МЕЖДУНАРОДНЫЙ СОЮЗ ЭЛЕКТРОСВЯЗИ

МСЭ-Т Z.141

СЕКТОР СТАНДАРТИЗАЦИИ
ЭЛЕКТРОСВЯЗИ МСЭ

(03/2006)

СЕРИЯ Z: ЯЗЫКИ И ОБЩИЕ АСПЕКТЫ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ДЛЯ СИСТЕМ ЭЛЕКТРОСВЯЗИ

Методы формального описания (FDT) – Нотация тестирования и управления тестированием (TTCN)

Нотация тестирования и управления тестированием, версия 3 (TTCN-3): Формат табличного представления (TFT)

Рекомендация МСЭ-Т Z.141

РЕКОМЕНДАЦИИ МСЭ-Т СЕРИИ Z

ЯЗЫКИ И ОБЩИЕ АСПЕКТЫ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ДЛЯ СИСТЕМ ЭЛЕКТРОСВЯЗИ

МЕТОДЫ ФОРМАЛЬНОГО ОПИСАНИЯ (FDT)	
Язык спецификации и описания (SDL)	Z.100–Z.109
Применение методов формального описания	Z.110–Z.119
Диаграмма последовательности сообщений (MSC)	Z.120–Z.129
Расширенный язык описания объектов (eODL)	Z.130–Z.139
Нотация тестирования и управления тестированием (TTCN)	**Z.140–Z.149**
Нотация требований пользователя (URN)	Z.150–Z.159
ЯЗЫКИ ПРОГРАММИРОВАНИЯ	
CHILL: язык высокого уровня МСЭ-Т	Z.200–Z.209
ЯЗЫК "ЧЕЛОВЕК–МАШИНА"	
Общие принципы	Z.300–Z.309
Базисный синтаксис и диалоговые процедуры	Z.310–Z.319
Расширенный язык MML для видеотерминалов	Z.320–Z.329
Спецификация интерфейса "человек–машина"	Z.330–Z.349
Информационно-ориентированные интерфейсы "человек–машина"	Z.350–Z.359
Интерфейсы "человек–машина" для управления сетями электросвязи	Z.360–Z.379
КАЧЕСТВО	
Качество программного обеспечения электросвязи	Z.400–Z.409
Аспекты качества рекомендаций, относящихся к протоколам	Z.450–Z.459
МЕТОДЫ	
Методы проверки и тестирования	Z.500–Z.519
ПРОМЕЖУТОЧНОЕ ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ	
Среда распределенной обработки	Z.600–Z.609

Для получения более подробной информации просьба обращаться к перечню Рекомендаций МСЭ-Т.

Рекомендация МСЭ-Т Z.141

Нотация тестирования и управления тестированием, версия 3 (TTCN-3): Формат табличного представления (TFT)

Резюме

В настоящей Рекомендации определяется TFT – табличный формат для TTCN-3. TFT – это формат табличного представления для базового языка TTCN-3 (*Нотации тестирования и управления тестированием 3*), определенного в Рек. МСЭ-Т Z.140. По внешнему представлению и функциональным характеристикам он аналогичен формату TTCN-2, определенному в Рек. МСЭ-Т X.292 для тестирования на соответствие. Табличный формат обеспечивает альтернативный способ вывода базового языка, а также выделения тех аспектов, которые характерны для требований стандартизированного набора тестов на соответствие. Тогда как базовый язык может использоваться независимо от формата табличного представления, табличный формат не может использоваться без базового языка. Использовать и внедрять формат табличного представления следует на основе базового языка. В настоящей Рекомендации определяются проформы, синтаксические отображения, дополнительная статическая семантика, рабочие семантические ограничения, визуализация и другие атрибуты. Эти характеристики образуют вместе формат табличного представления.

TFT воспринимает все важнейшие характеристики базового языка и предназначен для описания последовательностей тестирования, которые не зависят от платформ, методов тестирования, уровней протоколов и протоколов. TTCN-3 может использоваться для описания всех типов реактивного системного тестирования относительно разнообразных портов связи. Типичные области применения – тестирование протоколов (включая протокол для подвижной связи и протокол Интернет), тестирование услуг (включая дополнительные услуги), тестирование модулей, тестирование платформ на основе CORBA и интерфейсов прикладного программирования (API). Описание наборов тестов для протоколов физического уровня не входит в сферу применения настоящей Рекомендации.

Источник

Рекомендация МСЭ-Т Z.141 была утверждена 16 марта 2006 года 17-й Исследовательской комиссией МСЭ-Т (2005–2008 гг.) в соответствии с процедурой, изложенной в Рекомендации МСЭ-Т A.8.

ПРЕДИСЛОВИЕ

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

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

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

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

ПРИМЕЧАНИЕ

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

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

ПРАВА ИНТЕЛЛЕКТУАЛЬНОЙ СОБСТВЕННОСТИ

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

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

© ITU 2006

Все права сохранены. Никакая часть данной публикации не может быть воспроизведена с помощью каких-либо средств без письменного разрешения МСЭ.

СОДЕРЖАНИЕ

Рекомендация МСЭ-Т Z.141

Нотация тестирования и управления тестированием, версия 3 (TTCN-3): Формат табличного представления (TFT)

1 Сфера применения

В настоящей Рекомендации определяется формат табличного представления для TTCN версии 3 (или TTCN-3). Эта Рекомендация основана на базовом языке TTCN-3, определенном в Рек. МСЭ-Т Z.140 [1].

Описание других форматов не входит в сферу применения настоящей Рекомендации.

2 Справочные документы

Указанные ниже Рекомендации МСЭ-Т и другие источники содержат положения, которые путем ссылки на них в данном тексте составляют положения настоящей Рекомендации. На момент публикации указанные издания были действующими. Все рекомендации и другие источники могут подвергаться пересмотру; поэтому всем пользователям данной Рекомендации предлагается изучить возможность применения последнего издания рекомендаций и других источников, перечисленных ниже. Перечень действующих в настоящее время рекомендаций МСЭ-Т регулярно публикуется. Ссылка на документ в данной Рекомендации не придает ему как отдельному документу статус рекомендации.

[1] ITU-T Recommendation Z.140 (2006), *Testing and Test Control Notation version 3 (TTCN-3): Core language*.

[2] ITU-T Recommendation Z.143 (2006), *Testing and Test Control Notation version 3 (TTCN-3): Operational semantics*.

3 Аббревиатуры

Для целей настоящей Рекомендация применяются следующие аббревиатуры:

ASN.1 Абстрактно-синтаксическая нотация версии один

ATS Абстрактный набор тестов

BNF Форма Бэкуса-Наура

MTC Основной компонент тестирования

PICS Свидетельство о соответствии реализации протоколу

PIXIT Дополнительная информация для тестирования реализации протокола

TFT Формат табличного представления для TTCN-3

TTCN Нотация тестирования и управления тестированием

4 Введение

Формат табличного представления для TTCN-3 (TFT) – это графический формат, который по внешнему представлению и функциональным характеристикам аналогичен более ранним версиям TTCN, ориентированным на тестирование на соответствие. Базовый язык TTCN-3 определен в Рек. МСЭ-Т Z.140 [1] и обеспечивает синтаксис на основе полного текста, статическую семантику, а также определение использования языка при ASN.1. Рабочая семантика определена в Рек. МСЭ-Т Z.143 [2]. Табличный формат обеспечивает дополнительное средство визуализация базового языка, а также выделения тех аспектов, которые характерны для требований стандартизированного набора тестов на соответствие.

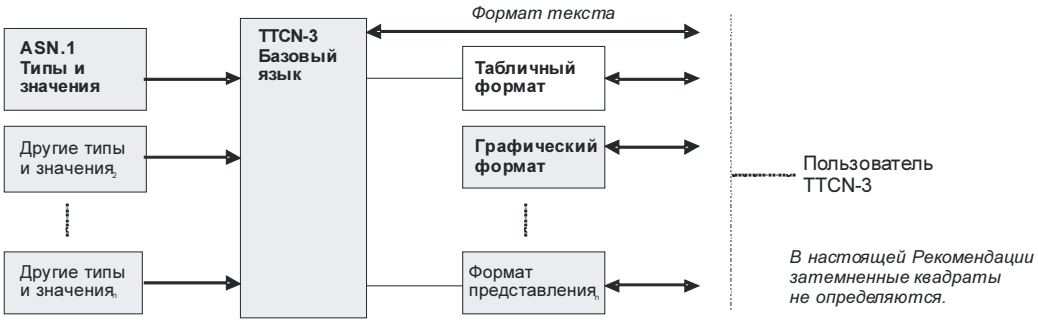

Рисунок 1/Z.141 – Видимые для пользователя форматы базового языка и различные форматы представления

Базовый язык может использоваться независимо от формата табличного представления. Однако табличный формат не может использоваться без базового языка. Использовать и внедрять формат табличного представления следует на основе базового языка.

В настоящей Рекомендации определяются:

a) проформы;

b) синтаксические отображения;

c) дополнительная статическая семантика;

d) рабочие семантические ограничения;

e) вывод и другие атрибуты.

Эти характеристики образуют вместе формат табличного представления.

5 Условные обозначения

В настоящей части определяются условные обозначения, которые использовались при определении проформ TTCN и грамматики базового языка TTCN.

5.1 Синтаксическая метанотация

В таблице 1 определяется метанотация, используемая для описания расширенной грамматики BNF для TTCN (в дальнейшем BNF).

Таблица 1/Z.141 – Синтаксическая метанотация TTCN.MP

::=	определяется как
abc xyz	за abc следуют xyz
|	альтернативный вариант
[abc]	0 или 1 копии abc
{abc}	0 или более копий abc
{abc}+	1 или более копий abc
(...)	группировка текста
abc	неконечный символ abc
abc	конечный символ abc
"abc"	конечный символ abc

Продукция BNF определена в Приложении A/Z.140 [1].

5.2 Текст описания

a) **Выделенный полужирным текст** используется для ссылок на поля проформы.

b) *Выделенный курсивом текст* используется для ссылок на продукцию BNF базового языка для TTCN-3.

c) **`Выделенный полужирным шрифтом Bold courier new`** текст используется для ключевых слов базового языка.

5.3 Проформы

a) **Выделенный полужирным текст** приводится дословно в каждой действительной таблице в модуле TTCN-3.

b) *Выделенный курсивом текст* не приводится дословно в модуле TTCN-3. Этот шрифт используется для указания на то, что имеющийся текст заменяется для выделенного курсивом символа. Синтаксические требования для имеющегося текста можно найти, либо следуя определению проформы, либо в BNF базового языка для TTCN-3. Квадратные скобки перед и после *выделенного курсивом текста* указывают на то, что включение текста в конкретное поле проформы является факультативным.

5.4 Базовый язык

a) **Выделенный полужирным текст** с помещенными в кавычках символами (например, "{") используется для зарезервированных ключевых слов и терминалов в базовом языке.

b) *Выделенный курсивом текст* не приводится дословно в модуле TTCN-3. Этот шрифт используется для указания на то, что имеющийся текст заменяется для выделенного курсивом символа. Синтаксические требования для имеющегося текста можно найти, либо следуя определению проформы, либо в BNF базового языка для TTCN-3.

c) Нотация "..." резервирует место для любого произвольно выбранного контента, не показанного в явной форме.

5.5 Общие правила отображения

Отображение формата табличного представления в базовый язык TTCN-3 состоит из ряда преобразований. Соответствующее преобразование имеется для каждого синтаксического элемента в рамках каждой проформы. Кроме того, преобразования обеспечивают возможность представить любой модуль базового языка в табличной форме.

Такие преобразования подразделяются на два класса. Первый класс обеспечивает прямое конвертирование табличного элемента и структурного компонента базового языка, имеющих одинаковое значение. Второй класс обеспечивает конвертирование табличного элемента и соответствующего структурного компонента базового языка, которые не имеют значения на уровне базового языка.

Типичным примером преобразований первого класса было бы поле идентификатора. Это поле может быть прямо преобразовано из табличной формы в базовый язык и сохранять свое значение, т. е. определяя тот же самый элемент языка.

Второй класс преобразований – это обычно та или иная форма примечания или указания относительно того, как элемент языка должен быть выведен в данном формате представления. Эти элементы не имеют прямого значения в базовом языке и выражаются с использованием *WithStatement* (с оператором).

Указанные в настоящей Рекомендации синтаксис и семантика являются характерными для формата табличного представления ETSI. С целью однозначного определения в базовом языке того, какой формат представления используется, необходимо указать следующий специальный оператор визуализации в качестве первого оператора визуализации, связанного с модулем базового языка TTCN-3:

```
1: module TTCN3ModuleId    "{"
2:      ...
3: "}" with "{"
4:      display """ "presentation format" ":=" "ETSI Tabular version"
5:      MajorVersion "." MinorVersion """ ";"
6:      ...
7: "}"
```

ПРИМЕЧАНИЕ. – Все *WithStatements,* связанные с определенной проформой, должны быть сгруппированы вместе в непрерывном списке.

Поля **Группа** в проформах никогда не переводятся в *WithStatements,* а выводятся из фактической структуры группы, приведенной в описании модуля.

6 Проформы

6.1 Управление набором тестов

Управление набором тестов	
Название модуля	*TTCN3ModuleId*
Версия	*[TabFreeText]*
Дата	*[TabFreeText]*
Базовая стандартная ссылка	*[TabFreeText]*
Тестовая стандартная ссылка	*[TabFreeText]*
Ссылка PICS	*[TabFreeText]*
Ссылка PIXIT	*[TabFreeText]*
Метод(ы) тестирования	*[TabFreeText]*
Кодирование	*[TabFreeText]*
Примечания	*[TabFreeText]*

Местное определение названия	**Тип**	**Исходная величина**	**Примечания**
[VarConstOrTimerIdentifier]	*[ConstTypeOrTimer]*	*[Expression]*	*[TabFreeText]*
...	...	...	...

Поведение
ModuleControlBody

Подробные примечания	*[TabFreeText]*

Рисунок 2/Z.141 – Проформа управления набором тестов

6.1.1 Отображение

Проформа управления набором тестов переводится в три части. Первая часть состоит из полей заголовка и поля **Подробные примечания**, которые конвертируются в атрибуты визуализации в рамках *WithStatement,* связанного с общим модулем TTCN-3. Поле **Название модуля** отображается в идентификатор модуля.

Вторая часть состоит из местных постоянных, переменных и таймеров, определенных в части управления. Эти определения могут появляться в любых местах части управления базового языка, однако для проформы они отделяются от остальной части текста модуля управления и выводятся в отдельной таблице. Следует сохранять порядок определений, поскольку они могут быть взаимозависимыми. Столбец **Тип** устанавливается на ключевом слове **timer** для всех таймеров и на постоянном типе, которому предшествует ключевое слово **const,** для всех постоянных. Поля **Примечания** таблицы местных определений конвертируются в атрибуты визуализации в рамках *WithStatement,* связанного с частью управления модуля базового языка TTCN-3.

Третья часть – это часть управления модуля базового языка TTCN-3 за вычетом местных постоянных, переменных и таймеров.

```
 1: module TTCN3ModuleId "{"
 2:     control "{"
 3:  var Type VarIdentifier [":=" Expression] ";"
 4:  timer TimerIdentifier [":=" Expression] ";"
 5:  const Type ConstIdentifier ":=" ConstantExpression;
 6:         ModuleControlBody
 7:     "}" with "{"
 8:         { VarConstOrTimerCommentsAttribute }
 9:     "}"
10: "}" with "{"
11:     ModuleAttributes
12:  [EncodeAttribute;]
13: "}"
```

ПРИМЕР:

Управление последовательностью тестирования			
Название модуля	Пример1		
Версия	1.01		
Дата	19 июля 2001 г.		
Базовая стандартная ссылка	Рекомендация МСЭ-Т Q.123		
Тестовая стандартная ссылка	Рекомендация МСЭ-Т Q.123.1		
Ссылка PICS	Рекомендация МСЭ-Т Q.123.2, Приложение A		
Ссылка PIXIT	Рекомендация МСЭ-Т Q.123.2, Приложение B		
Метод(ы) тестирования	местный		
Кодирование	BER (коэффициент ошибок по битам)		
Примечания	ATS, составленный STF 133		
Местное определение названия	**Тип**	**Исходная величина**	**Примечания**
PI	пост. допуск	3.14	коэффициент
x	допуск	PI * 2	двойной PI
t1	таймер	15	15-сек. таймер
Поведение			

/* group1/ */
/* group1_1/ */
execute(test1);
execute(test2);
/* group1_2/ */
execute(test3);
execute(test4);
/* group2/ */
execute(test5);			

Подробные примечания	подробные примечания		

Отображается в:

```
  1: module Example1 {
  2:     control {
  3:         const float PI := 3.14;
  4:         var float x := PI * 2;
  5:         timer t1 := 15;
  6:
  7:         /* group1/ */
  8:             /* group1_1/ */
  9:                 execute(test1());
 10:                 execute(test2());
 11:             /* group1_2/ */
 12:                 execute(test3());
 13:                 execute(test4());
 14:             /* group2/ */
 15:                 execute(test5());
 16:     } with {
 17:         display (PI) "comments := the ratio";
 18:         display (x) "comments := double PI";
 19:         display (t1) "comments := a 15 second timer";
 20:     }
 21: } with {
 22:     display "presentation format := ETSI Tabular version 1.0";
 23:     display "module version := 1.01";
 24:     display "module date := 19 July 2001";
 25:     display "module base standards ref := ITU-T Recommendation Q.123";
 26:     display "module test standards ref := ITU-T Recommendation Q.123";
 27:     display "module pics ref := ITU-T Recommendation Q.123, Annex A";
 28:     display "module pixit ref := ITU-T Recommendation Q.123, Annex A";
 29:     display "module test method := local";
 30:     display "module comments := ATS written by STF 133";
 31:     display "module detailed comments := detailed comments";
 32:     encode "BER";
 33: }
```

6.2 Параметры набора тестов

Параметры набора тестов				
Название	**Тип**	**Исходная величина**	**Ссылка PICS/PIXIT**	**Примечания**
. *ModuleParIdentifier* .	. *ModuleParType* .	. *[ConstantExpression]* .	. *[TabFreeText]* .	. *[TabFreeText]* .
Подробные примечания	*[TabFreeText]*			

Рисунок 3/Z.141 – Проформа параметров набора тестов

6.2.1 Отображение

Все записи в проформе параметров набора тестов отображаются в *ModuleParList*s в *ModuleParameterDefs* соответствующего модуля TTCN-3. При наличии более одного *ModuleParameterDef* собираются все *ModuleParList*s и представляются в одной проформе **Параметры набора тестов**.

Поля **Ссылка PICS/PIXIT** и **Примечания** отображаются в атрибуты визуализации, классифицируемые идентификатором параметра, в рамках *WithStatement*s, связанных с включающим их *ParamDef*. Поле **Подробные примечания** отображается в атрибуты визуализации в рамках *WithStatement,* связанного с включающим его *ParamDef*.

```
 1: module TTCN3ModuleId "{"
 2: parameters "{" ModuleParList "}"
 3:  with "{"
 4:   [ModuleParPicsPixitRefAttribute ";"]
 5:   [ModuleParComments ";"]
 6:   [DetailedComments ";"]
 7: "}"
 8: "}"
```

ПРИМЕР:

Параметры последовательности тестирования				
Название	**Тип**	**Исходная величина**	**Ссылка PICS/PIXIT**	**Примечания**
CAP_1	булевский	истинная	A.1.3	вариант 1 внедрен
Tall	допуск	600,0	A.1.4	таймер всего модуля
Подробные примечания	подробные примечания			

Отображается в:

```
 1: module MyModule{
 2:  parameters { boolean CAP_1 := true, float Tall := 600.0 }
 3:  with {
 4:  display (CAP_1) "pics/pixit ref := A.1.3";
 5:  display (CAP_1) "comments := option 1 implemented";
 6:  display (Tall) "pics/pixit ref := A.1.4";
 7:  display (Tall) "comments := overall module timer";
 8:  display "detailed comments := detailed comments"
 9:  }
10: }
```

6.3 Операции импорта модуля

Операции импорта		
Исходное название	*GlobalModuleId [**рекурсивное**]*	
Исходный язык	*[LanguageSpec]*	
Группа	*[GroupReference]*	
Исходная ссылка	*[TabFreeText]*	
Кодирование	*[TabFreeText]*	
Примечания	*[TabFreeText]*	
Тип		**Название**
. *[ImportType]* .		. *ImportSpecification* .
Подробные примечания	*[TabFreeText]*	

Рисунок 4/Z.141 – Проформа операций импорта

6.3.1 Отображение

Проформа операций импорта отображается в операторе *ImportDef* в базовом языке TTCN-3. Поля **Исходное название, Исходный язык, Тип** и **Название** непосредственно используются в соответствующем операторе *ImportDef* базового языка. Поля **Исходная ссылка, Примечания** и **Подробные примечания** переводятся в атрибуты визуализации в рамках *WithStatement*, связанного с оператором *ImportDef*. Поле **Кодирование** переводится в атрибут кодирования в рамках *WithStatement*, связанного с оператором *ImportDef*

Если все определения в модуле импортируются, тогда графа *ImportType* будет пустой, а в *ImportSpecification* будет использоваться ключевое слово **all**.

```
 1: module TTCN3ModuleId "{"
 2:  ImportDef
 3:     with "{"
 4:  [ImportsSourceRefAttribute ";"]
 5:  [CommentsAttribute ";"]
 6:  [ImportsSourceDefinitionCommentsAttribute ";"]
 7:  [DetailedCommentsAttribute ";"]
 8:  [EncodeAttribute ";"]
 9:  "}"
10: "}"
```

ПРИМЕР:

Операции импорта		
Исходное название	МодульA, рекурсивный	
Исходный язык	ASN.1:1997	
Группа		
Исходная ссылка	EN 800 900 версия 2	
Кодирование	BER (коэффициент ошибок по битам)	
Примечания	объявления об импорте от ATS	
Тип	**Название**	**Примечания**
постоянная	все, за исключением метасинтаксических переменных (foobar)	
Тип	MyType	метасинтаксические переменные(foobar)
Группа	AtoU_CTR	
Подробные примечания	подробные примечания	

Отображается в:

```
 1: module MyModule {
 2:  import from ModuleA recursive language "ASN.1997" {
 3:  const all except foobar;
 4:  type MyType;
 5:  Group AtoU_CTR;
 6:  } with {
 7:  display "imports source ref := EN 800 900 version 2";
 8:  display "comments := importing declarations from ATS";
 9:  display "detailed comments := detailed comments";
10:  encode "BER";
11:  }
12: }
```

6.4 Простые типы

Простые типы			
Группа	*[GroupReference]*		
Название	**Определение**	**Кодирование**	**Примечания**
. *SubTypeIdentifier* .	. *Type [ArrayDef]* *[SubTypeSpec]* .	. *[TabFreeText]* .	. *[TabFreeText]* .
Подробные примечания	*[TabFreeText]*		

Рисунок 5/Z.141 – Проформа простых типов

6.4.1 Отображение

Проформа простых типов отображается в серии операторов определений простого типа на том же уровне групп. Все определения простого типа являются определениями типа *SubTypeDef*.

Поле **Подробные примечания** отображается в атрибут визуализации в рамках *WithStatement,* связанного с включающими его группой или модулем. Поля **Кодирование** и **Примечания** отображаются в атрибуты кодирования и визуализации, соответственно, в рамках *WithStatement,* связанного с соответствующим определением простого типа.

```
 1: module TTCN3ModuleId "{"
 2:  type Type SubTypeIdentifier [ArrayDef] [SubTypeSpec] with "{"
 3:  [EncodeAttribute ";"]
 4:  [CommentsAttribute ";"]
 5: "}" with "{"
 6:  [SimpleTypesDetailedCommentsAttribute ";"]
 7: "}"
```

ПРИМЕР:

Простые типы			
Группа	SimpleTypes/		
Название	**Определение**	**Кодирование**	**Примечания**
EQ_NUMBER	целое (1 .. 20)	PER	Бог его знает
Detailed Comments	подробные примечания		

Отображается в:

```
 1: module MyModule {
 2:  group SimpleTypes {
 3:  type integer EQ_NUMBER (1..20) with {
 4:  encode "PER";
 5:  display "comments := God knows";
 6:  }
 7:  } with {
 8:  display "simple types detailed comments := detailed comments";
 9:  }
10: }
```

6.5 Структурный тип

Структурный тип			
Название	*StructTypeIdentifier[StructDefFormalParList]*		
Группа	*[GroupReference]*		
Структура	*StructureType*		
Кодирование	*[TabFreeText]*		
Примечания	*[TabFreeText]*		
Название поля	**Тип поля**	**Кодирование поля**	**Примечания**
.	.	.	.
FieldIdentifier	*Type [ArrayDef]*	*[TabFreeText]*	*[TabFreeText]*
.	*[SubTypeSpec]*	.	.
.	*[OptionalKeyword]*	.	.
.	.	.	.
Подробные примечания	*[TabFreeText]*		

Рисунок 6/Z.141 – Проформа структурного типа

6.5.1 Отображение

Проформа структурного типа отображается в операторе определения структурного типа в TTCN-3. Эта проформа будет использоваться следующими типами: *RecordDef*, *UnionDef* и *SetDef*.

Поля **Примечания** и **Подробные примечания** отображаются в атрибуты визуализации в рамках соответствующего *WithStatement*, а поле **Кодирование** отображается в атрибуте кодирования в соответствующем *WithStatement*. Поля **Примечания** и **Кодирование поля** каждого элемента поля отображаются в атрибуты визуализации и кодирования, соответственно, классифицируемые *FieldIdentifier* в соответствующем *WithStatement*.

```
 1: module TTCN3ModuleId "{"
 2:   type StructureType StructTypeIdentifier [StructDefFormalParList] "{"
 3:   {Type FieldIdentifier [ArrayDef] [SubtypeSpec] [OptionalKeyword]}
 4:   "}" with "{"
 5:   [EncodeAttribute ";"]
 6:   [CommentsAttribute ";"]
 7:   {FieldCommentsAttribute ";"}
 8:   {FieldEncodeAttribute ";"}
 9:   [DetailedCommentsAttribute ";"]
10:   "}"
11: "}"
```

ПРИМЕР:

Тип структуры			
Название	routing_label(SLSel_Type)		
Группа			
Структура	запись		
Кодирование	BER (коэффициент ошибок по битам)		
Примечания	заголовок для информации о маршрутизации		
Название элемента	**Тип определения**	**Кодирование поля**	**Примечания**
DestPC	BIT_14		код пункта назначения
OrigPC	BIT_14		код пункта происхождения
SLSel	SLSel_Type	PER	выбор линии передачи сигналов
Подробные примечания	заменяет предыдущие определения		

Отображается в:

```
 1: module MyModule {
 2:  type record routing_label(SLSel_Type) {
 3:  BIT_14 DestPC,
 4:  BIT_14 OrigPC,
 5:  SLSel_Type SLSel
 6:  } with {
 7:  encode "BER";
 8:  display "comments := header for routing info";
 9:  display (DestPC) "comments := destination point code";
10:  display (OrigPC) "comments := origination point code";
11:  display (SLSel) "comments := signalling link selection";
12:  encode (SLSel) "PER";
13:  display "detailed comments := overrides previous definition";
14:  }
15: }
```

6.6 Типы SequenceOf (последовательность из)

Типы SequenceOf					
Группа	*[GroupReference]*				
Название	**Тип**	**Вид**	**Длина**	**Кодирование**	**Примечания**
. *StructTypeIdentifier* .	. *Type [SubTypeSpec]* .	. *RecordOrSet* .	. *[StringLength]* .	. *[TabFreeText]* .	. *[TabFreeText]* .
Подробные примечания	*[TabFreeText]*				

Рисунок 7/Z.141 – Проформа типов SequenceOf

6.6.1 Отображение

Проформа типов SequenceOf отображается в серию операторов определений типа SequenceOf на том же уровне группы. Эта проформа используется для определений типа *RecordOfDef* и *SetOfDef*.

Поле **Подробные примечания** отображается в атрибут визуализации в рамках *WithStatement,* связанного с включающими его группой или модулем. Поля **Кодирование** и **Примечания** отображаются в атрибуты кодирования и визуализации, соответственно, в рамках *WithStatement*, связанного с соответствующим определением типа SequenceOf.

```
 1: module TTCN3ModuleId "{"
 2:  type record of [StringLength] Type StructTypeIdentifier [SubTypeSpec]
 3:  with {
 4:  [EncodeAttribute ";"]
 5:  [CommentsAttribute ";"]
 6:  }
 7:  type set of [StringLength] Type StructTypeIdentifier [SubTypeSpec]
 8:  with {
 9:  [EncodeAttribute ";"]
10:  [CommentsAttribute ";"]
11:  }
12: } with {
13:  [SequenceOfTypesDetailedCommentsAttribute ";"]
14: }
```

ПРИМЕР:

Типы SequenceOf					
Группа	SequenceOfTypes/				
Название	**Тип**	**Вид**	**Длина**	**Кодирование**	**Примечания**
RecordOfIntegers	целое(1..10)	запись	10	BER	десять целых
SetOfBooleans	булевский	набор	3	PER	три булевских
Подробные примечания	пример типов sequenceOf				

Отображается в:

```
 1: module MyModule {
 2:  group SequenceOfTypes {
 3:  type record of length(10) integer RecordOfIntegers(1..10) with {
 4:  encode "BER";
 5:  display "comments := ten integers";
 6:  }
 7:  type set of length(3) boolean SetOfBooleans with {
 8:  encode "PER";
 9:  display "comments := three booleans";
10:  }
11:  } with {
12:  display "sequenceof types detailed comments := example sequenceof types";
13:  }
14: }
```

6.7 Перечислимый тип

Перечислимый тип			
Название	*EnumTypeIdentifier*		
Группа	*[GroupReference]*		
Кодирование	*[TabFreeText]*		
Примечания	*[TabFreeText]*		
Название перечисления		**Значение перечисления**	**Примечания**
. *EnumerationIdentifier* .		. *[Number]* .	. *[TabFreeText]* .
Подробные примечания	*[TabFreeText]*		

Рисунок 8/Z.141 – Проформа перечислимого типа

6.7.1 Отображение

Проформа перечислимого типа отображается в операторе определения перечислимого типа в базовом языке TTCN-3. Поля **Примечания** и **Дополнительные примечания** отображаются в атрибуты визуализации в соответствующем *WithStatement*, а поле **Кодирование** отображается в атрибут кодирования в рамках соответствующего *WithStatement*. Поля **Примечания** в каждом перечне отображаются в атрибуты визуализации, классифицируемые *EnumerationIdentifier* в соответствующем *WithStatement*.

```
 1: module TTCN3ModuleId "{"
 2:  type enumerated EnumTypeIdentifier "{"
 3:  EnumerationIdentifier ["(" Number ")"]
 4:  {"," EnumerationIdentifier ["(" Number ")"]}
 5:  } with {
 6:  [EncodeAttribute ";"]
 7:  [CommentsAttribute ";"]
 8:  {NamedValueCommentsAttribute ";"}
 9:  [DetailedCommentsAttribute ";"]
10:  }
11: }
```

ПРИМЕР:

Перечислимый тип		
Название	дни недели	
Группа		
Кодирование	BER (коэффициент ошибок по битам)	
Примечания	названия дней недели	
Название перечисления	**Значение перечисления**	**Примечания**
Понедельник	1	
Вторник	2	
Среда	3	здесь половина дня
Четверг	4	
Пятница	5	TGIF
Суббота	6	
Воскресенье	7	
Подробные примечания	скорее бы пятница	

Отображается в:

```
 1: module MyModule {
 2:  type enumerated Weekdays {
 3:  Monday(1), Tuesday(2), Wednesday(3), Thursday(4), Friday(5),
 4:  Saturday(6), Sunday(7)
 5:  } with {
 6:  encode "BER";
 7:  display "comments := days of the week";
 8:  display (Wednesday) "comments := half way there";
 9:  display (Friday) "comments := TGIF";
10:  display "detailed comments := wish it were Friday";
11:  }
12: }
```

6.8 Типы портов

Тип порта		
Название	*PortTypeIdentifier*	
Группа	*[GroupReference]*	
Модель связи	*PortModelType*	
Примечания	*[TabFreeText]*	
Тип/подпись	**Направление**	**Примечания**
. *TypeOrSignature* .	. *InOutOrInout* .	. *[TabFreeText]* .
Подробные примечания	*[TabFreeText]*	

Рисунок 9/Z.141 – Проформа типа порта

6.8.1 Отображение

Проформа типа порта отображается в определении типа порта в базовом языке TTCN-3. Поля **Примечания** и **Подробные примечания** отображаются в атрибуты визуализации в соответствующем *WithStatement*. Поля **Примечания** таблицы типов и подписей отображаются в атрибуты визуализации в соответствующем *WithStatement,* классифицируемом идентификатором типа или подписи. Для каждого типа или подписи всегда будет одна строка.

Поле **Тип/подпись** установлено на ключевое слово **all**, если все типы или все подписи для процедуры, определенные в этом модуле, могут обойти данный порт связи.

```
 1: module TTCN3ModuleId "{"
 2:  type port PortTypeIdentifier PortModelType "{"
 3:  PortTypeDef
 4:  "}" with "{"
 5:  [CommentsAttribute ";"]
 6:  {TypeOrSignatureCommentsAttribute ";"}
 7:  [DetailedCommentsAttribute ";"]
 8:  "}"
 9: "}"
```

ПРИМЕР:

Тип порта		
Название	MyPortType	
Группа		
Модель связи	сообщение	
Примечания	пример типа порта	
Тип/подпись	**Направление**	**Примечания**
MsgType1	в	первое примечание
MsgType2	в	второе примечание
MsgType3	из	
Подробные примечания	подробное примечание	

Отображается в:

```
 1: module MyModule {
 2:  type port MyPortType message {
 3:  in MsgType1;
 4:  in MsgType2;
 5:  out MsgType3;
 6:  } with {
 7:  display "comments := example port type";
 8:  display (MsgType1) "comments := first comment";
 9:  display (MsgType2) "comments := second comment";
10:  display "detailed comments := detailed comment";
11:  }
12: }
```

6.9 Типы компонентов

Тип компонента			
Название	*ComponentTypeIdentifier*		
Группа	*[GroupReference]*		
Примечания	*[TabFreeText]*		
Местное определение названия	**Тип**	**Исходная величина**	**Примечания**
. *VarConstOrTimerIdentifier* .	. *TypeOrTimer [ArrayDef]* .	. *[ConstantExpression | Expression]* .	. *[TabFreeText]* .
Название порта	**Тип порта**		**Примечания**
. *PortIdentifier* .	. *PortType[ArrayDef]* .		. *[TabFreeText]* .
Подробные примечания	*[TabFreeText]*		

Рисунок 10/Z.141 – Проформа типа компонента

6.9.1 Отображение

Проформа типа компонента отображается в определение типа компонента в базовом языке TTCN-3. Проформа переводится в три части.

Первая часть состоит из заголовка полей **Примечания** и **Подробные примечания,** которые конвертируются в атрибуты визуализации в рамках *WithStatement,* связанного с определением типа компонента.

Вторая часть состоит из местных постоянных, переменных и таймеров, определенных в типе компонента. Эти определения могут появляться в любых местах определения типа компонента базового языка, однако для проформы они отделяются от вариантов портов и выводятся в отдельной таблице. Следует сохранять порядок определений, поскольку они могут быть взаимозависимыми. Столбец **Тип** устанавливается на ключевом слове **`timer`** для всех таймеров и на постоянном типе, которому предшествует ключевое слово **`const,`** для всех постоянных. Для каждой постоянной, переменной или таймера всегда будет одна строка. Столбец **Примечания** этой таблицы трансформируется в атрибуты визуализации, классифицируемые местным идентификатором определения в рамках *WithStatement,* связанного с определением типа компонента.

Третья часть состоит из вариантов портов, определенных в типе компонента. К типу порта прилагаются любые определения массива данных. Для каждого варианта порта всегда имеется одна строка. Столбец **Примечания** данной таблицы конвертируется в атрибуты визуализации, классифицируемые *PortIdentifier* в рамках *WithStatement*, связанного с определением типа компонента.

```
 1: module TTCN3ModuleId "{"
 2:  type component ComponentTypeIdentifier "{"
 3:  var Type VarIdentifier [":=" Expression] ";"
 4:  timer TimerIdentifier [":=" Expression] ";"
 5:  const Type ConstIdentifier ":=" ConstantExpression ";"
 6:  PortList
 7:  "}" with "{"
 8:  [CommentsAttribute ";"]
 9:  {PortCommentsAttribute ";"}
10:  [DetailedCommentsAttribute ";"]
11:  "}"
12: "}"
```

ПРИМЕР:

Тип компонента			
Название	MyComponentType		
Группа			
Примечания	пример типа компонента		
Местное определение названия	**Тип**	**Исходная величина**	**Примечания**
PI	пост. допуск	3.14	коэффициент
x	допуск	PI * 2	двойной PI
t1	синхронизатор	15 мин.	15-сек. синхронизатор
Название порта	**Тип порта**		**Примечания**
PCO1	MyMessagePortType		первое примечание
PCO2	MyProcedurePortType		второе примечание
Подробные примечания	подробные примечания		

Отображается в:

```
 1: module MyModule {
 2:  type component MyComponentType {
 3:  const float PI := 3.14;
 4:  var float x := PI * 2;
 5:  timer t1 := 15;
 6:  port MyMessagePortType PCO1;
 7:  port MyProcedurePortType PCO2;
 8:  } with {
 9:  display "comments := an example component type";
10:  display (PI) "comments := the ratio";
11:  display (x) "comments := double PI";
12:  display (t1) "comments := a 15 second timer";
13:  display (PCO1) "comments := first comment";
14:  display (PCO2) "comments := second comment";
15:  display "detailed comments := detailed comments";
16:  }
17: }
```

6.10 Постоянные

Постоянные			
Группа	*[GroupReference]*		
Название	**Тип**	**Значение**	**Примечания**
. *ConstIdentifier | ExtConstIdentifier* .	. *Type [ArrayDef]* .	. *ConstantExpression* | **external**	. *[TabFreeText]* .
Подробные примечания	*[TabFreeText]*		

Рисунок 11/Z.141 – Проформа постоянных

6.10.1 Отображение

Проформа постоянных отображается в серию операторов определений постоянных и внешних постоянных на том же уровне группы. Поле **Подробные примечания** отображается в атрибут визуализации в рамках *WithStatement,* связанного с включающими его группой или модулем. Поля **Примечания** отображаются в атрибуты визуализации в рамках *WithStatement,* связанного с соответствующим определением постоянной. Поле **Значение** для внешней постоянной установлено на ключевое слово **external.**

```
 1: module TTCN3ModuleId "{"
 2:  const Type ConstIdentifier[ArrayDef] ":=" ConstantExpression with "{"
 3:  [CommentsAttribute ";"]
 4:  "}"
 5:  external const Type ConstIdentifier with "{"
 6:  [CommentsAttribute ";"]
 7:  "}"
 8: "}" with "{"
 9:  [ConstantsDetailedCommentsAttribute ";"]
10: "}"
```

ПРИМЕР:

Постоянные			
Группа	Constants1		
Название	**Тип**	**Значение**	**Примечания**
TOTO	целое	внешнее	определено в другом месте
SEL2	булевский	(5 + TOTO) < 10	достигнут предел TOTO
T1	целое[1..3]	{1,3,2}	
Подробные примечания	подробные примечания		

Отображается в:

```
 1: module MyModule {
 2:  group Constants1 {
 3:  external const integer TOTO with {
 4:  display "comments := defined somewhere else";
 5:  }
 6:  const boolean SEL2 := (5 + TOTO) < 10 with {
 7:  display "comments := TOTO limit reached";
 8:  }
 9:  const integer T1[1..3] := {1,3,2};
10:  } with {
11:  display "detailed comments := detailed comments";
12:  }
13: }
```

6.11 Определение подписи

Определение подписи		
Название	*SignatureIdentifier*([*SignatureFormalParList*])	
Группа	*[GroupReference]*	
Тип возврата функции	*[Type]* | **noblock**	
Примечания	*[TabFreeText]*	
Тип исключения		**Примечания**
. *[ExceptionType]* .		. *[TabFreeText]* .
Подробные примечания	*[TabFreeText]*	

Рисунок 12/Z.141 – Проформа определения подписи

6.11.1 Отображение

Проформа определения подписи отображается в определение подписи в базовом языке TTCN-3. Поля **Примечания** и **Подробные примечания** отображаются в атрибуты визуализации в рамках соответствующего *WithStatement*. Поля **Примечания** таблицы исключений отображаются в атрибуты визуализации, классифицируемые типом исключения в соответствующем *WithStatement*. В процедурах без блокировки в качестве типа возврата функции указывается ключевое слово **noblock**.

```
 1: module TTCN3ModuleId "{"
 2:  signature SignatureIdentifier "(" [SignatureFormalParList] ")"
 3:  [return Type | noblock]
 4:  [exception "(" ExceptionTypeList ")"]
 5:  with "{"
 6:  [CommentsAttribute ";"]
 7:  [ExceptionCommentsAttribute ";"]
 8:  [DetailedCommentsAttribute ";"]
 9:  "}"
10: "}"
```

ПРИМЕР:

Определение подписи		
Название	прочитать(integer fields, inout charstring buf, integer nbyte)	
Группа		
Тип возврата функции	целое	
Примечания	прочитать из файла	
Тип исключения		**Примечания**
целое		код ошибки
MyException		определяется пользователем
Подробные примечания	необходимо: unistd.h	

Отображается в:

```
 1: module MyModule {
 2:  signature read_syscall(in integer fields,
 3:  inout charstring buf,
 4:  in integer nbyte)
 5:  return integer
 6:  exception (integer)
 7:  with {
 8:  display "comments := reads from a file";
 9:  display (integer) "comments := error code of system call";
10:  display "detailed comments := required: unistd.h";
11:  }
12: }
```

6.12 Простые шаблоны

Простые шаблоны					
Группа	*[GroupReference]*				
Название	**Тип**	**Производное**	**Значение**	**Кодирование**	**Примечания**
. *Template Identifier* .	. *BaseTemplate* .	. *[DerivedDef]* .	. *TemplateBody* .	. *[TabFreeText]* .	. *[TabFreeText]* .
Подробные примечания	*[TabFreeText]*				

Рисунок 13/Z.141 – Проформа простых шаблонов

6.12.1 Отображение

Проформа простых шаблонов отображается в серию операторов определений простых шаблонов на том же уровне группы. Все определения простых шаблонов являются определениями шаблонов, которые в качестве *TemplateBody* имеют *SimpleSpec* или *ArrayValueOrAttrib*. Соответствующие типы определены в проформах простых типов, типов SequenceOf и перечислимых типов.

Поле **Подробные примечания** отображается в атрибут визуализации в рамках *WithStatement*, связанного с включающими его группой или модулем. Поля **Примечания** и **Кодирование** отображаются в атрибуты визуализации и кодирования, классифицируемые *TemplateIdentifier* в рамках *WithStatement*, связанного с соответствующим оператором определения простого шаблона.

```
1: module TTCN3ModuleId "{"
2:  template BaseTemplate[DerivedDef] := TemplateBody with "{"
3:  [EncodeAttribute ";"]
4:  [CommentsAttribute ";"]
5:  "}"
6: "}" with "{"
7:  [SimpleTemplatesDetailedCommentsAttribute ";"]
8: "}"
```

ПРИМЕР:

Простые шаблоны					
Группа	SimpleTemplates1				
Название	**Тип**	**Производное**	**Значение**	**Кодирование**	**Примечания**
MyTemplate1	MyType1		3	BER	метасинтаксические переменные (foobar)
MyTemplate11 (индекс целого)	MyType1	MyTemplate1	3*индекс	PER	имеющийся индекс
Подробные примечания	пример				

Отображается в:

```
 1: module MyModule {
 2:  group SimpleTemplates {
 3:  template MyType1 MyTemplate1 with {
 4:  encode "BER";
 5:  display "comments := foobar";
 6:  }
 7:  template MyType1 MyTemplate11(integer index)
 8:  modifies MyTemplate1 := 3 * index
 9:  with {
10:  encode "PER";
11:  display "comments := the current index";
12:  }
13:  } with {
14:  display "simple templates detailed comments := an example";
15:  }
16: }
```

6.13 Структурный шаблон

Структурный шаблон			
Название	*TemplateIdentifier[(TemplateFormalParList)]*		
Группа	*[GroupReference]*		
Тип/подпись	*TypeIdentifier | SignatureIdentifier*		
Производное от	*[TemplateRef]*		
Кодирование	*[TabFreeText]*		
Примечания	*[TabFreeText]*		
Название элемента	**Значение элемента**	**Кодирование элемента**	**Примечания**
. *FieldReference* .	. *FieldValueOrAttrib* .	. *[TabFreeText]* .	. *[TabFreeText]* .
Подробные примечания	*[TabFreeText]*		

Рисунок 14/Z.141 – Проформа структурного шаблона

6.13.1 Отображение

Проформа структурного шаблона отображается в оператор определения структурного шаблона TTCN-3. Определения структурированного шаблона являются всеми определениями шаблона, которые имеют *FieldSpecList* в качестве основной части шаблона. Соответствующие типы определены в проформе структурного типа.

Поля **Примечания** и **Основные примечания** отображаются в атрибуты визуализации в рамках *WithStatement,* связанного с определением структурного шаблона. Поле **Кодирование** отображается в атрибут кодирования в рамках *WithStatement,* связанного с определением структурного шаблона.

Поля **Примечания** таблицы элементов отображаются в атрибуты визуализации, классифицируемые ссылкой на поле в рамках *WithStatement,* связанного с определением структурного шаблона. Поля **Кодирования элементов** отображаются в атрибуты кодирования, классифицируемые ссылкой на поле в рамках *WithStatement,* связанного с определением структурного шаблона.

```
module TTCN3ModuleId "{"
 template BaseTemplate [DerivedDef] ":=" TemplateBody with "{"
 [EncodeAttribute ";"]
 [CommentsAttribute ";"]
 [FieldEncodeAttribute ";"]
 [FieldCommentsAttribute ";"]
 [DetailedCommentsAttribute ";"]
 "}"
"}"
```

ПРИМЕР:

Структурный шаблон			
Название	MyStructuredTemplate11(целое пункт1, булевское пункт2)		
Группа			
Тип/подпись	MyStructuredType		
Производное от	MyStructuredTemplate1		
Кодирование	BER (коэффициент ошибок по битам)		
Примечания	пример структурного шаблона		
Название элемента	**Значение элемента**	**Кодирование элемента**	**Примечания**
field1	13		первое поле
field2	para2	PER	второе поле
field3	para1		третье поле
Подробные примечания	подробные примечания		

Отображается в:

```
module MyModule {
 template MyStructuredType MyStructuredTemplate11(integer para1,
 boolean para2)
 modifies MyStructuredTemplate1 := {
 field1 := 13,
 field2 := para2,
 field3 := para1
 } with {
 encode "BER";
 display "comments := example structured template";
 display (field1) "comments := first field";
 encode (field2) "PER";
 display (field2) "comments := second field";
 display (field3) "comments := third field";
 display "detailed comments := detailed comments";
 }
}
```

6.14 Функция

Функция			
Название	*FunctionIdentifier([FunctionFormalParList])*		
Группа	*[GroupReference]*		
Работает на	*[ComponentType]*		
Тип возврата функции	*[Type]*		
Примечания	*[TabFreeText]*		
Местное определение названия	**Тип**	**Исходная величина**	**Примечания**
. *VarConstOrTimerIdentifier* .	. *TypeOrTimer* .	. *[Expression | ConstantExpression]*	. *[TabFreeText]* .
Поведение			
. *FunctionStatement* | **внешнее** .			
Подробные примечания	*[TabFreeText]*		

Рисунок 15/Z.141 – Проформа функции

6.14.1 Отображение

Проформа функции отображается в оператор определения функции TTCN-3 или внешнее определение функции. Она переводится в три части.

Первая часть состоит из заголовков полей. Поля **Примечания** и **Подробные примечания** отображаются в атрибуты визуализации в рамках *WithStatement,* связанного с определением функции.

Вторая часть состоит из местных постоянных, переменных и таймеров, определенных в определении функции. Эти определения могут появляться в любых местах основной части функции базового языка, однако для проформы они отделяются от остальной основной части функции и выводятся в отдельную таблицу. Следует сохранять порядок определений, поскольку они могут быть взаимозависимыми. Столбец **Тип** устанавливается на ключевом слове **`timer`** для всех таймеров и на постоянном типе, которому предшествует ключевое слово **`const`,** для всех постоянных. Поля **Примечания** конвертируются в атрибуты визуализации, классифицируемые местным идентификатором в рамках *WithStatement,* связанного с определением функции.

Третья часть состоит из основной части функции базового языка TTCN-3 за вычетом местных постоянных, переменных и таймеров.

Для внешней функции поведение содержит только ключевое слово **`external`**.

```
 1: module TTCN3ModuleId "{"
 2:  function FunctionIdentifier "(" [FunctionFormalParList] ")"
 3:  [runs on ComponentType]
 4:  [return Type] "{"
 5:  var Type VarIdentifier [":=" Expression] ";"
 6:  timer TimerIdentifier [":=" Expression]";"
 7:  const Type ConstIdentifier ":=" ConstantExpression ";"
 8:  {FunctionStatement}
 9:  "}" with "{"
10:  [CommentsAttribute ";"]
11:  [VarConstOrTimerCommentsAttribute ";"]
12:  [DetailedCommentsAttribute ";"]
13:  "}"
14: "}"
```

ПРИМЕР:

Функция			
Название	MyFunction(целое para1)		
Группа			
Работает на	MyComponentType		
Тип возврата функции	булевский		
Примечания	пример определения функции		
Местное определение названия	**Тип**	**Исходная величина**	**Примечания**
MyLocalVar	булевский	ошибочная	местная переменная
MyLocalConst	пост. допуск	60	местная постоянная
MyLocalTimer	таймер	15 * MyLocalConst	местный таймер
Поведение			

if (para1 == 21) {
 MyLocalVar := true;
}
if (MyLocalVar) {
 MyLocalTimer.start;
 MyLocalTimer.timeout;
}
return (MyLocalVar);			

Подробные примечания	подробные примечания		

Отображается в:

```
 1: module MyModule {
 2:  function MyFunction(in integer para1)
 3:  runs on MyComponentType
 4:  return boolean {
 5:  var boolean MyLocalVar := false;
 6:  const float MyLocalConst := 60;
 7:  timer MyLocalTimer := 15 * MyLocalConst;
 8:
 9:  if (para1 == 21) {
10:  MyLocalVar := true;
11:  }
12:  if (MyLocalVar) {
13:  MyLocalTimer.start;
14:  MyLocalTimer.timeout;
15:  }
16:  return (MyLocalVar);
17:  } with {
18:  display "comments := example function definition";
19:  display (MyLocalVar) "comments := local variable";
20:  display (MyLocalConst) "comments := local constant";
21:  display (MyLocalTimer) "comments := local timer";
22:  display "detailed comments := detailed comments";
23:  }
24: }
```

6.15 Altstep

Altstep			
Название	*AltstepIdentifier([AltstepFormalParList])*		
Группа	*[GroupReference]*		
Цель	*[TabFreeText]*		
Работает на	*[ComponentType]*		
Примечания	*[TabFreeText]*		
Местное определение названия	**Тип**	**Исходная величина**	**Примечания**
. *VarConstOrTimerIdentifier* .	. *TypeOrTimer [ArrayDef]* .	. *[Expression | ConstantExpression]* .	. *[TabFreeText]* .
Поведение			
. *AltGuardList* .			
Подробные примечания	*[TabFreeText]*		

Рисунок 16/Z.141 – Проформа Altstep

6.15.1 Отображение

Проформа Altstep отображается в оператор определения altstep в TTCN-3. Она переводится в три части.

Первая часть состоит из заголовков полей. Поля **Цель**, **Примечания** и **Подробные примечания** отображаются в атрибуты визуализации в рамках *WithStatement,* связанного с определением altstep.

Вторая часть состоит из местных постоянных, переменных и таймеров, определенных в определении altstep. Эти определения могут появляться в любых местах основной части altstep базового языка, однако для проформы они отделяются от остальной основной части altstep и выводятся в отдельную таблицу. Следует сохранять порядок определений, поскольку они могут быть взаимозависимыми. Столбец **Тип** устанавливается на ключевом слове **`timer`** для всех таймеров и на постоянном типе, которому предшествует ключевое слово **`const`,** для всех постоянных. Поля **Примечания** конвертируются в атрибуты визуализации, классифицируемые местным идентификатором в рамках *WithStatement,* связанного с определением altstep.

Третья часть состоит из *AltGuardList* для altstep базового языка TTCN-3.

```
 1: module TTCN3ModuleId "{"
 2:  teststep AltstepIdentifier "(" [AltstepFormalParList] ")"
 3:  [runs on ComponentType] "{"
 4:  AltGuardList
 5:  "}" with "{"
 6:  [PurposeAttribute ";"]
 7:  [CommentsAttribute ";"]
 8:  [VarConstOrTimerCommentsAttribute ";"]
 9:  [DetailedCommentsAttribute ";"]
10:  "}"
11: "}"
```

ПРИМЕР:

Altstep			
Название	MyAltstep(целое para1)		
Группа			
Работает на	MyComponentType		
Цель	сделать что-либо		
Примечания	пример определения altstep		
Местное определение названия	**Тип**	**Исходная величина**	**Примечания**
MyLocalVar	булевский	ошибочная	местная переменная
MyLocalConst	пост. допуск	60	местная постоянная
MyLocalTimer	таймер	15 * MyLocalConst	местный таймер
Поведение			

[] PCO1.receive(MyTemplate(para1, CompVar) {
verdict.set(inconc);
}
[] PCO2.receive {
repeat;
}
[] CompTimer.timeout {
verdict.set(fail);
stop;
}			

Подробные примечания	подробные примечания		

Отображается в:

```
 1: module MyModule {
 2:  altstep MyTeststep(integer para1) runs on MyComponentType {
 3:  var boolean MyLocalVar := false;
 4:  const float MyLocalConst := 60;
 5:  timer MyLocalTimer := 15 * MyLocalConst;
 6:
 7:  [] PCO1.receive(MyTemplate(para1, CompVar)) {
 8:  verdict.set(inconc);
 9:  }
10:  [] PCO2.receive {
11:  repeat;
12:  }
13:  [] CompTimer.timeout {
14:  verdict.set(fail);
15:  stop;
16:  }
17:  } with {
18:  display "purpose := to do something";
19:  display "comments := example altstep definition";
20:  display "detailed comments := detailed comments";
21:  }
22: }
```

6.16 Набор тестовых данных

Набор тестовых данных			
Название	*TestcaseIdentifier([TestcaseFormalParList])*		
Группа	*[GroupReference]*		
Цель	*[TabFreeText]*		
Интерфейс системы	*[ComponentType]*		
Тип МТС	*ComponentType*		
Примечания	*[TabFreeText]*		
Местное определение названия	**Тип**	**Исходная величина**	**Примечания**
. *VarConstOrTimerIdentifier* .	. *TypeOrTimer* .	. *[Expression | ConstantExpression]*	. *[TabFreeText]* .
Поведение			
. *FunctionStatement* .			
Подробные примечания	*[TabFreeText]*		

Рисунок 17/Z.141 – Проформа набора тестовых данных

6.16.1 Отображение

Проформа набора тестовых данных отображена в оператор определения набора тестовых данных TTCN-3. Она переводится в три части.

Первая часть состоит из заголовков полей. Поля **Цель**, **Примечания** и **Подробные примечания** отображаются в атрибуты визуализации в рамках *WithStatement,* связанного с определением набора тестовых данных.

Вторая часть состоит из местных постоянных, переменных и таймеров, определенных в определении наборов тестовых данных. Эти определения могут появляться в любых местах основной части набора тестовых данных базового языка, однако для проформы они отделяются от остальной основной части набора тестовых данных и выводятся в отдельную таблицу. Следует сохранять порядок определений, поскольку они могут быть взаимозависимыми. Столбец **Тип** устанавливается на ключевом слове **`timer`** для всех таймеров и на постоянном типе, которому предшествует ключевое слово **`const`** для всех постоянных. Поля **Примечания** конвертируются в атрибуты визуализации, классифицируемые местным идентификатором в рамках *WithStatement,* связанного с определением набора тестовых данных.

Третья часть состоит из основной части набора тестовых данных базового языка TTCN-3 за вычетом местных постоянных, переменных и таймеров.

```
 1: module TTCN3ModuleId "{"
 2:  testcase TestcaseIdentifier [TestcaseFormalParList]
 3:  [runs on ComponentType]
 4:  [system ComponentType] "{"
 5:  var Type VarIdentifier [":=" Expression] ";"
 6:  timer TimerIdentifier [":=" Expression] ";"
 7:  const Type ConstIdentifier ":=" ConstantExpression;
 8:  {FunctionStatement}
 9:  "}" with "{"
10:  [CommentsAttribute ";"]
11:  [PurposeAttribute ";"]
12:  [VarConstOrTimerCommentsAttribute ";"]
13:  [DetailedCommentsAttribute ";"]
14:  "}"
15: "}"
```

ПРИМЕР:

Набор тестовых данных			
Название	MyTestcase(целое para1)		
Группа			
Цель	сделать что-либо полезное		
Интерфейс системы	MyComponentType		
Тип МТС	MyComponentType		
Примечания	пример определения набора тестовых данных		
Местное определение названия	**Тип**	**Исходная величина**	**Примечания**
MyLocalVar	булевский	ошибочная	местная переменная
MyLocalConst	пост. допуск	60	местная постоянная
MyLocalTimer	таймер	15 * MyLocalConst	местный таймер

Поведение

```
default.activate { [expand] OtherwiseFail(); }; /* Default activation */
ISAP1.send(ICONreq {}); /* Inline template definition */
alt {
    [] MSAP2.receive(Medium_Connection_Request()) { /* use of a template */
        MSAP2.send(MDATreq Medium_Connection_Confirmation());
        alt {
            [] ISAP1.receive(ICONconf {} ); {
                ISAP1.send(Data_Request(TestSuitePar) );
                alt {
                    [] MSAP2.receive(Medium_Data_Transfer()) {
                        MSAP2.send(MDATreq cmi_synch1());
                        ISAP1.send(IDISreq {});
                    }
                    [] ISAP1.receive(IDISind {}) {
                        verdict.set(inconclusive);
                        stop();
                    }
                }
            }
            [] MSAP2.receive(MDATind_Connection_Request()) {
                verdict.set(inconclusive);
                stop();
            }
            [] ISAP1.receive(IDISind {}) {
                verdict.set(inconclusive);
                stop();
            }
        }
    }
    [] ISAP1.receive(IDISind {}) {
        verdict.set(inconclusive);
        stop();
    }
}
```

Подробные примечания	подробные примечания

Отображается в:

```
 1: module MyModule {
 2:     testcase MyTestcase(in integer para1)
 3:         runs on MyComponentType
 4:         system MyComponentType {
 5:         var boolean MyLocalVar := false;
 6:         const float MyLocalConst := 60;
 7:         timer MyLocalTimer := 15 * MyLocalConst;
 8:         var default MyDefault := activate(OtherwiseFail());
 9:
10:         ISAP1.send(ICONreq:{}); /* Inline template definition */
11:         alt {
12:             /* use of a template */
13:             [] MSAP2.receive(Medium_Connection_Request()) {
14:     MSAP2.send(MDATreq:Medium_Connection_Confirmation());
15:                 alt {
16:                     [] ISAP1.receive(ICONconf:{}) {
17:                         ISAP1.send(Data_Request(TestSuitePar));
18:                         alt {
19:                             [] MSAP2.receive(Medium_Data_Transfer()) {
20:                                 MSAP2.send(MDATreq:cmi_synch1());
21:                                 ISAP1.send(IDISreq:{});
22:                             }
23:                             [] ISAP1.receive(IDISind:{}) {
24:                                 verdict.set(inconc);
25:                                 stop;
26:                             }
27:                         }
28:                     }
29:                     [] MSAP2.receive(MDATind_Connection_Request()) {
30:                         verdict.set(inconc);
31:                         stop;
32:                     }
33:                     [] ISAP1.receive(IDISind:{}) {
34:                         verdict.set(inconc);
35:                         stop;
36:                     }
37:                 }
38:             }
39:             [] ISAP1.receive(IDISind:{}) {
40:                 verdict.set(inconc);
41:                 stop;
42:             }
43:         }
44:     } with {
45:         display "purpose := do something useful";
46:         display "comments := example testcase definition";
47:         display (MyLocalVar) "comments := local variable";
48:         display (MyLocalConst) "comments := local constant";
49:         display (MyLocalTimer) "comments := local timer";
50:         display "detailed comments := detailed comments";
51:     }
    }
```

7 Производство BNF

```
1.  TabFreeText ::= [ExtendedAlphaNum]

2.  GroupReference ::= {GroupIdentifier "/"}+

3.  EncRuleIdentifier ::= Identifier

4.  CommentsAttribute :: = display """ "comments" ":=" TabFreeText """

5.  DetailedCommentsAttribute :: = display """ "detailed comments" ":=" TabFreeText """

6.  TTCN3ModuleId ::= ModuleIdentifier [ DefinitiveIdentifier ]
```

```
7.  ModuleAttributes ::= TabularPresentationFormatAttribute ";"
    ModuleVersionAttribute ";"
    ModuleDateAttribute ";"
    ModuleBaseStandardRefAttribute ";"
    ModuleTestStandardRefAttribute ";"
    ModulePICSRefAttribute ";"
    ModulePIXITRefAttribute ";"
    ModuleTestMethodAttribute ";"
    ModuleCommentsAttribute ";"
    ModuleDetailedCommentsAttribute ";"

8.  TabularPresentationFormatAttribute ::=
    display """ "presentation format := ETSI Tabular version" MajorVersion "." MinorVersion """

9.  MajorVersion ::= Number

10. MinorVersion ::= Number

11. ModuleVersionAttribute ::=
    display """ "module version" ":=" TabFreeText """

12. ModuleDateAttribute ::=
    display """ "module date" ":=" TabFreeText """

13. ModuleBaseStandardRefAttribute ::=
    display """ "module base standards ref" ":=" TabFreeText """

14. ModuleTestStandardRefAttribute ::=
    display """ "module test standards ref" ":=" TabFreeText """

15. ModulePICSRefAttribute ::=
    display """ "module pics ref" ":=" TabFreeText """

16. ModulePIXITRefAttribute ::=
    display """ "module pixit ref" ":=" TabFreeText """

17. ModuleTestMethodAttribute ::=
    display """ "module test method" ":=" TabFreeText """

18. ModuleCommentsAttribute ::=
    display """ "module comments" ":=" TabFreeText """

19. ModuleDetailedCommentsAttribute ::=
    display """ "module detailed comments" ":=" TabFreeText """

20. ModuleParPicsPixitRefAttribute ::=
    display "(" ModuleParIdentifier ")"
    """ "pics/pixit ref" ":=" TabFreeText """

21. ModuleParComments ::=
    display "(" ModuleParIdentifier ")"
    """ "comments" ":=" TabFreeText """

22. ImportsSourceRefAttribute ::=
    display """ "imports source ref" ":=" TabFreeText """

23. ImportsSourceDefinitionCommentsAttribute ::=
    display "(" ImportIdentifier ")"
    """ "comments" ":=" TabFreeText """

24. ImportSpecification ::= ( (Identifier | FullGroupIdentifier) | AllKeyword ) [ ExceptionsDef ]
    /* STATIC SEMANTIC: FullGroupIdentifier shall only be used for group imports. */

25. EncodeAttribute ::= encode """ TabFreeText """

26. SimpleTypesDetailedCommentsAttribute ::=
    display """ "simple types detailed comments" ":=" TabFreeText """

27. StructureType ::= record | union | set

28. FieldCommentsAttribute ::=
    display "(" FieldIdentifier ")" """ "comments" ":=" TabFreeText """

29. FieldEncodeAttribute ::=
    encode "(" FieldIdentifier ")" """ TabFreeText """

30. SequenceOfTypesDetailedCommentsAttribute ::=
    display """ "sequenceof types detailed comments" ":=" TabFreeText """
```

```
31. NamedValueCommentsAttribute ::=
    display "(" NamedValueIdentifier ")"
    """ "comments" ":=" TabFreeText """

32. TypeOrSignatureCommentsAttribute ::=
    display "(" TypeOrSignatureIdentifier ")"
    """ "comments" ":=" TabFreeText """

33. PortCommentsAttribute ::=
    display "(" PortIdentifier ")"
    """ "comments" ":=" TabFreeText """

34. ConstantsDetailedCommentsAttribute ::=
    display """ "simple types detailed comments" ":=" TabFreeText """

35. ExceptionCommentsAttribute ::=
    display "(" Type ")"
    """ "comments" ":=" TabFreeText """

36. VarConstOrTimerCommentsAttribute ::=
    display "(" VarConstOrTimerIdentifier ")"
    """ "comments" ":=" TabFreeText """

37. PurposeAttribute ::= display """ "purpose" ":=" TabFreeText """

38. SimpleTemplatesDetailedCommentsAttribute ::= display """ "simple templates detailed comments"
    ":=" TabFreeText """
```

СЕРИИ РЕКОМЕНДАЦИЙ МСЭ-Т

Серия A	Организация работы МСЭ-Т
Серия D	Общие принципы тарификации
Серия E	Общая эксплуатация сети, телефонная служба, функционирование служб и человеческие факторы
Серия F	Нетелефонные службы электросвязи
Серия G	Системы и среда передачи, цифровые системы и сети
Серия H	Аудиовизуальные и мультимедийные системы
Серия I	Цифровая сеть с интеграцией служб
Серия J	Кабельные сети и передача сигналов телевизионных и звуковых программ и других мультимедийных сигналов
Серия K	Защита от помех
Серия L	Конструкция, прокладка и защита кабелей и других элементов линейно-кабельных сооружений
Серия M	Управление электросвязью, включая СУЭ и техническое обслуживание сетей
Серия N	Техническое обслуживание: международные каналы передачи звуковых и телевизионных программ
Серия O	Требования к измерительной аппаратуре
Серия P	Качество телефонной передачи, телефонные установки, сети местных линий
Серия Q	Коммутация и сигнализация
Серия R	Телеграфная передача
Серия S	Оконечное оборудование для телеграфных служб
Серия T	Оконечное оборудование для телематических служб
Серия U	Телеграфная коммутация
Серия V	Передача данных по телефонной сети
Серия X	Сети передачи данных, взаимосвязь открытых систем и безопасность
Серия Y	Глобальная информационная инфраструктура, аспекты межсетевого протокола и сети последующих поколений
Серия Z	**Языки и общие аспекты программного обеспечения для систем электросвязи**

Отпечатано в Швейцарии
Женева, 2006 г.