TTC 標準
TTC STANDARD

JT-X725
ネットワーク管理についての一般関係モデル

OSI Management
- Structure of Management Information -
General relationship model

第1版

1998年4月28日制定

社団法人
情報通信技術委員会

THE TELECOMMUNICATION TECHNOLOGY COMMITTEE

Telecommunication
Technology
Committee

ＪＴ－Ｘ７２５　ネットワーク管理についての一般関係モデル

＜参考＞

１．英文の記述レベル

適用レベル：Ｅ１

本標準の本文，付属資料および付録の文章および図に英文記述を含んでいる．

２．国際勧告等との関連

本標準は、１９９５年版　ＩＴＵ－Ｔ勧告　Ｘ．７２５：Ｇｅｎｅｒａｌ　Ｒｅｌａｔｉｏｎｓｈｉｐ Ｍｏｄｅｌに基づいて定めたものである。

３．上記国際勧告等に対する追加項目等

なし。

４．改版の履歴

版数	制定日	改版内容
第１版	平成10年4月28日	制定

５．工業所有権

本標準に関わる「工業所有権の実施の権利に係る確認書」の提出状況は、ＴＴＣホームページでご覧になれます。

６．その他

（１）　参照している勧告、標準等

ＴＴＣ標準：ＪＴ－Ｘ７００、ＪＴ－Ｘ７１０、ＪＴ－Ｘ７２０，ＪＴ－Ｘ７２１、ＪＴ－Ｘ７２２、
ＪＴ－Ｘ７２３、ＪＴ－Ｘ７２４

ＩＴＵ－Ｔ勧告：Ｘ．７００、Ｘ．７０１、Ｘ．７１１、Ｘ．７３１、Ｘ．７３２、Ｘ．２０８、
Ｘ．２９１、Ｘ．２９６

目次

１.適用範囲　（Scope）

本勧告｜国際標準は、以下を提供する。

a)　リソース間の関係の検討、表現、管理、及び、再利用可能な仕様の開発を行うためのモデル

b)　関係、関係の表現、及び、管理を、明確に記述するための記述ツール

c)　関係の表現、及び、管理に使用することができる汎用管理情報の定義

d)　適合性宣言プロフォーマの開発のためのガイドライン

e)　定義の例

一般関係モデルを７節に定義する。記述ツールを付属資料 A に定義する。汎用管理情報を８節と付属資料 B に定義する。適合性宣言プロフォーマの開発のためのガイドラインを付属資料 C、及び D に示す。表現方法の図解と定義の例を付属資料 E、及び、F に示す。本テキストに対するコメントを付属資料 G に示す。

本勧告｜国際標準は、関係付けにより暗黙的に要求されるリソース間の一貫性維持を実現する機構を提供するものではない。

CCITT Rec.X.732|ISO/IEC 10164-3 は、属性を用いて表現される関係のモデル、及び、特定の関係の型を表現するために用いる汎用属性の集合を定義している。本勧告｜国際標準で定義するモデル化の概念及び定義ツールは、一般の関係の定義に適用可能であり、それゆえ、 CCITT Rec. X.732 | ISO/IEC 10164-3 でモデル化されている、属性を用いて表現する関係に対しても適用可能である。

２.規範参照文献　（Normative reference）

以下に示す勧告と国際標準は、本文書から参照しており、本勧告｜国際標準の規定を構成する規定を含んでいる。本文書の出版の時点では、示している版数は正しい。全ての勧告｜国際標準は改版を免れ得ないものであるため、本勧告｜国際標準に同意する団体に対しては、以下に示す勧告及び国際標準の最新版の適用可能性を調査することを奨励する。IEC と ISO の参加メンバが、現在有効な国際標準の登録票を維持管理している。ITU-T の事務局が、現在有効な ITU-T 勧告の一覧を維持管理している。

2.1　ITU-T 勧告｜国際標準（共通テキスト）

（Identical Recommendations | International Standards）

－　CCITT Recommendation X.701 (1992) | ISO/IEC 10040: 1992, *Information technology - Open Systems Interconnection - System management overview.*

－　CCITT Recommendation X.720 (1992) | ISO/IEC 10165-1: 1993, *Information technology - Open Systems Interconnection - Structure of management information: Management Information Model.*

－　CCITT Recommendation X.721 (1992) | ISO/IEC 10165-2: 1992, *Information technology - Open Systems Interconnection - Structure of management information: Definition of management information.*

－　CCITT Recommendation X.722 (1992) | ISO/IEC 10165-4: 1992, *Information technology - Open Systems Interconnection - Structure of management information: Guidelines for definition of managed objects.*

－　CCITT Recommendation X.724 (1993) | ISO/IEC 10165-6: 1994, *Information technology - Open Systems Interconnection - Structure of management information: Requirements and guidelines for implementation conformance statement proformas associated with OSI management.*

－　CCITT Recommendation X.731 (1992) | ISO/IEC 10164-2: 1993, *Information technology - Open Systems Interconnection - System management: State management function.*

－　CCITT Recommendation X.732 (1992) | ISO/IEC 10164-3: 1993, *Information technology - Open Systems nterconnection - System management: Attributes for representing relationships.*

2.2　ITU-T 勧告｜国際標準（同一内容テキスト）

（Paired Recommendations | International Standards equivalent in technical content）

- CCITT Recommendation X.208 (1988), *Specification of Abstract Syntax Notation One (ASN.1).*

  ISO/IEC 8824: 1990, *Information technology - Open System Interconnection - Specification of Abstract Syntax Notation One (ASN.1).*
- CCITT Recommendation X.291 (1992), *OSI conformance testing methodology and framework for protocol Recommendations for CCITT applications - Abstract test suit specification.*

  ISO/IEC 9646-2: 1994, *Information technology - Open Systems Interconnection - Conformance testing methodology ad framework - Part 2: Abstract Test Suit specification.*
- ITU-T Recommendation X.296 (1995), *OSI conformance testing methodology and framework for protocol Recommendations for ITU-T applications - Implementation conformance statements.*

  ISO/IEC 9646-7: 1995, *Information technology - Open Systems Interconnection - Conformance testing methodology ad framework - Part 7: Implementation Conformance Statement.*
- CCITT Recommendation X.700 (1992), *Management framework for Open System Interconnection (OSI) for CCITT applications.*

  ISO/IEC 7498-4: 1989, *Information processing system - Open System Interconnection - Basic Reference Model - Part 4: Management framework.*
- CCITT Recommendation X.710 (1991), *Common management information service definition for CCITT applications.*

  ISO/IEC 9595: 1991, *Information technology - Open Systems Interconnection - Common management information service definition.*
- CCITT Recommendation X.711 (1991), *Common management information protocol for CCITT applications*

  ISO/IEC 9596: 1991, *Information technology - Open Systems Interconnection - Common management information protocol - Part 1: Specification.*

３.定義　（Definitions）

本勧告｜国際標準では、以下に示す定義を用いる。

3.1　「管理フレームワーク」の定義　（Management framework definitions）

本勧告｜国際標準は、CCITT Rec. X.700 | ISO/IEC 7498-4 で定義されている以下の用語を使用する。

- 管理オブジェクト（managed object）

3.2　「システム管理概要」の定義　（Systems management overview definitions）

本勧告｜国際標準は、CCITT Rec. X.701 | ISO/IEC 10040 で定義されている以下の用語を使用する。

a)　管理オブジェクトクラス（managed object class）

b)　マネジャ（manager）

c)　MOCS（MOCS）

d)　MOCS プロフォーマ（MOCS proforma）

e)　通知（notification）

f)　（システム管理）操作（(system management) operation）

3.3　「CMIS」の定義　（CMIS definitions）

本勧告｜国際標準は、 CCITT Rec. X.710 | ISO/IEC 9595 で定義されている以下の用語を使用する。

– 属性（attribute）

3.4 「管理情報モデル」の定義 （Management information model definitions）

本勧告｜国際標準は、 CCITT Rec. X.720 | ISO/IEC 10165-1 で定義されている以下の用語を使用する。

a) アクション（action）

b) 属性グループ（attribute group）

c) 属性タイプ（attribute type）

d) 振舞い（behaviour）

e) 特性（characteristic）

f) 包含（containment）

g) 継承（inheritance）

h) 不変式（invariant）

i) 多重継承（multiple inheritance）

j) ネームバインディング（name binding）

k) ネーミングツリー（naming tree）

l) パッケージ（packages）

m) パラメタ（parameter）

n) 結果条件（post-condition）

o) 前提条件（pre-condition）

p) 特化（specialization）

q) サブクラス（subclass）

r) 下位オブジェクト（subordinate object）

s) スーパクラス（superclass）

t) 上位オブジェクト（superior object）

3.5 「管理オブジェクト定義のガイドライン」の定義 （Guidelines for the definition of managed objects definitions）

本勧告｜国際標準は、 CCITT Rec. X.722 | ISO/IEC 10165-4 で定義されている以下の用語を使用する。

a) 管理オブジェクトクラス定義（naged object class definition）

b) テンプレート（template）

3.6 「OSI 管理に関する実装適合性宣言プロフォーマの要件およびガイドライン」の定義 （Requirement and guidelines for implementation conformance statement proformas associated with OSI management definitions）

本勧告｜国際標準は、 ITU-T Rec. X.724 | ISO/IEC 10165-5 で定義されている以下の用語を使用する。

a) 管理関係適合性宣言（MRCS）（Managed Relationship Conformance Summary (MRCS)）

b) MRCS プロフォーマ（MRCS proforma）

c) 管理情報定義宣言（MIDS）（Management Information Definition Statement (MIDS)）

d) MIDS プロフォーマ（MIDS proforma）

3.7 「状態管理機能」の定義 （State management function definitions）

本勧告｜国際標準は、 CCITT Rec. X.731 | ISO/IEC 10164-2 で定義されている以下の用語を使用する。

a) 運用状態（administrative state）

b) 操作状態（operational state）

c) 使用状態（usage state）

3.8　追加定義　(Additional definitions)

3.8.1　結合　(binding)：

管理オブジェクトと、管理関係内の指定されたロールとを関連づけること。

3.8.2　結合のサポート　(binding support)：

管理関係の存在期間中に、管理オブジェクトと指定されたロールとを結合できる管理関係の能力。

3.8.3　(サブクラスの）一貫性　(consistency (of a subclass))：

管理関係クラスの特化により派生したサブクラスのインスタンスが、マネジャシステムの機能に影響を与えずに、スーパクラスのインスタンスの代用となれること。

3.8.4　管理関係　(managed relationship)：

管理オブジェクトの特性に関連する不変式を伴う、管理オブジェクトの集合。

3.8.5　管理関係クラス　(managed relationship class)：

同一の定義を共有する管理関係の名前づけられた集合。

3.8.6　パーティシパント　(participant)：

管理関係のロールを担う管理オブジェクト。

3.8.7　パーティシパントポインタ　(participant pointer)：

管理関係の特定のロールとなっているパーティシパントを指定する属性。

3.8.8　関係カーディナリティ　(relationship cardinality)：

ある管理オブジェクトが同一のロールとして参加している、同一の管理関係クラスの関係インスタンスの数。

3.8.9　関係カーディナリティ制約　(relationship cardinality constraint)：

関係カーディナリティの制限の範囲となる値の集合。

3.8.10　関係管理通知　(relationship management notification)：

管理関係からの通知であり、１つ以上のシステム管理通知にマッピングされる。

3.8.11　関係管理操作　(relationship management operation)：

管理関係に対する操作であり、１つ以上のシステム管理操作にマッピングされる。

3.8.12　ロールカーディナリティ　(role cardinality)：

管理関係の指定されたロールに参加している管理オブジェクトの数。

3.8.13　ロールカーディナリティ制約　(role cardinality constraint)：

指定されたロールのロールカーディナリティの制限の範囲となる値の集合。

3.8.14　関係マッピング仕様　（relationship mapping specification）：

管理関係クラスの特性から、１つ以上の管理オブジェクトの特性へのマッピングを定義する、名前付けられた仕様。

3.8.15　関係クラス仕様　（relationship class specification）：

管理関係の特性を定義する、名前付けられた仕様。

3.8.16　ロール　（role）：

管理関係の特定の種類のパーティシパントに共通な性質。

3.8.17　解放　（unbinding）：

管理関係の指定されたロールから、管理オブジェクトを分離すること。

3.8.18　解放のサポート　（unbinding support）：

管理関係の存在期間中に、指定されたロールから管理オブジェクトを分離できる管理関係の能力。

４.略語　（Abbreviations）

本勧告｜国際標準では、以下に示す略語を用いる。

ASN.1	抽象構文記法１（Abstract Syntax Notation One） （CCITT Rec. X.208 | ISO/IEC 8824 を参照）
CMIS	共通管理情報サービス（Common Management Information Service） （CCITT Rec. X.710 | ISO/IEC 9595 を参照）
GDMO	管理オブジェクト定義のガイドライン （Guidelines for the Definition of Managed Objects） （CCITT Rec. X.722 | ISO/IEC 10165-4 を参照）
MIM	管理情報モデル（Management Information Model） （CCITT Rec. X.720 | ISO/IEC 10165-4 参照）
MRCS	管理関係適合性宣言（Managed Relationship Conformance Statement）
MOCS	管理オブジェクト適合性宣言（Managed Object Conformance Statement）
OSI	開放型システム間相互接続（Open System Interconnection）
SMI	管理情報構造（Structure of Management Information）
MIDS	管理情報定義宣言（Management Information Definition Statement）

５.記法　（Conventions）

本勧告｜国際標準を通して、**ASN.1** 記法、**GDMO** 記法、もしくは、付属資料 A で定義されている記法ツールを用いる箇所に、異なる字体とサイズの文字を使用する。

本勧告｜国際標準が含む記述ツールを、CCITT Rec. X.722 | ISO/IEC 10165-4 が定義している記法に従って定義する。

６.要件　（Requirements）

システム管理では、リソースが互いに影響を及ぼし合うという意味で、リソース間の関係が存在する可能性がある。管理者は、そのような関係を管理する能力、特に、以下を必要とする。

－　リソースの配置と関係の表現方法とに依存しない、リソース間の関係のモデル

－　関係を定義するための記述ツール
－　OSI システム管理の考え方に沿った、関係の表現方法と関係の管理のモデル
－　OSI システム管理の考え方に沿った、関係の表現方法と関係の管理を定義するための記述ツール
－　再利用可能な定義を開発するためのモデル

７.モデル　（Model）

システム管理を行う目的で、リソースを管理オブジェクトとしてモデル化している。そのため、リソース間の関係は、管理オブジェクト間の管理関係（**managed relationship**）としてモデル化する。管理関係は、管理オブジェクトに関連する不変式を伴なった管理オブジェクトの集合として定義する。不変式の例を以下に示す。

a)　供給者ー消費者関係の供給者を担う管理オブジェクトは、操作可能状態を維持しなくてはならない。
b)　バックアップ管理関係のバックアップ実施ロールを担う管理オブジェクトは、操作可能状態かつ、使用状態がアイドル状態かつ、運用状態がアンロックでなくてはならない。
c)　上位ー下位管理関係では、下位ロールを担う管理オブジェクトが少なくとも１つ存在する必要がある。

管理関係は、管理情報構造の追加的な情報モデル化の概念であるが、既存の管理情報モデルを用いた表現と管理が可能である（CCITT Rec. X.720 | ISO/IEC 10165-1 を参照）。同一の定義を共有する管理関係を、管理関係クラス（**managed relationship class**）としてグループ化する。管理関係クラスの定義に用いる表記法を、付属資料 A に示す。

本モデルでは、同一の管理関係クラスを、管理情報モデルの範囲内で、異なる方法で表現することを特に許容する。特定の表現は、関係マッピング（**managed relationship**）で定義する。関係マッピングの詳細な説明は、７.４節で行う。関係マッピングを定義する表記法を、付属資料 A に定義する。

7.1　管理関係　（Managed relationship）

管理関係は、表現に依存しない管理オブジェクト間の特性のモデル化を、ロール、振舞い、関係管理操作、関係管理通知、継承、関係を特徴付ける性質（qualifying properties）を用いて行う。また、ロールのモデル化を、適合管理オブジェクトクラス、ロールカーディナリティ、関係カーディナリティ、結合操作と解放操作のサポートを用いて行う。モデル化の概念を、以下の副節で詳細に示す。

7.1.1　関係管理操作及び通知　（Relationship management operations and notifications）

関係管理操作及び関係管理通知は、管理関係がサポートしている操作及び通知を、表現に依存しない操作及び通知としてモデル化する。以下に示すプロトタイプ操作及び通知を用いて、それらを表現する。

ESTABLISH	管理関係を確立する。
TERMINATE	管理関係を終了する。
BIND	管理オブジェクトを、管理関係のロールに結合する。
UNBIND	管理オブジェクトを、管理関係のロールから解放する。
QUERY	管理関係に関する情報を要求する。
NOTIFY	管理関係に関する事象を報告する。
USER DEFINED	ユーザ定義の操作。そのセマンティクスは、関連する管理関係の振舞いで、モデル化する。

これらのプロトタイプ操作及び通知のセマンティクスは、８節で定義する。

管理関係は、サポートされる実管理操作と通知、及び、８節で定義されるセマンティクスに対して追加するセマンティクスをモデル化する。

管理関係は、１つのプロトタイプに対して、複数の実関係管理操作及び通知をモデル化してもよい。例えば、ある管理関係は、あるパーティシパントに対する解放操作により、他の全てのパーティシパントに対して解放と消去を要求するような解放操作と、他のパーティシパントに影響を与えないような第２の解放操作をモデル化することができる。管理関係は、それぞれのプロトタイプに対する関係操作及び通知をモデル化する必要は無い。

7.1.2　管理関係の振舞い　(Managed relationship behaviour)

管理関係の振舞いは、ロールに対する不変式と、関係操作及び通知に対する不変式、前提条件、結果条件とを用いて、表現に依存しない振舞をモデル化する。

7.1.2.1　不変式　(invariant)：

ある範囲の間、真のままである論理的記述。範囲は、管理関係の生存期間や、関係管理操作の実行期間であってもよい。

7.1.2.2　前提条件　(pre-condition)　(関係管理操作、又は、関係管理通知の)：

関係管理操作の実行直前、及び関係管理通知の発出直前に、真でなくてはならない論理的記述。

7.1.2.3　結果条件　(post-condition)　(関係管理操作、又は、関係管理通知の)：

関係管理操作の実行直後、及び関係管理通知の発出直後に、真でなくてはならない論理的記述。

7.1.3　関係特性　(Relationship qualification)

関係特性は、管理関係全体に関連し、また、使用した表現方法によらずに実装可能な属性をモデル化する。例えば、電話の呼は、２つの加入者ロール管理オブジェクト間の管理関係としてモデル化できるが、呼の保留時間は、両加入者の特性というよりも、呼の特性とするべきである。しかしながら、特定の実装や、使用した表現方法に依存して、呼の保留時間属性は、加入者管理オブジェクト、もしくは、関係オブジェクトの双方にマッピングすることができる。

7.1.4　ロール　(Roles)

管理関係に結合されているそれぞれの管理オブジェクトは、パーティシパントであり、管理関係において一つ以上のロールを担う。ロールは、パーティシパントと管理関係に対して条件を課す。関係に参加する管理オブジェクトは、あるロールを担うためには、ある特性を保持しなくてはならない。管理関係は、ロールが課す条件に従わなくてはならない。

同一クラスの複数の管理オブジェクトは、同一の管理関係の異なるロールを担ってもよい。また、ある管理オブジェクトは、ある管理関係の１つ以上のロールを担ってもよい。また、ある管理オブジェクトは、１つ以上の管理関係のインスタンスに参加してもよい。

7.1.4.1　パーティシパントの特性 (Participant properties)

特定のロールを担うために管理オブジェクトが保持しなくてはならない特性は、適合管理オブジェクトクラス[1]を用い

[1] 適合性の概念は、ＭＩＭの５．２節で議論されている。

てモデル化される。一般的には、適合クラスはロールに固有な特性のみをモデル化する。特定の実装では、ロールを担う管理オブジェクトは、付加的な特性を保持してもよいが、少なくとも、適合クラスの特性を保持する必要がある。それゆえ、その管理オブジェクトは、適合クラスと同質異形（allomorphic）でなくてはならない。

7.1.4.2　ロールカーディナリティ（Role cardinality）

一般的には、指定されたロールとして、複数の管理オブジェクトが関係に参加してもよい。この管理オブジェクトの数を、そのロールのカーディナリティと呼ぶ。管理関係の実装は、２つの種類のロールカーディナリティの制約に従わなくてはならない。１つは、許容ロールカーディナリティであり、もう１つは、要求ロールカーディナリティである。それぞれの制約は、非負整数の値集合を用いて定義する。これは、、連続的な範囲の値集合となることが多い。

許容ロールカーディナリティ制約は、実装を許容するロールカーディナリティの範囲に関する制約であるのに対し、要求カーディナリティ制約は、実装を要求するロールカーディナリティの範囲に関する制約である。要求カーディナリティ制約の値集合は、許容カーディナリティ制約の値集合と等しいか、もしくは、その部分集合でなくてはならない。

7.1.4.3　結合と解放のサポート（Support of binding and unbinding）

関係管理は、関係の存在期間中の管理オブジェクトの動的な結合及び解放を、ロール単位でサポートできる。そのような管理関係は、関係管理操作 BIND 及び、UNBIND をそれぞれサポートする。

関係管理が BIND をサポートする場合、ロールカーディナリティ制約に違反しない限り、関係の存在期間中に、管理オブジェクトは関係に動的に参加することができる。制約に違反する場合は、結合要求は失敗する。

管理関係が UNBIND をサポートする場合、ロールカーディナリティ制約に違反しない限り、関係の存在期間中に、パーティシパントを関係から動的に解放することができる。制約に違反する場合は、解放要求は失敗する。

7.1.4.4　関係カーディナリティ（Relationship cardinality）

１つの管理オブジェクトは、同一の管理関係クラスの複数のインスタンスに対して、同一のロールで参加することができる。この管理関係のインスタンスの数を、そのロールの関係カーディナリティと呼ぶ。関係管理の実装は、関係カーディナリティに関する一つの制約　―許容関係カーディナリティ制約―　に従わなくてはならない。この制約は、非負整数の値集合を用いて定義される。これは、連続的な範囲の値集合となることが多い。許容関係カーディナリティ制約は、実装を許容する関係カーディナリティの範囲に関する制約である。

7.2　関係マッピング　（Relationship mappings）

関係マッピングは、一つ以上の管理オブジェクトの特性を用いて、以下のように管理関係の表現をモデル化する。

- 関係のロールと関係特性の、管理オブジェクトクラスへのマッピング
- 関係操作及び通知の、システム管理操作及び通知へのマッピング
- 関係オブジェクト
- パーティシパントポインタ

特定の管理関係クラスに関連する関係マッピングは、複数存在してもよい。

7.2.1　パーティシパントポインタ　（Participant pointers）

関係管理のパーティシパントと各々のロールの対応を、パーティシパントポインタ（participant pointer）属性を用いて

指定してもよい。パーティシパントポインタ属性の値が、関係に参加している管理オブジェクトを指定し、属性の型が、その関係オブジェクトが担っているロールを指定する。特定の関係管理もしくは関係マッピングに関連する全ての制約の下で行う、属性に基づく操作、又は、オブジェクトに基づく操作によるパーティシパントポインタ属性値の変更を、管理オブジェクトの参加状態を変更するために使用してもよい。パーティシパントポインタの定義は、付属資料 B の **participantPointer** 属性の定義から導出される。

7.2.2　関係管理操作及び通知　（Relationship management operations and notifications）

本勧告｜国際標準は、関係管理操作及び通知とシステム管理操作及び通知のマッピングを規定するものではない。しかしながら、付属資料 A はそのようなマッピングを定義するためのテンプレートを提供しており、また、付属資料 B は、管理関係の名前、関係クラス、実際に使用する関係マッピングを指定するための適当な属性を定義している。

関係管理操作及び通知の潜在的に可能なマッピングは、選択した表現方法により制約を受けることがある。すなわち、関係マッピングは、管理関係クラスの特定の表現方法に対するマッピングをモデル化する。特定の表現方法が課す制約を、７.４節の副節でさらに詳細に示す。単一の関係管理操作又は通知を、１つ以上のシステム管理操作及び通知にマッピングしてもよい。

関係管理操作及び通知のマッピングは、関係管理操作及び通知の前提条件、結果条件、不変式、及び、関係管理の不変式を、システム管理操作及び通知へ関連づけるように定義する。関係マッピングは、この条件を正確に満たす機構をモデル化する。例えば、上位－下位管理関係において、下位ロールの管理オブジェクトを少なくとも１つ要求する場合、関係マッピングは ESTABLISH のマッピングを以下のようにモデル化する。

- 上位及び下位ロールを担う管理オブジェクトの明示的な生成操作と、パーティシパントポインタを適切な値に設定する属性の操作、もしくは、
- 上位ロールを担う管理オブジェクトの単一の生成操作を実施し、その後は、被管理システムが下位ロールを担う管理オブジェクトの生成、及び、パーティシパントポインタの値の設定を行う。

7.2.3　振舞い　（Behaviour）

関係マッピングの振舞いは、関係管理とそれに関連する関係管理操作及び通知に対する表現に依存しない振舞いから、表現に依存する振舞いへのマッピングを、パーティシパントに関する不変式、及び、関連するパーティシパントのシステム管理操作及び通知に関する不変式、前提条件、結果条件を用いてモデル化する。関係マッピングは、表現方法に関連する全ての付加的振舞いもモデル化する。

7.3　仕様の再利用　（Re-usable specifications）

管理関係クラス、継承、特化は、再利用可能な仕様を開発するためのモデルを構成する。このモデルの本質は特化　－既存の管理関係クラスを継承し、更に定義を追加することによりクラスを導出すること－　である。

１つ以上の管理関係クラスから継承した特性と、管理関係クラステンプレートに定義した特性とを組み合わせることにより、関係管理クラスを特化することができる。特化されたクラスは、（複数の）オリジナルのクラスのサブクラスとなり、（複数の）オリジナルのクラスは、特化されたクラスの（複数の）スーパクラスとなる。付属資料 A に定義された特化の規則は、管理関係のサブクラスが（複数の）スーパクラスと一貫性を持つこと（consistent）を保証する。サブクラスがその（複数の）スーパクラスに対して持つ一貫性とは、管理関係のサブクラスのインスタンスが、マネジャシステムの機能に影響を与えずに、その複数のスーパクラスの中の一つのクラスのインスタンスの代用となれることである。

7.4　管理関係の表現及び管理

（Representation and management of managed relationships）

管理情報モデル　（CCITT Rec. X.720 | ISO/IEC 10165-1 参照）　で定義されている構文に基づく以下の方法を用いて、管理関係を表現できる。

- 名前付け
- パーティシパントポインタ
- 関係オブジェクト
- システム管理操作

管理関係の全ての範疇が、上記全ての表現方法により表現できるわけではない。単一の関係マッピング内で、これらの表現方法を組み合わせて使用してもよい。例えば、３つのロールを持つ管理関係の関係マッピングを、２つのロールを名前付けを使用して表現し、３つ目のロールをパーティシパントポインタを使用して表現してもよい。

7.4.1　名前付けによる表現及び管理　（Representation and management by means of naming）

関係マッピングは、名前付けを用いて、管理関係の表現及び管理をモデル化してもよい。１つの上位管理オブジェクトのネーミングスコープに含まれる複数の下位管理オブジェクトが名前付けられる。関係マッピングの中では、関係管理に関連するネームバインディングを差し示す。

関係管理操作は、上位管理オブジェクト、もしくは、下位管理オブジェクトに対するシステム管理操作として、マッピングできる。可能なマッピングを表１に列挙するが、ある特定の表現に対するマッピングは、関連する関係マッピングによりモデル化される。

指定されたネームバインディングの下位の識別名の構成要素を解析することにより、関係管理に参加しているパーティシパントを発見することができる。

表１－名前付け関係に対する操作のマッピング

プロトタイプ関係管理操作	上位管理オブジェクトに対するシステム管理操作の候補	下位管理オブジェクトに対するシステム管理操作の候補
BIND	Create、Action	Create
UNBIND	Delete、Action	Delete
QUERY	Action	Get＋名前解析、Action
ESTABLISH	Create、Action	Create、Action
TERMINATE	Delete、Action	Delete、Action

7.4.2　パーティシパントポインタによる表現及び管理

（Representation and management by means of participant pointers）

関係マッピングは、管理関係に参加しているパーティシパントが公開しているパーティシパントポインタ属性を用いて、管理関係の表現及び管理をモデル化してもよい。関係管理操作は、パーティシパントポインタに対する属性に基づく操作、もしくは、関係に参加している管理オブジェクトに対するオブジェクトに基づく操作として、マッピングできる。可能なマッピングを表２に列挙するが、ある特定の表現に対するマッピングは、関連する関係マッピングによりモデル化される。

表２－関係属性に対する操作のマッピング

プロトタイプ関係管理操作	関係に参加している 管理オブジェクトに対する 属性に基づく システム管理操作の候補	関係に参加している 管理オブジェクトに対する オブジェクトに基づく システム管理操作の候補
BIND	Replace、Add	Create、Action
UNBIND	Replace、Remove	Delete、Action
QUERY	Get	Action
ESTABLISH	Replace、Add	Create、Action
TERMINATE	Replace、Remove	Delete、Action

7.4.3　関係オブジェクトによる表現及び管理
（Representation and management by means of a relationship object）

関係マッピングは、関係オブジェクト　（relationship　object）　と呼ばれる管理オブジェクトを用いて、管理関係の表現及び管理をモデル化してもよい。全ての関係オブジェクトのスーパクラスである gegnericRelationshipObject は、以下に示す属性を持つ。

a)　**関係の名前**（relationship name）**－関係管理の名前を指定する。**

b)　**関係クラス**（relationship class）**－関係管理のクラスを指定する。**

c)　**関係マッピング**（ralationship mapping）**－実際に使用する関係マッピングを指定する。**

関係オブジェクトクラスは、関係管理のインスタンスに参加しているパーティシパントを識別するために、関係クラスで定義された各々のロールに対するパーティシパントポインタを持つ。

genericRelationshipObject は、 relationships 属性グループ（CCITT Rec. X.732 | ISO/IEC 10164-3 **参照）を公開している。この属性グループが、全てのパーティシパントポインタ属性を含んでもよい。**

関係管理操作は、関係オブジェクトに対する、オブジェクトに基づく操作、もしくは、属性に基づく操作として、マッピングできる。可能なマッピングを表３に列挙するが、ある特定の表現に対するマッピングは、関連する関係マッピングによりモデル化される。

表３－関係オブジェクトに対する操作のマッピング

プロトタイプ関係管理操作	関係オブジェクトに対する 属性に基づく システム管理操作の候補	関係オブジェクトに対する オブジェクトに基づく システム管理操作の候補
BIND	Replace、Add	Create、Action
UNBIND	Replace、Remove	Delete、Action
QUERY	Get	Action
ESTABLISH	Replace、Add	Create、Action
TERMINATE	Replace、Remove	Delete、Action

7.4.4　システム管理操作による表現及び管理

（Representation and management by means of system management operations）

関係マッピングは、関係に参加している管理オブジェクトに対するオブジェクトに基づくシステム管理操作を用いて、管理関係の表現及び管理をモデル化してもよい。関係管理操作は、関係に参加している管理オブジェクトに対するオブジェクトに基づく操作として、マッピングできる。可能なマッピングを表４に列挙するが、ある特定の表現に対するマッピングは、関連する関係マッピングによりモデル化される。

表４－管理操作により表現される関係に対する操作のマッピング

プロトタイプ関係管理操作	関係に参加している 関係オブジェクトに対する オブジェクトに基づく システム管理操作の候補
BIND	Create、Action
UNBIND	Delete、Action
QUERY	Action
ESTABLISH	Create、Action
TERMINATE	Delete、Action

８.汎用定義　（Generic definitions）

本勧告｜国際標準は、汎用管理情報と、プロトタイプ関係管理操作及び通知のセマンティクスを定義する。汎用管理情報のシンタックスの形式的な定義を、付属資料Ｂに示す。

8.1　関係管理操作及び通知　（Relationship management operations and notification）

8.1.1　ESTABLISH

不変式：　ロールカーディナリティ及び関係カーディナリティに違反しないこと。

前提条件：　管理関係が存在していないこと。ESTABLISH 操作で結合するように指定された管理オブジェクトが､指定されたロールのパーティシパントとなることを許可されたクラスであること。

結果条件：　管理関係が存在していること。ESTABLISH 操作で結合するように指定された管理オブジェクトが存在し、管理関係に結合されていること。

8.1.2　TERMINATE

前提条件：　管理関係が存在していること。

結果条件：　管理関係が存在していないこと。関係に結合されていた管理オブジェクトが､関係に結合されていないこと。

8.1.3　BIND

不変式：　管理関係が存在していること｡ロールカーディナリティ及び関係カーディナリティに違反しないこと。

前提条件：　BIND 操作で指定された管理オブジェクトのクラスが、指定されたロールのパーティシパントとなることを許可されたクラスであること。管理関係が、指定されたロールに対して、BIND 操作をサポートしていること。

結果条件：　BIND 操作で指定された管理オブジェクトが存在し、関係に結合されていること。

8.1.4 UNBIND

不変式：　管理関係が存在していること。ロールカーディナリティ及び関係カーディナリティに違反しないこと。

前提条件：　UNBIND 操作で指定された管理オブジェクトが存在し、管理関係に結合されていること。管理関係が、指定されたロールに対して、UNBIND 操作をサポートしていること。

結果条件：　UNBIND 操作で指定された管理オブジェクトが、関係管理に結合されていないこと。

8.1.5 QUERY

前提条件：　真（TRUE）。

結果条件：　管理関係が変化していないこと。

8.1.6 NOTIFY

前提条件：　真（TRUE）。

結果条件：　管理関係が変化していないこと。

8.1.7 USER DEFINED

本勧告｜国際標準は、本プロトタイプ操作に対しては、いかなるセマンティクスも定義しない。

8.2 管理オブジェクトクラス－genericRelationshipObject

（Managed object class - genericRelationshipObject）

全ての関係オブジェクトクラスは、**genericRelationshipObject** を特化したものである。**genericRelationshipObject** は、**relationshipMapping** 属性、**ralationshipClass** 属性、**relationshipName** 属性を含んでいる。特定の関係クラスために用いる関係オブジェクトクラスは、その管理関係クラスで定義された各々のロール毎に、 **participantPointer** 属性から派生する属性を持つ。

8.3 ネームバインディング－genericRelationshipObject-system

（Name binding - genericRelationshipObject-system)

関係オブジェクトの名前付けを **relationshipName** 属性を用いて行うために、本ネームバインディングを使用する。

8.4 属性　(Attribute)

8.4.1 属性－relationshipName　(Attribute - relationshipName)

管理関係の名前付け及び関係オブジェクトの名前付けを行うために、本属性を使用する。

8.4.2 属性ーrelationshipClass　（Attribute - relationshipClass）

関係管理のクラスを特定するために、本属性を使用する。この属性の値は、各々の管理関係テンプレートに割り当てられた値である。

8.4.3 属性ーrelationshipMapping　（Attribute - relationshipMapping）

実際に使用する関係マッピングを特定するために、本属性を使用する。この属性の値は、各々の関係マッピングテンプレートに割り当てられた値である。

8.4.4 属性ーparticipantPointer　(Attribute - participantPointer)

participantPointer 属性は未登録の属性であり、全てのパーティシパントポインタ属性に対するプロトタイプを提供する。本属性のシンタックスは、管理オブジェクト名の集合である。本属性の **MATCH FOR** 節には、**equality**、**set comparison**、**set intersection** を記述する。本属性は、特定エラーとして、**roleCardinalityViolation**、**relationshipCardinalityViolation**、**noSuchObject**、及び、**roleInstanceConflict** をサポートする。

派生した **participantPointer** 属性の値は、現在ロールを担っている管理オブジェクトを示す。本属性の型は、ロールを示す。

8.5 属性グループーrelationships　(Attribute group - relationships)

本属性は、CCITT Rec. X.723 | ISO/IEC 10164-3 に定義されており、全てのパーティシパントポインタをグループ化するために使用される。

8.6 パラメータ　(Parameters)

8.6.1 パラメーターnoSuchObject　(Parameter - noSuchObject)

本特定エラーは、関係管理操作 BIND が、実行者が認識していない管理オブジェクト名を指定したことを通知するために使用する。本パラメータの値は、BIND 操作で指定された名前である。

8.6.2 パラメーターroleCardinalityViolation　(Parameter - roleCardinalityViolation)

本特定エラーは、関係管理操作 BIND、もしくは、UNBIND が、関係管理のロールカーディナリティ制約の一つに違反することを通知するために使用する。本パラメータの値は、ヌル値である。

8.6.3 パラメーターroleInstanceConflict　(Parameter - roleInstanceConflict)

本特定エラーは、関係管理操作 BIND が、管理関係の関係マッピングで許容していないクラスの管理オブジェクト名を指定したことを通知するために使用する。本パラメータの値は、BIND 操作で指定された名前である。

8.6.4 パラメーターrelationshipCardinalityViolation　(Parameter - relationshipCardinalityViolation)

本特定エラーは、関係管理操作 BIND、もしくは、UBIND が関係管理の関係カーディナリティ制約に違反することを通知するために使用される。本パラメータの値は、ヌル値である。

Annex A

Relationship templates

(This annex forms an integral part of this Recommendation | International standard)

A.1 Relationship class template

A.1.1 Overview

The relationship class template forms the basis of the formal definition of a managed relationship. Constructs in the template allow the various characteristics of the managed relationship class to be defined, namely:

a) relationship inheritance;

b) relationship qualification;

c) relationship behaviour;

d) role compatibility;

e) role cardinality constraints;

f) bind and unbind support;

g) relationship cardinality constraints.

The following template labels and supporting definitions used in the relationship class template are defined in GDMO:

<behaviour-label>

<class-label>

<attribute-label>

object-identifier

type-reference

The following supporting definitions, used in the relationship class template, are defined in ASN.1:

– identifier.

Labels values shall be unique within the assigning document.

A.1.1.1 Inheritance

The managed relationship class template permits the specification of the managed relationship superclass(es) from which a managed relationship class has been derived. Characteristics of the superclass(es) are inherited by the subclass. The specialization of a subclass is such that a subclass of a managed relationship is consistent with its superclass(es).

A.1.1.2 Relationship qualification

The managed relationship class template permits the definition of characteristics that qualify the relationship as a whole and are independent of the particular representation method.

A.1.1.3 Behaviour

The managed relationship class template requires the specification of behaviour of a managed relationship that is independent of the particular representation method. Behaviour that is dependent on the particular representation method shall be specified in the relationship mapping template.

A.1.1.4 Roles

The managed relationship class template permits the definition of the roles of the relationship and their associated characteristics.

A.1.1.5 Relationship class identifier

The managed relationship class template requires the specification of an object identifier which may be used to reference the relationship class in management protocol.

A.1.2 Template structure

```
<relationship-class-label>     RELATIONSHIP CLASS
    [DERIVED FROM <relationship-class-label>
        [, <relationship-class-label>]* ;]
    BEHAVIOUR <behaviour-label> [, <behaviour-label]*;
    [SUPPORTS supported [, supported]*;]
    [QUALIFIED BY <attribute-label> [, <attribute-label>]*;]
    [role-specifier]*;
REGISTERED AS object-identifier;
```

supporting productions

```
supported–>
    ESTABLISH [operation-name]
    | TERMINATE [operation-name]
    | QUERY [operation-name]
    | NOTIFY [notification-name]
    | USER DEFINED [operation-name]
```

```
role-specifier–>
    ROLE role-name
        [COMPATIBLE-WITH <class-label> ]
        [PERMITTED-ROLE-CARDINALITY-CONSTRAINT type-reference]
        [REQUIRED-ROLE-CARDINALITY-CONSTRAINT type-reference]
        [BIND-SUPPORT [operation-name]]
        [UNBIND-SUPPORT [operation-name]]
        [PERMITTED-RELATIONSHIP-CARDINALITY-CONSTRAINT type-reference]
    [REGISTERED AS object-identifier]
```

```
role-name –> <identifier>
operation-name –> <identifier>
notification-name –> <identifier>
```

A.1.3 Supporting definitions

A.1.3.1 DERIVED FROM <relationship-class-label> [, <relationship-class-label>]*

This construct shall be used to specify the superclass(es) from which the managed relationship class inherits its characteristics including any which may, in turn, have been inherited from other managed relationship class(es). The managed relationship class is a specialization of the inherited characteristics and those specified in the balance of the completed template; the specialization is such that the subclass is consistent with its superclass(es). If this construct is absent, the managed relationship class is not specialized from other relationship classes.

Specification of characteristics that are inherited from other managed relationship classes shall not be repeated in the specification of the subclass unless one of the techniques described in CCITT Rec. X.722 | ISO/IEC 10165-4 for extending a specification inherited from a superclass is being used.

The rules for specifying managed relationship subclasses to ensure consistency are as follows:

a) **SUPPORTS:** The specialized relationship management operations shall be the union of the relationship management operations of the superclasses and those specified in the subclass; inheritance and specialization shall not introduce additional relationship management notifications into a subclass.

b) **QUALIFIED BY:** Permitted and required value-sets of attribute ranges shall not be changed in a subclass.ehaviou

c) **BEHAVIOUR:** The br of a subclass shall be:

- the disjunctive combination of the pre-conditions inherited from its superclass(es) and those specified in the subclass;
- the conjunctive combination of the post-conditions inherited from its superclass(es) and those specified in the subclass;
- the conjunctive combination of the invariants inherited from its superclass(es) and those specified in the subclass; if the invariants are mutually contradictory, a subclass cannot be specified.

d) **ROLE:**

- Additional role specifications may be included in the subclass definition.
- A managed objects class introduced by the **COMPATIBLE WITH** clause in the subclass shall be compatible[2] to those referenced in similar clauses in the superclass(es).
- The inherited **PERMITTED-ROLE-CARDINALITY-CONSTRAINT** value of a role inherited from more than one superclass shall be the set intersection of the values specified for that role in the superclasses; any permitted role cardinality constraint value specified in the subclass shall be a subset of, or equal to, the inherited permitted role cardinality constraint value; the specialized permitted role cardinality constraint value shall be the set intersection of the inherited values and that specified in the subclass.
- The inherited **REQUIRED-ROLE-CARDINALITY CONSTRAINT** value of a role inherited from more than one superclass shall be the set union of the values specified for the role in the superclasses set-intersected with the inherited permitted role cardinality constraint value; any required role cardinality constraint value specified in the subclass shall be a superset of, or equal to, the inherited required role cardinality constraint value; the specialized required role cardinality constraint value shall be the set union of the inherited value and that specified in the subclass set-intersected with the value of the specialized permitted role cardinality constraint.
- **BIND-SUPPORT** may be added in the subclass specification.
- **UNBIND-SUPPORT** may be added in the subclass specification.
- The inherited **PERMITTED-RELATIONSHIP-CARDINALITY-CONSTRAINT** value of a role inherited from more than one superclass shall be the set intersection of the values specified for the role in the superclasses; any permitted relationship cardinality constraint value specified in the subclass shall be a subset of, or equal to, the inherited permitted relationship cardinality constraint value; the specialized permitted relationship cardinality constraint value shall be the set intersection of the inherited value and that specified in the subclass.

e) **REGISTERED AS:** The subclass registration shall replace any registration inherited from other definitions.

A.1.3.2 BEHAVIOUR <behaviour-label>[, <behaviour-label>]

This construct shall be used to specify the representation-independent behaviour of the managed relationship. It shall be stated in terms of an invariant over the managed relationship and invariant and pre- and post-conditions for the relationship management operations and notification. The construct references behaviour templates as defined in CCITT Rec. X.722 | ISO/IEC 10165-4.

A.1.3.3 SUPPORTS supported [, supported]*

This construct shall be used to define the relationship management operations and notifications that a managed relationship supports. The **supported** supporting production shall be used to specify the prototypical operation or notification on which the relationship management operation or notification is based, namely:

- **ESTABLISH [operation-name];**
- **TERMINATE [operation-name];**
- **QUERY [operation-name];**
- **NOTIFY [notification-name];**
- **USER DEFINED [operation-name].**

The **operation-name** and **notification-name** shall be used, where necessary, to:

- provide a link to an optional specification, in behaviour templates referenced by the **BEHAVIOUR** construct, of behaviour additional to that specified for the referenced prototypical operation;
- disambiguate relationship management operations or notifications that are based on the same prototypical operation or notification respectively;
- provide a link to the related systems management operations and notifications specified in the relationship mapping template.

[2] The concept of compatibility is discussed in 5.2 of MIM.

A.1.3.4 QUALIFIED BY <attribute-label>[, <attribute-label>]*

This construct shall be used to specify attributes that are associated with the managed relationship as a whole. Qualifying attributes shall be made available in all implementations of the managed relationship irrespective of the representation method used. The relationship mapping template shall be used to specify how these attributes are made available by a particular representation.

A.1.3.5 ROLE role-name

This construct shall be used to specify the roles associated with the managed relationship class; the label role-name shall be used as a reference name to the role.

A.1.3.5.1 COMPATIBLE WITH <class-label>

This construct shall be used to specify the characteristics required of a managed object to fulfil the requirements of the role; the characteristics shall be specified in terms of a compatible[3)] managed object class. If the construct is not present the characteristics of **top** (see CCITT Rec. X.721 | ISO/IEC 10165-2) are assumed. The role specification is independent of the representation method.

A.1.3.5.2 PERMITTED-ROLE-CARDINALITY-CONSTRAINT type-reference

This construct shall be used to specify any restriction on the number of managed objects that a managed relationship is permitted to support in the role. It shall reference an ASN.1 subtype value set of non-negative integers.

For example, if the construct specifies a value set of **INTEGER (1..3)**, a managed relationship is permitted to support either 1, 2, or 3 managed objects in the role but it is not permitted to support more than 3 managed objects in the role. An implementation is required to enforce the constraint.

If the value set contains 0, the role is optional; however an optional role does not imply support of either the bind- or unbind-operations. If the construct is absent, the inherited permitted role cardinality constraint shall be used as the default; if no constraint has been inherited, a value set **INTEGER (0..MAX)** is assumed for the constraint.

The **PERMITTED-ROLE-CARDINALITY-CONSTRAINT** value set shall be a superset of, or equal to, the **REQUIRED-ROLE-CARDINALITY-CONSTRAINT** value set.

A.1.3.5.3 REQUIRED-ROLE-CARDINALITY-CONSTRAINT type-reference

This construct shall be used to specify any restriction on the number of managed objects that a managed relationship is required to support in the referenced role. The constraint shall be specified in terms of an ASN.1 subtype value set of non-negative integers. For example, if the construct specifies a value set of **INTEGER (1, 3, 4)**, a managed relationship is required to support either 1, 3, or 4 managed objects in the role but it is not required to support either 2 managed objects or more than 4 managed objects in the role. An implementation is required to enforce the constraint.

If the value set contains 0, the role is optional; however an optional role does not imply support of either the bind- or unbind-operations. If the construct is absent, the inherited required role cardinality constraint value shall be used as the default; if no value has been inherited, the managed relationship is not required to support the constraint.

The **REQUIRED-ROLE-CARDINALITY-CONSTRAINT** value set shall always be a subset of, or equal to, the **PERMITTED-ROLE-CARDINALITY-CONSTRAINT** value set.

A.1.3.5.4 BIND-SUPPORT [operation-name]

This construct shall be used to specify that managed objects may become participants in the role during the existence of the relationship provided that role cardinality constraints are not violated. Absence of this construct implies that managed objects may not become participants in the role during the existence of the relationship.

The operation-name shall be used, where necessary, to:

- provide a link to an optional specification, in behaviour templates referenced by the **BEHAVIOUR** construct, of behaviour additional to that specified for the referenced **BIND** prototypical operation;
- disambiguate multiple relationship management operations that are based on the **BIND** prototypical operation;
- provide a link to the related systems management operations specified in the relationship mapping template.

[3)] The concept of compatibility is discussed in 5.2 of MIM.

A.1.3.5.5 UNBIND-SUPPORT [operation-name]

This construct shall be used to specify that participants may be released from the role during the existence of the relationship provided that role cardinality constraints are not violated. Absence of this construct implies that participants may not be released from the role during the existence of the relationship.

The operation-name shall be used, where necessary, to:

- provide a link to an optional specification, in behaviour templates referenced by the **BEHAVIOUR** construct, of behaviour additional to that specified for the referenced **UNBIND** prototypical operation;
- disambiguate multiple relationship management operations that are based on the **UNBIND** prototypical operation;
- provide a link to the related systems management operations specified in the relationship mapping template.

A.1.3.5.6 PERMITTED-RELATIONSHIP-CARDINALITY-CONSTRAINT type-reference

This construct shall be used to specify a restriction on the number of relationships of the referenced class in which managed object is permitted to participate in the referenced role. The constraint shall be specified in terms of an ASN.1 subtype value set of non-negative integers. For example, if the construct specifies a value set of **INTEGER (0..3)**, a managed object is permitted to participate in up to and including, but not more than, three instances of the referenced managed relationship class in the given role. An implementation is required to enforce the constraint. If the construct is absent, the inherited permitted relationship cardinality constraint shall be used as the default; if no constraint has been inherited, a value set **INTEGER (0..MAX)** is assumed for the constraint.

A.1.3.5.7 REGISTERED AS object-identifier

This construct shall be used to specify a globally-unique identifier which registers the role; the identifier may be used in protocol to unambiguously identify the role. If the role has been inherited, this construct shall not be present.

A.1.3.6 REGISTERED AS object-identifier

This construct shall be used to specify a globally-unique identifier which registers the managed relationship class; the identifier may be used in protocol to unambiguously identify the managed relationship class.

A.2 Relationship mapping template

A.2.1 Overview

The relationship mapping template forms the basis of the formal definition of a relationship mapping. Constructs in the template allow the various elements of the representation to be defined, namely:

a) relationship mapping behaviour;

b) relationship objects;

c) candidate classes from which managed objects may be drawn to fulfil roles;

d) representational methods;

e) qualification attributes;

f) operation and notification mappings.

The following template labels and supporting definitions used in the relationship mapping template are defined in GDMO:

<action-label>
<attribute-label>
<behaviour-label>
<class-label>
<name-binding-label>
<notification-label>
<parameter-label>
object-identifier

The following supporting definition, used in the relationship mapping template, is defined in ASN.1:

- identifier.

Labels values shall be unique within the assigning document.

A.2.1.1 Behaviour

The relationship mapping template specifies any behaviour that is peculiar to the representation method defined in the template.

A.2.1.2 Representation methods

The relationship mapping template requires the specification of the method used to represent a managed relationship and any relevant management information associated with the role representation.

A.2.1.3 Roles

The relationship mapping template requires the specification of the mapping of roles and relationship qualifications to managed object classes.

A.2.2 Template structure

```
<relationship-mapping-label> RELATIONSHIP MAPPING
    RELATIONSHIP CLASS <relationship-class-label> ;
    BEHAVIOUR <behaviour-label> [, <behaviour-label>]*;
    [RELATIONSHIP OBJECT <class-label> [QUALIFIES <attribute-label>
        [, <attribute-label>]*];]
    role-mapping-specification [, role-mapping-specification]*;
    [OPERATIONS MAPPING relationship-operation maps-to
        [, relationship-operation maps-to ]* ;]
REGISTERED AS object-identifier;

supporting productions

role-mapping-specification –>
    ROLE role-name RELATED-CLASSES <class-label> [<class-label>]*
    [REPRESENTED-BY representation]
    [QUALIFIES <attribute-label> [ <attribute-label>]*]

representation ->
    NAMING <name-binding-label> USING superiorOrSubordinate
    | ATTRIBUTE <attribute-label>
    | RELATIONSHIP-OBJECT-USING-POINTER <attribute-label>
    | OPERATION

superiorOrSubordinate –>
    SUPERIOR|SUBORDINATE

relationship-operation –>
    ESTABLISH [operation-name]
    | TERMINATE [operation-name]
    | BIND [operation-name] [role-name]
    | UNBIND [operation-name] [role-name]
    | QUERY [operation-name] [role-name]
    | NOTIFY [notification-name]
    | USER DEFINED [operation-name]

maps-to –>
    MAPS-TO-OPERATION systems-management-operation
        OF role-or-relObject [systems-management-operation
        OF role-or-relObject]*

systems-management-operation –>
    GET <attribute-label> [<parameter-label>]*
    | REPLACE <attribute-label> [<parameter-label>]*
    | ADD <attribute-label> [<parameter-label>]*
    | REMOVE <attribute-label> [<parameter-label>]*
    | CREATE [<class-label>] [<parameter-label>]*
    | DELETE [<parameter-label>]*
    | ACTION <action-label> [<parameter-label>]*
    | NOTIFICATION <notification-label> [<parameter-label>]*

role-or-relObject –> role-name | RELATIONSHIP OBJECT
role-name –> <identifier>
operation-name –> <identifier>
notification-name –> <identifier>
```

A.2.3 Supporting definitions

A.2.3.1 RELATIONSHIP CLASS <relationship-class-label>

This construct shall be used to specify the managed relationship class to which this relationship mapping is related.

A.2.3.2 BEHAVIOUR <behaviour-label> [, <behaviour-label>]

This construct shall be used to specify the representation-dependent behaviour of the managed relationship and its relationship operations and notifications. It shall be stated in terms of an invariant over the participating managed objects and an invariant and pre- and post-conditions over systems management operations and notifications related to participating managed objects. The construct shall not specify behaviour in addition to that already exhibited by the participating managed objects.

A.2.3.3 RELATIONSHIP OBJECT <class-label> [QUALIFIES <attribute-label> [, <attribute-label>]*]

This construct is present in templates which specify the representation of a managed relationship by means of a relationship object. The **<class-label>** shall be used to indicate the class of the relationship object; in a real implementation the class of the relationship object shall that referenced by **<class-label>** or a subclass thereof. The managed object class referenced by **<class-label>** shall be a subclass of **genericRelationshipObject** and shall exhibit participant pointer attributes for each of the roles specified in the associated managed relationship class template.

The **QUALIFIES <attribute-label> [, <attribute-label>]*** construct shall be used to specify the relationship qualification attributes, defined in the referenced relationship class template, that are to be realized by the relationship object.

A.2.3.4 ROLE role-name RELATED-CLASSES <class-label> [<class-label>]* [REPRESENTED-BY representation] [QUALIFIES <attribute-label> [<attribute-label>]*]

This construct shall be used to identify candidate managed object classes, referenced by **<class-label> [<class-label>]***, that may fulfil the role, referenced by **role-name**. Therole shall be one of the roles specified in the referenced managed relationship class template; the classes shall be compatible with that referenced in the **COMPATIBLE WITH** clause of the referenced relationship class template. Only managed objects of the classes specified in the **<class-label> [, <class-label>]*** construct and their subclasses shall be permitted to fulfil the role in an instance of the referenced relationship class that uses this mapping.

The **representation** supporting definition shall specify the method by which the referenced role is to be represented and any associated management information. A choice of one of the following productions shall be used to specify representation by naming, participant pointers, relationship object, or systems management operations respectively:

- **NAMING <name-binding-label> USING superiorOrSubordinate**: The role referenced by role-name shall be represented by an object of either the **SUPERIOR OBJECT CLASS** or the **SUBORDINATE OBJECT CLASS** indicated in the name binding referenced by **<name-binding-label>**; the expansion of the **superiorOrSubordinate** supporting production, **SUPERIOR** or **SUBORDINATE**, shall indicate either the **SUPERIOR OBJECT CLASS** or the **SUBORDINATE OBJECT CLASS** respectively.
- **ATTRIBUTE <attribute-label>**: The type of the attribute referenced by **<attribute-label>** shall indicate the referenced role; the value of the attribute shall specify participant(s) fulfilling that role.
- **RELATIONSHIP-OBJECT-USING-POINTER <attribute-label>**: The type of the attribute referenced by **<attribute-label>** shall indicate the referenced role; the value of the attribute shall specify participant(s) fulfilling that role.
- **OPERATION**: The mapping of relationship management operations to systems management operations shall be specified in the **OPERATIONS MAPPING** construct.

The **QUALIFIES <attribute-label> [<attribute-label>]*** construct identifies relationship qualification attributes, defined in the referenced relationship class template, that are to be realized by the referenced managed object classes.

A.2.3.5 OPERATIONS MAPPING relationship-operation maps-to [, relationship-operation maps-to]*

This construct shall be used to specify the mapping of a relationship management operation to one or more systems management operations.

The **relationship-operation** supporting definition specifies a choice of one of the following productions which shall be used to indicate the respective relationship management operation or notification and the role to which it refers:

- **ESTABLISH [operation-name];**
- **TERMINATE [operation-name];**
- **BIND [operation-name][role-name];**

- **UNBIND [operation-name] [role-name];**
- **QUERY [operation-name] [role-name];**
- **NOTIFY [notification-name];**
- **USER DEFINED [operation-name].**

The **operation-name** or **notification-name** specified shall be one of those defined in the related relationship class template and shall form, where required, the link between the semantics of the relationship management operations and notifications and their representation in terms of systems management operations and notifications. Where a managed relationship defines only one role, the specification of a **role-name** is optional.

The **maps-to** supporting definition specifies the following production:

- **MAPS TO OPERATION systems-management-operation OF role-or-relObject [systems-management-operation OF role-or-relObject]***

The **systems-management-operation** supporting definition specifies a choice of one of the following productions each of which indicates the respective systems management operation or notification and related systems management information; the **[<parameter-label>]** shall be used to specify any parameters to be associated with the systems management operation or notification:

- **GET <attribute-label> [<parameter-label>]*–**The attribute referenced by **<attribute-label>** shall specify the attribute value to be retrieved.
- **REPLACE <attribute-label> [<parameter-label>]*–**The attribute referenced by **<attribute-label>** shall specify the attribute value to be replaced.
- **ADD <attribute-label> [<parameter-label>]*–**The attribute referenced by **<attribute-label>** shall specify the attribute to which the value is to be added.
- **REMOVE <attribute-label> [<parameter-label>]*–**The attribute referenced by **<attribute-label>** shall specify the attribute from which the value is to be removed.
- **CREATE [<class-label>] [<parameter-label>]*–**The class referenced by the **<class-label>** shall specify the class to which the created managed object is to belong.
- **DELETE [<parameter-label>]*.**
- **ACTION <action-label> [<parameter-label>]*–**The action referenced by the **<action-label>** shall specify the action to be issued.
- **NOTIFICATION <notification-label> [<parameter-label>]*–**The notification referenced by the **<notification-label>** shall specify the notification to be issued.

The **role-or-relObject** supporting definition specifies target or source managed objects for the referenced systems management operation. A choice of one of the following productions is permitted which shall be used to specify either the managed object fulfilling the role referenced in role-name or the relationship object respectively:

- **role-name**;
- **RELATIONSHIP-OBJECT**.

A.2.3.6 REGISTERED AS object-identifier

This construct shall be used to specify a globally-unique identifier which registers the relationship mapping; the identifier may be used in protocol to unambiguously identify the relationship mapping.

Annex B

Definition of management information[4]

(This annex forms an integral part of this Recommendation | International Standard)

B.1 Allocation of object identifiers

This Recommendation | International Standard allocates the following object identifiers:

```
GRMD {joint-iso-itu-t ms(9) smi(3) part7(7) asn1Module(2) 1}

DEFINITIONS ::= BEGIN

grm-Object OBJECT IDENTIFIER  ::=  {joint-iso-itu-t ms(9) smi(3) part7(7) managedObjectClass(3)}
grm-Package OBJECT IDENTIFIER   ::=  {joint-iso-itu-t ms(9) smi(3) part7(7) package(4)}
grm-Parameter OBJECT IDENTIFIER   ::=  {joint-iso-itu-t ms(9) smi(3) part7(7) parameter(5)}
grm-NameBinding OBJECT IDENTIFIER   ::=  {joint-iso-itu-t ms(9) smi(3) part7(7) nameBinding(6)}
grm-Attribute OBJECT IDENTIFIER   ::=  {joint-iso-itu-t ms(9) smi(3) part7(7) attribute(7)}
grm-RelationshipClass OBJECT IDENTIFIER   ::=  {joint-iso-itu-t ms(9) smi(3) part7(7) relationshipClass(11)}
grm-RelationshipMapping OBJECT IDENTIFIER   ::=  {joint-iso-itu-t ms(9) smi(3) part7(7) relationshipMapping(12)}
grm-RelationshipRole OBJECT IDENTIFIER   ::=  {joint-iso-itu-t ms(9) smi(3) part7(7) relationshipRole(13)}

END
```

B.2 Definition of managed object classes

```
genericRelationshipObject   MANAGED OBJECT CLASS
      DERIVED FROM "CCITT Rec. X.721 | ISO/IEC 10165-2":top;
      CHARACTERIZED BY genericRelationshipObjectPackage PACKAGE
         ATTRIBUTES   relationshipName GET,
                  relationshipClass GET,
                  relationshipMapping GET;
         ATTRIBUTE GROUPS
      "CCITT Rec. X.721 | ISO/IEC 10165-2":relationships;;;
REGISTERED AS {GRMD.grm-Object 1};
```

B.3 Definition of name bindings

```
genericRelationshipObject-system   NAME BINDING
   SUBORDINATE OBJECT CLASS   genericRelationshipObject AND SUBCLASSES;
   NAMED BY SUPERIOR OBJECT CLASS   "CCITT Rec. X.721 | ISO/IEC 10165-2":system AND SUBCLASSES;
   WITH ATTRIBUTE  relationshipName;
REGISTERED AS {GRMD.grm-NameBinding 1};
```

B.4 Definition of attributes

```
relationshipName   ATTRIBUTE
   WITH ATTRIBUTE SYNTAX   GRM-ASN1Module.SimpleNameType;
REGISTERED AS {GRMD.grm-Attribute 1};

relationshipClass   ATTRIBUTE
      WITH ATTRIBUTE SYNTAX   GRM-ASN1Module.RelationshipClass;
      MATCHES FOR EQUALITY;
REGISTERED AS {GRMD.grm-Attribute 2};

relationshipMapping   ATTRIBUTE
      WITH ATTRIBUTE SYNTAX   GRM-ASN1Module.RelationshipMapping;
      MATCHES FOR EQUALITY;
REGISTERED AS {GRMD.grm-Attribute 3};
```

[4] Users of this Recommendation | International Standard may freely reproduce the contents of this annex so that it can be used for its intended purpose.

```
participantPointer ATTRIBUTE
        WITH ATTRIBUTE SYNTAX  GRM-ASN1Module.GroupObjects;
        MATCHES FOR EQUALITY, SET-INTERSECTION, SET-COMPARISON;
        PARAMETERS    noSuchObject,
                roleInstanceConflict,
                roleCardinalityViolation,
                relationshipCardinalityViolation;;
--    An implementation may choose to apply ASN.1 subtyping restrictions to the attribute syntax of the
--    participantPointer attribute to reflect the permitted role cardinality constraints defined in a
--    specification.
```

B.5 Definition of parameters

```
noSuchObject PARAMETER
        CONTEXT SPECIFIC-ERROR;
        WITH SYNTAX GRM-ASN1Module.ObjectInstance;
REGISTERED AS {GRMD.grm-Parameter 1};

roleCardinalityViolation PARAMETER
        CONTEXT SPECIFIC-ERROR;
        WITH SYNTAX GRM-ASN1Module.Null;
REGISTERED AS {GRMD.grm-Parameter 3};

roleInstanceConflict PARAMETER
        CONTEXT SPECIFIC-ERROR;
        WITH SYNTAX GRM-ASN1Module.ObjectInstance;
REGISTERED AS {GRMD.grm-Parameter 2};

relationshipCardinalityViolation PARAMETER
        CONTEXT SPECIFIC-ERROR;
        WITH SYNTAX GRM-ASN1Module.Null;
REGISTERED AS {GRMD.grm-Parameter 4};
```

B.6 Abstract syntax definitions

```
GRM-ASN1Module {joint-iso-itu-t ms(9) smi(3) part7(7) asn1Module(2) 2}
DEFINITIONS ::= BEGIN

IMPORTS   ObjectInstance   FROM CMIP-1 {joint-iso-itu-t ms(9) cmip(1) version(1) protocol(3) }
SimpleNameType, GroupObjects
            FROM Attribute-ASN1Module {joint-iso-itu-t ms(9) smi(3) part2(2) asn1Module(2) 1}
RelationshipClass ::= OBJECT IDENTIFIER
RelationshipMapping ::= OBJECT IDENTIFIER
Null   ::=   NULL

END
```

Annex C

Managed relationship conformance statement proforma[5] *for General Relationship Model*

(This annex forms an integral part of this Recommendation | International Standard)

C.1 Introduction

The purpose of the proforma in this annex is to provide guidelines the for Managed Relationship Conformance Statement (MRCS) so that a supplier of an implementation which claims to conform to a managed relationship class can provide conformance information in a standard form. The proforma defined in this annex is an additional proforma to that specified in ITU-T Rec. X.724 | ISO/IEC 10165-6.

C.2 Instructions for completing the MRCS proforma

The MRCS proforma contained in this annex is comprised of information in tabular form. The supplier of the implementation shall state which items are supported in Tables C.1 to C.3 and if necessary provide additional information.

C.3 Symbols, abbreviations and terms

The following common notations defined in CCITT Rec. X.291 | ISO/IEC 9646-2 are used for the status columns:

m	Mandatory
o	Optional
c	Conditional
x	Prohibited
–	Not applicable

The following common notations, defined in CCITT Rec. X.291 | ISO/IEC 9646-2 and ITU-T Rec. X.296 | ISO/IEC 9646-7 are used for support columns:

Y	Implemented
N	Not implemented
–	No answer required
Ig	The item is ignored (i.e. processed syntactically but not semantically)

C.4 Managed relationship support

The supplier of the implementation shall state the managed relationship class and the relationship mappings supported using Table C.1.

Table C.1 – Managed relationship support

Index	Relationship class template label	Value of object identifier for relationship class	Relationship mapping template label	Value of object identifier for relationship mapping	Status	Support	Additional information
1							

C.4.1 Roles support

For each role identified in the relationship mapping, the supplier of the implementation shall indicate support using Table C.2.

[5] Users of this Recommendation | International Standard may freely reproduce the MRCS proforma in this annex so that it can be used for its intended purpose.

Table C.2 – Roles support

Index	Role label	Constraints and values	Status	Supported	Value of object identifier for actual participants managed object class	MOCS reference for actual participants managed object class	Additional information
1							
2							

C.4.1.1 Relationship management operations, notifications, and parameters support

The supplier of the implementation shall indicate the relationship management operations and notifications supported using Table C.3.

The supplier of the implementation shall indicate support for the parameters, if any, specified in the relationship mapping template by using the parameter support table specified in Annex D.

Table C.3 – Relationship management operations and notifications support

Index	Relationship management operation or notification	Systems management operation or notification	Constraints and values	Status	Support	Additional information
1						
2						

C.4.2 Relationship object support

The supplier of the implementation shall indicate support for the relationship object class, if any, specified in the relationship mapping template by using the MOCS proforma defined in ITU-T Rec. X.724 | ISO/IEC 10165-6 and MIDS proforma defined in Annex D. The relationship object class shall be a subclass of **genericRelationshipObject**.

Annex D

MIDS (attribute) proforma[6]

(This annex forms an integral part of this Recommendation | International Standard)

D.1 Introduction

The purpose of the proforma in this annex is to provide guidelines for the Management Information Definition Statement (MIDS) so that a supplier of an implementation which claims to conform to a managed relationship class can provide conformance information in a standard form.

D.2 Attributes

See Table D.1.

Table D.1 – Attribute support

				Set by create		Get		Replace	
Index	Attribute template label	Value of object identifier for attribute	Constraints and values	Status	Support	Status	Support	Status	Support
1	relationshipName	{joint-iso-itu-t ms(9) smi(3) part7(7) attribute(7) 1}		o		m		x	
2	relationshipClass	{joint-iso-itu-t ms(9) smi(3) part7(7) attribute(7) 2}		o		m		x	
3	roleMapping	{joint-iso-itu-t ms(9) smi(3) part7(7) attribute(7) 3}		o		m		x	
4	participantPointer	–		o		o		o	

Table D.1 *(concluded)* **– Attribute support**

	Add		Remove		Set to default		
Index	Status	Support	Status	Support	Status	Support	Additional information
1	–		–		–		
2	–		–		–		
3	–		–		–		
4	o		o		–		

D.3 Parameters

See Table D.2.

[6] Users of this Recommendation | International Standard may freely reproduce the MIDS proforma in this annex so that it can be used for its intended purpose. Instructions for constructing MIDS (attribute) proforma are specified in ITU-T Rec. X.724 | ISO/IEC 10165-6.

Table D.2 – Parameter support

Index	Parameter template label	Value of object identifier for parameter	Constraints and values	Status	Support	Additional information
1	noSuchObject	{joint-iso-itu-t ms(9) smi(3) part7(7) parameter(5) 1}		o		
2	roleCardinalityViolation	{joint-iso-itu-t ms(9) smi(3) part7(7) parameter(5) 3}		o		
3	roleInstanceConflict	{joint-iso-itu-t ms(9) smi(3) part7(7) parameter(5) 2}		o		
4	relationshipCardinalityViolation	{joint-iso-itu-t ms(9) smi(3) part7(7) parameter(5) 4}		o		

Annex E
Illustration of representation methods

(This annex does not form an integral part of this Recommendation | International Standard)

This annex presents a graphical interpretation of the layout and use of the Relationship Class and Relationship Mapping Templates (see Figures E.1 and E.2).

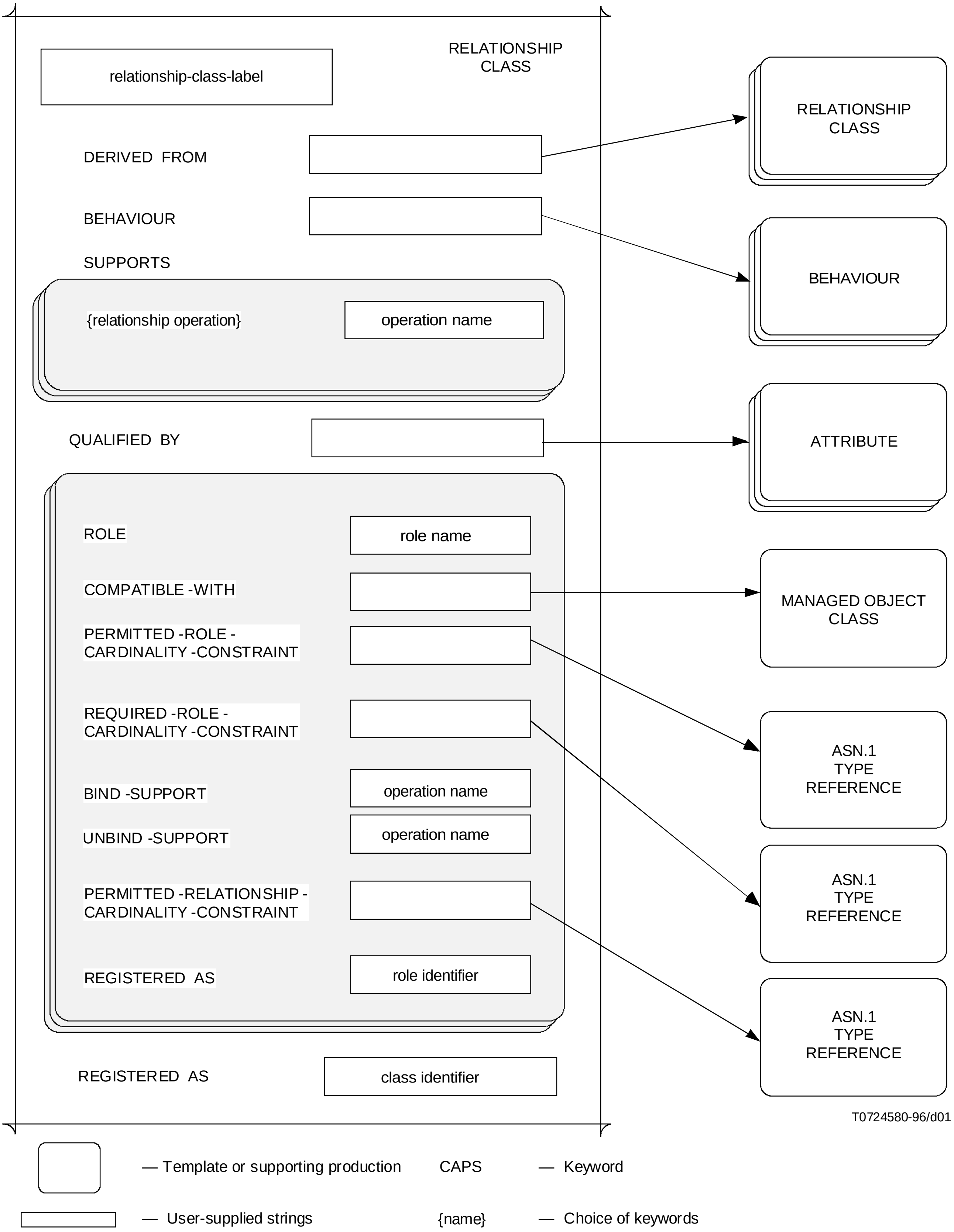

Figure E.1 – Relationship Class Template

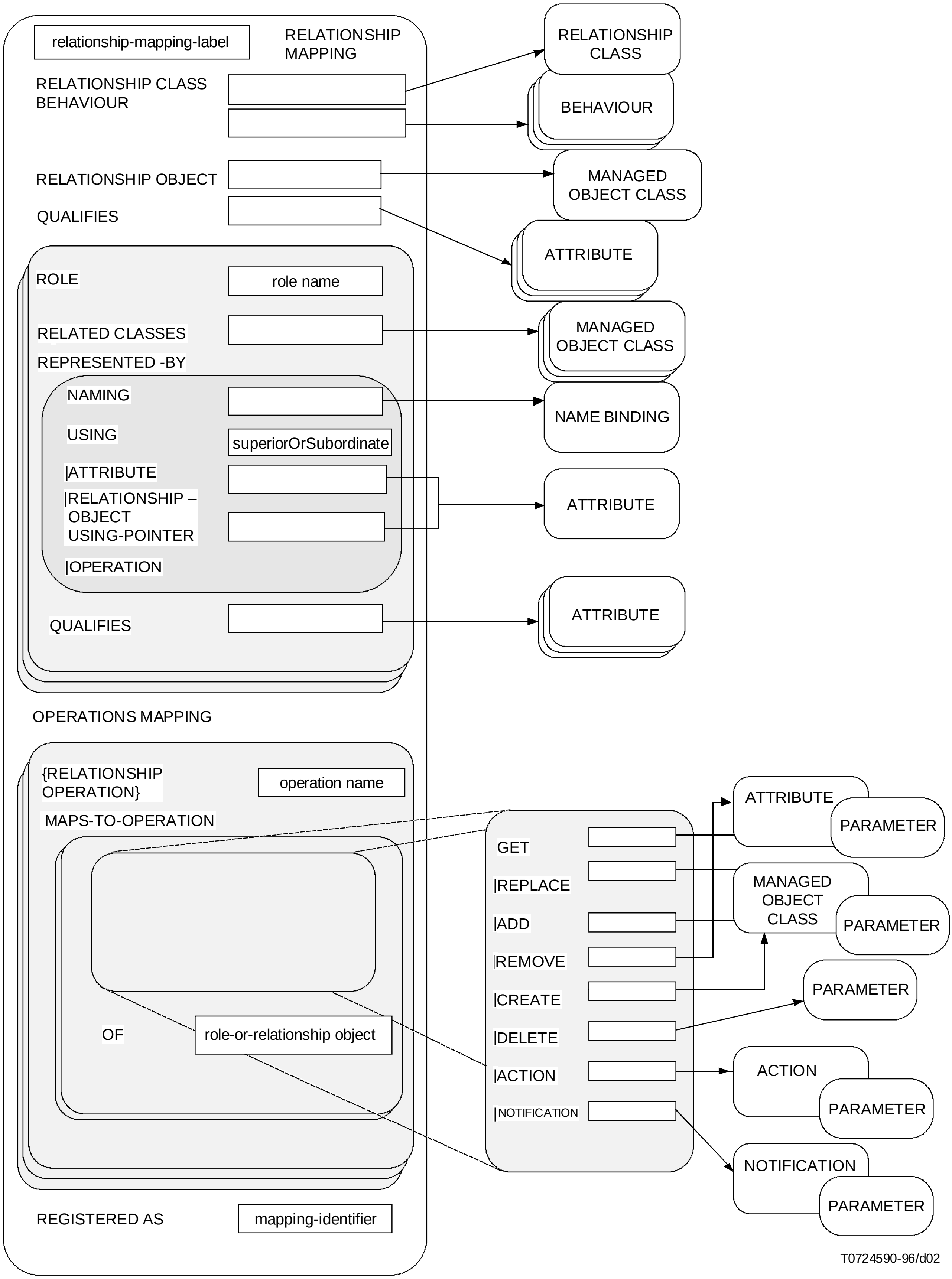

Figure E.2 – Relationsphip Mapping Template

Annex F

Examples of use of templates

(This annex does not form an integral part of this Recommendation | International Standard)

The examples shown in this annex are intended to provide illustration of the concepts identified in this Recommendation | International Standard and to give examples of the use of the **RELATIONSHIP CLASS** and **RELATIONSHIP MAPPING** template notations. These examples are not intended to provide definitions which are necessarily useful in real implementations.

F.1 Allocation of object identifiers

```
GRMExample {joint-iso-itu-t ms(9) smi(3) part7(7) asn1Module(2) exampleASN1(99)}

DEFINITIONS ::=
BEGIN
    grmEx-Role OBJECT IDENTIFIER ::=
        {joint-iso-itu-t ms(9) smi(3) part7(7) grm-Role(13) exampleRole(99)}
    grmEx-RelationshipClass OBJECT IDENTIFIER ::=
        {joint-iso-itu-t ms(9) smi(3) part7(7) grm-RelationshipClass(11)exampleRelationshipClass(99)}
    grmEx-RelationshipMapping OBJECT IDENTIFIER ::=
        {joint-iso-itu-t ms(9) smi(3) part7(7) grm-RelationshipMapping(12) exampleRelationshipMapping(99)}
    grmEx-Object OBJECT IDENTIFIER ::=
        {joint-iso-itu-t ms(9) smi(3) part7(7) managedObjectClass(3) exampleObjectClass(99)}
    grmEx-Attribute OBJECT IDENTIFIER ::=
        {joint-iso-itu-t ms(9) smi(3) part7(7) attribute(7) exampleAttribute(99)}
    grmEx-NameBinding OBJECT IDENTIFIER ::=
        {joint-iso-itu-t ms(9) smi(3) part7(7) nameBinding(6) exampleNameBinding(99)}
    grmEx-Package OBJECT IDENTIFIER ::=
        {joint-iso-itu-t ms(9) smi(3) part7(7) package(4) examplePackage(99)}

    PersonName ::=      GraphicString
    SingleValued ::=    GroupObjects (SIZE (1))

    ZeroToTwo ::=       INTEGER (0..2)
    One ::=             INTEGER (1..1)
    OneToFive ::=       INTEGER (1..5)
    OneToMax ::=        INTEGER (1..MAX)
    Two ::=             INTEGER (2..2)
    TwoToMax ::=        INTEGER (2..MAX)

END
```

F.2 Symmetric relationship example

The following example shows how the relationship class template may be used to define a generic, single-role relationship between objects of the same class and how the relationship mapping template may be used to define a representation.

F.2.1 Symmetric relationship class definition

```
symmetricRelationship    RELATIONSHIP CLASS
    BEHAVIOUR    symmetricRelationshipBehaviour;

    SUPPORTS
        ESTABLISH  establishSymmetricRelationship,
        TERMINATE       terminateSymmetricRelationship,
        QUERY           querySymmetricRelationship;

    ROLE            peerRole
        PERMITTED-ROLE-CARDINALITY-CONSTRAINT               GRMExample.TwoToMax
        REQUIRED-ROLE-CARDINALITY-CONSTRAINT                GRMExample.Two
        PERMITTED-RELATIONSHIP-CARDINALITY-CONSTRAINT       GRMExample.One
    REGISTERED AS {GRMExample.grmEx-Role x};
REGISTERED AS {GRMExample.grmEx-RelationshipClass x};
```

symmetricRelationshipBehaviour BEHAVIOUR DEFINED AS "

INVARIANT: This relationship has one role – the peer role – for which the minimum permitted and minimum required role cardinality constraint is 2. The existence of an instance of this relationship class implies the existence of at least two corresponding managed objects fulfilling the peer role.

OPERATIONS:

ESTABLISH establishSymmetricRelationship

Signature: The class and identity of the proposed participants in the peer role to be bound in a new instance of the Symmetric Relationship class.

Precondition: The instance of the Symmetric Relationship relationship class does not exist.

Postcondition: The participants in the peer role exist; the instance of the Symmetric Relationship relationship class exists; the participants in the peer role referenced in the signature are bound in this instance of the Symmetric Relationship class.

TERMINATE terminateSymmetricRelationship

Signature: The identity of the Symmetric Relationship relationship instance to be terminated.

Precondition: The instance of the Symmetric Relationship relationship class referenced in the signature exists; the participants in the peer role bound in this instance of the Symmetric Relationship class exist.

Postcondition: The referenced instance of the Symmetric Relationship relationship class does not exist; the participants in the peer role which were bound in this instance of the Symmetric Relationship class exist.";

F.2.2 Symmetric relationship represented by a relationship object

```
symmetricRelationshipMapping RELATIONSHIP MAPPING
   RELATIONSHIP CLASS symmetricRelationship;

   BEHAVIOUR symmetricRelationshipMappingBehaviour BEHAVIOUR DEFINED AS "

   This representation of the symmetric relationship uses a relationship object to represent the relationship. Objects
   fulfilling the peer role are identified by the peerPointer attribute of the symmetric relationship managed object.";;

   RELATIONSHIP OBJECT symmetricRelationshipObject;

   ROLE peerRole RELATED-CLASSES "CCITT Rec. X.721 | ISO/IEC 10165-2":top
      REPRESENTED-BY RELATIONSHIP-OBJECT-USING-POINTER peerPointer;

   OPERATIONS MAPPING
      ESTABLISHestablishSymmetricRelationship
         MAPS-TO-OPERATION CREATE OF RELATIONSHIP OBJECT,
      TERMINATE     terminateSymmetricRelationship
         MAPS-TO-OPERATION DELETE OF RELATIONSHIP OBJECT,
      QUERY         querySymmetricRelationship
         MAPS-TO-OPERATION GET peerPointer OF RELATIONSHIP OBJECT;
REGISTERED AS {GRMExample.grmEx-RelationshipMapping x};

symmetricRelationshipObject MANAGED OBJECT CLASS
   DERIVED FROM genericRelationshipObject;
   CHARACTERIZED BY symmetricRelationshipPackage PACKAGE
      ATTRIBUTES peerPointer GET-REPLACE ADD-REMOVE;;;
REGISTERED AS {GRMExample.grmEx-Object x};

peerPointer ATTRIBUTE
   DERIVED FROM participantPointer;
REGISTERED AS {GRMExample.grmEx-Attribute x};
```

F.3 Dependency relationship example

The following example illustrates a relationship of dependency of one or more objects that assume a dependency role on a single object that assumes a parent role. The example illustrates mappings in terms of participant pointers, a relationship object, and naming.

The dependency relationship class might be useful to represent a directed acyclic graph by means of relationship specialization. In such a DAGDependency relationship class, the level of a dependent relative to its parent in the graph should be introduced and represented by the addition of an appropriate attribute. An invariant should be added stating that the value of the level attribute in a dependent must always be greater than that in its parents. The dependency relationship class might also be useful to represent a family relationship by the specialization of the person managed object class into three subclasses:

– parent;

– son; and

– daughter.

F.3.1 Dependency relationship class definition

```
dependency     RELATIONSHIP CLASS
    BEHAVIOUR    dependencyBehaviour;

    SUPPORTS
        ESTABLISHestablishDependency,
        TERMINATE      terminateDependency,
        QUERY          queryDependents,
        QUERY          queryParent;

    QUALIFIED-BY   timeOfEstablishment;

    ROLE       parentRole
            PERMITTED-ROLE-CARDINALITY-CONSTRAINT                 GRMExample.One
            REQUIRED-ROLE-CARDINALITY-CONSTRAINT                  GRMExample.One
            PERMITTED-RELATIONSHIP-CARDINALITY-CONSTRAINT         GRMExample.One
    REGISTERED AS {GRMExample.grmEx-Role x},

    ROLE       dependentRole
            PERMITTED-ROLE-CARDINALITY-CONSTRAINT                 GRMExample.OneToMax
            REQUIRED-ROLE-CARDINALITY-CONSTRAINT                  GRMExample.One
            PERMITTED-RELATIONSHIP-CARDINALITY-CONSTRAINT         GRMExample.One
            BIND-SUPPORT            bindDependent
            UNBIND-SUPPORT          unbindDependent
    REGISTERED AS {GRMExample.grmEx-Role x};
REGISTERED AS {GRMExample.grmEx-RelationshipClass x};
```

dependencyBehaviour BEHAVIOUR DEFINED AS "

INVARIANT: There exist two roles in this relationship class – parent role and dependent role. The existence of a participant in the dependent role implies the existence of at least one corresponding participant in the parent role. A managed object may not participate in both roles.

COMMENTS: An object instance fulfilling the dependent role may only participate in one instance of this dependency relationship, that is, the relationship cardinality is equal to one. An object instance able to fulfil the parent role may exist outside a dependency relationship an object fulfilling a dependent role shall not. The qualifying attribute, timeOfEstablishment, indicates the time, in UTC time format, of establishment of the relationship.

OPERATIONS:

ESTABLISH establishDependency

Signature: The class and identity of the proposed participant object in the dependent role to be created by the ESTABLISH operation; the class and identity of the proposed participant in the parent role.

Precondition: The proposed participant in the dependent role does not exist; the proposed participant in the parent role exists.

Postcondition: A new instance of the dependency relationship class exists; the participants in the parent role and the dependent role proposed in the ESTABLISH signature exist and are bound in the new instance of the dependency relationship class. The qualifying attribute, timeOfEstablishment, is set to the current value of UTC time.

BIND bindDependent

Signature: The class and identity of a participant in the parent role; the class and identity of the proposed participant to be created in the dependent role.

Precondition: The participant in the parent role exists and is bound into an instance of the dependency relationship class; the proposed participant in the dependent role does not exist.

Postcondition: The participant in the dependent role referenced in the BIND signature exists and is bound into the same dependency relationship as that in which the participant in the parent role referenced in the BIND signature is bound.

UNBIND unbindDependent

Signature: The class and identity of a participant in the parent role; the class and identity of the participant in a dependent role.

Precondition: The two participants identified in the UNBIND signature exist and are bound into the same instance of a dependency relationship; the dependency relationship exists; there exists at least one other participant in the dependent role bound into the relationship.

Postcondition: The participant in the dependent role referenced in the UNBIND signature does not exist; all other participants bound into the instance of dependency relationship class exist and remain bound in the instance of the dependency relationship class.

TERMINATE terminateDependency

Signature: The identity of a dependency relationship instance to be terminated.

Precondition: The instance of the dependency relationship class identified in the signature exists; only a single participant in dependent role is bound into the identified dependency relationship.

Postcondition: The instance of the dependency relationship class referenced in the signature does not exists; the participant that was in the parent role exists. The participant(s) in the dependent role do not exist. The value of the qualifying attribute, timeOfEstablishment, is undefined.";

```
person MANAGED OBJECT CLASS
    DERIVED FROM "CCITT Rec. X.721 | ISO/IEC 10165-2":top;
    CHARACTERIZED BY
        personPackage    PACKAGE
            ATTRIBUTES personName GET;;;
REGISTERED AS {GRMExample.grmEx-Object x};

personName ATTRIBUTE
    WITH ATTRIBUTE SYNTAX GRMExample.PersonName;
REGISTERED AS {GRMExample.grmEx-Attribute x};

timeOfEstablishment    ATTRIBUTE
    WITH ATTRIBUTE SYNTAX UTCTime;
REGISTERED AS {GRMExample.grmEx-Attribute x};
```

F.3.2 Dependency relationship class represented by means of conjugate pointers

```
dependencyAttributeRepresentation    RELATIONSHIP MAPPING
    RELATIONSHIP CLASS dependency;
    BEHAVIOUR    dependencyAttributeRepresentationBehaviour;

    ROLE    parentRole
        RELATED-CLASSES                aPerson
        REPRESENTED-BY ATTRIBUTE       parent
        QUALIFIES timeOfEstablishment,

    ROLE      dependentRole
        RELATED-CLASSES                bPerson
        REPRESENTED-BY ATTRIBUTE       dependents;
    OPERATIONS MAPPING
        ESTABLISHestablishDependency
            MAPS-TO-OPERATION CREATE OF dependentRole,
        TERMINATE    terminateDependency
            MAPS-TO-OPERATION DELETE OF dependentRole,
        BIND             bindDependent
            MAPS-TO-OPERATION CREATE OF dependentRole ,
        UNBIND       unbindDependent
            MAPS-TO-OPERATION DELETE OF dependentRole,
        QUERY        queryParent parentRole
            MAPS-TO-OPERATION GET parent OF dependentRole,
        QUERY       queryDependents dependentRole
            MAPS-TO-OPERATION GET dependents OF parentRole;
REGISTERED AS {GRMExample.grmEx-RelationshipMapping x};
```

dependencyAttributeRepresentationBehaviour BEHAVIOUR DEFINED AS "

This representation of the dependency relationship class uses conjugate participant pointers to represent an instance of the relationship; participant pointer consistency is to be maintained.

The relationship management operations ESTABLISH establishDependency and BIND bindDependent both map to a create of a participant in the dependent role: the distinction being that the relationship management operation ESTABLISH establishDependency is used when a participant is the first to fulfil the dependent role; the relationship management operation BIND bindDependent is used when there is at least one other participant in the dependent role at the time of binding. After creation of an object of class bPerson with the attribute, parent, identifying an object of class aPerson, the value of the attribute, dependents, in the object of class aPerson identifies the corresponding object of class bPerson.

Similarly, the relationship management operations TERMINATE terminateDependency and UNBIND unbindDependent both map to a delete of a participant in the dependentRole: the distinction being that the relationship management operation TERMINATE terminateDependency is used if there is only one participant fulfilling the dependentRole; the relationship management operation UNBIND unbindDependent is used if there is more than one participant fulfilling the dependentRole at time of deletion. Upon deletion of an object of class bPerson fulfilling the dependentRole, the value of the attribute, dependents, in the object of class aPerson object fulfilling the parentRole is modified by removing the identity of the corresponding object of class bPerson.

The QUERY queryDependents relationship management operation maps to a GET of the dependents attribute in the aPerson object fulfilling a parentRole; the QUERY queryParent operation maps to a GET of the parent attribute in the bPerson object fulfilling a dependentRole.

The creation of a bPerson managed object class (or bPerson subclass) results in the establishment of an instance of the dependency relationship with dependencyAttributeRepresentation RELATIONSHIP MAPPING when the value of the parent attribute in the object of class bPerson is set-by-create to an instance of a managed object of class aPerson and the value of the dependents attribute in the object of class aPerson is an empty set.

The creation of a bPerson (or bPerson subclass) managed object results in its being bound to an instance of the dependency relationship with dependencyAttributeRepresentation RELATIONSHIP MAPPING when the value of the parent attribute in the bPerson object is set-by-create to an instance of aPerson managed object class and the dependents attribute in the aPerson object is a non-empty set.

The deletion of a bPerson (or bPerson subclass) managed object results in its being unbound from an instance of the dependency relationship with dependencyAttributeRepresentation RELATIONSHIP MAPPING when the value of the dependents attribute in the aPerson object is not empty after the deletion and associated update of the dependents attribute.

The deletion of a bPerson (or bPerson subclass) managed object results in the termination of an instance of the dependency relationship with dependencyAttributeRepresentation RELATIONSHIP MAPPING when the value of the dependents attribute in the aPerson object is empty after the deletion and the associated update of the dependents attribute.";

```
aPerson    MANAGED OBJECT CLASS
    DERIVED FROM person;
    CHARACTERIZED BY    parentPackage    PACKAGE
        ATTRIBUTES      dependents GET,
                        timeOfEstablishment GET;;;
REGISTERED AS {GRMExample.grmEx-Object x};
```

```
bPerson   MANAGED OBJECT CLASS
    DERIVED FROM person;
    CHARACTERIZED BY
        dependentPackage   PACKAGE
            ATTRIBUTES        parent PERMITTED VALUES GRMExample.SingleValued GET;;;
REGISTERED AS {GRMExample.grmEx-Object x};

dependent ATTRIBUTE
    DERIVED FROM participantPointer;
REGISTERED AS {GRMExample.grmEx-Attribute x};

parent ATTRIBUTE
    DERIVED FROM participantPointer;
REGISTERED AS {GRMExample.grmEx-Attribute x};
```

F.3.3 Dependency relationship class represented by means of a relationship object

```
dependencyObjectRepresentation      RELATIONSHIP MAPPING
    RELATIONSHIP CLASS dependency;
    BEHAVIOUR dependencyObjectRepresentationBehaviour;
    RELATIONSHIP OBJECT dependencyRelationshipObject
        QUALIFIES     timeOfEstablishment;

    ROLE parentRole
            RELATED-CLASSES person
            REPRESENTED-BY RELATIONSHIP-OBJECT-USING-POINTER parent,

    ROLE dependentRole
            RELATED-CLASSES person
            REPRESENTED-BY RELATIONSHIP-OBJECT-USING-POINTER dependents;

    OPERATIONS MAPPING
        ESTABLISH establishDependency
            MAPS-TO-OPERATION CREATE OF RELATIONSHIP OBJECT,
        TERMINATE terminateDependency
            MAPS-TO-OPERATION DELETE OF RELATIONSHIP OBJECT,
        BIND bindDependent dependentRole
            MAPS-TO-OPERATION ADD dependents OF RELATIONSHIP OBJECT,
        UNBIND unbindDependent dependentRole
            MAPS-TO-OPERATION REMOVE dependents OF RELATIONSHIP OBJECT,
        QUERY queryDependents dependentRole
            MAPS-TO-OPERATION GET dependents OF RELATIONSHIP OBJECT,
        QUERY queryParents parentRole
            MAPS-TO-OPERATION GET parent OF RELATIONSHIP OBJECT;
REGISTERED AS {GRMExample.grmEx-RelationshipMapping x};

dependencyObjectRepresentationBehaviour BEHAVIOUR DEFINED AS"
```

This representation of the dependency relationship uses a relationship object to represent an instance of the relationship and to relate the participants. The relationship management operation ESTABLISH establishDependency maps to a CREATE of a dependencyRelationshipObject object and the relationship management operation TERMINATE terminateDependency maps to a DELETE of the dependencyRelationshipObject object. The relationship management operation BIND bindDependent maps to an ADD operation on the dependents attribute in a dependencyRelationshipObject object. The relationship management operation UNBIND unbindDependent maps to a REMOVE operation on the dependents attribute in the dependencyRelationshipObject object.

The creation of a DependencyRelationshipObject object results in the establishment of a dependency relationship with the dependencyObjectRepresentation RELATIONSHIP MAPPING. Because the parent role is not dynamic (i.e. BIND-SUPPORT and UNBIND-SUPPORT are not defined for the parent role), the parent attribute within the DependencyRelationshipObject must be set-by-create to the value of exactly one instance of person object fulfilling the parentRole; the value of the parent attribute cannot be changed during the lifetime of the dependency operation.

The addition of a value representing a person object to the dependents attribute of a dependencyRelationshipObject object results in the person object's being bound into the relationship corresponding to the dependencyRelationshipObject object in the dependentRole.

The removal of a value representing a person object from the dependents attribute of a dependencyRelationshipObject object, results in the person object's being unbound from the relationship corresponding to the dependencyRelationshipObject object.

The deletion of a dependencyRelationshipObject results in the termination of the corresponding dependency relationship with the dependencyObjectRepresentation RELATIONSHIP MAPPING.";

```
dependencyRelationshipObject          MANAGED OBJECT CLASS
    DERIVED FROM genericRelationshipObject;
    CHARACTERIZED BY
    dependencyRelationshipObjectPackage PACKAGE
        ATTRIBUTES
            dependents   GET-REPLACE ADD-REMOVE,
            parent       GET,
            timeOfEstablishment GET;
    REGISTERED AS {GRMExample.grmEx-Package x};;
REGISTERED AS {GRMExample.grmEx-Object x};
```

F.3.4 Dependency relationship represented by means of naming

```
dependencyNamingRepresentation      RELATIONSHIP MAPPING
    RELATIONSHIP CLASS dependency;
    BEHAVIOUR dependencyNamingRepresentationBehaviour ;

    ROLE parentRole
            RELATED-CLASSES cPerson
            REPRESENTED-BY NAMING aNameBinding USING SUPERIOR
            QUALIFIES    timeOfEstablishment,

    ROLE dependentRole
            RELATED-CLASSES person
            REPRESENTED-BY NAMING aNameBinding USING SUBORDINATE;

    OPERATIONS MAPPING
        ESTABLISH establishDependency
            MAPS-TO-OPERATION CREATE OF dependentRole,
        BIND bindDependent dependentRole
            MAPS-TO-OPERATION CREATE OF dependentRole,
        UNBIND unbindDependent dependentRole
            MAPS-TO-OPERATION DELETE OF dependentRole,
        TERMINATE terminateDependency
            MAPS-TO-OPERATION DELETE OF dependentRole,
        QUERY queryDependents dependentRole MAPS-TO-OPERATION GET
                "CCITT Rec. X.721 | ISO/IEC 10165-2":nameBinding OF dependentRole,
        QUERY queryParent parentRole MAPS-TO-OPERATION GET
                "CCITT Rec. X.721 | ISO/IEC 10165-2":nameBinding OF dependentRole;
REGISTERED AS {GRMExample.grmEx-RelationshipMapping x};

dependencyNamingRepresentationBehaviour BEHAVIOUR DEFINED AS "
```

This representation of the dependency relationship uses naming to represent an instance of the relationship.

The relationship management operations ESTABLISH establishDependency and BIND bindDependent both map to a create of a person (or person subclass) object participant in the dependentRole using a name binding with a cPerson (or cPerson subclass) object as the superior object in the parentRole. The distinction is that: the relationship management operation ESTABLISH establishDependency is used when the proposed participant in the dependent role would be the first object in the role; the relationship management operation BIND bindDependent is used when there is at least one other participant in the dependent role at the time of creation.

Similarly, the relationship management operations TERMINATE terminateDependency and UNBIND unbindDependent both map to a delete of a participant in the dependent role, the distinction being that: the relationship management operation TERMINATE terminateDependency is used if the participant is the only one fulfilling the dependentRole and the relationship management operations UNBIND unbindDependent is used if at least one other participant remains fulfilling the dependent role after deletion.

The QUERY queryDependents relationship management operation maps to a scoped get of the nameBinding attribute with a scope level of one on the person object in the parent role to determine the contained person objects that have the value of their name binding attribute equal to aNameBinding; such objects are fulfilling the dependents role.

The QUERY queryParent relationship management operation maps to a get of the nameBinding attribute of the subordinate object to determine that the value of its name binding attribute is equal to aNameBinding; subsequent analysis of the RDN of the subordinate object name will indicate the parent object.

The creation of a person (or person subclass) managed object as a subordinate to a cPerson (or cPerson subclass) object with aNameBinding name binding results in the establishment of an instance of the dependency relationship with dependencyNamingRepresentation RELATIONSHIP MAPPING if there are no other subordinates with aNameBinding name binding.

The creation of a person (or person subclass) managed object as a subordinate of a cPerson (or cPerson subclass) object with aNameBinding name binding results in the binding of the created object into a dependency relationship with the dependencyNamingRepresentation RELATIONSHIP MAPPING if there is at least one other subordinate with aNameBinding name binding.

The deletion of a person (or person subclass) managed object bound in the dependent role of a dependency relationship with the dependencyNamingRepresentation RELATIONSHIP MAPPING, results in the unbinding of the deleted object from the dependency relationship when at least one other dependents with aNameBinding will exist after the deletion.

The deletion of a person (or person subclass) managed object bound in the dependent role of a dependency relationship with the dependencyNamingRepresentation RELATIONSHIP MAPPING, results in the termination of the dependency relationship when there will exist no other dependents with aNameBinding after the deletion.";

```
aNameBinding NAME BINDING
    SUBORDINATE OBJECT CLASS person AND SUBCLASSES;
    NAMED BY SUPERIOR OBJECT CLASS cPerson AND SUBCLASSES;
        WITH ATTRIBUTE personName;
        CREATE;
        DELETE;
REGISTERED AS {GRMExample.grmEx-NameBinding x};
```

```
cPerson    MANAGED OBJECT CLASS
    DERIVED FROM person;
    CHARACTERIZED BY
        timePackage    PACKAGE
            ATTRIBUTES timeOfEstablishment GET;;;
REGISTERED AS {GRMExample.grmEx-Object x};
```

F.4 General composition relationship example

This example illustrates the use of the relationship class template to define a generic composition relationship between a single object in a composite role and one or more objects in a component role and how the template might be refined. Such a relationship might be useful for modelling an assembly/sub-assembly relationship.

```
generalCompositionRelationship    RELATIONSHIP CLASS
    BEHAVIOUR generalCompositionRelationshipBehaviour;

    SUPPORTS
        ESTABLISH establishGeneralComposition,
        TERMINATE terminateGeneralComposition;

    ROLE    compositeRole
            PERMITTED-ROLE-CARDINALITY-CONSTRAINT    GRMExample.OneToOne
            REQUIRED-ROLE-CARDINALITY-CONSTRAINT     GRMExample.OneToOne
    REGISTERED AS {GRMExample.grmEx-Role x},

    ROLE    componentRole
            PERMITTED-ROLE-CARDINALITY-CONSTRAINT    GRMExample.OneToMax
            REQUIRED-ROLE-CARDINALITY-CONSTRAINT     GRMExample.OneToOne
            BIND-SUPPORT bindComponent
            UNBIND-SUPPORT unbindComponent
    REGISTERED AS {GRMExample.grmEx-Role x};
REGISTERED AS {GRMExample.grmEx-RelationshipClass x};
```

generalCompositionRelationshipBehaviour BEHAVIOUR DEFINED AS "

INVARIANT: The existence of an instance of this relationship class implies the existence of exactly one participant in the composite role and one or more participants in the component role. At least one property of the composite participant is such that it depends upon properties of the components. At least the identity of the composite participant is such that it is independent of the existence or properties of the components; that is, creating, updating, or deleting any component does not change the identity of the composite.

OPERATIONS:

ESTABLISH establishGeneralComposition

Signature: The class and identity of the proposed participant in the composite role and the class and identity of the proposed participant(s) in the component role to be bound in an instance of the generalCompositionRelationship.

Precondition: The proposed participants are not already bound in the same instance of the generalCompositionRelationship class or a subclass thereof.

Postcondition: An instance of the generalCompositionRelationship class exists; the participants referenced in the signature are bound into this instance of the generalCompositionRelationship class.

BIND bindComponent

Signature: The class and identity of a proposed participant in the component role; the identity of a generalCompositionRelationship.

Precondition: The referenced instance of the generalCompositionRelationship class exists; the proposed participant in the component role is not bound into this instance of generalCompositionRelationship class; there exists at least one other participant in the component role bound into this instance of the generalCompositionRelationship class.

Postcondition: The participant in the component role referenced in the signature exists and is bound in this instance of the generalCompositionRelationship class.

UNBIND unbindComponent

Signature: The class and identity of a participant in the component role; the identity of a generalCompositionRelationship.

Precondition: The instance of the generalCompositionRelationship class referenced in the signature exists; the participant in the component role referenced in the signature is bound into the referenced instance of generalCompositionRelationship class; there exists at least one other participant in the component role bound into the referenced instance of the generalCompositionRelationship class.

Postcondition: The referenced participant in the component role exists but is not bound into the referenced instance of the generalCompositionRelationship class; the referenced instance of the generalCompositionRelationship class exists.

TERMINATE terminateGeneralComposition

Signature: The identity of a generalCompositionRelationship instance.

Precondition: The referenced instance of the generalCompositionRelationship class exists.

Postcondition: The referenced instance of the generalCompositionRelationship class does not exist; the participants in the composite role and in the component role that were bound into the relationship exist.";

F.4.1 Subclass of general composition relationship

subclassedCompositionRelationship RELATIONSHIP CLASS
DERIVED FROM generalCompositionRelationship;
BEHAVIOUR subclassedCompositionRelationshipBehaviour
BEHAVIOUR DEFINED AS"

This relationship class refines the required role cardinality of the component role of the generalCompositionRelationship class to be the range 1 to 5; all other characteristics of this relationship class are inherited from the generalCompositionRelationship class.";;

ROLE componentRole
REQUIRED-ROLE-CARDINALITY-CONSTRAINT GRMExample.OneToFive;
REGISTERED AS {GRMExample.grmEx-Object x};

F.5 Access control domain example

accessControlDomain RELATIONSHIP CLASS
BEHAVIOUR accessControlDomainBehaviour BEHAVIOUR DEFINED AS"

This relationship class binds managed objects which are subject to access control (memberObjectRole) to managed objects representing the access enforcement function (aefRole) and access decision function (adfRole) respectively.";;

```
    SUPPORTS QUERY queryAccessControlDomain;

    ROLE     memberObjectRole
             REQUIRED-ROLE-CARDINALITY-CONSTRAINT GRMExample.OneToTwo
             BIND-SUPPORT bindMember
             UNBIND-SUPPORT unbindMember
    REGISTERED AS {GRMExample.grmEx-Role memberObjectRoleArc(1) },

    ROLE     aefRole
             COMPATIBLE-WITH "ITU-T Rec. X.741 | ISO/IEC 10164-9":notificationEmitter
             PERMITTED-ROLE-CARDINALITY-CONSTRAINT     GRMExample.OneToOne
             REQUIRED-ROLE-CARDINALITY-CONSTRAINT      GRMExample.OneToOne
    REGISTERED AS {GRMExample.grmEx-Role aefRoleArc(2) },

    ROLE     adfRole
             COMPATIBLE-WITH "ITU-T Rec. X.741 | ISO/IEC 10164-9":accessControlRules
             PERMITTED-ROLE-CARDINALITY-CONSTRAINT     GRMExample.OneToOne
             REQUIRED-ROLE-CARDINALITY-CONSTRAINT      GRMExample.OneToOne
    REGISTERED AS {GRMExample.grmEx-Role adfRoleArc(3) };
REGISTERED AS {GRMExample.grmEx-RelationshipClass accessControlDomainArc(1) };
```

F.5.1 Access control domain relationship represented by attributes and naming

```
simpleAccessControlDomain RELATIONSHIP MAPPING
    RELATIONSHIP CLASS accessControlDomain;
    BEHAVIOUR simpleAccessControlDomainBehaviour BEHAVIOUR DEFINED AS"
```

In this mapping of the accessControlDomain managed relationship class, the accessControlDomainObject class (a subclass of the accessControlRules class) participates in the adfRole and the notificationEmitter class participates in the aefRole; any managed object may participate in the memberObjectRole. The memberObjectAttribute in the accessControlDomainObject identifies the participants in the memberObjectRole and the notificationEmitter-accessControlRules name binding contains the aef participant within the adf participant.

The QUERY queryAccessControlDomain relationship management operation maps to two operations, namely :

(a) a GET of the memberObjectAttribute of the object fulfilling the adfRole; followed by

(b) a scoped GET of the nameBinding attribute with a scope level of one on the object fulfilling the adf role to determine the contained objects that have the value of their name binding attribute equal to "ITU-T Rec. X.741 | ISO/IEC 10164-9":notificationEmitter-accessControlRules.";;

```
    ROLE       memberObjectRole RELATED-CLASSES    "ITU-T Rec. X721 | ISO/IEC 10165-2":top,
             REPRESENTED-BY ATTRIBUTE memberObjectAttribute ;

    ROLE       aefRole
            RELATED-CLASSES "ITU-T Rec. X.741 | ISO/IEC 10164-9":notificationEmitter
               REPRESENTED-BY NAMING
               "ITU-T Rec. X.741 | ISO/IEC 10164-9":notificationEmitter-accessControlRules USING
               SUBORDINATE,

    ROLE       adfRole
             RELATED-CLASSES accessControlDomainObject
               REPRESENTED-BY NAMING
               "ITU-T Rec. X.741 | ISO/IEC 10164-9":notificationEmitter-accessControlRules USING
               SUPERIOR,

    OPERATIONS MAPPING
       BIND bindMember memberObjectRole
          MAPS-TO-OPERATION ADD memberObjectAttribute OF adfRole,
       UNBIND unbindMember memberObjectRole
          MAPS-TO-OPERATION REMOVE memberObjectAttribute OF adfRole,
       QUERY queryAccessControlDomain memberObjectRole
          MAPS-TO-OPERATION GET memberObjectAttribute OF adfRole
          MAPS-TO-OPERATION GET
             "CCITT Rec. X.721 | ISO/IEC 10165-2":nameBinding OF adfRole;
REGISTERED AS {GRMExample.grmEx-RelationshipMapping simpleAccesControlDomainArc(1) };
```

```
accessControlDomainObject MANAGED OBJECT CLASS
    DERIVED FROM "ITU-T Rec. X.741 | ISO/IEC 10164-9":accessControlRules;
        CHARACTERIZED BY accessControlDomainPackage PACKAGE
            BEHAVIOUR accessControlDomainBehaviour
                BEHAVIOUR DEFINED AS "
```

Membership of the access control domain is identified and modified by operations upon the memberObjectAttribute.";;

```
            ATTRIBUTES memberObjectAttribute GET-REPLACE ADD-REMOVE;;;
REGISTERED AS {GRMExample.grmEx-Object accessControlDomainObjectArc(1) };
```

F.5.2 Access control domain relationship representation using a relationship object

```
coordinatedAccessControlDomain RELATIONSHIP MAPPING
    RELATIONSHIP CLASS accessControlDomain;
    BEHAVIOUR coordinatedAccessControlDomainBehaviour
        BEHAVIOUR DEFINED AS"
```

In this mapping of the accessControlDomain managed relationship class, the accessControlRules class participates in the adfRole and the notificationEmitter class participates in the aefRole; any managed object may participate in the memberObjectRole. The relationship is represented by the accessControlDomainCoordinator, a subclass of the genericRelationshipObject, using the memberObjectAttribute, aefAttribute, and adfAttribute attributes.

The QUERY queryAccessControlDomain relationship management operation maps to three GET operations on the relationship object, namely:

(a) a GET of the memberObjectAttribute;

(b) a GET of the aefAttribute; and

(c) a GET of the adfAttribute.";;

```
RELATIONSHIP OBJECT accessControlDomainCoordinator;

    ROLE        memberObjectRole
            RELATED-CLASSES "CCITT Rec. X.721 | ISO/IEC 10165-2":top
            REPRESENTED-BY RELATIONSHIP-OBJECT-USING-POINTER memberObjectAttribute,

    ROLE        aefRole
            RELATED-CLASSES
            "ITU-T Rec. X.741 | ISO/IEC 10164-9":notificationEmitter
            REPRESENTED-BY RELATIONSHIP-OBJECT-USING-POINTER aefAttribute,

    ROLE        adfRole
            RELATED-CLASSES "ITU-T Rec. X.741 | ISO/IEC 10164-9":accessControlRules
            REPRESENTED-BY RELATIONSHIP-OBJECT-USING-POINTER adfAttribute;

    OPERATIONS MAPPING
        BIND bindMember
            MAPS-TO-OPERATION ADD memberObjectAttribute OF RELATIONSHIP OBJECT,
        UNBIND unbindMember
            MAPS-TO-OPERATION REMOVE memberObjectAttribute OF RELATIONSHIP OBJECT,
        QUERY queryAccessControlDomain
            MAPS-TO-OPERATION GET memberObjectAttribute OF RELATIONSHIP OBJECT
            MAPS-TO-OPERATION GET aefAttribute OF RELATIONSHIP OBJECT
            MAPS-TO-OPERATION GET adfAttribute OF RELATIONSHIP OBJECT;
REGISTERED AS
    {GRMExample.grmEx-RelationshipMapping coordinatedAccessControlDomainArc(2)};

accessControlDomainCoordinator MANAGED OBJECT CLASS
    DERIVED FROM genericRelationshipObject;
    CHARACTERIZED BY accessControlDomainCoordinatorPackage PACKAGE

        ATTRIBUTES
            memberObjectAttribute
                ATTRIBUTE DERIVED FROM participantPointer;
                REGISTERED AS { GRMExample.grmEx-Attribute memberObjectAttributeArc(1) };
                    GET-REPLACE ADD-REMOVE,
            aefAttribute
                ATTRIBUTE DERIVED FROM participantPointer;
                REGISTERED AS { GRMExample.grmEx-Attribute aefAttributeArc(1) }; GET,
            adfAttribute
                ATTRIBUTE DERIVED FROM participantPointer;
                REGISTERED AS { GRMDExample.grmEx-Attribute adfAttributeArc(1) }; GET;
REGISTERED AS {GRMExample.grmEx-Object accessControlDomainCoordinatorArc(1)};
```

Annex G

Commentary

(This annex does not form an integral part of this Recommendation | International Standard)

G.1 Introduction

The following commentary has been developed from the list of issues that was maintained over the development of the standard.

G.2 Dependency between managed objects in a managed relationship

- **Issue:** The essence of the GRM is that managed objects participating in a managed relationship affect one another; this is expressed as an invariant over the properties of the participants. How should this invariant be specified?
- **Commentary:** Previous drafts of the GRM have attempted to single out, and provide notational support for, various types of invariants such as attribute-value constraints or existence dependency. Recognizing that behaviour templates can potentially express all types of invariants, the inclusion of notational support for particular invariant types has not retained consistent NB support. Hence all invariants are expressed in terms of managed relationship behaviour. The invariant is specified in terms of properties of the managed relationship (roles, relationship management operations, etc.). The relationship mapping template may provide a mapping of the invariant in terms of the representation method (participating managed objects, relationship objects, participant pointers, etc.).

Invariants are, by definition, requirements and conformant implementations must meet these requirements. The GRM prescribes no general mechanism for meeting these requirements though the relationship mapping template does provide the tools for managed relationship specifiers to prescribe such mechanisms in particular cases of relationship mapping.

G.3 Consistency of views

- **Issue:** A representation method may specify management information (e.g. participant pointers, relationship objects) that is related solely to the representation method. How is this information to be kept consistent?
- **Commentary:** It is a fundamental concept of the GRM that the semantics of the managed relationship be consistently expressed in the elements of an implementation; in other words the relationship drives the representation, not the other way round. Thus, if a relationship mapping chooses to represent the semantics of managed object participation as conjugate pointers in the participant objects, then an implementation must ensure that the pointers are always consistent. Furthermore, if a relationship mapping chooses to represent the BIND operation as an attribute-based addOperation on one of a pair of conjugate participant pointers, an implementation is required to adjust the other pointer to maintain consistency. The GRM only specifies requirements for consistency of information; it does not specify mechanisms for maintaining consistency either within a single managed system or across multiple managed systems.

G.4 Expression of relationship management operations and notifications

- **Issue:** How are relationship management operations and notifications expressed and how are they mapped to systems management operations?
- **Commentary:** Relationship management operations and notifications are expressed in terms of a number of prototypical operations and a notification which are subsequently mapped to systems management operations and notifications. The final text gives full details and examples of the technique.

G.5 Generic management

– **Issue:** Can mechanisms be defined to permit the management of a broad range of managed relationship types?

– **Commentary:** A companion standard, the General Relationship Management Function, was raised in parallel to the GRM. However, given the broad range of relationship types that could be defined, subsequent investigation indicated that generic management tools for managing relationships across the board would be of limited use. It was thus agreed to provide managed-relationship specifiers tools to specify such mechanisms on a relationship-by-relationship basis. The GRM defines a template for mapping managed relationship operations and notifications and defines generic management information.

However, managed relationship subclasses are consistent with their superclasses and, in this sense, generic management is provided within an inheritance hierarchy.

G.6 Relationship awareness

– **Issue:** How does a managed object “know” that it is in a managed relationship?

– **Commentary:** An anthropomorphic view of a relationship is not helpful. A managed object must fulfil the requirements of the role as modelled by the managed relationship. In the final analysis, an implementation must ensure that the semantics of the relationship are preserved and that implementations of managed objects fulfil the requirements of the role.

G.7 Role specification

– **Issue:** Should a role be specified out of line?

– **Commentary:** Initially roles were seen as independent, re-usable specifications. Subsequent reflection has indicated that roles are intimately connected with their managed relationship and out-of-line specification is of limited value.

G.8 Re-use of specifications

– **Issue:** Re-use of specifications is an important facet of OSI systems management; how is it implemented in the GRM?

– **Commentary:** Subclasses of managed relationship classes are *consistent* with their supertypes in that an instance of a subclass can be substituted for an instance of a superclass without affect the operation of the managing system. Subclasses are, in fact, subtypes within the Open Distributed Processing definition of the term. Thus the inheritance and specialization tools provide a mechanism for re-use of specifications.

G.9 AND SUBCLASSES

– **Issue:** The **AND SUBCLASSES** clause was not carried over from **GDMO name-binding template** to the **role-mapping-specification** supporting production of the **RELATIONSHIP MAPPING** template.

– **Commentary:** The ability of a subclass to support of a role is regarded as a fundamental property of a managed object and should thus be unconditionally inherited.

G.10 Relationship between relationships

– **Issue:** How can relationships between relationships be modelled?

– **Commentary:** Whilst the GRM models relationships between managed objects, if relationships are represented by relationship objects, then there is no reason why the GRM cannot model relationships between relationships. The GRM provides no particular support for this, but any additional semantics could be specified in the BEHAVIOUR template.

G.11 Naming Scope of relationship objects

- **Issue:** What should be the scope for the naming of relationship objects?
- **Commentary:** There was discussion regarding the naming of all relationship objects in a managed system within the scope of a single object of a particular class – often referred to as the *anchor object* class – particularly with a view to being able to discover all relationship objects in a managed system by means of CMIS scoping. It was concluded that, since existing management standards regard naming structure as a local matter, it would be inconsistent for the GRM to prescribe a particular structure.

G.12 Allowable representation methods

- **Issue:** Can representation methods represent all types of relationships?
- **Commentary:** No; some representation methods are inherently restricted in the type of the relationships they can represent. Table G.1 shows the types of relationships that can be represented by the various methods.

Table G.1 – Allowable representation methods

	Relationship Cardinality = 1			Relationship Cardinality > 1		
	Role Cardinality			Role Cardinality		
Representation Method	1:n	n:m	n:m:p	1:n	n:m	n:m:p
Naming	Yes	No	No	No	No	No
Participant Pointers	Yes	Yes	Yes	No	No	No
Relationship Object	Yes	Yes	Yes	Yes	Yes	Yes
Systems Management Operations	Yes	Yes	Yes	Yes	Yes	Yes

第１版作成協力者（１９９８年1月２９日時点）

第四部門委員会

部門委員長	齋藤　幸男	**日本電信電話（株）**
副部門委員長	蓮池　和夫	**国際電信電話（株）**
副部門委員長	渡辺　芳明	**日本アイ・ビー・エム（株）**
委員	川野　強	**住友電気工業（株）**
〃	堀　潔洋	**（株）東芝**
〃	斉藤　裕一	**（株）リコー**
〃	森　仁	**松下電器産業（株）**
〃	青島　一倫	WG4-1**委員長・日本電気（株）**
〃	伊藤　均	WG4-1**副委員長・富士通（株）**
〃	小池　淳	WG4-2**委員長・国際電信電話（株）**
〃	細田　隆明	WG4-2副委員長**・沖電気工業（株）**
〃	菅野　政孝	WG4-3委員長**・ｴﾇ･ﾃｨ･ﾃｨ･ﾃﾞｰﾀ通信（株）**
〃	青山　敬	WG4-3副委員長**・（株）日立製作所**
〃	近藤　貴士	WG4-4委員長**・シャープ（株）**
〃	斉藤　隆一	WG4-4副委員長**・日本電信電話（株）**
〃	千田　昇一	WG4-Obj委員長**・日本電信電話（株）**
〃	岩本　裕司	WG4-Obj副委員長**・日本アイ・ビー・エム（株）**

第四部門委員会第一専門委員会

専門委員長	青島　一倫	**日本電気（株）**
副専門委員長	伊藤　均	**富士通（株）**
委員	青木　俊行	**国際電信電話（株）**
〃	村椿　喜一	**日本テレコム（株）**
〃	田場　洋哉	**日本電信電話（株）**
〃	横山　和広	**（株）インテック**
〃	森福　茂	**沖電気工業（株）**
〃	中川　和三	**住友電気工業（株）**
〃	大神　和正	**（株）東芝**
〃	吉川　康司	**日本電気（株）**
〃	若島　数英	**日本無線（株）**
〃	金子　勲	**日本ユニシス（株）**
〃	荻野　啓	**（株）日立製作所**
〃	坂田　雅岳	**富士通（株）**
〃	妹尾　尚一郎	**三菱電機（株）**
〃	岩波　道昭	**明星電気（株）**
〃	山田　俊明	**（株）リコー**
〃	川口　直幸	**日本情報通信ｺﾝｻﾙﾃｨﾝｸﾞ（株）**
〃	中谷　辰五郎	**（株）エヌ・ケー・エクサ**
特別専門委員	寺本　昌弘	SWG2リーダ**・日本電信電話（株）**
特別専門委員	鈴木　三知男	SWG5リーダ**・（株）日立製作所**

事務局	近藤　康生	TTC第四技術部

ＪＴ－Ｘ７２５検討グループ（ＳＷＧ５）

リーダ＊１	鈴木　三知男	**（株）日立製作所**
サブリーダ＊1	小林　秀承	**日本電信電話（株）**
サブリーダ＊1	須藤　誠	**富士通（株）**
特別専門委員	水本　政弘	**国際電信電話（株）**
〃	曽根　進	**国際電信電話（株）**
委員	村椿　喜一	**日本テレコム（株）**
特別専門委員	羽室　大介	**日本電信電話（株）**
〃	片山　善博	**沖電気工業（株）**
〃	三坂　亮	**沖電気工業（株）**
〃	片岡　好広	**（株）東芝**
〃	川田　裕哉	**日本電気（株）**
〃	小林　勝	**日本ユニシス（株）**
〃	安藤　貞人	**（株）日立製作所**
〃	小林　修	**富士通（株）**
〃	堀内　栄一	**三菱電機（株）**
〃	奥村　公治	**（株）エヌ・ケー・エクサ**
委員	中谷　辰五郎	**（株）エヌ・ケー・エクサ**

＊１　特別専門委員