МЕЖДУНАРОДНЫЙ СОЮЗ ЭЛЕКТРОСВЯЗИ

МСЭ-Т T.808

(01/2005)

СЕКТОР СТАНДАРТИЗАЦИИ
ЭЛЕКТРОСВЯЗИ МСЭ

СЕРИЯ T: ОКОНЕЧНОЕ ОБОРУДОВАНИЕ ДЛЯ ТЕЛЕМАТИЧЕСКИХ СЛУЖБ

Информационные технологии − Система кодирования изображений JPEG 2000: Инструменты интерактивности, интерфейсы API и протоколы

Рекомендация МСЭ-Т T.808

МЕЖДУНАРОДНЫЙ СТАНДАРТ ИСО/МЭК 15444-9
РЕКОМЕНДАЦИЯ МСЭ-Т Т.808

Информационные технологии – Система кодирования изображений JPEG 2000: Инструменты интерактивности, интерфейсы API и протоколы

Резюме

Целью данной Рекомендации | Международного стандарта является предоставить сетевой протокол, который предусматривает диалоговую и последовательную передачу кодированных данных и файлов JPEG 2000 от сервера к клиенту. Этот протокол позволяет клиенту запрашивать только порции изображения (с помощью области, качества или уровня разрешающей способности), которые применимы к потребностям клиента. Протокол также позволяет клиенту получать доступ к метаданным или к другому содержимому из файла.

Источник

Рекомендация МСЭ-Т Т.808 была утверждена 8 января 2005 года 16-й Исследовательской комиссией МСЭ-Т (2005−2008 гг.) в соответствии с процедурой, изложенной в Рекомендации МСЭ-Т А.8. Идентичный текст также опубликован в виде документа ИСО/МЭК 15444-9.

ПРЕДИСЛОВИЕ

Международный союз электросвязи (МСЭ) является специализированным учреждением Организации Объединенных Наций в области электросвязи. Сектор стандартизации электросвязи МСЭ (МСЭ-Т) − постоянный орган МСЭ. МСЭ-Т отвечает за изучение технических, эксплуатационных и тарифных вопросов и за выпуск Рекомендаций по ним с целью стандартизации электросвязи на всемирной основе.

На Всемирной ассамблее по стандартизации электросвязи (ВАСЭ), которая проводится каждые четыре года, определяются темы для изучения Исследовательскими комиссиями МСЭ-Т, которые, в свою очередь, вырабатывают Рекомендации по этим темам.

Утверждение Рекомендаций МСЭ-Т осуществляется в соответствии с процедурой, изложенной в Резолюции 1 ВАСЭ.

В некоторых областях информационных технологий, которые входят в компетенцию МСЭ-Т, необходимые стандарты разрабатываются на основе сотрудничества с ИСО и МЭК.

ПРИМЕЧАНИЕ

В настоящей Рекомендации термин "администрация" используется для краткости и обозначает как администрацию электросвязи, так и признанную эксплуатационную организацию.

Соблюдение положений данной Рекомендации носит добровольный характер. Однако в Рекомендации могут содержаться определенные обязательные положения (например, для обеспечения возможности взаимодействия или применимости), и соблюдение положений данной Рекомендации достигается в случае выполнения всех этих обязательных положений. Для выражения необходимости выполнения требований используется синтаксис долженствования и соответствующие слова (такие, как "должен" и т. п.), а также их отрицательные эквиваленты. Использование этих слов не предполагает, что соблюдение положений данной Рекомендации является обязательным для какой-либо из сторон.

ПРАВА ИНТЕЛЛЕКТУАЛЬНОЙ СОБСТВЕННОСТИ

МСЭ обращает внимание на вероятность того, что практическое применение или реализация этой Рекомендации может включать использование заявленного права интеллектуальной собственности. МСЭ не занимает какую бы то ни было позицию относительно подтверждения, обоснованности или применимости заявленных прав интеллектуальной собственности, независимо от того, отстаиваются ли они членами МСЭ или другими сторонами вне процесса подготовки Рекомендации.

На момент утверждения настоящей Рекомендации МСЭ не получил извещение об интеллектуальной собственности, защищенной патентами, которые могут потребоваться для выполнения этой Рекомендации. Однако те, кто будет применять Рекомендацию, должны иметь в виду, что это может не отражать самую последнюю информацию, и поэтому им настоятельно рекомендуется обращаться к патентной базе данных БСЭ.

СОДЕРЖАНИЕ

РИСУНКИ

ТАБЛИЦЫ

Введение

Рекомендация МСЭ-Т Т.800 | ИСО/МЭК 15444-1 (JPEG 2000) является спецификацией, которая описывает систему сжатия изображений, что обеспечивает большую гибкость, не только для сжатия изображений, но также и для доступа в кодовый поток. Кодовый поток предоставляет некоторое количество механизмов для определения местонахождения и извлечения порций данных сжатых изображений для целей передачи, хранения, отображения или редактирования. Этот доступ позволяет осуществлять хранение, поиск и выборку данных сжатых изображений, соответствующих для заданного приложения без декодирования.

Целью этой Рекомендации | Международного стандарта является предоставление сетевого протокола, который позволяет осуществлять диалоговую и последовательную передачу кодированных данных и файлов JPEG 2000 от сервера к клиенту. Этот протокол разрешает клиенту запрашивать только порции изображения (с помощью области, качества или уровня разрешающей способности), которые применимы к потребностям клиента. Протокол также позволяет клиенту получать доступ к метаданным или к другому содержимому (контенту) из файла.

Любой организации, намеревающейся использовать эту Рекомендацию | Международный стандарт, следует тщательно рассмотреть ее применимость.

Международный союз электросвязи (МСЭ), Международная организация по стандартизации (ИСО) и Международная электротехническая комиссия (МЭК) обращают внимание на тот факт, что считается, что соответствие этой Рекомендации | Международному стандарту может затрагивать использование патента.

МСЭ, ИСО и МЭК не занимает определенной позиции относительно очевидности, законности и сферы применения этого патентного права.

Держатель этого патентного права уверил МСЭ, ИСО и МЭК, что он желает договариваться о лицензиях на основании разумных и справедливых условий с претендентами во всем мире. В этом отношении заявление держателя этого патентного права регистрируется с помощью МСЭ, ИСО и МЭК. Информация может быть получена от компаний, перечисленных в Приложении М.

Обращается внимание на возможность того, что некоторые из элементов этой Рекомендации | Международного стандарта могут быть предметом других патентных прав, отличающихся от тех, что определены в Приложении М. МСЭ, ИСО и МЭК не должны считаться ответственными за определение каких-либо или всех таких патентных прав.

МЕЖДУНАРОДНЫЙ СТАНДАРТ
РЕКОМЕНДАЦИЯ МСЭ-Т

Информационные технологии − Система кодирования изображений JPEG 2000: Инструменты интерактивности, интерфейсы API и протоколы

1 Сфера применения

Эта Рекомендация | Международный стандарт определяет, подробным образом, синтаксисы и методы для дистанционного опроса и дополнительной модификации кодовых потоков и файлов JPEG 2000 в соответствии с их определением в следующих частях документа ИСО/МЭК 15444:

- Рекомендация МСЭ-Т Т.800 | ИСО/МЭК 15444-1:2004 и ее определение кодового потока JPEG 2000 и файлового формата JP2.
- семейство файловых форматов JPEG 2000, как определяется в последующих частях документа ИСО/МЭК 15444.

В этой Рекомендации | Международном стандарте определяемые синтаксисы и методы упоминаются как Диалоговый протокол JPEG 2000, "JPIP", а диалоговые приложения, использующие протокол JPIP, упоминаются как "системы JPIP."

JPIP определяет протокол, состоящий из структурированного ряда взаимодействий между клиентом и сервером, посредством которого метаданными файла изображения, структурой и кодовыми потоками частичных или целых изображений можно обмениваться способом, эффективным для передачи информации. Эта Рекомендация | Международный стандарт включает в себя определения семантики и значений, которыми должны обмениваться, и предлагает, как они могут быть переданы далее с использованием многообразия транспортных средств существующих сетей.

С помощью протокола JPIP изменяющимися, совместимыми способами могут быть выполнены следующие задачи:

- обмен возможностями;
- согласование возможностей для использования в сеансе;
- запрос и перенос следующих элементов из разнообразия таких контейнеров, как файлы семейств JPEG 2000, кодовые потоки JPEG 2000 и файлы других контейнеров:
  - выборочные сегменты данных;
  - выборочные и определенные структуры;
  - части изображения или их связанные метаданные.

2 Нормативные справочные документы

Нижеследующие Рекомендации и Международные стандарты содержат положения, которые, путем ссылки в этом тексте, составляют положения этой Рекомендации | Международного стандарта. На момент публикации указанные издания были действующими. Все Рекомендации и Стандарты являются предметом пересмотра; поэтому всем участникам соглашений, основанных на этой Рекомендации | Международном стандарте, предлагается изучить возможность применения самого современного издания Рекомендаций и Стандартов, перечисленных ниже. Члены МЭК и ИСО поддерживают реестры действующих в данный момент Международных стандартов. Бюро стандартизации электросвязи МСЭ поддерживает перечень действующих в данный момент Рекомендаций МСЭ-Т.

- Рекомендация МСЭ-Т Т.800 (2002 г.) | ИСО/МЭК 15444-1:2004, *Информационная технология – Система кодирования изображений JPEG 2000: Основы системы кодирования.*
- Рекомендация МСЭ-Т Т.801 (2002 г.) | ИСО/МЭК 15444-2:2004, *Информационная технология – Система кодирования изображений JPEG 2000: Добавления.*
- ITU-T Recommendation T.802 (2005) | ISO/IEC 15444-3:2005, *Information technology – JPEG 2000 image coding system: Motion JPEG 2000.*
- ISO/IEC 15444-6:2003, *Information technology – JPEG 2000 image coding system – Part 6: Compound image file format.*
- IETF RFC 768 (1980), *Датаграммный протокол пользователя*. Доступно из "Всемирной паутины": <http://www.ietf.org/rfc/rfc0768.txt>.

- IETF RFC 793 (1981), *Протокол управления передачей*. Доступно из "Всемирной паутины": <http://www.ietf.org/rfc/rfc0793.txt>.
- IETF RFC 2046 (1996), *Многоцелевые почтовые расширения Интернет (MIME) [Multipurpose Internet Mail Extensions], Часть два: Типы носителей информации*. Доступно из "Всемирной паутины": <http://www.ietf.org/rfc/rfc2046.txt>.
- IETF RFC 2234 (1997), *Дополненная форма BNF для Синтаксических спецификаций: ABNF*. Доступно из "Всемирной паутины": <http://www.ietf.org/rfc/rfc2234.txt>.
- IETF RFC 2396 (1998), *Унифицированные идентификаторы ресурса (URI) [Uniform Resource Identifiers]: Общий синтаксис*. Доступно из "Всемирной паутины": <http://www.ietf.org/rfc/rfc2396.txt>.
- IETF RFC 2616 (1999), *Протокол переноса гипертекста – HTTP/1.1*. Доступно из "Всемирной паутины": <http://www.ietf.org/rfc/rfc2616.txt>.

3 Определения

Для целей этой Рекомендации | Международного стандарта применяются следующие определения.

3.1 Определения Части 1 JPEG 2000

К этой Рекомендации | Международному стандарту также применяются определения, приведенные в разделе 3 Рекомендации МСЭ-Т Т.800 | ИСО/МЭК 15444-1:2004 и в разделе 3 Рекомендации МСЭ-Т Т.801 | ИСО/МЭК 15444-2:2004.

3.2 Определения протокола HTTP

Для согласования с протоколом HTTP/1.1 предназначены следующие определения. В случае какого-либо различия должны использоваться эти определения.

3.2.1 Соединение: Виртуальная цепь транспортного уровня, установленная между двумя программами в целях передачи информации.

3.2.2 Объект: Информация, переносимая как полезная нагрузка запроса или отклика. Объект состоит из метаинформации в форме полей заголовков объектов и содержимого (контента) в форме тела объекта.

3.2.3 Посредник: Промежуточная программа, которая действует и как сервер, и как клиент в целях осуществления запросов по поручению других клиентов. Запросы обслуживаются внутренним образом или путем пересылки их далее, с возможной трансляцией, к другим серверам.

3.3 Определения JPIP

В рамках этой Рекомендации | Международного стандарта используются следующие определения. В некоторых случаях эти определения отличаются от тех определений, которые используются в других стандартах и/или Рекомендациях.

3.3.1 кэш-память (сторона клиента): Кэш-память на стороне Клиента является запоминающим устройством бункеров данных JPIP. Клиент может иметь ограниченную кэш-память и может иметь возможность время от времени очищать помещенные в кэш-память бункеры данных JPIP.

3.3.2 снабженный кэш-памятью: Отклик снабжен кэш-памятью, если кэш-памяти разрешено сохранять копию сообщения отклика для использования в ответе на последующие запросы. Даже если источник снабжен кэш-памятью, могут быть дополнительные ограничения относительно того, может ли кэш-память использовать сохраненную копию для конкретного запроса.

3.3.3 модель кэш-памяти (сторона сервера): Оценка сервером порций бункеров данных, имеющихся в кэш-памяти клиента. Сервер может добавлять пункты к своей оценке кэш-памяти клиента, поскольку он предполагает успешную доставку, или поскольку он получил подтверждения о переданных данных, или из-за утверждений об обновлении модели кэш-памяти.

3.3.4 канал: Механизм для группирования запросов и откликов так, что в момент времени внутри группы активным является только один запрос/отклик. Многократные одновременные запросы и отклики требуют многократных каналов.

3.3.5 клиент: Программа, которая устанавливает соединения в целях отправки запросов.

3.3.6 область изображения кодового потока: Область изображения кодового потока является пересечением между изображением и областью, определенной Смещением и Размером области. Область изображения кодового потока может быть пустой (нет площади).

3.3.7 бункер данных: Набор байтов одного и того же типа данных, которые могут доставляться частично.

3.3.8 возрастающий кодовый поток: Представление кодового потока в качестве совокупности бункеров данных (бункеров данных главного заголовка, заголовка элемента мозаичного изображения, зоны или элементов мозаичного изображения), имеющих тот же самый идентификатор кодового потока.

3.3.9 таблица индексов JPIP: Блок файлового формата, который предоставляет информацию о местонахождении порций файла или кодового потока.

3.3.10 логическая цель: Характерное представление характерного первоначального именованного ресурса, или диапазон байтов из такого характерного первоначального именованного ресурса, к которому направляется запрос JPIP. Это характерное представление могло быть преобразовано по коду из первоначального именованного ресурса.

3.3.11 сообщение: Набор байтов из отдельного бункера данных и заголовок, обозначающий те байты и бункер данных.

3.3.12 необработанный кодовый поток: Представление кодового потока как отдельного бункера метаданных.

3.3.13 запрос: Группа полей и значений, посылаемых от клиента к серверу для получения порций изображения или метаданных.

3.3.14 ресурс: Объект или услуга сетевых данных, которые могут быть определены с помощью унифицированного идентификатора ресурса (URI). Цель HTTP.

3.3.15 отклик: Байты, посылаемые из сервера к клиенту после получения запроса.

3.3.16 сервер: Прикладная программа, которая признает соединения, чтобы обслуживать запросы путем отсылки откликов обратно. Любая заданная программа может быть способна быть как клиентом, так и сервером; использование этих терминов относится только к роли, выполняемой с помощью программы для конкретного соединения, а не для возможностей программ вообще.

3.3.17 сеанс: Совокупность запросов и откликов, применяющихся к тому же самому источнику, для которого сервер поддерживает модель кэш-памяти.

3.3.18 на основе сеанса: Там, где сервер поддерживает модель кэш-памяти.

3.3.19 без изменения состояния в процессе исполнения: Отдельный запрос, где сервер не использует модель кэш-памяти в определении отклика.

3.3.20 цель: Логическая идентификация данных JPIP. Имя главной цели (часто имя файла на сервере).

ПРИМЕЧАНИЕ. – Файлы или кодовые потоки JPEG 2000 могут иметься в многократных представлениях (например, тип возврата, размер зоны) или изменяться другим образом, каждый обозначаемый как единственная логическая цель.

3.3.21 заголовок элемента мозаичного изображения: Все заголовки частей элементов мозаичного изображения для характерного элемента мозаичного изображения.

3.3.22 окно обзора: Порция данных изображения, которую желает клиент, как выражается сочетанием следующих полей, которые появляются в запросе: Размер области, Смещение, Размер кадра, Кодовый поток, Контекст кодового потока, Скорость дискретизации, ROI и Слои. Окно обзора часто бывает меньше, чем данные целого изображения. Если окно обзора подразумевается, но не определяется, то тогда оно должно быть взято как окно обзора на данных полной совокупности изображений логической цели.

3.4 Символы

Для целей этой Рекомендации | Международного стандарта применяются следующие символы. К этой Рекомендации | Международному стандарту также применяются символы, определенные в разделе 4 Рекомендации МСЭ-Т Т.800 | ИСО/МЭК 15444-1:2004 и в разделе 4 Рекомендации МСЭ-Т Т.801 | ИСО/МЭК 15444-2:2004.

c	Индекс (начинающийся от 0) компонента изображения, к которому принадлежит зона
fx	Размер кадра x-оси для окна обзора запроса клиента
fy	Размер кадра y-оси для окна обзора запроса клиента
fx'	Размер кадра x-оси для подходящей разрешающей способности кодового потока
fy'	Размер кадра y-оси для подходящей разрешающей способности кодового потока
$fx"$	Модифицированный размер кадра x-оси jpx для подходящей разрешающей способности
$fy"$	Модифицированный размер кадра y-оси jpx для подходящей разрешающей способности
H_{cod}	Высота кодового потока, как записано в блоке Заголовка изображения (ihdr) (см. Приложение I.5.3.1 Рекомендации МСЭ-Т Т.800 | ИСО/МЭК 15444-1:2004)

$\mathbf{H_{comp}}$	Высота усредненного результата, подаваемого в блоке вариантов состава JPX (см. Приложение M.11.10.1 Рекомендации МСЭ-Т Т.801 | ИСО/МЭК 15444-2:2004)
$\mathbf{H_{reg}}$	Высота слоя наложения изображений, как он появляется на сетке регистрации слоя наложения изображений
$\mathbf{Hs_{inst}}$	Обрезаемая высота
$\mathbf{Ht_{inst}}$	Усредненная высота
$\boldsymbol{l}$	Уникальный идентификатор зоны внутри ее кодового потока
$\boldsymbol{N_L}$	Является количеством уровней разложения
num_components	Количество кодируемых компонентов
num_tiles	Количество элементов мозаичного изображения в кодовом потоке
ox	Смещение x-оси для окна обзора в запросе клиента
ox'	Смещение x-оси для подходящей области кодового потока
ox"	Модифицированное смещение x-оси jpx для подходящей области
oy	Смещение y-оси для окна обзора в запросе клиента
oy'	Смещение y-оси для подходящей области кодового потока
oy"	Модифицированное смещение y-оси jpx для подходящей области
$\boldsymbol{r}$	Уровень разрешающей способности
$\boldsymbol{s}$	Порядковый номер, который определяет зону внутри его компонента элемента мозаичного изображения
sx	Размер x-оси окна обзора в запросе клиента
sx'	Размер x-оси для соответствующей области кодового потока
sx"	Модифицированный размер x-оси jpx для подходящей области
sy	Размер y-оси окна обзора в запросе клиента
sy'	Размер y-оси для соответствующей области кодового потока
sy"	Модифицированный размер y-оси jpx для подходящей области
$\boldsymbol{t}$	Индекс (начинающийся от 0) элемента мозаичного изображения, к которому принадлежит зона
$\mathbf{W_{cod}}$	Ширина кодового потока, как записано в блоке Заголовка изображения (ihdr) (см. Приложение I.5.3.1 Рекомендации МСЭ-Т Т.800 | ИСО/МЭК 15444-1:2004)
$\mathbf{W_{comp}}$	Ширина усредненного результата, подаваемого в блок вариантов состава JPX (см. Приложение M.11.10.1 Рекомендации МСЭ-Т Т.801 | ИСО/МЭК 15444-2:2004)
$\mathbf{W_{reg}}$	Ширина слоя наложения изображений, как он появляется на сетке регистрации слоя наложения изображений
$\mathbf{Ws_{inst}}$	Усеченная ширина
$\mathbf{Wt_{inst}}$	Усредненная ширина
$\mathbf{XC_{inst}}$	Смещение усечения x-оси, подаваемое через соответствующую инструкцию (см. Приложение M.11.10.2.1 Рекомендации МСЭ-Т Т.801 | ИСО/МЭК 15444-2:2004)
$\mathbf{XO_{inst}}$	Усредненное смещение x-оси, описываемое через соответствующую инструкцию наложения изображений (см. Приложение M.11.10.2.1 Рекомендации МСЭ-Т Т.801 | ИСО/МЭК 15444-2:2004)
$\mathbf{XO_{reg}}$	Смещение регистрации кодового потока x-оси
XOsiz	Горизонтальное смещение от начала эталонной сетки соответствующего сегмента маркера SIZ кодового потока
$\mathbf{XR_{reg}}$	Коэффициент дискретизации при регистрации кодового потока x-оси, описываемый в начале любого блока регистрации кодового потока (см. Приложение M.11.7.7 Рекомендация МСЭ-Т Т.801 | ИСО/ИЭК 15444-2:2004)
Xsiz	Ширина эталонной решетки соответствующего сегмента маркера SIZ кодового потока

XS_{reg} Точность регистрации x-оси, описываемая в начале любого блока регистрации кодового потока (см. Приложение M.11.7.7 Рекомендации МСЭ-Т Т.801 | ИСО/МЭК 15444-2:2004)

YC_{inst} Смещение усечения y-оси, подаваемое через соответствующую инструкцию (см. Приложение M.11.10.2.1 Рекомендации МСЭ-Т Т.801 | ИСО/МЭК 15444-2:2004)

YO_{inst} Усредненное смещение y-оси, описываемое через соответствующую инструкцию наложения изображений (см. Приложение M.11.10.2.1 Рекомендации МСЭ-Т Т.801 | ИСО/МЭК 15444-2:2004)

YO_{reg} Смещение регистрации кодового потока y-оси

YOsiz Вертикальное смещение от начала эталонной сетки соответствующего сегмента маркера SIZ кодового потока

YR_{reg} Коэффициент дискретизации при регистрации кодового потока y-оси, описываемый в начале любого блока регистрации кодового потока (см. Приложение M.11.7.7 Рекомендация МСЭ-Т Т.801 | ИСО/ИЭК 15444-2:2004)

Ysiz Высота эталонной сетки соответствующего сегмента маркера SIZ кодового потока

YS_{reg} Точность регистрации y-оси, описываемая в начале любого блока регистрации кодового потока (см. Приложение M.11.7.7 Рекомендации МСЭ-Т Т.801 | ИСО/МЭК 15444-2:2004).

4 Сокращения

Для целей этой Рекомендации | Международного стандарта применяются следующие сокращения.

ABNF	Дополненная форма Backus-Naur
DICOM	Цифровые изображения и передача информации в медицине
DWT	Дискретное преобразование элементарных волн
EOR	Конец отклика
HTML	Язык разметки гипертекста
IP	Межсетевой протокол (протокол Интернет)
JP3D	JPEG 2000 Часть 10: 3-D и данные плавающей запятой
JPIP	Диалоговый протокол JPEG 2000
JPP	Зона JPIP
JPSEC	Часть 8 JPEG 2000: Безопасная JPEG 2000
JPT	Часть элемента мозаичного изображения JPIP
JPWL	Часть 11 JPEG 2000: Радиосвязь
JTC 1	Объединенный технический комитет 1
MTF	Передаточная функция модуляции
PDF	Формат переносимого документа
SC 29	Подкомитет 29
SVG	Масштабируемая векторная графика
TCP	Протокол управления передачей
UDP	Датаграммный протокол пользователя
UUID	Универсальный уникальный идентификатор
VBAS	Выровненный сегмент байта переменной длины
WG 1	Рабочая группа 1
XHTML	Наращиваемый язык разметки гипертекста
XML	Наращиваемый язык разметки

5 Соглашения об условных обозначениях

5.1 Правила ABNF

Эта Рекомендация | Международный стандарт использует обозначение ABNF, определенное в документе RFC 2234, включая опорные синтаксические правила ABNF: ALPHA (буквы), CR (возврат каретки), CRLF (стандартная новая строка Интернет), CTL (управляющие символы), DIGIT (десятичные цифры), HEXDIG (шестнадцатеричные цифры), LF (перевод строки), LWSP (линейный белый пробел) и SP (пробел). Для целей этой Рекомендации | Международного стандарта также применяются следующие правила.

```
NZDIGIT = %x31-39              ; 1-9
UPPER = %x41-5A                ; A-Z
LOWER = %x61-7A                ; a-z
UINT    = 1*DIGIT
NONZERO = *"0" NZDIGIT *DIGIT
UINT-RANGE = UINT ["-" [UINT]]
UFLOAT = 1*DIGIT ["." 1*DIGIT]
ENCODED-CHAR = "%" HEXDIG HEXDIG
UUID = 16(HEXDIG)
TOKEN = 1*(ALPHA / DIGIT / "." / "_")
```

Эта Рекомендация | Международный стандарт также определяет `PATH` [*тракт*], представляющий файл или имя тракта. В общем случае, значения `PATH` могут содержать любой знак, хотя для заданной архитектуры сервера, сервер должен отвергать любые знаки, которые не являются законными на таком конкретном сервере. Кроме того, `PATH` должен быть соответствующим образом кодирован, как определено транспортной технологией.

`UINT-RANGE` указывает диапазон значений целых чисел. Первое целое число в диапазоне определяет начало диапазона. Если определяются два значения, то первое и второе значения задают содержащиеся пределы начала и окончания к диапазону. Если определяются только первое значение и знак "`-`", то диапазон включает в себя все значения, которые больше или равны первому значению.

Численное значение, непосредственно предшествующее элементу ABNF, относится к повторению параметра, который сопровождает число, для числа раз, задаваемого численным значением, без промежуточных пробелов между каждым появлением.

Конструктивный элемент "`1#`" относится к одному или более повторениям параметра, который следует, каждое появление которого отделяется запятой.

Конструктивный элемент "`1$`" относится к одному или более повторениям параметра, который следует, каждое появление которого разделяется точкой с запятой.

5.2 Правила ABNF файлового формата

```
compatibility-code = 4(ALPHA / DIGIT / "_" / ENCODED-CHAR)
box-type = 4(ALPHA / DIGIT / "_" / ENCODED-CHAR)
box-type-list = "*" / 1#(box-type)
```

Элемент `box-type` [*блок-тип*] определяет четыре знака типа блока. Для каждого знака в типе блока, если знак является буквенно-цифровым (A..Z, a..z или 0..9), знак записывается прямо в строку. Если знак является пробелом (0x20), то такой знак должен кодироваться как знак подчеркивания ("_"). Для любого другого знака на его месте записывается строка из 3-х знаков, состоящая из знака процентов ("%"), за которым следуют две шестнадцатеричные цифры, представляющие значение знака из типа блока в шестнадцатеричном исчислении. Элемент `compatibility-code` [*совместимость-код*] кодируется тем же самым способом, которым кодируется элемент `box-type`.

Элемент `box-type-list` [*блок-тип-перечень*] указывает перечень типов блоков. Если значение поля `box-type-list` есть "*", то поле относится ко всем типам блоков.

5.3 Ключ к графическому описанию блоков (информативное)

Описание каждого блока сопровождается рисунком, который показывает порядок и взаимоотношение параметров в блоке. Рисунок 1 показывает пример этого типа рисунка. Прямоугольник используется для указания параметров в блоке. Ширина прямоугольника пропорциональна количеству байтов в параметре. Заштрихованный прямоугольник (диагональные полосы) указывает, что параметр имеет изменяющийся размер. Два параметра с верхними надстрочными индексами и серая площадь между ними указывают выполнение нескольких из этих параметров. Последовательность из двух групп многозначных параметров с верхними надстрочными индексами, разделенная серой площадью, указывает выполнение такой группы параметров

(один набор каждого параметра в группе, сопровождаемый следующим набором каждого параметра в группе). Дополнительные параметры или блоки будут показаны с помощью пунктирного прямоугольника.

Рисунок сопровождается перечнем, который описывает значение каждого параметра в блоке. Если параметры повторяются, то определяется длина и сущность выполнения параметров. Как пример, на рисунке 1, параметры A, B, C и D есть, соответственно, 8, 16, 32 битов и изменяющаяся длина. Обозначения E^0 и E^{N-1} подразумевают, что в ряду имеются N различных параметров, E^i. Группа параметров F^0 и F^{M-1}, а также G^0 и G^{M-1} указывают, что блок будет содержать F^0, за которым следует G^0, сопровождаемое F^1 и G^1, продолжаясь до F^{M-1} и G^{M-1} (в сумме M экземпляров каждого параметра). Кроме того, поле D является дополнительным и может отсутствовать в этом блоке.

Дополнительно на рисунке, описывающем содержимое суперблока, будет использоваться многоточие (…) для указания того, что содержимое файла между двумя блоками конкретно не определяется. Вместо многоточия может быть найден любой блок (или последовательность блоков), пока путем определения такого блока не указано другое условие.

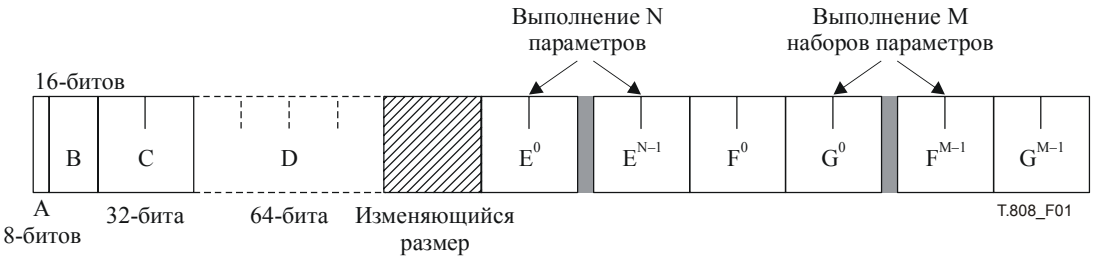

Рисунок 1 − Пример рисунков описаний блоков

Например, суперблок, показанный на рисунке 2, должен содержать блок AA и блок BB, а блок BB должен сопровождать блок AA. Однако между блоками AA и BB могут быть обнаружены другие блоки. Действия с неизвестными блоками обсуждается в Приложении I.8 Рекомендации МСЭ-Т T.800 | ИСО/МЭК 15444-1:2004.

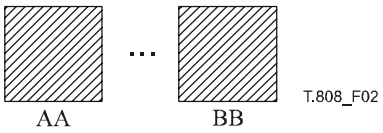

Рисунок 2 − Пример рисунков описаний суперблоков

6 Общее описание

6.1 Протокол JPIP

Эта Рекомендация | Международный стандарт описывает синтаксисы и методы, которые используются тогда, когда клиент осуществляет доступ к совокупности сжатых изображений и к связанным данным совокупности изображений JPEG 2000, находящимся на сервере, который обеспечивает протокол JPIP. Эта Рекомендация | Международный стандарт делает возможным гибкость и функциональные возможности, планируемые в Рекомендации МСЭ-Т T.800 | ИСО/МЭК 15444-1:2004, которые должны быть реализованы через многократные транспортные средства клиент/сервер.

Протокол JPIP определяет диалоговый протокол для выполнения эффективного обмена совокупностями изображений и связанными данными совокупностей изображений JPEG 2000. Протокол определяет взаимодействия между клиентом и сервером, основанные на запросе клиента и отклике сервера, как показано на рисунке 3. Эта Рекомендация | Международный стандарт определяет запросы клиентов JPIP и ответы серверов JPIP. Как примеры возможных транспортных средств для протокола JPIP показаны документы HTTP/1.1 (RFC 2616), TCP (RFC 793) и UDP (RFC 768). Клиент использует запрос Окна обзора для определения разрешающей способности, размера, местоположения, компонентов, слоев и других параметров для совокупности изображений и связанных данных совокупности изображений, которые запрашиваются клиентом. Сервер откликается, доставляя совокупность изображений и связанные данные совокупности изображений с помощью потоков на основе зоны, потоков на основе элементов мозаичного изображения или целых изображений. Протокол также позволяет обсуждение возможностей и ограничений клиента и сервера. Клиент может запрашивать информацию об изображении, как определено в таблицах индексов JPIP из сервера, что позволяет клиенту уточнять свой запрос Окна обзора к характерным параметрам изображения (например, запросы диапазона байтов). Модель кэш-памяти сервера основывается на возможностях, определяемых клиентом, и на сохранении данных сеанса.

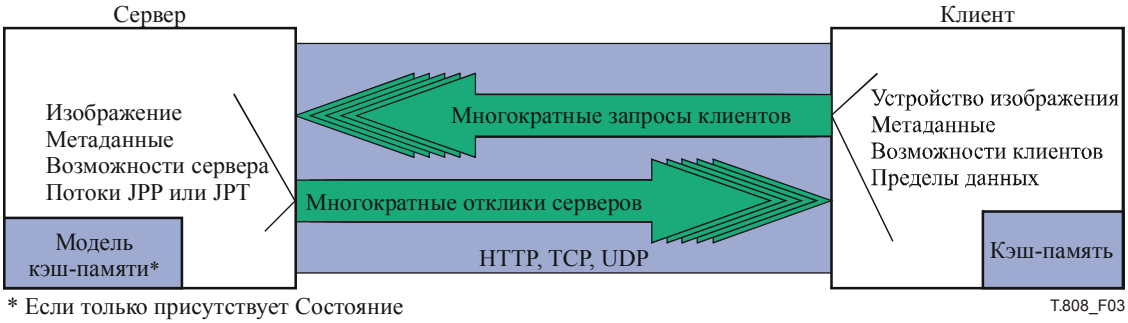

Рисунок 3 − Обзор протокола JPIP

Этот протокол может быть использован поверх нескольких различных транспортных средств, как показано на рисунке 4. Эта Рекомендация | Международный стандарт включает в себя информативные приложения по использованию протокола JPIP поверх протоколов HTTP и TCP и предоставляет советы для других примерных реализаций.

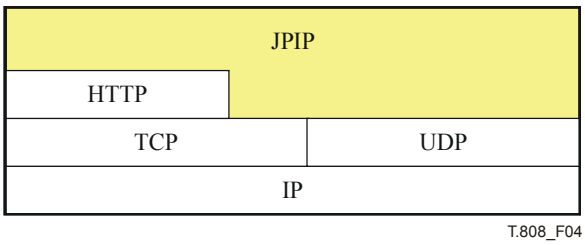

Рисунок 4 − Стек протокола JPIP

Были включены положения для расширения протокола JPIP, чтобы поддерживать текущие стандарты JPEG 2000: Рекомендация МСЭ-Т Т.802 | ИСО/МЭК 15444-3, JPEG 2000 "Движение" и документ ИСО/МЭК 15444-6, Составные документы, а также будущие части JPEG 2000 (в настоящее время JP3D, JPSEC и JPWL).

6.2 Назначение

Эта Рекомендация | Международный стандарт определяет синтаксис и методы, требуемые как для клиента, так и для сервера. Каждое приложение определяет компонент, который требуется для достижения взаимодействия и функциональных возможностей между клиентом и сервером поверх нескольких транспортных средств. Каждое приложение может быть требованием клиента, сервера или их обоих. Эти приложения описываются ниже.

- Приложение A описывает потоки на основе элемента мозаичного изображения и на основе зоны, которые требуются как для клиента, так и для сервера. От сервера требуется производить потоки, соответствующие JPP-потокам и JPT-потокам, и понимать загружаемые в сервер JPP-потоки и JPT-потоки. От клиента требуется понимать и должным образом декодировать эти потоки, а также нести ответственность за производство соответствующих потоков при загрузке частичной совокупности изображений в сервер.
- Приложение B описывает сеанс и моделирование кэш-памяти для сеанса клиент/сервер и требуется и для клиента, и для сервера.
- Приложение C определяет синтаксис запроса клиента. Клиент должен порождать соответствующие запросы, а сервер должен быть способен понимать и откликаться на все соответствующие запросы.
- Приложение D определяет синтаксис отклика сервера. Сервер должен порождать соответствующие отклики, а клиент должен быть способен понимать соответствующие отклики.
- Приложение E определяет синтаксис и методы для загрузки в сервер частичного изображения для систем, которые для загрузки используют протокол JPIP.
- Приложения F, G и H определяют методы и процедуры для взаимодействий "клиент/сервер" JPIP поверх нескольких различных транспортных протоколов.

- Приложение I определяет синтаксис информации индексирования, содержащийся в блоке JPEG 2000, что может использоваться клиентом и сервером для более эффективного получения доступа к совокупности изображений и к связанным данным совокупности изображений.
- Приложение J определяет, как этот стандарт может быть расширен путем регистрации.
- Приложение K описывает несколько примеров использования этой Рекомендации | Международного стандарта для нескольких различных приложений.

7 Соответствие

Соответствие с этой Рекомендацией | Международным стандартом со стороны клиента означает, что запросы JPIP клиента являются правильно структурированными, правомерными и соответствующими запросам клиентов JPIP, как определяется этой Рекомендацией | Международным стандартом. Клиенты должны поддерживать все типы нормативных запросов.

Соответствие с этой Рекомендацией | Международным стандартом со стороны сервера означает, что отклики JPIP сервера являются правильно структурированными, правомерными и соответствующими сигнализации отклика сервера JPIP, как определено этой Рекомендацией | Международным стандартом. Серверы должны поддерживать все типы нормативных запросов.

В то время как подразумевается, что эту Рекомендацию | Международный стандарт следует осуществлять так, что данные для совокупности изображений запрашиваются на основе требований приложений со стороны клиента в результативных запросах JPIP, никакое соответствующее поведение не определяется.

В равной степени, данные для совокупности изображений следует обслуживать на основе результативных откликов сервера JPIP, сводя к минимуму количество обслуживаемых данных вне сообщенных интересов клиента и избыточных данных, которыми уже обладает клиент. Однако никакое соответствующее поведение не определяется.

Ожидается, что приложения сервера могут уменьшать эффективность, посылая дополнительные данные или избыточные данные в зависимости от сетевого качества обслуживания. Такие решения по реализации являются зависимыми от приложения и обеспечивают систему JPIP с высокой полезностью. Однако эта Рекомендация | Международный стандарт не определяет соответствие для проектирования этих решений по реализации.

Приложение A

Типы носителей информации JPP-потока и JPT-потока

(Это приложение образует составную часть этой Рекомендации | Международного стандарта)

A.1 Введение

JPP-поток и JPT-поток являются типами носителей информации, полезными для представления кодовых потоков и данных о файловом формате JPEG 2000 в произвольном порядке. Каждый тип носителя информации состоит из сцепленной последовательности сообщений, где каждое сообщение содержит порцию одиночного бункера данных, которому предшествует заголовок сообщения. Бункеры данных содержат порции представления сжатого изображения JPEG 2000, так что есть возможность составлять поток, который полностью представляет информацию, присутствующую в файле или в кодовом потоке JPEG 2000. Каждое сообщение полностью само себя описывает таким образом, что последовательность сообщений может быть завершена в любой точке, а сообщения могут быть перестроены в зависимости от минимальных ограничений без потери их смысла. По этим причинам типы носителей информации JPP-потока и JPT-потока полезны для серверов JPIP, а протокол JPIP разрабатывается конкретно с учетом этих типов носителей информации. Это приложение определяет типы носителей информации JPP-потока и JPT-потока без ссылки на протокол JPIP.

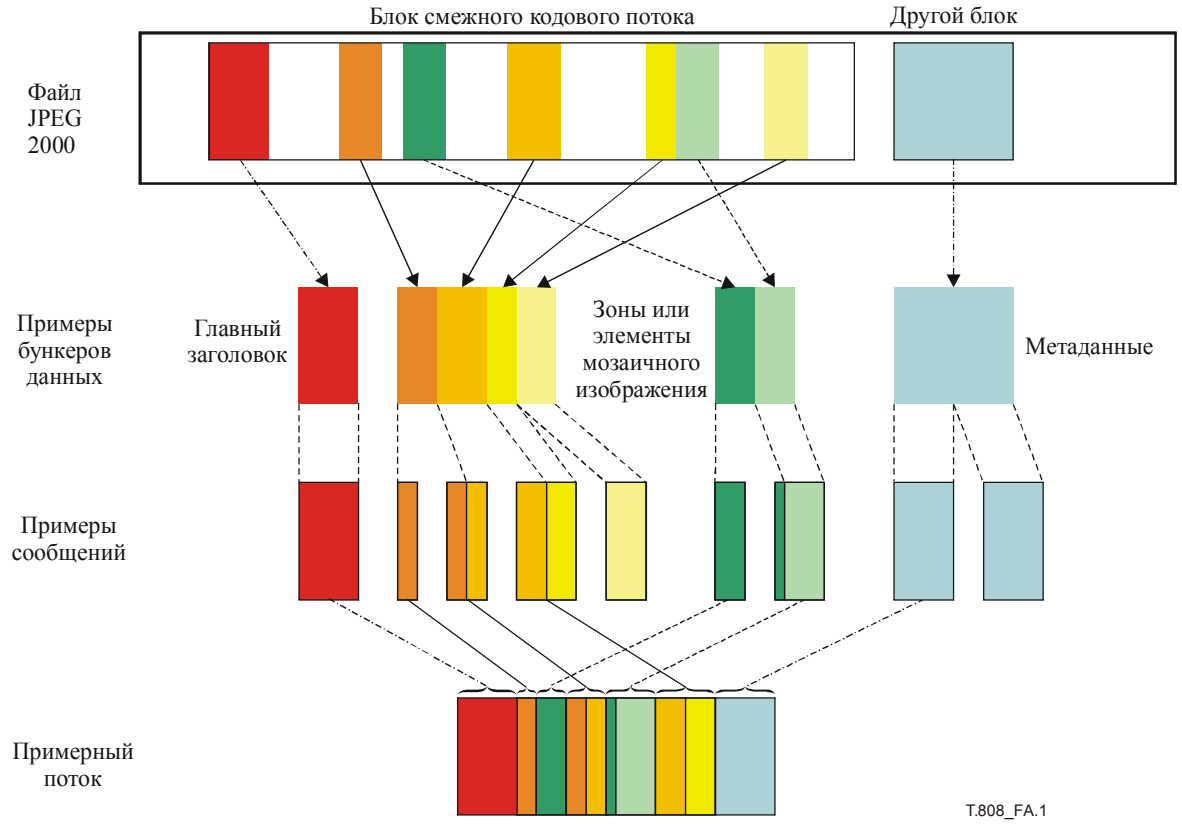

Рисунок A.1 − Примеры взаимоотношений файла JPEG 2000, бункеров данных JPIP и JPIP-потока (согласно G.J. Colyer и R.A. Clark, IEEE Trans. Consumer Electronics, 49 (2003), стр. 850−854)

Рисунок A.1 является иллюстративным примером взаимоотношения между потоками битов из файла JPEG 2000, бункерами данных JPIP и потоком JPIP. Рисунок показывает главный заголовок, кодированный красным цветом, 2 зоны с пакетами, кодированными тенями оранжево-желтого и зеленого цвета, и блок метаданных, кодированный голубым цветом. Сообщения JPIP с самоописанием формируются из этих бункеров данных и объединяются для образования потока JPIP.

Поток JPIP состоит из одного или более сцепленных сообщений JPIP. Каждое сообщение JPIP состоит из заголовка и тела. Заголовок предоставляет описательную информацию для определения соответствующего бункера данных. Тело представляет собой данные из бункера данных. Если не предусматривается дальнейшая сигнализация, то сообщение является сцепкой заголовка с телом.

ПРИМЕЧАНИЕ. – В этой Рекомендации | Международном стандарте все предоставленные примеры образуют двоичные сообщения путем сцепки заголовка и тела. Если для заголовка и тела обеспечивается другая сигнализация, то она зависит от осуществления транспортных средств и приложения. Например, для приложений на основе радиосвязи может быть реализована вспомогательная сигнализация с изменяющейся защитой от ошибок.

A.2 Структура заголовка сообщения

A.2.1 Общие положения

Каждое сообщение представляет порцию в точности одного бункера данных. Заголовок сообщения состоит из последовательности выровненных по байтам сегментов изменяющейся длины (VBAS) [*variable-length byte-aligned segments*]. Каждый сегмент VBAS состоит из последовательности байтов, все из которых, кроме последнего, имеют самый старший бит (бит 7), равный 1, как видно из рисунка A.2. Наименьшие значащие 7 битов каждого байта в сегменте VBAS сцеплены для формирования потока битов, который используется различными способами для различных сегментов VBAS.

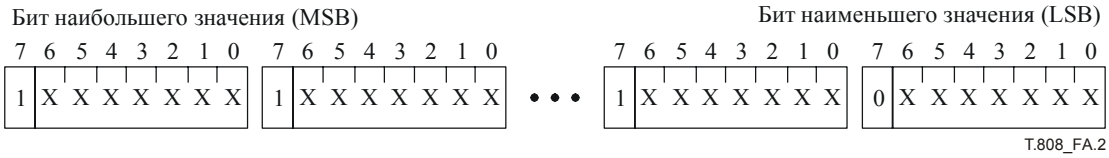

Рисунок A.2 – Структура VBAS

Заголовок сообщения служит для обозначения конкретного бункера данных и диапазона байтов, что представляется телом сообщения. Заголовки сообщений могут принимать независимую форму и зависимую форму. Независимая форма является удлиненной формой, где заголовки сообщений являются полностью описываемыми самостоятельно; их истолкование не зависит от любых других заголовков сообщений. Дополнительные более короткие заголовки сообщений, зависящие от формы, используют информацию в заголовках предыдущих сообщений; их декодирование зависит от предыдущего сообщения. Приложения могут выбирать использование заголовков сообщений удлиненной формы; эти сообщения могут быть повторно расставлены в любом произвольном порядке. В качестве альтернативы, приложения могут использовать заголовки сообщений более короткой формы, которые действительно зависят от предыдущих заголовков сообщений; они являются более короткими сообщениями, но будут создавать ошибочные результаты, если сообщения не расставлены в правильной последовательности при декодировании. От приложения зависит, действительно ли упорядочение последовательности полученных сообщений может предполагаться надежным и, если это так, использовать ли заголовки сообщений более короткой формы.

Заголовок сообщения состоит из следующих сегментов VBAS (дополнительные сегменты VBAS, определяемые путем использования квадратных скобок):

Bin-ID [, Class] [, CSn], Msg-Offset, Msg-Length [, Aux]

Существование сегментов VBAS вида Class [*Класс*] и CSn определяется путем изучения сегмента VBAS Bin-ID [*бункер-идентификатор*]. Существование сегмента Aux VBAS определяется сегментом Class VBAS или предыдущим сегментом Class VBAS, если отсутствует сегмент Class VBAS в заголовке текущего сообщения.

Сегмент Bin-ID VBAS играет несколько ролей. Биты 6 и 5 первого байта сегмента Bin-ID VBAS, маркированные как 'b' на рисунке A.3, указывают, присутствуют ли сегменты VBAS вида Class и CSn в заголовке сообщения. Таблица A.1 определяет значения битов и их смысл.

Бит 4 первого байта сегмента Bin-ID VBAS, маркированный как 'c' на рисунке A.3, показывает, содержит ли это сообщение последний байт в связанном бункере данных, или нет: '0' означает, что он не является последним байтом в бункере данных; '1' указывает, что он является последним байтом в бункере данных. Получение сообщения с этим наборов битов позволяет определять длину полного бункера данных, хотя это не предполагает, что полный JPP-поток или JPT-поток содержит достаточно сообщений для компоновки всех байтов из бункера данных.

Остающиеся 4 бита первого байта и 7 битов нижнего порядка для любых остающихся байтов в сегменте Bin-ID VBAS (маркированном как 'd' на рисунке A.3) образуют "идентификатор внутри класса" [*"in-class identifier"*], который используется для однозначного определения бункера данных внутри его класса способом, описанным в разделе A.2.3.

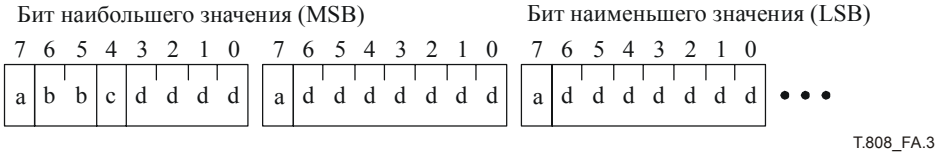

Рисунок A.3 − Структура сегмента Bin-ID VBAS

Таблица A.1 − Дополнительная индикация VBAS идентификатора Bin-ID

Индикатор Биты 'bb'	Смысл
00	Запрещенное.
01	В заголовке сообщения не присутствуют VBAS вида Class или вида CSn.
10	В заголовке сообщения присутствует VBAS вида Class, но отсутствует вида CSn.
11	В заголовке сообщения присутствуют VBAS вида Class и вида CSn.

Сегмент VBAS Class, если присутствует, предоставляет идентификатор класса сообщения. Идентификатор класса сообщения является неотрицательным целым числом с обратным порядком байтов, образованным путем сцепки 7 наименьших значащих битов каждого байта из сегмента VBAS. Если сегмент VBAS Class не присутствует, то идентификатор класса сообщения не изменяется относительно идентификатора, что связан с предыдущим сообщением. Если сегмент VBAS Class отсутствует, и нет предыдущего сообщения, то идентификатор класса сообщения есть 0. Правомерные идентификаторы классов сообщений описываются в разделе A.2.2.

Сегмент VBAS CSn, если присутствует, определяет индекс (начинающийся с 0) кодового потока, к которому принадлежит бункер данных. Индекс кодового потока формируется сцепкой наименьших значащих 7 битов каждого байта в сегменте VBAS с обратным порядком байтов. Если сегмент VBAS CSn отсутствует, то индекс кодового потока остается без изменения по сравнению с предыдущим сообщением. Если сегмент VBAS CSn отсутствует, и нет предыдущего сообщения, то индекс кодового потока есть 0.

Каждый сегмент VBAS вида Msg-Offset [*смещение*] и Msg-Length [*длина*] представляет собой значения неотрицательных целых чисел, образованных путем сцепки наименьших значащих 7 битов каждого байта в сегменте VBAS, с обратным порядком байтов. Целое число Msg-Offset определяет смещение данных в этом сообщении от начала бункера данных. Целое число Msg-Length определяет общее количество байтов в теле сообщения.

Может присутствовать сегмент VBAS Aux. Его присутствие, и смысл, если присутствует, определяется идентификатором класса сообщения, найденным внутри VBAS Bin-ID, как объясняется в разделе A.2.2. Если он присутствует, то VBAS Aux представляет значение неотрицательного целого числа, образованного путем сцепки наименьших значащих 7 битов каждого байта в сегменте VBAS, с обратным порядком байтов.

ПРИМЕЧАНИЕ. − Информация в VBAS Aux не может воздействовать на длину тела сообщения.

A.2.2 Идентификаторы классов сообщений

Идентификаторы классов сообщений, определяемые в этой Рекомендации | Международном стандарте, являются неотрицательными целыми числами, показанными в таблице A.2. Истолкование классов бункеров данных, к которым они относятся, описывается в разделе A.3. Все другие значения идентификатора класса сообщения резервируются, и декодерам, не опознающим значение, следует игнорировать связанные сообщения.

Идентификаторы классов выбираются так, что VBAS Aux присутствует в том случае, если и только если, идентификатор является нечетным. Это свойство позволяет правильно проводить синтаксический анализ для заголовков неопознанных сообщений и игнорировать содержимое.

Сообщения бункеров данных расширенных зон имеют то же самое истолкование, что и сообщения бункеров данных нерасширенных зон, и они относятся точно к тем же самым бункерам данных зон. Сообщения расширенных зон включают в себя сегмент VBAS Aux, который обозначает количество полных пакетов (слоев качества), которые имелись бы для зоны, если бы байты в этом сообщении были объединены со всеми предыдущими байтами той же самой зоны. Если сообщение также содержит последний байт бункера данных, то сегмент VBAS Aux указывает общее число слоев качества, связанных с зоной в первоначальном кодовом потоке. В противном случае, сегмент VBAS Aux указывает слой качества, к которому принадлежит байт, непосредственно сопровождающий последний байт в сообщении. Информация в сегменте VBAS Aux может быть полезной для определенных клиентов.

Таблица A.2 − Идентификаторы классов для различных классов сообщений бункеров данных

Идентификатор класса	Класс сообщения	Класс бункера данных	Тип потока
0	Сообщение бункера данных зоны	Бункер данных зоны	Только JPP-поток
1	Сообщение бункера расширенной зоны	Бункер данных зоны	Только JPP-поток
2	Сообщение бункера данных заголовка элемента мозаичного изображения	Бункер данных заголовка элемента мозаичного изображения	Только JPP-поток
4	Сообщение бункера данных элемента мозаичного изображения	Бункер данных мозаичного изображения	Только JPT-поток
5	Сообщение бункера данных расширенного элемента мозаичного изображения	Бункер данных мозаичного изображения	Только JPT-поток
6	Сообщение бункера данных главного заголовка	Бункер данных главного заголовка	JPP-поток и JPT-поток
8	Сообщение бункера метаданных	Бункер метаданных	JPP-поток и JPT-поток

Сообщения бункеров данных расширенных элементов мозаичного изображения имеют то же самое истолкование, как и сообщения бункеров данных нерасширенных элементов мозаичного изображения, и они относятся в точности к тем же самым бункерам данных элементов мозаичного изображения. Сообщения расширенного элемента мозаичного изображения включают в себя сегмент VBAS Aux, который определяет наименьшее n таким образом, что, во всех компонентах, для которых $(N_L - n)$ является неотрицательным, уровень разрешающей способности $(N_L - n)$ и все уровни нижней разрешающей способности были завершены, когда байты в этом сообщении объединяются со всеми предыдущими байтами того же самого элемента мозаичного изображения, где N_L является количеством уровней разложения, которые могут изменяться компонентом. Если никакие уровни разрешающей способности любого компонента не были завершены, значение сегмента VBAS Aux является единицей плюс максимальное значение N_L по всем компонентам. Значение нуль достигается тогда, когда все разрешающие способности во всех компонентах были завершены. Поскольку разрешающие способности в элементе мозаичного изображения не обязательно появляются в должном порядке, то некоторые уровни разрешающей способности выше значения, сообщенного сегментом VBAS, могли быть завершены, но это не может быть определено из заголовка сообщения. Информация в сегменте VBAS Aux может быть полезна для определенных клиентов.

A.2.3 Идентификаторы внутри класса

Четыре наименее значащих бита первого байта и 7 наименее значащих битов всех других байтов из VBAS Bin-ID образуют сцепку с обратным порядком байтов для формирования отдельного слова, имеющего $7k$-3 битов, где k является количеством байтов в сегменте VBAS. Это слово представляет собой незнаковое целое число, которое служит для однозначного определения бункера данных в пределах его класса и кодового потока. Раздел A.3 предоставляет описание различных классов бункеров данных, вместе с соответствующими идентификаторами внутри класса.

A.3 Бункеры данных

A.3.1 Введение

Бункеры данных содержат порции файла или кодового потока JPEG 2000. Эти данные могут основываться на таких элементах совокупностей изображений, как данные на основе зоны, данные на основе элементов мозаичного изображения и заголовки. Они могут также основываться на метаданных. Безотносительно содержимого бункеров данных, каждый бункер данных обрабатывается как индивидуальный поток битов.

A.3.2 Бункеры данных зон

A.3.2.1 Формат бункера данных зоны

Бункеры данных зон появляются только внутри типа носителя информации JPP-потока. Каждый бункер данных зоны соответствует одной зоне внутри одного кодового потока. Идентификатор в составе класса определяется Уравнением A-1.

$$I = t + (c + s \times \text{num\_components}) \times \text{num\_tiles}\,, \qquad \text{(A-1)}$$

где:

- I есть уникальный идентификатор зоны внутри ее кодового потока;
- t есть индекс (начинающийся от 0) элемента мозаичного изображения, к которому принадлежит зона;
- c есть индекс (начинающийся от 0) компонента изображения, к которому принадлежит зона;
- s есть номер последовательности, который определяет зону внутри ее компонента элемента мозаичного изображения.

Внутри каждого компонента элемента мозаичного изображения зонам назначаются смежные порядковые номера, *s*, следующим образом. Все зоны уровня наименьшей разрешающей способности (который содержит только отсчеты подполосы LL) сначала выстраиваются в последовательность, начиная от 0, следуя порядку растрового сканирования. Зоны из каждого последовательного уровня разрешающей способности выстраиваются в очередь, опять следуя порядку растрового сканирования внутри их уровня разрешающей способности.

Из этого следует, что идентификатор зоны 0 относится к верхней левосторонней зоне из подполосы LL компонента изображения 0 в элементе мозаичного изображения 0.

Каждый бункер данных зоны соответствует строке байтов, сформированной путем сцепки всех пакетов кодового потока, завершенных с помощью всех соответствующих заголовков пакетов, которые принадлежат зоне. Считается, что заголовки пакетов будут упакованы в сегменты маркеров PPM или PPT, которые должны тогда принадлежать к бункерам данных главных заголовков или заголовков элементов мозаичного изображения; в этом случае бункер данных зоны удерживал бы только тела пакетов. В любом случае, потоку данных зоны следует совпадать со смежным сегментом байтов, которые были бы найдены внутри кодового потока JPEG 2000, имея одну из последовательностей прогрессии, подчиненной слою (CPRL, PCRL или RPCL).

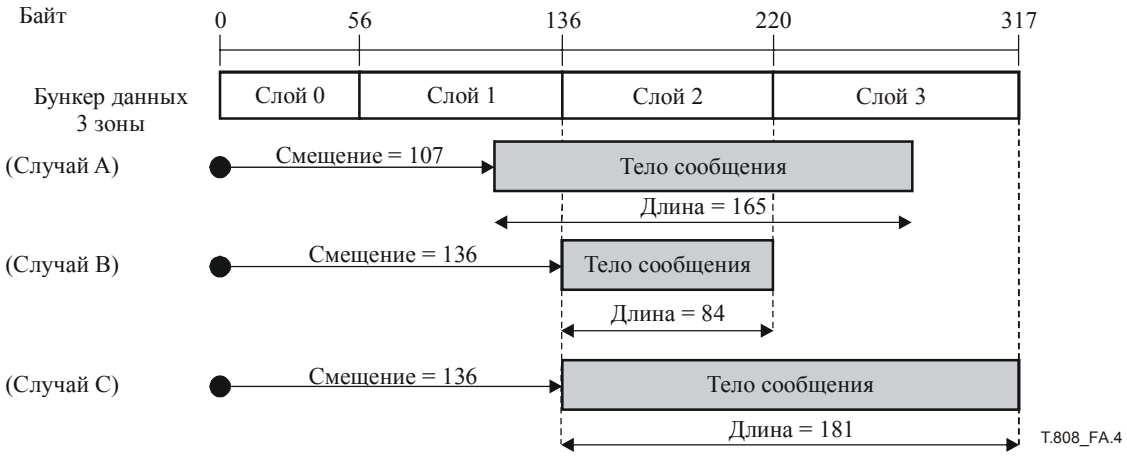

Рисунок A.4 − Примерный бункер данных зоны

A.3.2.2 Пример бункера данных зоны (информативный)

Рисунок A.4 показывает примерный бункер данных зоны (идентификатор 3 внутри класса) с 4 слоями качества (или пакетами).

Для Случаев A, B и C заголовок сообщения показан ниже, основанный на структурах сообщений бункеров данных расширенной и нерасширенной зон. Подчеркнутые данные обозначают сегмент VBAS Aux для определения количества слоев, которые завершаются сообщением.

(Случай A)

Нерасширенный заголовок: `00100011 01101011 10000001 00100101 xxxxxxxx …`

Первоначальный бит 0 указывает, что в VBAS Bin-ID используется только один байт. Следующие два бита ("01") указывают, что отсутствуют сегменты VBAS видов Class или CSn. Следующий бит "0" указывает, что бункер данных не завершается этим сообщением. Остающиеся биты первого байта ("0011") указывают, что идентификатор bin-ID есть 3. Первый бит второго байта указывает, что имеется только один байт, используемый в сегменте VBAS Msg-Offset. Следующие 7 битов ("1101011") означают, что смещение составляет 107. Первый бит 3-го байта указывает, что оба этих байта, и, по меньшей мере, следующий байт являются частью сегмента VBAS Msg-Length. Бит 0, дающий начало 4-му байту, указывает, что он является последним байтом сегмента VBAS Msg-Length. Таким образом, все биты нижнего порядка из 3-го и 4-го байтов сцеплены для определения длины. В этом случае, "0000001 0100101" = 165.

Расширенный заголовок: `01000011 00000001 01101011 10000001 00100101 00000011 xxxxxxxx …`

(Случай B)

Нерасширенный заголовок: `00100011 10000001 00001000 01010100 xxxxxxxx …`

Расширенный заголовок: `01000011 00000001 10000001 00001000 01010100 00000011 xxxxxxxx …`

(Случай C)

Нерасширенный заголовок: `00110011 10000001 00001000 10000001 00110101 xxxxxxxx …`

Расширенный заголовок: `01010011 00000001 10000001 00001000 10000001 00110101 00000100 xxxxxxxx …`

Отметим, что поскольку данные отклика содержат последний байт бункера данных в Случае C, сегмент VBAS Bin-ID указывает, что это есть "завершенное" сообщение.

A.3.3 Бункеры данных заголовков элементов мозаичного изображения

Бункеры данных заголовков элементов мозаичного изображения появляются только внутри типа носителя информации JPP-потока. Для бункеров данных, принадлежащих к этому классу, идентификатор в составе класса удерживает индекс (начинающийся от 0) элемента мозаичного изображения, к которому обращается бункер данных. Этот бункер данных состоит из маркеров и сегментов маркеров для элемента мозаичного изображения *n*. Он не должен содержать сегмент маркера SOT. Включение маркеров SOD является необязательным. Этот бункер данных может быть сформирован из допустимого кодового потока, путем сцепки всех сегментов маркеров, кроме SOT и POC, во всех заголовках частей элемента мозаичного изображения для элемента мозаичного изображения *n*.

A.3.4 Бункеры данных элементов мозаичного изображения

Бункеры данных элементов мозаичного изображения должны использоваться только с типом носителя информации JPT-потока. Для бункеров данных, принадлежащих к этому классу, идентификатор в составе класса является индексом (начинающимся от 0) элемента мозаичного изображения, к которому принадлежит бункер данных. Каждый бункер данных элемента мозаичного изображения соответствует строке байтов, сформированной путем сцепки всех частей элементов мозаичного изображения, принадлежащих к элементу мозаичного изображения, чтобы завершаться с помощью их SOT, SOD и всех других соответствующих сегментов маркеров.

A.3.5 Бункер данных главного заголовка

Типы носителей информации и JPP-потока, и JPT-потока используют бункер данных главного заголовка. Для бункеров данных, принадлежащих к классу главного заголовка кодового потока (завершенной или незавершенной разновидностей), идентификатор в составе класса должен быть равен 0. Этот бункер данных состоит из сцепленного перечня всех маркеров и сегментов маркеров, начиная с маркера SOC. Он не содержит маркеры SOT, SOD или EOC.

A.3.6 Бункеры метаданных

A.3.6.1 Введение в бункеры метаданных

Типы носителей информации и JPP-потока, и JPT-потока используют бункеры метаданных. Бункеры метаданных используются для перемещения метаданных из логической цели, что содержит кодовый поток или кодовые потоки, чьи элементы могут упоминаться другими бункерами данных, связанными с JPP-потоком или JPT-потоком. Для целей этой Рекомендации | Международного стандарта термин "метаданные" относится к любой совокупности "блоков" из файла семейства JPEG 2000. Индекс кодового потока должен игнорироваться в любом сообщении, которое имеет идентификатор класса бункера метаданных.

В отличие от числовых идентификаторов ID, используемых для других типов бункеров данных, идентификаторы ID бункера метаданных не преобразуются алгоритмически в некоторую конструкцию файлового формата или смещения байта. Сервер свободен в выборе любого числового идентификатора ID для любого конкретного бункера метаданных. Одним и единственным исключением этого является то, что бункеру метаданных, содержащему корень логической цели, должен быть дан идентификатор ID, равный 0.

ПРИМЕЧАНИЕ. − Механизм для назначения зависит от осуществления; однако имеется информативное предложение, чтобы серверы назначали идентификаторы ID бункеров, используя последовательные номера.

A.3.6.2 Разделение логической цели, содержащей файл JPEG 2000 в бункерах метаданных

Очевидно, все метаданные могли быть включены в бункер метаданных 0. В этом случае все блоки из логической цели принадлежали бы бункеру метаданных 0, появляясь в своем первоначальном порядке. Поскольку файловые форматы семейства JPEG 2000 состоят только из последовательности блоков, это фактически означает, что бункер метаданных 0 состоял бы из полной логической цели. В целом, однако, полезно разбивать логическую цель на части, которые могут быть переданы управляемым способом. Это позволяет серверам изображений преднамеренно опускать порции логической цели, которые в настоящее время не требуются клиентом. Кроме того, протокол JPIP определяет новый специальный тип блока, известный как "Блок Указателя места заполнения". Блок Указателя места заполнения служит для определения размера и типа блока из логической цели, в то же время указывая на другой бункер данных, который удерживает содержимое блока. Указатели места заполнения также способны представлять кодовые потоки из логической цели. Это особенно существенно ввиду факта, что сжатые данные, представленные любым заданным кодовым потоком, можно доставлять с приращением через другие типы бункеров данных (бункеры данных заголовка и бункеры данных зоны или бункеры данных элемента мозаичного изображения).

Формально, бункер метаданных 0 состоит из всех блоков из логической цели, появляющихся в своем первоначальном порядке, за исключением того, что указатель места заполнения может заменять любой заданный блок. Блок Указателя места заполнения содержит первоначальный заголовок блока, который был заменен, вместе с идентификатором бункера метаданных, который удерживает содержимое такого блока, не включая сам заголовок. Каждый бункер метаданных, отличающийся от бункера метаданных 0, должен состоять из содержимого некоторого блока, чей заголовок появляется в указателе места заполнения, что ссылается на такой бункер данных. Это содержимое блока может само включать в себя субблоки, любой из которых можно заменить дальнейшими указателями места заполнения.

Для примерных иллюстраций бункера метаданных будет использоваться следующая цветная схема (рисунок A.5):

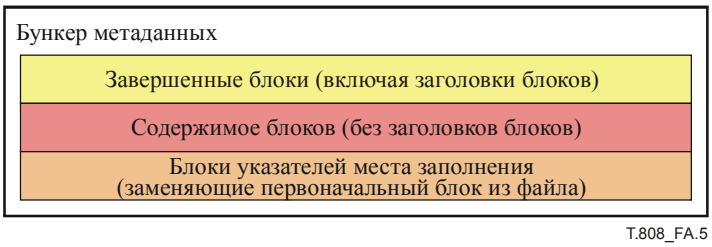

Рисунок A.5 − Примерная цветная схема бункера метаданных

В качестве примера рассмотрим простой файл JP2 со следующей структурой блока (рисунок A.6):

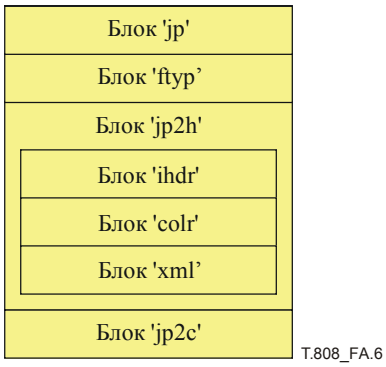

Рисунок A.6 − Файл JP2 отсчета

Этот файл может быть разделен на три бункера метаданных: один должен представлять верхний уровень первоначального файла (бункер данных 0); один должен представлять блок Заголовка JP2; и один должен представлять кодовый поток. Это деление показано на рисунке A.7.

В то время как содержимое любого бункера метаданных должно быть содержимым блока или файла, представленных таким бункером, фактические данные, имеющиеся в таком содержимом, могут концептуально изменяться в зависимости от типа блока. Например, в Бункере метаданных 1 на рисунке 7, представляющем содержимое блока Заголовка JP2, содержимое такого блока буквально является последовательностью других завершенных блоков, поскольку блок Заголовка JP2 является суперблоком. Никакие другие данные, кроме последовательности таких завершенных блоков, не могут быть найдены внутри бункера метаданных 1, поскольку нет других данных в блоке Заголовка JP2. Напротив, данные внутри бункера метаданных 2 являются необработанным содержимым блока Смежного кодового потока, без заголовков блока, поскольку такой блок не является суперблоком.

Одной точкой особого интереса, которую нужно отметить из примера на рисунке 7, является то, что доступ к данным кодового потока может быть обеспечен двумя способами. Бункер второго указателя места заполнения используется для замены блока смежного кодового потока (jp2c) в первоначальном файле. Он обозначает бункер метаданных 2 как удерживающий первоначальное содержимое этого блока, т. е. сам необработанный кодовый поток. Для удобства описания в этой Рекомендации | Международном стандарте это будет называться представлением "необработанного кодового потока". Необработанные кодовые потоки обслуживаются из бункеров метаданных.

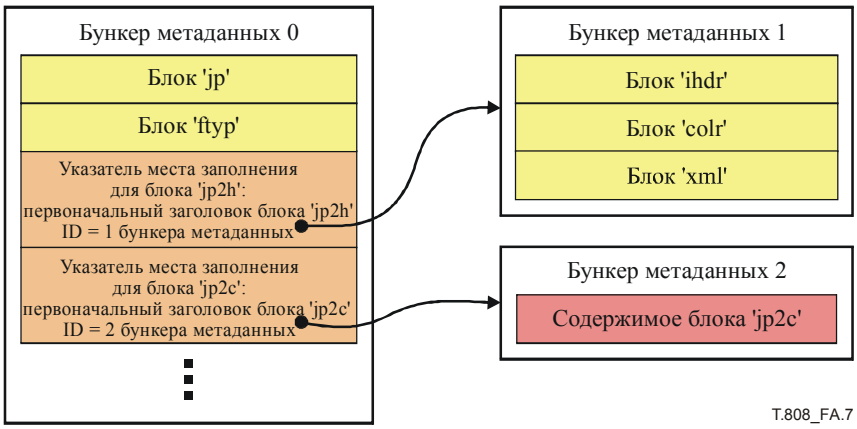

Рисунок A.7 – Файл JP2 отсчета, разделенный на три бункера метаданных

Указатель места заполнения может также предоставлять идентификатор кодового потока. Любые бункеры данных, принадлежащие к классам бункеров главного заголовка, заголовка элемента мозаичного изображения, зоны или элемента мозаичного изображения, имея этот же самый идентификатор кодового потока, перемещают сжатые данные, связанные с тем же самым кодовым потоком, как они найдены в бункере метаданных 2. Для удобства описания в этой Рекомендации | Международном стандарте это должно называться представлением "возрастающего кодового потока". Возрастающие кодовые потоки обслуживаются из этих бункеров данных.

В общем случае, указатели места заполнения, которые ссылаются на данные кодового потока, могут делать так, либо ссылаясь на отдельный бункер метаданных (необработанный кодовый поток), либо обеспечивая идентификатор кодового потока (возрастающий кодовый поток), либо на оба. Даже если обеспечиваются оба метода, данные JPP-потока или данные JPT-потока, имеющиеся у клиента или агента визуализации изображения, могли бы иметь только содержимое (контент) необработанного кодового потока, или иметь только данные от возрастающего кодового потока. Более того, если имеются и необработанные, и возрастающие версии того же самого кодового потока, нет никакой гарантии, что эти два представления будут иметь совместимые параметры кодирования. Только восстановленные отсчеты изображения, связанные с этими двумя представлениями, как гарантируют, будут непротиворечивыми.

Есть также возможность использовать блоки Указателей места заполнения, чтобы связывать множественные кодовые потоки с одиночным первоначальным блоком. Истолкование такой ассоциации зависит от замещаемого блока. Дальнейшая дискуссия по этому аспекту имеется в подразделе A.3.6.4.

В простом примере рисунка 7 блоки Указателя места заполнения появляются только на верхнем уровне файла, в бункере метаданных 0. Как уже было отмечено, однако, указатели места заполнения могут использоваться для замены любого блока, в любом бункере метаданных. Это позволяет расчленять сложные файлы иерархическим способом. По существу, отдельный первоначальный файл может быть пакетирован в разнообразие различных структур бункеров метаданных, в зависимости от того, как используются указатели места заполнения. **Однако отдельный JPP-поток или JPT-поток должны выбирать только одно такое пакетирование.** В приложениях "клиент-сервер", сервер в общем случае будет определять подходящую структуру бункера метаданных для файла, назначая уникальный идентификатор к результирующему потоку и используя ту же самую структуру бункера метаданных во всей связи со всеми клиентами, которые ссылаются на тот же самый идентичный уникальный идентификатор.

Когда указатель места заполнения перемещает блок в новый бункер метаданных, заголовок такого блока (поля LBox, TBox и XLBox) сохраняется, неизменным, в блоке Указателя места заполнения. Если клиент или агент визуализации нуждаются в преобразовании конкретных блоков в их смещения первоначальных файлов, они могут так сделать, используя заголовки первоначальных блоков, которые появляются в блоках Указателя места заполнения. Эта информация, в конечном счете, позволяет любому местоположению в первоначальном файле преобразовываться в конкретное местоположение в конкретном бункере метаданных, если содержимое такого бункера данных существует. Это важно, поскольку некоторые семейства файлов JPEG 2000 содержат блоки, которые ссылаются на другие блоки через их местоположение внутри файла.

В то время как существует значительная свобода в решении, как лучше всего делить файл в бункерах метаданных, есть одно ограничение. Любой блок Указателя места заполнения, который появляется внутри бункера метаданных, должен заменять блок верхнего уровня внутри такого бункера данных. Эквивалентным образом, везде, где субблок должен быть заменен указателем места заполнения, его непосредственный содержащий суперблок должен находиться внутри своего собственного бункера метаданных. Например, в

файле отсчета, показанном на рисунке 6, данные XML, содержащиеся внутри блока Заголовка JP2, могут быть помещены в отдельный бункер данных из других блоков. Это позволяет серверу доставлять только те бункеры данных, которые фактически требуются для декодирования и отображения изображений, если данные XML явно не требуются. Подходящая структура бункера данных показана на рисунке 8.

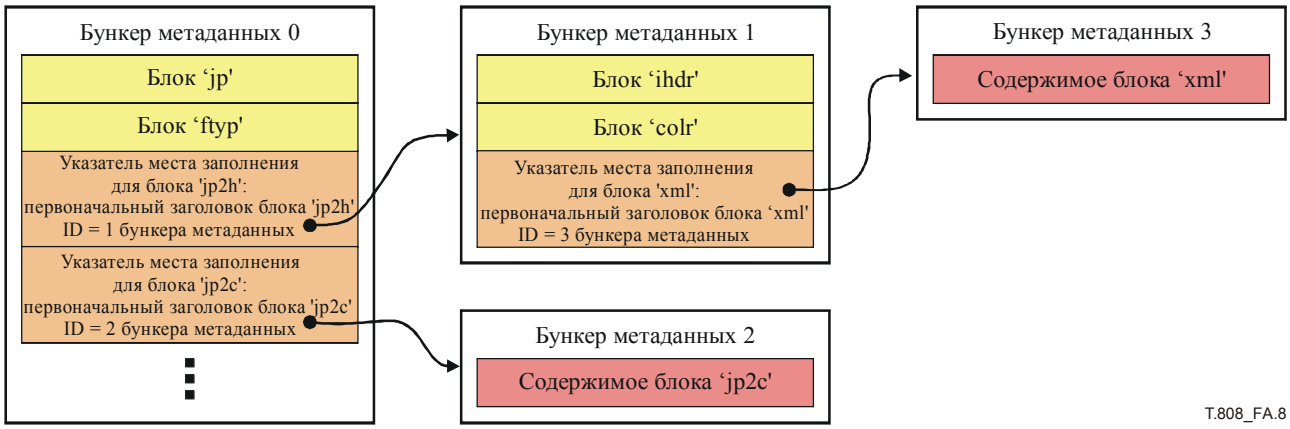

Рисунок A.8 − Суперблок с бункером справочных метаданных

Не было бы допустимым, однако, для блока Заголовка JP2 оставаться в бункере метаданных 0, как показано на рисунке 9.

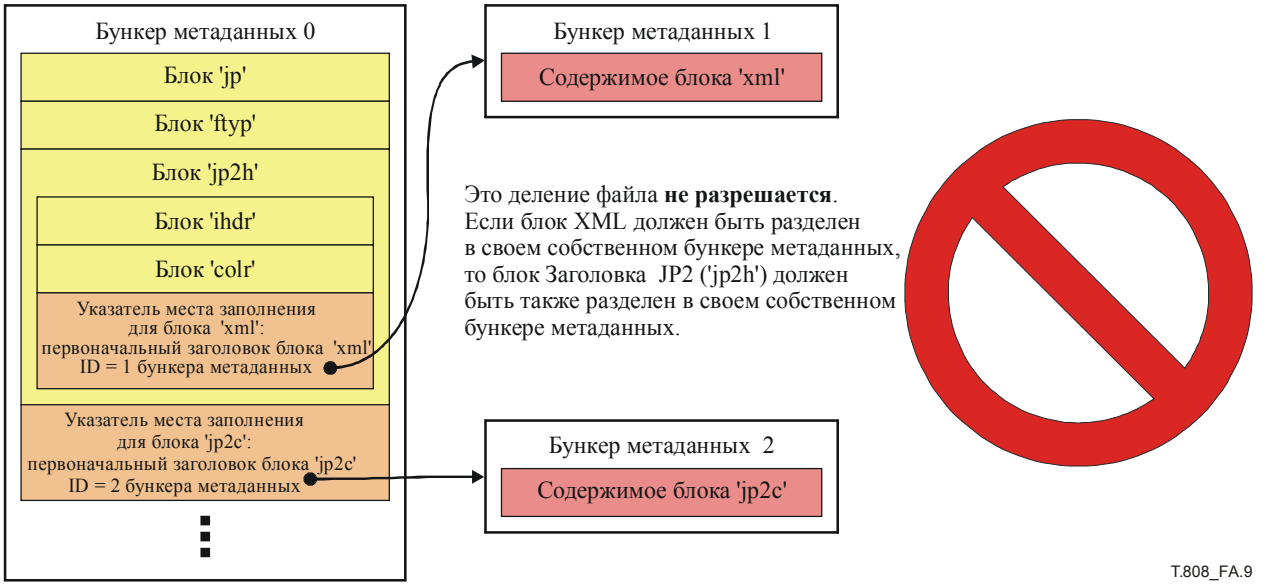

Рисунок A.9 − Недопустимое деление файла в бункерах метаданных

ПРИМЕЧАНИЕ. − Эквивалентным способом выражения этого же самого ограничения является следующее. Везде, где указатель места заполнения заменяет субблок, указатель места заполнения должен также заменять блок, содержащий его. Это ограничение гарантирует, что для клиента или агента визуализации всегда есть возможность восстанавливать длины и местоположения первоначальных блоков внутри файла, даже если некоторые из блоков не поняты клиентом.

В дополнение к обеспечению первоначального содержания блока в отдельном бункере метаданных, также разрешаются JPP-поток и JPT-поток для обеспечения альтернативных представлений блока, которые явно не появлялись внутри первоначального файла. Эти альтернативные представления известны как "эквиваленты потоков". Например, первоначальный файл мог бы содержать блок Перекрестной ссылки, чей блок перечня фрагментов собирает один или более фрагментов файла, чтобы воссоздать блок Спецификации цветового пространства. В то время как клиенту или агенту визуализации необходимо следовать за уместными указателями файла, чтобы восстанавливать блок Спецификации цветового пространства, более удобное представление JPP-потока или JPT-потока могло бы содержать указатель места заполнения, который ссылается на бункер данных, содержащий восстановленный блок Спецификации цветового пространства, как на эквивалент потока. Чтобы сделать это, указатель места заполнения включает заголовок блока для эквивалента потока, вместе с идентификатором бункера метаданных, который удерживает содержимое блока эквивалента потока.

Следующий пример (показанный на рисунке 10) иллюстрирует использование эквивалентов потоков для блоков Перекрестной ссылки. В этом случае бункер данных, который удерживает содержимое эквивалентного потока, также упоминается как удерживающий первоначальное содержимое другого блока. В то время как это, вероятно, будет обычной ситуацией, где первоначальный файл содержал блоки перекрестной ссылки, нет никакой потребности в эквиваленте потока для указания на бункер метаданных, который связан с иерархией первоначального файла. Содержимое блока эквивалента потока может быть создано из рабочей области, или можно обратиться к содержимому, которое первоначально существовало внутри других файлов. Это позволяет блокам Перекрестной ссылки, перечень фрагментов которых ссылается на другие файлы или на указатели URL, полностью пакетироваться внутри отдельного JPP-потока или JPT-потока.

Эквиваленты потоков могут использоваться в любой ситуации, где сервер может создать дополнительную форму содержимого блока, что обеспечивает некоторую выгоду клиенту; они имеются не только для того, чтобы обеспечивать доступ к явным данным перекрестной ссылки.

В дополнение к указанию на фактические или эквивалентные данные блока, блок указателя места заполнения может указывать на один или более кодовых потоков, где заменяемый блок эквивалентен таким кодовым потокам. Например, блок Смежного кодового потока может быть заменен блоком указателя места заполнения, который ссылается на идентификатор ID возрастающего кодового потока, содержащийся внутри такого блока Смежного кодового потока. Другим примером была бы замена блока Смещения фрагмента данных [*Chunk*] в файле JPEG 2000 "Движение" указателем места заполнения, что определяет массив идентификаторов ID кодовых потоков. Такие идентификаторы ID кодовых потоков относятся к кодовым потокам, которые указываются блоком Смещения фрагмента данных.

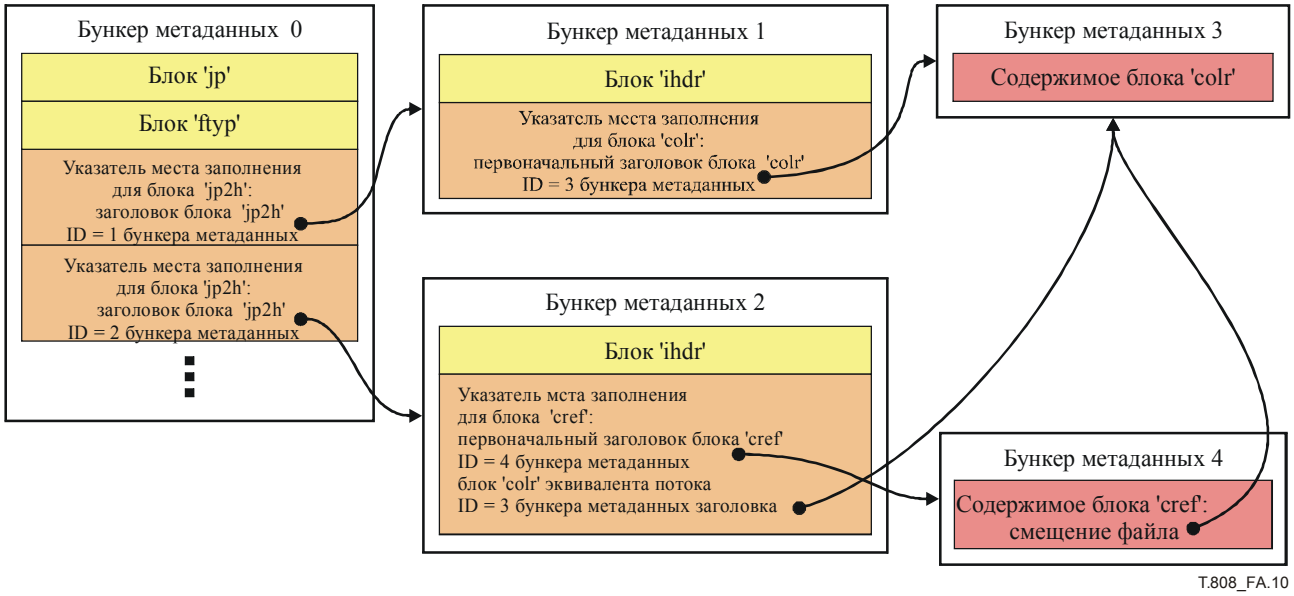

Рисунок A.10 – Пример использования эквивалентов потоков

A.3.6.3 Формат блока Указателя места заполнения

Рисунок A.11 показывает формат блока Указателя места заполнения, включая заголовок блока (в отличие от определения большинства блоков в Приложении I и в других частях этой Рекомендации | Международного стандарта); он определил этот способ подчеркнуть, что использование поля длины в заголовке блока для блока Указателя места заполнения является более ограничительным, чем для других блоков.

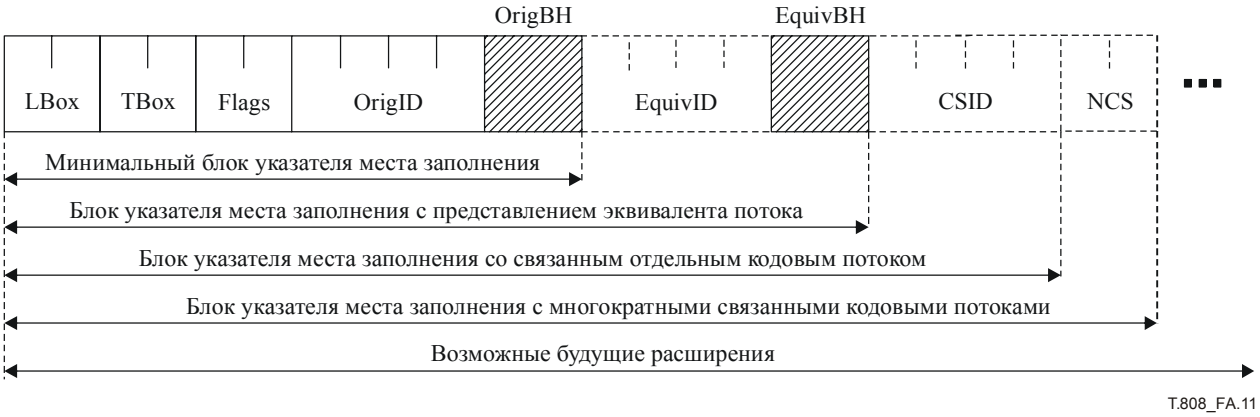

Рисунок A.11 − Структура блока Указателя места заполнения

LBox: Это стандартное 4-байтовое поле длины с обратным порядком байтов для блока. Для блока Указателя места заполнения значение не должно быть 1, означая, что поле XLBox не должно присутствовать.

TBox: Это стандартное поле типа блока из 4 байтов для блока. Значение типа для Указателя места заполнения должно быть 'phld' (0x7068 6c64).

Флаги: Это поле указывает, какие элементы Указателя места заполнения содержат действительные данные. Это поле кодируется как целое 4-байтовое число с обратным порядком байтов. Правомерные значения для поля Флага указываются в таблице A.3.

OrigID: Это поле указывает идентификатор ID бункера метаданных для бункера, вмещающего содержимое первоначального блока, представленного блоком Указателя места заполнения. Оно кодируется как незнаковое целое 8-байтовое число с обратным порядком байтов.

OrigBH: Это поле указывает первоначальный заголовок (LBox, TBox и XLBox, как необходимо) первоначального блока, упоминаемого блоком Указателя места заполнения. Длина этого поля составляет 8 байтов, если поле Lbox заголовка первоначального блока не равно 1, и 16 байтов в противном случае.

EquivID: Это поле указывает идентификатор ID бункера метаданных для бункера, который содержит форму эквивалента потока из этого блока. Это поле кодируется как незнаковое целое 8-байтовое число с обратным порядком байтов.

EquivBH: Это поле указывает заголовок блока эквивалента потока (LBox, TBox и XLBox, как необходимо) из блока, упоминаемого этим блоком Указателя места заполнения. Длина этого поля составляет 8 байтов, если поле Lbox заголовка эквивалентного блока не равно 1, и 16 в противном случае.

CSID: Это поле указывает идентификатор ID первого кодового потока, связанного с заменяемым блоком. Это идентификатор ID, который связывается со всеми бункерами данных заголовков, зон и/или элементов мозаичного изображения, используемых для увеличивающейся передачи содержимого первого кодового потока, связанного с заменяемым блоком. Это поле кодируется как незнаковое целое 8-байтовое число с обратным порядком байтов.

NCS: Это поле указывает количество кодовых потоков в массиве кодовых потоков, который эквивалентен замещаемому блоку. Значения идентификаторов ID этих кодовых потоков идут смежным образом от значения, указанного полем CSID. Это поле кодируется как незнаковое целое 4-байтовое число с обратным порядком байтов.

ExtendedBoxList: Это поле конкретно на рисунке A.11 не показывается. Поле NCS может сопровождаться последовательностью блоков, содержащих расширенную информацию от сервера. Существование любого блока, сопровождающего поле NCS, должно быть указано с помощью бита в поле Флага. Однако ни расширенные блоки, ни любые дополнительные флаги битов не определяются этой Рекомендацией | Международным стандартом. Клиенты должны игнорировать любой блок в ExtendedBoxList, который не понят.

Значение бита "x" в таблице A.3 показывает, что указываемое значение включает в себя случаи, где такой бит устанавливается либо в "1", либо в "0". Биты, указываемые как "y", не используются этим стандартом, и должны быть установлены в 0 серверами и проигнорированы клиентами.

Не все из полей, определенных для блока Указателя места заполнения, должны появляться в каждом блоке Указателя места заполнения. Как указано стрелками в рисунке A.11, если никакой блок идентификатора ID эквивалентного или возрастающего кодового потока не обеспечивается, то блок может быть завершен в конце поля OrigBH. Подобным образом, если никакой идентификатор ID возрастающего кодового потока не обеспечивается, блок может быть завершен в конце поля EquivBH, а если обеспечивается не более чем один идентификатор ID возрастающего кодового потока, то блок может быть завершен в конце CSID.

Таблица A.3 – Правомерные значения для поля Флагов в блоке Указателя места заполнения

Значение	Смысл
yyyy yyyy yyyy yyyy yyyy yyyy yyyy xxx1	Доступ обеспечивается к первоначальному содержимому этого блока через бункер метаданных, определяемый в поле OrigID
yyyy yyyy yyyy yyyy yyyy yyyy yyyy xxx0	Не обеспечивается доступ к первоначальному содержимому этого блока, а значение поля OrigID должно игнорироваться
yyyy yyyy yyyy yyyy yyyy yyyy yyyy xx1x	Обеспечивается блок эквивалента потока, чье содержимое находится в бункере метаданных, определяемом полем EquivID
yyyy yyyy yyyy yyyy yyyy yyyy yyyy xx0x	Не обеспечивается блок эквивалента потока, а значение любых полей EquivID и EquivBH должно игнорироваться
yyyy yyyy yyyy yyyy yyyy yyyy yyyy 01xx	Доступ к изображению, представленному этим блоком, обеспечивается одиночным возрастающим кодовым потоком, который определяется полем CSID. Значение поля NCS должно обрабатываться так, как если бы оно было установлено в "1", вне зависимости от фактического значения этого поля
yyyy yyyy yyyy yyyy yyyy yyyy yyyy 11xx	Доступ к изображению, представленному этим блоком, обеспечивается одним или более возрастающими кодовыми потоками, как указывается полями CSID и NCS.
yyyy yyyy yyyy yyyy yyyy yyyy yyyy x0xx	Этот указатель места заполнения не обеспечивает доступ к изображению, представляющему первоначальный блок как возрастающий кодовый поток, поля CSID и NCS должны игнорироваться
Другие значения	Зарезервированы для использования ИСО

A.3.6.4 Ссылка на возрастающие кодовые потоки с помощью указателей мест заполнения

Везде, где существуют заголовок, зона или бункеры данных элементов мозаичного изображения, их идентификатор ID кодового потока должен появляться в блоке Указателя места заполнения внутри соответствующего бункера метаданных. Единственное исключение к этому требованию делается для развернутых кодовых потоков JPEG 2000, которые не встроены внутри файлового формата семейства JPEG 2000.

Значения идентификаторов ID кодовых потоков, которые появляются внутри уместного блока Указателя места заполнения, должны соответствовать любым требованиям, наложенным с помощью содержащегося файлового формата. Например, файлы JPX формально назначают номер последовательности каждому кодовому потоку, который появляется на верхнем уровне файла, или через блок Смежного кодового потока, или через блок Таблицы фрагментов. Первый кодовый поток верхнего уровня в логической цели должен иметь идентификатор ID кодового потока в значении 0; следующий будет иметь идентификатор ID кодового потока в значении 1; и т. д.

Указатели мест заполнения, которые ссылаются на идентификаторы ID многократных кодовых потоков, могут использоваться только там, где смысл таких кодовых потоков хорошо определен типом блока, который заменяется.

A.3.6.5 Использование блоков Указателей мест заполнения с MJ2

Эта Рекомендация | Международный стандарт определяет только два типа блоков, подходящих для указателей мест заполнения с файлами JPEG 2000 "Движение" (MJ2). Конкретно, или блок смещения фрагмента данных ('stco'), или блок большого смещения фрагмента данных ('co64') могут быть заменены блоком Указателя места заполнения, который обозначает идентификаторы ID многократных кодовых потоков.

Каждая видеодорожка в файле MJ2 содержит в точности один блок смещения фрагмента данных (или 'stco', или 'co64'), что, в сочетании с отсчетом к блоку фрагмента данных ('stsc'), служит для обозначения местоположения всех смежных блоков кодовых потоков, которые принадлежат видеодорожке. Если блок смещения фрагмента данных заменяется указателем места заполнения, что обеспечивает один или более идентификаторов ID кодовых потоков, то должен быть точно один идентификатор ID кодового потока для каждого смежного кодового потока в видеодорожке. Если блок ввода визуального отсчета ('mjp2') обозначает счет поля в значении 2, должно быть *2N* идентификаторов ID кодовых потоков в диапазоне, предоставляемом блоком Указателя места заполнения, где N − количество отсчетов видео (т. е. N есть количество кадров). Иначе, должно быть только N идентификаторов ID кодовых потоков в диапазоне, обеспеченном блоком Указателя места заполнения. Идентификаторы ID кодовых потоков должны быть упорядочены с помощью номера отсчета (номера кадра) и номера поля внутри каждого отсчета.

ПРИМЕЧАНИЕ. − Для файлов MJ2 в представлении JPP-потока или JPT-потока нет никакой необходимости для включения в поток содержимого первоначального блока смещения фрагмента данных, отсчета к блоку фрагмента данных ('stsc'), или блока размера отсчета ('stsz'). Эта информация индексации может быть восстановлена, при необходимости, если представление потока преобразуется в файл MJ2.

A.4 Соглашения для синтаксического анализа и доставки потоков JPP-потоков и JPT-потоков (информативные)

Блоки указателей мест заполнения создают дополнительную гибкость и некоторую потенциальную двусмысленность и для клиентов, и для серверов в том, как они проводят синтаксический анализ или доставляют JPP-потоки и JPT-потоки. Сервер может выбирать расчленение первоначальных блоков из файла семейства JPEG 2000 на бункеры метаданных, используя любую из стратегий широкого диапазона, вводя блоки Указателей мест заполнения в соответствующих точках. Сервер должен сделать это непротиворечивым образом так, чтобы бункеры данных, связанные с JPP-потоком или JPT-потоком, имели то же самое номинальное содержимое для всех клиентов, которые получают доступ к той же самой логической цели (возможно, оцененной идентификатором ID уникальной цели), всякий раз, когда они получают доступ к цели.

Более значительно, однако, то, что блоки Указателей мест заполнения позволяют серверам строить отдельный JPP-поток или JPT-поток, бункеры данных которых обеспечивают многократные дополнительные представления того же самого первоначального содержимого. Это может случаться тогда, когда потоковый эквивалент обозначается внутри указателя места заполнения, и/или когда идентификатор ID возрастающего кодового потока определяется внутри указателя места заполнения. В этих случаях первоначальный блок мог быть сделан доступным в бункере метаданных, будучи при этом также доступным как эквивалент потока еще в одном бункере метаданных, и/или будучи также доступным как возрастающий кодовый поток через бункеры данных заголовков, зон или элементов мозаичного изображения. В то время как серверы могли бы распределять содержимое всех бункеров данных, которые представляют первоначальный блок, ожидается, что по причинам эффективности серверы распределяли бы только достаточную информацию, чтобы перемещать первоначальное содержимое, если от них явно не спрашивают распределение избыточных бункеров данных. Синтаксические анализаторы на стороне клиента для JPP-потоков или JPT-потоков, когда сталкиваются с многократными представлениями первоначального блока, могли бы выбирать игнорирование всех представлений, кроме одного. Ожидаемому соглашению клиента следует иметь существенное влияние в вопросе, какие бункеры метаданных сервер фактически выбирает для отправки клиенту.

С учетом вышеуказанного, эта Рекомендация | Международный стандарт предлагает следующие соглашения:

- Пока сервер не имеет повода верить иному, он должен предполагать, что синтаксический анализатор клиента будет осуществлять синтаксический анализ блока эквивалента потока в предпочтении к первоначальному блоку, если указателями мест заполнения было сообщено клиенту о присутствии обоих типов блоков.
- Пока сервер не имеет повода верить иному, он должен предполагать, что синтаксический анализатор клиента будет использовать представление возрастающего кодового потока (бункеры заголовков, зон или данных элементов мозаичного изображения) в предпочтении к необработанному кодовому потоку, если указателями мест заполнения было сообщено клиенту о присутствии обоих типов блоков.

A.5 Соглашения для взаимодействия JPP-потока или JPT-потока (информативные)

Это соглашение описывает файловый формат обмена для JPP-потока и JPT-потока, которые здесь обозначаются соответственно jpp-файлом и jp-файлом. Такой файл может содержать полученные данные JPEG 2000 из сеанса JPIP (например, кэш-память клиента) или поднабор этих данных. Имеется возможность для другого клиента JPIP читать и использовать этот файл, поскольку JPP-поток и JPT-поток являются типами носителей информации, которые сами себя описывают.

Эти файлы формируются путем сцепки сообщений JPP-потока или JPT-потока. Например, они могут быть сформированы простой сцепкой всех таких сообщений, полученных клиентом в отдельном сеансе или из многократных сеансов. Улучшенная ситуация была бы там, где клиенты порождали бы правомерный JPT-поток или JPP-поток, используя отдельный Заголовок сообщения и Сообщение на каждый бункер данных.

Рекомендуется, чтобы для этих файлов использовались расширения ".jpp" и ".jpt", и, если соответствует, чтобы имя файла включало ссылку на соответствующий маркер JPIP `target` [*цель*] или на маркер `target-id`.

Это соглашение не определяет реализацию или структуру кэш-памяти для клиента. Например, клиент может использовать базу данных, чтобы она служила в качестве его реализации функции кэш-памяти, а не система кэш-памяти на основе файла.

Приложение B

Сеансы, каналы, модель кэш-памяти и наборы моделей

(Это приложение образует составную часть этой Рекомендации | Международного стандарта)

B.1 Запросы внутри сеанса по отношению к запросам, не меняющим своего состояния в процессе исполнения

Протокол JPIP проводит ясное различие между двумя различными типами запросов: запросами, не меняющими своего состояния в процессе исполнения, и запросами, которые принадлежат сеансу.

Цель сеансов состоит в том, чтобы уменьшить количество явной передачи информации, требуемой между клиентом и сервером. Ожидается, что внутри сеанса сервер будет помнить возможности клиента и предпочтения, доставленные в предыдущих запросах, чтобы эту информацию не нужно было посылать в каждом запросе. Что еще более важно, сервер обычно поддерживал бы журнал регистрации информации, которую он уже отослал клиенту в отклике на предыдущие запросы, чтобы эту информацию не нужно было повторно посылать в отклике на будущие запросы. Этот журнал регистрации был бы постоянен для продолжительности сеанса. Если явно не поручено иное, сервер может предполагать, что клиент помещает в кэш-память отклики на все запросы, выпущенные внутри сеанса, и может моделировать кэш-память клиента, посылая только те порции данных или метаданных сжатых изображений, которых клиент уже не имеет в своей кэш-памяти.

Запросы, не меняющие своего состояния в процессе исполнения, не связаны ни с каким сеансом и поэтому должны быть полностью самодостаточными. Следует отметить, что термин "не меняющий состояния в процессе исполнения" применяется только серверу, а не к клиенту. Что касается сеансов, то клиенту следует в общем случае помещать в кэш-память отклики от предыдущих запросов, связанных с той же самой логической целью. Клиентам, которые выпускают многократные запросы, не меняющие своего состояния в процессе исполнения, для той же самой цели, следует вообще включать информацию об их содержимом кэш-памяти с каждым запросом, чтобы избегать передачи избыточных данных. Таким образом, выгодами сеансов являются меньшие, менее сложные запросы и/или менее избыточные данных отклика от сервера. Выгода от обмена информацией, не меняющей своего состояния в процессе исполнения, состоит в том, что сервер не должен поддерживать информацию состояния между запросами; это означает, что тот же самый ведущий узел (хост) не должен, в конечном счете, обслуживать все запросы для одиночного целевого изображения, которые исходят от отдельного клиента.

B.2 Каналы и сеансы

С каждым сеансом связаны следующие элементы:

- Одна или более логических целей (обычно файлы изображения), содержимое которых не изменяется в течение сеанса.
- Тип возврата данных отдельного изображения для каждой логической цели, связанной с сеансом.
- Для каждой логической цели, связанной с сеансом, должна поддерживаться модель содержимого кэш-памяти клиента вне зависимости от того, является ли тип возврата данных одним из потоков "jpp-stream" или "jpt-stream". Отметим, однако, что от этой модели не требуется полностью отражать фактическое состояние кэш-памяти клиента. Правила, управляющие поддержанием моделей кэш-памяти, очерчиваются в разделе B.3.
- Один или более каналов JPIP. В общем случае клиенты могут открывать многократные каналы внутри того же самого сеанса. Каждый канал JPIP может быть связан с отдельным основным транспортным каналом (например, отдельное соединение TCP), хотя это могло не быть правильным. Многократные каналы позволяют клиентам выпускать одновременные запросы для многократных областей изображений, с ожиданием, что сервер ответит на эти запросы одновременно. Каналы также учитывают разумное распределение ширины полосы частот среди различных типов запросов или внутри отдельного целевого изображения, или через многократные цели.
- Там, где многократные каналы связаны с той же самой логической целью, по всем каналам применяется модель кэш-памяти сеанса. Многократные клиенты могут открывать каналы JPIP внутри того же самого сеанса, хотя это могло бы иметь нежелательные побочные эффекты, если каналы относятся к той же самой логической цели.

С каждым каналом связаны следующие элементы:

- Отдельная логическая цель (обычно файл изображения).

- Назначенный сервером идентификатор, который должен быть включен в каждый запрос. Протокол JPIP не определяет идентификатор отдельного сеанса, поскольку идентификатор канала является достаточным, чтобы связывать запрос с его сеансом.
- Запись возможностей и предпочтений клиента, которые могут быть настроены через соответствующие поля запросов.
- Что касается степени, до которой сервер выстраивает очередь для запросов, то ему следует обеспечивать отдельную очередь для каждого канала JPIP.

Есть взаимнооднозначное соответствие для запроса клиента и отклика клиента на канале. Различные каналы JPIP могут быть на том же самом транспортном канале или на различных транспортных каналах. Запросы, которые используют различные каналы JPIP, могут прибывать в сервер асинхронно, если для транспортирования запросов используются отдельные транспортные каналы. Отклики, которые используют различные каналы JPIP, могут прибыть к клиенту асинхронно, если для транспортирования откликов используются отдельные транспортные каналы. Обслуживание многократных каналов оставлено на усмотрение сервера; однако полю запроса скорости доставки и предпочтениям максимальной ширины полосы частот и среза ширины полосы частот следует направлять сервер.

B.3 Управление моделью кэш-памяти

Как уже было отмечено, одной из основных функций сеанса является моделирование кэш-памяти клиента со стороны сервера. Если явно не сообщено иное, сервер **может** предполагать, что клиент сохранил в памяти всю информацию, посланную в отклике на запросы внутри сеанса: эта информация **не нуждается** в повторной передаче. Отметим, однако, что сервер не обязан поддерживать полную модель кэш-памяти или действительно любую модель кэш-памяти вообще: избыточные данные **могут** быть переданы в отклике на запросы.

В дополнение к воздействию переданных данных, явные утверждения (операторы) об обработке (манипуляции) модели кэш-памяти в запросах клиентов могут обновлять модель кэш-памяти сервера. Эти утверждения должны быть обработаны перед определением данных, которые следует возвращать клиенту в отклике на его запрос. Имеются два типа утверждений обработки модели кэш-памяти: добавляющее и вычитающее.

Добавляющие утверждения обработки модели кэш-памяти служат для увеличения модели кэш-памяти сервера, добавляя бункеры данных или порции бункеров данных к существующей модели. Для клиента они обеспечивают механизм, чтобы сообщать серверу об информации, которую он получил в предыдущем сеансе, или используя предыдущие запросы, не меняющие своего состояния в процессе исполнения. Серверу **следует** пытаться использовать любые добавляющие утверждения обработки модели кэш-памяти, которые появляются в запросах клиента. Однако серверы не обязаны поддерживать полную модель кэш-памяти, поэтому сервер может игнорировать, или частично игнорировать добавляющие утверждения обработки модели кэш-памяти.

Вычитающие утверждения служат для удаления бункеров данных или порций бункеров данных из модели кэш-памяти сервера. Клиент мог бы выпускать вычитающие утверждения обработки модели кэш-памяти, чтобы сообщать серверу, что он сохранил или сбросил некоторые данные, которые предварительно были посланы сервером. В противном случае, сервер свободен предполагать, что клиент сохранил в памяти все данные, переданные в течение сеанса. Сервер **должен** удалить всю информацию, определяемую вычитающим утверждением модели кэш-памяти из любой модели кэш-памяти (завершенной или иной), которую он поддерживает.

Запросы JPIP на основе сеанса имеют побочные результаты, которые могут затронуть отклик на будущие запросы. Это верно также для запросов, которые содержат утверждения обработки модели кэш-памяти − воздействия обработки модели кэш-памяти являются устойчивыми. Кроме того, побочные результаты запроса, прибывающего в один канал JPIP, отражаются в отклике на любые запросы, которые могли бы принадлежать различному каналу JPIP, который связан с той же самой логической целью. Это следует из факта, что есть только одна модель кэш-памяти для каждой логической цели в сеансе.

B.4 Опрос и обработка наборов моделей

Там, где логическая цель, связанная с сеансом, содержит большое количество кодовых потоков (например, цель видео), или клиент остается подключенным на длительный период времени, частичное моделирование кэш-памяти становится все более и более похожим на стратегию для практических реализаций серверов. Также становится все более и более вероятным, что клиенты будут не способны сохранять в кэш-памяти всю информацию, посланную сервером. Чтобы избегать неэффективности связи в таких обстоятельствах, вводится понятие "mset" (модель-набор). Модель "mset" есть совокупность кодовых потоков, для которых содержимое кэш-памяти клиента моделируется сервером.

В любом запросе клиент может проинструктировать сервер ограничивать его "mset" конкретным набором кодовых потоков. Это обеспечивает удобный механизм для клиентов, чтобы сбрасывать целые кодовые потоки из их кэш-памяти, не подвергаясь риску, что сервер будет порождать неполные отклики на будущие запросы о таких кодовых потоках.

Модель "mset" запрашивает также результат в откликах сервера, указывающих фактический набор кодовых потоков, для которых поддерживается информация модели кэш-памяти. Это позволяет клиентам определять, действительно ли утверждения обработки модели кэш-памяти, которые относятся к разнообразию кодовых потоков, будут сброшены сервером.

В отсутствие любой явной обработки или опроса "mset", клиент может только предполагать, что "mset" сервера включает все кодовые потоки, для которых данные отклика порождаются к его запросу. Так как серверы вообще имеют право ограничивать сферу действия запроса клиента до меньшего количества кодовых потоков, чем количество, которое было определено первоначально, нет никакой гарантии, что "mset" сервера будет включать все кодовые потоки, упомянутые в запросе, если запрос не упоминал только один кодовый поток. Эти вопросы объясняются далее в подразделе C.8.6.

Приложение C

Запрос клиента

(Это приложение образует составную часть этой Рекомендации | Международного стандарта)

C.1 Синтаксис запроса

C.1.1 Введение

Это приложение описывает все возможные элементы в запросе JPIP. Каждый главный подраздел описывает группу полей и возможных значений для таких полей. В общем случае, запрос будет состоять из полей более чем одной группы, но некоторые группы являются несовместимыми. Далее, внутри каждой группы некоторые поля запросов являются несовместимыми. Некоторые противоположные допустимые запросы могут не быть действительными для использования в некоторых ситуациях (например, сеансы), даже если это не обозначено синтаксисом BNF. Наконец, даже с допустимым запросом, сервер может не поддерживать все возможные поля запросов или их сочетания.

C.1.2 Структура запроса

Запрос JPIP состоит из следующих полей:

- Полей идентификации цели;
- Полей управления сеансом и каналом;
- Полей запросов окон обзора;
- Поля метаданных;
- Полей запросов ограничения данных;
- Полей запросов управления сервером;
- Полей запросов управления кэш-памятью;
- Полей запросов загрузки в сервер;
- Полей возможности и предпочтения клиента.

Элементы в запросе должны посылаться в соответствии с выбранным транспортным протоколом. Например, в протоколе HTTP запросы выражаются как знаки, перечисленные в синтаксисе BNF, многократные параметры присоединяются со знаком "&", и запросы могут быть частью поля вопроса из запроса GET [*получить*] или тела из запроса POST [*сообщить*]. Для подробностей, см. Приложения F, G и H.

ПРИМЕЧАНИЕ. – Знаков, зарезервированных идентификатором URI, можно избежать. Например, "request=a:b" в указателе URL GET протокола HTTP будет иметь своим результатом "request=a%3Ab", где зарезервированный знак URL ':' переключается на '%3A'.

```
jpip-request-field = target-field
                   / channel-field
                   / view-window-field
                   / metadata-field
                   / data-limit-field
                   / server-control-field
                   / cache-management-field
                   / upload-field
                   / client-cap-pref-field

target-field       = target                          ; C.2.2
                   / subtarget                       ; C.2.3
                   / tid                             ; C.2.4

channel-field      = cid                             ; C.3.2
                   / cnew                            ; C.3.3
                   / cclose                          ; C.3.4
                   / qid                             ; C.3.5
```

```
view-window-field  = fsiz                    ; C.4.2
                   / roff                    ; C.4.3
                   / rsiz                    ; C.4.4
                   / comps                   ; C.4.5
                   / stream                  ; C.4.6
                   / context                 ; C.4.7
                   / srate                   ; C.4.8
                   / roi                     ; C.4.9
                   / layers                  ; C.4.10

metadata-field     = metareq                 ; C.5.2

data-limit-field   = len                     ; C.6.1
                   / quality                 ; C.6.2

server-control-field = align                 ; C.7.1
                   / wait                    ; C.7.2
                   / type                    ; C.7.3
                   / drate                   ; C.7.4

cache-management-field = model               ; C.8.1
                   / tpmodel                 ; C.8.3
                   / need                    ; C.8.4
                   / tpneed                  ; C.8.5
                   / mset                    ; C.8.6

upload-field       = upload                  ; C.9.1

client-cap-pref-field = cap                  ; C.10.1
                   / pref                    ; C.10.2
                   / csf                     ; C.10.3
```

C.1.3 Ограничения на сочетание полей запросов

Каждый тип поля запроса JPIP должен случаться не более одного раза в одиночном запросе.

В общем случае, запросы для данных изображения (запросы окон обзора) могут быть объединены с запросами для дополнительных метаданных. Однако имеются ограничения, как можно объединять поля запросов.

Поле запроса при загрузке файла в сервер не должно объединяться с полями `metadata-field` [*метаданные−поле*], `data-limit-field` [*данные−предел−поле*] или `server-control-field` [*сервер−управление−поле*].

C.2 Поля идентификации целей

C.2.1 Введение в логические цели

Каждый запрос JPIP направляется к характерному представлению характерного первоначального именованного ресурса или к характерной порции такого ресурса. Такой ресурс может быть физически сохраненным файлом или объектом, или может быть кое-чем, что виртуально создано сервером по запросу.

Характерное представление, является ли оно первоначальной кодируемой формой или формой с преобразованием кода, или является ли оно характерным диапазоном байтов, или является полным ресурсом, упоминается как логическая цель. Логическая цель определяется с помощью трех полей запроса: идентификатора ID Цели [*Target ID*], Цели [*Target*] и Подцели [*Sub-target*].

Поле запроса Цели определяет первоначальный именованный ресурс, к которому направляется запрос. Он определяется, используя `PATH` [*тракт*], который мог быть простой строкой или идентификатором URI. Если поле Цели не указывается, и запрос переносится поверх протокола HTTP, то тогда запрос JPIP должен быть направлен к ресурсу, указанному через компонент тракта URL запроса JPIP. Этот первоначальный именованный ресурс может быть фактическим файлом или другим объектом, физически сохраненным на сервере, или он может быть чем-то, что сервер создает в отклике на запрос JPIP.

Поле запроса Подцели указывает характерный диапазон байтов первоначального именованного ресурса (указанного через поле запроса Цели), к которому направляется запрос. Если поле запроса Подцели не указывается, то запрос направляется к полному диапазону байтов первоначального ресурса.

Поле запроса идентификатора ID Цели может использоваться, чтобы далее определять конкретное кодирование ресурса в ситуациях, где клиент и сервер предварительно обменялись данными из этого ресурса. Например,

сервер мог предварительно поставить клиенту перекодированную версию файла на основе полученной информации и условий по предыдущему запросу. Если такой клиент сохранил предварительно переданные данные в своей кэш-памяти, он пожелает продолжать получать данные, используя то же самое перекодирование, чтобы он мог продолжить использовать данные в кэш-памяти. Идентификатор ID Цели является определенной сервером строкой идентификации, к которой сервер предварительно был присоединен с тем характерным представлением такого характерного первоначального именованного ресурса, или диапазона байтов некоторого характерного первоначального именованного ресурса.

Если клиент определяет и первоначальный именованный ресурс (или через поле запроса Цели, или через компонент тракта URL Запроса JPIP), и идентификатор ID Цели, то сервер должен проверить, действительно ли он может ответить на запрос в той же самой манере, как тогда, когда он первоначально назначил такой идентификатор ID Цели такому ресурсу. Если сервер не может откликнуться в той же самой манере, он должен использовать заголовок отклика JPIP-tid, чтобы информировать клиента нового идентификатора ID Цели, в какой точке клиент будет знать, что он должен сбросить любые предварительно сохраненные данные.

Если логическая цель должна обслуживаться с помощью сообщений JPP-потока или JPT-потока, связанные бункеры данных должны оставаться совместимыми в течение всех откликов, которые выпускаются внутри того же самого сеанса. Там, где сервер или связанный сервер, также выпускают идентификатор ID Цели, бункеры данных должны оставаться совместимыми по всем откликам, выпущенным с тем же самым идентификатором ID Цели, вне зависимости от того, выпущены ли они внутри того же самого сеанса, или нет.

Если этот запрос является частью идентификатора ID сеанса и канала, как было назначено сервером, клиент может определять идентификатор ID канала через поле запроса ID Канала вместо указания полей запросов Цели, Подцели и ID Цели. Если логическая цель определяется и через сочетание полей Цели, Подцели и ID Цели, и через поле запроса ID Канала, сервер должен откликнуться сообщением об ошибке.

Следующие примеры показывают спецификацию логических целей:

ПРИМЕР 1: Для URL запроса JPIP из

"http://one.jpeg.org/imageserver.cgi?target= http%3A%2F%2Fone.jpeg.org%2Fimages%2Fpicture.jp2&fsiz=200,200" логическая цель является полным диапазоном байтов, содержащимся в пределах URL "http://one.jpeg.org/images/picture.jp2, " относительно каталога корневого документа сервера.

ПРИМЕР 2: Для URL запроса JPIP из

"http://one.jpeg.org/imageserver.cgi? target= http%3A%2F%2Fone.jpeg.org%2Fimages%2Fpicture.jp2&tid=4384-5849-af4d-3dca&fsiz=200,200" логическая цель является полным диапазоном байтов, содержащимся в пределах URI "http://one.jpeg.org/images/picture.jp2," относительно каталога корневого документа сервера, с представлением, указанным с помощью определенного сервером идентификатора ID Цели 4384-5849-af4d-3dca.

ПРИМЕР 3: Для URL запроса JPIP из

"http://one.jpeg.org/imageserver.cgi?target= http%3A%2F%2Fone.jpeg.org%2Fimages%2Fpicture.jp2&subtarget=1038-13458&fsiz=200,200" логическая цель является диапазоном байтов, начинающимся с байта 1038, и со всеми байтами до, и включая байты 13458, которые содержатся внутри идентификатора URI "http://one.jpeg.org/images/picture.jp2," относительно каталога корневого документа сервера.

ПРИМЕР 4: Для URL запроса JPIP из "http://one.jpeg.org/imageserver.cgi?cid=1234-5849-af4d-3dca&fsiz=200,200" логическая цепь является ресурсом, к которому сервер был соединен каналом с идентификатором ID 1234-5849-af4d-3dca.

ПРИМЕР 5: Для URL запроса JPIP из "http://one.jpeg.org/images/picture.jp2?fsiz=200,200" логическая цель является полным диапазоном байтов, содержащимся внутри файла "images/picture.jp2," относительно каталога корневого документа сервера.

ПРИМЕР 6: Для URL запроса JPIP из "http://one.jpeg.org/images/picture.jp2?subtarget=1038-13458&fsiz=200,200" логическая цель является диапазоном байтов, начинающимся с байта 1038, и со всеми байтами до, и включая байт 13458, содержащимися внутри файла "images/picture.jp2," относительно каталога корневого документа сервера.

C.2.2 Цель (target)

```
target = "target" "=" PATH
```

Это поле используется для указания первоначального именованного ресурса (часто имени файла на сервере). Если поле запроса Цели отсутствует, то тогда первоначальный именованный ресурс определяется другими средствами.

C.2.3 Подцель (subtarget)

```
subtarget = "subtarget" "=" byte-range

byte-range = UINT-RANGE
```

Это поле может быть использовано для оценки первоначального именованного ресурса через спецификацию диапазона байтов. Логическая цель должна истолковываться, как указано диапазоном байтов первоначального именованного ресурса.

Нижняя и верхняя границы поставляемого диапазона байтов являются включающими, а 0 относится к первому байту файла цели.

C.2.4 Идентификатор ID Цели (tid)

```
tid = "tid" "=" target-id

target-id = TOKEN
```

Это поле может быть использовано для подачи строки `target-id`, которая была ранее порождена сервером для абсолютного определения логической цели, к которой осуществляется доступ, включая любую используемую по собственному усмотрению перекодировку, выполняемую сервером. Имя логической цели не является обязательно уникальным и не обязательно соответствует единственному кодированию его содержимого, тогда как строке `target-id`, вместе с именем первоначального ресурса и диапазоном байтов, следует, безусловно, определять как совокупность изображений, так и ее кодирование.

Если `target-id` есть "0", логическая цель определяется через использование компонентов тракта URL Цели, Подцели и JPIP, и клиент недвусмысленно запрашивает, чтобы север информировал его о назначенном идентификаторе `target-id`, если такой имеется. Сервер должен включать заголовок ID Цели в свой отклик ко всем клиентам с помощью `target-id` вида "0".

Длина идентификатора `target-id` не должна превышать 255 знаков.

C.3 Поля для работы с сеансами и каналами

C.3.1 Введение

Запрос должен не менять своего состояния в процессе исполнения, пока не возникают одно или оба из следующих состояний:

- Запрос включает правомерное поле ID Канала;
- Запрос включает поле Нового канала [*New Channel*] (см. ниже), а отклик сервера включает в себя заголовок отклика Нового канала с вновь выпущенным идентификатором `channel-id`.

См. Раздел B.2 для обсуждения по сеансам и каналам.

C.3.2 Идентификатор ID Канала (cid)

```
cid = "cid" "=" channel-id

channel-id = TOKEN
```

- Это поле используется, чтобы связывать запрос с конкретным каналом JPIP, и, следовательно, с сеансом, к которому принадлежит канал.

C.3.3 Новый канал (cnew)

```
cnew = "cnew" "=" 1#transport-name

transport-name = TOKEN
```

Это поле используется для запроса нового канала JPIP. Если поле запроса ID Канала не присутствует, то осуществляется запрос для нового сеанса. В противном случае, запрос осуществляется для нового канала в том же самом сеансе, что и канал, определенный полем запроса ID Канала.

Строка значения определяет имена одного или более транспортных протоколов, что клиент желает принять. Эта Рекомендация | Международный стандарт определяет только транспортные имена, "http" и "http-tcp," хотя ожидается, что где-то в другом месте могут быть определены такие другие транспортные средства, как "udp". Подробности использования протокола JPIP поверх транспортного средства "http" показаны в Приложении F, в то время как подробности использования протокола JPIP поверх транспортного средства "http-tcp" показаны в Приложении G.

Если сервер имеет желание открыть новый канал, используя один из указанных транспортных протоколов, он должен возвратить маркер идентификатора нового канала, используя заголовок отклика Нового канала (см. раздел D.2.3). В этом случае присутствующий запрос является первым запросом внутри нового канала.

Для клиента есть возможность открывать канал для новой логической цели внутри того же самого сеанса. Чтобы сделать это, запрос клиента должен указывать и существующий идентификатор ID Канала, и логическую цель. При открытии канала для той же самой логической цели, которая связана с существующим каналом, нет необходимости подробно указывать логическую цель.

Если сервер не желает открывать новый канал, он не должен возвращать заголовок отклика Нового канала, но запрос должен обслуживаться так, как если бы поле запроса Нового канала не было включено. Это означает, что запрос, который определяет существующий идентификатор ID Канала, будут рассматривать как запрос внутри такого канала, в то время как запрос, который не включает никакое поле запроса ID Канала, должен обрабатываться как запрос, не меняющий своего состояния в процессе исполнения. Когда запрос Нового канала определяет различную логическую цель по отношению к той, которая связана с поставленным существующим идентификатором ID Канала, сервер не будет способен откликнуться на запрос либо без выпуска нового идентификатора ID Канала, либо без возвращения кода ошибки.

ПРИМЕР 1: "target=nice.jp2&cnew=http" запрашивает первый канал нового сеанса для изображения "nice.jp2", используя транспортное средство "http". Если сервером канал не назначается, то запрос будет обрабатываться как не меняющий своего состояния в процессе исполнения.

ПРИМЕР 2: "cid=013ac8&cnew=http-tcp" запрашивает новый канал внутри того же самого сеанса, который связан с идентификатором ID Канала 013ac8. Новый канал собирается использовать транспортное средство "http-tcp" и ссылается на ту же самую логическую цель, как и идентификатор ID Канала 013ac8. Этими каналами совместно используется единственная модель кэш-памяти. Если сервером канал не назначается, то запрос будет обрабатываться так, как если бы поле запроса Нового канала было опущено.

ПРИМЕР 3: "target=nice.jp2&cid=013ac8&cnew=http" запрашивает новый канал внутри того же самого сеанса, который связывается с помощью идентификатора ID Канала "013ac8." Новый канал собирается использовать транспортное средство "http". Логическая цель, связанная с новым каналом, отличается от цели, что связана с идентификатором ID Канала "013ac8", и для нового канала используется модель отдельной кэш-памяти. Модели кэш-памяти для обеих целей связываются с этим общим сеансом.

C.3.4 Канал закрыть (cclose) [*Channel Close*]

```
cclose = "cclose" "=" ("*" / 1#channel-id)
```

Это поле используется для того, чтобы закрывать один или более открытых каналов. Если поле значения содержит один или более маркеров channel-id, они должны принадлежать к тому же самому сеансу. В этом случае поле запроса ID Канала не является необходимым, но если оно обеспечивается, оно должно упоминать канал, принадлежащий к тому же самому сеансу.

Если поле значения есть "*", то все каналы, связанные с сеансом, будут закрыты. В этом случае сеанс должен указываться путем включения поля запроса ID Канала.

Сервер должен завершить свой отклик на любом канале, указанном в запросе "Канал закрыть", перед фактическим закрытием канала.

C.3.5 Идентификатор ID Запроса (qid)

```
qid = "qid" "=" UINT
```

Это поле используется для определения значения ID Запроса [*Request ID*]. Каждый канал имеет свою собственную очередь запросов, со своим собственным счетчиком ID Запроса. Запросы, которые были получены внутри любого заданного канала (как указано значением ID Канала), должны обрабатываться в порядке их значений ID Запроса, где используется поле ID Запроса. Сервер может обрабатывать запросы, которые не содержат поле ID Запроса на основе принципа "первым пришел - первым обслужен". Однако он не должен обрабатывать запрос, который прибывает со значением ID Запроса, равным *n*, до тех пор, пока он не обработал все запросы со значением ID Запроса меньше, чем *n*, которое связывается с тем же самым каналом, исключая *n*=0. Клиент не должен выпускать запрос, который указывает то же самое значение ID Запроса, как любой другой запрос, связанный с тем же самым каналом, и не должен выпускать идентификаторы ID Запроса, которые меньше, чем любой ранее выпущенный идентификатор ID Запроса на этом канале.

C.4 Поля запросов окон обзора

C.4.1 Преобразование запросов окон обзора в разрешающие способности и области изображения кодовых потоков

Цель протокола JPIP состоит в том, чтобы обеспечивать порции изображения JPEG 2000 и связанные метаданные в отклике на запросы от клиента. Это делается через последовательность запросов и откликов. Для порции изображения запрашиваемые данные могут быть меньше, чем полное изображение в понятиях размера кадра изображения, области, качества и/или компонентов.

В самом простом случае рассматриваемая часть изображения определяется непосредственно по отношению к эталонной сетке высокой разрешающей способности кодового потока (потоков) JPEG 2000, указанной в запросе, а не к сетке дискретизации какого-нибудь конкретного компонента изображения. В общем случае, однако, клиенты могут запрашивать объекты изображения верхнего уровня (например, слои наложения изображений JPX или видеодорожки MJ2) через поле запроса Контекста кодового потока [*Codestream Context*] (см. C.4.7). В этом случае запрашиваемая часть изображения, возможно, должна подвергаться преобразованию координаты, чтобы определять часть каждого связанного кодового потока, который запрашивается. Эти преобразования координат описаны в C.4.7, и их нужно понимать в понятиях следующего описания областей изображения кодового потока.

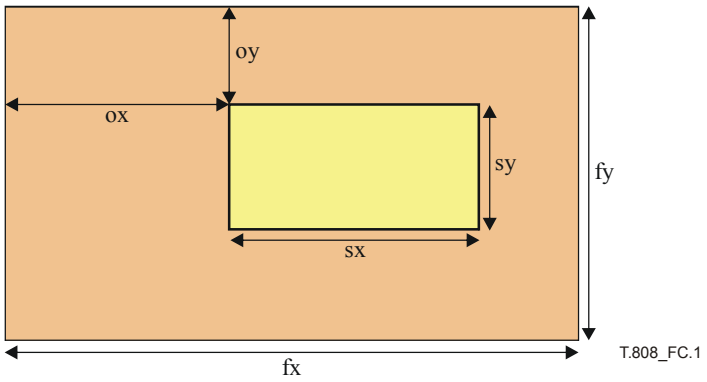

Рисунок C.1 − Желательная область внутри изображения

Области изображений кодовых потоков описываются с использованием 3 двумерных параметров, как показано на рисунке C.1. Параметры размера (`sx` и `sy`) и параметры смещения (`ox` и `oy`) определяют ширину и высоту желательной области изображения кодового потока и верхний левый угол такой области относительно целого изображения, которое имеет размер заданного кадра (`fx` и `fy`).

ПРИМЕР 1: Клиент, желающий заполнить полным изображением устройство отображения с размерами 640 × 480, мог бы сделать запрос следующего вида "`fsiz=640,480&rsiz=640,480&roff=0,0`". Отметим, что это может быть сделано независимо от первоначального размера изображения (и конечно, без знания о первоначальном размере изображения).

Когда ни одна из имеющихся разрешающей способностей изображения в кодовом потоке JPEG 2000 не соответствует точно запрашиваемому размеру кадра, возвращаемые данные изображения могут быть больше или меньше, чем требуемый размер кадра, и могут даже отличаться от формата кадра. Сервер должен определить подходящую разрешающую способность изображения кодового потока, обозначенную параметрами размера `fx' and fy'`, и подходящую область на кодовом потоке, обозначенном параметрами `sx'`, `sy'`, `ox'` и `oy'`, как показано в рисунке C.2. Хотя клиент может указывать направление для округления, как часть поля запроса Размера кадра [*Frame Size*], клиент должен быть готов иметь дело с возвращаемыми данными, которые не соответствуют точно требуемым параметрам.

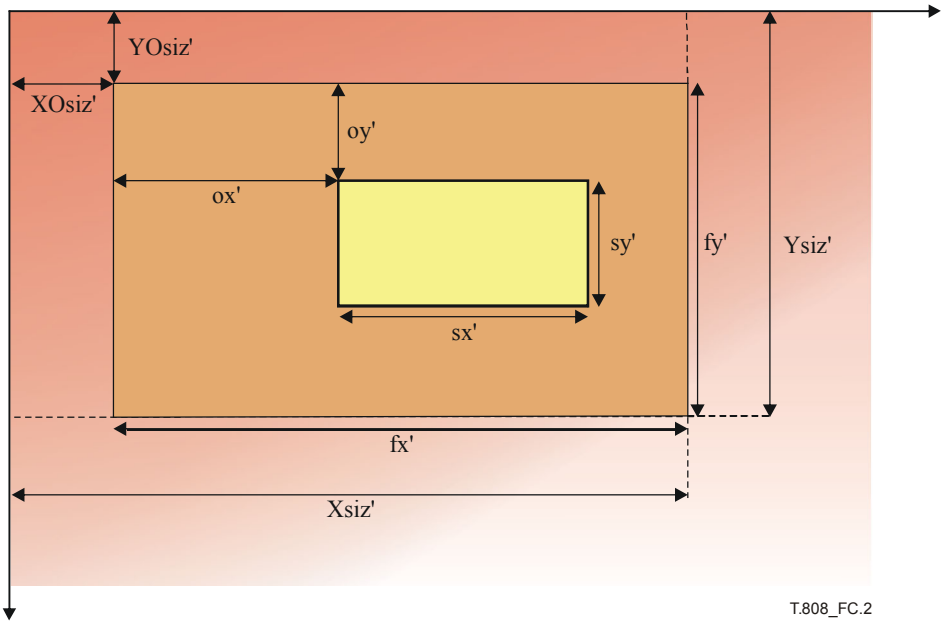

Рисунок C.2 − Желательная область по отношению к эталонной сетке с пониженной скоростью дискретизации

Как показано на рисунке C.2, размер разрешающей способности изображения подходящего кодового потока дается параметрами `fx' = Xsiz' -XOsiz'` и `fy' = Ysiz' - YOsiz'`, где `XOsiz'`, `YOsiz'`, `Xsiz'` и `Ysiz'` извлекаются, используя Уравнение C-1.

$$XOsiz' = \left\lceil \frac{XOsiz}{2^r} \right\rceil; \quad YOsiz' = \left\lceil \frac{YOsiz}{2^r} \right\rceil; \quad Xsiz' = \left\lceil \frac{Xsiz}{2^r} \right\rceil; \quad Ysiz' = \left\lceil \frac{Ysiz}{2^r} \right\rceil, \qquad \text{(C-1)}$$

где:

r определяется сервером, чтобы согласовать размер запрашиваемого изображения (`fx` и `fy`) как можно ближе, в зависимости от предпочтений по округлению, подаваемых через поле запроса Размера кадра.

Здесь XOsiz, YOsiz, Xsiz и Ysiz берутся из соответствующего сегмента маркера SIZ кодового потока. Принято истолковывать *r* как количество сброшенных наивысших уровней DWT, и конечно, *r* должно быть целым числом не менее 0. Однако значение *r* не ограничивается количеством уровней DWT, которые были использованы для сжатия какого-либо компонента элемента мозаичного изображения в кодовом потоке.

Как только подходящие размеры кадров, `fx'` и `fy'`, были найдены, размер области, `sx'` и `sy'`, и смещения, `ox'` и `oy'`, связанные с областью изображения кодового потока, определяются Уравнением C-2.

$$ox' = \left\lfloor ox \cdot \frac{fx'}{fx} \right\rfloor; \quad oy' = \left\lfloor oy \cdot \frac{fy'}{fy} \right\rfloor; \quad sx' = \left\lceil (sx + ox) \cdot \frac{fx'}{fx} \right\rceil - ox'; \quad sy' = \left\lceil (sy + oy) \cdot \frac{fy'}{fy} \right\rceil - oy' \qquad \text{(C-2)}$$

ПРИМЕР 2: Предположим, что запрашиваемый Размер кадра равен 128 × 128, а изображение на эталонной сетке высокой разрешающей способности описывается с помощью XOsiz=127, Xsiz=648, YOsiz=0 и Ysiz=504. Предположим также, что в кодовом потоке существуют 3 уровня преобразования элементарных волн [*wavelet transform*] для всех компонентов изображения. Тогда имеющиеся размеры изображения кодовых потоков составят:

$$521\times504 \quad \left(\left\lceil\frac{648}{1}\right\rceil-\left\lceil\frac{127}{1}\right\rceil \text{ by } \left\lceil\frac{504}{1}\right\rceil-0\right)$$

$$260\times252 \quad \left(\left\lceil\frac{648}{2}\right\rceil-\left\lceil\frac{127}{2}\right\rceil \text{ by } \left\lceil\frac{504}{2}\right\rceil-0\right)$$

$$130\times126 \quad \left(\left\lceil\frac{648}{4}\right\rceil-\left\lceil\frac{127}{4}\right\rceil \text{ by } \left\lceil\frac{504}{4}\right\rceil-0\right)$$

$$65\times63 \quad \left(\left\lceil\frac{648}{8}\right\rceil-\left\lceil\frac{127}{8}\right\rceil \text{ by } \left\lceil\frac{504}{8}\right\rceil-0\right)$$

Таким образом, если запрос осуществляется для большего размера кадра (`round-direction` есть `round-up`) [*округление-направление есть округление в большую сторону*], возвращаемый размер кадра будет 260 × 252. Если запрос осуществляется для кадра меньшего размера (`round-direction` есть `round-down`) [*округление-направление есть округление в меньшую сторону*], тогда будет использоваться размер кадра 65 × 63. Отметим, что, как в этом примере, имеющиеся размеры кадров кодовых потоков в общем случае не являются точными степенями от 2.

Дискретизация с пониженной скоростью компонентов изображения, как определяется с помощью `XRsiz` и `YRsiz`, не оказывает воздействия на истолкование области запрашиваемого изображения или разрешающей способности изображения внутри любого запрашиваемого кодового потока.

ПРИМЕР 3: Запрос для области 256 × 256 из верхнего левого угла изображения с размерами 512 × 512 может быть сделан с помощью:

fsiz=512,512&rsiz=256,256

Предположим, что кодовый поток содержит изображение, обработанное на пониженной скорости дискретизации в компонентах 1 и 2, но не в компоненте 0. Более точно, предположим `Xsiz`=1024, `Ysiz`=1024, `XOsiz`=0, `YOsiz`=0, и $\texttt{XRsiz}^0$=1, $\texttt{YRsiz}^0$=1, $\texttt{XRsiz}^1$=2, $\texttt{YRsiz}^1$=2, $\texttt{XRsiz}^2$=2, и $\texttt{YRsiz}^2$=2. Сервер не обратил бы внимания на уровень наивысшей разрешающей способности всех трех компонентов, и вернул бы элементы мозаичного изображения или зоны, достаточные для обеспечения 256 × 256 отсчетов компонента 0, но только 128 × 128 отсчетов компонентов 1 и 2. Таким образом, клиент имеет данные для отображения верхнего левого угла на половине размера полного изображения и все еще обработанного на пониженной скорости дискретизации. Если клиент желает отобразить компоненты цветности без дискретизации с пониженной скоростью, он мог выпустить такой дополнительный запрос, как:

fsiz=1024,1024&rsiz=512,512&comps=1,2

Сервер затем вернул бы достаточные данные для обеспечения 256 × 256 отсчетов компонентов 1 и 2, которые могли быть объединены с данными компонента 0, уже принятыми для получения изображения без дискретизации на пониженной скорости, но половинного размера.

Если все три компонента были подвергнуты дискретизации на пониженной скорости, сервер предоставил бы только 128 × 128 отсчетов всех трех компонентов для первоначального запроса (`fsiz=512,512&rsiz=256,256`), поскольку разрешающая способность изображения и области изображений оцениваются относительно эталонной сетки каждого запрашиваемого кодового потока.

C.4.2 Размер кадра (fsiz)

```
fsiz = "fsiz" "=" fx "," fy ["," round-direction]

fx = UINT

fy = UINT

round-direction = "round-up" / "round-down" / "closest"
```

Это поле используется для определения разрешающей способности, связанной с запрашиваемым окном обзора. Значения `fx` и `fy` точно определяют размеры желаемой разрешаемой способности изображения. Значение `round-direction` определяет, как разрешающая способность изображения имеющегося кодового потока должна выбираться для каждого запрашиваемого кодового потока, если разрешающая способность запрашиваемого изображения не имеется внутри такого кодового потока. Размер запрашиваемого кадра преобразуется в разрешающую способность изображения кодового потока, следуя процедуре, описанной

в C.4.1, возможно, с добавлением преобразований координат, запрашиваемых через поле запроса Контекста кодового потока (см. C.4.7). Клиент, желающий управлять точным количеством отсчетов, получаемых для компонента конкретного изображения, может нуждаться в увеличении размера запрашиваемого кадра, как объясняется в C.4.1. Варианты выбора `round-direction`, определяемые этой Рекомендацией | Международным стандартом, описываются в таблице C.1.

Таблица C.1 − Варианты направления округления

`Округление-направление`	**Смысл**
`"round-up"` [*округление−в большую сторону*]	Для каждого запрашиваемого кодового потока должна быть выбрана наименьшая разрешающая способность изображения кодового потока, чьи и ширина, и высота больше, чем определяемый размер, или равны ему. Если такой нет, то тогда должна быть использована наибольшая разрешающая способность изображения имеющегося кодового потока.
`"round-down"` [*округление− в меньшую сторону*]	Для каждого запрашиваемого кодового потока должна быть выбрана наибольшая разрешающая способность изображения кодового потока, чьи и ширина, и высота меньше, чем определяемый размер, или равны ему. Это есть значение по умолчанию, когда параметр `round-direction` не определяется.
`"closest"` [*ближайшее*]	Для каждого запрашиваемого кодового потока должна быть выбрана разрешающая способность изображения кодового потока, которая есть `closest` к определяемому размеру в площади (где площадь = fx × fy). Там, где разрешающие способности изображений двух кодовых потоков имеют площади, которые равноудалены от fx × fy, должна из двух выбираться та, что больше.

Если поле запроса Размера кадра опускается из запроса окна обзора и параметр `metadata-only` [*метаданные - только*] не определяется в поле запроса метаданных (см. C.5.1), то запрашиваемое окно обзора не включает в себя данные сжатого изображения и заголовки, зависящие от элементов мозаичного изображения; но оно действительно включает всю другую информацию заголовка (кодовый поток и файловый формат), которая была бы возвращена, если бы клиент включил поле запроса Размер заголовка. См. C.5.1 для дальнейших сведений относительно информации файлового формата (метаданные), которая неявным образом запрашивается вместе с запросом окна обзора.

C.4.3 Смещение (roff)

```
roff = "roff" "=" ox "," oy

ox = UINT

oy = UINT
```

Это поле используется для определения левостороннего верхнего угла (смещение) пространственной области, связанной с запрашиваемым окном обзора; если оно не присутствует, то смещения по умолчанию устанавливаются в 0. Фактический сдвиг области изображения кодового потока от левостороннего верхнего угла изображения, на фактической разрешающей способности изображения кодового потока, выбранной сервером, получается, следуя процедуре, описанной в C.4.1, возможно, с добавлением преобразований координат, запрашиваемых через поле запроса Контекста кодового потока (см. C.4.7).

Использование поля Смещение [*Offset*] является правомерным только в союзе с полем запроса Размера кадра.

Если область изображения кодового потока, определенная с использованием Размера области и/или Смещения, оказывается пустой (нет площади), отклику сервера не следует включать в себя никакие данные сжатых изображений для того кодового потока. В частности, откликам типа JPP-потока или JPT-потока не следует содержать никаких сообщений, которые ссылаются на бункеры данных зон, элементов мозаичного изображения или заголовков элементов мозаичного изображения того кодового потока. Сервер может, по своему усмотрению, предпочесть возвращать сообщения главного заголовка или бункера метаданных, которые были бы возвращены в отклике на запрос, который не включил поле запроса Размера кадра.

C.4.4 Размер области (rsiz)

```
rsiz = "rsiz" "=" sx "," sy

sx = UINT

sy = UINT
```

Это поле используется для определения горизонтального и вертикального расстояний (размера) пространственной области, связанной с запрашиваемым окном обозрения; если оно не присутствует, то область простирается до правостороннего нижнего угла изображения. Фактические размеры области изображения кодового потока, на фактической разрешающей способности изображения кодового потока, выбранной сервером, вычисляются, следуя процедуре, описанной в C.4.1, возможно, с дополнением преобразований координат, которые запрашиваются через поле запроса Контекста кодового потока (см. C.4.7). Запрашиваемая область изображения кодового потока не обязательно полностью должна содержаться внутри кодового потока; в этом случае сервер просто берет пересечение между областью изображения имеющегося кодового потока и запрашиваемой областью.

Использование поля запроса Размера области является правомерным только в союзе с полем запроса Размера кадра.

Область изображения кодового потока может быть пуста, например, если `sx` или `sy` были нулями. Если область пуста, то отклику сервера не следует включать никакие данные сжатого изображения для такого кодового потока. В частности, откликам типа JPP-потока или JPT-потока не следует содержать никаких сообщений, которые ссылаются на бункеры данных зон, элементов мозаичных изображений или заголовков элементов мозаичных изображений такого кодового потока. Сервер, по своему усмотрению, может предпочесть возвращать сообщения главного заголовка или бункера метаданных, что были бы возвращены в отклике на запрос, который опустил поле запроса Размер кадра.

C.4.5 Компоненты (comps)

```
comps = "comps" "=" 1#UINT-RANGE
```

Это поле используется для определения компонентов изображения, которые должны быть включены в запрашиваемое окно обзора; если оно не присутствует, то запрос, как понимают, должен включать в себя все имеющиеся компоненты изображения всех кодовых потоков, определенных через поле запроса Кодового потока [*Codestream*], и все уместные компоненты всех кодовых потоков, запрашиваемые через поле запроса Контекста кодового потока (см. C.4.7). Этими "уместными" компонентами являются те компоненты, которые вовлечены в воспроизведение объектов изображения (например, слои наложения изображений JPX или видеодорожки MJ2), которые определяются через поле запроса Кодового потока.

Значения в этом поле запроса представляют собой индексы рассматриваемых компонентов изображений. Индексы компонентов изображений начинаются от 0 и имеют истолкование, назначенное им синтаксисом кодового потока JPEG 2000, как описывается в Рекомендации МСЭ-Т T.800 | ИСО/МЭК 15444-1; но отметим, что они являются компонентами, которые получаются путем декодирования и инверсного преобразования элементарных волн в сжатых данных, до применения инверсного преобразования компонента RCT или ICT. Для кодовых потоков, соответствующих Рекомендации МСЭ-Т T.801 | ИСО/МЭК 15444-2, указанные здесь компоненты, являются компонентами, которые определены как "пространственные компоненты", т. е. как те, что получены путем декодирования и инверсного преобразования элементарных волн сжатых данных, до применения любого инверсного многокомпонентного преобразования, преобразования компонента зависимости или многокомпонентного преобразования элементарных волн.

Несуществующие компоненты в любых запрашиваемых кодовых потоках должны быть сброшены.

C.4.6 Кодовый поток (поток)

```
stream = "stream" "=" 1#sampled-range

sampled-range = UINT-RANGE [":" sampling-factor]

sampling-factor = UINT
```

Это поле используется для обозначения, какой кодовый поток или кодовые потоки принадлежат к запрашиваемому окну обзора. Если поле опускается и кодовый поток (потоки) не может быть определен другими способами, по умолчанию будет одиночный кодовый поток с идентификатором 0. Отметим, что поле запроса Контекста кодового потока (см. C.4.7) предоставляет дополнительные средства для запрашиваемых кодовых потоков.

Для целей семейства JPEG 2000, индексами кодового потока являются те индексы, которые вложены в соответствующий блок Указателя места заполнения [*Placeholder*], что появляется внутри соответствующего бункера метаданных, как описано в разделе A.3.6. Для файловых форматов, которые подразумевали идентификаторы кодовых потоков, тем идентификатором следует соглашаться с индексами, используемыми здесь.

Там, где диапазон кодовых потоков обозначается, отсутствие верхней границы означает, что диапазон простирается на все кодовые потоки с большими идентификаторами. Там, где верхняя граница обеспечивается, верхняя граница предоставляет абсолютный идентификатор последнего кодового потока в диапазоне.

Обеспечивается ли верхняя граница, или нет, диапазон кодового потока может быть оценен с помощью дополнительного параметра `sampling-factor` [*дискретизация-коэффициент*]. Параметр `sampling-factor`, если обеспечивается, должен быть строго положительным целым числом, *F*. Диапазон тогда включает все идентификаторы кодового потока *L+Fk*, которые лежат в пределах неограниченного диапазона, где *L* является идентификатором первого кодового потока в диапазоне. Индекс клиента рассматриваемых кодовых потоков есть *k* и *k* есть UINT.

C.4.7 Контекст кодового потока (контекст)

```
context = "context" "=" 1#context-range

context-range = jpxl-context-range / mj2t-context / reserved-context

jpxl-context-range = "jpxl" "<" jpx-layers ">" [ "[" jpxl-geometry "]" ]

jpx-layers = sampled-range

jpxl-geometry = "s" jpx-iset "i" jpx-inum

jpx-iset = UINT

jpx-inum = UINT

mj2t-context = "mj2t" "<" mj2-track ">" [ "[" mj2t-geometry "]" ]

mj2-track = NONZERO ["+" "now" ]

mj2t-geometry = "track" / "movie"

reserved-context = 1*( TOKEN / "<" / ">" / "[" / "]" / "-" / ":" / "+" )
```

Это поле может быть использовано для запроса кодовых потоков косвенно через объекты изображения "верхнего уровня". Эта Рекомендация | Международный стандарт определяет контексты, соответствующие слоям наложения изображений JPX (слой наложения изображений JPX может содержать один или более кодовых потоков) и видеодорожкам MJ2; однако механизм разрабатывается для расширяемости.

Если поле запроса Контекста кодового потока поставляется, то запрашиваемое окно обзора включает в себя каждый из кодовых потоков, которые связаны с запрашиваемым контекстом (контекстами), в дополнение к любым кодовым потокам, запрашиваемым через поле запроса Кодового потока.

Тело поля запроса Контекста кодового потока состоит из одного из значений `context-range` [*конекст−диапазон*]. Каждое значение `context-range` связывается с набором кодовых потоков, которые могут быть определены сервером. Значение `context-range` может также обозначать преобразования повторного отображения координат, которые должны быть применены к параметрам Размера кадра, Размера области и Смещения, чтобы определять разрешающую способность изображения кодового потока и область изображения кодового потока для каждого из кодовых потоков, связанных со значением `context-range`. Там, где сервер подготовлен для обработки значения `context-range`, он должен распознавать кодовые потоки, которые связаны с таким значением `context-range`, с помощью заголовка отклика Контекста кодового потока.

Эта Рекомендация | Международный стандарт определяет два особых типа `context-range`, которые предназначены для обращения к потребностям файловых форматов JPX и MJ2. Первый из этих типов `context-range,jpxl-context-range`, используется для распознавания одного или более слоев наложенных изображений JPX. Индексы слоев наложения изображений, связанных с `jpxl-context-range,` поставляются в форме `sampled-range` [*дискретизированный−диапазон*], следующий той же самой семантике, как и диапазоны подвергнутых дискретизации кодовых потоков в поле запроса Кодового потока. Там, где `jpxl-context-range` обрабатывается сервером, кодовые потоки, принадлежащие соответствующему слою (слоям) наложения изображений, должны распознаваться внутри заголовка отклика Контекста кодового потока.

Параметр `jpxl-context-range` может обозначать дополнительное преобразование повторного отображения координат, подлежащее использованию в заключении разрешающей способности изображения кодового потока и области изображения кодового потока для каждого из своих кодовых потоков. Это преобразование повторного отображения координат определяется с помощью двух неотрицательных целых чисел, `jpx-iset` и `jpx-inum`. Вместе эти два целых числа обозначают особую инструкцию наложения изображений внутри блока JPX Composition (comp) [*наложение изображений*], найденную внутри сферы применения логической цели. Конкретная рассматриваемая инструкция располагается в блоке набора

инструкций (iset), чья порядковая позиция (начинающаяся от 0) внутри блока наложения изображений дается значением `jpx-iset`. Значение `jpx-inum` дает порядковую позицию (начинающуюся от 0) для инструкции внутри блока набора инструкций. Истолкование этих индексов не зависит от подсчетов повторений, которые могут появляться внутри блока наложения изображений JPX.

Когда значения `jpx-iset` и `jpx-inum` обрабатываются сервером, размер запрашиваемого кадра и параметры областей `fx`, `fy`, `sx`, `sy`, `ox` и `oy`, первоначально должны быть преобразованы в модифицированный размер кадра и параметры областей `fx"`, `fy"`, `sx"`, `sy"`, `ox"` и `oy"`, используя выражения в Уравнении C-3. Эти модифицированные параметры областей должны быть вычислены отдельно для каждого запрашиваемого кодового потока и затем должны быть использованы вместо `fx`, `fy`, `sx`, `sy`, `ox` и `oy` , когда определяются разрешающая способность изображения кодового потока и область изображения кодового потока, следуя процедуре, описанной в разделе C.4.1.

$$
\begin{aligned}
\mathtt{fx"} &= \left\lceil \mathtt{fx} \cdot \frac{\mathtt{XR}_{reg}}{\mathtt{XS}_{reg}} \cdot \frac{\mathtt{Wt}_{inst}}{\mathtt{Ws}_{inst}} \cdot \frac{\mathtt{W}_{cod}}{\mathtt{W}_{comp}} \right\rceil ; \quad \mathtt{fy"} = \left\lceil \mathtt{fy} \cdot \frac{\mathtt{YR}_{reg}}{\mathtt{YS}_{reg}} \cdot \frac{\mathtt{Ht}_{inst}}{\mathtt{Hs}_{inst}} \cdot \frac{\mathtt{H}_{cod}}{\mathtt{H}_{comp}} \right\rceil \\
\mathtt{sx"} &= \min\{(\mathtt{ox}+\mathtt{sx}), x_{lim}\} - \max\{\mathtt{ox}, x_{min}\} \\
\mathtt{sy"} &= \min\{(\mathtt{oy}+\mathtt{sy}), y_{lim}\} - \max\{\mathtt{oy}, y_{min}\} \\
\mathtt{ox"} &= \max\{\mathtt{ox}, x_{min}\} - \left\lceil \left(\mathtt{XO}_{inst} - \left(\mathtt{XC}_{inst} - \frac{\mathtt{XO}_{reg}}{\mathtt{XS}_{reg}} \right) \cdot \frac{\mathtt{Wt}_{inst}}{\mathtt{Ws}_{inst}} \right) \cdot \frac{\mathtt{fx}}{\mathtt{W}_{comp}} \right\rceil \\
\mathtt{oy"} &= \max\{\mathtt{oy}, y_{min}\} - \left\lceil \left(\mathtt{YO}_{inst} - \left(\mathtt{YC}_{inst} - \frac{\mathtt{YO}_{reg}}{\mathtt{YS}_{reg}} \right) \cdot \frac{\mathtt{Ht}_{inst}}{\mathtt{Hs}_{inst}} \right) \cdot \frac{\mathtt{fy}}{\mathtt{H}_{comp}} \right\rceil \\
x_{min} &= \left\lfloor \mathtt{XO}_{inst} \cdot \frac{\mathtt{fx}}{\mathtt{W}_{comp}} \right\rfloor ; \quad y_{min} = \left\lfloor \mathtt{YO}_{inst} \cdot \frac{\mathtt{fy}}{\mathtt{H}_{comp}} \right\rfloor \\
x_{lim} &= \left\lceil (\mathtt{XO}_{inst} + \mathtt{Wt}_{inst}) \cdot \frac{\mathtt{fx}}{\mathtt{W}_{comp}} \right\rceil ; \quad y_{lim} = \left\lceil (\mathtt{YO}_{inst} + \mathtt{Ht}_{inst}) \cdot \frac{\mathtt{fy}}{\mathtt{H}_{comp}} \right\rceil
\end{aligned}
\tag{C-3}
$$

Отметим, что модифицированная область окна обзора, определяемая с помощью `sx"`, `sy"`, `ox"` и `oy"`, может лежать слегка слева или выше начала. То есть, `ox"` и/или `oy"` могут быть отрицательными. При определении области изображения кодового потока любую часть области окна обзора, которая лежит слева или выше начала, следует игнорировать, следуя процедуре, описанной в разделе C.4.1.

Если значения `jpx-iset` и `jpx-inum` не поставляются, то модифицированные параметры областей, которые должны использоваться взамен `fx`, `fy`, `sx`, `sy`, `ox` и `oy`, даются выражениями в Уравнении C-4. Как и ранее, эти модифицированные параметры должны использоваться при определении разрешающей способности изображения кодового потока, следуя процедуре в разделе C.4.1.

$$
\begin{aligned}
\mathtt{fx"} &= \left\lceil \mathtt{fx} \cdot \frac{\mathtt{XR}_{reg}}{\mathtt{XS}_{reg}} \cdot \frac{\mathtt{W}_{cod}}{\mathtt{W}_{reg}} \right\rceil ; \quad \mathtt{fy"} = \left\lceil \mathtt{fy} \cdot \frac{\mathtt{YR}_{reg}}{\mathtt{YS}_{reg}} \cdot \frac{\mathtt{H}_{cod}}{\mathtt{H}_{reg}} \right\rceil \\
\mathtt{ox"} &= \mathtt{ox} - \left\lceil \frac{\mathtt{XO}_{reg}}{\mathtt{XS}_{reg}} \cdot \frac{\mathtt{fx}}{\mathtt{W}_{reg}} \right\rceil ; \quad \mathtt{oy"} = \mathtt{oy} - \left\lceil \frac{\mathtt{YO}_{reg}}{\mathtt{YS}_{reg}} \cdot \frac{\mathtt{fy}}{\mathtt{H}_{reg}} \right\rceil \\
\mathtt{sx"} &= \mathtt{sx}; \quad \mathtt{sy"} = \mathtt{sy}
\end{aligned}
\tag{C-4}
$$

Второй тип `context-range`, описываемый этой Рекомендацией | Международным стандартом, `mj2t-context`, позволяет клиентам запрашивать характерные дорожки из файла MJ2. Идентификатор `mj2-track` должен быть в точности положительным целым числом, поскольку 1 является наименьшим дозволенным идентификатором дорожки, разрешенным внутри файла MJ2. Если идентификатор `mj2-track` включает в себя дополнительный суффикс "+now" [*сейчас*], то `mj2t-context` состоит из всех кодовых потоков, принадлежащих видеодорожке MJ2, начинающихся с кодового потока, чье время захвата соответствует времени, в которое получают запрос. Это является полезным, когда источником является поток "живого" видео. В противном случае, сервер может связывать суффикс " now" с любым кодовым потоком, который видится ему подходящим. Если суффикс "+now" не включается, то `mj2-context` состоит из всех кодовых потоков, принадлежащих видеодорожке MJ2.

Тип `mj2t-context` может точно определять преобразование повторного отображения координат, которое должно использоваться в заключении разрешающей способности изображения кодового потока и областей изображений кодовых потоков для каждого из своих кодовых потоков. Если он не присутствует, то параметры размера изображений и областей, поставляемые через поля запросов Размера кадра, Смещения и Размера области, должны истолковываться непосредственно, следуя процедуре, очерченной в разделе C.4.1. В противном случае, запрашивается один из двух типов преобразования координаты, как обозначается появлением одного из маркеров "дорожка" [*"track"*] или "фильм" [*"movie"*].

Там, где определяется" дорожка", поля запросов Размера кадра, Смещения и Размера области используются для обозначения желательного размера представления и желательной прямоугольной области внутри наименьшего ограничивающего прямоугольника, который содержит представление дорожки, в этом желательном размере представления. Геометрические преобразования, описываемые блоком Заголовка дорожки MJ2 [*Track Header*] (tkhd), должны быть применены для определения соответствующей разрешающей способности и области на каждом кодовом потоке, связанном с дорожкой.

Там, где определяется "фильм", поля запросов Размера кадра, Смещения и Размера области используются для обозначения желательного размера для полного (возможно, наложенного) воспроизводимого фильма и желательной прямоугольной области внутри наименьшего ограничительного прямоугольника, который содержит фильм, в этом желательном размере. Геометрические преобразования, описанные блоком Заголовка дорожки MJ2 (tkhd) должны быть объединены с геометрическими преобразованиями, описанными блоком Заголовка фильма [*Movie Header*] (mvhd) и примененными для определения соответствующей разрешающей способности изображения и области на каждом кодовом потоке, связанном с дорожкой.

В случае, когда сервер не способен применить любое из геометрических преобразований `mj2t-context`, описанных выше, он предоставляет модифицированную строку `mj2t-context` в своем заголовке отклика Контекста кодового потока.

ПРИМЕЧАНИЕ 1. − Использование поля запроса Контекста кодового потока вместе с полем запроса Кодового потока может иметь своим результатом кодовый поток, запрашиваемый многократно с различными геометрическими преобразованиями полей запросов Размера кадра, Размера области и Смещения. Там, где это случается, фактически запрашиваются многократные непересекающиеся или перекрывающиеся порции изображений такого кодового потока.

ПРИМЕЧАНИЕ 2 . − Выражения в Уравнении C-4 могут быть эквивалентным образом получены путем установки $XS_{comp}=Ws_{inst}=Wt_{inst}=W_{reg}$, $YS_{comp}=Hs_{inst}=Ht_{inst}=H_{reg}$ и $XO_{inst}=YO_{inst}$ в Уравнение C-3, когда пределы на `sx"`, `sy"`, `ox"` и `oy"` не ограничиваются с помощью $x_{\lim}, x_{\min}, y_{\lim}, y_{\min}$.

ПРИМЕР 1: "context=jpxl<0-4:2>[s5i2]"
В этом случае сервер просят возвращать кодовые потоки, которые используются слоями наложения изображений JPX 0, 2 и 4, повторно отображая требуемый размер кадра и область изображения согласно геометрическим подстройкам, представленным третьей инструкцией шестого блока набора инструкций внутри блока наложения изображений (файлы JPX имеют, самое большее, один блок наложения изображений).

ПРИМЕР 2: "stream=0&context=mj2t<1+now>[track]"
В этом случае сервер просят возвращать кодовый поток 0, а также все кодовые потоки, принадлежащие первой дорожке файла MJ2, начиная от кодового потока, чье время дискретизации соответствует текущему времени. Кроме того, сервер просят повторно отобразить требуемый размер кадра и область изображения согласно геометрическим подстройкам, описанным в блоке Заголовка дорожки, игнорируя любые дополнительные геометрические подстройки, которые могут быть описаны в блоке Заголовка фильма [*Movie Header*].

C.4.8 Скорость дискретизации (srate)

```
srate = "srate" "=" streams-per-second

streams-per-second = UFLOAT
```

Если это поле поставляется, то кодовые потоки, которые принадлежат окну обзора, получаются путем дискретизации на пониженной скорости тех потоков, которые упомянуты полем запроса Кодового потока, в дополнение к потокам, расширенным от значений диапазона контекста в поле запроса Контекста кодового потока (см. C.4.7), чтобы достигнуть средней скорости дискретизации, которая не больше, чем значение потоков в секунду. Это возможно только в том случае, если кодовые потоки имеют связанную информацию синхронизации (например, если они принадлежат логической цели, соответствующей файловому формату MJ2).

Это поле запроса служит только для определения, какие кодовые потоки следует считать принадлежащими окну обзора. Сервер должен просмотреть все кодовые потоки, которые были бы иначе включены в окно обзора, сбрасывая кодовые потоки, как требуется, чтобы гарантировать, что среднее разделение между временами источников кодовых потоков составляет не меньше, чем обратная величина от значения потоков в секунду. Эта Рекомендация | Международный стандарт не предписывает алгоритм для дискретизации на пониженной скорости или точное истолкование для термина "среднее разделение".

Если никакой источник синхронизации не имеется, то окно обзора будет состоять из всех кодовых потоков, опознанных с помощью поля запроса Кодового потока и поля запроса Контекста кодового потока; но это поле запроса может, тем не менее, затронуть истолкование поля запроса Скорости доставки [*Delivery Rate*], если оно присутствует.

C.4.9 ROI (roi)

```
roi = "roi" "=" region-name

region-name = 1*(DIGIT / ALPHA / "_")
            / "dynamic"
```

Это поле определяет желательную пространственную область изображения через имя, а не через координаты. Преобразование между `region-name` [*область-имя*] и определенной пространственной областью изображения может поступать от нескольких мест; оно может быть определено внутри блока описания ROI в пределах логической цели или оно может быть определено внутри реализации самого сервера.

Значение `region-name` "динамическое" (*dynamic ROI*) резервируется, чтобы представлять непостоянную область внутри изображения, которая преобразовывается в пространственную область независимо для всех запросов без разбора. Сервер может использовать любую информацию о клиенте и любые другие параметры запроса, когда он определяет, какую пространственную область он будет обеспечивать для такого конкретного запроса. Например, если сервер знает, что физическое устройство отображения у клиента является очень маленьким, то он может выбрать обеспечение только области переднего плана изображения на более высокой разрешающей способности, а не полную область изображения на более низкой разрешающей способности. Серверы не обязаны поддерживать динамические ROI.

Если поле ROI существует, и сервер знает, как обращаться с запросом ROI, то поле ROI имеет приоритет по отношению к полю запроса Смещения и к полям Размера области, которые сервер должен игнорировать. Если поле ROI существует, но по какой-нибудь причине сервер не знает, как с ним обращаться, то сервер должен игнорировать поле ROI и использовать поля Размера области и Смещения. Если эти поля опускаются, то должны использоваться значения этих полей по умолчанию.

Если клиент определяет Размер кадра, а также ROI, и сервер понимает указанное ROI, то значение поля запроса Размера кадра определяет разрешающую способность изображения, в которой запрашивается ROI.

C.4.10 Слои (слои)

```
layers = "layers" "=" UINT
```

Это поле может использоваться для ограничения количества слоев качества кодовых потоков, которые принадлежат запросу окна обзора. По умолчанию, представляют интерес все имеющиеся слои. Значение определяет количество начальных слоев качества, которые представляют интерес. Серверу не следует пытаться увеличивать любые бункеры данных зоны вне уместной границы слоя. Серверу не следует пытаться увеличивать любые бункеры данных элементов мозаичного изображения вне точки, в которой все остающееся содержимое лежит вне уместной границы слоя. Из-за порядка данных внутри элемента мозаичного изображения, для сервера может быть необходимым возвращать данные вне границы запрашиваемого слоя только для JPT-потока.

C.5 Поля запросов метаданных

C.5.1 Метаданные, запрашиваемые неявно с запросами окон обзора

Поле запроса Кодового потока и поле запроса Контекста кодового потока обозначают один или более кодовых потоков, которые связаны с запрашиваемым окном обзора. Даже если ни одно из полей запросов не присутствует, окно обзора связывается, по крайней мере, с одним кодовым потоком, как упомянуто в разделе C.4.6. Более того, как отмечено в разделе C.4.2, даже если поле запроса Размера кадра опускается, запрашиваемое окно обзора включает в себя, по крайней мере, главный заголовок кодового потока для каждого запрашиваемого кодового потока. Единственным исключением для этого является случай, когда `metadata-only` [*метаданные–только*] определяется в поле запроса Метаданных (см. C.5.2). Кроме этого случая, клиент также неявно запрашивает, чтобы любые блоки метаданных могли быть затребованы из файлового формата, если такой имеется, чтобы использовать совокупность изображений, представленную запрашиваемыми кодовыми потоками. Чтобы гарантировать способность к взаимодействию между компонентами клиента и сервера, этот подраздел обозначает минимальный набор метаданных, который серверы должны расценить как неявно запрашиваемый, наряду с окном обзора. Там, где сервер знает об уместных дополнительных элементах метаданных, он может доставлять их тоже.

Для файлов JP2 и JPX должны быть рассмотрены следующие элементы метаданных, которые подлежат запросу вместе с окном обзора:

a) Полное содержимое бункера метаданных 0.

b) Полное содержимое каждого из следующих блоков, где бы они не находились на верхнем уровне файла:

   1) Подпись JP2 ("jP ");
   2) Тип файла ("ftyp");
   3) Требования программ чтения ("rreq");
   4) Наложение изображений ("comp").

c) Все непосредственные заголовки суперблоков от каждого из следующих суперблоков:

   1) любой блок Заголовка JP2 ("jp2h")
   2) любой блок Заголовка кодового потока ("jpch"), связанный с запрашиваемым кодовым потоком;
   3) любой блок Заголовка слоя наложения изображений ("jplh"), связанный со слоем наложения изображений JPX, запрашиваемым через поле запроса Контекста кодового потока.

d) Полное содержание каждого из следующих блоков, где бы эти блоки не находились внутри одного из вышеуказанных суперблоков:

   1) Заголовок изображения ("ihdr");
   2) Биты на каждый Компонент ("bpcc");
   3) Глубина цвета ("pclr");
   4) Преобразование компонента ("cmap");
   5) Определение канала ("cdef");
   6) Разрешающая способность ("res ");
   7) Регистрация кодового потока ("creg");
   8) Непрозрачность ("opct").

e) Для файлов JP2, совместимых файлов JP2 и файлов JPX, один или более блоков Описания цветового пространства ("colr"), связанных с каждым кодовым потоком или слоем наложения JPX, запрошенными через поле запроса Контекста кодового потока следующим образом:

   1) Если сервер способен точно определять, какой блок предпочитается, то сервер должен послать только такой блок, даже если это означает отказ от отправки первого блока для совместимых файлов JP2 или JP2 (например, если второй блок есть Любой ICC, и предпочтения цветового пространства определяют, что клиент предпочитает Любой ICC). Если сервер не способен точно определять, какой блок предпочитается, он должен послать полный блок Описания цветового пространства.
   2) Для всех блоков, которые не посылаются, сервер должен отсылать порцию содержимого блока, чтобы клиент мог определить, хочет ли он позже запросить другую спецификацию цветового пространства.
      - Для нумерованных блоков, сервер должен послать, по крайней мере, первые 7 байтов из содержимого блока (по меньшей мере, вплоть до поля EnumCS).
      - Для блоков цветового пространства, определяемых поставщиком, сервер должен послать, по меньшей мере, первые 19 байтов из содержимого блока (по меньшей мере, вплоть до поля VCLR).
      - Для Ограниченных и Любых ICC блоков цветового пространства сервер должен послать, по меньшей мере, первые 3 байта из содержимого блока (по меньшей мере, поля METH, APPROX и PREC).

Сервер просят возвращать начальный префикс каждого бункера метаданных, который содержит любые из метаданных, упомянутых выше, простираясь от первого байта бункера метаданных и продолжаясь до конца всех запрашиваемых метаданных из того бункера метаданных. В результате, фактическое количество метаданных, возвращаемых сервером, может зависеть от конкретного способа, которым логическая цель была разделена в бункерах метаданных. Обсуждение этих проблем можно найти в подразделе A.3.6.2.

Эта Рекомендация | Международный стандарт не дает совета относительно того, что составляет неявные метаданные MJ2 для запросов окон обзора, однако это может быть определено в будущем стандарте.

C.5.2 Запрос метаданных (metareq)

```
metareq = "metareq" "=" 1#("[" 1$(req-box-prop) "]" [root-bin] [max-depth])
                        [metadata-only]

req-box-prop = box-type [limit] [metareq-qualifier] [priority]

limit = ":" (UINT / "r")

metareq-qualifier = "/" 1*("w" / "s" / "g" / "a")

priority = "!"

root-bin = "R" UINT

max-depth = "D" UINT

metadata-only = "!!"
```

Это поле определяет, какие метаданные нужны в отклике на этот запрос, в дополнение к любым метаданным, требуемым для клиента, чтобы декодировать или истолковывать запрашиваемые данные изображения (см. C.5.1). Строка значения в этом поле запроса является перечнем независимых запросов; однако сервер может обрабатывать запросы как группу, и между запросами может быть перекрытие.

Каждый запрос относится к бункеру данных, определяемому его значением `root-bin`. [*корень−бункер*]. Если значение `root-bin` не указывается, то корнем является бункер метаданных 0. Запрос принадлежит только к данным, находящимся внутри или упоминаемого (через блоки Указателей мест заполнения) такого конкретного бункера данных.

Если указывается значение для `max-depth` [*максимальная−глубина*], то тогда запрашиваются только блоки, содержащиеся внутри корневого бункера метаданных, и такие, где не больше, чем `max-depth` уровней в файловой иерархии ниже такого блока. Если значение для `max-depth` не определяется, то не существует предела по глубине файловой иерархии для этого запроса.

Порция `req-box-prop` запроса определяет перечень типов блоков, которые представляют интерес для клиента. Специальная строка "*" может быть заменена для типа блока, в этом случае предполагаются все типы блоков. Каждый тип блока (или "*") может сопровождаться любым сочетанием трех атрибутов: предельное значение, определитель metareq и флаг приоритета.

Атрибут `limit` [*предел*] определяет, какой тип информации и сколько содержимого из блока клиент запрашивает для такого типа блока. Предельный параметр принимает форму двоеточия, сопровождаемого значением (предельным значением), которое должно быть либо незнаковым целым числом, либо знаком "r".

Если предельное значение является целым числом *n*, которое больше нуля, то сервер просят возвращать только первые *n* байтов содержимого соответствующих блоков такого типа блока, в дополнение к заголовкам блоков. Если предельное значение равно 0, то запрашиваются только заголовки блоков такого типа. Если `limit` не определяется, тогда клиент запрашивает полное содержимое всех блоков такого типа блока, которое соответствует другим аспектам запроса, независимо от того, являются ли блоки такого типа суперблоками или нет. Также, в случае числового или не указанного предельного значения на суперблоке, сервер просят обеспечивать количество данных, запрашиваемых атрибутом `limit`, независимо от того, действительно ли иерархия, содержащаяся внутри такого суперблока, более глубока, или нет, чем была бы достигнута на основе значений `root-bin` и `max-depth`, и независимо от типов блока субблоков, найденных внутри суперблока.

Если предельное значение есть "r", тогда сервер просят посылать заголовок блока, а не содержимое блока, для любого блока с указанным типом блока, а также для всех его понижающихся субблоков (независимо от их типа блока), вниз до максимальной глубины, указанной в запросе. Это в действительности является запросом о каркасе такой порции иерархии блока. Если сервер не способен определять, действительно ли блок является суперблоком, или нет, он мог бы не обладать способностью возвращения в блоки субблока, поэтому он мог бы не откликаться полностью на некоторые запросы метаданных. Серверам следует обладать способностью распознавать состояние суперблока всех блоков, определенных файловыми форматами, которые они намерены поддерживать.

В то время как предельное значение предела "r" означает, что клиент запрашивает каркас структуры блока, состоящий из заголовков блока, разделение логической цели в бункерах метаданных может заставить сервер возвращать дополнительные данные, включая содержимое некоторых блоков и заголовков и/или содержимое других не запрашиваемых блоков. Это потому, что сервер просят возвращать все байты от начала каждого бункера метаданных, который содержит запрашиваемые байты блока вплоть до последнего байта запрашиваемого блока.

Атрибут `metareq-qualifier` [*определитель*] принимает форму "/", он сопровождается одним или более флагами "g", "s", "w" и "a". Каждый флаг обозначает контекст, из которого могут быть извлечены блоки,

соответствующие запросу. Истолкование для каждого из этих контекстов дается в таблице C.2. Если предоставляется более одного флага, то должен быть взят союз соответствующего контекста. Если `metareq-qualifier` не предоставляется, то должен использоваться союз контекстов "g", "s" и "w". Путем видоизменения, отметим, что контексты "g", "s" и "w" являются взаимно исключающими, но их союз обычно меньше, чем контекст "a" вместилища.

Если определяется флаг `priority` [*приоритет*], тогда клиент запрашивает те блоки типа `box-type` [*блок−тип*], которые соответствует другим элементам запроса, подлежащим доставке, с более высоким приоритетом, чем данные изображения.

Для любого типа блока, не определенного в перечне `req-box-prop`, никакие данные для блоков такого типа не запрашиваются.

Если `metadata-only` [*метаданные−только*] определяется в конце поля запроса метаданных, клиент запрашивает, чтобы отклик сервера состоял только из метаданных, без каких-либо данных изображения или заголовков кодовых потоков, независимо от того, были ли использованы такие поля запросов, как Размер кадра. Для типов возврата JPP-потока и JPT-потока это означает, что возвращаемые сообщения JPIP все будут сообщениями бункеров метаданных.

ПРИМЕР 1: "metareq=[*]R31D4"

В этом случае, сервер просят возвращать полное содержимое всех блоков, которые он находит в содержимом бункера 31. Пока было определено ограничение на желательную глубину, сервер должен игнорировать такое ограничение, так как содержимое тех блоков не было ограничено через параметр `limit`.

ПРИМЕР 2: "metareq=[*:r,drep]R31D4"

Параметр "*:r" означает, что от сервера запросили возврат заголовков блоков для всех блоков, содержавшихся в бункере метаданных 31, и в любых бункерах, упоминаемых указателями мест заполнения, которые содержались внутри такого бункера, вплоть до глубины 4 уровней от содержимого бункера 31, но не включая содержимое тех блоков. Дополнительный "drep" `req-box-prop` определяет, что сервер запрашивают возвращать полное содержимое любого блока "drep", что содержится внутри бункера метаданных 31, и любых бункеров, на которые ссылаются указатели мест заполнения внутри такого бункера, вплоть до глубины 4 уровней от содержимого бункера 31.

ПРИМЕР 3: "metareq=[drep]R31D4"

В этом случае, сервер все еще просят возвращать полное содержимое любых блоков "drep", которые он находит в содержимом бункера 31 или в любых бункерах, упоминаемых таким бункером, вплоть до глубины четырех уровней от содержимого бункера 31. Однако, потому что никакие другие блоки не были определены, сервер просят посылать только такое количество других данных, какое необходимо для определения позиции любого блока "drep" в файловой иерархии относительно блока, содержавшегося внутри бункера метаданных 31.

Независимо от спецификаций блоков, обеспечиваемых через поле Запроса метаданных, сервер может посылать другие данные, или потому что он решил, что для клиента требуются другие данные, чтобы декодировать или истолковывать запрашиваемые данные изображения, или потому что сервер предварительно разделил логическую цель на бункеры данных, используя различные критерии, и дополнительные данные должны быть посланы, чтобы обеспечивать непротиворечивый и значащий обзор бункеров метаданных для этой логической цели.

Таблица C.2 – Флаги определителей запросов метаданных

Флаг	Истолкование
"w"	Этот контекст metareq включает в себя блоки, которые известны как связанные с конкретной пространственной областью изображения внутри одного или более кодовых потоков, и которые принадлежат окну обзора, где пространственная область, разрешающие способности и компоненты изображений, к которым принадлежит окно обозрения, пересекаются с такими атрибутами окна обозрения. Такая ассоциация могла бы, например, быть установлена с помощью блока "asoc" в файле JPX.
"s"	Этот контекст metareq включает в себя все блоки, которые известны как связанные с одним или более кодовыми потоками, которые принадлежат к окну обзора, или с одном или более запрашиваемыми контекстами кодовых потоков (например, слои наложения изображений JPX или видеодорожки MJ2), где эти блоки не являются связанными исключительно с конкретными пространственными областями. Такая ассоциация, например, могла бы быть установлена с помощью блока "asoc" в файле JPX.
"g"	Этот контекст metareq включает в себя все блоки, которые относятся к запрашиваемому окну обзора, принимая во внимание запрашиваемые кодовые потоки и контексты запрашиваемых кодовых потоков, исключая те блоки, которые включены в контексты metareq "w" и "s".
"a"	Этот контекст metareq включает в себя все блоки в логической цели, без исключения (Примечание).
ПРИМЕЧАНИЕ. − Этот контекст metareq годится для запросов, которые желают исследовать файловую структуру независимо от окна.	

C.6 Поля запросов, ограничивающие данные

C.6.1 Максимальная длина отклика (len)

```
len = "len" "=" UINT
```

Это поле определяет ограничение на количество данных, которые клиент хочет, чтобы сервер посылал в отклике на этот запрос. Единицей должны быть байты. Если это поле не присутствует, сервер должен посылать данные изображения к клиенту до такой точки, пока все из уместных данных не были посланы, не достигается предел качества (см. C.6.2), или отклик не прерывается прибытием нового запроса, который не включает поле запроса Ожидания со значением "да" (см. C.7.2). Клиенту следует использовать `len=0`, если он не требует никаких данных заголовков откликов и отклика.

C.6.2 Качество (quality)

```
quality = "quality" "=" (1*2DIGIT / "100")        ; 0 to 100
```

Это поле может использоваться для ограничения передачи данных до уровня качества (между 0 для самого низкого качества и 100 для самого высокого качества), связанного с изображением. Пределы качества трудно сформулировать надежным образом, и сервер может игнорировать этот запрос, отвечая с помощью значения"–1" (см. D.2.16). Тем не менее, полезно позволить клиенту обеспечивать некоторый признак максимального качества изображения, которое могло бы представлять интерес. Коэффициент качества может пытаться аппроксимировать специальное для данного случая Качество, обычно используемое для управления сжатием JPEG. Клиенту следует ожидать, что размер возвращаемых данных монотонно не уменьшается с увеличением качества, т. е. увеличение значения качества в общем случае соответствует увеличению размера возвращаемых данных.

ПРИМЕЧАНИЕ. – Если сервер поддерживает этот запрос, и два различных клиента осуществляют одинаковые запросы к одной и той же логической цели, имеющей то же самое значение качества, например, `quality=80`", серверу следует иметь непротиворечивую политику осуществления при возврате данных из бункеров данных.

C.7 Поля запросов для управления сервером

C.7.1 Выравнивание (align)

```
align = "align" "=" ("yes" / "no")
```

Это поле определяет, должны ли данные откликов сервера быть выровнены на естественных границах. Значение по умолчанию равно "нет" [*no*]. Если значение равно "да" [*yes*], то любое доставленное в отклике на этот запрос сообщение JPT-потока или JPP-потока, которое пересекает любую "естественную границу", должно заканчиваться в любой последующей "естественной границе". Естественные границы для каждого типа бункера данных перечисляются в таблице C.3. Сообщение, как говорят, пересекает естественную границу, если оно включает в себя последний байт до границы и первый байт после границы. Например, бункер данных зоны пересекает естественную границу, если он включает в себя последний байт одного пакета и первый байт следующего пакета. Осторожно отметим, что от выровненных сообщений откликов не обязательно требуется завершение на естественной границе, если они не пересекают границу. Это означает, например, что отклик может включать в себя частичные пакеты из зон, которые могут быть необходимы, если преобладающий предел байта предотвращает поставку полных пакетов.

Таблица C.3 – Границы выравнивания, основанные на типе бункера

Тип бункера	Естественная граница
Бункер данных зоны	Конец пакета (одна граница для каждого слоя качества)
Бункер данных элементов мозаичного изображения	Конец части элемента мозаичного изображения (одна граница для каждой части элемента мозаичного изображения)
Бункер данных заголовка элемента мозаичного изображения	Конец бункера (только одна граница)
Бункер данных главного заголовка	Конец бункера (только одна граница)
Бункер метаданных	Конец блока на верхнем уровне бункера данных (одна граница для каждого блока)

C.7.2 Ожидание (wait)

```
wait = "wait" "=" ("yes" / "no")
```

Это поле используется для указания, должен ли сервер закончить отклик на предыдущий запрос. Если значение равно "да", то сервер должен полностью ответить на предыдущий запрос на том же самом ресурсе канала, определенном через поле идентификатора ID Канала перед началом отклика на этот запрос

Если значение этого поля равно "нет", то сервер может изящно завершить обработку любого предыдущего запроса на том же самом ресурсе канала (указанном через поле идентификатора ID Канала) до окончания и может начать отвечать на этот новый запрос. В этом контексте, "изящное завершение" подразумевает, что сервер должен, по крайней мере, закончить текущее сообщение.

Значение по умолчанию этого поля равно "нет".

C.7.3 Тип Возврата изображения (type)

```
type = "type" "=" 1#image-return-type
image-return-type = media-type / reserved-image-return-type
media-type = TOKEN "/" TOKEN *( ";" parameter )
reserved-image-return-type = TOKEN *( ";" parameter )
parameter = attribute "=" value
attribute = TOKEN
value = TOKEN
```

Это поле используется для указания типа (или типов) запрашиваемых данных отклика. Сервер, не желающий предоставлять какой-либо из запрашиваемых типов возврата, должен выпустить отклик ошибки.

Значение поля запроса Типа возврата изображения должно быть либо типом носителя информации (определенным в документе RFC 2046), либо одним из резервных типов возврата изображений, определяемых в таблице C.4.

Таблица C.4 – Правомерные типы возврата изображений

Тип	Истолкование
"jpp-stream"	JPP-поток, как определяется в Приложении A. Тип "jpp-stream" может на необязательной основе сопровождаться ";ptype=ext", в этом случае тип запрашиваемого возврата является типом, в котором все заголовки сообщений бункеров данных зон имеют расширенную форму (см. A.2.2)
"jpt-stream"	JPT-поток, как определяется в Приложении. Тип "jpt-stream" может на необязательной основе сопровождаться ";ttype=ext"; в этом случае тип запрашиваемого возврата является типом, в котором все запрашиваемые заголовки сообщений бункера данных элементов мозаичного изображения имеют расширенную форму (см. A.2.2)
"raw" [*необработанный*]	Клиент запрашивает полную последовательность байтов в логической цели, подлежащую доставке без изменения
Другие значения	Зарезервировано для использования ИСО

Если поле запроса `type` опускается, то тип возврата следует определять другими средствами.

В сеансе, в котором запросы включают в себя поле запроса идентификатора ID Канала, значение параметра возврата должно поддерживаться в последовательных откликах для запросов данных изображения или метаданных, которые соответствуют той же самой логической цели.

ПРИМЕЧАНИЕ 1. − Другие типы носителей информации изображения (например, jp2, jpeg, tiff, png), если имеются, могут быть предоставлены сервером как услуга перекодирования с функциональными возможностями JPIP.

ПРИМЕЧАНИЕ 2. − Для типа возврата необработанного кодового потока, данным откликов следует состоять полностью из запрашиваемого объекта. Поэтому многие из других возможных полей запросов клиентов не имели бы значения, и были бы проигнорированы сервером.

C.7.4 Скорость доставки (drate)

```
drate = "drate" "=" rate-factor
rate-factor = UFLOAT
```

Это поле используется для определения скорости доставки различных кодовых потоков. Если это поле поставляется, то сервер должен доставлять данные, принадлежащие различным кодовым потокам в окне обзора, следуя временно упорядоченному графику. Кодовые потоки, которые принадлежат окну обзора, все являются потоками, опознаваемыми через поле запроса Кодового потока и поле запроса Контекста кодового потока, возможно, с дискретизацией на пониженной скорости в соответствии с полем запроса Скорости дискретизации.

Чтобы обеспечивать значение к этому полю запроса, информация синхронизации должна быть связана с различными кодовыми потоками в окне обзора. Если кодовые потоки принадлежат файлу MJ2, то информация синхронизации обеспечивается таким файлом. Файл MJ2 обеспечивает преобразование между

каждым кодовым потоком и номинальным временем воспроизведения, которое опознается здесь как "исходный момент времени".

Если кодовые потоки не имеют источника информации синхронизации, но поле запроса Скорости дискретизации присутствует, сервер должен предполагать, что кодовые потоки в окне обзора имеют исходные моменты времени, которые отделены с помощью обратной величины из значения в поле запроса Скорости дискретизации.

Если кодовые потоки не имеют источника информации синхронизации, а поле запроса Скорости дискретизации не присутствует, сервер должен предполагать, что кодовые потоки в окне обзора имеют исходные моменты времени, которые отделены друг от друга точно на одну секунду.

Поле запроса Скорости доставки обеспечивает коэффициент масштабирования между скоростями доставки и источника. Если коэффициент скорости дается как 1, серверу следует попытаться доставлять кодовые потоки клиенту на скорости, предложенной с помощью их исходных моментов времени, отмечая, что эти исходные моменты времена могли бы не обязательно быть систематическими. Обычно, если коэффициент скорости есть *F*, серверу следует попытаться доставлять кодовые потоки клиенту на скорости, которая в раз *F* быстрее, чем предложено их исходными моментами времени.

Если сервер не способен доставлять все уместные данные для каждого кодового потока на требуемой скорости (например, из-за ограничений полосы пропускания), он должен доставлять только часть данных для каждого кодового потока, чтобы избежать нарушения запрашиваемой скорости доставки. Порция данных каждого кодового потока, которая не доставляется, может зависеть от значения `view-window-pref` `[обзор−окно−предпочтение]`, поставляемого в поле запроса Предпочтений клиента (см. C.10.2). Если предпочтение является "прогрессивным" или никакое такое предпочтение не опознается, серверу следует попытаться доставлять унифицированное, максимальное качество изображения через окно обзора, в зависимости от ограничения скорости доставки. Если было поставлено значение `view-window-pref` из `fullwindow`" [*полное окно*], сервер мог бы усечь представление, связанное с каждым кодовым потоком некоторым другим способом. В любом случае, поведение должно быть подобно тому поведению, которое следовало бы из выпуска клиентом последовательности запросов для каждого из уместных кодовых потоков по очереди, на скорости доставки.

Если сервер способен доставлять все уместные данные для каждого кодового потока, на запрашиваемой скорости, ему следует сделать неработающими соединения, как требуется для обеспечения того, что скорость доставки не превышается.

Если это поле не поставляется и если значение `view-window-pref` из "`fullwindow`" не было определено, серверу следует попытаться упорядочить соответствующие данные таким образом, чтобы постепенно наращивать качество всех кодовых потоков однородным образом.

C.8 Поля запросов для управления кэш-памятью

C.8.1 Модель (model)

C.8.1.1 Общие положения

```
model = "model" "=" 1#model-item

model-item = [codestream-qualifier ","] model-element

model-element = ["-"] bin-descriptor

bin-descriptor = explicit-bin-descriptor    ; C.8.1.2
               / implicit-bin-descriptor    ; C.8.1.3

codestream-qualifier = "[" 1$(codestream-range) "]"

codestream-range = first-codestream-id ["-" [last-codestream-id]]

first-codestream-id = UINT

last-codestream-id = UINT
```

Это поле может использоваться в запросах, основанных на сеансах, или в запросах, не меняющих своего состояния в процессе исполнения. Запрос на основе сеанса является любым запросом, который включает в себя поле идентификатора Channel-ID, так как каналы связаны с сеансом, управляемым сервером. Поле "модель" содержит один или более описателей бункеров, каждый из которых опознает бункер данных, или диапазон бункеров данных, о которых передается информация кэш-памяти. Для запросов внутри сеанса эта информация кэш-памяти служит, чтобы обновлять модель сервера для кэш-памяти клиента. Имеется только одна модель кэш-памяти для каждой логической цели, связанной с сеансом. Для запросов, не меняющих состояния

в процессе исполнения, модель сервера кэш-памяти клиента пуста в начале запроса, но обновляется с помощью поля "модель" (если существует) прежде, чем сервер сформулирует свой отклик. При завершении обработки запроса, не меняющего своего состояния в процессе исполнения, вся информация модели кэш-памяти сбрасывается сервером.

Чтобы облегчать эффективный обмен информацией о модели кэш-памяти, для значений описателей бункеров обеспечиваются две формы. Они называются "явной" и "неявной" формами и описываются в следующем подразделе. Клиенты могут выпускать запросы, используя любую форму, и могут смешивать две формы описателя бункера внутри поля запроса единственной "модели", если это желательно.

Если описателю бункера предшествуют символ "-", то он называется вычитающим. В противном случае он называется добавляющим. Вычитающий описатель бункера сообщает серверу, что уместные данные следует удалить из модели сервера для кэш-памяти клиента. Удаление элементов из модели кэш-памяти означает, что сервер не должен предполагать, что клиент уже имеет эти элементы. Значения дескрипторов бункеров обрабатываются по порядку.

Добавляющий описатель бункера (тот, которому не предшествует символ "-") сообщает серверу о данных, которые клиент уже имеет в своей кэш-памяти. Сервер может добавлять эту информацию к своей модели кэш-памяти и может предполагать, что клиент уже имеет указанные данные.

Поле "модель" может сослаться на данные бункера, которые не относятся к окну обзора, обозначаемому другими полями запроса (Размер кадра, Размер области, Смещение и т. д.). Там, где это случается, обработка модели кэш-памяти могла бы не затрагивать отклик на текущий запрос, но, тем не менее, может затрагивать отклик на будущие запросы (если запрос не является запросом, меняющим свое состояние в процессе исполнения).

Везде, где перечень пунктов модели включает в себя определитель кодового потока, все последующие элементы модели должны добавляться или должны вычитаться (по обстановке) из всех кодовых потоков, идентификаторы которых перечисляются определителем кодового потока. Определители кодового потока могут быть вкраплены по всему перечню, чтобы постепенно изменять совокупность кодовых потоков, которые должны быть затронуты последующими элементами модели. Любой элемент модели, которому не предшествует определитель кодового потока, применяется к первому кодовому потоку, который запрашивается через поле запроса Кодового потока. Если поле запроса Кодового потока не присутствует, то значения элементов моделей, которым не предшествует определитель кодового потока, должны ссылаться на кодовый поток 0, независимо от того, включено ли поле запроса Контекста кодового потока, или нет. Если идентификатор id последнего кодового потока [*last-codestream-id*] не присутствует, но дефис определителя имеется, то тогда это должно означать идентификатор id первого кодового потока [*first-codestream-id*], и все последующие кодовые потоки включаются.

Запросы внутри сеанса не должны включать в себя никакой определитель кодового потока, который упоминает более чем единственный поток.

ПРИМЕЧАНИЕ 1. − Серверу следует попытаться эксплуатировать утверждения обработки добавляющей модели кэш-памяти, но быть свободным в игнорировании некоторых или всех из них в возможных затратах транспортной эффективности. Клиентам следует знать, что серверы, вероятно, могли бы проигнорировать утверждения обработки добавляющей модели кэш-памяти, что относятся к бункерам данных, принадлежащим к кодовым потокам, которые не будут обслуживаться текущим запросом. Чтобы убрать такие сомнения там, где вовлечены многократные кодовые потоки, можно использовать поле запроса "mset" для определения набора моделируемых кодовых потоков.

ПРИМЕЧАНИЕ 2. − Обработка модели кэш-памяти сервера на основе сеанса в общем случае затрагивает отклик и на текущий запрос, и на любые будущие запросы. Более того, все каналы внутри сеанса, которые связаны с отдельной логической целью, совместно используют ту же самую модель кэш-памяти. Таким образом, поля "модель" в запросах, которые прибывают, используя один канал (поле идентификатора ID Канала), могут затронуть отклик на запросы, которые прибывают, используя различный канал. Важно отметить, что запросы, которые используют различные каналы JPIP (различные значения идентификаторов ID Канала), могут прибывать в сервер асинхронно, если используются отдельные каналы TCP, чтобы транспортировать запрос либо непосредственно от клиента, либо косвенно в промежуточном посреднике. Клиентам следует предпринять любое действие, которое необходимо, чтобы гарантировать, что их инструкции обработки модели кэш-памяти в свете этих соображений остаются значащими.

C.8.1.2 Явная форма

```
explicit-bin-descriptor = explicit-bin
                          [":" (number-of-bytes / number-of-layers )]

explicit-bin = codestream-main-header-bin
             / meta-bin
             / tile-bin
             / tile-header-bin
             / precinct-bin

number-of-bytes = UINT

number-of-layers = %x4c UINT              ; "L"
```

```
codestream-main-header-bin = %x48 %x6d   ; "Hm"
meta-bin = %x4d bin-uid                  ; "M"
tile-bin = %x54 bin-uid                  ; "T"
tile-header-bin = %x48 bin-uid           ; "H"
precinct-bin = %x50 bin-uid              ; "P"
bin-uid = UINT / "*"
```

Значения описателей бункеров, которые явно относятся к бункерам данных, имеют следующие типы: M (бункеры метаданных), Hm (бункеры данных главных заголовков), H (бункеры данных заголовков элементов мозаичного изображения), P (бункеры данных зоны) или T (бункеры данных элементов мозаичного изображения). Явные описатели бункеров распознают соответствующий бункер данных (или бункеры данных) внутри соответствующих кодовых потоков, используя или уникальный целочисленный идентификатор, или знак группового символа, "*". Единственным исключением этого является бункер данных главного заголовка кодового потока, чей описатель бункера есть "Hm". Для всех других классов бункеров данных уникальный идентификатор идентичен значению, сообщенному идентификатором в составе класса в заголовках сообщений JPP-потока и/или JPT-потока (см. Приложение A).

Символ обобщения, "*", должен использоваться только в запросах, не меняющих своего состояния в процессе исполнения. Там, где он используется, описатель бункера обращается одновременно ко всем бункерам данным уместного класса (метаданные, зона, заголовок элемента мозаичного изображения или элемент мозаичного изображения), соответствующего окну обзора.

Каждый описатель бункера может быть оценен количеством байтов. Добавляющий описатель бункера, который оценивается количеством байтов, *B*, указывает, что клиент в своей кэш-памяти уже имеет, по крайней мере, первые *B* байтов обозначенного бункера данных; сервер может добавить первые *B* байтов бункера данных к своей модели кэш-памяти. Вычитающий описатель бункера, который оценивается количеством байтов, *B*, указывает, что клиент имеет, самое большее, первые *B* байтов обозначенного бункера данных; сервер должен удалить любые байты после первых *B* байтов бункера данных из своей модели кэш-памяти.

ПРИМЕР 1: Описатель бункера оцененного вычитающего устройства типа "-P23:10" означает, что серверу следует удалить из своей модели кэш-памяти все байты, кроме первых 10 байтов бункера 23 данных зоны. Это не подразумевает, что клиент имеет первые 10 байтов бункера 23 данных зоны в своей кэш-памяти, и серверу не следует допускать это путем добавления этих байтов к своей модели кэш-памяти, если они уже не присутствовали.

Описатели бункеров зон альтернативно могут быть оценены количеством слоев. Добавляющий описатель бункера, который оценивается количеством слоев, *L*, указывает, что клиент уже имеет, по крайней мере, первые *L* слоев (первые *L* пакетов) обозначенной зоны; сервер может добавить байты, соответствующие этим слоям, к своей модели кэш-памяти. Вычитающий описатель бункера зоны, который оценивается количеством слоев, *L*, указывает, что клиент имеет, самое большее, первые *L* слоев (*L* пакетов) обозначенной зоны; из своей модели кэш-памяти сервер должен удалить байты, соответствующие любым последующим слоям такой зоны.

Описатель бункера без определителя количества байтов или количества слоев означает полный явный бункер данных.

ПРИМЕР 2: "model=M0,Hm,H7:20,P3" означает, что в своей кэш-памяти клиент имеет, по меньшей мере, всё из бункера 0 метаданных, всё из главного заголовка кодового потока, первые 20 байтов заголовка 7 элемента мозаичного изображения и всё из зоны 3.

ПРИМЕР 3: "model=P3:256,P5:L2,-P6:20" означает, что клиент имеет, по меньшей мере, первые 256 байтов зоны 3 и первые два слоя (пакета) зоны 5, но (самое большее) он не имеет ничего за пределами 20-го байта зоны 6 (или он может не иметь первые 20 байтов).

ПРИМЕР 4: "model=M*,-M5,-H*,-P*:L3" означает, что клиент имеет (или готов разрешить серверу верить, что он имеет) все бункеры метаданных, кроме бункера 5 метаданных, не имеет бункеров данных заголовков элементов мозаичного изображения, которые относятся к окну обзора, и, самое большее, первые 3 слоя любой зоны, которая относится к окну обзора. Отметим, что групповые символы, используемые здесь, разрешены только тогда, когда утверждение "модель" появляется в запросе, не меняющем своего состояния в процессе исполнения.

ПРИМЕР 5: "model=[30-200],Hm,H*,M*,P0,[0-29],-Hm,-H*,-M*,-P*" означает, что клиент имеет все заголовки и метаданные, плюс бункер 0 данных зоны из кодовых потоков от 30 до 200 включительно, но что он удалил все бункеры данных заголовка, метаданных и зоны из первых 30 кодовых потоков.

C.8.1.3 Неявная форма

```
implicit-bin-descriptor = 1*implicit-bin [":" number-of-layers]

implicit-bin  = implicit-bin-prefix (data-uid / index-range-spec)

implicit-bin-prefix = %x74    ; t -- tile
                    / %x63    ; c -- component
                    / %x72    ; r -- resolution level
                    / %x70    ; p -- position

index-range-spec = first-index-pos "-" last-index-pos

first-index-pos = UINT

last-index-pos = UINT

data-uid = UINT / "*"
```

Значения описателей бункеров, которые неявно относятся к бункерам данных, имеют следующие типы: t (элемент мозаичного изображения, к которому принадлежит зона), c (компонент изображения, к которому принадлежит зона), r (уровень разрешающей способности компонента элемента мозаичного изображения, к которому принадлежит зона) или p (позиция зоны внутри ее разрешающей способности компонента элемента мозаичного изображения). Неявные описатели бункеров используются для распознавания бункеров данных зоны через индексы. Все индексы должны начинаться от 0. Индекс уровня разрешающей способности 0, `r0`, относится к самому низкому уровню разрешающей способности (подполоса LL) компонента элемента мозаичного изображения. Индексы позиции, `p`, запускаются слева направо и от вершины к основанию прогрессии разрешающей способности компонента элемента мозаичного изображения, в манере сканирования строки, как описано в Рекомендации МСЭ-Т Т.800 | ИСО/МЭК 15444-1.

В запросах, не меняющих своего состояния в процессе исполнения, любое или всё из элемента мозаичного изображения, компонента, уровня разрешающей способности или спецификатора неявного бункера позиции может быть заменено диапазоном индексов или символом обобщения, "*". В любом случае, описатель бункера расширяется, чтобы включить все значения диапазона индексов, относящиеся к окну обзора. Ни один из этих вариантов не должен использоваться для запросов внутри сеанса.

В запросах, не меняющих своего состояния в процессе исполнения, любое или всё из элемента мозаичного изображения, компонента, уровня разрешающей способности или индексов позиций также может быть заменено отдельным диапазоном индексов. Значение first-index-pos в index-range-spec дает первый индекс в диапазоне. Значение last-index-pos дает последний индекс в диапазоне и должно быть больше или равно значению first-index-pos. Оба определяемые индекса являются включающими. Индекс last-index-pos не может быть опущен. Если задается диапазон индексов элементов мозаичного изображения ("t"), то диапазон относится к прямоугольному массиву элементов мозаичного изображения, чей левосторонний верхний угол имеет значение first-index-pos, и чей правосторонний нижний угол имеет значение last-index-pos. Подобным образом, если задается диапазон индексов позиций ("p"), то диапазон относится к прямоугольному массиву позиций зон, чьи верхний левый и нижний правый углы задаются соответственно значениями first-index-pos и last-index-pos. Что касается групповых символов, то диапазоны не должны использоваться в запросах внутри сеанса.

Неявные описатели бункеров зоны могут быть оценены количеством слоев, для которых синтаксис и истолкование идентичны тем, что есть для оцененных явных описателей бункеров зон слоя, описанных ранее.

ПРИМЕР 1: "model=t0c2r3p4:L5" указывает, что клиент имеет первые 5 пакетов из 5-ой зоны в последовательности, из четвертого уровня разрешающей способности, из третьего компонента, из элемента мозаичного изображения 0.

ПРИМЕР 2: "model=t10r0,t*r1:L4" означает, что клиент имеет все слои индекса 10 элемента мозаичного изображения на уровне 0 разрешающей способности, и первые 4 слоя всех элементов мозаичного изображения, относящихся к окну обзора на уровне 1 разрешающей способности. Отметим, что групповой символ соответствует только для запросов, не меняющих своего состояния в процессе исполнения.

ПРИМЕР 3: "model=t0-10:L2" указывает, что клиент имеет первые 2 слоя из элементов мозаичного изображения с 0 по 10. Отметим, что диапазон соответствует только для запросов, не меняющих своего состояния в процессе исполнения.

ПРИМЕР 4: "model=t*r0-2:L4" указывает, что клиент имеет первые 4 слоя из уровней разрешающей способности с 0 по 2 из всех элементов мозаичного изображения, уместных для окна обзора. Отметим, что групповой символ и диапазон соответствует только для запросов, не меняющих своего состояния в процессе исполнения.

C.8.2 Резюме вариантов описателей кэш-памяти (информативное)

Таблица C.5 – Резюме вариантов описателей кэш-памяти

Тип формы	Групповой символ		диапазон индексов	количество уровней (например, ":L3")	количество байтов (например, ":256")
	не меняющий состояния в процессе исполнения	на основе сеанса			
Явная форма	Разрешается	Не разрешается	Не разрешается	Разрешается	Разрешается
Неявная форма	Разрешается	Не разрешается	Разрешается только для не меняющих состояния в процессе исполнения	Разрешается	Не разрешается

C.8.3 Модель части элемента мозаичного изображения, включающая в себя JPT-потоки (tpmodel)

```
tpmodel =  "tpmodel" "=" 1#tpmodel-item

tpmodel-item = [codestream-qualifier "," ] tpmodel-element

tpmodel-element = ["-"] tp-descriptor

tp-descriptor = tp-range / tp-number

tp-range = tp-number "-" tp-number

tp-number = tile-number "." part-number

tile-number = UINT

part-number = UINT
```

Это поле может использоваться для указания характерных частей элементов мозаичного изображения, которые клиент хотел бы добавить или вычесть из модели кэш-памяти сервера. Подобно полю "модель", оно может использоваться как в запросах на основе сеанса, так и в запросах, не меняющих своих состояний в процессе исполнения. В случае запросов, не меняющих своего состояния в процессе исполнения, модель кэш-памяти пуста в начале запроса и не сохраняется между запросами, но она все еще обеспечивает полезный механизм для опознавания элементов мозаичного изображения, которые уже находятся в кэш-памяти клиента.

Если описателю части элементов мозаичного изображения предшествует знак "-", он называется вычитающим. В противном случае он является добавляющим. Добавляющий описатель части элементов мозаичного изображения указывает, что клиент в своей кэш-памяти уже имеет указанную часть элемента мозаичного изображения или диапазон частей элементов мозаичного изображения; сервер может добавить эти элементы к своей модели кэш-памяти. Вычитающий описатель части элемента мозаичного изображения указывает, что в своей кэш-памяти клиент не имеет указанной части элемента мозаичного изображения или диапазона частей элемента мозаичного изображения; сервер должен удалить эти элементы из своей модели кэш-памяти.

Первое значение в номере части элемента мозаичного изображения является индексом элемента мозаичного изображения (начиная от 0); второе значение является номером части (начиная от 0) внутри элемента мозаичного изображения. Считается, что диапазон tp независимо содержит элементы мозаичного изображения от первого номера элемента мозаичного изображения до второго номера элемента мозаичного изображения и частей элементов мозаичного изображения от первого номера части элемента мозаичного изображения до второго номера части элемента мозаичного изображения. Таким образом, 4.0-5.1 включает в себя части элементов мозаичного изображения 4.0, 4.1, 5.0 и 5.1, но не 4.2 или 5.2.

Внутри одиночного запроса могут появляться поля запросов и "tpmodel", и "модель". В этом случае, однако, сервер должен отражать результаты поля "модель" на своей модели кэш-памяти перед обработкой поля "tpmodel".

Значения определителя кодового потока могут быть вкраплены среди перечня элементов tpmodel, чтобы изменять совокупность кодовых потоков, к которым следующие элементы tpmodel обращаются, следуя в точности тем же самым принципам, как для поля запроса "модель".

ПРИМЕЧАНИЕ. − В отличие от поля запроса "модель", внутри поля запроса "tpmodel" разрешаются и диапазоны частей элементов мозаичного изображения, и диапазоны кодовых потоков (в определителях кодового потока), независимо от того, появляется ли оно внутри запроса на основе сеанса или запроса, не меняющего своего состояния в процессе исполнения.

ПРИМЕР 1: "tpmodel=4.0,4.1,5.0-6.2" указывает, что в своей кэш-памяти клиент уже имеет первые две части элементов мозаичного изображения 4 и первые 3 части элементов мозаичного изображения 5 и 6.

ПРИМЕР 2: "tpmodel=-4.0-6.254" указывает, что в своей кэш-памяти клиент не имеет частей элементов мозаичного изображения из элементов мозаичного изображения 4, 5 или 6.

ПРИМЕР 3: "tpmodel=3.0,[131-133],4.0,[100],-0.0-65534.254" указывает, что клиент имеет часть 0 элемента мозаичного изображения из элемента 3 мозаичного изображения из кодового потока 0, упомянутого в запросе, плюс часть 0 элемента мозаичного изображения из элемента мозаичного изображения 4 из каждого из кодовых потоков с 131 по 133 включительно, и что он исключает все части элемента мозаичного изображения из своей кэш-памяти кодового потока 100.

C.8.4 Необходимость в запросах, не меняющих своего состояния в процессе исполнения (need)

```
need = "need" "=" 1#need-item

need-item = [codestream-qualifier "," ] bin-descriptor
```

Это поле может появляться только в запросе, не меняющем своего состояния в процессе исполнения, т. е. в тех, которые не включают в себя поле запроса идентификатора ID Канала. Оно имеет тот же самый синтаксис, что и поле запроса модели, за исключением того, что описателям бункера не должен предшествовать символ "-". Поле запроса "необходимость" [*"need"*] не должно появляться внутри того же самого запроса, как поле "модель" или "tpmodel".

Поле запроса "необходимость" указывает набор бункеров данных (или суффиксов бункеров данных), которые представляют потенциальный интерес для клиента. Серверу не нужно посылать информацию, которая не имеет потенциального интереса. Независимо от того, насколько большим может быть набор потенциально интересных бункеров данных, сервер должен посылать только информацию, которая является уместной для полей запроса окон обзора или полей запроса метаданных.

Воздействие поля "необходимость" на запрос сервера можно объяснить, используя понятие временной модели кэш-памяти. Временная модель кэш-памяти устанавливается в исходное состояние (пусто) непосредственно перед тем, как запрос обрабатывается, и сбрасывается после того, как отклик порождается. Если в запросе появляется поле "необходимость", все возможные бункеры данных добавляются в модель кэш-памяти, после чего все элементы, упоминаемые описателями бункеров в поле "необходимость", удаляются из модели кэш-памяти. Сервер затем обрабатывает запрашиваемое окно обзора, используя эту модель кэш-памяти, чтобы определить элементы, которые не нужно посылать клиенту.

Определители кодового потока могут быть вкраплены среди перечня описателей бункера, чтобы изменять совокупность кодовых потоков, к которым обращаются следующие описатели бункера, следуя в точности тем же самым принципам, как и для полей запроса "модель" и "tpmodel".

ПРИМЕР 1: "need=M1,H0:20,P0" означает, что клиенту необходим весь бункер 1 метаданных, данные из 20-го байта бункера 0 данных заголовка элемента мозаичного изображения и всё из бункера 0 данных зоны.

ПРИМЕР 2: "need=P1:256,P5:L2" означает, что клиенту необходимы данные после 256-го байта (или из байта 256) из бункера 1 данных зоны и слои после 2-го слоя из бункера 5 данных зоны.

ПРИМЕР 3: "need= H*,P*:L3" означает, что клиенту необходимы все бункеры данных заголовков элементов мозаичного изображения, уместные с окном обзора и слои после 3-го слоя всех бункеров данных зон, уместных с окном обзора.

ПРИМЕР 4: "need=t10r0,t*r1:L4" означает, что клиенту необходимы все слои индекса 10 элемента мозаичного изображения на уровне 0 разрешающей способности и слои после 4-го слоя всех элементов мозаичного изображения, соответствующих окну обзора на уровне 1 разрешающей способности.

ПРИМЕР 5: "need=t*r0-2:L4" означает, что клиенту необходимы все слои из уровня 4 всех бункеров данных зоны в уровнях разрешающей способности с 0 по 2 (0, 1 и 2) во всех элементах мозаичного изображения и компонентах, соответствующих запросу окна обзора.

ПРИМЕР 6: "need=[120-131],r0,[140;143-145],r0-1" означает, что клиенту необходимы уровень 0 разрешающей способности кодовых потоков от 120 до 131 включительно, а также уровни 0 и 1 разрешающей способности кодовых потоков 140 и от 143 до 145 включительно.

C.8.5 Часть элемента мозаичного изображения, необходимая для запросов, не меняющих своих состояний в процессе исполнения (tpneed)

```
tpneed = "tpneed" "=" 1#tpneed-item

tpneed-item = [codestream-qualifier "," ] tp-descriptor
```

Это поле может появиться только в запросах, не меняющих своего состояния в процессе исполнения, т. е. в тех запросах, которые не включают в себя поле запроса ID Канала. Оно имеет тот же самый синтаксис, как и поле запроса tpmodel, за исключением того, что tp-описателям не должен предшествовать символ "-". Поле запроса "tpneed" не должно появляться внутри того же самого запроса, как поле запроса "модель" или "tpmodel".

Поле запроса "tpneed" указывает набор частей элемента мозаичного изображения, которые представляют потенциальный интерес для клиента. Сервер не нуждается в отправке информации, которая не имеет потенциального интереса. Независимо от того, насколько большим может быть набор потенциально интересных частей элемента мозаичного изображения, сервер должен посылать только информацию, которая является уместной для полей запроса окон обозрения или для поля запроса метаданных.

Воздействие поля "tpneed" на запрос сервера можно объяснить, используя понятие временной модели кэш-памяти. Временная модель кэш-памяти устанавливается в исходное состояние (пусто) непосредственно перед тем, как запрос обрабатывается, и сбрасывается после того, как отклик порождается. Если в запросе появляется поле "tpneed", то все возможные части элементов мозаичного изображения и бункеры данных добавляются в модель кэш-памяти, после чего все элементы, упоминаемые описателями бункеров в поле "необходимость", и все части элементов мозаичного изображения в поле "tpneed" удаляются из модели кэш-памяти. Сервер затем обрабатывает запрашиваемое окно обзора, используя эту модель кэш-памяти, чтобы определять элементы, которые не нужно посылать клиенту.

Определители кодового потока могут быть вкраплены среди перечня частей элементов мозаичного изображения, чтобы изменять совокупность кодовых потоков, к которым обращаются следующие части элементов мозаичного изображения, следуя в точности тем же самым принципам, как и для полей запроса "модель" и "tpmodel".

C.8.6 Набор модели для запросов внутри сеанса (mset)

```
mset = "mset" "=" 1#sampled-range
```

Это поле служит двум целям. Прежде всего, оно сообщает серверу о наборе кодовых потоков, для которых клиент готов занести в кэш-память данные, доставленные сервером. Во втором случае, оно обеспечивает механизм, чтобы клиент узнавал о кодовых потоках, для которых сервер готов моделировать кэш-память клиента. Более точно, если совокупность индексов кодовых потоков, поставляемая в запросе "mset", отличается любым образом от набора кодовых потоков, через которые сервер в настоящее время готов предложить моделирование кэш-памяти, сервер должен обеспечить заголовок отклика Набора модели, как обсуждается в D.2.18.

Строка параметра поля запроса "mset состоит из разделенного запятыми перечня диапазонов индексов кодовых потоков, возможно, с дискретизацией на пониженной скорости, следуя соглашениям, выделенным в соединении с полем запроса Кодового потока в C.4.6.

В дополнение к кодовым потокам, упомянутым в запросе "mset", сервер может также обеспечивать модель кэш-памяти для всех кодовых потоков, связанных с его откликом на текущий запрос. Это есть совокупность кодовых потоков, опознаваемых с помощью запроса клиента (см. поля запроса Кодового потока и Контекста кодового потока C.4.7), если сервер не указывает уменьшенный набор кодовых потоков через заголовок отклика Кодового потока (см. D.2.9). Если поле запроса "mset" не предоставляется, клиенту не следует предполагать, что сервер обеспечивает модель кэш-памяти для любых других кодовых потоков, кроме потоков, связанных с его откликом; однако он может моделировать другие кодовые потоки. Если поле запроса "mset" задается, то сервер должен сбросить любую информацию модели кэш-памяти, которую он имеет для всех других кодовых потоков, кроме упомянутых или в запросе "mset", или в наборе кодовых потоков, связанных с его данными отклика. Кроме того, результаты любой обработки модели кэш-памяти через поля запросов "модель" или "tpmodel" должны быть ограничены только этим кодовым потокам.

Сервер, по своему усмотрению, может уменьшать количество кодовых потоков в "mset"; в этом случае он должен выдать заголовок отклика "mset", опознающий фактический набор кодовых потоков, которые моделируются; кроме того, этот набор моделируемых кодовых потоков должен, по крайней мере, включать в себя все кодовые потоки, связанные с данными отклика сервера (те, которые запрашиваются клиентом или опознаются с помощью заголовка отклика Кодового потока сервера, если они имеют место). В этом случае эти утверждения применяются к тем кодовым потокам, которые содержатся в "mset", обозначаемом сервером. Сервер не может обозначать набор кодовых потоков, который больше, чем тот, что упоминается в запросе "mset" клиента, объединенный с теми кодовыми потоками, которые связаны с данными отклика сервера.

Отметим, что сервер может менять свое поле "mset" от запроса к запросу, поэтому клиенты, которым нужно отслеживать и/или сильно ограничивать "mset" сервера, могли бы выбирать включение поля запроса "mset" с каждым запросом.

C.9 Параметры запросов загрузки

C.9.1 Загрузка в сервер (upload)

```
upload = "upload" "=" upload-type
upload-type = image-return-type                    ; C.7.3
```

Это поле указывает, что клиент осуществляет загрузку в сервер нового изображения или метаданных. Значение `upload-type` [*загрузка−тип*] может быть любым из действительных значений `image-return-type` [*изображение−возврат−тип*], которые могли использоваться с полем запроса типа. См. Приложение E для информации относительно загрузки данных в сервер.

C.10 Поля запросов возможностей и предпочтений клиентов

C.10.1 Возможность клиента (cap)

```
cap = "cap" "=" 1#capability-group
capability-group = processing-capability
                 / depth-capability
                 / config-capability
processing-capability = compatibility-capability
                      / vendor-capability
compatibility-capability = "cc." compatibility-code
vendor-capability = "vc." vendor-code [":" vendor-value]
vendor-code = 1*(LOWER / DIGIT / "." / "-")
vendor-value = TOKEN
depth-capability = "depth:" UINT
config-capability = "config:" UINT
```

Это поле определяет возможности клиента. Для запросов на основе сеанса (тех, которые включают в себя поле запроса ID Канала) любые поля возможностей, передаваемые клиентом, должны затрагивать только канал, связанный с запросом, и должны считаться постоянными. Возможности не должны передаваться клиентом повторно для последующих запросов на том же самом канале.

Когда новый канал создается из существующего канала, его клиентские возможности наследуются. Для запросов, не меняющих своих состояний в процессе исполнения, и для запросов, выпущенных внутри канала, возможности которого никогда не определялись или не наследовались, способности клиента могут быть определены или предугаданы другими средствами. Возможности, связанные с каналом, могут быть изменены включением поля запроса Возможностей клиентов внутри любого запроса.

Если поле запроса Возможности клиента обозначает один или более вариантов `processing-capability` [*обработка−возможность*], сервер должен предполагать, что клиент не имеет ни одного из других вариантов `processing-capability`, которые могли быть упомянуты. Если никакие варианты `processing-capability` в поле запроса Возможностей клиентов не поставляются, сервер должен продолжать использовать любые предыдущие знания, которые он имел, касающееся обработки возможностей. Варианты `processing-capability`, определенные в соответствии с этой Рекомендацией | Международным стандартом, описаны в таблице C.6.

Таблица C.6 – Допустимые возможности элемента `processing-capabilities`

Возможность	Смысл
`compatibility-capability`	Клиент поддерживает все файлы, которые содержат `compatibility-code` [*совместимость-код*] в перечне совместимости в блоке Типа файла. Например, для указания, что клиент поддерживает все файлы JP2, клиент в поле запроса Возможности передавал бы "cc.jp2_". Значение кода совместимости "jp2c" должно использоваться, чтобы указывать поддержку для необработанных кодовых потоков JPEG 2000.
`vendor-capability`	Клиент поддерживает возможности поставщика, определяемые с помощью параметра `vendor-code` [*поставщик-код*]. Параметр `vendor-code` должен быть строкой, указывающей обратное доменное имя поставщика, что определило свойство, сопровождаемое именем свойства поставщика. Например, если example.com определил свойство, называемое "дистанция" [*"distance"*], то тогда значение `vendor-code` для этого свойства должно быть "com.example.distance". Параметр `vendor-value` указывает дополнительное значение, как определено свойством конкретного поставщика.

Если поставляется параметр `depth-capability` [*глубина−возможность*], то он указывает максимальную битовую глубину дискретизации (точность), на которой клиент способен эксплуатировать разложенную совокупность изображений. Если клиент поддерживает различные битовые глубины для различных компонентов изображений, то это поле должно определять битовую глубину компонента, для которой клиент имеет наибольшую возможность битовой глубины.

ПРИМЕЧАНИЕ 1. − Если клиент поддерживает 12 битов для сигнала яркости и 8 битов для информации о цветности, значение возможности глубины должно быть 12.

ПРИМЕЧАНИЕ 2. − Клиенты, имеющие возможность обработки только *N* битов на отсчет, в общем случае все еще будут способны обрабатывать кодовые потоки, маркер SIZ которых указывает битовую глубину намного больше, чем *N*.. Однако этот флаг может использоваться сервером для определения соответствующего способа, которым доставлять запрашиваемые данные изображения.

Если поставляется параметр `config-capability` [*конфигурация−возможность*], то он должен находиться в диапазоне от 0 до 255, представляя слово из 8 битов, чьи индивидуальные биты истолковываются как флаги конфигурации. Истолкование флагов конфигурации обеспечивается в таблице С.7.

Таблица С.7 – Допустимые значения параметра возможности конфигурации

Значение	Смысл
1xxx yyyy	Клиент способен обрабатывать данные цветного изображения .
0xxx yyyy	Клиент не способен обрабатывать данные цветного изображения и желает, чтобы сервер передавал любые области запрашиваемых изображений как шкалу серого.
x1xx yyyy	Клиент обладает координатно-указательным устройством для взаимодействия конечного пользователя
x0xx yyyy	Клиент не обладает координатно-указательным устройством для взаимодействия конечного пользователя
xx1x yyyy	Клиент обладает клавиатурой для взаимодействия конечного пользователя
xx0x yyyy	Клиент не обладает клавиатурой для взаимодействия конечного пользователя
xxx1 yyyy	Клиент обладает возможностями звукового вывода
xxx0 yyyy	Клиент не обладает возможностями звукового вывода
Другие значения	Зарезервировано для использования ИСО

Значение бита "x" в таблице С.7 указывает, что заданное значение включает в себя случаи, где такой бит устанавливается или в "1", или в "0". Биты, обозначенные, как "y", не используются этой Рекомендацией | Международным стандартом, они должны быть установлены в 0 клиентами, а сервер должен их игнорировать.

С.10.2 Предпочтения клиентов (pref)

С.10.2.1 Общие положения

```
pref = "pref" "=" 1#(related-pref-set ["/r"])

related-pref-set = view-window-pref            ; C.10.2.2
                 / colour-meth-pref            ; C.10.2.3
                 / max-bandwidth               ; C.10.2.4
                 / bandwidth-slice             ; C.10.2.5
                 / placeholder-pref            ; C.10.2.6
                 / codestream-seq-pref         ; C.10.2.7
                 / other

other = TOKEN
```

Это поле определяет предпочтения клиентов для образа действий сервера. Для запросов на основе сеанса (тех, которые включают поле запроса ID Канала) любые поля предпочтений, передаваемые клиентом, должны затрагивать только канал, связанный с запросом, и должны считаться постоянными. Предпочтения не должны повторно передаваться клиентом для последующих запросов на том же самом канале. Каждое предпочтение должно происходить не больше, чем однажды в отдельном поле запроса предпочтения.

Когда новый канал создается из существующего канала, его предпочтения наследуются. Для запросов, не меняющих своего состояния в процессе исполнения, и для запросов, выпущенных внутри канала, чьи предпочтения никогда не определялись или не наследовались, предпочтения клиентов могут быть определены или предугаданы другими средствами. Если клиент желает изменить свои предпочтения, он должен снова послать полностью затронутый параметр `related-pref-set` [*относящееся−предпочтение−набор*].

Если не изложено иначе, каждый параметр `related-pref-set` определяет заказанный перечень индивидуальных маркеров предпочтений, от наиболее предпочтительного до наименее предпочтительного. Там, где возможно, сервер должен уважать предпочтения клиентов, опознаваемые этим полем запроса. Если параметр `related-pref-set` сопровождается модификатором "/r" (требуемый), сервер должен или поддержать одно из предпочтений, перечисленных в `related-pref-set`, или иначе он должен откликнуться сигналом ошибки. В последнем случае сервер должен возвратить заголовок отклика предпочтения "Нет в наличии", который отождествляет любой `related-pref-sett`, который имел модификатор "/r", но не мог быть поддержан. См. подраздел D.2.20 для более подробной информации по заголовку отклика предпочтения "Нет в наличии".

Например, рассмотрим следующий запрос Предпочтений клиента:

```
pref=fullwindow/r,color-ricc:2;color-icc
```

Этот запрос предпочтения требует, чтобы сервер возвращал полное запрашиваемое окно обзора, независимо от того, насколько большим может быть такое окно обзора (см. подраздел C.10.2.2 для обсуждения предпочтения "fullwindow" [*полное окно*]). Так как был использован модификатор "/r", сервер должен возвращать отклик ошибки, если он не способен поддерживать это предпочтение. Кроме того, клиент предпочитает использовать профили ICC Ограниченные, а не произвольные профили ICC, если профиль ICC Ограниченный имеет, по крайней мере, "исключительное качество". См. подраздел C.10.2.3 для обсуждения предпочтений цветового пространства.

Сервер должен игнорировать любое значение для `related-pref-set`, которое он не понимает, и которое сразу же не сопровождается модификатором "/r". Если значение, которое не понято, сопровождается модификатором "/r", то тогда сервер должен возвращать заголовок отклика предпочтения "Нет в наличии", указывая предпочтение, которое не способен исполнить.

Значения маркера `other` [*другие*] резервируются для использования ИСО.

C.10.2.2 Предпочтения обработки окон обзора

```
view-window-pref = "fullwindow" / "progressive"
```

Эта Рекомендация | Международный стандарт определяет два варианта для определения образа действий сервера, когда запрос не может быть обслужен точно, как установлено, следуя качественно-поступательному порядку данных отклика. Эти два варианта определяются в таблице C.8.

Таблица C.8 – Предпочтения при обработке окон обзора

Вариант	Смысл
"fullwindow"	Сервер должен уважать запрос окна обзора, но разрешено возвращать данные не в качественно-поступательном порядке.
"progressive"	Сервер может модифицировать параметры запросов окон обзора, чтобы сохранять качественно-поступательные свойства данных отклика. В случае, когда сервер действительно модифицирует параметры запросов окон обзора, модифицированному окну обзора следует быть поднабором первоначально запрашиваемого окна обзора.

Если ни вариант "fullwindow" [*полное окно*], ни вариант "progressive" [*поступательный*] в поле запроса Предпочтения клиента не определяются, сервер должен предполагать, что предпочтением клиента является вариант "progressive".

Отметим, что истолкование доставки "progressive" может быть затронуто присутствием поля запроса Скорости доставки, как объясняется в подразделе C.7.4.

C.10.2.3 Предпочтение метода цветового пространства

```
color-meth-pref = 1$(color-meth [":" meth-limit])

color-meth = "color-enum" / "color-ricc" / "color-icc" / "color-vend"

meth-limit = UINT
```

Эта Рекомендация | Международный стандарт определяет четыре варианта, которые устанавливают, какие формы спецификации данных цветового пространства следует возвращать с помощью сервера. Отдельный файл JPEG 2000 может содержать многократные спецификации цветового пространства для отдельного кодового потока или слоя наложения изображений. Это позволяет автору файла обеспечивать оптимальную спецификацию цветового пространства, все еще обеспечивая решения, имеющие возможности взаимодействия.

Однако не все программы ввода будут поддерживать все методы цветового пространства, а данные, предусмотренные для некоторых методов цветового пространства, могут быть существенного размера. В таких случаях сервер должен посылать только данные спецификации цветового пространства, которые желательны для клиента.

Если поле запроса Предпочтения клиента не содержит никакого предпочтения метода цветового пространства, то тогда поддерживаемые методы цветового пространства определяются согласно информации, содержащейся внутри поля Возможности, и никакое предпочтение не определяется.

Каждое предпочтение метода цветового пространства состоит из двух частей: из конкретного метода цветового пространства и дополнительного предела по такому предпочтению. Допустимые значения метода цветового пространства определяются в таблице C.9.

Таблица C.9 – Предпочтения клиентов по методу цветового пространства

Метод	Смысл
"color-enum"	Клиент предпочитает спецификации цветового пространства, которые используют метод Нумерованный [*Enumerated Method*]
"color-ricc"	Клиент предпочитает спецификации цветового пространства, которые используют метод ICC Ограниченный [*Restricted ICC Method*]
"color-icc"	Клиент предпочитает спецификации, которые используют метод ICC Любой [*Any ICC Method*]
"color-vend"	Клиент предпочитает спецификации, которые используют метод Поставщика [*Vendor Method*]

Дополнительное значение `meth-limit` [*метод−предел*] определяет предел на значении APPROX для такого конкретного метода цветового пространства. При использовании этих предпочтений, чтобы выбрать спецификацию цветового пространства, сервер должен рассмотреть спецификацию метода цветового пространства со значением APPROX из `meth-limit` или меньше, как если бы фактическое значение APPROX было 1 (точно). Это позволяет клиентам определять точку, в которой цветовая точность не является важной для конкретного метода цветового пространства, для текущего приложения. Например, приложение расположения страницы, которое заинтересовано только выравниванием данных изображения с другими элементами на странице, вообще может не заботиться о цветовой точности и устанавливать `meth-limit` в значение 4, имея в виду, что точность методов цветового пространства не является важной. Другое приложение, которое отображает изображения на экране низкого качества, может устанавливать `meth-limit` в значение 3 для указания, что пока цветовая точность приемлема, оно было бы удовлетворено. Знаки поля должны истолковываться как незнаковое десятичное целое число. Допустимые значения характеризуются определением поля APPROX в таблице M.24 Рекомендации МСЭ-Т T.801 | ИСО/МЭК 15444-2, а также расширениями и поправками к такой Рекомендации | Международному стандарту.

При отборе, какой блок Спецификации цветового пространства передавать клиенту, сервер должен использовать следующий алгоритм, как показано в рисунке C.3.

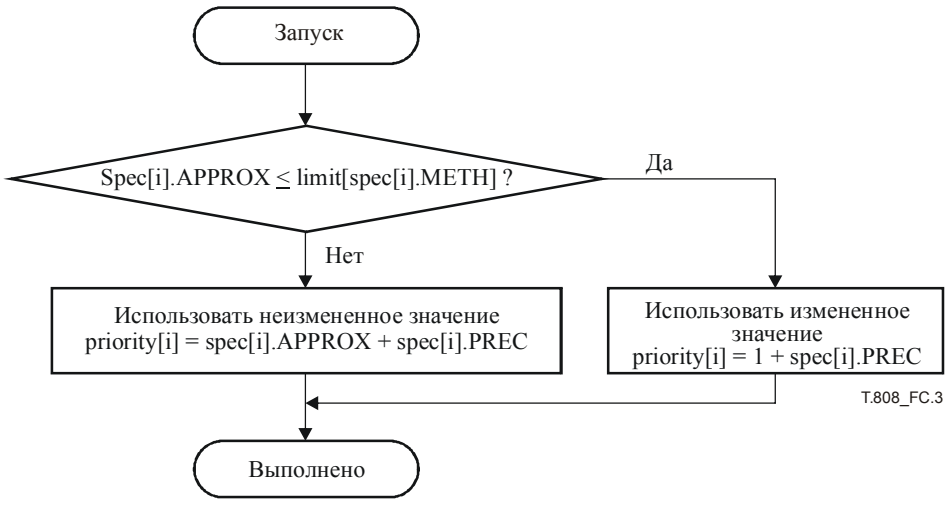

Рисунок C.3 − Процедура выбора блока спецификации цветового пространства

Для каждого блока Спецификации цветового пространства, который использует метод, что поддерживается клиентом, где:

- spec[] есть массив, содержащий все блоки Спецификации цветового пространства из заданной логической цели.
- spec[i].APPROX есть значение поля APPROX для i-го блока Спецификации цветового пространства, как он появляется в логической цели.
- spec[i].METH есть значение поля METH для блока Спецификации цветового пространства, как он появляется в логической цели.
- spec[i].PREC есть значение поля PREC для i-го блока Спецификации цветового пространства, как он появляется в логической цели.
- limit[] есть массив, который содержит значения `meth-limit`, что определяются в поле запроса, снабженном указателем с помощью допустимых значений поля METH в блоке Спецификации цветового пространства.
- priority[] есть массив вычисленных приоритетных значений для каждого блока Спецификации цветового пространства в заданной логической цели. Элемент priority[i] соответствует spec[i].

Если сервер знает, что клиент не поддерживает конкретный блок Спецификации цветового пространства, то сервер должен игнорировать такой блок для целей выбора предпочитаемого блока Спецификации цветового пространства. Как только были вычислены значения priority[] для каждого поддерживаемого блока Спецификации цветового пространства, сервер должен выбрать блок с самым низким значением приоритета. В случае, когда многократные блоки имеют приоритетное значение, равное минимальному значению для этой логической цели, сервер должен выбрать метод цветового пространства, используя следующий порядок предпочтения:

1) Нумерованный метод;
2) Метод поставщика;
3) Метод ICC Ограниченный;
4) Метод ICC Любой.

Независимо от предпочтения клиента для блоков Спецификации цветового пространства, сервер может возвращать больше блоков Спецификации цветового пространства, чем одиночный цветовой блок, определяемый этим алгоритмом, в зависимости от разделения файла в бункерах метаданных.

C.10.2.4 Максимальная ширина полосы частот

```
max-bandwidth = "mbw:" mbw

mbw = UINT ["K" / "M" / "G" / "T"]
```

Это предпочтение подает сигнал о максимальной скорости, на которой клиент хотел бы посылать данные в логическую цель. Если значение `mbw` заканчивается буквой "K", то значение будет в килобитах в секунду, где 1 килобит = 1024 бита. Если значение `mbw` заканчивается буквой "M", то значение будет в мегабитах в секунду, где 1 мегабит = = 1024^2 бита. Если значение `mbw` заканчивается буквой "G", то значение будет в гигабитах в секунду, где 1 гигабит = 1024^3 бита. Если значения `mbw` заканчивается буквой "T", то значение будет в терабитах в секунду, где 1 терабит = 1024^4 бита. В противном случае, значение будет в битах в секунду. Способность или сервера, или сети может далее ограничивать имеющуюся максимальную ширину полосы частот для службы JPIP.

C.10.2.5 Срез ширины полосы частот

```
bandwidth-slice = "slice:" slice

slice = NONZERO
```

Это предпочтение может использоваться для обозначения доли имеющейся ширины полосы частот, которую следует распределить этому каналу. Значение `slice` [*срез*] точно будет больше, чем 0. Доля ширины полосы частот получается путем деления значения среза каждого канала на сумму всех значений срезов каналов. Если не определяются, значения среза канала по умолчанию устанавливаются в 1.

Как пример, низкое значение `slice` могло бы использоваться для запроса окна обзора "фон" [*"background"*], а более высокое значение `slice` могло бы использоваться для окна обзора "передний план" [*"foreground"*]. Если сеанс содержит каналы, которые связаны с различными логическими целями, то значения срезов затрагивают пропорцию имеющейся ширины полосы частот, назначенной этим различным целям (изображениям).

C.10.2.6 Предпочтение Указателя места заполнения

```
placeholder-pref = "meta:" placeholder-branch

placeholder-branch = "incr" / "equiv" / "orig"
```

Это предпочтение может использоваться для указания предпочтительной обработки блоков Указателей мест заполнения. Там, где блоки Указателей мест заполнения появляются внутри метаданных в JPP-потоке или JPT-потоке, могут быть целых три различных представления содержимого блока: первоначальный блок; потоковый эквивалентный блок; и возрастающий кодовый поток (сообщенный через индекс). Эти возможности объясняются в подразделах A.3.6 и A.4. Как объясняется в разделе A.4, рекомендованное предположение, заданное по умолчанию, состоит в том, что клиент предпочел бы получать возрастающий кодовый поток, если имеется, а при неудаче предпочитал бы получать потоковый эквивалентный блок, если имеется. Клиент может сигнализировать об альтернативном предпочтении, используя механизм, описанный здесь. Допустимые значения предпочтения Указателя места заполнения определяются в таблице C.10.

Таблица C.10 – Предпочтения Указателей мест заполнения

Метод	Смысл
`"orig"`	Клиент предпочел бы получать первоначальный блок, если имеется. В случае неудачи, клиент предпочел бы получать потоковый эквивалентный блок, если имеется.
`"equiv"`	Клиент предпочел бы получать потоковый эквивалентный блок, если имеется. В случае неудачи, клиент предпочел бы получать первоначальный блок, если имеется.
`"incr"`	Клиент предпочел бы получать бункеры данных возрастающих кодовых потоков, если имеются. В случае неудачи, клиент предпочел бы получать потоковый эквивалентный блок, если имеется. Это является тем же самым, что и рекомендованная политика по умолчанию.

Не является правомерным обеспечивать более одного значения для предпочтения указателя места заполнения.

C.10.2.7 Упорядочение кодовых потоков

```
codestream-seq-pref = "codeseq:" codestream-seq-option

codestream-seq-option = "sequential" / "reverse-sequential"
                      / "interleaved"
```

Это предпочтение может использоваться для указания, как клиент желает, чтобы сервер доставлял многократные кодовые потоки, которые были запрошены внутри отдельного запроса. Допустимые значения при предпочтении упорядочения Кодового потока определяются в таблице C.11.

Таблица C.11 – Предпочтения при упорядочении кодовых потоков

Метод	Смысл
`"sequential"`[*последовательный*]	Клиент предпочел бы получать многократные кодовые потоки в последовательном порядке кадра (например, обслуживать многократные кадры в файле JPEG 2000 "Движение" в последовательном порядке).
`"reverse-sequential"` [*обратный−последовательный*]	Клиент предпочел бы получать многократные кодовые потоки в последовательном порядке кадра (т. е. многократные кадры в файле JPEG 2000 "Движение"), в обратном порядке.
`"interleaved"`[*чередующийся*]	Клиент предпочел бы получать многократные кодовые потоки в чередующемся порядке (например, чередуемые сервером многократные слои наложения изображений из файла JPX).

Не является правомерным обеспечивать более одного значения для предпочтения в упорядочении кодовых потоков.

C.10.3 Контрастная чувствительность (csf)

```
csf = "csf" "=" 1#csf-sample-line

csf-sample-line = csf-density [";" csf-angle] ";" 1$sensitivity

csf-density = "density" ":" UFLOAT

csf-angle = "angle" ":" UFLOAT

sensitivity = UFLOAT
```

Это поле может использоваться для подачи информации, касающейся контрастной чувствительности. В то время как эта информация может представлять воздействия и визуальной чувствительности, и передаточной функции модуляции устройства отображения, она наиболее легко описывается в понятиях предполагаемой гипотетической передаточной функции модуляции. При воспроизведении в размере кадра, обозначенном полем запроса Размера кадра, предполагается, что совокупность изображений проходит через устройство, чья передаточная функция модуляции (MTF) [*modulation transfer function*] есть $m(\omega_1, \omega_2)$, после которой она рассматривается субъектом, чья человеческая визуальная система имеет совершенно однородную функцию контрастной чувствительности. Параметр функции MTF $m(\omega_1, \omega_2)$ описывается через совокупность отсчетов. Отсчеты логарифмически размещаются в радиальном направлении, вдоль одной или более ориентированных осей. Сервер может интерполировать эти отсчеты, используя любой метод, который он считает подходящим, чтобы восстанавливать функцию MTF, которая, в свою очередь, может использоваться для регулирования порядка, в котором диапазоны байтов из бункеров данных сообщаются клиенту через сообщения JPT-потока или JPP-потока.

Каждый элемент `csf-sample-line` представляет собой отсчеты функции MTF $m(\omega_1, \omega_2)$ заданных частот $\omega_1 = \pi d^n \cos\psi$, $\omega_2 = \pi d^n \sin\psi$, где n есть индекс отсчета, начинающийся с $n = 0$ для первого отсчета `csf-density` в `csf-sample-line`, ψ есть ориентация строки отсчета CSF, выраженная в градусах (значения по умолчанию будут 0, если нет никакого значения `csf-angle`), и d − плотность дискретизации; она должна быть не больше, чем 1,0. Значение ω_1 описывает горизонтальную частоту в радианах, где $\omega_1 = \pi$ есть горизонтальная частота Найквиста (*Nyquist*). Значение ω_2 описывают вертикальную частоту в радианах, где $\omega_2 = \pi$ есть вертикальная частота Найквиста.

Значения отсчетов MTF имеют смысл только друг относительно друг друга; нет никакого конкретного истолкования для их абсолютных значений.

Приложение D

Сигнализация отклика сервера

(Это приложение образует составную часть этой Рекомендации | Международного стандарта)

D.1 Синтаксис ответа

D.1.1 Введение

Это приложение описывает все возможные элементы в отклике JPIP. Каждый главный подраздел описывает код состояния и его связанное выражение причины, заголовки откликов и возможные значения для таких заголовков, а также данные отклика. В общем случае, отклик будет состоять из многократных заголовков откликов.

D.1.2 Структура ответа

Отклик JPIP состоит из следующих элементов:

- код состояния [*status-code*];
- выражение причины [*reason-phrase*];
- заголовок отклика jpip [*jpip-response-header*];
- данные отклика [*response data*].

Элементам в отклике следует соблюдать выбранный транспортный протокол. Как пример, в протоколе HTTP, код состояния и выражение причины появляются в строке состояния, заголовки отклика JPIP появляются в заголовках отклика HTTP, а данные отклика (если имеются) появляются в теле объекта HTTP.

```
Status-Code = 3DIGIT
Reason-Phrase = *<TEXT, excluding CR and LF>
jpip-response-header =
                      / JPIP-tid                    ; D.2.2
                      / JPIP-cnew                   ; D.2.3
                      / JPIP-qid                    ; D.2.4
                      / JPIP-fsiz                   ; D.2.5
                      / JPIP-rsiz                   ; D.2.6
                      / JPIP-roff                   ; D.2.7
                      / JPIP-comps                  ; D.2.8
                      / JPIP-stream                 ; D.2.9
                      / JPIP-context                ; D.2.10
                      / JPIP-roi                    ; D.2.11
                      / JPIP-layers                 ; D.2.12
                      / JPIP-srate                  ; D.2.13
                      / JPIP-metareq                ; D.2.14
                      / JPIP-len                    ; D.2.15
                      / JPIP-quality                ; D.2.16
                      / JPIP-type                   ; D.2.17
                      / JPIP-mset                   ; D.2.18
                      / JPIP-cap                    ; D.2.19
                      / JPIP-pref                   ; D.2.20
```

Строке выражения причины следует идеальным способом давать текстовое разъяснение кода состояния. Для приложений JPIP могут быть достаточными следующие коды состояний.

D.1.3 Коды состояний и выражения причин

D.1.3.1 Общие положения

●●● ●●●●●●●●● [*Status-Code*] является результирующим кодом целого числа из 3 цифр для попытки понять и удовлетворить запрос. Используется поднабор кодов состояний и выражений причин из протокола HTTP/1.1. Клиентам JPIP следует ожидать следующие коды. Клиенты JPIP, действующие поверх протокола HTTP, также могут видеть другие коды состояний.

D.1.3.2 200 (OK)

Серверу следует использовать этот код состояния, если он соглашается с запросом окна обзора для обработки, возможно, с некоторыми изменениями к запрашиваемому окну обзора, как указывается дополнительными заголовками, включенными в ответ.

D.1.3.3 202 (Принят)

Серверам следует выпускать этот код состояния, если запрос окна обзора был приемлемым, но в очереди был найден последующий запрос окна обзора, который в результате заменил запрос (поскольку `wait=no` [*ожидать=нет*]). Когда первый запрос становится неуместным прежде, чем сервер способен обработать и начать передачу отклика, тогда должен использоваться код состояния 202. Это является обычным случаем в практике, так как диалоговый пользователь может заменять его/ее область интереса множество раз прежде, чем сервер заканчивает отвечать на более ранний запрос, или прежде, чем сервер готов прервать продолжающуюся обработку.

D.1.3.4 400 (Плохой запрос)

Серверам следует выпускать этот код состояния, если запрос неправильно форматирован или содержит нераспознанное поле в строке вопроса.

D.1.3.5 404 (Не найден)

Этот код состояния следует выпускать, если сервер не может согласовать запрашиваемый ресурс с выпущенным идентификатором ID Цели. Это может следовать из неправомочных попыток доступа или, более вероятно, из истечения предела времени. Если клиент пропускает это окно времени, из-за плохого соединения, он может обнаружить, что идентификатор ID Цели больше не активен.

D.1.3.6 415 (Неподдерживаемый тип носителя информации)

Этот код состояния может быть использован, если отдельный тип изображения, указанный в поле запроса Типа возврата изображения, не может быть обслужен.

D.1.3.7 501 (Не осуществлен)

Этот код состояния может быть использован, если порция этой Рекомендации | Международного стандарта, требуемая запросом, не может быть обслужена.

D.1.3.8 503 (Услуга недействительная)

Этот код состояния следует использовать, если идентификатор канала, указанный в поле запроса ID Канала, является недействительным.

D.2 Заголовки откликов JPIP

D.2.1 Введение в заголовки откликов JPIP

В ответе на запрос клиента сервер может изменить некоторые аспекты запроса. Если сервер изменяет запрос, то измененные параметры должны опознаваться через заголовки откликов. Имя каждого заголовка отклика получается из имени области запроса, чьи параметры изменяются путем задания префиксов имени поля запроса с помощью "`JPIP-`". Пока не определено иначе, если параметры, обозначаемые в заголовке отклика, были первоначально определены в запросе клиента, тогда сервер ответил бы тем же самым способом, кроме отклика, который теперь не содержал бы эти заголовки откликов. Кроме того, заголовки откликов JPIP могут быть посланы сервером, чтобы информировать клиента о значениях других неуказанных полей запроса для использования в будущих запросах.

Отклик `JPIP-qid` является исключением в том, что он должен посылаться всякий раз, когда в запрос клиент включил идентификатор ID Запроса, и тогда значение `JPIP-qid` должно быть всегда тем же самым, как `qid`.

Параметры к полученному заголовку отклика, указанному тем же самым элементом BNF, что и параметры в первоначальном поле запроса, имеют то же самое значение и форматирование, как параметры к первоначальному полю запроса.

Единственные исключения к этому правилу находятся в соединении с заголовками откликов Нового канала и Качества.

D.2.2 Идентификатор ID Цели (JPIP-tid)

```
JPIP-tid = "JPIP-tid" ":" LWSP target-id
```

Сервер должен посылать этот заголовок отклика, если уникальный целевой идентификатор сервера отличается любым способом от идентификатора, поставляемого с полем запроса ID Цели, или если клиент не определил поле запроса ID Цели. Строка `target-id` является произвольной, назначенной сервером, с длиной, не превышающей 255 знаков. Если поле запроса ID Цели определяет значение "0", сервер обязан включить заголовок отклика ID Цели, указывая фактический идентификатор target-id. Если сервер не способен назначить уникальные идентификаторы к запрашиваемой логической цели, и, следовательно, не может гарантировать ее целостность между многократными запросами или сеансами, тогда заголовок отклика ID Цели должен указывать значение 0. Если сервер поставляет `target-id`, который отличается от определенного в запросе, он должен игнорировать все поля запросов `model`, `tpmodel`, `need` и `tpneed` при ответе на этот запрос.

D.2.3 Новый канал (JPIP-cnew)

```
JPIP-cnew = "JPIP-cnew" ":" LWSP "cid" "=" channel-id
            ["," 1#(transport-param "=" TOKEN)]
transport-param = TOKEN
```

Сервер должен посылать этот заголовок отклика, если и только если, он назначает новый канал в отклике на поле запроса Нового канала. Строка значения состоит из разделенного запятыми перечня пар name=value [*имя=значение*], первая пара которого обозначает маркер channel-id нового канала.

Этой Рекомендацией | Международным стандартом определяются следующие маркеры `transport-param` [*транспорт−параметр*] (см. таблицу D.1).

Таблица D.1 – Допустимые значения transport-param

Значение	Смысл
`"transport"`	Это параметр должен назначаться одному из значений в перечне приемлемых транспортных имен, поставляемых в поле запроса Нового канала. Если в поле запроса были поставлены многократные транспортные имена, то заголовок отклика должен обозначать фактические транспортные средства, которые будут использованы с каналом.
`"host"`[*ведущий узел*]	Этот параметр обозначает имя или адрес IP ведущего узла (хоста) для сервера JPIP, который управляет новым каналом. Параметр не нужно возвращать, если ведущий узел не отличается от того ведущего узла, к которому фактически был послан запрос.
`"path"`[*тракт*]	Этот параметр обозначает компонент тракта URL, подлежащий использованию в построении будущих запросов с этим каналом. Параметр не нужно возвращать, если имя тракта не отличается от того имени, которое было использовано в фактически посланном запросе.
`"port"` [*порт*]	Этот параметр обозначает числовой номер порта (десятичный), на котором сервер JPIP, что управляет новым каналом, прослушивает запросы. Параметр не нужно возвращать, если главный узел и номер порта идентичны тем, к которым был послан первоначальный запрос. Параметр также не нужно возвращать, если главный узел отличается от того, к которому был послан запрос, и подлежит использованию номер порта по умолчанию, связанный с соответствующим транспортным средством.
`"auxport"`	Это параметр используется с транспортными средствами, требующими второго физического канала. Если используется транспортное средство "http-tcp", то используется вспомогательный порт для подключения вспомогательного канала TCP. Для дальнейших подробностей, см. Приложение G. Параметр не нужно возвращать, если первоначальный запрос вовлекал канал, который также занимал вспомогательный канал, имея тот же самый номер вспомогательного порта. В противном случае, параметр нужно возвращать только в том случае, если номер вспомогательного порта отличается от значения по умолчанию, связанного с выбранным транспортным средством.

D.2.4 Идентификатор ID Запроса (JPIP-qid)

```
JPIP-qid = "JPIP-qid" ":" LWSP UINT
```

Сервер должен посылать этот заголовок отклика, если запрос клиента включил ID Запроса `qid`. Значение `JPIP-qid` должно быть идентично `qid`. Сервер не должен включать заголовок отклика ID Запроса, когда соответственный запрос клиента не включил в себя ID Запроса.

ПРИМЕЧАНИЕ. − Идентификатор ID Запроса сервера, `JPIP-qid`, всегда должен быть идентичен идентификатору ID Запроса клиента. Таким образом, идентификатор ID Запроса отличается тем, что этот заголовок отклика посылается, когда клиент использовал ID Запроса, а не тогда, когда сервер изменяет значение.

D.2.5 Размер кадра (JPIP-fsiz)

```
JPIP-fsiz = "JPIP-fsiz" ":" LWSP fx "," fy
```

Серверу следует посылать этот заголовок отклика, если размер кадра, для которого будут предоставлены данные отклика, отличается от кадра, запрашиваемого через поле запроса Размера кадра.

D.2.6 Размер области (JPIP-rsiz)

```
JPIP-rsiz = "JPIP-rsiz" ":" LWSP sx "," sy
```

Серверу следует посылать этот заголовок отклика, если размер области, для которого будут предоставлены данные отклика, отличается от запрашиваемого размера.

D.2.7 Смещение (JPIP-roff)

```
JPIP-roff = "JPIP-roff" ":" LWSP ox "," oy
```

Серверу следует посылать этот заголовок отклика, если смещение области, для которого будут предоставлены данные, отличается от запрашиваемого смещения.

D.2.8 Компоненты (JPIP-comps)

```
JPIP-comps = "JPIP-comps" ":" LWSP 1#UINT-RANGE
```

Серверу следует посылать этот заголовок отклика, если компоненты, для которых он будет предоставлять данные, отличаются от тех, которые запрашиваются через поле запроса Компонентов. Он не обязан посылать этот заголовок отклика, если запрашиваемые компоненты изображения не существуют внутри ни одного из запрашиваемых кодовых потоков.

D.2.9 Кодовый поток (JPIP-stream)

```
JPIP-stream = "JPIP-stream" ":" LWSP 1#(prefixed-range / sampled-range)
prefixed-range = "<" ctxt-id ":" ctxt-elt ">" sampled-range
ctxt-id = UINT
ctxt-elt = UINT
```

Серверу следует посылать этот заголовок отклика для информирования клиента кодового потока или кодовых потоков, для которых он будет предоставлять данные, если он не обслуживает данные в отклике ко всем кодовым потокам, запрашиваемым непосредственно через поле запроса любого Кодового потока, и ко всем кодовым потокам, запрашиваемым косвенно через поле запроса любого Контекста кодового потока. Серверу следует использовать синтаксис `prefixed-range` [*предварительно заданный−диапазон*], чтобы обозначать те кодовые потоки, для которых данные предоставляются в ответ на транслируемое поле запроса Контекста кодового потока. В этом случае, значение `ctxt-id` должно обозначать характерный `context-range` [*контекст−диапазон*] из поля запроса Контекста кодового потока, чья трансляция производит уместные кодовые потоки. Кроме того, значение `ctxt-elt` должно обозначать конкретный элемент внутри `context-range`, обозначаемый с помощью `ctxt-id`, чья трансляция производит уместные кодовые потоки.

Значение 0 для `ctxt-id` означает, что первый `context-range` в поле запроса Контекста кодового потока есть контекст, который произвел диапазон кодовых потоков, который сопровождает префикс. Точно так же значение 1 для `ctxt-id` означает, что второй `context-range` в поле запроса Контекста кодового потока есть контекст, который произвел следующий диапазон кодовых потоков, и т. д.

Значение 0 для `ctxt-elt` означает, что первый контекст в уместном контексте `context-range` есть контекст, который произвел диапазон кодовых потоков, сопровождающий префикс.

Пример:

Запрос клиента:

поток=0&context=jpxl<2-7:2>[s0i0],jpxl<3-5>[s1i3]

Отклик сервера:

JPIP-контекст: jpxl<2-7:2>[s0i0]=0,1;jpxl<9-10>[s1i3]=0

JPIP-поток: 0,<0:1>1,<1:0>0,<1:1>0

Это означает, что сервер откликается с помощью данных, появляющихся из:

1) прямого приложения окна обзора к кодовому потоку 0 (как запрашивается через "поток=0");
2) трансляции окна обзора к слою 4 наложения изображений JPX, согласно инструкции 0 наложения изображений в наборе 0 инструкций наложения изображений, как он применяется к кодовому потоку 1;
3) трансляции окна обзора к слою 9 наложения изображений JPX, согласно инструкции 3 наложения изображений в наборе 1 инструкций по наложению изображений, как он применяется к кодовому потоку 0; и
4) трансляции окна обзора к слою 10 наложения изображений JPX, согласно инструкции 3 наложения изображений в наборе 1 инструкций по наложению изображений, как он применяется к кодовому потоку 0.

D.2.10 Контекст кодового потока (JPIP-context)

```
JPIP-context = "JPIP-context" ":" LWSP 1$(context-range "=" 1#sampled-range)
```

Серверу следует посылать этот заголовок отклика, если он способен обрабатывать любое из значений `context-range`, поставляемое через поле запроса Контекста кодового потока. Заголовок описывает каждый `context-range`, который обрабатывается вместе с индексами всех кодовых потоков, связанных с таким `context-range`. Сервер может опустить некоторые значения `context-range`, которые были первоначально предоставлены в поле запроса Контекста кодового потока, если они не обрабатываются. Сервер может также изменить значения `context-range`, первоначально предоставленные в поле запроса Контекста кодового потока. Разрешаются два типа модификации:

a) сервер может ограничивать совокупность элементов изображений (например, слои наложения изображений), которые были первоначально запрошены;

b) сервер может сбрасывать модификаторы геометрических преобразований, которые он не способен поддерживать (например, модификатор "дорожка" или "фильм" внутри строки `mj2t-context`).

D.2.11 ROI (JPIP-roi)

```
JPIP-roi = "JPIP-roi" ":" LWSP
           "roi" "=" region-name ";"
           "fsiz" "=" UINT "," UINT ";"
           "rsiz" "=" UINT "," UINT ";"
           "roff" "=" UINT "," UINT ";"

region-name = 1*(DIGIT / ALPHA / "_")
```

В отклике на запрос клиента для ROI, сервер должен определять через заголовок отклика ROI степень фактически обслуживаемого ROI. Если сервер не способен выполнить запрос ROI, он должен откликнуться заголовком отклика ROI, просто установленным в положение: "JPIP-roi: roi=no-roi". В дополнение к ROI сервер через заголовки откликов Размера кадра, Размера области и Смещения также определяет область изображения, что он обслуживает в качестве резерва.

Если сервер способен обслуживать ROI, но по некоторым причинам должен изменять размеры порции возвращаемого изображения, он должен послать заголовок отклика ROI, описывающий ROI и заголовки откликов Размера кадра, Размера области и Смещения, описывающие часть возвращаемого ROI.

D.2.12 Слои (JPIP-слои)

```
JPIP-layers = "JPIP-layers" ":" LWSP UINT
```

Серверу следует посылать этот заголовок отклика, если количество слоев, которые он будет обслуживать, меньше, чем значение, указанное полем запроса слоев. Так как окно обзора обычно обслуживается способом последовательного качества, сервер не обязан (и действительно может быть не способен) определять количество слоев, которые охвачены данными откликов, которые он доставляет. Однако если требуемое количество слоев превышает количество слоев, имеющихся от любых кодовых потоков в окне обзора, серверу следует, по крайней мере, обозначить максимальное количество имеющихся слоев. Любой сервер, который соглашается с полем запроса Выравнивания (см. C.7.1), должен предоставлять отклик JPIP-слоев, если количество слоев, которые он будет обслуживать, меньше, чем значение, указанное полем запроса слоев.

D.2.13 Скорость дискретизации (JPIP-srate)

```
JPIP-srate = "JPIP-srate" ":" LWSP UFLOAT
```

Серверу следует посылать этот заголовок отклика, если средняя частота дискретизации кодовых потоков, которую он будет посылать клиенту, как ожидается, будет отличаться от той, которая запрашивается через поле запроса Частоты дискретизации, а частота дискретизации известна. Если исходные кодовые потоки не имеют информации синхронизации, этот заголовок отклика не следует посылать.

D.2.14 Запрос метаданных (JPIP-metareq)

```
JPIP-metareq = "JPIP-metareq" ":" LWSP
               1#( "[" 1$(req-box-prop) "]" [root-bin] [max-depth] )
               [metadata-only]

req-box-prop = box-type [limit] [metareq-qualifier] [priority]
```

Серверу следует посылать этот заголовок отклика, если он осуществляет видоизменение значений `max-depth`, `limit`, `metareq-qualifier` or `priority`, предоставляемых в поле запроса Метаданных.

D.2.15 Максимальная длина отклика (JPIP-len)

```
JPIP-len = "JPIP-len" ":" LWSP UINT
```

Серверу следует посылать этот заголовок отклика, если предел байта, определенный в поле запроса Максимальной длины отклика, был слишком мал, чтобы позволить непустой отклик, если предел байта не был равен нулю. Если он возвращается, `JPIP-len` должно быть значением, которое сообщает клиенту о подходящей максимальной длине ответа, `len`, для последующих запросов. Если `len=0`, серверу следует откликнуться на запрос заголовками откликов и без данных отклика.

D.2.16 Качество (JPIP-качество)

```
JPIP-quality = "JPIP-quality" ":" LWSP (1*2DIGIT / "100" / "-1")
```

Сервер может посылать этот заголовок отклика, чтобы сообщать клиенту значение качества, которое будет связано с данными изображения, возвращенными, как только этот запрос был закончен. Если запрос прерывается другим запросом (не имеющим "wait=yes" [*"ожидать=да"*]), это значение качества может не быть точным. Значение качества относится только к запрашиваемому окну обзора и имеет тоже же самое истолкование, как поле запроса Качества. Если сервер игнорировал запрос клиента, должно быть возвращено значение "-1".

D.2.17 Тип возврата изображения (JPIP-type)

```
JPIP-type = "JPIP-type" ":" LWSP image-return-type
```

Серверу следует включать этот заголовок отклика, если другой механизм не обозначает подтип MIME данных изображения возврата. Примеры других механизмов включают в себя:

- заголовок "Content-Type:" [*"Содержимое-тип"*] протокола HTTP,
- Отклики на запросы, которые связаны с сеансом, чей тип изображения возврата уже был сообщен.

D.2.18 Набор моделей (JPIP-mset)

```
JPIP-mset = "JPIP-mset" ":" LWSP 1#sampled-range
```

Серверу следует включать этот заголовок отклика, если запрос клиента содержит поле запроса Набора моделей, а совокупность кодовых потоков, обозначаемых полем запроса Набора моделей клиента, отличается любым способом от совокупности кодовых потоков, для которых сервер фактически готов сохранять информацию модели кэш-памяти. Набор кодовых потоков, для которых сервер сохраняет информацию модели кэш-памяти, должен включать в себя все кодовые потоки, которые связаны с данными отклика сервера (или те, что обозначены в запросе клиента, или те, что обозначены заголовком отклика Кодового потока сервера, если имеются). Кроме таких кодовых потоков, "mset" сервера может быть не больше, чем обозначено полем запроса Набора моделей.

D.2.19 Необходимая возможность (JPIP-cap)

```
JPIP-cap = "JPIP-cap" ":" LWSP 1#capability-code
```

Этот заголовок ответа определяет, что клиенту следует поддерживать конкретное свойство, чтобы истолковывать логическую цель в манере матрицы отображения. Допустимые возможности являются теми же, что определены для поля запроса Возможностей в таблице C.6.

D.2.20 Недействительное предпочтение (JPIP-pref)

```
JPIP-pref = "JPIP-pref" ":" LWSP 1#related-pref-set
```

Этот заголовок ответа следует предоставлять, если и только если, поле запроса Предпочтения клиента содержало `related-pref-set` с модификатором "/r" (требуемый), который сервер не желал поддерживать. В этом случае следует также возвращать значение ошибки для кода состояния отклика. Строка значения состоит из одного или более `related-pref-sets`, которые не могли быть поддержаны, повторенных точно в той же самой форме, как они появлялись в запросе Предпочтений клиентов.

Хотя желательно, но не является необходимым перечислять для этого заголовка отклика все из запрашиваемых `related-pref-sets`, что не могут быть поддержаны. Таким образом, для сервера дозволительно входить в поле запроса Предпочтений клиентов только до тех пор, пока он не сталкивается с `related-pref-set`, которое определяет "/r", и не может быть поддержано. См. подраздел C.10.2.1 для получения дополнительной информации о том, когда должен использоваться этот заголовок ответа.

Таблица D.2 – Коды определенных причин

Код причины	Причина	Пояснение
1	Изображение готово	Сервер перенес к клиенту всю имеющуюся информацию изображения (не только информацию, относящуюся к запрашиваемому окну обзора). Этот код причины имеет особый смысл для запросов на основе сеанса. Для запросов на основе сеанса этот код причины подразумевает, что клиент получил все данные, которые могли быть посланы в отклике на любой запрос на основе сеанса, связанный с этой логической целью. С возможным исключением запросов, которые включают в себя поля запросов для управления кэш-памятью, любой последующий запрос на основе сеанса будет получать отклик без данных отклика и `R=1 EOR`.
2	Окно готово	Сервер перенес всю имеющуюся информацию, относящуюся к запрашиваемому окну обзора. Этот код причины имеет особый смысл для запросов на основе сеанса. Для запросов на основе сеанса этот код причины подразумевает, что клиент получил все данные, которые могли быть посланы в отклике на этот запрос, и данные отклика не ограничены любым полем ограничения данных (`len` или `quality`) в запросе или обработкой последующего запроса. С возможным исключением запросов, которые включают в себя поля запросов для управления кэш-памятью, любое последующее повторение запроса будет получать отклик без данных отклика и `R=2 EOR`.
3	Изменение окна	Север завершает свой отклик, чтобы обслужить новый запрос, который не определяет `Wait=yes`.
4	Достигнут предел байта	Сервер завершает свой отклик, поскольку достигнут предел байта, определенный в поле запроса Максимальной длины отклика.
5	Достигнут предел качества	Сервер завершает свой отклик, поскольку достигнут предел качества, определенный в поле запроса Качества.
6	Достигнут предел сеанса	Сервер завершает свой отклик, поскольку достигнут некоторый предел на ресурсы сеанса, например, предел времени. Не следует выпускать дальнейший запрос, используя ID канала, связанный с таким сеансом.
7	Достигнут предел отклика	Сервер завершает свой отклик, поскольку достигнут некоторый предел, например, предел времени. Если запрос выпускается в сеансе, дальнейшие запросы могут еще выпускаться с использованием ID канала, связанного с таким сеансом.
0xFF	Неопределен-ная причина	Сервер завершает свой отклик по причине, которая не определена.
Другие значения		Зарезервировано для использования ИСО.

D.3 Данные отклика

Для любых типов возврата, отличающихся от типов возврата изображения JPP-потока или JPT-потока, включая необработанный кодовый поток, данным отклика следует полностью состоять из запрашиваемого объекта. Для типов возврата изображения JPP-потока или JPT-потока, данные отклика состоят из последовательности сообщений, как определено в Приложении A, завершаемой единичным сообщением EOR (Конец отклика). Сообщение EOR не определяется в Приложении A и формально не является частью типов носителей информации JPP-потока или JPT-потока.

Сообщение EOR состоит из заголовка и тела. Заголовок сообщения EOR состоит из однобайтового идентификатора, 0x00, сопровождаемого однобайтовым кодом причины, R, и затем отдельным подсчетом байтов VBAS, указывающим количество байтов в теле сообщения EOR. Эта Рекомендация | Международный стандарт не обеспечивает никакого нормативного истолкования для содержимого тела сообщения EOR.

Отметим, что тело сообщения EOR не вносит вклад в ограничение подсчета байтов, связанное с полем запроса Максимальной длины отклика, как определено в Приложении C.

Отметим, что сообщение EOR означает, что сервер доставил все подходящее содержимое уместных бункеров данных для запроса клиента. Это не обязательно полное содержание таких бункеров данных. Отклик завершается, когда достигнут предел, определенный клиентом. Если предел не был определен, то тогда сообщение EOR подразумевало бы, что подано все содержимое уместных бункеров данных.

Коды причин в настоящее время определены (см. таблицу D.2).

Приложение E

Загрузка изображений в сервер

(Это приложение образует составную часть этой Рекомендации | Международного стандарта)

E.1 Введение

Ожидается, что изображения будут помещаться в сервер разнообразными способами, находящимися вне сферы действия этой Рекомендации | Международного стандарта. Цель этого приложения состоит в том, чтобы описать механизм, который позволяет загружаться в сервер частям изображения.

E.2 Запрос загрузки

E.2.1 Структура запроса

Запрос загрузки в сервер состоит из одного или более полей запроса, определенных в Приложении C, и тела запроса.

E.2.2 Поля запроса загрузки

Поля запроса для загрузки должны содержать поле запроса Загрузки [*Upload*]. Могут также использоваться поля запросов Цели, Подцели и ID Цели (см. C.2.2, C.2.3 и C.2.4). Для загрузки типа носителя информации полного изображения используются поля запросов Размера кадра, Смещения и Размера области (см. C.4.2, C.4.3 и C.4.4) для указания положения загружаемой части внутри полного изображения. Для загрузок JPT-потока и JPP-потока не являются необходимыми количество бункеров данных (и, следовательно, количество элементов мозаичного изображения или зон) наряду с главным заголовком для указания местоположения закодированных данных и поля запроса окна обзора.

E.2.3 Тело запроса загрузки

E.2.3.1 Общие положения

Тело запроса загрузки состоит из одного из поддерживаемых типов изображений: JPP-поток, JPT-поток или тип носителя информации полного изображения. Тело содержит данные, которые клиент запрашивает для обработки сервером. Эта Рекомендация | Международный стандарт не поддерживает загрузку необработанных данных изображения.

E.2.3.2 JPT-поток

Тело запроса содержит все бункеры данных, которые клиент хочет, чтобы сервер заменил (бункеры данных заголовка, бункеры метаданных и бункеры данных элементов мозаичного изображения). Если клиент не загружает бункер данных главного заголовка, то бункеры данных элементов мозаичного изображения должны кодироваться совместимым способом с текущим главным заголовком.

E.2.3.3 JPP-поток

Тело запроса содержит все бункеры данных, которые клиент хочет, чтобы сервер заменил (бункеры данных заголовка, бункеры данных элементов мозаичного изображения, бункеры метаданных и бункеры данных зон). Если клиент не загружает бункер данных главного заголовка или бункер данных элементов мозаичного изображения, то зоны должны кодироваться совместимым способом с текущим главным заголовком и заголовками элементов мозаичного изображения.

E.2.3.4 Загрузка полного изображения

Тело запроса содержит тип носителя информации полного изображения, представляющий те отсчеты, которые клиент желает изменить.

В случае загрузки полного изображения запрос может включать в себя поля запросов Размера кадра, Размера области и Смещения. Поле запроса Размера кадра должно быть размером эталонной сетки изображения. В случае загрузки полного изображения, сжатие не должно осуществляться совместимым способом с логической целью на сервере. Если размер загружаемого изображения превышает предел в поле запроса Размера области, то серверу следует ограничить изменения до предела, указанного в поле запроса Размера области.

E.3 Отклик сервера

E.3.1 Общие положения

Серверу следует откликаться на запрос загрузки с помощью кода состояния и выражения причины из Приложения D. Полезные коды возврата и выражения причин для загрузки изображения в сервер представляются в следующих подразделах.

E.3.2 201 (Созданный)

Серверу следует использовать этот код состояния, если, после получения запроса на загрузку, на сервере был определен новый ресурс. Сервер должен завершить создание перед возвращением этого запроса. Если будет задержка, то сервер должен возвратить код 202 (Принятый) [*Accepted*] вместо кода 201 (Созданный) [*Created*].

Серверу следует включать заголовок с откликом с новым полем ID цели для обновленного ресурса.

Нет необходимости возвращать тело.

E.3.3 202 (Принятый)

Серверу следует использовать этот код состояния, если загрузка в сервер создает новый ресурс, но сервер еще не готов ее обслуживать. Сервер может также использовать этот код состояния для обновления текущего ресурса.

E.3.4 400 (Плохой запрос)

Серверам следует выпускать этот код состояния, если запрос неправильно форматирован, или если вопрос содержит поля запросов, которые являются несовместимыми с загрузкой, или содержит нераспознанное поле в строке вопроса.

E.3.5 404 (Не найден)

Этот код состояния следует выпускать, если сервер не может согласовать запрашиваемый ресурс с выпущенным идентификатором ID цели.

E.3.6 415 (Не поддерживаемый тип носителя информации)

Этот код состояния может быть выпущен для указания, что в то время как загрузки в сервер поддерживаются, загрузки конкретного типа (например, полное изображение, JPT-поток или JPP-поток), включенные с помощью запроса, не поддерживаются.

E.3.7 501 (Не осуществлено)

Это состояние могло бы использоваться, если сервер не поддерживает загрузку или не поддерживает конкретный вариант с загрузкой.

E.4 Слияние данных на сервере

E.4.1 Обновление изображения

После получения загруженных данных сервер может создавать новую версию логической цели и предоставлять новую версию клиентам, обращающимся к новому или старому указателю URL. Однако сервер не должен использовать поле запроса старого идентификатора ID Цели для обеспечения доступа к любым слившимся или обновленным данным.

Если клиент включает поле запроса ID Цели в запрос загрузки, а такой ID цели не соответствует текущему ID цели сервера для ресурса, серверу не следует обновлять изображение. Это несоответствие может указывать, что клиент редактировал предыдущую версию изображения, которая уже была изменена. Серверы могут отказаться принимать загрузки, которые не содержат поле запроса ID Цели. Это является одним способом предотвращения многократных одновременных изменений цели различными клиентами. Серверы, обеспечивающие возможности редактирования, могут позаботиться о таких проблемах, как захват цели некоторыми другими средствами.

Клиент JPIP может загружать часть нового изображения в сервер, определяя ID цели как 0, или используя новый указатель URL, или цель, которую сервер не имеет. Серверу следует для загрузки выпускать ID цели. Клиент может продолжать загрузку дополнительных частей нового изображения, используя идентификатор ID цели, возвращенный сервером с предыдущей загрузкой.

E.4.2 JPT-поток

Сервер, принимающий данные бункера данных элементов мозаичного изображения, должен сначала удалить все старые данные бункера данных элементов мозаичного изображения для таких загружаемых элементов мозаичного изображения, а затем включить в кодовый поток новые данные бункера данных элементов мозаичного изображения. Не может быть выполнено обновление, которое приводит к изменению в количестве, величине или местоположении элементов мозаичного изображения: структура изображения не может быть изменена путем загрузки. В частности, серверу не следует допускать загрузок бункеров данных элементов мозаичного изображения для кодового потока, содержащего сегмент маркера PPM в главном заголовке, если с помощью загрузки клиент не обеспечивает новый главный заголовок. Любые сегменты маркеров PLM или TLM должны быть удалены или обновлены. Бункер данных главного заголовка потока JPT должен быть загружен для новых изображений.

Как формируются части элемента мозаичного изображения кодового потока из бункера данных элемента мозаичного изображения, не определяется. Клиенту не нужно обязательно обеспечивать все части элементов мозаичного изображения и не нужно завершать последнюю часть элемента мозаичного изображения. Сервер должен обновлять главный заголовок и любые затронутые части файлового формата (например, длина блока кодового потока).

При слиянии данных количество или размер элементов мозаичного изображения не должны быть изменены, а данные, которые не заменяются процессом загрузки в сервер, должны иметь тот же самый смысл, какой они первоначально имели перед загрузкой.

E.4.3 JPP-поток

Сервер, принимающий сообщения бункеров данных зон, должен сначала удалить соответствующие старые бункеры данных зон для таких загружаемых зон, а затем включить новые данные бункера данных зоны. Изменение не может быть выполнено для заголовка, что приводит к изменению количества зон, или смысла идентификатора зоны, или местоположения, или размера каждой зоны внутри ее разрешающей способности компонента элемента мозаичного изображения. Бункеры данных заголовков элементов мозаичного изображения JPP-потока и бункеры данных главных заголовков должны быть загружены в сервер для новых изображений.

Как формируются пакеты зон из бункера данных зоны, не определяется. Клиенту не нужно обязательно обеспечивать все пакеты зоны или даже завершать последний предоставляемый пакет.

При слиянии данных количество или размер зон не должны быть изменены, а данные, которые не заменяются процессом загрузки, будут иметь тот же самый смысл, какой они первоначально имели перед загрузкой.

E.4.4 Бункеры метаданных JPP-потока и JPT-потока

Бункер метаданных может быть загружен в сервер, заменяя содержимое в существующем бункере метаданных. Так как сервер имеет управление разделением распределения метаданных в бункерах метаданных, клиент должен следовать за структурой бункера метаданных сервера. Клиент не должен изменять указатели мест заполнения в бункере метаданных, кроме полного удаления указателя места заполнения. При загрузке полного бункера метаданных клиенты могут прибавлять новые метаданные, добавляя к концу старого бункера метаданных, или вставляя новые метаданные между блоками в старом бункере метаданных. Сервер должен управлять указателями мест заполнения и структурой бункера метаданных. Это включает обновление всех указателей мест заполнения, указывающих на любые отмершие блоки метаданных, которые были изменены или затронуты заменой. Сервер должен удалять любые блоки метаданных, которые были указаны указателем места заполнения, который клиент удалил. Сервер может повторно систематизировать метаданные после того, как загрузка принимается, но прежде, чем создается новый ресурс. Если после загрузки в файле остаются неиспользованные секции, то для заполнения этих секций должны использоваться блоки Свободные [*Free*].

E.4.5 Загрузка полного изображения

В случае приемлемой загрузки полного изображения серверу следует распаковать (если требуется) загруженное под-изображение, распаковать некоторую порцию полного изображения на сервере, заменить те пиксели (точки растра) в (распакованном) пространственном домене и повторно сжать все элементы мозаичного изображения или зоны, затронутые операцией обновления.

ПРИМЕЧАНИЕ. − Этот метод требует большего количества вычислений на сервере; однако он устраняет возможность того, что клиент будет использовать данные сжатого изображения несовместимым способом (например, неправильное количество уровней преобразования элементарных волн).

Приложение F

Использование протокола JPIP поверх протокола HTTP

(Это приложение образует составную часть этой Рекомендации | Международного стандарта)

F.1 Введение

Это приложение определяет метод использования протокола JPIP с протоколом HTTP и для запросов, и для откликов. Параметры запроса JPIP из Приложения C пакетируются в допустимые структуры запросов HTTP. Отклики сервера (включая коды состояния, заголовки, сообщения и коды откликов) из Приложения D пакетируются в допустимые отклики HTTP. Все запросы и отклики должны кодироваться, как определено стандартом HTTP.

Отметим, что текст и примеры в этом приложении описывают использование протокола JPIP поверх протокола HTTP. Ожидается, что идентичное связывание может использоваться для безопасного протокола HTTP.

F.2 Запросы

F.2.1 Введение запросов

Приложение C определяет поля запросов. При транспортировке через HTTP запрос JPIP может появляться как строка вопроса для запроса "GET" ["*ПОЛУЧИТЬ*"] протокола HTTP или как тело запроса "POST" ["*ОБЪЯВИТЬ*"] протокола HTTP. Поскольку некоторые системы HTTP ограничивают длину строки вопроса, предоставляемой в запросе "GET", запрос "POST" является предпочтительным для продолжительных запросов JPIP.

ПРИМЕЧАНИЕ 1. – Запрос протокола HTTP определяется в секции 5 документа RFC 2616 следующим образом:

```
Request = Request-Line               ; HTTP Section 5.1
          0*(( general-header        ; HTTP Section 4.5
            / request-header         ; HTTP Section 5.3
            / entity-header ) CRLF)  ; HTTP Section 7.1
           CRLF
           [ message-body ]          ; HTTP Section 4.3
```

ПРИМЕЧАНИЕ 2. – Запросы протокола HTTP `Request-Line` [*Запрос-строка*] и `Request-URI` определяются таким образом:

```
Request-Line = Method SP Request-URI SP HTTP-Version CRLF

Request-URI = "*" / absoluteURI / abs_path / authority
```

ПРИМЕЧАНИЕ 3. – Документ RFC 2396 определяет:

```
absoluteURI   = scheme ":" ( hier_part / opaque_part )

hier_part     = ( net_path / abs_path ) [ "?" query ]

abs_path      = "/"  path_segments
```

F.2.2 Запросы GET

Запрос JPIP может быть предоставлен серверу в качестве запроса HTTP. Для запроса "GET" запрос HTTP ограничивается следующим образом:

- "Method" ["*Метод*"] должен быть "GET".
- "query" [*"вопрос"*] должен быть нулем или более `jpip-request-field`, отделенных с помощью '&'.

Примером запроса JPIP, пакетированного в запросе "GET" протокола HTTP, есть:

```
GET /images/kids.jp2?rsiz=640,480&roff=320,240&fsiz=1280,1024 HTTP/1.1

Host: get.jpeg.org

CRLF
```

Эквивалентный пример, использующий `absoluteURI` вместо `abs_path`, есть:

```
GET http://get.jpeg.org/images/kids.jp2?rsiz=640,480&roff=320,240
&fsiz=1280,1024 HTTP/1.1

CRLF
```

ПРИМЕЧАНИЕ. – Эта Рекомендация | Международный стандарт не налагает ограничения на компонент схемы absoluteURI.

F.2.3 Запросы POST

Запрос JPIP может быть предоставлен серверу пакетированным в запросе "POST" протокола HTTP. Для запроса "POST" запрос HTTP ограничивается следующим образом:

- "Method" должен быть "POST".
- "entity-body" ["*объект-тело*"] должно быть нулем или более `jpip-request-field`, отделенным с помощью '&'.
- Строке заголовка "Content-type:" [*"содержимое-тип"*] следует быть включенной как "entity-header" ["*объект-заголовок*"] и содержать значение "application/x-www-form-urlencoded".

Пример запроса JPIP, пакетированного в запросе "POST" протокола HTTP, есть:

```
POST /cgi-bin/j2k_server.cgi HTTP/1.1
Host: post.jpeg.org
Content-type: application/x-www-form-urlencoded
Content-length: 62
CRLF
target=/images/kids.jp2&rsiz=640,480&roff=320,240&fsiz=1280,1024
```

F.2.4 Запросы загрузки

Запрос загрузки является допустимым запросом HTTP, ограниченным следующим образом:

- "Method" должен быть "POST".
- Указатель URL должен содержать поле вопроса загрузки.
- Content-type [*Содержимое-тип*] должен быть типом тела изображения: image/jpt-stream, image/jpp-stream, или типом носителя информации полного изображения.

Пример запроса загрузки JPIP есть:

```
POST /images/kids.jp2?rsiz=640,480&roff=320,240&fsiz=1280,1024 HTTP/1.1
Host: post.jpeg.org
Content-type: image/jpt-stream
CRLF
```

F.3 Установление сеанса

Сеанс на основе сеанса HTTP устанавливается с использованием поля запроса Нового канала со значением "http", т. е. "cnew=http" как часть запроса. Этот запрос обычно доставляется протоколом HTTP. Запрос может содержать запрос окна обзора, который становится первым запросом в новом канале. Отклик на этот запрос возвращается на том же самом соединении, на котором был сделан запрос.

Клиент может открыть соединение HTTP и выпустить запрос, который включает заголовок HTTP "Connection: keep-alive." [*"Соединение: сохранять-действующим"*] Это полезно для эффективных сеансов, но оно не является необходимым и не является достаточным, чтобы иметь сеанс. Отдельное соединение HTTP может использоваться для трафика для различных целей, различных каналов или даже трафика, не соответствующего протоколу JPI, например, запросы о файлах HTML. Запрос JPIP, который является частью сеанса, может прибывать на соединениях HTTP, отличающихся от соединения HTTP, используемого для запроса и выпуска нового канала, хотя это не поощряется.

F.4 Отклики

F.4.1 Введение

Каждый компонент отклика из Приложения D может быть пакетирован как часть допустимого отклика HTTP.

ПРИМЕЧАНИЕ. – Элемент `Response` [*отклик*] протокола HTTP определяется в секции 6 документа RFC 2616:

```
Response = Status-Line                    ; HTTP Section 6.1
           0*(( general-header            ; HTTP Section 4.5
             / response-header            ; HTTP Section 6.2
             / entity-header ) CRLF)      ; HTTP Section 7.1
            CRLF
            [ message-body ]              ; HTTP Section 7.2
```

Отклики JPIP, транспортируемые поверх протокола HTTP, должны быть допустимыми откликами HTTP, с дальнейшими ограничениями на некоторые части отклика HTTP, как описывается в следующих подразделах.

F.4.2 Код состояния и выражение причины

Все из кодов состояний, перечисленных в подразделе D.1.3, могут использоваться непосредственно как коды состояний HTTP. Кроме того, сервер, обеспечивающий протокол JPIP поверх протокола HTTP, может использовать любой код состояния HTTP, считающийся полезным, например, 402.

Все значения для Reason-Phrase [*Причина-выражение*], обеспеченного в подразделе D.1.3, могут использоваться непосредственно как Reason-Phrase протокола HTTP. Reason-Phrase должен быть соответствующим для кода состояния. Сервер, обеспечивающий протокол JPIP поверх протокола HTTP, может использовать любой Reason-Phrase протокола HTTP, считающийся полезным, например, Payment required [*требуемая Оплата*].

F.4.3 Информация заголовка

F.4.3.1 Заголовки JPIP

Строки заголовков из раздела D.2 должны быть включены в качестве "entity-header" в отклик HTTP без изменения.

F.4.3.2 Использование заголовка Accept протокола HTTP

Сервер, обеспечивающий протокол JPIP поверх протокола HTTP, может использовать строку заголовка "Accept" [*"Принимать"*] HTTP, найденную в запросе для определения типа отклика JPIP. Если запрос содержит параметр вопроса "type="[*тип*], то типом возвращения должен быть один из типов, перечисленных в параметре типа. Если запрос содержит и параметр вопроса "type=", и строку заголовка "Accept:", сервер может использовать приоритеты, указанные в строке "Accept:", чтобы выбирать между типами, указываемыми в параметре вопроса "type=". Если параметр вопроса "type=" не присутствует в запросе, сервер может выбирать тип возвращения, поддерживаемый основным JPIP сервером из перечня типов в "Accept:".

F.4.3.3 Использование заголовка Cache-Control

Отметим, что кэш-память в серверах-посредниках HTTP отличается от кэш-памяти и модели кэш-памяти в JPIP.

Любой запрос JPIP с полем запроса Нового канала является частью сеанса, и такие отклики в общем случае не могут быть занесены в кэш-память серверами-посредниками HTTP. Подобным образом любой отклик, который включает в себя заголовок отклика Нового канала, также является частью сеанса. В обоих случаях отклику сервера следует включать в себя строку заголовка "Cache-Control:" [*кэш-память - управление*] протокола HTTP со значением "no-cache" [*нет кэш-памяти*].

F.4.3.4 Использование заголовка Content-type

Серверу, обеспечивающему протокол JPIP поверх протокола HTTP, следует включать строку заголовка "Content-type:"[*содержимое-тип*], указывающую тип данных в теле, наиболее обычным это есть изображение/jpip-поток [*image/jpp-stream*] или изображение/jpt-поток [*image/jpt-stream*].

F.4.3.5 Использование заголовка Redirect

Заголовок Redirect [*Переадресовать*] протокола HTTP может быть полезным для информирования клиента о том, что ресурсы перемещены или к ним следует получать доступ от другого узла.

Отметим, что отклик JPIP также определяет способ переадресования. Отклику JPIP следует оказывать предпочтение внутри сеанса.

F.4.4 Тело

Сообщения из Приложения D должны быть включены как тело отклика HTTP. Отметим, что отклик HTTP должен иметь механизм для определения длины отклика. Если сервер не планирует прерывать отклик, он может предоставить эту информацию со строкой заголовка "Content-Length" [*содержимое-длина*]. Предпочтительный метод обеспечения длины состоит в том, чтобы использовать строку заголовка "Transfer-Encoding: chunked" [*перенос-кодирование: фрагментированные данные*] протокола HTTP, а затем обеспечивать тело во фрагментах данных с размером, определяемым сервером и указываемым перед каждым фрагментом данных. Указание конца отклика путем закрытия соединения HTTP не поощряется.

F.5 Дополнительные свойства протокола HTTP

F.5.1 Использование метода HEAD протокола HTTP

От клиентов и серверов JPIP не требуется использовать или поддерживать метод "HEAD" [*Головная часть*] протокола HTTP. Сервер, выбирающий осуществление метода "HEAD", должен сделать так, как определено в Секции 9.4 документа RFC 2616. В частности, "Метод HEAD идентичен методу GET, за исключением того, что сервер не должен возвращать тело сообщения в отклике".

Для клиентов может оказаться полезным выпускать запросы "HEAD" протокола HTTP как средства для определения, будет ли сервер изменять любой из параметров запроса, как определено в Приложении D. Клиентам не следует выпускать запрос "HEAD" протокола HTTP с полями вопросов модели кэш-памяти, поскольку это может заставить сервер обновить свою модель кэш-памяти.

Отметим, что клиент, желающий обновить модель кэш-памяти сервера без получения отклика, может использовать поле запроса Максимальной длины отклика.

Серверы могут отклонять любые или все запросы "HEAD". В отличие от типичных запросов "HEAD" протокола HTTP, которые для выполнения требуют относительно малого усилия для сервера, некоторые реализации сервера JPIP могли бы получать данные от нескольких местоположений в логической цели, вычислять характер отклика, а затем сбрасывать тело отклика, чтобы отвечать на запрос "HEAD".

F.5.2 Использование метода OPTIONS протокола HTTP

От клиентов и серверов JPIP не требуется использовать или поддерживать метод "OPTIONS" [*варианты*] протокола HTTP.

F.5.3 Использование Etag

Отметим, что протокол HTTP определяет механизм ярлыка объекта (ETag), который подобен полю запроса ID Цели JPIP в том, что он используется для обозначения изменений в ресурсе. Если и ярлык объекта, и ID цели связываются с ресурсом, то рекомендуется, чтобы ETag, определенный протоколом HTTP, изменялся всякий раз, когда изменяется `target-id`.

F.5.4 Использование кодирования переноса фрагментированных данных

Поскольку отклики, содержащие сжатые данные, могут быть очень большими и таким образом занимать долгое время для передачи, важно быть способным останавливаться в середине передачи. Если не определяется "Transfer-Encoding: chunked", запросы HTTP должны определять полную длину тела в заголовке "Content-Length:" или указывать конец данных, закрывая соединение. Ни один из этих вариантов не является желательным в диалоговом протоколе, так как может быть необходимым останавливать текущий отклик и посылать больше данных на том же самом соединении для нового отклика.

ПРИМЕЧАНИЕ 1. − Секция 19.4.6 документа RFC 2616 предоставляет алгоритм для удаления кодирования переноса фрагментированных данных.

ПРИМЕЧАНИЕ 2. − Кодирование переноса фрагментированных данных может быть полезным с протоколом JPIP при доставке поверх протоколов, отличающихся от протокола HTTP.

F.6 Протокол HTTP и поле запроса длины (информативное)

С каналом возврата HTTP сервер не получает непрерывную обратную связь от клиента и может легко проталкивать множество данных в программный канал, которые должны быть полностью получены прежде, чем какие-либо данные для нового окна могут быть обработаны. Чтобы поддерживать способность к реагированию, клиентам следует использовать поле запроса Максимальной длины отклика для регулирования потока трафика и, следовательно, поддерживать способность к реагированию. В общем случае клиенты будут нуждаться в осуществлении своих собственных алгоритмов управления потоком, чтобы регулировать длину запроса применительно к изменяющимся сетевым условиям.

Приложение G

Использование JPIP с запросами HTTP и возвратами TCP

(Это приложение образует составную часть этой Рекомендации | Международного стандарта)

G.1 Введение

Сам по себе протокол JPIP нейтрален относительно основных транспортных механизмов для запросов клиентов и ответов серверов, кроме запросов каналов, представленных полем запроса Нового канала ("cnew") (см. C.3.3) и заголовком отклика Нового канала ("JPIP-cnew") (см. D.2.3), где должны быть переданы подробности, зависящие от транспортных средств. Эта Рекомендация | Международный стандарт определяет два конкретных транспортных средства, которые обозначаются строками "http" и "http-tcp" в строке значения, связанной с запросами Нового канала. Это приложение обеспечивает подробности второго транспортного средства, которое должно быть обозначено в этом тексте как HTTP-TCP. Первое транспортное средство обозначается в этом тексте как HTTP и описывается в Приложении F.

Транспортное средство HTTP-TCP использует точно те же самые механизмы, как и транспортное средство HTTP, чтобы посылать запросы клиентов серверу и получать заголовки ответов серверов и коды состояний. Однако данные отклика сервера (не заголовки откликов) поставляются по вспомогательному соединению TCP. Информация, транспортируемая на этом вспомогательном соединении TCP, идентична той, которая транспортировалось бы как тело объекта чистого отклика HTTP, за исключением того, она обрамляется во фрагменты данных [*chunks*], каждый из которых имеет порядковый номер фрагмента данных.

Клиент явным образом подтверждает прибытие каждого фрагмента данных, посылая его порядковый номер обратно к серверу на тракте возвращения вспомогательного соединения TCP. Одна из принципиальных выгод от транспортного средства HTTP-TCP заключается в том, что сервер получает возрастающее уведомление о прибытии своих фрагментов данных откликов через этот механизм подтверждения клиента. Это позволяет серверу управлять потоком данных таким способом, чтобы поддерживать способность к реагированию и сетевую эффективность.

Все запросы, посланные через транспортное средство HTTP, должны кодироваться так, как определено стандартом HTTP.

G.2 Запросы клиентов

Запросы поставляются на первичном канале точно так же, как запросы HTTP. Они имеют точно ту же самую форму, как запросы, выпущенные по каналу, что использует транспортное средство HTTP, описанное в Приложении F. В частности, могут быть использованы и запрос "GET", и запрос "POST" протокола HTTP.

G.3 Установление сеанса

G.3.1 Установление канала

Новый канал может быть установлен к серверу JPIP путем выпуска запроса, который включает в себя поле запроса Нового канала (см. C.3.3). Как пример, такой запрос мог быть выпущен, используя протокол HTTP, хотя он мог бы также выпущен к характерному для JPIP серверу, используя любой подходящий транспортный механизм. Если отклик сервера (через заголовок отклика Нового канала в D.2.3) указывает, что новый канал был создан для работы с транспортным средством HTTP-TCP, клиент должен установить вспомогательное соединение TCP, используя номер вспомогательного порта, возвращаемый через заголовок отклика Нового канала. Более того, запрос, который включал в себя поле запроса Нового канала, затем обрабатывается так, как если бы он был выпущен внутри недавно созданного транспортного канала HTTP-TCP, означая, что данные отклика, порожденные таким запросом, должны быть возвращены через вспомогательное соединение TCP, как только оно было подключено.

Для установления вспомогательного соединения TCP клиент выпускает к ведущему узлу сервера запрос соединения TCP, обозначаемый через заголовок отклика Нового канала, на порте, определяемом заголовком отклика Нового канала. Клиент затем немедленно посылает отдельную строку текста кода ASCII, состоящую из строки нового идентификатора id канала, сопровождаемую двумя последовательными парами CR-LF. Это является единственной информацией, ориентированной на текст, доставляемой по вспомогательному соединению TCP.

Клиент затем ожидает получения данных отклика сервера по вспомогательному соединению TCP. Эти данные отклика не могут быть пустыми, так как каждый запрос, выпускаемый внутри транспортированного канала HTTP-TCP, должен иметь поток данных отклика, который состоит, по крайней мере, из сообщения EOR (см. D.3). См. раздел G 4 для более подробной информации об этом.

G.3.2 Формирование кадров сервера для данных отклика

Все данные отклика, посланные сервером через вспомогательное соединение TCP, должны быть обрамлены во фрагменты данных. Каждый фрагмент данных состоит из 8-байтового заголовка фрагмента данных, сопровождаемого телом фрагмента данных, которое удерживает данные отклика сервера, как показано на рисунке G 1. Первое 2-байтовое слово заголовка фрагмента данных содержит незнаковое целое число с обратным порядком байтов, представляющее полную длину фрагмента данных, включая само слово длины. Содержимое остающихся 6 байтов заголовка фрагмента данных не определяется этой Рекомендацией | Международным стандартом. Оно может использоваться для дополнительной передачи сигнализации, характерной для сервера. Клиент будет возвращать полный 8-байтовый заголовок фрагмента данных в своих сообщениях подтверждений фрагментов данных.

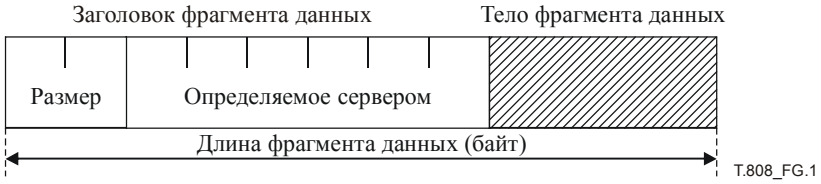

Рисунок G.1 − Структура данных отклика на соединении http-tcp

G.3.3 Подтверждение клиента из фрагментов данных откликов сервера

По получении фрагмента данных отклика сервера на вспомогательном соединении TCP клиент должен послать 8-байтовый заголовок фрагмента данных обратно к серверу как не имеющий кадров поток данных, используя обратный тракт соединения TCP. Каждый полученный фрагмент данных должен быть подтвержден в последовательности.

G.4 Отклики серверов

В отклике на каждый запрос клиента сервер посылает параграф ответа HTTP обратно к клиенту по первичному каналу. Параграф ответа содержит код состояния, выражение причины и все уместные заголовки откликов JPIP, а также любые соответствующие заголовки откликов HTTP. Однако через первичный канал никакие данные отклика не возвращаются. По этой причине не должно быть никакого тела объекта HTTP в отклике HTTP-TCP. Не должен также использоваться ни заголовок отклика"Content-length:", ни заголовок отклика "Transfer-encoding:".

Данные самого отклика доставляются по вспомогательному каналу TCP, обрамленные во фрагменты данных способом, описанным в подразделе G.3.2. Так как транспортное средство HTTP-TCP может использоваться только с сеансами, и, следовательно, только с типами возврата изображения JPP-потока и JPT-потока, данные отклика неизменно состоят из последовательности сообщений JPP-потока или JPT-потока.

Данные отклика, являющиеся результатом каждого запроса, должны состоять из целого числа фрагментов данных, означая, что никакой фрагмент данных не может содержать данные отклика, произведенные в отклике на два различных запроса.

Отклик на всякий и каждый запрос должен быть завершен сообщением EOR (см. раздел D.3), даже если данные отклика в противном случае были бы пусты. Сообщение EOR рассматривается как часть данных отклика и обрамляется во фрагменты данных наряду с фактическими сообщениями JPP-потока и JPT-потока.

Это означает, что каждый запрос, выпущенный на канале JPIP, транспортированном на HTTP-TCP, приводит к порождению, по крайней мере, одного непустого фрагмента данных отклика от сервера, и что последний фрагмент данных, произведенный в отклике на каждый запрос, заканчивается сообщением EOR.

Отметим, что нет никакого фактического требования для транспортируемых фрагментов данных откликов HTTP-TCP по выравниванию на границах сообщений.

G.5 TCP и поле запроса длины (информативное)

Может иметься несущественная причина или отсутствовать причина для использования поля запроса Максимальной длины отклика с каналом возвращения TCP, где сервер способен тщательно регулировать поток данных отклика к клиенту, чтобы поддерживать способность к реагированию.

Приложение H

Использование JPIP с альтернативными транспортными средствами

(Это приложение не образует составную часть этой Рекомендации | Международного стандарта)

H.1 Введение

Эта Рекомендация | Международный стандарт не определяет никакой другой определенный транспортный протокол, отличающийся от транспортного средства "`http`", описанного в Приложении F, и от транспортного средства `http-tcp`", описанного в Приложении G. Цель этого приложения состоит в том, чтобы обеспечить рекомендации по развертыванию протокола JPIP поверх ненадежных транспортных средств и предоставить общий подход, который может быть применен к широкому разнообразию транспортных средств.

В развитии общего подхода полезно делить аспекты связи на два логических транспортных соединения, называемые "соединение запроса" [*request connection*] и "соединение данных" [*data connection*]. Каждое логическое соединение, как понимают, обеспечивает и прямой тракт связи, и обратный тракт связи. Роли, играемые этими трактами, следующие:

- Прямой тракт соединения запроса используется для доставки запросов JPIP от клиента к серверу.
- Обратный тракт соединения запроса используется сервером для подтверждения получения запросов и заголовков откликов возврата к клиенту.
- Прямой тракт соединения данных используется для доставки сообщений потоков JPIP от сервера к клиенту.
- Обратный тракт соединения данных используется клиентом для подтверждения получения сообщений потоков JPIP от сервера.

Читатель заметит, что эти роли совместимы с теми, которые обслуживаются прямым и обратным трактами связи двух каналов TCP, используемых транспортным средством "http-tcp, которое описано в Приложении G. Действительно, материал в этом приложении может истолковываться как расширение транспортного средства "http-tcp" на ненадежные транспортные средства. Отметим, однако, что хотя это приложение описывается в понятиях двух различных логических соединений, нет никакой причины, почему связь нельзя осуществлять по отдельному транспортному соединению.

Наконец, предполагается, что каждое логическое соединение обеспечивает один из следующих двух типов услуг:

a) Такую надежную услугу, ориентированную на поток, как услуга, предлагаемая протоколом TCP.

b) Такую ненадежную услугу, ориентированную на передачу пакетов, как услуга, предлагаемая протоколом UDP. В этом случае пакеты могут прибывать в беспорядке или не прибывать вообще, а подтверждение установления связи должно быть осуществлено недвусмысленно, чтобы определять, прибыл ли пакет успешно, или нет.

В этом приложении рассматриваются два сценария. Предполагается, что в первом случае тракт соединения запроса должен предлагать надежную услугу, ориентированную на поток, но тракт соединения данных является ненадежным. Во втором случае и тракт соединения запроса, и тракт соединения данных являются ненадежными. Полезно рассмотреть эти два сценария по порядку.

H.2 Надежные запросы с ненадежными данными

В этом подразделе соединение запроса является надежным, означая, что запросы прибывают в сервер по порядку без потерь, а ответы серверов получаются клиентом по порядку и опять без потерь. В этом случае поля запросов и заголовки откликов могут быть сообщены точно, как в протоколе "http-tcp", и конечно, протокол HTTP рекомендуется для транспортирования запросов и заголовков откликов. Транспортный протокол этой особенности мог быть назван "http-udp", но такие специфические особенности выходят за сферу действия этого приложения.

Сообщения потоков JPIP, включая сообщение EOR (см. раздел D.3), должны быть разделены на пакеты и доставлены по ненадежному соединению данных (например, по протоколу UDP). Клиент должен подтверждать получение каждого такого пакета, посылая заголовок пакета обратно к серверу. Это позволяет серверу оценивать сетевые условия и определять, действительно ли оправдана, или нет, повторная передача пакетов. Когда окно обзора клиента изменилось, сервер мог бы решать не передавать повторно неподтвержденный пакет.

Следует соблюдать следующие общие рекомендации при построении транспортных протоколов этого типа:

a) Каждому запросу следует включать в себя поле запроса ID Запроса (см. C.3.5).

b) Для каждого запроса должно быть соответствующее сообщение EOR, даже если в отклике на запрос никакие сообщения потоков JPIP не посылаются. Это требование также применяется в случае транспортного средства "http-tcp".

c) Каждый пакет соединения данных, составленный сервером, должен состоять из целого числа сообщений потоков JPIP и/или сообщений EOR. Более того, первое сообщение потока JPIP в каждом пакете должно содержать полный заголовок, не полагаясь на повторение идентификатора кодового потока или компонентов кода класса предыдущего сообщения.

d) Все сообщения потоков JPIP (не обязательно сообщения EOR), найденные в пакете соединения данных, должны принадлежать отклику от отдельного запроса, а соответствующий идентификатор ID Запроса должен кодироваться в заголовке пакета.

e) Сообщения EOR могут быть найдены или в конце пакета, несущего то же самое значение ID Запроса, как запрос, чей отклик заканчивается, или в блоке из одного или более последовательных сообщений EOR, найденных в начале первого пакета, сопровождающего последний пакет, несущий такой идентификатор ID Запроса. Этот алгоритм позволяет сообщениям EOR, соответствующим одному или более последовательным пустым откликам (например, из-за заранее использованных запросов), быть связанными в первый пакет последующего непустого отклика.

f) В дополнение к значению ID Запроса, каждому заголовку пакета следует включать в себя порядковый номер пакета. Счетчик последовательности пакета устанавливается в 0 для первого пакета, связанного с любым конкретным значением ID Запроса. Последующие пакеты с тем же самым значением ID Запроса имеют последовательные порядковые номера. Этот алгоритм позволяет клиенту обозначать любые сообщения EOR, которые, возможно, не были получены из-за потери пакета. Важно, чтобы клиент был способен связывать запросы с данными отклика, чтобы синхронизировать воздействия утверждений обработки кэш-памяти в сервере с состоянием их собственной кэш-памяти.

g) Клиенты должны подтверждать получение каждого пакета, посылая сообщения подтверждений к серверу на тракте соединения данных отклика. Каждому сообщению подтверждения следует содержать копию заголовка полученного пакета, но можно было бы, вероятно, содержать и дополнительную информацию. Клиент, по его усмотрению, может приобщать сообщения подтверждения к нескольким пакетам при составлении пакетов подтверждений. Однако чрезмерное объединение может затронуть надежность, с которой серверы могут оценивать сетевую статистику.

h) Сервер не обязан повторно передавать любой неподтвержденный пакет, а клиентам не следует ожидать повторной передачи пропавших пакетов. Интеллектуальный сервер мог бы, например, выбрать повторную передачу неподтвержденных пакетов, в зависимости от их значимости для текущего окна обзора.

H.3 Ненадежные запросы с ненадежными данными

Этот подраздел касается транспортных средств, где и соединение запроса, и соединение данных являются ненадежными. Рекомендации для соединения данных являются точно такими, как описанные в разделе H.2 для случая, где данные доставляются ненадежно. С ненадежным соединением запроса, однако, есть возможность, что один или более запросов могли быть утеряны или прибыть в сервер без должного порядка. Протокол JPIP хорошо приспособлен к обработке этой ситуации, так как серверы имеют свободу предотвращать предыдущие запросы, когда прибывает новый запрос.

Следует соблюдать следующие рекомендации при обработке ненадежных запросов, в дополнение к тем, что перечислены в разделе H.2 для ненадежных соединений данных.

a) Каждому пакету запроса следует включать в себя заголовок, обозначающий значение ID Запроса.

b) Каждому пакету запроса следует также включать в себя порядковый номер, несущий достаточную информацию для определения, действительно ли были получены все пакеты, связанные с запросом.

c) Во многих случаях серверы могут просто игнорировать отсутствующие пакеты запросов, когда прибывает новый запрос. Чтобы сделать это, сервер должен только послать сообщения EOR на соединении данных, указывая, что отсутствующий запрос был немедленно заменен. Нет никакой потребности в сообщениях подтверждений, которые будут посланы в ответ на пакеты запросов. Нет никакой потребности в любых заголовках ответов, которые будут посланы в отклике на запросы, которые немедленно заменяются, потому что некоторые или все пакеты запросов были потеряны.

d) Для каждого запроса, который прибывает в сервер в полном составе, из сервера следует послать один или более пакетов откликов, которые обозначают ID Запроса и включают в себя любые заголовки откликов. Это верно даже в том случае, если запрос прибывает после того, как отклик был выпущен к любым последующим запросам (например, потому что некоторые пакеты запросов были неправильно задержаны). Это обеспечивает клиента механизмом для определения, действительно ли важный запрос был получен сервером, или нет.

e) Некоторые типы запросов должны обрабатываться сервером, чтобы избегать потери синхронизации с клиентом. Наиболее важными из них являются запросы, которые включают в себя поля обработки вычитающих моделей кэш-памяти. Чтобы позволить серверу обнаруживать такие запросы, не имея необходимости полностью преобразовывать поток запроса в последовательную форму, заголовкам пакетов запроса следует включать в себя следующие два поля:

   1) Флаг, указывающий, действительно ли пакет принадлежит запросу, который должен быть обработан перед обработкой последующих запросов.

   2) Идентификатор ID Запроса, связанный с самым последним запросом, для которого был установлен флаг, упомянутый в e1.

   Если сервер не получает один или более пакетов запроса с флагом e1 (т. е. запросы с условием e2 прибывают, а запрос с флагом e1 отсутствует), он должен быть свободным до тех пор, пока клиент повторно не передаст пакеты.

H.4 Синтаксис запроса и отклика

При проектировании новых транспортных механизмов для протокола JPIP необходимо следовать синтаксису запроса и отклика, описанному в Приложениях C и D. Однако допускается развивать эквивалентные двоичные представления различных полей запросов и заголовков откликов.

H.5 Установление сеанса

Поле запроса Нового канала (см. D.2.3) и соответствующий заголовок отклика могут использоваться для создания каналов, связанных с другими транспортными протоколами, которые отличаются от транспортных средств "http" и "http-tcp", описанных нормативным образом в этой Рекомендации | Международном стандарте. Для этой цели имена новых транспортных протоколов могут быть зарегистрированы с помощью комиссии по регистрации, определяемой в Приложении J. Процедуре для создания каналов для новых транспортных средств необходимо следовать тем же самым общим соглашениями, которые очерчены для "http-tcp". В частности, заголовки откликов для запроса, который создает новый канал, следует возвращать на транспортном средстве, которое использовалось для создания канала, в то время как данные отклика следует доставлять, используя транспортное средство нового канала.

Приложение I

Индексация файлов JPEG 2000 для JPIP

(Это приложение образует составную часть этой Рекомендации | Международного стандарта)

I.1 Введение (информативное)

Рекомендация МСЭ-Т Т.800 | ИСО/МЭК 15444-1:2004 и другие стандарты определяют семейство файловых форматов JPEG 2000. Семейство использует общий синтаксис, основным элементом которого является контейнер, называемый блоком. Это приложение определяет новые блоки файловых форматов, содержащие информацию индексации, включение которой в семейство файлов JPEG 2000 может облегчить развертывание таких файлов в системе JPIP, позволяя программам чтения файлов определять внутри файлов элементы, которые требуются для создания изображений возрастающим образом.

В частности, эти блоки могут быть полезны:

- для осуществления протокола JPIP на стороне сервера;
- клиенту, осуществляющему доступ к изображению дистанционно, используя более простой протокол, который позволяет осуществлять доступ к определенным диапазонам байтов в файле.

Это приложение определяет блоки индексов, соответствующие как информации на уровне файла, так и информации кодового потока. Блоки могут быть разбиты на категории следующим образом:

- Суперблок Индекса кодового потока (cidx) снабжает указателями информацию кодового потока, соответствующую классам бункеров данных главного заголовка, заголовка элемента мозаичного изображения, элемента мозаичного изображения и зоны JPP-потока и JPT-потока. Он содержит блок Искателя кодового потока (cptr), указывающий на индексированный кодовый поток, блок Декларации (manf), суммирующий остальную часть содержимого, и блоки таблиц индексов, которыми являются блок Таблицы индексов заголовков (mhix), суперблок Таблицы индексов части элемента мозаичного изображения (tpix), суперблок Таблицы индексов заголовка элемента мозаичного изображения (thix), суперблок Таблицы индексов пакета зоны (ppix) и суперблок Таблицы индексов заголовка пакета (phix). Блоки таблиц индексов соответствуют различным типам данных кодовых потоков, представленных классами бункеров данных в JPP-потоке и JPT-потоке, определенным в Приложении A. Блоки таблиц индексов, которые являются суперблоками, содержат блоки Индекса массива фрагментов (faix) или Таблицу индексов заголовка, перечисляющих фактические элементы кодовых потоков. Суперблоки Таблицы индексов заголовка, Пакета зоны и Заголовка пакета также каждый содержат блок Декларации.
- Суперблок Файлового индекса (fidx) снабжает указателями информацию на уровне файла, соответствующую классу бункера метаданных JPP-потока и JPT-потока. Если он не снабжает указателями верхний уровень файла, в случае которого он называется корневым блоком Файлового индекса, он содержит блок Файлового искателя (fptr), указывающий на индексированный суперблок. Он может содержать блоки Посредников (prxy), представляющие содержимое индексированного файла или суперблока.
- Блок Искателя индекса (iptr) указывает на корневой Файловый индекс, обеспечивая обнаружение его местоположения.

Рисунок I.1 иллюстрирует пример файла JPEG 2000, содержащего блоки индексов JPIP:

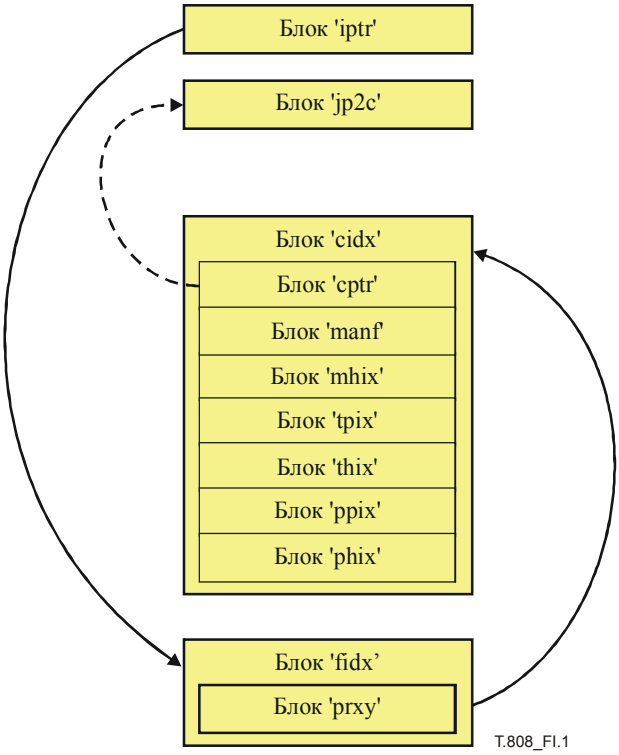

Рисунок I.1 − Часть примерного файла JPEG 2000, содержащего блоки индексов JPIP

I.2 Обозначение использования блоков индексов JPIP в перечне совместимости файлового формата JPEG 2000

Файлы, которые содержат один или более блоков индексов, определенных в этой Рекомендации | Международном стандарте, могут содержать поле CLi в блоке Файлового типа (как определено в Приложении I Рекомендации МСЭ-Т Т.800 | ИСО/МЭК 15444-1) со значением 'jpip' (0x6a70 6970).

I.3 Определенные блоки

I.3.1 Общие положения

Таблица I.1 перечисляет все блоки, определенные как часть этой Рекомендации | Международного стандарта. Для наложения ограничений на каждый блок, см. соответствующий подраздел, определяющий такой блок.

Таблица I.1 является информативной. Нормативные определения каждого блока содержатся внутри индивидуального подраздела, упоминаемого в таблице.

Таблица I.1 – Определенные блоки (информативная)

Имя блока	Тип	Суперблок	Комментарии
Блок индекса кодового потока (I.3.2)	'cidx' (0x6369 6478)	Да	Этот блок содержит индексирующую информацию относительно кодового потока JPEG 2000.
Блок Искателя кодового потока (I.3.2.2)	'cptr' (0x6370 7472)	Нет	Этот блок указывает на кодовый поток JPEG 2000.
Блок Таблицы индексов заголовков (I.3.2.4.3)	'mhix' (0x6D68 6978)	Нет	Этот блок определяет индекс сегментов маркеров в главном заголовке кодового потока или в заголовках частей элементов мозаичного изображения из элемента мозаичного изображения.
Блок Таблицы индексов части элемента мозаичного изображения (I.3.2.4.4)	'tpix' (0x7470 6978)	Да	Этот блок определяет местоположения и длины каждой части элемента мозаичного изображения в кодовом потоке.
Блок Таблицы индексов заголовков элемента мозаичного изображения (I.3.2.4.5)	'thix' (0x7468 6978)	Да	Этот блок определяет местоположения и длины каждой части кодового потока, необходимые для составления заголовков элемента мозаичного изображения для каждого элемента мозаичного изображения для правильного декодирования данных пакета зоны.
Блок Таблицы индексов пакетов зон (I.3.2.4.6)	'ppix' (0x7070 6978)	Да	Этот блок определяет местоположения и длины пакетов внутри кодового потока.
Блок Таблицы индексов заголовков пакетов (I.3.2.4.7)	'phix' (0x7068 6978)	Да	Этот блок определяет местоположения и длины заголовков пакетов внутри кодового потока.
Блок Декларации (I.3.2.3)	'manf' (0x6D61 6E66)	Нет	Этот блок суммирует блоки, которые непосредственно и смежным образом сопровождают его, внутри его содержащегося блока или файла на том же самом уровне, как и блок Декларации.
Блок индекса массива фрагментов (I.3.2.4.2)	'faix' (0x6661 6978)	Нет	Этот блок определяет местоположения и длины элементов кодового потока.
Блок Файлового индекса (I.3.3)	'fidx' (0x6669 6478)	Да	Этот блок может быть использован для нахождения других индексов и произвольных данных внутри файла
Блок Файлового искателя (I.3.3.2)	'fptr' (0x6670 7472)	Нет	Этот блок указывает на индексированный блок
Блок Посредника (I.3.3.3)	'prxy' (0x7072 7879)	Нет	Этот блок в блоке Файлового индекса представляет блок где-то в другом месте в файле
Блок Искателя индекса (I.3.4)	'iptr' (0x6970 7472)	Нет	Этот блок указывает корневой блок Файлового индекса из файла.

I.3.2 Блок Индекса кодового потока (суперблок)

I.3.2.1 Общие положения

Блок Индекса кодового потока содержит информацию индексации о кодовом потоке JPEG 2000. Тип блока Индекса кодового потока должен быть 'cidx' (0x6369 6478). Содержимое блока Индекса кодового потока должно быть следующим (рисунок I.2):

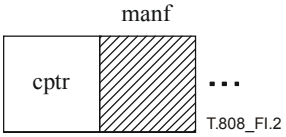

Рисунок I.2 – Организация содержимого блока Индекса кодового потока

cptr: Блок Искателя кодового потока. Этот блок указывает на кодовый поток, снабженный указателями с помощью блока Индекса кодового потока. Его структура определяется в подразделе I.3.2.2.

manf: Блок Декларации. Этот блок суммирует таблицы индексов, сопровождающие его внутри блока Индекса кодового потока. Его структура определяется в подразделе I.3.2.3.

I.3.2.2 Блок Искателя кодового потока

Блок Искателя кодового потока указывает на кодовый поток JPEG 2000. Тип блока Искателя кодового потока должен 'cptr' (0x6370 7472). Содержимое блока Искателя кодового потока должно быть следующим (рисунок I.3):

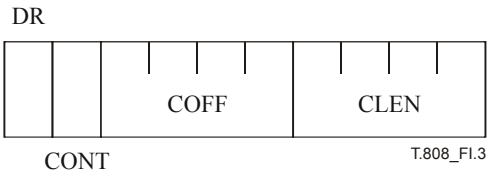

Рисунок I.3 – Организация содержимого блока Искателя кодового потока

DR: Эталон данных. Это поле определяет местоположение кодового потока или блока Таблицы фрагментов [*fragment*], поддерживающего его. Если он есть 0, то кодовый поток или его блок Таблицы фрагментов существует в текущем файле. В противном случае, количество определяет запись в блок Эталона данных в текущем файле. В этом случае запись Эталона данных, обозначаемая с помощью DR, указывает ресурс, который содержит кодовый поток или блок Таблицы фрагментов. Это поле хранится как 2-байтовое незнаковое целое число с обратным порядком байтов.

CONT: Тип Контейнера. Это поле хранится как 2-байтовое незнаковое целое число с обратным порядком байтов. Значения, определяемые в этой Рекомендации | Международном стандарте, описываются в таблице I.2.

COFF: Смещение кодового потока. Это поле определяет местоположение кодового потока или блока Перечня фрагментов, по обстановке, относительно начала файла или ресурса, обозначаемого с помощью DR. Это поле хранится как 8-байтовое незнаковое целое число с обратным порядком байтов.

CLEN: Длина кодового потока. Это поле определяет длину кодового потока или блока Перечня фрагментов, по обстановке. Это поле хранится как 8-байтовое незнаковое целое число с обратным порядком байтов.

Таблица I.2 – Значения типов контейнеров

CONT	Смысл
0	Полный кодовый поток появляется как смежный диапазон байтов внутри своего файла или ресурса. В этом случае значения смещений и длин, задаваемые здесь, относятся к самому кодовому потоку. Отметим, что кодовый поток может быть внутри блока Смежного кодового потока, но значения смещений и длин относятся к самому кодовому потоку, начинаясь на маркере SOC и заканчиваясь сразу же после маркера EOC.
1	Кодовый поток фрагментируется, а значения местоположений и длин относятся к блоку Перечня фрагментов (включая его заголовок блока), описывающему местоположения и длины каждого из фрагментов, которые представляют кодовый поток. Отметим, что все последующие местоположения и длины выражаются относительно начала кодового потока, как если бы он появился после воссоздания всех фрагментов, обозначенных в блоке Перечня фрагментов.
Все другие значения	Зарезервировано для использования ИСО.

I.3.2.3 Блок Декларации

Блок Декларации суммирует блоки, которые непосредственно и смежным образом сопровождают его внутри его содержащегося блока или файла на том же самом уровне, что и блок Декларации.

ПРИМЕЧАНИЕ. − Блок Декларации может быть использован для облегчения произвольного доступа в эти следующие блоки, например, блоки индексов, сопровождающие его внутри блока Индекса кодового потока.

Тип блока Декларации должен быть 'manf' (0x6D61 6E66). Содержимое блока Декларации должно быть следующим (рисунок I.4):

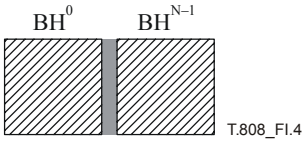

Рисунок I.4 − Организация содержимого блока Декларации

BH^{i}: Заголовок блока. Это поле содержит полный заголовок блока для *i*-го блока, непосредственно следующего за блоком Декларации. Длина этого поля составляет 16 байтов, если значение поля Lbox, содержащееся внутри такого заголовка блока равно 1, или 8 байтов в противном случае.

Количество блоков, *N*, чьи заголовки содержатся внутри блока Декларации, определяется длиной блока Декларации. При использовании внутри блока Таблицы индексов пакетов зон или блока Таблицы индексов заголовков пакетов, *N* есть количество компонентов кодового потока.

Внутри блока Индекса кодового потока, блока Таблицы индексов заголовков элементов мозаичного изображения, блока Таблицы индексов пакетов зон или блока Таблицы индексов заголовков пакетов, блок Декларации должен включать в себя все блоки, которые следуют за ним, вплоть до конца содержащегося блока.

I.3.2.4 Таблицы индексов

I.3.2.4.1 Общие положения

Блок Индекса кодового потока может содержать таблицу индексов для каждого из следующих видов данных кодового потока: главный заголовок, части элементов мозаичного изображения, заголовки элементов мозаичного изображения, пакеты (зон) и заголовки пакетов. Каждая таблица индексов является различным типом блока. В блоке Индекса кодового потока должен быть не больше, чем один из каждого вида таблицы.

Блоки Таблицы индексов частей элементов мозаичного изображения, Таблицы индексов пакетов зон и Таблицы индексов заголовков пакетов являются суперблоками, содержащими блоки Индексов массивов фрагментов. Блок Таблицы индексов заголовков элементов мозаичного изображения является суперблоком, содержащим блоки Таблиц индексов заголовков. Ниже определяются сначала блок Индекса массива фрагментов, а затем блоки Таблиц индексов.

I.3.2.4.2 Блок Индексов массива фрагментов

Блок Индексов массива фрагментов перечисляет местоположения и длины элементов кодового потока. Он используется внутри суперблоков Таблицы индексов частей элементов мозаичного изображения, Таблицы индексов пакетов зон и Таблицы индексов заголовков пакетов.

Тип блока Индекса массива фрагментов должен быть 'faix' (0x6661 6978). Содержимое блока Индекса массива фрагмента должно быть следующим (рисунок I.5):

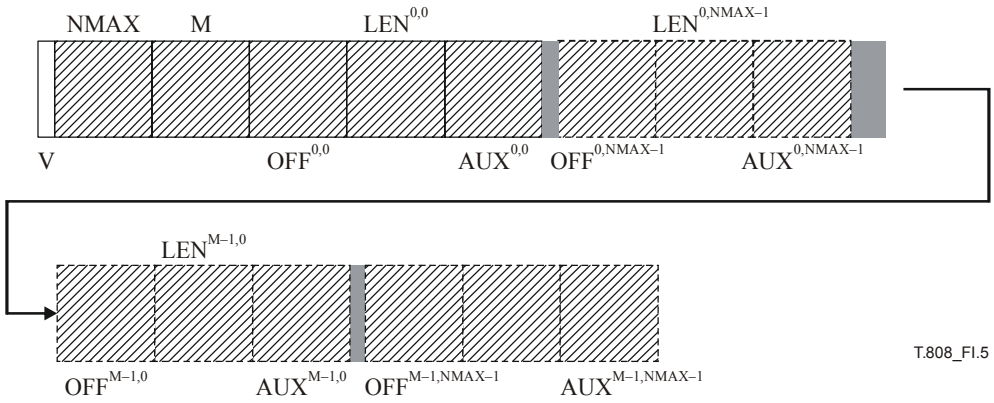

Рисунок I.5 – Организации содержимого блока Индексов массива фрагментов

V: Версия. Это поле кодируется как 1-байтовое незнаковое целое число. Значения, определяемые в этой Рекомендации | Международном стандарте, описываются в таблице I.3.

NMAX: Максимальное количество действительных элементов в любом ряду массива. При использовании внутри таблицы индексов кодовых потоков, NMAX есть максимальное количество элементов, что будут определены для любых элементов мозаичного изображения.

M: Количество рядов массива. При использовании внутри таблицы индексов кодовых потоков, M есть количество элементов мозаичного изображения.

OFF^{i,j}: Смещение. Это поле определяет смещение в байтах (относительно начала кодового потока) для *j*-го элемента в ряду *i* массива.

LEN^{i,j}: Длина. Это поле определяет длину в байтах *j*-го элемента в ряду *i* массива.

AUX^{i,j}: Вспомогательное. Это поле определяет вспомогательную информацию относительно *j*-го элемента в ряду *i* массива. Значение этого поля должно быть 0, если другое не разрешено суперблоком, содержащим этот блок. Все ненулевые значения этого поля зарезервированы.

В то время как все ряды массива, определенного в блоке Индекса массива фрагментов, должны быть сохранены с количеством NMAX элементов, объект, описываемый таким рядом, может иметь меньшее количество элементов для определения. В этом случае, где для любого ряда *i*, содержащего *J* действительных элементов, где *J* меньше, чем NMAX, значения от $OFF^{i,J}$ до $OFF^{i,NMAX-1}$ и от $LEN^{i,J}$ до $LEN^{i,NMAX-1}$ должны быть установлены в нуль.

Таблица I.3 − Значения версий

CONT	Смысл
0	NMAX, M и все поля $OFF^{i,j}$ и $LEN^{i,j}$ кодируются как 4-байтовые незнаковые целые числа с обратным порядком байтов, а поля $AUX^{i,J}$ не присутствуют.
1	NMAX, M и все поля $OFF^{i,j}$ и $LEN^{i,j}$ кодируются как 8-байтовые незнаковые целые числа с обратным порядком байтов, а поля $AUX^{i,J}$ не присутствуют.
2	Все поля, отличающиеся от V, кодируются как 4-байтовые незнаковые целые числа с обратным порядком байтов.
3	NMAX, M и все поля $OFF^{i,j}$ и $LEN^{i,j}$ кодируются как 8-байтовые незнаковые целые числа с обратным порядком байтов, а все поля $AUX^{i,J}$ кодируются как 4-байтовые незнаковые целые числа с обратным порядком байтов.
Все другие значения	Зарезервировано для использования ИСО.

I.3.2.4.3 Блок Таблицы индексов заголовков

Блок Таблицы индексов заголовков снабжает указателями главный заголовок кодового потока или заголовки частей элементов мозаичного изображения из элемента мозаичного изображения, указывая полную длину главного заголовка или длину первой части элемента мозаичного изображения, а также местоположения и длины сегментов маркеров в заголовке. Все сегменты маркеров должны быть включены, за исключением того, что сегмент маркера SOT может быть опущен для заголовков частей элементов мозаичного изображения, которые состоят только из SOT и SOD. Сегменты маркеров не должны перечисляться в порядке, в котором они появляются в кодовом потоке. Блок Таблицы индексов заголовков может иметь место только внутри блока Индекса кодового потока. На верхнем уровне он снабжает указателями кодовый поток и должен встречаться не более одного раза. Внутри блока Таблицы индексов заголовков элементов мозаичного изображения он снабжает указателями заголовки части элемента мозаичного изображения.

ПРИМЕЧАНИЕ. − Намерение состоит в обеспечении эффективных средств для того, чтобы перескакивать через информацию указателя в заголовке, который не требуется для эффективного просмотра файла, но может излишне загромождать заголовок. Составление списка многократных сегментов маркеров с тем же самым кодом маркера смежным образом в блоке Таблицы индексов заголовков позволит программам чтения перескакивать через сегменты маркеров, в которых они не заинтересованы.

Тип блока Таблицы индексов заголовков должен быть 'mhix' (0x6D68 6978). Содержимое блока таблицы индексов заголовков должно быть следующим (рисунок I.6):

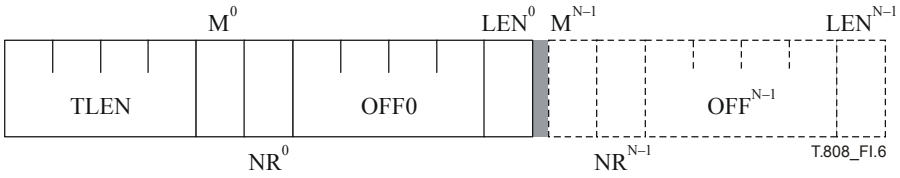

Рисунок I.6 – Организация содержимого блока Таблицы индексов заголовков

TLEN: Длина. Когда блок Таблицы индексов заголовков снабжает указателями главный заголовок, это поле определяет общую длину главного заголовка. Когда блок Таблицы индексов заголовков снабжает указателями заголовки частей элементов мозаичного изображения, это поле определяет общую длину заголовка первой части элемента мозаичного изображения. Значение этого поля кодируется как 8-байтовое незнаковое целое число с обратным порядком байтов.

M^i: Код маркера. Это поле определяет код маркера, начинающий *i*-ый сегмент маркера, перечисляемый в этом блоке. Значение этого поля кодируется как 2-байтовое незнаковое целое число с обратным порядком байтов.

NR^i: Сохранение номера. Это поле указывает, что (по меньшей мере) перечисляются NR^i сегментов маркеров с тем же самым кодом маркера M^i, сразу же и смежным образом сопровождая *i*-ый сегмент маркера в этом перечне. Значение этого поля кодируется как 2-байтовое незнаковое целое число с обратным порядком байтов.

OFF^i: Смещение. Это поле определяет смещение в байтах, относительно начала кодового потока, параметров сегментов маркеров (включая параметр длины, но не сам маркер) для *i*-го сегмента маркера в этом перечне. Значение этого поля кодируется как 8-байтное незнаковое целое число с обратным порядком байтов.

LEN^i: Длина. Это поле определяет длину в байтах параметров сегментов маркеров (включая два байта параметра длины, но не два байта самого маркера) для *i*-го сегмента маркера в этом перечне. Значение этого поля кодируется как 2-байтовое незнаковое целое число с обратным порядком байтов и является тем же самым, как и значение параметра длины в самом сегменте маркера.

Количество сегментов маркеров, *N*, перечисляемых в блоке Таблицы индексов заголовков, определяется длиной блока таблицы индексов заголовков.

I.3.2.4.4 Блок Таблицы индексов части элемента мозаичного изображения (суперблок)

Блок Таблицы индексов части элемента мозаичного изображения снабжает указателями местоположения и длины каждой части элемента мозаичного изображения в кодовом потоке, где каждая часть элемента мозаичного изображения начинается ее маркером SOT и заканчивается последним пакетом части мозаичного изображения.

Тип блока Таблицы индексов части элемента мозаичного изображения должен быть 'tpix' (0x7470 6978). Содержимое блока Таблицы индексов части элемента мозаичного изображения должно быть следующим (рисунок I.7):

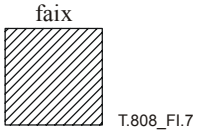

Рисунок I.7 − Организация содержимого блока Таблицы индексов части элемента мозаичного изображения

faix: Блок Индексов массива фрагментов. Этот блок перечисляет местоположения и длины всех частей элементов мозаичного изображения в кодовом потоке. Его структура определяется в подразделе I.3.2.4.2. Ряд *m*-ый в этой таблице соответствует *m*-му элементу мозаичного изображения в кодовом потоке. Записи на этом ряде удерживают местоположения и длины всех частей элемента мозаичного изображения в соответствующем элементе мозаичного изображения, в порядке кодового потока. Если блок Индексов массива фрагментов имеет Версию, равную 2 или 3, то Вспомогательные поля определяют для каждой части элемента мозаичного изображения наименьшее *n*, так, что во всех компонентах, для которых $(N_L - n)$ является неотрицательным, уровень разрешающей способности $(N_L - n)$ и все нижние уровни разрешающей способности были завершены, когда эта часть элемента мозаичного изображения объединяется со всеми предыдущими частями элементов мозаичного изображения того же самого элемента мозаичного изображения, где N_L есть количество уровней разложения, которые могут изменяться компонентом. Если никакие уровни разрешающей способности любого компонента не были завершены, значение Вспомогательного поля есть один плюс максимальное значение N_L по всем компонентам. Нуль значения достигается тогда, когда все разрешающие способности во всех компонентах были завершены. Поскольку разрешающие способности не обязательно появляются в должном порядке в элементе мозаичного изображения, некоторые уровни разрешающей способности выше значения, сообщенного Вспомогательным полем, возможно, были завершены.

I.3.2.4.5 Блок Таблицы индексов заголовков элементов мозаичного изображения (суперблок)

Блок Таблицы индексов заголовков элементов мозаичного изображения снабжает указателями заголовки элементов мозаичного изображения каждого элемента мозаичного изображения для правильного декодирования пакетных данных зон.

Тип блока Таблицы индексов заголовков элементов мозаичного изображения должен быть 'thix' (0x7468 6978). Содержимое блока Таблицы индексов заголовков элементов мозаичного изображения должно быть следующим (рисунок I.8).

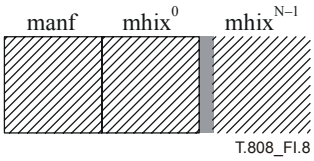

Рисунок I.8 − Организация содержимого блока Таблицы индексов заголовков элементов мозаичного изображения

Количество блоков Таблицы индексов заголовков элементов мозаичного изображения, N, есть количество элементов мозаичного изображения.

manf: Блок Декларации. Этот блок суммирует блоки, определенные с помощью $mhix^i$ внутри этого блока Таблицы индексов заголовков элементов мозаичного изображения. Его структура определяется в подразделе I.3.2.3.

$mhix^i$: Блок Таблицы индексов заголовков. Этот блок снабжает указателями заголовки частей элементов мозаичного изображения для *i*-го элемента мозаичного изображения. Его структура определяется в подразделе I.3.2.4.3.

I.3.2.4.6 Блок Таблицы индексов пакетов зон (суперблок)

Блок Таблицы индексов пакетов зон снабжает указателями пакеты внутри кодового потока. Тип блока Таблицы индексов пакетов зон должен быть 'ppix' (0x7070 6978). Содержимое блока Таблицы индексов пакетов зон должно быть следующим (рисунок I.9):

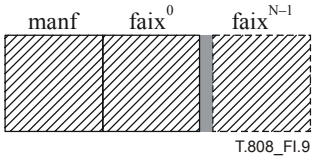

Рисунок I.9 – Организация содержимого блока Таблицы индексов пакетов зон

Количество блоков индексов массивов фрагментов, N, должно быть не больше, чем количество компонентов кодовых потоков.

manf: Блок Декларации. Этот блок суммирует блоки, определенные с помощью $faix^i$ внутри блока таблицы индексов пакетов зон. Его структура определяется в подразделе I.3.2.3.

$faix^i$: Блок *i*-ый Индекса массива фрагментов соответствует *i*-му компоненту изображения в кодовом потоке. Ряд *m*-ый в этой таблице соответствует *m*-му элементу мозаичного изображения в кодовом потоке. Записи на этом ряде удерживают местоположения и длины всех пакетов в соответствующем компоненте элемента мозаичного изображения. Пакеты появляются смежным образом, возрастая в порядке слоя, внутри их соответствующих зон, а зоны появляются в порядке, связанном с порядковым номером *s*, определенным в подразделе A.3.2.1. Однако фиксированный порядок пакетов не обязательно является тем же самым, как определено в любых сегментах маркеров COD/POC внутри кодового потока. Структура блока Индексов массива фрагментов определяется в подразделе I.3.2.4.2.

Если заголовки пакета упакованы в сегменты маркеров PPM или PPT, то соответствующие записи в массиве фрагментов относятся только к местоположению и длине тела пакета, как оно появляется внутри тела своей части элемента мозаичного изображения. Записям, которые относятся к несуществующим пакетам (или потому что уместный компонент элемента мозаичного изображения содержит меньше пакетов, чем другой компонент элемента мозаичного изображения в том же самом массиве, или потому что кодовый поток был обрезан перед точкой, в которой существовал бы такой пакет), следует иметь свое поле местоположения, установленным в нуль. Записи, которые относятся к пакетам, чье тело является пустым и чей заголовок состоит из точно одного байта, 0x80, могут быть определены, используя значение длины в виде нуля. Такие пакеты часто встречаются в кодовых потоках JPEG 2000; приложения могут избегать предзаголовка из выборки явно таких пакетов, чье содержимое является предсказуемым. Если уместный сегмент маркера COD определяет, что маркеры EPH должны появляться после каждого заголовка пакета в некотором элементе мозаичного изображения, то специальное значение длины в виде нуля в таком элементе мозаичного изображения должно истолковываться как значащее, что пакет состоит из байта 0x80, сопровождаемого маркером EPH.

I.3.2.4.7 Блок Таблицы индексов заголовков пакетов (суперблок)

Блок Таблицы индексов заголовков пакетов снабжает указателями заголовки пакетов внутри кодового потока. Тип блока Таблицы индексов заголовков пакетов должен быть 'phix' (0x7068 6978). Содержимое блока Таблицы индексов заголовков пакетов должно быть следующим (рисунок I.10):

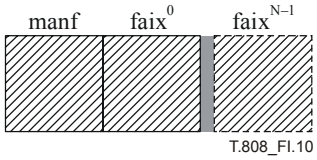

Рисунок I.10 – Организация содержимого блока Таблицы индексов заголовков пакетов

Количество блоков Индексов массивов фрагментов, N, должно быть не больше, чем количество компонентов кодовых потоков.

manf: Блок Декларации. Этот блок суммирует блоки, определяемые с помощью $faix^i$ внутри блока Таблицы индексов заголовков пакетов. Его структура определяется в подразделе I.3.2.3.

$faix^i$: Блок *i*-ый Индекса массива фрагментов соответствует *i*-му компоненту изображения в кодовом потоке. Ряд *m*-ый в этой таблице соответствует *m*-му элементу мозаичного изображения в кодовом потоке. Записи на этом ряде удерживают местоположения и длины всех заголовков пакетов в соответствующем компоненте элемента мозаичного изображения. Заголовки пакетов появляются смежным образом, возрастая в порядке слоя, внутри их соответствующих зон, а зоны появляются в порядке, связанном с порядковым номером *s*, определенном в подразделе A.3.2.1. Однако фиксированный порядок заголовков пакетов необязательно является тем же самым, что определяется в любых сегментах маркеров COD/POC внутри кодового потока. Структура блока Индексов массива фрагментов определяется в подразделе I.3.2.4.2.

Записям, которые относятся к несуществующим пакетам (или потому что уместный компонент элемента мозаичного изображения содержит меньше пакетов, чем другой компонент элемента мозаичного изображения в том же самом массиве, или потому что кодовый поток был обрезан перед точкой, в которой существовал бы такой пакет), следует иметь их поле местоположения, установленным в нуль. Записи, которые относятся к пакетам, чье тело является пустым и чей заголовок состоит из точно одного байта, 0x80, могут быть определены, используя значения длины вида нуль. Такие пакеты часто встречаются в кодовых потоках JPEG 2000; приложения могут избегать предзаголовка из выборки явно таких пакетов, чье содержимое является предсказуемым. Если уместный сегмент маркера COD определяет, что маркеры EPH должны появляться после каждого заголовка пакета в некотором элементе мозаичного изображения, то специальное значение длины в виде нуля в таком элементе мозаичного изображения должно истолковываться как значащее, что пакет состоит из байта 0x80, сопровождаемого маркером EPH.

I.3.3 Блоке Файлового индекса (суперблок)

I.3.3.1 Общие положения

Блок Файлового индекса может быть использован для нахождения других индексов (в частности, индекса кодового потока, соответствующего кодовому потоку) и произвольных данных внутри файла.

Корневой блок Файлового индекса снабжает указателями верхний уровень файла. Любой другой Файловый индекс снабжает указателями суперблок внутри файла. Должно быть не больше одного блока Файлового индекса с заданной сферой действия (суперблок верхнего уровня или конкретный суперблок) внутри заданного файла.

Тип блока Файлового индекса должен быть 'fidx' (0x6669 6478). Содержимое блока Файлового индекса должно быть следующим (рисунок I.11):

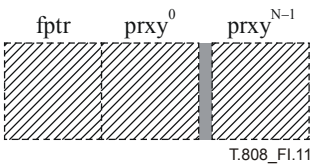

Рисунок I.11 − Организация содержимого блока Файлового индекса

fptr: Блок Файлового искателя. Корневой блок Файлового индекса не должен включать в себя этот блок. Любой другой блок Файлового индекса должен включать в себя этот блок, который должен указывать на суперблок, снабженный указателем блока Файлового индекса. Структура блока Файлового искателя определяется в подразделе I.3.3.2.

prxy^{i}: Блок Посредника. Этот блок представляет блок в порции файла, снабженной указателями с помощью блока Файлового индекса. Корневой блок Файлового индекса должен включать в себя посредников только для блоков на верхнем уровне файла. Любой другой блок Файлового индекса должен включать в себя посредников только для блоков на верхнем уровне суперблока, снабженного указателями с помощью блока Файлового индекса. Посредники должны встречаться в том же самом порядке, что и блоки, но не все блоки нуждаются в посредниках. Структура блока Посредника определяется в подразделе I.3.3.3.

ПРИМЕЧАНИЕ. − Поскольку в некоторых случаях присутствие, отсутствие или упорядочение блоков в файле является существенным, для приложений может быть полезным, если, перед любыми такими блоками с посредниками, никакие блоки внутри сферы действия индекса не опускаются из индекса.

I.3.3.2 Блок Файлового искателя

Блок Файлового искателя указывает на блок. Тип блока Файлового искателя должен быть 'fptr' (0x6670 7472). Содержимое блока Файлового искателя должно быть следующим (рисунок I.12):

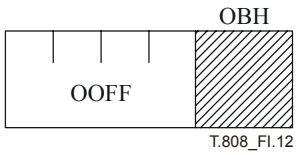

Рисунок I.12 – Организация содержимого блока Файлового искателя

OOFF: Первоначальное смещение. Это поле определяет смещение в байтах (относительно начала файла) блока, указанного этим блоком Файлового искателя. Значение этого поля кодируется как 8-байтовое незнаковое число с обратным порядком байтов.

OBH: Заголовок первоначального блока. Это поле содержит заголовок завершенного блока из блока, указанного блоком Файлового искателя. Длина этого поля составляет 16, если значение поля Lbox, содержащееся в таком заголовке блоке, равно 1, или 8 байтов в противном случае.

I.3.3.3 Блок Посредника

В блоке Файлового индекса блок Посредника представляет блок где-то в другом месте в файле, указывая его местоположение и длину, местоположение и длину любого индекса к блоку, а также префикс содержимого блока.

Тип блока Посредника должен быть 'prxy' (0x7072 7879). Содержимое блока Посредника должно быть следующим (рисунок I.13):

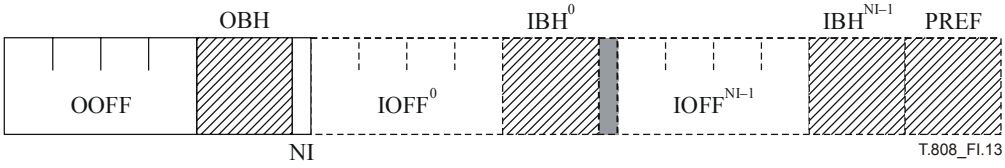

Рисунок I.13 – Организация содержимого блока Посредника

OOFF: Первоначальное смещение. Это поле определяет смещение в байтах (относительно начала файла) блока, представленного этим блоком Посредника. Значение этого поля кодируется как 8-байтовое незнаковое число с обратным порядком байтов.

OBH: Заголовок первоначального блока. Это поле содержит заголовок завершенного блока из блока, представленного блоком Посредника. Длина этого поля составляет 16 байтов, если значение поля Lbox, содержащегося в заголовке блока, равно 1, или 8 байтов в противном случае.

NI: Количество индексов. Это поле указывает количество указателей индексов, включенных в этот блок Посредника. Каждый набор последующих полей IOFF^{i} и IBH^{i} указывает либо на блок Файлового индекса, либо на блок Индекса кодового потока, который снабжает

указателями блок, представленный этим блоком Посредника. Все другие значения являются зарезервированными. Значение этого поля кодируется как 1-байтовое незнаковое целое число.

IOFF^{i}: Смещение индекса. Это поле содержит смещение в байтах (относительно начала файла) для *i*-го блока индекса. Значение этого поля должно кодироваться как 8-битовое незнаковое целое число с обратным порядком байтов.

IBH^{i}: Заголовок блока индекса. Это поле содержит заголовок завершенного блока из блока *i*-го индекса. Длина этого поля составляет 16 байтов, если значение поля Lbox, содержащееся внутри такого заголовка блока, есть 1, или 8 байтов в противном случае.

PREF: Префикс. Это поле содержит произвольный префикс данных в блоке, представленном блоком Посредника. Оно может иметь любую длину от нуля вплоть до длины содержимого первоначального блока.

I.3.4 Блок Искателя индекса

Блок Искателя индекса указывает на корневой блок Файлового индекса. Он должен встречаться только в том случае, если файл содержит корневой блок Файлового индекса. Тип блока Искателя индекса должен быть 'iptr' (0x6970 7472). Содержание блока Искателя индекса должно быть следующим (рисунок I 14):

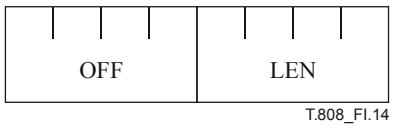

Рисунок I.14 − Организация содержимого блока Искателя индекса

OFF: Смещение. Этот поле определяет местоположение корневого блока Файлового индекса относительно начала файла. Это поле хранится как 8-байтовое незнаковое целое число с обратным порядком байтов.

LEN: Длина. Это поле определяет размер корневого блока Файлового индекса. Это поле хранится как 8-байтовое незнаковое число с обратным порядком байтов.

I.4 Ассоциация индексов кодовых потоков с кодовыми потоками

В файле JP2, JPX или JPM блок Индекса кодового потока должен встречаться на верхнем уровне файла, а *i*-ый блок Индекса кодового потока должен соответствовать *i-му* кодовому потоку также на верхнем уровне файла. Блок Искателя кодового потока внутри блока Индекса кодового потока также указывает кодовый поток, который снабжен указателями с помощью блока Индекса кодового потока.

I.5 Ограничения по размещению (информативное)

На блоки, определенные в этом приложении, были наложены несколько ограничений по размещению. Они могут быть помещены в конце файла, если желательно; это, вероятно, будет удобным, когда неиндексированный файл позднее снабжается указателем. Однако может быть полезным размещать блок Искателя индекса около начала файла, лучше − непосредственно после любых блоков, которые обязаны находиться в смежной группе в начале файла (такого, как после блока Файлового типа в файле JP2, или после блока Требований программ чтения в файле JPX), где его можно легко найти программами чтения файлов. Чтобы свести к минимуму движение файловых блоков, на добавление этого блока и дополнительно на добавление кода 'jpip' к перечню совместимости в блоке Файлового типа, может быть использован блок Свободный (определенный в Приложении M. 11.20 Рекомендации МСЭ-Т T.801 | ИСО/МЭК 15444-2) как указатель места заполнения для него во всех файлах, все еще подлежащих снабжению указателями.

Приложение J

Регистрация расширений к этой Рекомендации | Международному стандарту

(Это приложение образует составную часть этой Рекомендации | Международного стандарта)

J.1 Введение в регистрацию

Регистрация является процессом добавления расширений к этой Рекомендации | Международному стандарту после того, как она была опубликована. В этой Рекомендации | Международном стандарте много возможностей может быть расширено путем регистрации. Этот подраздел обозначает те пункты, которые могут быть расширены путем регистрации, процесс, с помощью которого возможности могут быть зарегистрированы, и процесс, с помощью которого Орган регистрации будет издавать такие расширения. Путем регистрации могут быть расширены только пункты, которые определены в этом подразделе.

J.2 Элементы регистрации

Процесс регистрации складывается из следующих элементов.

- **Орган регистрации**: Организационный объект, ответственный за обзор, поддержание распределение и действие как точка контакта для всех мероприятий, относящихся к регистрации. Орган регистрации подлежит определению.
- **Податель**: Подателем является организация или личность, которая запрашивает, чтобы пункт был зарегистрирован.
- **Комиссия по пересмотру**: Комиссия по пересмотру является организационным объектом, который одобряет регистрацию предложенного пункта. Она складывается из специального комитета, назначенного Председателем комиссии по пересмотру. Комиссия по пересмотру должна быть подгруппой Рабочей группы 1 JPIP Подкомитета 29 Объединенного технического комитета 1 ИСО/МЭК [ISO/IEC JTC 1/SC 29/WG 1 JPIP].
- **Председатель комиссии по пересмотру**: Председатель комиссии по пересмотру несет ответственность за рассмотрение каждого пункта, являющегося соискателем. Он осуществляет связь с подателем через Орган регистрации. Председатель комиссии по пересмотру должен быть председателем подгруппы ISO/IEC JTC 1/SC 29/WG 1 JPIP.
- **Испытание**: Логическое обоснование того, что именно Комиссии по пересмотру должна использовать для определения, следует ли зарегистрировать представление/пункт.
- **Представление/Пункт**: Это − предложение для регистрации. Каждое предложение должно включать в себя название пункта, подлежащего расширению, предлагаемый признак (тэг)/ тождественность для расширения и целесообразность/причину для расширения.

J.3 Критерии оценки регистрации

Комиссия по пересмотру должна оценивать все представления на основе следующих критериев:

- Удовлетворяет ли оно потребностям, которые не удовлетворяются стандартом или другими расширениям?
- Является ли расширение достаточно определенным?
- Удовлетворяет ли расширение общим потребностям (например, приложениям потокового видео вообще) или потребностям, характерным только для поставщика (например, конкретное осуществление потокового видео поставщика)?

J.4 Пункты, которые могут быть расширены путем регистрации

J.4.1 Расширенные блоки внутри блока указателя места заполнения

Новые типы блоков для блоков, которые будут использоваться внутри поля ExtendedBoxList в блоке Указателя места заполнения (A.3.6.3), должны быть регистрируемыми. Предложение для регистрации нового типа блока должно содержать полное определение такого блока (тип блока и содержимое блока), инструкции о том, когда сервер может записать этот блок внутри блока Указателя места заполнения, а также инструкции о том, что клиент может делать, когда он сталкивается с блоком Указателя места заполнения, содержащим этот блок.

J.4.2 Контекст кодового потока

Новые значения `context-range` [*контекст−диапазон*] для запроса конкретных кодовых потоков с использованием поля Контекста кодового потока (C.4.7) должны быть регистрируемыми. Предложение зарегистрировать новое значение `context-range` должно содержать полное определение значения, инструкции о том, как сервер должен отображать такое значение в имеющихся кодовых потоках в логической цели, и инструкции о том, как сервер должен откликаться в заголовке отклика Контекста кодового потока.

J.4.3 Канальные транспортные средства

Новые канальные транспортные средства (Приложение H) должны быть регистрируемыми. Предложение зарегистрировать новые канальные транспортные средства должно содержать полное определение транспортного средства, включая идентификатор, подлежащий использованию для такого транспортного средства.

J.4.4 Предпочтения

Новые предпочтения клиентов должны быть регистрируемыми. Это включает в себя новые наборы предпочтений (новые значения `related-pref-set`, как определяется в C.10.2.1), или новые варианты для существующего или зарегистрированного набора предпочтений. Предложение зарегистрировать новый вариант предпочтения или набор предпочтений должно содержать полное определение синтаксиса, смысл новых вариантов и инструкции о том, как сервер должен откликаться при действии согласно такому предпочтению.

J.5 Процесс регистрации

Процессом регистрации является следующее.

a) Податель создает пункт соискателя для регистрации.

b) Пункт соискателя подается Органу регистрации.

c) Орган регистрации передает далее пункт соискателя Председателю комиссии по пересмотру.

d) Председатель комиссии по пересмотру распространяет пункт соискателя для Комиссии по пересмотру и расписание собраний, телефонные вызовы и т. д., по обстановке, для рассмотрения пункта.

e) Комиссия по пересмотру должна оценивать все представления. Если текст представления не удовлетворяет требованиям, то он должен быть возвращен подателю для видоизменения. Одобрение будет даваться решениям, которые являются более общими, а предложенные решения, которые являются весьма характерными только для поставщиков, могут быть возвращены подателю, чтобы сделать их более общими и более применимыми для промышленности в целом.

f) При одобрении Председатель передает одобрение в адрес Органа регистрации, который уведомляет ИСО и подателя и делает зарегистрированный или изданный пункт доступным.

g) При отклонении Председатель готовит ответный документ, указывающий, почему пункт было решено отклонить, и передает это далее в адрес Органа регистрации, который уведомляет подателя.

J.6 Рамки времени для процесса регистрации

Комиссия по пересмотру должна ответить на все запросы о регистрации в течение семи месяцев от даты подачи. В пределах такого периода времени Комиссия по пересмотру соберется на официальное собрание ISO/IEC JTC 1/SC 29/WG 1 JPIP, чтобы оценить предложение, принять решение и подготовить проект ответа.

Приложение K

Примеры приложений

(Это приложение не образует составную часть этой Рекомендации | Международного стандарта)

K.1 Введение

Это приложение представляет некоторые информативные примеры аспектов реализаций JPIP.

K.2 Использование JPIP с кодовыми потоками в других файловых форматах

Протокол JPIP может использоваться для осуществления доступа к кодовым потокам JPEG 2000, сохраненным в других файловых форматах, которые отличаются от семейства файлов JPEG 2000. Например, файлы и DICOM, и PDF имеют возможность содержать кодовые потоки JPEG 2000. Для определения местонахождения кодового потока JPEG 2000 в окружающей среде "клиент-сервер" могут использоваться некоторые процедуры, не указанные в этой Рекомендации | Международном стандарте. Запросы и отклики JPIP могут использоваться на объекте, как только определяется место кодового потока. Как раз для такой ситуации предназначено поле запроса Подцели. Альтернативно, сервер мог бы обеспечивать доступ к кодовым потокам через различный указатель URL.

K.3 Методы осуществления части элемента мозаичного изображения

K.3.1 Определение сервером соответствующих частей элементов мозаичного изображения для запроса окна обзора

Для связи через часть элемента мозаичного изображения преобразование окна обзора в набор элементов мозаичного изображения является простым. Желательная область изображения преобразуется в "единицы эталонной сетки". Порции XTsiz и YTsiz из сегмента маркера SIZ используются для определения, какие элементы мозаичного изображения пересекаются с окном обозрения.

ПРИМЕЧАНИЕ. − Хотя на эталонной сетке все элементы мозаичного изображения имеют одинаковые размеры, на эталонной сетке на пониженной скорости дискретизации, после разложения подполосы, элементы мозаичного изображения не обязательно все имеют одинаковые размеры. Элемент мозаичного изображения, пересекающий окно обзора, даже элемент мозаичного изображения, полностью содержащийся внутри него, не могут внести никаких отсчетов в окно обзора на самых нижних уровнях разрешающей способности; однако реализациям не нужно пользоваться преимуществом этого случая путем опускания элемента мозаичного изображения из отклика в целом.

Для определения необходимых частей элементов мозаичного изображения используются уровень разрешающей способности и качество. Блок Таблицы индексов части элемента мозаичного изображения, если имеется, может использоваться для получения информации о местоположении частей элементов мозаичного изображения в кодовом потоке и (если включены Вспомогательные поля) для завершения уровней разрешающей способности внутри частей элементов мозаичного изображения. Сегменты маркеров SOT также дают индексы элементов мозаичного изображения и частей элементов мозаичного изображения, а также количество байтов в каждой части элемента мозаичного изображения. Из кодового потока подходящие байты, соответствующие частям элементов мозаичного изображения, которые необходимо отослать, передаются к клиенту. В случае, когда окно обзора изменяется, и соответствующие уместные элементы мозаичного изображения также изменяются, то чтобы обновить изображение отображения, нужно послать только уместные части элементов мозаичного изображения, которые не были посланы ранее.

K.3.2 Декодирование изображения из сообщений возвращенных JPT-потоков

Протокол JPIP определяет механизмы для передачи данных сжатых изображений и метаданных между клиентом и сервером. Механизмы для клиента, чтобы отображать возвращаемые данные, не определяются, и, конечно, будут широко различаться между приложениями. Этот подраздел предоставляет информацию относительно получения отсчетов компонентов из возвращаемых данных.

Приложение клиента, которое получило всё из данных главного заголовка (указанное с помощью завершенного бункера данных, появляющегося в сообщении отклика для бункера 0 данных заголовка), может сцеплять такой бункер данных с завершенными частями элементов мозаичного изображения из бункеров данных элемента мозаичного изображения, чтобы сформировать допустимый кодовый поток JPEG 2000. Этот кодовый поток может быть предоставлен к соответствующему декодеру JPEG 2000, а результат отображен. Конечно, в целях эффективности, клиент может иметь желание предоставлять параметры окна обзора к интеллектуальному декодеру наряду с кодовым потоком так, что будут отображены только части, необходимые для текущего окна обзора.

K.3.3 Вспомогательная сигнализация для частей элементов мозаичного изображения

Таблицы K.1 и K.2 иллюстрируют использование Вспомогательных полей в расширенных сообщениях бункеров данных элементов мозаичного изображения и в блоке Таблицы индексов частей элементов мозаичного изображения.

ПРИМЕЧАНИЕ. − В этом примере определение *r* отличается от того, которое используется в других местах в этой Рекомендации | Международном стандарте, но является совместимым с Приложением B Рекомендации МСЭ-Т T.800 | ИСО/МЭК 15444-1:2004.

Таблица K.1 иллюстрирует простой случай, в котором все компоненты элементов мозаичного изображения из элемента мозаичного изображения с последовательной разрешающей способностью имеют одинаковое количество уровней разложения, и в котором границы сообщений (в случае бункера данных) или границы частей элементов мозаичного изображения (в случае блока индекса) встречаются только между каждым следующим уровнем разрешающей способности.

Таблица K.1 – Пример использования вспомогательных полей в простом случае

Порядковый номер сообщения в бункере данных, или номер части элемента мозаичного изображения в элементе мозаичного изображения	Уровень разрешающей способности *r*	$n = N_L - r$	Вспомогательное значение
0	0	2	2
1	1	1	1
2	2	0	0

Таблица K.2 иллюстрирует более сложный случай, в котором количество уровней разложения изменяется компонентом элемента мозаичного изображения. Комментарий делается в заключительной колонке таблицы при первом случае каждого нового Вспомогательного значения. Этот случай соответствует элементу мозаичного изображения из трехкомпонентного изображения в порядке продвижения RC…, например, в порядке продвижения LRCP с одиночным слоем, или в порядке продвижения RPCL с одиночной зоной в элементе мозаичного изображения. Границы сообщений (в случае бункера данных) или границы частей элементов мозаичного изображения (в случае блока индекса) встречаются между каждым компонентом каждого уровня разрешающей способности, а также между уровнями разрешающей способности. Компоненты 0 и 1 имеют два уровня разложения ($N_L = 2$), а компонент 2 имеет одиночный уровень разложения ($N_L = 1$).

Таблица K.2 – Пример использования вспомогательных полей в более сложном случае

Порядковый номер сообщения в бункере данных, или номер части элемента мозаичного изображения в элементе мозаичного изображения	Индекс компонента *c*	Уровень разрешающей способности *r*	$n = N_L - r$	Вспомогательное значение	Комментарий
0	0	0	2	3	Нет завершенного уровня
1	1	0	2	2	$n = 2$ сейчас завершен
2	2	0	1	2	
3	0	1	1	2	
4	1	1	1	1	$n = 1$ сейчас также завершен
5	2	1	0	1	
6	0	2	0	1	
7	1	2	0	0	Все уровни завершены

K.4 Методы реализаций на основе зоны

K.4.1 Определение сервером соответствующих зон для запроса окна обзора

Когда связь включает в себя тип носителя информации JPP-потока, сервер транслирует запрашиваемую клиентом область изображения в наборе зон, которые являются уместными для запроса. Первая часть этого процесса включает в себя трансляцию параметров fx, fy, sx, sy, ox и oy, поставляемых полями запросов Размера

кадра, Размера области и Смещения области, в параметры размера кадра кодового потока, размера области и смещения fx', fy', sx', sy', ox' и oy', для каждого уместного кодового потока. Эта трансляция продолжается тем же самым способом и для услуг на основе зоны, и для услуг на основе элемента мозаичного изображения и основывается на Уравнениях C-1 и C-2, возможно, измененных согласно Уравнениям C-3 и C-4. Этот подраздел описывает, как серверу следует устанавливать зоны, которые относятся к области, определяемой параметрами fx', fy', sx', sy', ox' и oy', внутри конкретного кодового потока.

Пусть *r* будет неотрицательным целым числом в Уравнении C-1, которое было использовано сервером для нахождения fx' и fy', на основе запроса клиента. Как было упомянуто в связи с таким уравнением, *r* наиболее легко истолковывается как количество сброшенных уровней DWT наивысшей разрешающей способности, даже если *r* позволяется превысить фактическое количество уровней DWT, которые имеются для любого заданного компонента элемента мозаичного изображения. Удобно сначала преобразовать область, описываемую с помощью sx', sy', ox' и oy', в сетку высокой разрешающей способности кодового потока. Это приводит к области, чей левосторонний верхний угол дается с помощью $(E_1^{\text{reg}}, E_2^{\text{reg}})$, и чей правосторонний нижний угол дается с помощью $(F_1^{\text{reg}} - 1, F_2^{\text{reg}} - 1)$, где:

$$E_1^{\text{reg}} = \text{XOsiz} + 2^r \cdot \text{ox}', \quad E_2^{\text{reg}} = \text{YOsiz} + 2^r \cdot \text{oy}', \quad F_1^{\text{reg}} = E_1^{\text{reg}} + 2^r \cdot \text{sx}', \quad \text{и} \quad F_2^{\text{reg}} = E_2^{\text{reg}} + 2^r \cdot \text{sy}'$$

Серверу нужно рассматривать только те элементы мозаичного изображения, которые пересекаются с этой областью на сетке высокой разрешающей способности кодового потока. Для каждого такого элемента мозаичного изображения серверу необходимо рассматривать только те компоненты изображения, которые запрашиваются клиентом, способом, описанным в соединении с полями запроса Компонента и Контекста кодового потока. Для каждого рассматриваемого компонента элемента мозаичного изображения, обозначенного с помощью *t* и *c*, пусть $D_{t,c}$ будет количеством уровней DWT, которые были использованы для сжатия такого компонента элемента мозаичного изображения. Если $D_{t,c} \geq r$, то серверу следует сбросить все зоны, принадлежащие уровням высокой разрешающей способности компонента *r* элемента мозаичного изображения; в противном случае, ему следует сбросить все зоны, принадлежащие уровням самой высокой разрешающей способности $D_{t,c}$ компонента элемента мозаичного изображения, оставляя только те зоны, которые представляют самую нижнюю подполосу LL компонента элемента мозаичного изображения.

Для каждой зоны, которая остается после сбрасывания элементов мозаичного изображения, компонентов и уровней разрешающей способности, упомянутых выше, серверу следует обозначать, способствуют или нет кодовые блоки, которые принадлежат к такой зоне, восстановлению области, определяемой с помощью $E_1^{\text{reg}}, E_2^{\text{reg}}$ и $F_1^{\text{reg}}, F_2^{\text{reg}}$ на сетке высокой разрешающей способности кодового потока. Кодовый блок содействует этой области, если любой из его отсчетов затрагивает восстановление любого отсчета компонента изображения полной разрешающей способности, чьи координаты *(x,y)* удовлетворяют:

$$E_1^{\text{reg}} \leq \text{XRsiz}^c \cdot x < F_1^{\text{reg}} \quad \text{и} \quad E_2^{\text{reg}} \leq \text{YRsiz}^c \cdot y < F_2^{\text{reg}}$$

где XRsiz^c и YRsiz^c обозначают горизонтальный и вертикальный коэффициенты дискретизации на пониженной скорости для уместного компонента, *c*, в сегменте маркера SIZ кодового потока.

Важно иметь в виду, что восстановление компонента изображения полной разрешающей способности затрагивает синтез элементарных волн [*wavelet*], который в своей основе является расширительным процессом. Таким образом, область, которой содействует любая заданная зона, в общем случае, накладывается на области, которым содействуют ее соседние зоны. Серверу следует быть готовым учитывать эти расширительные воздействия преобразования элементарных волн при определении зон, которые относятся к запросу клиента.

Секция 10.6.4 книги "JPEG2000: основные положения, стандарты и практика сжатия изображений" [11] описывает один способ для вычисления отсчетов любой заданной подполосы, которые содействуют заданной области на сетке высокой разрешающей способности кодового потока. Из областей подполос простым вопросом является выведение содействующих кодовых блоков и, следовательно, зон.

K.4.2 Декодирование изображения из сообщений возвращаемых JPP-потоков

Протокол JPIP определяет механизмы для обмена данными сжатых изображений и метаданными между клиентом и сервером. Механизмы для клиента, чтобы отображать возвращаемые данные, не определяются, и конечно, будут широко различаться между приложениями.

K.5 Интерпретации протоколов JPIP

K.5.1 Введение

В следующих примерных интерпретациях текст после символов "<<" в начале строки посылается от клиента к серверу, текст после символов ">>" в начале строки посылается от сервера к клиенту, а текст после символов "--" является комментарием и фактически не передается. Комментарии могут указывать, что некоторые из переданных данных не показаны.

K.5.2 Использование протокола HTTP

Следующая интерпретация показывает пять запросов, посланных от клиента к серверу, и отклик сервера.

Первый запрос запрашивает файл JP2, называемый phoenix.jp2, запрашивается первый кодовый поток в файле, на отклик накладывается максимальная длина, запрашивается идентификатор id цели, запрашиваемые данные должны возвращаться как JPP-поток, и запрашивается установление сеанса поверх протокола HTTP. Окно и, следовательно, данные изображения не запрашиваются.

Сервер отвечает, предоставляя идентификатор ID цели для изображения и идентификатор ID для недавно установленного канала. Строка заголовка, начинающая как "JPIP-cnew", указывает новый тракт, который может использоваться для получения доступа к файлу изображения. Значение для тракта "jpip" может быть трактом к программе CGI на сервере, разработанной, чтобы иметь дело со всеми диалоговыми командами JPIP. Некоторые данные из файла возвращаются в теле; они будут блоками файловых форматов и, возможно, главным заголовком первого кодового потока.

2-ой запрос клиента использует новый тракт, "jpip.cgi" и идентификатор ID канала, чтобы обозначать желаемое изображение (никакое имя изображения или ID цели не являются необходимыми). Этот запрос также определяет конкретное интересуемое окно.

Отклик на 2-й запрос указывает, что окно обзора было изменено и возвращается меньшее окно, сосредоточенное в требуемом окне обзора. Сервер начинает возвращать данные для этого окна.

Перед получением завершенного отклика на 2-й запрос клиент выпускает 3-й запрос. Клиент отрегулировал свое окно обзора до размера, указанного сервером.

Некоторое время сервер продолжает откликаться на 2-й запрос, затем начинает отклик на 3-й запрос. В течение этого отклика клиент выпускает 4-й запрос с немного различной областью. Сервер некоторое время продолжает откликаться на 3-й запрос, затем начинает откликаться на 4-й запрос.

Клиент ждет, пока не закончился 4-ый отклик, затем выпускает запрос для завершения и сеанса, и соединения HTTP. Данные отклика в этом случае не показываются, поскольку соединение закрывается.

```
<< GET /phoenix.jp2?stream=0&len=2000&tid=0&type=jpp-stream&cnew=http
HTTP/1.1
<< Host: dst-m
<<
    >> HTTP/1.1 200 OK
    >> JPIP-tid: 281B6E135135BBC0BC588452AC9B73C5
    >> JPIP-cnew: cid=JPH_033C38BE48115AC9,path=jpip.cgi,transport=http
    >> Cache-Control: no-cache
    >> Transfer-Encoding: chunked
    >> Content-Type: image/jpp-stream
    >>
    >> 102
       -- 258 bytes of binary data
    >> 0
    >>
<< GET /jpip.cgi?fsiz=834,834&roff=0,0&rsiz=834,790&comps=0-
2&stream=0&len=2000&cid=JPH_033C38BE48115AC9 HTTP/1.1
<< Host: dst-m
<< Cache-Control: no-cache
<<
    >> HTTP/1.1 200 OK, with modifications
    >> JPIP-roff: 120,114
    >> JPIP-rsiz: 593,561
    >> Cache-Control: no-cache
    >> Transfer-Encoding: chunked
```

```
    >> Content-Type: image/jpp-stream
    >>
    >> 393
       -- 915 bytes of binary data
<< GET /jpip.cgi?fsiz=834,834&roff=120,114&rsiz=593,561&comps=0-
2&stream=0&len=2000&cid=JPH_033C38BE48115AC9 HTTP/1.1
<< Host: dst-m
<< Cache-Control: no-cache
<<
    >> 3f9
       -- 1017 bytes of binary data
    >> 0
    >>
    >> HTTP/1.1 200 OK
    >> Cache-Control: no-cache
    >> Transfer-Encoding: chunked
    >> Content-Type: image/jpp-stream
    >>
    >> 359
       -- 857 bytes of binary data
<< GET /jpip.cgi?fsiz=834,834&roff=309,297&rsiz=121,86&comps=0-
2&stream=0&len=3906&cid=JPH_033C38BE48115AC9 HTTP/1.1
<< Host: dst-m
<< Cache-Control: no-cache
<<
    >> 234
       -- 564 bytes of binary data
    >> 3d0
       -- 976 bytes of binary data
    >> 24f
       -- 591 bytes of binary data
    >> 0
    >>
    >> HTTP/1.1 200 OK
    >> Cache-Control: no-cache
    >> Transfer-Encoding: chunked
    >> Content-Type: image/jpp-stream
    >>
    >> 3b2
       -- 946 bytes of binary data
    >> 400
       -- 1024 bytes of binary data
    >> 263
       -- 611 bytes of binary data
    >> 356
       -- 854 bytes of binary data
    >> 209
       -- 521 bytes of binary data
    >> 0

<< GET /jpip.cgi?cclose=JPH_033C38BE48115AC9&len=0 HTTP/1.1
<< Host: dst-m
<< Connection: close
<< Cache-Control: no-cache
<<
```

Нижеследующее является примером GET на основе сеанса HTTP с запросом модели.

```
<< GET /jpip.cgi?fsiz=1024,768&cid=JPH_5&model=Hm,H*,M*,P* HTTP/1.1
<< Host: jpip.jpeg.org
<< Cache-Control: no-cache

    >> HTTP/1.1 200 OK
    >> Cache-control: no-cache
```

```
    >> Transfer-Encoding: chunked
    >> 3
    -- 3 bytes of binary data
    >> 0
```

Нижеследующее является примером запроса GET протокола HTTP, не меняющего своего состояния в процессе исполнения, с запросом модели.

```
<< GET /images/kids.jp2?fsiz=1024,768&model=M0,Hm,H0:20,P0 HTTP/1.1
<< Host: jpip.jpeg.org
<< Cache-Control: no-cache

    >> HTTP/1.1 200 OK
    >> Cache-Control: no-cache
    >> Transfer-Encoding: chunked
    >> Content-Type: image/jpp-stream
    >> 400
    -- 1024 bytes of binary data
    >> 3f8
    -- 1016 bytes of binary data
    >> 0
```

K.5.3 Использование HTTP с возвратом TCP

```
<< GET /phoenix.jp2?stream=0&len=2000&tid=0&type=jpp-stream&cnew=http-
tcp,http HTTP/1.1
<< Host: dst-m
<<
    >> HTTP/1.1 200 OK
    >> JPIP-tid: 281B6E135135BBC0BC588452AC9B73C5
    >> JPIP-cnew: cid=JPHT033C38BE481154F9,path=jpip,transport=http-
       tcp,auxport=80
    >> Cache-Control: no-cache
    >>
        << JPHT033C38BE481154F9 – [Примечание. – Это сообщение соединения
канала TCP]
        <<

<< GET /jpip.cgi?fsiz=834,834&roff=0,0&rsiz=834,790&comps=0-
2&stream=0&cid=JPHT033C38BE481154F9 HTTP/1.1
<< Host: dst-m
<< Cache-Control: no-cache
<<
    >> HTTP/1.1 200 OK, with modifications
    >> JPIP-roff: 120,114
    >> JPIP-rsiz: 593,561
    >> Cache-Control: no-cache
    >>

<< GET /jpip.cgi?fsiz=834,834&roff=229,254&rsiz=155,113&comps=0-
2&stream=0&cid=JPHT033C38BE481154F9 HTTP/1.1
<< Host: dst-m
<< Cache-Control: no-cache
<<
    >> HTTP/1.1 200 OK
    >> Cache-Control: no-cache
    >>

<< GET /jpip.cgi?fsiz=1667,1667&roff=457,507&rsiz=310,226&comps=0-
2&stream=0&cid=JPHT033C38BE481154F9 HTTP/1.1
<< Host: dst-m
<< Cache-Control: no-cache
<<
    >> HTTP/1.1 200 OK
```

```
    >> Cache-Control: no-cache
    >>

<< GET /jpip.cgi?fsiz=3334,3334&roff=914,1014&rsiz=620,452&comps=0-
2&stream=0&cid=JPHT033C38BE481154F9 HTTP/1.1
<< Host: dst-m
<< Cache-Control: no-cache
<<
    >> HTTP/1.1 200 OK
    >> Cache-Control: no-cache
    >>

<< GET /jpip.cgi?cclose=JPHT033C38BE481154F9 HTTP/1.1
<< Host: dst-m
<< Cache-Control: no-cache
<<
```

K.6 Использование протокола JPIP с языком HTML

Система JPIP может использоваться со страницами языков HTML и XHTML разнообразными способами. Если сервер JPIP включает в себя способность к перекодированию порций изображения в JPEG или другие типы носителей полного изображения, то тогда язык HTML может использоваться для получения доступа к порциям изображения JPEG 2000 без любых изменений к текущим окнам просмотра (браузерам) [*browsers*].

Рассмотрим вэб-страницу, содержащую следующий фрагмент HTML:

```
<img

src="http://jpip.jpeg.org/name.jp2?fsiz=128,128&rsiz=128,128&type=image/jpeg"
width="128" height="128">
```

Любой вэб-браузер [окно просмотра "Всемирной паутины"], желающий отобразить свою вэб-страницу с изображениями, будет выпускать запрос для получения изображения. Этот запрос будет начинаться так:

```
GET /name.jp2?fsiz=128,128&rsiz=128,128&type=image/jpeg
Host: jpip.jpeg.org
```

и будет включать много других строк заголовков HTTP, типовым образом обозначая окно просмотра и типы предметов, которые принимает окно просмотра. Этот запрос HTTP является законным запросом JPIP и сервер JPIP, который получает этот запрос, должен или возвратить сообщение ошибки, или определить соответствующую порцию файла JP2 для обеспечения доступа и трансляции ее в файл JPEG. Возвращаемое сообщение могло бы выглядеть так:

```
HTTP/1.1 200 OK
Content-type: image/jpeg
Content-length: 20387
CRLF
JPEG-Compressed-Image-Data
```

Это сообщение является законным откликом JPIP, а также является законным откликом HTTP, который все окна просмотра изображений знают, как показывать. Отметим, что предпочитается, но не требуется для сервера, использовать кодирование переноса фрагментированных данных так, чтобы этот запрос мог быть прерван. Предыдущий пример не является примером кодирования переноса фрагментированных данных.

Есть также возможность записывать вэб-страницы, которые будут использовать JPEG, когда имеется только JPEG, использовать JPEG 2000, когда имеется, и JPT-поток или JPP-поток, когда они имеются в окне просмотра клиента. Рассмотрим фрагмент HTML:

```
<img src="http://jpip.jpeg.org/name.jp2?rsiz=128,128" width="128"
     height="128">
```

В этом случае нет явного запрашиваемого типа. Серверу JPIP, использующему протокол HTTP, поэтому следует изучить строку запроса "Accept:" протокола HTTP, выпущенную клиентом. В зависимости от присутствия image/jp2, или image/jpt-stream, или image/jpp-stream, или image/jpeg, сервер может определить совместимый формат для возвращения.

Приложение L

Совокупность ABNF JPIP

(Это приложение не образует составную часть этой Рекомендации | Международного стандарта)

L.1 ABNF запрос JPIP

```
;================================
; C.1.1 Request structure
;================================

jpip-request-field = target-field
                   / channel-field
                   / view-window-field
                   / metadata-field
                   / data-limit-field
                   / server-control-field
                   / cache-management-field
                   / upload-field
                   / client-cap-pref-field

target-field       = target                          ; C.2.2
                   / subtarget                       ; C.2.3
                   / tid                             ; C.2.4

channel-field      = cid                             ; C.3.2
                   / cnew                            ; C.3.3
                   / cclose                          ; C.3.4
                   / qid                             ; C.3.5

view-window-field  = fsiz                            ; C.4.2
                   / roff                            ; C.4.3
                   / rsiz                            ; C.4.4
                   / comps                           ; C.4.5
                   / stream                          ; C.4.6
                   / context                         ; C.4.7
                   / srate                           ; C.4.8
                   / roi                             ; C.4.9
                   / layers                          ; C.4.10

metadata-field     = metareq                         ; C.5.2

data-limit-field   = len                             ; C.6.1
                   / quality                         ; C.6.2

server-control-field = align                         ; C.7.1
                   / wait                            ; C.7.2
                   / type                            ; C.7.3
                   / drate                           ; C.7.4

cache-management-field = model                       ; C.8.1
                   / tpmodel                         ; C.8.3
                   / need                            ; C.8.4
                   / tpneed                          ; C.8.5
                   / mset                            ; C.8.6

upload-field       = upload                          ; C.9.1
```

```
client-cap-pref-field = cap                          ; C.10.1
                      / pref                         ; C.10.2
                      / csf                          ; C.10.3;
================================
; C.2.2 Target(target)
;================================

target = "target" "=" PATH

;================================
; C.2.3 Sub-target (subtarget)
;================================

subtarget = "subtarget" "=" byte-range

byte-range = UINT-RANGE

;================================
; C.2.4 Target ID (tid)
;================================

tid = "tid" "=" target-id

target-id = TOKEN

;================================
; C.3.1 Channel ID (cid)
;================================

cid = "cid" "=" channel-id

channel-id = TOKEN

;================================
; C.3.2 New Channel (cnew)
;================================

cnew = "cnew" "=" 1#transport-name

transport-name = TOKEN

;================================
; C.3.3 Channel Close (cclose)
;================================

cclose = "cclose" "=" ("*" / 1#channel-id)

;================================
; C.3.4 Request ID (qid)
;================================

qid = "qid" "=" UINT

;================================
; C.4.2 Frame Size (fsiz)
;================================

fsiz = "fsiz" "=" fx "," fy ["," round-direction]

fx = UINT

fy = UINT

round-direction = "round-up" / "round-down" / "closest"

;================================
; C.4.3 Offset (roff)
;================================

roff = "roff" "=" ox "," oy

ox = UINT

oy = UINT
```

```
;================================
; C.4.4 Region Size (rsiz)
;================================

rsiz = "rsiz" "=" sx "," sy

sx = UINT

sy = UINT

;================================
; C.4.5 Components (comps)
;================================

comps = "comps" "=" 1#UINT-RANGE

;================================
; C.4.6 Codestream (stream)
;================================

stream = "stream" "=" 1#sampled-range

sampled-range = UINT-RANGE [":" sampling-factor]

sampling-factor = UINT

;================================
; C.4.7 Codestream Context (context)
;================================

context = "context" "=" 1#context-range

context-range = jpxl-context-range / mj2t-context / reserved-context

jpxl-context-range = "jpxl" "<" jpx-layers ">" [ "[" jpxl-geometry "]" ]

jpx-layers = sampled-range

jpxl-geometry = "s" jpx-iset "i" jpx-inum

jpx-iset = UINT

jpx-inum = UINT

mj2t-context = "mj2t" "<" mj2-track ">" [ "[" mj2t-geometry "]" ]

mj2-track = NONZERO ["+" "now" ]

mj2t-geometry = "track" / "movie"

reserved-context = 1*( TOKEN / "<" / ">" / "[" / "]" / "-" / ":" / "+" )

;================================
; C.4.8 Sampling Rate (srate)
;================================

srate = "srate" "=" streams-per-second

streams-per-second = UFLOAT

;================================
; C.4.9 ROI (roi)
;================================

roi = "roi" "=" region-name

region-name = 1*(DIGIT / ALPHA / "_")
            / "dynamic"

;================================
; C.4.10 Layers (layers)
;================================

layers = "layers" "=" UINT
```

```
;================================
; C.5.2 Metadata Request (metareq)
;================================

metareq = "metareq" "=" 1#("[" 1$(req-box-prop) "]" [root-bin] [max-depth])
                        [metadata-only]

req-box-prop = box-type [limit] [metareq-qualifier] [priority]

limit = ":" (UINT / "r")

metareq-qualifier = "/" 1*("w" / "s" / "g" / "a")

priority = "!"

root-bin = "R" UINT

max-depth = "D" UINT

metadata-only = "!!"

;================================
; C.6.1 Maximum Response Length (len)
;================================

len = "len" "=" UINT

;================================
; C.6.2 Quality (quality)
;================================

quality = "quality" "=" (1*2DIGIT / "100")        ; 0 to 100

;================================
; C.7.1 Alignment (align)
;================================

align = "align" "=" ("yes" / "no")

;================================
; C.7.2 Wait (wait)
;================================

wait = "wait" "=" ("yes" / "no")

;================================
; C.7.3 Image Return Type (type)
;================================

type = "type" "=" 1#image-return-type

image-return-type = media-type / reserved-image-return-type

media-type = TOKEN "/" TOKEN *( ";" parameter )

reserved-image-return-type = TOKEN *( ";" parameter )

parameter = attribute "=" value

attribute = TOKEN

value = TOKEN

;================================
; C.7.4 Delivery Rate (drate)
;================================

drate = "drate" "=" rate-factor

rate-factor = UFLOAT
```

```
;================================
; C.8.1.1 Model (model)
;================================

model = "model" "=" 1#model-item

model-item = [codestream-qualifier ","] model-element

model-element = ["-"] bin-descriptor

bin-descriptor = explicit-bin-descriptor    ; C.8.1.2
               / implicit-bin-descriptor    ; C.8.1.3

codestream-qualifier = "[" 1$(codestream-range) "]"

codestream-range = first-codestream-id ["-" [last-codestream-id]]

first-codestream-id = UINT

last-codestream-id = UINT

;================================
; C.8.1.2 Explicit Form
;================================

explicit-bin-descriptor = explicit-bin
                          [":" (number-of-bytes / number-of-layers )]

explicit-bin = codestream-main-header-bin
             / meta-bin
             / tile-bin
             / tile-header-bin
             / precinct-bin

number-of-bytes = UINT

number-of-layers = %x4c UINT                        ; "L"

codestream-main-header-bin = %x48 %x6d ; "Hm"

meta-bin = %x4d bin-uid                                  ; "M"

tile-bin = %x54 bin-uid                                  ; "T"

tile-header-bin = %x48 bin-uid                      ; "H"

precinct-bin = %x50 bin-uid                         ; "P"

bin-uid = UINT / "*"

;================================
; C.8.1.3 Implicit Form
;================================

implicit-bin-descriptor = 1*implicit-bin [":" number-of-layers]

implicit-bin  = implicit-bin-prefix (data-uid / index-range-spec)

implicit-bin-prefix = %x74    ; t -- tile
                    / %x63    ; c -- component
                    / %x72    ; r -- resolution level
                    / %x70    ; p -- position

index-range-spec = first-index-pos "-" last-index-pos

first-index-pos = UINT

last-index-pos = UINT

data-uid = UINT / "*"
```

```
;================================
; C.8.3 Tile-part Model involving JPT-streams (tpmodel)
;================================
tpmodel =  "tpmodel" "=" 1#tpmodel-item
tpmodel-item = [codestream-qualifier "," ] tpmodel-element
tpmodel-element = ["-"] tp-descriptor
tp-descriptor = tp-range / tp-number
tp-range = tp-number "-" tp-number
tp-number = tile-number "." part-number
tile-number = UINT
part-number = UINT
;================================
; C.8.4 Need for Stateless Requests (need)
;================================
need = "need" "=" 1#need-item
need-item = [codestream-qualifier "," ] bin-descriptor
;================================
; C.8.5 Tile-part Need for Stateless Requests (tpneed)
;================================
tpneed = "tpneed" "=" 1#tpneed-item
tpneed-item = [codestream-qualifier "," ] tp-descriptor
;================================
; C.8.6 Model Set for Requests within a session (mset)
;================================
mset =  "mset" "=" 1#sampled-range
;================================
; C.9.1 Upload (upload)
;================================
upload = "upload" "=" upload-type
upload-type = image-return-type                    ; C.7.3
;================================
; C.10.1 Client Capability (cap)
;================================
cap = "cap" "=" 1#capability-group
capability-group = processing-capability
                 / depth-capability
                 / config-capability
processing-capability = compatibility-capability
                      / vendor-capability
compatibility-capability = "cc." compatibility-code
vendor-capability = "vc." vendor-code [":" vendor-value]
vendor-code = 1*(LOWER / DIGIT / "." / "-")
vendor-value = TOKEN
depth-capability = "depth:" UINT
config-capability = "config:" UINT
```

```
;================================
; C.10.2.1 General
;================================

pref = "pref" "=" 1#(related-pref-set ["/r"])

related-pref-set = view-window-pref            ; C.10.2.2
                 / colour-meth-pref            ; C.10.2.3
                 / max-bandwidth               ; C.10.2.4
                 / bandwidth-slice             ; C.10.2.5
                 / placeholder-pref            ; C.10.2.6
                 / codestream-seq-pref         ; C.10.2.7
                 / other

other = TOKEN

;================================
; C.10.2.2 View-window handling preferences
;================================

view-window-pref = "fullwindow" / "progressive"

;================================
; C.10.2.3 Colour space method preference
;================================

color-meth-pref = 1$(color-meth [":" meth-limit])

color-meth = "color-enum" / "color-ricc" / "color-icc" / "color-vend"

meth-limit = UINT

;================================
; C.10.2.4 Max bandwidth
;================================

max-bandwidth = "mbw:" mbw

mbw = UINT ["K" / "M" / "G" / "T"]

;================================
; C.10.2.5 Bandwidth slice
;================================

bandwidth-slice = "slice:" slice

slice = NONZERO

;================================
; C.10.2.6 Placeholder preference
;================================

placeholder-pref = "meta:" placeholder-branch

placeholder-branch = "incr" / "equiv" / "orig"

;================================
; C.10.2.7 Codestream sequencing
;================================

codestream-seq-pref = "codeseq:" codestream-seq-option

codestream-seq-option = "sequential" / "reverse-sequential"
                      / "interleaved"

;================================
; C.10.3 Contrast sensitivity (csf)
;================================

csf = "csf" "=" 1#csf-sample-line

csf-sample-line = csf-density [";" csf-angle] ";" 1$sensitivity

csf-density = "density" ":" UFLOAT
```

```
csf-angle = "angle" ":" UFLOAT

sensitivity = UFLOAT
```

L.2 BNF отклик JPIP

```
;================================
; D.1.1 Reply structure
;================================

Status-Code = 3DIGIT

Reason-Phrase = *<TEXT, excluding CR and LF>

jpip-response-header =
                      / JPIP-tid                    ; D.2.2
                      / JPIP-cnew                   ; D.2.3
                      / JPIP-qid                    ; D.2.4
                      / JPIP-fsiz                   ; D.2.5
                      / JPIP-rsiz                   ; D.2.6
                      / JPIP-roff                   ; D.2.7
                      / JPIP-comps                  ; D.2.8
                      / JPIP-stream                 ; D.2.9
                      / JPIP-context                ; D.2.10
                      / JPIP-roi                    ; D.2.11
                      / JPIP-layers                 ; D.2.12
                      / JPIP-srate                  ; D.2.13
                      / JPIP-metareq                ; D.2.14
                      / JPIP-len                    ; D.2.15
                      / JPIP-quality                ; D.2.16
                      / JPIP-type                   ; D.2.17
                      / JPIP-mset                   ; D.2.18
                      / JPIP-cap                    ; D.2.19
                      / JPIP-pref                   ; D.2.20

;================================
; D.2.2 Target ID (JPIP-tid)
;================================

JPIP-tid = "JPIP-tid" ":" LWSP target-id

;================================
; D.2.3 New Channel (JPIP-cnew)
;================================

JPIP-cnew = "JPIP-cnew" ":" LWSP "cid" "=" channel-id
            ["," 1#(transport-param "=" TOKEN)]

transport-param = TOKEN

;================================
; D.2.4 Request ID (JPIP-qid)
;================================

JPIP-qid = "JPIP-qid" ":" LWSP UINT

;================================
; D.2.5 Frame Size (JPIP-fsiz)
;================================

JPIP-fsiz = "JPIP-fsiz" ":" LWSP fx "," fy

;================================
; D.2.6 Region Size (JPIP-rsiz)
;================================

JPIP-rsiz = "JPIP-rsiz" ":" LWSP sx "," sy
```

```
;================================
; D.2.7 Offset (JPIP-roff)
;================================

JPIP-roff = "JPIP-roff" ":" LWSP ox "," oy

;================================
; D.2.8 Components (JPIP-comps)
;================================

JPIP-comps = "JPIP-comps" ":" LWSP 1#UINT-RANGE

;================================
; D.2.9 Codestream (JPIP-stream)
;================================

JPIP-stream = "JPIP-stream" ":" LWSP 1#(prefixed-range / sampled-range)
prefixed-range = "<" ctxt-id ":" ctxt-elt ">" sampled-range
ctxt-id = UINT
ctxt-elt = UINT

;================================
; D.2.10 Codestream Context (JPIP-context)
;================================

JPIP-context = "JPIP-context" ":" LWSP 1$(context-range "=" 1#sampled-range)

;================================
; D.2.11 ROI (JPIP-roi)
;================================

JPIP-roi = "JPIP-roi" ":" LWSP
           "roi" "=" region-name ";"
           "fsiz" "=" UINT "," UINT ";"
           "rsiz" "=" UINT "," UINT ";"
           "roff" "=" UINT "," UINT ";"

region-name = 1*(DIGIT / ALPHA / "_")

;================================
; D.2.12 Layers (JPIP-layers)
;================================

JPIP-layers = "JPIP-layers" ":" LWSP UINT

;================================
; D.2.13 Sampling Rate (JPIP-srate)
;================================

JPIP-srate = "JPIP-srate" ":" LWSP UFLOAT

;================================
; D.2.14 Metadata request (JPIP-metareq)
;================================

JPIP-metareq = "JPIP-metareq" ":" LWSP
               1#( "[" 1$(req-box-prop) "]" [root-bin] [max-depth] )
               [metadata-only]

req-box-prop = box-type [limit] [metareq-qualifier] [priority]

;================================

; D.2.15 Maximum Response Length (JPIP-length)

;================================

JPIP-len = "JPIP-len" ":" LWSP UINT

;================================
; D.2.16 Quality (JPIP-quality)
;================================
```

```
JPIP-quality = "JPIP-quality" ":" LWSP (1*2DIGIT / "100" / "-1")

;================================
; D.2.17 Image Return Type (JPIP-type)
;================================

JPIP-type = "JPIP-type" ":" LWSP image-return-type

;================================
; D.2.18 Model Set (JPIP-mset)
;================================

JPIP-mset = "JPIP-mset" ":" LWSP 1#sampled-range

;================================
; D.2.19 Needed Capability (JPIP-cap)
;================================

JPIP-cap = "JPIP-cap" ":" LWSP 1#capability-code

;================================
; D.2.20 Unavailable Preference (JPIP-pref)
;================================

JPIP-pref = "JPIP-pref" ":" LWSP 1#related-pref-set
```

Приложение M

Патентные заявления

(Это приложение не образует составную часть этой Рекомендации | Международного стандарта)

Международная организация по стандартизации (ИСО) и Международная электротехническая комиссия (МЭК) обращают внимание на тот факт, что считается, что согласование с этой частью Рекомендации | Международного стандарта 15444 может включать в себя использование патентов.

ИСО и МЭК не занимают определенной позиции относительно очевидности, законности и сферы применения этих патентных прав.

Держатели этих патентных прав уверили ИСО и МЭК, что они желают договариваться о лицензиях с разумными и справедливыми условиями с претендентами во всем мире. В этом отношении, заявления держателей этих патентных прав зарегистрированы с помощью ИСО и МЭК. Информация может быть получена от компаний, перечисленных ниже.

Обращается внимание на возможность, что некоторые из элементов этой части ИСО/МИЭК 15444 могут быть предметом патентных прав, отличающихся от тех, что определены в этом приложении. ИСО и МЭК не должны считаться ответственными за опознание каких-либо или всех таких патентных прав.

	Компания
1	Canon Inc.
2	Ricoh Company, Limited.

Приложение N

Библиография

(Это приложение не образует составную часть этой Рекомендации | Международного стандарта)

[1] TAUBMAN (D.): Remote Browsing of JPEG 2000 Images, *Proc. Int. Conf. on Image Processing*, Vol. 1, pp. 229-232, Sept. 2002.

[2] LI (J.), SUN (H.), LI (H.), ZHANG (Q.), LIN (X.): Vfile – A Virtual File Media Access Mechanism and its Application in JPEG2000 Images for Browsing over Internet, *ISO/IEC JTC 1/SC 29/WG 1 Document Register: N1473*, Nov. 1999.

[3] BOLIEK (M.), WU (G.K.), GORMISH (M.J.): JPEG 2000 for Efficient Imaging in a Client/Server Environment, *Proc. SPIE Conf. on Applications of Digital Image Processing*, Vol. 4472, pp. 212-223, Dec. 2001.

[4] DESHPANDE (S.), ZENG (W.): Scalable Streaming of JPEG2000 Images Using Hypertext Transfer Protocol, *Proc. ACM Conf. on Multimedia*, pp. 372-381, Oct. 2001.

[5] WRIGHT (A.), CLARK (R.), COLYER (G.): An Implementation of JPIP Based on HTTP, *ISO/IEC JTC 1/SC 29/WG 1 Document Register: N2426*, Feb. 2002.

[6] GORMISH (M.), BANERJEE (S.): Tile-Based Transport of JPEG 2000, N. Garcia, J.M. Martinez, L. Salgado (Eds.), VLVB03, LNCS 2849, pp. 217-224, 2003.

[7] TAUBMAN (D.), ROSENBAUM (R.): Rate-Distortion Optimized Interactive Browsing of JPEG2000 Images, *Proc. Int. Conf. on Image Processing*, Sept. 2003.

[8] TAUBMAN (D.), PRANDOLINI (R.): Architecture, Philosophy and Performance of JPIP: Internet Protocol Standard for JPEG2000, presented at *Visual Communications and Image Processing*, Lugano, Switzerland, 2003.

[9] GORMISH (M.J.): TRUEW: Transport of Reversible and Unreversible EmbeddedWavelets (A JPIP Proposal), *ISO/IEC JTC 1/SC 29/WG 1 Document Register: N2602*, July 2002.

[10] CANON: Proposal for JPIP Tier 2 protocol, *ISO/IEC JTC 1/SC 29/WG 1 Document Register: N2608*, June 2002.

[11] TAUBMAN (D.), MARCELLIN (M.): JPEG2000: image compression fundamentals, standards and practice, *Kluwer Academic Publishers*, Boston, 2001.

СЕРИИ РЕКОМЕНДАЦИЙ МСЭ-Т

Серия A	Организация работы МСЭ-Т
Серия D	Общие принципы тарификации
Серия E	Общая эксплуатация сети, телефонная служба, функционирование служб и человеческие факторы
Серия F	Нетелефонные службы электросвязи
Серия G	Системы и среда передачи, цифровые системы и сети
Серия H	Аудиовизуальные и мультимедийные системы
Серия I	Цифровая сеть с интеграцией служб
Серия J	Кабельные сети и передача сигналов телевизионных и звуковых программ и других мультимедийных сигналов
Серия K	Защита от помех
Серия L	Конструкция, прокладка и защита кабелей и других элементов линейно-кабельных сооружений
Серия M	Управление электросвязью, включая СУЭ и техническое обслуживание сетей
Серия N	Техническое обслуживание: международные каналы передачи звуковых и телевизионных программ
Серия O	Требования к измерительной аппаратуре
Серия P	Качество телефонной передачи, телефонные установки, сети местных линий
Серия Q	Коммутация и сигнализация
Серия R	Телеграфная передача
Серия S	Оконечное оборудование для телеграфных служб
Серия T	**Оконечное оборудование для телематических служб**
Серия U	Телеграфная коммутация
Серия V	Передача данных по телефонной сети
Серия X	Сети передачи данных, взаимосвязь открытых систем и безопасность
Серия Y	Глобальная информационная инфраструктура, аспекты межсетевого протокола и сети последующих поколений
Серия Z	Языки и общие аспекты программного обеспечения для систем электросвязи

Отпечатано в Швейцарии
Женева, 2006 г.