ГОСТ Р ИСО 10303-21—2002

ГОСУДАРСТВЕННЫЙ СТАНДАРТ РОССИЙСКОЙ ФЕДЕРАЦИИ

Системы автоматизации производства и их интеграция

ПРЕДСТАВЛЕНИЕ ДАННЫХ ОБ ИЗДЕЛИИ И ОБМЕН ЭТИМИ ДАННЫМИ

Часть 21

Методы реализации Кодирование открытым текстом структуры обмена

Издание официальное

ГОССТАНДАРТ РОССИИ
Москва

Предисловие

1 РАЗРАБОТАН Научно-исследовательским центром (НИЦ) CALS-технологий «Прикладная логистика» и Всероссийским научно-исследовательским институтом стандартизации (ВНИИстандарт)

ВНЕСЕН Техническим комитетом по стандартизации ТК 431 «CALS-технологии»

2 ПРИНЯТ И ВВЕДЕН В ДЕЙСТВИЕ Постановлением Госстандарта России от 20 декабря 2002 г. № 496-ст

3 Настоящий стандарт представляет собой аутентичный текст международного стандарта ИСО 10303-21:2002 «Системы автоматизации производства и их интеграция. Представление данных об изделии и обмен этими данными. Часть 21. Методы реализации. Кодирование открытым текстом структуры обмена»

4 ВЗАМЕН ГОСТ Р ИСО 10303-21—99

5 ПЕРЕИЗДАНИЕ. Май 2006 г.

Содержание

Введение

Стандарты серии ГОСТ Р ИСО 10303 распространяются на машинно-ориентированное представление данных об изделии и обмен этими данными. Целью является создание механизма, позволяющего описывать данные об изделии на протяжении всего его жизненного изделия независимо от конкретной системы. Характер такого описания делает его пригодным не только для обмена инвариантными файлами, но также и для создания баз данных об изделиях, коллективного пользования этими базами и архивации соответствующих данных.

Настоящий стандарт устанавливает механизм, который позволяет представлять данные об изделии для передачи из одной вычислительной системы в другую, используя язык EXPRESS, описанный в ГОСТ Р ИСО 10303-11.

Основные разделы настоящего стандарта:

- определение синтаксиса структуры обмена;
- преобразование из EXPRESS-схемы в заданный синтаксис.

Примечания

1 Примеры использования EXPRESS в настоящем стандарте не соответствуют каким-либо правилам стиля. Напротив, иногда в примерах используют искаженный стиль, чтобы сохранить место или сконцентрироваться на важных вопросах. Примеры не ставят целью отразить содержание информационных моделей, определенных в других частях стандартов серии ГОСТ Р ИСО 10303. Данные примеры предназначены для показа конкретных особенностей EXPRESS или структуры обмена. Многие примеры даны с аннотациями, не согласующимися с синтаксическими правилами настоящего стандарта. Такие аннотации введены символическими стрелками: или горизонтальными «→», или вертикальными «↑». При составлении правил просмотра текста эти аннотации должны быть игнорированы. Должно быть также игнорировано любое сходство между примерами и нормативными моделями, определенными в других стандартах серии ГОСТ Р ИСО 10303. В настоящем стандарте приведены несколько примеров отображения. В некоторые примеры для улучшения читаемости вставлены дополнительные пробелы и новые строки. Эти пробелы и новые строки не должны появляться в структуре обмена.

2 Стандарт дополнен следующими приложениями:

- A — описывающим правила представления файла на носителе данных;
- B — описывающим соглашения по синтаксической нотации Вирта;
- C — устанавливающим идентификаторы информационных объектов, присвоенные настоящему стандарту и описанной в нем схеме;
- D — описывающим основной латинский алфавит и набор используемых графических символов;
- E — описывающим форму заявки о соответствии реализации протоколу (ЗСРП);
- F — описывающим множество EXPRESS-схем в структуре обмена;
- G — содержащим руководство по распечатке структуры обмена;
- H — содержащим пример полной структуры обмена.

3 В настоящем стандарте объекты и конструкции языка EXPRESS в ряде случаев выделены полужирным шрифтом (например, **file_description**).

По сравнению с ГОСТ Р ИСО 10303-21—99 в настоящий стандарт внесены следующие изменения:

- исключена структура SCOPE (&SCOPE/ENDSCOPE);
- в структуру обмена может быть внесено несколько секций (разделов) данных;
- в заголовочной секции структуры обмена по умолчанию может быть задан язык описания строковых атрибутов экземпляров объектов (сущностей), закодированных в секции данных;
- в заголовочной секции структуры обмена может быть задана информация, описывающая контексты кодирования экземпляров объектов, используемые в секции данных;
- значения перечислений могут кодироваться с использованием соответствующих сокращенных наименований (при их наличии).

Все структуры обмена, закодированные согласно ГОСТ Р 10303-21—99 и не входящие в структуру SCOPE, соответствуют настоящему стандарту.

ГОСУДАРСТВЕННЫЙ СТАНДАРТ РОССИЙСКОЙ ФЕДЕРАЦИИ

Системы автоматизации производства и их интеграция

ПРЕДСТАВЛЕНИЕ ДАННЫХ ОБ ИЗДЕЛИИ И ОБМЕН ЭТИМИ ДАННЫМИ

Часть 21

Методы реализации. Кодирование открытым текстом структуры обмена

Industrial automation systems and integration. Product data representation and exchange.
Part 21. Implementation methods. Clear text encoding of the exchange structure

Дата введения 2003—07—01

1 Область применения

Настоящий стандарт устанавливает формат структуры обмена, использующий кодирование открытым текстом данных об изделии, для которого концептуальная модель определена в языке EXPRESS (ГОСТ Р ИСО 10303-11). Формат обмена пригоден для передачи данных об изделии между вычислительными системами.

Определено преобразование из языка EXPRESS в синтаксис структуры обмена. В синтаксис структуры обмена может быть преобразована любая EXPRESS-схема.

2 Нормативные ссылки

В настоящем стандарте использованы ссылки на следующие стандарты:

ГОСТ Р ИСО/МЭК 8824-1—2001 Информационная технология. Абстрактная синтаксическая нотация версии один (АСН.1). Часть 1. Спецификация основной нотации

ГОСТ Р ИСО 10303-1—99 Системы автоматизации производства и их интеграция. Представление данных об изделии и обмен этими данными. Часть 1. Общие представления и основополагающие принципы

ГОСТ Р ИСО 10303-11—2000 Системы автоматизации производства и их интеграция. Представление данных об изделии и обмен этими данными. Часть 11. Методы описания. Справочное руководство по языку EXPRESS

ГОСТ Р ИСО 10303-41—99 Системы автоматизации производства и их интеграция. Представление данных об изделии и обмен этими данными. Часть 41. Интегрированные обобщенные ресурсы. Основы описания и поддержки изделий

ИСО 639-2—98* Коды для представления наименований языков. Часть 2. Код Alpha-3

ИСО 3788—90* Обработка информации. 9-дорожечная магнитная лента для обмена информацией шириной 12,7 мм (0,5 дюйма) с использованием фазового кодирования плотностью 126 перехода потока на миллиметр (3200 переходов потока на дюйм) — 63 символа на миллиметр (1600 символов на дюйм)

ИСО 8601—2000* Элементы данных и форматы обмена. Обмен информацией. Представление дат и времени

ИСО/МЭК 8859-1—98* Обработка информации. 8-битные однобайтные кодированные наборы графических символов. Часть 1. Латинский алфавит № 1

ИСО/МЭК 8859-2—99* Обработка информации. 8-битные однобайтные кодированные наборы графических символов. Часть 2. Латинский алфавит № 2

* Оригиналы стандартов ИСО (ИСО/МЭК) — во ВНИИКИ Госстандарта России.

Издание официальное

ИСО/МЭК 8859-3—99* Обработка информации. 8-битные однобайтные кодированные наборы графических символов. Часть 3. Латинский алфавит № 3

ИСО/МЭК 8859-4—99* Обработка информации. 8-битные однобайтные кодированные наборы графических символов. Часть 4. Латинский алфавит № 4

ИСО/МЭК 8859-5—99* Обработка информации. 8-битные однобайтные кодированные наборы графических символов. Часть 5. Алфавит латинский / кириллица

ИСО/МЭК 8859-6—99* Обработка информации. 8-битные однобайтные кодированные наборы графических символов. Часть 6. Алфавит латинский / арабский

ИСО/МЭК 8859-7—87* Обработка информации. 8-битные однобайтные кодированные наборы графических символов. Часть 7. Алфавит латинский / греческий

ИСО/МЭК 8859-8—99* Обработка информации. 8-битные однобайтные кодированные наборы графических символов. Часть 8. Алфавит латинский/иврит

ИСО/МЭК 8859-9—99* Обработка информации. 8-битные однобайтные кодированные наборы графических символов. Часть 9. Латинский алфавит № 5

ИСО/МЭК 10646-1—2000* Обработка информации. Многобайтный кодированный набор символов. Часть 1. Архитектура и основной многоязычный уровень

3 Определения

3.1 Термины, определенные в ИСО/МЭК 8859-1

В настоящем стандарте использованы следующие термины:

- **байт;**
- **символ;**
- **графический символ.**

3.2 Термины, определенные в ИСО/МЭК 10646-1

В настоящем стандарте использован термин **основной многоязычный уровень.**

3.3 Термины, определенные в ГОСТ Р ИСО 10303-1

В настоящем стандарте использованы следующие термины:

- **прикладной протокол**;
- **структура обмена.**

3.4 Термины, определенные в ГОСТ Р ИСО 10303-11

В настоящем стандарте использованы следующие термины:

- **экземпляр сложного объекта;**
- **тип данных**;
- **объект;**
- **частное значение сложного объекта;**
- **экземпляр простого объекта;**
- **лексема.**

3.5 Другие определения

В настоящем стандарте использованы следующие термины с соответствующими определениями:

3.5.1 **основной алфавит** (basic alphabet): Набор символов из ИСО 8859-1 от G (02/00) до G (07/14).

3.5.2 **кодирование открытым текстом** (clear text encoding): Кодирование информации с использованием последовательности кодов для символов в основном алфавите.

3.5.3 **управляющая директива** (control directive): Последовательность символов в основном алфавите.

3.5.4 **ключевое слово** (keyword): Особая последовательность символов, обозначающая объект или определенный тип в структуре обмена.

3.5.5 **секция** (section): Набор данных одной и той же категории информации.

3.5.6 **последовательный файл** (sequential file): Файл, который может быть доступен только последовательным способом.

3.5.7 **разделитель лексем** (token separator): Последовательность из одного или нескольких 8-битных байтов, которая разделяет любые две лексемы.

---

* Оригиналы стандартов ИСО (ИСО/МЭК) — во ВНИИКИ Госстандарта России.

3.6 Сокращения

В настоящем стандарте использованы следующие сокращения:

ОМУ (BMP) — основной многоязычный уровень (Basic multilingual plane);

СНВ (WSN) — синтаксическая нотация Вирта (Wirth Syntax Notation)

4 Основополагающие концепции и допущения структуры обмена

4.1 Введение

Для того чтобы облегчить синтаксический анализ с помощью программных средств, структура обмена описана однозначной, контекстно-свободной грамматикой. Грамматика выражена в синтаксической нотации Вирта, которая описана в приложении В. Представление данных об изделии в структуре обмена определяется с использованием отображения из языка EXPRESS в синтаксис структуры обмена.

4.2 Нотационные и типографские соглашения

Любые *кавычки*, используемые в настоящем стандарте, не являются частью текста структуры обмена, но служат для отделения этого текста. Это положение применимо ко всем местам в тексте, где используются *кавычки*. Таблицы 2—4 определяют исключения из этого правила, так как кавычки, используемые в этих таблицах, составляют часть правил СНВ.

Стандартами серии ИСО 8859 каждому символу присваивается обозначающее имя. Когда это имя используют в настоящем стандарте, для отличия от обычного текста, оно выделено курсивом. Так, *запятую (comma)* используют для ссылки на ", ", *подчеркивание (low line)* — для ссылки на "_", а *прописную букву А (capital letter A)* — для ссылки на «А».

В примерах настоящего стандарта, где требуются пояснения, они введены последовательностью →.

4.3 Соответствие

Установлены два уровня соответствия:

- синтаксическое соответствие структуры обмена: структура обмена соответствует настоящему стандарту, если удовлетворены требования этого стандарта;
- схематическое соответствие структуры обмена: экземпляры, представленные в структуре обмена, соответствуют схемам, указанным в заголовочной секции соответствующей структуры обмена, если удовлетворены все требования и ограничения данных схем и требования отображения каждого экземпляра или группы экземпляров, установленные в разделах 9 и 10.

Примечание — В приложении F определены методы оценки соответствия схемы, когда структура обмена содержит ряд секций данных, связанных с различными EXPRESS-схемами.

Синтаксическое соответствие является необходимым условием для схематического соответствия.

Настоящим стандартом установлены два класса синтаксического соответствия в зависимости от метода, выбранного для кодирования экземпляров сложных объектов (см. 10.2.5). Реализация, претендующая на синтаксическое соответствие настоящему стандарту, должна читать или (и) записывать файлы, которые демонстрируют синтаксическое соответствие по крайней мере в одном из этих двух классов соответствия.

Реализация, претендующая на схематическое соответствие настоящему стандарту, должна читать или (и) записывать файлы, которые демонстрируют схематическое соответствие наряду с синтаксическим соответствием.

5 Формальные определения

5.1 Формальная нотация

В настоящем стандарте для определения синтаксиса структуры обмена использована синтаксическая нотация Вирта (СНВ), описанная в приложении В.

5.2 Определение основного алфавита

Алфавит структуры обмена определен как символы от G(02/00) до G(07/14) по ИСО/МЭК 8859-1. Этот алфавит представлен в структуре обмена набором 8-битных байтов с десятичными значениями от 32 до 126. Таблица 1 делит основной алфавит на подмножества. G(x/y) является нотацией для символа, находящегося в позиции (16 раз по x) + y в кодовой таблице по ИСО/МЭК 8859-1.

Примечание — Таблица D.1 определяет соответствие между 8-битными байтами и их графическим представлением в ИСО/МЭК 8859-1.

Таблица 1 — Определение подмножеств основного алфавита по CHB

```
SPACE      =  " "

DIGIT      =  "0"     |  "1"     |  "2"     |  "3"     |  "4"     |  "5"     |  "6 "|     |"7"
              |  "8"     |  "9"     .

LOWER      =  "a"     |  "b"     |  "c"     |  "d"     |  "e"     |  "f"     |  "g"     |  "h"
              |  "i"     |  "j"     |  "k"     |  "l"     |  "m"     |  "n"     |  "o"     |  "p"
              |  "q"     |  "r"     |  "s"     |  "t"     |  "u" |  |  "v"     |  "w"     |  "x"
              |  "y"     |  "z"     .

UPPER      =  "A"     |  "B"     |  "C"     |  "D"     |  "E"     |  "F"     |  "G"     |  "H"
              |  "I" |  |  "J"     |  "K"     |  "L"     |  "M"     |  "N"     |  "O"     |  "P"
              |  "Q"     |  "R"     |  "S"     |  "T"     |  "U"     |  "V"     |  "W"     |  "X"
              |  "Y"     |  "Z"     |  "_"     .

SPECIAL    =  "!"     |  ""&""   |  "*"     |  "$"     |  "%"     |  "&"     |  "."     |  "#"
              |  "+"     |  ","     |  "-"     |  "("     |  ")"     |  "?"     |  "/"     |  ":"
              |  ";"     |  "<"     |  "="     |  ">"     |  "@"     |  "["     |  "]"     |  "{"
              |  "|"     |  "}"     |  "^"     |  "`"     |  "~"     .

REVERSE_SOLIDUS = "\"

APOSTROPHE = "'"

CHARACTER = SPACE | DIGIT | LOWER | UPPER | SPECIAL

                | REVERSE_SOLIDUS | APOSTROPHE
```

5.3 Структура обмена

Структура обмена должна быть представлена в виде последовательного файла с использованием кодирования открытым текстом. Структура обмена должна содержать заголовочную секцию и секцию данных. Заголовочная секция представляет данные, относящиеся к самой структуре обмена. Структура заголовочной секции определена в разделе 8. Секция данных представляет данные, которые должны быть переданы. Структура секции данных определена в разделе 9. Структура обмена определена с помощью CHB в таблице 3.

Структура обмена является потоком 8-битных байтов, которые кодируются графическими символами основного алфавита. Графические символы группируются в распознаваемые последовательности, называемые лексемами. Лексемы могут быть отделены разделителями. Структуру обмена можно рассматривать как последовательность лексем и их разделителей.

5.4 Определение лексем

Лексемы, используемые в структуре обмена, определены средствами CHB в таблице 2.

Таблица 2 — CHB определений лексем

```
KEYWORD          = USER_DEFINED_KEYWORD | STANDARD_KEYWORD.
USER_DEFINED_KEYWORD  = "!" UPPER { UPPER | DIGIT }.
STANDARD_KEYWORD      = UPPER { UPPER | DIGIT } .
SIGN                  = "+" | "—".
INTEGER               = [SIGN] DIGIT { DIGIT } .
REAL                  = [SIGN] DIGIT { DIGIT } "." { DIGIT }
                        ["E" [SIGN" DIGIT { DIGIT } ].
NON_Q_CHAR            = SPECIAL | DIGIT | SPACE | LOWER | UPPER.
STRING                = "'" { NON_Q_CHAR |
                        APOSTROPHE APOSTROPHE |
                        REVERSE_SOLIDUS REVERSE_SOLIDUS |
                        CONTROL_DIRECTIVE } "'".
ENTITY_INSTANCE_NAME  = "#" DIGIT { DIGIT } .
ENUMERATION           = "." UPPER { UPPER | DIGIT } ".".

HEX                   = "0" |  "1" |  "2" |  "3" |  "4" |  "5" |  "6" |  "7" |
                        "8" |  "9" |  "A" |  "B" |  "C" |  "D" |  "E" |  "F".

BINARY          = """" ("0" |  "1" |  "2" |  "3" )  { HEX } """" .
```

5.5 CHB структуры обмена

Синтаксис структуры обмена установлен в таблице 3. Таблица 3 ссылается на лексемы, определенные в таблице 2. Отношение между синтаксисом и EXPRESS-схемой установлено в разделе 10.

Таблица 3 — CHB структуры обмена

```
EXCHANGE_FILE  = "ISO-10303-21;"
                 HEADER_SECTION DATA_SECTION { DATA_SECTION }
                 "END-ISO-10303-21;".
HEADER_SECTION = "HEADER;"
                 HEADER_ENTITY HEADER_ENTITY HEADER_ENTITY
                 [HEADER_ENTITY_LIST]
                 "ENDSEC;".
HEADER_ENTITY_LIST  = HEADER_ENTITY { HEADER_ENTITY } .
HEADER_ENTITY       = KEYWORD "(" [PARAMETER LIST] ")" ";".
PARAMETER_LIST      = PARAMETER { "," PARAMETER }.
PARAMETER           = TYPED_PARAMETER |
                      UNTYPED_PARAMETER | OMITTED_ PARAMETER.
TYPED_PARAMETER     = KEYWORD "(" PARAMETER ")".
UNTYPED_ PARAMETER  = "$" | INTEGER | REAL | STRING | ENTITY_INSTANCE_NAME
                      | ENUMERATION | BINARY | LIST.
OMITTED_ PARAMETER  = "*".
LIST                = "(" [PARAMETER { "," PARAMETER } ]").
DATA_SECTION        = "DATA" [ "(" PARAMETER _LIST ")" ] ";"
                      ENTITY_INSTANCE_LIST "ENDSEC;".
ENTITY_INSTANCE_LIST      = { ENTITY_INSTANCE }.
ENTITY_INSTANCE = SIMPLE_ENTITY_INSTANCE | COMPLEX_ENTITY_INSTANCE
SIMPLE_ENTITY_INSTANCE = ENTITY_INSTANCE_NAME "=" SIMPLE_RECORD ";".
COMPLEX_ENTITY_INSTANCE = ENTITY_INSTANCE_NAME "=" SUBSUPER_RECORD ";

SIMPLE_RECORD       = KEYWORD "(" [ PARAMETER_LIST ] ")".
SUBSUPER_RECORD          = ("SIMPLE_RECORD_LIST ")".
SIMPLE_RECORD _LIST = SIMPLE_RECORD { SIMPLE_RECORD }.
```

5.6 Разделители лексем

Разделитель лексемы является элементом, отделяющим две лексемы. Разделителями являются *пробел,* явные директивы управления печатью и комментарии. Разделитель может появиться между терминальными или нетерминальными порождениями таблицы 3. В том месте, где может появиться один разделитель, может появиться любое число разделителей. Разделитель не должен появляться внутри лексем, за исключением того, что явные директивы управления печатью могут появляться внутри чисел в двоичном представлении и внутри строк. Директивы управления печатью определены в разделе 11.

Примечание — *Пробелом* является символ пробела, описанный в основном алфавите в 5.2. Разделители строк, такие как *перевод строки* или *возврат каретки,* могут быть использованы в структуре обмена в соответствии с приложением А, но они не входят в основной алфавит, и согласно этому приложению А должны быть проигнорированы при обработке структуры обмена. Поэтому разделители строк могут присутствовать в структуре обмена, включая разделители лексем.

Комментарий должен быть закодирован как *косая черта, звездочка* "/*", за которым следует любое число символов из основного алфавита, и завершаться "*/". Любое появление комбинации "*/" после первого появления не имеет значения, т. е. комментарии не могут быть вложены. Все графические символы, появляющиеся внутри комментария, не имеют значения для структуры обмена и предназначены только для чтения людьми.

6 Лексемы

В структуре обмена лексема является специальной лексемой, ключевым словом или кодированием простого типа данных.

6.1 Специальные лексемы

Для открытия структуры обмена должна быть использована специальная лексема "ISO-10303-21;", а для закрытия структуры обмена — "END-ISO-10303-21;".

Для того чтобы открыть или закрыть заголовочную секцию структуры обмена, должны быть использованы специальные лексемы "HEADER" или "ENDSEC" соответственно.

Для того чтобы открыть или закрыть секцию данных структуры обмена, должны быть использованы специальные лексемы "DATA" или "ENDSEC" соответственно.

Специальную лексему *знак доллара* "$" используют для представления объекта, чье значение не представлено в структуре обмена.

Специальную лексему *звездочка* "*" используют для представления предмета, значение которого не представлено в структуре обмена, но может быть выведено из других величин в соответствии с правилами, приведенными в EXPRESS-схеме (см. 10.2.6).

Специальные лексемы *точка с запятой* ";", *скобки* "(", ")", *запятая* "," и *косая черта* "/" используют как знаки препинания в структуре обмена.

6.2 Ключевые слова

Ключевые слова являются последовательностями графических символов, указывающими объект или определенный тип в структуре обмена. Ключевые слова должны состоять из *прописных букв, цифр, подчеркивания* и, возможно, *восклицательного знака* "!". *Восклицательный знак* должен появляться не более одного раза и только как первый символ в ключевом слове.

Ключевые слова могут быть таковыми, определенными в схеме или заданными пользователем. Ключевые слова, которые не начинаются с *восклицательного знака*, являются определенными в схеме. Ключевые слова, которые начинаются с *восклицательного знака*, являются определенными пользователем. Определенное пользователем ключевое слово является идентификатором для поименованного типа (типа данных объекта или определенного типа) в EXPRESS-схеме, управляющей структурой обмена. Смысл ключевого слова, определенного пользователем, является предметом соглашения между партнерами, использующими структуру обмена.

6.3 Кодирование простых типов данных

В структурах обмена используют кодирование шести простых типов данных: целое (integer), вещественно (real), строка (string), имя экземпляра объекта (entity instance name), перечисление (enumeration) и двоичное (binary).

6.3.1 Целое (Integer)

Целое должно быть закодировано как последовательность из одной или нескольких цифр, согласно таблице 2, которой может (но необязательно) предшествовать *знак плюс* "+" или *минус*

"—". Целое должно быть выражено в десятичном основании. Если целое не имеет знака, его считают положительным.

Пример

Верное представление целого в файле	Значение
16	Положительное 16
+12	Положительное 12
—349	Отрицательное 349
012	Положительное 12
00	Нуль
Неверное представление целого в файле	Ошибка
26 54	Содержит *пробелы*
32.0	Содержит *точку*
+12	Содержит *пробел* между *знаком плюс* и цифрами

6.3.2 Вещественное (Real)

Вещественное должно быть закодировано, как указано в таблице 2. Код должен состоять из десятичной мантиссы, за которой (необязательно) следует десятичный показатель степени. Десятичная мантисса состоит в порядке следования из необязательного *знака плюс* "+" или *минус* "—", последовательности из одной или более цифр, *точки* ".", последовательности из нуля или нескольких цифр. Десятичный показатель степени состоит из *прописной буквы E*, за которой следует необязательный *знак плюс* "+" или *минус* "—" с одной или несколькими цифрами.

Примечание — В настоящем стандарте не сделано никаких попыток выразить концепцию точности. Когда необходимо указать значение точности, посылающая и принимающая стороны должны достигнуть соглашения по этому вопросу. Там, где точность требуется как элемент описания типа данных объекта, ее значение должно быть включено в определение типа данных объекта в EXPRESS-схеме.

Пример

Верное представление вещественного	Значение
+0.0E0	0.0
—0.0E-0	0.0
1.5	1.5
—32.178E+02	—3217.8
0.25E8	25 миллионов
0.E25	0.
2.	2.
5.0	5.0
Неверное представление вещественного	Ошибка
1.2E3.	В обозначении показателя степени не допускается десятичная точка
1E05	В обозначении мантиссы требуется десятичная точка
1,000.00	*Запятая* не допускается
3.E	В обозначении показателя степени должна быть хотя бы одна цифра
.5	Десятичной точке должна предшествовать хотя бы одна цифра
1	В обозначении мантиссы требуется десятичная точка

6.3.3 Строка (String)

Строка должна быть закодирована как *апостроф* "'", за которым следует нуль или несколько 8-битных байтов, и заканчиваться "'". Нулевая строка (строка нулевой длины) должна быть закодирована последовательностью из двух *апострофов* "''". Внутри строки единичный *апостроф* должен

быть закодирован как два последовательных *апострофа*. Внутри строки единичная *косая обратная черта* "\" должна быть закодирована как две *косые обратные черты* "\\". 8-битные байты, разрешенные внутри строки, являются десятичными эквивалентами чисел от 32 до 126 (включительно) по ИСО/МЭК 8859-1, которые определяют графические символы основного алфавита.

Примечание — Таблица D.1 определяет соответствие между 8-битными байтами и их графическим представлением по ИСО/МЭК 8859-1. *Кавычки* при появлении в строке не должны дублироваться. В таблице 1 появляются *двойные кавычки* потому, что в СНВ они являются метасимволом (см. приложение В).

Дополнительные символы должны быть закодированы с использованием шестнадцатеричных цифр (см. HEX в таблице 2), как определено в 6.3.3.1 и 6.3.3.2. СНВ управляющих директив для закодированных строк приведена в таблице 4.

Таблица 4 — Управляющие директивы для строк

```
CONTROL_DIRECTIVE = PAGE | ALPHABET | EXTENDED2
                          | EXTENDED4 | ARBITRARY.
PAGE = REVERSE_SOLIDUS "S" REVERSE_SOLIDUS CHARACTER.
ALPHABET = REVERSE_SOLIDUS "P" UPPER REVERSE_SOLIDUS.
EXTENDED2 = REVERSE_SOLIDUS "X2" REVERSE_SOLIDUS
            HEX_TWO { HEX_TWO } END_EXTENDED.
EXTENDED4 = REVERSE_SOLIDUS "X4" REVERSE_SOLIDUS
            HEX_FOUR { HEX_FOUR } END_EXTENDED.
END_EXTENDED = REVERSE_SOLIDUS "X0" REVERSE_SOLIDUS.
ARBITRARY = REVERSE_SOLIDUS "X" REVERSE_SOLIDUS HEX_ONE.
HEX_ONE  = HEX HEX.
HEX_TWO = HEX_ONE HEX_ONE.
HEX_FOUR = HEX_TWO HEX_TWO.
```

6.3.3.1 Кодирование полного алфавита по стандартам серии ИСО/МЭК 8859 внутри строки

В стандартах серии ИСО/МЭК 8859 G (*x/y*) является обозначением символа в "колонке" *x* "столбце" *y*, т.е. значением кода (16**x*) + *y* в таблице кодов. Каждая часть ИСО/МЭК 8859 (ИСО/МЭК 8859-1 — ИСО/МЭК 8859-9) включает в себя основной алфавит (см. 5.2) как позиции от G (02/00) до G (07/14). Различные части стандартов серии ИСО/МЭК 8859 отличаются символами расширенного набора символов — позициями от G (10/00) до G (15/14). Для того чтобы включить в строку символы из расширенного набора, необходимо использовать управляющие директивы.

Управляющую директиву PAGE — *обратная косая черта, прописная буква S, обратная косая черта* ("\S\") CHARACTER (см. таблицу 4) — используют в строке для того, чтобы позволить символу основного алфавита представить символ в соответствующей позиции расширенного алфавита. Управляющую директиву PAGE следует интерпретировать в строке как одиночный символ G [(*x* + 8) /*y*], где G(*x* / *y*) — символ основного алфавита, следующего за "\S\". Таким образом, если символ основного алфавита имеет значение кода *v*, его следует интерпретировать как символ со значением кода *v* + 128.

Для того чтобы указать, что только в данной строке последующие управляющие директивы *обратная косая черта, прописная буква S, обратная косая черта* будут интерпретироваться как ссылки на расширенный алфавит, определенный в той части стандартов серии ИСО 8859, которая определяется значением UPPER, должна быть использована управляющая директива *обратная косая черта, прописная буква P,* UPPER, *обратная косая черта. Прописная буква* (обозначенная как UPPER) должна быть одной из следующих: "A", "B", "C", "D", "E", "F", "G", "H", "I". В данном контексте *буква A* определяет ИСО/МЭК 8859-1; *буква B* — ИСО/МЭК 8859-2 и т.д. Если данная управляющая директива не появляется в строке, подразумевается значение "A", т.е. должен быть тот расширенный алфавит, который определен в ИСО/МЭК 8859-1.

Пример

Хранящаяся строка	Содержание	Комментарии
'CAT'	CAT	
'Don"t'	Don't	
""	'	
"		Строка нулевой длины
'\S\Drger'	**Ärger**	
'h\S\ttel'	**hôtel**	
'\PE\\S*\S\U\S\b'	**Нет**	Кириллица, **'Нет'**

6.3.3.2 Кодирование внутри строки набора символов из стандартов серии ИСО/МЭК 10646

В ГОСТ Р ИСО 10303-11 (см. 8.1.6) определено, что в строке может появиться любой символ из стандартов серии ИСО/МЭК 10646. Настоящий стандарт устанавливает три управляющие директивы, которые позволяют кодировать символы из ИСО/МЭК 10646.

Стандарты серии ИСО/МЭК 10646 определяют каноническую форму, которая использует четыре восьмибитные группы для представления любого символа из полного множества кодирования. Эти символы определяют соответственно группу, уровень, ряд и ячейку. Дополнительно стандарты серии ИСО/МЭК 10646 определяют основной многоязычный уровень (ОМУ), представляющий уровень 00 группы 00 полного множества кодирования. Символы в ОМУ представлены двумя байтами, определяющими ряд и ячейку.

Примечание — ОМУ включает в себя символы, в основном используемые в алфавитных, послоговых и идеографических записях вместе с различными знаками и цифрами.

Для того чтобы показать, что очередная последовательность из кратных четырем шестнадцатеричным символом будет интерпретироваться как закодированная двухбайтным представлением символов из ОМУ по ИСО/МЭК 10646-1, должна быть использована управляющая директива *обратная косая черта, прописная буква X, цифра два, обратная косая черта* "\X2\". Кодирование в строке в структуры обмена должно быть следующим:

- каждый символ представления из стандартов серии ИСО/МЭК 10646, подлежащий кодированию, должен быть преобразован в два 8-битных байта, как определено в ИСО/МЭК 10646-1;
- каждый из двух полученных в результате 8-битных байтов должен быть закодирован как два шестнадцатеричных символа в основном алфавите, соответствующих графическому представлению шестнадцатеричных цифр.

Пример 1 — Латинскую *прописную букву B* преобразуют с помощью таблицы 1 из ИСО/МЭК 10646-1 в шестнадцатеричное значение '0042'. Шестнадцатеричными цифрами, соответствующими этому значению, являются 0, 0, 4 и 2. Кодирование в структуре обмена с использованием основного алфавита представляет собой четыре последовательных символа: 0042.

Для того чтобы показать, что очередная последовательность из кратных восьми шестнадцатеричных значений должна интерпретироваться как закодированная четырехбайтным представлением символов из полного множества кодирования по стандартам серии ИСО/МЭК 10646, должна быть использована управляющая директива *обратная косая черта, прописная буква X, цифра четыре, обратная косая черта* "\X4\". Кодирование в строке в структуре обмена должно быть следующим:

- каждый символ в представлении из стандартов серии ИСО/МЭК 10646, который должен быть закодирован, следует преобразовать в четыре 8-битных байта, как определено в ИСО/МЭК 10646-1;
- каждый из четырех получаемых в результате 8-битных байтов должен быть закодирован как два шестнадцатеричных значения из основного алфавита, соответствующих графическому представлению шестнадцатеричных цифр.

Пример 2 — Латинскую *прописную букву B* преобразуют с помощью таблицы 1 из ИСО/МЭК 10646-1 в шестнадцатеричное значение '00000042'. Шестнадцатеричными цифрами являются 0, 0, 0, 0, 0, 0, 4 и 2. Кодирование в структуре обмена с использованием основного алфавита представляет собой восемь последовательных символов 00000042.

Для того чтобы показать окончание кодирования символов по стандартам серии ИСО/МЭК 10646 в строке и возврат к прямому кодированию с использованием основного алфавита, должна быть использована управляющая директива *обратная косая черта, прописная буква X, цифра ноль, обратная косая черта* "\X0\".

6.3.3.3 Кодирование в строке единичного 8-битного байта

В строке может быть закодирован 8-битный байт со значением от 0 до 255. Для того чтобы указать, что следующие два шестнадцатеричных символа должны быть интерпретированы как 8-битный байт, должна быть использована управляющая директива *обратная косая черта, прописная буква X, обратная косая черта* "\X\", интерпретируемая как 8-битный байт ячейки символа уровня 0 ОМУ по стандартам серии ИСО/МЭК 10646.

Примечание — Символы, определенные в ИСО/МЭК 10646-1 и ИСО/МЭК 8859-1, должны быть указаны в соответствующем диапазоне.

Пример

Хранящаяся строка	Содержание	Комментарии
'see \X\A7 4.1'	см. 4.1	—
'line one\X0Aline two'	строка один строка два	содержит новые строки

6.3.3.4 Максимальная длина строки

Максимальная длина строки, сохраняемая в структуре обмена, ограничена 32769 8-битными байтами, включая начальный и конечный *апострофы*. Если в хранящуюся строку включены *кавычки, обратная косая черта, апострофы*, директивы управления печатью (см. раздел 11) или символы, закодированные в соответствии с 6.3.3.1, 6.3.3.2 или 6.3.3.3, максимальная длина действительного содержания строки будет меньше, чем 32767 графических символов. Действительным содержанием является последовательность графических символов, полученная после того, как будут выполнены соглашения по кодированию.

6.3.4 Имена экземпляров объектов

Имя экземпляра объекта должно быть закодировано как *знак номера* "#", за которым следует целое без знака. Целое должно представлять любую комбинацию из одной или нескольких десятичных цифр. По меньшей мере одна цифра не должна быть "0". Предшествующие нули в имени экземпляра объекта не имеют значения.

СНВ для имен экземпляров объектов приведена в таблице 2 в правиле подстановки ENTITY_INSTANCE_NAME.

Пример

Правильное выражение имени	Значение
#12	Именует экземпляр объекта или ссылается на экземпляр объекта с идентификатором 12
#023	Именует экземпляр объекта или ссылается на экземпляр объекта с идентификатором 23
Неправильное выражение имени	**Ошибка**
#+023	Содержит знак '+'
#00.1	Содержит десятичную точку
74	Не начинается со *знака номера*
#439A6	Содержит символы алфавита

Если имена экземпляров объектов появляются внутри списка атрибутов экземпляра объекта, то их используют как ссылки на другие экземпляры объектов. Разрешены ссылки как вперед, так и назад.

6.3.5 Перечисляемые значения

Перечисляемое значение должно быть закодировано как последовательность *прописных букв* или *цифр*, начинающаяся с *прописной буквы*, ограниченная *точками*. Смысл заданного перечисляемого значения задается EXPRESS-схемой и соответствующими определениями из объявлений перечисляемого типа.

Пример

Правильное перечисляемое выражение	Значение
.STELL.	Показывает значение STELL
Неправильное перечисляемое выражение	Ошибка
.RED	Пропущена *точка* в конце
.123.	Начинается не с алфавитного символа

6.3.6 Двоичное число (Binary)

Двоичным числом является последовательность битов (0 или 1). Двоичное число должно быть закодировано так, как определено следующей процедурой:

- подсчитывают количество битов в последовательности. Называют результат *p*;
- определяют число *n*, 0 <= *n* <= 3, так что $k = p + n$ является кратным четырем;
- дополняют слева двоичное число *n* нулевыми битами. Разделяют последовательность на группы по четыре бита;
- перед последовательностью вводят четырехбитное представление *n*;
- если десятичный эквивалент 4-битной группы 9 или меньше, чтобы получить 8-битный байт, добавляют к десятичному значению 48; если десятичный эквивалент 4-битной группы больше 9, чтобы получить 8-битный байт, добавляют к десятичному значению 55.

Примечание — Это преобразование двоичного числа в шестнадцатеричное;

- закодированное двоичное число состоит из $k/4+1$ шестнадцатеричных цифр. Первая цифра является значением *n*. За ней следуют шестнадцатеричные цифры, представляющие двоичное число;
- двоичное число отделяют кавычками “”".

Пример

Двоичное значение	Представление
'null' или 'empty'	“0"
0	“30"
1	“31"
111011	“23B”
100100101010	“092A”

7 Структурированные типы данных

Единственным структурированным типом данных, который появляется в структуре обмена, является LIST (список), как это определено в таблице 3. Список является последовательностью (возможно пустой) из PARAMETER (параметров), каждый из которых может быть:

- кодированием простого типа, как описано в 6.3, или
- специальной лексемой *знак доллара* “$”, или
- TYPED_PARAMETER, представляющим экземпляр выбранного типа (см. 10.1.8), или
- LIST, представляющим экземпляр (вложенного) структурированного типа.

Данный список может содержать более одной из вышеперечисленных форм. В структуре обмена список начинается с *левой скобки* “(” и кончается *правой скобкой* “)”. Экземпляры разделяют *запятыми*. Список может быть вложенным на любую глубину.

Пример

Структурированный тип данных	Представление
Список целых	(0, 1, 2, 3, 7, 2, 4)
Список строк	('CAT', 'HELLO')
Список списков вещественных значений	((0.0,1.0,2.0), (3.0,4.0,5.0))
Список списков вещественных значений	((0.0,1.0,2.0), ())

В последнем списке списков вещественных значений второй вложенный список пустой.

8 Заголовочная секция

Заголовочная секция содержит информацию, которая относится ко всей структуре обмена. Эта секция должна быть представлена в каждой структуре обмена. Секция должна начинаться со специальной лексемы “HEADER;” и заканчиваться специальной лексемой “ENDSEC;”.

Примечание — В приложении Н представлен пример заголовочной секции внутри структуры обмена.

8.1 Объекты заголовочной секции

Определены три объекта заголовочной секции, и в каждой структуре обмена требуется наличие одного экземпляра каждого из объектов. Объектами заголовочной секции являются **file_description**, **file_name** и **flie_schema**, и они должны располагаться в этой последовательности. Экземпляры объектов **file_population**, **section_language** и **section_context** могут располагаться после объекта **file_schema.** Если имеются экземпляры определенных пользователем объектов заголовочной секции, то они должны появляться после обязательных экземпляров объектов заголовочной секции в произвольном порядке. Синтаксис экземпляров объектов заголовочной секции приведен в СНВ в таблице 3. Каждое имя объекта должно быть преобразовано в KEYWORD (ключевое слово) правила HEADER_ENTITY. Раздел 10 определяет преобразование простых и составных (агрегатных) типов данных в PARAMETER_LIST (список параметров) для значений атрибутов данных экземпляров объектов.

8.2 Схема заголовочной секции

Данный подраздел определяет объекты заголовочной секции и типы, которые появляются в заголовочной секции структуры обмена. Объекты заголовочной секции определены с помощью языка EXPRESS.

EXPRESS-спецификация:

```
*)
SCHEMA header_section_schema;
TYPE exchange_structure_identifier = STRING;
END_TYPE
*)
```

Данная схема определяет объекты заголовочной секции, которые специфичны для процесса передачи данных об изделии с использованием структуры обмена.

Примечание — Тип **exchange_structure_identifier** соответствует типу идентификатора, установленному в ГОСТ Р ИСО 10303-41, но должен быть задан отдельно, чтобы обеспечить независимость требований настоящего стандарта от моделей данных, описанных в группе интегрированных ресурсов стандартов серии ГОСТ Р ИСО 10303.

8.2.1 Объект file_description

Объект **file_description** определяет версию настоящего стандарта, использованную для создания структуры обмена, а также ее содержание.

EXPRESS-спецификация

```
*)
ENTITY file_description;
    description          : LIST [1 : ?] OF STRING (256);
    implementation_level : STRING (256);
END_ENTITY;
(*
```

Описание атрибутов:

description — неформальное описание содержимого структуры обмена;

implementation_level — обозначение требований, которым соответствует кодирование в данной структуре обмена, и любых соответствующих вариантов, применяемых при кодировании. Значение этого атрибута должно указывать соответствие данной версии настоящего стандарта наличием значения “3; 1", или ”3; 2". Для структур обмена, принадлежащих к классу соответствия 1, значение должно быть задано в виде “3; 1". Для структур обмена, принадлежащих к классу соответствия 2, значение должно быть ”3; 2".

Если встречаются ниже перечисленные ограничения, тогда для указания соответствия настоящему стандарту может быть использовано значение “2; 1" или ”2; 2":

- конкретная структура обмена должна содержать единственную секцию данных, а ключевое слово "DATA" не должно располагаться за PARAMETER_LIST;
- заголовочная секция структуры обмена не должна содержать объектов FILE_POPULATION;
- заголовочная секция структуры обмена не должна содержать объектов SECTION_LANGUAGE;
- заголовочная секция структуры обмена не должна содержать объектов SECTION_CONTEXT;
- перечисляемые значения (ENUMERATION) на языке EXPRESS не должны кодироваться с использованием сокращенных наименований.

Значения "2; 1" и "2; 2" должны обозначать структуры обмена, относящиеся к классу соответствия 1 и 2.

Примечания

1 Классы соответствия 1 и 2 определены в 10.2.5.

2 Общая форма для значения имеет вид "v; cc", где *v* — номер версии настоящего стандарта, как определено в приложении С, а *cc* — кодирование класса соответствия. Будущие версии настоящего стандарта могут определить дополнительные значения для *v* и *cc*.

3 Значения "2; 1" и "2; 2" обеспечивают совместимость с реализациями, основанными на предыдущей версии настоящего стандарта.

8.2.2 Объект file-name

Объект **file-name** представляет доступную для прочтения человеком информацию о структуре обмена. Содержание атрибутов данного объекта, за исключением атрибута **time_stamp**, не определяется настоящим стандартом.

EXPRESS-спецификация

```
*)
ENTITY file_name;
    name                : STRING (256);
    time_stamp          : time_stamp_text;
    author              : LIST [1:?] OF STRING (256);
    organization        : LIST [1:?] OF STRING (256);
    preprocessor_version : STRING (256);
    originating_system  : STRING (256);
    authorization       : STRING (256);
END_ENTITY;

TYPE time_stamp_text = STRING (256);
END_TYPE;
*)
```

Описания атрибутов:

name — строка графических символов, используемая для наименования данного конкретного экземпляра структуры обмена.

Примечание — Имя предназначается для связи между отправителем и получателем, подобно тому, как оно используется между людьми;

time_stamp — дата и время, показывающие, когда была создана структура обмена. Содержание строки должно соответствовать расширенному формату полной календарной даты, как определено в 4.2.1.1 ИСО 8601, объединенному с расширенным форматом времени дня, установленным в 4.3.1.1 или в 4.3.3 ИСО 8601. Дата и время должны быть разделены *прописной буквой Т*, как определено в 4.4.1 ИСО 8601. Другие форматы из 4.3.1.1 и 4.3.3 позволяют дополнительно включать определитель часового пояса;

author — имя и почтовый адрес лица, ответственного за создание структуры обмена;

organization — группа или организация, с которой связан автор;

preprocessor_version — система, используемая для создания структуры обмена, включая имя и версию системного изделия;

originating_system — система, от которой выданы данные в эту структуру обмена;

authorization — имя и почтовый адрес лица, уполномоченного посылать структуру обмена.

Пример

Элемент штампа времени	Полный расширенный формат
Календарная дата — 12 апреля 1993 г.	1993—04—12
Время дня — 15 ч 27 мин 46 с	15:27:46
Часовой пояс — 5 ч на запад от Гринвича	Поле часового пояса является необязательным —05:00
Вышеприведенные дата, время и часовой пояс, закодированные в поле time_stamp	1991—04—12 T 15:27:46—05:00

8.2.3 Объект file_schema

Объект **file_schema** указывает EXPRESS-схемы, в которых определены экземпляры объектов в секции данных. Атрибут **schema_identifiers** должен состоять из списка строк, каждая из которых должна содержать имя схемы, за которым следует необязательный идентификатор объекта, присвоенный этой схеме.

Если имя схемы содержит *строчные буквы*, то они должны быть преобразованы в соответствующие *прописные буквы*. В строках **schema_name** должны быть только *прописные буквы*.

Если известен идентификатор объекта, он должен иметь форму, установленную в ГОСТ Р ИСО/МЭК 8824-1. Использование идентификаторов объектов в стандартах серии ГОСТ Р ИСО 10303 описано в 4.3 ГОСТ Р ИСО 10303-1. Рекомендуется, по возможности, применять идентификатор объекта, так как это обеспечивает однозначное обозначение схемы.

Примечание — Общим форматом представления идентификатора описываемого предмета (физического объекта) являются отдельные целые числа. Последовательность этих чисел представляют в соответствующих фигурных скобках "{","}".

EXPRESS-спецификация

```
*)
ENTITY file_shema;
    schema_identifiers : LIST [1:?] OF UNIQUE schema_name;
END_ENTITY;

TYPE schema_name = STRING (1024);
END_TYPE;
(*
```

Описание атрибутов:

schema_identifiers — схемы, которые определяют экземпляры объектов в секции данных.

Пример — Нижеописанный экземпляр EXPRESS-схемы назван 'CONFIG_CONTROL_-DESIGN':

```
FILE_SCHEMA (('CONFIG_CONTROL_DESIGN'));
```

Следующий экземпляр использует идентификатор описываемого предмета (физического объекта)' AUTOMOTIVE_DESIGN' для указания конкретной версии EXPRESS-схемы:

```
FILE_SCHEMA (('AUTOMOTIVE_DESIGN {1 0 10303 214 1 1 1 }'));
```

8.2.4 Объект file_population

Объект **file_population** задает набор (коллекцию) экземпляров объектов в структуре обмена с целью определения их схематического соответствия конкретной EXXPRESS-схеме. Данная коллекция должна быть задана посредством алгоритма, определенного атрибутом **determination_method** для набора секций данных, указанных атрибутом **governed_sections**. Если в этом атрибуте не указано конкретное значение, данный алгоритм должен быть использован для всех секций данных в структуре обмена.

Структура обмена может содержать ноль, один или несколько экземпляров **file_population**. Имя секции данных может быть указано в атрибуте **governed_sections** нулевого, одного или нескольких экземпляров **file_population**.

Примечания

1 В F.2 заданы три возможных метода определения данного объекта.

2 Если в атрибуте **governed_sections** не задано конкретное значение, его кодируют *знаком доллара* ("$") в соответствии с 10.2.2, а не в виде пустого списка.

```
EXPRESS-спецификация
*)
ENTITY file_population;
    governing_schema      : schema_name;
    determination_method : exchange_strucrure_identifier;
    governed_sections     : OPTIONAL SET [1 : ?] OF section_name;
END_ENTITY;
TYPE section_name = exchange_strucrure_identifier;
END_TYPE;
(*
```

Описание атрибутов:

governing_schema: — имя EXPRESS-схемы, используемой для совокупности экземпляров объектов, заданных в заголовочной секции объекта **file_population**. Данное имя может быть размещено в заголовочной секции объекта file_schema;

determination_method: — строка графических символов, применяемая для определения алгоритма, используемого при выборе экземпляров объектов для обмена;

governed_sections: — имена секций данных, использованные в качестве исходных данных для выбранного метода определения данного объекта.

8.2.5 Объект section_language

Объект **section_language** определяет по умолчанию язык описания строковых значений в секции данных. Атрибут **default_language** должен содержать наименование конкретного языка. Наименование языка должно быть закодировано в соответствии с библиографическим кодом Alpha-3, определенным в ИСО 639-2.

Атрибут **section** должен содержать наименование секции данных в структуре обмена, для которой по умолчанию задан соответствующий язык. Если в структуре обмена содержится единственная не поименованная секция данных, тогда значение атрибута **section** не определяют (см. 10.2.2). Заголовочная секция структуры обмена должна содержать по крайней мере один экземпляр объекта **section_language** с неопределенным значением атрибута **section.** При необходимости заданный в этом экземпляре по умолчанию язык должен быть использован для всех секций данных в структуре обмена, в которых не определены экземпляры других объектов **section_language.**

Пример — Примерами значений атрибута **default_language** являются: 'end' — для английского языка, 'fre' — для французского, 'rus' — для русского или 'ger' — для немецкого.

```
EXPRESS-спецификация
*)
ENTITY section_language;
    section          : OPTIONAL section_name;
    default_language : language_name;
UNIQUE
    UR1              : section;
END_ENTITY;
TYPE language_name = exchange_strucrure_identifier;
END_TYPE;
(*
```

Описание атрибутов:

section: — имя секции данных, для которой применяют язык, заданный по умолчанию атрибутом **default_language**;

default_language: — наименование языка, используемого для строковых значений.

8.2.6 Объект section_context

Объект **section_context** задает информацию, описывающую контексты использования экземпляров, закодированных в структуре обмена.

Атрибут **section** должен содержать имя секции данных в структуре обмена, для которой используют конкретные идентификаторы контекстов. Если в структуре обмена содержится единственная не поименованная секция данных, тогда значение атрибута **section** не определяют (см. 10.2.2). Заголовочная секция структуры обмена должна содержать по крайней мере один экземпляр объекта **section_context** с неопределенным значением атрибута **section**. При необходимости заданные в этом

экземпляре идентификаторы контекстов должны быть использованы для всех секций данных в структуре обмена, в которых не определены экземпляры других объектов **section_context**.

EXPRESS-спецификация

```
*)
ENTITY section_context;
    section             : OPTIONAL section_name;
    context_identifiers : LIST [1:?] OF context_name;
UNIQUE
    UR1 : section;
END_ENTITY;

TYPE context_name = STRING;
END_TYPE;
(*
```

Описание атрибутов:

section: — имя секции данных, для которой применяют контексты, заданные атрибутом **context_identifiers**;

context_identifiers: — идентификаторы, содержащие информацию о контекстах экземпляров, закодированных в структуре обмена.

Примечание — В прикладном протоколе могут быть заданы символические идентификаторы для каждого класса соответствия этого протокола. Идентификаторы контекстов для секции могут задавать конкретный список классов соответствия прикладного протокола, обеспечиваемый данными из этой секции.

Пример 1 — Язык и идентификатор контекста заданы для структуры обмена в единственной не поименованной секции данных:

```
HEADER;
    .
    .
FILE_SCHEMA (('GEMETRY '));
SECTION_LANGUAGE ($, 'end'); ——————————> A
SECTION_CONTEXT ($,'tag_a'); ——————————> B
ENDSEC;
DATA;
    .
ENDSEC;
```

A: для секции данных выбран английский язык, закодированный как 'eng'.

B: для секции данных выбран признак (тег) контекста 'tag_a'.

Пример 2 — Язык и идентификатор контекста заданы для структуры обмена в нескольких секциях данных:

```
HEADER;
    .
    .
FILE_SCHEMA (('GEMETRY'));
SECTION_LANGUAGE ('DS1', 'ger'); ——————————> A
SECTION_LANGUAGE ('DS2', 'epo'); ——————————> B
SECTION_LANGUAGE ($, 'haw'); ——————————> C
SECTION_CONTEXT ('DS1', ('tag_a', 'tag b')); ————> D
SECTION_CONTEXT ('DS2', ('tag_c')); ——————————> E
SECTION_CONTEXT ($, ('tag_d')); ——————————> F
ENDSEC;
DATA ('DS1', (GEOMETRY'));
    .
ENDSEC;
DATA ('DS2', (GEOMETRY'));
    .
ENDSEC;
DATA ('DS3', (GEOMETRY'));
    .
ENDSEC;
```

```
DATA ('DS4', ('GEOMETRY'));
    .
ENDSEC;

A: для секции данных с именем 'DS1' выбран немецкий язык, закодированный как 'ger'.
B: для секции данных с именем 'DS2' выбран язык Эсперанто, закодированный как 'epo'.
C: для секций данных с именами 'DS3' и 'DS4' выбран гавайский язык, закодированный как 'haw'.
D: для секции данных с именем 'DS1' выбраны идентификаторы контекстов 'tag_a' и 'tag_b'.
E: для секции данных с именем 'DS2' выбран идентификатор контекста 'tag_c'.
F: для секций данных с именами 'DS3' 'DS4' выбран идентификатор контекста 'tag_d'.

*)
END_SCHEMA;
(*
```

8.3 Объекты заголовочной секции, определенные пользователем

В заголовочной секции могут быть помещены экземпляры объектов заголовочной секции, определенные пользователем, с конкретными ограничениями, перечисленными ниже.

a) Экземпляры объектов заголовочной секции, определенные пользователем, должны соблюдать тот же синтаксис, что и все экземпляры объектов заголовочной секции, с дополнительным требованием, что первый символ ключевого слова должен быть *восклицательным знаком* "!".

b) Атрибуты объектов заголовочной секции, определенные пользователем, должны иметь типы данных из языка EXPRESS и отображаться в заголовочной секции, как определено в разделе 10.

```
Пример
HEADER;
    .
    .
FILE_SCHEMA (('GEOMETRY'));
!A_SPECIAL_ENTITY ('ABC', 123); ----> ОПРЕДЕЛЕННЫЙ ПОЛЬЗОВАТЕЛЕМ ОБЪЕКТ
    .
    .
ENDSEC;
```

9 Секции данных

Секция данных содержит экземпляры, передаваемые через структуру обмена. В каждой структуре обмена должна быть представлена по крайней мере одна секция данных. Каждая секция данных содержит экземпляры объектов, соответствующие одной EXPRESS-схеме, определенной в заголовочной секции.

Синтаксис секции данных задан в таблице 3. Каждая секция данных должна начинаться с ключевого слова «DATA». Если в структуру обмена включено несколько секций данных, в каждой из них за ключевым словом «DATA» должен быть представлен PARAMETER_LIST, содержащий параметр строки (STRING) или списка (LIST).

Первым параметром должна быть строка (STRING), содержащая индивидуальное имя секции. Вторым параметром должен быть список (LIST), содержащий одну строку (STRING). Данная строка должна содержать имя схемы, управляющей заданной секцией данных. Имя данной схемы должно входить в заголовочную секцию объекта **file_schema**.

Если структура обмена содержит только одну секцию данных, тогда список параметров (PARAMETER_LIST) может быть опущен. В этом случае в заголовочной секции объекта **file_schema** должна быть определена только одна схема, управляющая заданной секцией данных.

Каждая секция данных должна заканчиваться специальной лексемой "ENDSEC".

Примечание — В приложении H представлен полный пример секции данных в структуре обмена.

9.1 Экземпляры объектов секции данных

Каждый экземпляр объекта должен отображаться на конструкцию ENTITY_INSTANCE (см. таблицу 3) в секции данных, как определено в 10.2. Каждый экземпляр объекта должен быть представлен в секции данных не более одного раза; различающиеся экземпляры объектов должны иметь различные имена. Экземпляры объектов в структуре обмена не нуждаются в упорядочении. Ссылка на имя экземпляра объекта может появляться до того, как оно будет определено ENTITY_INSTANCE в структуре обмена.

9.2 Экземпляры объектов секции данных, определяемые пользователем

Экземпляр объекта, определяемый пользователем, не входит в EXPRESS-схему, определенную в заголовочной секции. Экземпляры объекта, определяемые пользователем, должны соответствовать синтаксису всех экземпляров объектов этой секции данных, исключая выбор USER_DEFINED_KEYWORD, который должен быть использован в SIMPLE_RECORD, входящем в данное описание. Смысловое содержание экземпляров объектов, определяемых пользователем, их количество, типы данных и содержание соответствующих атрибутов являются предметом соглашения между сторонами, использующими заданную структуру обмена.

Пример

```
DATA;
    .
    .
    .
#1 = pT (1.0, 2.0, 3.0); ————> ОБУСЛОВЛЕННЫЙ ЭКЗЕМПЛЯР ОБЪЕКТА
#2 = pT (1.0, 2.0, 5.0);
    .
    .
    .
#12 = !MYCURVE (0.0, 0.0, 0.0, 1.0, $,$, $); ——> ЭКЗЕМПЛЯР ОБЪЕКТА, ОПРЕДЕЛЕННЫЙ ПОЛЬЗОВАТЕЛЕМ
    .
    .
    .
ENDSEC;
```

Примечание — До использования синтаксиса, определенного пользователем в соответствии с настоящим разделом, следует описать EXPRESS-схему, определяющую информацию, задаваемую пользователем, и ее кодирование в отдельной секции данных.

10 Отображение из EXPRESS в структуру обмена

Данный раздел описывает, каким образом экземпляры типов данных, определенных в языке EXPRESS, отображаются в структуру обмена.

Язык EXPRESS включает в себя объявления TYPE (типов), ENTITY (объектов) и CONSTANT (констант), спецификации ограничений и описания алгоритмов. Только экземпляры типов данных, определенные как типы данных EXPRESS с помощью объявлений TYPE и ENTITY, отображаются в структуру обмена. Другие элементы языка в структуру обмена не отображаются (см. таблицу 5).

Таблица 5 — Краткая справочная таблица отображений

Элемент EXPRESS	Отображается в:
ARRAY	список (list)
BAG	список (list)
BOOLEAN	булевскую переменную (boolean)
BINARY	двоичное (binary)
CONSTANT	НЕ ОТОБРАЖАЕТСЯ
DERIVED ATTRIBUTE	НЕ ОТОБРАЖАЕТСЯ
ENTITY	экземпляр объекта
ENTITY AS ATTRIBUTE	имя экземпляра объекта
ENUMERATION	перечисление (enumeration)
FUNCTION	НЕ ОТОБРАЖАЕТСЯ
INTEGER	целое (integer)
INVERSE	НЕ ОТОБРАЖАЕТСЯ
LIST	список (list)
LOGICAL	перечисление (enumeration)
NUMBER	вещественное (real)
PROCEDURE	НЕ ОТОБРАЖАЕТСЯ
REAL	вещественное (real)

Окончание таблицы 5

Элемент EXPRESS	Отображается в:
REMARKS	НЕ ОТОБРАЖАЕТСЯ
RULE	НЕ ОТОБРАЖАЕТСЯ
SCHEMA	НЕ ОТОБРАЖАЕТСЯ
SELECT	см. 10.1.8
SET	список (list)
STRING	строку (string)
TYPE	см. 10.1.6
UNIQUE RULE	НЕ ОТОБРАЖАЕТСЯ
WHERE RULES	НЕ ОТОБРАЖАЕТСЯ

10.1 Отображение типов данных EXPRESS

Данный подраздел определяет преобразование из элементов EXPRESS, являющихся типами данных, в структуру обмена.

10.1.1 Отображение простых типов данных EXPRESS

10.1.1.1 Тип данных integer (целое)

Значения данных в EXPRESS типа INTEGER отображаются в структуру обмена как целочисленный тип данных. Состав целочисленного типа данных описан в 6.3.1.

10.1.1.2 Тип данных string (строковый)

Значения данных в EXPRESS типа STRING отображаются в структуру обмена как строковый тип данных. Состав строкового типа данных описан в 6.3.3.

10.1.1.3 Тип данных boolean (булевский)

Значения данных в EXPRESS типа BOOLEAN отображаются в структуру обмена как данные перечисляемого типа. Состав данных перечисляемого типа описан в 6.3.5. EXPRESS-данные типа BOOLEAN должны быть обработаны как предопределенный перечисляемый тип данных со значением, кодированным графическим символом "T" или "F". Эти значения соответствуют true (истина) и false (ложь).

10.1.1.4 Тип данных logical (логический)

Значения данных в EXPRESS типа LOGICAL отображаются в структуру обмена как данные перечисляемого типа. В 6.3.5 описан состав перечисляемого типа данных. EXPRESS-данные типа LOGICAL должны быть обработаны как предопределенный перечисляемый тип данных со значением, кодированным графическими символами "T", "F" или "U". Эти значения соответствуют true (истина), false (ложь) и unknown (неизвестно).

10.1.1.5 Тип данных real (вещественный)

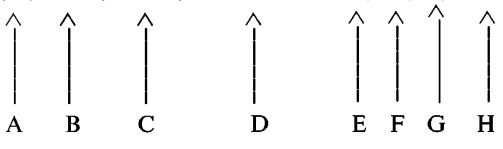
Значения данных в EXPRESS типа REAL должны быть отображены в структуру обмена как вещественный тип данных. В 6.3.2 описано содержание вещественного типа данных.

Пример — Определение объекта в языке EXPRESS:

```
ENTITY widget;
   il : INTEGER;  ————> A
   i2 : INTEGER;  ————> B
   s1 : STRING (3); ————> C
   s2 : STRING;   ————> D
   l  : LOGICAL;  ————> E
   b  : BOOLEAN;  ————> F
   r1 : REAL (4);  ————> G
   r2 :  REAL;     ————> H
END_ENTITY;
```

Образец экземпляра объекта в секции данных:

```
#2 = WIDGET (99, 99999, 'ABC', 'ABCDEFG', .T., .F., 9., 1.2345);
```



A: в этом экземпляре объекта атрибут i1 имеет значение 99.
B: в этом экземпляре объекта атрибут i2 имеет значение 99999.
C: в этом экземпляре объекта атрибут s1 имеет значение 'ABC'. Это значение попадает в диапазон (три символа), определенный для данного атрибута.
D: в этом экземпляре объекта атрибут s2 имеет значение 'ABCDEFG'.
E: в этом экземпляре объекта атрибут l имеет значение TRUE (истина).
F: в этом экземпляре объекта атрибут b имеет значение FALSE (ложь).
G: в этом экземпляре объекта атрибут r1 имеет значение 9. Требование точности не влияет на кодирование.
H: в этом экземпляре объекта атрибут r2 имеет значение 1.2345.

10.1.1.6 Тип данных binary (двоичный)

Значения данных в EXPRESS двоичного типа отображают в структуру обмена как двоичный тип данных. Описание двоичного типа данных — по 6.3.6.

Пример — Определение объекта в языке EXPRESS:

```
ENTITY picture;
   bn : BINARY;
END_ENTITY;
```

Образец экземпляра объекта в секции данных:

```
#4 = PICTURE ("1556FB0");
                ^
                |
                A
```

A: в этом экземпляре bn закодирован как "1556FB0", что соответствует последовательности битов 101 0101 0110 1111 1011 0000.

10.1.1.7 Тип данных number (числовой)

Значения данных в EXPRESS типа NUMBER должны быть отображены в структуру обмена как вещественный тип данных. Формирование вещественного типа данных — по 6.3.2.

10.1.2 Тип данных list (список)

Значения данных в EXPRESS типа LIST отображают в структуру обмена как данные спискового типа. В разделе 7 описан состав данных спискового типа. Если список пустой, то он должен быть закодирован как *левая скобка* "(", за которой сразу следует *правая скобка* ")". Внутри списка каждый экземпляр типа элемента должен быть закодирован, как требуется (согласно разделу 10) для каждого типа данных в EXPRESS.

Примечание — Если в конкретном экземпляре объекта отсутствует значение для необязательного (OPTIONAL) атрибута, чей тип данных — LIST, атрибут должен быть закодирован *знаком доллара* "$", как определено в 10.2.2, и это не пустой список.

Пример — Определение объекта в языке EXPRESS:

```
ENTITY widget;
   attribute1 : LIST [0 : ?] OF INTEGER; ----------> A
   attribute2 : LIST [1 : ?] OF INTEGER; ----------> B
   attribute3 : OPTIONAL LIST [1 : ?] OF INTEGER; ---> C
   attribute4 : REAL; ------------------------------> D
END_ENTITY;
```

Образец экземпляра объекта в секции данных:

```
#4 = WIDGET ((), (1, 2, 4), $, 2.56);
          ^   ^   ^   ^
          |   |   |   |
          A   B   C   D
```

A: attribute 1 является пустым списком (список, содержащий нуль элементов).
B: в этом экземпляре attribute 2 содержит три элемента.
C: в этом экземпляре attribute 3 не имеет значения.
D: в этом экземпляре attribute 4 имеет значение 2.56.

10.1.3 Тип данных array (массив)

Значения данных в языке EXPRESS типа ARRAY должны быть отображены в структуру обмена как данные спискового типа. Раздел 7 описывает состав данных спискового типа. Если атрибут в языке EXPRESS является многомерным массивом, он должен быть закодирован как список списков, вложенный на глубину, равную размерности массива. В создании таких списков самый внутренний список, содержащий только экземпляры типа элемента, должен соответствовать самому правому из определений ARRAY в операторе языка EXPRESS, определяющем объект. Упорядочение элементов при кодировании осуществляется так, что для каждого элемента следующего внешнего списка кодируются элементы внутреннего списка. Данный порядок означает, что в каждом списке самый правый индекс будет изменяться первым. Внутри списка каждый экземпляр типа элемента должен быть закодирован, как требуется (согласно разделу 10) для каждого типа данных в EXPRESS. Если тип данных массива имеет необязательные (OPTIONAL) элементы, любой элемент, для которого отсутствует значение, должен быть закодирован *знаком доллара* "$".

Примечание — Если в конкретном экземпляре объекта отсутствует значение для необязательного (OPTIONAL) атрибута, тип данных которого ARRAY, этот атрибут должен быть закодирован *знаком доллара* ("$"), как определено в 10.2.2, и это не пустой список.

Пример 1 — Определение объекта в EXPRESS:

```
X : ARRAY [1:5] OF ARRAY [100:102] OF INTEGER;
```

Это кодируется в следующем порядке:

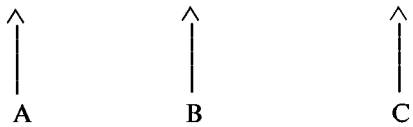
```
((X [1, 100], X [1, 101], X [1, 102] ),
(X [2, 100], X [2, 101], X [2, 102] ),
(X [3, 100], X [3, 101], X [3, 102] ),
(X [4, 100], X [4, 101], X [4, 102] ),
(X [5, 100], X [5, 101], X [5, 102] ))
```

Пример 2 — Определение объекта в языке EXPRESS:

```
ENTITY widget;
    attribute1: ARRAY [-1 : 3] OF INTEGER; ————————————> A
    attribute2 : ARRAY [1 : 5] OF OPTIONAL INTEGER;  ————————> B
    attribute3 : ARRAY [1 : 2] OF ARRAY [1 : 3] OF INTEGER; ———> C
END_ENTITY;
```

Образец экземпляра объекта в секции данных:

```
#30 = WIDGET ((1, 2, 3, 4, 5), (1, 2, 3, $, 5), ((1, 2, 3), (4, 5, 6)));
```



A : attribute1 содержит следующие значения:

```
attribute1 [-1] = 1
attribute1 [0] = 2
attribute1 [1] = 3
attribute1 [2] = 4
attribute1 [3] = 5
```

B : attribute2 содержит следующие значения:

```
attribute2 [1] = 1
attribute2 [2] = 2
attribute2 [3] = 3
attribute2 [4] = отсутствует
attribute2 [5] = 5
```

Пропуск значения в массиве предусмотрен в определении EXPRESS-схемы.

C : attribute3 содержит следующие значения:

```
attribute3 [1,1] = 1
attribute3 [1,2] = 2
attribute3 [1,3] = 3
```

attribute3 [2,1] = 4
attribute3 [2,2] = 5
attribute3 [2,3] = 6

10.1.4 Тип данных set (набор)

Значения данных в языке EXPRESS типа SET должны быть отображены в структуру обмена как данные спискового типа. Раздел 7 описывает состав данных спискового типа. Внутри списка каждый экземпляр типа элемента должен быть закодирован, как требуется (согласно разделу 10) для каждого типа данных в EXPRESS. Если SET пустой, список должен кодироваться как *левая скобка* "(", за которой сразу следует *правая скобка* ")".

Примечание — Если в конкретном экземпляре объекта отсутствует значение для необязательного (OPTIONAL) атрибута, чьим типом данных является SET, этот атрибут должен быть закодирован *знаком доллара* "$", как определено в 10.2.2, и это не пустой список.

Пример — Определение объекта в языке EXPRESS:
ENTITY widget;
a_number : SET OF INTEGER;
END_ENTITY;

Пример экземпляра объекта в секции данных:

#2 = WIDGET ((0, 1, 2)); ————> A
#3 = WIDGET ((0, $, 2)); ————> B
#4 = WIDGET ((0, 0, 2)); ————> C

A: в этом экземпляре атрибут a_number определен набором чисел 0, 1, 2.
B: синтаксически экземпляр правилен. Однако экземпляр неправилен относительно определения SET в EXPRESS, поскольку SET не может иметь пропущенных элементов.
C: синтаксически экземпляр правилен. Однако экземпляр неправилен относительно определения SET в EXPRESS, поскольку SET не может иметь одинаковых элементов.

10.1.5 Тип данных bag (мультимножество)

Значения данных в языке EXPRESS типа BAG должны быть отображены в структуру обмена как данные спискового типа. Раздел 7 описывает состав данных спискового типа. Внутри списка каждый экземпляр типа элемента должен быть закодирован, как требуется (согласно разделу 10) для каждого типа данных в EXPRESS. Если BAG пустое, список должен кодироваться как *левая скобка* "(", за которой сразу следуют *правая скобка* ")".

Примечание — Если в конкретном экземпляре объекта отсутствует значение для необязательного (OPTIONAL) атрибута, чьим типом данных является BAG, атрибут должен быть закодирован *знаком доллара* "$", как определено в 10.2.2, и это не пустой список.

Пример — Определение объекта в языке EXPRESS:
ENTITY widget;
a_numbers : BAG OF INTEGER;
END_ENTITY;

Пример экземпляра объекта в секции данных:

#2 = WIDGET ((0, 1, 1, 2)); ————> A
#3 = WIDGET ((0, $, 2)); ————> B
A: В этом экземпляре атрибут a_numbers определен набором чисел 0, 1, 1, 2.
B: Синтаксически экземпляр правилен. Однако экземпляр неправилен относительно определения BAG в EXPRESS, поскольку BAG не может иметь пропущенных элементов.

10.1.6 Простые определенные типы

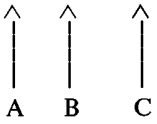
Простой определенный тип является типом, определяемым объявлением EXPRESS-типа, в котором опорным типом не является перечисляемый тип (ENUMERATION) или выбираемый тип (SELECT). Простой определенный тип должен быть отображен в структуру обмена как тот тип данных, который использовался в его определении.

Пример — Определение объекта в языке EXPRESS:
TYPE
type1 = INTEGER;
END_TYPE;

```
TYPE
    type2 = LIST [1:2] OF REAL;
END_TYPE;
ENTITY widget;
    attribute 1 : LOGICAL; ————> A
    attribute 2 : TYPE1;   ————> B
    attribute 3 : TYPE2;   ————> C
END_ENTITY;
```

Пример экземпляра объекта в секции данных:

```
#4 = WIDGET (.T., 256, (1.0, 0.0));
```



A: В этом экземпляре значением атрибута attribute 1 является TRUE (истина).
B: Type 1 является целым типом и, следовательно, значение 256 допустимо.
C: Type 2 является списком и, следовательно, список из двух вещественных (REAL) элементов допустим.

10.1.7 Тип данных enumeration (перечисление)

Значения данных в языке EXPRESS типа ENUMERATION должны быть отображены в структуру обмена как перечисляемый тип данных. В 6.3.5 описано содержание перечисляемого типа данных.

Если документ, определяющий конкретную схему, экземплярами которой является предмет этой секции данных, задающий набор сокращенных наименований для перечисляемых значений в этой схеме, тогда фактическим значением конкретного экземпляра перечисления (ENUMERATION) может быть сокращенное наименование, соответствующее одному из перечисляемых значений в данной EXPRESS-схеме. В противном случае фактическим значением должно быть одно из перечисляемых значений в EXPRESS-схеме. В любом случае *строчные буквы* должны быть преобразованы в *прописные буквы*, а значение ограничено отделенными *точками* ".", как это определено в выводе ENUMERATION в таблице 2.

Пример — Определение объекта в языке EXPRESS:

```
TYPE
    primary_colorur = ENUMERATION OF (red, green, blue);
END_TYPE;
ENTITY widget;
    p_colour : primary_colour; ————> A
END_ENTITY;
```

Пример экземпляра объекта в секции данных:

```
#2 = WIDGET (.RED.);
```

A: В этом экземпляре объекта значением атрибута p_colour является RED.

10.1.8 Выбираемый тип данных

Выбираемый тип данных в языке EXPRESS определяет список типов данных, называемый "список-выбора" ("select-list"), значениями которого являются правильные экземпляры данных выбираемого типа. Экземпляр данных выбираемого типа должен быть значением по меньшей мере одного из типов данных в списке-выбора:

- если значение является экземпляром типа данных объекта в списке-выбора, оно должно быть отображено в структуру обмена как имя экземпляра объекта (см. 6.3.4);
- если значение является экземпляром простого определяемого типа в списке-выбора, оно должно быть отображено в структуру обмена как TYPED_PARAMETER (см. таблицу 3), в котором KEYWORD должно обозначать простой определяемый тип, как определено ниже, а PARAMETER должен быть кодированием значения простого определяемого типа, как определено в 10.1.6;

- если значение является экземпляром перечисляемого типа данных в списке-выбора, оно должно быть отображено в структуру обмена как TYPED_PARAMETER (см. таблицу 3), в котором KEYWORD должно обозначать перечисляемый тип данных, как определено ниже, а PARAMETER должен быть кодированием значения перечисляемого типа данных, как определено в 10.1.7;

- если значение является экземпляром (вложенного) выбираемого типа данных, оно должно быть отображено в структуру обмена в виде экземпляра выбираемого типа, как указано в данном разделе.

Для экземпляров простых определяемых типов и перечисляемых типов данных KEYWORD должно обозначать тип данных экземпляра. Обозначенный тип данных должен быть одним из типов в списке-выбора. Если документ, определяющий схему, экземпляры которой являются предметом секции данных, также определяет набор сокращенных имен простых определяемых и перечисляемых типов в данной схеме, KEYWORD должно быть сокращенным именем, соответствующим типу данных экземпляра. В противном случае KEYWORD должно быть самим именем простого определяемого типа или перечисляемого типа данных. В обоих случаях все строчные буквы должны быть преобразованы в соответствующие прописные буквы, т.е. кодирование не должно содержать строчных букв.

Примечания

1 Если имеется тип данных (в списке-выбора), значениями экземпляров которого является сам выбираемый тип данных, тогда настоящий пункт должен быть использован рекурсивно для кодирования значения. Реально могут быть закодированы только типы данных объекта, простые определяемые типы и перечисляемые типы данных (см. пример 2).

2 В соответствии с ГОСТ Р ИСО 10303-11 экземпляр подтипа типа данных объекта является экземпляром типа данных объекта. Так, "экземпляр типа данных объекта в списке-выбора" включает в себя экземпляры подтипов соответствующих типов данных объекта.

3 Если типы данных объекта в списке-выбора не являются взаимоисключающими, тогда значение выбираемого типа данных может быть приписано нескольким типам данных объекта в списке-выбора (см. пример 1).

4 Если значение не является экземпляром объекта, то оно является экземпляром простого определяемого типа или перечисляемого типа данных. Однако значение может быть фактическим экземпляром нескольких (вложенных) выбираемых типов данных и, следовательно, приписываться нескольким типам в исходном списке-выбора (см. пример 2).

Пример 1 — Определение объекта в языке EXPRESS:

```
ENTITY Leader SUBTYPE OF (Employee);
    project: STRING;
END_ENTITY;
ENTITY Manager SUBTYPE OF (Employee);
    unit: STRING;
END_ENTITY;
ENTITY Employee;
    name: STRING;
END_ENTITY;
TYPE Supervisor = SELECT (Manager, Leader);
END_TYPE;
ENTITY Meeting;
    date : STRING;
    attendees: SET [2:?] OF Supervisor;
END_ENTITY;
```

Образцы экземпляров секции данных:

```
#1 = LEADER ('J. Brahms', 'Academic Festival');
#2 = MANAGER ('S. Ozawa', 'Tokyo Symphony');
#3 = (EMPLOYEE ('G. Verdi') LEADER ('Aida') MANAGER ('La Scala'));
#4 = MEETING ('14921012', (#1, #2, #3));
```

Вторым атрибутом экземпляра #4 являются участники: SET OF Supervisor. Экземпляром #1 является Leader, и поэтому Supervisor верен. Экземпляром #2 является Manager, и поэтому Supervisor верен. Экземпляр #3 является экземпляром двух типов Leader и Manager из списка-выбора Supervisor. Все экземпляры отображаются согласно 6.3.4.

Пример 2 — Определение объекта в языке EXPRESS:

```
TYPE Mass = SELECT(Mass_Spec, Mass_Substitute);
```

```
END_TYPE;
TYPE Mass_Spec = SELECT(Measured_Mass, Computed_Mass, Estimated_Mass);
END_TYPE;
TYPE Measured_Mass = REAL;
END_TYPE;
TYPE Computed_Mass = Extended_Real;
END_TYPE;
TYPE Estimated_Mass = REAL;
END_TYPE;
TYPE Mass_Subsitute = SELECT (Weight, Estimated_Mass);
END_TYPE;
TYPE Weight = REAL;
END_TYPE;
TYPE Extended_Real = SELECT (FloatingNumber, NotaNumber);
END_TYPE;
TYPE FloatingNumber = REAL(7);
END_TYPE;
TYPE NotaNumber = ENUMERATION OF (plus_infinity, minus_infinity, indeterminate, invalid);
END_TYPE;
ENTITY Steel_Bar;
bar_length: Extended_Real;
bar_mass : Mass;
END_ENTITY;
```

Образец конкретизации секции данных:

```
#1 = STELL_BAR(FLOATINGNUMBER(77.0), MEASURED_MASS (13.25));
#2 = STELL_BAR(NOTANUMBER(.INDETERMINATE.),
       ESTIMATED_MASS (10.0));
#3 = STEEL_BAR(FLOATINGNUMBER(77.0),
       COMPUTED_MASS(FLOATINGNUMBER(14.77719)));
```

Первый атрибут экземпляра #1 представляет значение Extended_Real, которое является FloatingNumber. Оно отображается (согласно 10.1.8 для выбираемого типа данных Extended_Real) как TYPED_PARAMETER с KEYWORD FLOATINGNUMBER (простой определяемый тип в списке-выбора). Значение 77.0 PARAMETER отображается в структуру обмена, согласно 10.1.6 для FloatingNumber, как значение простого типа REAL.

Второй атрибут экземпляра #1 представляет значение Measure_Mass, которое является правильным значением Mass_Spec и поэтому также правильным значением Mass. Оно отображается (посредством рекурсивного применения 10.1.8, поскольку Mass является выбранным типом данных, а Mass_Spec — выбираемым типом данных) как TYPED_PARAMETER с KEYWORD MEASURED_MASS (простой определяемый тип в списке-выбора). Значение 13.25 PARAMETER отображается в структуру обмена, согласно 10.1.6 для Measure_Mass, как значение простого типа REAL.

Первый атрибут экземпляра #2 представляет значение Extended_Real, являющееся значением NotaNumber. Оно отображается (согласно 10.1.8 для Extended_Real) как TYPED_PARAMETER с KEYWORD NOTANUMBER (перечисляемый тип в списке-выбора). Значение "indeterminate" PARAMETER отображается в структуру обмена согласно 10.1.7 как перечисляемый тип NotaNumber.

Второй атрибут экземпляра #2 представляет значение Estimated_Mass. Оно является правильным значением Mass_Spec, а также правильным значением Mass_Subsitute и поэтому правильным значением Mass. Это значение фактически представлено двумя (выбираемыми) типами в списке-выбора Mass. Оно отображается (посредством рекурсивного применения 10.1.8, поскольку Mass является выбранным типом данных, а Mass_Spec — выбираемым типом данных) как TYPED_PARAMETER с KEYWORD ESTIMATED_MASS (простой определяемый тип в списке-выбора). Значение 10.0 PARAMETER отображается в структуру обмена, согласно 10.1.6 для Estimated_Mass, как значение простого типа REAL.

Первый атрибут экземпляра #3 тот же, что и первый атрибут экземпляра #1.

Второй атрибут экземпляра #3 представляет значение Computed_Mass, которое является правильным значением Mass_Spec и поэтому правильным значением Mass. Оно отображается (посредством рекурсивного применения 10.1.8, поскольку Mass является выбранным типом данных, а Mass_Spec — выбираемым типом данных) как TYPED_PARAMETER с KEYWORD COMPUTED_MASS (простой определяемый тип в списке-выбора). Значение PARAMETER кодируется, согласно 10.1.6 для значения Computed_Mass, которое является значением Extended_Real. Extended_Real является выбранным типом данных. Согласно 10.1.8 значение Extended_Real кодируется как TYPED_PARAMETER с KEYWORD FLOATINGNUMBER (простой определяемый тип в списке-выбора) и как значение 14.77719 PARAMETER, кодируемое согласно 10.1.6 для FloatingNumber.

10.2 Отображение типов данных объекта из языка EXPRESS

Экземпляр типа данных объекта из EXPRESS должен быть отображен в структуру обмена как ENTITY_INSTANCE.

Как определено в ГОСТ Р ИСО 10303-11, "экземпляр простого объекта" ("simple entity instance") является экземпляром объекта, не являющегося экземпляром подтипа какого-либо типа данных объекта. Все прочие экземпляры объекта называются "экземплярами сложного объекта" ("complex entity instances"). Экземпляр простого объекта должен быть отображен согласно 10.2.1, а экземпляр сложного объекта — согласно 10.2.5.

Примечание — Экземпляр простого объекта является экземпляром объекта, который полностью описывается *единственным* объявлением объекта в EXPRESS. Экземпляр сложного объекта является экземпляром, описание которого включает в себя несколько объявлений объекта, даже если только одно из них содержит явные атрибуты. Экземпляр простого объекта может быть экземпляром супертипа, пока последний не является экземпляром какого-либо подтипа, но экземпляр подтипа всегда является «сложным».

Только явные атрибуты объекта в EXPRESS отображаются в структуру обмена. Специальные средства, однако, применяются к необязательным (OPTIONAL) явным атрибутом (см. 10.2.2), явным атрибутам, значениями которых являются экземпляры объекта (см. 10.2.4), и всем переобъявлениям явных атрибутов (см. 10.2.6—10.2.8).

Примечание — К одному и тому же атрибуту может быть применено несколько средств.

10.2.1 Отображение экземпляра простого объекта

Экземпляр простого объекта должен быть отображен в структуру обмена как SIMPLE_ENTITY_INSTANSE. Имя типа данных объекта должно быть отображено в KEYWORD для SIMPLE_RECORD, как определено в 10.2.11.

Каждый явный атрибут должен быть отображен непосредственно в PARAMETER для SIMPLE_RECORD в структуре обмена. Порядок параметров (PARAMETER) в структуре обмена должен быть тем же, что и порядок соответствующих атрибутов в объявлении объекта в EXPRESS. Первый PARAMETER должен быть значением первого явного атрибута, второй PARAMETER — значением второго явного атрибута и т.д. Если тип данных объекта в EXPRESS не имеет явных атрибутов, список параметров (PARAMETER_LIST) должен быть пустым.

Форма каждого PARAMETER должна зависеть от типа данных соответствующего атрибута, как определено в 10.1.

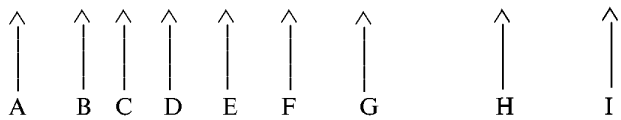
Пример — Определение в языке EXPRESS:

```
TYPE
    primary_colour_abbreviation = ENUMERATION OF (r, g, b);
END_TYPE;

ENTITY widget; ————————————————————————> A
    attribute1: INTEGER; ——————————————————> B
    attribute2: STRING; ———————————————————> C
    attribute3: LOGICAL; ——————————————————> D
    attribute4: BOOLEAN; ——————————————————> E
    attribute5: REAL; ———————————————————> F
    attribute6: LIST [1 : 2] OF LOGICAL; ——————————> G
    attribute7: ARRAY [-1 : 3] OF INTEGER; ————————> H
    attribute8: PRIMARY_COLOUR_ABBREVIATION; ————> I
END_ENTITY;
```

Образец экземпляра объекта в секции данных:

```
#1 = WIDGET(1,  'A',  .T.,  .F.,  1.0,  (.T.,.F.),  (1, 0, 1, 2, 3),  .R.);
```



A: имя объекта widget из EXPRESS отображается в стандартное ключевое слово WIDGET объекта секции данных;

B: в этом экземпляре объекта attribute1 имеет значение 1;
C: в этом экземпляре объекта attribute2 имеет значение A;
D: в этом экземпляре объекта attribute3 имеет значение T (истина);
E: в этом экземпляре объекта attribute4 имеет значение F (ложь);
F: в этом экземпляре объекта attribute5 имеет значение 1.0;
G: в этом экземпляре объекта attribute6 является списком логических переменных. Список значений:
ATTRIBUTE6(1) = T
ATTRIBUTE6(2) = F
H: в этом экземпляре объекта attribute7 является массивом целых. Список значений:
ATTRIBUTE7(-1) = 1
ATTRIBUTE7(0) = 0
ATTRIBUTE7(1) = 1
ATTRIBUTE7(2) = 2
ATTRIBUTE7(3) = 3
I: в этом экземпляре объекта attribute8 является перечислением. attribute8 имеет значение R.

10.2.2 Отображение необязательных (OPTIONAL) явных атрибутов

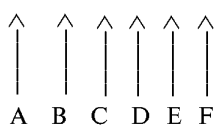
Явный атрибут, объявленный как OPTIONAL, не обязан иметь значение в заданном экземпляре объекта. Если необязательное значение поставляется в экземпляре объекта, оно должно быть закодировано в соответствии с типом данных атрибута, как определено в 10.1. Когда необязательное значение атрибута отсутствует в экземпляре объекта, оно должно быть закодировано в структуре обмена как *знак доллара* "$".

Пример — Определение объекта в языке EXPRESS:

```
ENTITY xxx;
    attribute1: REAL;
    attribute2 : REAL;
END_ENTITY;
ENTITY yyy; ————————————————————> A
attribute1 : OPTIONAL LOGICAL; ————————> B
attribute2 : xxx; ————————————————————> C
attribute3 : xxx; ————————————————————> D
attribute4 : OPTIONAL INTEGER; ——————————> E
attribute5 : OPTIONAL REAL; ———————————> F
END_ENTITY;
```

Образец экземпляров объекта в секции данных:

```
#1 = XXX (1.0, 2.0);
#2 = XXX (3.0, 4.0);
#3 = YYY ($, #2, #1, $, $);
```



A: имя объекта yyy в EXPRESS отображается в стандартное ключевое слово YYY объекта секции данных;
B: в этом экземпляре объекта attribute1 не имеет значения;
C: attribute2 является ссылкой на объект xxx экземпляром объекта #2;
D: attribute3 является ссылкой на объект xxx экземпляром объекта #1;
E: в этом экземпляре объекта attribute4 не имеет значения;
F: в этом экземпляре объекта attribute5 не имеет значения.

10.2.3 Отображение вычисляемых атрибутов

Вычисляемые атрибуты объекта не должны отображаться в структуру обмена. Когда вычисляемый атрибут в подтипе переобъявляется как атрибут в супертипе, отображение должно происходить, как описано в 10.2.6.

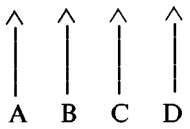
Пример — Определение объекта в языке EXPRESS:

```
ENTITY yyy;
    q0 : REAL;
    q1 : REAL;
    q2 : REAL;
END_ENTITY;
```

```
ENTITY xxx; ————> A
    p0 : yyy; ————> B
    p1 : yyy; ————> C
    p2 : yyy; ————> D
DERIVE
attrib5 : vector : = func_normal (p0, p1, p2); ————> E
attrib4 : real : = func_diameter (p0, p1, p2); ————> F
END_ENTITY;
```

Образец экземпляров объекта в секции данных:

```
#9 = YYY(0.0, 0.0, 0.0);
#10 = YYY(1.0, 2.0, 3.0);
#11 = YYY(4.0, 5.0, 6.0);
#12 = XXX(#9, #10, #11);
```



A: имя объекта xxx в EXPRESS отображается в стандартное ключевое слово XXX объекта секции данных;
B: p0 является ссылкой на объект yyy экземпляром объекта #9;
C: p1 является ссылкой на объект yyy экземпляром объекта #10;
D: p2 является ссылкой на объект yyy экземпляром объекта #11;
E: attrib5 в этом экземпляре объекта не отображается, поскольку он является вычисляемым атрибутом;
F: attrib4 в этом экземпляре объекта не отображается, поскольку он является вычисляемым атрибутом.

10.2.4 Отображение атрибутов, значения которых являются экземплярами объектов

Если экземпляр объекта определен как атрибут другого (ссылающегося на него) экземпляра объекта, первый (на который ссылаются) экземпляр объекта должен быть отображен в структуру обмена как имя экземпляра объекта (см. 6.3.4). Ссылка на этот экземпляр объекта может быть внутри секций данных, то есть где-то в пределах секций данных экземпляр объекта, на который ссылаются, должен быть указан слева от *знака равенства* "=". Это определение может предшествовать применению экземпляра объекта в качестве атрибута или следовать за ним. Его описание не должно входить в эту секцию в качестве атрибута используемого экземпляра объекта.

Примечание — В приложении F описаны методы оценки соответствия схемы при наличии в структуре обмена нескольких секций данных, основанных на разных EXPRESS-схемах, включая ссылки между экземплярами объектов, определенных в этих секциях на основе разных EXPRESS-схем.

Пример — Определение объекта в языке EXPRESS:

```
ENTITY yyy;
    x : REAL;
    y : REAL;
    z : REAL;
END_ENTITY;

ENTITY xxx;
    p0 : yyy; ————> A
    p1 : yyy; ————> B
END_ENTITY;
```

Образец экземпляра объекта в секции данных:

```
#1 = YYY (3., 4., 5.);
#2 = XXX (#1, #3);
#3 = YYY (1., 2., 3.);
```

10.2.5 Объекты, определенные как подтипы других объектов

ГОСТ Р ИСО 10303-11 определяет экземпляры объекта, имеющего раздел SUBTYPE (подтип), являющийся "экземплярами сложных объектов", так, что они могут включать в себя атрибуты из нескольких объявлений типов объектов. В настоящем пункте определено, как экземпляры сложных объектов должны быть отображены в структуру обмена.

Экземпляры сложного объекта должны быть отображены в структуру обмена на основе одного из двух правил: внутреннего отображения или внешнего отображения. К каждому экземпляру подтипа объекта должно быть применено одно правило отображения.

Примечания

1 Выбор отображения зависит в большей мере от экземпляра объекта, чем от его типа. Для разных экземпляров одного и того же типа данных объекта возможно использование разных отображений в зависимости от того, являются ли они экземплярами подтипов и какие подтипы они представляют.

2 Настоящий пункт применим только к экземплярам сложного объекта. Нет необходимости применять его к каждому экземпляру объекта супертипа. В частности, он не применяется к экземпляру супертипа, который не является экземпляром любого подтипа. Такие экземпляры могут существовать, если супертип не является абстрактным супертипом и подтипом какого-либо другого объекта. Такие экземпляры отображают согласно 10.2.1.

Правило отображения, которое следует использовать для каждого экземпляра объекта, выбирают в зависимости от класса соответствия, выбранного для реализации. Для реализаций с классом соответствия 1 выбор отображения описан в 10.2.5.1. Для реализаций с классом соответствия 2 для всех экземпляров сложных объектов должно быть использовано внешнее отображение, описанное в 10.2.5.3.

10.2.5.1 Выбор отображения по умолчанию

Набор определений типа данных объекта, которые связаны выражениями подтипа и явного или неявного супертипа, определяет набор структур экземпляров сложного объекта, на который ссылаются как на определяемое множество в приложении В ГОСТ Р ИСО 10303-11. Каждый член определяемого множества устанавливает список имен типов данных объектов.

Каждый конкретный экземпляр типа данных объекта соответствует одному элементу определяемого множества. Отображение, применяемое к конкретному экземпляру, зависит от члена определяемого множества, которому соответствует экземпляр.

Для того чтобы установить, какое из правил отображения надо применить к данному экземпляру объекта:

a) определяют список имен типов данных объекта, который становится элементом определяемого множества, соответствующим экземпляру объекта;

b) отбирают из списка все типы объектов, не имеющие подтипы, и все типы объектов, которые могут иметь подтипы, но для которых не определены подтипы в списке (члене определяемого множества) для данного экземпляра;

c) в случае определения только одного типа данных объекта его следует считать “конечным типом данных объекта” (“leaf entity data type”), и должно быть применено внутреннее отображение. В противном случае должно быть использовано внешнее отображение.

Примечание — При реализации положения b) должен быть отобран по меньшей мере один тип объекта.

10.2.5.2 Внутреннее отображение

Если используется внутреннее отображение, экземпляр объекта должен быть отображен в SIMPLE_ENTITY_INSTANCE (см. таблицу 3). Ключевое слово (KEYWORD) должно быть именем конечного типа данных объекта, как указано в 10.2.11. Список параметров (PARAMETER_LIST) должен содержать значения унаследованных явных атрибутов всех объектов супертипа и явных атрибутов конечного типа данных объекта. Порядок, в котором унаследованные и явные атрибуты будут появляться в структуре обмена, должен быть определен следующим образом:

- все унаследованные атрибуты должны появляться последовательно перед явными атрибутами любого объекта;
- атрибуты объекта супертипа должны наследоваться в порядке их появления в самом объекте супертипа;
- если объект супертипа сам является подтипом другого объекта, атрибуты более высокого супертипа должны наследоваться первыми;
- когда указано несколько объектов супертипа, атрибуты объектов супертипа должны быть обработаны в порядке, определенном в выражении SUBTYPE OF.

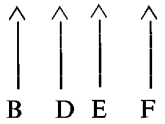
В результате этой процедуры на объект супертипа может быть несколько ссылок. В этом случае все ссылки, кроме первой, должны быть игнорированы.

Пример 1 — Простое отношение подтип/супертип. Определение объекта в языке EXPRESS:

```
ENTITY aa ABSTRACT SUPERTYPE OF (ONEOF(bb, cc)); ————————> A
    attrib_a : zz; ————————————————————————————————> B
END_ENTITY;
ENTITY bb SUBTYPE OF (aa)
                ABSRTACT SUPERTYPE OF (ONEOF(xx)); ————————> C
    attrib_b1 : yy;————————————————————————————————> D
    attrib_b2 : yy; ————————————————————————————————> E
END_ENTITY;
ENTITY cc SUBTYPE OF (aa);
    attrib_c : REAL;
END_ENTITY;
ENTITY xx SUBTYPE OF (bb);
    attrib_x : REAL; ————————————————————————————————> F
END_ENTITY;
ENTITY zz;
    attrib_z : STRING;
END_ENTITY;
ENTITY yy
    attrib_1 : REAL;
    attrib_2 : REAL;
    attrib_3 : REAL;
END_ENTITY;
```

Образец экземпляра объекта типа данных объекта xx в секции данных:

```
#1 = ZZ(ZATTR');
#2 = YY(1.0, 2.0, 0.0);
#3 = YY (2.0, 2.0, 0.0);
#4 = XX(#1, #2, #3, 4.0);
```



A: поскольку объект aa является абстрактным супертипом, он не отображается в структуру обмена;

B: атрибут attrib_a будет отображен в секции данных как унаследованный атрибут того объекта, который прямо или косвенно является подтипом объекта aa. В этом случае attrib_a представлен первым атрибутом экземпляра объекта xx и ссылается на zz экземпляр объекта #1;

C: поскольку объект bb является абстрактным супертипом, он не отображается в структуре обмена;

D: атрибут attrib_b1 будет отображен в секции данных как унаследованный атрибут объекта, который прямо или косвенно является подтипом объекта bb. В этом случае attrib_b1 представлен вторым атрибутом экземпляра объекта xx и ссылается на yy — экземпляр объекта #2;

E: атрибут attrib_b2 будет отображен в секции данных как унаследованный атрибут объекта, который прямо или косвенно является подтипом объекта bb. В этом случае attrib_b2 представлен третьим атрибутом экземпляра объекта xx и ссылается на yy — экземпляр объекта #3;

F: атрибут attrib_x представлен своим значением 4.0.

Пример 2 — Отображение супертипа, который не является абстрактным (ABSTRACT) супертипом. Определение объекта в языке EXPRESS:

```
ENTITY aa SUPERTYPE OF (ONEOF(bb, dd)); ————————> A
    attrib_a : STRING;
END_ENTITY;
ENTITY bb SUBTYPE OF (aa); ————————————————————> B
    attrib_b : INTEGER;
END_ENTITY;
ENTITY cc SUBTYPE OF (bb); ————————————————————> C
    attrib_c : REAL;
END_ENTITY;
ENTITY dd SUBTYPE OF (aa); ————————————————————> D
    attrib_d : REAL;
END_ENTITY;
ENTITY ee; ————————————————————————————————————> E
```

```
    attrib_e : aa;
END_ENTITY;
```

Образец экземпляра объекта в секции данных:

```
#1 = AA ('SAMPLE STRING'); ————————> A
#2 = BB )'ABC'); ——————————————————> B
#3 = CC ('DEF', 123); ————————————> C
#4 = DD('XYZ', 99.99); ———————————> D
#5 = EE(#1); ——————————————————————> E
#6 = EE (#2); —————————————————————> E
#7 = EE(#3); —————————————————————> E
#8 = EE(#4); —————————————————————> E
```

A: поскольку объект aa не является абстрактным супертипом, он может присутствовать в структуре обмена. Объект имеет только один атрибут attrib_a, когда присутствует в структуре.

B: объект bb является подтипом aa и, следовательно, его экземпляры будут содержать атрибуты и aa, и bb, но так как объект bb не определяет все атрибуты, в данном атрибуте должен присутствовать только список параметров attrib_a.

C: объект cc является подтипом bb и, следовательно, его экземпляры будут содержать атрибуты aa, bb и cc.

D: объект dd является подтипом aa и, следовательно, его экземпляры должны содержать атрибуты объектов aa и bb.

Пример 3 — Отображение объекта с несколькими супертипами в выражении SUBTYPE OF. Определение объекта в языке EXPRESS.

```
ENTITY base SUPERTYPE OF (branch_one, brabch_two); ————————> A
    attrib_a : STRING;
END_ENTITY;
ENTITY brabch one SUBTYPE OF (base); ——————————————————————> B
    attrib_b : INTEGER;
END_ENTITY;
ENTITY brabch_two SUBTYPE OF (base); ——————————————————————> C
    attrib_c : BOOLEAN;
END_ENTITY;
ENTITY leaf SUBTYPE OF (branch one, brabch_two); ——————————> D
    attrib_d : REAL;
END_ENTITY;
```

Образец экземпляра объекта в секции данных:

```
#1 = BASE('SAMPLE STRING'); ————————> A
#2 = BRABCH_ONE('ABC', 123); ————————> B
#3 = BRABCH_TWO('DEF', .T.); ————————> C
#4 = LEAF('XYZ', 123, .T., 99.99); ————————> D
```

A: Объект не имеет супертипов. При внесении в структуру обмена список его параметров должен содержать только значение атрибута attrib_a.

B: объект branch_one является подтипом основного типа. При внесении в структуру обмена список его параметров должен содержать унаследованные атрибуты, следующие за атрибутом branch_one.

C: объект branch_two является подтипом основного типа. При внесении в структуру обмена список его параметров должен содержать унаследованные атрибуты, следующие за атрибутом branch_two.

D: объект перечисления, являющийся подтипом объектов branch_one и branch_two. При внесении в структуру обмена список его параметров должен содержать унаследованные атрибуты branch_one, включающие в себя атрибуты основного объекта, следующие за унаследованными атрибутами объекта branch_two. Атрибуты основного объекта должны быть описаны однократно при описании атрибутов branch_one. Они должны быть проигнорированы при конфликте с атрибутами объекта branch_two.

10.2.5.3 Внешнее отображение

Если используется внешнее отображение, экземпляр объекта должен быть отображен в COMPLEX_ENTITY_INSTANCE (см. таблицу 3).

ГОСТ Р ИСО 10303-11 определяет "частное значение сложного объекта" ("partial complex entity value") как множество значений атрибутов, описанных единственным EXPRESS-объявлением объекта. Каждое имя типа данных объекта в элементе определяемого множества обозначает частное значение сложного объекта для данного экземпляра объекта. Таким образом, элемент определяемого множества обозначает множество частных значений сложного объекта, что, вместе с именем экземпляра объекта, полностью описывает заданный экземпляр объекта.

Каждое частное значение сложного объекта, обозначаемое именем типа данных объекта в элементе определяемого множества, должно быть отображено в SIMPLE_RECORD (единичная запись) внутри SUBSUPER_RECORD. Порядок SIMPLE_RECORD внутри SUBSUPER_RECORD должен быть возрастающей последовательностью имен типов данных объекта с использованием схемы упорядочения, приведенной в 5.2.

Каждая SIMPLE_RECORD должна кодировать одно частное значение сложного объекта. KEYWORD в каждой SIMPLE_RECORD должно кодировать соответствующее имя типа данных объекта, как определено в 10.2.11, а PARAMETER_LIST должен кодировать значения явных атрибутов, если таковые появляются в соответствующем объявлении объекта. Порядок PARAMETER в структуре обмена должен быть тем же, что и порядок соответствующих атрибутов в EXPRESS-объявлении объекта. Если EXPRESS-объявление объекта не содержит явных атрибутов, PARAMETER_LIST должен быть пустым. Форма каждого PARAMETER должна зависеть от типа данных соответствующего атрибута согласно требованиям 10.1.

Примечания

1 Последовательность частных значений объекта (SIMPLE_RECORD) определяется именем типа данных объекта (так называемым "длинным именем"), а не "сокращенным именем" (если таковое есть), которое может использоваться для кодирования.

2 Каждое частное значение объекта в определяемом множестве должно быть представлено, включая супертипы, не имеющие явных атрибутов.

Пример 1 — Отображение подтипов, связанных посредством ANDOR.

Определение объекта в языке EXPRESS:

```
ENTITY base SUPERTYPE OF (bb ANDOR cc); ————————> A
    attrib_a : STRING;
END_ENTITY;
ENTITY bb SUBTYPE OF (aa); ——————————————————> B
    attrib_b : INTEGER;
END_ENTITY;
ENTITY cc SUBTYPE OF (aa); ——————————————————> C
    attrib_c : REAL;
END_ENTITY;
ENTITY dd; ——————————————————————————————> D
    attrib_d : aa;
END_ENTITY;
```

Образец экземпляра объекта в секции данных:

```
#1 = BB('sample string', 15); ————————————> A
#2 = CC('S', 3.0); ———————————————————————> B
#3 = (AA('ASTRID')BB(17)CC(4.0)); ————————> C
#4 = DD(#1); —————————————————————————————> D
#5 = DD(#2); —————————————————————————————> D
#6 = DD#3; ———————————————————————————————> D
#7 = AA('ABC');——————————————————————————> E
```

A: #1 является экземпляром комбинации aa и bb.

B: #2 является экземпляром комбинации aa и cc.

C: #3 является экземпляром комбинации aa, bb и cc.

D: объект dd ссылается на объект aa как на атрибут. Следовательно, экземпляр объекта dd может иметь разрешенные ссылки на любой из экземпляров #1, #2 или #3.

E: aa не является абстрактным супертипом и может быть представлен экземплярами, для которых применяется внутреннее отображение, поскольку в этом случае определяемое множество состоит только из одного элемента.

Пример 2 — Отображение более сложного графа подтип/супертип. Определение объекта в языке EXPRESS:

```
ENTITY x;
    attrib_x : INTEGER;
END_ENTITY;
ENTITY a ABSTRACT SUPERTYPE OF (ONEOF(b, c)); ————> A
    attrib_a : x; ————————————————————————————> B
END_ENTITY;
ENTITY b SUPERTYPE OF (d ANDOR e);
```

```
        SUBTYPE OF (a);
    attrib_b : REAL; ————————————> B
END_ENTITY;
ENTITY c SUBTYPE OF (a); ————————> C
    attrib_c : REAL;
END_ENTITY;
ENTITY d SUBTYPE OF (b); ————————> D
    attrib_d: x;
END_ENTITY;
ENTITY e ABSTRACT SUPERTYPE
        SUBTYPE OF (b); ————————————> A
    attrib_e : x; ————————————————————> B
END_ENTITY;
ENTITY f SUPERTYPE OF (h);
    attrib_f: x; ————————————————————> B
END_ENTITY;
ENTITY g SUBTYPE OF (e); ————————> E
    attrib_g: INTEGER;
END_ENTITY;
ENTITY h SUBTYPE OF (e, f); ————————> E
    attrib_h: INTEGER;
END_ENTITY;
```

A: поскольку объекты *a* и *e* являются абстрактными супертипами, они не могут присутствовать в структуре обмена в виде самостоятельных экземпляров.

B: поскольку attrib_a, attrib_b, attrib_e, и attrib_f являются атрибутами объектов супертипа, они будут отображены как унаследованные атрибуты в том случае, если при отображении подтипа применено внутреннее отображение. Эти атрибуты будут отображены как атрибуты объектов, в которых они объявлены, если подтип отображен с использованием внешнего отображения.

C: поскольку объект *c* участвует в операторе ONEOF, а его супертип не участвует ни в каких операторах с супертипами, для объекта *c* будет использовано внутреннее отображение.

D: отображение *d* зависит от структуры определяемого множества, в котором оно появляется.

E: поскольку объекты *g* и *h* имеют супертип (объект *e*), который участвует в операторе ANDOR, их отображение будет зависеть от структуры определяемого множества, в котором они появились.

3 Экземпляр объекта, показывающий внутреннее отображение.

```
#1 = X(1);
#2 = C(#1, 2.0);
```

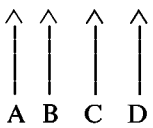
A: определяемым множеством для '#2' является [c & a] и, следовательно, используется внутреннее отображение.

B: attrib_a унаследован типом данных объекта *c*. Определяемое множество является ссылкой на экземпляр типа данных объекта *x*.

C: attrib_c появляется после всех унаследованных атрибутов.

4 Экземпляр объекта, показывающий внутреннее отображение.

```
#4 = X(3);
#1 = X(1);
#2 = D(#1, 2.0, #4);
```



A: поскольку экземпляр объекта #2 принадлежит определяемому множеству [a & b & d], которое имеет точно одну вершину (d), он отображается внутренним образом.

B: атрибут объекта *a* с именем attrib_a унаследован типом данных объекта *d*.

C: attrib_b унаследован типом данных объекта *d*.

D: attrib_d является последним атрибутом в экземпляре *d* потому, что атрибуты, унаследованные от объектов супертипа *a* и *b*, появляются первыми.

5 Экземпляр объекта, показывающий внешнее отображение.
#1 =X(1);
#2 = (A(#1) B(9.0) D(#1) E(#1) F(#1) H(4)); ————————> A

A: поскольку экземпляр объекта #2 является членом [a & b & d & e & f & h] и это определяемое множество оценки имеет более одной вершины (листа) (d и h), используется внешнее отображение. Не существует единого имени типа данных объекта, которое может быть связано с объектом, скорее объект должен иметь составное имя a-b-d-e-f-h. *Пробелы* между записями объектов необязательны и добавлены в этом примере для того, чтобы облегчить чтение.

10.2.6 Явные атрибуты, переобъявленные как DERIVE

Если объект подтипа переобъявляет атрибут своего супертипа с помощью оператора DERIVE, а исходный атрибут является явным, то значение исходного атрибута в супертипе должно кодироваться звездочкой "*".

Пример — Определение объекта в языке EXPRESS:

```
ENTITY point;
    x : REAL;
    y : REAL;
    z : REAL;
END_ENTITY;
ENTITY point_on_curve SUBTYPE OF (point);
    u : REAL;
    c : curve;
DERIVE
    SELF\point.x : real := fx(u, c);
    SELF\point.y : real := fy(u, c);
    SELF\point.z : real := fz(u, c);
END_ENTITY;
ENTITY curve;
    attr : STRING;
END_ENTITY;
```

Образец экземпляра объекта в секции данных:

#1 = CURVE('curve_attribute');
#2 = POINT _ON_CURVE(*, *, *, 0.55, #1); ————————> A
#3 = POINT(2.0, 3.0, 4.0); ————————————————————> B

A: поскольку здесь используется подтип с вычисляемыми атрибутами, атрибуты *x, y и z* заменены *звездочками.*

B: поскольку POINT не является абстрактным супертипом, в структуре обмена может присутствовать экземпляр POINT. Атрибуты *x, y и z* появляются как нормальные атрибуты.

10.2.7 Атрибуты, переобъявленные как INVERSE

Если объект подтипа переобъявляет атрибут своего супертипа с помощью оператора INVERSE, это не влияет на кодирование. Переобъявленный атрибут в любом случае не кодируют.

10.2.8 Атрибуты, переобъявленные как явные атрибуты

Если объект подтипа переобъявляет атрибут одного из своих супертипов как явный атрибут, т.е. не с помощью оператора INVERSE или DERIVE, это не влияет на кодирование.

Значение атрибута должно быть закодировано как атрибут супертипа (см. 10.2.5) с применением отображения, определенного в разделе 10 для типа данных атрибута в супертипе. Переобъявленный атрибут должен игнорироваться, т.е. он не должен применяться в качестве атрибута подтипа для целей кодирования.

Пример — Определение объекта в языке EXPRESS:

```
ENTITY aaa SUPERTYPE OF (ONEOF (bbb, ccc));
    a1 : NUMBER;
    a2 : curve;
INVERSE
    a3 : SET OF mmm FOR m1;
END_ENTITY;
ENTITY bbb SUBTYPE OF (aaa);
    SELF\aaa.a1: INTEGER;
    b          : REAL;
```

```
END_ENTITY;
ENTITY ccc SUBTYPE OF (aaa);
    SELF\aaa.a2 : line;
INVERSE
    SELF\aaa.a3 : SET [1:2] OF mmm FOR m1;
END_ENTITY;
ENTITY curve;
    . . .
END_ENTITY;
ENTITY line SUBTYPE OF (curve);
    . . .
END_ENTITY;
ENTITY mmm;
    m1 : aaa;
END_ENTITY;
```

Образцы экземпляров в секции данных:

```
#1 = LINE(. . .);
#2 = CURVE(. . .);
#3 = BBB(1.0, #2, 0.5);
#4 = CCC(1.5, #1,);
```

Для экземпляров #3 и #4 кодирование такое же, как если бы не было переобъявляемых атрибутов в объектах bbb и ccc.

10.2.9 Локальные правила для объектов

Для объектов локальными являются правила WHERE и UNIQUE, которые не должны отображаться в структуру обмена.

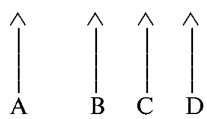
Пример — Определение объекта в языке EXPRESS:

```
ENTITY widget; ————————————————> A
    a : REAL; ————————————————> B
    b : REAL; ————————————————> C
    c : REAL; ————————————————> D
WHERE
    a ** 2 + b ** 2 + c ** 2 = 3.0; ————————> E
END_ENTITY;
```

Образец экземпляра объекта в секции данных:

Правила WHERE не представлены в экземпляре объекта.

```
    #2 = WIDGET(1.0, 1.0, 2.0);
```



A: имя объекта widget на языке EXPRESS отображено в ключевое слово типа данных объекта для объекта секции данных.

B: в экземпляре объекта атрибут *a* имеет значение 1.0.

C: в экземпляре объекта атрибут *b* имеет значение 1.0.

D: в экземпляре объекта атрибут *c* имеет значение 2.0.

E: правило WHERE не отображено в структуру обмена. Объект синтаксически правилен. Однако значения трех атрибутов не удовлетворяют правилу WHERE.

10.2.10 Отображение инверсных (INVERSE) атрибутов

Атрибуты, описанные в операторе INVERSE, не должны отображаться в структуру обмена.

10.2.11 Кодирование имен типов объектов

Если документ, определяющий схему, экземпляры которой являются предметом секции данных, определяет также и набор сокращенных имен для каждого типа данных объекта в схеме, то эти сокращенные имена должны быть использованы при кодировании имен типов данных объекта. В другом случае именами типов данных объекта при кодировании будут сами имена типов данных объекта. В обоих случаях все *строчные буквы* должны быть преобразованы в соответствующие *прописные буквы*, т.е. код не должен содержать никаких *строчных букв*.

10.3 Отображение элемента EXPRESS для SCHEMA

Элемент EXPRESS для SCHEMA не должен отображаться в структуру обмена. Имя схемы, которая описывает объекты, появляющиеся в структуре обмена, должно быть отображено в заголовочной секции структуры обмена с помощью экземпляра типа данных объекта **file_schema**, как это описано в 8.2.3.

10.4 Отображение элемента EXPRESS для CONSTANT

Элемент EXPRESS для CONSTANT не должен отображаться в структуру обмена.

Примечание — Когда данные отображают в структуру обмена, ссылки на одну и ту же константу не сохраняют.

10.5 Отображение элемента EXPRESS для RULE

Элементы EXPRESS для RULE не должны отображаться в структуру обмена.

10.6 Комментарии

Комментарии не должны отображаться в структуру обмена.

11 Печатное представление структур обмена

Для управления появлением структуры обмена в печатном виде могут быть использованы комбинации графических символов, как это описано в таблице 6. Эти директивы могут быть представлены в любой позиции, где может появиться разделитель лексем (token separator), а также внутри строк (string) и двоичных значений (binaries). Подробности, касающиеся разделителей лексем, приведены в таблице 3. Комбинации графических символов должны появляться вместе без вставки между ними каких-либо графических символов.

Внутри строк также разрешены явные директивы управления процедурой печати. Директивы "\N\" и "\F\" не включают в значимое содержание строки. При кодировании строк см. 6.3.3. Директивы управления процедуры печати "\N\" и "\F\" используют только для печатного представления структуры обмена, и в других случаях они должны игнорироваться. Приложение G представляет руководство по распечатке структуры обмена.

Таблица 6 — Директивы управления печатью

Последовательность графических символов	Значение
\N\ REVERSE_SOLIDUS N REVERSE_SOLIDUS	Новая строка
\F\ REVERSE_SOLIDUS F REVERSE_SOLIDUS	Перевод страницы

ПРИЛОЖЕНИЕ А
(обязательное)

Представление файла на носителе данных

Различают следующие виды носителей, содержащие структуры обмена:

- перенос с доступом к записям (Record-oriented transport) обычно характеризуется как магнитная лента, но может быть любым носителем данных с доступом к записям;
- перенос с доступом к строкам (Line-oriented transport) обычно характеризуется как дискета, но может быть любым носителем данных с доступом к строкам.

В обоих случаях целью является пересылка последовательности графических символов основного алфавита, как описано в настоящем стандарте. Для носителей, содержащих структуру обмена, применяют следующие соглашения.

А.1 Содержание носителя с доступом к записям

Магнитная лента может содержать несколько наборов данных. Каждый набор может быть последовательным файлом, содержание которого может быть интерпретировано как структура обмена, соответствующая настоящему стандарту.

Отправитель ленты несет ответственность за сообщение любому получателю ленты (через ленту или иным способом) информации о том, какие наборы данных являются структурами обмена, соответствующими настоящему стандарту.

Форматы транспортировки (транспортные форматы) для каждой структуры обмена следующие:

- файл должен состоять из последовательности записей;
- первыми графическими символами в первой записи должна быть специальная лексема “ISO-10303-21;”, которая инициирует структуру обмена. Форматирование носителя данных зависит от операционной системы и является предметом соглашения между отправляющей и получающей сторонами;
- последняя запись должна быть заполнена (если необходимо) *пробелами* за *точкой с запятой*, относящейся к специальной лексеме “END-ISO-10303-21;”.

А.1.1 Формат передачи для носителя на магнитной ленте

Для передачи на ленте обменных файлов предпочтительный формат обмена определяют следующие характеристики, установленные в ИСО 3788:

- ширина ленты 12,7 мм (0,5 дюйма);
- неразмеченная лента;
- девятидорожечная лента;
- плотность записи на ленте — 63 символ/мм (1600 бит/дюйм);
- размер физического блока — 4000 октетов (8-битных байтов);
- блоки разделены пропусками между записями;
- за последним блоком должен следовать маркер ленты.

Также может быть использован другой стандарт носителя на магнитной ленте, например магнитная лента с плотностью записи 246 символ/мм (6250 бит/дюйм).

А.1.2 Другие носители данных с доступом к записи

Другие носители, на которых структуры обмена хранятся как последовательности записей, могут использовать также формат передачи, определенный для лент.

А.2 Содержание носителя с доступом к строкам

Некоторые магнитные носители содержат наборы данных, хранящиеся как последовательность строк. Каждый из этих наборов может быть последовательным файлом, содержание которого может быть интерпретировано как файл, соответствующий настоящему стандарту.

Файл должен состоять из последовательности строк:

- первые графические символы в первой строке должны быть специальной лексемой “ISO-10303-21;”, которая инициирует структуру обмена;
- последняя строка должна быть дополнена при необходимости пробелами за специальной лексемой “END-ISO-10303-21;”, завершающей структуру обмена.

Средства ограничения строк и представления конца файла зависят от операционной системы и являются предметом соглашения между отправителем и получателем с учетом того, что ограничители не должны использовать каких-либо символов основного алфавита. При любом их представлении ограничители строк и файла должны игнорироваться при обработке структуры обмена.

А.2.1 Формат передачи для носителя на дискете

Для передачи структуры обмена может быть использован любой из популярных носителей на дискетах (3,5 дюйма, 5,25 дюйма, низкой или высокой плотности записи).

Поставщик такой дискеты несет ответственность за сообщение любому принимающему информацию (через дискету или иным способом) о том, какие наборы данных являются структурами обмена, соответствующими настоящему стандарту.

А.2.2 Другие носители

Другие носители, на которых файлы хранятся как последовательности строк, могут использовать тот же формат передачи, который определен для дискет. В частности, этот формат может быть пригоден для передачи через сети связи (E-mail).

А.3 Обработка многотомных файлов

Может быть необходимым распределить структуру обмена на нескольких физических томах. Способ организации многотомных файлов в настоящем стандарте не рассматривается.

Примечание — В зависимости от конкретных условий объединять физически раздельные тома многотомного файла в одну многотомную структуру можно с помощью специальных программных средств или операционной системы.

ПРИЛОЖЕНИЕ В
(обязательное)

Соглашения по записи в синтаксической нотации Вирта

Синтаксис структуры обмена определен в синтаксической нотации Вирта (CHB), опубликованной Никлаусом Виртом в "Communications of the ACM, 20:11 (Nov 77), 822-823". CHB состоит из набора выводов или правил подстановки. Элемент в левой части вывода (т. е. перед знаком равенства) может быть использован для представления появления образца в правой части. Элементарные символы, которые появляются только в правой части таких выводов, называются терминальными. Элементы, которые появляются в левой части выводов, называются нетерминальными.

Соглашения по записи даны ниже. Таблица В.1 представляет самоопределенную CHB.

Строка *прописных букв* является элементом языка; строка является именем элемента (для удобства *строчные буквы* используются для неопределенных идентификаторов и комментариев).

Любая строка, заключенная в *кавычки*, точно определяет содержимое внутри *кавычек*. Единственным исключением из этого правила является знак *кавычек* внутри текста. Чтобы реализовать его, знак *кавычек* сразу же повторяют один раз. Последовательность ’’’’’’’интерпретируется как", а последовательность “AB””C" — как AB"C.

Знак равенства “=” означает вывод. Определено, что элемент слева должен быть комбинацией элементов справа. Любые *пробелы*, появляющиеся между элементами вывода, не являются значащими, если они появляются не внутри литерала. Вывод завершается *точкой* “.”.

Фигурные скобки “{ }” означают нуль или более повторений.

Квадратные скобки “[ ]” означают необязательный параметр.

Вертикальная линия “|” означает логическое ИЛИ.

Скобки “( )” показывают приоритет операций. В частности, там, где скобки включают в себя элементы, разделенные *вертикальными линиями*, один из элементов должен быть выбран в сочетании с любой другой операцией.

Пример — Последовательность “A(B|C|D)” эквивалентна “AB|AC|AD”.

Таблица В.1 — Самоопределенная синтаксическая нотация Вирта (CHB)

```
SYNTAX     = { PRODUCTION }.
PRODUCTION = IDENTIFIER “=” EXPRESSION “.”.
EXPRESSION = TERM { “|” TERM }.
TERM       = FACTOR { FACTOR }.
FACTOR     = IDENTIFIER
           | LITERAL
           | “[” EXPRESSION “]”
           | “(” EXPRESSION “)”
           | “{” EXPRESSION “}”
IDENTIFIER = letter { letter }.
LITERAL    = ’’’’’’’’ character { character } ’’’’’’’’.
```

ПРИЛОЖЕНИЕ С
(обязательное)

Регистрация информационного объекта

С.1 Обозначение документа

Для того чтобы обеспечить однозначное обозначение информационного объекта в открытой системе, настоящему стандарту присвоен идентификатор объекта

{ iso standard 10303 part(21) version(3) }.

Смысл этого значения определен в ГОСТ Р ИСО/МЭК 8824-1 и описан в ГОСТ Р ИСО 10303-1.

С.2 Обозначение схемы

Для того чтобы обеспечить однозначное обозначение **header_section_schema** в открытой информационной системе, **header_section_schema** (см. раздел 8) присвоен идентификатор объекта

{ iso standard 10303 part(21) version(3) object (1) header-section-schema (1) }.

Смысл этого значения определен в ГОСТ Р ИСО/МЭК 8824-1 и описан в ГОСТ Р ИСО 10303-1.

ПРИЛОЖЕНИЕ D
(обязательное)

Основной алфавит и набор графических символов

Графическими символами, используемыми в структуре обмена, являются символы столбцов 02-07 Латинского алфавита № 1 согласно ИСО/МЭК 8859-1.

Примечание — Графические символы изображены в таблице D.1. В верхней строке приведены старшие четыре бита данного байта, в самом левом столбце — младшие четыре бита.

Таблица D.1 — Набор символов основного алфавита, используемый в структуре обмена

	0010 32	0011 48	0100 64	0101 80	0110 96	0111 112
0000 0		**0**	**@**	**P**		**p**
0001 1	**!**	**1**	**A**	**Q**	**a**	**q**
0010 2	**“**	**2**	**B**	**R**	**b**	**r**
0011 3	**#**	**3**	**C**	**S**	**c**	**s**
0100 4	**$**	**4**	**D**	**T**	**d**	**t**
0101 5	**%**	**5**	**E**	**U**	**e**	**u**
0110 6	**&**	**6**	**F**	**V**	**f**	**v**
0111 7	**‘**	**7**	**G**	**W**	**g**	**w**
1000 8	**(**	**8**	**H**	**X**	**h**	**x**
1001 9	**)**	**9**	**I**	**Y**	**i**	**y**
1010 10	*****	**:**	**J**	**Z**	**j**	**z**
1011 11	**+**	**;**	**K**	**[**	**k**	**{**
1100 12	**,**	**<**	**L**	****	**l**	**|**
1101 13	**-**	**=**	**M**	**]**	**m**	**}**
1110 14	**.**	**>**	**N**	**^**	**n**	**~**
1111 15	**/**	**?**	**O**	**_**	**o**	

ПРИЛОЖЕНИЕ Е
(обязательное)

Форма заявки о соответствии реализации протоколу

Форма заявки о соответствии реализации протоколу (ЗСРП) предназначена для обеспечения оценки соответствия реализаций настоящему стандарту. Настоящее приложение составлено в форме вопросника. Данный вопросник должен быть заполнен реализатором и может быть использован испытательной лабораторией при подготовке аттестационного тестирования заявленной реализации.

Все реализаторы должны ответить на вопросы, заданные в пунктах Е.1 и Е.2.

Е.1 Соответствие заданной функции

Проверяют соответствие указанному в ЗСРП.

Е.1.1 Кодирование экземпляра объекта

Обеспечивает ли реализация класс соответствия 1?:

___ для чтения, ___ для записи.

Обеспечивает ли реализация класс соответствия 2?:

___ для чтения, ___ для записи.

Е.1.2 Кодирование сокращенных имен

Обеспечивает ли реализация представление сокращенных имен объектов?:

___ для чтения, ___ для записи.

Обеспечивает ли реализация представление сокращенных имен для значений выбранных типов?:

___ для чтения, ___ для записи.

Обеспечивает ли реализация представление сокращенных имен для перечисляемых значений?:

___ для чтения, ___ для записи.

Е.1.3 Кодирование строки

Обеспечивает ли реализация кодирование в формате \X\ для 8-битных байтов?:

___ для чтения; если да, каково двоичное представление, использованное в реализации?,

___ для записи.

Обеспечивает ли реализация кодирование символов в форматах \S\ и \P\ по стандартам серии ИСО/МЭК 8859?:

_ для чтения; если да, каково двоичное представление, использованное в реализации?,

_ для записи.

Обеспечивает ли реализация кодирование символов в форматах \X2\ по стандартам серии ИСО/МЭК 10646?:

___ для чтения; если да, каково двоичное представление, использованное в реализации?,

___ для записи.

Обеспечивает ли реализация кодирование символов в форматах \X4\ по стандартам серии ИСО/МЭК 10646?:

___ для чтения; если да, каково двоичное представление, использованное в реализации?,

___ для записи.

Е.2 Ограничения реализации

Каково максимальное число схем, на которые можно ссылаться в структуре обмена?

Каково максимальное число секций данных, входящих в структуру обмена?

Каково максимальное число экземпляров объектов, входящих в секцию данных?

Каково максимальное число экземпляров объектов, входящих в структуру обмена?

Каково максимальное значение (или двоичное представление, использованное в реализации) для идентификаторов экземпляров объектов?

Каковы максимальные и минимальные значения (или двоичное представление, использованное в реализации) для целых чисел (INTEGER) в соответствии с языком EXPRESS?

Каковы ограничения по точности (или двоичное представление, использованное в реализации) для вещественных чисел (REAL) в соответствии с языком EXPRESS?

Какова максимальная длина строки (STRING) в соответствии с языком EXPRESS?

Какова максимальная длина двоичного числа (BINARY) в соответствии с языком EXPRESS?

Каково максимальное число элементов, могущих входить в агрегат (массив) при его кодировании?

Какова максимальная глубина вложения при кодировании вложенных агрегатов (массивов)?

ПРИЛОЖЕНИЕ F
(обязательное)

Множество EXPRESS-схем в структуре обмена

В структуре обмена с несколькими секциями данных могут быть приведены ссылки на экземпляры объектов, определенные в секциях данных по различным EXPRESS-схемам. При реализации ссылок между схемами возникают два вопроса: 1) какие ссылки можно считать верными?; 2) какие экземпляры следует считать правильными, основанными на конкретной схеме, при определении схематического соответствия структуре обмена.

F.1 Допустимые ссылки

В настоящем подразделе описаны два метода определения правильности ссылок между экземплярами объектов при установлении схематического соответствия для структуры обмена. Можно использовать другие методы, не определенные в настоящем стандарте.

Примечание — В данном подразделе описаны способы, посредством которых автор документов, определяющих EXPRESS-схемы, может так же задать правильные ссылки между двумя или несколькими схемами.

При определении схематического соответствия реализация может ссылаться на конкретный документ, описывающий данную схему в рамках определений на языке EXPRESS, сокращенных имен и других требований или ограничений. Если заданную схему используют в сочетании с другими, данный документ так же должен определять правильность ссылок между этими схемами.

Если предполагается, что данное определение делается однократно и охватывает ряд схем, не следует дополнительно определять верность ссылок в конкретной структуре обмена.

F.1.1 Метод спецификации интерфейса на языке EXPRESS

При использовании данного метода ссылки между экземплярами объектов различных схем должны быть заданы посредством определения интерфейса на языке EXPRESS в соответствии с разделом 11 ГОСТ Р ИСО 10303-11.

На экземпляр типа, заданный в схеме, можно ссылаться через экземпляр типа, определенный в другой схеме, если этот тип связан с последней посредством конструктивов USE или REFERENCE.

Пример — Рассматриваются две схемы и основанная на них структура обмена.

```
SCHEMA base;
ENTITY a;
    range : REAL;
END_ENTITY;
ENTITY b;
    name : STRING;
END_ENTITY;
END_SCHEMA;
SCHEMA extension;
USE FROM base (a, b);
ENTITY c;
    addressed_item : b;
    address : STRING;
END_ENTITY;
RULE a_range_positive FOR (a);
WHERE
    WR1: SIZEOF (QUERY (inst <* a | inst.range < 0)) = 0;
END_RULE;
END_SCHEMA;
ISO-10303-21;
HEADER;
    .
    .
FILE_ SCHEMA (( 'BASE', 'EXTENSION'));
ENDSEC;
DATA ('ONE', ('BASE'));
#1 = A(-3.5);
#2 = B('Sam Smith');
```

```
#3 = B('John Doe');
ENDSEC;
DATA ('TWO', ('EXTENSION'));
#4 = C(#2, '100 Main Street');
#5 = C(#3, '1300 Elmwood Avenue');
ENDSEC;
END-ISO-10303-21;
```

При использовании метода задания требований к интерфейсу на языке EXPRESS для определения правильности ссылок реализация должна учитывать, что объект типа B явно связан со схемой 'EXTENSION', поэтому допустимы ссылки из #4 на #2 и из #5 на #3.

F.1.2 Метод определения эквивалентности области значений СИДД

При использовании данного метода ссылки между экземплярами объектов, заданными в разных схемах, должны быть выполнены на основе методов определения эквивалентности области значений, установленных в ГОСТ Р ИСО 10303-22.

При задании экземпляра типа, определенного в схеме, экземпляры любых типов объектов из данной схемы, ссылающиеся на данный тип, также могут ссылаться на экземпляры любых типов, объявленных в эквивалентной области значений первого типа.

Пример — Рассматриваются две схемы и основанная на них структура обмена.

```
SCHEMA LONGA;
ENTITY a;
    range : REAL;
END_ENTITY;
ENTITY b;
    name : STRING;
END_ENTITY;
END_SCHEMA;
SCHEMA LONGB;
ENTITY a;
    range : REAL;
END_ENTITY;
ENTITY b;
    name : STRING;
END_ENTITY;
ENTITY c;
    addressed_item : b;
    address : STRING;
END_ENTITY;
RULE a_range_positive FOR (a);
WHERE
    WR1: SIZEOF (QUERY (inst <* a | inst.range < 0)) = 0;
END_RULE;
END_SCHEMA;
ISO-10303-21;
HEADER;
    .
    .
FILE_ SCHEMA (( 'LONGA', 'LONGB'));
ENDSEC;
DATA ('ONE', ('LONGA'));
#1 = A(-3.5);
#2 = B('Sam Smith');
#3 = B('John.Doe');
ENDSEC;
DATA ('TWO', ('LONGB'));
#4 = C(#2, '100 Main Street');
#5 = C(#3, '1300 Elmwood Avenue');
ENDSEC;
END-ISO-10303-21;
```

В настоящем примере предполагается наличие информации об области эквивалентности значений, заданной с использованием одного из методов, описанных в ГОСТ Р ИСО 10303-22, устанавливающей эквивалентность области значений типов A и B в обеих схемах. Это представлено в виде:

LONGA.A is DEQ to LONGB.A
LONGB.A is DEQ to LONGB.A

LONGA.B is DEQ to LONGB.B
LONGB.B is DEQ to LONGA.B

При использовании области эквивалентности для определения правильности ссылок реализация должна учитывать, что тип объекта LONGA.B является областью эквивалентности значений для LONGB.B, что допускает ссылки из #4 на #2 и из #5 на #3.

F.2 Определение совокупности схемы

Объект **file_population** в заголовочной секции связывает EXPRESS-схему и набор (коллекцию) экземпляров объектов с конкретной структурой обмена. Атрибут **determination_method** определяет алгоритм выбора коллекции экземпляров объектов, заданных в наборе секций данных. В настоящем подразделе описаны три метода их определения. Можно использовать другие методы, не определенные в настоящем стандарте.

При определении схематического соответствия структуры обмена коллекция экземпляров объектов, заданная в соответствующих **file_population**, должна быть проверена на соответствие EXPRESS-схеме. Если на какую-либо секцию данных нет ссылки из какого-либо объекта **file_population**, эта секция должна быть проверена на соответствие определенной в ней схемы по методу ограничения секции, описанному ниже.

При проверке соответствия схемы установленным требования и ограничениям ссылки на экземпляры объектов, сделанные вне коллекции данных экземпляров, должны восприниматься как ошибочные.

Примечание — Последующая процедура описывает метод проверки структуры обмена в соответствии с 8.2.4 и вышеизложенным параграфом.

Для каждого объекта **file_population**, имеющего в структуре обмена значение "F":

- находят набор экземпляров посредством использования метода, заданного объектом **F.determination_method**, для секций данных, поименованных в объекте **F.governed_sections**. Если объект **F.governed_sections** является пустым (неопределенным), то данный метод используют для всех секций данных в структуре обмена;
- проверяют данный набор на соответствие правилам и ограничениям, заданным в объекте **F.governing_schema;**
- отмечают секции данных, являющиеся исходными данными для объекта **F.determination_method**.

Для каждой немаркированной секции данных "D" проверяют набор экземпляров в этой секции на соответствие правилам и ограничениям, заданным в схеме.

F.2.1 Метод ограничения секции

При использовании метода ограничения секции атрибут **F.determination_method** должен иметь значение "SECTION_BOUNDARY". Набор (коллекция) экземпляров объектов, заданный в качестве исходных данных для одной или нескольких секций, должен содержать:

- все экземпляры из заданной секции данных.

Примеры

1 Рассмотрим схемы и структуру обмена, описанные в примере из F.1.1. Заголовочная секция не содержит каких-либо экземпляров объекта **file_population**. При определении схематического соответствия структуры обмена должно быть учтено следующее:

- на секцию данных ONE не ссылаются из какого-либо объекта **file_population**, поэтому следует проверить все экземпляры объектов этой секции на соответствие задающей ее схеме. Все экземпляры в секции данных ONE должны удовлетворять требованиям и ограничениям схемы BASE. В данном примере совокупность удовлетворяет всем ограничениям схемы BASE;
- на секцию данных TWO не ссылаются из какого-либо объекта **file_population**, поэтому следует проверить все экземпляры объектов этой секции на соответствие задающей ее схеме. Все экземпляры в секции данных ONE должны удовлетворять требованиям и ограничениям схемы EXTENSION. В настоящем примере секция данных не содержит каких-либо экземпляров объекта A, поэтому она удовлетворяет правилу **a_range_positive**. Атрибут **addressed_item** экземпляров #4 и #5 ссылается на экземпляры вне данной совокупности, поэтому эти ссылки следует рассматривать как ошибочные. Атрибут **addressed_item** не является факультативным (необязательным), поэтому данная совокупность не удовлетворяет ограничениям схемы EXTENSION.

2 Рассмотрим схемы и структуру обмена, описанные в примере из F.1.1, но с заголовочной секцией, приведенные ниже:

```
HEADER;
    .
    .
FILE_SCHEMA (('BASE', 'EXTENSION'));
FILE_POPULATION ('BASE', 'SECTION_BOUNDARY', ('ONE'));
FILE_POPULATION ('EXTENSION', 'SECTION_BOUNDARY', ('ONE', 'TWO'));
ENDSEC;
```

При определении схематического соответствия структуры обмена должно быть учтено следующее:

- первый объект **file_population** определяет набор (коллекцию) экземпляров, управляемых схемой BASE. Все экземпляры в секции данных ONE должны удовлетворять требованиям и ограничениям схемы BASE. В данном примере совокупность удовлетворяет всем ограничениям схемы BASE;
- второй объект **file_population** определяет набор (коллекцию) экземпляров, управляемых схемой EXTENSION. Все экземпляры в секции данных ONE и TWO должны удовлетворять требованиям и ограничениям схемы EXTENSION. В рассматриваемом примере правило **a_range_positive** нарушено в экземпляре #1, поэтому данная совокупность не соответствует ограничениям схемы EXTENSION.

F.2.2 Описание всех совместимых методов

Атрибут **determination_method** должен иметь значение "INCLUDE_ALL_COMPATIBLE" в случае использования всех совместимых методов. Набор (коллекция) экземпляров объектов, заданный в качестве исходных данных для одной или нескольких секций данных должен охватывать:

- все экземпляры в регулируемых секциях данных;
- все экземпляры в других секциях данных, где тип объекта экземпляра позволяет ссылаться на экземпляры, заданные схемой управления совокупностью.

Пример — Рассмотрим схемы и структуру обмена, описанные в примере из F.1.1, но с заголовочной секцией, приведенные ниже:

```
HEADER;
    .
    .
FILE_SCHEMA (('BASE', 'EXTENSION'));
FILE_POPULATION ('BASE', 'INCLUDE_ALL_COMPATIBLE', ('ONE'));
FILE_POPULATION ('EXTENSION', 'INCLUDE_ALL_COMPATIBLE', ('TWO'));
ENDSEC;
```

При определении схематического соответствия структуры обмена должно быть учтено следующее:

- первый объект **file_population** определяет набор (коллекцию) экземпляров, управляемых схемой BASE. Данный набор содержит все экземпляры из секции данных ONE. Он так же должен содержать экземпляры из секции данных TWO, допускающие ссылки на экземпляры, заданные схемой BASE. В настоящем примере это не рассмотрено. Однако секция данных TWO включает экземпляры типов A и B, которые могут быть ограниченными. В данном примере совокупность удовлетворяет всем ограничениям схемы BASE;
- второй объект **file_population** определяет набор (коллекцию) экземпляров, управляемых схемой EXTENSION. Этот набор содержит все экземпляры из секции данных TWO. Он также содержит все экземпляры типов A и B из секции данных ONE, потому что их типы позволяют ссылаться на экземпляры, определенные схемой EXTENSION. В рассматриваемом примере правило **a_range_positive** нарушено в экземпляре #1, поэтому данная совокупность не соответствует ограничениям схемы EXTENSION.

F.2.3 Описание методов ссылки на экземпляр

Атрибут **determination_method** должен иметь значение "INCLUDE_REFERENCED" в случае использования метода ссылки на экземпляр. Набор (коллекция) экземпляров объектов, заданный в качестве исходных данных для одной или нескольких секций данных, должен охватывать:

- все экземпляры в регулируемых секциях данных;
- экземпляры из других секций данных, которые ссылаются на экземпляры в заданных секциях данных.

Пример — Рассмотрим схемы и структуру обмена, описанные в примере из F.1.1, но с заголовочной секцией, приведенные ниже:

```
HEADER;
    .
    .
FILE_SCHEMA (('BASE', 'EXTENSION'));
FILE_POPULATION ('BASE', 'INCLUDE_ REFERENCED ', ('ONE'));
FILE_POPULATION ('EXTENSION', 'INCLUDE_ REFERENCED', ('TWO'));
ENDSEC;
```

При определении схематического соответствия структуры обмена должно быть учтено следующее:

- первый объект **file_population** определяет набор (коллекцию) экземпляров, управляемых схемой BASE. Данный набор содержит все экземпляры из секции данных ONE. Он так же должен содержать экземпляры из секции данных TWO, допускающие ссылки на экземпляры, заданные в секции данных ONE. В настоящем примере это не рассмотрено. В данном примере совокупность удовлетворяет всем ограничениям схемы BASE;
- второй объект **file_population** определяет набор (коллекцию) экземпляров, управляемых схемой EXTENSION. Этот набор содержит все экземпляры из секции данных TWO. Он также содержит все экземпляры #2 и #3 из секции данных ONE, потому что их типы позволяют ссылаться на экземпляры, из секции данных TWO. В рассматриваемом примере экземпляр #1 не является членом совокупности, поэтому правило **a_range_positive** удовлетворено. Данная совокупность соответствует всем ограничениям схемы EXTENSION.

ПРИЛОЖЕНИЕ G
(справочное)

Руководство по распечатке структуры обмена

Структура обмена допускает включение необязательных директив, которые явно управляют представлением структуры в напечатанном виде. Когда такие директивы не включены и должно быть создано печатное представление структуры обмена, следует использовать набор неявных указаний по управлению процессом печати, определенный в G.2. Явные директивы управления при печати перекрывают неявные указания по управлению процессом печати.

G.1 Явные директивы управления процессом печати

Явные директивы управления процессом печати могут быть использованы в случаях, когда отправитель нуждается в точном управлении появлением структуры обмена в напечатанном виде.

Директива *обратная косая черта, прописная буква N, обратная косая черта* "\N\" указывает, что первый символ, следующий сразу за директивой, появляется в начале новой строки. Директива *обратная косая черта, прописная буква F, обратная косая черта* "\F\" указывает, что печать будет продолжена с новой страницы. В любом случае директивы управления процессом печати сами в напечатанном виде не появляются.

Примечание — Процесс печати структуры обмена обычно не должен управляться явными или неявными директивами управления при печати. Подразумевается, что директивы управления процессом печати будут использоваться только тогда, когда отправитель требует управления представлением структуры обмена в напечатанном виде. Эта ситуация может иметь место, если структура обмена является частью официального контракта.

G.2 Неявные директивы управления процессом печати

При отсутствии явных директив управления при печати структуры обмена могут быть использованы следующие директивы:

a) строка выключается слева;

b) каждая секция начинается с новой строки;

c) объекты заголовочной секции должны начинаться с начала новой строки;

d) имя экземпляра объекта, предшествующее *знаку равенства* "=", должно начинаться с новой строки;

e) лексемы, не являющиеся строками (string) и двоичными значениями (binary), не должны разделяться между строками. Строки и двоичные значения могут быть разделены только в том случае, если они целиком не умещаются в одной строке. Директива h) определяет, где должны разделяться строки и двоичные значения;

f) комментарии должны начинаться с новой строки;

g) разделитель лексем, не являющийся комментарием, не должен разделяться между строками. Комментарий может разделяться, если он не помещается в одной строке. Директива h) определяет, где будут разделяться комментарии;

h) строки должны иметь длину не больше чем 72 символа. Если символ не может быть напечатан в качестве 73-й позиции, его следует напечатать в первой позиции новой строки.

ПРИЛОЖЕНИЕ Н
(справочное)

Пример полной структуры обмена

Н.1 Введение

Ниже представлен пример определения средствами EXPRESS схемы, таблицы сокращенных имен и структуры обмена. Приведенная EXPRESS-схема не отражает содержания какого-либо стандарта серии ГОСТ Р ИСО 10303.

Н.2 Пример схемы

Определение EXPRESS-схемы, используемые в примере структуры обмена.

```
SHEMA example_ geometry
TYPE length_measure = NUMBER;
END_TYPE;
ENTITY geometry
SUPERTYPE OF (ONEOF(point));
END_ENTITY;
ENTITY point
SUPERTYPE OF (ONEOF(cartesian_point));
SUBTYPE OF (geometry);
END_ENTITY;
ENTITY cartesian_ point
SUBTYPE OF (point);
    x_coordinate : length_measure;
    y_coordinate : length_measure;
    z_coordinate : OPTIONAL length_measure;
END_ENTITY;
TYPE edge_or_logical = SELECT (edge, edge_logical_structure);
END_TYPE;
ENTITY topology
SUPERTYPE OF (ONEOF(vertex, edge, loop));
END_ENTITY;
ENTITY vertex
SUBTYPE OF (topology);
    vertex_point : OPTIONAL point;
END_ENTITY;
ENTITY edge
SUBTYPE OF (topology);
    edge_start : vertex;
    edge_end  : vertex;
END_ENTITY;
ENTITY edge_logical_structure;
    edge_element : edge;
    flag : BOOLEAN;
END_ENTITY;
ENTITY loop
SUPERTYPE OF (ONEOF(edge_loop));
SUBTYPE OF (topology);
END_ENTITY;
ENTITY edge_loop
SUBTYPE OF (loop);
    loop edges : LIST [1:?] OF edge_or_logical;
END_ENTITY;
END_SCHEMA;
```

Н.3 Пример сокращенных имен

Ниже даны сокращенные имена объектов вышеприведенной схемы.

Имя объекта	Сокращенное имя
cartesian_point	cpt
vertex	vx
edge	ed
edge_logical_structure	ed_strc
edge_loop	ed_loop

Н.4 Пример структуры обмена

Ниже приведен пример полной структуры обмена. Поскольку схема является только примером, в заголовочной секции экземпляра объекта **file_schema** в атрибуте **schema_name** отсутствует регистрационный идентификатор схемы (id).

```
ISO-10303-21;
HEADER;
FILE_DESCRIPTION(('THIS FILE CONTAINS A SMALL SAMPLE STEP MODEL'), '3; 1');
FILE_NAME('EXAMPLE STEP FILE #1',
'1992-02-11T15:30:00',
('JOHN DOE',
'ACME INC.',
'METROPOLIS USA'),
('ACME INC. A SUBSIDIARY OF GIANT INDUSTRIES', METROPOLIS USA'),
'CIM/STEP VERSION2',
'SUPER CIM SYSTEM RELEASE 4.0',
'APPROVED BY JOE BLOGGS');
FILE_SCHEMA(('EXAMPLE GEOMETRY'))'
ENDSEC;
DATA;
/*
СЛЕДУЮЩИЕ 13 ОБЪЕКТОВ ПРЕДСТАВЛЯЮТ КОНТУР СТОРОН ТРЕУГОЛЬНИКА
*/
#1 = CPT(0.0, 0.0, 0.0);                ./* ЭТО ОБЪЕКТ ТИПА "ДЕКАРТОВА ТОЧКА" */
#2 = CPT(0.0, 1.0, 0.0);
#3 = CPT(1.0, 0.0, 0.0);
#11 = VX(#1);                           /* ЭТО ОБЪЕКТ ТИПА "ВЕРШИНА" */
#12 = VX(#2);
#13 = VX(#3);
#16 = ED(#11, #12);                     /* ЭТО ОБЪЕКТ ТИПА "СТОРОНА" */
#17 = ED(#11, #13);
#18 = ED(#13, #12);
#21 = ED_STRC(#17, .F.);      /* ЭТО ОБЪЕКТ ТИПА "ЛОГИЧЕСКАЯ СТРУКТУРА СТОРОНЫ" */
#22 = ED_STRC(#18, .F.);
#23 = ED _STRC(#16, .T.);
#24 = ED_LOOP((#21, #22, #23));  /* ЭТО ОБЪЕКТ ТИПА "КОНТУР СТОРОН" */
/*
*/ ВОЗМОЖНЫ ДРУГИЕ СИНТАКСИЧЕСКИЕ ПРЕДСТАВЛЕНИЯ. ПРЕДЫДУЩИЙ ПРИМЕР
ПРЕДСТАВЛЯЕТ ОДИН ИЗ ВОЗМОЖНЫХ МЕТОДОВ.
*/
ENDSEC;
END-ISO-10303-21;
```

Примечание — Данный пример структуры обмена был отредактирован для улучшения его читаемости, для чего добавлены пробелы, не являющиеся обязательными.

Тематический указатель

УДК 656.072:681.3:006.354 ОКС 25.040.40 П87 ОКСТУ 4002

Ключевые слова: автоматизация, средства автоматизации, прикладные автоматизированные системы, промышленные изделия, данные, представление данных, обмен данными, преобразование данных, реализация, открытый текст, кодирование

Редактор *Т.А. Леонова*
Технический редактор *В.Н. Прусакова*
Корректор *Н.Л. Рыбалко*
Компьютерная верстка *И.А. Налейкиной*

Подписано в печать 24.05.2006. Формат 60 × 84$^{1}/_{8}$. Бумага офсетная. Гарнитура Таймс.
Печать офсетная. Усл. печ.л. 6,51. Уч.-изд.л. 6,10. Тираж 14 экз. Зак. 155. С 2873

ФГУП «Стандартинформ», 123995 Москва, Гранатный пер., 4.
www.gostinfo.ru info@gostinfo.ru
Набрано и отпечатано во ФГУП «Стандартинформ»