

ALT Linux
ADTS / ТЕХНИЧЕСКАЯ ДОКУМЕНТАЦИЯ

SDMX/SDML

Спецификация формата

Версия 1.0

Содержание

1 Назначение и область применения 3
1.1 Нормативная терминология 3
1.2 Нормативный статус 3
1.3 Термины 3
1.4 Соответствие спецификации 4
2 Обзор архитектуры 4
2.1 Абстрактное состояние 4
2.2 Обработка комплекта 5
3 Комплект и пространства имён 5
3.1 Общие лексические правила 6
4 Документ SDMX 6
4.1 Корневой элемент и разделы 6
4.2 Политика 7
5 Элементы данных 7
5.1 Логическое значение 8
5.2 Целое значение 8
5.3 Перечисление и флаги 8
5.4 Строки и списки 9
5.5 Коллекция 9
6 Назначения и типы хранения 10
6.1 `sectionValue` 10
6.2 `registryValue` 11
6.3 Составные назначения коллекций 11
7 Ограничения между политиками 12
7.1 Профиль OCL `sdmx-1.0` 12
7.2 Уведомления 13
8 Документ SDML 13
8.1 Таблица строк 14
8.2 Таблица представлений 14
9 Построение `GptTmpl.inf` 15
9.1 Общие правила файла 15
9.2 Минимальная оболочка 16
9.3 Строки и principal 16
9.4 Примеры wire-записей 16
9.5 Особенности целевых параметров 17
10 Безопасность реализации 17
11 Ссылки 18
11.1 Нормативные источники 18
11.2 Справочные источники 18
12 Приложение A. Схема SDMX 1.0 19
13 Приложение B. Схема SDML 1.0 26

1 Назначение и область применения

SDMX (Security Definition Markup XML) описывает машинную модель параметров безопасности: типы данных, ограничения и назначение значений в шаблоне безопасности. SDML (Security Definition Markup Language) содержит локализованные строки и формы представления этих данных. Разделение модели и представления следует архитектуре ADMX/ADML: политика ссылается на `presentation`, а элементы управления связываются с машинными элементами через `refId`.

SDMX/SDML не является форматом Group Policy компании Microsoft. Результатом обработки является `GptTmpl.inf`, синтаксис и применение которого определены в MS-GPSB, раздел 2.2 «Message Syntax». Связь security policy с административным UI, INF-файлом и клиентским расширением описана в MS-GPSB, раздел 1.3.2 «Security Extension Overview».

Формат не содержит произвольного исполняемого кода, сведений об источнике определения, истории изменения и состояния применения политики. Все варианты синтаксиса закрыты XSD. Неизвестный элемент или атрибут является ошибкой.

1.1 Нормативная терминология

Слова «ДОЛЖЕН», «НЕ ДОЛЖЕН», «СЛЕДУЕТ» и «МОЖЕТ» задают обязательность требования. Они соответствуют MUST, MUST NOT, SHOULD и MAY из RFC 2119.

«Документ» означает один XML-файл. «Комплект» означает все SDMX и SDML-файлы, загруженные приложением. «Определённая политика» означает, что пользователь явно включил её в создаваемую конфигурацию и предоставил значения элементов. Неопределённая политика не равна политике с начальными значениями.

1.2 Нормативный статус

Нормативными являются XML-модель, правила загрузки комплекта, разрешения ссылок, проверки значений, вычисления ограничений и построения INF, а также обе XSD из приложений. Примеры иллюстративны. Конкретные SID, SDDL, пути и значения в них не являются рекомендациями по настройке системы.

XSD является источником истины для локальной грамматики документа: порядка и кратности элементов, типов атрибутов и закрытых вариантов. Основной текст задаёт правила, которые XSD 1.0 выразить не может: связи между документами, совместимость элементов и назначений, профиль OCL и физическое кодирование INF. Документ соответствует версии 1.0 только при выполнении обоих слоёв требований.

1.3 Термины

Термин	Значение
Определение	Один SDMX и соответствующие ему SDML выбранных локалей.
Политика	Элемент `securityPolicy`, объединяющий ресурсы, данные и ограничения одного параметра безопасности.
Элемент данных	Типизированное машинное значение внутри `elements` либо поля коллекции.
Назначение	Структурированное описание записи элемента в INF шаблона безопасности.
Представление	Локализованная форма `presentation` в SDML, связанная с элементами данных через `refId`.
Конфигурация	Отображение расширенных имён политик в состояния `Undefined` либо `Defined`.
Principal	Имя учётной записи или группы либо каноническая строка SID.
Wire-запись	Промежуточная запись `(section, optional key, value)` до сборки `GptTmpl.inf`.

1.4 Соответствие спецификации

Реализация чтения соответствует версии 1.0, если она:

1. принимает документы только с XML namespace и атрибутами версии 1.0;
2. проверяет каждый документ соответствующей XSD до семантической обработки;
3. проверяет уникальность пространств имён и ссылочную целостность комплекта;
4. строит типизированную модель элементов, назначений и представлений;
5. проверяет статические типы выражений профиля OCL;
6. формирует INF по правилам назначения и физического типа;
7. отклоняет неизвестные варианты модели и не исполняет XML как код.

XSD проверяет локальный синтаксис отдельного файла. Уникальность, разрешение межфайловых ссылок, совместимость типов и полнота отображений ДОЛЖНЫ проверяться после XSD. Ошибка одного определения запрещает выдачу частичного INF.

2 Обзор архитектуры

SDMX описывает идентичность политики, рекурсивную модель её данных и назначения значений. SDML описывает локализованные строки и один из возможных интерфейсов для этих данных. Начальные значения, диапазоны, допустимые варианты и машинные отображения находятся в SDMX и не зависят от локали или наличия контрола.

```
SDMX + SDML
    -> проверенный комплект определений
    -> пользовательская конфигурация
    -> локальные ограничения и constraints OCL
    -> назначения элементов
    -> wire-записи
    -> GptTmpl.inf
    -> обработчик MS-GPSB
```

Переключатель состояния `Defined`, заголовок политики, прокручиваемое объяснение и действия формы принадлежат общей оболочке приложения. `presentationTable` описывает только контролы значений. Виджеты Adwaita, размеры и визуальные стили не являются частью формата.

Версия 1.0 применима к параметрам безопасности, которые выражаются поддерживаемыми элементами данных и назначениями в security policy INF. Настройки, требующие чтения текущего состояния машины, произвольного кода или другого wire-формата, находятся вне области применения.

2.1 Абстрактное состояние

Объект	Содержимое
`DefinitionSet`	Индексы пространств имён, политик, элементов, строк и представлений принятого комплекта.
`SecurityPolicyState`	`Undefined` либо `Defined(ElementStates)`.
`ElementState`	`Unset` либо `Set(Value)` для каждого корневого элемента и присутствующего поля.
`Configuration`	Отображение расширенного имени политики в `SecurityPolicyState`.
`WireRecord`	`section`, необязательный `key`, строковый `value` и происхождение.
`Diagnostic`	Класс, код, сообщение, файл и, если известны, securityPolicy и element.

Начальное значение применяется UI только при создании соответствующего значения: при переходе политики из `Undefined` в `Defined` либо при добавлении записи коллекции. Если инициализатор отсутствует, элемент остаётся `Unset` до ввода пользователем или загрузки значения. Открытие

определения, загрузка конфигурации или создание INF НЕ ДОЛЖНЫ синтезировать пустую строку, пустой список, дескриптор или другое значение.

`Defined` с `Unset` допустим как незавершённое состояние редактора, но не как вход для экспорта. При экспорте каждый корневой элемент и каждое обязательное поле ДОЛЖНЫ иметь значение. Необязательное поле может отсутствовать; это не равно присутствующему полю с пустым значением.

2.2 Обработка комплекта

Загрузка выполняется атомарно:

1. обнаружить SDMX и одноимённые SDML выбранной локали;
2. безопасно разобрать XML и проверить XSD;
3. построить индексы пространств имён, политик, элементов и ресурсов;
4. разрешить ссылки, проверить типы назначений и привязки представлений;
5. разобрать и статически типизировать ограничения OCL;
6. принять весь комплект только при отсутствии ошибок.

Для построения результата реализация сначала проверяет полноту обязательных значений и локальные ограничения, затем вычисляет все активные constraints над одним неизменяемым снимком конфигурации, проверяет `inputDomain`, создаёт wire-записи всех назначений и проверяет их конфликты. Только после этого собирается INF.

Ошибки определения, конфигурации и построения INF запрещают выдачу нового файла. `notice` является локализованным уведомлением и не меняет допустимость конфигурации.

3 Комплект и пространства имён

Комплект строится по одинаковому basename:

```
definitions/
  account-password.sdmx
  en-US/account-password.sdml
  ru-RU/account-password.sdml
```

Для выбранной локали каждый SDMX ДОЛЖЕН иметь ровно один SDML с тем же basename. Локаль определяется каталогом; внутри SDML отдельного атрибута локали нет. Реализация МОЖЕТ определить цепочку резервного выбора локалей, но она не является частью формата.

Оба вида документа используют общее пространство имён:

```
http://www.altlinux.org/GroupPolicy/SecurityDefinitions
```

Корневые атрибуты `schemaVersion` и `revision` обязательны. `schemaVersion` фиксирован значением `1.0`; `revision` имеет тип `major.minor`, а документы данного выпуска используют редакцию `1.0`. Эти атрибуты разделяют версию грамматики и редакцию документа. У корня SDMX дополнительно обязателен `format="securityTemplate"`.

`securityNamespaces/target/@namespace` идентифицирует конкретное определение и ДОЛЖЕН быть уникален в комплекте. Это значение не совпадает по назначению с XML namespace схемы. `target/@prefix` и `using/@prefix` являются псевдонимами пространств политик и не объявляют XML-префиксы. Все псевдонимы документа ДОЛЖНЫ различаться, а namespace каждого `using` ДОЛЖЕН принадлежать загруженному SDMX того же комплекта.

`using` разрешает квалифицированные ссылки на политики другого определения. Неквалифицированная ссылка разрешается в target namespace текущего SDMX; `prefix:id` — в namespace соответствующего `using`. Все ссылки разрешаются регистрозависимо.

3.1 Общие лексические правила

SDMX и SDML являются XML 1.0. Писатель ДОЛЖЕН выдавать UTF-8 без BOM. Комментарии и processing instructions не входят в модель. DTD, XInclude, `xsi:nil`, смешанное содержимое и элементы из чужих пространств имён не предусмотрены.

Машинные идентификаторы соответствуют шаблону:

```
[a-z][a-z0-9_.-]*
```

Идентификаторы политик уникальны в SDMX, элементов — внутри политики, вариантов — внутри enum или flags, полей — внутри коллекции. Сравнение регистрозависимо по Unicode code point.

Строковые ресурсы используют типы `pd:resourceID`, `pd:stringReference` и `pd:presentationReference` из базовой схемы ADMX. Ссылка занимает всё значение атрибута:

```
$(string.minimum_password_length)
$(presentation.minimum_password_length)
```

Конкатенация ссылок с литералами не выполняется. Каждый `string` и `presentation` ДОЛЖЕН существовать ровно один раз в одноимённом SDML выбранной локали.

Совпадающие по назначению типы используются непосредственно из ADMX: `pd:LocalizedString` для ресурсов `string`, `pd:registryKey` и `pd:registryValueName` для атрибутов `registryValue/@key` и `@valueName`, `pd:versionString` для `schemaVersion`, `revision` и `minRequiredRevision`, `pd:PolicyNamespaceAssociation` для `target` и `using`. Версия схемы дополнительно фиксируется значением `1.0`. Непустое содержимое строковых ресурсов, ключей и имён значений реестра проверяется семантическим валидатором комплекта.

Собственные типы сохраняются при отличии модели: машинные идентификаторы допускают точки и дефисы, целые значения являются знаковыми 64-битными, а элементы управления содержат привязки и атрибуты SDML. Составные типы с дочерними элементами SDMX/SDML также сохраняют своё пространство имён.

4 Документ SDMX

4.1 Корневой элемент и разделы

```
<securityDefinitions
    xmlns="http://www.altlinux.org/GroupPolicy/SecurityDefinitions"
    schemaVersion="1.0" revision="1.0" format="securityTemplate">
  <securityNamespaces>
    <target prefix="account_password"
            namespace="urn:altlinux:sdmx:policies:account-password"/>
  </securityNamespaces>
  <resources minRequiredRevision="1.0" fallbackCulture="en-US"/>
  <categories>
    <category id="account.password"
              displayName="$(string.category)"/>
  </categories>
  <securityPolicies>...</securityPolicies>
</securityDefinitions>
```

Порядок разделов фиксирован: `securityNamespaces`, `resources`, `categories`, `securityPolicies`. `resources/@minRequiredRevision` задаёт минимальную допустимую редакцию SDML, а `fallbackCulture` — культуру ресурсов по умолчанию. Каталог этой культуры ДОЛЖЕН присутствовать в комплекте.

`category` имеет `id`, `displayName` и необязательный `parentCategory/@ref`. Родитель ДОЛЖЕН разрешаться в текущем или объявленном namespace, а граф категорий ДОЛЖЕН быть ацикличным.

Редакции имеют форму `major.minor` и сравниваются как пары целых чисел. `revision` выбранного SDML ДОЛЖЕН быть не меньше `minRequiredRevision` SDMX.

4.2 Политика

Атрибуты политики:

Атрибут	Назначение
`id`	Стабильный машинный идентификатор политики.
`displayName`	Локализованное название.
`explainText`	Развёрнутое локализованное объяснение.
`presentation`	Ссылка `$(presentation.ID)` на форму SDML.
`allowedValues`	Необязательное дополнительное описание допустимых значений.
`defaultText`	Необязательное человекочитаемое описание начального значения.

Внутри `securityPolicy` следуют `parentCategory`, `elements`, необязательные `constraints` и `notices`. Политика содержит один или более корневых элементов данных. Их `id` обязательны и уникальны внутри политики. Несколько элементов позволяют одной политике описывать составное значение без искусственной привязки модели к одной форме UI.

`allowedValues` и `defaultText` являются только локализованными пояснениями и не задают ограничения или значения. Если у политики нет применимого инициализатора, `defaultText` СЛЕДУЕТ опустить; в частности, он не может подменять отсутствующий начальный SDDL.

```
<securityPolicy id="account.password.minimum_password_length"
    displayName="$(string.minimum_password_length)"
    explainText="$(string.minimum_password_length_help)"
    presentation="$(presentation.minimum_password_length)">
  <parentCategory ref="account.password"/>
  <elements>
    <integer id="value" initialValue="0">
      <range min="0" max="14"/>
      <sectionValue section="System Access"
                    key="MinimumPasswordLength" type="integer"/>
    </integer>
  </elements>
</securityPolicy>
```

5 Элементы данных

Тип элемента определяет логическое значение конфигурации. Назначение задаёт его физическое представление в INF. Элемент управления UI не влияет на тип, обязательность или способ записи.

Одна и та же закрытая группа типов используется для корневых элементов и полей коллекций: `boolean`, `integer`, `enum`, `flags`, `string`, `sddl`, `stringList`, `principal`, `principalList`, `serviceName`, `registryKey`, `filePath`, `collection`. Поэтому коллекции могут быть вложенными, а состав записи не ограничен заранее заданным набором типов.

У корневого элемента `id` обязателен. Внутри `field` идентичность задаёт `field/@id`: `id` вложенного элемента либо отсутствует, либо совпадает с ним. Идентификаторы корневых элементов политики и полей одной коллекции уникальны в своей области.

Инициализатором считается `initialValue`, `initialOption` либо набор `initialSelection`. Если `initialValue` или `initialOption` не объявлен, начальное состояние равно `Unset`. Для `flags` отсутствие `initialSelection` означает заданное пустое множество. Типы без конструкции инициализации — в том числе SDDL, списки и коллекции — требуют явного или загруженного значения.

5.1 Логическое значение

`boolean` хранит логическое значение. `initialValue` необязателен. Элементы `trueValue` и `falseValue` задают его машинное отображение и либо оба отсутствуют, либо оба присутствуют. Каждый содержит один литерал: `integer/@value`, текстовый `string` либо `binary` в лексической форме `xs:hexBinary`. Типы двух литералов совпадают и совместимы со всеми назначениями элемента.

```
<boolean id="value" initialValue="false">
  <trueValue><binary>01</binary></trueValue>
  <falseValue><binary>00</binary></falseValue>
  <registryValue hive="HKEY_LOCAL_MACHINE"
      key="System\CurrentControlSet\Control\Lsa"
      valueName="FullPrivilegeAuditing" type="REG_BINARY"/>
</boolean>
```

5.2 Целое значение

`integer` хранит знаковое 64-битное логическое значение. Он может содержать ноль или более включительных `range`. При отсутствии `range` разрешён весь диапазон `int64`. Объявленные диапазоны не пересекаются, для каждого выполняется `min <= max`. Необязательный `initialValue` входит в их объединение.

Ноль или более `substitute/@input,@output` выполняют точечное преобразование перед записью. `input` уникален. Если совпадения нет, используется исходное значение. Необязательный `inputDomain` содержит один или более диапазонов и проверяется до подстановки. Он выражает ограничение целевого параметра, которое может быть уже диапазона редактора.

```
<integer id="value" initialValue="0">
  <range min="0" max="999"/>
  <substitute input="0" output="-1"/>
  <inputDomain><range min="0" max="999"/></inputDomain>
  <sectionValue section="System Access"
                key="MaximumPasswordAge" type="integer"/>
</integer>
```

5.3 Перечисление и флаги

`enum` содержит один или более `option`. Вариант имеет уникальный `id`, необязательный локализованный `displayName` и необязательный типизированный литерал `value`. При отсутствии `displayName` UI использует `id` как подпись. Необязательный `initialOption` ссылается на существующий вариант. Значением конфигурации является идентификатор варианта; при наличии назначения оно получает литерал выбранного варианта.

`flags` содержит один или более вариантов. При числовом назначении каждый вариант содержит целочисленный литерал. Подпись варианта задаётся и резервно выбирается так же, как у `enum`. Значением является множество идентификаторов, а выходное число равно побитовому OR соответствующих неотрицательных значений `uint32`. Каждая маска ненулевая, и маски разных вариантов не пересекаются. `initialSelection/@option` задаёт начальное множество без повторов.

```
<flags id="value">
  <option id="require_ntlmv2_session_security"
      displayName="$(string.ntlmv2)">
    <value><integer value="524288"/></value>
  </option>
  <option id="require_128_bit_encryption"
      displayName="$(string.encryption_128)">
    <value><integer value="536870912"/></value>
  </option>
  <registryValue hive="HKEY_LOCAL_MACHINE"
      key="System\CurrentControlSet\Control\Lsa\MSV1_0"
      valueName="NTLMMinClientSec" type="REG_DWORD"/>
</flags>
```

5.4 Строки и списки

`string` хранит строку и может иметь `initialValue`. `sddl` хранит Security Descriptor Definition Language и не допускает `initialValue`: новый редактор SDDL всегда начинается с `Unset`, пока дескриптор не введён или не загружен. Пустая строка не подставляется вместо отсутствующего дескриптора. Приложение ДОЛЖНО проверить синтаксис заданного SDDL по MS-DTYP §2.5.1.1.

`stringList` хранит упорядоченный список строк. `principalList` хранит упорядоченный список principal, а `principal` — одно такое значение. `allowedKind="group"` ограничивает значение группами; при отсутствии `allowedKind` разрешены поддерживаемые виды principal. SID ДОЛЖЕН соответствовать MS-DTYP §2.4.2.1.

`serviceName`, `registryKey` и `filePath` являются отдельными семантическими строковыми типами для внутреннего имени службы, полного пути раздела реестра и пути файла или каталога. Их нельзя взаимозаменять только потому, что все они представлены строкой. Пустой список или пустая строка являются заданными значениями и отличаются от `Unset`.

5.5 Коллекция

`collection` хранит упорядоченный список однотипных записей. Каждое `field` имеет обязательный `id`, необязательный `displayName`, атрибут `required` со значением по умолчанию `true` и ровно один элемент любого типа из общей группы. Вложенный `collection` тем самым создаёт рекурсивную структуру конечной глубины, заданной самим XML-документом.

Необязательный `uniqueBy` ссылается на обязательное поле текущей записи и требует уникальности его значений между записями коллекции. `duplicateResource` при наличии задаёт локализованную диагностику нарушения. Отсутствие `uniqueBy` означает, что формат не вводит собственного ключа уникальности для этой коллекции.

Обязательное поле ДОЛЖНО иметь значение перед экспортом. Необязательное поле может отсутствовать; присутствующее пустое значение остаётся значением. Назначения коллекции следуют после полей и не повторяют id владеющей коллекции.

6 Назначения и типы хранения

Назначение отделяет логический тип элемента от физического формата записи. Элемент может иметь ноль, одно или несколько назначений. Элемент без назначения является чистыми данными: он участвует в UI, конфигурации и constraints, но сам не создаёт wire-записей. Несколько назначений получают одно логическое значение и обрабатываются в порядке объявления. Каждое из них ДОЛЖНО быть совместимо с типом элемента и его машинным отображением.

Для boolean обязательны оба отображения `trueValue`/`falseValue`, если хотя бы одно назначение требует литерал. Для enum и flags каждый вариант ДОЛЖЕН иметь `value`, если его требует хотя бы одно назначение. У data-only элемента эти отображения могут отсутствовать.

Допустимые сочетания закрыты:

Назначение	Элементы	Требуемое представление
`sectionValue/integer`	`integer`, `boolean`, `enum`, `flags`	Целое; для boolean и enum — целочисленные литералы, для flags — маски.
`sectionValue/text`	`string`, `sddl`, `principal`, `serviceName`, `registryKey`, `filePath`	Строка.
`sectionValue/text`	`boolean`, `enum`	Строковые литералы.
`sectionValue/principalList`	`principalList`	Список principal.
`registryValue/REG_DWORD`	`integer`, `boolean`, `enum`, `flags`	`uint32`; отображения имеют целочисленные литералы.
`registryValue/REG_SZ`	`string`, `sddl`, `principal`, `serviceName`, `registryKey`, `filePath`	Строка.
`registryValue/REG_SZ`	`boolean`, `enum`	Строковые литералы.
`registryValue/REG_SZ`	`integer`	Десятичный текст и обязательный `storeAsText="true"`.
`registryValue/REG_EXPAND_SZ`	`string`, `sddl`, `principal`, `serviceName`, `registryKey`, `filePath`	Строка.
`registryValue/REG_EXPAND_SZ`	`boolean`, `enum`	Строковые литералы.
`registryValue/REG_BINARY`	`boolean`, `enum`	Непустые бинарные литералы.
`registryValue/REG_MULTI_SZ`	`stringList`	Список непустых строк.

Другие сочетания являются ошибкой определения. Для boolean типы `trueValue` и `falseValue` совпадают. Для enum типы значений всех вариантов совпадают.

6.1 `sectionValue`

`section` и `key` задают ключевую запись. `type` имеет значения:

- `integer` — каноническое десятичное целое без кавычек;
- `text` — результат `infQuote`;
- `principalList` — элементы `infPrincipal`, соединённые запятыми.

Совместимы соответственно числовой литерал, строковое значение и `principalList`. Определённый пустой список principal создаёт пустую правую часть, например `SeServiceLogonRight =`. Такая форма не входит в `SidList` MS-GPSB §2.2.6 и является явно заявленным расширением wire-профиля 1.0; совместимость целевой системы ДОЛЖНА быть проверена до применения.

```
<principalList id="value">
  <sectionValue section="Privilege Rights"
                key="SeServiceLogonRight" type="principalList"/>
</principalList>
```

6.2 `registryValue`

`registryValue` всегда создаёт запись в `[Registry Values]` и задаёт адрес тремя частями: `hive`, `key`, `valueName`. В версии 1.0 `hive` равен `HKEY_LOCAL_MACHINE`; `key` не содержит hive и конечное имя значения.

type	Код INF	Payload
REG_SZ	1	`infQuote(string)` либо каноническое десятичное целое при `storeAsText="true"`.
REG_EXPAND_SZ	2	`infQuote(string)`.
REG_BINARY	3	Десятичные байты через запятую.
REG_DWORD	4	Беззнаковое десятичное `uint32`.
REG_MULTI_SZ	7	`infQuote` каждого элемента через запятую.

Полное wire-имя строится как `MACHINE\` плюс `key`, обратная косая черта и `valueName`. Формат `name=type,value` и коды типов определены в MS-GPSB §2.2.5.

`storeAsText="true"` обязателен для целого с `REG_SZ` и запрещён для остальных сочетаний: после подстановки значение записывается каноническими десятичными цифрами без кавычек. Значения строковых семантических типов и строковые литералы boolean/enum проходят через `infQuote`.

```
<integer id="value" initialValue="10">
  <range min="0" max="50"/>
  <registryValue hive="HKEY_LOCAL_MACHINE"
      key="Software\Microsoft\Windows NT\CurrentVersion\Winlogon"
      valueName="CachedLogonsCount" type="REG_SZ" storeAsText="true"/>
</integer>
```

Для значения `10` получается:

```
[Registry Values]
MACHINE\Software\Microsoft\Windows NT\CurrentVersion\Winlogon\CachedLogonsCount=1,10
```

При `REG_MULTI_SZ` каждый элемент ДОЛЖЕН быть непустым. Пустой список допустим и создаёт только префикс `7,`. REG_MULTI_SZ сохраняет порядок элементов. Для `REG_BINARY` каждая пара шестнадцатеричных цифр задаёт один байт; байты выводятся в исходном порядке как беззнаковые десятичные числа через запятую. Для `REG_DWORD` итог после подстановки или отображения ДОЛЖЕН находиться в диапазоне 0..4 294 967 295.

6.3 Составные назначения коллекций

`groupRecords` ссылается через `groupField` на обязательное поле `principal` с `allowedKind="group"`. Его дочерние `members` и `memberOf` ссылаются на необязательные `principalList`-поля; `emit` в версии 1.0 равен `when-present`. Присутствующее поле создаёт запись, отсутствующее — нет. Пустой присутствующий список создаёт пустую правую часть.

```
<groupRecords groupField="group">
  <members listField="members" emit="when-present"/>
  <memberOf listField="member_of" emit="when-present"/>
</groupRecords>
```

Ключи равны `infPrincipal(group) + "__Members"` и `infPrincipal(group) + "__Memberof"`. Семантика и формат заданы в MS-GPSB §2.2.10.

`serviceRecord` ссылается на поля `serviceName`, `enum` режима запуска и `sddl`. Числовые литералы вариантов `enum` задают режим запуска. Для каждой записи в `[Service General Setting]` создаётся безключевая строка:

```
infQuote(service-name),startup-value,infQuote(sddl)
```

Значения `2` Automatic, `3` Manual и `4` Disabled и формат определены в MS-GPSB §2.2.8.

`objectRecord` ссылается на путь, `enum` режима распространения и SDDL. При `target="registry"` путь имеет тип `registryKey`, секция равна `[Registry Keys]`; при `target="file"` путь имеет тип `filePath`, секция равна `[File Security]`. Для каждой записи создаётся:

```
infQuote(path),propagation-value,infQuote(sddl)
```

Значения `0` propagate, `1` replace children и `2` protect описаны в MS-GPSB §2.2.7 и MS-GPSB §2.2.9.

Роли составного назначения разрешаются относительно записи владеющей коллекции. Роли `serviceRecord` и `objectRecord`, а также `groupField`, ссылаются на обязательные поля требуемых типов. `members` и `memberOf` ссылаются на `principalList`; для `memberOf` допустимы только группы. Поля, не упомянутые данным назначением, остаются обычными данными и могут использоваться другим назначением, вложенным UI или constraints.

Назначения вложенного элемента вычисляются в контексте текущей записи коллекции. Если несколько записей или назначений создают один и тот же физический `(section, key)`, применяется обычное правило конфликта: экспорт отклоняется даже при совпадении значений.

7 Ограничения между политиками

Локальные ограничения — диапазоны, обязательность, варианты и уникальность — задаются структурой элементов. `constraints` предназначен для зависимостей между политиками и содержит один или более `constraint`.

```
<constraints>
  <constraint
      id="rule_1"
      language="OCL"
      profile="sdmx-1.0"
      message="$(string.max_service_age_dependencies)">
self.securityPolicy('account.kerberos.max_ticket_age').isDefined and
self.integer('account.kerberos.max_service_age').value &lt;=
self.integer('account.kerberos.max_ticket_age').value * 60</constraint>
</constraints>
```

Ограничение вычисляется только когда владеющая политика находится в состоянии `Defined`. Контекст `self` — неизменяемая `Configuration` текущего документа. Идентификатор ограничения уникален внутри политики. `language` равен `OCL`, `profile` — `sdmx-1.0`, а `message` ссылается на локализованную диагностику.

7.1 Профиль OCL `sdmx-1.0`

Профиль является закрытым подмножеством выражений OCL 2.4 с типизированными операциями доступа к модели SDMX. Он не заявляет поддержку полного языка OCL.

Поддерживаются:

- литералы `true`, `false`, `null`, `invalid`, целые и строки;
- скобки и `if ... then ... else ... endif`;
- `not`, `and`, `or`, `implies`;
- `=`, `<>`, `<`, `<=`, `>`, `>=`, `+`, `-`, `*`;
- `self.securityPolicy('ID').isDefined`;
- `self.integer('ID').value`;

- `self.enumeration('ID').value`;
- `self.integer('ID', 'ELEMENT').value`;
- `self.enumeration('ID', 'ELEMENT').value`.

Приоритет операторов от высокого к низкому: унарные `not` и `-`, затем `*`, `+`/`-`, отношения `<`/`<=`/`>`/`>=`, равенство `=`/`<>`, `and`, `or`, `implies`. Все инфиксные операторы левоассоциативны. Скобки изменяют порядок вычисления. Эти правила соответствуют порядку из OCL 2.4 §7.4.8.

Строковый литерал заключается в одинарные кавычки. Поддерживаются escape-последовательности `\\`, `\'`, `\n`, `\r`, `\t`. Последовательность `--` начинает комментарий до конца строки. Другие escape-последовательности запрещены.

Аргумент `ID` является строковым литералом и разрешается статически. Допустимы квалифицированные ссылки через псевдонимы `using`. `securityPolicy` принимает политику любого типа. Одноаргументные `integer` и `enumeration` допустимы только для политики с единственным корневым элементом соответствующего типа. В двухаргументной форме `ELEMENT` статически выбирает корневой элемент по его `id`; к полям вложенных коллекций эта форма не обращается. Значение `enumeration(...).value` — id выбранного варианта, а не его wire-литерал. `securityPolicy(...).isDefined` всегда возвращает Boolean, в том числе `false` для неопределённой политики. `true` означает наличие состояния политики, но не гарантирует, что каждый её элемент уже получил значение.

Чтение `.value` неопределённой политики или элемента в состоянии `Unset` даёт `invalid`. Поэтому выражение ДОЛЖНО явно охранять чтение политик, которые могут быть `Undefined`, проверкой `.isDefined`; полнота обязательных значений отдельно проверяется перед экспортом. Все ссылки на состояния, включая значение самой владеющей политики, задаются явно; неявного `self.value` нет.

`allInstances`, итераторы коллекций, циклы, рекурсия, динамические запросы, пользовательские операции и навигация, отсутствующая в списке профиля, запрещены. Целочисленная арифметика математическая; реализация НЕ ДОЛЖНА допускать тихое переполнение машинного типа. Ограничения выходного диапазона проверяются перед созданием назначения.

Длина выражения не превышает 32 768 символов, число токенов — 4096, глубина синтаксической и абстрактной синтаксической структуры — 128. Превышение любого предела является ошибкой определения.

Результат `true` означает успех, `false` — нарушение с сообщением `message`, а `null` или `invalid` — ошибку вычисления конфигурации. Короткое замыкание логических операций допускается только в соответствии с семантикой OCL и не может изменять итоговый результат. В частности, `false and invalid` даёт `false`, а `true or invalid` — `true`; доминирующее логическое значение маскирует `invalid` независимо от порядка вычисления операндов.

Нормативная семантика базовых операторов и специальных значений следует OMG OCL 2.4. Типы и операции доступа к модели определены настоящей спецификацией.

7.2 Уведомления

`notices` содержит один или более `notice/@message`. Уведомление показывается, когда владеющая политика `Defined`, не вычисляет выражение и не запрещает сохранение. Машинные правила НЕ ДОЛЖНЫ извлекаться из текста уведомления. `notice` относится к отдельной информационной степени серьёзности: он не является ни нарушением constraint, ни ошибкой определения или конфигурации.

8 Документ SDML

SDML содержит локализованные строки и представления. Он не повторяет типы, диапазоны, варианты и начальные значения SDMX.

```
<securityDefinitionResources
    xmlns="http://www.altlinux.org/GroupPolicy/SecurityDefinitions"
    schemaVersion="1.0" revision="1.0">
  <resources>
    <stringTable>
      <string id="minimum_password_length">Minimum password length</string>
      <string id="length_label">Number of characters:</string>
      <string id="password_not_required">Password is not required.</string>
    </stringTable>
    <presentationTable>
      <presentation id="minimum_password_length">
        <integerTextBox refId="value" label="$(string.length_label)">
          <labelWhen value="0" label="$(string.password_not_required)"/>
        </integerTextBox>
      </presentation>
    </presentationTable>
  </resources>
</securityDefinitionResources>
```

8.1 Таблица строк

`stringTable` содержит один или более `string`. `id` использует `pd:resourceID`; текст непуст и не состоит только из whitespace. Абзацы объяснения разделяются пустой строкой в XML character data. SDML не содержит HTML или Markdown как структурных элементов.

Каждая ссылка на строку ДОЛЖНА разрешаться в SDML выбранной локали. Разные локали могут иметь разные вспомогательные строки и структуру формы, поэтому равенство полных множеств `string/@id` не требуется. Тексты являются интерфейсными ресурсами; они не задают диапазоны, преобразования или зависимости.

8.2 Таблица представлений

`presentationTable` содержит `presentation` с уникальными `id`. Ссылка каждой политики ДОЛЖНА разрешаться в SDML выбранной локали. Представление может быть пустым либо содержать рекурсивную последовательность элементов управления. Каждый элемент управления данными имеет `refId`, который разрешается в текущей области и совместим по типу:

Контрол	Допустимый элемент
`integerTextBox`	`integer`
`booleanChoice`	`boolean`
`dropdownList`	`enum`
`flagList`	`flags`
`textBox`	`string`
`sddlEditor`	`sddl`
`listBox`	`stringList`
`principalPicker`	`principal`
`principalList`	`principalList`
`servicePicker`	`serviceName`
`registryKeyPicker`	`registryKey`
`filePicker`	`filePath`
`collectionEditor`	`collection`

В корне `presentation` областью являются корневые элементы политики. Внутри `collectionEditor` областью становятся поля одной записи связанной коллекции. Вложенный `collectionEditor` повторяет то же правило рекурсивно. `group` имеет необязательный `label`, группирует любые элементы общей control group и не меняет область разрешения `refId`.

`label` является необязательной ссылкой на строку. У `booleanChoice` обязательны `falseLabel` и `trueLabel`. `integerTextBox` имеет необязательный локализованный `suffix` и `showRange`, по умолчанию

`true`. Атрибут `readOnly` доступен у каждого элемента управления данными и по умолчанию равен `false`. При `true` контрол показывает значение, но не изменяет его.

`readOnly="true"` у `collectionEditor` запрещает изменения всей показанной коллекции и всех вложенных контролов. Дочерний `readOnly="false"` не отменяет это ограничение. Та же коллекция может редактироваться через другую доступную для записи привязку; поскольку состояние модели общее, изменения отображаются и в read-only представлении.

`labelWhen` меняет только подпись `integerTextBox`, когда логическое значение равно `value`. Он не меняет значение, допустимый диапазон или назначение. `value` ДОЛЖЕН входить в диапазон связанного integer и не повторяться внутри одного `integerTextBox`.

`text/@ref` вставляет самостоятельный локализованный текст в форму и не имеет `refId`. Он не участвует в проверке конфигурации.

Один `refId` может встречаться несколько раз. Все такие контролы показывают одно значение модели, а изменение через любой доступный для записи контрол синхронно отражается в остальных. Представление также может не ссылаться на часть элементов и полей. Значения для невидимых и `readOnly`-полей должны поступить из конфигурации, инициализатора или другого контрола; UI не создаёт для них синтетических значений. Отсутствующее обязательное значение блокирует экспорт независимо от структуры формы.

Порядок, группировка, набор контролов, повторные привязки и вспомогательный текст могут различаться между локалями одного basename. Машинная модель SDMX, типы каждой фактически используемой привязки и полнота данных перед экспортом остаются общими. Комплект данного пакета ДОЛЖЕН содержать `en-US` и `ru-RU`; другие обнаруженные каталоги локалей проходят те же проверки ссылок и типов.

9 Построение `GptTmpl.inf`

9.1 Общие правила файла

Назначения создают логические записи `(section, optional key, value)`. Итоговый файл ДОЛЖЕН использовать UTF-16LE с BOM, CRLF и preamble `[Unicode]`/`[Version]`, как требует MS-GPSB §2.2.

Key-value запись имеет вид `key = value`. Безключевая запись содержит только `value`. Неопределённая политика не создаёт записей. Определённая data-only политика также может не создать записей; это не меняет её состояния в модели. Пустая секция пропускается.

Порядок секций после preamble фиксирован:

```
System Access
Kerberos Policy
System Log
Security Log
Application Log
Event Audit
Registry Values
Privilege Rights
Registry Keys
Service General Setting
File Security
Group Membership
```

Политики обрабатываются по возрастанию `target/@namespace`, затем `securityPolicy/@id` по Unicode code point. Элементы следуют порядку объявления в SDMX, записи коллекции — порядку конфигурации, назначения одного элемента — порядку XML. Вложенные элементы вычисляются в порядке полей текущей записи. `groupRecords` создаёт сначала `__Members`, затем `__Memberof`.

Две ключевые записи с одинаковыми section и key являются ошибкой, даже если значения совпадают. Реализация СЛЕДУЕТ также диагностировать различие только в регистре. Конфликты проверяются до записи байтов.

Обычные ключевые записи и Group Membership имеют один пробел с обеих сторон `=`. Registry Values использует `name=type,value` без пробелов. Каждая строка, включая последнюю, завершается CRLF.

9.2 Минимальная оболочка

```
[Unicode]
Unicode=yes

[Version]
signature="$CHICAGO$"
Revision=1
```

Если политики не определены, результат содержит только эту оболочку. Поскольку ABNF MS-GPSB §2.2 требует хотя бы одну секцию политики, такой результат НЕ СЛЕДУЕТ передавать клиентскому расширению как обновление: его следует считать отсутствием нового `GptTmpl.inf`.

9.3 Строки и principal

`infQuote(s)`:

1. отклоняет NUL, CR и LF;
2. заменяет `%` на `%%`;
3. заменяет `"` на `""`;
4. заключает результат в двойные кавычки.

Удвоение соответствует общим правилам INF Microsoft. Поддержка Unicode и удвоенных кавычек является частью wire-профиля SDMX 1.0 и требует совместимого INF-обработчика.

`infPrincipal(p)` возвращает `*` плюс каноническую SID-строку, если `p` — SID; имя возвращается без префикса. Вход `*SID` запрещён, чтобы исключить двойное кодирование. Формат определён в MS-GPSB §2.2.6.

9.4 Примеры wire-записей

Логическое `true`, отображённое в integer `1`:

```
[System Access]
PasswordComplexity = 1
```

Строковая запись:

```
[System Access]
NewAdministratorName = "Local Admin"
```

Список principal:

```
[Privilege Rights]
SeServiceLogonRight = *S-1-5-32-544,*S-1-5-32-545
```

REG_DWORD с двумя выбранными флагами (`524288 OR 536870912`):

```
[Registry Values]
MACHINE\System\CurrentControlSet\Control\Lsa\MSV1_0\NTLMMinClientSec=4,537395200
```

REG_BINARY с одним байтом:

```
[Registry Values]
MACHINE\System\CurrentControlSet\Control\Lsa\FullPrivilegeAuditing=3,1
```

REG_MULTI_SZ и пустой список:

```
[Registry Values]
MACHINE\System\CurrentControlSet\Services\LanManServer\Parameters\NullSessionPipes=7,"srvsvc","browser"
MACHINE\System\CurrentControlSet\Services\LanManServer\Parameters\NullSessionPipes=7,
```

Group Membership с присутствующим пустым `member_of`:

```
[Group Membership]
*S-1-5-32-545__Members = *S-1-5-32-544,*S-1-5-32-546
*S-1-5-32-545__Memberof =
```

Служба и защищаемый объект:

```
[Service General Setting]
"Spooler",4,"D:(A;;CCLCSWLOCRRC;;;SY)(A;;CCLCSWLOCRRC;;;BA)"

[Registry Keys]
"MACHINE\Software\Contoso",2,"D:PAI(A;OICI;KA;;;SY)(A;OICI;KA;;;BA)"
```

9.5 Особенности целевых параметров

Числа в INF являются значениями полей шаблона безопасности, а не обязательно готовыми значениями внутренних структур Windows. Например, `MaxTicketAge` задан в часах, `MaxServiceAge` — в минутах; множитель в OCL только приводит их к одной единице для сравнения. `AuditLogRetentionPeriod = 2` остаётся числом 2, хотя обработчик задаёт `Retention = 0xFFFFFFFF`; см. MS-GPSB §3.2.5.5.

Для `MaxServiceAge` допустимы `0` либо `10..99999` минут. Ноль означает неограниченный срок действия и записывается как `MaxServiceAge = 0`, без подстановки другого числа. Это специальное значение описано в документации политики Windows. В таблице MS-GPSB §2.2.2 указан минимум 10 без этого исключения; поддержка нуля является явно заданным расширением wire-профиля SDMX 1.0 относительно этой таблицы и должна быть проверена у целевого обработчика.

Некоторые параметры хранят числа как REG_SZ. `AllocateDASD` и `ScRemoveOption` используют строковые литералы, а `CachedLogonsCount` — `storeAsText="true"`. Тип логического элемента не определяет тип реестра.

Пустая SDDL-строка НЕ ДОЛЖНА автоматически означать удаление значения или пустую DACL. Неопределённость представлена отсутствием создаваемой записи; отдельной операции удаления формат 1.0 не содержит.

`__Members` задаёт состав управляемой группы, а `__Memberof` обеспечивает её включение в перечисленные группы и не удаляет из остальных. Поэтому пустая `__Memberof` не очищает все связи членства. Семантика описана в KB 279301.

10 Безопасность реализации

XML-парсер ДОЛЖЕН запрещать DTD, внешние сущности, XInclude и сетевую загрузку схем. Проверка выполняется только по локальным доверенным XSD. Реализация ДОЛЖНА иметь конечные пределы размера файла, глубины XML, числа политик, ресурсов и записей коллекций, длины строк и итогового INF. Превышение предела является ошибкой, а не поводом усечь значение.

Basename и тег локали используются только как компоненты пути внутри корня комплекта. `..`, NUL и разделители пути запрещены. После разрешения символических ссылок открываемый файл ДОЛЖЕН оставаться внутри разрешённого корня; допустим полный запрет символических ссылок.

INF сначала полностью строится и проверяется в памяти или временном файле. Запись на диск СЛЕДУЕТ выполнять атомарной заменой с ограниченными правами. Ошибка НЕ ДОЛЖНА оставлять частично обновлённый `GptTmpl.inf`.

SDDL, principal, путь, раздел реестра и имя службы после синтаксической проверки остаются административными данными, способными изменить безопасность системы. Компонент предварительного просмотра не должен автоматически получать привилегии компонента применения.

11 Ссылки

11.1 Нормативные источники

- MS-GPREG §7.1 — Base ADMX Schema.
- MS-GPREG §7.3 — ADMX File Schema.
- MS-GPREG §2.2.2.2.3 — ADMX/ADML File Reference Examples.
- MS-GPSB — Group Policy: Security Protocol Extension.
- MS-GPSB §2.2 — Message Syntax.
- MS-GPSB §2.2.1 — System Access.
- MS-GPSB §2.2.2 — Kerberos Policy.
- MS-GPSB §2.2.3 — Event Log Policies.
- MS-GPSB §2.2.4 — Event Audit Policies.
- MS-GPSB §2.2.5 — Registry Values.
- MS-GPSB §2.2.6 — Privilege Rights.
- MS-GPSB §2.2.7 — Registry Keys.
- MS-GPSB §2.2.8 — Service General Setting.
- MS-GPSB §2.2.9 — File Security.
- MS-GPSB §2.2.10 — Group Membership.
- MS-DTYP §2.4.2.1 — SID String Format Syntax.
- MS-DTYP §2.5.1.1 — SDDL Syntax.
- OMG Object Constraint Language 2.4.
- W3C — XML 1.0.
- W3C — XML Schema Part 1: Structures.
- W3C — XML Schema Part 2: Datatypes.
- RFC 2119.

11.2 Справочные источники

- MS-GPOL — Group Policy: Core Protocol.
- Microsoft Learn — General Syntax Rules for INF Files.
- Microsoft — Cached domain logon information.

12 Приложение A. Схема SDMX 1.0

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
           xmlns:sdmx="http://www.altlinux.org/GroupPolicy/SecurityDefinitions"
           xmlns:pd="http://www.microsoft.com/GroupPolicy/PolicyDefinitions"
           targetNamespace="http://www.altlinux.org/GroupPolicy/SecurityDefinitions"
           elementFormDefault="qualified">

  <xs:import namespace="http://www.microsoft.com/GroupPolicy/PolicyDefinitions"
             schemaLocation="microsoft/PolicyNamespaceTypes.xsd"/>

  <xs:simpleType name="identifierType">
    <xs:restriction base="xs:token">
      <xs:pattern value="[a-z][a-z0-9_.-]*"/>
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="nonBlankStringType">
    <xs:restriction base="xs:string">
      <xs:minLength value="1"/>
      <xs:pattern value="[\s\S]*\S[\s\S]*"/>
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="itemReferenceType">
    <xs:restriction base="xs:token">
      <xs:pattern value="((\p{L}|\p{N}|_)+:)?[a-z][a-z0-9_.-]*"/>
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="cultureNameType">
    <xs:restriction base="xs:token">
      <xs:pattern value="[A-Za-z]{2,3}(-[A-Za-z0-9]{2,8})*"/>
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="registryHiveType">
    <xs:restriction base="xs:token">
      <xs:enumeration value="HKEY_LOCAL_MACHINE"/>
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="registryValueType">
    <xs:restriction base="xs:token">
      <xs:enumeration value="REG_DWORD"/>
      <xs:enumeration value="REG_SZ"/>
      <xs:enumeration value="REG_EXPAND_SZ"/>
      <xs:enumeration value="REG_BINARY"/>
      <xs:enumeration value="REG_MULTI_SZ"/>
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="sectionValueType">
    <xs:restriction base="xs:token">
      <xs:enumeration value="integer"/>
      <xs:enumeration value="text"/>
      <xs:enumeration value="principalList"/>
    </xs:restriction>
  </xs:simpleType>

  <xs:complexType name="literalType">
    <xs:choice>
      <xs:element name="integer">
        <xs:complexType>
          <xs:attribute name="value" type="xs:long" use="required"/>
        </xs:complexType>
```

```
      </xs:element>
      <xs:element name="string" type="xs:string"/>
      <xs:element name="binary" type="xs:hexBinary"/>
    </xs:choice>
  </xs:complexType>

  <xs:complexType name="rangeType">
    <xs:attribute name="min" type="xs:long" use="required"/>
    <xs:attribute name="max" type="xs:long" use="required"/>
  </xs:complexType>

  <xs:complexType name="substitutionType">
    <xs:attribute name="input" type="xs:long" use="required"/>
    <xs:attribute name="output" type="xs:long" use="required"/>
  </xs:complexType>

  <xs:complexType name="inputDomainType">
    <xs:sequence>
      <xs:element name="range" type="sdmx:rangeType" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="sectionValueDestinationType">
    <xs:attribute name="section" type="sdmx:nonBlankStringType" use="required"/>
    <xs:attribute name="key" type="sdmx:nonBlankStringType" use="required"/>
    <xs:attribute name="type" type="sdmx:sectionValueType" use="required"/>
  </xs:complexType>

  <xs:complexType name="registryValueDestinationType">
    <xs:attribute name="hive" type="sdmx:registryHiveType" use="required"/>
    <xs:attribute name="key" type="pd:registryKey" use="required"/>
    <xs:attribute name="valueName" type="pd:registryValueName" use="required"/>
    <xs:attribute name="type" type="sdmx:registryValueType" use="required"/>
    <xs:attribute name="storeAsText" type="xs:boolean" default="false"/>
  </xs:complexType>

  <xs:group name="scalarDestinationGroup">
    <xs:choice>
      <xs:element name="sectionValue" type="sdmx:sectionValueDestinationType"/>
      <xs:element name="registryValue" type="sdmx:registryValueDestinationType"/>
    </xs:choice>
  </xs:group>

  <xs:complexType name="optionType">
    <xs:sequence>
      <xs:element name="value" type="sdmx:literalType" minOccurs="0"/>
    </xs:sequence>
    <xs:attribute name="id" type="sdmx:identifierType" use="required"/>
    <xs:attribute name="displayName" type="pd:stringReference"/>
  </xs:complexType>

  <xs:complexType name="booleanElementType">
    <xs:sequence>
      <xs:element name="trueValue" type="sdmx:literalType" minOccurs="0"/>
      <xs:element name="falseValue" type="sdmx:literalType" minOccurs="0"/>
      <xs:group ref="sdmx:scalarDestinationGroup" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="id" type="sdmx:identifierType"/>
    <xs:attribute name="initialValue" type="xs:boolean"/>
  </xs:complexType>

  <xs:complexType name="integerElementType">
    <xs:sequence>
      <xs:element name="range" type="sdmx:rangeType" minOccurs="0" maxOccurs="unbounded"/>
                <xs:element   name="substitute"   type="sdmx:substitutionType"   minOccurs="0"
maxOccurs="unbounded"/>
      <xs:element name="inputDomain" type="sdmx:inputDomainType" minOccurs="0"/>
      <xs:group ref="sdmx:scalarDestinationGroup" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="id" type="sdmx:identifierType"/>
```

```
    <xs:attribute name="initialValue" type="xs:long"/>
  </xs:complexType>

  <xs:complexType name="enumElementType">
    <xs:sequence>
      <xs:element name="option" type="sdmx:optionType" maxOccurs="unbounded"/>
      <xs:group ref="sdmx:scalarDestinationGroup" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="id" type="sdmx:identifierType"/>
    <xs:attribute name="initialOption" type="sdmx:identifierType"/>
  </xs:complexType>

  <xs:complexType name="initialSelectionType">
    <xs:attribute name="option" type="sdmx:identifierType" use="required"/>
  </xs:complexType>

  <xs:complexType name="flagsElementType">
    <xs:sequence>
      <xs:element name="option" type="sdmx:optionType" maxOccurs="unbounded"/>
          <xs:element  name="initialSelection"  type="sdmx:initialSelectionType"  minOccurs="0"
maxOccurs="unbounded"/>
      <xs:group ref="sdmx:scalarDestinationGroup" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="id" type="sdmx:identifierType"/>
  </xs:complexType>

  <xs:complexType name="stringElementType">
    <xs:sequence>
      <xs:group ref="sdmx:scalarDestinationGroup" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="id" type="sdmx:identifierType"/>
    <xs:attribute name="initialValue" type="xs:string"/>
  </xs:complexType>

  <xs:complexType name="sddlElementType">
    <xs:sequence>
      <xs:group ref="sdmx:scalarDestinationGroup" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="id" type="sdmx:identifierType"/>
  </xs:complexType>

  <xs:simpleType name="principalKindType">
    <xs:restriction base="xs:token">
      <xs:enumeration value="any"/>
      <xs:enumeration value="group"/>
    </xs:restriction>
  </xs:simpleType>

  <xs:complexType name="listElementType">
    <xs:sequence>
      <xs:group ref="sdmx:scalarDestinationGroup" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="id" type="sdmx:identifierType"/>
  </xs:complexType>

  <xs:complexType name="principalElementType">
    <xs:sequence>
      <xs:group ref="sdmx:scalarDestinationGroup" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="id" type="sdmx:identifierType"/>
    <xs:attribute name="allowedKind" type="sdmx:principalKindType"/>
  </xs:complexType>

  <xs:complexType name="principalListElementType">
    <xs:sequence>
      <xs:group ref="sdmx:scalarDestinationGroup" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="id" type="sdmx:identifierType"/>
    <xs:attribute name="allowedKind" type="sdmx:principalKindType"/>
  </xs:complexType>
```

```
<xs:complexType name="semanticTextElementType">
  <xs:sequence>
    <xs:group ref="sdmx:scalarDestinationGroup" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="id" type="sdmx:identifierType"/>
</xs:complexType>

<xs:complexType name="fieldType">
  <xs:sequence>
    <xs:group ref="sdmx:dataElementGroup"/>
  </xs:sequence>
  <xs:attribute name="id" type="sdmx:identifierType" use="required"/>
  <xs:attribute name="displayName" type="pd:stringReference"/>
  <xs:attribute name="required" type="xs:boolean" default="true"/>
</xs:complexType>

<xs:complexType name="serviceRecordType">
  <xs:attribute name="serviceField" type="sdmx:identifierType" use="required"/>
  <xs:attribute name="startupField" type="sdmx:identifierType" use="required"/>
  <xs:attribute name="descriptorField" type="sdmx:identifierType" use="required"/>
</xs:complexType>

<xs:complexType name="objectRecordType">
  <xs:attribute name="target" use="required">
    <xs:simpleType>
      <xs:restriction base="xs:token">
        <xs:enumeration value="registry"/>
        <xs:enumeration value="file"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:attribute>
  <xs:attribute name="pathField" type="sdmx:identifierType" use="required"/>
  <xs:attribute name="propagationField" type="sdmx:identifierType" use="required"/>
  <xs:attribute name="descriptorField" type="sdmx:identifierType" use="required"/>
</xs:complexType>

<xs:complexType name="conditionalListRecordType">
  <xs:attribute name="listField" type="sdmx:identifierType" use="required"/>
  <xs:attribute name="emit" use="required" fixed="when-present">
    <xs:simpleType>
      <xs:restriction base="xs:token">
        <xs:enumeration value="when-present"/>
      </xs:restriction>
    </xs:simpleType>
  </xs:attribute>
</xs:complexType>

<xs:complexType name="groupRecordsType">
  <xs:sequence>
    <xs:element name="members" type="sdmx:conditionalListRecordType"/>
    <xs:element name="memberOf" type="sdmx:conditionalListRecordType"/>
  </xs:sequence>
  <xs:attribute name="groupField" type="sdmx:identifierType" use="required"/>
</xs:complexType>

<xs:group name="collectionDestinationGroup">
  <xs:choice>
    <xs:element name="serviceRecord" type="sdmx:serviceRecordType"/>
    <xs:element name="objectRecord" type="sdmx:objectRecordType"/>
    <xs:element name="groupRecords" type="sdmx:groupRecordsType"/>
  </xs:choice>
</xs:group>

<xs:complexType name="collectionElementType">
  <xs:sequence>
    <xs:element name="field" type="sdmx:fieldType" maxOccurs="unbounded"/>
    <xs:group ref="sdmx:collectionDestinationGroup" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:attribute name="id" type="sdmx:identifierType"/>
  <xs:attribute name="uniqueBy" type="sdmx:identifierType"/>
```

```
  <xs:attribute name="duplicateResource" type="pd:stringReference"/>
</xs:complexType>

<xs:group name="dataElementGroup">
  <xs:choice>
    <xs:element name="boolean" type="sdmx:booleanElementType"/>
    <xs:element name="integer" type="sdmx:integerElementType"/>
    <xs:element name="enum" type="sdmx:enumElementType">
      <xs:unique name="uniqueEnumOptionId">
        <xs:selector xpath="sdmx:option"/>
        <xs:field xpath="@id"/>
      </xs:unique>
    </xs:element>
    <xs:element name="flags" type="sdmx:flagsElementType">
      <xs:unique name="uniqueFlagOptionId">
        <xs:selector xpath="sdmx:option"/>
        <xs:field xpath="@id"/>
      </xs:unique>
    </xs:element>
    <xs:element name="string" type="sdmx:stringElementType"/>
    <xs:element name="sddl" type="sdmx:sddlElementType"/>
    <xs:element name="stringList" type="sdmx:listElementType"/>
    <xs:element name="principal" type="sdmx:principalElementType"/>
    <xs:element name="principalList" type="sdmx:principalListElementType"/>
    <xs:element name="serviceName" type="sdmx:semanticTextElementType"/>
    <xs:element name="registryKey" type="sdmx:semanticTextElementType"/>
    <xs:element name="filePath" type="sdmx:semanticTextElementType"/>
    <xs:element name="collection" type="sdmx:collectionElementType">
      <xs:unique name="uniqueCollectionFieldId">
        <xs:selector xpath="sdmx:field"/>
        <xs:field xpath="@id"/>
      </xs:unique>
    </xs:element>
  </xs:choice>
</xs:group>

<xs:complexType name="elementsType">
  <xs:sequence>
    <xs:group ref="sdmx:dataElementGroup" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>

<xs:complexType name="constraintType">
  <xs:simpleContent>
    <xs:extension base="sdmx:nonBlankStringType">
      <xs:attribute name="id" type="sdmx:identifierType" use="required"/>
      <xs:attribute name="language" use="required" fixed="OCL">
        <xs:simpleType>
          <xs:restriction base="xs:token">
            <xs:enumeration value="OCL"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:attribute>
      <xs:attribute name="profile" use="required" fixed="sdmx-1.0">
        <xs:simpleType>
          <xs:restriction base="xs:token">
            <xs:enumeration value="sdmx-1.0"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:attribute>
      <xs:attribute name="message" type="pd:stringReference" use="required"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>

<xs:complexType name="constraintsType">
  <xs:sequence>
    <xs:element name="constraint" type="sdmx:constraintType" maxOccurs="unbounded"/>
  </xs:sequence>
```

```
  </xs:complexType>

  <xs:complexType name="noticeType">
    <xs:attribute name="message" type="pd:stringReference" use="required"/>
  </xs:complexType>

  <xs:complexType name="noticesType">
    <xs:sequence>
      <xs:element name="notice" type="sdmx:noticeType" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="parentCategoryType">
    <xs:attribute name="ref" type="sdmx:itemReferenceType" use="required"/>
  </xs:complexType>

  <xs:complexType name="securityPolicyType">
    <xs:sequence>
      <xs:element name="parentCategory" type="sdmx:parentCategoryType"/>
      <xs:element name="elements" type="sdmx:elementsType"/>
      <xs:element name="constraints" type="sdmx:constraintsType" minOccurs="0"/>
      <xs:element name="notices" type="sdmx:noticesType" minOccurs="0"/>
    </xs:sequence>
    <xs:attribute name="id" type="sdmx:identifierType" use="required"/>
    <xs:attribute name="displayName" type="pd:stringReference" use="required"/>
    <xs:attribute name="explainText" type="pd:stringReference" use="required"/>
    <xs:attribute name="presentation" type="pd:presentationReference" use="required"/>
    <xs:attribute name="allowedValues" type="pd:stringReference"/>
    <xs:attribute name="defaultText" type="pd:stringReference"/>
  </xs:complexType>

  <xs:complexType name="categoryType">
    <xs:sequence>
      <xs:element name="parentCategory" type="sdmx:parentCategoryType" minOccurs="0"/>
    </xs:sequence>
    <xs:attribute name="id" type="sdmx:identifierType" use="required"/>
    <xs:attribute name="displayName" type="pd:stringReference" use="required"/>
  </xs:complexType>

  <xs:complexType name="securityNamespacesType">
    <xs:sequence>
      <xs:element name="target" type="pd:PolicyNamespaceAssociation"/>
               <xs:element   name="using"   type="pd:PolicyNamespaceAssociation"   minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="documentResourcesType">
    <xs:attribute name="minRequiredRevision" type="pd:versionString" use="required"/>
    <xs:attribute name="fallbackCulture" type="sdmx:cultureNameType" use="required"/>
  </xs:complexType>

  <xs:complexType name="categoriesType">
    <xs:sequence>
      <xs:element name="category" type="sdmx:categoryType" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="securityPoliciesType">
    <xs:sequence>
      <xs:element name="securityPolicy" type="sdmx:securityPolicyType" maxOccurs="unbounded">
        <xs:unique name="uniqueSecurityPolicyElementId">
          <xs:selector xpath="sdmx:elements/*"/>
          <xs:field xpath="@id"/>
        </xs:unique>
        <xs:unique name="uniqueConstraintId">
          <xs:selector xpath="sdmx:constraints/sdmx:constraint"/>
          <xs:field xpath="@id"/>
        </xs:unique>
      </xs:element>
```

```
    </xs:sequence>
  </xs:complexType>

  <xs:element name="securityDefinitions">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="securityNamespaces" type="sdmx:securityNamespacesType"/>
        <xs:element name="resources" type="sdmx:documentResourcesType"/>
        <xs:element name="categories" type="sdmx:categoriesType"/>
        <xs:element name="securityPolicies" type="sdmx:securityPoliciesType"/>
      </xs:sequence>
      <xs:attribute name="schemaVersion" type="pd:versionString" use="required" fixed="1.0"/>
      <xs:attribute name="revision" type="pd:versionString" use="required"/>
      <xs:attribute name="format" use="required" fixed="securityTemplate">
        <xs:simpleType>
          <xs:restriction base="xs:token">
            <xs:enumeration value="securityTemplate"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:attribute>
    </xs:complexType>
    <xs:unique name="uniqueTargetNamespaceAlias">
      <xs:selector xpath="sdmx:securityNamespaces/*"/>
      <xs:field xpath="@prefix"/>
    </xs:unique>
    <xs:unique name="uniqueCategoryId">
      <xs:selector xpath="sdmx:categories/sdmx:category"/>
      <xs:field xpath="@id"/>
    </xs:unique>
    <xs:unique name="uniqueSecurityPolicyId">
      <xs:selector xpath="sdmx:securityPolicies/sdmx:securityPolicy"/>
      <xs:field xpath="@id"/>
    </xs:unique>
  </xs:element>

</xs:schema>
```

13 Приложение B. Схема SDML 1.0

```
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
           xmlns:sdml="http://www.altlinux.org/GroupPolicy/SecurityDefinitions"
           xmlns:pd="http://www.microsoft.com/GroupPolicy/PolicyDefinitions"
           targetNamespace="http://www.altlinux.org/GroupPolicy/SecurityDefinitions"
           elementFormDefault="qualified">

  <xs:import namespace="http://www.microsoft.com/GroupPolicy/PolicyDefinitions"
             schemaLocation="microsoft/BaseTypes.xsd"/>

  <xs:simpleType name="bindingReferenceType">
    <xs:restriction base="xs:token">
      <xs:pattern value="[a-z][a-z0-9_.-]*"/>
    </xs:restriction>
  </xs:simpleType>

  <xs:complexType name="labelWhenType">
    <xs:attribute name="value" type="xs:long" use="required"/>
    <xs:attribute name="label" type="pd:stringReference" use="required"/>
  </xs:complexType>

  <xs:attributeGroup name="dataControlAttributes">
    <xs:attribute name="refId" type="sdml:bindingReferenceType" use="required"/>
    <xs:attribute name="label" type="pd:stringReference"/>
    <xs:attribute name="readOnly" type="xs:boolean" default="false"/>
  </xs:attributeGroup>

  <xs:complexType name="controlReferenceType">
    <xs:attributeGroup ref="sdml:dataControlAttributes"/>
  </xs:complexType>

  <xs:complexType name="integerTextBoxType">
    <xs:sequence>
     <xs:element name="labelWhen" type="sdml:labelWhenType" minOccurs="0" maxOccurs="unbounded"/
>
    </xs:sequence>
    <xs:attributeGroup ref="sdml:dataControlAttributes"/>
    <xs:attribute name="suffix" type="pd:stringReference"/>
    <xs:attribute name="showRange" type="xs:boolean" default="true"/>
  </xs:complexType>

  <xs:complexType name="booleanChoiceType">
    <xs:attributeGroup ref="sdml:dataControlAttributes"/>
    <xs:attribute name="falseLabel" type="pd:stringReference" use="required"/>
    <xs:attribute name="trueLabel" type="pd:stringReference" use="required"/>
  </xs:complexType>

  <xs:complexType name="textControlType">
    <xs:attribute name="ref" type="pd:stringReference" use="required"/>
  </xs:complexType>

  <xs:complexType name="collectionEditorType">
    <xs:sequence>
      <xs:group ref="sdml:controlGroup" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attributeGroup ref="sdml:dataControlAttributes"/>
  </xs:complexType>

  <xs:complexType name="groupType">
    <xs:sequence>
      <xs:group ref="sdml:controlGroup" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="label" type="pd:stringReference"/>
  </xs:complexType>

  <xs:group name="controlGroup">
```

```
    <xs:choice>
      <xs:element name="integerTextBox" type="sdml:integerTextBoxType">
        <xs:unique name="uniqueConditionalLabelValue">
          <xs:selector xpath="sdml:labelWhen"/>
          <xs:field xpath="@value"/>
        </xs:unique>
      </xs:element>
      <xs:element name="booleanChoice" type="sdml:booleanChoiceType"/>
      <xs:element name="dropdownList" type="sdml:controlReferenceType"/>
      <xs:element name="flagList" type="sdml:controlReferenceType"/>
      <xs:element name="textBox" type="sdml:controlReferenceType"/>
      <xs:element name="sddlEditor" type="sdml:controlReferenceType"/>
      <xs:element name="listBox" type="sdml:controlReferenceType"/>
      <xs:element name="principalPicker" type="sdml:controlReferenceType"/>
      <xs:element name="principalList" type="sdml:controlReferenceType"/>
      <xs:element name="servicePicker" type="sdml:controlReferenceType"/>
      <xs:element name="registryKeyPicker" type="sdml:controlReferenceType"/>
      <xs:element name="filePicker" type="sdml:controlReferenceType"/>
      <xs:element name="collectionEditor" type="sdml:collectionEditorType"/>
      <xs:element name="group" type="sdml:groupType"/>
      <xs:element name="text" type="sdml:textControlType"/>
    </xs:choice>
  </xs:group>

  <xs:complexType name="presentationType">
    <xs:sequence>
      <xs:group ref="sdml:controlGroup" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:attribute name="id" type="pd:resourceID" use="required"/>
  </xs:complexType>

  <xs:complexType name="stringTableType">
    <xs:sequence>
      <xs:element name="string" type="pd:LocalizedString" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="presentationTableType">
    <xs:sequence>
      <xs:element name="presentation" type="sdml:presentationType" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>

  <xs:complexType name="localizedResourcesType">
    <xs:sequence>
      <xs:element name="stringTable" type="sdml:stringTableType"/>
      <xs:element name="presentationTable" type="sdml:presentationTableType"/>
    </xs:sequence>
  </xs:complexType>

  <xs:element name="securityDefinitionResources">
    <xs:complexType>
      <xs:sequence>
        <xs:element name="resources" type="sdml:localizedResourcesType"/>
      </xs:sequence>
      <xs:attribute name="schemaVersion" type="pd:versionString" use="required" fixed="1.0"/>
      <xs:attribute name="revision" type="pd:versionString" use="required"/>
    </xs:complexType>
    <xs:unique name="uniqueResourceId">
      <xs:selector xpath="sdml:resources/sdml:stringTable/sdml:string"/>
      <xs:field xpath="@id"/>
    </xs:unique>
    <xs:unique name="uniquePresentationId">
      <xs:selector xpath="sdml:resources/sdml:presentationTable/sdml:presentation"/>
      <xs:field xpath="@id"/>
    </xs:unique>
  </xs:element>

</xs:schema>
```