UNION INTERNATIONALE DES TÉLÉCOMMUNICATIONS

UIT-T Q.816.1

SECTEUR DE LA NORMALISATION
DES TÉLÉCOMMUNICATIONS
DE L'UIT

(08/2001)

SÉRIE Q: COMMUTATION ET SIGNALISATION

Interface Q3

Services RGT à architecture CORBA: extensions pour la prise en charge des interfaces à granularité grossière

Recommandation UIT-T Q.816.1

RECOMMANDATIONS UIT-T DE LA SÉRIE Q

COMMUTATION ET SIGNALISATION

SIGNALISATION DANS LE SERVICE MANUEL INTERNATIONAL	Q.1–Q.3
EXPLOITATION INTERNATIONALE AUTOMATIQUE ET SEMI-AUTOMATIQUE	Q.4–Q.59
FONCTIONS ET FLUX D'INFORMATION DES SERVICES DU RNIS	Q.60–Q.99
CLAUSES APPLICABLES AUX SYSTÈMES NORMALISÉS DE L'UIT-T	Q.100–Q.119
SPÉCIFICATIONS DES SYSTÈMES DE SIGNALISATION N° 4 ET N° 5	Q.120–Q.249
SPÉCIFICATIONS DU SYSTÈME DE SIGNALISATION N° 6	Q.250–Q.309
SPÉCIFICATIONS DU SYSTÈME DE SIGNALISATION R1	Q.310–Q.399
SPÉCIFICATIONS DU SYSTÈME DE SIGNALISATION R2	Q.400–Q.499
COMMUTATEURS NUMÉRIQUES	Q.500–Q.599
INTERFONCTIONNEMENT DES SYSTÈMES DE SIGNALISATION	Q.600–Q.699
SPÉCIFICATIONS DU SYSTÈME DE SIGNALISATION N° 7	Q.700–Q.799
Généralités	Q.700
Sous-système transport de messages	Q.701–Q.709
Sous-système commande des connexions sémaphores	Q.711–Q.719
Sous-système utilisateur de téléphonie	Q.720–Q.729
Services complémentaires du RNIS	Q.730–Q.739
Sous-système utilisateur de données	Q.740–Q.749
Gestion du système de signalisation n° 7	Q.750–Q.759
Sous-système utilisateur du RNIS	Q.760–Q.769
Sous-système application de gestion des transactions	Q.770–Q.779
Spécification des tests	Q.780–Q.799
INTERFACE Q3	**Q.800–Q.849**
SYSTÈME DE SIGNALISATION D'ABONNÉ NUMÉRIQUE N° 1	Q.850–Q.999
RÉSEAUX MOBILES TERRESTRES PUBLICS	Q.1000–Q.1099
INTERFONCTIONNEMENT AVEC LES SYSTÈMES MOBILES À SATELLITES	Q.1100–Q.1199
RÉSEAU INTELLIGENT	Q.1200–Q.1699
PRESCRIPTIONS ET PROTOCOLES DE SIGNALISATION POUR LES IMT-2000	Q.1700–Q.1799
RNIS À LARGE BANDE	Q.2000–Q.2999

Pour plus de détails, voir la Liste des Recommandations de l'UIT-T.

Recommandation UIT-T Q.816.1

Services RGT à architecture CORBA: extensions pour la prise en charge des interfaces à granularité grossière

Résumé

La présente Recommandation définit des extensions de l'ensemble des services RGT à architecture CORBA, qui sont nécessaires afin de prendre en charge les interfaces à granularité grossière. Elle spécifie la façon dont les services communs aux objets de l'architecture CORBA sont utilisés pour prendre en charge les interfaces à granularité grossière et définit des extensions aux services supports qui sont propres au RGT et qui sont définis dans la Rec. UIT-T Q.816. Un module d'architecture CORBA est présenté afin de définir en langage IDL les interfaces avec les nouveaux services supports propres au RGT.

Source

La Recommandation Q.816.1 de l'UIT-T, élaborée par la Commission d'études 4 (2001-2004) de l'UIT-T, a été approuvée le 13 août 2001 selon la procédure définie dans la Résolution 1 de l'AMNT.

Mots clés

Architecture de courtier commun de requête d'objets (CORBA), granularité grossière, interfaces RGT, langage de définition d'interface (IDL), objets gérés, services CORBA, traitement réparti.

AVANT-PROPOS

L'UIT (Union internationale des télécommunications) est une institution spécialisée des Nations Unies dans le domaine des télécommunications. L'UIT-T (Secteur de la normalisation des télécommunications) est un organe permanent de l'UIT. Il est chargé de l'étude des questions techniques, d'exploitation et de tarification, et émet à ce sujet des Recommandations en vue de la normalisation des télécommunications à l'échelle mondiale.

L'Assemblée mondiale de normalisation des télécommunications (AMNT), qui se réunit tous les quatre ans, détermine les thèmes d'étude à traiter par les Commissions d'études de l'UIT-T, lesquelles élaborent en retour des Recommandations sur ces thèmes.

L'approbation des Recommandations par les Membres de l'UIT-T s'effectue selon la procédure définie dans la Résolution 1 de l'AMNT.

Dans certains secteurs des technologies de l'information qui correspondent à la sphère de compétence de l'UIT-T, les normes nécessaires se préparent en collaboration avec l'ISO et la CEI.

NOTE

Dans la présente Recommandation, l'expression "Administration" est utilisée pour désigner de façon abrégée aussi bien une administration de télécommunications qu'une exploitation reconnue.

DROITS DE PROPRIÉTÉ INTELLECTUELLE

L'UIT attire l'attention sur la possibilité que l'application ou la mise en œuvre de la présente Recommandation puisse donner lieu à l'utilisation d'un droit de propriété intellectuelle. L'UIT ne prend pas position en ce qui concerne l'existence, la validité ou l'applicabilité des droits de propriété intellectuelle, qu'ils soient revendiqués par un Membre de l'UIT ou par une tierce partie étrangère à la procédure d'élaboration des Recommandations.

A la date d'approbation de la présente Recommandation, l'UIT n'avait pas été avisée de l'existence d'une propriété intellectuelle protégée par des brevets à acquérir pour mettre en œuvre la présente Recommandation. Toutefois, comme il ne s'agit peut-être pas de renseignements les plus récents, il est vivement recommandé aux responsables de la mise en œuvre de consulter la base de données des brevets du TSB.

TABLE DES MATIÈRES

Recommandation UIT-T Q.816.1

Services RGT à architecture CORBA: extensions pour la prise en charge des interfaces à granularité grossière

1 Domaine d'application

L'architecture RGT qui est définie dans la Rec. UIT-T M.3010 (2000) présente les concepts découlant du traitement réparti et inclut l'utilisation de multiples protocoles de gestion. Les Rec. UIT-T Q.816 et X.780 définissent ensuite, à l'intérieur de cette architecture, un cadre d'application de l'architecture de courtier commun de requête d'objets (CORBA, *common object request broker architecture*) en tant qu'un des protocoles de gestion RGT.

La présente Recommandation, de concert avec la Rec. UIT-T X.780.1, ajoute des spécifications à ce cadre afin de lui permettre de prendre en charge un mode d'interaction entre systèmes gérants et systèmes gérés légèrement différent de celui qui a été spécifié dans le cadre original. Ce mode d'interaction présente certains avantages, le principal étant qu'il dispense un système gérant d'avoir à extraire une adresse logicielle de type objet pour chaque ressource gérable à laquelle ce système souhaite accéder. Ces adresses logicielles pourraient se chiffrer par millions dans de grands systèmes. Ce mode d'interaction modifie également un peu la façon dont le logiciel est structuré dans les systèmes gérés, d'une façon que certains fournisseurs de systèmes peuvent préférer.

Le domaine d'application de la présente Recommandation est celui du cadre RGT original de l'architecture CORBA. Ce cadre et ces extensions s'appliquent à toutes les interfaces du RGT où l'architecture CORBA peut être utilisée. L'on anticipe cependant que toutes les interfaces RGT n'auront pas besoin de toutes les capacités et de tous les services ici définis. Il en découle que le cadre peut être utilisé pour des interfaces entre systèmes de gestion à tous les niveaux d'abstraction (administration interdomaniale et intradomaniale) ainsi qu'entre systèmes de gestion et éléments de réseau.

1.1 Objet

L'objet de la présente Recommandation est d'élargir le cadre RGT de l'architecture CORBA afin de permettre son utilisation dans une plus large gamme d'applications. Ces extensions permettent un mode d'interaction légèrement différent entre les systèmes gérants et gérés, qui peut être préféré dans de nombreuses situations. La présente Recommandation est donc destinée à être utilisée par divers groupes spécifiant des interfaces de gestion de réseau.

1.2 Application

L'approche choisie dans les Recommandations sur le cadre RGT de l'architecture CORBA consiste à modéliser les ressources gérables du réseau par des objets logiciels accessibles au moyen de l'architecture CORBA. Des modèles informationnels, rédigés en langage de définition d'interface CORBA (IDL, *interface definition language*), décrivent les interfaces avec ces objets.

L'architecture CORBA assure la transparence spatiale, qui permet à chaque objet logiciel d'interagir avec un autre objet logiciel quel que soit son emplacement. C'est ce qui est appelé en architecture CORBA référence d'objet interexploitable (IOR, *interoperable object reference)* qui permet d'accéder à un objet logiciel.

Le cadre RGT original de l'architecture CORBA modélise chaque ressource gérable par un objet CORBA indépendant, possédant chacun sa propre référence IOR unique. Cette approche permet de manière flexible à chaque objet de résider à n'importe quel endroit. Elle nécessite cependant que les systèmes gérants disposent d'une référence IOR pour chaque objet auquel ils souhaitent accéder.

Cela constitue une contrainte que de nombreuses compagnies et administrations de l'industrie des télécommunications ont cherché à éviter. Cela pourrait également nécessiter qu'un système géré prenne en charge de grands nombres de références IOR, ce que certains fournisseurs de système géré voudraient éviter. La présente Recommandation définit, conjointement avec la Rec. UIT-T X.780.1, la façon dont le cadre RGT de l'architecture CORBA doit être élargi de façon à éviter la nécessité de grands nombres de références IOR.

Les interfaces en architecture CORBA qui utilisent la méthode permettant d'accéder à chaque ressource gérable au moyen d'une unique référence IOR sont appelées maintenant interfaces "à granularité fine". En revanche, celles dans lesquelles une référence IOR n'est pas attribuée à chaque ressource gérable sont appelées interfaces "à granularité grossière".

Etant donné que la Rec. UIT-T X.780.1 définit une approche légèrement différente pour la modélisation des ressources gérables aux interfaces à granularité grossière, les spécifications des modèles d'interface seront légèrement différentes pour les approches à granularité fine et grossière.

1.3 Structure de la Recommandation

La présente Recommandation est structurée comme suit:

Paragraphe 1	Introduction, structure et mises à jour.
Paragraphe 2	Références.
Paragraphes 3 et 4	Définition des termes et abréviations utilisés dans la présente Recommandation.
Paragraphe 6	Ajout au cadre de considérations relatives à la conception dont il faut tenir compte pour la prise en charge des interfaces à granularité grossière.
Paragraphe 7	Aperçu général des exigences du cadre RGT de l'architecture CORBA et des interfaces à granularité grossière.
Paragraphe 8	Exigences relatives à l'utilisation des services communs aux objets du groupe OMG afin de prendre en charge les interfaces de gestion de réseau à granularité grossière.
Paragraphe 9	Exigences relatives à l'utilisation de services supports spécifiques du RGT afin de prendre en charge les interfaces de gestion de réseau à granularité grossière. Le cadre original a défini certains services nouveaux qui devront faire l'objet d'une extension pour prendre en charge les interfaces à granularité grossière. Un nouveau service support est également requis.
Paragraphe 10	Directives d'observance et de conformité.
Annexe A	Spécification en langage IDL des services supports à granularité grossière spécifiques du RGT.

2 Références normatives

La présente Recommandation se réfère à certaines dispositions des Recommandations UIT-T et textes suivants qui, de ce fait, en sont partie intégrante. Les versions indiquées étaient en vigueur au moment de la publication de la présente Recommandation. Toute Recommandation ou tout texte étant sujet à révision, les utilisateurs de la présente Recommandation sont invités à se reporter, si possible, aux versions les plus récentes des références normatives suivantes. La liste des Recommandations de l'UIT-T en vigueur est régulièrement publiée.

[1] UIT-T Q.816 (2001), *Services RGT à architecture CORBA*.

[2] UIT-T X.780 (2001), *Directives concernant le RGT pour la définition d'objets gérés CORB*A.

[3] UIT-T X.780.1 (2001), *Directives concernant le RGT pour la définition d'objets gérés CORBA à granularité grossière*.

[4] OMG Document formal/99-10-07, *The Common Object Request Broker: Architecture and Specification*, Révision 2.3.1.

[5] OMG Document formal/01-02-65, *Naming Service Specification*.

[6] OMG Document formal/00-06-20, *Notification Service Specification*, Version 1.0.

[7] OMG Document formal/00-01-04, *Telecom Log Service Specification*, Version 1.0.

[8] OMG Document formal/00-06-25, *Security Services Specification*, Version 1.5.

[9] OMG Document formal/00-06-28, *Transaction Service Specification*, Version 1.1.

[10] OMG TC Document orbos/98-05-05, *CORBA Messaging*.

[11] IETF RFC 2246 (1999), *The TLS Protocol Version 1.0*.

3 Définitions

3.1 Définitions extraites de la Rec. UIT-T X.701

Les termes suivants, utilisés dans la présente Recommandation, sont définis dans la Rec. UIT-T X.701:

– classe d'objets gérés;

– gestionnaire;

– agent.

3.2 Définitions extraites de la Rec. UIT-T X.703

Le terme suivant, utilisé dans la présente Recommandation, est défini dans la Rec. UIT-T X.703:

– notification.

3.3 Définitions extraites de la Rec. UIT-T Q.816

Le terme suivant, utilisé dans la présente Recommandation, est défini dans la Rec. UIT-T Q.816:

– canal d'événements.

4 Abréviations

La présente Recommandation utilise les abréviations suivantes:

AMI invocation de méthode asynchrone (*asynchronous messaging invocation*)

API interface de programmation d'application (*application programming interface*)

CMIP protocole commun de transfert d'informations de gestion (*common management information protocol*)

CORBA architecture de courtier commun de requête d'objets (*common object request broker architecture*)

COS services communs aux objets (*common object services*)

DN nom distinctif (*distinguished name*)

EMS système de gestion d'élément (*element management system*)

GDMO directives pour la définition d'objets gérés (*guidelines for the definition of managed objects*)

ID identificateur

IDL langage de définition d'interface (*interface definition language*)

IIOP protocole d'interopérabilité Internet (*Internet interoperability protocol*)

IOR référence d'objet interexploitable (*interoperable object reference*)

MO objet géré (*managed object*)

MOO opération sur objets multiples (*multiple object operation*)

NE élément de réseau (*network element*)

NMS système de gestion de réseau (*network management system*)

OAM&P exploitation, administration, maintenance et fourniture (*operations, administration, maintenance and provisioning*)

OID identificateur d'objet (*object identifier*)

OMG groupe de gestion par objets (*object management group*)

ORB gestionnaire en ressources d'objets (*object request broker*)

OSI interconnexion des systèmes ouverts (*open systems interconnection*)

PDU unité de données protocolaire (*protocol data unit*)

PM gestion de la performance (*performance management*)

POA adaptateur d'objet portable (*portable object adapter*)

QS qualité de service

RDN nom distinctif relatif (*relative distinguished name*)

RGT réseau de gestion des télécommunications

SSL couche des numéros de connexion logique sécurisés (*secure socket layer*)

TII invocation indépendante du temps (*time-independent invocation*)

TLS sécurité de couche de transport (*transport layer security*)

UIT-T Union internationale des télécommunications – Secteur de la normalisation des télécommunications

5 Conventions

5.1 Conventions applicables à la Recommandation

Un petit nombre de conventions sont appliquées dans la présente Recommandation afin d'informer le lecteur de l'intention du texte. Bien que la plus grande partie de la Recommandation soit normative, les alinéas qui décrivent succinctement les exigences obligatoires qu'un système de gestion (gérant et/ou géré) doit respecter sont précédés de la lettre "R" en caractère gras entre parenthèses, suivie d'un nom court indiquant le sujet de la prescription et d'un numéro. Par exemple:

(R) EXEMPLE-1 Exemple d'exigence obligatoire.

Les exigences dont l'application par un système de gestion est facultative sont précédées de la lettre "O" au lieu de "R". Par exemple:

(O) OPTION-1 Exemple d'exigence facultative.

Les déclarations d'exigences servent à créer des profils d'observance et de conformité.

La présente Recommandation donne, dans l'Annexe A, de nombreux exemples en langage IDL de l'architecture CORBA et de spécifications IDL des services spécifiques du RGT ainsi que de types de données supports. Les spécifications en langage IDL sont écrites en caractères "courrier" de corps 9:

```
// Example IDL
interface foo {
     void operation1 ();
};
```

5.2 Compilation du langage IDL

L'utilisation du langage IDL pour spécifier des interfaces de gestion de réseau est que ce langage peut être "compilé" en code de programmation par des outils qui accompagnent un gestionnaire ORB, ce qui automatise en fait la mise au point d'une partie du code nécessaire pour que les applications de gestion de réseau puissent interfonctionner. La présente Recommandation comporte en annexe le code que les réalisateurs pourront juger utile d'extraire et de compiler. L'Annexe A est normative. Il y a lieu qu'elle soit utilisée par les développeurs réalisant des systèmes conformes à la présente Recommandation. Le langage IDL contenu dans cette Recommandation a été vérifié avec deux compilateurs afin de garantir son exactitude. Il faut utiliser un compilateur prenant en charge la version de l'architecture CORBA qui est spécifiée dans la Rec. UIT-T Q.816.

L'Annexe A a été formatée de façon qu'il soit simple de la transférer par couper-coller dans des fichiers en texte clair pouvant ensuite être compilés. L'on trouvera ci-après quelques conseils pour effectuer cette opération.

1) Le couper-coller semble mieux fonctionner à partir de la version Microsoft® Word® de la présente Recommandation. Le couper-coller à partir du format de fichier Adobe® Acrobat® semble inclure les en-têtes et bas de page, qui ne peuvent pas être compilés.
2) L'ensemble de l'Annexe A, à partir de la ligne "/* Ce code IDL …" jusqu'à la fin doit être sauvegardé dans un fichier nommé "itut_q816_1.idl" à l'intérieur d'un répertoire où il sera trouvé par le compilateur.
3) Les en-têtes imbriqués dans l'annexe n'ont pas besoin d'être supprimés. Ils ont été encapsulés dans des commentaires IDl et seront laissés de côté par le compilateur.
4) Les commentaires qui commencent par la séquence spéciale "/**" sont reconnus par les compilateurs qui convertissent le langage IDL en HTML. Ces commentaires contiennent souvent des instructions de formatage spéciales pour ces compilateurs. Ceux qui travailleront avec le langage IDL souhaiteront peut-être produire du code HTML car les fichiers HTML résultants contiendront des liens permettant une navigation rapide entre les fichiers.
5) L'annexe a été formatée avec des espaces de tabulation à intervalles de 8 espaces et avec des retours à la ligne devant permettre à presque tous les éditeurs de texte de travailler avec ce texte.

6 Considérations relatives à la conception d'une interface à granularité grossière

Ce paragraphe relève plusieurs considérations relatives à la conception dont le cadre doit tenir compte lorsqu'on y ajoute la prise en charge des interfaces à granularité grossière.

6.1 Nombre réduit de références IOR

Lorsqu'on ajoute la prise en charge des interfaces à granularité grossière au cadre, celui-ci doit permettre la croissance du nombre de ressources gérées (comme les terminaisons) sans augmentation du nombre de références IOR mises en jeu de part et d'autre de l'interface de gestion.

6.2 Capacité de construire des références IOR

Les premières discussions sur les avantages des interfaces à granularité grossière se sont principalement concentrées sur la nécessité de réduire le nombre de références IOR prises en charge par un système géré. La raison en était qu'au cours de la plupart des années 1990, les gestionnaires en ressources d'objets (ORB, *object request broker*) n'offraient aux développeurs de système aucun moyen normalisé d'assurer la rémanence de l'état d'un objet entre les invocations de méthode. Tous les objets assortis d'une référence IOR en instance devaient donc être conservés en mémoire, ce qui limitait le nombre de références IOR qu'un système pouvait prendre en charge. Le groupe OMG y a toutefois remédié par la spécification CORBA 2.2 et aujourd'hui la plupart des gestionnaires ORB prennent en charge la norme adaptateur d'objet portable (POA, *portable object adapter*) du groupe OMG, qui permet aux systèmes de conserver en mémoire rémanente l'état d'un objet entre invocations de méthode. Le nombre d'objets qu'un système géré peut maintenant prendre en charge n'est donc limité, théoriquement, que par le nombre d'objets qu'il peut mémoriser dans son espace disque.

Cette évolution n'enlève cependant pas tous les mérites des interfaces à granularité grossière. Il s'avère que le bénéficiaire sera surtout le système gérant. Lorsqu'un système gérant utilise une interface CORBA à granularité fine pour interagir avec chaque ressource gérée, il doit à un moment donné extraire la référence IOR de chacune de ces ressources. Dans un système comportant des millions de ressources gérées, cela correspond à des millions de références IOR. Les systèmes gérants implémenteront probablement une série de stratégies pour traiter de grands nombres de références IOR. La solution la plus simple consistera à résoudre simplement le nom d'une ressource gérée en fonction de sa référence IOR avant chaque interaction. Mais ce processus est lent et gaspille les ressources réseau de communication de données. Une solution consisterait à extraire une seule fois tous les noms et leurs références assorties puis à les mémoriser dans le système gérant. Une autre stratégie consisterait à mettre en mémoire cache les noms et les références IOR dans le système gérant en gardant à disposition les références IOR utilisées le plus récemment afin de pouvoir les consulter rapidement et en rejetant celles qui n'ont pas été utilisées depuis un certain temps afin de les extraire du système géré si elles sont de nouveau nécessaires. D'autres stratégies pourraient encore apparaître.

Les interfaces à granularité grossière offrent aux systèmes gérants la possibilité de ne pas avoir à implémenter de tels procédés. Elles permettent à un système gérant d'extraire initialement d'un système géré et de mémoriser un petit nombre de références IOR seulement. A cette fin, un système gérant doit cependant être en mesure d'indiquer, sur la seule base du nom d'une ressource gérée, quelle est la référence IOR déjà extraite qu'il y a lieu d'utiliser pour interagir avec cette ressource. Autrement dit, en fonction d'un nom, un système gérant doit être en mesure d'en déduire la référence IOR de l'interface avec cette ressource gérée spécifique. Si par contre le système gérant doit interroger le système géré avec le nom afin de découvrir la référence IOR correspondante, ce système gérant revient à l'implémentation des procédés décrits ci-dessus et l'avantage de l'adjonction au cadre de la prise en charge des interfaces à granularité grossière sera largement perdu.

6.3 Application des services du protocole cadre

La prise en charge des interfaces à granularité élevée doit toujours être ajoutée au cadre de façon que les services existants de ce protocole (comme le service d'opération sur objets multiples et le service terminateur) puissent être appliqués. Cela ne préjuge pas les modifications aux réalisations de ces services afin de prendre en charge les interfaces à granularité grossière.

6.4 Distinction entre les deux types d'objet

Un système gérant doit pouvoir opérer une distinction entre ressources gérées par interfaces à granularité fine ou à granularité grossière.

6.5 Nommage hiérarchique

Le nommage hiérarchique (fondé sur les confinements) de ressources gérées avec des interfaces à granularité grossière doit être pris en charge.

6.6 Migration d'une approche à une autre par les entités modélisées

Le cadre doit permettre aux réalisations de passer de la méthode à granularité grossière à la méthode à granularité fine (et inversement) pour accéder à un type particulier de ressource gérée.

6.7 Migration d'une technique à une autre par les entités modélisées

Le cadre doit permettre aux réalisations de passer à diverses techniques logicielles, comme les langages C++ et JavaBeans.

6.8 Noms distinctifs équivalents

Le nom distinctif d'une ressource gérée ne doit pas dépendre du fait que l'accès à la ressource s'effectue par interface à granularité fine ou grossière. Il doit également être possible de connaître les informations de relation de confinement à partir d'un emplacement donné, que l'on accède aux ressources du système géré par interface à granularité fine ou par interface à granularité grossière.

7 Aperçu général du cadre et de ses exigences

Le paragraphe 6 décrivait les considérations relatives à la conception qui doivent être résolues lorsque l'on ajoute au cadre la prise en charge des interfaces à granularité grossière. Le présent § et le reste de la Recommandation détaillent la façon dont le cadre sera étendu pour tenir compte de ces considérations. La Rec. UIT-T X.780.1 décrit la façon dont les interfaces à granularité grossière seront modélisées. L'on présentera d'abord un bref aperçu général du cadre actuel, puis un aperçu général de ses extensions.

7.1 Aperçu général du cadre

Le cadre des interfaces RGT en architecture CORBA est un ensemble de capacités. Un élément central de ce cadre est un ensemble OMG de services communs aux objets. Le cadre définit leur rôle dans les interfaces de gestion de réseau ainsi que leurs conventions d'utilisation. Le cadre définit également les services qui n'ont pas été normalisés en tant que services communs aux objets de l'OMG mais qui sont appelés à être normalisés aux interfaces de gestion de réseau conformément à ce cadre.

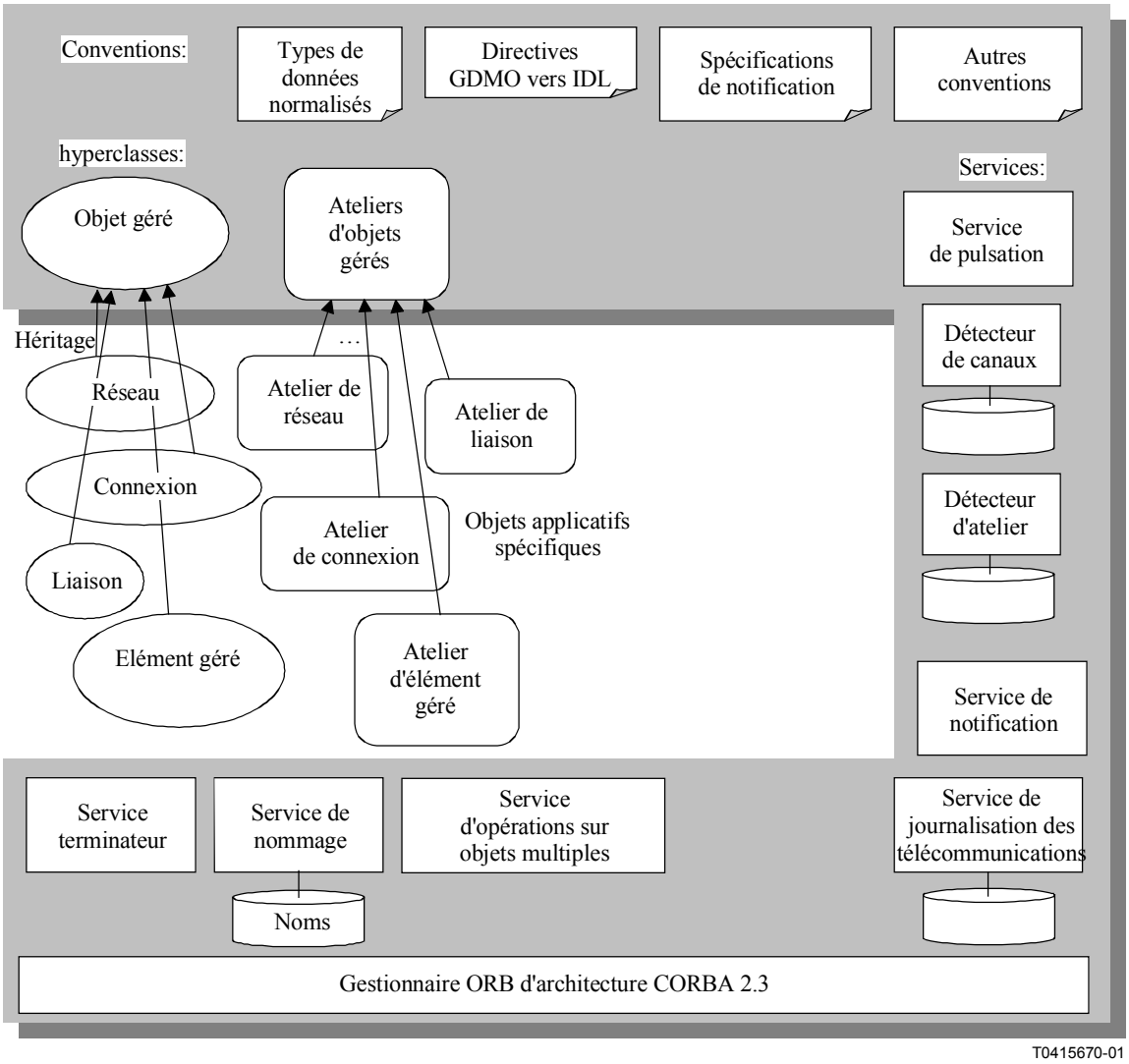

Figure 1/Q.816.1 – Aperçu général du cadre

Dans la Figure 1 ci-dessus, le cadre est représenté en gris. Au centre se trouvent les objets applicatifs spécifiques qui sont pris en charge par le cadre. Dans la rangée du bas, le cadre contient le gestionnaire ORB de l'architecture CORBA. Au-dessus, un certain nombre de cases contenant des noms représentent les services qui composent le cadre général. (Certaines cases contiennent des icônes décrivant les bases de données qu'elles devront conserver afin d'exécuter leurs fonctions.) Ces services sont définis dans la Rec. UIT-T Q.816 dans le cadre des exigences relatives à la version du gestionnaire ORB. En haut de la figure, des icônes représentent deux hyperclasses, l'une pour les objets gérés, l'autre pour les ateliers d'objets gérés. Chacun des objets gérés et des ateliers correspondants, pris en charge par ce cadre, doivent finalement hériter de ces deux hyperclasses respectives. La figure montre également des icônes de pages aux coins retournés, qui représentent des conventions normalisées de modélisation d'objet. Ces conventions et hyperclasses sont définies dans la Rec. UIT-T X.780.

7.2 Aperçu général des extensions à granularité grossière

Ce paragraphe donne un aperçu général des extensions qu'il faut apporter au cadre pour prendre en charge les interfaces à granularité grossière.

7.2.1 Structure nominale d'une façade

La plus importante modification qu'il est nécessaire d'apporter au cadre pour prendre en charge les interfaces à granularité grossière est la façon dont les objets gérés font l'objet d'un accès. Le nombre d'objets gérés dans un système géré doit pouvoir augmenter même si le nombre de références IOR prises en charge par ce système n'augmente pas. Il reste cependant souhaitable que l'accès aux objets gérés reste fortement dépendant du type. Cela conduit à l'utilisation d'une structure nominale désignée ci-après par le terme de *façade*. Une façade peut être considérée comme étant une interface fictive ou un portail. Au moyen de la structure nominale de façade, un système géré peut prendre en charge un petit nombre d'interfaces de façade, ce nombre étant au moins égal à 1 mais habituellement non supérieur à une dizaine pour chaque type d'objet géré dans ce système. Un système gérant invoquera alors une opération sur un objet géré en orientant sa demande d'opération réelle vers une façade pour ce type d'objet géré de ce système. Dans la structure nominale de façade, les objets gérés n'ont pas à affronter une interface CORBA et peuvent donc ne pas avoir de référence IOR individuelle. Il en découle qu'un système géré qui prend en charge la méthode des façades n'a pas besoin d'implémenter les interfaces d'objets gérés à granularité fine.

Il est préférable de considérer une façade, non pas comme un objet géré mais comme un objet intermédiaire qui permet à un système gérant d'accéder à des objets gérés. L'objet façade possède une interface CORBA et peut ne pas être directement accessible par cette architecture. La façade proprement dite ne représente pas une ressource réseau gérable. Sa fonction est de permettre l'interaction avec les objets qui représentent effectivement des ressources gérables. Tous les objets de façade sont créés automatiquement par le système géré et existent tant que les objets gérés sont accessibles par l'intermédiaire de cette façade. Plusieurs façades peuvent exister pour le même type d'objet géré à une interface à granularité grossière mais un objet géré doit être accessible par une seule façade. Voir la Figure 2 ci-dessous.

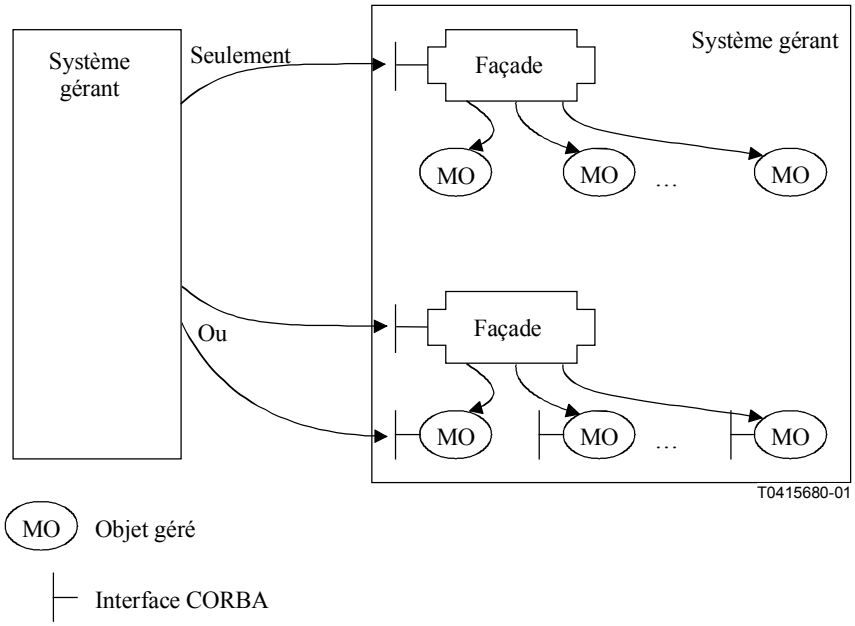

Figure 2/Q.816.1 – Rôle de façade

Cette figure montre un système gérant qui accède à un système géré prenant en charge l'approche par granularité grossière. Le système géré possède deux interfaces de façade qui permettent au système gérant d'accéder à deux ensembles différents d'objets gérés. Les objets gérés qui sont situés en haut de la figure ne peuvent faire l'objet d'un accès que par l'intermédiaire de la façade. Les objets gérés situés en bas prennent également en charge les interfaces CORBA directes et peuvent faire l'objet

d'un accès soit par l'intermédiaire de la façade soit directement. L'accès CORBA direct est facultatif mais un système géré qui prend en charge la méthode de façade doit fournir des interfaces-façades pour chacune de ses instances d'objet géré.

Une façade peut utiliser une interface CORBA d'objet géré pour invoquer une opération sur cet objet ou un autre moyen propre à la réalisation. En fait, un système géré n'a même pas besoin d'implémenter, sur le plan interne, des objets gérés en tant qu'objets individuels. Mais en implémentant une interface fondée sur ce cadre, il donne cependant l'illusion que des objets gérés sont implémentés sur le plan interne en tant qu'objets.

Lorsqu'une opération est invoquée au sujet d'un objet géré par l'intermédiaire d'une façade, celle-ci doit ensuite invoquer cette opération au sujet de l'objet géré ou de l'entité gérée en question. Etant donné que de nombreux objets gérés feront l'objet d'un accès par l'intermédiaire d'une seule façade, celle-ci doit être informée des objets gérés qui forment la cible réelle de l'opération. Cette information sera donnée par l'adoption de la convention d'inclure le nom des objets gérés cibles comme premier paramètre de chaque opération de façade visant un objet géré.

Bien que les objets gérés ne puissent plus avoir de références IOR uniques, ils ont toujours des noms uniques et peuvent toujours être considérés comme des entités individuelles représentant des ressources gérables.

7.2.2 Extension du nom d'un objet géré

Comme indiqué ci-dessus, les objets gérés ayant fait l'objet d'un accès par l'intermédiaire d'une façade conserveront un nom bien qu'ils n'aient peut-être plus d'interface CORBA individuelle. Il importe qu'un système gérant soit en mesure de déterminer la façade à utiliser sur la base du nom d'un objet géré. S'il n'y parvient pas, il devra interroger le système géré ou associer en permanence une référence IOR de façade à chaque nom d'objet géré. De façon à prendre en charge la capacité de déterminer la façade d'un objet géré sur la seule base de son nom, une légère extension sera apportée aux noms des objets gérés accessibles par une façade, par rapport aux noms des objet gérés non accessibles par une façade. Dans le composant nominatif final, qui possédera toujours une chaîne *ID* avec la valeur "Object" (ou, selon les extensions de la présente Recommandation, une valeur <empty>), la chaîne de sorte (*kind*) est mise à la valeur d'un identificateur de façade attribué à la façade par laquelle l'objet peut faire l'objet d'un accès. Dans le cas des objets gérés non accessibles par une façade, cette chaîne *kind* sera vide. Le § 8.1.2 donne de plus amples détails sur la façon dont l'identificateur de façade est utilisé pour désigner une façade.

7.2.3 Services supports pour objets gérés à façade accessible

Les services supports du cadre fournis aux interfaces qui utilisent la méthode de façade seront essentiellement les mêmes que les services supports définis dans la Rec. UIT-T Q.816. Certains, comme les services de détecteur d'atelier et de détecteur de canaux, ne nécessitent aucune modification que ce soit. D'autres, comme le service terminateur et le service d'opération sur objets multiples (MOO, *multiple object operation*), ne nécessitent pas de modification de leurs interfaces ou de la façon dont ils sont utilisés par les systèmes gérants mais peuvent nécessiter de légères modifications de leurs implémentations s'ils accèdent à des objets gérés au moyen des interfaces-façades d'objets gérés (plutôt que par une méthode propre à l'implémentation). Le § 9 donne des détails sur les modifications des services supports du cadre qui sont nécessaires afin de prendre en charge les interfaces à granularité grossière.

Le plus grand changement apporté aux services supports relève du domaine de la prise en charge du nommage. Les interfaces-façades sont liées aux noms contenus dans le service de nommage, de façon très semblable à la liaison des interfaces aux services supports. Aux interfaces qui utilisent des façades, les noms des objets gérés ne sont cependant pas nécessairement liés à des références IOR dans le service de nommage OMG. Par contre, un nouveau service est introduit en tant que lieu de mémorisation des noms d'objets gérés et des informations de relation de confinement de ces objets. Ce nouveau service, appelé *service de confinement*, est défini au § 9.6. Il doit être pris en charge par les systèmes qui utilisent des façades d'objets gérés mais il ne sera pas requis par les systèmes qui n'utilisent pas de façades.

7.2.4 Modélisation des façades

Une nouvelle interface de base est introduite afin de prendre en charge la structure nominale de façade et la définition des façades utilisables avec ce cadre. Cette interface de base sera appelée *interface-façade d'objet géré*. Elle jouera dans les interfaces à granularité grossière le même rôle que l'interface d'objet géré dans les interfaces à granularité fine. En d'autres termes, ce sera l'interface de base dont doivent hériter directement ou indirectement toutes les interfaces-façades d'objet géré pour fonctionner avec le cadre. L'interface-façade d'objet géré est tout à fait similaire à l'interface d'objet géré qui est définie dans la Rec. UIT-T X.780. L'on trouvera dans la Rec. UIT-T X.780.1 la définition de l'interface-façade d'objet géré, ainsi que d'autres directives de modélisation des informations à granularité grossière.

Les modifications apportées au cadre sont représentées dans la Figure 3 ci-dessous. Une nouvelle hyperclasse, *ManagedObjectFacade*, est ajoutée à cette figure, ainsi que le service de confinement. Noter que cette interface donne accès à une base de données contenant des noms d'objet géré. Cette base, tenue à jour par le service de nommage, est représentée par des pointillés indiquant qu'il n'est pas nécessaire de mémoriser les noms et les références IOR des objets gérés. Le service de nommage est cependant encore nécessaire pour permettre aux systèmes gérants de trouver les interfaces-façades et les références des services supports. Finalement, les objets gérés sont également représentés en pointillés afin d'indiquer qu'il n'est pas nécessaire qu'ils soient directement accessibles.

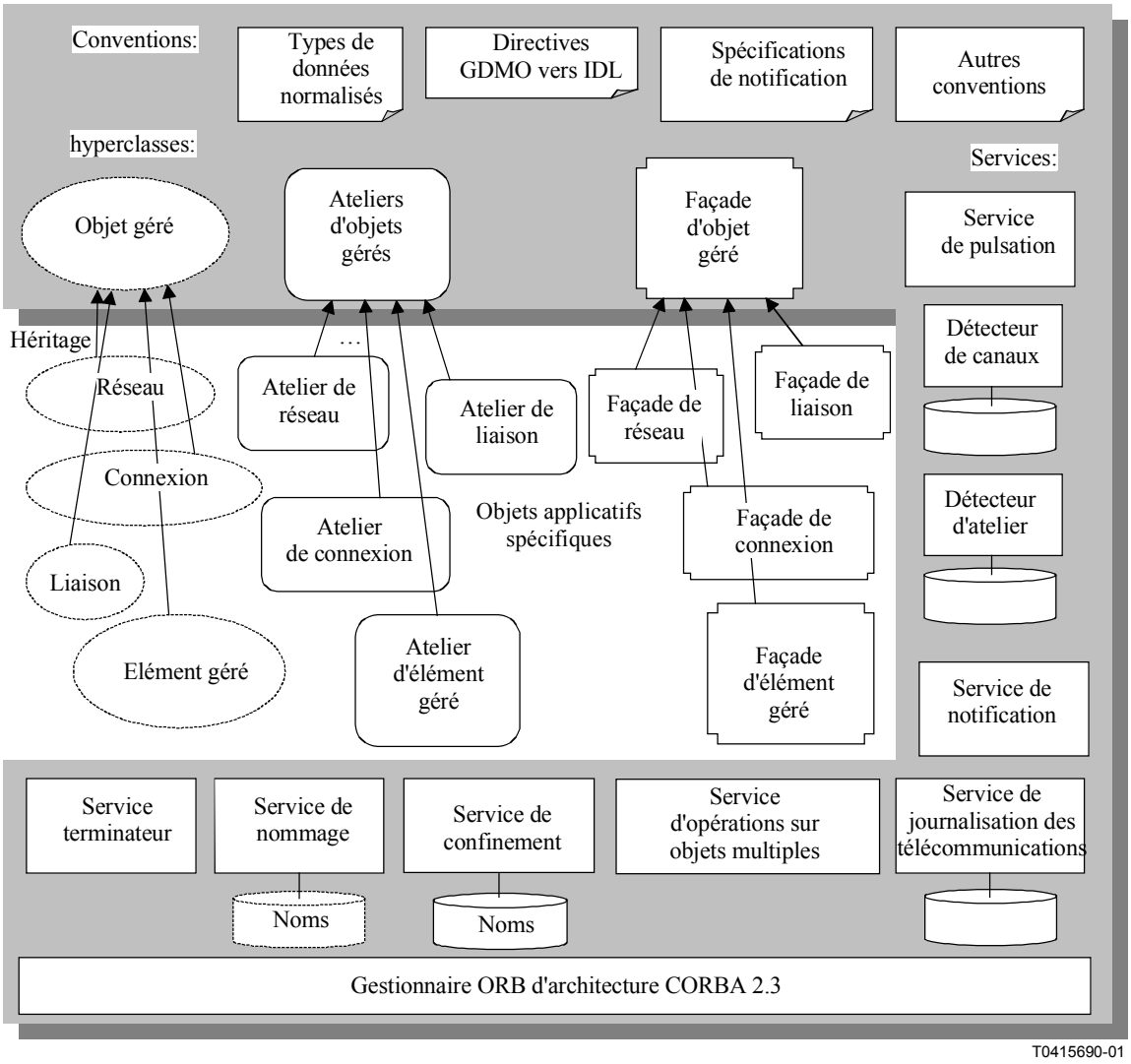

Figure 3/Q.816.1 – Cadre avec extensions de prise en charge d'interfaces à granularité grossière

8 Exigences cadres des services communs aux objets pour la prise en charge des interfaces à granularité grossière

La Rec. UIT-T Q.816 décrit comment le cadre original contient plusieurs des services communs aux objets de l'OMG. Il s'agit des services définis par le groupe OMG pour usage général dans toute application CORBA. Le cadre définit le service commun aux objets qui doit être pris en charge par un système géré, avec leurs conventions d'emploi. Ce paragraphe présente les conventions et exigences qu'il faut ajouter afin de prendre en charge les interfaces à granularité grossière selon chaque service commun aux objets du cadre.

8.1 Service de nommage

Ce paragraphe décrit, d'une part la façon dont le service de nommage de l'OMG est utilisé dans les interfaces à granularité grossière et, d'autre part, l'extension des noms d'objet géré qui est nécessaire pour prendre en charge les façades.

8.1.1 Utilisation du service de nommage aux interfaces à granularité grossière

Le rôle du service de nommage peut être beaucoup réduit aux interfaces à granularité grossière. Aux interfaces à granularité fine, les noms et les références IOR des objets gérés sont mémorisés dans le service de nommage. Dans les interfaces à granularité grossière, les objets gérés ne sont pas appelés à indiquer leurs références IOR et les corrélations de noms qui les concernent ne sont pas appelées à être mémorisées dans le service de nommage.

Le concept de contexte de nommage en racine locale s'applique également ici. Il ne faut pas perdre de vue le fait que le contexte de nommage en racine locale contient des corrélations de noms pour un ou plusieurs objets qui forment chacun la racine d'un arbre d'objets gérés associés par confinement. Par lui-même, un contexte de nommage en racine locale possède un nom unique et tous les objets qui en dépendent sont nommés par rapport à lui. Un contexte de nommage en racine locale contient également des corrélations avec les services du cadre qui prennent en charge les objets gérés qui ont été nommés par rapport à cette racine locale.

Les corrélations de noms d'objets gérés n'ont pas à entrer dans le service de nommage, alors que les corrélations pour les interfaces à façade le doivent. Les interfaces à façade possèdent bien des références IOR, dont il existe un nombre relativement restreint, de sorte que les corrélations de noms correspondantes sont simplement placées dans le contexte de nommage en racine locale avec les corrélations relatives aux services supports. Le format des noms correspondants est défini ci-dessous.

Le résultat de ces règles est que, dans un système géré muni d'une interface à granularité grossière, le service de nommage ne peut contenir que des contextes de nommage en racine locale. Etant donné que de nombreux systèmes ne posséderont qu'un seul contexte de nommage en racine locale, ces systèmes n'auront qu'un seul objet de contexte de nommage dans le service de nommage. Cet objet de contexte ne contiendra que les corrélations de noms pour les services supports et pour les interfaces-façades d'objet géré implémentées par le système.

8.1.2 Nom des interfaces de façade

Comme mentionné ci-dessus, les interfaces-façades possèdent des références IOR auxquelles les noms seront associés dans le contexte de nommage en racine locale. Le présent paragraphe décrit le format des noms associés aux références IOR d'une interface-façade.

Etant donné que les noms de toutes les façades d'un système sont placés dans le même contexte de nommage, ils doivent être uniques les uns par rapport aux autres. De même, il est utile qu'un système gérant puisse reconnaître qu'un nom appartient à une façade, par opposition aux noms des services supports, qui sont également associés dans le contexte de nommage racine. Finalement, comme décrit au § 6.2, un système gérant doit pouvoir déterminer la façade à utiliser sur la base du nom d'un objet géré. Pour répondre à cette dernière exigence, il faut assurer une certaine coordination entre noms de façade et noms d'objet géré.

Lorsqu'une référence IOR est associée à un nom dans un contexte de nommage OMG, ce nom se compose de deux chaînes: une chaîne *ID* et une chaîne de sorte. Toutes les interfaces-façades doivent être associées à la chaîne *ID* de valeur "façade" et la chaîne de sorte doit être mise à une valeur unique parmi tous les noms de façade de ce système. Cette valeur sera désignée par le terme "identificateur de façade". L'identificateur de façade entre cependant dans le champ de sorte plutôt que dans le champ *ID* car l'on verra plus loin qu'il doit toujours être mis en concordance avec la valeur contenue dans la chaîne de sorte d'un nom d'objet géré. L'on a estimé que la mise en concordance d'une valeur de sorte avec une autre valeur de type de *sorte* prêtait moins à confusion que l'insertion d'un identificateur dans un champ de sorte.

Etant donné que les noms des façades auront tous la même valeur "Façade" dans le champ *ID*, les chaînes de sorte doivent toujours être uniques de façon que la corrélation de nom soit unique, comme prescrit par le service de nommage OMG. La valeur de la chaîne de sorte doit être choisie par le réalisateur du système géré. La valeur réelle n'est pas importante mais, comme décrit ci-dessous, chaque objet géré faisant l'objet d'un accès par la façade aura la même valeur de chaîne de sorte dans le composant final de son nom.

8.1.3 Extension des noms d'objet géré

Les objets gérés accessibles par l'intermédiaire d'une façade auront des noms identiques à ceux qui ne sont pas accessibles par une façade, sauf pour les valeurs des chaînes *ID* et de sorte contenues dans le composant final du nom. La Rec. UIT-T Q.816 précise que les noms d'objet géré doivent avoir un composant nominatif "supplémentaire" ajouté à leur nom distinctif proprement dit. Ce composant nominatif final est nécessaire pour représenter des relations de confinement dans le service de nommage OMG. La Rec. UIT-T Q.816 prescrit que la chaîne d'identification contenue dans ce composant nominatif final ait une valeur égale à "Object" et que la chaîne de sorte ait une valeur néant. L'extension du cadre pour la prise en charge des interfaces à granularité grossière modifie ces exigences pour les objets accessibles par l'intermédiaire d'une façade.

Les objets gérés qui sont accessibles par l'intermédiaire d'une façade doivent avoir un composant nominatif final dont la chaîne de sorte est mise à la valeur de la chaîne de sorte contenue dans la corrélation de noms de la façade utilisée pour accéder à cet objet. En d'autres termes, la chaîne de sorte est mise à l'identificateur de façade choisi pour cette façade. Voir au § 8.1.2 ci-dessus les détails des noms de façade. Les objets gérés qui ne sont accessibles que par l'intermédiaire d'une façade doivent avoir une valeur de chaîne *ID* néant dans le composant final de leur nom. Les objets gérés qui sont accessibles, aussi bien directement que par l'intermédiaire d'une façade, doivent avoir une valeur *ID* égale à "Object" et une valeur de sorte égale à l'identificateur de façade contenu dans leur composant nominatif final.

Afin d'illustrer la façon dont les noms d'objet géré peuvent être réduits à une référence IOR de façade, considérons un système EMS détenu par l'exploitant "XYZ Communications". Ce système EMS peut avoir un contexte de nommage racine comportant le nom mondialement unique "ems17.ems.xyz.com". Admettons que la carte d'équipement de façade de ce système soit associée au nom "Façade.cp" dans le contexte de nommage racine, où "Façade" est la valeur de la chaîne d'identification et où "cp" est l'identificateur de façade inséré dans la chaîne de sorte. Une instance de l'objet géré de carte d'équipement pourra donc avoir le nom suivant:

```
ems17\.ems\.xyz\.com/me1.ManagedElement/bay1.Equipment/shelf1.Equipment/
slot1.Equipment/cp1.Equipment/.cp
```

(Lorsque des noms CORBA, composés d'une séquence de structures de données contenant chacune une chaîne nommée "ID" et une chaîne nommée de sorte, sont représentés sous forme de chaîne, la spécification du service de nommage de l'OMG [5] indique que ces noms doivent être représentés dans le format suivant:

$ID_1.Kind_1/ID_2.Kind_2/ID_3.Kind_3\ldots$

Où ID_1 représente la valeur de la chaîne *ID* dans le premier composant du nom, etc. Une barre oblique ("/") sépare donc les valeurs d'un composant donné des valeurs du composant suivant et un point (".") sépare la valeur de la chaîne *ID* de chaque composant de la chaîne de sorte. La chaîne *ID* vient avant la chaîne de sorte et si celle-ci est vide, le point est éliminé. Si une valeur de chaîne *ID* ou de sorte comporte un point, une barre oblique inverse ('\'), ce caractère est évité si on le fait précéder d'une barre oblique inverse. C'est la raison pour laquelle des barres obliques inverses précèdent les points dans le nom ci-dessus.)

Ce nom étant posé, un système gérant peut déterminer si l'objet est accessible par l'intermédiaire d'une façade car la valeur de la chaîne de sorte n'est pas vide dans le composant final. Le système gérant peut ensuite trouver, en deux étapes, la référence IOR de la façade pour cet objet. Tout d'abord, le système gérant doit comparer le nom avec les contextes radicaux qui ont été enregistrés avec lui afin de trouver la meilleure correspondance. En l'occurrence, il s'agira du contexte radical nommé "ems17\.ems\.xyz\.com". Une fois cette correspondance trouvée, le système sait que le nom de la façade pour cet objet est "ems17\.ems\.xyz\.com/Facade.cp". Il tire cette déduction du fait que toutes les façades de ce système sont associées à ce contexte radical avec une chaîne *ID* de valeur "Façade" et que la chaîne de sorte doit correspondre à la valeur de chaîne de sorte contenue dans le composant final du nom d'objet géré: "cp". En admettant que le système gérant met en mémoire cache les références IOR pour les façades, ce système peut aller plus loin et invoquer une opération sur l'objet au moyen de la référence IOR de la façade et du nom de l'objet. Noter que le système gérant n'est pas tenu d'extraire ou de mémoriser un grand nombre de références IOR, ni d'interroger le système géré quant à la référence IOR avant chaque opération.

Etant donné que le nom d'un objet dépend dans une certaine mesure de la façon dont il fait l'objet d'un accès, le changement du mode d'accès à un objet entraîne l'effet secondaire qu'il faut changer son nom. Ainsi, par exemple, si un objet qui n'avait jusqu'alors été accessible que par l'intermédiaire d'une façade est rendu directement accessible, vraisemblablement grâce à une mise à jour logicielle, celle-ci doit contenir la mise à jour du nom d'objet. De même, si un objet qui avait été accessible par l'intermédiaire d'une seule façade est transféré vers une autre façade, son nom doit changer. En raison de ce niveau de difficulté, la modification de la façon dont on accède à un objet n'est pas jugée devoir se produire fréquemment.

8.1.4 Comparaison de noms d'objet

Le Tableau 1 compare la façon de construire les valeurs nominatives des différentes sortes d'interface.

Tableau 1/Q.816.1 – Comparaisons entre noms d'objet

Type d'interface	Association du nom dans le contexte nominatif radical?	Avant-dernier composant nominatif		Dernier composant nominatif	
		ID	Sorte	ID	Sorte
Service support	Toujours	Non applicable – 1 seul composant utilisé	Non applicable – 1 seul composant utilisé	Valeur unique extraite par le système géré comme "ChannelFinder1"	Nom du service entièrement détecté à l'interface comme "itut_q816:: ChannelFinder"
Façade	Toujours	Comme ci-dessus	Comme ci-dessus	"Façade"	Valeur unique dans les corrélations de nom de façade contenues dans le contexte radical

Tableau 1/Q.816.1 – Comparaisons entre noms d'objet

Type d'interface	Association du nom dans le contexte nominatif radical?	Avant-dernier composant nominatif		Dernier composant nominatif	
		ID	Sorte	ID	Sorte
Objet géré inaccessible par façade	Les objets supérieurs ont un contexte de nommage associé au contexte radical	Valeur choisie selon l'application pour nommer l'objet par rapport à son ascendant – le nom RDN	Valeur de la chaîne constante de sorte dans le module de nommage cité en référence lors de la création de l'objet	"Object"	<vide>
Objet géré accessible par une façade	Il n'est pas nécessaire d'avoir un nom dans le service de nommage mais les objets supérieurs seront nommés directement comme subordonnés du contexte de nommage radical	Comme ci-dessus	Comme ci-dessus	<vide>	Valeur choisie par le système géré pour la chaîne de sorte utilisée dans la corrélation de nom pour la façade par laquelle on a accès à l'objet. Voir la cellule expliquant les valeurs de chaîne de sorte pour les façades
Objet géré accessible par une façade ainsi que directement	Comme ci-dessus	Comme ci-dessus	Comme ci-dessus	"Object"	Comme ci-dessus

8.1.5 Mémorisation de noms d'objet géré dans le service de nommage

Bien que cela ne soit pas exigé, certains systèmes faisant appel à des façades peuvent continuer à choisir de mémoriser certains ou la totalité des noms d'objet géré dans le service de nommage OMG. Cela peut être dû au fait que certains ou la totalité des objets se relient directement à des interfaces CORBA et possèdent donc des références IOR qui peuvent être associées à leur nom.

Si un objet géré possède une référence IOR liée à son nom dans le service de nommage et qu'il soit également accessible par l'intermédiaire d'une façade, sa corrélation de nom dans le service de nommage aura un identificateur de valeur "Object" et une chaîne de sorte contenant l'identificateur de sa façade.

Un système qui ne mémorise que certains de ses noms d'objet géré dans le service de nommage peut avoir à traiter le problème d'associer une référence IOR à un nom d'objet géré situé, dans l'arbre de nommage, au-dessous d'un objet géré ne possédant pas de référence IOR. Noter que, normalement, cela ne se produira pas car les objets situés aux "feuilles" de l'arbre de nommage tendent à être les plus nombreux et sont donc les plus susceptibles de ne pas avoir de références IOR individuelles. Les objets intérieurs à l'arbre tendent à être moins nombreux et sont donc plus susceptibles d'avoir

des références IOR lorsque l'on choisit une réalisation mixte à granularité fine et grossière. Si cette situation se produit quand même, il faut la traiter en créant un contexte de nommage pour l'objet supérieur sans référence IOR, associé à la valeur d'identification "Object". Le système géré doit créer au-dessous de cet objet un contexte pour le nouvel objet géré et doit associer sa référence IOR à la valeur d'identification "Object" contenue dans ce contexte. La chaîne de sorte doit être mise à la valeur de la chaîne de sorte associée à la façade utilisée pour accéder à l'objet.

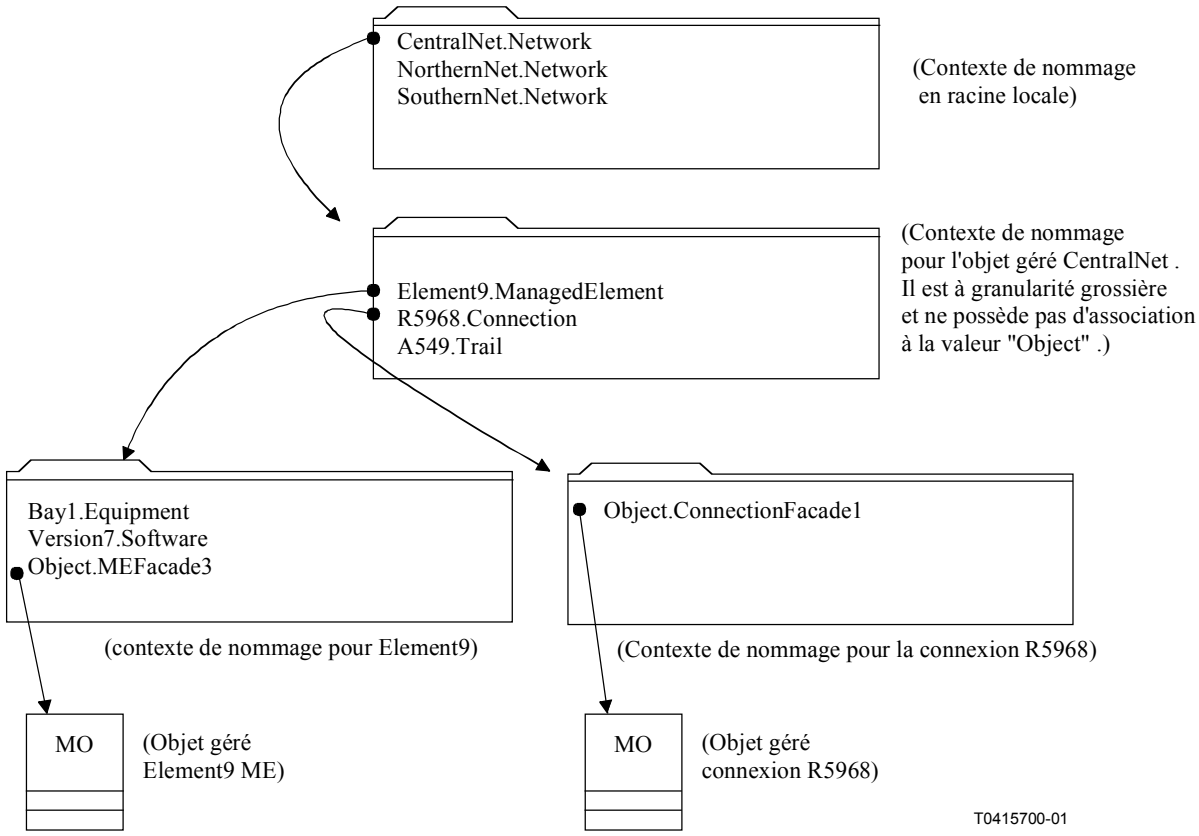

Figure 4/Q.816.1 – Graphe de nommage combinant des objets à granularité fine et grossière

Dans cet exemple, le nom de l'objet d'élément géré "Element9" et l'objet de connexion nommé "R5968" sont tous les deux contenus dans l'objet de réseau nommé "CentralNet", lequel est accessible par l'intermédiaire d'une façade mais indirectement. Les deux objets Element9 et R5968 sont accessibles aussi bien directement que par l'intermédiaire de façades. Afin de représenter cela dans le service de nommage, un contexte de nommage est créé pour CentralNet (dessiné en tant que deuxième répertoire à partir du sommet). Mais ce contexte n'a pas de corrélation avec une référence IOR vers CentralNet car celui-ci ne possède pas de référence IOR directe. Si CentralNet possédait effectivement une interface CORBA directe, le contexte de nommage CentralNet contiendrait une corrélation avec cet objet, dont la valeur ID serait égale à "Object". Les deux objets Element9 et R5968 possèdent des corrélations pour leur référence IOR, qui sont indiquées en bas de la figure.

Etant donné que les systèmes qui prennent en charge les façades ne sont pas tenus de communiquer leurs références IOR pour des objets gérés ni d'associer des noms à des références IOR pour des objets gérés dans le service de nommage, un système gérant utilisant une interface à granularité grossière ne peut pas compter sur le service de nommage pour recevoir des informations complètes au sujet de la relation de confinement. Un système géré peut n'associer aucun nom, ou n'associer que quelques noms à des références IOR du service de nommage. Dans les interfaces prenant en charge les façades, les systèmes gérants doivent donc utiliser le service de confinement pour extraire les

informations de confinement. Le service de nommage ne peut être utilisé que pour réduire des noms à des références IOR pour les objets qui possèdent de telles références individuelles.

Si le service de nommage est utilisé pour mémoriser des noms d'objet, il appartient au système géré de conserver ces informations dans le service de nommage de façon précise et synchronisée avec les informations contenues dans le service de confinement. A cette fin, un moyen peut consister à implémenter les deux services afin d'accéder à une base de données commune. D'autres moyens peuvent également être implémentés.

8.1.6 Exigences du service de nommage pour les interfaces à granularité grossière

Ce paragraphe définit les exigences relatives au service de nommage que les systèmes gérés doivent observer lors de l'utilisation d'interfaces à granularité grossière.

(R) NAME-1. Un système géré doit associer des noms aux références IOR de chaque façade utilisée dans le système. Une corrélation de nom doit être contenue pour chaque façade dans chaque contexte de nommage en racine locale. L'arborescence inférieure à cette racine contient les objets auxquels l'on accède par l'intermédiaire de la façade.

(R) NAME-2. La chaîne d'identification de chaque nom de façade doit avoir la valeur "Facade". La chaîne de sorte doit avoir une valeur choisie par le système géré de façon à être unique parmi les interfaces-façades associées dans un contexte de nommage radical particulier.

(R) NAME-3. S'il est mémorisé dans le service de nommage, le dernier composant nominatif du nom d'un objet géré accessible par l'intermédiaire d'une façade doit avoir une chaîne d'identification ayant la valeur "Object" et une chaîne de sorte ayant une valeur identique à la chaîne de sorte utilisée dans la corrélation de nom relative à la façade ayant permis l'accès à l'objet géré.

(R) NAME-4. Si le système géré mémorise tout ou partie des noms d'objet géré dans le service de nommage, ce système géré doit veiller à l'exactitude de ces noms et à l'intégrité de l'arbre de nommage.

(R) NAME-5. Si le système géré mémorise tout ou partie des noms d'objet dans le service de nommage, il utilisera des contextes de nommage sans référence IOR associée à la valeur d'identification "Object" pour les objets gérés ne possédant pas de référence IOR individuelle. Celles-ci n'ont besoin d'être utilisées que lorsqu'un objet géré possédant une référence IOR est associé à un nom dans l'arbre de nommage situé au-dessous d'un objet géré sans référence IOR.

8.2 Service de notification

Aucune exigence additionnelle n'est prescrite au sujet de l'utilisation du service de notification afin de prendre en charge les interfaces à granularité grossière. Le service de notification peut être utilisé pour acheminer, comme dans le cas des autres objets gérés, des notifications issues d'objets gérés accessibles par l'intermédiaire d'une façade.

8.3 Service de journalisation des télécommunications

Aucune exigence additionnelle n'est prescrite au sujet de l'utilisation du service de journalisation des télécommunications afin de prendre en charge les interfaces à granularité grossière.

8.4 Service de messagerie

Aucune exigence additionnelle n'est prescrite au sujet de l'utilisation du service de messagerie afin de prendre en charge les interfaces à granularité grossière.

8.5 Service de sécurité

Au moyen de la méthode des façades, le service de sécurité de l'OMG ne peut assurer la protection qu'au niveau des instances individuelles de façade. Il y a lieu d'utiliser un mode d'accès à granularité fine s'il y a des prescriptions visant à protéger les objets gérés au niveau des instances au moyen du service de sécurité OMG.

8.6 Service de transaction

Aucune exigence additionnelle n'est prescrite au sujet de l'utilisation du service de transaction afin de prendre en charge les interfaces à granularité grossière.

9 Exigences cadres des services supports pour la prise en charge des interfaces à granularité grossière

En plus des règles d'utilisation des services communs aux objets du groupe OMG, la Rec. UIT-T Q.816 définit quelques nouveaux services supports à utiliser aux interfaces CORBA-RGT. Ces services remplissent des fonctions communes qui sont propres à la gestion de réseau et qui ne sont pas fournies par les services à usage général communs aux objets du groupe OMG.

Les paragraphes ci-dessous définissent les exigences additionnelles que ces services doivent observer afin de prendre en charge les interfaces à granularité grossière. De même, un seul nouveau service est défini afin de conserver les relations de confinement de ressource aux interfaces à granularité grossière. Le langage IDL décrivant l'interface de ce nouveau service est reproduit dans l'Annexe A.

9.1 Service détecteur d'atelier

Le service détecteur d'atelier est défini dans la Rec. UIT-T Q.816 de façon à permettre aux systèmes gérants de trouver des "ateliers" concernant les systèmes gérés. Un système géré crée un nouvel objet relatif à un système géré en invoquant une opération liée à un atelier qui est lui-même un objet (il s'agit d'une structure nominale couramment employée en architecture CORBA). Un système gérant trouve un atelier en interrogeant ce service, qui est fourni par le système géré, avec le nom de classe d'un atelier. Le service répond par une référence à un atelier de ce type. Le système gérant peut ensuite invoquer l'opération appropriée concernant cet atelier puis créer l'objet. Le service détecteur d'atelier se trouve par recherche dans le service de nommage. Le système géré est tenu d'y placer une référence dans le contexte de nommage radical.

Cette même approche est utilisée pour les objets gérés qui doivent faire l'objet d'un accès par l'intermédiaire d'une façade. La seule différence est que si l'objet ne prend pas en charge une interface CORBA directe, l'atelier renvoie une référence vide. Il renverra cependant un nom approprié de façon que le système gérant puisse accéder à l'objet nouvellement créé par l'intermédiaire de sa façade.

Etant donné que la méthode de création d'objets gérés aux interfaces à granularité grossière est semblable à la méthode de création d'objets gérés aux interfaces à granularité fine, le service détecteur d'atelier n'a pas besoin d'être modifié. Aucune exigence additionnelle n'est donc prescrite au sujet de l'emploi du service détecteur d'atelier afin de prendre en charge les interfaces à granularité grossière.

9.2 Service détecteur de canaux

Le service détecteur de canaux est défini dans la Rec. UIT-T Q.816 afin de permettre aux systèmes gérants de découvrir les canaux d'événements présents dans un système géré ainsi que de découvrir les notifications que chaque canal manipule. Un petit système géré peut n'avoir qu'un seul canal mais un système plus complexe peut avoir plusieurs canaux, éventuellement pour différents ensembles de

ressources gérées ou pour différents types de notification. Le service détecteur de canaux ne permet pas à un système gérant de créer de nouveaux canaux d'événement relatifs au système géré, ni de modifier les événements manipulés par l'un des canaux. Les canaux d'événement individuels ne permettent cependant pas aux systèmes gérants d'ajouter des filtres et des destinations pour les notifications manipulées par un canal. Le système géré inscrit donc des informations sur la configuration des canaux événementiels auprès du service détecteur de canaux, informations que le système gérant pourra extraire et utiliser par la suite afin de déterminer les canaux qu'il doit surveiller. L'on trouve le service détecteur de canaux en le recherchant dans le service de nommage. Le système géré est tenu d'y placer une référence dans le contexte de nommage radical.

Lorsque le service détecteur de canaux fournit des informations au sujet des objets pour lesquels il manipule des événements, il identifie les objets par leur nom. Comme les objets gérés possèdent, aux interfaces à granularité grossière, un nom individuel fondé sur leur confinement, exactement comme les objets gérés des interfaces à granularité fine, le détecteur de canaux peut être utilisé de la même manière aux interfaces à granularité grossière. Aucune exigence nouvelle n'est donc prescrite concernant l'emploi du service détecteur de canaux pour prendre en charge les interfaces à granularité grossière.

9.3 Service terminateur

La Rec. UIT-T Q.816 définit le service terminateur comme un service commun implémentant les fonctions nécessaires pour supprimer un objet géré. Chaque objet géré pris en charge par le cadre possède un attribut de politique de suppression qui est réglé lors de la création de l'objet. Le service terminateur veille à ce que cette politique de suppression soit suivie lors de la suppression de l'objet. De même, les objets gérés sont nommés sur la base de relations de confinement. Il importe donc que les objets qui contiennent d'autres objets ne soient supprimés que si les objets contenus sont également supprimés. Le service terminateur effectue cette opération et assure la validité de l'arbre de nommage fondé sur le confinement.

Les objets gérés seront traités de la même manière à une interface à granularité grossière. Le service terminateur restera donc nécessaire pour jouer son rôle aux interfaces à granularité grossière. Il y a cependant quelques différences liées aux interfaces à granularité grossière qui auront une incidence sur la façon dont le service terminateur jouera son rôle. Premièrement, si le service terminateur utilise l'interface CORBA des objets gérés pour en extraire leur politique de suppression (par opposition à un moyen non transparent), il devra le faire de la façon prescrite par la Rec. UIT-T X.780.1 afin d'interagir avec les objets gérés par l'intermédiaire d'une façade. Deuxièmement, comme les noms des objets gérés ne peuvent pas être mémorisés dans le service de nommage OMG, le service terminateur doit extraire les informations de confinement à partir du service de confinement (voir § 9.6) et y maintenir l'exactitude de la hiérarchie de nommage. Si les noms d'objet géré sont mémorisés dans le service de nommage ainsi que dans le service de confinement, le service terminateur doit mettre à jour les arbres de nommage dans les deux services, à moins que ceux-ci ne se synchronisent eux-mêmes.

Ces modifications n'imposeront cependant pas de modification de l'interface IDL du service terminateur. La Rec. UIT-T Q.816 définit cette interface avec deux opérations de suppression: l'une qui prend un nom d'objet géré pour identifier l'objet, l'autre qui prend une référence d'objet géré (IOR). L'opération qui prend une référence d'objet géré ne s'appliquera pas aux interfaces à granularité grossière puisque les objets gérés ne peuvent pas avoir de références IOR individuelles et uniques. L'opération est définie de façon que la référence soit de type *ManagedObject* ou soit une sous-classe. Les interfaces-façades n'héritent pas de cette interface, de sorte qu'un système gérant n'aura pas la possibilité d'essayer, par erreur, d'utiliser cette interface pour effacer un objet géré ne possédant pas de référence IOR.

Un système gérant supprimera des objets gérés aux interfaces à granularité grossière en transmettant le nom de ces objets à l'opération *deleteByName* du service terminateur, exactement comme il peut le

faire pour un objet géré à une interface à granularité fine. Le service terminateur doit détecter que le nom se rapporte à un objet accessible par façade, sur la base des informations contenues dans le nom proprement dit (voir § 8.1.3). Le service terminateur doit ensuite déterminer l'interface-façade qui peut être utilisée pour accéder à cet objet géré puis extraire la politique de suppression de l'objet en examinant son nom. Si le service terminateur finit par supprimer l'objet, il le fait par invocation de l'opération *destroy* à la même façade, de nouveau en examinant le nom de l'objet. Cette opération est très semblable au cas des objets gérés situés aux interfaces à granularité fine, sauf qu'avec ces objets le service terminateur utilise directement l'interface de l'objet, plutôt qu'une interface-façade.

(R) TERM-1. Le service terminateur doit détecter les différences de nom pour les objets accessibles par l'intermédiaire d'une façade et pour les objets qui ne le sont pas. Il doit y accéder en conséquence.

(R) TERM-2. Lors de la suppression d'un objet géré accessible en façade et lors d'un accès à cet objet en architecture CORBA, le service terminateur doit utiliser l'interface-façade de cet objet afin d'en extraire sa politique de suppression et de le supprimer (à moins qu'il n'utilise une méthode d'accès non CORBA, propre à la réalisation).

(R) TERM-3. Lors de la suppression d'un objet accessible en façade, le service terminateur doit accéder aux informations de confinement mémorisées dans le service de confinement. Le service terminateur doit également veiller à ce que les noms contenus dans le service de confinement soient corrects au fur et à mesure de la suppression des objets.

9.4 Service d'opérations sur objets multiples

La Rec. UIT-T Q.816 définit le service d'opérations sur objets multiples comme un service commun qui peut être utilisé par un système gérant afin d'invoquer des opérations relatives à des groupes d'objets gérés au moyen d'une seule invocation (ou d'un petit nombre d'invocations) de méthode entre système gérant et système géré. Pour utiliser ce service, le système gérant invoque une seule opération par le service d'opérations MOO qui réside dans le système géré. Les opérations prises en charge sont *get* (requête) afin d'extraire une ou plusieurs valeurs d'un groupe d'objets gérés; *set* (mise à jour) afin de modifier une ou plusieurs valeurs d'un groupe d'objets gérés; et *delete* (suppression) afin de supprimer un groupe d'objets gérés. Le groupe d'objets visé par les opérations est désigné par un domaine de visibilité et par un filtre. Un domaine de visibilité désigne un objet de base assorti d'un certain ensemble d'objets contenus dans l'objet de base, d'après leur nom. Un filtre est une instruction logique qui contrôle les valeurs des attributs des objets contenus dans le domaine de l'opération. Si l'instruction donne la valeur "vrai" pour un objet particulier, l'opération est appliquée à cet objet. Si l'opération est appliquée à de nombreux objets, les résultats peuvent être trop volumineux pour faire l'objet d'un retour dans un même lot. Le service d'opérations MOO utilise la structure nominale de l'itérateur afin que le système gérant puisse extraire les résultats en volumes de lot gérables.

La Rec. UIT-T X.780.1 définit également, en fournissant une liste de noms d'objet lors d'une opération de "requête générale" une opération de façade commune qui permet à un système gérant d'extraire des valeurs d'attribut de multiples objets faisant l'objet d'un accès par cette façade. Cette opération de requête générale n'offre cependant pas la flexibilité et la puissance du service d'opérations MOO. Il est donc souhaitable que le service MOO soit également applicable aux interfaces à granularité grossière.

Il y a toutefois quelques différences avec les interfaces à granularité grossière qui affecteront la façon dont le service d'opérations MOO remplit son rôle. Premièrement, étant donné que les noms d'objet géré ne peuvent pas être mémorisés dans le service de nommage OMG, le service d'opérations MOO devra extraire les informations de confinement à partir du service de confinement (voir § 9.6). Deuxièmement, si le service d'opérations MOO utilise l'interface CORBA d'objets gérés (et non pas un moyen non transparent) pour extraire des valeurs d'attribut pour le filtrage et l'invocation de l'opération proprement dite, ce service devra utiliser la façade d'objet géré.

Troisièmement, alors que les interfaces à granularité grossière ne possèdent pas de méthode pour renvoyer un nom d'objet, le service d'opérations MOO doit toujours renvoyer le nom de chaque objet dans le cadre des résultats d'une opération de requête cadrée, exactement comme il le fait pour les interfaces à granularité fine. Il doit également permettre aux systèmes gérants d'énumérer les noms d'objet contenus dans la liste d'attributs qu'il souhaite extraire.

Ces modifications n'obligeront cependant pas à modifier l'interface IDL du service MOO ou de ses itérateurs. La Rec. UIT-T Q.816 définit l'interface du service MOO au moyen de trois opérations: requête, mise à jour et suppression, dont chacune prend le nom de l'objet de base, un indicateur de domaine et une instruction de filtrage. Si les attributs doivent faire l'objet d'une action, ils sont identifiés par un nom. L'ensemble est identique aux interfaces à granularité grossière ou fine. Ce qui change avec les interfaces à granularité grossière est l'endroit où les informations de confinement sont situées et la façon dont les attributs d'objet géré font l'objet d'un accès.

Les informations de confinement sont mémorisées dans le service de confinement au lieu du service de nommage. Afin d'accéder aux attributs d'un objet géré, le service MOO doit déterminer l'interface-façade de cet objet et y invoquer l'opération en fournissant le nom de l'objet géré. L'utilisation du service de confinement pour extraire des informations de confinement peut en fait être plus simple aux interfaces à granularité grossière puisque le service de confinement accepte le nom et le domaine d'un objet de base. Le service MOO peut simplement transmettre au service de confinement le nom et le domaine de l'objet de base qui lui ont été fournis puis effectuer une itération sur les objets confinés afin de déterminer s'ils traversent le filtre. Il n'a pas à naviguer dans l'arbre de nommage.

L'accès à des attributs d'objet géré par l'intermédiaire d'une façade n'est pas très différent de l'accès direct aux attributs. Pour un nom donné, la façade correcte est déterminée par mise en correspondance de la sorte d'objet (indiquée dans le champ de sorte du dernier composant nominatif) avec une façade inscrite dans le contexte de nommage radical. L'opération est ensuite appliquée à la façade avec fourniture du nom de l'objet géré en tant que premier paramètre de l'opération.

(R) MOO-1. Le service d'opérations MOO doit reconnaître – et traiter en conséquence – les différences entre les noms d'objets gérés accessibles par l'intermédiaire d'une façade et les noms de ceux qui ne le sont pas.

(R) MOO-2. Lors de l'accès à un objet géré accessible en façade au moyen de l'architecture CORBA, le service d'opérations MOO doit utiliser l'interface-façade de cet objet géré (à moins qu'il n'utilise une méthode d'accès non-CORBA, propre à la réalisation).

(R) MOO-3. Le service d'opérations MOO doit déterminer les objets situés dans le domaine de visibilité d'une opération par accès aux informations de confinement mémorisées dans le service de confinement.

9.5 Service de pulsation

Etant donné que l'architecture CORBA assure la transparence des emplacements, les applications de gestion de réseau perdent la visibilité des connexions à des systèmes spécifiques. Cela facilite la mise au point des applications mais pourrait empêcher un système gérant de connaître le moment de la perte de communications avec un système géré. Pour contourner ce problème, la Rec. UIT-T Q.816 définit le service de pulsation, qui permet à un système gérant de configurer un système géré de façon qu'il émette périodiquement une brève notification. Celle-ci passe par chacun des canaux d'événement d'un système géré, de sorte qu'en plus du contrôle de la liaison de communication elle vérifie également le fonctionnement des canaux d'événement de notification. Un système gérant peut être alerté de problèmes s'il ne reçoit pas la notification périodique qu'il est censé recevoir d'un système géré.

Les interfaces à granularité grossière utiliseront les canaux d'événement au même titre que les interfaces à granularité fine. Le service de pulsation peut continuer à être utilisé afin de vérifier le

fonctionnement de ces canaux et le réseau de communication de données reliant le système géré à ses systèmes gérants. Aucune exigence additionnelle n'est imposée concernant l'utilisation du service de pulsation afin de prendre en charge les interfaces à granularité grossière.

9.6 Service de confinement

Les relations de confinement entre objets gérés aux interfaces à granularité fine sont représentées par les noms mémorisés dans le service de nommage de l'OMG. Une capacité similaire est requise pour les objets gérés aux interfaces à granularité grossière. En d'autres termes, une fonction doit pouvoir signaler les objets qui sont confinés dans un objet supérieur, afin de vérifier qu'un tel objet existe avant qu'un subordonné soit créé, afin de s'assurer que l'on ne crée pas deux objets portant le même nom, etc. Le cadre sera élargi afin de prendre en charge cette fonction par ajout d'un nouveau service de confinement.

(R) CONTAINMENT-1. Un système géré doit instancier au moins un objet de service de confinement. De même, chaque contexte de nommage en racine locale d'un système doit avoir au moins une corrélation de noms pour un objet du service de confinement. La valeur de la chaîne d'identification de cette corrélation doit simplement désigner le serveur, éventuellement avec une valeur de type "Containment1". La chaîne de sorte de la corrélation doit désigner la classe de l'objet ("itut_q816_1::ContainmentService").

9.6.1 Justification du service de confinement

Il existe de nombreux emplacements différents dans lesquels des noms d'objet géré pourraient être mémorisés: le service de nommage, les objets de façade, un service existant comme le service terminateur ou d'opérations MOO, ou un nouveau service.

La mise des noms dans le service de nommage n'est pas vraiment logique car la vraie fonction de ce service est d'associer des noms à des références IOR et les objets gérés situés à des interfaces à granularité grossière ne peuvent pas avoir leurs propres références IOR. L'association d'un nom à la référence IOR de la façade serait inopérante. Tous les objets gérés auxquels on accède par l'intermédiaire d'une façade partageraient la même référence IOR, de sorte que leurs noms seraient associés à la même référence IOR dans le service de nommage. Le résultat probable serait la mémorisation dans le service de nommage de grands nombres de copies de quelques références IOR seulement.

L'inscription des noms dans les objets de façade ou dans les objets gérés eux-mêmes, est une option logique mais une des fonctions du cadre est d'offrir des emplacements pour implémenter des fonctions communes. La duplication de la gestion des informations de confinement à chaque façade ou dans chaque objet géré va dans le sens contraire de ce qui précède.

Le service terminateur et le service d'opérations MOO se fondent tous les deux sur les informations de confinement afin d'exécuter leurs tâches, de sorte qu'une possibilité consiste à étendre ces services afin de gérer les informations de confinement. La tâche principale de ces services n'est cependant pas de gérer ces données. Le fait de placer ailleurs la gestion des informations de confinement permet à ces services de rester focalisés sur leurs tâches utiles.

Bien que l'ajout d'un nouveau service soit une modification notable du cadre, il semble pourtant que ce soit la meilleure solution. Un nouveau service offre un répertoire particulier aux informations de confinement ainsi que la possibilité d'introduire des fonctions permettant d'accéder à des informations de confinement non prises en charge par le service de nommage, comme l'utilisation du domaine de visibilité pour rechercher des objets confinés.

9.6.2 Description du service de confinement

La principale fonction qui doit être prise en charge par le service de confinement est de permettre à un système gérant d'interroger un système géré avec le nom d'un objet et de recevoir en retour le nom des objets contenus par cet objet. L'on définira également un moyen d'obtenir les noms ajoutés

au service et retranchés de celui-ci. Ces noms ne sont pas à l'usage des systèmes gérants mais à l'usage interne des objets gérés, des ateliers et d'autres parties d'un système géré. Ils sont fournis afin de faciliter la mise au point de composants réutilisables, éventuellement par des tierces parties, et sont définis à une interface distincte de celle qui est utilisée par les systèmes gérants. Finalement, un nombre éventuellement important de noms pourra être renvoyé en réponse à une requête, de sorte que l'on fait appel à la structure nominale de l'itérateur, décrite au § 9.6.2.3.

9.6.2.1 Interface avec le service de confinement

Le service de confinement offre trois opérations permettant d'extraire les informations de confinement. Le module IDL décrivant l'interface avec le service de confinement (sans commentaires) est reproduit ci-dessous.

```
interface Containment {

     boolean exists (in NameType name)
               raises (ApplicationError);

     NameSetType getContained (in NameType base,
               in ScopeType scope,
               in unsigned short howMany,
               out NameIterator iterator)
               raises (ApplicationError);

     NameSetType getContainedByKind (in NameType base,
               in ScopeType scope,
               in KindType kind,
               in unsigned short howMany,
               out NameIterator iterator)
               raises (ApplicationError);

};   // fin de l'interface avec le service de confinement
```

L'opération *exists* prend un nom et renvoie la valeur "vrai" si ce nom est inscrit dans le service de confinement. Les deux autres opérations renvoient le nom des objets confinés par l'objet désigné dans le paramètre *base*. Le paramètre *scope* peut être utilisé dans ces deux opérations afin de spécifier la partie à extraire de l'arbre d'objets confinés au-dessous de l'objet de base. La troisième opération, *getContainedByKind*, prend un paramètre *kind* afin de demander au service de confinement de renvoyer les noms d'objet d'une certaine sorte.

Il importe de noter, au sujet des noms échangés de part et d'autre de cette interface, que les noms fournis par le système gérant dans ces requêtes n'a pas besoin de contenir le composant final du nom lorsque la chaîne d'identification a la valeur "néant" ou "Object" et que la chaîne de sorte est soit vide ou mise à l'identificateur de façade. Les noms renvoyés au système gérant doivent cependant contenir toujours ce composant nominatif final. Si un système gérant possède le nom d'un objet mais ne contient pas le composant final, il peut donc l'obtenir en interrogeant le service de confinement afin de trouver les objets confinés mais après avoir mis le domaine de visibilité à la valeur "objet de base seulement". Un système gérant peut se retrouver avec un nom d'objet dépourvu du composant final si par exemple il tronque le nom d'un objet afin de trouver son ascendant.

Comme on le verra plus loin, les noms des contextes de nommage en racine locale doivent être inscrits avant que d'autres noms puissent être ajoutés au service de confinement. Les contextes de nommage en racine inscrits dans le service de confinement peuvent être extraits par présentation d'un nom de base de longueur nulle à l'opération *getContained* en même temps qu'un domaine de visibilité indiquant le premier niveau d'objets confinés. La présentation du nom d'un contexte de nommage radical produit simplement l'extraction des noms des objets gérés qui lui sont subordonnés. Il en est de même pour tout autre objet géré.

(R) CONTAINMENT-2. L'interface prise en charge par l'objet (les objets) du service de confinement doit être l'interface de confinement décrite ci-dessus et définie dans l'Annexe A en langage IDL de l'architecture CORBA.

(R) CONTAINMENT-3. En réponse à une invocation de l'opération *exists*, le service de confinement doit renvoyer la valeur "vrai" si le nom est déjà inscrit au service et la valeur "faux" dans le cas contraire. Si une erreur de serveur empêche cette détermination, une exception d'erreur d'application appropriée doit être transmise.

(R) CONTAINMENT-4. En réponse à une invocation de l'opération *getContained*, le service de confinement doit renvoyer la liste des noms des objets confinés par l'objet nommé dans le paramètre *base*. La liste d'objets confinés doit être déterminée conformément au paramètre *scope*. (Voir dans la Rec. UIT-T Q.816 une description des informations de visibilité.) Le nombre maximal de noms à renvoyer dans l'ensemble du résultat doit être la valeur du paramètre *howMany*. S'il faut renvoyer un nombre de noms supérieur à la valeur *howMany*, le service de confinement doit renvoyer une référence à un itérateur de nom et doit rendre le nom additionnel disponible de part et d'autre de cette interface. Si tous les noms peuvent être renvoyés dans l'ensemble du résultat, la référence de l'itérateur de nom doit être vide. Si un nom de base de longueur nulle est soumis, le premier niveau de noms confinés doit être celui des noms des contextes de nommage radicaux qui ont été inscrits. Si une erreur empêche le renvoi de la liste, une exception appropriée d'erreur d'application doit être transmise. En particulier, si le nom de base n'est pas inscrit, une exception d'erreur d'application de type *InvalidParameter* doit être transmise.

(R) CONTAINMENT-5. Le service de confinement doit répondre à l'invocation de l'opération *getContainedByKind* comme décrit dans la prescription CONTAINMENT-4, sauf que seuls les noms qui correspondent au paramètre *kind* doivent être renvoyés. Il convient de ne pas perdre de vue le fait que le composant final d'un nom possède un identificateur de valeur "Object". C'est l'avant-dernier composant dont l'identificateur contient le nom distinctif relatif de l'objet et dont le champ de sorte contient la valeur de sorte de l'objet. En réponse à une invocation de l'opération *getContainedByKind*, le service de confinement doit donc ne renvoyer que les noms dont le champ de sorte dans l'avant-dernier composant nominatif correspond à la valeur du paramètre *kind*.

9.6.2.2 Interface avec le composant du service de confinement

Quelques opérations supplémentaires sont définies à une interface distincte pour usage interne dans le système géré. Ces opérations sont définies afin de faciliter la mise au point de composants réutilisables du service de confinement, éventuellement par des tierces parties. La description en langage IDL de l'interface avec le composant du service de confinement est reproduite ci-dessous (sans commentaires):

```
interface ContainmentComponent : Containment {

     void registerLocalRoot (in NameType name,
                             in NamingContext localRoot)
                    raises (ApplicationError);

     void unRegisterLocalRoot (in NameType name)
                    raises (ApplicationError, DeleteError);

     void addName (in NameType name)
                    raises (ApplicationError, CreateError);

     void removeName (in NameType name)
                    raises (ApplicationError, DeleteError);
};
```

Les deux premières opérations servent à inscrire et à désinscrire des contextes de nommage en racine locale dans le service de confinement. Lorsqu'un nom est ajouté à ce service, le nom de son objet supérieur doit déjà être inscrit sinon la tentative d'adjonction de nom échoue. Au sommet de l'arbre de nommage, l'on trouvera cependant des objets sans objet supérieur dans le système géré. Ces objets

seront nommés par rapport à un contexte de nommage en racine locale. Sans indication des noms des contextes de nommage en racine locale, le service de confinement ne peut pas déterminer si un nouveau nom, dont l'objet supérieur n'est pas reconnu, est incorrect ou s'il s'agit d'un objet sommital d'un arbre de nommage.

Les deux opérations finales servent à ajouter et à supprimer des noms dans le service de confinement. Les clients les plus probables de ces opérations seront respectivement les ateliers d'objet géré et le service terminateur. Lorsqu'un nom est ajouté, il est vérifié afin de s'assurer qu'il est unique et que le nom de l'objet supérieur a été ajouté (à moins qu'il ne soit nommé directement par rapport à un contexte de nommage radical). Un nom n'est supprimé que s'il ne reste plus de noms subordonnés inscrits.

(O) CONTAINMENT-6. L'interface prise en charge par l'objet (les objets) du service de confinement peut être l'interface avec le composant de confinement décrite ci-dessus et définie dans l'Annexe A en IDL/CORBA.

(O) CONTAINMENT-7. Lorsqu'un contexte de nommage en racine locale est inscrit au moyen de l'opération *registerLocalRoot*, le service de confinement doit l'ajouter à sa liste de contextes de nommage en racine locale. Si une erreur de serveur l'en empêche, une exception appropriée d'erreur d'application doit être transmise.

(O) CONTAINMENT-8. Lorsqu'un contexte de nommage en racine locale est désinscrit au moyen de l'opération *unRegisterLocalRoot*, le service de confinement doit le retirer de sa liste de contextes de nommage en racine locale inscrits mais seulement si aucun nom d'objet subordonné n'est déjà inscrit. Si une erreur de serveur l'en empêche, une exception appropriée d'erreur d'application doit être transmise.

(O) CONTAINMENT-9. Lorsqu'un nom d'objet géré est ajouté au moyen de l'opération *addName*, le service de confinement doit ajouter ce nom à sa liste de noms d'objets inscrits mais seulement si ce nom est unique et directement subordonné à une racine locale inscrite ou à un autre nom inscrit. Si le nom n'est pas unique ou si aucun nom supérieur n'est inscrit, le service de confinement ne doit pas ajouter ce nom mais transmettre l'exception appropriée d'erreur de création. Noter que si deux noms sont identiques à l'exception de différences dans le composant nominatif final (dont la chaîne d'identification a la valeur "Object" et dont la chaîne *kind* désigne une façade), ces noms ne sont pas uniques. Si une erreur de serveur empêche l'opération de s'exécuter, une exception appropriée d'erreur d'application doit être transmise.

(O) CONTAINMENT-10. Lorsqu'un nom d'objet géré est supprimé au moyen de l'opération *removeName*, le service de confinement doit supprimer ce nom de sa liste de noms d'objets inscrits, mais seulement si aucun nom subordonné n'est déjà inscrit. Si un ou plusieurs noms subordonnés existent, le service de confinement ne doit pas supprimer le nom mais plutôt transmettre une exception appropriée d'erreur d'application. Si une erreur de serveur empêche l'opération de s'exécuter, une exception appropriée d'erreur d'application doit être transmise.

9.6.2.3 L'itérateur de nom

Lorsque le service de confinement renvoie une liste de noms d'objets confinés, il peut y en avoir trop à renvoyer en même temps. C'est afin de permettre au service de confinement de renvoyer un nombre théoriquement illimité de noms que la structure nominale d'itérateur est utilisée. Comme décrit ci-dessus, si le service de confinement ne peut pas renvoyer une liste de noms en réponse à une opération, il en renvoie la plus grande partie possible avec une référence non vide à une interface avec un itérateur. La description en langage IDL de l'interface avec l'itérateur de nom est reproduite ci-dessous.

```
interface NameIterator {

     boolean getNext(in unsigned short howMany,
                     out NameSetType results)
          raises (ApplicationError);

     void destroy();

}; // fin de l'interface NameIterator
```

(**R) CONTAINMENT-11**. Le service de confinement doit prendre en charge l'utilisation des itérateurs de nom avec des interfaces correspondant à la description ci-dessus et aux définitions données en IDL dans l'Annexe A.

(R) CONTAINMENT-12. Chaque fois qu'un client invoque une opération *getNext* concernant un itérateur de nom, l'itérateur doit renvoyer l'ensemble de résultats suivant. L'itérateur doit conserver la trace du nombre de résultats qui ont déjà été extraits par le client puis doit renvoyer tous les résultats une seule fois. Les résultats initialement renvoyés en réponse à une opération concernant l'interface du service de confinement ne doivent pas être renvoyés de nouveau par l'itérateur. Celui-ci doit renvoyer en réponse à une opération *getNext* au plus le nombre de noms indiqué par la valeur du paramètre *howMany*. L'itérateur peut renvoyer moins que l'effectif du lot demandé afin de trouver un compromis entre l'efficacité du renvoi de résultats dans un grand lot avec l'éventuelle nécessité de bloquer ce renvoi en attendant qu'un plus grand nombre de résultats soit disponible. S'il y a plus de résultats à renvoyer (par rapport à ceux qui le sont), la valeur de retour de l'opération *getNext* doit être "vrai", sinon "faux". L'itérateur ne doit pas renvoyer d'ensemble de résultats vide, à moins que le paramètre *howMany* ait été mis à zéro ou qu'il n'y ait plus de résultats à renvoyer, car cela forcerait le client à interroger l'itérateur.

(R) CONTAINMENT-13. Le système géré doit commander la durée de vie de l'itérateur. Une opération de destruction est cependant prévue si le gestionnaire souhaite mettre fin à l'extraction des résultats avant d'arriver à la dernière itération. Sur invocation de l'opération *destroy*, l'itérateur doit libérer les éventuelles ressources qu'il est en train d'utiliser puis doit se supprimer. L'itérateur doit se détruire dès qu'il renvoie le dernier résultat. Il peut également être détruit par le système géré s'il reste inutilisé pendant une durée excessive.

10 Observance et conformité

Ce paragraphe définit les critères qui doivent être observés par d'autres normes revendiquant la conformité au présent cadre ainsi que les fonctions qui doivent être remplies par les systèmes revendiquant la conformité à la présente Recommandation.

10.1 Conformité du système

10.1.1 Points de conformité

Ce paragraphe résume les fonctions individuelles qui ont été décrites plus haut. Ces points de conformité sont ensuite combinés dans des profils qui doivent être pris en charge par les systèmes revendiquant la conformité à la présente Recommandation.

1) Nommage à granularité grossière: une réalisation revendiquant la conformité aux prescriptions du service de nommage à granularité grossière doit toujours:
   - prendre en charge toutes les exigences du service de nommage qui sont spécifiées au § 8.1.
2) Service terminateur à granularité grossière: une réalisation revendiquant la conformité aux prescriptions du service de nommage à granularité grossière doit toujours:
   - prendre en charge toutes les exigences du service terminateur qui sont spécifiées au § 9.3.

3) Service d'opérations MOO à granularité grossière: une réalisation revendiquant la conformité aux prescriptions du service d'opérations MOO à granularité grossière doit toujours:
   - prendre en charge toutes les exigences du service d'opérations MOO qui sont spécifiées au § 9.4.

4) Service de confinement: une réalisation revendiquant la conformité aux prescriptions du service de confinement doit toujours:
   - prendre en charge les exigences obligatoires du service de confinement qui sont spécifiées au § 9.6. Les exigences obligatoires sont précédées des caractères "**(R)**". Celles qui sont précédées des caractères "**(O)**" peuvent être prises en charge mais ne sont pas requises.

10.1.2 Profil de conformité de base

Un système revendiquant la conformité au profil de base de la Rec. UIT-T Q.816.1 doit prendre en charge:

1) le profil de base de la Rec. UIT-T Q.816 **sauf**:
   - qu'il n'est pas nécessaire que les noms d'objet géré soient associés à des références IOR d'objet géré dans le service de nommage OMG;

2) les prescriptions de nommage à granularité grossière (voir point de conformité 1 ci-dessus);

3) les prescriptions du service terminateur à granularité grossière (voir point de conformité 2 ci-dessus);

4) les prescriptions du service d'opérations MOO à granularité grossière (voir point de conformité 3 ci-dessus);

5) les prescriptions du service de confinement (voir point de conformité 4 ci-dessus).

10.2 Directives de déclaration de conformité

Les utilisateurs de ce cadre doivent prendre des précautions lors de la rédaction de déclarations de conformité. Etant donné que des modules en langage IDL sont utilisés comme espaces de mémorisation de noms, ces modules peuvent, comme cela est autorisé par les règles IDL de l'OMG, être répartis entre plusieurs fichiers. Ainsi, lorsqu'un module est étendu, son nom n'est pas modifié mais un nouveau fichier IDL est simplement ajouté. Se limiter à déclarer le nom d'un module dans une déclaration de conformité ne suffira donc pas à identifier un ensemble d'interfaces IDL. La déclaration de conformité doit donc désigner un document et une année de publication afin de s'assurer que l'on a identifié la version correcte du langage IDL.

ANNEXE A

Spécification IDL des services supports du cadre à granularité grossière

```
/* Ce code IDL est destiné à être mémorisé dans un fichier nommé "itut_q816_1.idl" qui est
situé dans le chemin de recherche utilisé par les compilateurs IDL de votre système. */

#ifndef ITUT_Q816_1_IDL
#define ITUT_Q816_1_IDL

#include <CosNaming.idl>
#include <itut_q816.idl>
#include <itut_x780.idl>

#pragma prefix "itu.int"


module itut_q816 {
```

```
// TYPES IMPORTÉS

     // Types importés de CosNaming
     typedef CosNaming::NamingContext NamingContext;
     typedef CosNaming::Istring KindType;

// INTERFACES

     /** L'interface avec l'itérateur de nom sert à extraire les résultats d'une
     opération getContained ou getContainedByKind d'une interface de confinement au
     moyen de la structure nominale de l'itérateur. */

     interface NameIterator {

          /** Cette méthode sert à extraire le nombre "howMany" de prochains
          résultats de l'ensemble de résultats.
          @param howMany  Nombre maximal d'éléments à renvoyer dans les résultats.
                          Un plus petit nombre peut être renvoyé si c'est tout ce
                          qui reste ou pour trouver un compromis entre délai et
                          efficacité.
          @param results  Prochain lot de résultats.
          @return         Vrai s'il y a encore des résultats après ceux qui sont
                          actuellement renvoyés. Si la valeur de retour est vraie,
                          l'ensemble de résultats ne devrait pas être vide car cela
                          forcerait le client à interroger l'itérateur pour obtenir
                          des résultats. Au lieu de cela, la communication doit
                          marquer un blocage.
          */

          boolean getNext(in unsigned short howMany,
               out NameSetType results)
               raises (itut_x780::ApplicationError);

          /** Cette méthode est utilisée pour détruire l'itérateur et libérer ses
          ressources. L'itérateur est cependant automatiquement détruit après le
          retour des derniers résultats. Il peut être détruit s'il n'est pas
          utilisé pendant une période excessivement longue.*/

          void destroy();

     }; // fin de l'interface NameIterator


     /** L'interface avec le service de confinement sert à extraire des informations
     d'un système géré au sujet des relations de confinement entre les objets gérés du
     système. Les relations de confinement sont représentées par des noms. Un objet
     qui est confiné par un autre objet est nommé par rapport à celui-ci. Tous les
     noms d'objet géré sont inscrits dans le service de confinement. */

     interface Containment {

          /** Cette méthode sert à vérifier si un nom est inscrit dans le service
          de confinement.
          @param name          Nom à vérifier pour voir s'il est inscrit
          @return              Vrai si le nom est inscrit. Tous les composants
                               du nom soumis correspondent à un nom inscrit.
          */

          boolean exists (in NameType name)
               raises (itut_x780::ApplicationError);

          /** Cette méthode sert à extraire les noms des objets confinés par un
          objet cible. Un objet est confiné par l'objet cible si son nom commence
          par tout composant sauf le dernier du nom de l'objet cible. La structure
          nominale d'itérateur est utilisée pour prendre en charge le retour de
          nombres éventuellement importants de noms. Si le nom n'existe pas, une
          exception d'erreur d'application est transmise pour indiquer un paramètre
          invalide (voir X.780 pour les codes d'exception d'erreur d'application)
```

```
        @param name     Nom de l'objet pour lequel les objets confinés sont
                        recherchés. N'a pas besoin de contenir le composant
                        nominatif final (dans lequel ID=Object, kind=facade)
        @param scope    Le domaine de visibilité est utilisé pour identifier la
                        partie de l'arbre d'objets confinés qu'il y a lieu de
                        renvoyer.
        @param howMany  Nombre maximal d'éléments à renvoyer dans les résultats.
                        S'il y a plus que "howMany" noms, un itérateur doit être
                        utilisé afin de renvoyer le reste.
        @param iterator Référence à un itérateur afin de renvoyer des résultats
                        additionnels. Si tous les résultats sont renvoyés en
                        réponse à l'appel, ce champ doit être vide.
        @return         Noms des objets confinés. Doit toujours contenir le
                        composant nominatif final.
        */

        NameSetType getContained (in NameType name,
             in ScopeType scope,
             in unsigned short howMany,
             out NameIterator iterator)
             raises (itut_x780::ApplicationError);

        /** Cette méthode sert à extraire les noms des objets confinés dans un
        objet cible qui correspond à une certaine sorte. Une sorte d'objet est la
        valeur du champ de sorte contenu dans l'avant-dernier composant de son
        nom. Ce champ doit correspondre à la valeur de sorte soumise.
        @see getContained
        @param name     Nom de l'objet pour lequel les objets confinés sont
                        recherchés. N'a pas besoin de contenir le composant
                        nominatif final (dans lequel ID=Object, kind=facade)
        @param scope    Le domaine de visibilité est utilisé pour identifier la
                        partie de l'arbre d'objets confinés qu'il y a lieu de
                        renvoyer.
        @param kind     Valeur qui doit être mise en correspondance avec le champ
                        de sorte de l'avant-dernier composant contenu dans les
                        noms renvoyés.
        @param howMany  Nombre maximal d'éléments à renvoyer dans les résultats.
                        S'il y a plus que "howMany" noms, un itérateur doit être
                        utilisé afin de renvoyer le reste.
        @param iterator Référence à un itérateur afin de renvoyer des résultats
                        additionnels. Si tous les résultats sont renvoyés en
                        réponse à l'appel, ce champ doit être vide.
        @return         Noms des objets confinés. Doit toujours contenir le
                        composant nominatif final.
        */

        NameSetType getContainedByKind (in NameType name,
             in ScopeType scope,
             in KindType kind,
             in unsigned short howMany,
             out NameIterator iterator)
             raises (itut_x780::ApplicationError);

};   // fin de l'interface Containment


/** L'interface avec le composant de confinement développe l'interface de
confinement afin d'ajouter des fonctions utilisées à l'intérieur d'un système.
géré */

interface ContainmentComponent : Containment {

        /** Cette méthode sert à inscrire un contexte de nommage en racine locale
        dans le service terminateur. La réinscription d'un nom produit la
        réutilisation subséquente par le service de la référence qui vient d'être
        fournie, si ce service a besoin d'accéder au contexte de nommage radical.  */

        void registerLocalRoot     (in NameType name,
                                    in NamingContext localRoot)
                                    raises (itut_x780::ApplicationError);
```

```
        /** Cette méthode sert à supprimer une inscription de contexte de nommage
        en racine locale. */

        void unregisterLocalRoot (in NameType name)
             raises (itut_x780::ApplicationError);

        /** Cette méthode sert à ajouter un nom au service de confinement. Le nom
        doit toujours se rapporter à un nom existant ou à un nom de racine
        locale. En d'autres termes, tous les composants nominatifs, sauf les deux
        derniers, du nom soumis doivent correspondre à tous les composants
        nominatifs, sauf le dernier, d'un nom inscrit. De même, le nom doit
        toujours être unique. En d'autres termes, tous les composants nominatifs
        du nom soumis, sauf le dernier, ne peuvent pas correspondre à tous les
        composants nominatifs, sauf le dernier, de tout autre nom inscrit. Si le
        nom n'a pas d'association avec un objet supérieur inscrit, le nom n'est
        pas inscrit et une exception d'erreur de création est transmise avec la
        cause mise à la valeur "badName". Si le nom n'est pas unique, il n'est
        pas inscrit et une exception d'erreur de création est transmise avec la
        cause mise à la valeur "duplicateName". (Voir X.780 pour les codes
        d'exception d'erreur de création.)
        @param name     Nom à ajouter.
        */

        void addName (in NameType name)
             raises (itut_x780::ApplicationError,
                  itut_x780::CreateError);

        /** Cette méthode sert à supprimer un nom du service de confinement. Il
        ne doit pas y avoir de noms inscrits dans le service de confinement
        lorsqu'un nom est supprimé. En d'autres termes, il ne doit pas y avoir
        d'autres noms commençant par tous les composants nominatifs, sauf le
        dernier, du nom à supprimer. S'il y en a, le nom n'est pas supprimé et
        une exception "DeletError" est transmise avec la cause mise à la valeur
        "containsObjects" (voir X.780). Si le nom n'existe pas, une erreur
        d'application est transmise avec la cause mise à la valeur
        "invalidParameter" (voir X.780).
        @param name     Nom à supprimer.
        */

        void removeName (in NameType name)
             raises     (itut_x780::ApplicationError,
                         itut_x780::DeleteError);


     };   // fin de l'interface ContainmentComponent

}; // fin du module itut_q816

#endif // fin du module #ifndef ITUT_Q816_1_IDL
```

SÉRIES DES RECOMMANDATIONS UIT-T

Série A	Organisation du travail de l'UIT-T
Série B	Moyens d'expression: définitions, symboles, classification
Série C	Statistiques générales des télécommunications
Série D	Principes généraux de tarification
Série E	Exploitation générale du réseau, service téléphonique, exploitation des services et facteurs humains
Série F	Services de télécommunication non téléphoniques
Série G	Systèmes et supports de transmission, systèmes et réseaux numériques
Série H	Systèmes audiovisuels et multimédias
Série I	Réseau numérique à intégration de services
Série J	Réseaux câblés et transmission des signaux radiophoniques, télévisuels et autres signaux multimédias
Série K	Protection contre les perturbations
Série L	Construction, installation et protection des câbles et autres éléments des installations extérieures
Série M	RGT et maintenance des réseaux: systèmes de transmission, circuits téléphoniques, télégraphie, télécopie et circuits loués internationaux
Série N	Maintenance: circuits internationaux de transmission radiophonique et télévisuelle
Série O	Spécifications des appareils de mesure
Série P	Qualité de transmission téléphonique, installations téléphoniques et réseaux locaux
Série Q	**Commutation et signalisation**
Série R	Transmission télégraphique
Série S	Equipements terminaux de télégraphie
Série T	Terminaux des services télématiques
Série U	Commutation télégraphique
Série V	Communications de données sur le réseau téléphonique
Série X	Réseaux de données et communication entre systèmes ouverts
Série Y	Infrastructure mondiale de l'information et protocole Internet
Série Z	Langages et aspects généraux logiciels des systèmes de télécommunication

Imprimé en Suisse
Genève, 2002