскольку метаданные наследуются от контейнеров более высокого уровня, все родительские метаданные должны быть представлены в канонической форме (существенно уплощая иерархию). Для сохранения текущих значений метаданных, относящихся к сериализуемому объекту, не перезаписанные метаданные включаются как из непосредственного родительского контейнера, так и от родительских контейнеров более высокого уровня.

Канонический формат должен обладать следующими свойствами:

- рекурсивное представление JSON для объекта данных, совместимое с остальной частью CDMI;
- метаданные пользователя и системы данных для каждого объекта данных/контейнера;
- содержимое потока данных для каждого объекта данных и очереди;
- двоичные данные представляются в форме строк JSON;
- типы значений данных совместимы с JSON представлениями CDMI.

15.3.2 Пример канонического сериализованного формата JSON

Пример – Для сериализации выбраны объект данных и очередь в контейнере:

```
{
«objectType»: «application/cdmi-container»,
«objectID»: «00007E7F00102E230ED82694DAA975D2»,
«objectName»: «MyContainer/»,
«parentURI»: «/»,
«parentID»: «00007E7F0010128E42D87EE34F5A6560»,
«domainURI»: «/cdmi_domains/MyDomain/»,
«capabilitiesURI»: «/cdmi_capabilities/container/»,
```

«completionStatus»: «Complete»,
«metadata»: {},
«exports» : {
«OCCI/iSCSI»: {
«identifier»: «00007E7F00104BE66AB53A9572F9F51E»,
«permissions»: [
«http://example.com/compute/0/»,
«http://example.com/compute/1/»
]
}, «Network/NFSv4» : {
«identifier» : «/users»,
«permissions» : «domain»
}
},
«childrenrange» : «0-1»,
«children» : [
{
«objectType» : «application/cdmi-object»,
«objectID» : «0000706D0010B84FAD185C425D8B537E»,
«objectName» : «MyDataObject.txt»,
«parentURI» : «/MyContainer/»,
«parentID» : «00007E7F00102E230ED82694DAA975D2»,
«domainURI» : «/cdmi_domains/MyDomain/»,
«capabilitiesURI» : «/cdmi_capabilities/dataobject/»,
«completionStatus» : «Complete»,
«mimetype» : «text/plain»,
«metadata» : {
},
«valuerange» : «0-36»,
«valuetransferencoding»: «utf-8»,
«value» : «This is the Value of this Data Object»
},
{
«objectType» : «application/cdmi-queue»,
«objectID» : «00007E7F00104BE66AB53A9572F9F51E»,
«objectName» : «MyQueue»,
«parentURI» : «/MyContainer/»,
«parentID» : «00007E7F00102E230ED82694DAA975D2»,
«domainURI» : «/cdmi_domains/MyDomain/»,
«capabilitiesURI» : «/cdmi_capabilities/queue/»,
«completionStatus» : «Complete»,
«metadata» : {

},
«queueValues» : «0-1»,
«mimetype»: [
«text/plain»,
«text/plain»
],
«valuetransferencoding»: [
«utf-8»,
«utf-8»
],»valuerange» : [
«0-2»,
«0-3»
],
«value» : [
«red»,
«blue»
]
}
] 145 }

При сериализации контейнеров в JSON массив children должен быть последним элементом канонического формата сериализации JSON для эффективной десериализации в потоковом режиме.

16 Метаданные

16.1 Управление доступом

Управление доступом предоставляет механизмы, позволяющие авторизовывать и либо разрешать, либо запрещать различные способы доступа к объектам и контейнерам. CDMI™ использует известный механизм списков контроля доступа (Access Control List, ACL), определенный в стандарте NFSv4 (см. RFC 3530). ACL представляют собой списки записей, предоставляющих или запрещающих доступ, которые называются записи управления доступом (access control entry, ACE).

16.1.1 Структуры ACL и ACE

ACL представляет собой упорядоченный список ACE. Существуют два типа ACE в модели CDMI: ALLOW и DENY. Запись ALLOW разрешает некоторый тип доступа соответствующему принципалу (пользователю или группе пользователей, представленными идентификаторами); наоборот, запись DENY запрещает принципалу какой-то тип доступа. Например, DENY может запрещать запись метаданных или ACL в объект, но не запрещать любые другие формы доступа. Однако, если другая запись ACE ALLOW разрешает запись в объект, принципалу разрешено записывать данные объекта, но ничего больше.

ACE включают четыре поля: type, who, flags и access_mask, см. RFC 3530. Поля type, flags и access_mask должны быть либо строковым представлением целого беззнакового числа в шестнадцатеричной записи, либо списком строковых битовых масок из таблиц 112, 114 115, разделенных запятыми.

16.1.2 Типы ACE

В таблице 112 приведены типы ACE, согласно NFSv4.

Таблица 112 – Типы ACE

Строковая форма	Описание	Константа	Битовая маска
"ALLOW"	Предоставляет принципалу права доступа	CDMI_ACE_ACCESS_ALLOW	0x00000000
"DENY"	Запрещает принципалу доступ	CDMI_ACE_ACCESS_DENY	0x00000001
"AUDIT"	Создает запись-отчет, когда принципал пытается осуществить определенный доступ	CDMI_ACE_SYSTEM_AUDIT	0x00000002

Примечание – Строковые формы могут быть безопасно сокращены, так как они локальны для структуры ACE, в отличие от относительно глобальных констант.

Порядок ACE в ACL задается клиентом. Сервер не должен задавать какой-либо порядок, и должен хранить и анализировать записи ACE в порядке, заданном клиентом.

16.1.3 Поле Who

Особые идентификаторы «who» должны интерпретировать универсально, а не в контексте данного внешнего домена безопасности (см. таблицу 113). Некоторые из этих идентификаторов могут недоступны для клиентов, когда клиент CDMI получает доступ к серверу, но они могут быть значимыми, когда локальный процесс получает доступ к файлу. CDMI должен предоставить возможность показывать и изменять эти разрешения, даже если ни один из методов доступа на сервере не использует эти идентификаторы.

Таблица 113 – Идентификаторы Who

Идентификаторы	Описание
"OWNER@"	Владелец файла
"GROUP@"	Группа, ассоциированная с файлом
"EVERYONE@"	Любой принципал
"ANONYMOUS@"	Доступ без аутентификации
"AUTHENTICATED@"	Любой аутентифицированный пользователь (противоположность ANONYMOUS)
"ADMINISTRATOR@"	Пользователь со статусом администратора, например, root
"ADMINUSERS@"	Группа, члены которой имеют права администратора

Чтобы избежать конфликта имен, специальные идентификаторы заканчиваются символом «@» без указания доменного имени.

16.1.4 Поле flags

CDMI позволяет организовывать вложенные контейнеры и передачу полномочий, когда объекты и вложенные контейнеры могут наследовать разрешения на доступ от родительских контейнеров. Однако недостаточно просто наследовать все разрешения от родителя, может быть необходимо, например иметь различные разрешения по умолчанию для потомков и вложенных контейнеров данного контейнера. Флаги, описанные в таблице 114, управляют этим поведением.

Таблица 114 – Флаги ACE

Строковая форма	Описание	Константа	Битовая маска
"NO_FLAGS"	Флаги не установлены	CDMI_ACE_FLAGS_NONE	0x00000000
"OBJECT_INHERIT"	Запись ACE, на которую установлен OBJECT_INHERIT, наследуется объектами как фактическая ACE: флаг OBJECT_INHERIT снимается для объекта-потомка. Если ACE наследуется контейнером, OBJECT_INHERIT сохраняется для наследования, кроме того, устанавливается, INHERIT_ONLY.	CDMI_ACE_FLAGS_OBJECT_INHERIT_ACE	0x00000001
"CONTAINER_INHERIT"	Запись ACE, на которую установлен CONTAINER_INHERIT, наследуется вложенным контейнером как фактическая ACE. Флаги INHERIT_ONLY, и CONTAINER_INHERIT снимаются у контейнера-потомка.	CDMI_ACE_FLAGS_CONTAINER_INHERIT_ACE	0x00000002
"NO_PROPAGATE"	Запись ACE, на которую установлен NO_PROPAGATE, не наследуется никакими объектами или вложенными контейнерами. Она применяется лишь к контейнеру, на который была установлена.	CDMI_ACE_FLAGS_NO_PROPAGATE_ACE	0x00000004
"INHERIT_ONLY"	Запись ACE, на которую установлен INHERIT_ONLY, наследуется потомками через наследование ACL в соответствии с OBJECT_INHERIT и CONTAINER_INHERIT. Запись ACE игнорируется при оценке доступа к контейнеру, для которого она была установлена и всегда игнорируется, если установлена для объекта.	CDMI_ACE_FLAGS_INHERIT_ONLY_ACE	0x00000008
"IDENTIFIER_GROUP"	Запись ACE, на которую установлен IDENTIFIER_GROUP, показывает, что "who" ссылается на идентификатор группы.	CDMI_ACE_FLAGS_IDENTIFIER_GROUP	0x00000040
"INHERITED"	Запись ACE, на которую установлен INHERITED, показывает, что эта ACE наследована от родительской директории. Сервер, который поддерживает автоматическое наследование, помещает этот флаг на любую ACE, наследованную от родительской директории при создании нового объекта.	CDMI_ACE_FLAGS_INHERITED_ACE	0x00000080

16.1.5 Битовые маски ACE

Поле mask в ACE содержит 32 бита. В таблице 115 приведены битовые маски ACE в CDMI; их значения взяты из IETF NFSv4 RFC 3530.

Таблица 115 – Битовые маски ACE

Строковая форма	Описание	Константа	Битовая маска
"READ_OBJECT"	Разрешение на чтение значения из объекта данных	CDMI_ACE_READ_OBJECT	0x00000001
"LIST_CONTAINER"	Разрешение на перечисление потомков объекта-контейнера	CDMI_ACE_LIST_CONTAINER	0x00000001
"WRITE_OBJECT"	Разрешение на изменение значения объекта данных	CDMI_ACE_WRITE_OBJECT	0x00000002
"ADD_OBJECT"	Разрешение на добавление нового объекта-потомка (объекта данных или очереди) к контейнеру	CDMI_ACE_ADD_OBJECT	0x00000002

Окончание таблицы 115

Строковая форма	Описание	Константа	Битовая маска
"APPEND_DATA"	Разрешение на добавление данных к значению объекта данных	CDMI_ACE_APPEND_DATA	0x00000004
"ADD_SUBCONTAINER"	Разрешение создавать вложенный контейнер в контейнере	CDMI_ACE_ADD_SUBCONTAINER	0x00000004
"READ_METADATA"	Разрешение на чтение не-ACL метаданных объекта	CDMI_ACE_READ_METADATA	0x00000008
"WRITE_METADATA"	Разрешение на запись не-ACL метаданных объекта	CDMI_ACE_WRITE_METADATA	0x00000010
"EXECUTE"	Разрешение на исполнение объекта	CDMI_ACE_EXECUTE	0x00000020
"DELETE_OBJECT"	Разрешение на удаление объекта-потомка (объект данных или очередь) из контейнера	CDMI_ACE_DELETE_OBJECT	0x00000040
"DELETE_SUBCONTAINER"	Разрешение на удаление вложенного контейнера из контейнера	CDMI_ACE_DELETE_SUBCONTAINER	0x00000040
"READ_ATTRIBUTES"	Разрешение на чтение полей объекта, не относящихся к метаданным или значению/потомкам	CDMI_ACE_READ_ATTRIBUTES	0x00000080
"WRITE_ATTRIBUTES"	Разрешение на изменение полей объекта, не относящихся к метаданным или значению/потомкам	CDMI_ACE_WRITE_ATTRIBUTES	0x00000100
"WRITE_RETENTION"	Разрешение на изменение атрибутов отложенного удаления объекта	CDMI_ACE_WRITE_RETENTION	0x00000200
"WRITE_RETENTION_HOLD"	Разрешение на изменение атрибутов удержания объекта	CDMI_ACE_WRITE_RETENTION_HOLD	0x00000400
"DELETE"	Разрешение на удаление объекта	CDMI_ACE_DELETE	0x00010000
"READ_ACL"	Разрешение на чтение ACL объекта	CDMI_ACE_READ_ACL	0x00020000
"WRITE_ACL"	Разрешение на запись ACL объекта	CDMI_ACE_WRITE_ACL	0x00040000
"WRITE_OWNER"	Разрешение на изменение владельца объекта	CDMI_ACE_WRITE_OWNER	0x00080000
"SYNCHRONIZE"	Разрешение на локальный доступ к объекту на сервере с синхронными чтением и записью	CDMI_ACE_SYNCHRONIZE	0x00100000

Реализации должны использовать корректную строковую форму для отображения разрешений, если тип объекта известен. Если тип объекта неизвестен, должна использоваться форма строки «object».

16.1.6 Обработка ACL

При оценке, предоставлены ли права доступа к объекту O для принципала P, сервер должен проследить логический ACL объекта (его ACL после обработки наследования от родительского контейнера) в порядке, указанном в списке, используя временную битовую маску разрешения m, изначально пустую (заполненную нулями).

- Если объект не содержит ACL, алгоритм завершается, и доступ запрещен всем пользователям и группам. Такое состояние не ожидается, так как реализации CDMI требуют наследования ACL по умолчанию от всех корневых контейнеров.
- ACE, не относящиеся к принципалу P, игнорируются.
- Если встреченная ACE запрещает доступ P для некоторых из битов маски, доступ запрещается и алгоритм завершается.
- Если встреченная ACE разрешает принципалу P доступ, маска разрешений m складывается по правилу XOR с маской разрешений из ACE. Если m достаточна для запрашиваемой операции, доступ разрешается, и алгоритм завершается.
- При достижении конца ACL, если доступ не был разрешен и не был прямо запрещен, доступ запрещается, и алгоритм завершается, если только объект O не является корневым контейнером. В этом случае сервер должен:

– разрешить доступ владельцу контейнера, ADMINISTRATOR@, либо любому члену группы ADMINUSERS@; и

– записать в лог событий произошедшее.

Если разрешение на операцию не было явно дано, даже ADMINISTRATOR@ и эквивалентные пользователи не могут получить доступ к объектам, отличным от корневого контейнера. Если администратору требуется доступ к объекту, следует обратиться к корневому контейнеру и изменить его наследуемые ACE, чтобы разрешить доступ к необходимому объекту. В результате создается запись в логе для отслеживания доступа.

Если создается корневой контейнер без указания ACL, сервер должен разместить для контейнера ACL, содержащий следующие ACEs:

```
«cdmi_acl»:
[
        {
                «acetype»: «ALLOW»,
                «identifier»: «OWNER@»,
                «aceflags»: «OBJECT_INHERIT, CONTAINER_INHERIT»,
                «acemask»: «ALL_PERMS»
        },
        {
                «acetype»: «ALLOW»,
                «identifier»: «AUTHENTICATED@»,
                «aceflags»: «OBJECT_INHERIT, CONTAINER_INHERIT»,
                «acemask»: «READ»
        }
]
```

Так как ACL является метаданными системы хранения, они сохраняются и предоставляются через поле metadata в запросе PUT или GET. Синтаксис ACL представлен ниже. Используются строковые константы из таблиц 112, 114, 115.

```
ACL = { ACE [, ACE ...] }
ACE = { acetype , identifier , aceflags , acemask }
acetype = uint_t | acetypeitem
identifier  = utf8string_t
aceflags    = uint_t | aceflagsstring
acemask     = uint_t | acemaskstring
acetypeitem = aceallowedtype |
        acedeniedtype |
        aceaudittype
aceallowedtype = «CDMI_ACE_ACCESS_ALLOWED_TYPE» | 0x0
acedeniedtype  = «CDMI_ACE_ACCESS_DENIED_TYPE» | 0x01
aceaudittype   = «CDMI_ACE_SYSTEM_AUDIT_TYPE» | 0x02
aceflagsstring = aceflagsitem [| aceflagsitem ...]
aceflagsitem   = aceobinheritem |
        acecontinheritem |
        acenopropagateitem |
        aceinheritonlyitem
aceobinheritem   = «CDMI_ACE_OBJECT_INHERIT_ACE» | 0x01
acecontinheritem = «CDMI_ACE_CONTAINER_INHERIT_ACE» | 0x02
acenopropagateitem = «CDMI_ACE_NO_PROPAGATE_INHERIT_ACE» | 0x04
aceinheritonlyitem = «CDMI_ACE_INHERIT_ONLY_ACE» | 0x08
acemaskstring  =  acemaskitem [| acemaskitem ...]
acemaskitem    =  acereaditem | acewriteitem |
        aceappenditem | acereadmetaitem |
        acewritemetaitem | acedeleteitem |
        acedelselfitem | acereadaclitem |
        acewriteaclitem | aceexecuteitem |
        acereadattritem | acewriteattritem |
        aceretentionitem
acereaditem      = «CDMI_ACE_READ_OBJECT» |
```

«CDMI_ACE_LIST_CONTAINER» | 0x01
acewriteitem = «CDMI_ACE_WRITE_OBJECT» |
«CDMI_ACE_ADD_OBJECT» | 0x02
aceappenditem = «CDMI_ACE_APPEND_DATA» |
«CDMI_ACE_ADD_SUBCONTAINER» | 0x04
acereadmetaitem = «CDMI_ACE_READ_METADATA» | 0x08
acewritemetaitem = «CDMI_ACE_WRITE_METADATA» | 0x10
acedeleteitem = «CDMI_ACE_DELETE_OBJECT» |
«CDMI_ACE_DELETE_SUBCONTAINER» | 0x40
acedelselfitem = «CDMI_ACE_DELETE» | 0x10000
acereadaclitem = «CDMI_ACE_READ_ACL» | 0x20000
acewriteaclitem = «CDMI_ACE_WRITE_ACL» | 0x40000
aceexecuteitem = «CDMI_ACE_EXECUTE» | 0x80000
acereadattritem = «CDMI_ACE_READ_ATTRIBUTES» | 0x00080
acewriteattritem = «CDMI_ACE_WRITE_ATTRIBUTES» | 0x00100
aceretentionitem = «CDMI_ACE_SET_RETENTION» | 0x10000000

Если маски ACE показываются в числовом формате, они должны всегда быть шестнадцатеричными и начинаться с «0x». Этот формат позволяет как серверам, так и клиентам быстро определять, какая из двух форм использована для данной константы. Если маски представлены в строковой форме, они должны быть преобразованы в числовой формат и затем анализироваться с использованием стандартных побитовых операций.

Если при создании объекта ACL не указан и не наследуется от родительского контейнера (или родительский контейнер отсутствует), сервер должен разместить на объекте следующий ACL:

```
«cdmi_acl»:
[
        {
                «acetype»: «ALLOW»,
«               identifier»: «OWNER@»,
                «aceflags»: «OBJECT_INHERIT, CONTAINER_INHERIT»,
                «acemask»: «ALL_PERMS»
        }
]
```

16.1.7 Пример битовых выражений ACE

Примеры
1 «READ_ALL» | 0x09--0x02
вычисляется как 0x09 | 0x02 == 0x0
2 0x001F07FF
вычисляется как 0x001F07FF == «ALL_PERMS»
3 «RW_ALL» | DELETE
вычисляется как 0x000601DF | 0x00100000 == 0x000701DF

16.1.8 Канонический формат шестнадцатеричных параметров ACE

Битовые выражения ACE должны всегда вычисляться и приводиться к единственному шестнадцатеричному значению перед передачей в датаграмму протокола HTTP. Приложения или утилиты, отображающие их для пользователей, должны преобразовывать их в текстовое представление и принимать текстовый ввод от пользователя. Для этого следует использовать побитовые операторы «|» и «&».

Для разложения маски в строки следует использовать следующий алгоритм. Таблица масок и строковых эквивалентов должна поддерживаться в порядке от большего к меньшему:

```
0x001F07FF  «ALL_PERMS»      «ALL_PERMS»
0x0006006F  «RW_ALL»         «RW_ALL»
0x0000001F  «RW»             «RW»              ...
0x00000002  «WRITE_OBJECT»   «ADD_OBJECT»
0x00000001  «READ_OBJECT»    «LIST_CONTAINER»
```

Например, для маски M следует повторять следующие шаги, пока не установится M == 0:

1 Выбрать наибольшую маску m из таблицы, такую что M & m == m.

2 Если объект является контейнером, выбрать строку из 3-й колонки, иначе выбрать строку из 2-й колонки.

3 Побитово вычесть m из M, т.е., M = M xor m.

В полном текстовом представлении выбранные строки соединяются символом «, «, например, «ALL_PERMS, WRITE_OWNER». Строки должны появляться в порядке от большего значения к меньшему.

Такой же способ должен применяться для других наборов соответствий между шестнадцатиричными значениями и строками.

При правильном программировании этот алгоритм требует лишь одного (зачастую частичного) прохода через таблицу соответствий строк и чисел.

16.1.9 Представление ACL в нотации JSON

Флаги и маски ACE входят в 32-битный объект, который часто обрабатывается в виде шестнадцатеричного представления. Формат данных JSON не поддерживает шестнадцатеричные целые, поэтому все шестнадцатеричные целые должны в ACL модели CDMI быть представлены как закавыченные строки, начинающиеся с «0x».

ACL, содержащий одну или несколько ACE, в формате JSON представляется как:

```
{
«cdmi_acl» : [
{
«acetype» : «0xnn»,
«identifier» : «<user-or-group-name>»,
«aceflags» : «0xnn»,
«acemask» : «0xnn»
},
{
«acetype» : «0xnn»,
«identifier» : «<user-or-group-name>»,
«aceflags» : «0xnn»,
«acemask» : «0xnn»
}
]195   }
```

ACE в таком ACL должны сравниваться в порядке появления.

Пример – ACL, внедренный в ответ на запрос GET:

```
HTTP/1.1 200 OK
Content-Type: application/cdmi-object
X-CDMI-Specification-Version: 1.0.2
{
«objectType» : «/application/cdmi-object»,
«objectID» : «0000706D0010734CE0BAEB29DD542B51»,
«objectName» : «MyDataItem.txt»,
«parentURI» : «/MyContainer/»,
«domainURI» : «/cdmi_domains/MyDomain/»,
«capabilitiesURI» : «/cdmi_capabilities/dataobject/»,
«completionStatus» : «Complete»,
«mimetype» : «text/plain»,
«metadata» : {
«cdmi_size» : «17»,
«cdmi_acl» : [
{
«acetype» : «0x00»,
«identifier» : «EVERYONE@»,
«aceflags» : «0x00»,
«acemask» : «0x00020089»
}
]
},
```

«valuerange» : «0-16»,
«value» : «Hello CDMI World!»
}

16.2 Поддержка пользовательских метаданных

Все объекты CDMI должны поддерживать метаданные, позволяющие включение произвольных определенных пользователем элементов, но имя таких пользовательских метаданных не должно начинаться с префикса «cdmi_».

- Максимальное количество элементов пользовательских метаданных указано в опции «cdmi_ metadata_maxitems».
- Максимальный размер каждого элемента пользовательских метаданных укаан в опции «cdmi_ metadata_maxsize».

16.3 Поддержка метаданных системы хранения

После создания объекта метаданные системы хранения, указанные в таблице 116, должны быть сгенерированы облачной системой хранения и немедленно сделаны доступными клиенту CDMI в виде метаданных, возвращаемых в ответ на операции создания или последующее использование.

Таблица 116 – Метаданные системы хранения

Имя элемента метаданных	Тип	Описание	Требование
cdmi_size	Строка JSON	Количество байтов, которые занимает объект. Этот элемент метаданных рассчитывается системой хранения, и все попытки изменить или установить его должны игнорироваться.	Опционально
cdmi_ctime	Строка JSON	Время создания объекта, в формате ISO-8601, как описано в 5.14.	Опционально
cdmi_atime	Строка JSON	Время доступа к объекту, в формате ISO-8601, как описано в 5.14. Доступ или изменение объекта-потомка не считается доступом к родительскому контейнеру (событие доступа/изменения не распространяется по дереву).	Опционально
cdmi_mtime	Строка JSON	Время последнего изменения объекта, в формате ISO-8601, как описано в 5.14. Изменение объекта-потомка не считается изменением родительского контейнера (событие изменения не распространяется по дереву).	Опционально
cdmi_acount	Строка JSON	Количество доступов к объекту со времени его создания. Доступом считаются все операции чтения, записи и перечисления.	Опционально
cdmi_mcount	Строка JSON	Количество изменений объекта со времени его создания. Изменением считаются все операции изменения значения или метаданных. Модификации метаданных в результате чтения (например, изменение *atime*) не считаются изменением объекта.	Опционально
cdmi_hash	Строка JSON	Хеш значения объекта, кодируемого по правилам base 16, описанным в RFC 4648. Этот элемент метаданных должен присутствовать, когда элемент cdmi_value_hash метаданных системы данных данного или родительского объекта указывает, что значение объекта должно хешироваться.	Опционально
cdmi_owner	Строка JSON	Имя принципала – владельца объекта.	Обязательно
cdmi_acl	Массив JSON объектов JSON	Стандартные метаданные ACL. Если не указаны при создании объекта, этот элемент заполняется системой.	Опционально

16.4 Поддержка метаданных системы данных

Если указаны, метаданные системы данных указывают системе хранения, как осуществлять службы управления данными через интерфейс CDMI.

Метаданные системы данных, приведенные в таблице 117, наследуются от родительского объекта каждым потомком. Если наследник содержит метаданные системы данных, метаданные потомка должны перекрывать соответствующие элементы родительского объекта.

Таблица 117 – Метаданные системы данных

Имя элемента метаданных	Тип	Описание	Требование
cdmi_data_ redundancy	Строка JSON	Если данный элемент метаданных присутствует и установлен равным строковому представлению положительного числа, он показывает сколько полных копий запросил клиент. Дополнительные копии создаются в соответствии с установленным значением. Если данный элемент отсутствует или не установлен равным положительному числу, он не должен использоваться.	Опционально
cdmi_ immediate_ redundancy	Строка JSON	Если данный элемент метаданных присутствует и установлен равным "true", это означает, что клиент запрашивает, чтобы как минимум количество копий, установленных в cdmi_data_redundancy, содержали вновь записанное значение до завершения операции. Этот элемент обеспечивает запись на постоянный носитель нескольких копий данных для предотвращения их потери. Если данный элемент отсутствует или не равен "true", он не должен использоваться. Если заданное количество копий не может быть создано за время ожидания операции HTTP, транзакция должна завершиться, но элемент метаданных cdmi_immediate_redundancy_provided должен быть установлен в "false".	Опционально
cdmi_ assignedsize	Строка JSON	Если данный элемент метаданных присутствует и установлен равным строковому представлению положительного числа, он показывает, что клиент передает размер в байтах, необходимый для экспорта контейнера через другие протоколы (см. 9.1.1). Система не обязана резервировать этот объем, и он может быть выделен по механизму тонкого резервирования. Таким образом, запрошенное значение может быть больше, чем реальный объем использоеанного пространства. Если этот элемент отсутствует или не содержит положительное число, он не должен использоваться. Данный элемент применим только к контейнерам и не наследуется их дочерними объектами.	Опционально
cdmi_ infrastructure _redundancy	Строка JSON	Если данный элемент метаданных присутствует и установлен равным строковому представлению положительного числа, он указывает на то, что клиент запрашивает соответствующее число независимых инфраструктур хранения для поддержки нескольких копий данных. Данный элемент используется для того, чтобы копии, запрошенные в cdmi_data_redundancy, хранились на нескольких независимых устройствах. Если этот элемент отсутствует или не равен положительному числу, он не должен использоваться.	Опционально
cdmi_data_ dispersion	Строка JSON	Если данный элемент метаданных присутствует и установлен равным строковому представлению положительного числа, он указывает на то, что клиент запрашивает минимальное расстояние (в км) между инфраструктурами, поддерживающими хранение копий (число копий равно cdmi_infrastructure _redundancy) данных. Этот элемент используется для повышения надежности хранения копий данных, предотвращая их потерю вследствие местных аварий. Если этот элемент отсутствует или не равен положительному числу, он не должен использоваться.	Опционально
cdmi_ geographic_ placement	Массив JSON строк JSON	Если данный элемент метаданных присутствует и соответствует некоторому количеству геополитических идентификаторов, это означает, что клиент накладывает ограничение на географические регионы, где может храниться объект. Каждый геополитический идентификатор должен быть либо строкой, содержащей корректный код страны/административного образования в соответствии со стандартом ISO 3166, показывающей, где разрешено хранение объекта, либо строкой, начинающейся с символа "!" перед кодом по стандарту ISO 3166 страны/административного образования, удаляющей страну/административное образование из предыдущего списка геополитических регионов. Данный список обрабатывается слева направо, обработка заканчивается, если кандидат-хранилище находится в списке как разрешенный или запрещенный регион, или администратиное образование в составе разрешенного или запрещенного региона. В дополнение к кодам ISO 3166, "*" указывает на все регионы.	Опционально

Продолжение таблицы 117

Имя элемента метаданных	Тип	Описание	Требование
		Если место-кандидат не соответствует ни одной из записей списка, оно рассматривается как запрещенное. Если данный элемент метаданных отсутствует, он не должен быть использован. Если данный элемент присутствует, но не содержит корректный идентификатор геополитического региона, операции создания, изменения или десериализации должны возвращать отказ с кодом ошибки HTTP 400 Bad Request. Если данный элемент присутствует и корректен, но нет доступных допустимых метоположений хранилищ, операции создания, изменения или десериализации должны возвращать отказ с кодом ошибки HTTP 403 Forbidden.	
cdmi_ retention_id	Строка JSON	Если данный элемент метаданных присутствует и установлен равным непустой строке, он показывает, что клиент запрашивает добавление данной строки к объекту в качестве метки, обозначающей, что объект подпадает под определенную политику отложенного удаления. Данный элемент метаданных не является обязательным для задания отложенного удаления объекта, но полезен, если требуется найти все объекты, подпадающие под определенную политику отложенного удаления. Если этот элемент отсутствует или является пустой строкой, он не должен использоваться.	Опционально
cdmi_ retention_ period	Строка JSON	Если данный элемент метаданных присутствует и содержит корректный временной интервал в соответствии с ISO 8601:2004 (см. 5.14), он показывает, что клиент запрашивает отложенное удаление данного объекта (см. 17.3). Если этот элемент метаданных отсутствует, он не должен использоваться. Если значение этого элемента не является корректным временным интервалом по ISO 8601:2004, операции создания, изменения или десериализации должны возвращать отказ с кодом ошибки 400 Bad Request. Если данный элемент метаданных обновляется, и новое окончание интервала раньше текущего окончания интервала, операция изменения должна возвращать отказ с кодом ошибки HTTP 403 Forbidden.	Опционально
cdmi_ retention_aut odelete	Строка JSON	Если данный элемент метаданных присутствует и установлен равным "true", он показывает, что клиент запрашивает, чтобы объект, для которого установлено отложенное удаление, был удален после завершения срока хранения. Если этот элемент отсутствует или не равен "true", он не должен использоваться.	Опционально
cdmi_hold_id	Массив JSON строк JSON	Если данный элемент метаданных присутствует и не является пустым массивом, он показывает, что клиент запрашивает удержание объекта (см. 17.4). Каждая строка массива должна содержать уникальный идентификатор удержания, установленный пользователем. Если этот элемент отсутствует или является пустым JSON массивом, он не должен использоваться. Если данный элемент изменяется, и ранее присуисивующая строка удержания удаляется или изменяется в результате, операция создания изменения должна возвращать отказ с кодом ошибки 403 Forbidden. (см. 17.4 касаемо снятия удержания)	Опционально
cdmi_encryp- tion	Строка JSON	Если данный элемент метаданных присутствует и не является пустой строкой, он показывает, что клиент запрашивает фоновое шифрование объекта. Шифрование подразумевает шифрование всех данных и метаданных объекта. Соответствующие значения алгоритма/режима/длины ключа указываются в опции cdmi_encryption. Если этот элемент отсутствует, он не должен использоваться. Если данный элемент присутствует, но не содержит корректной спецификации шифрования, система может проигнорировать данный элемент метаданных, вернув код ошибки HTTP 400 Bad Request, либо самостоятельно выбрать алгоритм/режим/длина ключа.	Опционально

Продолжение таблицы 117

Имя элемента метаданных	Тип	Описание	Требование
		Поддерживаемые алгоритмы шифрования выражаются строкой в форме ALGORITHM_MODE_KEYLENGTH, где: "ALGORITHM" алгоритм шифрования (например, "AES" или "3DES"). "MODE" режим шифрования (например, "XTS", "CBC" или "CTR"). "KEYLENGTH" длина ключа в битах (например, "128", "192", "256"). Для улучшения совместимости различных реализаций CDMI следующие обозначения должны быть использованы для наиболее распространенных параметров шифрования: "3DES_ECB_168" алгоритм TripleDES с тремя ключами, режим электронной кодовой книги (Electronic Code Book, ECB), ключ 168 бит; "3DES_CBC_168" алгоритм TripleDES с тремя ключами, режим сцепления блоков шифртекста (Cipher Block Chaining, CBC), ключ 168 бит; "AES_CBC_128" алгоритм AES, режим CBC, ключ 128 бит; "AES_CBC_256" алгоритм AES, режим CBC, ключ 256 бит; "AES_XTS_128" алгоритм AES, режим XTS, ключ 128 бит; "AES_XTS_256" алгоритм AES, режим XTS, ключ 256 бит.	
cdmi_value_hash	Строка JSON	Если данный элемент метаданных присутствует и не является пустой строкой, он показывает, что клиент запрашивает вычисление системой хэш-кода объекта с использованием указанного алгоритма и длиной хэш-кода. Результат должен быть помещен в элемент cdmi_hash метаданных системы хранения. Используемые значения алгоритма/длины хэш-кода указаны в опции cdmi_value_hash. Если этот элемент отсутствует, он не должен использоваться. Если данный элемент присутствует, но не содержит корректной спецификации алгоритма хэширования, система может проигнорировать данный элемент метаданных, вернуть код ошибки HTTP 400 Bad Request, либо самостоятельно выбрать алгоритм и размер хэш-кода. Поддерживаемые алгоритмы хэширования выражаются строкой в форме ALGORITHM LENGTH, где: "ALGORITHM" алгоритм хэширования (например, "SHA"). "LENGTH" размер хэш-кода в байтах (например, "160", "256"). Для улучшения совместимости различных реализаций CDMI следующие обозначения должны быть использованы для наиболее распространенных параметров хэширования: "SHA160" для SHA-1, "SHA256" для SHA-2.	Опционально
cdmi_latency	Строка JSON	Если данный элемент метаданных присутствует и установлен равным строковому представлению положительного числа, он показывает, что клиент запрашивает желаемое максимальное время до первого байта, в миллисекундах. Этот элемент устанавливает желаемую задержку, измеренную от края облака, и не включает задержки между клиентом и облаком. Например, этот элемент может совместимым образом определять, какой тип хранилища следует использовать. Если этот элемент отсутствует или не равен положительному числу, он не должен использоваться.	Опционально
cdmi_throughput	Строка JSON	Если данный элемент метаданных присутствует и установлен равным строковому представлению положительного числа, он показывает, что клиент запрашивает желаемую максимальную скорость получения данных, в байтах в секунду. Этот элемент устанавливает желаемую скорость, измеренную от края облака, и не учитывает пропускную способность между клиентом и облаком. Например, этот элемент может совместимым образом определять, какой тип соединения следует использовать. Если этот элемент отсутствует или не равен положительному числу, он не должен использоваться.	Опционально

Окончание таблицы 117

Имя элемента метаданных	Тип	Описание	Требование
cdmi_ sanitization_ method	Строка JSON	Если данный элемент метаданных присутствует и не является пустой строкой, он показывает, что клиент запрашивает использование системой определенного метода очистки для удаления данных, обеспечивающего невосстановимость данных после операций обновления или удаления. Поддерживаемый метод очистки указан в опции cdmi_sanitization_method. Если этот элемент отсутствует, он не должен использоваться. Если данный элемент присутствует, но не содержит корректного определения метода очистки, система может проигнорировать данный элемент метаданных, вернуть код ошибки HTTP 400 Bad Request, либо самостоятельно определить метод очистки. Поддерживаемые методы очистки выражаются системозависимыми строками.	Опционально
cdmi_RPO	Строка JSON	Если данный элемент метаданных присутствует и установлен равным строковому представлению положительного числа, он показывает, что клиент запрашивает наибольшую продолжительность между изменением или созданием объекта и моментом, когда объект может быть восстановлен. Этот элемент метаданных используется для указания на желаемую частоту резервного копирования из первичной копии или копий во вторичную копию или копии. Это максимально приемлемый временной промежуток перед отказом или неисправностью, во время которого изменение данных может быть потеряно из-за восстановления. Если этот элемент отсутствует или не равен положительному числу, он не должен использоваться.	Опционально
cdmi_RTO	Строка JSON	Если данный элемент метаданных присутствует и установлен равным строковому представлению положительного числа, он показывает, что клиент запрашивает максимально допустимое время восстановления данных, в секундах. Этот элемент данных используется для задания приемлемого времени восстановления первичной копии или копий из вторичной копии или копий. Если этот элемент отсутствует или не равен положительному числу, он не должен использоваться.	Опционально

16.5 Поддержка метаданных, предоставленных системой данных

Для каждого элемента метаданных в системе данных имеется текущее значение, которое реализация может получить в данный момент (см. таблицу 118).

Таблица 118 – Метаданные, предоставляемые системой данных

Имя элемента метаданных	Тип	Описание	Требование
cdmi_data_redundancy_ provided	Строка JSON	Содержит текущее число полных копий объекта	Опционально
cdmi_immediate_ redundancy_provided	Строка JSON	Если присутствует и равно "true", указывает, что для объекта поддерживается резервное копирование	Опционально
cdmi_infrastructure_ redundancy_provided	Строка JSON	Содержит текущее количество независимых инфраструктур хранения, поддерживающих данные.	Опционально
cdmi_data_dispersion_ provided	Строка JSON	Содержит минимальное расстояние (км) между любыми двумя инфораструктурами, хранящими данные	Опционально
cdmi_geographic_ placement_provided	Массив JSON строк JSON	Содержит идентификатор ISO-3166, указывающий на геополитический регион, в котором храненится объект	Опционально
cdmi_retention_period_ provided	Строка JSON	Содержит временной интервал в соответствии с ISO 8601:2004 (см. 5.14), указывающий интервал отложенного удаленния объекта	Опционально
cdmi_retention_ autodelete_provided	Строка JSON	Содержит "true", если объект будет автоматически удален после окончания периода отложенного удаления	Опционально

Окончание таблицы 118

Имя элемента метаданных	Тип	Описание	Требование
cdmi_hold_id_provided	Массив JSON строк JSON	Содержит определенный пользователем идентификатор удержания, действующего в текущий момент	Опционально
cdmi_encryption_provided	Строка JSON	Содержит алгоритм, использованный для шифрования, режим работы и длину ключа. (Формат cdmi_encryption см. в табл. 117)	Опционально
cdmi_value_hash_provided	Строка JSON	Содержит алгоритм и длину хэш-кода объекта	Опционально
cdmi_latency_provided	Строка JSON	Содержит максимальное время до первого байта	Опционально
cdmi_throughput_provided	Строка JSON	Содержит максимальную скорость получения данных	Опционально
cdmi_sanitization_method_provided	Строка JSON	Содержит использованный метод очистки	Опционально
cdmi_RPO_provided	Строка JSON	Содержит продолжительность, в секундах, между обновлением и временем, когда обновление может быть восстановлено	Опционально
cdmi_RTO_provided	Строка JSON	Срдержит продолжительность восстановления данных, в секундах	Опционально

17 Управление отложенным удалением и удержанием

17.1 Введение

Облачная система хранения может опционально поддерживать средства управления отложенным удалением объектов в системе облачного хранилища. Реализация системы отложенного удаления и удержания обозначается присутствием соответствующих общесистемных опций.

Управление отложенным удалением включает политику отложенного удаления, определение политики удержания объектов для специальных целей (например, судебного оспаривания) и определение влияния правил удаления объектов на отложенное удаление и удержание. Удаление объекта CDMI™ не есть опция управления отложенным удалением как такового, но общесистемная опция. В данном разделе, тем не менее, описаны последствия применения политики удержания и отложенного удаления к объекту.

Управление отложенным удалением может применяться к следующим объектам:

- объекты данных;
- объекты-очереди;
- объекты-контейнеры.

17.2 Управление отложенным удалением

В CDMI управление отложенным удалением, удержанием и удалением оказывает влияние на всех клиентов CDMI клиент, создающих или удаляющих объекты CDMI, а также как система хранения управляет объектами с момента создания и до момента удаления.

Управление отложенным удалением CDMI состоит из трех управляющих элементов: отложенное удаление, удержание и удаление:

- отложенное удаление CDMI использует временные критерии для определения периода времени, в течение которого запрещено удаление соответствующего объекта. При этом не разрешается и изменение объекта, даже после истечения периода удержания, за исключением случаев, перечисленных ниже;
- удержание CDMI запрещает удаление или изменение объекта, пока не сняты все удерживающие блокировки;
- CDMI система не должна допускать удаление CDMI объекта пока не истечет время отложенного удаления или выполняется удержание. Любые попытки удаления объекта должны возвращать ошибку;
- после завершения отложенного удаления и снятия удержания эти механизмы не должны оказывать влияния на возможность удаления объекта;

– после начала периода отложенного удаления или наложения удержания, изменение данных или метаданных объекта не допускается, за исключением метаданных подсистем отложенного удаления и удержания. Метаданные системы отложенного удаления могут добавляться и изменяться, расширяя период отложенного удаления, а метаданные системы удержания могут модифицироваться для добавления новых удержаний. Другие попытки изменения объекта должны вызывать ошибку.

17.3 Отложенное удаление CDMI

Отложенное удаление CDMI позволяет применять в каждый момент времени лишь одну политику отложенного удаления к каждому объекту.

Управление отложенным удалением использует временной критерий для определения периода, когда удаление объекта из CDMI системы запрещено. Критерий отложенного удаления в системе CDMI обозначается следующими метаданными системы данных:

– идентификатор критерия отложенного удаления, строка, полученная от клиента CDMI и идентифицирующая класс записи о постоянном хранении (cdmi_retention_id);

– начало периода отложенного удаления и его продолжительность (cdmi_retention_period).

Если клиент CDMI пытается удалить объект, облачная система хранения должна проверить, выполняется ли критерий отложенного удаления, и вернуть ошибку в случае, если удаление невозможно.

При копировании объекта, к которому применена политика отложенного удаления, свойства, связанные с отложенным удалением, не должны переноситься от источника к создаваемому объекту, и копия не попадает под действие политики отложенного удаления.

На рисунке 10 показано, как установить отложенное удаление по времени с помощью идентификатора отложенного удаления.

Рисунок 10 – Отложенное удаление объекта

Определенный код ошибки HTTP (403) должен возвращаться при попытке изменить или удалить объект, находящийся на постоянном хранении.

Облачная система хранения не должна препятствовать изменениям метаданных, расширяющим период отложенного удаления.

17.4 Удержание CDMI

Удержание CDMI устанавливает доступ к объекту только для чтения и запрещает его удаление. Облачная система хранение должна допускать множественные удержания для одного и того же объекта.

Если объект удерживается, облачная система хранения устанавливает доступ к объекту только для чтения и запрещает его удаление.

При копировании удерживаемого объекта, его свойства, связанные с удержанием, не должны копироваться в новый объект, и копия объекта на должна удерживаться.

Управление удержанием использует идентификатор удержания для определения периодов времени, в которые объект CDMI и его метаданные не могут быть изменены или удалены. Критерии удержания CDMI объекта должны указываться метаданными системы данных, в частности, предоставленным клиентом строковыми идентификаторами, которые определяют удержания и их порядок.

Клиент CDMI может поместить объект в удержание добавлением соответствующего идентификатора в элемент метаданных системы данных cdmi_hold_id. Когда объект удерживается, операции клиента CDMI, связанные с изменением объекта (успешные в обычном состоянии объекта) должны возвращать ошибку.

На рисунке 11 показано, как установка удержания на объект влияет на доступ к нему для изменения или удаления.

Рисунок 11 – Удержание объекта

На рисунке 12 показано совместное действие идентификаторов удержания и отложенного удаления. Значение метаданных системы данных не должно изменяться в период отложенного удаления; метаданные для идентификаторов удержания не запрещают удаление удержания.

Удаление удержания находится за рамками настоящего стандарта.

Рисунок 12 – Удержание объекта при постоянном хранении

На рисунке 13 показано, как установка нескольких удержаний на объект влияет на доступ к нему для изменения или удаления.

Рисунок 13 – Объект с множественными удержаниями

Облачная система хранения должна поддерживать возможность чтения удерживаемого объекта, но запрещать его удаление, как явное, так и автоматическое.

Клиенты CDMI должны допускать отказы операций из-за удержания или изменение его состояния.

Отмена удержания находится за рамками настоящего стандарта и обычно производится при помощи отдельного механизма, определяемого реализацией.

Определенный код ошибки HTTP (403) должен возвращаться при попытке изменить или удалить объект, находящийся в состоянии удержания. Для приложений это должно означать ошибку.

17.5 Автоудаление CDMI

Удаление CDMI управляет облачным хранилищем в аспектах, касающихся удаления объектов. Облачная система хранения может автоматически удалить объ-ект, как только для него исчезли блокирующие условия удержания или отложенного удаления (см. cdmi_retention_autodelete в таблице 117.)

Объекты CDMI должны автоматически удаляться системой после окончания времени отложенного удаления, если соответствующим образом установлен флаг cdmi_retention_autodelete. Этот флаг показывает системе, что доступ к объекту будет невозможен после выполнения критерия отложенного удаления.

Система должна обеспечить, что объект не доступен более через CDMI интер-фейс. Если критерий отложенного удаления выполнен, но объект находится в состоянии удержания, система не должна прекращать доступ к объекту или удалять его. Если к объекту одновременно применяются отложенное удаление и удержание, обе блокировки должны быть сняты для автоматического удаления объекта.

17.6 Замечания о безопасности отложенного удаления

Точность и целостность значений начала отложенного удаления и его продолжительность зависят от точности и целостности таймера, использованного для их установки. Одинаково важны относительная точность и безопасность таймера, который определяет, закончился ли период отложенного удаления и часов, использованных для установки начала хранения. Относительная разница во времени между этими часами может привести к нежелательному поведению подсистем отложенного удаления и удаления.

Важно устанавливать системные часы по надежному источнику. Время слоя 1 непосредственно устанавливается по стандартным часам и находится на вершине иерархии серверов времени. Относительная разница времени между стандартными и системными часами и эталонными часами может приводить к нежелательному отметкам времени отложенного удаления и проблемам в обработке событий таймера.

Пример – Объект создан в облачном хранилище в момент времени 0 с периодом хранения 8 лет и включенным автоматическим удалением. В момент времени 1 год системные часы переведены вперед на 9 лет. Теперь, несмотря на то, что фактическое время существования объекта равно 1 году, он будет удален, так как выполнен соответствующий критерий.

Спецификация точности и целостности хранения времени находится за рамка-ми настоящего стандарта. Тем не менее, необходимо отметить важность этих моментов для обеспечения правильности поведения системы.

18 Спецификация условий запросов

18.1 Введение

CDMI™ предоставляет стандартизованный механизм для определения наборов объектов, обладающих определенными свойствами. Этот механизм известен как спецификация условмй запросов (scope) CDMI. Спецификации условий запросов обычно используются для предоставления клиенту CDMI возможности определения интересующей группы объектов.

Каждый объект JSON в рамках спецификации условий запросов представляет набор условий, которые должны быть все истинными, чтобы объект считался соответствующим запросу (логическое отношение И). В случае запроса (query) соответствующий объект должен быть возвращен в результате запроса. Пустая спецификация условия должна рассматриваться как истинная. Для выражения логического отношения ИЛИ используются несколько объектов JSON; в этом случае объект, для которого хотя бы один из объектов JSON истинный, считается удовлетворяющим условию запроса.

Каждый объект JSON строится по тем же структурным принципам, что и объект CDMI. Для иллюстрации этой структуры рассмотрим результат запроса CDMI GET к объекту данных:

```
HTTP/1.1 200 OK
Content-Type: application/cdmi-object
X-CDMI-Specification-Version: 1.0.2
{
«objectType» : «application/cdmi-object»,
«objectID» : «00007E7F0010EB9092B29F6CD6AD6824»,
«objectName» : «MyDataObject.txt»,
«parentURI» : «/MyContainer/»,
«parentID» : «00007E7F00102E230ED82694DAA975D2»,
«domainURI» : «/cdmi_domains/MyDomain/»,
«capabilitiesURI» : «/cdmi_capabilities/dataobject/»,
«completionStatus» : «Complete»,
«mimetype» : «text/plain»,
«metadata» : {
   «cdmi_size» : «108263»
},
«valuerange» : «0-108262», «value» : «...»
}
```

18.2 Примеры

Каждое поле в объекте JSON спецификации условия запроса отражает условие, которое должно выполняться для соответствующего поля объекта.

Примеры

1 Запрос на поиск всех объектов, принадлежащих домену /cdmi_domains/MyDomain/, структурирован следующим образом:

```
[
   {
         «domainURI» : «== /cdmi_domains/MyDomain/»
   }
]
```

2 Для запроса всех объектов, принадлежащих домену /cdmi_domains/MyDomain/ И находящихся в контейнере MyContainer, спецификация условия запроса такова:

```
[
   {
         «parentURI» : «== /MyContainer/»,
         «domainURI» : «== /cdmi_domains/MyDomain/»
   }
]
```

3 Для запроса всех объектов, принадлежащих домену MyDomain ИЛИ находящихся в контейнере MyContainer, структура такова:

```
[
   {
         «parentURI» : «== /MyContainer/»,
   },
   {
         «domainURI» : «== /cdmi_domains/MyDomain/»
   }
]
```

Запросы могут относиться к любым полям, возвращаемым в результате запроса GET системой облачного хранения данных.

4 Для запроса элементов метаданных, объект metadata включается в запрос, например –

```
[
{
```

«metadata» : {
«colour» : «== blue»
}
}
]

Этот подход позволяет устанавливать соответствие с использованием произвольно вложенных структур метаданных.

Для запроса значения объекта, в запрос включается поле value. В запросах значения всегда представляются с использованием кодировки base 64.

5
{
[
{
«value»: «== Ymx1ZQ==»
}
]
}

Запрос на значение объекта опционален и обозначается присутствием опции cdmi_query_value.

18.3 Выражения для запросов

В таблице 119 приведены выражения для запросов.

Таблица 119 – Выражения для запросов

Выражение	Описание
"field" : "*"	Выражение "существует" проверяет наличие заданного поля. Если поле существует, даже если оно пусто, условие считается выполненным.
"field" : "!"	Выражение "не существует" проверяет отсутствие заданного поля. Если поле не существует, условие считается выполненным.
"field" : "== constant"	Выражение "равно" проверяет равенство значения поля и заданной константы. Эта проверка регистрозависимая. Пробел между "==" и значением константы не включается в проверку. Если значение константы равно значению поля, условие считается выполненным. Если выражение запроса начинается с "#" (то есть "#=="), значение поля считается числовым в рамках сравнения. Строковые представления чисел должны обрабатываться в соответствии с представлением JSON, описанным в RFC 4627. Если числовая константа сравнивается с нечисловым полем, условие считается не выполненным.
"field" : "!= constant"	Выражение "не равно" проверяет неравенство значения поля и заданной константы. Эта проверка регистрозависимая. Пробел между "!=" и значением константы не включается в проверку. Если значение константы не равно значению поля, условие считается выполненным. Если выражение запроса начинается с "#" (то есть "#!="), значение поля считается числовым в рамках сравнения. Строковые представления чисел должны обрабатываться в соответствии с представлением JSON, описанным в RFC 4627. Если числовая константа сравнивается с нечисловым полем, условие считается выполненным.
"field" : "> constant"	Выражение "больше" лексикографически сравнивает значения поля и заданной константы. Эта проверка регистрозависимая. Пробел между ">" и значением константы не включается в проверку. Если значение константы больше значения поля, условие считается выполненным. Если выражение запроса начинается с "#" (то есть "#>"), значение поля считается числовым в рамках сравнения. Строковые представления чисел должны обрабатываться в соответствии с представлением JSON, описанным в RFC 4627. Если числовая константа сравнивается с нечисловым полем, условие считается не выполненным.

Продолжение таблицы 119

Выражение	Описание
"field" : ">= constant"	Выражение "больше или равно" лексикографически сравнивает значения поля и заданной константы. Эта проверка регистрозависимая. Пробел между ">=" и значением константы не включается в проверку. Если значение константы больше значения поля или равно ему, условие считается выполненным. Если выражение запроса начинается с "#" (то есть "#>="), значение поля считается числовым в рамках сравнения. Строковые представления чисел должны обрабатываться в соответствии с представлением JSON, описанным в RFC 4627. Если числовая константа сравнивается с нечисловым полем, условие считается не выполненным.
"field" : "< constant"	Выражение "меньше чем" лексикографически сравнивает значения поля и заданной константы. Эта проверка регистрозависимая. Пробел между "<" и значением константы не включается в проверку. Если значение константы vtymit значения поля, условие считается выполненным. Если выражение запроса начинается с "#" (то есть "#<"), значение поля считается числовым в рамках сравнения. Строковые представления чисел должны обрабатываться в соответствии с представлением JSON, описанным в RFC 4627. Если числовая константа сравнивается с нечисловым полем, условие считается не выполненным.
"field" : "<= constant"	Выражение "меньше или равно чем" лексикографически сравнивает значения поля и заданной константы. Эта проверка регистрозависимая. Пробел между "<=" и значением константы не включается в проверку. Если значение константы меньше значения поля или равно ему, условие считается выполненным. Если выражение запроса начинается с "#" (то есть "#<="), значение поля считается числовым в рамках сравнения. Строковые представления чисел должны обрабатываться в соответствии с представлением JSON, описанным в RFC 4627. Если числовая константа сравнивается с нечисловым полем, условие считается не выполненным.
"field" : "starts constant"	Выражение "начинается с" проверяет, начинается ли значение поля с указанной константы. Пробел между "starts" и значением константы не включается в проверку. Эта проверка регистрозависимая. Если указанная константа совпадает с началом значения поля, условие считается выполненным.
"field" : "!starts constant"	Выражение "не начинается с" проверяет, что значение поля не начинается с указанной константы. Пробел между "!starts" и значением константы не включается в проверку. Эта проверка регистрозависимая. Если указанная константа не совпадает с началом значения поля, условие считается выполненным.
"field" : "ends constant"	Выражение "заканчивается на" проверяет, заканчивается ли значение поля указанной константой. Пробел между "ends" и значением константы не включается в проверку. Эта проверка регистрозависимая. Если указанная константа совпадает с окончанием значения поля, условие считается выполненным.
"field" : "!ends constant"	Выражение "не заканчивается на" проверяет, что значение поля не заканчивается указанной константой. Пробел между "!ends" и значением константы не включается в проверку. Эта проверка регистрозависимая. Если указанная константа не совпадает с окончанием значения поля, условие считается выполненным.
"field" : "contains constant"	Выражение "содержит" проверяет, включает ли значение поля указанную константу. Пробел между "contains" и значением константы не включается в проверку. Эта проверка регистрозависимая. Если указанная константа найдена как подстрока в значении поля, условие считается выполненным. Оператор "содержит" поддерживается, только если присутствует опция cdmi_query_contains.

Окончание таблицы 119

Выражение	Описание
"field" : "!contains constant"	Выражение "не содержит" проверяет, что значение поля не включает указанную константу. Пробел между "!contains" и значением константы не включается в проверку. Эта проверка регистрозависимая. Если указанная константа не найдена как подстрока в значении поля, условие считается выполненным. Оператор "содержит" поддерживается, только если присутствует опция cdmi_query_contains.
"field" : "tag constant"	Выражение "метка" проверяет, содержит ли поле указанную метку-константу. Пробел между "tag" и константой не включается в проверку. Эта проверка регистронезависимая. Если указанная константа найдена как метка-подстрока в значении поля, условие считается выполненным. Подстроки начинаются в начале строки или после "," и заканчиваются перед следующим "," либо концом строки. Пробел перед или после "," не учитывается в данной проверке. Оператор "метка" поддерживается, только если присутствует опция cdmi_query_tags.
"field" : "!tag constant"	Выражение "не метка" проверяет, что поле не содержит указанную метку-константу. Пробел между "tag" и константой не включается в проверку. Эта проверка регистронезависимая. Если указанная константа не найдена как метка-подстрока в значении поля, условие считается выполненным. Подстроки начинаются в начале строки или после "," и заканчиваются перед следующим "," либо концом строки. Пробел перед или после "," не учитывается в данной проверке. Оператор "метка" поддерживается, только если присутствует опция cdmi_query_tags.
"field" : "=~ constant"	Выражение "соответствует регулярному выражению" проверяет, удовлетворяет ли значение поля приведенному регулярному выражению-константе. Пробел между "=~" и константой не включается в проверку. Если результатом применения регулярного выражения к значению поля является значение "true", условие считается выполненным. Строковые регулярные выражения должны обрабатываться согласно стандарту POSIX Extended Regular Expression (ERE), как указано в IEEE Std 1003.1. Данная проверка поддерживается, только если присутствует опция cdmi_query_regex.
"field" : "!~ constant"	Выражение "не соответствует регулярному выражению" проверяет, что значение поля не удовлетворяет приведенному регулярному выражению-константе. Пробел между "!~" и константой не включается в проверку. Если результатом применения регулярного выражения к значению поля является значение "false", условие считается выполненным. Строковые регулярные выражения должны обрабатываться согласно стандарту POSIX Extended Regular Expression (ERE), как указано в IEEE Std 1003.1. Данная проверка поддерживается, только если присутствует опция cdmi_query_regex.

Все поля объектов, которые не включены в спецификацию условий запроса, должны быть игнорированы при поисковом запросе.

Если в качестве константы для операторов равенства или неравенства для полей parentURI, domainURI и capabilitiesURI используется URI, возможно указывать URI по пути или по ID объекта, эти способы взаимозаменяемы.

Примеры

1 В запросе на принадлежность объекта некоторому домену два типа запросов считаются эквивалентными:

```
[
{
        «domainURI» : «== /cdmi_domains/MyDomain/»
}
]
```

и
[
{
«domainURI» : «== /cdmi_objectid/00007E7F001074C86AD256DA5C67180D/»
}
]

2 Аналогично, запрос на поиск объектов с заданным родительским контейнером может иметь две формы:

[
{
«parentURI» : «== /MyContainer/»
}
]
и
[
{
«parentURI» : «== /cdmi_objectid/0000706D0010B84FAD185C425D8B537E/»
}
]

Если в поисковом запросе используется ID объекта (поля objectID или parentID), ID объектов должны обрабатываться как регистронезависимые.

19 Спецификация результатов

19.1 Введение

CDMI™ предоставляет стандартизованный механизм определения подмножеств содержимого объектов. Этот механизм известен как спецификация результатов CDMI. Спецификации результатов обычно используются для предоставления клиенту CDMI возможность определить, с каким подмножеством содержимого объекта CDMI он будет работать.

Каждый объект JSON в спецификации результатов представляется набором полей, которые возвращаются для каждого подходящего объекта.

Объект JSON результатов должен строиться аналогично структуре объектов CDMI. Для иллюстрации рассмотрим следующий результат запроса GET к объекту CDMI:

HTTP/1.1 200 OK
Content-Type: application/cdmi-object
X-CDMI-Specification-Version: 1.0.2
{
«objectType» : «application/cdmi-object»,
«objectID» : «00007E7F0010EB9092B29F6CD6AD6824»,
«objectName» : «MyDataObject.txt»,
«parentURI» : «/MyContainer/»,
«parentID» : «00007E7F00102E230ED82694DAA975D2»,
«domainURI» : «/cdmi_domains/MyDomain/»,
«capabilitiesURI» : «/cdmi_capabilities/dataobject/»,
«completionStatus» : «Complete»,
«mimetype» : «text/plain»,
«metadata» : {
«cdmi_size» : «108263»
},
«valuerange» : «0-108262»,
«value» : «...»
2}

19.2 Примеры

Каждое поле JSON объекта спецификации результатов указывает, какое поле должно включаться в результат.

Примеры –

1 Данная спецификация результатов требует, чтобы в результате были возвращены поля метаданных objectID и cdmi_size:

```
{
«cdmi_results_specification» : {
«objectID» : «»,
«metadata» : {
        «cdmi_size» : «»
}
}
}
```

2 Если объект подходит под запрос, результат ставится в очередь как:

```
{
«objectID» : «00007E7F0010EB9092B29F6CD6AD6824»,
«metadata» : {
«cdmi_size» : «108263»
}
}
```

Чаще всего клиент запрашивает в спецификации cdmi_results_specification либо objectID, либо objectName и parentURI, либо все три поля. Если запрашивается parentURI или objectName, это поле должно быть возвращено только для объектов, расположенных в контейнере.

3 Для запроса всех элементов метаданных каждого подходящего под запрос объекта, следует использовать такую спецификацию cdmi_results_specification:

```
{
«cdmi_results_specification» : {
«metadata» : «»
}
}
```

4 Для запроса всех полей и всех элементов метаданных каждого подходящего под запрос объекта, следует использовать такую спецификацию cdmi_results_specification:

```
6061        {
  «cdmi_results_specification» : «»
  }
```

Значение поля всегда возвращается в кодировке base 64, если включено в результат запроса. При этом поле valuetransferencoding указывает на ожидаемую кодировку, если запрос GET выполняется для чтения объекта.

20 Ведение журналов событий

20.1 Обзор

Ведение журналов событий в системе CDMI™ делится на три функциональные области, различающиеся уровнем детализации. Они включают:

- журналы событий объектов;
- журналы событий безопасности;
- журналы событий управления данными.

CDMI не определяет формат сообщений журналов событий. Предполагается, что это станет предметов стандартов ведения журналов событий в будущем.

Клиенты CDMI получить доступ к данным журналов событий посредством создания очереди журналов событий, которая определяет содержание сообщений о событиях, которые необходимо получить (см. 20.5). Если пользователь имеет достаточный уровень разрешений для создания очереди журнала событий, все сообщения журнала, на которые он подписан, будут ставиться в эту очередь, которая затем может быть доступна для обработки или архивного хранения.

Если определены несколько очередей журнала событий, каждая очередь получит запись о соответствующем событии. Если нет очередей, которые подписаны на то или иное событие или класс сообщений, такие сообщения могут не создаваться облачной системой хранения.

20.2 Ведение журналов событий объекта

Если ведение журналов событий поддерживается облачной системой хранения, все операции, производимые над объектами CDMI (объекты данных, контейнеры, домены, очереди и опции), должны быть сохранены во все определенные очереди журналов событий.

Сообщения журнала событий должны содержать минимум приведенной ниже информации в формате, определенном реализацией:

- время в формате ISO-8601 (см. 5.14);
- домен, в котором была выполнена операция;
- выполненная операция;
- URI объекта, над которым была проведена операция;
- информация о принципале, от имени которого была проведена операция;
- результат операции.

Операции, сохраненные в журнале событий, должны включать операции, проведенные экспортированной файловой системой.

20.3 Ведение журналов событий безопасности

Все события, связанные с безопасностью, включая установку сессии, аутентификацию и авторизацию, модификации доменов и делегирования, должны быть записаны в журналы событий безопасности. Ведение журналов событий безопасности включает управление пользователями и доменами, события, связанные с разрешениями (например, валидация списка отзыва сертификатов), и должно включать операции за пределами системы, влияющие на безопасность облачного хранилища (тое есть изменения свойств безопасности домена CDMI через графический интерфейс администратора).

Если облачная система хранения поддерживает очереди типа cdmi_logging_queue и класс cdmi_logging_class из cdmi_security_logging (см. 20.5), это означает, что система поддерживает ведение журнала событий аудита. В свою очередь, общесистемная опция cdmi_security_audit (см. таблицу 101 в 12.1.3) должна быть установлена в «true». В противном случае cdmi_security_audit не должна присутствовать.

20.4 Ведение журналов событий управления данными

В дополнение к записи сообщений, относящихся непосредственно к изменению метаданных системы данных, в журнал событий должны включать все условия, при которых могут изменяться метаданные системы данных для объектов. Например, если клиент изменяет количество необходимых копий объекта, должно инициироваться событие, приводящее к появлению соответствующей записи в журнале событий. Аналогично, изменение системой текущего количества копий объекта также должно быть отражено в журнале событий.

Данный тип журналов событий также должен включать сообщения об удержаниях и отложенном удалении объектов.

20.5 Очереди журналов событий

Очереди журналов событий позволяют клиентам CDMI получать подробную информацию о событиях, связанных с работой системы облачного хранения. Так как данные в очереди являются хранимыми (persistent), от клиента не требуется поддерживать сессию. Если разные клиенты используют различные очереди журналов событий, каждый клиент действует независимо (например, анализирующее приложение получает информацию о действиях в определенном домене или над группой объектов с использованием специально сконфигурированной очереди).

Очереди журналов событий отличаются от очередей уведомлений (см. раздел 21) тем, что в первом случае предоставляется значительно более подробная информация, и ведение таких очередей ограничено небольшим подмножеством привилегированных клиентов.

Когда клиент запрашивает информацию журналов событий, он может вначале проверить, способна ли система предоставить такую информацию. Для этого необходимо проверить наличие опции cdmi_logging среди опций корневого контейнера. Если данная опция отсутствует, очередь журналов событий будет создана, но записи в нее добавляться не будут.

При создании очереди журналов событий должны быть предоставлены метаданные, описанные в таблице 120. Попытки изменения элементов метаданных, приведенных в этой таблице, должно возвращать ошибку HTTP 403 Forbidden. После создания очереди журналов событий, метаданные, приведенные в таблице, не могут быть изменены, за исключением cdmi_queue_type; последний элемент может быть только удален, показывая системе, что очередь журнала событий больше не будет принимать сообщения журнала и должна обрабатываться как обычный объект-очередь CDMI.

Таблица 120 – Необходимые метаданные для очереди журнала событий

Имя метаданных	Тип	Описание	Требование
cdmi_queue_type	Строка JSON	Тип очереди показывает, как облачная система хранения должна обрабатывать объект-очередь. Для очереди журнала событий определен тип cdmi_logging_queue.	Обязательно
cdmi_logging_class	Массив JSON строк JSON	Содержит массив JSON, показывающий, сообщения каких журналов событий должны помещаться в очередь. Допустимые значения: - cdmi_object_logging – Получение сообщений об операциях с объектами; - cdmi_datasystem_logging - Получение сообщений об изменении метаданных системы данных; - cdmi_security_logging - Получение сообщений о событиях безопасности. Клиенты могут включать желаемые классы сообщений о событиях в массив cdmi_logging_class. Если необходимо получать сообщения всех типов, следует указать пустой массив JSON.	Обязательно
cdmi_scope_ specification	Массив JSON объектов JSON	Спецификация задачи условий запроса определяет набор объектов, сообщения о событиях для которых помещаются в очередь. Если необходим сбор журналов событий, относящихся ко всем объектам, следует использовать пустой JSON массив. Для ведения журналов событий безопасности, спецификация задачи игнорируется. Описание построения спецификации цели описано в разделе 18.	Обязательно

Примеры

1 Пример метаданных, связанных с очередью журналов событий:

```
{
«metadata» : {
«cdmi_queue_type» : «cdmi_logging_queue»,
«cdmi_logging_class» : [
«cdmi_object_logging»,
«cdmi_security_logging»
],
«cdmi_scope_specification» : [
        {
                «domainURI» : «== /cdmi_domains/MyDomain/»
}
]
}
}
```

Когда сообщения журнала извлекаются из очереди, содержимое каждого значения в очереди должно содержать объект JSON и иметь тип MIME «application/json». Этот объект JSON содержит одну или более строку/объект JSON, каждая из которых соответствует одному сообщению журнала.

Сообщения включаются в очереди журналов событий, только если пользователь, создавший очередь, имеет права доступа к объекту, с которым связано сообщение о событии (т.е., пользователь обладает подходящей записью ACE из 16.1.5).

2 Если очередь журналов событий была создана администратором, то все подходящие объекты, без ограничений, включаются в результат. Если очередь журналов событий создана пользователем «jdoe», то в результат будут включены лишь сообщения о событиях, относящиеся к объектам, к которым «jdoe» имеет доступ.

В таблице 121 приведены метаданные, созданные системой и дающие подробную информацию об очереди журналов событий.

Таблица 121 – Метаданные статуса ведения журналов событий

Имя метаданных	Тип	Описание	Требование
cdmi_logging_ status	Строка JSON	Строка, показывающая состояние ведения журналов событий. Определены следующие значения: - Processing – Очередь журналов событий находится в состоянии поиска результатов; - Halted – Новые сообщения о событиях не будут более добавляться в очередь; - Current – Очередь журналов событий содержит все имеющиеся на текущий момент сообщения о событиях; - Error – Метаданные очереди сообщений о событиях некорректные, либо возникли другие ошибки, которые не позволяют включать сообщения о событиях в очередь. Строка "Error" может завершаться произвольным текстом, зависящим от реализации.	Обязательно

20.6 Замечания о безопасности при ведении журналов событий

Точность и целостность временных меток элементов журнала зависят от точности и целостности часов, использующихся для их установки. Точные временные метки важны для разрешения проблем, экспертного анализа распределенных атак, разрешения споров и доказательства транзакций, чувствительных к времени. В частности, устранение ошибок, безопасность, учет и аутентификация основываются на корреляции событий (т.е., точном знании порядка событий), и эти соображения безопасности зависят от хорошей синхронизации времени.

Спецификация точности и целостности временных меток находится за рамками настоящего стандарта; тем не менее, для демонстрации надежности временных меток их следует приводить к стандартному времени, и необходимо показать, что системное время не может быть произвольным образом изменено.

21 Очереди уведомлений

Облачная система хранения может опционально реализовывать функциональность уведомлений. Наличие этой функциональности обозначается присутствием соответствующих общесистемных опций и предполагает поддержку очередей CDMI™.

Очереди уведомлений позволяют CDMI клиентам эффективно обнаруживать, какие изменения происходят в системе. Так как очередь уведомлений является хранимой (persistent), не требуется, чтобы клиент постоянно поддерживал сессию. Если разными клиентами используются различные очереди уведомлений, каждый клиент действует независимо от остальных (например, приложение для управления хранением может использовать очередь уведомлений для поддержания своей базы данных без необходимости проводить полное сканирование контейнера для определения, какие объекты данных были добавлены, изменены или удалены).

Если клиент запрашивает получение уведомлений, в первую очередь он может проверить, способна ли система генерировать такие уведомления (проверка наличия опции cdmi_notification среди опций корневого контейнера). Если эта опция отсутствует, создание очереди уведомлений будет успешным, но никакие уведомления не будут добавляться в эту очередь.

Для создания очереди уведомлений клиент создает обычную очередь CDMI и добавляет метаданные, указывающие системе хранения, что данная очередь должна обрабатываться как очередь уведомлений. Эти метаданные также показывают системе, какие типы уведомлений должны генерироваться, и какая информация должна включаться в каждое уведомление.

После создания очереди уведомлений последующие подходящие события должны приводить к возникновению соответствующих уведомлений и добавлению их в очередь. Стандарт CDMI не требует специального упорядочения уведомлений, и клиент должен быть способен обрабатывать уведомления, поступившие вне очереди.

При создании очереди уведомлений необходимо предоставить метаданные, указанные в таблице 122. Попытки изменения метаданных, указанных в этой таблице, должны приводить к ошибке HTTP 403 Forbidden. После создания очереди уведомлений, элементы этих метаданных не должны меняться, за исключением cdmi_queue_type; последний элемент может быть только удален, показывая системе, что очередь уведомлений теперь не должна принимать уведомления и должна обрабатываться как обычная очередь CDMI.

Таблица 122 – Необходимые метаданные для очереди уведомлений

Имя метаданных	Тип	Описание	Требование
cdmi_queue_ type	Строка JSON	Тип очереди определяет, как облачная система хранения обрабатывает объект-очередь. Для очереди уведомлений определен тип cdmi_notification_queue.	Обязательно
cdmi_notification_events	Массив JSON строк JSON	Содержит массив JSON, определяющий, какие события вызывают уведомления. Определены следующие значения: - cdmi_create_processing – Уведомление генерируется когда новый объект находится в процессе создания. - cdmi_create_complete – Уведомление генерируется, когда создан новый объект или когда состояние создаваемого объекта меняется из "Processing" в "Complete". - cdmi_read – Уведомление генерируется при чтении объекта. - cdmi_modify_processing – Уведомление генерируется при существующий объект, когда находится в процессе изменения. - cdmi_modify_complete – Уведомление генерируется, когда существующий объект был изменен или когда состояние изменяемого объекта меняется из "Processing" в "Complete". - cdmi_rename – Уведомление генерируется, когда объект переименован в ходе операции перемещения. - cdmi_copy – Уведомление генерируется, когда новый объект создан в результате копирования. - cdmi_reference – Уведомление генерируется при создании ссылки. - cdmi_delete – Уведомление генерируется при удалении объекта. - cdmi_export – Уведомление генерируется при экспорте контейнера. - cdmi_snapshot – Уведомлене создается пр создании снимка состояния контейнера. <события, зависящие от реализации> Клиенты могут включать необходимые типы событий в массив cdmi_notification_events. Если требуются уведомления всех типов, следует указать пустой массив JSON.	Обязательно
cdmi_scope_ specification	Массив JSON объектов JSON	Спецификация условий запроса определяет набор объектов, операции над которыми генерируют уведомления. Если уведомления требуются при операции над всеми объектами, следует указать пустой массив JSON. Создание спецификации условий запроса описано в разделе 18.	Обязательно
cdmi_results_ specification	Объект JSON	Содержит поля JSON, которые должны быть возвращены для каждого объекта, подходящего под спецификацию условий запроса. Создание спецификации результата описано в разделе 19. Дополнительно к полям, определенным в разделе 19, для уведомлений определены следующие поля: - cdmi_event – Указывает на событие, вызвавшее уведомление, в соответствии с полем cdmi_notification_events; - cdmi_event_result – Указывает на статус события, вызвавшего уведомление. Статус совпадает с возвращаемым статусом HTTP, т.е., 200 OK, 404 Not Found, и т.д.; - cdmi_event_time – Указывает время события, вызвавшего уведомление, в формате ISO-8601 (см. 5.14 и ISO 8601:2004); - cdmi_event_user – Указывает имя пользователя (имя ACL), который вызвал событие, сгенерировавшее уведомление. Если событие вызвано системой, это поле должно быть пустой строкой.	Обязательно

Пример – Метаданные, связанные с очередью уведомлений:

```
{
«metadata» : {
«cdmi_queue_type» : «cdmi_notification_queue»,
«cdmi_notification_events» : [
«cdmi_create_complete»,
«cdmi_read»,
«cdmi_modify_complete»,
«cdmi_delete»
```

```
],
«cdmi_scope_specification» : [
{
«domainURI» : «== /cdmi_domains/MyDomain/»,
«parentURI» : «starts /sandbox»,
«metadata» : {
«cdmi_size» : «>+100000»
}
}
],
«cdmi_results_specification» : {
«cdmi_event» : «»,
«cdmi_event_result» : «»,
«cdmi_event_time» : «»,
«objectID» : «»,
«metadata» : {
«cdmi_size» : «»
}
}
}
}
```

Если результаты уведомления хранятся в очереди уведомлений, каждый элемент очереди должен быть объектом JSON типа MIME «application/json». Этот объект JSON содержит значения, запрошенные согласно полую метаданных cdmi_results_specification очереди уведомлений.

Пример – JSON результата уведомления:

```
{
«cdmi_event» : «cdmi_read»,
«cdmi_event_result» : «200 OK»,
«cdmi_event_time» : «2010-11-15T13:12:52.342324Z»,
«objectID» : «00007E7F0010EB9092B29F6CD6AD6824»,
«metadata» : {
«cdmi_size» : «108263»
}
}
```

Объекты должны включаться в уведомление лишь только если пользователь, создавший очередь уведомлений, обладает правами на чтение соответствующего объекта.

Если очередь уведомлений была создана администратором, то все подходящие объекты, без ограничений, включаются в результат. Если очередь уведомлений создана пользователем «jdoe», то в результат будут включены лишь уведомления-логи, относящиеся к объектам, к которым «jdoe» имеет доступ.

В таблице 123 приведены метаданные, созданные системой, показывающие подробности состояния очереди уведомлений.

Таблица 123 – Метаданные состояния уведомлений

Имя метаданных	Тип	Описание	Требование
cdmi_notification_status	Строка JSON	Строка, указывающая на состояние очереди уведомлений. Определены значения: - Processing – Очередь уведомлений находится в состоянии поиска результатов; - Halted – Новые уведомления не будут более добавляться в очередь; - Current – Очередь уведомлений содержит все имеющиеся на текущий момент уведомления; - Error – Метаданные очереди уведомлений некорректные, либо возникли другие ошибки, которые не позволяют добавлять уведомления в очередь. Строка "Error" может завершаться произвольным текстом, зависящим от реализации. Отсутствие данного элемента метаданных означает, что уведомления еще не начали добавляться в очередь.	Обязательно

22 Очереди запросов

22.1 Обзор

Облачная система хранения может опционально реализовывать функциональность запросов метаданных и/или полнотекстовых запросов. Наличие этой функциональности обозначается присутствием соответствующих общесистемных опций и предполагает поддержку очередей CDMI™.

Очереди запросов позволяют клиентам CDMI эффективно обнаруживать, какое содержимое соответствует заданным условиям поиска по метаданным или полному тексту. Клиенты создают или обновляют очередь запросов, указывая метаданные, которые определяют критерии соответствия (условия запросов), а также указывая результаты, которые должны быть возвращены для подходящих объектов (результат запроса). Реализация CDMI должна затем провести поиск по имеющемуся содержимому и сохранить результат поиска в очереди запросов. После нахождения результатов запроса, они добавляются в очередь, а когда поиск завершен, элемент метаданных cdmi_query_status очереди изменяется, чтобы указать на завершение поиска. Подходящие объекты, созданные или измененные за время отработки поискового запроса, могут включаться или не включаться в результат (например, как следствие связности в конечном итоге).

Если клиент запрашивает поиск по запросу, в первую очередь он должен проверить, способна ли система выполнять такой поиск (проверка наличия опции cdmi_query среди опций корневого контейнера). Если эта опция отсутствует, создание очереди запросов будет успешным, но никакие результаты не будут попадать в эту очередь.

При создании очереди сообщений необходимо предоставить метаданные, указанные в таблице 124. Попытки изменения метаданных, указанных в этой таблице, должны приводить к ошибке HTTP 403 Forbidden. После создания очереди сообщений элементы этих метаданных не должны меняться, за исключением cdmi_queue_type. Если значение cdmi_queue_type изменяется с «cdmi_query_queue», это изменение указывает системе, что обрабатывающийся поисковый запрос должен быть остановлен, очередь запросов более не принимает результатов запросов, и должна обрабатываться как обычный объект-очередь CDMI. Для начала нового поиска с использованием существующей очереди, значение cdmi_queue_type должно быть изменено обратно на «cdmi_query_queue». Настоящий стандарт не определяет механизма приостановки выполняющегося поиска и возобновления остановленного поиска.

Таблица 124 – Необходимые метаданные для очереди запросов

Имя метаданных	Тип	Описание	Требование
cdmi_queue_type	Строка JSON	Показывает, как облачная система хранения должна обрабатывать очередь. Для очереди запросов определен тип cdmi_query_queue.	Обязательно
cdmi_scope_specification	Массив JSON объектов JSON	Спецификация условий запроса определяет, какие объекты включаются в результат запроса. Это эквивалентно оператору "WHERE" в языках типа SQL. Для запроса по всем объектам, следует указать пустой массив JSON. Конструирование спецификации условий запроса описано в гл. 18.	Обязательно
cdmi_results_specification	Объект JSON	Содержит JSON поля, которые должны быть возвращены для каждого объекта, подходящего под запрос. Это эквивалент оператора "SELECT" в языках типа SQL. Конструирование спецификации результатов описано в разделе 19.	Обязательно

Пример – Метаданные, связанные с очередью запросов:

```
{
«metadata» : {
«cdmi_queue_type» : «cdmi_query_queue»,
«cdmi_scope_specification» : [
{
«domainURI» : «== /cdmi_domains/MyDomain/»,
«parentURI» : «starts /sandbox»,
«metadata» : {
«cdmi_size» : «#> 100000»
}
}
```

```
],
«cdmi_results_specification» : {
«objectID» : «»,
«metadata» : {
«cdmi_size» : «»
}
}
}
}
46
```

Если результат хранится в очереди запросов, каждый элемент очереди должен быть объектом JSON типа MIME «application/json». Этот объект JSON содержит запрошенные значения согласно метаданным очереди запросов cdmi_results_specification.

Пример – Объект JSON с результатом запроса:

```
{
«objectID» : «00007E7F0010EB9092B29F6CD6AD6824»,
«metadata» : {
«cdmi_size» : «108263»
}
}
```

В таблице 125 приведены созданные системой метаданные, детально описывающие статус очереди запросов.

Таблица 125 – Метаданные статуса запроса

Имя метаданных	Тип	Описание	Требование
cdmi_query_status	Строка JSON	Если присутствует, этот элемент метаданных показывает состояние очереди запросов. Определены следующие значения: - Processing – Очередь запросов находится в состоянии поиска результатов; - Halted – Новые результаты не будут более добавляться в очередь; - Current – Очередь запросов содержит все имеющиеся на текущий момент результаты; - Error – Метаданные очереди запросов некорректные, либо возникли другие ошибки, которые не позволяют добавлять результаты в очередь. Строка "Error" может завершаться произвольным текстом, зависящим от реализации.	Обязательно

Объекты должны включаться в результат запроса, только если пользователь, создавший очередь запросов, обладает правами на чтение соответствующего объекта. Если очередь запросов была создана администратором, то все подходящие объекты, без ограничений, включаются в результат. Если очередь запросов создана пользователем «jdoe», то в результат будут включены лишь объекты, к которым «jdoe» имеет доступ.

22.2 Расширение запроса CDMI

Разработчик сервера CDMI может расширить запросы CDMI добавлением специальных выражений-проверок. Если разработчик добавляет в реализацию специальные элементы метаданных, эти поля также могут обрабатываться поисковым запросом с использованием базовой функциональности.

Разработчик сервера CDMI может расширить запросы CDMI, разрешив создавать специальные очереди запросов, отличающиеся по типу от cdmi_query_queue.

Приложение А
(обязательное)

Безопасность передачи

А.1 Введение

Во многих реализациях CDMI™ протокол передачи гипертекста (Hypertext Transfer Protocol, HTTP) служит для передачи CDMI сообщений. В данном приложении описаны детали, связанные с обеспечением безопасности этого транспортного протокола.

А.2 Общие требования к реализациям HTTP

Требования безопасности к реализациям HTTP относятся и к клиентам, и к серверам CDMI. Клиент CDMI должен соответствовать требованиям безопасности, которые накладывает на клиентов использование HTTP. Следующие общие требования обеспечивают безопасность передачи при использовании HTTP:

- необходима реализация либо базовой аутентификации HTTP, либо дайджест-аутентификации HTTP;
- пользовательские идентификаторы и права доступа (пароли) должны использовать базовую аутентификацию HTTP ТОЛЬКО в сочетании с безопасностью транспортного уровня (Transport Layer Security, TLS);
- пользовательские идентификаторы и права доступа, использованные в одном из типов аутентификации HTTP (т.е., базовой или дайджест) никогда не должны затем использоваться с другим типом аутентификации HTTP. Чтобы не снижать целостность более сильной системы защиты, одни и те же пароли не должны использоваться в системах защиты разной силы;
- в CDMI следует использовать по меньшей мере TLS 1.0, а использование более новых версий TLS (например, v1.1 и v1.2) очень желательно. Использование TLS в системах CDMI опционально, но должно использоваться для защиты критических данных;
- хотя HTTP должен быть реализован всеми сущностями CDMI, его использование опционально.

Реализация использования HTTP над TLS (HTTPS) опциональна. К таким реализациям предъявляются следующие требования:

- Для обеспечения минимального уровня безопасности и совместимости различных реализаций следует поддерживать следующие системы шифрования:
  - TLS_DHE_DSS_WITH_3DES_EDE_CBC_SHA (обязательно для TLS 1.0),
  - TLS_RSA_WITH_AES_128_CBC_SHA (обязательно для TLS 1.1/1.2), и
  - TLS_RSA_WITH_NULL_SHA (для TLS без шифрования).

Примечание — Разработчики могут использовать дополнительные системы шифрования, но для них нет гарантии совместимости различных реализаций;

- Для связи клиентов и серверов необходимо использовать совместимый подход к безопасности. Правильно конфигурированные клиент и сервер не смогут сообщаться, если один из них использует порт 80, а другой – порт 443. Клиенты, которые не могут подключиться к CDMI серверу по протоколу HTTPS по порту TCP 443, должны воспользоваться HTTP портом 80, если это допускается их политикой безопасности;
- серверы могут ускорять поиск клиентом безопасного канала, отвечая контактам HTTP на TCP порт 80 статусом HTTP REDIRECT на подходящий HTTPS: URI (HTTP над TLS на порт TCP 443), чтобы избежать задержки после запроса клиентом по HTTP. В такой ситуации клиенты должны вести себя в соответствии с указанной переадресацией;
- все сертификаты, включая корневые сертификаты CA, используемые клиентом для проверки сертификата, должны быть заменяемыми;
- должны быть поддержка сертификатов в кодировках DER X.509, base 64 X.509 и формате PKCS#12;
- списки отозванных сертификатов (Certificate Revocation Lists) должны поддерживаться в форматах кодировок DER X.509 и base 64 X.509.

Примечание — Так как в области безопасности нет абсолютных правил, если указанные версии оказываются уязвимыми или некорректными, реализации CDMI как можно скорее должны переключиться на использование обновленной версии TLS и более сильное шифрование.

А.3 Базовая безопасность HTTP

HTTP является обязательным механизмом передачи в данной версии CDMI. Важно отметить, что HTTP сам по себе не предоставляет никакой конфиденциальности или защиты целостности.

Клиенты CDMI сами обеспечивают аутентификацию пользователей на каждом сервере CDMI, к которому требуется доступ. Сервер CDMI действует как аутентификатор и получает права доступа пользователя через операции аутентификации HTTP.

IETF RFC 2616 и IETF RFC 2617 определяют требования к аутентификации HTTP, которая обычно начинается запросом HTTP от клиента, например, <GET Request-URI> (где Request-URI – запрошенный ресурс). Если запрос клиента не включает заголовок «Authorization», и авторизация обязательна, сервер отвечает кодом 401 Unauthorized и строкой заголовка WWW-Authenticate.

Клиент HTTP должен затем ответить с использованием подходящей строки заголовка Authorization в следующем запросе. Формат строк заголовка WWW-Authenticate и Authorization зависит от типа необходимой аутентификации (базовой или дайджест). Если аутентификация пройдена успешно, HTTP сервер возвращает код 200 OK.

Базовая аутентификация включает отсылку имени пользователя и пароля в открытом виде, и должна поэтому использоваться только в защищенной сети или в сочетании с механизмом, обеспечивающим конфиденциальность, например, TLS (см. А.4). Дайджест-аутентификация отсылает безопасный хеш пользовательского имени и пароля (и другой необходимой информации, включая контрольное слово), так что пароль не показывается. Ответ 401 Unauthorized не должен включать выбор аутентификации.

Аутентификация клиента на сервере CDMI основана на сервисе аутентификации (локальном и/или внешнем). Могут поддерживаться различные схемы аутентификации, включая узловые, Kerberos, PKI и другие, рассмотрение сервисов аутентификации находится за рамками настоящего стандарта.

А.4 HTTP над TLS (HTTPS)

CDMI может также включать механизм для обеспечения безопасности передачи данных по HTTP, когда данные, пересылаемые между сервером и клиентом предварительно шифруются. Эта безопасность достигается передачей HTTP поверх TLS (т.е. HTTPS); URI безопасного соединения начинается с https:// вместо http://. Важно, что клиент CDMI взаимодействует с сервером CDMI по HTTPS через порт TCP 443 (порт TCP 80 используется для HTTP). Приложение А.5 описывает важные детали, связанные с TLS.

Если TLS используется для обеспечения безопасности HTTP, клиент и сервер обычно проводят аутентификацию объектов. Особенности этой процедуры зависят от использованной системы шифрования, которая указывает на алгоритм шифрования и алгоритм построения дайджеста для использования при TLS соединении. В широко распространенном сценарии в качестве основы однонаправленной аутентификации объекта использованются сертификаты на стороне сервера, которым клиент доверяет. Допускается как отсутствие аутентификация (например, анонимная аутентификация), так и наоборот, может требоваться взаимная аутентификация клиента и сервера.

А.5 Безопасность уровня транспорта (TLS)

Серверы CDMI должны реализовать протокол TLS; однако, его использование клиентами опционально. TLS 1.0 должен быть реализован согласно RFC 2246, TLS 1.1 описан в RFC 4346, а TLS 1.2 – в RFC 5246.

Первичная задача протокола TLS – предоставить защиту информации и целостность данных при взаимодействии двух приложений. TLS позволяет клиент-серверным приложениям осуществлять обмен даными, избегая перехвата данных, несанкционированного вмешательства или подделки сообщений. TLS располагается поверх надежного протокола транспортного уровня (например, TCP) и используется для передачи сообщений различных протоколов более высокого уровня (например, HTTP).

TLS предоставляет аутентификацию оконечных точек и безопасный обмен данными в сети посредством использования криптографии. Обычно лишь сервер проходит аутентификацию, а клиент нет; таким образом, конечный пользователь (человек или приложение) уверен в том, с чем установлено соединение. Взаимная аутентификация требует, с малочисленными исключениями, предоставления клиентом цифрового сертификата.

TLS включает три основные фазы:

- согласование сторонами поддерживаемого алгоритма;
- обмен ключами и аутентификация;
- симметричное шифрование и аутентификация сообщений.

Во время первой фазы клиент и сервер обмениваются сведениями о поддерживаемых криптосистемах (см. А.5.1), определяя, какой шифр должен быть использован, схемы обмена ключами и алгоритмы аутентификации, а также алгоритмы аутентификации сообщений (MAC). Обмен ключами и аутентификация, как правило, представлены алгоритмами шифрования открытым ключом. MAC конструируются из алгоритмов криптографических хэш-функций.

А.5.1 Криптосистемы

TLS объединяет алгоритмы создание ключа, конфиденциальности, подписи и хэширования в криптосистему. Каждой криптосистеме ставится в соответствие 16-битное число, индекс криптосистемы.

Пример – Согласование ключей RSA, RSA подпись, конфиденциальность посредством Advanced Encryption Standard (AES) в режиме Cipher Block Chaining (CBC), и алгоритм хэширования Secure Hash Algorithm (SHA-1) соответствуют в TLS шестнадцатеричному числу {0x000F}.

TLS сессия всегда начинается клиентом, который начинает обмен пакетами шифрования, отсылая сообщение установления соединения, содержащее набор поддерживаемых криптосистем (в виде их индексов). Сервер отвечает сообщением, содержащим индекс криптосистемы, выбранной из списка клиента, или строкой «abort» как

описано ниже. Хотя клиент должен упорядочить список по увеличению «силы» криптосистемы, сервер может выбрать ЛЮБУЮ из предложенных клиентом криптосистем. Таким образом, НЕ гарантируется, что при таком обмене будет выбрана самая надежная криптосистема. Если взаимно поддерживаемых криптосистем нет, соединение разрывается. Когда обмен согласований, использующих опциональные сертификаты открытых ключей и случайные даннын для генерации ключа, используемого в криптографическом алгоритме, завершен, производится обмен специальными сообщениями для переключения соединения в защищенный режим.

Для обеспечения хотя бы минимального уровня безопасности и совместимости реализаций все клиенты и серверы CDMI должны поддерживать:

– криптосистему TLS_DHE_DSS_WITH_3DES_EDE_CBC_SHA (индекс {0x0013}), обязательную криптоситсему для TLS 1.0 (см. RFC 2246 раздел 9);

– криптосистему TLS_RSA_WITH_AES_128_CBC_SHA (индекс {0x002F}), обязательную для TLS 1.2;

– криптосистема TLS_RSA_WITH_NULL_SHA (индекс {0x0002}) должна поддерживаться CDMI клиентами и серверами для обеспечения аутентифицированного незашифрованного соединения. При использовании этой криптосистемы не следует использовать базовую аутентификацию HTTP;

– криптосистема TLS_RSA_WITH_AES_128_CBC_SHA256 (индекс {0x003C}) должна поддерживаться всеми реализациями с рекомендованным стандартом TLS 1.2 в рамках перехода к 112-битному шифрованию.

Разработчики вправе использовать дополнительные криптосистемы.

А.5.2 Цифровые сертификаты

Клиенты и серверы CDMI могут быть атакованы с использованием фальшивого CDMI сервера для получения идентификатора и пароля пользователя или с использованием ненаблюдаемого прокси между клиентом и сервером CDMI. Наиболее эффективная контрмера против такой атаки – проверка клиентом сертификата сервера посредством TLS, в предположении, что фальшивый сервер не сможет получить правильный сертификат. В частности, это может быть реализовано настройкой клиента всегда использовать TLS при аутентификации HTTP и доверять лишь сертификатам от особого локального удостоверяющего центра.

При использовании с CDMI, TLS должен использовать сертификаты форомата X.509 версии 3, которые удовлетворяют прфилю, определенному в гл. 4 RFC 3280 (X.509v3 Certificate and CRL). Этот профиль сертификатов и списка отозванных сертификатов (CRL) определяет обязательные поля для включения в сертификат, а также опциональные поля и расширения, которые могут включаться в сертификат.

Серверные сертификаты должны поддерживаться всеми серверами CDMI, а клиентские сертификаты могут поддерживаться CDMI клиентами. Сервер предоставляет клиенту серверный сертификат, в свою очередь, клиент предоставляет серверу клиентский сертификат для аутентификации. Использование серверных сертификатов довольно обычно для публичных веб-серверов, предлагающих безопасное соединение через TLS, однако клиентские сертификаты используются достаточно редко, так как клиент обычно аутентифицируется другими способами.

Пример – Сайт электронной коммерции аутентифицирует клиента по номеру кредитной карты, пользовательскому имени/паролю, и т.д., когда происходит покупка. С этой точки зрения гораздо более критично, что сервер доказывает свою идентичность, и поэтому со стороны сервера используется серверный сертификат.

Такие сертификаты X.509 используют цифровую подпись для установления связи открытого ключа и идентичности. Такие подписи часто выпускаются удостоверяющими центрами (УЦ), связанными с внутренними или внешними инфраструктурами открытых ключей (PKI); однако, альтернативой может быть самоподписанный сертификат (сертификат, подписанный той же парой ключей, открытая часть которых находится в данных сертификата). Модели, связанные с этими двумя подходами, различны. В случае сертификатов PKI следует принимать во внимание иерархию доверия и доверенную третью сторону при валидации сертификата, что повышает безопасность ценой усложнения. Самоподписанные сертификаты могут использоваться в форме доверенной сети (решение о доверии находится в руках пользователей или администраторов), однако это считается менее безопасным, так как отсутствует центральный орган, который может проверить подлинность сертификата и отозвать его при необходимости. Тем не менее, даже такая пониженная безопасность в ряде случаев может быть адекватной мерой, а ее реализация гораздо проще.

В случае сертификатов PKI часто необходимо пройти по цепочке доверителей до корневого удостоверяющего центра. Такой корневой УЦ может быть внутренним УЦ, сертификат которого подписан УЦ более высокого уровня, или находиться на конце цепочки как УЦ наивысшего уровня. Последний является главной аттестационной организацией в данной схеме PKI, и его сертификат (корневой сертификат) может быть только самоподписанным. Установление якоря доверия на уровне корневого сертификата, особенно в случае коммерческих СА, может иметь нежелательные побочные эффекты, связанные с неявным доверием ко всем сертификатом, выпускаемым данным коммерческим СА. В идеале якорь должен располагаться на низшем практически доступном уровне.

А.5.2.1 Проверка сертификата

Клиенты и серверы CDMI должны провести простейшую проверку пути, расширенную проверку пути и проверку CRL как описано в гл. 6 RFC 3280, для всех представленных сертификатов. Эти проверки включают (но не ограничены) следующими:

– построение сертификата корректно;
– подпись сертификата корректна;
– дата выдачи сертификата (срок действия не истек);
– сертификат не был отозван (только для сертификатов PKI);
– цепь сертификатов построена правильно (включая сертификат стороны, участвующей в коммуникации, и его выпускающий орган, вплоть до максимально достижимого уровня цепи, только для сертификатов PKI).

Если серверы и клиенты CDMI используют списки отозванных сертификатов, они должны использовать CRL X.509 версию 2, которая соответствует требованиям раздела 5 RFC 3280 (только для сертификатов PKI.)

Когда сертификаты PKI и самоподписанные сертификаты используются совместно в одном домене, важно понимать, что уровень безопасности понижается до уровня, который обеспечивают самоподписанные сертификаты. Самоподписанные сертификаты сами по себе лишь предлагают основу для обеспечения конфиденциальности и целостности при коммуникации. Единственная проверка идентичности самоподписанных сертификатов заключается в процессах их принятия, как описано ниже.

А.5.2.2 Форматы сертификата

Все интерфейсы для конфигурирования сертификатов (в особенности импорт) должны поддерживать следующие форматы сертификатов:

– сертификат X.509 в кодировке DER. См. спецификацию и технические данные в ISO/IEC 9594-8:2008;
– сертификат X.509 в кодировке Base 64 (PEM). См. гл. 6.8 в RFC2045;
– PKCS#12. См. спецификацию и технические данные в PKS12.

Все программы проверки сертификатов должны поддерживать локальные списки отозванных сертификатов, при этом на каждый корневой сертификат должен приходиться по крайней мере один такой список. Необходима поддержка как кодировок DER и base 64, но она может осуществляться программной поддержкой одного из форматов и конвертацией в него другого формата. OCSP и другие средства мгновенной онлайн проверки сертификатов не обязательны, так как связь с выпускающим УЦ не может быть гарантирована.

А.5.2.3 Управление сертификатами

Все сертификаты и связанные с ними открытые ключи должны быть заменяемыми. Клиенты и серверы CDMI должны обладать одной из следующих возможностей:

– импорт внешним образом сгенерированного сертификата и открытого ключа;
– генерирование и установка нового самоподписанного сертификата с соответствующим открытым ключом.

При использовании клиентами и серверами CDMI сертификатов PKI все реализации должны включать поддержку возможности импорта, установки/хранения и удаления корневых сертификатов УЦ, а также поддержку множественных доверенных сертификатов УЦ. Сертификаты УЦ используются для проверки того, что данный сертификат подписан ключом приемлемого УЦ.

Все интерфейсы сертификатов должны поддерживать ограничения доступа, которые разрешают доступ лишь администраторам с соответствующими правами. Администратор должен иметь возможность заблокировать использование неопознанных сертификатов, см. А.5.2.1 и А.5.2.2.

Поддержка формата сертификата PKCS#7 опущена из списка требований. Этот формат в основном используется для онлайн взаимодействий с УЦ, эта функциональность не требуется от всех приложений CDMI, кроме того, существуют средства конвертации сертификатов PKCS#7 в и из других форматов.

А.5.2.4 Руководство по цифровым сертификатам для TLS

Для упрощения использования сертификатов, реализации CDMI должны включать конфигурируемые механизмы, позволяющие в любой момент форсировано использовать один из следующих взаимоисключающих режимов для работы с конечными сертификатами (т.е., сертификатами, не используемыми УЦ для подписи другиx сертификатов):

– не поддающиеся проверке (самоподписанные) сертификаты конечных систем, в случае использования, автоматически устанавливаются как доверенные; такие сертификаты должны определяться как корневые сертификаты, не принадлежащие УЦ, и не должны использоваться для проверки других сертификатов. Если для верификациии узла используется непосредственно сертификат УЦ, он должен быть отклонен. Для клиентов CDMI следует реализовать при наличии технической возможности вариант этого режима, при котором запрашивается решение пользователя.

Примечание — Использование этого режима должно быть ограничено обучающим или подготовительным периодом, когда устанавливаются соединения с другими облачными системами, требующими безопасного соединения. Рекомендуется выход из этого режима по таймауту;

– не поддающиеся проверке (самоподписанные) сертификаты конечных систем могут быть импортированы и установлены как доверенные вручную (подобно ручному импорту и установке корневого сертификата УЦ), но они не добавляются автоматически, когда встречаются впервые. Для ручного импорта и установки сертификата могут требоваться права администратора;
– этот режим работы считается нормальным. Все политики принятия сертификатов клиентами и серверами CDMI должны быть настраиваемыми. Настраиваемый механизм определяет, как данная реализация CDMI обрабатывает представленные сертификаты. В нормальном рабочем режиме серверы CDMI не должны принимать сертификаты от неизвестных удостоверяющих центров (т.е., когда корневые сертификаты УЦ не установлены).

Интерактивные клиенты должны предоставлять возможность запроса пользователя на принятие сертификата от неопознанной организации (т.е. соответствующий корневой сертификат УЦ не установлен в клиента), и принимать ответ, позволяющий использовать либо данный сертификат, либо все сертификаты от данного выпускающего УЦ. Серверы не должны поддерживать принятие нераспознанных сертификатов; ожидается, что ограниченное число СА будет приемлемо для клиентских сертификатов в каждом месте, где они используются.

Предварительная настройка для использования корневых сертификатов от широко известных УЦ опциональна, но она упрощает начальную настройку системы безопасности, основанной на сертификатах, так как разрешает принятие сертификатов от этих УЦ. Эти корневые сертификаты могут быть экспортированы из широко распространенных веб-браузеров.

Приложение ДА
(справочное)

Сведения о соответствии ссылочных международных стандартов национальным стандартам Российской Федерации

Таблица ДА.1

Обозначение ссылочного международного стандарта	Степень соответствия	Обозначение и наименование соответствующего национального стандарта
ИСО 3166	–	*
ИСО 4217:2008	–	*
ИСО 8601:2004	–	*
ИСО/МЭК 9594-8-94	IDT	ГОСТ Р ИСО/МЭК 9594-8-98 "Информационная технология. Взаимосвязь открытых систем. Справочник. Часть 8. Основы аутентификации"
ИСО/МЭК 14776-414	–	*
IEEE Std 1003.1, 2004	–	*

* Соответствующий национальный стандарт отсутствует. До его утверждения рекомендуется использовать перевод на русский язык данного международного стандарта. Перевод данного международного стандарта находится в Федеральном информационном фонде технических регламентов и стандартов.

Примечание – В настоящей таблице использовано следующее условное обозначение степени соответствия стандартов:
- IDT – идентичные стандарты.

Библиография

[1] [CRC], Williams, Ross, «A Painless Guide to CRC Error Detection Algorithms», Chap-ter 16, August 1993, http://www.repairfaq.org/filipg/LINK/F_crc_v3.html

[2] [OCCI], «Open Cloud Computing Interface», Version 1.1, June 2011. Specification - http://occi-wg.org/about/ specification/

[3] [PKS12], RSA Laboratories, PKCS #12: Personal Information Exchange Syntax, Ver-sion 1.0, June 1999. Specification and Technical Corrigendum - http://www.rsa.com/rsalabs/node.asp?id=2138

[4] [REST], «Representational State Transfer» - http://www.ics.uci.edu/ ~fielding/pubs/dissertation/rest_arch_style.htm

[5] [RESTful Web], Richardson, Leonard and Sam Ruby, RESTful Web Services, O'Reilly, 2007.

[6] [INCITS 464-2010], Information Technology - Information Management - Extensible Access Method (XAM™)

УДК 004.057:006.354 ОКС 35.040

Ключевые слова: информационные технологии, облачные вычисления, облачные хранения данных

Редактор *М.Ю. Сухинина*
Технический редактор *А.Б. Заварзина*
Корректор *В.Г. Смолин*
Компьютерная верстка *Д.Е. Першин*

Сдано в набор 24.09.2015. Подписано в печать 25.10.2015. Формат 60x841/8. Гарнитура Ариал.
Усл. печ. л. 18,60. Уч.-изд. л. 17.10. Тираж 30 экз. Зак 3361.

Набрано в ООО «Академиздат»
www.academizdat.com lenin@academizdat.ru

Издано и отпечатано во
ФГУП «СТАНДАРТИНФОРМ», 123995 Москва, Гранатный пер., 4.
www.gostinfo.ru info@gostinfo.ru