ФЕДЕРАЛЬНОЕ АГЕНТСТВО
ПО ТЕХНИЧЕСКОМУ РЕГУЛИРОВАНИЮ И МЕТРОЛОГИИ

НАЦИОНАЛЬНЫЙ СТАНДАРТ РОССИЙСКОЙ ФЕДЕРАЦИИ

ГОСТ Р ИСО 20242-4 — 2012

Системы промышленной автоматизации и интеграция

СЛУЖЕБНЫЙ ИНТЕРФЕЙС ДЛЯ ИСПЫТАТЕЛЬНЫХ ПРИКЛАДНЫХ ПРОГРАММ

Часть 4

Шаблон профиля возможностей устройства

ISO 20242-4:2011
Industrial automation systems and integration — Service interface for testing applications — Part 4: Device capability profile template (IDT)

Издание официальное

Москва
Стандартинформ
2014

gost.e-ecolog.ru

Предисловие

1 ПОДГОТОВЛЕН АНО «Международная академия менеджмента и качества бизнеса» на основе собственного аутентичного перевода на русский язык международного стандарта, указанного в пункте 4

2 ВНЕСЕН Техническим комитетом по стандартизации ТК 100 «Стратегический и инновационный менеджмент»

3 УТВЕРЖДЕН И ВВЕДЕН В ДЕЙСТВИЕ Приказом Федерального агентства по техническому регулированию и метрологии от 29 ноября 2012 г. № 1716-ст

4 Настоящий стандарт идентичен международному стандарту ИСО 20242-4:2011 «Системы промышленной автоматизации и интеграция. Служебный интерфейс для испытательных прикладных программ. Часть 4. Шаблон профиля возможностей устройства» (ISO 20242-4:2011 «Industrial automation systems and integration — Service interface for testing applications — Part 4: Device capability profile template»).

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

5 ВВЕДЕН ВПЕРВЫЕ

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

© Стандартинформ, 2014

Настоящий стандарт не может быть полностью или частично воспроизведен, тиражирован и распространен в качестве официального издания без разрешения Федерального агентства по техническому регулированию и метрологии

Содержание

1 Область применения . . . 1
2 Нормативные ссылки . . . 1
3 Термины и определения . . . 1
4 Сокращения . . . 2
5 Концепция профиля возможностей устройства . . . 2
5.1 Общие положения . . . 2
5.2 Процедура создания DCD-, CCD- и PID-описаний . . . 3
6 Обобщенный шаблон профиля возможностей устройства . . . 5
6.1 Общие сведения . . . 5
6.2 Модель обобщенного DCPT-шаблона . . . 5
6.3 XML-схема для обобщенного DCPT-шаблона . . . 6
7 Общие правила применения DCPT-шаблона . . . 10
7.1 Общие сведения . . . 10
7.2 Заголовок DCPT-шаблона . . . 10
7.3 Дополнение шаблона профиля . . . 12
7.4 Закрепление текстовой информации . . . 13
7.5 Создание PID-описания . . . 13
8 Многоязычные текстовые элементы . . . 13
Приложение А (справочное) Шаблон профиля возможностей GDI-интерфейса ASAM . . . 15
Приложение В (справочное) Шаблоны профилей возможностей устройства для промышленного применения . . . 32
Приложение С (справочное) Шаблоны профилей возможностей открытого сетевого робототехнического интерфейса (OriN) . . . 51
Приложение ДА (справочное) Сведения о соответствии ссылочных международных стандартов ссылочным национальным стандартам Российской Федерации . . . 69
Библиография . . . 70

Введение

Настоящий стандарт разработан с целью облегчения интеграции измерительных и автоматических устройств, а также других периферийных устройств в различных компьютеризированных областях применения. В стандарте определены принципы создания драйверов устройств и режимы их работы в области применения измерительных автоматических средств.

Основной целью комплекса международных стандартов ИСО 20242 является обеспечение:

- независимости пользователя от операционной системы;
- независимости пользователя от технологии соединения (интерфейс устройства/сеть);
- независимости пользователя от поставщиков устройств;
- возможности сертификации драйверов устройств с подсоединенными к ним устройствами и выбранными режимами работы (с учетом используемой компьютерной платформы);
- независимости пользователя от последующих технологических усовершенствований устройств.

Стандарты комплекса ИСО 20242 не распространяются на разработку новых семейств устройств или использование специальных технологий для интерфейсов (сетей). В стандартах приведены общие описания сетей существующих устройств и их коммуникационных интерфейсов, обеспечивающих совместимость интерфейсов с другими устройствами аналогичного типа и назначения.

Комплекс стандартов ИСО 20242 включает в себя требования, распространяющиеся на:

- служебный интерфейс для управления ресурсами;
- служебный интерфейс виртуального устройства;
- шаблон функциональных характеристик устройства;
- служебный интерфейс прикладных программ;
- методы проверки на совместимость, критерии и отчеты о проведенных проверках.

Комплекс стандартов ИСО 20242 состоит из следующих частей:

- часть 1: Общий обзор;
- часть 2: Служебный интерфейс управления ресурсами.
- часть 3: Служебный интерфейс виртуального устройства;
- часть 4: Шаблон профиля возможностей устройства.

НАЦИОНАЛЬНЫЙ СТАНДАРТ РОССИЙСКОЙ ФЕДЕРАЦИИ

Системы промышленной автоматизации и интеграция

СЛУЖЕБНЫЙ ИНТЕРФЕЙС ДЛЯ ИСПЫТАТЕЛЬНЫХ ПРИКЛАДНЫХ ПРОГРАММ

Часть 4

Шаблон профиля возможностей устройства

Industrial automation systems and integration.
Service interface for testing applications.
Part 4. Device capability profile template

Дата введения — 2014—01—01

1 Область применения

В настоящем стандарте определены правила форматирования, а также синтаксические и семантические правила, предназначенные для описания:

- функциональных возможностей (далее — возможности) устройства и координатора (согласующего устройства) с использованием XML схем и
- конфигурации устройств на языке XML.

Примечание — Настоящий стандарт не распространяется на конфигурацию согласующего устройства, однако этот вопрос будет включен в следующее издание настоящего стандарта или в дополнение к нему.

2 Нормативные ссылки

В настоящем стандарте использованы нормативные ссылки на следующие стандарты, которые необходимо учитывать при использовании настоящего стандарта. В случае ссылок на документы, у которых указана дата утверждения, необходимо пользоваться только указанной редакцией. В случае, когда дата утверждения не приведена, следует пользоваться последней редакцией ссылочных документов, включая любые поправки и изменения к ним:

ИСО 15745-1:2003 Системы промышленной автоматизации и интеграция. Прикладная среда интегрирования открытых систем. Часть 1. Общее эталонное описание (ISO/IEC 15745-1:2003, Industrial automation systems and integration — Open systems application integration framework — Part 1: Generic reference description)

ИСО 20242-1 Системы промышленной автоматизации и интеграция. Служебный интерфейс для испытательных прикладных программ. Часть 1. Общие сведения (ISO 20242-1, Industrial automation systems and integration — Service interface for testing applications — Part 1: Overview)

ИСО 20242-3 Системы промышленной автоматизации и интеграция. Часть 3. Служебный интерфейс для испытательных прикладных программ (ИСО 20242-31, Industrial automation systems and integration — Service interface for testing applications — Part 3: Virtual device service interface)

3 Термины и определения

В настоящем стандарте используются термины, определенные в ИСО 20242-1 и ИСО 20242-3, а также следующие термины с соответствующими определениями:

3.1 **объект связи** (communication object): Объект, с которым может быть установлено коммуникационное соединение для записи или считывания значений параметров.

[ИСО 20242-1:2005, пункт 2.3]

3.2 **координатор; согласующее устройство** (coordinator): Программа с определенным интерфейсом для управления доступом прикладной программы к одному или нескольким драйверам устройств, а также для управления в реальном масштабе времени приложениями, синхронизацией и событиями.

[ИСО 20242-1:2005, пункт 2.4]

Издание официальное

3.3 **профиль возможностей согласующего устройства** (coordinator capability description): Текстовый файл, содержащий информацию о функциональных характеристиках виртуальных устройств, зарегистрированную в установленном формате (т. е. с заданной структурой, синтаксисом и т. д.).

[ИСО 20242-1:2005, пункт 2.5]

3.4 **описание возможностей устройства** (device capability description): Информация о функциональных возможностях виртуальных устройств.

[ИСО 20242-3:2011, раздел 3.2]

3.5 **драйвер устройства** (device driver): Компьютерный программный модуль, обеспечивающий интерфейс с сервисными (служебными) функциями (согласно ИСО 20242) и открывающий доступ адаптера платформы к физическим устройствам.

[ИСО 20242-2:2010, пункт 3.1]

3.6 **функциональный объект** (function object): Класс или экземпляр класса, определяющий одну функциональную возможность виртуального устройства.

[ИСО 20242-3:2011, пункт 3.4]

3.7 **операция** (operation): Класс или экземпляр класса, определяющий одну законченную процедуру.

[ИСО 20242-3:2011, пункт 3.5]

3.8 **параметрическое описание экземпляра класса** (parameterization instance description): Информация о конфигурациях согласующего устройства (координатора) и виртуальных устройствах.

3.9 **виртуальное устройство** (virtual device): Представление одного или нескольких физических устройств и/или автономных программных объектов для предоставления однозначного мнения относительно ресурсов интерфейса связи.

[ИСО 20242-3:2011, пункт 3.7]

4 Сокращения

CCD — описание возможностей согласующего устройства (координатора) (Coordinator Capability Description);

DCD — описание возможностей устройства (Device Capability Description);

DCPT — шаблон профиля возможностей устройства (Device Capability Profile Template);

PID — параметрическое описание экземпляра класса (Parameterization Instance Description);

VD — виртуальное устройство (Virtual Device);

VDSI — служебный интерфейс виртуального устройства (Virtual Device Service Interface);

XML — расширяемый язык разметки (eXtensible Markup Language).

5 Концепция профиля возможностей устройства

5.1 Общие положения

На рисунке 1 приведена диаграмма классов, используемых в концепции профиля возможностей устройства в соответствии с настоящим стандартом. Обобщенный шаблон профиля возможностей устройства (далее — DCPT-шаблон) определяют на основе использования обобщенной информации, получаемой из шаблона профиля обмена данными по ИСО 15745-1. Зависящий от выбираемой технологии DCPT-шаблон дополняет обобщенный DCPT-шаблон, что позволяет описывать возможности устройства на языке XML. DCD-описание позволяет дополнить зависящий от выбираемой технологии DCPT-шаблон и описать возможности согласующего устройства на языке XML. CCD-описание позволяет импортировать DCD-описание драйверов устройств и сделать описание возможностей системы. PID-описание определяют путем создания экземпляра CCD-описания (связь один к одному) и DCD-описания (связь один ко многим). PID-описание является реализацией профиля информационного обмена данными согласно ИСО 15745-1 и может использоваться вместе с другими профилями, указанными в настоящем стандарте.

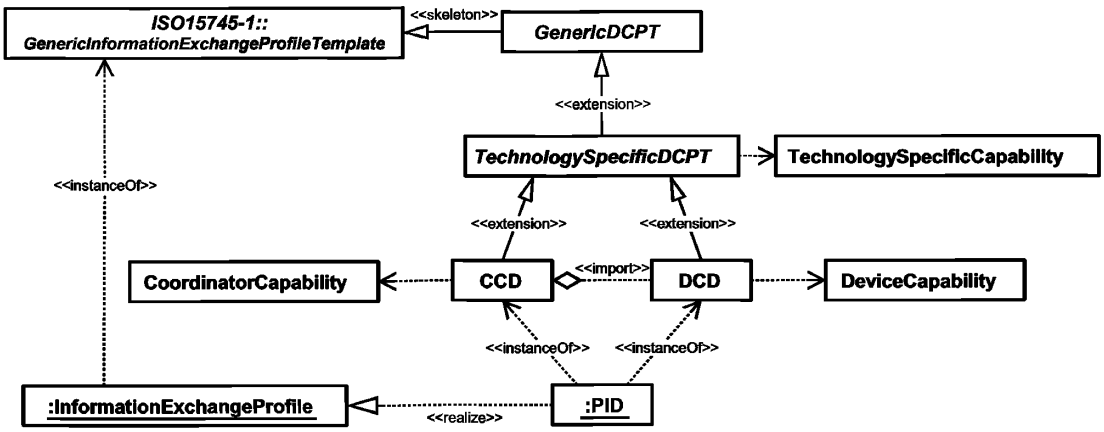


Рисунок 1 — Диаграмма классов, используемая в концепции профиля возможностей устройства

На рисунке 2 приведена диаграмма CCD- и DCD-классов, используемых в данной концепции. Обобщенный DCPT-шаблон позволяет определить совокупность обобщенных CCD- и DCD-описаний. Обобщенное CCD-описание характеризует обобщенные возможности согласующего устройства, обобщенное DCD-описание — обобщенные возможности виртуального устройства. Зависящий от выбираемой технологии CCD-класс в зависящем от технологии DCPT-классе содержит обобщенное CCD-описание и характеризует зависящие от технологии возможности согласующего устройства. Зависящий от выбираемой технологии DCD-класс содержит обобщенное DCD-описание и характеризует зависящие от выбираемой технологии возможности виртуальных устройств.

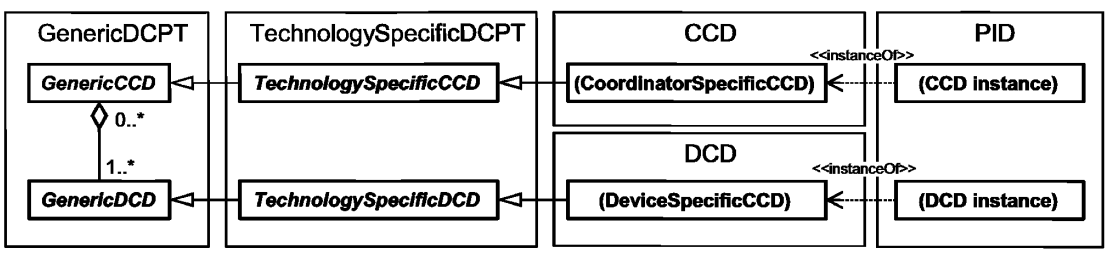


Рисунок 2 — Диаграмма CCD- и DCD-классов

Зависящий от согласующего устройства CCD-класс входит в зависящий от технологии CCD-класс и характеризует зависящие от согласующего устройства возможности. Имя объекта в зависящем от согласующего устройства CCD-классе может быть задано с помощью зависящего от устройства имени. Обобщенные CCD- и DCD-классы состоят в неразрывной связи, поэтому зависящий от согласующего устройства CCD-класс и зависящий от устройства DCD-класс связаны между собой. CCD-класс позволяет импортировать DCD-описания драйверов устройств. Экземпляр CCD-класса в PID-классе позволяет использовать зависящий от выбираемого согласующего устройства CCD-класс в виде XML-схемы и записать его как XML-экземпляр. Имя XML-тэга экземпляра CCD-класса будет совпадать с именем зависящего от выбираемого согласующего устройства CCD-класса, а имя XML-тэга экземпляра DCD-класса — с именем зависящего от выбираемого устройства DCD-класса.

5.2 Процедура создания DCD-, CCD- и PID-описаний

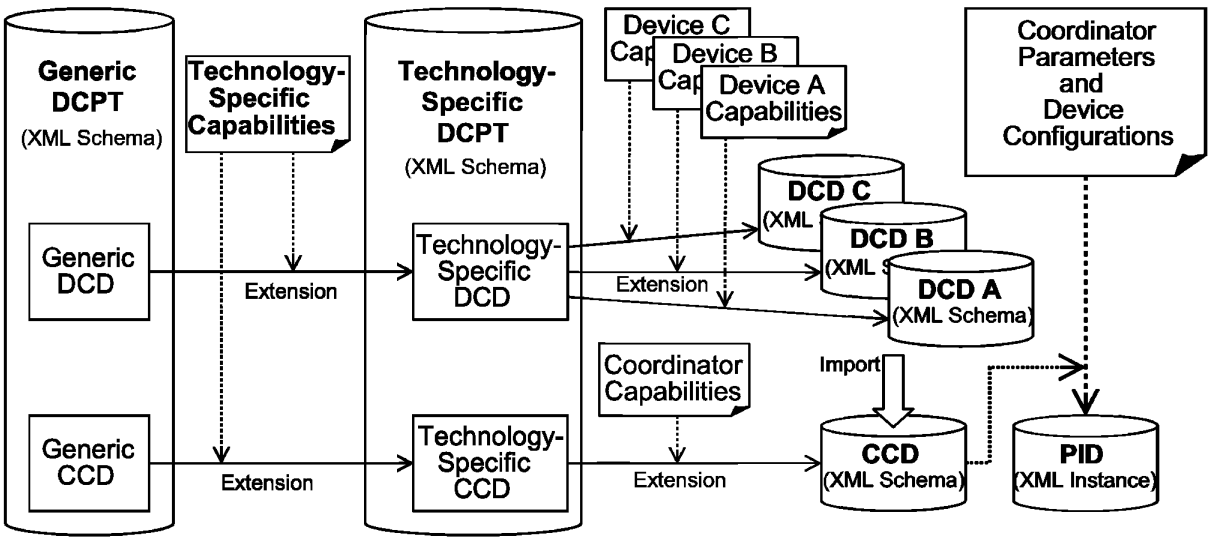
5.2.1 Общие положения

Рисунок 3 иллюстрирует процедуру создания DCD- и CCD-описаний. В разделе 6 настоящего стандарта в рамках XML-языка определен обобщенный DCPT-шаблон. Технология каждого служебного

интерфейса определяет зависящие от выбираемой технологии возможности и зависящий от выбираемой технологии DCPT-шаблон.

Примечание — Описания зависящих от выбираемой технологии DCPT-шаблонов для обобщенного интерфейса устройства (GDI) ASAM и технологии совместного использования информации для обмена данными (MICX-технология) приведены в приложениях А и В.

Поставщик устройства или оборудования расширяет часть зависящих от выбираемой технологии DCD-описаний в зависящем от выбираемой технологии DCPT-шаблоне и приводит возможности драйвера для этого устройства или оборудования в DCD-описании, а затем предоставляет его вместе с драйвером устройства. Поставщик согласующего устройства расширяет часть зависящих от выбираемой технологии CCD-описаний в зависящем от выбираемой технологии DCPT-шаблоне и приводит возможности этого согласующего устройства в CCD-описании. В средствах конфигурирования используются CCD-описания, которые позволяют импортировать требуемые DCD-описания и создавать PID-описание, содержащее описания всех требуемых экземпляров классов с их именами и численными значениями. Согласующее устройство обеспечивает считывание PID-описания, осуществляет его инсталляцию, конфигурирование устройств и обеспечивает служебный интерфейс прикладных программ после применения ИСО 20242-5.



Примечание — Объекты в виде цилиндров означают XML-файлы, а объекты в виде листков — информацию относительно функциональных возможностей (ФВ) и конфигурации. Сплошные стрелки на диаграмме указывают на создание XML-файла, стрелки в виде точек — на ввод информации относительно ФВ, а пунктирные стрелки — на применение шаблона.

Generic DCPT (XML Schema) — файл обобщенного DCPT-шаблона (с XML-структурой); Generic DCD — обобщенное DCD-описание; Generic CCD — обобщенное CCD-описание; Technology specific capabilities — информация относительно зависящих от технологии возможностей; Extension — расширение; Technology-specific DCPT (XML Schema) — файл зависящего от технологии DCPT—шаблона (с XML-структурой); Technology-specific DCD — зависящее от технологии DCD-описание; Technology-specific CCD — зависящее от технологии CCD-описание; Device capabilities — возможности устройства; DCD (XML Schema) — файл DCD-описания устройства (XML-схема); Import — импорт; CCD (XML Schema) — файл CCD-описания; Coordinator parameters and device configurations — информация о параметрах согласующего устройства и конфигурации устройства; PID (XML Instance) — файл PID-описания (XML-экземпляр класса); Coordinator capabilities — возможности согласующего устройства.

Рисунок 3 — Блок-схема процедуры создания CCD- и DCD-описаний

5.2.2 Описание возможностей устройства (DCD-описание)

DCD-описание обычно содержит:

- идентификационную информацию относительно драйвера устройства;
- описание возможностей виртуальных устройств, поддерживаемых драйвером.

5.2.3 Описание возможностей согласующего устройства (CCD-описание)

CCD-описание обычно содержит:

- идентификационную информацию, относящуюся к программному обеспечению согласующего устройства;
- описание возможностей согласующего устройства;
- информацию об аттестации служебного интерфейса, поддерживающего прикладные программы.

5.2.4 Параметрическое описание экземпляров класса (PID-описание)

PID-описание обычно содержит:

- идентификационную информацию, относящуюся к PID-описанию профиля обмена информацией по ИСО 15745;
- параметрическое описание экземпляров класса с зависящими от выбираемого приложения именами;
- конфигурационные данные для драйверов устройств.

6 Обобщенный шаблон профиля возможностей устройства

6.1 Общие сведения

Обобщенный DCPT-шаблон позволяет определить общую структуру DCPT-шаблона, не зависящую от технической реализации служебного интерфейса. Зависящий от выбираемой технологии DCPT-шаблон является дополнительным к обобщенному DCPT-шаблону.

6.2 Модель обобщенного DCPT-шаблона

Обобщенный DCPT-шаблон использует шаблон профиля обмена обобщенной информацией, получаемый согласно ИСО 15745-1 в качестве основы и дополненный информацией относительно модели VDSI-интерфейса согласно ИСО 20242-3 и ИСО 20242-5 (служебный интерфейс прикладных программ). На рисунке 4 представлена структура классов обобщенного DCPT-шаблона. Профиль обмена информацией является корневым классом и содержит заголовок по ИСО 15745, а также основной текст по ИСО 15745, заголовок по ИСО 15745 содержит идентификационную информацию, относящуюся к используемому профилю. Основной текст по ИСО 15745 содержит одно или несколько обобщенных CCD-описаний. Профиль обмена информацией, заголовок по ИСО 15745 и основной текст по ИСО 15745 определены в ИСО 15745-1.

Обобщенное CCD-описание относится к абстрактному классу и характеризует обобщенные возможности согласующего устройства. Номер обобщенного CCD-описания совпадает с номером этого устройства.

Обобщенное CCD-описание содержит обобщенные DCD-описания. Обобщенное DCD-описание относится к (принадлежит) абстрактному классу и характеризует обобщенные возможности драйвера устройства. Номер обобщенного DCD-описания совпадает с номером этого драйвера.

Обобщенное DCD-описание содержит описания виртуальных устройств, которые относятся к (принадлежат) абстрактному классу и характеризуют обобщенные возможности виртуального устройства.

Виртуальное устройство содержит функциональные объекты, которые относятся к (принадлежат) абстрактному классу и характеризуют обобщенные возможности устройства.

Функциональный объект содержит объекты связи и рабочие операции.

Объект связи относится к (принадлежит) абстрактному классу и характеризует обобщенные возможности объекта связи, определенные в ИСО 20242-3.

Рабочая операция относится к (принадлежит) абстрактному классу и характеризует обобщенные возможности этой операции, определенные в ИСО 20242-3.

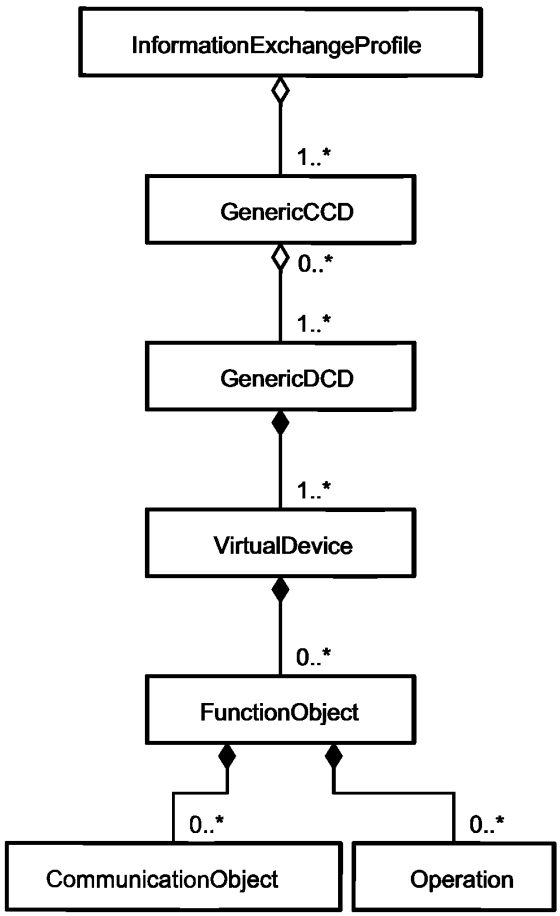


Рисунок 4 — Диаграмма классов модели обобщенного DCPT-описания

6.3 XML-схема для обобщенного DCPT-шаблона

XML-схема для обобщенного DCPT-шаблона содержится в шаблоне профиля обмена информацией (см. рисунок 5), включает в себя XML-схему для обобщенного CCD-описания и относится к элементу этого описания.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<xsd:schema xmlns:xsd="gost.e-ecolog.ru "
xmlns="gost.e-ecolog.ru "
targetNamespace="gost.e-ecolog.ru
elementFormDefault="qualified">
<xsd:annotation>
<xsd:appinfo source="DCPTHeader.xsd">
<DCPTHeader>
<DCPTIdentification>GenericDCPT</DCPTIdentification>
<DCPTRevision>1.0</DCPTRevision>
<DCPTName>Generic DCPT</DCPTName>
<DCPTSource>GenericDCPT.xsd</DCPTSource>
<DCPTDate>2011-07-01</DCPTDate>
</DCPTHeader>
</xsd:appinfo>
</xsd:annotation>
<!-- * Include GenericCCD * -->
```

```
<xsd:include schemaLocation="GenericCCD.xsd"/>
<!-- * ISO15745 Profile Root * -->
<xsd:element name="ISO15745Profile">
<xsd:complexType>
<xsd:sequence>
<xsd:element ref="ProfileHeader"/>
<xsd:element ref="ProfileBody"/>
</xsd:sequence>
</xsd:complexType>
</xsd:element>
<!-- * HEADER DATA TYPES * -->
<xsd:element name="ProfileHeader">
<xsd:complexType>
<xsd:sequence>
<xsd:element name="ProfileIdentification" type="xsd:string"/>
<xsd:element name="ProfileRevision" type="xsd:string"/>
<xsd:element name="ProfileName" type="xsd:string"/>
<xsd:element name="ProfileSource" type="xsd:string"/>
<xsd:element name="ProfileClassID" type="ProfileClassID_DataType"
fixed="InformationExchange"/>
<xsd:element name="ProfileDate" type="xsd:date"
minOccurs="0"/>
<xsd:element name="AdditionalInformation" type="xsd:anyURI" minOccurs="0"/>
<xsd:element name="ISO15745Reference" type="ISO15745Reference_DataType"/>
<xsd:element name="IASInterfaceType" type="IASInterface_DataType"
fixed="CSI" minOccurs="0" maxOccurs="unbounded"/>
</xsd:sequence>
</xsd:complexType>
</xsd:element>
<xsd:complexType name="ISO15745Reference_DataType">
<xsd:sequence>
<xsd:element name="ISO15745Part" type="xsd:string"
fixed="1"/>
<xsd:element name="ISO15745Edition" type="xsd:string"
fixed="1"/>
<xsd:element name="ProfileTechnology" type="xsd:string"
fixed="None"/>
</xsd:sequence>
</xsd:complexType>
<xsd:simpleType name="ProfileClassID_DataType">
<xsd:restriction base="xsd:string">
<xsd:enumeration value="AIP"/>
<xsd:enumeration value="Process"/>
<xsd:enumeration value="InformationExchange"/>
<xsd:enumeration value="Resource"/>
<xsd:enumeration value="Device"/>
<xsd:enumeration value="CommunicationNetwork"/>
<xsd:enumeration value="Equipment"/>
<xsd:enumeration value="Human"/>
<xsd:enumeration value="Material"/>
</xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="IASInterface_DataType">
<xsd:union>
<xsd:simpleType>
<xsd:restriction base="xsd:string">
<xsd:enumeration value="CSI"/>
<xsd:enumeration value="HCI"/>
<xsd:enumeration value="ISI"/>
<xsd:enumeration value="API"/>
```

```
<xsd:enumeration value="CMI"/>
<xsd:enumeration value="ESI"/>
<xsd:enumeration value="FSI"/>
<xsd:enumeration value="MTI"/>
<xsd:enumeration value="SEI"/>
<xsd:enumeration value="USI"/>
</xsd:restriction>
</xsd:simpleType>
<xsd:simpleType>
<xsd:restriction base="xsd:string">
<xsd:length value="4"/>
</xsd:restriction>
</xsd:simpleType>
</xsd:union>
</xsd:simpleType>
<!-- * BODY SECTION * -->
<xsd:element name="ProfileBody">
<xsd:complexType>
<xsd:sequence>
<xsd:element ref="GenericCCD" maxOccurs="unbounded"/>
</xsd:sequence>
</xsd:complexType>
</xsd:element>
</xsd:schema>
```

Рисунок 5 — XML-схема для обобщенного DCPT-шаблона

XML-схема для обобщенного CCD-описания содержится в шаблоне обобщенного CCD-описания (см. рисунок 6), включает в себя XML-схему для обобщенного DCD-описания и относится к элементу этого описания.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<xsd:schema xmlns:xsd="gost.e-ecolog.ru "
xmlns="gost.e-ecolog.ru "
targetNamespace="gost.e-ecolog.ru "
elementFormDefault="qualified">
<xsd:annotation>
<xsd:appinfo source="DCPTHeader.xsd">
<DCPTHeader>
<DCPTIdentification>GenericCCD</DCPTIdentification>
<DCPTRevision>1.0</DCPTRevision>
<DCPTName>Generic CCD</DCPTName>
<DCPTSource>GenericCCD.xsd</DCPTSource>
<DCPTDate>2011-07-01</DCPTDate>
</DCPTHeader>
</xsd:appinfo>
</xsd:annotation>
<!-- * Include GenericDCD * -->
<xsd:include schemaLocation="GenericDCD.xsd"/>
<!-- * Elements Declaration * -->
<xsd:element name="GenericCCD" type="GenericCCDType" abstract="true"/>
<xsd:complexType name="GenericCCDType" abstract="true">
<xsd:sequence>
<xsd:element ref="GenericDCD" maxOccurs="unbounded"/>
</xsd:sequence>
<xsd:attribute name="name" type="xsd:string"/>
```

```
<xsd:attribute name="category" type="xsd:string" use="required" fixed="CCD"/>
</xsd:complexType>
</xsd:schema>
```

Рисунок 6 — XML-схема для обобщенного CCD-описания

XML-схема для обобщенного DCD-описания содержится в шаблоне обобщенного DCD-описания (см. рисунок 7) и включает в себя шаблоны для виртуального устройства, функционального объекта, объекта связи и рабочей операции.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<xsd:schema xmlns:xsd="gost.e-ecolog.ru "
xmlns="gost.e-ecolog.ru "
targetNamespace="gost.e-ecolog.ru "
elementFormDefault="qualified">
<xsd:annotation>
<xsd:appinfo source="DCPTHeader.xsd">
<DCPTHeader>
<DCPTIdentification>GenericDCD</DCPTIdentification>
<DCPTRevision>1.0</DCPTRevision>
<DCPTName>Generic DCD</DCPTName>
<DCPTSource>GenericDCD.xsd</DCPTSource>
<DCPTDate>2009-03-16</DCPTDate>
</DCPTHeader>
</xsd:appinfo>
</xsd:annotation>
<!-- * Elements Declaration * -->
<xsd:element name="GenericDCD"
type=»GenericDCDType" abstract="true"/>
<xsd:complexType name="GenericDCDType" abstract="true">
<xsd:sequence>
<xsd:element ref="VirtualDevice" minOccurs="0" maxOccurs="unbounded"/>
</xsd:sequence>
<xsd:attribute name="name" type="xsd:string"/>
<xsd:attribute name="category" type="xsd:string"
use="required" fixed="DCD"/>
</xsd:complexType>
<xsd:element name="VirtualDevice"
type="VirtualDeviceType" abstract="true"/>
<xsd:complexType name="VirtualDeviceType" abstract="true">
<xsd:sequence>
<xsd:element ref="FunctionObject" maxOccurs="unbounded"/>
</xsd:sequence>
<xsd:attribute name="name" type="xsd:string"/>
<xsd:attribute name="category" type="xsd:string"
use="required" fixed="MODULE"/>
</xsd:complexType>
<xsd:element name="FunctionObject"
type="FunctionObjectType" abstract="true"/>
<xsd:complexType name="FunctionObjectType" abstract="true">
<xsd:sequence>
<xsd:element ref="CommunicationObject" minOccurs="0" maxOccurs="unbounded"/>
<xsd:element ref="Operation" minOccurs="0" maxOccurs="unbounded"/>
</xsd:sequence>
<xsd:attribute name="name" type="xsd:string"/>
<xsd:attribute name="category" type="xsd:string"
use="required" fixed="INTERFACE"/>
</xsd:complexType>
<xsd:element name="CommunicationObject"
```

```
type="CommunicationObjectType" abstract="true"/>
<xsd:complexType name="CommunicationObjectType" abstract="true">
<xsd:attribute name="name" type="xsd:string"/>
</xsd:complexType>
<xsd:element name="Operation" type="OperationType" abstract="true"/>
<xsd:complexType name="OperationType" abstract="true">
<xsd:attribute name="name" type="xsd:string"/>
<xsd:attribute name="category" type="xsd:string"
use="required" fixed="OPERATION"/>
</xsd:complexType>
</xsd:schema>
```

Рисунок 7 — XML-схема для обобщенного DCD-описания

7 Общие правила применения DCPT-шаблона

7.1 Общие сведения

Эти правила необходимо использовать для расширения (дополнения) DCPT-шаблона. Имена XML-элементов (тэги) могут быть определены специально для согласующего и других устройств. Для специальных целей в этот шаблон могут быть введены любые атрибуты и элементы, на которые настоящий стандарт не распространяется.

7.2 Заголовок DCPT-шаблона

Идентификационную информацию относительно DCD- и CCD-описаний приводят с помощью элемента xsd:appinfo в элементе xsd:annotation XML-схемы. Заголовок DCPT-шаблона содержит атрибуты, указанные в таблице 1.

Таблица 1 — Элементы заголовка DCPT-шаблона

Наименование элемента	Описание элемента
DCPTIdentification	Идентификатор DCPT-шаблона. Тип XML-данных: строка. Пример — ABC-123-XX
DCPTRevision	Редакция DCPT-шаблона. Тип XML-данных: строка. Пример — 2.34
DCPTName	Описательное имя DCPT-шаблона. Тип XML-данных: строка. Пример — DCD Thermometer
DCPTSource	Идентификатор разработчика DCPT-шаблона. Тип XML-данных: строка. Пример — ASAM
DCPTClassID	Идентификатор класса профиля. Тип XML-данных: строка. Действующими классами профилей являются: GenericDCPT TechnlogySpecificDCPT CCD DCD Пример — DCD
DCPTDate	Дата выпуска данной редакции профиля в формате CCYY-MM-DD. Тип XML-данных: дата 2011-09-21

Окончание таблицы 1

Наименование элемента	Описание элемента
Additionalinformation	Расположение диаграмм /дополнительной информации относительно профиля. Данное поле является дополнительным. Тип XML-данных: любой унифицированный идентификатор ресурса (URI), anyURI Пример — gost.e-ecolog.ru
ISO20242Reference	Идентификатор части ИСО 20242 (см. элемент ISO20242Part), вместе с его редакцией (см. элемент ISO20242Edition) и используемой технологией (см. элемент Technology). Множественные ссылки разрешены, например для устройств с более чем одним связующим интерфейсом
ISO20242Edition	Редакция ссылочной части ИСО 20242. Тип XML-данных: положительное целое число. Пример — 1
Technology	Наименование ссылочной технологии Тип XML-данных: строка При отсутствии применяемой в ИСО 20242 технологии необходимо использовать значение «None». Пример — None

Соответствующая XML-схема приведена на рисунке 8.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<xsd:schema xmlns:xsd="gost.e-ecolog.ru "
elementFormDefault="qualified">
<xsd:element name="DCPTHeader">
<xsd:complexType>
<xsd:sequence>
<xsd:element name="DCPTIdentification" type="xsd:string"/>
<xsd:element name="DCPTRevision" type="xsd:string"/>
<xsd:element name="DCPTName" type="xsd:string"/>
<xsd:element name="DCPTSource" type="xsd:string"/>
<xsd:element name="DCPTClassID" type="DCPTClassID_DataType"/>
<xsd:element name="DCPTDate" type="xsd:string"/>
<xsd:element name="AdditionalInformation" type="xsd:anyURI" minOccurs="0"/>
<xsd:element name="ISO20242Reference" type="ISO20242Reference_DataType"/>
</xsd:sequence>
</xsd:complexType>
</xsd:element>
<xsd:simpleType name="DCPTClassID_DataType">
<xsd:restriction base="xsd:string">
<xsd:enumeration value="GenericDCPT"/>
<xsd:enumeration value="TechnlogySpecificDCPT"/>
<xsd:enumeration value="CCD"/>
<xsd:enumeration value="DCD"/>
</xsd:restriction>
</xsd:simpleType>
<xsd:complexType name="ISO20242Reference_DataType">
<xsd:sequence>
<xsd:element name="ISO20242Part" type="xsd:string" fixed="4"/>
<xsd:element name="ISO20242Edition" type="xsd:string" fixed="1"/>
<xsd:element name="Technology" type="xsd:string"/>
```

```
</xsd:sequence>
</xsd:complexType>
</xsd:schema>
```

Рисунок 8 — XML-схема для заголовка DCPT-шаблона

7.3 Дополнение шаблона профиля

7.3.1 Создание DCD-описания

DCD-описание содержит зависящие от выбираемого типа устройства элементы, которые получаются из зависящих от выбираемой технологии элементов и заменяют абстрактные элементы обобщенного DCPT-шаблона.

Примечание — В соответствии с настоящим стандартом не требуется импортировать обобщенные или зависящие от выбираемой технологии XML-схемы в зависящее от выбираемого устройства CD-описание, однако независимо созданное DCD-описание должно содержать информацию, аналогичную DCD-описанию импортируемых XML-схем, соответствующих настоящему стандарту.

Имена элементов (тэги) для DCD-описания могут быть специфичными для различных типов устройств. Элементы должны быть снабжены атрибутом «category», содержащим ключ для типа этого элемента, указанного в таблице 2. Содержанием этого атрибута должна быть ссылка на соответствующий элемент обобщенного DCPT-шаблона.

Таблица 2 — Ключи для типа элемента

Элемент обобщенного DCPT-шаблона	Содержание XML-атрибута «category»
GenericCCD	CCD
GenericDCD	DCD
Virtual Device	MODULE
FunctionObject	INTERFACE
Operation	OPERATION

Дополнительные элементы могут иметь специфическое содержимое атрибута «category» в зависящих от выбираемой технологии приложениях.

7.3.2 Закрепление зависящих от выбираемого устройства элементов

Зависящий от выбираемого устройства элемент DCD-описания содержит один или несколько элементов виртуального устройства, которые не должны принадлежать (относиться к) какому-либо другому зависящему от выбираемого устройства элементу DCD-описания.

Зависящий от выбираемого устройства элемент виртуального устройства заменяет обобщенный элемент виртуального устройства и содержит (или не содержит) несколько зависящих от выбираемого устройства элементов функционального объекта, которые не должны принадлежать (относиться к) каким-либо другим зависящим от выбираемого виртуального устройства элементам функционального объекта и виртуального устройства.

Зависящий от выбираемого устройства элемент функционального объекта заменяет элемент обобщенного функционального объекта и содержит (или не содержит) несколько зависящих от выбираемого объекта связи элементов, которые не должны принадлежать (относиться к) какому-либо другому зависящему от выбираемого устройства элемента функциональному объекту.

7.3.3 Порядок реализации VDSI-интерфейса

Для создания виртуальных устройств, функциональных объектов и объектов связи с помощью сервисов VDSI_Initiate, VDSI_CreateFuncObject и VDSI_CreateCommObject, рассмотренных в ИСО 20242-3, их должен сопровождать дополнительный XML-атрибут «initOrder», относящийся к типу unsignedInt, определенный в XML-схеме и содержащий порядок создания экземпляра элемента с помощью VDSI-интерфейса.

7.3.4 Параметризация объектов связи

Если значение закреплено за объектом связи, то его необходимо записать с помощью согласующего устройства и сервиса VDSI_Write для VDSI-интерфейса. Можно определить дополнительный порядок записи значений в объекты связи, выполняемой с использованием XML-атрибута «initOrder».

Примечание — Способ многократной записи объекта связи приведен в приложении А.

7.4 Закрепление текстовой информации

Любой элемент DCPT-шаблона может иметь два вида присоединяемой текстовой информации — краткого и развернутого сообщения. Эти сообщения должны быть определены в дополнительном XML-экземпляре и пронумерованы согласно разделу 8. Нумерация сообщений организована иерархически в пронумерованных областях текста с помощью пронумерованных текстовых элементов. Номера текстов закрепляют за элементами DCPT-шаблона с помощью дополнительных XML-атрибутов типа xsd:unsignedInt, указанных в таблице 3.

Таблица 3 — XML-атрибуты присвоения текстовой информации

Имя XML-атрибута	Описание содержимого атрибута
areaMsg	Номер области короткого сообщения message
infMsg	Номер короткого сообщения
areaText	Номер области развернутого сообщения
infText	Номер развернутого сообщения

7.5 Создание PID-описания

7.5.1 XML-экземпляры CCD-классов

XML-экземпляр зависящего от выбранного согласующего устройства CCD-класса позволяет характеризовать особенности согласующего устройства. CCD-класс содержит один или несколько XML-экземпляров DCD-классов.

7.5.2 XML-экземпляры DCD-классов

XML-экземпляр зависящего от выбираемого устройства DCD-класса позволяет характеризовать драйвер устройства, который является элементом VDSI-интерфейса согласно ИСО 20242-3. Экземпляр DCD-класса содержит один или несколько XML-экземпляров зависящих от выбираемого устройства классов виртуальных устройств.

7.5.3 XML-экземпляры классов виртуальных устройств

XML-экземпляр зависящего от выбираемого устройства класса виртуальных устройств позволяет характеризовать экземпляр класса виртуального устройства с помощью сервиса VDSI_CreateFuncObject для VDSI-интерфейса. Может существовать (или не существовать) несколько экземпляров зависящего от выбираемого устройства класса виртуальных устройств.

7.5.4 XML-экземпляры классов функциональных объектов

XML-экземпляр зависящего от выбираемого устройства класса функциональных объектов позволяет характеризовать экземпляр класса функционального объекта, созданного с помощью сервиса для VDSI-интерфейса. Может существовать несколько экземпляров зависящего от выбираемого устройства класса виртуальных устройств. Экземпляр класса виртуального устройства может содержать (или не содержать) несколько экземпляров зависящих от выбираемого устройства классов объектов связи и/или несколько экземпляров зависящих от выбираемого устройства классов рабочих операций.

7.5.5 XML-экземпляры классов объектов связи

XML-экземпляр зависящего от выбираемого класса объектов связи позволяет характеризовать объект связи, созданный с помощью сервиса VDSI_CreateCommObject для VDSI-интерфейса. В одном конкретном классе объектов связи может быть только один экземпляр этого класса.

7.5.6 XML-экземпляры классов рабочих операций

XML-экземпляр зависящего от выбираемого класса рабочих операций позволяет характеризовать экземпляр класса рабочих операций, создаваемый в точном соответствии с реализацией класса связанных функциональных объектов с помощью VDSI-интерфейса. В одном зависящем от выбираемого устройства классе рабочих операций должен существовать только один экземпляр этого класса.

8 Многоязычные текстовые элементы

Для текстовой информации используют три вида сообщений — короткие информационные сообщения, расширенные информационные сообщения и сообщения об ошибках. Указанные сообщения должны содержаться в XML-экземпляре со структурой, определяемой согласно рисунку 9.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="gost.e-ecolog.ru "
elementFormDefault="qualified">
<xsd:element name="DIT">
<xsd:complexType>
<xsd:sequence>
<xsd:element name="Area" minOccurs="0" maxOccurs="unbounded">
<xsd:complexType>
<xsd:all>
<xsd:element name="SubAreaText" type="SubAreaType"/>
<xsd:element name="SubAreaMessage" type="SubAreaType"/>
<xsd:element name="SubAreaError" type="SubAreaType"/>
</xsd:all>
<xsd:attribute name="number" type="xsd:unsignedInt" use="required"/>
<xsd:attribute name="name" type="xsd:Name" use="optional"/>
</xsd:complexType>
</xsd:element>
</xsd:sequence>
</xsd:complexType>
</xsd:element>
<xsd:complexType name="SubAreaType">
<xsd:sequence>
<xsd:element name="Entry" type="EntryType"
minOccurs="0" maxOccurs="unbounded"/>
</xsd:sequence>
</xsd:complexType>
<xsd:complexType name="EntryType">
<xsd:sequence>
<xsd:element name="Text" type="LangSpecType"
minOccurs="0" maxOccurs="unbounded"/>
</xsd:sequence>
<xsd:attribute name="number" type="xsd:unsignedInt" use="required"/>
<xsd:attribute name="name" type="xsd:Name" use="optional"/>
</xsd:complexType>
<xsd:complexType name="LangSpecType">
<xsd:simpleContent>
<xsd:extension base="xsd:normalizedString">
<xsd:attribute name="lang" type="xsd:language" use="required"/>
</xsd:extension>
</xsd:simpleContent>
</xsd:complexType>
```

Рисунок 9 — XML-схема для текстовой информации об устройстве

Расширенные информационные сообщения вводят в XML-элемент <SubAreaText>, короткие информационные сообщения — в элемент <SubAreaMessage>, а сообщения об ошибках — в элемент <SubAreaError>. Зависящие от выбираемого языка сообщения вводят в XML-элементы <Text> с XML-атрибутом «lang» для идентификации выбираемого языка сообщений.

Приложение А
(справочное)

Шаблон профиля возможностей GDI-интерфейса ASAM

А.1 Общие сведения

Обобщенный интерфейс устройства (GDI-интерфейс) ASAM определяет интерфейс для испытательных применений. В данном приложении описан зависящий от GDI-интерфейса шаблон возможностей устройства (версия 4.4 ASAM GDI). Примеры DCD-, CCD- и PID-описаний приведены далее.

А.2 Зависящая от выбираемого GDI-интерфейса модель профиля

А.2.1 Общие сведения

Зависящая от выбираемого GDI-интерфейса модель профиля содержит информацию, необходимую для описания возможностей данного устройства и параметризации. На рисунке А.1 приведена диаграмма классов для зависящего от выбираемого GDI-интерфейса шаблона профилей возможностей устройства.

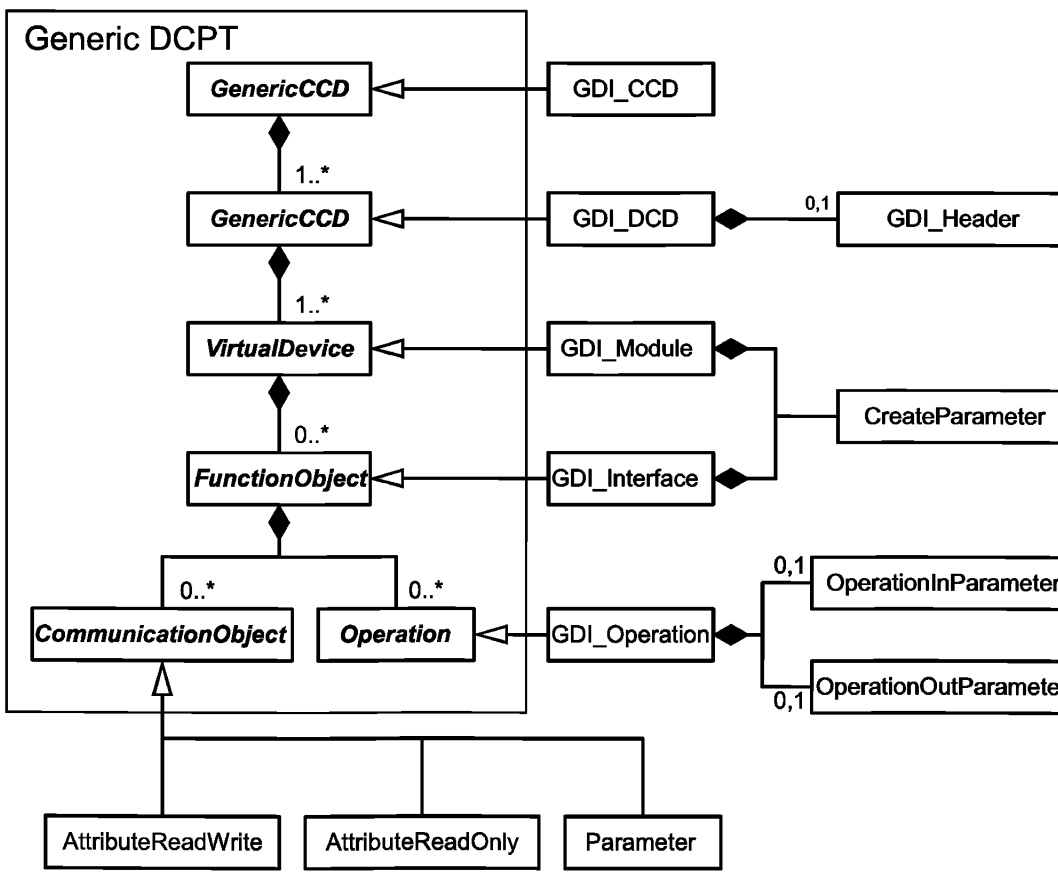


Рисунок А.1 — Диаграмма классов для зависящей от GDI-интерфейса модели DCPT-шаблона

А.2.2 Класс GDI_CCD (CCD-описание GDI-интерфейса)

Этот класс характеризует возможности зависящего от выбираемого GDI-интерфейса согласующего устройства, который входит в класс GenericCCD (обобщенное CCD-описание) и является абстрактным классом. Зависящий от выбираемого согласующего устройства класс CCD-описаний должен наследовать его и определять возможности конкретного согласующего устройства.

А.2.3 Класс GDI_DCD (DCD-описание GDI-интерфейса)

Этот класс характеризует возможности зависящего от выбираемого GDI-интерфейса драйвера устройства, который входит в класс GenericDCD (обобщенное DCD-описание) и является абстрактным классом. Зависящий от выбираемого согласующего устройства класс GDI-DCD должен наследовать его и определять возможности конкретного драйвера устройства.

А.2.4 Класс GDI_Header (заголовок для GDI-интерфейса)

Этот класс содержит дополнительную информацию, используемую для реализации драйвера устройства. Элементы класса GDI_Header указаны в таблице А.1. Определение класса GDI_Header также приведено в файле common.xsd GDI-интерфейса (см. А.6.2).

Таблица А.1 — Элементы класса GDI_Header

Элементы класса GDI_Header		Тип элемента	Описание элемента
DCD_Version		xsd:unsignedInt	Номер версии DCD-описания
DeviceVersion		xsd:unsignedInt	Номер варианта исполнения устройства
DriverName		xsd:string	Наименование драйвера
DriverVersion		xsd:unsignedInt	Номер версии драйвера
Factory		xsd:string	Наименование изготовителя
DIT		xsd:string	Имя XML-текстового файла
GDI_Version	Major	rxsd:unsignedByte	Номер полной версии
	Minor	rxsd:unsignedByte	Номер минимальной версии
	Revision	rxsd:unsignedByte	Номер редакции

А.2.5 Класс GDI_Module (модуль GDI-интерфейса)

Этот класс характеризует возможности зависящего от выбираемого GDI-интерфейса виртуального устройства, который входит в класс VirtualDevice (виртуальное устройство) и является абстрактным классом. Зависящий от выбираемого согласующего устройства класс GDI_Module должен наследовать его и определять возможности конкретного виртуального устройства. Класс GDI _Module может содержать класс CreateParameter и быть идентифицирован по номеру, содержащемуся в дополнительном XML-атрибуте «moduleId» (типа «xsd:unsignedShort»).

А.2.6 Класс GDI _Interface (GDI-интерфейс)

Этот класс характеризует возможности функциональных объектов зависящего от выбираемого GDI-интерфейса виртуального устройства, который входит в класс FunctionObject (функциональный объект) и является абстрактным классом. Зависящий от выбираемого согласующего устройства класс GDI_Interface должен наследовать его свойства и определять возможности конкретного функционального объекта. Класс GDI_Interface может содержать класс CreateParameter и быть идентифицирован по номеру, содержащемуся в дополнительном XML-атрибуте «funcId» (типа «xsd:unsignedShort»).

А.2.7 Класс CreateParameter (создание параметра)

Этот класс предназначен для описания созданного параметра функционального объекта или виртуального устройства и является абстрактным классом. Каждый зависящий от устройства класс CreateParameter должен наследовать его свойства и определять тип данных созданного параметра. Сервисы VDSI-интерфейса VDSI_Initiate и VDSI_CreateFuncObject используют созданные параметры.

А.2.8 Класс GDI_Operation (работа GDI-интерфейса)

Этот класс предназначен для описания работы зависящего от выбираемого GDI-интерфейса виртуального устройства, который входит в класс Operation (рабочая операция) и является абстрактным классом. Каждый зависящий от выбираемого устройства класс GDI_Operation должен наследовать его свойства и определять возможности рабочей операции. Этот класс имеет один входной и один рабочий выходной параметры, которые могут быть использованы один раз (или не появляться) в XML-экземпляре. Закрепление значения за входным рабочим параметром в этом экземпляре (PID-описании) указывает на то, что операция должна быть выполнена с целью конфигурирования. Класс GDI_Operation идентифицируют с помощью номера, содержащегося в дополнительном XML-атрибуте, называемом «operationIdId» (типа «xsd:unsignedShort»).

А.2.9 Класс OperationInParameter (выходной параметр операции)

Класс OperationInParameter предназначен для отметки выполнения рабочей операции и является абстрактным классом. Каждый зависящий от выбираемого устройства класс OperationInParameter должен наследовать свойства этого класса и определять тип данных входного параметра.

А.2.10 Класс OperationOutParameter (выходной параметр операции)

Этот класс предназначен для указания возвращаемого значения в рабочей операции, который содержит значение выходного параметра. Каждый зависящий от выбираемого устройства класс OperationOutParameter должен наследовать свойства этого класса и определять тип данных значения этого параметра.

А.2.11 Класс AttributeReadOnly (только считывание атрибута)

Этот класс предназначен для указания значения времени обработки функционального объекта, которое может только считываться. Данный класс наследует свойства класса CommunicationObject и является абстрактным классом. Каждый зависящий от выбираемого устройства класс AttributeReadOnly должен наследовать свойства этого класса и определять функциональные возможности каждого значения времени обработки и типа данных значения этого параметра.

А.2.12 Класс AttributeReadWrite (считывание/запись атрибута)

Этот класс предназначен для указания значения времени обработки функционального объекта, которое может быть как считано, так и записано. Данный класс наследует свойства класса CommunicationObject и является абстрактным классом. Каждый зависящий от выбираемого устройства класс AttributeReadWrite должен наследовать свойства этого класса и определять возможности каждого значения времени обработки и типа данных этого значения.

А.2.13 Класс Parameter (параметр)

Этот класс характеризует возможности параметра функционального объекта, который может быть как считан и записан. Данный класс наследует свойства класса CommunicationObject и является абстрактным классом. Каждый зависящий от выбираемого устройства класс Parameter должен наследовать свойства этого класса и определять возможности каждого параметра и его тип данных.

А.2.14 Идентификация объектов связи GDI-интерфейса

Экземпляры классов AttributeReadOnly, AttributeReadWrite и Parameter идентифицируют с помощью номеров, которые являются числом этих экземпляров в представителе FunctionObject, начиная с 1.

А.3 Элементы типовых данных и сценариев конфигурирования

А.3.1 Общие сведения

В 7.3.3 XML-атрибут initOrder был введен для описания последовательностей конфигурирования устройств, связанных с созданием виртуальных устройств, функциональных объектов и объектов связи, что является другим сценарием конфигурирования, определенным для объектов связи, которые в различные моменты времени могут принимать различные значения. С этой целью вводят класс OrderedValue, который не требуется в случае задания сценария записи данных, однако в этом случае может быть использовано то же значение, что и в атрибуте initOrder классов CommunicationObject и OrderedValue.

Элемент Value определяют для упрощения наследования типов данных без учета влияния наследования структурных классов. С помощью этого метода определение типов данных для объектов связи и формирование параметров разделяются в определении объекта связи. Значения элемента не требуются, если типы данных в объектах связи определены точно.

А.3.2 Класс OrderedValue

Этот класс характеризует значение, которое можно многократно устанавливать в различные моменты времени. Установка этого значения означает его запись с помощью сервисов VDSI_Write или VDSI_Execute, что применимо к объектам классов AttributeReadWrite, Parameter и OperationInParameter. Упорядоченное значение относится к абстрактному классу, который содержит порядок инициализации в его XML-атрибуте initOrder, который описывает, когда значение должно быть установлено, а также само значение. Каждый зависящий от выбираемого устройства класс OrderedValue должен наследовать класс OrderedValue и определять тип данных для значения.

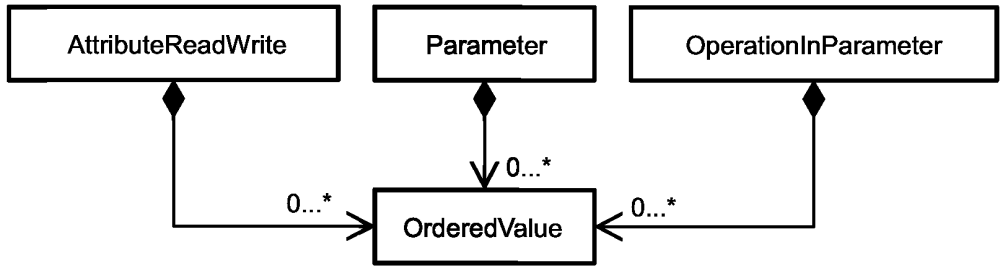


Рисунок А.2 — Элемент OrderedValue для сценариев конфигурирования

А.3.3 Класс Value (значение)

Класс Value характеризует элемент типа xsd:anyType, который может ограничиваться любым другим типом и поэтому является меткой-заполнителем для любого типа элемента, определенного в описании возможностей устройства. Использование этих дополнительных элементов облегчает разделение наследования для определения типов данных.

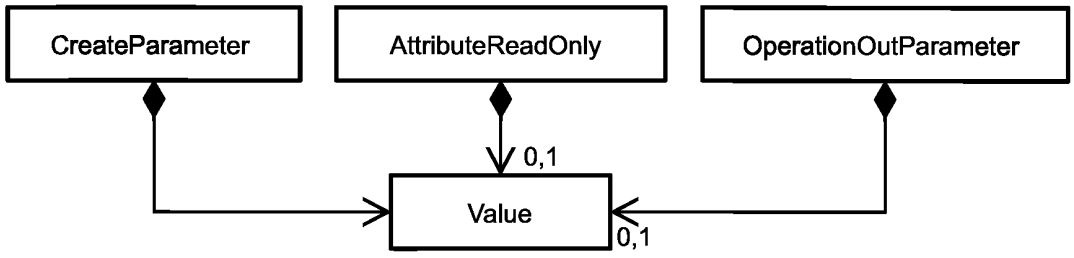


Рисунок А.3 — Элемент Value для разделенных определений типов

Класс Value содержит данные, которые заменяются на служебный примитивный параметр CreateParameter, после чего используют либо сервис VDSI_Initiate для реализации виртуального устройства, либо сервис VDSI_CreateFuncObject — для реализации функционального объекта.

При конфигурировании не происходит замены на объект связи AttributeReadOnly и OperationOutParameter класса GDI_Operation, однако класс Value может быть использован для представления типа данных объектов.

Класс Value также может быть использован вместе с классом OrderedValue для классов AttributeReadWrite, Parameter и OperationInParameter, в случае если требуется сценарий конфигурирования с различными значениями в различные моменты времени. Если нет необходимости использования этого сценария, класс Value может быть использован вместо класса OrderedValue с классами AttributeReadWrite, Parameter и OperationInParameter.

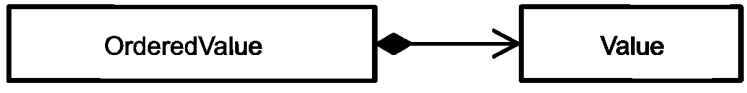


Рисунок А.4 — Элементы классов Value и OrderedValue в сценарии конфигурирования

А.4 Дополнительные идентификаторы типа для обобщенного DCPT-шаблона

В таблице А.2 указаны зависящие от выбираемой технологии типы идентификаторов параметра create, объектов связи и рабочих параметров.

Таблица А.2 — Идентификаторы типов объектов связи

Элемент зависящего от выбираемой технологии DCPT-шаблона	Содержимое XML-атрибута «category»	Содержимое XML-атрибута «readonly»
AttributeReadWrite	ATTRIBUTE	false
AttributeReadOnly	ATTRIBUTE	true
Parameter	PARAMETER	(false)

Примечание — Дополнительный XML-атрибут «readonly» типа xsd:boolean является обязательным только для классов AttributeReadWrite и AttributeReadOnly. Если также используется класс Parameter, следует использовать содержимое «false».

Таблица А.3 — Идентификаторы типов рабочих параметров

Элемент зависящего от выбираемой технологии DCPT-шаблона	Содержимое XML-атрибута «category»
OperationInParameter	IN
OperationOutParameter	OUT

Таблица А.4 — Идентификаторы типов созданных параметров

Элемент зависящего от выбираемой технологии DCPT-шаблона	Содержимое XML-атрибута «category»
Create ParameterCR	CREATEPARAMETER

А.5 Дополнительные XML-атрибуты для обобщенного DCPT-шаблона

Объекты связи AttributeReadOnly и AttributeReadWrite могут использоваться для передачи незатребованных данных с помощью сервисов VDSI_InfReport и VDSI_Accept. Запрос на использование указанных сервисов во время работы выполняется во время конфигурирования путем установки XML-атрибутов infReport и приема типа xsd:Boolean на значение true.

Таблица А.5 — XML-атрибуты для передачи незатребованных данных

Имя XML-атрибута	Использование с объектом	Описание содержимого
infReport	AttributeReadWrite, AttributeReadOnly	Запрос на использование сервиса VDSI_InfReport
accept	AttributeReadWrite	Запрос на использование сервиса VDSI_Accept

А.6 Зависящий от выбираемого GDI-интерфейса шаблон профиля возможностей устройства.

А.6.1 Общие сведения

Файл GDIcommon.xsd содержит XML-схему с базовыми классами, которые могут использоваться для описания XML-схемы и могут содержать всю информацию относительно описания возможностей, необходимых для оценки (определения) параметров. Определенные типы Parameter и AttributeReadWrite могут быть дополнены элементами OrderedValue и Value. В этом случае класс Value может иметь тип данных, который может быть определен в дополнительной XML-схеме. Кроме того, определенные типы классов CreateParameter, AttributeReadOnly и OperationOutParameter могут быть дополнены элементами класса Value с внешне определенными типами данных.

Примечание — Если элементы классов OrderedValue и Value не используются, а объекты связи содержат значения простых или комплексных типов данных, то они могут иметь другую структуру XML-схемы, основанную на дополнении типов данных. В этом случае файл GDIcommon.xsd может использоваться как модель структуры результирующего XML-экземпляра и необходимых XML-атрибутов.

А.6.2 XML-схема: Файл GDIcommon.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="gost.e-ecolog.ru "
elementFormDefault="qualified" attributeFormDefault="unqualified">
<!— Header information is contained in each schema file —>
<xsd:annotation>
<xsd:appinfo source="gost.e-ecolog.ru ">
<DCPTHeader>
<DCPTIdentification>GDI_Common</DCPTIdentification>
<DCPTRevision>1.0</DCPTRevision>
<DCPTName>GDIcommon</DCPTName>
<DCPTSource>GDIcommon.xsd</DCPTSource>
```

```
<DCPTClassID>TechnlogySpecificDCPT</DCPTClassID>
<DCPTDate>2009-03-16</DCPTDate>
<ISO20242Reference>
<ISO20242Edition>1</ISO20242Edition>
<Technology>ASAM-GDI</Technology>
</ISO20242Reference>
</DCPTHeader>
</xsd:appinfo>
</xsd:annotation>
<!-- Valid content for XML-Attribut category (kind of DCD element) -->
<xsd:simpleType name="Category">
<xsd:restriction base="xsd:string">
<xsd:enumeration value="MODULE"/>
<xsd:enumeration value="INTERFACE"/>
<xsd:enumeration value="CREATEPARAMETER"/>
<xsd:enumeration value="PARAMETER"/>
<xsd:enumeration value="ATTRIBUTE"/>
<xsd:enumeration value="OPERATION"/>
<xsd:enumeration value="IN"/>
<xsd:enumeration value="OUT"/>
<xsd:enumeration value="DCD"/>
<xsd:enumeration value="CCD"/>
</xsd:restriction>
</xsd:simpleType>
<!-- Group of attributes for assigning multilingual text to elements -->
<xsd:attributeGroup name="TextAttributes">
<xsd:attribute name="areaMsg" type="xsd:unsignedShort" use="optional"/>
<xsd:attribute name="infMsg" type="xsd:unsignedShort" use="optional"/>
<xsd:attribute name="areaText" type="xsd:unsignedShort" use="optional"/>
<xsd:attribute name="infText" type="xsd:unsignedShort" use="optional"/>
</xsd:attributeGroup>
<!-- Basic type definition for most elements used in the DCD -->
<xsd:complexType name="TNamedDCDElement" abstract="true">
<xsd:attribute name="name" type="xsd:string" use="optional"/>
<xsd:attribute name="initOrder" type="xsd:unsignedInt" use="optional"/>
<xsd:attributeGroup ref="TextAttributes"/>
</xsd:complexType>
<xsd:complexType name="TOrderedValue" abstract="true">
<xsd:sequence>
<xsd:element name="Value" type="xsd:anyType"/>
</xsd:sequence>
<xsd:attribute name="initOrder" type="xsd:unsignedInt" use="required"/>
</xsd:complexType>
<!-- Virtual Devices have additional XML-attributes "moduleId" and category="MODULE". -->
<xsd:complexType name="GDI_Module" abstract="true">
<xsd:complexContent>
<xsd:extension base="TNamedDCDElement">
<xsd:attribute name="moduleId" type="xsd:unsignedShort" use="required"/>
<xsd:attribute name="category" type="Category" use="required" fixed="MODULE"/>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<!-- Function Objects have additional XML-attributes "funcId" and category="INTERFACE". -->
<xsd:complexType name="GDI_Interface" abstract="true">
<xsd:complexContent>
<xsd:extension base="TNamedDCDElement">
<xsd:attribute name="funcId" type="xsd:unsignedShort" use="required"/>
<xsd:attribute name="category" type="Category" use="required" fixed="INTERFACE"/>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<!-- Create Parameter have an additional XML-attribute category="CREATEPARAMETER" -->
```

```
<xsd:complexType name="CreateParameter" abstract="true">
<xsd:attribute name="category" type="Category" use="required" fixed="CREATEPARAMETER"/>
<xsd:attributeGroup ref="TextAttributes"/>
</xsd:complexType>
<!-- Communication Objects which may be changed have readonly="false" -->
<xsd:complexType name="CommunicationObjectReadWrite" abstract="true">
<xsd:complexContent>
<xsd:extension base="TNamedDCDElement">
<xsd:attribute name="readonly" type="xsd:boolean" fixed="false"/>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<!-- Communication Objects which may not be changed have readonly="true" -->
<xsd:complexType name="CommunicationObjectReadOnly" abstract="true">
<xsd:complexContent>
<xsd:extension base="TNamedDCDElement">
<xsd:attribute name="readonly" type="xsd:boolean" use="required" fixed="true"/>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<!-- GDI-Parameter are Communication Objects that may be changed and have category="PARAMETER" -->
<xsd:complexType name="Parameter" abstract="true">
<xsd:complexContent>
<xsd:extension base="CommunicationObjectReadWrite">
<xsd:attribute name="category" type="Category" use="required" fixed="PARAMETER"/>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<!-- GDI-Attribute have to be separated in readonly and read/write Communication Objects -->
<xsd:complexType name="AttributeReadWrite" abstract="true">
<xsd:complexContent>
<xsd:extension base="CommunicationObjectReadWrite">
<xsd:attribute name="category" type="Category" use="required" fixed="ATTRIBUTE"/>
<xsd:attribute name="infReport" type="xsd:boolean"/>
<xsd:attribute name="accept" type="xsd:boolean"/>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="AttributeReadOnly" abstract="true">
<xsd:complexContent>
<xsd:extension base="CommunicationObjectReadOnly">
<xsd:attribute name="category" type="Category" use="required" fixed="ATTRIBUTE"/>
<xsd:attribute name="infReport" type="xsd:boolean"/>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<!-- Operations -->
<xsd:complexType name="GDI_Operation" abstract="true">
<xsd:complexContent>
<xsd:extension base="TNamedDCDElement">
<xsd:attribute name="operationId" type="xsd:unsignedShort" use="required"/>
<xsd:attribute name="category" type="Category" use="required" fixed="OPERATION"/>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="OperationOutParameter" abstract="true">
<xsd:attribute name="category" type="Category" use="required" fixed="OUT"/>
<xsd:attributeGroup ref="TextAttributes"/>
</xsd:complexType>
<xsd:complexType name="OperationInParameter" abstract="true">
<xsd:attribute name="category" type="Category" use="required" fixed="IN"/>
<xsd:attributeGroup ref="TextAttributes"/>
```

```
</xsd:complexType>
<!-- Device Driver (when instantiated) -->
<xsd:complexType name="TVersion">
<xsd:sequence>
<xsd:element name="Major" type="xsd:unsignedByte"/>
<xsd:element name="Minor" type="xsd:unsignedByte"/>
<xsd:element name="Revision" type="xsd:unsignedByte"/>
</xsd:sequence>
</xsd:complexType>
<xsd:complexType name="THeader">
<xsd:sequence>
<xsd:element name="DCD_Version" type="xsd:unsignedInt"/>
<xsd:element name="DeviceVersion" type="xsd:unsignedInt"/>
<xsd:element name="DriverName" type="xsd:string"/>
<xsd:element name="DriverVersion" type="xsd:unsignedInt"/>
<xsd:element name="Factory" type="xsd:string"/>
<xsd:element name="DIT" type="xsd:string"/>
<xsd:element name="GDI_Version" type="TVersion"/>
</xsd:sequence>
</xsd:complexType>
<xsd:complexType name="GDI_DCD" abstract="true">
<xsd:sequence>
<xsd:element name="GDI_Header" type="THeader" minOccurs="0"/>
</xsd:sequence>
<xsd:attribute name="category" type="Category" use="required" fixed="DCD"/>
<xsd:attributeGroup ref="TextAttributes"/>
<xsd:attribute name="dllPath" type="xsd:string" use="required"/>
<xsd:attribute name="driverVersion" type="xsd:int" use="required"/>
</xsd:complexType>
</xsd:schema>
```

А.7 Примеры DCD-описания

А.7.1 Общие сведения

Файлы DCDa1.xsd и DCDa2.xsd содержат XML-схемы примеров DCD-описания, которые позволяют дополнять XML-схему в файле GDICommon.xsd в соответствии с возможностями драйверов устройства. Имена XML-элементов могут быть определены в соответствии с требованиями пользователя. Структурную информацию, имеющую отношение к XML-элементу, определяют с помощью содержимого XML-атрибута «category».

А.7.2 XML-схема: Файл DCDa1.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="gost.e-ecolog.ru
elementFormDefault="qualified" attributeFormDefault="unqualified">
<xsd:annotation>
<xsd:appinfo source="gost.e-ecolog.ru
<DCPTHeader>
<DCPTIdentification>DCD1</DCPTIdentification>
<DCPTRevision>1.0</DCPTRevision>
<DCPTName>DCD1</DCPTName>
<DCPTSource>DCD1.xsd</DCPTSource>
<DCPTClassID>DCD</DCPTClassID>
<DCPTDate>2009-03-16</DCPTDate>
<ISO20242Reference>
<ISO20242Edition>1</ISO20242Edition>
<Technology>ASAM-GDI</Technology>
</ISO20242Reference>
</DCPTHeader>
</xsd:appinfo>
</xsd:annotation>
<xsd:include schemaLocation="GDICommon.xsd"/>
<!-- Main type is GDI_DCD with contained Virtual Device Types.
There may be many different Virtual Device Types contained. -->
<xsd:complexType name="Driver01">
<xsd:complexContent>
```

```
<xsd:extension base="GDI_DCD">
<xsd:sequence>
<xsd:element name="myDevice01" type="Device01" minOccurs="0" maxOccurs="unbounded"/>
</xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<!-- Virtual Device Types have fixed numbers for identification in moduleID.
This is a restriction of type GDI_Module. If you want to avoid
restriction, move attribute moduleId from type GDI_Module with a fixed
number to type Device01 below, which then is an extension of GDI_Module
and type Device01_ will be obsolete. -->
<xsd:complexType name="Device01_">
<xsd:complexContent>
<xsd:restriction base="GDI_Module">
<xsd:attribute name="moduleId" type="xsd:unsignedShort" use="required" fixed="1000"/>
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
<!-- Virtual Device Types typically contain a Create Paramter of a specific
type and several Function Objects of different types.
This exemplary Function Object type describes an input channel and it is
possible to use up to 16 channels with each Virtual Device instance. -->
<xsd:complexType name="Device01">
<xsd:complexContent>
<xsd:extension base="Device01_">
<xsd:sequence>
<xsd:element name="NumOfChannel" type="CRParamDevice"/>
<xsd:element name="fnADInput" type="IFADInput" minOccurs="0" maxOccurs="16"/>
</xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<!-- A simple Create Parameter for the Virtual Device of type Device01-->
<xsd:complexType name="CRParamDevice">
<xsd:complexContent>
<xsd:extension base="CreateParameter">
<xsd:sequence>
<xsd:element name="Value" type="xsd:unsignedShort" default="5"/>
</xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<!-- Function Object Types have fixed numbers for identification in funcID.
This is a restriction of type GDI_Interface. If you want to avoid
restriction, move attribute funcId from type GDI_Interface with a fixed
number to type IFADInput below, which then is an extension of
GDI_Interface and type IFADInput_ will be obsolete. -->
<xsd:complexType name="IFADInput_">
<xsd:complexContent>
<xsd:restriction base="GDI_Interface">
<xsd:attribute name="funcId" type="xsd:unsignedShort" use="required" fixed="1077"/>
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
<!-- A simple Function Object type for the Virtual Device of type Device01.
Function Object types typically contain a Create Parameter and several
Communication Objects of kind GDI-Attribute or GDI-Parameter.
An example with additional GDI-Operation is given in DCD2.xsd -->
<xsd:complexType name="IFADInput">
<xsd:complexContent>
<xsd:extension base="IFADInput_">
```

```
<xsd:sequence>
<xsd:element name="Interrupt" type="CRParamIFADInput"/>
<xsd:element name="Polarity" type="PolarityType" minOccurs="0"/>
<xsd:element name="Channel" type="ChannelType" minOccurs="0"/>
<xsd:element name="ADValue" type="ADValueType" minOccurs="0"/>
</xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<!-- A simple Create Parameter for the Function Object of type IFADInput -->
<xsd:complexType name="CRParamIFADInput">
<xsd:complexContent>
<xsd:extension base="CreateParameter">
<xsd:sequence>
<xsd:element name="Value" type="xsd:unsignedShort" default="5"/>
</xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<!-- A simple readonly GDI-Attribute for the Function Object of type IFADInput -->
<xsd:complexType name="ADValueType">
<xsd:complexContent>
<xsd:extension base="AttributeReadOnly">
<xsd:sequence>
<xsd:element name="OrderedValue" type="OrderedValueAD" minOccurs="0"/>
</xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<!-- A simple Value-Type for GDI-Attribute of type ADValueType (above) -->
<xsd:complexType name="OrderedValueAD">
<xsd:complexContent>
<xsd:restriction base="TOrderedValue">
<xsd:sequence>
<xsd:element name="Value" type="xsd:unsignedShort" default="0"/>
</xsd:sequence>
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
<!-- A simple read/write GDI-Attribute for the Function Object of type IFADInput -->
<xsd:complexType name="PolarityType">
<xsd:complexContent>
<xsd:extension base="AttributeReadWrite">
<xsd:sequence>
<xsd:element name="OrderedValue" type="OrderedValuePolarity"
minOccurs="0" maxOccurs="unbounded"/>
</xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<!-- A simple Value-Type for GDI-Attribute of type PolarityType (above) -->
<xsd:complexType name="OrderedValuePolarity">
<xsd:complexContent>
<xsd:restriction base="TOrderedValue">
<xsd:sequence>
<xsd:element name="Value" type="TePolarity" default="0"/>
</xsd:sequence>
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
<xsd:simpleType name=»TePolarity»>
<xsd:restriction base="xsd:short">
```

```
<xsd:enumeration value="0"/>
<xsd:enumeration value="1"/>
</xsd:restriction>
</xsd:simpleType>
<!-- A simple GDI-Parameter for the Function Object of type IFADInput -->
<xsd:complexType name="ChannelType">
<xsd:complexContent>
<xsd:extension base="Parameter">
<xsd:sequence>
<xsd:element name="OrderedValue" type="OrderedValueChannel"
minOccurs="0" maxOccurs="unbounded"/>
</xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<!-- A simple Value-Type for GDI-Parameter of type ChannelType (above) -->
<xsd:complexType name="OrderedValueChannel">
<xsd:complexContent>
<xsd:restriction base="TOrderedValue">
<xsd:sequence>
<xsd:element name="Value" type="xsd:unsignedShort" default="0"/>
</xsd:sequence>
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
</xsd:schema>
```

A.7.3 XML-схема: Файл DCDa2.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="gost.e-ecolog.ru "
elementFormDefault="qualified" attributeFormDefault="unqualified">
<xsd:annotation>
<xsd:appinfo source="gost.e-ecolog.ru ">
<DCPTHeader>
<DCPTIdentification>DCD2</DCPTIdentification>
<DCPTRevision>1.0</DCPTRevision>
<DCPTName>DCD2</DCPTName>
<DCPTSource>DCD2.xsd</DCPTSource>
<DCPTClassID>DCD</DCPTClassID>
<DCPTDate>2007-03-16</DCPTDate>
<ISO20242Reference>
<ISO20242Edition>1</ISO20242Edition>
<Technology>ASAM-GDI</Technology>
</ISO20242Reference>
</DCPTHeader>
</xsd:appinfo>
</xsd:annotation>
<xsd:include schemaLocation="GDICommon.xsd"/>
<!-- Main type is GDI_DCD with contained Virtual Device Types.
There may be many different Virtual Device Types contained. -->
<xsd:complexType name="Driver02">
<xsd:complexContent>
<xsd:extension base="GDI_DCD">
<xsd:sequence>
<xsd:element name="myDevice02" type="Device02" minOccurs="0" maxOccurs="unbounded"/>
</xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<!-- Virtual Device Types have fixed numbers for identification in moduleID.
This is a restriction of type VDInstance. If you want to avoid
restriction, move attribute moduleId from type GDI_Module with a fixed
```

```
number to type Device02 below, which then is an extension of GDI_Module
and type Device02_ will be obsolete. -->
<xsd:complexType name="Device02_">
<xsd:complexContent>
<xsd:restriction base="GDI_Module">
<xsd:attribute name="moduleId" type="xsd:unsignedShort" use="required" fixed="1002"/>
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
<!-- This exemplary Virtual Device Type only contains one Function Object
type without restricting the number of instances. -->
<xsd:complexType name="Device02">
<xsd:complexContent>
<xsd:extension base="Device02_">
<xsd:sequence>
<xsd:element name="myFunction02" type="MyFunction02Type"
minOccurs="0" maxOccurs="unbounded"/>
</xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<!-- Function Object Types have fixed numbers for identification in funcID.
This is a restriction of type GDI_Interface. If you want to avoid
restriction, move attribute funcId from type GDI_Interface with a fixed
number to type MyFunction02Type below, which then is an extension of
GDI_Interface and type MyFunction02Type_ will be obsolete. -->
<xsd:complexType name="MyFunction02Type_">
<xsd:complexContent>
<xsd:restriction base="GDI_Interface">
<xsd:attribute name="funcId" type="xsd:unsignedShort" use="required" fixed="1008"/>
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
<!-- This Function Object type example contains only the Create Parameter
and one optional Operation. -->
<xsd:complexType name="MyFunction02Type">
<xsd:complexContent>
<xsd:extension base="MyFunction02Type_">
<xsd:sequence>
<xsd:element name="myCRPar02" type="cpType02"/>
<xsd:element name="myOperation02" type="myOperation02Type" minOccurs="0"/>
</xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<!-- The Create Parameter of this Function Object type is a simple structure for a typical poor serial interface with
restricted speed and fixed bits per transmitted character. -->
<xsd:complexType name="cpType02">
<xsd:complexContent>
<xsd:extension base="CreateParameter">
<xsd:sequence>
<xsd:element name="Value" type="serialCom"/>
</xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="serialCom">
<xsd:sequence>
<xsd:element name="speed" type="speedType"/>
<xsd:element name="length" type="xsd:integer" fixed="8"/>
</xsd:sequence>
</xsd:complexType>
```

```
<xsd:simpleType name="speedType">
<xsd:restriction base="xsd:int">
<xsd:enumeration value="2400"/>
<xsd:enumeration value="4800"/>
<xsd:enumeration value="9600"/>
</xsd:restriction>
</xsd:simpleType>
<!-- This Operation has optional input and output values. -->
<xsd:complexType name="myOperation02Type_">
<xsd:complexContent>
<xsd:restriction base="GDI_Operation">
<xsd:attribute name="operationId" type="xsd:unsignedShort" use="required" fixed="1009"/>
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="myOperation02Type">
<xsd:complexContent>
<xsd:extension base="myOperation02Type_">
<xsd:sequence>
<xsd:element name="OutValue" type="op02outType" minOccurs="0"/>
<xsd:element name="InValue" type="op02inType" minOccurs="0"/>
</xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<!-- The input parameters for this operation may be a sequence of ordered
values. If accordingly instantiated, from this follow multiple requests
(and confirmations) of service VDSI_Execute (ISO 20242.3) -->
<xsd:complexType name="op02inType">
<xsd:complexContent>
<xsd:extension base="OperationInParameter">
<xsd:sequence>
<xsd:element name="Input" type="orderedvalueop02in" maxOccurs="unbounded"/>
</xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="orderedvalueop02in">
<xsd:complexContent>
<xsd:restriction base="TOrderedValue">
<xsd:sequence>
<xsd:element name="Value" type="xsd:double"/>
</xsd:sequence>
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
<!-- The output parameters for this operation is a special type "void",
which defines, that the accordingly XML element has to be empty.
Implementations of service VDSI_Execute (e.g. Annex A of ISO 20242.3)
will define this case also (e.g. NULL pointer). It simply means,
that there will be no output value for this operation -->
<xsd:complexType name="op02outType">
<xsd:complexContent>
<xsd:extension base="OperationOutParameter">
<xsd:sequence>
<xsd:element name="Value" type="void"/>
</xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="void"/>
</xsd:schema
```

А.8 Пример CCD-описания

А.8.1 Общие сведения

Файл CCDa.xsd содержит XML-схему примера CCD-описания, которая позволяет описывать XML-схему параметризации двух драйверов устройств.

А.8.2 XML-схема: Файл CCDa.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="gost.e-ecolog.ru "
elementFormDefault="qualified" attributeFormDefault="unqualified">
<xsd:annotation>
<xsd:appinfo source="gost.e-ecolog.ru ">
<DCPTHeader>
<DCPTIdentification>CCD1</DCPTIdentification>
<DCPTRevision>1.0</DCPTRevision>
<DCPTName>CCDa</DCPTName>
<DCPTSource>CCDa.xsd</DCPTSource>
<DCPTClassID>CCD</DCPTClassID>
<DCPTDate>2009-03-16</DCPTDate>
<ISO20242Reference>
<ISO20242Edition>1</ISO20242Edition>
<Technology>ASAM-GDI</Technology>
</ISO20242Reference>
</DCPTHeader>
</xsd:appinfo>
</xsd:annotation>
<!-- A coodinator capability description contains the capability
descriptions of all devices, which will be usable -->
<xsd:include schemaLocation="DCDa1.xsd"/>
<xsd:include schemaLocation="DCDa2.xsd"/>
<!-- * ISO15745 Profile Root * -->
<xsd:element name="ISO15745Profile">
<xsd:complexType>
<xsd:sequence>
<xsd:element ref="ProfileHeader"/>
<xsd:element ref="ProfileBody"/>
</xsd:sequence>
</xsd:complexType>
</xsd:element>
<!-- * HEADER DATA TYPES * -->
<xsd:element name="ProfileHeader">
<xsd:complexType>
<xsd:sequence>
<xsd:element name="ProfileIdentification" type="xsd:string"/>
<xsd:element name="ProfileRevision" type="xsd:string"/>
<xsd:element name="ProfileName" type="xsd:string"/>
<xsd:element name="ProfileSource" type="xsd:string"/>
<xsd:element name="ProfileClassID" type="ProfileClassID_DataType" fixed="InformationExchange"/>
<xsd:element name="ProfileDate" type="xsd:date" minOccurs="0"/>
<xsd:element name="AdditionalInformation" type="xsd:anyURI" minOccurs="0"/>
<xsd:element name="ISO15745Reference" type="ISO15745Reference_DataType"/>
<xsd:element name="IASInterfaceType" type="IASInterface_DataType" fixed="CSI"
minOccurs="0" maxOccurs="unbounded"/>
</xsd:sequence>
</xsd:complexType>
</xsd:element>
<xsd:complexType name="ISO15745Reference_DataType">
<xsd:sequence>
<xsd:element name="ISO15745Part" type="xsd:string" fixed="1"/>
<xsd:element name="ISO15745Edition" type="xsd:string" fixed="1"/>
<xsd:element name="ProfileTechnology" type="xsd:string" fixed="None"/>
</xsd:sequence>
</xsd:complexType>
```

```
<xsd:simpleType name="ProfileClassID_DataType">
<xsd:restriction base="xsd:string">
<xsd:enumeration value="AIP"/>
<xsd:enumeration value="Process"/>
<xsd:enumeration value="InformationExchange"/>
<xsd:enumeration value="Resource"/>
<xsd:enumeration value="Device"/>
<xsd:enumeration value="CommunicationNetwork"/>
<xsd:enumeration value="Equipment"/>
<xsd:enumeration value="Human"/>
<xsd:enumeration value="Material"/>
</xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="IASInterface_DataType">
<xsd:union>
<xsd:simpleType>
<xsd:restriction base="xsd:string">
<xsd:enumeration value="CSI"/>
<xsd:enumeration value="HCI"/>
<xsd:enumeration value="ISI"/>
<xsd:enumeration value="API"/>
<xsd:enumeration value="CMI"/>
<xsd:enumeration value="ESI"/>
<xsd:enumeration value="FSI"/>
<xsd:enumeration value="MTI"/>
<xsd:enumeration value="SEI"/>
<xsd:enumeration value="USI"/>
</xsd:restriction>
</xsd:simpleType>
<xsd:simpleType>
<xsd:restriction base="xsd:string">
<xsd:length value="4"/>
</xsd:restriction>
</xsd:simpleType>
</xsd:union>
</xsd:simpleType>
<!-- * BODY SECTION * -->
<xsd:element name="ProfileBody">
<xsd:complexType>
<xsd:sequence>
<xsd:element name="CCD" maxOccurs="unbounded">
<xsd:complexType>
<xsd:sequence>
<xsd:element name="DCD1" type="Driver01" minOccurs="0" maxOccurs="unbounded"/>
<xsd:element name="DCD2" type="Driver02" minOccurs="0" maxOccurs="unbounded"/>
</xsd:sequence>
<xsd:attribute name="category" type="Category" use="required" fixed="CCD"/>
</xsd:complexType>
</xsd:element>
</xsd:sequence>
</xsd:complexType>
</xsd:element>
</xsd:schema>
```

А.9 Пример PID-описания

А.9.1 Общие сведения

Файл SamplePIDa.xml содержит пример XML-данных для PID-описания, который позволяет использовать файл CCDa.xsd в качестве XML-схемы и определять экземпляры класса параметров согласующего устройства и драйверов устройства.

А.9.2 XML-схема: Пример файла PIDa.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<ISO15745Profile xmlns:xsi="
```

gost.e-ecolog.ru

```
xsi:noNamespaceSchemaLocation="CCDa.xsd">
<ProfileHeader>
<ProfileIdentification>ExampleOfPID</ProfileIdentification>
<ProfileRevision>1.0</ProfileRevision>
<ProfileName>Example of PID</ProfileName>
<ProfileSource>SamplePIDa.xml</ProfileSource>
<ProfileClassID>InformationExchange</ProfileClassID>
<ProfileDate>2009-03-16</ProfileDate>
<AdditionalInformation>gost.e-ecolog.ru                                                    >
<ISO15745Reference>
<ISO15745Part>1</ISO15745Part>
<ISO15745Edition>1</ISO15745Edition>
<ProfileTechnology>None</ProfileTechnology>
</ISO15745Reference>
<IASInterfaceType>CSI</IASInterfaceType>
</ProfileHeader>
<ProfileBody>
<CCD category="CCD">
<DCD1 driverVersion="1" dllPath="ndAD.dll" category="DCD">
<!-- initialize device 1-->
<myDevice01 initOrder="1" moduleId="1000" category="MODULE">
<NumOfChannel category="CREATEPARAMETER">
<Value>255</Value>
</NumOfChannel>
<!-- initialize fnADInput -->
<fnADInput funcId="1077" initOrder="2" category="INTERFACE">
<Interrupt category="CREATEPARAMETER">
<Value>5</Value>
</Interrupt>
<!-- initialize parameter Channel -->
<Channel initOrder="3" category="PARAMETER" readonly="false">
<!-- write channel several times -->
<OrderedValue initOrder="4">
<Value>0</Value>
</OrderedValue>
<OrderedValue initOrder="14">
<Value>1</Value>
</OrderedValue>
<OrderedValue initOrder="24">
<Value>2</Value>
</OrderedValue>
</Channel>
<!-- initialize the read-only attribute ADValue -->
<ADValue initOrder="3" category="ATTRIBUTE" readonly="true" infReport="true"/>
</fnADInput>
</myDevice01>
</DCD1
<DCD2 driverVersion="1" dllPath="dcd2.dll" category="DCD">
<!-- initialize device 02 -->
<myDevice02 initOrder="1" moduleId="1002" category="MODULE">
<!-- initialize myFunction02 -->
<myFunction02 funcId="1008" initOrder="2" category="INTERFACE">
<myCRPar02 category="CREATEPARAMETER">
<Value>
<speed>4800</speed>
<length>8</length>
</Value>
</myCRPar02>
<!-- initialize myOperation02 -->
<myOperation02 initOrder="3" category="OPERATION" operationId="1009">
<!-- execute the operation several times -->
<InValue category="IN">
```

```
<Input initOrder="13">
<Value>7.0</Value>
</Input>
<Input initOrder="23">
<Value>14.0</Value>
</Input>
<Input initOrder="33">
<Value>24.0</Value>
</Input>
</InValue>
</myOperation02>
</myFunction02>
</myDevice02>
</DCD2>
</CCD>
</ProfileBody>
</ISO15745Profile>
```

Приложение В
(справочное)

Шаблоны профилей возможностей устройства для промышленного применения

В.1 Общие сведения

MICX-технология позволяет определить методы совместного использования производственной информации, представляемой на языке XML и предназначенной для промышленных применений. В данном приложении приведено описание шаблона профиля зависящих от MICX-технологии возможностей служебного интерфейса как совместно используемых для информационной передачи с помощью спецификаций на характеристики программ (PPS-спецификаций) OASIS. Примеры DCD-, CCD- и PID-описаний приведены далее.

В.2 Модель профиля MICX-технологии

В.2.1 Общие сведения

Зависящая от MICX-технологии модель профиля содержит всю информацию, которая необходима для описания возможностей данного устройства и параметризации. На рисунке В.1 приведена диаграмма классов для зависящего от MICX-технологии шаблона профилей возможностей устройства.

В.2.2 Класс Device (устройство)

Этот класс характеризует возможности зависящего от MICX-технологии виртуального устройства и наследуется классом виртуальных устройств. Зависящий от устройства класс должен наследовать его свойства и определять возможности зависящего от данного устройства виртуального устройства.

В.2.3 Класс Activity (рабочая операция)

Этот класс характеризует возможности зависящей от MICX-технологии рабочей операции и наследуется классом функциональных объектов. Зависящий от рабочей операции класс должен наследовать его и определять возможности зависящей от устройства рабочей операции.

В.2.4 Класс Requester (инициатор запроса)

Этот класс характеризует возможности инициатора запроса, используемые в зависящей от MICX-технологии рабочей операции, наследуемые классом рабочих операций, обладающие несколькими классами сообщений (или не имеющие класса) и способные посылать ответные сообщения этих классов, которые могут приниматься. Инициатор запроса определен в части 2 PPS-спецификаций OASIS [3].

В.2.5 Класс Responder (респондент запроса)

Этот класс характеризует возможности респондента (ответчика) запроса, используемые в зависящей от MICX-технологии рабочей операции, наследуемые классом объектов связи, обладающие несколькими классами запрашиваемых сообщений (или не имеющие этого класса) и способные принимать ответные сообщения этих классов, которые могут посылаться. Респондент запроса определен в части 2 PPS-спецификаций OASIS [3].

В.2.6 Класс Sender (отправитель запроса)

Этот класс характеризует возможности отправителя запроса, используемые в зависящей от MICX-технологии рабочей операции, наследуемые классом объектов связи, обладающие несколькими классами уведомляющих сообщений (или не имеющие этого класса) и способные принимать ответные сообщения этих классов, которые могут посылаться. Данный класс имеет несколько классов доменных объектов (или не имеет этого класса) для записи данных в уведомляющие сообщения. Отправитель запроса определен в части 2 PPS-спецификаций OASIS [3].

В.2.7 Класс Receiver (получатель запроса)

Этот класс характеризует возможности получателя запроса, используемые в зависящей от MICX-технологии рабочей операции, наследуемые классом объектов связи, обладающие несколькими классами уведомляющих сообщений (или не имеющие этого класса) и способные принимать ответные сообщения этих классов, которые могут посылаться. Данный класс имеет несколько классов доменных объектов (или не имеет этого класса) для чтения данных из уведомляющих сообщений. Получатель запроса определен в части 2 PPS-спецификаций OASIS [3].

В.2.8 Класс Message (сообщение)

Этот класс характеризует возможности сообщения, которые позволяют зависящей от MICX-технологии рабочей операции получать и принимать его. Данный класс имеет один или несколько классов входных сообщений.

В.2.9 Класс Domain object (доменный объект)

Этот класс характеризует возможности доменного объекта, которые имеют респондер, отправитель или получатель запроса. Данный класс имеет один или несколько классов примитивных элементов.

В.2.10 Класс Transaction (транзакция)

Этот класс характеризует возможности входного сообщения, которые в зависимости от рабочей операции, используемой в MICX-технологии, способны посылать или принимать это сообщение. Данный класс содержит классы примитивных элементов. Каждый конкретный класс транзакции должен наследовать свойства абстрактного класса транзакций, определенные в части 2 PPS-спецификаций OASIS [3].

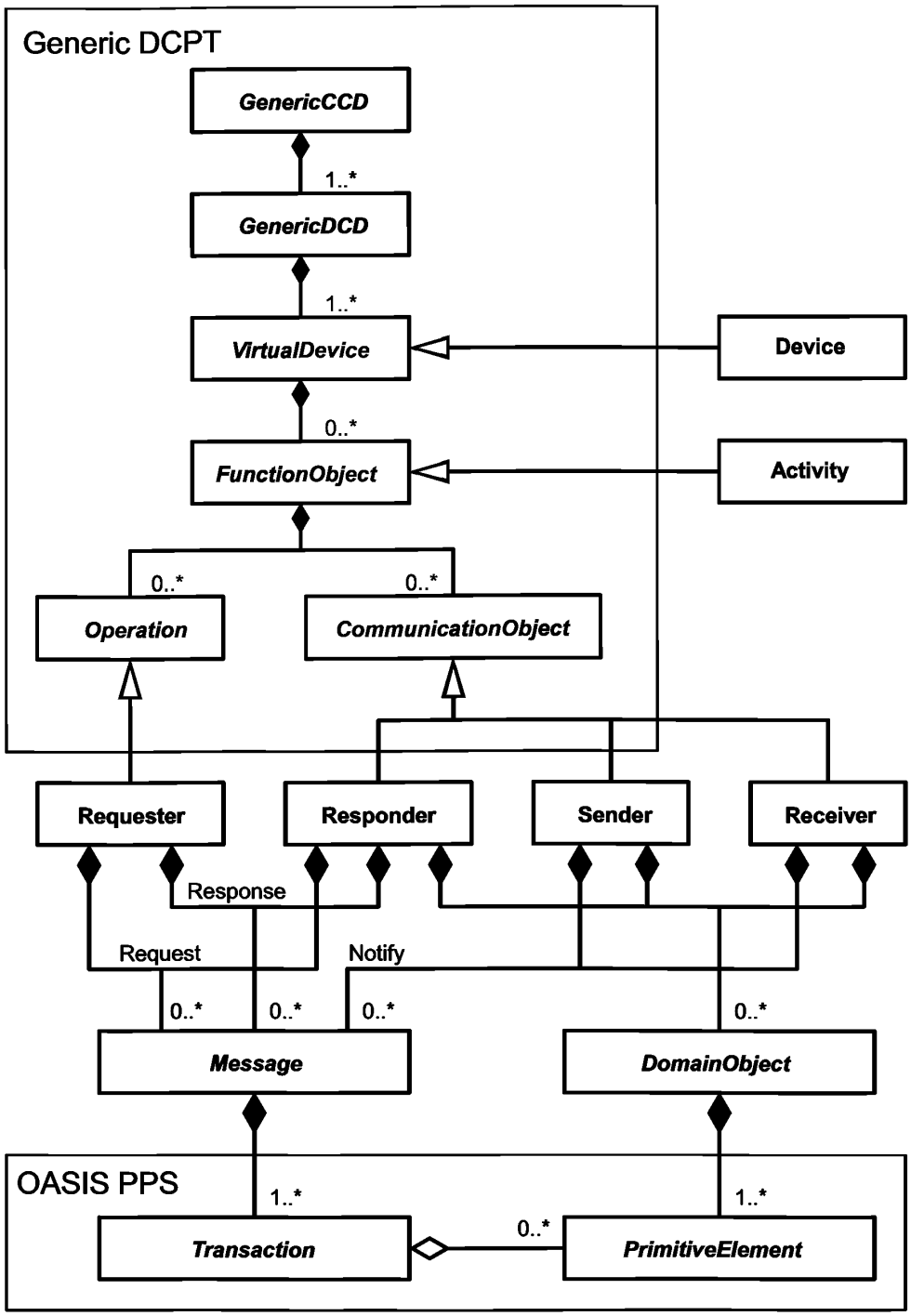


Рисунок В.1 — Диаграмма классов для зависящей от MICX-технологии модели DCPT-шаблона

B.2.11 Класс Primitive element (примитивный элемент)

Этот класс характеризует возможности OASIS PPS-примитивного элемента сообщения, которое может посылаться и приниматься с помощью зависящей от MICX-технологии рабочей операции. В OASIS PPS предусмотрено восемь типов примитивных элементов — план, заказ (порядок), сторона, задание, операция, партия, ресурс, процесс и товар. Примитивные элементы определены в части 1 PPS-спецификаций OASIS [2].

B.3 Дополнительная идентификация типа для обобщенного DCPT-шаблона

В таблице B.1 указаны зависящие от технологии типы идентификаторов объектов связи.

Т а б л и ц а B.1 — Идентификаторы типов объектов связи

Элемент зависящего от выбираемой технологии DCPT-шаблона	Содержимое XML-атрибута «category»
Responder	RESPONDER
Sender	SENDER
Receiver	RECEIVER

П р и м е ч а н и е — В MICX-технологии содержимое атрибута «category» не связано с состоянием виртуального устройства. При этом любое содержимое должно иметь тот же смысл, что и содержимое ATTRIBUTE в GDI-интерфейсе ASAM (см. приложение А).

B.4 Зависящий от выбираемой MICX-технологии шаблон профилей возможностей устройства

B.4.1 Общие сведения

Файл MICXCommon.xsd содержит XML-схему зависящего от выбираемой MICX-технологии DCPT-шаблона, которая позволяет расширять XML-схему для обобщенного DCPT-шаблона, приведенную в 6.3 в соответствии с зависящей от выбранной MICX-технологии моделью DCPT-шаблона (см. рисунок B.1). Этот файл позволяет импортировать XML-схемы в PPS-спецификации OASIS-системы (части 1 и 2).

B.4.2 XML-схема: Файл MICXCommon.xsd

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<xsd:schema xmlns:xsd="gost.e-ecolog.ru                                    "
xmlns="gost.e-ecolog.ru              )/micx/ISO20242-4/MICX"
xmlns:dcpt="gost.e-ecolog.ru                                                "
xmlns:pps="gost.e-ecolog.ru                                                 "
xmlns:ppst="gost.e-ecolog.ru              en.org/pps/ns/transaction-messages"
targetNamespace="gost.e-ecolog.ru              )/micx/ISO20242-4/MICX"
elementFormDefault="qualified">
<xsd:annotation>
<xsd:appinfo source="gost.e-ecolog.ru                                                    ">
<DCPTHeader>
<DCPTIdentification>MICXCommon</DCPTIdentification>
<DCPTRevision>1.0</DCPTRevision>
<DCPTName>MICX Specific DCPT</DCPTName>
<DCPTSource>MICXCommon.xsd</DCPTSource>
<DCPTClassID>InformationExchangeTemplate</DCPTClassID>
<DCPTDate>2009-02-25</DCPTDate>
<DCPTVender>FAOP</DCPTVender>
<TechnologyReference>
<Technology>FAOP-MICX</Technology>
<Revision>1.0</Revision>
</TechnologyReference>
</DCPTHeader>
</xsd:appinfo>
</xsd:annotation>
<!-- * Import GenericDCPT,* -->
<xsd:import namespace="gost.e-ecolog.ru                                      "
schemaLocation="GenericDCPT.xsd"/>
<!-- * Import PPS core elements,* -->
<xsd:import namespace="gost.e-ecolog.ru                                      "
schemaLocation="pps-core-elements-1.0.xsd"/>
```

```
<!-- * Import PPS transaction messages,* -->
<xsd:import namespace="gost.e-ecolog.ru
schemaLocation="pps-transaction-messages-1.0.xsd"/>
<!-- Valid content for XML-Attribut category (kind of DCD element) -->
<xsd:simpleType name="categoryType">
<xsd:restriction base="xsd:string">
<xsd:enumeration value="CCD"/>
<xsd:enumeration value="DCD"/>
<xsd:enumeration value="MODULE"/>
<xsd:enumeration value="INTERFACE"/>
<xsd:enumeration value="OPERATION"/>
<xsd:enumeration value="RESPONDER"/>
<xsd:enumeration value="SENDER"/>
<xsd:enumeration value="RECEIVER"/>
</xsd:restriction>
</xsd:simpleType>
<!-- Valid message type (kind of Messge element) -->
<xsd:simpleType name="messageType">
<xsd:restriction base="xsd:string">
<xsd:enumeration value="REQUEST"/>
<xsd:enumeration value="RESPONSE"/>
<xsd:enumeration value="NOTIFY"/>
</xsd:restriction>
</xsd:simpleType>
<!-- * Elements Declaration * -->
<xsd:element name="RequestMessage" type="MessageType" abstract="true"/>
<xsd:element name="ResponseMessage" type="MessageType" abstract="true"/>
<xsd:element name="NotifyMessage" type="MessageType" abstract="true"/>
<xsd:element name="DomainObject" type="DomainObjectType" abstract="true"/>
<xsd:element name="Transaction" type="ppst:TransactionType" abstract="true"/>
<xsd:element name="PrimitiveElement" type="pps:PrimitiveType" abstract="true"/>
<!-- * Type Definition * -->
<xsd:complexType name="DCDType">
<xsd:complexContent>
<xsd:extension base="dcpt:GenericDCDType"/>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="DeviceType">
<xsd:complexContent>
<xsd:extension base="dcpt:VirtualDeviceType"/>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="ActivityType">
<xsd:complexContent>
<xsd:extension base="dcpt:FunctionObjectType"/>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="RequesterType">
<xsd:complexContent>
<xsd:extension base="dcpt:OperationType">
<xsd:sequence>
<xsd:element ref="RequestMessage" minOccurs="0" maxOccurs="unbounded"/>
<xsd:element ref="ResponseMessage" minOccurs="0" maxOccurs="unbounded"/>
</xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="ResponderType">
<xsd:complexContent>
<xsd:extension base="dcpt:CommunicationObjectType">
<xsd:sequence>
<xsd:element ref="RequestMessage" minOccurs="0" maxOccurs="unbounded"/>
```

```
<xsd:element ref="ResponseMessage" minOccurs="0" maxOccurs="unbounded"/>
<xsd:element ref="DomainObject" minOccurs="0" maxOccurs="unbounded"/>
</xsd:sequence>
<xsd:attribute name="category" type="categoryType" use="required" fixed="RESPONDER"/>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="SenderType">
<xsd:complexContent>
<xsd:extension base="dcpt:CommunicationObjectType">
<xsd:sequence>
<xsd:element ref="NotifyMessage" minOccurs="0" maxOccurs="unbounded"/>
<xsd:element ref="DomainObject" minOccurs="0" maxOccurs="unbounded"/>
</xsd:sequence>
<xsd:attribute name="category" type="categoryType" use="required" fixed="SENDER"/>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="ReceiverType">
<xsd:complexContent>
<xsd:extension base="dcpt:CommunicationObjectType">
<xsd:sequence>
<xsd:element ref="NotifyMessage" minOccurs="0" maxOccurs="unbounded"/>
<xsd:element ref="DomainObject" minOccurs="0" maxOccurs="unbounded"/>
</xsd:sequence>
<xsd:attribute name="category" type="categoryType" use="required" fixed="RECEIVER"/>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="MessageType">
<xsd:choice minOccurs="0">
<xsd:element ref="Transaction" maxOccurs="unbounded"/>
</xsd:choice>
<xsd:attribute name="name" type="xsd:string"/>
<xsd:attribute name="message" type="messageType" use="required"/>
</xsd:complexType>
<xsd:complexType name="DomainObjectType">
<xsd:sequence minOccurs="0">
<xsd:element ref="PrimitiveElement" maxOccurs="unbounded"/>
</xsd:sequence>
</xsd:complexType>
<xsd:attribute name="name" type="xsd:string"/>
</xsd:schema>
```

B.5 Пример программы DCD-описания

B.5.1 Общие сведения

Файл DCDb.xsd содержит пример XML-схемы DCD-описания, которая позволяет дополнять XML-схему для зависящего от выбранной MICX-технологии DCPT -шаблона (см. B.4) в соответствии с возможностями драйвера устройства.

B.5.2 XML-схема: Файл DCDb.xsd

```
<?xml version="1.0" encoding="utf-8"?>
<xsd:schema xmlns:xsd="gost.e-ecolog.ru                                    "
xmlns="gost.e-ecolog.ru                                                    "
xmlns:micx="gost.e-ecolog.ru              )/micx/ISO20242-4/MICX"
xmlns:pps="gost.e-ecolog.ru               en.org/pps/ns/core-elements"
xmlns:ppst="gost.e-ecolog.ru               en.org/pps/ns/transaction-messages"
targetNamespace="gost.e-ecolog.ru                                                  "
elementFormDefault="qualified">
<xsd:annotation>
<xsd:appinfo source="gost.e-ecolog.ru                                                  ">
<DCPTHeader>
<DCPTIdentification>DCD</DCPTIdentification>
```

```
<DCPTRevision>1.0</DCPTRevision>
<DCPTName>DCD Example</DCPTName>
<DCPTSource>DCDb.xsd</DCPTSource>
<DCPTClassID>InformationExchangeTemplate</DCPTClassID>
<DCPTDate>2009-02-25</DCPTDate>
<DCPTVender>FAOP</DCPTVender>
<TechnologyReference>
<Technology>FAOP-MICX</Technology>
<Revision>1.0</Revision>
</TechnologyReference>
</DCPTHeader>
</xsd:appinfo>
</xsd:annotation>
<!-- * Import MICXCommon.xsd,* -->
<xsd:import namespace="gost.e-ecolog.ru                                    "
schemaLocation="MICXCommon.xsd"/>
<!-- * Import PPS core elements,* -->
<xsd:import namespace="gost.e-ecolog.ru                                    "
schemaLocation="pps-core-elements-1.0.xsd"/>
<!-- * Import PPS transaction messages,* -->
<xsd:import namespace="gost.e-ecolog.ru                                    "
schemaLocation="pps-transaction-messages-1.0.xsd"/>
<!-- * Elements Declaration * -->
<xsd:element name="Device" type="DeviceType"/>
<xsd:element name="ProductionControl" type="ProductionControlType"/>
<xsd:element name="LoadRecipe" type="LoadRecipeType"/>
<xsd:element name="ExecutionControl" type="ExecutionControlType"/>
<xsd:element name="GetInformation" type="GetInformationType"/>
<xsd:element name="EventReport" type="EventReportType"/>
<!-- * DCD Type Definition * -->
<xsd:complexType name="DCDType">
<xsd:sequence>
<xsd:element ref="Device" minOccurs="0" maxOccurs="unbounded"/>
</xsd:sequence>
<xsd:attribute name="name" type="xsd:string"/>
<xsd:attribute name="category" type="micx:categoryType" use="required" fixed="DCD"/>
</xsd:complexType>
<!-- * Device Type Definition * -->
<xsd:complexType name="DeviceType">
<xsd:sequence>
<xsd:element ref="ProductionControl" minOccurs="0" maxOccurs="unbounded"/>
</xsd:sequence>
<xsd:attribute name="name" type="xsd:string"/>
<xsd:attribute name="category" type="micx:categoryType" use="required" fixed="MODULE"/>
</xsd:complexType>
<!-- * ProductionControl(Activity) Type Definition * -->
<xsd:complexType name="ProductionControlType">
<xsd:sequence>
<xsd:element ref="LoadRecipe" minOccurs="0" maxOccurs="unbounded"/>
<xsd:element ref="ExecutionControl" minOccurs="0" maxOccurs="unbounded"/>
<xsd:element ref="GetInformation" minOccurs="0" maxOccurs="unbounded"/>
<xsd:element ref="EventReport" minOccurs="0" maxOccurs="unbounded"/>
</xsd:sequence>
<xsd:attribute name="name" type="xsd:string"/>
<xsd:attribute name="category" type="micx:categoryType" use="required" fixed="INTERFACE"/>
</xsd:complexType>
<!-- * LoadRecipe(Requester) Type Definition * -->
<xsd:complexType name="LoadRecipeType">
<xsd:sequence>
<xsd:element ref="LoadRecipeRequest" minOccurs="0" maxOccurs="unbounded"/>
<xsd:element ref="LoadRecipeResponse" minOccurs="0" maxOccurs="unbounded"/>
</xsd:sequence>
```

```
<xsd:attribute name="category" type="micx:categoryType" use="required" fixed="OPERATION"/>
</xsd:complexType>
<!-- * ExecutionControl(Requester) Type Definition * -->
<xsd:complexType name="ExecutionControlType">
<xsd:sequence>
<xsd:element ref="ExecutionRequest" minOccurs="0" maxOccurs="unbounded"/>
<xsd:element ref="ExecutionResponse" minOccurs="0" maxOccurs="unbounded"/>
</xsd:sequence>
<xsd:attribute name="name" type="xsd:string"/>
<xsd:attribute name="category" type="micx:categoryType" use="required" fixed="OPERATION"/>
</xsd:complexType>
<!-- * GetInformation(Requester) Type Definition * -->
<xsd:complexType name="GetInformationType">
<xsd:sequence>
<xsd:element ref="GetInformationRequest" minOccurs="0" maxOccurs="unbounded"/>
<xsd:element ref="GetInformationResponse" minOccurs="0" maxOccurs="unbounded"/>
</xsd:sequence>
<xsd:attribute name="name" type="xsd:string"/>
<xsd:attribute name="category" type="micx:categoryType"
use=»required» fixed="OPERATION"/>
</xsd:complexType>
<!-- * EventReportType(Receiver) Type Definition * -->
<xsd:complexType name="EventReportType">
<xsd:sequence>
<xsd:element ref="NotifyEvent" minOccurs="0" maxOccurs="unbounded"/>
<xsd:element ref="EventObject" minOccurs="0" maxOccurs="unbounded"/>
</xsd:sequence>
<xsd:attribute name="name" type="xsd:string"/>
<xsd:attribute name="category" type="micx:categoryType"
use=»required» fixed="RECEIVER"/>
</xsd:complexType>
<!-- * EventObject Type Definition * -->
<xsd:element name="EventObject">
<xsd:complexType>
<xsd:sequence>
<xsd:element name="EquipmentObject" type="EquipmentType"
minOccurs="0" maxOccurs="unbounded"/>
<xsd:element name="PersonnelObject" type="PersonnelType"
minOccurs="0" maxOccurs="unbounded"/>
<xsd:element name="ExecutionObject" type="ExecutionType"
minOccurs="0" maxOccurs="unbounded"/>
</xsd:sequence>
<xsd:attribute name="name" type="xsd:string"/>
</xsd:complexType>
</xsd:element>
<!-- Definition Type of Equipment elements extended on primitive element -->
<!-- Equipment element -->
<xsd:element name="Equipment" type="EquipmentType" substitutionGroup="pps:Resource"/>
<!-- Equipment Type -->
<xsd:complexType name="EquipmentType">
<xsd:complexContent>
<xsd:restriction base="pps:PrimitiveType">
<xsd:sequence>
<xsd:element ref="pps:Capacity" minOccurs="0" maxOccurs="unbounded"/>
<xsd:element ref="pps:Spec" minOccurs="0" maxOccurs="unbounded"/>
<xsd:element ref="pps:Event" minOccurs="0" maxOccurs="unbounded"/>
<xsd:element ref="pps:Description" minOccurs="0" maxOccurs="unbounded"/>
<xsd:element ref="pps:Author" minOccurs="0" maxOccurs="unbounded"/>
<xsd:element ref="pps:Date» minOccurs="0" maxOccurs="unbounded"/>
</xsd:sequence>
<xsd:attribute name="id" type="xsd:string" use="required"/>
<xsd:attribute name="key" type="xsd:long"/>
```

```
<xsd:attribute name="name" type="xsd:string" fixed="Equipment"/>
<xsd:attribute name="type" type="xsd:string"/>
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
<!-- Definition Type of Personnel element extended on primitive element -->
<!-- Personnel element -->
<xsd:element name="Personnel" type="PersonnelType" substitutionGroup="pps:Resource"/>
<!-- Personnel Type -->
<xsd:complexType name="PersonnelType">
<xsd:complexContent>
<xsd:restriction base="pps:PrimitiveType">
<xsd:sequence>
<xsd:element ref="pps:Capacity" minOccurs="0" maxOccurs="unbounded"/>
<xsd:element ref="pps:Spec" minOccurs="0" maxOccurs="unbounded"/>
<xsd:element ref="pps:Event" minOccurs="0" maxOccurs="unbounded"/>
<xsd:element ref="pps:Description" minOccurs="0" maxOccurs="unbounded"/>
<xsd:element ref="pps:Author" minOccurs="0" maxOccurs="unbounded"/>
<xsd:element ref="pps:Date" minOccurs="0" maxOccurs="unbounded"/>
</xsd:sequence>
<xsd:attribute name="id" type="xsd:string" use="required"/>
<xsd:attribute name="key" type="xsd:long"/>
<xsd:attribute name="name" type="xsd:string" fixed="Personnel"/>
<xsd:attribute name="type" type="xsd:string"/>
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
<!-- Definition Type of Recipe element extended on primitive element -->
<!-- Recipe element -->
<xsd:element name="Recipe" type="RecipeType" substitutionGroup="pps:Process"/>
<!-- customer Type -->
<xsd:complexType name="RecipeType">
<xsd:complexContent>
<xsd:restriction base="pps:PrimitiveType">
<xsd:sequence>
<xsd:element ref="pps:Produce" minOccurs="0" maxOccurs="unbounded"/>
<xsd:element ref="pps:Consume" minOccurs="0" maxOccurs="unbounded"/>
<xsd:element ref="pps:Assign" minOccurs="0" maxOccurs="unbounded"/>
<xsd:element ref="pps:Spec" minOccurs="0" maxOccurs="unbounded"/>
<xsd:element ref="pps:Description" minOccurs="0" maxOccurs="unbounded"/>
<xsd:element ref="pps:Author" minOccurs="0" maxOccurs="unbounded"/>
<xsd:element ref="pps:Date" minOccurs="0" maxOccurs="unbounded"/>
</xsd:sequence>
<xsd:attribute name="id" type="xsd:string" use="required"/>
<xsd:attribute name="key» type="xsd:long"/>
<xsd:attribute name="name" type="xsd:string"/>
<xsd:attribute name="type" type="xsd:string" fixed="Recipe"/>
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
<!-- Definition Type of Work element extended on primitive element -->
<!-- Work element -->
<xsd:element name="Work" type="WorkType" substitutionGroup="pps:Item"/>
<!-- customer Type -->
<xsd:complexType name="WorkType">
<xsd:complexContent>
<xsd:restriction base="pps:PrimitiveType">
<xsd:sequence>
<xsd:element ref="pps:Produce" minOccurs="0" maxOccurs="unbounded"/>
<xsd:element ref="pps:Consume" minOccurs="0" maxOccurs="unbounded"/>
<xsd:element ref="pps:Spec" minOccurs="0" maxOccurs="unbounded"/>
<xsd:element ref="pps:Description" minOccurs="0" maxOccurs="unbounded"/>
```

```
<xsd:element ref="pps:Author" minOccurs="0" maxOccurs="unbounded"/>
<xsd:element ref="pps:Date" minOccurs="0" maxOccurs="unbounded"/>
</xsd:sequence>
<xsd:attribute name="id" type="xsd:string" use="required"/>
<xsd:attribute name="key" type="xsd:long"/>
<xsd:attribute name="name" type="xsd:string"/>
<xsd:attribute name="parent" type="xsd:string"/>
<xsd:attribute name="type" type="xsd:string" fixed="Work"/>
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
<!-- Definition Type of Execution Order element extended on primitive element -->
<!-- Execution element -->
<xsd:element name="Execution" type="ExecutionType" substitutionGroup="pps:Operation"/>
<!-- Execution Type -->
<xsd:complexType name="ExecutionType">
<xsd:complexContent>
<xsd:restriction base="pps:PrimitiveType">
<xsd:sequence>
<xsd:element ref="pps:Produce" minOccurs="0" maxOccurs="unbounded"/>
<xsd:element ref="pps:Consume" minOccurs="0" maxOccurs="unbounded"/>
<xsd:element ref="pps:Assign" minOccurs="0" maxOccurs="unbounded"/>
<xsd:element ref="pps:Progress" minOccurs="0" maxOccurs="unbounded"/>
<xsd:element ref="pps:Spec" minOccurs="0" maxOccurs="unbounded"/>
<xsd:element ref="pps:Start" minOccurs="0" maxOccurs="unbounded"/>
<xsd:element ref="pps:End" minOccurs="0" maxOccurs="unbounded"/>
<xsd:element ref="pps:Event" minOccurs="0" maxOccurs="unbounded"/>
<xsd:element ref="pps:Description" minOccurs="0" maxOccurs="unbounded"/>
<xsd:element ref="pps:Author" minOccurs="0" maxOccurs="unbounded"/>
<xsd:element ref="pps:Date" minOccurs="0" maxOccurs="unbounded"/>
</xsd:sequence>
<xsd:attribute name="id" type="xsd:string" use="required"/>
<xsd:attribute name="key" type="xsd:long"/>
<xsd:attribute name="name" type="xsd:string"/>
<xsd:attribute name="type" type="xsd:string" fixed="Operation"/>
<xsd:attribute name="status" type="xsd:string"/>
<xsd:attribute name="process" type="xsd:string"/>
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
<!-- Definition Type of Accepted Work element extended on relational element -->
<!--AcceptedWork element -->
<xsd:element name="AcceptedWork" type="AcceptedWorkType"
substitutionGroup="pps:Produce"/>
<!--AcceptedWork Type -->
<xsd:complexType name="AcceptedWorkType">
<xsd:complexContent>
<xsd:restriction base="pps:RelationalType">
<xsd:sequence>
<xsd:element ref="pps:Qty" minOccurs="0" maxOccurs="unbounded"/>
</xsd:sequence>
<xsd:attribute name="type" type="xsd:string" fixed="Accepted"/>
<xsd:attribute name="lot" type="xsd:string"/>
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
<!-- Definition Type of DefectiveWork element extended on relational element -->
<!-- DefectiveWork element -->
<xsd:element name="DefectiveWork" type="DefectiveWorkType"
substitutionGroup="pps:Produce"/>
<!--DefectiveWork Type -->
<xsd:complexType name="DefectiveWorkType">
```

```
<xsd:complexContent>
<xsd:restriction base="pps:RelationalType">
<xsd:sequence>
<xsd:element ref="pps:Qty" minOccurs="0" maxOccurs="unbounded"/>
</xsd:sequence>
<xsd:attribute name="type" type="xsd:string" fixed="Defective"/>
<xsd:attribute name="lot" type="xsd:string"/>
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
<!--Definition Type of Assigned Equipment element extended on relational element -->
<!--AssignedEquipment element -->
<xsd:element name="AssignedEquipment" type="AssignedEquipmentType" substitutionGroup="pps:Assign"/>
<!--AssignedEquipment Type -->
<xsd:complexType name="AssignedEquipmentType">
<xsd:complexContent>
<xsd:restriction base="pps:RelationalType">
<xsd:sequence>
<xsd:element ref="pps:Qty" minOccurs="0" maxOccurs="unbounded"/>
</xsd:sequence>
<xsd:attribute name="type" type="xsd:string" fixed="Equipment"/>
<xsd:attribute name="resource" type="xsd:string"/>
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
<!--Definition Type of OrderName element extended on specific element -->
<!--OrderName element -->
<xsd:element name="OrderName" type="OrderNameType" substitutionGroup="pps:Spec"/>
<!--OrderName Type -->
<xsd:complexType name="OrderNameType">
<xsd:complexContent>
<xsd:restriction base="pps:SpecificType">
<xsd:sequence>
<xsd:element ref="pps:Char" minOccurs="0" maxOccurs="unbounded"/>
</xsd:sequence>
<xsd:attribute name="type" type="xsd:string" fixed="Order"/>
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
<!--Definition Type of Length element extended on specific element -->
<!-- Length element -->
<xsd:element name="Length" type="LengthType" substitutionGroup="pps:Spec"/>
<xsd:element name="Qty" substitutionGroup="pps:Qty"/>
<!-- Length Type -->
<xsd:complexType name="LengthType">
<xsd:complexContent>
<xsd:restriction base="pps:SpecificType">
<xsd:sequence>
<xsd:element ref="pps:Qty" minOccurs="0" maxOccurs="unbounded"/>
</xsd:sequence>
<xsd:attribute name="type" type="xsd:string" fixed="Length"/>
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
<!--Message -->
<!--Definition Request Message of LoadRecipe -->
<xsd:element name="LoadRecipeRequest">
<xsd:complexType>
<xsd:complexContent>
<xsd:extension base="micx:MessageType">
<xsd:sequence>
<xsd:element name="RecipeRecord">
```

```
<xsd:complexType>
<xsd:complexContent>
<xsd:restriction base="ppst:TransactionType">
<xsd:sequence>
<xsd:element ref="ppst:App" minOccurs="0"/>
<xsd:element ref="ppst:Condition" minOccurs="0" maxOccurs="unbounded"/>
<xsd:choice minOccurs="0">
<xsd:element ref="pps:Item" minOccurs="0" maxOccurs="unbounded"/>
<xsd:element ref="pps:Process" minOccurs="0" maxOccurs="unbounded"/>
</xsd:choice>
</xsd:sequence>
<xsd:attribute name="id" type="xsd:string" use="required"/>
<xsd:attribute name="action" type="xsd:string" fixed="Add"/>
<xsd:attribute name="transaction" type="xsd:string"/>
<xsd:attribute name="profile" type="xsd:string"/>
<xsd:attribute name="confirm" type="xsd:string"/>
<xsd:attribute name="create" type="xsd:dateTime"/>
<xsd:attribute name="sender" type="xsd:string"/>
<xsd:attribute name="description" type="xsd:string"/>
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
</xsd:element>
</xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
</xsd:element>
<!--Definition Response Message of LoadRecipe -->
<xsd:element name="LoadRecipeResponse">
<xsd:complexType>
<xsd:complexContent>
<xsd:extension base="micx:MessageType">
<xsd:sequence>
<xsd:element name="RecipeRecord">
<xsd:complexType>
<xsd:complexContent>
<xsd:restriction base="ppst:TransactionType">
<xsd:sequence>
<xsd:element ref="ppst:Error" minOccurs="0" maxOccurs="unbounded"/>
<xsd:element ref="ppst:App" minOccurs="0"/>
<xsd:choice minOccurs="0">
<xsd:element ref="pps:Item" minOccurs="0" maxOccurs="unbounded"/>
<xsd:element ref="pps:Process" minOccurs="0" maxOccurs="unbounded"/>
</xsd:choice>
</xsd:sequence>
<xsd:attribute name="id" type="xsd:string" use="required"/>
<xsd:attribute name="action" type="xsd:string" fixed="Confirm"/>
<xsd:attribute name="transaction" type="xsd:string"/>
<xsd:attribute name="profile" type="xsd:string"/>
<xsd:attribute name="create" type="xsd:dateTime"/>
<xsd:attribute name="sender" type="xsd:string"/>
<xsd:attribute name="description" type="xsd:string"/>
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
</xsd:element>
</xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
</xsd:element>
```

```
<!--Definition Request Message of Execution –>
<xsd:element name="ExecutionRequest">
<xsd:complexType>
<xsd:complexContent>
<xsd:extension base="micx:MessageType">
<xsd:sequence>
<xsd:element name="ExecutionOrder">
<xsd:complexType>
<xsd:complexContent>
<xsd:restriction base="ppst:TransactionType">
<xsd:sequence>
<xsd:element ref="ppst:App" minOccurs="0"/>
<xsd:element ref="ppst:Condition" minOccurs="0" maxOccurs="unbounded"/>
<xsd:choice minOccurs="0">
<xsd:element ref="pps:Operation" maxOccurs="unbounded"/>
</xsd:choice>
</xsd:sequence>
<xsd:attribute name="id" type="xsd:string" use="required"/>
<xsd:attribute name="action" type="xsd:string" fixed="Add"/>
<xsd:attribute name="transaction" type="xsd:string"/>
<xsd:attribute name="profile" type="xsd:string"/>
<xsd:attribute name="confirm" type="xsd:string"/>
<xsd:attribute name="create" type="xsd:dateTime"/>
<xsd:attribute name="sender" type="xsd:string"/>
<xsd:attribute name="description" type="xsd:string"/>
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
</xsd:element>
</xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
</xsd:element>
<!--Definition Response Message of Execution -->
<xsd:element name="ExecutionResponse">
<xsd:complexType>
<xsd:complexContent>
<xsd:extension base="micx:MessageType">
<xsd:sequence>
<xsd:element name="ExecutionOrder">
<xsd:complexType>
<xsd:complexContent>
<xsd:restriction base="ppst:TransactionType">
<xsd:sequence>
<xsd:element ref="ppst:Error" minOccurs="0" maxOccurs="unbounded"/>
<xsd:element ref="ppst:App" minOccurs="0"/>
<xsd:choice>
<xsd:choice minOccurs="0">
<xsd:element ref="pps:Operation" minOccurs="0" maxOccurs="unbounded"/>
</xsd:choice>
</xsd:choice>
</xsd:sequence>
<xsd:attribute name="id" type="xsd:string" use="required"/>
<xsd:attribute name="action» type="xsd:string" fixed="Confirm"/>
<xsd:attribute name="transaction" type="xsd:string"/>
<xsd:attribute name="profile" type="xsd:string"/>
<xsd:attribute name="create" type="xsd:dateTime"/>
<xsd:attribute name="sender" type="xsd:string"/>
<xsd:attribute name="description" type="xsd:string"/>
</xsd:restriction>
</xsd:complexContent>
```

```
</xsd:complexType>
</xsd:element>
</xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
</xsd:element>
<!--Definition Request Message of GetInformation -->
<xsd:element name="GetInformationRequest">
<xsd:complexType>
<xsd:complexContent>
<xsd:extension base="micx:MessageType">
<xsd:choice>
<xsd:element name="ExecutionOrder">
<xsd:complexType>
<xsd:complexContent>
<xsd:restriction base="ppst:TransactionType">
<xsd:sequence>
<xsd:element ref="ppst:App" minOccurs="0"/>
<xsd:element ref="ppst:Condition" minOccurs="0" maxOccurs="unbounded"/>
<xsd:element ref="ppst:Selection" maxOccurs="unbounded"/>
<xsd:element ref="ppst:Header" minOccurs="0"/>
<xsd:choice minOccurs="0">
<xsd:element ref="pps:Operation" minOccurs="0" maxOccurs="unbounded"/>
</xsd:choice>
</xsd:sequence>
<xsd:attribute name="id" type="xsd:string" use="required"/>
<xsd:attribute name="action" type="xsd:string" fixed="Get"/>
<xsd:attribute name="transaction" type="xsd:string"/>
<xsd:attribute name="profile" type="xsd:string"/>
<xsd:attribute name="create" type="xsd:dateTime"/>
<xsd:attribute name="sender" type="xsd:string"/>
<xsd:attribute name="description" type="xsd:string"/>
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
</xsd:element>
<xsd:element name="ResourceRecord">
<xsd:complexType>
<xsd:complexContent>
<xsd:restriction base="ppst:TransactionType">
<xsd:sequence>
<xsd:element ref="ppst:App" minOccurs="0"/>
<xsd:element ref="ppst:Condition" minOccurs="0" maxOccurs="unbounded"/>
<xsd:choice minOccurs="0">
<xsd:element ref="pps:Resource" maxOccurs="unbounded"/>
</xsd:choice>
</xsd:sequence>
<xsd:attribute name="id" type="xsd:string" use="required"/>
<xsd:attribute name="action" type="xsd:string"/>
<xsd:attribute name="transaction" type="xsd:string" fixed="Get"/>
<xsd:attribute name="profile" type="xsd:string"/>
<xsd:attribute name="confirm" type="xsd:string"/>
<xsd:attribute name="create" type="xsd:dateTime"/>
<xsd:attribute name="sender" type="xsd:string"/>
<xsd:attribute name="description" type="xsd:string"/>
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
</xsd:element>
<xsd:element name="InventryRecord">
<xsd:complexType>
```

```
<xsd:complexContent>
<xsd:restriction base="ppst:TransactionType">
<xsd:sequence>
<xsd:element ref="ppst:App" minOccurs="0"/>
<xsd:element ref="ppst:Condition" minOccurs="0" maxOccurs="unbounded"/>
<xsd:choice minOccurs="0">
<xsd:element ref="pps:Item" maxOccurs="unbounded"/>
</xsd:choice>
</xsd:sequence>
<xsd:attribute name="id" type="xsd:string" use="required"/>
<xsd:attribute name="action" type="xsd:string"/>
<xsd:attribute name="transaction" type="xsd:string" fixed="Get"/>
<xsd:attribute name="profile" type="xsd:string"/>
<xsd:attribute name="confirm" type="xsd:string"/>
<xsd:attribute name="create" type="xsd:dateTime"/>
<xsd:attribute name="sender" type="xsd:string"/>
<xsd:attribute name="description" type="xsd:string"/>
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
</xsd:element>
</xsd:choice>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
</xsd:element>
<!--Definition Response Message GetInformation -->
<xsd:element name="GetInformationResponse">
<xsd:complexType>
<xsd:complexContent>
<xsd:extension base="micx:MessageType">
<xsd:choice>
<xsd:element name="ExecutionOrder">
<xsd:complexType>
<xsd:complexContent>
<xsd:restriction base="ppst:TransactionType">
<xsd:sequence>
<xsd:element ref="ppst:Error" minOccurs="0" maxOccurs="unbounded"/>
<xsd:element ref="ppst:App" minOccurs="0"/>
<xsd:element ref="ppst:Header" minOccurs="0"/>
<xsd:choice minOccurs="0">
<xsd:element ref="pps:Operation" minOccurs="0" maxOccurs="unbounded"/>
</xsd:choice>
</xsd:sequence>
<xsd:attribute name="id" type="xsd:string" use="required"/>
<xsd:attribute name="action" type="xsd:string" fixed="Show"/>
<xsd:attribute name="transaction" type="xsd:string"/>
<xsd:attribute name="profile" type="xsd:string"/>
<xsd:attribute name="create" type="xsd:dateTime"/>
<xsd:attribute name="sender" type="xsd:string"/>
<xsd:attribute name="description" type="xsd:string"/>
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
</xsd:element>
<xsd:element name="ResourceRecord">
<xsd:complexType>
<xsd:complexContent>
<xsd:restriction base="ppst:TransactionType">
<xsd:sequence>
<xsd:element ref="ppst:Error" minOccurs="0" maxOccurs="unbounded"/>
<xsd:element ref="ppst:App" minOccurs="0"/>
```

```
<xsd:element ref="ppst:Header" minOccurs="0"/>
<xsd:choice minOccurs="0">
<xsd:element ref="pps:Resource" maxOccurs="unbounded"/>
</xsd:choice>
</xsd:sequence>
<xsd:attribute name="id" type="xsd:string" use="required"/>
<xsd:attribute name="action" type="xsd:string"/>
<xsd:attribute name="transaction" type="xsd:string" fixed="Show"/>
<xsd:attribute name="profile" type="xsd:string"/>
<xsd:attribute name="create" type="xsd:dateTime"/>
<xsd:attribute name="sender" type="xsd:string"/>
<xsd:attribute name="description" type="xsd:string"/>
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
</xsd:element>
<xsd:element name="InventryRecord">
<xsd:complexType>
<xsd:complexContent>
<xsd:restriction base="ppst:TransactionType">
<xsd:sequence>
<xsd:element ref="ppst:Error" minOccurs="0" maxOccurs="unbounded"/>
<xsd:element ref="ppst:App" minOccurs="0"/>
<xsd:element ref="ppst:Header" minOccurs="0"/>
<xsd:choice minOccurs="0">
<xsd:element ref="pps:Item" maxOccurs="unbounded"/>
</xsd:choice>
</xsd:sequence>
<xsd:attribute name="id" type="xsd:string" use="required"/>
<xsd:attribute name="action" type="xsd:string"/>
<xsd:attribute name="transaction" type="xsd:string" fixed="Show"/>
<xsd:attribute name="profile" type="xsd:string"/>
<xsd:attribute name="create" type="xsd:dateTime"/>
<xsd:attribute name="sender" type="xsd:string"/>
<xsd:attribute name="description" type="xsd:string"/>
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
</xsd:element>
</xsd:choice>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
</xsd:element>
<!--Definition Notify Message of NotifyEvent -->
<xsd:element name="NotifyEvent">
<xsd:complexType>
<xsd:complexContent>
<xsd:extension base="micx:MessageType">
<xsd:choice>
<xsd:element name="ExecutionOrder">
<xsd:complexType>
<xsd:complexContent>
<xsd:restriction base="ppst:TransactionType">
<xsd:sequence>
<xsd:element ref="ppst:App" minOccurs="0"/>
<xsd:element ref="ppst:Header" minOccurs="0"/>
<xsd:choice minOccurs="0">
<xsd:element ref="pps:Operation" maxOccurs="unbounded"/>
</xsd:choice>
</xsd:sequence>
<xsd:attribute name="id" type="xsd:string" use="required"/>
```

```
<xsd:attribute name="action" type="xsd:string" fixed="Notify"/>
<xsd:attribute name="transaction" type="xsd:string"/>
<xsd:attribute name="profile" type="xsd:string"/>
<xsd:attribute name="create" type="xsd:dateTime"/>
<xsd:attribute name="sender" type="xsd:string"/>
<xsd:attribute name="description" type="xsd:string"/>
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
</xsd:element>
<xsd:element name="ResourceRecord">
<xsd:complexType>
<xsd:complexContent>
<xsd:restriction base="ppst:TransactionType">
<xsd:sequence>
<xsd:element ref="ppst:App" minOccurs="0"/>
<xsd:element ref="ppst:Header" minOccurs="0"/>
<xsd:choice minOccurs="0">
<xsd:element ref="pps:Resource" maxOccurs="unbounded"/>
</xsd:choice>
</xsd:sequence>
<xsd:attribute name="id" type="xsd:string" use="required"/>
<xsd:attribute name="action" type="xsd:string"/>
<xsd:attribute name="transaction" type="xsd:string" fixed="Notify"/>
<xsd:attribute name="profile" type="xsd:string"/>
<xsd:attribute name="create" type="xsd:dateTime"/>
<xsd:attribute name="sender" type="xsd:string"/>
<xsd:attribute name="description" type="xsd:string"/>
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
</xsd:element>
</xsd:choice>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
</xsd:element>
</xsd:schema>
```

B.6 Пример CCD-описания

B.6.1 Общие сведения

Файл CCDb.xsd содержит XML-схему примера CCD-описания, которая позволяет дополнять XML-схему для зависящего от выбираемой MICX-технологии DCPT-шаблона (см. B.4) в соответствии с возможностями согласующего устройства, импортирующего DCD-описание драйверов устройства.

B.6.2 XML-схема: Файл CCDb.xsd

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<xsd:schema xmlns:xsd="gost.e-ecolog.ru
xmlns="gost.e-ecolog.ru
xmlns:micx="gost.e-ecolog.ru )/micx/ISO20242-4/MICX"
targetNamespace="gost.e-ecolog.ru
elementFormDefault="qualified">
<xsd:annotation>
<xsd:appinfo source="gost.e-ecolog.ru ">
<DCPTHeader>
<DCPTIdentification>CCD</DCPTIdentification>
<DCPTRevision>1.0</DCPTRevision>
<DCPTName>CCD Sample</DCPTName>
<DCPTSource>CCDb.xsd</DCPTSource>
<DCPTClassID>InformationExchangeTemplate</DCPTClassID>
<DCPTDate>2009-02-25</DCPTDate>
<DCPTVender>FAOP</DCPTVender>
<TechnologyReference>
```

```
<Technology>FAOP-MICX</Technology>
<Revision>1.0</Revision>
</TechnologyReference>
</DCPTHeader>
</xsd:appinfo>
</xsd:annotation>
<!-- * Import MICXCommon.xsd,* -->
<xsd:import namespace="gost.e-ecolog.ru "
schemaLocation="MICXCommon.xsd"/>
<xsd:include schemaLocation="DCDb.xsd"/>
<!-- * ISO15745 Profile Root * -->
<xsd:element name="ISO15745Profile">
<xsd:complexType>
<xsd:sequence>
<xsd:element ref="ProfileHeader"/>
<xsd:element ref="ProfileBody"/>
</xsd:sequence>
</xsd:complexType>
</xsd:element>
<!-- * HEADER DATA TYPES * -->
<xsd:element name="ProfileHeader">
<xsd:complexType>
<xsd:sequence>
<xsd:element name="ProfileIdentification" type="xsd:string"/>
<xsd:element name="ProfileRevision" type="xsd:string"/>
<xsd:element name="ProfileName" type="xsd:string"/>
<xsd:element name="ProfileSource" type="xsd:string"/>
<xsd:element name="ProfileClassID" type="ProfileClassID_DataType"
fixed="InformationExchange"/>
<xsd:element name="ProfileDate" type="xsd:date" minOccurs="0"/>
<xsd:element name="AdditionalInformation" type="xsd:anyURI" minOccurs="0"/>
<xsd:element name="ISO15745Reference" type="ISO15745Reference_DataType"/>
<xsd:element name="IASInterfaceType" type="IASInterface_DataType" fixed="CSI"
minOccurs="0" maxOccurs="unbounded"/>
</xsd:sequence>
</xsd:complexType>
</xsd:element>
<xsd:complexType name="ISO15745Reference_DataType">
<xsd:sequence>
<xsd:element name="ISO15745Part" type="xsd:string" fixed="1"/>
<xsd:element name="ISO15745Edition" type="xsd:string" fixed="1"/>
<xsd:element name="ProfileTechnology" type="xsd:string" fixed="None"/>
</xsd:sequence>
</xsd:complexType>
<xsd:simpleType name="ProfileClassID_DataType">
<xsd:restriction base="xsd:string">
<xsd:enumeration value="AIP"/>
<xsd:enumeration value="Process"/>
<xsd:enumeration value="InformationExchange"/>
<xsd:enumeration value="Resource"/>
<xsd:enumeration value="Device"/>
<xsd:enumeration value="CommunicationNetwork"/>
<xsd:enumeration value="Equipment"/>
<xsd:enumeration value="Human"/>
<xsd:enumeration value="Material"/>
</xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="IASInterface_DataType">
<xsd:union>
<xsd:simpleType>
<xsd:restriction base="xsd:string">
<xsd:enumeration value="CSI"/>
<xsd:enumeration value="HCI"/>
```

```
<xsd:enumeration value="ISI"/>
<xsd:enumeration value="API"/>
<xsd:enumeration value="CMI"/>
<xsd:enumeration value="ESI"/>
<xsd:enumeration value="FSI"/>
<xsd:enumeration value="MTI"/>
<xsd:enumeration value="SEI"/>
<xsd:enumeration value="USI"/>
</xsd:restriction>
</xsd:simpleType>
<xsd:simpleType>
<xsd:restriction base="xsd:string">
<xsd:length value="4"/>
</xsd:restriction>
</xsd:simpleType>
</xsd:union>
</xsd:simpleType>
<!-- * BODY SECTION * -->
<xsd:element name="ProfileBody">
<xsd:complexType>
<xsd:sequence>
<xsd:element name="CCD" maxOccurs="unbounded">
<xsd:complexType>
<xsd:sequence>
<xsd:element name="DCD1" type="DCDType" minOccurs="0" maxOccurs="unbounded"/>
</xsd:sequence>
<xsd:attribute name="name" type="xsd:string"/>
<xsd:attribute name="category" type="micx:categoryType"
use="required" fixed="CCD"/>
</xsd:complexType>
</xsd:element>
</xsd:sequence>
</xsd:complexType>
</xsd:element>
</xsd:schema>
```

B.7 Пример PID-описания

B.7.1 Общие сведения

Файл SamplePIDb.xml содержит XML-данные для примера PID-описания, который в качестве XML-схемы использует файлы CCDb.xsd и DCDb.xsd. В нем описаны параметрические экземпляры класса из набора параметров загрузки в сетевой компьютер.

B.7.2 XML-схема: Файл SamplePIDb.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<ISO15745Profile xmlns="gost.e-ecolog.ru                                    "
xmlns:xsi="gost.e-ecolog.ru                                    "
xsi:schemaLocation="gost.e-ecolog.ru                                    CCDb.xsd">
<ProfileHeader>
<ProfileIdentification>ExamplePIDb</ProfileIdentification>
<ProfileRevision>1.0</ProfileRevision>
<ProfileName>Example of PID for MICX</ProfileName>
<ProfileSource>SamplePIDb.xml</ProfileSource>
<ProfileClassID>InformationExchange</ProfileClassID>
<ProfileDate>2009-02-25</ProfileDate>
<AdditionalInformation>gost.e-ecolog.ru                    /micx/ISO20242-4/PID
</AdditionalInformation>
<ISO15745Reference>
<ISO15745Part>1</ISO15745Part>
<ISO15745Edition>1</ISO15745Edition>
<ProfileTechnology>None</ProfileTechnology>
</ISO15745Reference>
<IASInterfaceType>CSI</IASInterfaceType>
</ProfileHeader>
```

```
<ProfileBody>
<CCD name="ExamplePID" category="CCD">
<DCD1 name="MICX" category="DCD">
<Device name="NC_lathe" category="MODULE">
<ProductionControl name="maching" category="INTERFACE">
<LoadRecipe category="OPERATION">
<LoadRecipeRequest message="REQUEST">
<RecipeRecord id="001" action="Add" confirm="Always">
<Recipe id="f001" name="machining">
<AcceptedWork item="B"/>
<AssignedEquipment resource="NC_C"/>
<Length name="insideDiameter">
<Qty value="15" unit="mm"/>
</Length>
<Length name="outsideDiameter">
<Qty value="36" unit="mm"/>
</Length>
<Length name="thickness">
<Qty value="2.6" unit="mm"/>
</Length>
</Recipe>
</RecipeRecord>
</LoadRecipeRequest>
<LoadRecipeRequest message="REQUEST">
<RecipeRecord id="002" action="Add" sender="MESX" confirm="Always">
<Recipe id="f002" name="machining">
<AcceptedWork item="B"/>
<AssignedEquipment resource="NC_C"/>
<Length name="insideDiameter">
<Qty value="16" unit="mm"/>
</Length>
<Length name="outsideDiameter">
<Qty value="40" unit="mm"/>
</Length>
<Length name="thickness">
<Qty value="2.6" unit="mm"/>
</Length>
</Recipe>
</RecipeRecord>
</LoadRecipeRequest>
</LoadRecipe>
</ProductionControl>
</Device>
</DCD1>
</CCD>
</ProfileBody>
</ISO15745Profile>
```

Приложение С
(справочное)

Шаблоны профилей возможностей открытого сетевого робототехнического интерфейса (OriN)

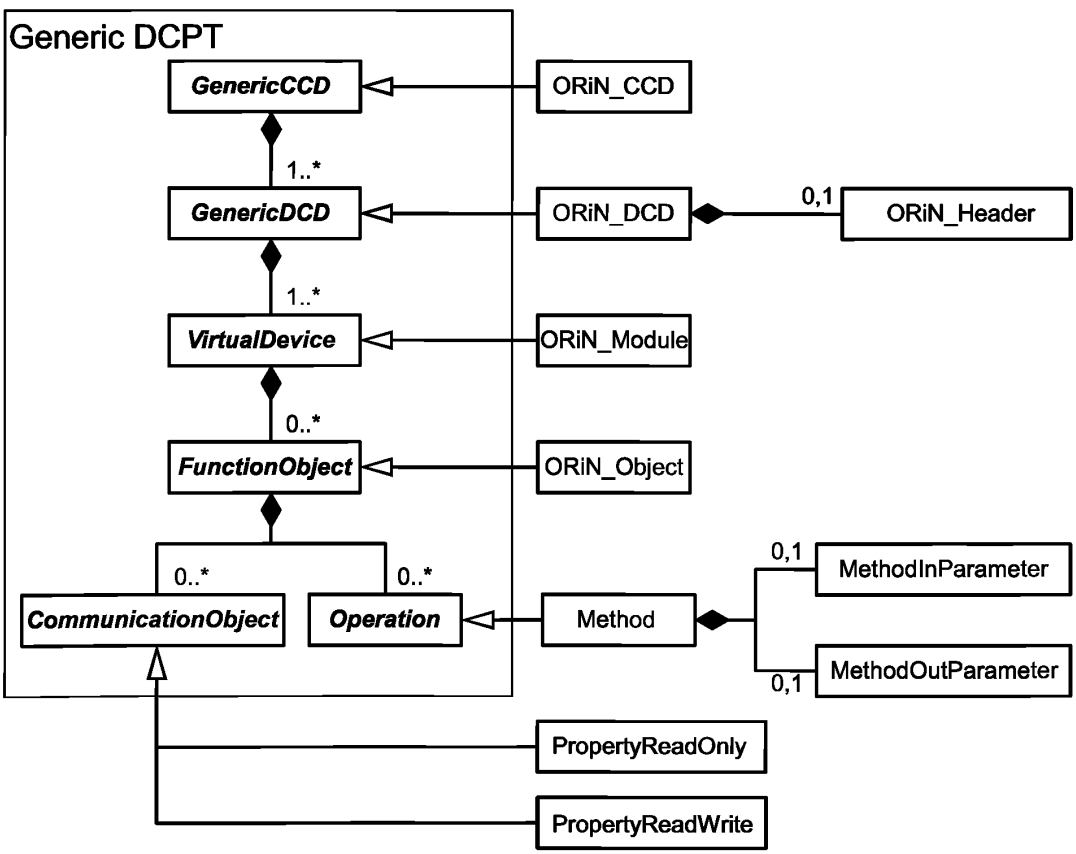
С.1 Общие сведения

Открытый сетевой робототехнический интерфейс (OriN) является обобщенным интерфейсом устройств, используемым для различных приложений. Ниже приведено описание зависящего от ORiN-интерфейса шаблона профиля возможностей устройства (версия 2.1), а также приведены примеры DCD-, CCD- и PID-описаний.

С.2 Зависящая от ORiN-интерфейса модель профиля

С.2.1 Общие сведения

Зависящая от выбираемого ORiN-интерфейса модель профиля содержит всю информацию, необходимую для описания возможностей устройств и их параметризации. На рисунке С.1 приведена диаграмма классов для зависящей от ORiN-интерфейса модели профиля шаблона (DCPT).



Generic DCPT — Обобщенный DCPT-шаблон.

Рисунок С.1 — Диаграмма классов для зависящей от ORiN-интерфейса модели профиля шаблона (DCPT)

С.2.2 Класс ORiN_CCD

Класс ORiN_CCD характеризует возможности зависящего от ORiN-интерфейса согласующего устройства (координатора). Он наследует класс GenericCCD и является абстрактным классом. Его наследует зависящий от согласующего устройства класс ORiN_CCD и определяет возможности этого устройства.

С.2.3 Класс ORiN_DCD

Класс ORiN_DCD характеризует возможности зависящего от ORiN-интерфейса драйвера согласующего устройства (координатора). Он наследует этот зависящий от ORiN-интерфейса драйвер. Он наследует класс GenericDCD и является абстрактным классом. Его наследует зависящий от устройства класс ORiN_DCD и определяет возможности драйвера этого устройства.

С.2.4 Класс ORiN_Header

Класс ORiN_Header содержит дополнительную информацию, которая используется для конкретизации драйвера устройства. Элементы этого класса указаны в таблице С.1. Определение класса ORiN_Header также приведено в ORiNcommon.xsd (см. А.6.2).

Таблица С.1 — Элементы класса ORiN_Header

Элемент класса ORiN_Header		Тип элемента	Описание элемента
DCD_Version		xsd:unsignedInt	Число версий для DCD version
DeviceVersion		xsd:unsignedInt	Число версий для устройства
ProviderName		xsd:string	Наименование провайдера ORiN
ProviderVersion		xsd:unsignedInt	Число версий для провайдера ORiN
Factory		xsd:string	Наименование производителя
DIT		xsd:string	Имя XML-текстового файла
ORiN_Version	Major	xsd:unsignedByte	Основной номер версии
	Minor	xsd:unsignedByte	Дополнительный номер версии
	Revision	xsd:unsignedByte	Номер редакции

С.2.5 Класс ORiN_Module

Класс ORiN_Module характеризует возможности зависящего от ORiN-интерфейса виртуального устройства. Он наследует класс VirtualDevice и является абстрактным классом. Зависящий от устройства класс ORiN_Module наследует его и определяет возможности конкретного виртуального устройства. Класс ORiN_Module может содержать параметр CreateParameter и идентифицироваться номером, содержащимся в дополнительном XML-атрибуте, называемом «moduleId» и принадлежащем типу «xsd:unsignedShort».

С.2.6 Класс ORiN_Object

Класс ORiN_Object характеризует возможности функциональных объектов зависящего от ORiN-интерфейса виртуального устройства. Он наследует класс FunctionObject и является абстрактным классом. Зависящий от устройства класс ORiN_Object наследует его и определяет возможности зависящего от устройства функционального объекта. Класс ORiN_Object может содержать параметр CreateParameter и идентифицироваться номером, содержащимся в дополнительном XML-атрибуте, называемом «funcId» и принадлежащем типу «xsd:unsignedShort».

С.2.7 Класс Method

Класс Method описывает работу зависящего от ORiN-интерфейса виртуального устройства, наследует класс Operation и является абстрактным классом. Каждый зависящий от устройства класс Method наследует его и характеризует возможности операции. Каждый класс Method содержит один входной и один выходной операционный параметр, которые могут быть нулевыми или (иногда) XML-экземплярами реализации. Присвоение значения входному операционному параметру в XML-экземпляре (PID) указывает на то, что операция должна выполняться для конфигурации. Класс Method идентифицируется номером, содержащимся в дополнительном XML-атрибуте, называемом «operationIdId» и принадлежащем типу «xsd:unsignedShort».

С.2.8 Класс MethodInParameter

Класс MethodInParameter отмечает выполнение операции и является абстрактным классом. Каждый зависящий от устройства класс MethodInParameter наследует его и определяет тип данных входного параметра.

С.2.9 Класс MethodOutParameter

Класс MethodOutParameter характеризует выходное значение для операции, является абстрактным классом и содержит значение выходного параметра. Каждый зависящий от устройства класс MethodOutParameter наследует его и определяет тип данных значения.

С.2.10 Класс PropertyReadOnly

Класс PropertyReadOnly характеризует значение времени прогона функционального объекта, который можно только считывать. Он наследует класс CommunicationObject и является абстрактным классом. Каждый зависящий

от устройства класс PropertyReadOnly наследует его и определяет возможности каждого значения времени прогона и его тип данных.

С.2.11 Класс PropertyReadWrite

Класс PropertyReadWrite характеризует значение времени прогона функционального объекта, который можно и считывать и записывать. Он наследует класс CommunicationObject и является абстрактным классом. Каждый зависящий от устройства класс PropertyReadWrite наследует его и определяет возможности каждого значения времени прогона и его тип данных.

С.2.12 Идентификация коммуникационных объектов ORiN-интерфейса

Экземпляры классов PropertyReadOnly и PropertyReadWrite определяют по номеру, который является подсчетом этих экземпляров в FunctionObject, начинающимся с единицы.

С.3 Элементы типовых данных и сценариев конфигурирования

С.3.1 Общие сведения

Элемент Value задается для облегчения наследования типов данных, не затрагивающего наследования структурных классов. С помощью данного метода определение типов данных для коммуникационных объектов и создание параметров само по себе не связано с этими объектами. Элемент Value не требуется в тех случаях, когда типы данных определены неявно по отношению к коммуникационному объекту.

С.3.2 Элемент Value

Класс Value описывает элемент типа xsd:anyType, которым можно пренебречь для любого другого типа, и поэтому он будет лишь полем для заполнения для любого типа элемента, определенного в каком-либо другом месте описания возможностей устройства. Использование этого дополнительного элемента облегчает наследование при определении типов данных (см. рисунок С.2).

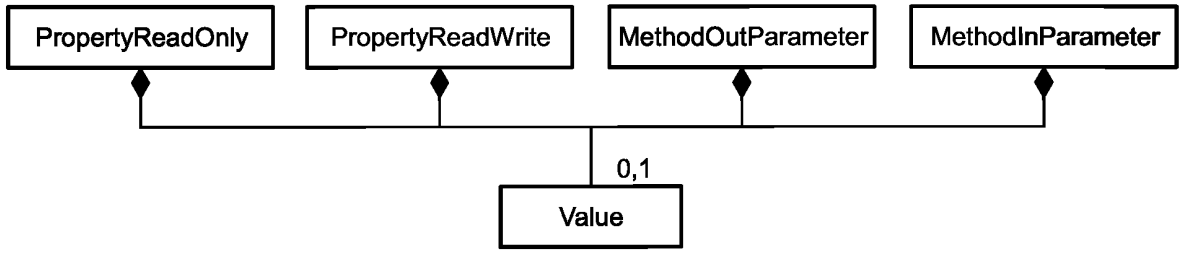


Рисунок С.2 — Элемент Value для несвязанных определений типа

С.4 Дополнительная идентификация типа обобщенного DCPT-шаблона

В таблицах С.2–С.4 указаны зависящие от используемой технологии типы идентификаторов для коммуникационных объектов и операционных параметров.

Таблица С.2 — Типы идентификаторов для коммуникационных объектов

Элемент зависящего от технологии DCPT-шаблона	Содержимое XML-атрибута «category»	Содержимое XML-атрибута «readonly»[a)]
AttributeReadWrite	**ATTRIBUTE**	ложное
AttributeReadOnly	**ATTRIBUTE**	истинное
Parameter	**PARAMETER**	(ложное)
[a)] Дополнительный XML-атрибут «readonly» типа xsd:boolean является обязательным для элементов AttributeReadWrite и AttributeReadOnly.		

Таблица С.3 — Типы идентификаторов для операционных параметров

Элемент зависящего от технологии DCPT-шаблона	Содержимое XML-атрибута «category»
OperationInParameter	IN
OperationOutParameter	OUT

Т а б л и ц а С.4 — Типы идентификаторов для создания параметров

Элемент зависящего от технологии DCPT-шаблона	Содержимое XML-атрибута «category»
CreateParameter	**CREATEPARAMETER**

С.5 Дополнительные XML-атрибуты для обобщенного DCPT-шаблона

Коммуникационные объекты AttributeReadOnly и AttributeReadWrite можно использовать для передачи незапрашиваемых данных вместе с сервисами VDSIinfReport и VDSI_Accept. Запрос на использование этих сервисов во время прогона выполняется в процессе конфигурирования путем установки XML-атрибутов infReport и присвоения типу «xsd:boolean» истинного значения (см. таблицу С.5).

Т а б л и ц а С.5 — XML-атрибуты для передачи незапрашиваемых данных

Наименование XML-атрибута	Используется для элемента	Описание содержимого атрибута
infReport	AttributeReadWrite, AttributeReadOnly	Запрос на использование сервиса VDSMnfReport
accept	AttributeReadWrite	Запрос на использование сервиса VDSI_Accept

С.6 Зависящий от ORiN-интерфейса шаблон профиля возможностей устройства

С.6.1 Общие сведения

ORiNcommon.xsd — это XML-схема (структура), содержащая базовые классы, которые можно использовать для описания этой структуры, содержащей всю информацию для описания функциональных возможностей устройства и параметризации. Задаваемые типы элемента AttributeReadWrite могут быть расширены элементами Value. В этом случае элементы Value могут иметь тип данных, определяемый в дополнительной XML-схеме. Кроме того, определяемые типы данных AttributeReadOnly и OperationOutParameter могут быть расширены элементами Value с внешне заданными типами данных.

П р и м е ч а н и е — Если элементы Value не используются, а коммуникационные объекты содержат значения простого или комплексного типа данных, то из-за расширения числа типов данных XML-схема может измениться. В этом случае ORiNcommon.xsd можно использовать в качестве образца (модели) структуры получаемого XML-экземпляра и необходимых XML-атрибутов.

С.6.2 XML-схема: ORiNcommon.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
;=============================================================================
; ORiNcommon.xsd Edition: 27-Feb-2010
;=============================================================================
-->
<xsd:schema xmlns:xsd="gost.e-ecolog.ru                    " elementFormDefault="qualified"
attributeFormDefault="unqualified">
<!-- Header information is contained in each schema file -->
<xsd:annotation>
<xsd:appinfo source="gost.e-ecolog.ru                    ">
<DCPTHeader>
<DCPTIdentification>ORiN_Common</DCPTIdentification>
<DCPTRevision>1.0</DCPTRevision>
<DCPTName>ORiNcommon</DCPTName>
<DCPTSource>ORiNcommon.xsd</DCPTSource>
<DCPTClassID>TechnlogySpecificDCPT</DCPTClassID>
<DCPTDate>2010-02-27</DCPTDate>
<ISO20242Reference>
<ISO20242Edition>1</ISO20242Edition>
<Technology>ORiN</Technology>
</ISO20242Reference>
```

```
</DCPTHeader>
</xsd:appinfo>
</xsd:annotation>
<!-- Valid content for XML-Attribut category (kind of DCD element) -->
<xsd:simpleType name="Category">
<xsd:restriction base="xsd:string">
<xsd:enumeration value="MODULE"/>
<xsd:enumeration value="INTERFACE"/>
<xsd:enumeration value="ATTRIBUTE"/>
<xsd:enumeration value="OPERATION"/>
<xsd:enumeration value="IN"/>
<xsd:enumeration value="OUT"/>
<xsd:enumeration value="DCD"/>
<xsd:enumeration value="CCD"/>
</xsd:restriction>
</xsd:simpleType>
<!-- Group of attributes for assigning multilingual text to elements -->
<xsd:attributeGroup name="TextAttributes">
<xsd:attribute name="areaMsg" type="xsd:unsignedShort "use="optional"/>
<xsd:attribute name="infMsg" type="xsd:unsignedShort" use="optional"/>
<xsd:attribute name="areaText" type="xsd:unsignedShort" use="optional"/>
<xsd:attribute name="infText" type="xsd:unsignedShort" use="optional"/>
</xsd:attributeGroup>
<!-- Basic type definition for most elements used in the DCD -->
<xsd:complexType name="TNamedDCDElement" abstract="true">
<xsd:attribute name="helpstring" type="xsd:string" use="optional"/>
<xsd:attributeGroup ref="TextAttributes"/>
</xsd:complexType>
<!-- Virtual Devices have additional XML-attributes "moduleId" and category="MODULE". -->
<xsd:complexType name="ORiN_Module" abstract="true">
<xsd:complexContent>
<xsd:extension base="TNamedDCDElement">
<xsd:attribute name="moduleId" type="xsd:unsignedShort" use="required"/>
<xsd:attribute name="category" type="Category" use="required" fixed="MODULE"/>
<xsd:attribute name="provider" type="xsd:string" use="required"/>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<!-- Function Objects have additional XML-attributes "funcId" and category="INTERFACE". -->
<xsd:complexType name="ORiN_Object" abstract="true">
<xsd:complexContent>
<xsd:extension base="TNamedDCDElement">
<xsd:attribute name="funcId" type="xsd:unsignedShort" use="required"/>
<xsd:attribute name="category" type="Category" use="required" fixed="INTERFACE"/>
<xsd:attribute name="name" type="xsd:string" use="optional"/>
<xsd:attribute name="option" type="xsd:string" use="optional"/>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<!-- Communication Objects which may be changed have readonly="false" -->
<!-- With respect to ORiN obejct model, they are called Properties here -->
<xsd:complexType name="PropertyReadWrite" abstract="true">
<xsd:complexContent>
<xsd:extension base="TNamedDCDElement">
<xsd:attribute name="readonly" type="xsd:boolean" fixed="false"/>
<xsd:attribute name="category" type="Category" use="required" fixed="ATTRIBUTE"/>
<xsd:attribute name="infReport" type="xsd:boolean"/>
<xsd:attribute name="accept" type="xsd:boolean"/>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<!-- Communication Objects which may not be changed have readonly="true" -->
```

```
<!-- With respect to ORiN obejct model, they are called Properties here -->
<xsd:complexType name="PropertyReadOnly" abstract="true">
<xsd:complexContent>
<xsd:extension base="TNamedDCDElement">
<xsd:attribute name="readonly" type="xsd:boolean" use="required" fixed="true"/>
<xsd:attribute name="category" type="Category" use="required" fixed="ATTRIBUTE"/>
<xsd:attribute name="infReport" type="xsd:boolean"/>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<!-- Operations / Methods -->
<xsd:complexType name="Method" abstract="true">
<xsd:complexContent>
<xsd:extension base="TNamedDCDElement">
<xsd:attribute name="operationId" type="xsd:unsignedShort" use="required"/>
<xsd:attribute name="category" type="Category" use="required" fixed="OPERATION"/>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="MethodOutParameter" abstract="true">
<xsd:attribute name="category" type="Category" use="required" fixed="OUT"/>
<xsd:attributeGroup ref="TextAttributes"/>
</xsd:complexType>
<xsd:complexType name="MethodInParameter" abstract="true">
<xsd:attribute name="category" type="Category" use="required" fixed="IN"/>
<xsd:attributeGroup ref="TextAttributes"/>
</xsd:complexType>
<!--
==========================================================================
Types of ISO 20242-4 Annex C
==========================================================================
-->
<xsd:complexType name="ORiNVersionType">
<xsd:sequence>
<xsd:element name="Major" type="xsd:unsignedByte" default="2"/>
<xsd:element name="Minor" type="xsd:unsignedByte" default="1"/>
<xsd:element name="Revision" type="xsd:unsignedByte" default="0"/>
</xsd:sequence>
</xsd:complexType>
<xsd:complexType name="HeaderType">
<xsd:sequence>
<xsd:element name="DCD_Version" type="xsd:unsignedInt"/>
<xsd:element name="DeviceVersion" type="xsd:unsignedInt"/>
<xsd:element name="ProviderName" type="xsd:string"/>
<xsd:element name="ProviderVersion" type="xsd:unsignedInt"/>
<xsd:element name="Factory" type="xsd:string"/>
<xsd:element name="DIT" type="xsd:string"/>
<xsd:element name="ORiN_Version" type="ORiNVersionType"/>
</xsd:sequence>
</xsd:complexType>
<xsd:complexType name="ORiN_DCD" abstract="true">
<xsd:sequence>
<xsd:element name="ORiN_Header" type="HeaderType" minOccurs="0"/>
</xsd:sequence>
<xsd:attribute name="category" type="Category" use="required" fixed="DCD"/>
<xsd:attributeGroup ref="TextAttributes"/>
<xsd:attribute name="dllPath" type="xsd:string" use="required"/>
<xsd:attribute name="ProviderVersion" type="xsd:int" use="required"/>
</xsd:complexType>
</xsd:schema>
```

С.7 Примеры описания возможностей устройства

С.7.1 Общие сведения

XML-схемы DCD1.xsd и DCD2.xsd являются примерами описания возможностей устройства, которые расширяют существующую XML-схему ORiNCommon.xsd в соответствии с возможностями драйверов устройств. Имена XML-элементов должны быть определены в соответствии с требованиями потребителей. Структурная информация об XML-элементах определена в содержании XML-атрибута «категория» (category).

С.7.2 XML-схема: DCD1.xsd

```
<?xml version="1.0" encoding="utf-8"?>
<xsd:schema xmlns:xsd="gost.e-ecolog.ru "
xmlns="gost.e-ecolog.ru jp/ISO20242-4/DCD1"
targetNamespace="gost.e-ecolog.ru " elementFormDefault="qualified">
<xsd:annotation>
<xsd:appinfo source="gost.e-ecolog.ru ">
<DCPTHeader>
<DCPTIdentification>DCD1</DCPTIdentification>
<DCPTRevision>1.0</DCPTRevision>
<DCPTName>DCD1</DCPTName>
<DCPTSource>DCD1.xsd</DCPTSource>
<DCPTClassID>DCD1</DCPTClassID>
<DCPTDate>2010-05-26</DCPTDate>
<ISO20242Reference>
<ISO20242Edition>1</ISO20242Edition>
<Technology>ORiN</Technology>
</ISO20242Reference>
</DCPTHeader>
</xsd:appinfo>
</xsd:annotation>
<xsd:include schemaLocation="ORiNcommon.xsd"/>
<!-- Main type is ORiN_DCD with contained Virtual Device Types.
There may be many different Virtual Device Types contained. -->
<xsd:element name="DCD1" type="DCDType"/>
<xsd:complexType name="DCDType">
<xsd:complexContent>
<xsd:extension base="ORiN_DCD">
<xsd:sequence>
<xsd:element ref="Provider" minOccurs="0" maxOccurs="unbounded"/>
</xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<!-- Virtual Device Types have fixed numbers for identification in moduleID.
This is a restriction of type VDInstance. If You want to avoid
restriction, move attribute moduleId from type ORiN_Module with a fixed
number to type ProviderType below, which than is an extension of ORiN_Module
and type ProviderBaseType will be obsolete. -->
<xsd:complexType name="ProviderBaseType">
<xsd:complexContent>
<xsd:restriction base="ORiN_Module">
<xsd:attribute name="provider" type="xsd:string" use="required" fixed="CaoProv.DCD1"/>
<xsd:attribute name="moduleId" type="xsd:unsignedShort" use="required" fixed="0"/>
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
<!-- This Virtual Device Type examplary only contains one Function Object
type without restricting the number of instances. -->
<xsd:element name="Provider" type="ProviderType"/>
<xsd:complexType name="ProviderType">
<xsd:complexContent>
<xsd:extension base="ProviderBaseType">
<xsd:sequence>
<xsd:element ref="CaoProvController" maxOccurs="unbounded"/>
```

```
<xsd:element ref="CaoProvRobot» minOccurs="0" maxOccurs="unbounded"/>
</xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<!-- Function Object Types have fixed numbers for identification in funcID.
This is a restriction of type ORiN_Object. If You want to avoid
restriction, move attribute funcId from type ORiN_Object with a fixed
number to type ControllerBaseType below, which than is an extension of
ORiN_Object and type ControllerBaseType will be obsolete. -->
<xsd:complexType name="ControllerBaseType">
<xsd:complexContent>
<xsd:restriction base="ORiN_Object">
<xsd:attribute name="funcId" type="xsd:unsignedShort" use="required" fixed="101"/>
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="RobotBaseType">
<xsd:complexContent>
<xsd:restriction base="ORiN_Object">
<xsd:attribute name="funcId" type="xsd:unsignedShort" use="required" fixed="104"/>
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
<!-- This Function Object type exemplary contains several Operations. -->
<xsd:element name="CaoProvController" type="ControllerType"/>
<xsd:complexType name="ControllerType">
<xsd:complexContent>
<xsd:extension base="ControllerBaseType">
<xsd:sequence>
<xsd:element name="Connect" type="CtrlConnectType"/>
<xsd:element name="Disconnect" type="CtrlDisconnectType"/>
<xsd:element name="GetRobot" type="CtrlGetRobotType"/>
</xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<!-- This Function Object type exemplary contains one Operation. -->
<xsd:element name="CaoProvRobot" type="RobotType"/>
<xsd:complexType name="RobotType">
<xsd:complexContent>
<xsd:extension base="RobotBaseType">
<xsd:sequence>
<xsd:element name="Move" type="RobotMoveType" minOccurs="0"/>
</xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<!-- This Operation has optional input and output values. -->
<xsd:complexType name="CtrlConnectBaseType">
<xsd:complexContent>
<xsd:restriction base="Method">
<xsd:attribute name="operationId" type="xsd:unsignedShort" use="required" fixed="3"/>
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="CtrlConnectType">
<xsd:complexContent>
<xsd:extension base="CtrlConnectBaseType">
<xsd:sequence>
<xsd:element name="Parameter" type="MethodInGetObjectType"/>
<xsd:element name="Result" type="MethodOutVoidType"/>
```

```
</xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="CtrlDisconnectBaseType">
<xsd:complexContent>
<xsd:restriction base="Method">
<xsd:attribute name="operationId" type="xsd:unsignedShort" use="required" fixed="4"/>
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="CtrlDisconnectType">
<xsd:complexContent>
<xsd:extension base="CtrlDisconnectBaseType">
<xsd:sequence>
<xsd:element name="Parameter" type="MethodInVoidType"/>
<xsd:element name="Result" type="MethodOutVoidType"/>
</xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="CtrlGetRobotBaseType">
<xsd:complexContent>
<xsd:restriction base="Method">
<xsd:attribute name="operationId" type="xsd:unsignedShort" use="required" fixed="7"/>
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="CtrlGetRobotType">
<xsd:complexContent>
<xsd:extension base="CtrlGetRobotBaseType">
<xsd:sequence>
<xsd:element name="Parameter" type="MethodInGetObjectType"/>
<xsd:element name="Result" type="MethodOUTGetObject"/>
</xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="RobotMoveBaseType">
<xsd:complexContent>
<xsd:restriction base="Method">
<xsd:attribute name="operationId" type="xsd:unsignedShort" use="required" fixed="72"/>
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="RobotMoveType">
<xsd:complexContent>
<xsd:extension base="RobotMoveBaseType">
<xsd:sequence>
<xsd:element name="Parameter" type="MethodInRobotMove"/>
<xsd:element name="Result" type="MethodOutVoidType"/>
</xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<!-- The input parameters for this operation may be a sequence of values. -->
<xsd:complexType name="MethodInVoidType">
<xsd:complexContent>
<xsd:extension base="MethodInParameter">
<xsd:sequence>
<xsd:element name="Value" type="void"/>
</xsd:sequence>
```

```
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="MethodInGetObjectType">
<xsd:complexContent>
<xsd:extension base="MethodInParameter">
<xsd:sequence>
<xsd:element name="Value">
<xsd:complexType>
<xsd:sequence>
<xsd:element name="Name" type="xsd:string"/>
<xsd:element name="Option" type="xsd:string"/>
</xsd:sequence>
</xsd:complexType>
</xsd:element>
</xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="MethodInRobotMove">
<xsd:complexContent>
<xsd:extension base="MethodInParameter">
<xsd:sequence>
<xsd:element name="Value">
<xsd:complexType>
<xsd:sequence>
<xsd:element name="Interpolation" type="xsd:integer"/>
<xsd:element name="Pose" type="xsd:anyType"/>
<xsd:element name="Option" type="xsd:string"/>
</xsd:sequence>
</xsd:complexType>
</xsd:element>
</xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<!-- The output parameters for this operation is a special type "void",
which defines, that the accordingly XML element has to be empty. -->
<xsd:complexType name="MethodOutVoidType">
<xsd:complexContent>
<xsd:extension base="MethodOutParameter">
<xsd:sequence>
<xsd:element name="Value" type="void"/>
</xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="MethodOUTGetObject">
<xsd:complexContent>
<xsd:extension base="MethodOutParameter">
<xsd:sequence>
<xsd:element name="Value" type="xsd:unsignedInt"/>
</xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<!-- empty type -->
<xsd:complexType name="void"/>
</xsd:schema>
```

C.7.3 XML-схема: DCD2.xsd

```
<?xml version="1.0" encoding="utf-8"?>
```

```
<xsd:schema xmlns:xsd="gost.e-ecolog.ru                                    "
xmlns="gost.e-ecolog.ru              jp/ISO20242-4/DCD2"
targetNamespace="gost.e-ecolog.ru                                    " elementFormDefault="qualified">
<xsd:annotation>
<xsd:appinfo source="gost.e-ecolog.ru                                          ">
<DCPTHeader>
<DCPTIdentification>DCD2</DCPTIdentification>
<DCPTRevision>1.0</DCPTRevision>
<DCPTName>DCD2</DCPTName>
<DCPTSource>DCD2.xsd</DCPTSource>
<DCPTClassID>DCD2</DCPTClassID>
<DCPTDate>2010-05-26</DCPTDate>
<ISO20242Reference>
<ISO20242Edition>1</ISO20242Edition>
<Technology>ORiN</Technology>
</ISO20242Reference>
</DCPTHeader>
</xsd:appinfo>
</xsd:annotation>
<xsd:include schemaLocation="ORiNcommon.xsd"/>
<!-- Main type is ORiN_DCD with contained Virtual Device Types.
There may be many different Virtual Device Types contained. –>
<xsd:element name="DCD2" type="DCDType"/>
<xsd:complexType name="DCDType">
<xsd:complexContent>
<xsd:extension base="ORiN_DCD">
<xsd:sequence>
<xsd:element ref="Provider" minOccurs="0" maxOccurs="unbounded"/>
</xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<!-- Virtual Device Types have fixed numbers for identification in moduleID.
This is a restriction of type ORiN_Module. If You want to avoid
restriction, move attribute moduleId from type ORiN_Module with a fixed
number to type ProviderType below, which than is an extension of ORiN_Module
and type ProviderBase Type will be obsolete. –>
<xsd:complexType name="ProviderBaseType">
<xsd:complexContent>
<xsd:restriction base="ORiN_Module">
<xsd:attribute name="moduleId" type="xsd:unsignedShort" use="required" fixed="0"/>
<xsd:attribute name="provider" type="xsd:string" use="required" fixed="CaoProv.DCD2"/>
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
<!-- Virtual Device Types typically contain a Create Paramter of a specific
type and several Function Objects of different types.
This exemplary Function Object type describes an input channel and it is
possible to use up to 16 channels with each Virtual Device instance. –>
<xsd:element name="Provider" type="ProviderType"/>
<xsd:complexType name="ProviderType">
<xsd:complexContent>
<xsd:extension base="ProviderBaseType">
<xsd:sequence>
<xsd:element ref="CaoProvController" maxOccurs="unbounded"/>
<xsd:element ref="CaoProvVariable" minOccurs="0" maxOccurs="unbounded"/>
</xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<!-- Function Object Types have fixed numbers for identification in moduleID.
This is a restriction of type ORiN_Object. If You want to avoid
```

```
restriction, move attribute funcId from type ORiN_Object with a fixed
number to type ControllerType below, which than is an extension of
ORiN_Object and type ControllerBase Type will be obsolete. -->
<xsd:complexType name="ControllerBaseType">
<xsd:complexContent>
<xsd:restriction base="ORiN_Object">
<xsd:attribute name="funcId" type="xsd:unsignedShort" use="required" fixed="101"/>
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="VariableBaseType">
<xsd:complexContent>
<xsd:restriction base="ORiN_Object">
<xsd:attribute name="funcId" type="xsd:unsignedShort" use="required" fixed="106"/>
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
<!-- This Function Object type exemplary contains several Operations. –>
<xsd:element name="CaoProvController" type="ControllerType"/>
<xsd:complexType name="ControllerType">
<xsd:complexContent>
<xsd:extension base="ControllerBaseType">
<xsd:sequence>
<xsd:element name="Connect" type="CtrlConnectType" minOccurs="0"/>
<xsd:element name="Disconnect" type="CtrlDisconnectType" minOccurs="0"/>
<xsd:element name="GetVariable" type="CtrlGetVariableType" minOccurs="0"/>
</xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<!-- This Function Object type exemplary contains several Communication Objects. –>
<xsd:element name="CaoProvVariable" type="VariableType"/>
<xsd:complexType name="VariableType">
<xsd:complexContent>
<xsd:extension base="VariableBaseType">
<xsd:sequence>
<xsd:element name="Attribute" type="PropertyROIntType" minOccurs="0"/>
<xsd:element name="Value" type="PropertyRWVariantType" minOccurs="0"/>
</xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<!-- This Operation has optional input and output values. -->
<xsd:complexType name="CtrlConnectType">
<xsd:complexContent>
<xsd:extension base="CtrlConnectBaseType">
<xsd:sequence>
<xsd:element name="Parameter" type="MethodInGetObjectType"/>
<xsd:element name="Result" type="MethodOutVoidType"/>
</xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="CtrlConnectBaseType">
<xsd:complexContent>
<xsd:restriction base="Method">
<xsd:attribute name="operationId" type="xsd:unsignedShort" use="required" fixed="3"/>
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="CtrlDisconnectType">
<xsd:complexContent>
```

```
<xsd:extension base="CtrlDisconnectBaseType">
<xsd:sequence>
<xsd:element name="Parameter" type="MethodInVoidType"/>
<xsd:element name="Result" type="MethodOutVoidType"/>
</xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="CtrlDisconnectBaseType">
<xsd:complexContent>
<xsd:restriction base="Method">
<xsd:attribute name="operationId" type="xsd:unsignedShort" use="required" fixed="4"/>
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="CtrlGetVariableType">
<xsd:complexContent>
<xsd:extension base="CtrlGetVariableBaseType">
<xsd:sequence>
<xsd:element name="Parameter" type="MethodInGetObjectType"/>
<xsd:element name="Result" type="MethodOUTGetObject"/>
</xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="CtrlGetVariableBaseType">
<xsd:complexContent>
<xsd:restriction base="Method">
<xsd:attribute name="operationId" type="xsd:unsignedShort"
use="required" fixed="9"/>
</xsd:restriction>
</xsd:complexContent>
</xsd:complexType>
<!-- The input parameters for this operation may be a sequence of values. -->
<xsd:complexType name="MethodInVoidType">
<xsd:complexContent>
<xsd:extension base="MethodInParameter">
<xsd:sequence>
<xsd:element name="Value" type="void"/>
</xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="MethodInGetObjectType">
<xsd:complexContent>
<xsd:extension base="MethodInParameter">
<xsd:sequence>
<xsd:element name="Value">
<xsd:complexType>
<xsd:sequence>
<xsd:element name="Name" type="xsd:string"/>
<xsd:element name="Option" type="xsd:string"/>
</xsd:sequence>
</xsd:complexType>
</xsd:element>
</xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<!-- The output parameters for this operation is a special type "void",
which defines, that the accordingly XML element has to be empty. -->
<xsd:complexType name="MethodOutVoidType">
```

```
<xsd:complexContent>
<xsd:extension base="MethodOutParameter">
<xsd:sequence>
<xsd:element name="Value" type="void"/>
</xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="MethodOUTGetObject">
<xsd:complexContent>
<xsd:extension base="MethodOutParameter">
<xsd:sequence>
<xsd:element name="Value" type="xsd:unsignedInt"/>
</xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<!-- A simple readonly Property for the Function Object -->
<xsd:complexType name="PropertyROIntType">
<xsd:complexContent>
<xsd:extension base="PropertyReadOnly">
<xsd:sequence>
<xsd:element name="Value" type="xsd:int"/>
</xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<!-- A simple read/write Property for the Function Object -->
<xsd:complexType name="PropertyRWVariantType">
<xsd:complexContent>
<xsd:extension base="PropertyReadWrite">
<xsd:sequence>
<xsd:element name="Value" type="xsd:anyType"/>
</xsd:sequence>
</xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<!-- empty type -->
<xsd:complexType name="void"/>
</xsd:schema>
```

С.8 Пример описания возможностей согласующего устройства

С.8.1 Общие сведения

XML-схема CCD.xsd является примером описания возможностей согласующего устройства, которая описывает XML-схему парметризации двух драйверов устройств.

С.8.2 XML-схема: CCD.xsd

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="gost.e-ecolog.ru                                    "
xmlns:dcd1="gost.e-ecolog.ru                .jp/ISO20242-4/DCD1"
xmlns:dcd2="gost.e-ecolog.ru                                    " elementFormDefault="qualified"
attributeFormDefault="unqualified">
<xsd:annotation>
<xsd:appinfo source="gost.e-ecolog.ru                                    ">
<DCPTHeader>
<DCPTIdentification>CCD</DCPTIdentification>
<DCPTRevision>1.0</DCPTRevision>
<DCPTName>CCD</DCPTName>
<DCPTSource>CCD.xsd</DCPTSource>
<DCPTClassID>CCD</DCPTClassID>
<DCPTDate>2010-05-24</DCPTDate>
<ISO20242Reference>
<ISO20242Edition>1</ISO20242Edition>
```

```
<Technology>ORiN</Technology>
</ISO20242Reference>
</DCPTHeader>
</xsd:appinfo>
</xsd:annotation>
<!-- A coodinator capability description contains the capability
descriptions of all devices, which will be usable -->
<xsd:include schemaLocation="ORiNcommon.xsd"/>
<xsd:import namespace="gost.e-ecolog.ru                    " schemaLocation="DCD1.xsd"/>
<xsd:import namespace="gost.e-ecolog.ru                    " schemaLocation="DCD2.xsd"/>
<!-- * ISO15745 Profile Root * -->
<xsd:element name="ISO15745Profile">
<xsd:complexType>
<xsd:sequence>
<xsd:element ref="ProfileHeader"/>
<xsd:element ref="ProfileBody"/>
</xsd:sequence>
</xsd:complexType>
</xsd:element>
<!-- * HEADER DATA TYPES * -->
<xsd:element name="ProfileHeader">
<xsd:complexType>
<xsd:sequence>
<xsd:element name="ProfileIdentification" type="xsd:string"/>
<xsd:element name="ProfileRevision" type="xsd:string"/>
<xsd:element name="ProfileName" type="xsd:string"/>
<xsd:element name="ProfileSource" type="xsd:string"/>
<xsd:element name="ProfileClassID" type="ProfileClassID_DataType" fixed="InformationExchange"/>
<xsd:element name="ProfileDate" type="xsd:date" minOccurs="0"/>
<xsd:element name="AdditionalInformation" type="xsd:anyURI" minOccurs="0"/>
<xsd:element name="ISO15745Reference" type="ISO15745Reference_DataType"/>
<xsd:element name="IASInterfaceType" type="IASInterface_DataType" fixed="CSI"
minOccurs="0" maxOccurs="unbounded"/>
</xsd:sequence>
</xsd:complexType>
</xsd:element>
<xsd:complexType name="ISO15745Reference_DataType">
<xsd:sequence>
<xsd:element name="ISO15745Part" type="xsd:string" fixed="1"/>
<xsd:element name="ISO15745Edition" type="xsd:string" fixed="1"/>
<xsd:element name="ProfileTechnology" type="xsd:string" fixed="None"/>
</xsd:sequence>
</xsd:complexType>
<xsd:simpleType name="ProfileClassID_DataType">
<xsd:restriction base="xsd:string">
<xsd:enumeration value="AIP"/>
<xsd:enumeration value="Process"/>
<xsd:enumeration value="InformationExchange"/>
<xsd:enumeration value="Resource"/>
<xsd:enumeration value="Device"/>
<xsd:enumeration value="CommunicationNetwork"/>
<xsd:enumeration value="Equipment"/>
<xsd:enumeration value="Human"/>
<xsd:enumeration value="Material"/>
</xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="IASInterface_DataType">
<xsd:union>
<xsd:simpleType>
<xsd:restriction base="xsd:string">
<xsd:enumeration value="CSI"/>
<xsd:enumeration value="HCI"/>
```

```
<xsd:enumeration value="ISI"/>
<xsd:enumeration value="API"/>
<xsd:enumeration value="CMI"/>
<xsd:enumeration value="ESI"/>
<xsd:enumeration value="FSI"/>
<xsd:enumeration value="MTI"/>
<xsd:enumeration value="SEI"/>
<xsd:enumeration value="USI"/>
</xsd:restriction>
</xsd:simpleType>
<xsd:simpleType>
<xsd:restriction base="xsd:string">
<xsd:length value="4"/>
</xsd:restriction>
</xsd:simpleType>
</xsd:union>
</xsd:simpleType>
<!-- * BODY SECTION * -->
<xsd:element name="ProfileBody">
<xsd:complexType>
<xsd:sequence>
<xsd:element name="CCD" maxOccurs="unbounded">
<xsd:complexType>
<xsd:sequence>
<xsd:element ref="dcd1:DCD1" minOccurs="0"/>
<xsd:element ref="dcd2:DCD2" minOccurs="0"/>
</xsd:sequence>
<xsd:attribute name="category" type="Category" use="required" fixed="CCD"/>
</xsd:complexType>
</xsd:element>
</xsd:sequence>
</xsd:complexType>
</xsd:element>
</xsd:schema>
```

С.9 Пример параметрического описания экземпляра класса

С.9.1 Общие сведения

SamplePID.xml является примером параметрического описания экземпляра класса в XML-формате, который использует файл CCD.xsd в качестве XML-схемы и описывает параметрические экземпляры координатора и драйверов устройств.

С.9.2 XML: SamplePID.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<ISO15745Profilexmlns:xsi="gost.e-ecolog.ru                    001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="../Schema/CCD.xsd">
<ProfileHeader>
<ProfileIdentification>ExampleOfPID</ProfileIdentification>
<ProfileRevision>1.0</ProfileRevision>
<ProfileName>Example of PID</ProfileName>
<ProfileSource>PID.xml</ProfileSource>
<ProfileClassID>InformationExchange</ProfileClassID>
<ProfileDate>2010-03-26</ProfileDate>
<AdditionalInformation>gost.e-ecolog.ru                                                    >
<ISO15745Reference>
<ISO15745Part>1</ISO15745Part>
<ISO15745Edition>1</ISO15745Edition>
<ProfileTechnology>None</ProfileTechnology>
</ISO15745Reference>
<IASInterfaceType>CSI</IASInterfaceType>
</ProfileHeader>
<ProfileBody>
<CCD category="CCD">
<DCD1 category="DCD" dllPath="CaoProvNetwoRC.dll" ProviderVersion="1"
xmlns="gost.e-ecolog.ru                                    ">
```

```
<ORiN_Header>
<DCD_Version>1</DCD_Version>
<DeviceVersion>1</DeviceVersion>
<ProviderName/>
<ProviderVersion>1</ProviderVersion>
<Factory/>
<DIT/>
<ORiN_Version>
<Major>1</Major>
<Minor>1</Minor>
<Revision>1</Revision>
</ORiN_Version>
</ORiN_Header>
<Provider category="MODULE" moduleId="0" provider="CaoProv.DCD1">
<CaoProvController category="INTERFACE" funcId="101">
<Connect category="OPERATION" operationId="3">
<Parameter category="IN">
<Value>
<Name>RC1</Name>
<Option/>
</Value>
</Parameter>
<Result category="OUT">
<Value/>
</Result>
</Connect>
<Disconnect category="OPERATION" operationId="4">
<Parameter category="IN">
<Value/>
</Parameter>
<Result category="OUT">
<Value/>
</Result>
</Disconnect>
<GetRobot category="OPERATION" operationId="7">
<Parameter category="IN">
<Value>
<Name>VS</Name>
<Option/>
</Value>
</Parameter>
<Result category="OUT">
<Value>100</Value>
</Result>
</GetRobot>
</CaoProvController>
<CaoProvRobot category="INTERFACE" funcId="104">
<Move category="OPERATION" operationId="72">
<Parameter category="IN">
<Value>
<Interpolation>0</Interpolation>
<Pose>P11</Pose>
<Option/>
</Value>
</Parameter>
<Result category="OUT">
<Value/>
</Result>
</Move>
</CaoProvRobot>
</Provider>
</DCD1>
```

```
<DCD2 category="DCD" dllPath="CaoProv.DataStore" ProviderVersion="1"
xmlns="gost.e-ecolog.ru ">
<ORiN_Header>
<DCD_Version>1</DCD_Version>
<DeviceVersion>1</DeviceVersion>
<ProviderName/>
<ProviderVersion>1</ProviderVersion>
<Factory/>
<DIT/>
<ORiN_Version>
<Major>1</Major>
<Minor>1</Minor>
<Revision>1</Revision>
</ORiN_Version>
</ORiN_Header>
<Provider category="MODULE" moduleId="0" provider="CaoProv.DCD2">
<CaoProvController category="INTERFACE" funcId="101">
<Connect category="OPERATION" operationId="3">
<Parameter category="IN">
<Value>
<Name>DS</Name>
<Option/>
</Value>
</Parameter>
<Result category="OUT">
<Value/>
</Result>
</Connect>
<Disconnect category="OPERATION" operationId="4">
<Parameter category="IN">
<Value/>
</Parameter>
<Result category="OUT">
<Value/>
</Result>
</Disconnect>
<GetVariable category="OPERATION" operationId="9">
<Parameter category="IN">
<Value>
<Name>@Vars</Name>
<Option>ID=10</Option>
</Value>
</Parameter>
<Result category="OUT">
<Value>123</Value>
</Result>
</GetVariable>
</CaoProvController>
<CaoProvVariable category="INTERFACE" funcId="106">
<Attribute category="ATTRIBUTE" readonly="true">
<Value>0</Value>
</Attribute>
<Value category="ATTRIBUTE" readonly="false">
<Value>ABC</Value>
</Value>
</CaoProvVariable>
</Provider>
</DCD2>
</CCD>
</ProfileBody>
</ISO15745Profile>
```

Приложение ДА
(справочное)

Сведения о соответствии ссылочных международных стандартов ссылочным национальным стандартам Российской Федерации

Таблица ДА. 1

Обозначение ссылочного международного стандарта	Степень соответствия	Обозначение и наименование соответствующего национального стандарта
ИСО 15745-1	IDT	ГОСТ Р ИСО 15745-1—2010 «Системы промышленной автоматизации и интеграция. Прикладная интеграционная среда открытых систем. Часть 1. Общее эталонное описание»
ИСО 20242-1:2005	IDT	ГОСТ Р ИСО 20242-1—2010 «Системы промышленной автоматизации и интеграция. Служебный интерфейс для испытательных прикладных программ. Часть 1. Общие положения»
ИСО 20242-3	IDT	ГОСТ Р ИСО 20242-3—2012 «Системы промышленной автоматизации и интеграция. Служебный интерфейс для испытательных прикладных программ. Часть 3. Служебный интерфейс виртуального устройства»
Примечание — В настоящей таблице использовано следующее условное обозначение степени соответствия стандартов: IDT — идентичные стандарты.		

Библиография

[1] ИСО 15745 (все части) Системы промышленной автоматизации и интеграция. Прикладная среда интегрирования открытых систем
(ISO 15745 (all parts)) (Industrial automation systems and integration - Open systems application integration framework)

[2] ИСО/МЭК 19501:2005 Информационные технологии. Открытая распределительная обработка. Унифицированный язык моделирования (UML). Версия 1.4.2
(ISO/IEC 19501:2005) (Information technology - Open Distributed Processing - Unified Modeling Language (UML) Version 1.4.2)

[3] Generic Device Interface Version 4.4 — Association for Standardization of Automation and Measuring Systems (ASAM)

[4] OASIS PPS-1, Production Planning and Scheduling Part 1: Core Elements, Ver1.0 Public Review Draft 01, 7th August, 2007

[5] OASIS PPS-2, Production Planning and Scheduling Part 2: Transaction Messages, Ver1.0 Public Review Draft 01, 7th August 2007

[6] ORiN 2.1 Specifications, Version 2.1, ORiN Forum

[7] REC-xml-20040204, Extensible Markup Language (XML) 1.0 Third Edition — W3C Recommendation 04 February 2004

[8] REC-xmlschema-1-20041028, XML Schema Part 1: Structures Second Edition — W3C Recommendation 28 October 2004

[9] REC-xmlschema-2-20041028, XML Schema Part 2: Datatypes Second Edition — W3C Recommendation 28 October 2004

УДК 658.52.011.56:006.354 ОКС 25.040.40 Т58

Ключевые слова: автоматизированные промышленные системы, интеграция, жизненный цикл систем, управление производством

Редактор *Т. А. Леонова*
Технический редактор *Е. В. Беспрозванная*
Корректор *В. Е. Нестерова*
Компьютерная верстка *Е. Н. Евтеевой*

Сдано в набор 07.10.2014. Подписано в печать 16.12.2014. Формат 60×84 1/8. Бумага офсетная. Гарнитура Ариал. Печать офсетная. Усл. печ. л. 8,37. Уч.-изд. л. 7,50. Тираж 40 экз. Зак. 1738

ФГУП «СТАНДАРТИНФОРМ», 123995 Москва, Гранатный пер., 4.

gost.e-ecolog.ru gost.e-ecolog.ru
Набрано и отпечатано в Калужской типографии стандартов, 248021 Калуга, ул. Московская, 256.

gost.e-ecolog.ru