UNION INTERNATIONALE DES TÉLÉCOMMUNICATIONS

UIT-T X.903

(11/95)

SECTEUR DE LA NORMALISATION
DES TÉLÉCOMMUNICATIONS
DE L'UIT

RÉSEAUX POUR DONNÉES ET COMMUNICATION ENTRE SYSTÈMES OUVERTS

TRAITEMENT OUVERT RÉPARTI

TECHNOLOGIES DE L'INFORMATION – TRAITEMENT RÉPARTI OUVERT – MODÈLE DE RÉFÉRENCE: ARCHITECTURE

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. Au sein de l'UIT-T, qui est l'entité qui établit les normes mondiales (Recommandations) sur les télécommunications, participent quelque 179 pays membres, 84 exploitations de télécommunications reconnues, 145 organisations scientifiques et industrielles et 38 organisations internationales.

L'approbation des Recommandations par les Membres de l'UIT-T s'effectue selon la procédure définie dans la Résolution n° 1 de la Conférence mondiale de normalisation des télécommunications (CMNT), (Helsinki, 1993). De plus, la CMNT, qui se réunit tous les quatre ans, approuve les Recommandations qui lui sont soumises et établit le programme d'études pour la période suivante.

Dans certains secteurs de la technologie 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. Le texte de la Recommandation X.903 de l'UIT-T a été approuvé le 21 novembre 1995. Son texte est publié, sous forme identique, comme Norme internationale ISO/CEI 10746-3.

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.

RECOMMANDATIONS UIT-T DE LA SÉRIE X

RÉSEAUX POUR DONNÉES ET COMMUNICATION ENTRE SYSTÈMES OUVERTS

(Février 1994)

ORGANISATION DES RECOMMANDATIONS DE LA SÉRIE X

Domaine	Recommandations
RÉSEAUX PUBLICS POUR DONNÉES	
Services et services complémentaires	X.1-X.19
Interfaces	X.20-X.49
Transmission, signalisation et commutation	X.50-X.89
Aspects réseau	X.90-X.149
Maintenance	X.150-X.179
Dispositions administratives	X.180-X.199
INTERCONNEXION DES SYSTÈMES OUVERTS	
Modèle et notation	X.200-X.209
Définition des services	X.210-X.219
Spécifications des protocoles en mode connexion	X.220-X.229
Spécifications des protocoles en mode sans connexion	X.230-X.239
Formulaires PICS	X.240-X.259
Identification des protocoles	X.260-X.269
Protocoles de sécurité	X.270-X.279
Objets gérés de couche	X.280-X.289
Test de conformité	X.290-X.299
INTERFONCTIONNEMENT DES RÉSEAUX	
Considérations générales	X.300-X.349
Systèmes mobiles de transmission de données	X.350-X.369
Gestion	X.370-X.399
SYSTÈMES DE MESSAGERIE	X.400-X.499
ANNUAIRE	X.500-X.599
RÉSEAUTAGE OSI ET ASPECTS DES SYSTÈMES	
Réseautage	X.600-X.649
Dénomination, adressage et enregistrement	X.650-X.679
Notation de syntaxe abstraite numéro un (ASN.1)	X.680-X.699
GESTION OSI	X.700-X.799
SÉCURITÉ	X.800-X.849
APPLICATIONS OSI	
Engagement, concomitance et rétablissement	X.850-X.859
Traitement des transactions	X.860-X.879
Opérations distantes	X.880-X.899
TRAITEMENT OUVERT RÉPARTI	X.900-X.999

TABLE DES MATIÈRES

Résumé

La présente Recommandation | Norme internationale contient la spécification des caractéristiques requises pour qu'un système de traitement réparti puisse être qualifié d'ouvert: il s'agit des contraintes que doivent respecter les norme de traitement réparti ouvert (ODP, *open distributed processing*). Cette Recommandation utilise les techniques descriptives décrites dans la Recommandation X.902.

Introduction

La croissance rapide des applications réparties a fait naître le besoin d'un cadre pour coordonner la normalisation du traitement réparti ouvert (ODP). Le Modèle de référence ODP fournit ce cadre. Il établit une architecture qui permet la prise en compte de la répartition, l'interfonctionnement et la portabilité.

Le Modèle de référence pour le traitement réparti ouvert (RM-ODP, *reference model of open distributed processing*), Rec. UIT-T X.901 à X.904 | ISO/CEI 10746, repose sur des concepts précis issus des développements récents dans le domaine des traitements répartis et s'appuie, dans la mesure du possible, sur l'utilisation des techniques de description formelle pour la spécification de l'architecture.

Le Modèle de référence ODP se compose:

– de la Rec. UIT-T X.901 | ISO/CEI 10746-1: **aperçu général**: elle contient un aperçu général du Modèle de référence ODP, en précise les motivations, le domaine d'application et la justification, et propose une explication des concepts clés, ainsi qu'une présentation de l'architecture ODP. Elle explique la façon d'interpréter le Modèle de référence ODP et la manière dont il peut être utilisé, en particulier, par les auteurs de norme et les architectes de systèmes ODP. Elle contient également une classification des domaines de normalisation en matière de systèmes répartis; cette classification s'appuie sur des points de référence de conformité identifiés dans la présente Recommandation | Norme internationale. Cette partie n'est pas normative;

– de la Rec. UIT-T X.902 | ISO/CEI 10746-2: **fondations**: elle contient la définition des concepts et le cadre analytique à utiliser pour la description normalisée de systèmes de traitement répartis (arbitraires). Elle introduit les principes de la conformité aux normes ODP et la manière dont ils s'appliquent. Elle s'en tient à un niveau de détail suffisant pour étayer la présente Recommandation | Norme internationale et établir les exigences de nouvelles techniques de spécification. Cette partie est normative;

– de la Rec. UIT-T X.903 | ISO/CEI 10746-3: **architecture**: elle contient la spécification des caractéristiques d'un système réparti ouvert. Ce sont les contraintes auxquelles les normes ODP doivent se soumettre. Elle utilise les techniques descriptives de la Rec. UIT-T X.902 | ISO/CEI 10746-2. Cette partie est normative;

– de la Rec. UIT-T X.904 | ISO/CEI 10746-4: **sémantique d'architecture**: elle contient une formalisation des concepts de modélisation ODP définis dans la Rec. UIT-T X.902 | ISO/CEI 10746-2, articles 8 et 9. La formalisation s'obtient en interprétant chaque concept à partir d'éléments des différentes techniques normalisées de description formelle. Cette partie est normative.

La présente Recommandation | Norme internationale comporte une annexe (cette annexe fait partie intégrante du présent Modèle de référence).

NORME INTERNATIONALE

RECOMMANDATION UIT-T

TECHNOLOGIES DE L'INFORMATION – TRAITEMENT RÉPARTI OUVERT – MODÈLE DE RÉFÉRENCE: ARCHITECTURE

1 Domaine d'application

La présente Recommandation UIT-T | Norme internationale:

- définit la manière dont les systèmes ODP sont spécifiés, en utilisant des concepts de la Rec. UIT-T X. 902 | ISO/CEI 10746-2;
- identifie les caractéristiques auxquelles doivent se soumettre les systèmes ODP.

Elle établit un cadre de référence pour la coordination du développement de normes de systèmes ODP existantes et futures, ces normes y faisant référence.

2 Références normatives

Les Recommandations et Normes internationales suivantes contiennent des dispositions qui, par suite de la référence qui y est faite, constituent des dispositions valables pour la présente Recommandation | Norme internationale. Au moment de la publication, les éditions indiquées étaient en vigueur. Toutes Recommandations et Normes sont sujettes à révision et les parties prenantes aux accords fondés sur la présente Recommandation | Norme internationale sont invitées à rechercher la possibilité d'appliquer les éditions les plus récentes des Recommandations et Normes indiquées ci-après. Les membres de la CEI et de l'ISO possèdent le registre des Normes internationales en vigueur. Le Bureau de la normalisation des télécommunications de l'UIT tient à jour une liste des Recommandations de l'UIT-T en vigueur.

2.1 Recommandations | Normes internationales identiques

- Recommandation UIT-T X.200 (1994) | ISO/CEI 7498-1:1994, *Technologies de l'information – Interconnexion des systèmes ouverts – Modèle de référence de base: le Modèle de référence de base.*
- Recommandation UIT-T X.810 (1995) | ISO/CEI 10181-1:1996, *Technologies de l'information – Interconnexion des systèmes ouverts – Cadres de sécurité pour les systèmes ouverts: aperçu général.*
- Recommandation UIT-T X.811 (1995) | ISO/CEI 10181-2:1996, *Technologies de l'information – Interconnexion des systèmes ouverts – Cadres de sécurité pour les systèmes ouverts: authentification.*
- Recommandation UIT-T X.812 (1995) | ISO/CEI 10181-3:1996, *Technologies de l'information – Interconnexion des systèmes ouverts – Cadres de sécurité pour les systèmes ouverts: contrôle d'accès.*
- Recommandation UIT-T X.813[1)] | ISO/CEI 10181-4…[1)], *Technologies de l'information – Interconnexion des systèmes ouverts – Cadres de sécurité pour les systèmes ouverts: non-répudiation.*
- Recommandation UIT-T X.814 (1995) | ISO/CEI 10181-5:1996, *Technologies de l'information – Interconnexion des systèmes ouverts – Cadres de sécurité pour les systèmes ouverts: confidentialité.*
- Recommandation UIT-T X.815 (1995) | ISO/CEI 10181-6:1996, *Technologies de l'information – Interconnexion des systèmes ouverts – Cadres de sécurité pour les systèmes ouverts: intégrité.*
- Recommandation UIT-T X.816 (1995) | ISO/CEI 10181-7:1996, *Technologies de l'information – Interconnexion des systèmes ouverts – Cadres de sécurité pour les systèmes ouverts: audit et alarmes.*
- Recommandation UIT-T X.902 (1995) | ISO/CEI 10746-2:1996, *Technologies de l'information – Traitement ouvert réparti – Modèle de référence: fondements.*

---

1) Actuellement à l'état de projet.

2.2 Paires de Recommandations | Normes internationales équivalentes par leur contenu technique

– Recommandation X.800 du CCITT (1991), *Architecture de sécurité pour l'interconnexion en systèmes ouverts d'applications du CCITT.*

ISO 7498-2:1989, *Systèmes de traitement de l'information – Interconnexion des systèmes ouverts – Modèle de référence de base – Partie 2: architecture de sécurité.*

3 Définitions

Pour les besoins de la présente Recommandation | Norme internationale, les définitions suivantes s'appliquent.

3.1 Définitions de description

Le présent Modèle de référence emploie le terme suivant défini dans la Rec. UIT-T X.200 | ISO/CEI 7498-1:

– syntaxe de transfert.

La présente Recommandation | Norme internationale emploie les termes suivants définis dans la Rec. UIT-T X.811 | ISO/CEI 10181-2:

– déclarant;
– information d'authentification d'échange;
– mandant;
– tierce partie de confiance.

La présente Recommandation | Norme internationale emploie les termes suivants définis dans la Rec. UIT-T X.812 | ISO/CEI 10181-3:

– information de contrôle d'accès;
– fonction de décision du contrôle d'accès;
– fonction de mise en œuvre du contrôle d'accès;
– initiateur;
– cible.

La présente Recommandation | Norme internationale emploie les termes suivants définis dans la Rec. UIT-T X.813 | ISO/CEI 10181-4:

– générateur de preuve;
– utilisateur de preuve;
– vérificateur de preuve;
– source (de données non répudiables);
– destinataire (de données non répudiables);
– témoignage de non-répudiation;
– demandeur de service de non-répudiation;
– notaire.

La présente Recommandation | Norme internationale emploie les termes suivants définis dans la Rec. UIT-T X.814 | ISO/CEI 10181-5:

– données à confidentialité protégée;
– cacher;
– source;
– destinataire;
– révéler.

La présente Recommandation | Norme internationale emploie les termes suivants définis dans la Rec. UIT-T X.815 | ISO/CEI 10181-6:

– données à intégrité protégée;
– source;

– destinataire;
– protéger;
– valider.

La présente Recommandation | Norme internationale emploie les termes suivants définis dans la Rec. UIT-T X.816 | ISO/CEI 10181-7:

– fonction de collecte d'alarmes;
– fonction d'examen d'alarme;
– fonction d'examen de journal d'audit;
– fonction d'archivage de journal d'audit;
– fonction d'enregistrement d'audit;
– fonction de collecte de journal d'audit.

La présente Recommandation | Norme internationale emploie les termes suivants définis dans l'ISO/CEI 11170-1 «cadre de gestion des clés»:

– génération de clé;
– enregistrement de clé;
– certification de clé;
– invalidation de clé;
– distribution de clé;
– stockage de clé;
– archivage de clé;
– suppression de clé.

Le Modèle de référence utilise les termes définis dans la Rec. UIT-T X.902 | ISO/CEI 10746-2 et indiqués à la Figure 1.

3.2 Abréviations

Pour les besoins de la présente Recommandation | Norme internationale, les abréviations suivantes sont utilisées:

ODP Traitement réparti ouvert (*open distributed processing*).

OSI Interconnexion des systèmes ouverts (*open systems interconnection*).

4 Cadre général

Le présent Modèle de référence définit un cadre comprenant:

– cinq *points de vue*, appelés entreprise, information, traitement, ingénierie et technologie, qui fournissent des éléments de base pour spécifier des systèmes ODP;
– un *langage de point de vue* pour chaque point de vue, définissant les concepts et règles de spécification des systèmes ODP pour le point de vue correspondant;
– la spécification des *fonctions* requises par les systèmes ODP;
– des *règles de transparence* précisant comment utiliser les fonctions ODP pour assurer une transparence à la répartition.

L'architecture des systèmes ODP et la composition des fonctions sont déterminées par la combinaison du langage de traitement, du langage d'ingénierie et des règles de transparence.

4.1 Points de vue

4.1.1 Concepts

4.1.1.1 Point de vue entreprise: point de vue sur un système ODP et son environnement détaillant les objectifs, le domaine d'application et les politiques de ce système.

abstraction	gabarit de <x>	point de référence d'échange
action	gestion de communication	point de référence d'interfonctionnement
action de génération	groupe de <x>	point de référence de programmation
activité	identificateur	point de référence physique
architecture	information	point de vue
atomicité	instance	politique
classe	instanciation	position dans le temps
communication	interaction	position dans l'espace
comportement	interdiction	qualité de service
comportement d'établissement	interface	raffinement
composition	introduction	relation de liaison
configuration	invariant	rôle
contexte de désignation	liaison	signature d'interface
contrat	nom	sous-domaine
courtage	normes ODP	sous-type
création	notification	stabilité
décomposition	objet	supertype
défaillance	objet client	suppression
domaine de <x>	objet consommateur	système
domaine de désignation	objet initiateur	système ODP
données	objet producteur	terme
entité	objet serveur	traitement réparti
environnement	obligation	traitement réparti ouvert
erreur	permission	transparence
état	persistance	transparence à la répartition
faute	point de conformité	type
fil d'exécution	point de référence	

Figure 1 – Termes extraits de la Rec. UIT-T X.902 | ISO/CEI 10746-2

4.1.1.2 Point de vue information: point de vue sur un système ODP et son environnement couvrant la sémantique de l'information et son traitement.

4.1.1.3 Point de vue traitement: point de vue sur un système ODP et son environnement permettant la prise en compte de la répartition grâce à la décomposition fonctionnelle du système en objets interagissant à leurs interfaces.

4.1.1.4 Point de vue ingénierie: point de vue sur un système ODP et son environnement couvrant les mécanismes et les fonctions nécessaires pour mettre en œuvre les interactions réparties entre objets du système.

4.1.1.5 Point de vue technologie: point de vue sur un système ODP et son environnement couvrant les choix de technologie dans ce système.

4.1.2 Utilisation des points de vue

Les points de vue entreprise, information, traitement, ingénierie et technologie ont été choisis comme constituant un ensemble nécessaire et suffisant pour répondre aux besoins des normes ODP. On peut, à un niveau d'abstraction approprié, appliquer les points de vue à un système ODP complet; l'environnement définit alors le contexte dans lequel opère le système ODP considéré. On peut aussi appliquer les points de vue à des composants spécifiques d'un système ODP; l'environnement du composant inclut alors à la fois une abstraction de l'environnement du système et des autres composants du système.

NOTE – Selon l'abstraction, environnement de système et autres composants peuvent constituer un objet unique.

4.2 Langages de point de vue ODP

4.2.1 Concept

4.2.1.1 Langage de <point de vue>: définitions de concepts et de règles de spécification d'un système ODP du <point de vue> donné; ainsi: **langage d'ingénierie**: définitions de concepts et de règles de spécification d'un système ODP du point de vue ingénierie.

4.2.2 Utilisation des langages de point de vue

Le présent Modèle de référence définit un ensemble de cinq langages, chacun correspondant à l'un des points de vue définis en 4.1.1. Chaque langage est utilisé pour spécifier un système ODP dans le point de vue correspondant. Ces langages sont appelés:

- langage d'entreprise (défini à l'article 5);
- langage d'information (défini à l'article 6);
- langage de traitement (défini à l'article 7);
- langage d'ingénierie (défini à l'article 8);
- langage de technologie (défini à l'article 9).

Chaque langage utilise des concepts provenant de la Rec. UIT-T X.902 | ISO/CEI 10746-2, affine ces concepts et introduit des règles prescriptives, ainsi que des concepts additionnels, spécifiques à ce point de vue. Ces concepts additionnels sont, à leur tour, définis en utilisant des concepts de la Rec. UIT-T X. 902 | ISO/CEI 10746-2.

Une spécification de système comprend des spécifications correspondant à un ou plusieurs points de vue. Ces spécifications doivent être mutuellement cohérentes. Des règles visant à structurer de façon cohérente les spécifications de point de vue sont données à l'article 10. Le spécificateur doit démontrer par d'autres moyens que les termes sont utilisés de manière cohérente dans les spécifications. Une spécification d'une partie d'un système faisant appel à plusieurs points de vue contraindra plus les réalisations qu'une spécification faisant appel à moins de points de vue. La spécification d'objets dans un point de vue donné utilise le langage de point de vue associé ou les langages associés à d'autres points de vue. Il n'est pas nécessaire de spécifier un objet complètement de chaque point de vue pour obtenir un ensemble cohérent de spécifications de cet objet.

NOTES

1 La liste des termes provenant de la Rec. UIT-T X.902 | ISO/CEI 10746-2 apparaît dans la Figure 1.

2 Quand un terme issu de la Rec. UIT-T X.902 | ISO/CEI 10746-2 est qualifié par le nom d'un point de vue (comme dans «objet de **traitement**», par exemple) il faut l'interpréter en prenant en compte les éventuelles dispositions supplémentaires spécifiées pour ce terme dans le langage de point de vue considéré.

3 Quand dans une spécification de point de vue donnée on utilise un terme issu de la Rec. UIT-T X.902 | ISO/CEI 10746-2 non qualifié par le nom du point de vue (par exemple, «interface») et si le langage de point de vue associé impose des contraintes supplémentaires sur ce terme, il faut l'interpréter comme s'il était qualifié (c'est-à-dire, «interface de traitement»).

4.3 Fonctions ODP

4.3.1 **Fonction ODP**: fonction requise pour mettre en œuvre le traitement réparti ouvert.

4.3.2 Utilisation des fonctions ODP

Le présent Modèle de référence fournit, dans les articles 11 à 15, une description des fonctions nécessaires pour assurer le traitement réparti ouvert.

Chaque description de fonction ODP contient:

- une explication de l'utilisation de la fonction pour le traitement réparti ouvert;
- des prescriptions, concernant la structure et le comportement de la fonction, suffisantes pour assurer l'intégrité globale du Modèle de référence;
- une présentation des autres fonctions ODP dont elle dépend.

4.4 Transparences à la répartition dans ODP

4.4.1 Concepts

4.4.1.1 **Transparence d'accès**: transparence à la répartition qui masque les différences dans la représentation des données et les mécanismes d'invocation pour permettre l'interfonctionnement entre objets.

4.4.1.2 **Transparence aux défaillances**: transparence à la répartition qui masque, à un objet, les défaillances et la possibilité de reprise d'autres objets (ou de lui-même), pour garantir la tolérance aux fautes.

4.4.1.3 **Transparence à la localisation**: transparence à la répartition qui masque l'utilisation d'information de position dans l'espace lors de l'identification et de la liaison à des interfaces.

4.4.1.4 Transparence à la migration: transparence à la répartition qui masque, à un objet, la capacité d'un système à changer la localisation de cet objet. La migration est souvent utilisée pour équilibrer la charge et réduire le temps de latence.

4.4.1.5 Transparence à la relocalisation: transparence à la répartition qui masque la relocalisation d'une interface aux interfaces qui lui sont liées.

4.4.1.6 Transparence à la duplication: transparence à la répartition qui masque la mise en œuvre d'une interface par un groupe d'objets de comportement compatible. La duplication est souvent utilisée pour augmenter les performances et la disponibilité.

4.4.1.7 Transparence à la persistance: transparence à la répartition qui masque, à un objet, la désactivation et la réactivation d'autres objets (ou de lui-même). La désactivation et la réactivation sont souvent utilisées pour maintenir la persistance d'un objet lorsqu'un système ne peut pas lui assurer, de manière continue, les fonctions de traitement, de stockage et de communication.

4.4.1.8 Transparence aux transactions: transparence à la répartition qui masque la coordination des activités au sein d'une configuration d'objets pour en assurer la cohérence.

4.4.2 Utilisation de la transparence à la répartition

La transparence à la répartition est un besoin important des utilisateurs de systèmes répartis. Ce Modèle de référence définit un ensemble de transparences à la répartition qui rend possible la réalisation de systèmes ODP transparents à la répartition du point de vue de leurs utilisateurs. Cette transparence est sélective; le Modèle de référence comprend des règles de sélection et de combinaison de transparences à la répartition dans les systèmes ODP.

Le présent Modèle de référence comporte, pour chaque transparence à la répartition définie aux 4.4.1.1 à 4.4.1.8, les définitions:

- d'un schéma, pour exprimer les exigences d'une transparence particulière;
- d'un raffinement, pour transformer une spécification contenant des prescriptions de la transparence à la répartition donnée en une spécification qui réalise explicitement le masquage impliqué par cette transparence.

NOTES

1 Dans certains cas (pour la transparence d'accès, par exemple), le schéma est vide; dans d'autres (pour la transparence aux transactions, par exemple), le schéma contient un ou plusieurs paramètres dictant la forme précise de la transparence requise.

2 Le raffinement implique habituellement l'introduction d'un comportement supplémentaire, y compris l'utilisation dans la spécification d'une ou de plusieurs fonctions ODP.

Les raffinements de l'article 16 sont décrits à un niveau de détail suffisant pour garantir l'intégrité globale du Modèle de référence.

4.5 Normes dérivées du cadre général

Le Modèle de référence fournit un cadre général pour la définition de nouvelles normes et l'utilisation en tant que normes ODP de celles déjà existantes.

Les normes ODP incluent:

- des normes de composants de systèmes ODP;
- des normes de composition de composants de systèmes ODP;
- des normes de modélisation et de spécification des systèmes ODP.

Les normes ODP:

- utilisent le langage d'entreprise pour spécifier les politiques;
- utilisent le langage d'information pour spécifier l'utilisation et l'interprétation cohérentes de l'information dans et entre les normes;
- utilisent le langage de traitement pour spécifier la configuration et le comportement des interfaces;
- utilisent le langage d'ingénierie pour spécifier les infrastructures dont elles ont besoin;
- utilisent le langage de technologie pour spécifier la conformité aux spécifications internationales, privées ou consensuelles.

Les normes de méthodologie, de modélisation, de programmation, de mise en œuvre et de test des systèmes ODP utilisent le cadre général dans son ensemble.

Les normes ODP peuvent se baser sur un sous-ensemble du présent Modèle de référence (par exemple, en excluant certaines formes d'interaction, des fonctions particulières ou des transparences). Ces normes peuvent également étendre le Modèle de référence, à condition que les extensions qu'elles introduisent ne modifient ni ne contredisent ses dispositions. Les extensions mettent en relation les nouveaux termes avec les termes définis dans le Modèle de référence: par exemple, en introduisant de nouveaux types et de nouvelles règles de typage.

Les normes ODP se conforment aux règles définies dans le Modèle de référence.

4.6 Conformité

On utilise les langages d'entreprise, d'information, de traitement et d'ingénierie pour spécifier les exigences de conformité des systèmes ODP. On peut utiliser le langage de technologie pour exprimer la conformité aux normes ODP dans les systèmes ODP. Toutes les interfaces définies comme points de conformité ont des spécifications d'information pour permettre l'interprétation des interactions présentes à ces interfaces. Les langages de traitement et d'ingénierie spécifient les règles d'identification des points de conformité.

Un système ODP est conforme à une norme ODP s'il satisfait aux exigences de conformité de cette norme.

5 Langage d'entreprise

Le langage d'entreprise comprend des concepts, des règles et des structures pour la spécification d'un système ODP du point de vue entreprise.

Une spécification d'entreprise définit les objectifs, le domaine d'application et les politiques d'un système ODP.

Dans le présent Modèle de référence, les règles et concepts concernant le point de vue entreprise sont limitées et portent sur le domaine d'application et la nature des spécifications d'entreprise.

5.1 Concepts

Le langage d'entreprise utilise les concepts de la Rec. UIT-T X.902 | ISO/CEI 10746-2 et ceux définis ici, soumis aux règles du 5.2.

5.1.1 Communauté: configuration d'objets constituée pour atteindre un objectif. L'objectif s'exprime dans les termes d'un contrat qui spécifie la manière dont l'objectif peut être atteint.

5.1.2 Fédération de <X>: communauté de domaines de <X>.

5.2 Règles de structuration

Une spécification d'entreprise définit, grâce au langage d'entreprise, les objectifs, le domaine d'application et les politiques d'un système ODP en fonction de chacun des éléments suivants:

- des rôles joués par le système;
- des activités entreprises par le système;
- des déclarations de politique du système, y compris celle ayant un rapport avec les contrats d'environnement.

Dans une spécification d'entreprise, un système ODP et l'environnement dans lequel il opère sont représentés sous forme de communauté. A un certain niveau de description, le système ODP est représenté sous forme d'un objet d'entreprise au sein de la communauté. Les objectifs et le domaine d'application du système ODP sont définis en termes de rôles qu'il joue au sein de la communauté à laquelle il appartient, et de déclarations de politique. Une communauté est définie en termes:

- d'objets d'entreprise comprenant la communauté;
- de rôles joués par chacun de ces objets;
- de politiques régulant les interactions entre les objets d'entreprise jouant des rôles;
- de politiques régulant la création, l'utilisation et la suppression de ressources, par des objets d'entreprise jouant des rôles;
- de politiques régulant la configuration d'objets d'entreprise et l'attribution de rôles à des objets d'entreprise;
- de politiques relatives aux contrats d'environnement qui régulent le système.

Un rôle est défini en termes de permissions, d'obligations, d'interdictions et de comportement de l'objet d'entreprise jouant un rôle. Un objet d'entreprise peut occuper un ou plusieurs rôles au sein d'une communauté, et ces rôles sont déterminés par le contrat sur lequel repose la communauté. Un objet d'entreprise peut tenir simultanément plusieurs rôles dans des communautés différentes, sous réserve des dispositions contenues dans les contrats des communautés impliquées. Les interactions entre les objets d'entreprise jouant des rôles dans différentes communautés peuvent être considérées comme des interactions entre ces communautés.

NOTE 1 – Exemples de rôles: administrateur de politique, président, fournisseur de services, propriétaire, gestionnaire, actionnaire, consommateur.

NOTE 2 – Exemples de contrats d'environnement dans les spécifications d'entreprise: exigences de sûreté, exigences légales, règles de conduite.

NOTE 3 – Dans une spécification d'entreprise, le terme «objet de <x>», où <x> est un rôle, signifie «objet d'entreprise tenant un rôle <x>»; lorsqu'un objet d'entreprise joue plusieurs rôles, les noms peuvent être concaténés, par exemple, «objet chauffeur propriétaire».

Lorsqu'il joue un rôle, l'objet devient assujetti à des permissions, des obligations et des interdictions, par délégation ou par transfert. Dans certains rôles, les objets sont autorisés à changer de politique. Il y a cinq types fondamentaux d'action en matière de contrat:

- un agent contracte une obligation vis-à-vis d'un autre agent (il doit être autorisé à un moment donné à contracter cette obligation);
- un agent remplit une obligation vis-à-vis d'un autre agent;
- un agent abandonne une obligation vis-à-vis d'un autre agent;
- un agent obtient d'un autre agent l'autorisation d'effectuer une action qu'il ne lui était pas permis d'effectuer antérieurement;
- on interdit à un agent d'effectuer une action qu'il était antérieurement autorisé à effectuer.

NOTE 4 – Obtenir l'autorisation d'effectuer une action performative, c'est-à-dire quand un objet dans un rôle subordonné obtient le droit de donner à son tour des autorisations ou de créer d'autres obligations pour le compte d'un objet dans un rôle supérieur, constitue un cas particulier important. Ceci correspond à la notion d'agence ou de délégation.

Les obligations incluent la comptabilisation et le coût liés à l'utilisation des ressources. La facturation et le paiement sont modélisés suivant la réaffectation des ressources entre les objets, conformément aux rôles qu'ils tiennent.

Une ressource peut être consommable ou non. Une ressource consommable se détruit après un certain temps d'utilisation. Dans la fédération de <x>, l'objectif définit les ressources que chaque domaine de <x> de la fédération partage avec les autres membres. L'objectif peut laisser à chaque domaine un degré d'autonomie défini pour l'utilisation de ses propres ressources. Le comportement établissant une fédération de <x> peut laisser à chaque domaine de <x> participant le choix de faire partie ou non de la fédération.

5.3 Conformité et points de référence

Les déclarations de conformité dans le langage d'entreprise exigent que le comportement d'un système ODP soit conforme à un ensemble particulier d'objectifs et de politiques.

Un réalisateur se déclarant conforme doit énumérer les points de référence d'ingénierie donnant accès au système ainsi que les spécifications d'ingénierie, de traitement et d'information qui s'appliquent à ces points. De ce fait, les points de référence identifiés deviennent des points de conformité. L'ensemble des interactions observables à ces points de conformité peut alors être interprété en termes de langage d'entreprise pour vérifier que la spécification d'entreprise n'est pas transgressée.

Les spécifications d'entreprise peuvent s'appliquer aux quatre classes de point de référence (points de référence de programmation, physiques, d'interfonctionnement et d'échange) énumérées dans la Rec. UIT-T X.902 | ISO/CEI 10746-2.

6 Langage d'information

Le langage d'information comprend des concepts, des règles et des structures pour la spécification d'un système ODP du point de vue information.

Une spécification d'information définit la sémantique de l'information et la sémantique du traitement de l'information dans un système ODP.

Dans le présent Modèle de référence, les règles et concepts concernant le point de vue information sont limitées et portent sur le domaine d'application et la nature des spécifications d'information.

6.1 Concepts

Le langage d'information utilise les concepts de la Rec. UIT-T X.902 | ISO/CEI 10746-2 et ceux définis ici, soumis aux règles du 6.2.

6.1.1 **Schéma d'invariant**: ensemble de prédicats portant sur un ou plusieurs objets d'information qui doivent toujours être vrais. Les prédicats déterminent les états possibles et les changements d'état des objets auxquels ils s'appliquent.

NOTE – Un schéma d'invariant correspond à la spécification des types d'un ou de plusieurs objets d'information devant toujours être satisfaits quel que soit le comportement que pourraient exhiber les objets.

6.1.2 **Schéma statique**: spécification de l'état d'un ou de plusieurs objets d'information, à un instant donné dans le temps. Le schéma statique doit respecter les contraintes de tout schéma d'invariant.

NOTE – Un schéma statique correspond donc à la spécification des types d'un ou de plusieurs objets d'information à un instant précis. Ces types sont des sous-types des types spécifiés dans le schéma d'invariant.

6.1.3 **Schéma dynamique**: spécification des changements d'état qui sont autorisés pour un ou plusieurs objets d'information. Le schéma dynamique doit respecter les contraintes de tout schéma d'invariant.

NOTES

1 Dans un système d'information le comportement peut être modélisé sous la forme de transitions d'un schéma statique à un autre. Cela correspond à la reclassification d'instances d'un type à un autre.

2 Dans le langage d'information, un changement d'état impliquant un ensemble d'objets peut être considéré comme une interaction entre ces objets. Il n'est pas nécessaire que les objets impliqués dans l'interaction changent tous d'état; certains peuvent n'intervenir qu'en lecture seulement.

6.2 Règles de structuration

Une spécification d'information définit la sémantique de l'information et la sémantique du traitement de l'information dans un système ODP en termes de configuration d'objets d'information, de comportement de ces objets et des contrats d'environnement du système.

Un gabarit d'objet d'information fait référence à un schéma statique, un schéma d'invariant et un schéma dynamique. Les relations entre les objets d'information peuvent être modélisées comme faisant partie de l'état de ces objets d'information. Les objets d'information sont soit atomiques, soit représentés sous forme d'une composition d'objets d'information composants. L'état d'un objet composite est représenté par la combinaison des états des objets d'information le composant. Un gabarit d'objet d'information atomique représente un concept primitif que l'on ne peut décomposer plus avant au niveau d'abstraction considéré. Un objet d'information composite représente un concept dérivé exprimé selon d'autres concepts. Etant donné que la composition d'objets inclut l'encapsulation, un objet d'information composant d'un objet composite ne peut pas être composant d'un autre. Par conséquent, les objets d'information résultant de l'instanciation d'un gabarit d'objet d'information composite n'existent qu'en tant qu'élément de l'objet composite instancié et n'ont de signification que dans cette composition.

Les changements d'état autorisés spécifiés par un schéma dynamique peuvent inclure la création de nouveaux objets d'information et la suppression d'objets impliqués dans un schéma dynamique. Ces états sont soumis à des contraintes temporelles et d'ordonnancement.

NOTE 1 – Le résultat d'un accès à un état d'un ou plusieurs objets d'information peut être modélisé comme la création d'un nouvel objet d'information.

Dans une spécification d'information, la configuration et le comportement des objets d'information ne se prêtent pas forcément à une répartition immédiate (par exemple, les interactions décrites dans une spécification d'information peuvent faire abstraction de défaillances ou de localisations déterminées).

NOTE 2 – Si une notation d'information utilise le concept d'interface, les interfaces définies dans une spécification ne constituent pas forcément des points de référence; c'est pourquoi ces interfaces ne doivent pas forcément apparaître dans une réalisation.

6.3 Conformité et points de référence

Les déclarations de conformité des spécifications d'information exigent la conformité du comportement d'un système ODP à un ensemble particulier de schémas statiques, dynamiques et d'invariant.

Un réalisateur se déclarant conforme doit énumérer les points de référence d'ingénierie donnant accès au système ainsi que les spécifications d'ingénierie et de traitement qui s'appliquent à ces points. De ce fait, les points de référence identifiés deviennent des points de conformité. Les interactions observables à ces points de référence peuvent alors être interprétées en termes de langage d'information pour vérifier que les schémas d'invariant, statique et dynamique sont utilisés de manière cohérente.

Les spécifications d'information peuvent être appliquées aux quatre classes de points de référence (de programmation, physique, d'interfonctionnement et d'échange) identifiées dans la Rec. UIT-T X.902 | ISO/CEI 10746-2.

7 Langage de traitement

Le langage de traitement comprend des concepts, des règles et des structures de spécification d'un système ODP du point de vue traitement.

Une spécification de traitement définit la décomposition fonctionnelle d'un système ODP en objets interagissant à des interfaces.

Dans le point de vue traitement, les applications et les fonctions ODP correspondent à des configurations d'objets de traitement interagissant.

7.1 Concepts

Le langage de traitement utilise les concepts de la Rec. UIT-T X.902 | ISO/CEI 10746-2 et ceux définis ici, soumis aux règles de structuration du 7.2.

7.1.1 **Signal**: action atomique partagée constituant une communication d'un objet initiateur vers un objet répondeur.

NOTE – Un signal est une interaction.

7.1.2 **Opération**: interaction entre un objet client et un objet serveur qui correspond soit à une interrogation, soit à une annonce.

7.1.3 **Annonce**: interaction – l'**invocation** – initialisée par un objet client aboutissant à l'acheminement d'informations d'un objet client vers un objet serveur, et demandant l'exécution d'une fonction par cet objet serveur.

7.1.4 **Interrogation**: interaction consistant en:

- une première interaction – l'**invocation** – initialisée par un objet client aboutissant à l'acheminement d'informations de cet objet client vers un objet serveur, et demandant l'exécution d'une fonction par cet objet serveur, suivie de:
- une seconde interaction – la **terminaison** – initialisée par l'objet serveur, aboutissant à l'acheminement d'informations de l'objet serveur vers l'objet client en réponse à l'invocation.

NOTE – Dans les interrogations, invocations et terminaisons vont toujours de pair. Les annonces ne comportent pas de terminaison. Une invocation d'opération ne peut être suivie d'une séquence de terminaisons.

7.1.5 **Flux**: abstraction d'une séquence d'interactions aboutissant à l'acheminement d'informations d'un objet producteur vers un objet consommateur.

NOTE – Un flux peut être utilisé, par exemple, comme abstraction de la structure exacte d'une séquence d'interactions ou, d'une interaction à flux continu, y compris d'une interaction à flux continu analogique.

7.1.6 **Interface signal**: interface dont toutes les interactions sont des signaux.

7.1.7 **Interface opération**: interface dont toutes les interactions sont des opérations.

7.1.8 **Interface flux**: interface dont toutes les interactions sont des flux.

7.1.9 **Gabarit d'objet de traitement**: gabarit d'objet qui comprend un ensemble de gabarits d'interface de traitement que l'objet peut instancier, une spécification de comportement et une spécification de contrat d'environnement.

7.1.10 **Gabarit d'interface de traitement**: gabarit d'interface pour une interface signal, une interface flux ou une interface opération. Un gabarit d'interface de traitement comprend une signature d'interface signal, une signature d'interface flux ou une signature d'interface opération, selon le cas, une spécification de comportement et une spécification de contrat d'environnement.

7.1.11 **Signature d'interface signal**: signature d'interface pour une interface signal comprenant un ensemble fini de gabarits d'action pour chacun des types de signal de l'interface. Chaque gabarit d'action comporte, pour l'objet qui l'instancie, le nom du signal, le nombre, les noms et les types de ses paramètres et une indication de causalité (initiateur ou répondeur, mais pas les deux à la fois).

7.1.12 Signature d'interface opération: signature d'interface pour une interface opération comprenant, pour l'objet qui l'instancie, un ensemble de signatures d'interrogation et d'annonce pour chacun des types d'opération de l'interface, ainsi qu'une indication de causalité (client ou serveur, mais pas les deux à la fois) pour l'interface dans son ensemble.

Chaque **signature d'annonce** correspond à un gabarit d'action comportant le nom de l'invocation et le nombre, les noms et les types de ses paramètres.

Chaque **signature d'interrogation** comprend un gabarit d'action avec les éléments suivants:

- le nom de l'invocation;
- le nombre, les noms et les types de ses paramètres;
- un ensemble fini non vide de gabarits d'action, pour chacun des types de terminaison possibles de l'invocation et contenant le nom de la terminaison ainsi que le nombre, les noms et les types de ses paramètres.

7.1.13 Signature d'interface flux: signature d'interface pour une interface flux comprenant un ensemble fini de gabarits d'action pour chacun des types de flux de l'interface flux. Chaque gabarit d'action d'un flux contient, en fonction de l'objet qui l'instancie, le nom du flux, le type d'informations du flux ainsi qu'une indication de causalité du flux (à savoir producteur ou consommateur, mais pas les deux à la fois).

NOTES

1 L'expression «signature d'interface complémentaire à *X*», où *X* représente lui-même une signature d'interface, décrit une signature d'interface identique à celles de *X* mais de causalités opposées.

2 De nombreux langages de définition d'interface (IDL, *interface definition languages*) prennent en compte uniquement les gabarits d'action d'une signature et sont dépendants du contexte dans lequel l'IDL est utilisé pour déterminer la causalité à appliquer.

7.1.14 Objet de liaison: objet de traitement assurant la liaison entre un ensemble d'autres objets de traitement.

NOTE – Les objets de liaison relèvent de dispositions particulières (voir 7.2.3).

7.2 Règles de structuration

Une spécification de traitement décrit la décomposition fonctionnelle d'un système ODP dans des termes transparents à la répartition, à savoir:

- une configuration d'objets de traitement (incluant des objets de liaison);
- les actions internes de ces objets;
- les interactions qui se produisent entre ces objets;
- les contrats d'environnement de ces objets et leurs interfaces.

Le langage de traitement définit des règles auxquelles doivent se soumettre les spécifications de traitement. Ces règles comprennent:

- des règles d'interaction (voir 7.2.2), de liaison (voir 7.2.3) et de typage (voir 7.2.4) qui assurent un interfonctionnement transparent à la répartition;
- de règles de gabarit (voir 7.2.5) qui s'appliquent à tous les objets de traitement;
- des règles de défaillance (voir 7.2.6) qui s'appliquent à tous les objets de traitement et qui identifient les points potentiels de défaillances des activités de traitement.

Les règles de portabilité (voir 7.2.7) fournissent des guides pour les auteurs de normes de portabilité ODP.

Une spécification de traitement définit un ensemble initial d'objets de traitement et leur comportement. La configuration change selon que les objets de traitement:

- instancient d'autres objets de traitement;
- instancient d'autres interfaces de traitement;
- exécutent des actions de liaison;
- invoquent des fonctions de contrôle sur des objets de liaison;
- suppriment des interfaces de traitement;
- suppriment des objets de traitement.

7.2.1 Règles de désignation

Chaque type de nom défini dans le langage de traitement est associé à un contexte associé, de la façon suivante:

- un nom de signal d'une signature d'interface signal est un identificateur dans le contexte de cette signature;
- un nom de flux d'une signature d'interface flux est un identificateur dans le contexte de cette signature;
- un nom d'invocation d'une signature d'interface opération est un identificateur dans le contexte de cette signature;
- un nom de terminaison d'une signature d'interface opération est un identificateur dans le contexte du gabarit d'opération dans lequel il apparaît;
- le nom d'un paramètre d'un gabarit de signal est un identificateur dans le contexte de ce gabarit;
- le nom d'un paramètre d'un gabarit d'invocation d'une signature d'interface opération est un identificateur dans le contexte de ce gabarit;
- le nom d'un paramètre d'un gabarit de terminaison d'une signature d'interface opération est un identificateur dans le contexte de ce gabarit;
- le nom d'un paramètre d'un gabarit de signal d'une signature d'interface signal est un identificateur dans le contexte de ce gabarit.

NOTE 1 – Les noms de signaux sont distincts dans n'importe quelle signature d'interface signal, mais les signaux de différentes signatures peuvent avoir le même nom, et ainsi de suite.

Un identificateur d'interface de traitement est non ambigu dans son propre contexte (c'est-à-dire qu'il ne peut pas être associé à plusieurs interfaces de traitement dans ce contexte). Le choix des contextes pour les identificateurs d'interface de traitement est une question de conception du langage et, par conséquent, n'est pas couvert par le domaine d'application du présent Modèle de référence. Ce Modèle de référence n'impose pas de contraintes en ce qui concerne l'étendue des contextes pour les identificateurs d'interface de traitement. C'est pourquoi, on ne peut pas préjuger:

- de l'étendue des contextes de désignation des identificateurs d'interface de traitement (par exemple, des hypothèses selon lesquelles ils seraient associés à des structures de langage d'ingénierie telles que les nœuds ou les domaines de communication);
- du caractère unique des identificateurs d'interface de traitement (c'est-à-dire que les synonymes sont autorisés);
- de l'idendité des interfaces dénotées par un même identificateur d'interface de traitement indépendamment du contexte dans lequel cet identificateur est utilisé (les noms peuvent ne pas être globaux).

NOTE 2 – Une notation de traitement particulière peut ne pas comporter de termes explicites dénotant des identificateurs d'interface de traitement; dans cette notation, les identificateurs d'interface de traitement sont implicites mais ils demeurent sujet aux règles ci-dessus.

7.2.2 Règles d'interaction

Chaque interaction d'objet de traitement se produit à l'une de ses interfaces de traitement. Le langage de traitement impose des contraintes sur le comportement autorisé à une interface de traitement. Une interaction au niveau d'une interface qui n'a pas été liée engendre une défaillance. Les règles de liaison (voir 7.2.3) imposent des contraintes sur la manière dont les interfaces peuvent être liées.

La partie interaction du langage de traitement comprend trois modèles d'interaction. Chaque modèle d'interaction est associé à une sorte d'interface de traitement déterminée:

- signaux et interfaces signal;
- flux et interfaces flux;
- opérations et interfaces opération.

En plus des différents types d'interaction pris en charge, les modèles d'interaction se distinguent par leurs propriétés en matière de défaillance. Les participants à un flux ou à une opération peuvent avoir une vue incohérente d'une interaction à différents moments, notamment après que des défaillances se soient produites. Contrairement aux flux et aux opérations, le concept de défaillance partielle d'un signal n'existe pas – un signal réussit ou échoue de manière identique pour les deux participants de l'interaction.

7.2.2.1 Règles d'interaction pour les signaux

Un objet de traitement offrant une interface signal d'un type d'interface signal donné:

- génère des signaux ayant une causalité initiateur dans la signature de l'interface;
- réagit aux signaux ayant une causalité répondeur dans la signature de l'interface.

7.2.2.2 Règles d'interaction pour les flux

Un objet de traitement offrant une interface flux:

- génère des flux ayant une causalité producteur dans la signature de l'interface;
- reçoit des flux ayant une causalité consommateur dans la signature de l'interface.

7.2.2.3 Règles d'interaction pour les opérations

Un objet client utilisant une interface opération invoque les opérations nommées dans la signature de l'interface. Un objet serveur offrant une interface opération attend les opérations nommées dans la signature d'interface. Dans le cas d'une interrogation, le serveur répond à l'invocation en initialisant l'une des terminaisons nommées pour l'opération dans la signature d'interface serveur. Le client attend l'une des terminaisons nommées pour l'opération dans la signature d'interface client. La durée d'une opération est arbitraire, sauf à être spécifiée différemment dans les contrats d'environnement applicables aux objets et aux interfaces concernées.

NOTE – Si un fil d'exécution client invoque une chaîne d'interrogations, l'apariement des invocations et des terminaisons garantit que le client obtiendra des réponses du serveur dans le même ordre que celui de ses invocations. Si le client invoque une chaîne d'annonces (ou une chaîne contenant à la fois des annonces et des invocations), l'ordre dans lequel le serveur répond à une annonce n'est pas déterminé, sauf à être spécifié de manière implicite dans les contrats d'environnement applicables à l'interaction. Il n'y a pas de garantie d'ordonnancement pour les interrogations ou les annonces situées dans des activités différentes issues d'une même action de division.

7.2.2.4 Règles de paramétrage

Les paramètres des signaux, des invocations et des terminaisons peuvent inclure des identificateurs pour les interfaces de traitement et les types de signature d'interface de traitement.

NOTE 1 – La possibilité de spécifier des signatures d'interface de traitement comme paramètre d'opération fait du système de signature d'interface de traitement un système de ce type d'ordre supérieur. L'utilisation explicite de type de signature en paramètre d'opération est nécessaire, par exemple, dans le courtage, où les paramètres d'opérations d'importation et d'exportation incluent des types de signature:

courtier.importation (T: Type, ...) : (service: T) -> échec (raison: chaîne)

courtier.exportation (T: Type, service: T) : (...) -> échec (raison: chaîne)

Cela introduit un besoin de contrôle dynamique de la relation de sous-typage de signature d'interface (voir 7.2.5.1).

Un paramètre formel qui est un identificateur d'interface de traitement est qualifié par un type de signature d'interface de traitement. Le paramètre réel correspondant doit faire référence à une interface ayant ce type de signature d'interface (ou l'un de ses sous-types). Le paramètre réel peut uniquement être utilisé comme s'il référençait une interface de traitement avec le même type de signature que le paramètre formel (ou l'un des supertypes du paramètre formel). Après une interaction, l'initiateur et le répondeur peuvent référencer l'interface identifiée, bien qu'elle puisse être référencée par des identificateurs d'interface de traitement différents.

NOTE 2 – Cette règle empêche l'utilisateur de l'interface référencée par le paramètre réel d'exécuter des interactions autres que celles du type de signature d'interface formelle, même si l'interface référencée par le paramètre réel est un sous-type associé au paramètre formel.

7.2.2.5 Flux, opérations et signaux

Flux et opérations peuvent être définis en termes de signaux. Ceci permet de définir la sémantique de liaisons multipoints, de caractéristiques de qualité de service de bout en bout, ainsi que de liaisons composites entre des interfaces de différentes sortes (comme, par exemple, des liaisons entre interfaces flux et interfaces opération).

La définition des flux en termes de signaux dépend du détail des interactions modélisées dans la spécification de l'interface concernée et sort du cadre du présent Modèle de référence.

Les opérations peuvent être modélisées par des signaux en faisant correspondre des interfaces signal aux interfaces opération client et serveur impliquées:

- dans une interface signal correspondant à une interface opération client, il y a émission d'un signal (**envoi d'invocation**) correspondant à chaque invocation avec les mêmes paramètres et, dans le cas d'une interface contenant des interrogations, émission d'un signal (**réception de terminaison**) correspondant à chaque terminaison possible avec les mêmes paramètres que cette terminaison;
- dans une interface signal correspondant à une interface opération serveur, il y a émission d'un signal (**réception d'invocation**) correspondant à chaque invocation avec les mêmes paramètres et, dans le cas d'une interface contenant des interrogations, émission d'un signal (**envoi de terminaison**) correspondant à chaque terminaison possible avec les mêmes paramètres que cette terminaison.

Cela crée une équivalence entre l'ensemble de signaux résultant et l'ensemble des invocations et terminaisons des interfaces opération décrites.

7.2.3 Règles de liaison

Dans le présent Modèle de référence, le terme liaison fait référence aux actions de liaison. L'utilisation de telles actions est appelée **liaison explicite**. Il existe deux types d'actions de liaison: les **actions de liaison primitives** et les **actions de liaison composites**.

Une action de liaison primitive lie directement deux objets de traitement. Une action de liaison composite peut se définir en termes d'actions de liaison primitives liant plusieurs objets de traitement via un objet de liaison. La présence d'un objet de liaison dans une liaison de traitement permet d'exprimer le contrôle de la configuration et de la qualité de service (voir 7.2.3.3).

Dans les notations dépourvues de termes exprimant les actions de liaison, la liaison est implicite. Le présent Modèle de référence ne définit des **liaisons implicites** que pour des interfaces opération. Définir une liaison implicite pour d'autres sortes d'interface n'est pas évident, car il est difficile de préciser dans quelles conditions doit s'établir la liaison, en particulier, de définir chez qui réside la responsabilité de l'établissement d'une liaison. Les informations complémentaires nécessaires peuvent être fournies dans une action de liaison explicite.

7.2.3.1 Règles de liaison implicite pour les interfaces opération serveur

Si une invocation effectuée par un objet client fait référence à une interface opération serveur à laquelle le client n'est pas lié, la mise en place d'une liaison implicite est requise. Si une interface opération client appropriée liée au serveur n'existe pas, l'établissement d'une liaison implicite nécessite la procédure suivante:

- création d'une interface opération client de type de signature complémentaire à l'interface serveur;
- liaison de l'interface opération client à l'interface opération serveur;
- invocation de l'objet serveur à l'aide de l'interface opération client;
- lorsque l'opération est terminée, suppression facultative de l'interface client.

7.2.3.2 Règles de liaison primitive

Les actions de liaison primitives permettent de rattacher une interface de l'objet qui lance l'action à une autre interface (d'un autre objet ou de lui-même). L'action de liaison est paramétrée par deux identificateurs, un pour chaque interface. Les conditions préalables d'une action de liaison sont les suivantes: les deux interfaces sont du même type (signaux, flux, opérations), elles sont de causalité complémentaire et leurs types de signature sont complémentaires.

La liaison primitive établit une liaison entre les deux interfaces concernées, ou échoue.

La suppression d'une interface ayant été liée à une autre interface par une action de liaison primitive entraîne la suppression de la liaison en même temps que celle de l'interface.

7.2.3.3 Règles de liaison composite

Les actions de liaison composites permettent de lier un ensemble d'interfaces à l'aide d'un objet de liaison prenant en charge la liaison. Excepté pour les dispositions du présent paragraphe, un objet de liaison est un objet de traitement ordinaire. Dans un gabarit d'objet de liaison, la spécification du comportement est exprimée en termes d'un ensemble de paramètres de rôle formel, chacun étant associé à un gabarit d'interface.

Les actions de liaison composites sont paramétrées par un gabarit d'objet de liaison et un ensemble d'interfaces, les interfaces qui sont à lier pour assurer l'interaction.

Pour chaque rôle formel du gabarit d'objet de liaison, les conditions préalables sont les suivantes:

- le paramètre d'interface correspondant doit être de la même sorte (signal, flux, opération) que le gabarit d'interface associé au rôle formel du gabarit d'objet de liaison;
- le paramètre d'interface correspondant doit avoir une causalité complémentaire au gabarit d'interface associé au rôle formel du gabarit d'objet de liaison;
- le paramètre d'interface correspondant doit être un sous-type du type de signature du gabarit d'interface associé au rôle formel du gabarit d'objet de liaison.

Une action de liaison composite comprend les étapes suivantes:

- instanciation d'un objet de liaison passé en paramètre de l'action;
- instanciation pour chaque rôle formel du gabarit d'objet de liaison d'une interface correspondante au sein du nouvel objet de liaison;
- liaison primitive par l'objet de liaison des interfaces ainsi créées avec les interfaces correspondantes passées en paramètre de l'action de liaison composite initiale;
- instanciation d'un ensemble d'interfaces de contrôle et retour des identificateurs de ces interfaces comme résultats de l'action de liaison composite (les identificateurs des interfaces de contrôle deviennent partie de l'état de l'objet qui a exécuté l'action – cet objet peut, par la suite, transmettre ces identificateurs en interagissant avec d'autres objets de traitement).

Les interfaces de contrôle d'un objet de liaison offrent tout ou partie des fonctions suivantes:

- supervision de l'utilisation de la liaison;
- supervision des modifications de la liaison;
- autorisation des modifications de la liaison;
- changement des participants à la liaison;
- modification du schéma de communication permis par la liaison;
- changement de la qualité de service de la liaison;
- suppression de la liaison.

L'effet sur un objet de liaison de la suppression d'une liaison est déterminé par le comportement de l'objet de liaison.

7.2.4 Règles de typage

Le présent Modèle de référence spécifie les règles de typage de signature pour les interfaces de traitement. Les règles de sous-typage de signature définissent les conditions minimales de remplacement d'une interface par une autre. Les règles sont basées sur la sémantique d'interaction des interfaces de traitement (interfaces signal, flux et opération). Elles suffisent pour garantir qu'une interface remplaçante interprète de manière cohérente la structure des interactions.

Les règles de sous-typage de signature pour les interfaces possédant une sémantique d'interaction alternative peuvent être définies en termes de signaux; ces définitions peuvent être introduites par une norme ODP.

7.2.4.1 Règles de sous-typage de signature pour les interfaces signal

La définition du sous-typage de signature d'interface signal est donnée dans l'Annexe A. Pour les types d'interface signal qui ne sont pas définis récursivement, les règles sont résumées ci-après.

Un type de signature d'interface signal *X* est un sous-type d'un type de signature d'interface signal *Y* si les conditions suivantes sont satisfaites:

- pour chaque signature de signal dans *Y* avec une causalité initiateur, il y a dans *X* une signature de signal avec causalité initiateur, avec le même nom, les mêmes nombre et noms de paramètres, et chaque type de paramètre dans *X* est un sous-type du paramètre correspondant dans *Y*;
- pour chaque signature de signal dans *X* avec une causalité répondeur, il y a dans *Y* une signature de signal avec causalité répondeur, avec le même nom, les mêmes nombre et noms de paramètres, et chaque type de paramètre dans *Y* est un sous-type du paramètre correspondant dans *X*.

7.2.4.2 Règles de sous-typage de signature pour les interfaces flux

Les règles de sous-typage pour la signature d'interface flux dépendent des détails des interactions abstraites dans la définition des interfaces flux concernées. Ces détails déterminent si les règles de sous-typage permettent ou non d'effectuer une mise en correspondance entre l'ensemble des deux interfaces. De fait, le présent Modèle de référence ne peut définir complètement des règles de sous-typage de signature d'interface flux. Il ne définit donc que des règles de sous-typage incomplètes. Les contraintes de sous-typage de signature d'interface flux sont données dans l'Annexe A. Pour les types d'interface flux qui ne sont pas définis récursivement, les contraintes sont résumées ci-après.

Une interface flux *X* est un sous-type de signature d'interface flux *Y* si les conditions suivantes sont satisfaites pour tous les flux portant le même nom:

- si la causalité est une causalité producteur, le type d'information dans *X* est un sous-type du type d'information dans *Y*;
- si la causalité est une causalité consommateur, le type d'information dans *Y* est un sous-type du type d'information dans *X*.

7.2.4.3 Règles de sous-typage de signature pour les interfaces opération

La définition du sous-typage de signature d'interface opération est donnée dans l'Annexe A. Pour les types d'interface qui ne sont pas définis récursivement, les règles sont résumées ci-après.

Une interface opération *X* est un sous-type de signature de l'interface *Y* si les conditions suivantes sont satisfaites:

- pour chaque interrogation dans *Y*, il y a dans *X* une signature d'interrogation (la signature correspondante dans *X*) qui définit une interrogation portant le même nom;
- pour chaque signature d'interrogation dans *Y*, la signature correspondante dans *X* a le même nombre et les mêmes noms de paramètres;
- pour chaque signature d'interrogation dans *Y*, chaque type de paramètre est un sous-type du type de paramètre correspondant dans la signature correspondante dans *X*;
- l'ensemble des noms de terminaison d'une signature d'interrogation dans *Y* contient l'ensemble des noms de terminaison de la signature correspondante dans *X*;
- pour chaque signature d'interrogation dans *Y*, une terminaison donnée dans la signature correspondante dans *X* a le même nombre et les mêmes noms de paramètres résultats dans la terminaison du même nom dans la signature dans *Y*;
- pour chaque signature d'interrogation dans *Y*, chaque type de résultat associé à une terminaison donnée dans la signature correspondante dans *X* est un sous-type du type de résultat qui porte le même nom dans la terminaison de même nom dans *Y*;
- pour chaque annonce dans *Y*, il existe une signature d'annonce dans *X* (la signature correspondante dans *X*) qui définit une annonce portant le même nom;
- pour chaque signature d'annonce dans *Y*, il existe une signature d'annonce correspondante dans *X* qui a le même nombre et les mêmes noms de paramètres;
- pour chaque signature d'annonce dans *Y*, chaque type de paramètre est un sous-type du type de paramètre correspondant dans la signature d'annonce correspondante dans *X*.

7.2.5 Règles de gabarit

7.2.5.1 Règles de gabarit d'objets de traitement

Un objet de traitement (y compris le cas particulier que constitue un objet de liaison) peut:

- initier ou répondre à des signaux;
- produire ou consommer des flux;
- initier des invocations d'opération;
- répondre à des invocations d'opération;
- initier des terminaisons d'opération;
- répondre à des terminaisons d'opération;
- instancier des gabarits d'interface;
- instancier des gabarits d'objet;
- lier des interfaces;

- accéder à son propre état et le modifier;
- détruire une ou plusieurs de ses interfaces;
- s'autodétruire;
- générer, effectuer le branchement et la jonction d'activités;
- obtenir un identificateur d'interface de traitement pour une instance de la fonction de courtage;
- tester si une signature d'interface de traitement est un sous-type d'une autre.

Toutes ces actions peuvent échouer.

7.2.5.2 Instanciation d'interface de traitement

L'instanciation d'interface de traitement établit un ou plusieurs identificateurs d'interface de traitement pour la nouvelle interface de l'objet exécutant l'instanciation.

7.2.5.3 Instanciation de gabarit d'objet de traitement

La définition du comportement dans un gabarit d'objet de traitement comprend une description du comportement qui se produit lors de l'instanciation du gabarit (le **comportement instancié**). La spécification du contrat d'environnement décrit le contrat à établir entre l'objet instancié et son environnement lors de l'instanciation du gabarit. Lorsque le comportement d'instanciation comprend une instanciation d'interface, l'instanciation établit des identificateurs pour ces interfaces dans l'objet qui a initié l'instanciation.

7.2.6 Règles de défaillance

Les types de défaillance observables par un objet sont déterminés par son comportement et des spécifications de contrat d'environnement.

Toute action de traitement décrite au 7.2.5.1 peut échouer et une telle défaillance peut être observée par l'objet exécutant l'action. L'interaction peut être interrompue suite à la défaillance des objets impliqués ou de la liaison qu'il y a entre eux, ou des deux. Dans le cas des signaux, la défaillance est identique et visible pour tous les participants de l'interaction. Dans le cas des flux et des opérations, la défaillance peut ne pas intervenir chez tous les participants, et elle peut avoir lieu à des moments différents et avec des paramètres différents pour chaque participant défaillant.

NOTE – Exemples de défaillances d'interaction: défaillances de sécurité, défaillances de communication et défaillances de ressource.

Pour les opérations, l'incapacité d'un objet serveur à répondre à des invocations ou à initier des terminaisons peut être observée par l'objet client impliqué.

L'instanciation d'un gabarit d'objet ou d'un gabarit d'interface de traitement échoue si le contrat d'environnement ne peut être satisfait. Une action de liaison échoue si l'un des contrats d'environnement des interfaces en cours de liaison n'est pas satisfait.

7.2.7 Règles de portabilité

Les normes de portabilité dans les systèmes ODP spécifient des gabarits d'action pour les actions décrites au 7.2.5.1. La spécification de tels gabarits relève de la conception de langage et se situe en dehors du domaine d'application du présent Modèle de référence. Parallèlement aux préoccupations syntaxiques, une norme de portabilité doit adresser des problèmes sémantiques spécifiques incluant:

- des règles de composition pour les gabarits d'action, comprenant les gabarits pour les actions de branchement et de jonction pour permettre la concurrence et la synchronisation;
- les termes utilisables pour la spécification des gabarits d'objet et d'interface, ainsi que leurs règles de composition;
- les garanties d'ordonnancement et de remise pour les annonces.

Une norme de portabilité peut représenter les actions autorisées directement (sous forme de fonctions de bibliothèque) ou indirectement par l'intermédiaire de structures syntaxiques. Il peut y avoir plusieurs catégories de normes de portabilité différant selon le style (par exemple, un modèle d'exécution basé sur les événements par rapport à un modèle reposant sur les fils d'exécution) ou le contenu (par exemple, le nombre d'actions de traitement prises en charge). Le présent Modèle de référence identifie deux de ces catégories, portabilité de base et portabilité étendue.

Une **norme de portabilité de base** inclut au minimum:

- la prise en compte d'interrogation;
- la fourniture de liaisons implicites;

- la possibilité d'instancier des objets de traitement;
- la possibilité d'instancier des interfaces de traitement;
- la possibilité d'accéder et de modifier l'état d'un objet;
- la prise en charge de fils d'exécution avec des actions de génération, de branchement et de jonction;
- la possibilité d'obtenir un identificateur pour une interface de traitement fournissant la fonction de courtage (pour autoriser la liaison avec cette interface et l'utilisation de la fonction correspondante);
- la vérification de la relation de sous-typage entre signatures d'interface.

Une **norme de portabilité étendue** inclut toutes les actions décrites au 7.2.5.1.

7.3 Conformité et points de référence

Dans le langage de traitement, il existe un point de référence au niveau de chaque interface de chaque objet. Chaque point de référence peut devenir un point de référence physique, de perception, d'interfonctionnement ou d'échange en fonction des exigences qui on été établies lorsque le point de référence a été désigné comme point de conformité par une norme spécifique ou dans la spécification du système.

Dans le langage de traitement, ces exigences sont spécifiées en termes de gabarits d'interface et d'objet déterminant l'interface de l'objet dont il faut vérifier la conformité.

Un réalisateur se déclarant conforme à une spécification de traitement doit énumérer les points de référence d'ingénierie qui correspondent au point de référence de traitement requis et indiquer quelles transparences et structures d'ingénierie s'y appliquent. De ce fait, les points de référence identifiés deviennent des points de conformité. L'ensemble des interactions en ces points de conformité peut alors être interprété en termes de langage de traitement pour déterminer que la spécification de traitement n'a pas été transgressée.

La conformité d'un objet à un point de référence de programmation peut être testée en fonction d'un langage de spécification d'interface normalisé et d'un couplage avec langage respectant les règles de portabilité. La conformité d'un objet à un point de conformité d'interfonctionnement peut être testée en termes d'interactions observables dans les protocoles de communications.

8 Langage d'ingénierie

Le langage d'ingénierie comprend des concepts, des règles et des structures pour la spécification d'un système ODP du point de vue ingénierie.

Une spécification d'ingénierie définit les mécanismes et les fonctions requis pour prendre en charge l'interaction répartie entre les objets d'un système ODP.

8.1 Concepts

Le langage d'ingénierie contient les concepts de la Rec. UIT-T X.902 | ISO/CEI 10746-2 et ceux définis ici, soumis aux règles du 8.2.

8.1.1 **Objet d'ingénierie de base**: objet d'ingénierie qui nécessite une mise en œuvre d'une infrastructure répartie.

8.1.2 **Grappe**: configuration d'objets d'ingénierie de base formant une unité de désactivation, de pose de point de reprise, de réactivation, de reprise et de migration.

NOTE – Un segment de mémoire virtuelle contenant des objets est un exemple de grappe.

8.1.3 **Gestionnaire de grappe**: objet d'ingénierie qui gère les objets d'ingénierie de base appartenant à une grappe.

8.1.4 **Capsule**: configuration d'objets d'ingénierie formant une unité d'encapsulation de traitement et de stockage.

NOTE – Une machine virtuelle (par exemple, un processus) est un exemple de capsule.

8.1.5 **Gestionnaire de capsule**: objet d'ingénierie qui gère les objets d'ingénierie appartenant à une capsule.

8.1.6 **Noyau**: objet d'ingénierie coordonnant les fonctions de traitement, de stockage et de communication destinées à être utilisées par les autres objets d'ingénierie du nœud auquel il appartient.

NOTE – Un (noyau de) système d'exploitation est un exemple de noyau.

8.1.7 Nœud: configuration d'objets d'ingénierie formant une unité constituée à des fins de localisation dans l'espace, et qui comprend un ensemble de fonctions de traitement, de stockage et de communication.

NOTES

1 Un ordinateur et les logiciels qu'il met en œuvre (système d'exploitation et applications) est un exemple de nœud.

2 Une spécification d'ingénierie ne se préoccupe pas de la structure interne d'un nœud. Un ordinateur parallèle avec un système d'exploitation unique est un exemple de nœud.

8.1.8 Canal: configuration de talons, objets lieurs, objets protocoles et intercepteurs assurant une liaison entre un ensemble d'interfaces d'objets d'ingénierie de base, et autorisant ainsi leurs interactions.

NOTE – Les liaisons qui nécessitent des canaux sont appelées **liaisons réparties** dans le langage d'ingénierie; les liaisons entre objets d'ingénierie ne nécessitant pas de canal (par exemple, entre des objets d'ingénierie d'une même grappe) sont appelées **liaisons locales**.

8.1.9 Talon: objet d'ingénierie appartenant à un canal qui interprète les interactions acheminées par le canal et qui, suivant cette interprétation, exécute toute transformation ou supervision nécessaire.

NOTE – Par exemple, un talon peut encoder des paramètres dans des tampons de communication, ou enregistrer l'activité à des fins d'audit.

8.1.10 Objet lieur: objet d'ingénierie appartenant à un canal qui maintient une liaison répartie entre des objets d'ingénierie de base interagissant.

8.1.11 Intercepteur de <x>: objet d'ingénierie appartenant à un canal placé à la frontière entre des domaines de <x>. Un intercepteur de <x>:

– réalise des vérifications afin d'appliquer ou de superviser l'application de politiques d'interaction concernant des objets d'ingénierie de base de différents domaines;

– réalise des transformations afin de masquer les différences dans l'interprétation des données que peuvent avoir des objets d'ingénierie de base appartenant à des domaines différents.

NOTE – Un relais de sous-réseau ou d'interréseau est un exemple d'intercepteur.

8.1.12 Objet protocole: objet d'ingénierie appartenant à un canal qui communique avec d'autres objets protocoles dans le même canal pour assurer l'interaction entre les objets d'ingénierie de base (ces objets pouvant être dans d'autres grappes, dans d'autres capsules ou dans d'autres nœuds).

8.1.13 Domaine de communication: ensemble d'objets protocoles capables d'interfonctionner.

8.1.14 Interface de communication: interface d'un objet protocole qui peut être liée, soit à une interface d'un objet intercepteur, soit à une interface d'un autre objet protocole à un point de référence d'interfonctionnement.

8.1.15 Identificateur d'extrémité de liaison: identificateur, dans le contexte de désignation d'une capsule, qui est utilisé par un objet d'ingénierie de base pour sélectionner l'une des liaisons dans laquelle il est impliqué, dans un but d'interaction.

NOTES

1 Une adresse mémoire (pour une structure de données représentant une interface d'ingénierie) est un exemple d'identificateur d'extrémité de liaison.

2 Une même forme d'identificateur d'extrémité de liaison peut être utilisée selon que la liaison soit locale ou répartie.

8.1.16 Référence d'interface d'ingénierie: identificateur, dans le contexte d'un domaine de gestion de référence d'interface d'ingénierie utilisé pour désigner une interface d'objet d'ingénierie disponible pour une liaison répartie.

NOTE – Une référence d'interface d'ingénierie est nécessaire lors de l'établissement de liaisons réparties, et est distincte des identificateurs d'extrémité de liaison utilisés par un objet d'ingénierie de base dans un but d'interaction.

8.1.17 Domaine de gestion de référence d'interface d'ingénierie: ensemble de nœuds formant un domaine de désignation dans le but d'attribuer des références d'interface d'ingénierie.

8.1.18 Politique de gestion de référence d'interface d'ingénierie: ensemble de permissions et d'interdictions régulant la fédération de domaines de gestion de référence d'interface d'ingénierie.

8.1.19 Gabarit de grappe: gabarit d'objet pour une configuration d'objets et toute activité nécessaire pour instancier ces objets ou établir des liaisons initiales.

8.1.20 Point de reprise: gabarit d'objet dérivé de l'état et de la structure d'un objet d'ingénierie, qui peut être utilisé pour instancier un autre objet d'ingénierie, dont l'état est cohérent avec celui qu'avait l'objet original au moment de la pose de point de reprise.

8.1.21 Pose de point de reprise: création de point de reprise. Les points de reprise ne peuvent être créés que lorsque l'objet d'ingénierie impliqué satisfait la précondition énoncée dans une politique de pose de point de reprise.

8.1.22 Point de reprise de grappe: gabarit de grappe contenant les points de reprise des objets d'ingénierie de base d'une grappe.

8.1.23 Désactivation: pose de point de reprise de grappe, suivie de la suppression de cette grappe.

8.1.24 Clonage: instanciation d'une grappe à partir d'un point de reprise de grappe.

8.1.25 Reprise: clonage d'une grappe suite à la défaillance ou à la suppression de cette grappe.

8.1.26 Réactivation: clonage d'une grappe suite à sa désactivation.

8.1.27 Migration: déplacement d'une grappe vers une capsule différente.

8.2 Règles de structuration

Une spécification d'ingénierie définit l'infrastructure nécessaire pour prendre en charge la répartition fonctionnelle d'un système ODP. Elle identifie:

- les fonctions ODP nécessaires pour gérer la répartition physique, les communications, les traitements et le stockage;
- les rôles des différents objets d'ingénierie assurant les fonctions ODP (par exemple, le noyau).

Une spécification d'ingénierie s'exprime en termes de:

- configuration d'objets d'ingénierie structurés sous forme de grappes, de capsule et de nœuds;
- d'activités se produisant à l'intérieur de ces objets;
- d'interactions entre ces objets d'ingénierie.

Une spécification d'ingénierie doit respecter les règles du langage d'ingénierie. Celles-ci comprennent:

- des règles de canal (voir 8.2.1), de référence d'interface (voir 8.2.2), de liaison répartie (voir 8.2.3) et de relocalisation (voir 8.2.4) qui permettent des interactions entre objets d'ingénierie de manière transparente à la répartition;
- des règles de grappe (voir 8.2.5), de capsule (voir 8.2.6) et de nœud (voir 8.2.7) régissant la configuration d'objets d'ingénierie;
- des règles de défaillance (voir 8.2.9).

8.2.1 Règles de canal

Un canal permet la réalisation d'interactions entre objets d'ingénierie d'une manière transparente à la répartition. Cela inclut, par exemple:

- l'exécution d'opération entre un objet client et un objet serveur;
- la multidiffusion entre groupes d'objets;
- des interactions de flux impliquant de multiples objets producteurs et consommateurs.

Une interaction entre objets d'ingénierie entraîne le transfert de tout ou partie des éléments suivants:

- références d'interface d'ingénierie;
- gabarits de grappe;
- données.

Un canal est une configuration de talons, d'objets lieurs, d'objets protocoles et d'intercepteurs interconnectant un ensemble d'objets d'ingénierie. La configuration est un graphe acyclique avec des talons aux nœuds les plus extérieurs comme le montrent les Figures 2 et 3. Un chemin dans ce graphe entre objets talons comprend dans l'ordre:

- soit, un objet lieur, un objet protocole, un objet protocole et un objet lieur;
- ou, un objet lieur, un objet protocole, un intercepteur, un objet protocole et un objet lieur.

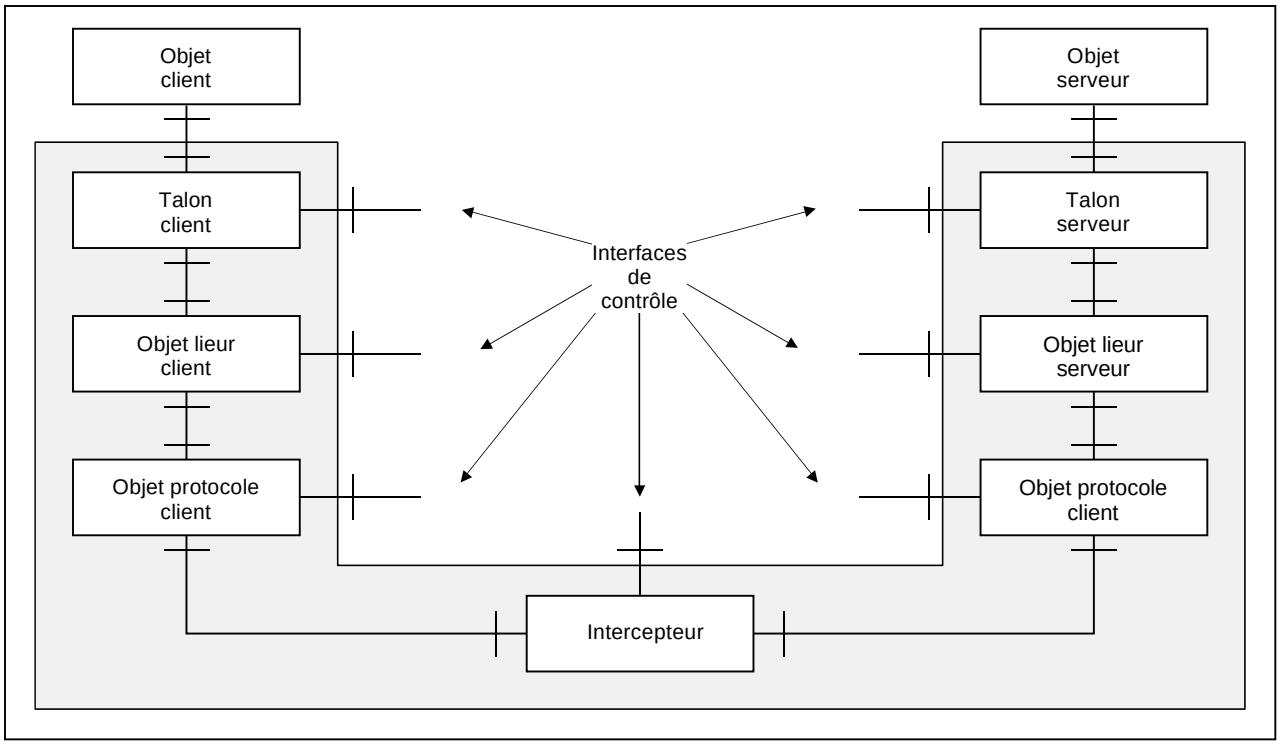

Figure 2 – Exemple de canal client/serveur

Le comportement d'un canal du point de vue de la gestion de qualité de service ou de la configuration de canal est commandé par les interfaces de contrôle des talons, des objets lieurs, des objets protocoles et des intercepteurs. Ces interfaces de contrôle sont optionnelles.

NOTES

1 Des talons, des objets lieurs, des objets protocoles et des intercepteurs appartenant à un canal peuvent avoir des liaisons (locales ou réparties) avec d'autres objets hors du canal. Par exemple, ces objets peuvent assurer des fonctions de relocalisation ou de coordination.

2 Selon le type de transformation, un intercepteur peut être décomposé en talons, en objets lieurs, en objets protocoles et en objets d'ingénierie de base, reflétant ainsi la structure d'un canal.

Les objets d'un canal peuvent eux-mêmes être des objets d'ingénierie de base pris en charge par d'autres canaux.

8.2.1.1 Talons

Les objets d'ingénierie de base qui sont en interaction via des canaux sont liés localement à des talons. Dans un canal, les talons assurent la conversion des données véhiculées par les interactions. Les talons peuvent exercer des contrôles et conserver des enregistrements (par exemple, pour la sécurité et la comptabilité). Les talons peuvent interagir, si nécessaire, avec des objets d'ingénierie hors du canal (par exemple, une fonction de sécurité). Un talon possède une interface utilisée par l'objet d'ingénierie qu'il supporte et une interface pour interagir avec un objet lieur. Il peut également disposer d'une interface de contrôle.

Quand des talons interconnectés utilisent des syntaxes de transfert différentes, il doit y avoir un intercepteur entre eux pour transformer les données d'une syntaxe à l'autre.

Un talon peut être:

a) spécifique à l'*instance d'interface* de l'objet d'ingénierie de base à laquelle il est lié;

b) spécifique au *type d'interface* de l'interface d'objet d'ingénierie de base à laquelle il est lié (par conséquent, un même talon peut être partagé par un certain nombre de canaux du même type);

c) générique (c'est-à-dire non spécifique à un type d'interface); de tels talons peuvent être partagés par un certain nombre de canaux de types différents.

NOTES

1 Dans a), les interactions entre l'objet d'ingénierie et le talon n'acheminent que les données d'interaction (par exemple, dans le cas d'une invocation, les noms d'opération et les paramètres). Le talon fait fonction de représentant local des autres objets d'ingénierie de base liés au canal.

2 Dans b), les interactions doivent inclure en plus un identificateur désignant le canal à utiliser.

3 Dans c), les interactions doivent inclure en plus un identificateur et un type pour le canal à utiliser, afin de permettre au talon de garantir que les données d'interaction sont compatibles avec le type de canal.

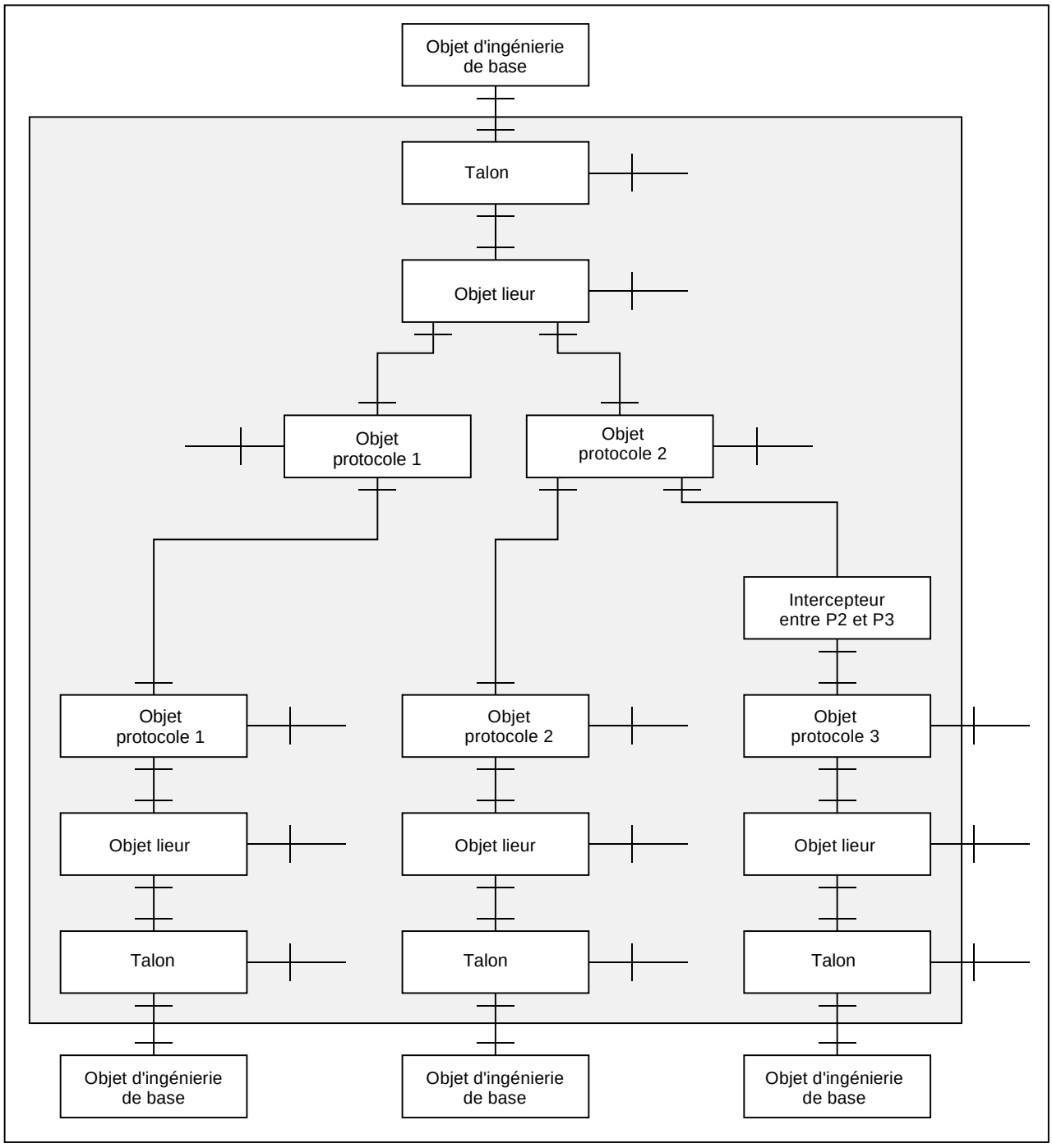

Figure 3 – Exemple d'un canal à l'extrémités multiples

8.2.1.2 Objets lieurs

Les objets lieurs appartenant à un canal assurent l'intégrité de bout en bout de ce canal. Si nécessaire, les objets lieurs assurent une transparence à la relocalisation en supervisant les défaillances de communication et en réparant les liaisons réparties détruites. Des objets lieurs d'un canal peuvent interagir avec des objets d'ingénierie hors du canal afin d'obtenir des données complémentaires nécessaires à l'exécution de leur fonction (par exemple, un relocalisateur pour obtenir des données de localisation). Un objet lieur peut posséder une interface de contrôle. Si elle est présente, cette interface permet d'effectuer des modifications dans la configuration du canal et de supprimer le canal.

8.2.1.3 Objets protocoles

Les objets protocoles assurent des fonctions de communication. Ils peuvent interagir avec des objets d'ingénierie hors d'un canal (par exemple, des fonctions d'annuaire) pour obtenir l'information dont ils ont besoin. Un objet protocole possède une interface permettant d'interagir avec un objet lieur et au moins une interface de communication pour interagir avec d'autres objets protocoles (via des intercepteurs, si nécessaire). Un objet protocole peut avoir une interface de contrôle. Lorsque les objets protocoles d'un canal sont de type différent, ils ont besoin d'un intercepteur pour assurer la conversion de protocole. Tous les objets protocoles appartenant à un domaine de communication peuvent communiquer directement à l'aide des facilités de ce domaine (ce point se situe en dehors du domaine d'application du présent Modèle de référence).

A tout instant dans le temps, un objet protocole est identifié par sa position dans l'espace, mais des objets protocoles différents peuvent occuper la même position dans l'espace à des instants différents (par exemple, des adresses réseau peuvent être recyclées). Quand des objets protocoles d'un canal sont du même type mais appartiennent à des domaines de communication disjoints, des conflits de désignation sont possibles (par exemple, des noms d'interface de communication pourraient être ambigus). Dans ce cas, un intercepteur est requis pour transformer les noms échangés lors du processus assurant l'établissement et la maintenance de l'intégrité du canal.

8.2.1.4 Intercepteurs

Un intercepteur de <x> appartenant à un canal se situe à la frontière entre les domaines de <x> et assure la vérification et la transformation des interactions qui traversent les frontières entre ces domaines de <x>. Selon la frontière traversée, les intercepteurs ont besoin de différentes informations pour exécuter leur tâche. Certains intercepteurs auront besoin de connaître les types de signature des interfaces d'objet d'ingénierie de base liées au canal dans lequel l'intercepteur est localisé, afin de pouvoir interpréter les interactions impliquant le canal. Un intercepteur dispose au moins de deux interfaces de communication et peut avoir une interface de contrôle.

8.2.2 Règles de référence d'interface

Pour les besoins des liaisons réparties, les interfaces d'ingénierie sont localisées dans l'espace et dans le temps par des références d'interface d'ingénierie. Les références d'interface d'ingénierie sont définies en fonction d'un domaine de gestion de référence d'interface d'ingénierie qui détermine une politique pour le contenu, l'attribution, le suivi et la validation des références d'interface d'ingénierie. Un domaine de gestion de référence d'interface d'ingénierie est constitué d'un ensemble de nœuds. Les domaines de référence d'interface d'ingénierie peuvent être fédérés si leurs politiques de gestion de référence d'interface d'ingénierie ne sont pas en conflit.

Une référence d'interface d'ingénierie contient des informations qui permettent d'établir des liaisons avec des interfaces d'objet d'ingénierie. Ces informations permettent aux objets noyaux de créer des canaux, et permettent également aux objets lieurs qui sont au sein des canaux de maintenir une liaison répartie entre des objets d'ingénierie interconnectés. Les informations contenues dans une référence d'interface d'ingénierie peuvent être sous forme:

- de données;
- d'identificateurs désignant les interfaces donnant accès à ces données;
- d'une combinaison de données et d'identificateurs.

Les données nécessaires à la liaison peuvent inclure tout ou partie des éléments suivants:

- le type de l'interface référencée;
- un gabarit de canal décrivant les intercepteurs, les objets protocoles, les objets lieurs et les souches pouvant être sélectionnés lors de la configuration d'un canal en vue de la prise en charge de la liaison répartie;
- la position dans le temps et l'espace des interfaces de communication au niveau desquelles le processus de liaison peut être initialisé (par exemple, une adresse de réseau);
- des informations permettant de détecter et réparer des liaisons réparties invalidées par la relocalisation d'objet d'ingénierie.

Un domaine de gestion de référence d'interface peut se décomposer en sous-domaines. Dans ce cas, les références d'interface de ce domaine sont organisées en un ensemble d'ensembles d'information, un pour chaque sous-domaine dans lequel une liaison peut être établie.

NOTE 1 – Si le noyau mettant en oeuvre une interface d'ingénierie prend en charge différents protocoles, différents processus de liaison et différentes syntaxes de transfert, la référence de l'interface d'ingénierie indiquera les combinaisons valides pouvant être choisies pour les liaisons réparties quelconques; différentes liaisons pourraient faire différents choix.

NOTE 2 – Le présent Modèle de référence ne prescrit pas de méthode pour obtenir à partir d'une référence d'interface d'ingénierie un gabarit de canal et la position dans l'espace et le temps des interfaces associées.

Les noyaux attribuent les références d'interface d'ingénierie à des interfaces fournissant la fonction de gestion de nœud (voir 12.1.3). La fonction de ramasse-miettes (voir 13.9) manipule les références d'interface d'ingénierie afin de détecter les interfaces d'ingénierie qui ne sont plus référencées. La fonction de relocalisation (voir 14.3) tient à jour les politiques régissant le rétablissement de liaison entre interfaces d'objets d'ingénierie qui ont été relocalisées et celles présidant à la mise à jour de leurs références d'interface d'ingénierie. Ces trois fonctions (gestion de nœud, ramasse-miettes et relocalisation) peuvent être coordonnées en utilisant la fonction de gestion de base d'information partagée (voir 14.2).

Les références d'interface d'ingénierie sont non ambiguës dans le contexte de désignation d'un domaine de gestion de référence d'interface d'ingénierie. Pour assurer cette non ambiguïté, les nœuds appartenant au domaine de gestion de référence d'interface d'ingénierie doivent allouer des références d'interface d'ingénierie de façon coordonnée. Les références d'interface d'ingénierie doivent être allouées de manière à empêcher qu'une référence d'interface d'ingénierie ne référence la mauvaise interface, même en cas de défaillances et de relocalisation d'interface. Une référence d'interface désignera, au pire, une interface qui n'existe pas (par exemple, suite à une défaillance de l'objet d'ingénierie supportant l'interface).

NOTE 3 – Dans les systèmes ODP pour lesquels la plupart des interfaces ne changent pas de position, la gestion des références d'interface peut être optimisée: le noyau peut attribuer des références d'interface d'ingénierie de manière autonome. Le type de canal et l'identificateur d'interface de communication associés à l'interface peuvent être stockés et transmis dans la référence d'interface d'ingénierie. La fonction de relocalisation peut servir à valider et à mettre à jour des références d'interface d'ingénierie pour les interfaces ayant été relocalisées.

Avant l'émission d'une référence d'interface d'ingénierie, le noyau construit un gabarit de canal définissant une configuration de talons, d'objets lieurs et d'objets protocoles appropriée pour assurer les interactions à l'interface. Par ailleurs, le noyau établit une structure locale suffisante pour permettre la liaison avec l'interface, et associe cette structure et le gabarit à une interface de communication. La référence d'interface d'ingénierie rend cette information disponible.

Un intercepteur se trouvant à une frontière de domaines de gestion de référence d'interface d'ingénierie maintient la correspondance entre les références d'interface d'ingénierie de ces domaines. Lorsqu'une référence d'interface d'ingénierie ou un gabarit de grappe contenant des références d'interface d'ingénierie traverse une frontière de domaine de référence d'interface d'ingénierie, les références d'interface d'ingénierie impliquées doivent être transformées pour être valides dans le nouveau domaine.

L'échange de références d'interface d'ingénierie d'un domaine de gestion de référence d'interface à un autre n'est possible que lorsqu'une procédure de mise en correspondance des références a été définie pour éviter toute ambiguïté.

8.2.3 Règles de liaison répartie

L'établissement d'un canal nécessite la création de talons, d'objets lieurs, d'objets protocoles et d'intercepteurs appropriés. Il peut être initialisé par un objet d'ingénierie quelconque. Chaque noyau peut établir un canal grâce à une fonction de son interface de gestion de nœud. Les liaisons réparties impliquent l'interaction avec les noyaux des nœuds dans lesquels les interfaces à lier sont localisées. L'établissement d'un canal est paramétré par un gabarit de canal et un ensemble de références d'interface, chacune étant assignée à un rôle particulier dans le gabarit de canal. Celui-ci doit être compatible avec les types de canal identifiés dans les références d'interface d'ingénierie des interfaces à lier. Pour chaque objet à lier le noyau crée au sein de son nœud une configuration de talons, d'objets lieurs et d'objets protocoles afin de prendre en charge les interfaces de cet objet. Cela inclut la configuration de leurs interfaces de contrôle. Les objets protocoles appartenant au canal sont connectés (via des intercepteurs, éventuellement) à leurs interfaces de communication. La sélection et la configuration de talons, d'objets lieurs, d'objets protocoles et d'intercepteurs sont déterminées par le gabarit et les types de canal des références d'interfaces impliquées. On attribue à chaque objet d'ingénierie de base lié par un canal un identificateur d'extrémité de liaison pour chacune de ses interfaces avec le canal. Les objets d'ingénierie de base utilisent des identificateurs d'extrémité de liaison pour indiquer à quelle interface doit se produire une interaction répartie.

NOTES

1 Un objet d'ingénierie quelconque peut établir un canal, y compris lorsqu'il dispose d'une interface devant être liée par ce canal.

2 Un objet d'ingénierie de base initialisant une liaison répartie requiert un ensemble de références d'interface. Celles-ci peuvent être obtenues de l'une ou l'autre des manières suivantes:

a) à l'initialisation de l'objet;

b) par interaction entre l'objet initiateur et le noyau lors de l'instanciation des interfaces de l'objet initiateur;

c) par l'intermédiaire d'une chaîne d'interactions avec les autres objets concernés (par exemple, par le passage de paramètres ou le courtage).

3 Un gabarit de canal peut contenir des configurations alternatives à appliquer dans des circonstances choisies. Par exemple, si des voies de communication ne sont pas sûres, des talons assurant un chiffrement pourraient être requis.

8.2.4 Règles de relocalisation

Les objets d'ingénierie peuvent être relocalisés suite à:

– une réactivation et une désactivation;

– une pose de point de reprise et une reprise;

– une migration;

– l'exécution de fonctions de gestion de domaine de communication (par exemple, la modification d'un identificateur d'interface de communication).

NOTES

1 Un identificateur d'une interface de communication peut être modifié suite au changement d'adresse de réseau d'un nœud.

2 Lorsque des canaux sont rétablis, des talons, objets lieurs et objets protocoles, différents de ceux utilisés avant la relocalisation, peuvent être utilisés. Un identificateur d'interface de communication n'est donc pas suffisant pour identifier une interface d'objet d'ingénierie.

La relocalisation peut entraîner la défaillance de canaux et peut invalider des références d'interface d'ingénierie. Des canaux défaillants peuvent être réparés si l'activité qui modifie la position de l'interface d'objet d'ingénierie notifie, de manière appropriée, la fonction de relocalisation (voir 14.3).

Les objets lieurs d'un canal détectent le moment où la relocalisation a invalidé le canal. Soit les objets lieurs collaborent pour corriger la mise en correspondance entre les références d'interface d'ingénierie et la structure du canal (dans ce cas la transparence à la relocalisation est nécessaire – voir 16.6), soit le canal échoue. Lorsque la transparence à la relocalisation est requise, les informations disponibles par l'intermédiaire de la référence d'interface d'ingénierie permettent aux objets lieurs d'utiliser la fonction de relocalisation pour déterminer les nouvelles positions des objets d'ingénierie de base impliqués.

8.2.5 Règles de grappe

Une grappe contient un ensemble d'objets d'ingénierie de base, auquel est associé un gestionnaire de grappe. Chaque membre de la grappe peut posséder une interface supportant la fonction de gestion d'objet. Chacune de ces interfaces de gestion d'objet est liée au gestionnaire de la grappe. Un objet d'ingénierie de base qui appartient à une grappe est toujours lié à son noyau, à une interface assurant la fonction de gestion de nœud, et à son gestionnaire de grappe. De plus, il peut être lié à d'autres objets d'ingénierie de base appartenant à la même grappe ou à d'autres. Chaque gestionnaire de grappe appartenant à une capsule est lié au gestionnaire de cette capsule. Cette structure est représentée dans la Figure 4.

Une grappe est toujours contenue dans une seule capsule. Une grappe est responsable de sa propre sécurité, mais peut être assistée par des fonctions de sécurité. Une telle fonction doit être assurée soit par un objet appartenant à la même capsule que la grappe, soit par un objet extérieur à la capsule, mais, dans ce cas, on devra accéder à l'objet par l'intermédiaire d'interactions sûres. Les objets d'ingénierie appartenant à la même grappe peuvent interagir à l'aide d'une liaison locale dans la grappe, ou en utilisant une liaison répartie supportée par un canal. Les objets d'ingénierie dans différentes grappes interagissent en utilisant des liaisons réparties supportées par des canaux.

NOTES

1 Les interactions à des points de référence physiques et d'échange ne sont pas exclues.

2 Bien qu'aucun canal ne soit nécessaire pour la mise en œuvre de liaisons locales entre des objets d'une même grappe, l'identificateur utilisé pour exécuter des invocations est du même type que celui utilisé entre des objets d'ingénierie de grappes différentes, et est toujours appelé identificateur d'extrémité de liaison.

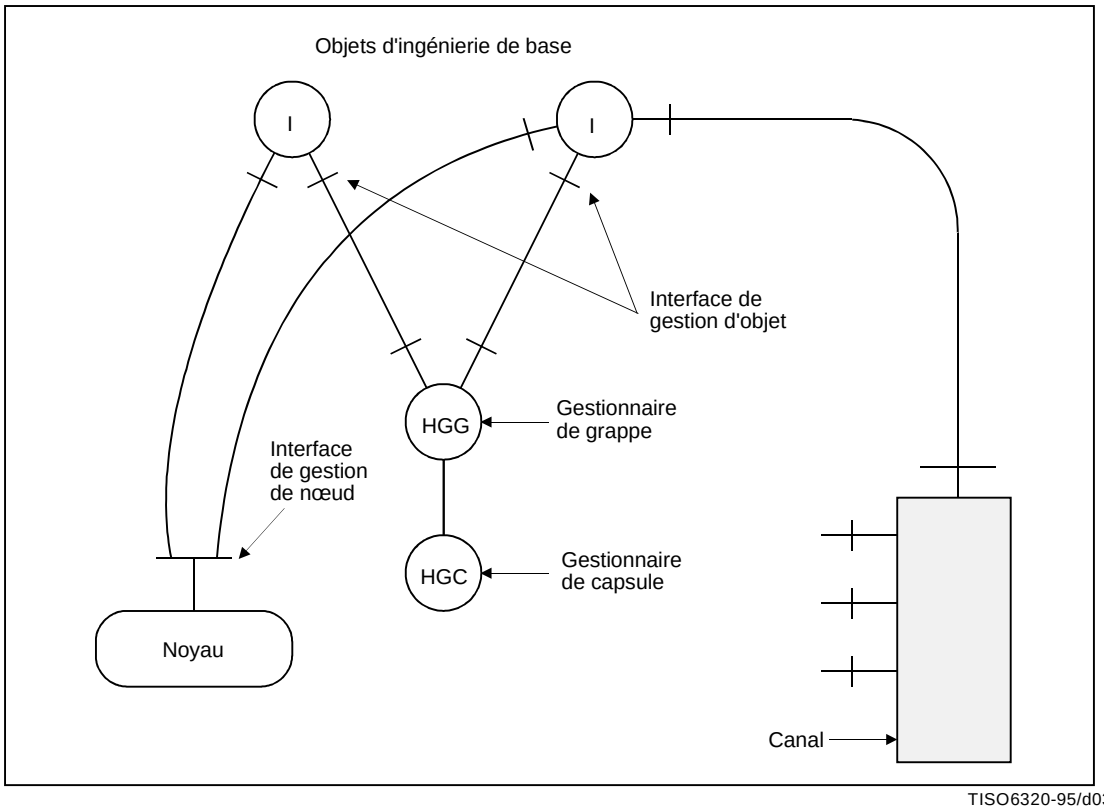

Figure 4 – Exemple de structure assurant le support d'un objet d'ingénierie de base

L'instanciation de grappe (le clonage en est un cas particulier) est réalisée par un gestionnaire de capsule.

Si le gabarit est un point de reprise de grappe, l'instanciation (à savoir le clonage) permet à la nouvelle grappe d'agir en tant que substitut à la grappe d'origine dont le gabarit de grappe est issu. Quand il est requis pour des raisons de transparence à la répartition, le processus de clonage inclut le rétablissement de toutes les liaisons réparties qui étaient maintenues par la grappe d'origine.

Une grappe possède un gestionnaire de grappe associé. Le gestionnaire de grappe assure la fonction de gestion de grappe. Il incorpore les politiques de gestion concernant les objets d'ingénierie appartenant à la grappe. Les politiques de gestion de grappe peuvent conduire le gestionnaire de grappe à interagir avec d'autres fonctions ODP pour réaliser des fonctions de gestion de grappe.

Un gestionnaire de grappe assiste son gestionnaire de capsule dans la gestion des références d'interface d'ingénierie des objets. Cela implique l'accès à la fonction de ramasse-miettes.

8.2.6 Règles de capsule

Une capsule se compose

- de grappe(s);
- de gestionnaires de grappe, un pour chacune des grappes de la capsule;
- d'un gestionnaire de capsule auquel chacun des gestionnaires de grappe est lié.

Des talons, des objets lieurs et des objets protocoles appartenant à un canal lié à une interface d'un objet d'ingénierie dans une grappe d'une capsule peuvent être inclus dans cette capsule. Tous les objets d'ingénierie de la capsule sont liés à la même interface de gestion de nœud. Les objets des autres capsules sont liés à des interfaces de gestion de nœud différentes. Une capsule est contenue dans un nœud. Une capsule possède un gestionnaire de capsule. Le gestionnaire de capsule, à une interface assurant la fonction de gestion de grappe, est lié à chaque gestionnaire de grappe de la capsule. Cette structure est illustrée dans la Figure 5.

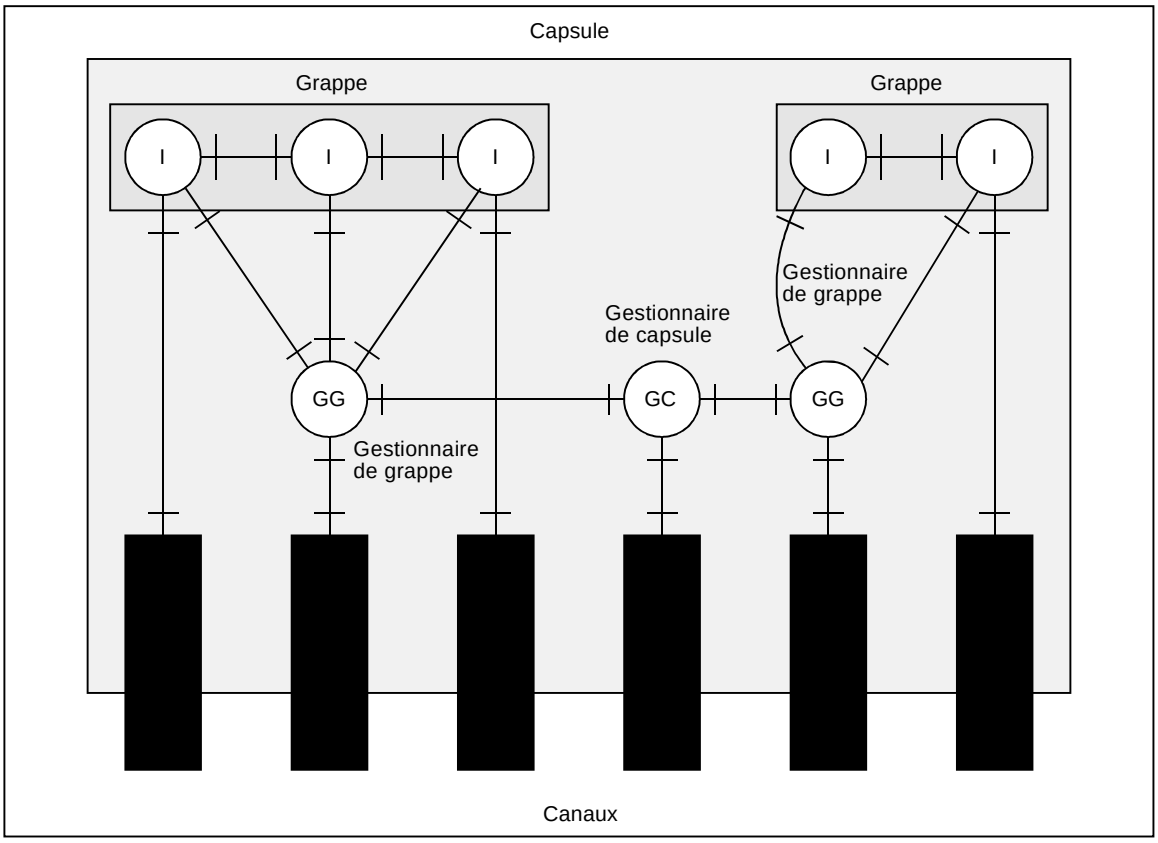

Figure 5 – Exemple de structure de capsule

L'instanciation de capsule est réalisée par le noyau à l'aide d'un gabarit de capsule qui spécifie la configuration initiale des objets d'ingénierie de la capsule, y compris les grappes, les gestionnaires de grappe, les talons, les objets lieurs, les objets protocoles et le gestionnaire de capsule.

Une capsule constitue un contexte de désignation pour les identificateurs d'extrémité de liaison. Le présent Modèle de référence n'exige pas que de tels identificateurs soient valides dans des contextes plus larges. Les références d'interface d'ingénierie sont utilisées pour communiquer la connaissance des interfaces d'objet d'ingénierie entre les capsules (pour des besoins de liaison).

Le gestionnaire de capsule incorpore les politiques de gestion concernant les grappes de la capsule. Les politiques de gestion de capsule peuvent conduire le gestionnaire de capsule à interagir avec d'autres fonctions pour réaliser des activités de gestion de capsule. Le gestionnaire de capsule possède une interface qui assure la fonction de gestion de capsule. Les moyens par lesquels les gestionnaires de grappe et de capsule interagissent entre eux et avec le noyau ne sont pas couverts par le présent Modèle de référence.

8.2.7 Règles de nœud

Un nœud se compose d'un noyau et d'un ensemble de capsules. Tous les objets d'ingénierie appartenant à un nœud partagent des fonctions communes de traitement, de stockage et de communication.

Un nœud est un membre d'un ou plusieurs domaines de gestion de référence d'interface d'ingénierie.

Le noyau fournit un ensemble d'interfaces de gestion de nœud, une pour chacune des capsules du nœud.

La structure d'un nœud est illustrée dans la Figure 6.

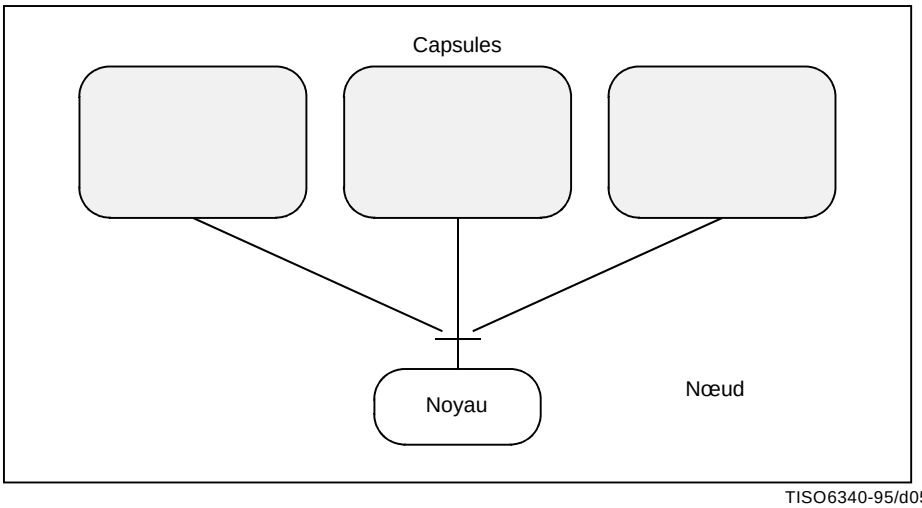

Figure 6 – Exemple de structure d'un nœud

La procédure d'instanciation de nœud n'entre pas dans le cadre du présent Modèle de référence. Elle doit résulter dans:

- l'introduction du noyau du nœud et de ses fonctions associées de traitement, de stockage et de communication, y compris l'introduction de fonctions de gestion de nœud pour permettre la liaison répartie des références d'interface d'ingénierie;
- l'introduction de toute fonction de courtage nécessaire au processus d'instanciation;
- l'instanciation de tous les canaux devant faire partie de la configuration initiale du nœud (par exemple, avec les objets de prise en charge tels qu'un relocalisateur).

L'ensemble des objets protocoles introduits durant l'instanciation du nœud détermine l'ensemble initial des domaines de communication dont le nœud fait partie. Le noyau assure la fonction de gestion de nœud et incorpore des politiques de gestion de nœud. Les politiques de gestion de nœud peuvent conduire le noyau à interagir avec d'autres fonctions ODP pour réaliser des activités de gestion de nœud. Une capsule est l'unité de base pour l'application des politiques de gestion de nœud; bien qu'un objet élémentaire puisse utiliser la fonction de gestion de nœud, il est soumis aux politiques applicables à sa capsule. Des capsules distinctes peuvent être soumises à différentes politiques de gestion de nœud.

8.2.8 Règles de gestion d'application

La gestion du cycle de vie (création, migration, désactivation, réactivation, pose de point de reprise, défaillance, reprise et destruction) d'ensembles de grappes coordonnées dépend de politiques de gestion spécifiques aux applications. Ces politiques peuvent s'appliquer à des grappes individuelles ou à des ensembles coordonnés de grappes. Un ensemble de grappes géré forme un **domaine de gestion d'application**. Une politique de gestion d'application peut être mise en œuvre par des fonctions de gestion spécifiques à l'application considérée qui effectuent leur travail en utilisant les mécanismes fournis par les fonctions de coordination et de gestion définies dans ce Modèle de référence (par exemple, la fonction de gestion de grappe).

Si nécessaire, en fonction des transparences à la répartition utilisées, les fonctions de gestion spécifiques à l'application reçoivent des notifications d'événements significatifs affectant les grappes qu'elles gèrent, et agissent en réponse à ces notifications. Par exemple, des rapports indiquant la défaillance d'une liaison peuvent conduire à la réactivation d'une grappe, ou des rapports indiquant une charge excessive peuvent entraîner la migration des grappes. Des requêtes et des notifications relatives à une grappe dans un domaine de gestion d'application peuvent conduire des fonctions de gestion spécifiques à l'application des actions de gestion de cycle de vie dans d'autres grappes du domaine.

Les détails de la gestion de domaine d'application ne sont pas couverts par le présent Modèle de référence.

8.2.9 Règles de défaillance

Les défaillances se divisent en plusieurs catégories: les défaillances de grappe, de capsule, de nœud ou de domaines de communication. L'analyse des défaillances exploite les hypothèses suivantes:

- une défaillance localisée dans une grappe peut être détectée par son gestionnaire de grappe;
- une défaillance localisée dans une capsule peut être détectée par son gestionnaire de capsule;
- la défaillance d'un nœud peut être détectée par des objets protocoles situés sur d'autres nœuds avec lesquels il est interconnecté;
- la défaillance d'un domaine de communication peut être détectée par des objets protocoles appartenant à d'autres domaines avec lesquels il est interconnecté.

NOTE – La détection de défaillance d'un nœud distant peut apparaître à un objet protocole comme une défaillance de communication et vice versa.

8.3 Conformité et points de référence

Il y a un point de référence de programmation à un point d'interaction entre un gestionnaire de grappe et un objet d'ingénierie de base.

Il y a un point de référence de programmation à un point d'interaction entre un objet d'ingénierie et un noyau.

Il y a un point de référence de programmation à un point d'interaction entre objets d'ingénierie de base.

Il peut y avoir un point de référence physique ou d'échange à une interface d'un objet d'ingénierie de base.

Lorsqu'un canal assure l'interaction entre des objets d'ingénierie de base, il y a, pour chaque objet d'ingénierie de base, un point de référence de programmation aux points d'interaction entre ces objets et les talons de ce canal.

Au sein de la configuration d'objets composant un canal, il y a un point de référence de programmation à chaque point d'interaction du canal:

- entre les talons (en faisant abstraction des objets lieurs, des objets protocoles et des intercepteurs du canal entre les talons);
- entre les talons et les objets lieurs;
- entre objets lieurs (en faisant abstraction des objets protocoles et intercepteurs du canal entre les objets lieurs);
- entre les objets lieurs et les objets protocoles;
- entre les objets de protocole et d'autres objets protocoles appartenant au même nœud (en faisant abstraction des intercepteurs entre les objets protocoles, s'il y en a);

de plus, il y a un point de référence d'interfonctionnement, à tout point d'interaction entre les objets protocoles et d'autres objets protocoles ou des intercepteurs appartenant à différents nœuds.

Les interfaces de contrôle des talons, des objets lieurs, des objets protocoles et des intercepteurs sont des points de référence de programmation.

Lorsque, dans un canal, des objets d'ingénierie interagissent avec les autres objets (dans ou en dehors de ce canal) via des interfaces qui ne se trouvent pas dans ce canal, les points de référence applicables à ces interfaces sont alors déterminés par l'application récursive de ces règles.

La définition de la conformité aux points de référence d'interfonctionnement rend possible l'interfonctionnement des systèmes.

La définition de la conformité aux points de référence de programmation rend possible la portabilité des objets d'ingénierie entre systèmes.

La conformité d'objets d'ingénierie individuels à des points de référence de programmation ne garantit pas en elle-même la portabilité de l'objet d'ingénierie sur tous les systèmes ou son interfonctionnement avec des objets d'ingénierie équivalents liés à d'autres noyaux.

9 Langage de technologie

Une spécification de technologie définit les choix effectués en matière de technologie pour un système ODP.

9.1 Concepts

Le langage de technologie se compose des concepts de la Rec. UIT-T X.902 | ISO/CEI 10746-2 et de ceux définis ici, soumis aux règles de structuration du 9.2.

9.1.1 Norme réalisable: gabarit d'objet technologique.

9.1.2 Mise en œuvre: processus d'instanciation dont la validité peut être testée.

9.1.3 IXIT: informations complémentaires nécessaires pour les tests de conformité.

9.2 Règles de structuration

Une spécification de technologie définit les choix effectués en matière de technologie pour un système ODP en termes:

- de configuration d'objets de technologie,
- d'interfaces entre ces objets.

Une spécification de technologie:

- définit la manière dont les spécifications d'un système ODP sont mises en œuvre;
- identifie des spécifications de technologie applicables à la construction de systèmes ODP;
- fournit une taxonomie pour de telles spécifications;
- spécifie les informations requises par les réalisateurs pour mettre en œuvre les tests de conformité.

La mise en œuvre d'une spécification écrite dans un autre langage de point de vue implique la constitution d'une spécification de technologie fournissant une interprétation des termes atomiques présents dans la spécification initiale.

La spécification de technologie d'une fonction ODP peut faire référence aux spécifications d'autres fonctions ODP.

Une spécification de technologie se compose d'assertions stipulant que certains objets de technologie sont des instances de normes réalisables données.

Une spécification de technologie comprend un formulaire pour l'IXIT de conformité qui énumère l'ensemble des gabarits requis et les noms descriptifs de tous les points de référence nécessaires.

Toutes les normes réalisables sont introduites en faisant référence à d'autres spécifications. Le langage de technologie ne définit pas d'autre règle contraignant le comportement des objets de technologie ou l'élaboration de normes réalisables.

9.3 Conformité et points de référence

On utilise le langage de technologie pour stipuler que des objets de technologie sont des instances de normes réalisables; ces normes contiendront en général des déclarations de conformité.

10 Règles de cohérence

Un ensemble de spécifications d'un système ODP écrites en différents langages de point de vue ne doit pas être contradictoire (voir 4.2.2); autrement dit, ces spécifications doivent être mutuellement cohérentes. Par conséquent, une spécification complète d'un système contient des déclarations de correspondances entre les termes et les constructions linguistiques reliant une spécification d'un point de vue à une autre, qui montrent que la prescription de cohérence est satisfaite. Pour qu'un ensemble de spécifications pour un système ODP soit cohérent, celles-ci doivent respecter les correspondances définies dans ce présent Modèle de référence et celles définies dans les spécifications elles-mêmes. Le présent Modèle de référence ne définit pas de correspondances génériques pour toutes les paires de langages de point de

vue. Le présent article se limite à la spécification des correspondances entre une spécification de traitement et une spécification d'information, et entre une spécification de traitement et une spécification d'ingénierie. Dans chaque cas, les correspondances sont exprimées en tant que relations d'interprétation reliant des termes d'un langage de point de vue à d'autres termes d'un autre langage de point de vue. Un ensemble de spécifications basé sur le présent Modèle de référence devra, en général, relier toutes les spécifications de point de vue.

La cohérence repose sur la notion de correspondance entre les spécifications, à savoir, une assertion selon laquelle certains termes ou structures d'une première spécification correspondent à d'autres termes et spécifications d'une seconde. On peut établir des correspondances entre deux spécifications différentes qu'elles soient définies dans un même langage ou dans deux langages différents. Les assertions de correspondance entre deux langages impliquent des correspondances équivalentes entre toute paire de spécifications exprimées dans ces langages.

L'analyse de cohérence dépend des techniques de vérification appliquées. La plupart d'entre elles sont basées sur des vérifications qui contrôlent des types particuliers d'incohérence; elles ne peuvent pas garantir, pour cette raison, une cohérence absolue. Une forme de définition de cohérence implique la définition d'une transformation d'un langage dans un autre. Ainsi, étant donné une spécification S_1, dans un langage de point de vue L_1 et une spécification S_2 dans un langage de point de vue L_2, où S_1 et S_2 spécifient le même système, une transformation T peut être appliquée à S_1, conduisant à une nouvelle spécification $T(S_1)$ de langage de point de vue L_2. La spécification $T(S_1)$ peut être directement comparée à S_2 pour contrôler, par exemple, la compatibilité de comportement entre des objets ou des configurations d'objets sensés équivalents.

10.1 Correspondances entre spécifications de traitement et d'information

Le présent Modèle de référence ne prescrit pas de correspondances strictes entre des objets d'information et des objets de traitement. Par exemple, les états définis dans une spécification de traitement ne correspondent pas nécessairement aux états d'une spécification d'information. Une spécification de traitement peut spécifier une série de changements d'état qui se trouve abstraite en une transition atomique dans une spécification d'information correspondante.

Si un objet d'information correspond à un ensemble d'objets de traitement, les schémas statiques et les schémas d'invariant de l'objet d'information correspondent à des états possibles des objets de traitement associés. Un changement d'état de l'objet d'information correspond à un ensemble d'interactions entre objets de traitement ou à une action interne d'un objet de traitement. Le schéma statique et le schéma d'invariant de l'objet d'information correspondent au comportement et au contrat d'environnement des objets de traitement.

NOTE – Si le concept d'interface est utilisé dans une spécification d'information, il n'y a pas nécessairement de correspondance entre une interface d'information et une interface de traitement.

10.2 Correspondances entre spécifications d'ingénierie et de traitement

Chaque objet de traitement qui n'est pas un objet de liaison correspond à un ensemble d'un ou plusieurs objets d'ingénierie de base (et comprenant des canaux qui les connectent). A tous les objets d'ingénierie de base de cet ensemble correspond uniquement cet objet de traitement.

Sauf dans le cas où des transparences à la répartition qui impliquent une duplication d'objet sont utilisées, à chaque interface de traitement correspond exactement une interface d'ingénierie, et à chaque interface d'ingénierie correspond uniquement cette interface de traitement.

NOTE 1 – L'interface d'ingénierie appartient à l'un des objets d'ingénierie de base correspondant à l'objet de traitement fournissant l'interface de traitement considérée.

Lorsque des transparences à la répartition qui impliquent une duplication d'objet sont utilisées, à chaque interface d'un objet de traitement dupliqué correspond un ensemble d'interfaces d'ingénierie, à raison d'une interface par objet d'ingénierie de base résultant de la duplication. Ces interfaces d'ingénierie correspondent toutes à l'interface de traitement d'origine.

Chaque interface de traitement est identifiée par un membre quelconque d'un ensemble de plusieurs identificateurs d'interface de traitement. Chaque interface d'ingénierie est identifiée par un membre quelconque d'un ensemble de plusieurs références d'interface d'ingénierie. Etant donné qu'une interface de traitement correspond à une interface d'ingénierie, un identificateur d'interface de traitement peut être représenté sans ambiguïté par une référence d'interface d'ingénierie de l'ensemble correspondant.

Chaque liaison de traitement (liaisons primitives ou liaisons composites avec des objets de liaison associés) correspond soit à une liaison d'ingénierie locale, soit à un canal d'ingénierie. Cette liaison ou ce canal d'ingénierie correspondent uniquement à cette liaison de traitement. Si la liaison de traitement autorise le transfert d'opérations, la liaison d'ingénierie locale ou le canal doivent au moins supporter l'échange:

- des noms de signatures de traitement;
- des noms d'opération de traitement;
- des noms de terminaison de traitement;
- des paramètres d'invocation et de terminaison (y compris des identificateurs d'interface de traitement et des signatures d'interface de traitement).

Sauf dans le cas où des transparences à la répartition qui impliquent une duplication d'objet sont utilisées, chaque interface de contrôle d'objet de liaison de traitement dispose d'une interface d'ingénierie correspondante et il existe une chaîne d'interactions liant cette interface aux souches, objets lieurs, objets de protocole et intercepteurs impliqués dans la mise en œuvre de la liaison de traitement.

NOTE 2 – L'ensemble des interfaces de contrôle impliquées dépend du type de l'objet de liaison.

A chaque interaction de traitement correspond une chaîne d'interactions d'ingénierie, commençant et se terminant par une interaction impliquant un ou plusieurs objets d'ingénierie correspondant aux objets de traitement en interaction.

Chaque signal de traitement correspond soit, à une interaction se produisant dans le cadre d'une liaison d'ingénierie locale, soit, à une chaîne d'interactions d'ingénierie qui respecte la sémantique de l'interaction de traitement.

Les définitions de transparence à l'article 16 précisent des correspondances additionnelles.

NOTE 3 – Des objets d'ingénierie de base correspondant à différents objets de traitement peuvent être membres d'une même grappe.

NOTE 4 – Dans un langage de traitement entièrement basé objet, les données sont représentées sous forme de types abstraits de données (à savoir, des interfaces d'objets de traitement).

NOTE 5 – Les paramètres qui correspondent à des interfaces de traitement (y compris ceux concernant les types abstraits de données) peuvent être passés par référence; de tels paramètres correspondent à des références d'interface d'ingénierie.

NOTE 6 – Les paramètres qui correspondent à des interfaces de traitement (y compris ceux concernant les types abstraits de données) peuvent être passés par migration ou duplication de l'objet fournissant l'interface. Dans le cas d'une migration, ces paramètres correspondent à des gabarits de grappe.

NOTE 7 – Si l'état abstrait d'un objet d'ingénierie fournissant une interface passée en paramètre est invariant, l'objet peut être cloné plutôt que migré.

NOTE 8 – Les gabarits de grappe peuvent être représentés sous forme de types abstraits de données. Aussi, peut-on se contenter d'une stricte correspondance entre paramètres de traitement et références d'interface d'ingénierie. Néanmoins, les passages de gabarits de grappe ou de données en paramètre constituent des exemples importants d'optimisation au niveau ingénierie et ne sont donc pas exclus.

11 Fonctions ODP

Les fonctions ODP définies dans la présente Recommandation | Norme Internationale correspondent à des fonctions qui sont soit fondamentales, soit largement applicables pour la construction des systèmes ODP.

Les spécifications des fonctions élémentaires ODP peuvent se combiner pour constituer des spécifications de composants de systèmes ODP. L'identification de ces composants relève de choix de normalisation, et n'est donc pas prescrite par le présent Modèle de référence: il ne contient que des descriptions donnant un aperçu de ces fonctions, définies à l'aide des concepts de la Rec. UIT-T X.902 | ISO/CEI 10746-2.

Certaines descriptions de fonction du présent Modèle de référence introduisent des objets pour simplifier la modélisation. Mis à part le cas où des contraintes explicites sont appliquées à la répartition de ces objets, elles ne définissent pas nécessairement une structure de mise en œuvre.

Les fonctions définies dans le présent Modèle de référence sont énumérées ci-après. Celles faisant partie intégrante du langage de traitement sont signalées par l'adjonction d'un «*», et celles faisant partie intégrante du langage d'ingénierie sont signalées par l'adjonction d'un «+».

a) *fonctions de gestion*

   1) fonction de gestion de nœud^{+};

   2) fonction de gestion d'objet^{+};

3) fonction de gestion de grappe+;
4) fonction de gestion de capsule+;

b) *fonctions de coordination*
1) fonction de notification d'événement;
2) fonction de pose de point de reprise et de reprise;
3) fonction de désactivation et de réactivation;
4) fonction de groupe;
5) fonction de duplication;
6) fonction de migration;
7) fonction de ramasse-miettes+;
8) fonction de transaction.

c) *fonctions de dépôt*
1) fonction de stockage;
2) fonction de gestion de base d'information;
3) fonction de relocalisation;
4) fonction de magasin de types;
5) fonction de courtage+*;

d) *fonctions de sécurité*
1) fonction de contrôle d'accès;
2) fonction d'audit de sécurité;
3) fonction d'authentification;
4) fonction d'intégrité;
5) fonction de confidentialité;
6) fonction de non-répudiation;
7) fonction de gestion de clé.

12 Fonctions de gestion

12.1 Fonction de gestion de nœud

La fonction de gestion de nœud contrôle les fonctions de traitement, de stockage et de communication à l'intérieur d'un nœud.

La fonction de gestion de nœud est assurée par chaque noyau à une ou plusieurs interfaces de gestion de nœud.

Chaque capsule utilise une interface de gestion de nœud distincte des interfaces de gestion de nœud utilisée par les autres capsules du même nœud.

La fonction de gestion de nœud:

– gère les fils d'exécution;
– accède aux horloges et gère les temporisateurs;
– crée des canaux et localise des interfaces.

Dans le cadre de l'architecture définie par le présent Modèle de référence, la fonction de gestion de nœud est utilisée par toutes les autres fonctions.

12.1.1 Gestion de fil d'exécution

Les interfaces de gestion de nœud fournissent des fonctions permettant, au sein d'une capsule, de créer des fils d'exécution, de causer leurs branchements, de les joindre, de les mettre en attente et de les synchroniser.

12.1.2 Gestion de l'accès horloge et du temporisateur

Les interfaces de gestion de nœud fournissent des fonctions permettant de déterminer le temps courant dans un domaine de gestion d'horloge spécifié, et d'initialiser, de superviser ou d'arrêter des temporisateurs.

12.1.3 Création de canal et localisation d'interface

Les interfaces de gestion de nœud fournissent des fonctions permettant:

a) d'activer la liaison entre un objet d'ingénierie appartenant à une capsule et une instance de la fonction de courtage;

b) de rendre disponible une interface d'ingénierie à des fins de liaison avec des objets appartenant à d'autres capsules;

c) d'établir une liaison entre un objet d'ingénierie d'une capsule et un ensemble d'autres objets d'ingénierie (identifiés par des références d'interface d'ingénierie);

d) de déterminer, à partir d'une référence d'interface d'ingénierie donnée, le type de canal et l'interface de communication.

NOTES

1 L'utilisation des interactions b) et c) est expliquée au 8.2.3.

2 L'interaction d) rend explicite le contenu de la référence d'interface, permettant ainsi de le stocker ou de le transformer afin de l'initialiser dans un domaine de gestion de référence d'interface différent.

Rendre disponible une interface à des fins de liaison avec des objets appartenant à d'autres capsules (interaction b) ci-dessus) se déroule de la façon suivante:

– attribution d'une référence d'interface d'ingénierie appartenant à un domaine de gestion de référence d'interface d'ingénierie donné;

– attribution d'une interface de communication à travers laquelle des liaisons avec l'interface peuvent être établies;

– attribution d'un type de canal à l'interface.

12.1.4 Instanciation de gabarit de capsule et suppression de capsule

Les interfaces de gestion de nœud fournissent des fonctions permettant d'instancier des gabarits de capsule et de supprimer des capsules.

L'instanciation de gabarit de capsule comprend les étapes suivantes:

– allocation des ressources de traitement, de stockage et de communication concernant une nouvelle capsule appartenant au même nœud que le noyau fournissant l'interface de gestion de nœud;

– création d'un gestionnaire de capsule pour la nouvelle capsule;

– création d'une interface de *gestion* de capsule dans le nouveau gestionnaire de capsule;

– allocation d'un identificateur pour l'interface de gestion de capsule;

– création d'une interface de *contrôle* de capsule pour la nouvelle capsule du noyau;

– allocation d'un identificateur pour l'interface de contrôle de capsule.

L'interface de contrôle de capsule produite par l'instanciation d'un gabarit de capsule permet la suppression de la capsule (par exemple, quand son gestionnaire connaît une défaillance).

La suppression d'une capsule supprime tous les objets de la capsule.

12.2 Fonction de gestion d'objet

La fonction de gestion d'objet permet de poser des points de reprise et de supprimer des objets.

Lorsqu'un objet appartient à une grappe pouvant être désactivée, contrôlée ou migrée, l'objet doit posséder une interface de gestion fournissant l'une ou plusieurs des fonctions suivantes:

– pose de point de reprise pour l'objet;

– suppression de l'objet.

NOTES

1 Des interfaces de gestion d'objet différentes peuvent donc avoir des types d'interface différents selon les fonctions qu'elles fournissent.

2 Poser un point de reprise pour un objet résulte en un ensemble d'information susceptibles d'être incorporé dans un point de reprise de grappe.

3 Quand un objet est supprimé, les talons, les objets lieurs, les objets protocoles et les intercepteurs assurant les liaisons de l'objet peuvent être supprimés.

La fonction de gestion d'objet est utilisée par la fonction de gestion de grappe.

12.3 Fonction de gestion de grappe

La fonction de gestion de grappe autorise le pose de point de reprise, la reprise, la migration, la désactivation ou la suppression de grappes. Elle est fournie par chaque gestionnaire de grappe à une interface de gestion de grappe, fournissant une ou plusieurs des fonctions suivantes relativement à la grappe gérée:

- modification des politiques de gestion de grappe (par exemple, pour la localisation des points de reprise de la grappe, pour l'utilisation de la fonction de relocalisation afin de déclencher la réactivation ou la reprise de la grappe);
- désactivation de la grappe;
- pose de point de reprise pour la grappe;
- remplacement de la grappe par une nouvelle grappe instanciée à partir d'un point de reprise de grappe (c'est-à-dire, suppression suivie par reprise);
- migration de la grappe vers une autre capsule (en utilisant la fonction de migration);
- suppression de la grappe.

NOTE – Des interfaces de gestion de grappe différentes peuvent donc avoir des types différents selon les fonctions qu'elles fournissent.

Les politiques de gestion d'une grappe contraignent le comportement du gestionnaire de grappe. La pose de point de reprise de grappe n'est possible que si tous les objets de la grappe possèdent des interfaces de gestion d'objet fournissant la fonction de pose de point de reprise d'objet. La désactivation et la suppression de grappe nécessitent que les objets de la grappe supportent la suppression d'objet.

Dans l'architecture définie par le présent Modèle de référence:

- la fonction de gestion de grappe est utilisée par la fonction de gestion de capsule, la fonction de désactivation et de réactivation, la fonction de pose de point de reprise et de reprise, la fonction de migration et la fonction de gestion de référence d'interface d'ingénierie;
- la fonction de gestion de grappe utilise la fonction de stockage pour stocker les points de reprise.

12.3.1 Point de reprise de grappe

Un point de reprise de grappe contient les informations nécessaires pour rétablir une grappe. Il comprend les éléments suivants:

a) les points de reprise des objets de la grappe;

b) la configuration des objets de la grappe;

c) des informations suffisantes pour rétablir les liaisons réparties impliquant les objets de la grappe.

NOTE – Les références d'interface d'ingénierie représentent une partie essentielle des informations requises pour établir des liaisons. Un point de reprise de grappe comportera toutes les références d'ingénierie issues des éléments a) et c) ci-dessus.

12.3.2 Suppression, désactivation et défaillance de grappe

La suppression de grappe supprime tous les objets appartenant à une grappe, le gestionnaire de grappe et tous les objets supportant la grappe ou son gestionnaire (par exemple, les souches et les objets lieurs). La désactivation de grappe est coordonnée par la fonction de désactivation et de réactivation; elle produit un point de reprise pour la grappe puis supprime la grappe et sa structure de support. La défaillance d'une grappe entraîne la suppression de tous les objets de la grappe et, dans certains cas, la suppression de la structure de support de la grappe.

12.3.3 Réactivation et reprise de grappe

Une grappe désactivée peut être réactivée à partir de l'un de ses points de reprise. La réactivation de grappe est nécessairement une fonction de gestion de capsule puisque le gestionnaire de grappe est supprimé lors de la désactivation de grappe. La reprise d'une grappe peut être assurée à partir de l'un de ses points de reprise. Si le gestionnaire de grappe associé n'a pas été supprimé, il peut initialiser la reprise; dans le cas contraire, la reprise est une fonction de gestion de capsule et inclut la création d'un nouveau gestionnaire de grappe.

12.3.4 Migration de grappe

La migration de grappe correspond au clonage d'une grappe source dans une capsule cible suivie par la suppression de la grappe source. Elle est coordonnée par la fonction de migration et est paramétrée par une interface de gestion de capsule appartenant à la capsule cible de la grappe migrée.

12.4 Fonction de gestion de capsule

La fonction de gestion de capsule instancie des grappes (y compris la reprise et la réactivation), contrôle toutes les grappes d'une capsule, désactive toutes les grappes d'une capsule et supprime les capsules. Elle est assurée par chaque gestionnaire de capsule à une interface de gestion de capsule fournissant une ou plusieurs des fonctions suivantes relativement à la capsule gérée:

- l'instanciation (au sein de la capsule) d'un gabarit de grappe;

  NOTE 1 – Cela recouvre la réactivation et la reprise.

- la désactivation de la capsule en désactivant toutes les grappes de la capsule (en utilisant la fonction de gestion de grappe);
- la pose de point de reprise de capsule en posant des points de reprise pour toutes les grappes de la capsule (en utilisant la fonction de gestion de grappe);
- la suppression de la capsule par suppression de toutes les grappes de la capsule, suivie par la suppression du gestionnaire de la capsule.

NOTE 2 – Des interfaces de gestion de capsule différentes peuvent donc avoir des types différents selon les fonctions qu'elles fournissent.

Le comportement d'un gestionnaire de capsule est régi par les politiques de gestion de cette capsule.

Dans le cadre de l'architecture définie par le présent Modèle de référence, la fonction de gestion de capsule est utilisée par la fonction de désactivation et de réactivation, la fonction de pose de point de reprise et de reprise et la fonction de migration.

12.4.1 Instanciation de gabarit de grappe

L'instanciation de gabarit de grappe est paramétrée par un gabarit de grappe et comprend les étapes suivantes:

- l'instanciation d'une grappe à partir d'un gabarit de grappe;
- l'introduction d'un gestionnaire de grappe pour la nouvelle grappe;
- l'allocation d'un identificateur d'interface de gestion de grappe pour le nouveau gestionnaire de grappe;
- la liaison de la nouvelle grappe avec d'autres objets conformément aux règles du langage d'ingénierie et aux informations de liaison contenues dans le gabarit de grappe.

NOTE – La réactivation et la reprise de grappe sont des cas particuliers d'instanciation de grappe, où le gabarit de grappe est un point de reprise de grappe.

Un gabarit de grappe peut contenir des informations spécifiques à un domaine. Si le gabarit doit être instancié dans un autre domaine, ces informations doivent être transformées. Les références d'interface d'ingénierie contenues dans la grappe doivent notamment être transformées si la grappe est instanciée dans un domaine de gestion de référence d'interface d'ingénierie différent.

12.4.2 Suppression de capsule

La suppression de capsule conduit à la suppression du gestionnaire de capsule et peut entraîner la suppression de talons, d'objets lieurs, d'objets protocoles et d'intercepteurs qui supportaient des objets de la capsule ou de son gestionnaire.

13 Fonctions de coordination

13.1 Fonction de notification d'événement

La fonction de notification d'événement enregistre et met à disposition l'historique des événements.

13.1.1 Concepts

13.1.1.1 Historique: objet représentant des actions significatives.

13.1.2 Règles

Les producteurs d'événement interagissent avec la fonction de notification d'événement pour créer des historiques. La fonction de notification d'événement notifie aux objets consommateurs d'événement la disponibilité des historiques.

La fonction de gestion d'événement supporte un ou plusieurs types d'historique et dispose d'une politique de gestion d'événement qui détermine le comportement de la fonction et en particulier:

- quels objets peuvent créer des historiques;
- à quels objets est notifiée la création d'un nouvel historique;
- les moyens par lesquels de telles notifications se produisent;
- les exigences de persistance et de stabilité pour les historiques;
- les relations d'ordre entre les interactions avec les objets producteurs d'événement et les objets consommateurs d'événement.

Un consommateur d'événement interagit avec la fonction de notification d'événement pour enregistrer la notification de nouveaux historiques. Suivant la politique de gestion d'événement, l'interaction peut:

- établir des liaisons avec les historiques disponibles;
- permettre la communication des historiques créés à la suite de l'interaction.

NOTE – Les historiques exigeant de la persistance stricte et de la stabilité peuvent être pris en charge en utilisant les fonctions de transaction et de duplication. Les notifications d'ordonnancement et de multidiffusion peuvent être prises en charge par la fonction de groupe.

13.2 Fonction de pose de point de reprise et de reprise

La fonction de pose de point de reprise et de reprise coordonne la pose de point de reprise et la reprise pour des grappes subissant des défaillances.

La fonction de pose de point de reprise et de reprise incorpore des politiques déterminant:

- quand des points de reprise de grappe doivent être posés;
- quand les grappes doivent faire l'objet d'une reprise;
- où les grappes doivent faire l'objet d'une reprise;
- où les points de reprise doivent être stockés;
- quel point de reprise fait l'objet d'une reprise.

La pose de point de reprise et la reprise de grappes sont soumises aux politiques de sécurité relatives à ces grappes, notamment, la position où le point de reprise est stocké et où il fait l'objet d'une reprise.

Dans le cadre de l'architecture définie par le présent Modèle de référence, la fonction de pose de point de reprise et de reprise utilise la fonction de gestion de grappe et la fonction de gestion de capsule.

13.2.1 Pose de point de reprise

La pose de point de reprise est de la responsabilité de la fonction de gestion d'objet et de la fonction de gestion de grappe. La pose de point de reprise d'une grappe est coordonnée par son gestionnaire de grappe: dans un premier temps, le gestionnaire de grappe utilise la fonction de gestion d'objet pour obtenir un point de reprise de chaque objet de la grappe; à partir de ces points de reprise d'objet, le gestionnaire de grappe construit un point de reprise de grappe qu'il rend persistant en utilisant la fonction de stockage.

Selon la politique de pose de point de reprise, la pose de point de reprise d'une grappe peut conduire à la création de points de reprise pour d'autres grappes qui participent à des activités communes à la grappe dont le point de reprise a été posé. La pose de point de reprise est soumise aux règles de cohérence suivantes:

- la grappe initiale doit être dans un état cohérent avant qu'il y ait pose de point de reprise;
- il doit y avoir une cohérence entre tous les points de reprise de grappe des différentes grappes qui sont posées conjointement (par exemple, tous les points de reprise doivent refléter le même ensemble d'interactions se produisant entre les grappes);
- quand un point de reprise de grappe est posé, des points de reprise doivent être posés sur toutes les autres grappes ayant des contraintes de point de reprise avec cette grappe. Cette règle doit être appliquée de façon récursive pour obtenir un ensemble fermé de grappes sur lesquelles on peut poser des points de reprise de façon cohérente.

13.2.2 Reprise

Une grappe peut faire l'objet d'une reprise:

- dans la capsule à partir de laquelle lui avait été préalablement posé un point de reprise;
- ou bien, dans une autre capsule (par exemple, quand la capsule dans laquelle a été posé le point de reprise a connu une défaillance par la suite).

La reprise d'une grappe est de la responsabilité du gestionnaire de grappe de cette grappe, ou, en l'absence de ce gestionnaire, d'un gestionnaire de capsule. La fonction de pose de point de reprise et de reprise interagit avec le gestionnaire de grappe ou le gestionnaire de capsule, selon le cas, pour instancier le point de reprise de grappe. Avant d'effectuer la reprise d'une grappe, la fonction de pose de point de reprise et de reprise doit s'assurer que la grappe a été supprimée (par exemple, suite à une défaillance). Les objets liés à des grappes ayant fait l'objet d'une reprise doivent être capables de détecter si la grappe a subi une reprise à partir d'un point de reprise (par exemple, de telle sorte qu'ils puissent réitérer des interactions qui se sont produites après l'établissement du point de reprise).

La reprise d'un grappe peut conduire à la reprise d'autres grappes, par exemple, celles qui constituent un point de reprise joint avec la grappe ayant fait l'objet d'une reprise.

13.3 Fonction de désactivation et de réactivation

La fonction de désactivation et réactivation coordonne la désactivation et la réactivation des grappes. Elle incorpore des politiques déterminant:

- quand des grappes doivent être désactivées;
- où le point de reprise associé à une désactivation doit être stocké;
- quand des grappes doivent être réactivées;
- quel point de reprise doit être réactivé (par exemple, le plus récent);
- où les grappes doivent être réactivées.

La désactivation et la réactivation des grappes sont soumises aux politiques de sécurité associée à ces grappes, en particulier, l'emplacement où le point de reprise est stocké et ou il est réactivé. Dans le cadre de l'architecture définie par le présent Modèle de référence, la fonction de désactivation et de réactivation utilise la fonction de gestion d'objet, la fonction de gestion de grappe et la fonction de gestion de capsule. La fonction de désactivation et de réactivation est utilisée par la fonction de migration.

13.3.1 Désactivation

La désactivation d'une grappe est une fonction de gestion de grappe et comprend les étapes suivantes:

- le gestionnaire de la grappe impliquée interagit avec chacun des objets de sa grappe afin d'obtenir un point de reprise pouvant être utilisé pour établir un point de reprise de grappe;
- le gestionnaire de grappe rend le point de reprise de grappe persistant en utilisant la fonction de stockage;
- le gestionnaire de grappe supprime la grappe (et peut lui-même être supprimé).

13.3.2 Réactivation

La fonction de désactivation et de réactivation réactive une grappe en utilisant la fonction de gestion de capsule pour instancier un point de reprise de la grappe appartenant à la capsule cible (et inclut la création d'un gestionnaire de grappe pour cette grappe). La capsule cible peut être la capsule dans laquelle elle a été précédemment désactivée, ou une autre capsule (par exemple, pour équilibrer la charge de l'infrastructure sur plusieurs nœuds).

13.4 Fonction de groupe

La fonction de groupe fournit les mécanismes nécessaires pour coordonner les interactions d'objets sur des liaisons multipoints.

13.4.1 Concepts

13.4.1.1 Groupe d'interaction: sous-ensemble d'objets participant à une liaison gérée par la fonction de groupe.

13.4.1.2 Règles

Pour chaque ensemble d'objets liés entre eux dans un groupe d'interactions, la fonction de groupe gère:

- **l'interaction**: pour décider quels membres du groupe participent à quelles interactions, suivant la politique d'interaction;
- **le récolement**: pour dériver une vue cohérente d'interactions (y compris des interactions défaillantes), suivant la politique de récolement;
- **l'ordonnancement**: pour s'assurer que les interactions entre les membres du groupe sont correctement ordonnées, suivant une politique d'ordonnancement;
- **l'appartenance**: pour traiter les défaillances des membres, leurs reprises, l'addition ou le retrait de membres, suivant une politique d'appartenance.

NOTE – Le comportement de l'objet de liaison liant les membres du groupe détermine la sémantique de l'interaction.

13.5 Fonction de duplication

La fonction de duplication est un cas particulier de la fonction de groupe dans laquelle les membres d'un groupe ont un comportement compatible (par exemple, parce qu'ils représentent des répliques provenant du même gabarit d'objet). La fonction de duplication garantit que le groupe apparaît aux autres objets comme s'il était un objet unique, en s'assurant que tous les membres participent à toutes les interactions dans le même ordre.

La politique d'appartenance d'un groupe dupliqué peut autoriser l'augmentation ou la diminution du nombre de membres de ce groupe dupliqué. L'augmentation de la taille d'un groupe dupliqué produit le même effet que si un membre du groupe avait été cloné puis ajouté au groupe dans une action atomique unique.

Pour que la fonction de duplication soit appliquée à une grappe, les objets appartenant à la grappe sont dupliqués et configurés dans un ensemble de grappes identiques. Les objets correspondants dans chaque grappe dupliquée constituent les groupes dupliqués. Une grappe dupliquée est donc un ensemble coordonné de groupes dupliqués.

La fonction de duplication est utilisée par la fonction de migration.

13.6 Fonction de migration

La fonction de migration coordonne la migration d'une grappe d'une capsule à une autre. Elle utilise la fonction de gestion de grappe et la fonction de gestion de capsule et incorpore des politiques déterminant quand des grappes doivent être migrées et où elles peuvent être localisées.

Il y a deux moyens de migration:

- la duplication; ou
- la désactivation dans une capsule, suivie par une réactivation dans une autre.

13.6.1 Duplication

La migration d'une grappe en utilisant la fonction de duplication comprend la série d'actions suivante:

- la grappe d'origine est considérée comme groupe dupliqué de grappe de taille un;
- une copie de la grappe d'origine est créée dans la capsule destinataire avec un gestionnaire de grappe;
- les objets des deux grappes sont constitués en groupes dupliqués (de taille deux);
- les objets de la grappe d'origine sont enlevés des groupes d'objets (en laissant les groupes de taille un);
- la grappe d'origine (et son gestionnaire) est supprimée.

13.6.2 Désactivation et réactivation

La migration d'une grappe par désactivation et réactivation est coordonnée par le gestionnaire de la grappe et inclut la désactivation de la grappe à sa position d'origine, suivie par la réactivation de la grappe à sa nouvelle position.

13.7 Fonction de transaction

13.7.1 Concepts

13.7.1.1 Transaction: activité qui entraîne un ensemble de changements d'état d'objet en conformité à un schéma dynamique (et les contraintes issues de son schéma d'invariant).

13.7.1.2 Action significative: action au sein d'une transaction qui entraîne un changement d'état significatif pour la transaction.

13.7.1.3 Visibilité: mesure dans laquelle une transaction peut accéder à l'état d'un objet de façon concurrente avec d'autres transactions.

13.7.1.4 Capacité de reprise: mesure dans laquelle des changements d'état d'objet résultant de transactions défaillantes sont annulés.

13.7.1.5 Permanence: mesure dans laquelle des défaillances peuvent affecter des changements d'état d'objet suite à l'achèvement de transactions.

13.7.2 Règles

La fonction de transaction coordonne et contrôle un ensemble de transactions afin d'atteindre un niveau spécifié de visibilité, de capacité de reprise et de permanence.

La fonction de transaction:

- interagit avec des objets pour superviser l'occurrence des actions significatives, l'annulation des effets des actions significatives et la causalité des actions significatives;
- détermine si des actions significatives sont en conflit;
- interagit avec des objets pour prévoir l'occurrence d'actions significatives, afin d'éviter des conflits;
- interagit avec des objets pour annuler les effets d'actions significatives qui se sont produites, afin de résoudre les conflits.

La fonction de transaction est soumise à des politiques déterminant:

- quelles actions sont des actions significatives;
- quelles actions significatives sont en conflit;
- quelles actions significatives doivent être annulées pour résoudre les conflits.

13.8 Fonction de transaction ACID

La fonction de transaction ACID est un cas particulier de la fonction de transaction générale dans laquelle:

- la visibilité se réduit à l'isolation des transactions les unes des autres;
- la capacité de reprise se réduit à l'obligation d'atomicité des transactions;

- la permanence se réduit à l'obligation de durabilité (c'est-à-dire, la stabilité) des changements d'état liés aux transactions;
- la cohérence est obtenue par l'exécution correcte de transactions respectant les propriétés d'atomicité, d'isolation et de durabilité en conformité avec les schémas dynamiques et d'invariant associés.

Les actions significatives de la fonction de transaction ACID sont:

- l'initialisation de la transaction;
- la validation de la transaction;
- l'interruption de la transaction;
- l'accès à l'état;
- la modification de l'état;
- l'annulation de n'importe laquelle de ces actions.

Les politiques de transaction s'expriment par des règles de sérialisation de transactions.

13.9 Fonction ramasse-miettes

La fonction ramasse-miettes (ou fonction de suivi de références) supervise le transfert de références d'interface d'ingénierie entre objets d'ingénierie appartenant à différentes grappes, afin de déterminer quand l'infrastructure associée aux interfaces d'ingénierie n'est plus requise (c'est-à-dire, lorsque aucun objet appartenant à une autre grappe n'est en mesure de se lier à l'interface référencée).

La fonction ramasse-miettes met à jour, dans le cadre de son domaine d'application:

- l'information de possession des références d'interface d'ingénierie;
- l'information d'existence des interfaces.

La fonction ramasse-miettes:

- est notifiée par les talons lorsqu'une référence d'interface d'ingénierie est transférée entre grappes;
- est notifiée par un gestionnaire de grappe lorsque toutes les copies d'une référence d'interface d'ingénierie sont supprimées dans sa grappe;
- détecte lorsque aucune copie d'une référence d'interface d'ingénierie n'est conservée en dehors de la grappe supportant l'interface d'ingénierie de référence et envoie des notifications au gestionnaire de grappe correspondant;
- envoie des notifications aux gestionnaires de capsules dans lesquels il y a des porteurs de références d'interface d'ingénierie concernant une interface défaillante ou supprimée.

NOTE – La notification des événements ramasse-miettes peut être réalisée en utilisant la fonction de notification d'événement.

La fonction ramasse-miettes est régie par les politiques de gestion de référence d'interface d'ingénierie des domaines auxquels elle s'applique.

14 Fonctions de dépôt

14.1 Fonction de stockage

La fonction de stockage stocke des données.

14.1.1 Concepts

14.1.1.1 Dépôt de données: objet fournissant la fonction de stockage.

14.1.1.2 Interface dépôt: interface d'un dépôt de données qui permet d'accéder aux données.

14.1.2 Règles

Un dépôt de données stocke des ensembles de données. Chaque ensemble de données est associé à une interface dépôt créée lorsque les données sont stockées. Une interface dépôt assure des fonctions destinées:

- à obtenir une copie des données stockées par cette interface;
- à modifier les données associées à cette interface;
- à supprimer l'interface dépôt et ses données associées.

NOTE – Les objets incorporent à la fois des actions et des états (c'est-à-dire, des traitements et des données); ils sont intrinsèquement persistants. La fonction de pose de point de reprise et de reprise ou la fonction de duplication peuvent être utilisées dans une infrastructure de fonction de stockage pour assurer la stabilité. Les fonctions peuvent utiliser d'autres instances de la fonction de stockage en tant que dépôt de points de reprise de grappe.

14.2 Fonction de gestion de base d'information

La fonction de gestion de base d'information gère un dépôt d'informations décrit par un schéma d'information et comprend certains ou tous les éléments suivants:

- modification et mise à jour du schéma d'information;
- requête auprès du dépôt à l'aide d'un langage de requête;
- modification et mise à jour du dépôt.

La forme et la nature des requêtes et réponses dépendent du langage de requête. La fonction de gestion de base d'information n'autorise pas de modification ni de mise à jour du dépôt d'informations qui soient incohérentes avec son schéma.

La fonction de gestion de base d'information peut être modélisée sous forme de dépôt d'objets (avec des interfaces de traitement) en correspondance avec des entités et des relations d'un système ODP. Ces objets fournissent des opérations en vue:

- de définir les attributs, les propriétés et les relations des objets du dépôt;
- d'ajouter et de supprimer des objets du dépôt;
- de définir et de supprimer des attributs, des propriétés et des relations concernant des objets sélectionnés dans le dépôt;
- de sélectionner les objets qui satisfont à un prédicat (à savoir un type) spécifié en termes d'attributs, de propriétés et de relations.

NOTES

1 La fonction de gestion de base d'information peut inférer des relations additionnelles venant s'ajouter à celles directement instanciées.

2 La fonction de gestion de base d'information peut servir à maintenir la relation entre une grappe et son point de reprise (c'est-à-dire une interface de stockage où un point de reprise peut être posé ou atteint).

3 Afin de permettre la reprise suite à l'interaction avec une grappe défaillante, la fonction de gestion de base d'information peut servir à maintenir une relation entre une grappe et l'interface à laquelle la reprise de la grappe peut être requise.

4 Afin de permettre la réactivation suite aux interactions avec une grappe désactivée, la fonction de gestion de base de l'information peut être utilisée pour maintenir une relation entre la grappe désactivée et l'interface à laquelle la réactivation de la grappe peut être requise.

5 L'information nécessaire à un relocalisateur pour valider ou mettre à jour une référence d'interface peut être tenue à jour par la fonction de gestion de base d'information.

14.3 Fonction de relocalisation

La fonction de relocalisation gère un dépôt des positions des interfaces, y compris les positions des fonctions de gestion liées à la grappe fournissant ces interfaces.

14.3.1 Concepts

14.3.1.1 Relocalisateur: objet assurant la fonction de relocalisation.

14.3.2 Règles

Un relocalisateur possède un annuaire des positions des interfaces qui ont changé de position suite à des activités de gestion de domaine de communication (par exemple, le changement de l'adresse réseau d'un nœud) ou à des activités de gestion de grappe, telles que la désactivation, la migration, la duplication ou la reprise d'un point de reprise de grappe.

Une interface peut être associée à un relocalisateur; les relocalisateurs peuvent être spécifiques à une seule interface ou communs à plusieurs interfaces. Lorsque le domaine d'application d'un relocalisateur couvre plus d'un domaine de gestion d'interface d'ingénierie, le mécanisme permettant d'accéder au relocalisateur doit indiquer clairement le domaine de gestion de référence d'interface qui s'applique.

Lorsqu'un relocalisateur est associé à une interface, les activités qui entraînent le changement de la position des interfaces doivent informer ce relocalisateur des nouvelles positions; en particulier, un gestionnaire de grappe doit notifier au relocalisateur quand la grappe est relocalisée et ce pour chaque interface de chacun des objets du gestionnaire de grappe. Seules les interfaces d'objets dont la position a changé, ont besoin d'être enregistrées. Quand une référence d'interface d'ingénierie doit rester valide même si l'objet qui la fournit se trouve dans une grappe désactivée dans une grappe défaillante mais pour laquelle on a auparavant poser un point de reprise, le relocalisateur doit comprendre une politique d'utilisation des fonctions de coordination. Cette politique doit permettre de restaurer la grappe, par réactivation ou reprise selon le cas, lorsqu'un autre objet essaie de valider la référence.

Un relocalisateur autorise des interactions permettant:

- l'enregistrement des changements de position d'une interface identifiée par une référence d'interface d'ingénierie;
- la validation de la position d'une interface identifiée par une référence d'interface d'ingénierie (comprenant, si nécessaire, la restauration de la grappe contenant l'objet supportant l'interface);
- la définition de politiques d'interaction avec des fonctions de coordination (par exemple, la fonction de désactivation et de réactivation) lors de la validation de la position d'une interface identifiée par une référence d'interface d'ingénierie.

NOTE – Un objet validant une référence d'interface d'ingénierie peut conserver les résultats de la validation pour optimiser l'utilisation future de la référence.

14.4 Fonction de dépôt de types

La fonction de dépôt de types gère un magasin de spécifications de types et de relations de types. Elle possède une interface pour chaque spécification de type qu'elle stocke.

14.4.1 Règles

La fonction de dépôt de types peut être amenée à gérer des relations entre types autres que de celles dérivables de la comparaison de spécification de types. Un dépôt de types interdit l'établissement de relations incohérentes.

Les spécifications de type sont immuables.

La fonction de dépôt de types inclut la création de type et des interfaces de type associées.

Une interface dépôt de types concernant un type spécifique assure des fonctions autorisant:

- la recherche de la spécification d'un type;
- la vérification des relations entre un type et les autres types;
- la recherche des relations entre le type et les autres types.

NOTE – Les relations de sous-typage concernant les types de signature de traitement sont déterminées par les règles de sous-typage de signature du 7.2.4. Il n'est pas exigé qu'un dépôt de types puisse vérifier la relation des sous-typages entre types de signature. Lorsque les types de signature sont inclus dans un dépôt de types, le dépôt de types ne peut être amené à gérer des règles typage de signature contredisant les dispositions du 7.2.4.

14.5 Fonction de courtage

La fonction de courtage permet de rendre public et de rechercher des interfaces.

14.5.1 Concepts

14.5.1.1 Offre de service: information sur une interface incluant un identificateur de l'interface et son type de signature d'interface de traitement.

NOTES

1 L'identificateur permet la liaison avec l'interface.

2 La signature de traitement permet au courtier de garantir que l'importation de service sélectionne les offres de service qui interagiront conformément aux attentes de l'objet importateur.

3 Des informations supplémentaires sur l'offre de service peuvent permettre d'assurer une plus grande discrimination que celles incorporées dans les signatures d'interface.

14.5.1.2 Exportation de service: interaction avec la fonction de courtage qui permet d'ajouter une offre de service à un ensemble donné d'offres de service afin de la rendre publique.

14.5.1.3 Importation de service: interaction avec la fonction de courtage qui permet de rechercher dans un ensemble donné d'offres de service des interfaces satisfaisant à un type spécifié.

14.5.2 Règles

La fonction de courtage assure l'importation et l'exportation de service et son comportement est régi par une politique de courtage qui définit les règles de satisfaction par une offre de service des critères de sélection identifiés dans l'importation de service. Lors de l'importation de service, une fonction de courtage ne doit sélectionner que les offres qui satisfont à la politique de la fonction de courtage, à la politique de l'exportateur de l'offre de service et à la politique de l'importateur de l'offre de service. L'importation de service entraîne la vérification du sous-type de la signature de l'interface de traitement (en particulier la vérification de relation de sous-typage). Elle peut, en plus, entraîner des vérifications supplémentaires de vérification, notamment des vérifications de la caractéristique de comportement et des contraintes d'environnement.

15 Fonctions de sécurité

15.1 Concepts

Les concepts suivants sont communs à toutes les fonctions de sécurité.

15.1.1 Politique de sécurité: ensemble de règles qui s'appliquent à un ou plusieurs ensembles d'activités concernant un ou plusieurs ensembles d'objets.

15.1.2 Autorité de sécurité: administrateur responsable de la mise en œuvre d'une politique de sécurité.

15.1.3 Domaine de sécurité: domaine dans lequel les membres sont obligés de se soumettre à une politique de sécurité établie et administrée par une autorité de sécurité.

NOTE – L'autorité de sécurité représente l'objet de contrôle du domaine de sécurité.

15.1.4 Politique d'interaction de sécurité: aspects des politiques de sécurité de différents domaines de sécurité qui sont nécessaires pour que des interactions se produisent entre ces domaines.

15.2 Fonction de contrôle d'accès

La fonction de contrôle d'accès empêche des interactions non autorisées avec un objet. Elle comprend une **fonction de décision du contrôle d'accès** et une **fonction de mise à exécution du contrôle d'accès**. Dans le contexte du contrôle d'accès, les objets peuvent jouer les rôles de **cible** ou d'**initiateur**. La fonction requiert des **informations de contrôle d'accès** concernant la cible, l'initiateur et l'interaction.

L'initiateur demande une interaction avec la cible en utilisant la fonction de contrôle d'accès. La fonction de décision du contrôle d'accès détermine si l'accès est autorisé ou refusé en fonction de l'information de contrôle d'accès et la décision est exécutée par la fonction de mise à exécution du contrôle d'accès.

NOTE – La fonction de décision du contrôle d'accès et la fonction de mise à exécution du contrôle d'accès peuvent être assurées par l'objet qui remplit le rôle de cible, ou par d'autres objets.

15.3 Fonction d'audit de sécurité

La fonction d'audit de sécurité assure la supervision et la collecte d'informations concernant les actions relatives à la sécurité, ainsi que l'analyse ultérieure de ces informations afin de réviser les politiques de sécurité, les contrôles et les procédures.

La fonction d'audit de sécurité inclut chacun des éléments suivants:

- **fonction de collecte d'alarme**;
- **fonction d'examen d'alarme**;
- **fonction d'analyse de journal d'audit**;
- **fonction d'archivage de journal d'audit**;
- **fonction d'enregistrement d'audit**;
- **fonction d'examen de journal d'audit**;
- **fonction de collecte de journal d'audit**.

15.4 Fonction d'authentification

La fonction d'authentification garantit la validité de l'identité déclarée d'un objet. Dans le contexte d'authentification, les objets remplissent les rôles suivants:

- **mandant**;
- **déclarant**;
- **tierce-partie de confiance**.

L'authentification d'accès utilise l'**information d'authentification d'échange**.

NOTES

1 Un objet quelconque d'un système ODP peut être le mandant pour l'authentification, notamment les objets qui modélisent des êtres humains et ceux qui modélisent les systèmes informatiques.

2 L'objet initialisant une authentification n'est pas nécessairement le déclarant.

Il y a deux formes d'authentification:

- l'**authentification d'entités homologues**, qui fournit la confirmation de l'identité d'un mandant dans le contexte d'une relation de communication;
- l'**authentification de l'origine des données**, qui fournit la confirmation de l'identité du principal responsable d'une unité de données spécifique.

NOTE – Les mécanismes d'authentification sont classés par catégorie dans la Rec. UIT-T X.811 | ISO/CEI 10181-2.

Dans une authentification impliquant deux objets, l'un ou l'autre ou les deux à la fois peuvent remplir le rôle de déclarant. Lorsque les deux objets ont le rôle de déclarant, le style de l'authentification est connu sous le nom d'**authentification mutuelle**. Les informations d'authentification d'échange sont transférées de l'objet initiateur à l'objet répondeur et d'autres informations d'authentification d'échange peuvent alors être transférées dans le sens inverse. Des échanges supplémentaires peuvent également se produire: des mécanismes d'authentification différents exigent des nombres d'échanges différents. L'authentification d'entités homologues implique toujours une interaction avec le déclarant. L'authentification de l'origine des données ne nécessite pas d'interaction avec le déclarant.

Un déclarant prend en charge des opérations permettant d'acquérir les informations nécessaires à une instance d'authentification et de générer l'information d'authentification d'échange. Un vérificateur dispose d'opérations permettant d'acquérir l'information nécessaire à une instance d'authentification, et de contrôler l'information d'authentification d'échange reçue et/ou de la générer. Des informations peuvent être échangées avec un serveur d'authentification et avec un déclarant ou bien un vérificateur, ou les deux à la fois, avant ou durant les échanges d'authentification.

La fonction d'authentification peut utiliser la fonction de gestion de clé.

15.5 Fonction d'intégrité

La fonction d'intégrité détecte et empêche la création, l'altération et la suppression non autorisées des données.

La fonction d'intégrité inclut les fonctions suivantes:

- **protéger**;
- **valider**;
- **enlever la protection**.

Dans le contexte d'intégrité, les objets remplissent les rôles suivants:

- **source de données à intégrité protégée**;
- **destinataire de données à intégrité protégée**.

Les données à intégrité protégée circulent de la source au destinataire. Une source de données à intégrité protégée fournit une interface assurant la fonction protéger. Un destinataire de données à intégrité protégée supporte une interface assurant les fonctions valider et enlever la protection.

La fonction d'intégrité peut utiliser la fonction de gestion de clé.

15.6 Fonction de confidentialité

La fonction de confidentialité protège contre la divulgation non autorisée d'information.

La fonction de confidentialité inclut les fonctions **cacher** et **révéler**.

Dans le contexte de confidentialité, les objets remplissent l'un ou les deux rôles suivants:

- **source de données à confidentialité protégée**;
- **destinataire de données à confidentialité protégée**.

Les données à confidentialité protégée circulent de la source au destinataire. Une source de données à confidentialité protégée supporte une interface assurant la fonction cacher. Un destinataire de données à confidentialité protégée supporte une interface assurant la fonction révéler.

La fonction de confidentialité peut utiliser la fonction de gestion de clé.

15.7 Fonction de non-répudiation

La fonction de non-répudiation garantit qu'un objet impliqué dans une interaction a bien participé à tout ou partie de cette interaction.

Dans le contexte de non-répudiation, les objets remplissent les rôles suivants:

- **source (de données non-répudiables)**;
- **destinataire (de données non-répudiables)**;
- **générateur de preuve**;
- **utilisateur de preuve**;
- **vérificateur de preuve**;
- **demandeur de service de non-répudiation**;
- **notaire**;
- **adjudicateur**.

La fonction de non-répudiation utilise le témoignage de non-répudiation. Dans la non-répudiation avec preuve de l'origine, la source remplit le rôle de générateur de preuve de non-répudiation pour l'interaction de création et inclut cette preuve dans un accusé de réception de participation à l'interaction. Le destinataire remplit le rôle d'utilisateur de preuve et utilise les services d'un vérificateur de preuve (qui peut être lui-même) pour s'assurer de la pertinence de la preuve. Dans la non-répudiation avec preuve de la remise, le destinataire remplit le rôle de générateur de preuve de non-répudiation pour l'interaction de remise et inclut cette preuve dans un accusé de réception de participation à l'interaction. La source remplit le rôle d'utilisateur de preuve et utilise les services d'un vérificateur de preuve (qui peut être lui-même) pour s'assurer de la pertinence de la preuve.

Un notaire fournit des fonctions dont ont besoin la source et/ou le destinataire. Elles peuvent inclure des opérations notariales, l'estampillage du temps, la supervision, la certification, la génération de certificat, la génération de signature, la vérification et la remise de signature, tels que définis dans la Rec. UIT-T X.813 | ISO/CEI 10181-4.

Dans le cas d'un litige, un adjudicateur collecte des informations et des preuves provenant des parties en litige (et optionnellement des notaires) et utilise une fonction de résolution comme défini dans la Rec. UIT-T X.813 | ISO/CEI 10181-4.

La fonction de non-répudiation peut utiliser la fonction de gestion de clé.

15.8 Fonction de gestion de clé

La fonction de gestion de clé fournit des facilités pour la gestion des clés de chiffrement et inclut tous les éléments suivants:

- **génération de clé**;
- **enregistrement de clé**;
- **certification de clé**;
- **invalidation de clé**;
- **répartition de clé**;

- **stockage de clé**;
- **archivage de clé**;
- **suppression de clé**.

Dans le contexte de gestion de clé, les objets peuvent avoir un ou plusieurs des rôles suivants:

- **autorité de certification**;
- **centre de distribution de clé**;
- **centre de traduction de clé**.

Une autorité de certification est une tierce partie de confiance qui crée et attribue des certificats comme défini dans l'ISO/CEI 11770-1 Gestion de clé. Un centre de distribution de clé permet d'établir en toute sécurité des informations de gestion de clé entre les objets autorisés à les obtenir. Un centre de traduction de clé est une forme spécifique de centre de distribution de clé qui a établi des informations de gestion de clés entre des objets appartenant à différents domaines de sécurité.

16 Transparence à la répartition dans ODP

Dans les systèmes ODP, la transparence à la répartition est sélective. Le présent Modèle de référence décrit comment réaliser les transparences à la répartition suivantes:

- transparence d'accès;
- transparence aux défaillances;
- transparence à la localisation;
- transparence à la migration;
- transparence à la persistance;
- transparence à la relocalisation;
- transparence à la duplication;
- transparence aux transactions.

Les normes ODP peuvent définir à la fois:

- des raffinements des descriptions du présent Modèle de référence;
- des transparences à la répartition additionnelles requises par ces normes.

Les transparences sont définies comme des contraintes qui s'exercent sur la mise en correspondance d'une spécification de traitement contenant un schéma de transparence, avec une spécification qui utilise des structures d'ingénierie et des fonctions ODP spécifiques afin de masquer les mécanismes requis.

Le comportement des talons, des objets lieurs, des objets protocoles et des intercepteurs au sein des canaux est déterminé par la combinaison de transparences à la répartition s'appliquant au canal (par exemple, les transparences d'accès, à la relocalisation).

Dans certaines normes réalisables qui nécessitent plus d'une transparence à la répartition, des règles de composition des objets mettant en œuvre ces transparences peuvent être spécifiées. Dans d'autres normes réalisables qui nécessitent plus d'une transparence à la répartition, un objet unique peut suffire à assurer ces différentes transparences de façon continue.

Les descriptions de transparence du présent Modèle de référence prennent au moins en charge les combinaisons suivantes:

a) transparence d'accès et à la localisation;

b) transparence à la relocalisation et à a);

c) transparence à la migration et à b);

d) transparence aux ressources et à b);

e) transparence aux défaillances et à b);

f) transparence aux transactions et à a);

g) transparence aux transactions et à b).

16.1 Transparence d'accès

La transparence d'accès masque les différences de représentation de données et de mécanismes d'invocation pour permettre l'interfonctionnement entre les objets.

La transparence d'accès est fournie par la sélection d'une structure de canal appropriée (dans laquelle, par exemple, les talons assurent des conversions appropriées, telles que la conversion dans une représentation de données canonique).

16.2 Transparence aux défaillances

La transparence aux défaillances masque à un objet la défaillance et la reprise éventuelle d'autres objets ou de lui-même, afin de garantir la tolérance aux fautes.

16.2.1 Concepts

16.2.1.1 Schéma de stabilité: spécification des modes de défaillances qu'un objet n'exhibera pas.

16.2.2 Règles

Le raffinement d'une spécification de traitement associée à la transparence aux défaillances peut satisfaire à un schéma de stabilité en utilisant l'une des méthodes suivantes:

- en plaçant l'objet sur un nœud qui possède une infrastructure qui exclut les défaillances spécifiées;
- en utilisant la fonction de pose de point de reprise et de reprise pour rendre l'objet stable;
- en utilisant la fonction de duplication pour rendre l'objet stable.

16.2.2.1 Duplication

Dans le cas de la duplication, le raffinement d'une spécification de traitement associée à la transparence aux défaillances comprend les étapes suivantes:

- la définition pour l'objet de traitement d'une interface de gestion d'objet supportant la pose de point de reprise et la suppression d'objet;
- l'introduction d'une fonction de duplication;
- la mise en place d'une politique de duplication pour les grappes contenant l'objet;
- l'association d'un relocalisateur qui supporte la duplication avec chacune des interfaces de l'objet.

Dans le cadre de l'architecture définie par le présent Modèle de référence, la transparence aux défaillances basée sur la duplication d'une grappe exige la transparence à la relocalisation.

16.2.2.2 Pose de point de reprise et reprise

Dans le cas de la pose de point de reprise et de la reprise, le raffinement d'une spécification de traitement associée à la transparence aux défaillances inclut chacune des étapes suivantes:

- la définition pour l'objet de traitement d'une interface de gestion d'objet supportant la pose de point de reprise et la suppression d'objet;
- l'introduction d'une fonction de pose de point de reprise et de reprise;
- la mise en place d'une politique de point de reprise et de reprise pour les grappes contenant l'objet;
- l'association d'un relocalisateur qui supporte la reprise avec chacune des interfaces de l'objet.

La transparence aux défaillances basée sur la pose de point de reprise et la reprise d'une grappe exige la transparence à la relocalisation.

16.3 Transparence à la localisation

La transparence à la localisation masque l'utilisation d'informations concernant la position dans l'espace lors de l'identification et de la liaison avec les interfaces. Cela permet aux objets d'accéder aux interfaces sans utiliser d'information de position.

16.4 Transparence à la migration

La transparence à la migration masque à un objet la capacité d'un système à changer la position de cet objet.

16.4.1 Concepts

16.4.1.1 Schéma de mobilité: spécification imposant des contraintes sur la mobilité d'un objet.

Un schéma de mobilité inclut:

- des contraintes de délais sur les interactions avec l'objet;
- des contraintes de performance sur les fils d'exécution de l'objet;
- des contraintes de sécurité sur la position de l'objet.

16.4.2 Règles

La transparence à la migration est assurée par l'utilisation de la fonction de migration pour coordonner la position d'un objet afin de satisfaire à un schéma de mobilité. Le raffinement de la spécification de traitement associée à la transparence à la migration inclut les étapes suivantes:

- la définition pour l'objet de traitement d'une interface de gestion d'objet supportant la pose de point de reprise et la suppression d'objet;
- l'introduction d'une fonction de migration;
- la mise en place d'une politique de migration pour les grappes contenant l'objet;
- l'association d'un relocalisateur avec chacune des interfaces de l'objet.

Dans le cadre de l'architecture définie par le présent Modèle de référence, la transparence à la migration exige une transparence à la relocalisation pour la grappe.

16.5 Transparence à la persistance

La transparence à la persistance masque à un objet la désactivation et la réactivation d'autres objets (ou de lui-même).

16.5.1 Concepts

16.5.1.1 Schéma de persistance: spécification de contraintes sur l'utilisation de fonctions spécifiques de traitement, de stockage et de communication.

16.5.2 Règles

La transparence à la persistance est assurée par l'utilisation de la fonction de réactivation et désactivation pour coordonner la désactivation et la réactivation de grappes afin de satisfaire à un schéma de persistance. Le raffinement de la spécification de traitement associée à la transparence à la persistance inclut les étapes suivantes:

- la définition pour l'objet de traitement d'une interface de gestion d'objet supportant la pose de point de reprise et la destruction d'objet;
- l'introduction d'une fonction de désactivation et de réactivation;
- la mise en place d'une politique de désactivation et de réactivation pour les grappes contenant l'objet;
- l'association d'un relocalisateur qui supporte la réactivation avec chacune des interfaces de l'objet.

Dans le cadre de l'architecture définie par le présent Modèle de référence, la transparence à la persistance exige une transparence à la relocalisation pour tous les canaux liés à la grappe.

16.6 Transparence à la relocalisation

La transparence à la relocalisation masque la relocalisation d'une interface à d'autres interfaces qui sont liées avec elle.

La transparence à la relocalisation exige:

- qu'un relocalisateur soit associé à chaque interface appartenant à une grappe;
- la propagation vers les relocalisateurs des informations sur les changements de position des objets (par exemple, par les gestionnaires de grappe);
- que les objets lieurs échangent des données supplémentaires en interagissant à travers un canal pour confirmer la validité de la liaison supportée par le canal. Ces données sont dérivées de la position dans le temps et dans l'espace associée aux références d'interface d'ingénierie des interfaces liées au canal.

Si un objet lieur détecte qu'un canal a été invalidé par la relocalisation d'un objet (par exemple, en observant une défaillance de communication), l'objet lieur doit valider les références d'interface d'ingénierie des interfaces auxquelles le canal était lié, et si nécessaire, rétablir le canal. La validation est réalisée par la fonction de relocalisation.

NOTE – Si le canal est invalidé par la désactivation d'un objet ou la défaillance d'un objet pour lequel un point de reprise a été précédemment posé, le relocalisateur incorporera une procédure pour la restauration de la grappe contenant l'objet dans le contexte de la validation.

16.7 Transparence à la duplication

La transparence à la duplication masque l'utilisation d'un groupe d'objets dont les comportements sont compatibles pour prendre en charge une interface.

16.7.1 Concepts

16.7.1.1 Schéma de duplication: spécification de contraintes sur la duplication d'un objet incluant des contraintes sur la disponibilité et les performances de l'objet.

16.7.2 Règles

La transparence à la duplication est obtenue en utilisant la fonction de duplication pour coordonner la duplication d'un objet afin de satisfaire à un schéma de duplication. Le raffinement de la spécification de traitement associée à la transparence à la duplication inclut les étapes suivantes:

- la définition pour l'objet de traitement d'une interface de gestion d'objet supportant la pose de point de reprise et la suppression d'objet;
- l'introduction d'une fonction de duplication;
- la mise en place d'une politique de duplication pour les grappes contenant l'objet;
- l'association d'un relocalisateur avec chacune des interfaces de l'objet.

Dans le cadre de l'architecture définie par le présent Modèle de référence, la transparence à la duplication exige la transparence à la relocalisation pour la grappe.

16.8 Transparence aux transactions

La transparence aux transactions masque la coordination d'activités au sein d'une configuration d'objets afin d'en assurer la cohérence.

16.8.1 Concept

16.8.1.1 Schéma transactionnel: schéma dynamique et schéma d'invariant définissant des transactions et leurs dépendances.

16.8.2 Règles

La transparence aux transactions est obtenue par l'utilisation de la fonction de transaction afin de coordonner le comportement d'un objet pour satisfaire à un schéma transactionnel. Le raffinement de la spécification de traitement associée à la transparence aux transactions inclut les étapes suivantes:

- la détermination des politiques de fonction de transaction à partir du schéma transactionnel;
- l'ajout d'opérations de pose de point de reprise et de reprise concernant l'état de l'objet;
- le remplacement de liaisons entre objets par la fonction de transaction;
- l'extension des interfaces des objets de traitement.

Les extensions aux interfaces de l'objet incluent la notification de l'occurrence des actions significatives et la reprise de l'état de l'objet après annulation.

Annexe A

Règles formelles de sous-typage dans le langage de traitement

(Cette annexe fait partie intégrante de la présente Recommandation | Norme Internationale)

La présente annexe définit des règles formelles de sous-typage pour les signatures d'interface de traitement. Les types de signatures d'interface de traitement peuvent être d'ordre supérieur ainsi que l'implique le 7.2.2.4 sur les règles de paramètre. Cette annexe ne formalise que le sous-ensemble du premier ordre des règles de sous-typage. La formalisation des caractéristiques d'ordre supérieur de types de signature d'interface de traitement est sujette à étude ultérieure.

Cette annexe définit un système de type du premier ordre qui comprend un langage de type simple et des règles d'égalité de type et de sous-typage. Elle décrit également un algorithme de comparaison de types complet et correct destiné au système de type. Les types de signature d'interface signal, de signature d'interface opération et de signature d'interface flux sont définis en utilisant le langage de type. Etant donné que le sous-typage de signature d'interface flux n'est que partiellement défini au 7.2.4.2, la présente annexe ne formalise que les règles de sous-typage qui s'appliquent entre flux du même nom.

A.1 Notations et conventions

Les notations suivantes sont utilisées:

- α, β, γ etc. désignent des types;
- *t, s* etc. désignent des identificateurs de type (c'est-à-dire, des variables de type) et des types de base (c'est-à-dire, des constantes de type); l'ensemble des variables de type (et des constantes de type) est appelé T_{var};
- *a, b, c,* a_1, a_2.,a_n, etc. désignent des identificateurs d'éléments de structure du langage de type; ils sont appelés *étiquettes*; l'ensemble des étiquettes est désigné par Λ;
- $\alpha[\beta/t]$ désigne le remplacement de β par *t* dans α;
- *Nil* désigne une constante de type prédéfinie.

A.2 Système de type

Le système de type comprend des constantes de type, des fonctions, des produits cartésiens, des enregistrements, des unions et des définitions récursives. La grammaire de la Figure A.1 donne le langage de type résultant, *Type*.

$$
\begin{array}{rcl}
a & ::= & t \\
 & | & \bot \\
 & | & \top \\
 & | & \alpha \rightarrow \beta \\
 & | & \alpha_1 \times \ldots \times \alpha_n \\
 & | & \langle a_1 : \alpha_1, \ldots, a_n : \alpha_n \rangle \\
 & | & [c_1 : \gamma_1, \ldots, c_n : \gamma_n] \\
 & | & \mu t.\alpha
\end{array}
$$

Figure A.1 – Syntaxe abstraite des déclarations de type

Les types de base sont appelés $\top$ (haut) et $\bot$ (bas). Ils jouent respectivement le rôle de plus grand et de plus petit élément dans la relation de sous-typage. Les fonctions sont désignées par $\alpha \rightarrow \beta$, les produits cartésiens par $\alpha_1 \times \ldots \times \alpha_n$, les unions par$[c_1 : \gamma_1 \ldots, c_n : \gamma_n]$, les enregistrements par: $\langle a_1 : \alpha_1, \ldots, a_n : \alpha_n \rangle$.

μ est un opérateur de liaison de variable. Les types récursifs peuvent être construits en liant les types à des identificateurs et en référençant un identificateur d'un type dans un autre.

Les parenthèses sont utilisées pour déterminer la précédence entre les différents opérateurs s'il y a lieu. En leur absence, $\rightarrow$ associe vers la droite et le champ de μ s'étend vers la droite aussi loin que possible.

L'ensemble des variables libres qui apparaissent dans α est désigné par $FV(\alpha)$.

A.2.1 Règles de typage

Ce paragraphe donne les règles d'égalité de type et les règles de sous-typage pour un langage donné.

Un type α est *contractant* dans une variable de type t, et est noté $\alpha\downarrow t$, si t n'est pas libre dans α, ou bien si α peut être réécrit comme un type ayant une des formes suivantes:

- $\alpha_1 \rightarrow \alpha_2$;
- $\langle a_1 : \alpha_1, \ldots, a_m : \alpha_m \rangle$;
- $[c_1 : \gamma_1, \ldots, c_n : \gamma_n]$;
- $\alpha_1 \times \ldots \times \alpha_n$.

Les règles d'égalité de type sont données à la Figure A.2. L'égalité de type est désignée par =.

(E.1)	$\alpha = \alpha$
(E.2)	$\alpha = \beta \Rightarrow \beta = \alpha$
(E.3)	$\alpha = \beta, \beta = \gamma \Rightarrow \alpha \Rightarrow \gamma$
(E.4)	$\alpha_1 = \alpha_2, \beta_1 = \beta_2 \Rightarrow \alpha_1 \rightarrow \beta_1 = \alpha_2 \rightarrow \beta_1$
(E.5)	$\alpha = \beta \Rightarrow \mu t.\alpha = \mu t.\beta$
(E.6)	$\forall i \varepsilon \{1, \ldots, n\}\ , \alpha_i = \beta_i \Rightarrow \alpha_1 \times \ldots \times \alpha_n = \beta_1 \times \ldots \times \beta_n$
(E.7)	$\forall i \varepsilon \{1, \ldots, n\}\ , \alpha_i = \beta_i \Rightarrow \langle a_1 : \alpha_1, \ldots, a_n : \alpha_n \rangle = \langle a_1 : \beta_1, \ldots , a_n : \beta_n \rangle$
(E.8)	$\forall i \varepsilon \{1, \ldots, n\}\ , \alpha_i = \beta_i \Rightarrow [a_1 : \alpha_1, \ldots, a_n : \alpha_n] = [a_1 : \beta_1, \ldots , a_n : \beta_n]$
(E.9)	$\mu t.t = \bot$
(E.10)	$\alpha[\mu t.\alpha/t] = \mu t.\alpha$
(E.11)	$\alpha[\beta/t] = \beta_1, \alpha[B_2/t] = \beta_2\ \alpha \downarrow t \Rightarrow \beta_1 = \beta_2$

Figure A.2 – Règles d'égalité de type

Les règles de sous-typage sont définies sous la forme de règles d'inférence portant sur des jugements qui ressemblent à un programme Prolog. Les jugements sont de la forme $\Gamma \vdash \alpha \leq \beta$, où Γ est un ensemble d'hypothèses de sous-typage sur les variables de type de la forme $\{t_1 \leq s_1, \ldots, t_n \leq s_n\}$. Une règle type peut prendre la forme suivante:

$$\Gamma \vdash \alpha_1 \leq \beta_1,\ \Gamma \vdash \alpha_2 \leq \beta_2 \Rightarrow \Gamma \vdash \alpha \leq \beta$$

De façon informelle, cela signifie que pour vérifier si $\Gamma \vdash \alpha \leq \beta$, on doit tout d'abord déterminer si $\Gamma \vdash \alpha_1 \leq \beta_1$ et si $\Gamma \vdash \alpha_2 \leq \beta_2$. Si ces sous-buts sont atteints, on peut alors en conclure que α est un sous-type de β.

Les règles de sous-typage sont données à la Figure A.3. On peut dire que α est un sous-type de β si $\varnothing \vdash \alpha \leq \beta$ peut être dérivé en utilisant les règles de sous-typage et les règles d'égalité.

A.2.2 Définitions relatives aux types

Les élément de *Type* sont définis par des ensembles d'équations mutuellement dépendantes, qui sont modélisées en tant qu'environnements bien-formés. Un environnement consiste en un nombre fini de correspondances entre des variables de type et des types appartenant à T, où T est un sous-ensemble non-récursif de *Type*. Un environnement bien formé Υ est un environnement dans lequel les variables libres d'un type α qui sont associées à une variable t du domaine de Υ appartiennent toutes au domaine de Υ. De façon intuitive, dans un environnement, chaque variable de type représente un type. On peut considérer que dans un environnement donné, les associations entre les variables de type et les éléments de T sont des équations mutuellement dépendantes définissant les types correspondants.

(S.1)	$\alpha = \beta \Rightarrow \Gamma \vdash \alpha \leq \beta$
(S.2)	$\Gamma \vdash \alpha \leq \beta, \Gamma \vdash \beta \leq \gamma \Rightarrow \Gamma \vdash \alpha \leq \gamma$
(S.3)	$t \leq s \in \Gamma \Rightarrow \Gamma \vdash t \leq s$
(S.4)	$\Gamma \vdash \bot \leq \alpha$
(S.5)	$\Gamma \vdash \alpha \leq \top$
(S.6)	$\Gamma \vdash \alpha_2 \leq \alpha_1, \Gamma \vdash \beta_1 \leq \beta_2 \Rightarrow \Gamma \vdash \alpha_1 \rightarrow \beta_1 \leq \alpha_2 \rightarrow \beta_2$
(S.7)	$\forall \iota \in \{1, \ldots, n\}, \Gamma \vdash \alpha_i \leq \beta_i \Rightarrow \Gamma \vdash \langle a_1 : \alpha_1, \ldots, a_m : \alpha_m \rangle \leq \langle a_1 : \beta_1, \ldots, a_n : \beta_n \rangle$ *avec* $n \leq m$
(S.8)	$\forall \iota \in \{1, \ldots, n\}, \Gamma \vdash \alpha_i \leq \beta_i \Rightarrow \Gamma \vdash [a_1 : \alpha_1, \ldots, a_n : \alpha_n] \leq [a_1 : \beta_1, \ldots, a_n : \beta_n]$ *avec* $n \leq m$
(S.9)	$\forall \iota \in \{1, \ldots, n\}, \Gamma \vdash \alpha_i \leq \beta_i \Rightarrow \Gamma \vdash \alpha_1 \times \ldots \times \alpha_n \leq \beta_1 \times \ldots \times \beta_n$
(S.10)	$\Gamma \cup \{t \leq s\} \vdash \alpha \leq \beta \Rightarrow \Gamma \vdash \mu t.\alpha \leq \mu s.\beta$ *avec t seulement en* α*; s seulement en* β*, t, s pas en* Γ

Figure A.3 – Règles de sous-typage

De façon formelle, soit Υ_{wf} un ensemble d'environnements bien formés et soit $f : A \rightarrow_f B$ une fonction partielle de A vers B dans un domaine fini, $FV(\alpha)$ désigne l'ensemble des variables libres apparaissant dans α):

$$\Upsilon_{wf} =_{def} \{\Upsilon : T_{var} \rightarrow_f Type \mid \forall\, t, t' \in dom(\Upsilon), t' \in FV(\Upsilon(t)) \Rightarrow t' \in dom(\Upsilon)\}$$

Soit $\Upsilon = \{t \mapsto \alpha, t1 \mapsto \alpha_1, \ldots, t_q \mapsto \alpha_q\}$. $\Upsilon \setminus t$ désigne l'environnement suivant:

$$\Upsilon \setminus t =_{def} \{t_1 \mapsto \alpha_1, \ldots, t_q \mapsto \alpha_q\}$$

Un type associé à une variable de type t dans un contexte d'environnement bien-formé Υ ($t \in dom(\Upsilon)$) est défini comme étant $Val(t, \Upsilon)$, où *Val* est une fonction définie sur des types et environnements. *Val est* définie sur la Figure A.4. Ainsi, tout élément de *Type* peut être défini comme $Val(t, \Upsilon)$,où Y est un ensemble bien-formé et $t \in dom(\Upsilon)$.

(IT.1)	$Val(\bot, \Upsilon) = \bot$
(IT.2)	$Val(\top, \Upsilon) = \top$
(IT.3)	$Val(Nil, \Upsilon) = Nil$
(IT.4)	$Val(\alpha \rightarrow \beta, \Upsilon) = Val(\alpha, \Upsilon) \rightarrow Val(\beta, \Upsilon)$
(IT.5)	$Val(\langle a_1 : \alpha_1, \ldots, a_n : \alpha_n \rangle, \Upsilon) = \langle a_1 : Val(\alpha_1, \Upsilon), \ldots, a_n : Val(\alpha_n, \Upsilon) \rangle$
(IT.6)	$Val([a_1 : \alpha_1, \ldots, a_n : \alpha_n], \Upsilon) = [a_1 : Val(\alpha_1, \Upsilon), \ldots, a_n : Val(\alpha_1, \Upsilon)]$
(IT.7)	$Val(\alpha_1 \times \ldots \times \alpha_n, \Upsilon) = Val(\alpha_1, \Upsilon) \times Val(\alpha_n, \Upsilon)$
(IT.8)	*si* $t \not\in dom(\Upsilon)$ *alors* $Val(t, \Upsilon) = t$
(IT.10)	*si* $t \in dom(\Upsilon)$ *alors* $Val(t, \Upsilon) = \mu t.Val(\Upsilon(t), \Upsilon \setminus t)$

Figure A.4 – Sémantique des définitions des types d'interface

A.2.3 Un algorithme de comparaison de types

Ce paragraphe définit un algorithme de comparaison de types correct et complet relativement aux règles d'égalité de type et sous-typage définies plus haut. L'algorithme travaille sur deux environnements bien-formés ε_1 et ε_2 tels que $dom(\varepsilon_1) \cap dom(\varepsilon_2) = \varnothing$ (les types à comparer sont associés à deux variables, l'une dans ε_1, l'autre dans ε_2). Il est décrit comme un ensemble de règles d'inférence impliquant $\varepsilon =_{def} \varepsilon_1 \cup \varepsilon_2$ et un ensemble Σ de la forme $\{t_1 \leq s_1, \ldots, t_n \leq s_n\}$, qui enregistre les relations entre les variables découvertes pendant l'exécution de l'algorithme. Une règle d'inférence correspond à une implication logique de *jugements* de la forme $\Sigma, \varepsilon \vdash \alpha \leq \beta$. De façon intuitive, un jugement capture l'assertion $\alpha \leq \beta$ dérivée dans le contexte de Σ et de ε. Les jugements initiaux $\{t_1 \leq s_1, \ldots, t_n \leq s_n\}$ doivent être tels que $\{t_1, s_1, \ldots, t_n, s_n\} \cap dom(\varepsilon) = \varnothing$.

Les règles d'inférence sont données à la Figure A.5. Sur la Figure A.5 α, $\beta \in Type$, t, s désignent des variables arbitraires, u désigne des variables qui n'appartiennent pas à $dom(\varepsilon)$.

Etant donné un but initial $\Sigma, \varepsilon \vdash \alpha \leq \beta$, l'algorithme consiste à appliquer les règles d'inférence d'avant en arrière, ce qui génère des sous-buts dans le cas où on applique les règles (rec), (fun), (rcd), (pro) et (uni). Un arbre de buts construit de cette manière est appelé arbre d'exécution. Un arbre d'exécution est toujours fini: si $t \leq s$ est une hypothèse qui est ajoutée à Σ, alors, t et s sont des variables de type dans $dom(\varepsilon)$; de plus, les règles (fun), (pro), (uni) and (rcd) réduisent la taille du but courant en le remplaçant par des sous-expressions du but, et chaque application de (rec) fait croître Σ.

(assmp)	$t \leq s \in \Sigma \Rightarrow \Sigma, \varepsilon \vdash t \leq s$
(bot)	$\Sigma, \varepsilon \vdash \bot \leq \beta$
(top)	$\Sigma, \varepsilon \vdash a \leq \top$
(var)	$\Sigma, \varepsilon \vdash u \leq u$
(fun)	$\Sigma, \varepsilon \vdash \alpha_2 \leq \alpha_1, \Sigma, \varepsilon \vdash \beta_1 \leq \beta_2 \Rightarrow \Sigma, \varepsilon \vdash \alpha_1 \to \beta_1 \leq \alpha_2 \to \beta_2 \leq$
(rcd)	$\forall \iota \in \{1, \ldots, n\}, \Sigma, \varepsilon \vdash \alpha_i \leq \beta_i \Rightarrow \Sigma, \varepsilon \vdash \langle a_1 : \alpha_1, \ldots, a_n : \alpha_n \rangle \leq \langle a_1 : \beta_1, \ldots, a_n : \beta_n \rangle$ *avec* $n \leq m$
(uni)	$\forall \iota \in \{1, \ldots, n\}, \Sigma, \varepsilon \vdash \alpha_i \leq \beta_i \Rightarrow \Sigma, \varepsilon \vdash [a_1 : \alpha_1, \ldots, a_n : \alpha_n] \leq \{a_1 : \beta_1, \ldots, a_m : \beta_m]$ *avec* $n \leq m$
(pro)	$\forall \iota \in \{1, \ldots, n\}, \Sigma, \varepsilon \vdash \alpha_i \leq \beta_i \Rightarrow \Sigma, \varepsilon \vdash \alpha_1 \times \ldots \times \alpha_n \leq \beta_1 \times \ldots \times \beta_n$
(rec)	$\Sigma \cup \{t \leq s\}, \varepsilon \vdash \varepsilon(t) \leq \varepsilon(s) \Rightarrow \Sigma, \varepsilon \vdash t \leq s$

Figure A.5 – Règles d'inférence pour les comparaisons de types

Un arbre d'exécution réussit lorsque toutes les feuilles correspondent à une application d'une des règles (assmp), (bot), (top) ou (var). Il échoue lorsqu'une au moins des feuilles correspond à un but non rempli (c'est-à-dire si aucune règle ne peut lui être appliquée). Si l'arbre d'exécution correspondant à l'objectif $\varnothing\varepsilon \vdash \alpha \leq \beta$ réussit, on notera $\vdash_A \alpha \leq \beta$.

Etant donné des types récursifs α et β, tels que $\alpha = Val(t_1, \varepsilon_1)$ et $\beta = Val(t_2, \varepsilon_2)$ (ε_1 et ε_2 comme plus haut) une relation de sous-typage, $\leq_A$, induit une relation définie par l'algorithme avec la définition suivante:

$$\alpha \leq_A \beta \Leftrightarrow \vdash_A t_1 \leq t_2$$

Cette nouvelle relation de sous-typage coïncide avec la relation antérieure, c'est-à-dire l'algorithme est correct et complet relativement aux règles d'égalité de type et de sous-typage de type:

- étant donné α, β dans T, si $\alpha \leq_A \beta$, alors $\alpha \leq_R \beta$;
- étant donné α, β dans T, si $\alpha \leq_R \beta$, alors $\alpha \leq_A \beta$.

A.3 Types de signature d'interface signal

On formalise les types de signature d'interface signal en les interprétant dans le langage *Type*. L'ensemble des types de signature d'interface signal est désigné par $Type_{sig}$. Les éléments de $Type_{sig}$. sont définis de manière abstraite à travers l'utilisation de deux fonctions *intype* : $Type_{sig} \to Type$ and *outype* : $Type_{sig} \to Type$. Dans un type de signature d'interface signal donné, *intype* décrit l'ensemble des signaux envoyés, et *outype* l'ensemble des signaux reçus.

Les éléments du langage *Type* qui sont associés à un type de signature d'interface signal à travers des fonctions i*ntype* et *outype* sont définies par des environnements bien-formés dont le codomaine est le sous-ensemble de *Type* défini par la grammaire de la Figure A.6, où les étiquettes a_i, $i \in \{1, \ldots, q\}$ sont supposées être distinctes. De fait, la Figure A.6 fournit une syntaxe abstraite pour les signatures d'interface signal. Les étiquettes a_i correspondent aux noms des signaux. Les productions *Arg* correspondent aux paramètres des signaux. Les productions *Sigsig* correspondent aux signatures de signaux. La forme fonctionnelle qui a été adoptée pour les signatures de signaux met l'accent sur l'analogie avec les signatures d'annonces.

La relation de sous-typage qui porte sur les types d'interface signal, $\leq_s$, est définie par:

$$\forall \iota_1, \iota_2 \in Type_{sig}, \iota_1 \leq \iota_2 \equiv \iota_1.intype \leq \iota_2.intype \wedge \iota_2.outype \leq \iota_1.outype$$

α	::=	$\langle a_i : Sigsig, \ldots, a_q : Sigsig \rangle$
Sigsig	::=	$Arg \rightarrow Nil$
Arg	::=	$Nil \mid t_1 \times \ldots \times t_p$

Figure A.6 – Syntaxe abstraite pour les types de signature d'interface signal

A.4 Types de signature d'interface opération

On formalise les types de signature d'interface opération en les interprétant dans le langage *Type* (les types de signature d'interface opération client peuvent être dérivés immédiatement par complément). L'ensemble des types de signature d'interface opération serveur est désigné par $Type_o$ (*S*). Les éléments de $Type_o$ (*S*) sont définis de manière abstraite à travers l'utilisation de la fonction *optype* : $Type_o$ (*S*) $\rightarrow$ *Type*.

Les éléments du langage *Type* qui sont associés à un type de signature d'interface opération à travers l'utilisation de la fonction *optype* sont définies par des environnements bien-formés dont le codomaine est le sous-ensemble de *Type* défini par la grammaire de la Figure A.7, où les étiquettes a_i, $i \in \{1, \ldots, q\}$, sont supposées être distinctes, et où les étiquettes c_i, $i \in \{1, \ldots, q\}$ sont supposées être distinctes du contexte d'une production *Opsig*.

α	::=	$\langle a_i : Opsig, \ldots, a_q : Opsig \rangle$
Opsig	::=	$Arg \rightarrow Term \mid Arg \rightarrow Nil$
Term	::=	$[c_1 : Arg, \ldots, c_q : Arg]$
Arg	::=	$Nil \mid t_1 \times \ldots \times t_p$

Figure A.7 – Syntaxe abstraite pour les types de signature d'interface opération

De fait, la Figure A.7 fournit une syntaxe abstraite pour les signatures d'interface opération. Les productions *Opsig* correspondent aux signatures d'opération. De façon spécifique, les productions *Opsig* qui ont la forme $Arg \rightarrow Term$ de la Figure A.1 correspondent aux interrogations. Les productions *Opsig* de la forme $Arg \rightarrow Nil$ correspondent aux annonces. Les produtions *Arg* correspondent aux paramètres d'invocation. Les productions *Term* correspondent aux terminaisons. Quand *Nil* se trouve sur la partie gauche d'une production *Opsig* cela signifie que l'invocation n'a aucun paramètre. Quand *Nil* se trouve à droite d'une production *Opsig* (c'est-à-dire, dans une signature d'annonce) cela signifie qu'aucune terminaison n'est en attente. Les étiquettes a_i correspondent à des noms d'opération. Les étiquettes c_i correspondent à des noms de terminaison.

La relation de sous-typage qui porte sur les types d'interface opération serveur, $\leq_o$, est définie par:

$$\forall\ \iota_1, \iota_2 \in Type_o\ (S),\ \iota_1 \leq \iota_2 \equiv \iota_1.optype \leq \iota_2.optype$$

A.5 Types de signature d'interface flux

La définition de règles de sous-typage de signature complètes pour les interfaces flot va au delà du domaine d'application du présent Modèle de référence (voir 7.2.4.2). Il est cependant à noter qu'on peut formaliser un type de signature de flux en l'interprétant dans le langage *Type*. Les éléments du langage *Type* qui sont associés à une signature de flux peuvent être définis par des environnements bien-formés dont le codomaine est le sous-ensemble de *Type* défini par la grammaire de la Figure A.8, où l'étiquette a_i correspond au nom du flux.

Les règles de sous-typage de 7.2.4.2 qui sont associées à des flux correspondants (sous réserve qu'ils aient la même causalité) correspondent exactement, dans ce cas, à la relation de sous-typage $\leq$.

α	::=	$\langle a_i : Flowsig \rangle$
Flowsig	::=	$Arg \rightarrow Nil$
Arg	::=	$Nil \mid t$

Figure A.8 – Syntaxe abstraite pour les types de signature d'interface flux

A.6 Exemple

Considérons les définitions de type de signature d'interface opération serveur suivantes (c'est-à-dire, l'environnement bien-formé):

$$\Upsilon = \{t \mapsto \alpha, f_t \mapsto \beta\}$$

où

$$\alpha =_{def} \quad \langle\, op : t \rightarrow [ok : Nil, nok : Nil], factory : Nil \rightarrow [ok : f_t] \,\rangle$$

$$\beta =_{def} \quad \langle\, new : Nil \rightarrow [ok : t] \,\rangle$$

et où $t, f_t \in Tvar$; *op*, *factory*, *new*, *ok* et $nok \in \Lambda$ (*op*, *factory* et *new* sont des noms d'opération; *ok* et *nok* sont des noms de terminaison).

De façon intuitive, l'environnement Υ correspond à la définition de deux types, *t* et f_t. f_t ne comporte qu'une seule opération, en l'occurence *new*, qui ne contient pas d'argument et retourne une référence à l'instance de type *t*. On peut imaginer, par exemple, que f_t est le type d'un objet usine qui crée des objets ayant une interface de type *t* à la demande, c'est-à-dire, à chaque invocation de l'opération *new*. *t* comporte deux opérations: *op* et *factory*. L'opération op passe comme argument une référence à une instance de type *t*: c'est un premier exemple de définition récursive. L'opération *factory* ne passe pas d'argument et retourne une référence à une instance de type f_t . On peut imaginer, par exemple pour des besoins de gestion, que chaque objet ayant une interface de type *t* peut suite à une requête (c'est-à-dire, suite à l'invocation de l'opération *factory*), retourner une référence à l'objet usine qui l'a créé. C'est un deuxième exemple de définition récursive, puisque la définition de f_t fait référence à *t*.

Si on applique la définition de *Val* qui est donnée plus haut, en notant $\Upsilon_1 =_{def} \{f_t \mapsto \beta\}$, en surchargeant le signe = et en utilisant la règle d'équivalence E.10, on obtient les équations suivantes:

$$\begin{aligned}
Val\,(t, \Upsilon) &= \mu\, t.\, Val\,(\alpha, \{\, f_t \mapsto \beta\} \\
&= \mu\, t.\, \langle op : Val(t, \Upsilon_1) \rightarrow [ok: Nil, nok : Nil] \\
&\quad factory: Nil \rightarrow [ok : Val\,(f_t, \Upsilon_1)]\rangle \\
&= \mu\, t.\, \langle op : t \rightarrow [ok: Nil, nok : Nil] \\
&\quad factory: Nil \rightarrow [ok : \mu\, f_t.Val\,(\beta, \varnothing)]\rangle \\
&= \mu\, t.\, \langle op : t \rightarrow [ok: Nil, nok : Nil] \\
&\quad factory: Nil \rightarrow [ok : \mu \langle\, f_t \langle new : Nil \rightarrow [ok : t]\rangle]\rangle \\
&= \mu\, t.\, \langle op : t \rightarrow [ok: Nil, nok : Nil] \\
&\quad factory: Nil \rightarrow [ok : \langle new : Nil \rightarrow [ok : t]\rangle]\rangle
\end{aligned}$$