UNION INTERNATIONALE DES TÉLÉCOMMUNICATIONS

UIT-T Q.775

SECTEUR DE LA NORMALISATION
DES TÉLÉCOMMUNICATIONS
DE L'UIT

(03/93)

SPÉCIFICATIONS DU SYSTÈME DE SIGNALISATION N° 7

SYSTÈME DE SIGNALISATION N° 7 – GUIDE D'UTILISATION DU GESTIONNAIRE DE TRANSACTIONS

Recommandation UIT-T Q.775

(Antérieurement «Recommandation du CCITT»)

AVANT-PROPOS

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

La Conférence mondiale de normalisation des télécommunications (CMNT), qui se réunit tous les quatre ans, détermine les thèmes que les Commissions d'études de l'UIT-T doivent examiner et à propos desquels elles doivent émettre des Recommandations.

La Recommandation révisée UIT-T Q.775, élaborée par la Commission d'études XI (1988-1993) de l'UIT-T, a été approuvée par la CMNT (Helsinki, 1-12 mars 1993).

NOTES

1 Suite au processus de réforme entrepris au sein de l'Union internationale des télécommunications (UIT), le CCITT n'existe plus depuis le 28 février 1993. Il est remplacé par le Secteur de la normalisation des télécommunications de l'UIT (UIT-T) créé le 1er mars 1993. De même, le CCIR et l'IFRB ont été remplacés par le Secteur des radiocommunications.

Afin de ne pas retarder la publication de la présente Recommandation, aucun changement n'a été apporté aux mentions contenant les sigles CCITT, CCIR et IFRB ou aux entités qui leur sont associées, comme «Assemblée plénière», «Secrétariat», etc. Les futures éditions de la présente Recommandation adopteront la terminologie appropriée reflétant la nouvelle structure de l'UIT.

2 Dans la présente Recommandation, le terme «Administration» désigne indifféremment une administration de télécommunication ou une exploitation reconnue.

TABLE DES MATIÈRES

Page

1 Introduction 1

1.1 Considérations générales 1

1.2 Environnement 1

2 Opérations 2

2.1 Définition 2

2.2 Exemples 2

2.3 Services, relatifs aux composants, offerts aux utilisateurs du TC 4

2.4 Situations anormales relatives aux composants 8

3 Dialogues 12

3.1 Groupement de composants dans un message 13

3.2 Services de gestion du dialogue 14

3.3 Service amélioré de gestion du dialogue 21

4 Directives pour l'élaboration des protocoles utilisateurs de TC 24

4.1 Introduction 24

4.2 Découpe fonctionnelle dans une application 24

4.3 Comment spécifier une AE entité d'application et un contexte d'application 24

4.4 Comment spécifier un ASE 25

4.5 Comment spécifier les opérations et les erreurs 25

4.6 Spécifications de types de données 32

4.7 Comment spécifier les syntaxes abstraites 33

4.8 Règles de codage 34

Recommandation Q.775

SYSTÈME DE SIGNALISATION N° 7 – GUIDE D'UTILISATION DU GESTIONNAIRE DE TRANSACTIONS

(Melbourne, 1988; modifiée à Helsinki, 1993)

1 Introduction

1.1 Considérations générales

La présente Recommandation a pour gestionnaire de servir de guide aux utilisateurs potentiels du gestionnaire de transactions (utilisateurs du TC). Les exemples donnés sont uniquement illustratifs; ils indiquent comment une application peut utiliser le sous-système application pour la gestion des transactions (TCAP) (*transaction capability application part*) et non comment le TCAP doit être utilisé dans tous les cas. Les bases techniques de cette Recommandation sont les Recommandations Q.771 à Q.774; elles seront considérées comme référence en cas de désaccord avec la présente Recommandation.

Le principal objectif du sous-système TCAP est de fournir un support aux applications interactives dans un environnement distribué. Le TCAP est basé sur la notion des opérations distantes définie dans les Recommandations X.219 et X.229 (ROSE) ainsi que sur certaines améliorations et additions spécifiques à l'environnement du système de signalisation n° 7 de manière à fournir les services attendus par les utilisateurs du TC. Les interactions entre entités d'application distribuées sont modélisées par des opérations. Une opération est lancée par une entité (origine), l'autre entité (destination) tente d'exécuter l'opération et éventuellement renvoie le résultat de cette tentative.

Le TCAP n'a pas pour objet de définir la sémantique d'une opération (représentée par ses nom et paramètres). Les services offerts par le TCAP sont indépendants d'une opération particulière. Un utilisateur du TC qui définit une application doit:

1) sélectionner les opérations (ce qui comprend la définition de la sémantique et de la syntaxe des données échangées pendant les lancements des opérations et leurs réponses);
2) sélectionner les services du TCAP nécessaires à la mise en œuvre de ces opérations. Ces services comprennent la gestion d'opérations individuelles ou d'opérations apparentées, attachées à une association entre utilisateurs du TC appelée dialogue;
3) définir le scénario d'application (par exemple lequel des deux homologues lance les opérations, l'ordre de l'échange des messages qui constitue le dialogue entre les utilisateurs du TC homologues et leurs réactions face à des situations anormales).

La présente Recommandation décrit le processus de sélection des opérations. Les opérations présentées dans la présente Recommandation sont fictives, elles sont uniquement utilisées à des fins d'illustration. La présente Recommandation décrit également les services offerts par le TCAP pour la gestion d'une opération, ou d'une suite d'opérations dans un dialogue. Par contre, la définition de suites d'opérations particulières fait partie intégrante de la définition du protocole d'application et n'est pas de la compétence de la présente Recommandation. Cependant, l'article 4 donne de brèves indications sur les informations qu'une spécification d'application devrait contenir.

Les services du TCAP sont accessibles aux utilisateurs du TC par l'intermédiaire de primitives. Ces primitives modélisent l'interface entre le TCAP et ses utilisateurs mais n'imposent aucune contrainte de réalisation à cette interface.

1.2 Environnement

Le TCAP définit le protocole de bout en bout, entre utilisateurs du TC localisés dans un réseau sémaphore n° 7. Actuellement, aucune interface standard n'est définie pour l'utilisation du TCAP sur un autre protocole ou réseau de base (par exemple X.25) que le SCCP du système de signalisation n° 7.

Le TC prend en considération les utilisateurs sensibles à l'aspect temps réel qui n'ont pas d'importants volumes de données à échanger. Il est considéré que, pour ces utilisateurs, les protocoles standards définis pour les couches 4 à 6 de l'OSI dans les Recommandations de la série X, pourraient les pénaliser de façon excessive et, par conséquent, ne sont pas utilisés.

En conclusion, le TCAP ne peut subvenir aux besoins de tous les types d'applications, certaines nécessitent des services plus élaborés tels que ceux spécifiés dans les Recommandations de la série X. En outre, en vue d'aider le concepteur d'une application à choisir les moyens de la mettre en œuvre, cette Recommandation, indiquant ce que le TCAP peut faire, indique ce qu'il ne peut faire.

2 Opérations

2.1 Définition

Une opération est lancée par un utilisateur du TC origine pour demander à un utilisateur du TC destination d'exécuter une action donnée.

Chaque opération relève d'une classe donnée. Elle indique si la destination doit signaler seulement les succès, seulement les échecs, les succès et les échecs ou ni les succès, ni les échecs.

La classe d'une opération n'est pas signalée à l'utilisateur du TC distant au moment où l'opération est lancée; en principe, les applications aux deux extrémités savent quelle classe est utilisée pour chaque opération.

Tout comme la classe, la définition de l'opération comprend une valeur de temporisation indiquant le délai maximal dans lequel l'opération doit être terminée et le résultat signalé.

La valeur de cette temporisation est une question locale; elle n'est pas transmise à l'extrémité éloignée par l'intermédiaire d'un protocole quelconque. Elle est choisie par l'utilisateur du TC lors de la définition de l'opération sur la base des prévisions du délai aller-retour d'un utilisateur du TC à un autre et des délais de traitement.

Une opération est définie par:

- son code d'opération et le type des paramètres associés à la demande d'opération;
- sa classe;
- si la classe requiert la signalisation des succès, les résultats possibles, correspondant aux exécutions réussies, sont définis par une liste de paramètres;
- si la classe requiert la signalisation des échecs, les résultats possibles, correspondant aux situations où l'opération n'a pu être complètement exécutée par l'utilisateur du TC distant, sont identifiés par des causes d'erreur spécifiques. La liste de ces causes d'erreur fait partie de la définition de l'opération. Une information de diagnostic peut être ajoutée à la cause d'erreur: si elle est présente, elle fait également partie de la définition;
- la liste des opérations corrélées éventuelles, si les réponses aux opérations corrélées sont autorisées pour cette opération. Les opérations corrélées doivent être décrites séparément;
- une valeur de temporisation indiquant le temps après lequel l'opération doit être terminée et éventuellement renvoyée. Cette valeur de temporisation peut être l'un des facteurs utilisés par une mise en œuvre pour gérer l'identificateur de lancement associé au lancement de l'opération. (Quand la temporisation associée au lancement d'une opération chute, l'identificateur de lancement est restitué à l'ensemble des identificateurs de lancement après une période de «gel» appropriée qui varie selon le mode de réalisation).

En règle générale, le choix de la classe d'une opération devrait reposer sur la sémantique associée à une opération et les opérations ne devraient pas être conçues de manière à être acheminées dans un message donné. Si par exemple, celui qui lance l'opération n'a pas besoin d'un accusé de réception lui permettant de savoir si l'opération peut ou non être effectivement menée à bien, une opération de classe 4 sera sans doute indiquée. S'il n'a pas besoin de savoir explicitement qu'une opération a été réalisée avec succès mais souhaite savoir si elle n'a pas pu être accomplie, une opération de classe 2 est justifiée. Définir par exemple une opération comme «annonce de lecture» comme une opération de classe 2 ou 4 suggère des intentions diverses de celui qui lance l'opération même si les actions de celui qui réalise les opérations peuvent être identiques.

2.2 Exemples

2.2.1 Traitement des opérations simples

NOTE – Le lancement de l'opération doit tenir dans un message, de même que la notification de succès. Les notifications de succès peuvent être fractionnées au moyen de résultat(s) partiel(s) et résultat complet.

Classe 1 (les succès et les échecs sont signalés)

Traduire un numéro libre appel en un numéro d'abonné demandé; renvoyer le numéro demandé si la traduction peut être réalisée, autrement indiquer pourquoi elle ne peut l'être; temps attribué: 2 secondes.

Lorsque aucune réponse n'est renvoyée à la suite d'un lancement d'opération après la chute de la temporisation, l'utilisateur du TC en est informé (annulation de l'opération par le TCAP); il peut penser que le lancement ou la réponse s'est perdu et, selon les applications, prendre les mesures correctives voulues (par exemple nouveau lancement de l'opération, information de la gestion locale, etc.).

Classe 2 (seuls les échecs sont signalés)

Réaliser un test de routine et envoyer une réponse uniquement en cas d'échec du test; temps attribué: 1 minute.

Dans une opération de classe 2, lorsque aucun résultat n'a été reçu, l'utilisateur du TC est informé par la chute de la temporisation. Elle est interprétée comme une exécution réussie, même en cas de perte du lancement d'opération.

Cet aspect doit être pris en compte lors du choix de la classe 2.

Classe 3 (seuls les succès sont signalés)

Réaliser un essai: cela correspond à un point de vue pessimiste, où un échec est considéré comme une option par défaut, ne nécessitant aucune réponse.

La chute de temporisation est indiquée à l'utilisateur du TC: elle doit être interprétée comme un échec de l'opération (mais elle est normale pour le gestionnaire de transactions lui-même qui considère que l'opération est terminée). Cet aspect doit être pris en compte lors du choix de la classe 3.

Classe 4 (ni les succès, ni les échecs ne sont signalés)

Envoyer un avis, sans attendre de réponse ou d'acquittement d'aucune sorte.

Dans ce cas, aucun résultat n'est reçu suite au lancement d'opération. L'utilisateur du TC s'en remet au TCAP et au réseau pour délivrer le lancement. La notification de la chute de temporisation est un problème local.

Comparaison avec les classes d'opérations de ROSE (Recommandation X.219)

ROSE offre cinq classes d'opérations: les classes 2 à 5, appelées classes asynchrones, sont identiques aux classes 1 à 4 du TCAP. La classe 1 de ROSE est une classe synchrone, autorisant les échanges bidirectionnels de composants. Elle n'a pas d'équivalent dans le TCAP.

Cependant, un utilisateur du TC peut décider de fonctionner en mode synchrone (voir 3.2.1).

2.2.2 Traitement d'opérations plus compliquées

Opérations avec résultats segmentés

Un résultat de succès peut être divisé en plusieurs segments, chacun d'eux est passé à l'origine de l'opération par une primitive. Ce service, utilisant la primitive TC-RESULT-NL, peut être utilisé par les utilisateurs du TC pour pallier l'absence de segmentation des couches sous-jacentes. Le dernier segment est passé à l'utilisateur par la primitive TC-RESULT-L.

La signalisation d'une erreur ne peut être segmentée.

Quand le concepteur du protocole peut s'assurer que la segmentation est prévue dans les couches sous-jacentes sur les voies de signalisation sur lesquelles les messages du TC sont transférés, l'utilisation de ce service de segmentation est déconseillée.

Le gestionnaire de transactions ne peut identifier un segment particulier dans le cas d'un résultat segmenté.

L'utilisateur du TC doit veiller à ce que chaque segment puisse être analysé (le paramètre Parameter de chaque primitive de demande TC-RESULT-NL/L devrait contenir suffisamment d'informations pour permettre la construction d'une valeur valable du type (ou du sous-type compatible) associé au résultat de l'opération.

Exemple E1: Une opération demande l'exécution d'un essai. Le résultat d'une exécution correcte est segmenté en trois parties P1, P2 et P3 à renvoyer à l'origine.

Une séquence possible de primitives pour l'exemple E1 est donnée au Tableau 1.

Opérations corrélées

Une autre extension des procédures de base est la possibilité de corréler un lancement d'opération à un autre lancement d'opération.

Typiquement, ce service couvre les situations où la destination de l'opération initiale (ou corrélée à l'opération initiale) requiert des informations supplémentaires afin de traiter cette opération: c'est le cas lorsque des services en mode menu sont utilisés (les services en mode menu permettent à un utilisateur de faire une suite de choix, chacun étant dépendant du précédent).

TABLEAU 1/Q.775

Utilisateur du TC A	Utilisateur du TC B
Demande TC-INVOKE (Essai, Classe = 1)	Indication TC-INVOKE (Essai) Demande TC-RESULT-NL (P1)
Indication TC-RESULT-NL (P1)	Demande TC-RESULT-NL (P2)
Indication TC-RESULT-NL (P2)	Demande TC-RESULT-L (P3)
Indication TC-RESULT-L (P3)	
Temps	
NOTE – La valeur de temporisation est spécifiée par l'utilisateur du TC d'origine au moment du lancement. Un résultat qui n'est pas final ne la relance pas.	

Exemple E2: L'opération est l'exécution d'un essai à plusieurs options. Avant d'exécuter l'essai, ces options sont présentées, pour sélection, à l'initiateur de l'essai (utilisateur du TC A). Les deux opérations sont imbriquées: l'opération 1 est l'essai lui-même; l'opération 2 est la sélection de l'option. L'utilisateur du TC A répond d'abord à l'opération 2 pour que l'utilisateur du TC B puisse exécuter l'essai avec la ou les options indiquées.

Une séquence possible de primitives pour l'exemple E2 est donnée au Tableau 2.

Il n'y a pas de limite au nombre de lancements d'opération qui peuvent être corrélés à un lancement d'opération donné.

Il est à noter que lorsqu'une opération B est corrélée à une autre opération A, elles n'ont pas à être imbriquées. La seule condition est que le lancement de B intervienne avant que le résultat de A ne soit reçu; cependant, l'opération B n'a pas à être terminée avant l'opération A.

2.3 Services, relatifs aux composants, offerts aux utilisateurs du TC

2.3.1 Lancement d'opération

Jusqu'ici, les opérations ont été considérées d'un point de vue statique. Le lancement introduit un aspect dynamique: un lancement d'opération particulier doit être différencié des autres lancements éventuels en cours pour la même opération ou pour une autre.

Chaque activation particulière d'une opération est identifiée par un identificateur de lancement. Cet identificateur doit être sans ambiguïté. Il est choisi par l'utilisateur du TC qui initie le lancement d'opération et est passé à l'utilisateur du TC distant, lequel le réutilisera dans sa réponse (ou dans chaque segment de sa réponse) ou dans un lancement corrélé, donc il établit une corrélation entre la réponse à un lancement (ou chaque segment d'une réponse) ou un lancement corrélé et le lancement lui-même.

L'utilisateur du TC est libre d'attribuer n'importe quelle valeur à l'identificateur de lancement (index, adresse, etc.) à condition que cette valeur corresponde à un nombre entier qui peut être codé dans un octet conformément aux règles de codage spécifiées dans la Recommandation Q.773. Il convient de remarquer que la valeur de ce nombre entier est comprise entre –128 et 127.

L'identificateur de lancement associé à un lancement est à nouveau utilisable lorsqu'un résultat complet ou partiel est reçu, ou lorsque certaines situations anormales sont signalées par le TCAP. Cependant, la valeur ne doit pas être immédiatement réattribuée pour une autre activation d'opération, cela pourrait nuire à la gestion correcte de certaines situations anormales (voir ci-dessous).

La période durant laquelle un identificateur de lancement est libre, mais inutilisable est dite gelée. Cette période varie selon le mode de réalisation.

TABLEAU 2/Q.775

Utilisateur du TC A	Utilisateur du TC B	
Demande TC-INVOKE (Essai, Classe = 1)		Début de l'opération 1
	Indication TC-INVOKE (Essai)	
	Demande TC-INVOKE (Sélection d'option Classe = 1)	Début de l'opération 2
Indication TC-INVOKE (Sélection d'options) Demande TC-RESULT-L (Options)		
	Indication TC-RESULT-L (Options) Demande TC-RESULT-L (Résultat de l'essai)	Fin de l'opération 2
Indication TC-RESULT-L (Résultat de l'essai)		
		Fin de l'opération 1
Temps		

Comme les identificateurs de lancement reçoivent leur valeur dynamiquement lors du lancement de l'opération, cette valeur ne peut apparaître dans la spécification des protocoles d'application. Elle est plutôt une valeur «logique», à laquelle une valeur réelle est substituée au moment de l'exécution afin d'identifier une opération dans un flot simple.

En tenant compte des identificateurs de lancement, la séquence de primitives de l'exemple E2 devient comme indiqué au Tableau 3:

TABLEAU 3/Q.775

Utilisateur du TC A	Utilisateur du TC B
Demande TC-INVOKE (1, Essai, Classe = 1)	
	Indication TC-INVOKE (1, Essai) Demande TC-INVOKE (2, 1, Sélection d'options, Classe = 1)
Indication TC-INVOKE (2, 1, Sélection d'options) Demande TC-RESULT-L (2, Options)	
	Indication TC-RESULT-L (2, Options) Demande TC-RESULT-L (1, Résultat de l'essai)
Indication TC-RESULT-L (1, Résultat de l'essai)	
Temps	

où le premier paramètre d'une primitive indique l'identificateur de lancement. Lorsque deux paramètres sont présents, le second est l'identificateur de corrélation. Ceci est une pure convention de notation.

2.3.2 Annulation (par l'utilisateur du TC)

L'utilisateur du TC demandant le lancement d'une opération peut arrêter, lorsqu'il le juge nécessaire, l'activité associée à l'identificateur de lancement correspondant. Cependant, l'annulation devrait, en principe, être réservée aux situations anormales: la méthode normale pour terminer une opération est la réception d'un résultat ou la chute de la temporisation.

L'annulation a un effet uniquement local: elle n'interdit pas à un utilisateur du TC distant d'envoyer des réponses à une opération annulée. Le TCAP rejettera ces réponses, lors de leur réception. Cela est décrit ci-dessous dans une séquence de primitives de l'exemple E1, où l'utilisateur du TC A annule l'essai après réception du premier résultat partiel.

Dans le Tablelau 4, le segment P2 de la réponse n'est pas reçu par l'utilisateur du TC A: le TCAP détecte une situation de rejet (identificateur de lancement inactif) et ne le remet donc pas à l'utilisateur du TC A. Par ailleurs, toute tentative de l'utilisateur du TC B d'envoyer davantage de segments de la réponse est rejetée en A.

TABLEAU 4/Q.775

Utilisateur du TC A	Utilisateur du TC B
Demande TC-INVOKE (1, Essai, Classe = 1)	Indication TC-INVOKE (1, Essai) Demande TC-RESULT-NL (1, P1)
Indication LTC-RESULT-NL (1, P1) Décision d'annulation: demande TC-CANCEL (1) Indication TC-L-REJECT (1, Code de problème) ...	Demande TC-RESULT-NL (1, P2) ...
Temps	

2.3.3 Rejet (par l'utilisateur du TC)

Il appartient à l'utilisateur du TC de décider du moment où il envoie un composant de rejet ou retourne une indication d'erreur (incapacité d'exécuter une opération). L'utilisateur du TC peut rejeter un composant lorsqu'il le juge nécessaire, à condition toutefois qu'un code de rejet approprié soit défini dans les spécifications du TCAP (par exemple élément d'information obligatoire manquant dans une opération, erreur de réponse, opération inattendue, opération inconnue, etc.) et qu'il puisse l'utiliser.

De même l'utilisateur du TC choisit le code d'erreur et l'information de diagnostic (qui est spécifiée dans le cadre des spécifications de protocole utilisateur du TC et acceptée par les deux utilisateurs du TC homologues uniquement à cette fin) qu'il utilise lorsqu'il envoie un composant d'erreur.

Le rôle de l'utilisateur du TC est illustré dans l'exemple ci-après.

L'utilisateur du TC en A s'attend, dans une situation donnée, à recevoir l'opération Y uniquement comme une opération corrélée. Lorsqu'il reçoit de B un composant de lancement avec un code d'opération se rapportant à Y mais pas d'identificateur corrélé, l'utilisateur du TC en A peut choisir:

- de ne pas exécuter l'opération et de renvoyer une erreur avec un paramètre de diagnostic déterminé au préalable dans les spécifications d'application de l'utilisateur du TC uniquement à cette fin;
- de rejeter le composant comme une «opération non reconnue».

L'interprétation du diagnostic d'erreur ou du code du problème de rejet renvoyé relève de l'utilisateur du TC en B et n'est pas décrite dans les Recommandations relatives au TCAP.

Le rejet d'un lancement d'opération, ou d'un résultat, affecte l'opération dans sa totalité: aucune réponse ultérieure ne sera acceptée par le TC pour ce lancement. Le rejet d'une opération corrélée n'affecte pas l'opération à laquelle elle est corrélée en ce qui concerne le TC. Les utilisateurs de TC devraient décrire leurs réactions face à des situations anormales de ce type dans leur script d'application.

Cela est illustré par le Tableau 5 où, dans l'exemple E2, l'utilisateur du TC A ne prévoit pas le traitement de sélection d'options (qui est peut-être une caractéristique optionnelle), et rejette l'opération avec le code de problème «opération corrélée inattendue». L'utilisateur du TC B peut alors décider d'exécuter l'essai en supposant une option par défaut.

TABLEAU 5/Q.775

Utilisateur du TC A	Utilisateur du TC B
Demande TC-INVOKE (1, Essai, Classe = 1)	Indication TC-INVOKE (1, Essai) Demande TC-INVOKE (2, 1, Sélection d'options, Classe = 1)
Indication TC-INVOKE (2, 1, Sélection d'options) Demande TC-U-REJECT (2, Code de problème)	
	Indication TC-U-REJECT (2, Code de problème) Demande TC-RESULT-L (1, Résultat de l'essai)
Indication TC-RESULT-L (1, Résultat de l'essai)	
Temps	

Lorsqu'un lancement d'opération est rejeté, l'utilisateur du TC peut décider d'un nouveau lancement avec un nouvel identificateur de lancement (par exemple, parce que le composant de lancement a été altéré). Il peut également décider d'abandonner le dialogue. Un dialogue très simple (question-réponse) peut ne pas définir de mécanismes de rétablissement, sauf si l'opération est d'une importance critique (par exemple, mise à jour de base de données).

2.3.4 Annulation à distance (par l'utilisateur du TC)

Le TC n'assure aucun service spécifique pour l'annulation à distance de l'exécution d'une opération en cours. Le service d'annulation prévu par la primitive de demande TC-U-CANCEL n'a qu'un effet local (voir 2.3.2).

Une procédure d'annulation à distance peut toutefois être définie au niveau de l'utilisateur du TC avec les services du TC existants. Une solution pour l'utilisateur du TC consiste à inclure dans l'un des éléments ASE utilisés par le contexte d'application une opération dont le but est d'annuler les lancements existants.

Le module de la notation ASN.1 ci-après décrit ce type d'opération. Ce type (et celui d'erreur corrélé) peut être importé dans l'un des modules utilisés par l'utilisateur du TC de sorte qu'une opération et des codes d'erreur adaptés peuvent être attribués. L'utilisateur a aussi la possibilité de concevoir sa propre opération d'après les mêmes principes.

```
TCAP-Tools { ccitt recommendation q 775 modules(2) tools(1) version1(1) }

DEFINITIONS ::=
BEGIN
EXPORTS Cancel, CancelFailed, Cancelled;

IMPORTS OPERATION, ERROR, InvokeIdType
FROM TCAPMessages { ccitt recommendation q 773 modules(2) messages(1) version2(2) };

Cancel ::= OPERATION
ARGUMENT InvokeIdType
-- un utilisateur TC peut redéfinir ce type pour inclure un résultat vide afin d'obtenir une opération
-- de classe 1
ERRORS { CancelFailed }
-- temporisateur = 15 s
```

```
CancelFailed ::= ERROR
PARAMETER SET {
  problem  [0] CancelProblem,
  invokeId  [1] InvokeIdType }

CancelProblem ::= ENUMERATED
{ unknownInvocation(0),
  tooLate (1),
  notCancellable (2) }
-- un utilisateur TC peut redéfinir ce type pour inclure des problèmes spécifiques à l'application

Cancelled ::= ERROR
-- une erreur de ce type doit être incluse dans la liste d'erreurs des opérations annulables

END
```

Il faut inclure une erreur «annulé» dans la liste des erreurs relatives aux opérations qui peuvent être annulées. Dans ce cas, l'utilisateur du TC qui reçoit la demande d'annulation émet une primitive de demande TC-U-ERROR pour mettre fin à l'opération. A réception d'un composant d'erreur comprenant ce code d'erreur et de l'indication TC-U-ERROR correspondante, il est mis fin à l'opération du côté du lancement.

L'opération qu'il convient d'annuler est identifiée par l'identificateur de lancement qui a été attribué au moment du lancement. Le lancement de l'opération d'annulation n'affecte pas le système d'état de lancement de l'opération qui doit être annulée car l'identificateur de lancement contenu dans l'argument de l'opération n'est pas visible par la sous-couche du composant.

Si l'annulation échoue, une erreur d'utilisateur est signalée avec trois diagnostics possibles:

- lancement inconnu: le lancement n'a jamais été fait ou a été oublié;
- trop tard: le lancement est toujours connu mais l'exécution en est à un stade qui ne permet pas une annulation;
- annulation impossible: l'identificateur de lancement dans l'argument de l'opération d'annulation correspond à une opération que les utilisateurs de TC jugent impossible à annuler par l'initiateur.

Lorsque l'annulation réussit et qu'aucune erreur de retour avec un code d'erreur indiquant «annulé» n'est reçue pour l'opération annulée, la temporisation associée à l'opération annulée chute, d'où l'émission par le TC d'une primitive d'indication TC-L-CANCEL.

L'utilisateur du TC qui demande l'annulation peut aussi décider d'émettre une demande TC-U-CANCEL immédiatement après avoir transmis la demande d'annulation de sorte qu'il n'existe pas d'autres activités locales pour l'opération qui doit être annulée.

L'utilisation du mécanisme d'annulation a un sens si l'opération d'annulation est lancée avant la chute de la temporisation associée à l'opération qu'il convient d'annuler. Après, l'identificateur de lancement risque de ne pas être reconnu du côté de l'exécution car l'opération peut être déjà exécutée et la primitive de réponse transmise. La possibilité d'annuler l'exécution d'une opération après la chute de la temporisation ne relève pas des présentes directives car cela signifierait que l'objectif n'est pas d'annuler l'exécution de l'opération mais les actions ultérieures qui peuvent avoir été provoquées du côté de l'exécution par le lancement.

2.4 Situations anormales relatives aux composants

2.4.1 Perte de composants

Le TCAP suppose un taux de perte de messages dans le réseau très faible. Si ce taux est jugé trop élevé pour une application, celle-ci doit s'orienter vers un service réseau en mode connexion. Si certaines informations de protocole requièrent une qualité de service améliorée (par exemple, informations de taxation), l'application doit introduire ses propres mécanismes afin d'obtenir une meilleure fiabilité pour ces informations.

Perte d'un lancement d'opération

Le Tableau 6 présente le cas de l'exemple E1, où aucune réponse à l'essai n'est reçue avant la chute de la temporisation.

TABLEAU 6/Q.775

Utilisateur du TC A	Utilisateur du TC B
Demande TC-INVOKE (1, Essai, Classe = 1)	
Chute de la temporisation: indication TC-L-CANCEL (1)	
Temps	

Lorsqu'une opération de classe 1 est perdue, l'utilisateur du TC est informé par la chute de la temporisation associée à l'opération. Lorsqu'une opération de classe 1 avec un résultat (dans un segment unique) est perdue, le TCAP ne peut indiquer lequel du lancement d'opération ou de la réponse a été perdu. Si l'application a besoin d'identifier lequel de ces deux composants a été perdu, elle doit le faire par un protocole d'application (par exemple, en utilisant une datation ou un acquittement du lancement d'opération avant d'y répondre).

Pour une opération de classe 2, la perte sera considérée comme un succès (que ce soit le lancement ou le résultat d'échec qui ait été perdu). Vu le taux de perte, cela peut être acceptable pour des opérations non critiques (par exemple, des mesures de nature statistique).

Pour une opération de classe 3, la perte sera considérée comme un échec (que ce soit le lancement ou le résultat de succès qui ait été perdu).

Pour une opération de classe 4, la perte n'est pas vue du TCAP.

Perte d'un résultat

- La perte d'un résultat partiel n'est jamais détectée par le TCAP.
- La perte d'un résultat complet sera éventuellement signalée à l'utilisateur du TC sur chute de temporisation, mais ne pourra jamais être interprétée, sans ambiguïté, comme la perte d'une réponse. En effet, lorsque aucun résultat n'est reçu, c'est peut-être parce que le lancement d'opération a été perdu.

Perte d'une opération corrélée

La perte d'une opération corrélée a le même effet que la perte d'une opération non corrélée. Elle n'a aucun effet sur l'opération à laquelle elle est corrélée.

Perte d'un composant de rejet

Ce cas devrait être extrêmement rare, et aucune application ne devrait éprouver le besoin de se prémunir contre de telles situations. Si la perte d'un rejet concerne un lancement d'opération, la chute de temporisation informera l'utilisateur du TC qui a lancé l'opération que le lancement (ou le résultat) a été perdu, lequel réagira en conséquence. Si la perte d'un rejet concerne un résultat, l'origine de ce résultat ne saura pas qu'il est incorrect; il appartiendra à l'origine de l'opération de détecter la perte.

2.4.2 Duplication de composants

Le cas de duplication de messages est très rare dans un réseau sémaphore n° 7 et, par conséquent, aucune application ne devrait avoir besoin de se prémunir contre de telles situations. Cependant, si une application juge les duplications inacceptables, il lui appartiendra de définir ses propres mécanismes de détection ou d'utiliser un service réseau en mode connexion.

Duplication d'un lancement d'opération

Lorsqu'un lancement d'opération est dupliqué (par le fournisseur de service), l'utilisateur du TC de destination (B) peut, ou non, détecter la duplication:

- L'utilisateur du TC B détecte la duplication: la duplication peut être rejetée avec le code de problème «identificateur de lancement dupliqué». Dans ce cas, l'utilisateur du TC distant peut interpréter le rejet comme le rejet de lancement initial.
- L'utilisateur du TC B ne détecte pas la duplication: cela peut se produire lorsque la relation entre A et B est de type maître-esclave. Dans ce cas, B exécute l'opération sans connaissance du contexte.

En supposant ce second cas dans l'exemple E1, une séquence possible pourrait être celle donnée au Tableau 7.

TABLEAU 7/Q.775

Utilisateur du TC A	Utilisateur du TC B
Demande TC-INVOKE (1, Essai, Classe = 1)	Indication TC-INVOKE (1, Essai) Indication TC-INVOKE (1, Essai) Demande TC-RESULT-NL (1, P1) Demande TC-RESULT-NL (1, P1)
Indication TC-RESULT-NL (1, P1) Indication TC-RESULT-NL (1, P1) A détecté une situation anormale et rejette: demande TC-U-REJECT (1, Code de problème) Le TC détecte une situation anormale et rejette P2: indication TC-L-REJECT (1, Code de problème)	Duplication non détectée Demande TC-RESULT-NL (1, P2) Indication TC-U-REJECT (1, Code de problème)
	Indication TC-R-REJECT (1, Code de problème)
Temps	

Dans cette séquence, l'utilisateur du TC B considère qu'il reçoit deux lancements indépendants et répond à chacun d'eux. Le premier résultat P1 est accepté, puis l'utilisateur du TC A détecte que P1 est reçu une seconde fois et le rejette, ce qui termine l'opération. Lorsque le résultat P2 est reçu, il est rejeté (par le TCAP), par conséquent, les deux activités à l'extrémité B seront terminées lors de la réception des rejets.

Duplication d'un résultat partiel

Si un résultat partiel est dupliqué, le TCAP ne pourra le détecter et le délivrera deux fois à l'utilisateur du TC. La détection de cette situation est du ressort de l'application.

Duplication d'un résultat complet

Si un résultat complet est dupliqué, le TCAP le détecte. Le second résultat complet est considéré comme anormal (l'opération a été terminée avec le premier résultat complet), et le TCAP le rejette.

Le Tableau 8 montre une séquence pour l'exemple E1 où le troisième segment du résultat est dupliqué (par le réseau).

Commentaire: La destruction par le TCAP des composants dupliqués est certainement utile. Il faut en effet noter que:

1) la définition de solutions plus élaborées nécessiterait un degré de complexité du TCAP plus important, en contradiction avec ses caractéristiques de base dans une approche sans connexion;

2) une telle situation est très rare, au moins dans le réseau sémaphore n° 7.

Si une application souhaite se prémunir contre ces situations, il serait préférable qu'elle utilise un service réseau en mode connexion, puisque les duplications seraient alors détectées et gérées par les couches inférieures.

TABLEAU 8/Q.775

Utilisateur du TC A	Utilisateur du TC B
Demande TC-INVOKE (1, Essai, Classe = 1) Indication TC-RESULT-NL (1, P1) Indication TC-RESULT-NL (1, P2) Indication TC-RESULT-L (1, P3) TC reçoit un résultat complet (1, P3) Duplication de P3: indication TC-L-REJECT (1, Code de problème)	 Indication TC-INVOKE (1, Essai) Demande TC-RESULT-NL (1, P1) Demande TC-RESULT-NL (1, P2) Demande TC-RESULT-L (1, P3)
	Indication TC-R-REJECT (1, Code de problème)
Temps	

2.4.3 Arrivée hors séquence de composants

L'ordre des résultats segmentés n'est pas significatif pour le TCAP. Il appartiendra donc au protocole d'application de fournir les mécanismes appropriés dans le cas où cet ordre est significatif pour l'utilisateur du TC (par exemple en introduisant un mécanisme de numérotation dans les réponses intermédiaires afin de les identifier, ou encore en utilisant un réseau en mode connexion).

Du fait d'une arrivée hors séquence, un résultat partiel peut arriver après un résultat complet. Lorsque cela se produit, le résultat partiel est rejeté par le TCAP. Cela est dû au fait que le résultat complet amène le TCAP à fermer le système d'état de lancement associé à cette opération; ainsi, lorsque le résultat partiel retardé est reçu, il ne peut pas être associé à un quelconque système d'état de lancement actif.

La séquence au Tableau 9 illustre ce qui se produit dans l'exemple E1 lorsque la dernière partie du résultat est reçue avant la seconde: les deux utilisateurs du TC sont informés.

Lorsqu'un lancement d'opération corrélé est reçu après le résultat complet de l'opération à laquelle il est corrélé (à la suite d'un mauvais séquencement), il est rejeté.

Le TCAP suppose une très faible probabilité d'arrivée hors séquence; si les performances du réseau sont jugées insuffisantes en la matière, il est préférable d'utiliser un réseau en mode connexion.

2.4.4 Rejet d'un composant par le TCAP

Un principe général du TCAP est de rejeter un composant (lancement d'opération ou réponse) reçu mal formé ou hors contexte (par exemple, réception d'une réponse sans lancement d'opération préalable). Ce rejet signifie:

1) que le TCAP forme un composant de rejet pour faire part à l'initiateur du composant concerné et il indique à l'utilisateur du TC local le composant de rejet qui attend d'être envoyé à l'extrémité distante, le TCAP fournissant toutes les informations disponibles sur la nature du composant rejeté (si l'utilisateur du TC éloigné n'a pas déjà mis fin au dialogue);

2) en réaction, l'utilisateur du TC local peut décider d'abandonner, de continuer ou de terminer le dialogue. Dans ces deux derniers cas, lorsque l'utilisateur du TC informe le TCAP de sa décision, l'utilisateur du TC homologue est informé du rejet.

TABLEAU 9/Q.775

Utilisateur du TC A	Utilisateur du TC B
Demande TC-INVOKE (1, Essai, Classe = 1)	Indication TC-INVOKE (1, Essai) Demande TC-RESULT-NL (1, P1)
Indication TC-RESULT-NL (1, P1) Indication TC-RESULT-L (1, P3) RR-NL (1, P2) PDU arrive Résultat hors séquence Rejet (pas de système d'état actif) Indication TC-L-REJECT (1, Code de problème)	Demande TC-RESULT-NL (1, P2) Demande TC-RESULT-L (1, P3)
	Indication TC-R-REJECT (1, Code de problème)
Temps	

Les cas possibles de rejet par le TCAP ont été présentés dans les paragraphes précédents. Le rejet par le TCAP entraîne la terminaison de l'opération à chaque fois que l'identificateur de lancement est reconnu: l'utilisateur du TC peut alors relancer l'opération terminée par un nouveau lancement d'opération. Lorsque le composant rejeté n'est pas identifiable, l'utilisateur du TC local est informé du composant malformé reçu et une valeur d'identificateur de lancement NULLE est incluse dans le composant de rejet qui attend d'être envoyé. Comme ce composant de rejet ne peut toutefois pas être associé à un lancement connu, la réaction appropriée pourrait être l'abandon du dialogue.

2.4.5 Chute de la temporisation d'opération

Lorsque le TCAP informe l'utilisateur du TC d'une chute de temporisation (indication TC-L-CANCEL), il indique que plus aucune information relative au lancement d'opération concerné ne peut être reçue (en particulier, aucun rejet). Si l'entité homologue continue d'envoyer des informations relatives à ce lancement d'opération, elles seront détruites lors de leur réception, à condition que l'identificateur de lancement de l'opération annulée n'ait pas été réattribué. La réattribution prématurée de valeurs d'identificateur de lancement est normalement évitée si les valeurs de temporisation sont correctement fixées et si la durée pendant laquelle un identificateur de lancement est «gelé» après que l'identificateur a été libéré a été déterminée. Cependant, un mécanisme, dépendant d'une réalisation, évitant la réattribution prématurée d'identificateurs de lancement est nécessaire afin de compenser les incertitudes concernant le temps de transfert d'une information, d'un utilisateur du TC à un autre, sans pour autant envisager le cas le plus défavorable (et, en général, le plus improbable).

L'indication de la chute de temporisation relate une situation anormale uniquement en cas d'opération de classe 1. L'utilisateur du TC est alors informé que, soit le lancement d'opération, soit le résultat a été perdu. Si aucun effet secondaire indésirable ne se produit, un autre lancement pour la même opération peut être effectué après la chute de la temporisation. Cela est illustré au Tablelau 10 par la séquence relative à l'exemple E1.

La chute de temporisation pour une opération de classe 2 indique qu'aucun résultat négatif n'a été reçu, ni ne sera accepté pour ce lancement: elle est une indication définitive de succès (classe 2) si l'on suppose qu'il n'y a pas de possibilité de perte de message dans le réseau. Une situation similaire s'applique à la classe 3 en cas d'échec. L'indication de chute de temporisation pour une opération de classe 4 est une décision locale.

3 Dialogues

A chaque fois qu'une des primitives de gestion des opérations considérées en 2 est émise, une demande est passée au TCAP, mais rien n'est envoyé à l'utilisateur du TC distant jusqu'à ce qu'une primitive demandant la transmission ne soit émise. Ces primitives et leurs relations avec les primitives de gestion des opérations sont abordées dans ce paragraphe.

TABLEAU 10/Q.775

Utilisateur du TC A	Utilisateur du TC B
Demande TC-INVOKE (1, Essai, Classe = 1)	Indication TC-INVOKE (1, Essai)
Chute de la temporisation: indication TC-L-CANCEL (1) Demande TC-INVOKE (2, Essai, Classe = 1)	
Temps	

3.1 Groupement de composants dans un message

L'émission par un utilisateur du TC d'une primitive de gestion de composant a pour effet de construire un composant à inclure dans un message (sauf si cette primitive n'a qu'un effet local). Le message n'est transmis que lorsque l'utilisateur du TC le demande.

Il est à noter qu'un composant peut également être engendré suite à une procédure de rejet par le TCAP: dans ce cas, ce composant est placé dans le prochain message du dialogue, sauf si le dialogue est abandonné.

Plusieurs composants peuvent être regroupés et envoyés à l'extrémité distante dans un même message dans la mesure où la taille maximale de message n'est pas dépassée, limitant ainsi la pénalisation en transmission. Cela est fait sous le contrôle de l'utilisateur du TC qui spécifie explicitement quand il veut envoyer un ou plusieurs composants. Les composants qui attendent d'être transmis sont ceux pour lesquels l'utilisateur du TC a préalablement émis une primitive de traitement des composants avec le même identificateur de dialogue.

Tant que la couche du SCCP du SS n° 7 n'a pas assuré la segmentation et réassemblé les fonctions, l'utilisateur du TC doit veiller à ce que la taille maximale d'un message de SS n° 7 ne soit pas dépassée.

L'exemple E3 donné au Tableau 11 montre le début d'un dialogue avec un point de commande de service où un commutateur demande des instructions (opération 1) et reçoit une demande pour connecter l'appel à une adresse de destination donnée et une demande pour envoyer l'information (par exemple, une annnonce parlée ou un message à présenter) au demandeur. Les deux composants sont contenus dans un seul message.

TC-BEGIN et TC-CONTINUE sont des primitives de transmission décrites en 3.2.

Il est possible d'avoir une primitive de transmission par composant, (et par conséquent un composant au maximum dans chaque message) ou moins de primitives de transmission que de composants, ce qui permet le groupement de plusieurs composants dans un message. De plus, l'information contenue dans les paramètres des primitives de transmission (par exemple, l'information d'adressage) s'applique à tous les composants du message.

A l'extrémité origine, la primitive demandant la transmission apparaît après les primitives de gestion des composants indiquant que la transmission de tous les composants précédents doit être faite immédiatement. Cela évite de lier la transmission de composants spécifiques à une primitive de transmission donnée et autorise les primitives de transmission sans aucun composant associé.

A l'extrémité de destination, la primitive indiquant la réception de composants transmis apparaît en premier. Elle contient l'information de commande nécessaire au TCAP pour délivrer chacun des composants du message (s'il y en a). Le dernier composant du message est signalé à l'utilisateur du TC par le paramètre «dernier composant». Les composants sont remis à l'utilisateur du TC de destination dans le même ordre qu'ils ont été passés au TCAP par l'utilisateur du TC de l'origine.

TABLEAU 11/Q.775

Utilisateur du TC A	Utilisateur du TC B
Demande TC-INVOKE (1, Fournir instructions, Classe = 1) Demande TC-BEGIN (Paramètres de commande)	
	Indication TC-BEGIN (Paramètres de commande) Indication TC-INVOKE (1, Fournir instructions) Demande TC-INVOKE (2, 1, Connecter appel) Demande TC-RESULT-L (1, Envoyer info) Demande TC-CONTINUE (Paramètres de commande)
Indication TC-CONTINUE (Paramètres de commande) Indication TC-INVOKE (2, 1, Connecter appel) Indication TC-RESULT-L (1, Envoyer info)	
Temps	

3.2 Services de gestion du dialogue

Lorsque deux utilisateurs du TC coopèrent dans une application, plus d'un lancement d'opération est généralement nécessaire. Le flot de composants résultant doit être identifié de telle sorte que:

1) les composants liés au même flot puissent être identifiés;

2) les flots correspondant à plusieurs instances d'une même application puissent être identifiés et se dérouler en parallèle.

Chacun de ces flots est qualifié de dialogue par l'utilisateur du TC et est identifié par un paramètre identificateur de dialogue correspondant. Le service de gestion de dialogue assuré à cette fin est le dialogue structuré.

Quand un seul message est nécessaire pour réaliser une application répartie, le message unidirectionnel du dialogue non structuré peut être utilisé. L'initiateur n'attend pas de notification du résultat de l'opération (il peut seulement lancer des opérations de classe 4), mais il peut recevoir notification d'une erreur de protocole, le cas échéant. La notification d'une erreur de protocole sera également acheminée dans un message unidirectionnel.

3.2.1 Dialogue structuré

3.2.1.1 Considérations générales

L'utilisation de dialogues permet à plusieurs flots de composants indépendants de coexister entre deux utilisateurs du TC. Le paramètre identificateur de dialogue est utilisé par les primitives de gestion des opérations et par les primitives de gestion de la transmission (dialogue) afin de déterminer quel(s) composant(s) appartiennent à quel dialogue.

Dans les exemples ci-après, le paramètre identificateur de dialogue est représenté (par convention) par le premier paramètre dans ces primitives, commençant par la lettre D. Chaque utilisateur du TC a sa propre référence pour un dialogue donné. Les références locales (celles utilisées à l'interface) sont représentées ici, la correspondance entre ces références locales et les références de protocoles (dénommées identificateurs de transaction) incluses dans les messages est faite par le TCAP.

Trois primitives ont été définies pour la gestion de dialogues dans des conditions normales. Elles indiquent l'établissement d'un dialogue (primitive TC-BEGIN), la continuation d'un dialogue (primitive TC-CONTINUE) ou la terminaison d'un dialogue (TC-END). Chacune des ces primitives peut être utilisée pour demander la transmission de 0, 1 ou plusieurs composants; ces composants peuvent contenir des informations relatives à une ou plusieurs opérations.

Le Tableau 12 illustre une séquence possible pour l'exemple E2, où la demande d'essai débute le dialogue, qui se termine lorsque le résultat de l'essai a été envoyé.

TABLEAU 12/Q.775

Utilisateur du TC A	Utilisateur du TC B
Demande TC-INVOKE (D1, 1, Essai, Classe = 1) Demande TC-BEGIN (D1, Adresse)	
	Indication TC-BEGIN (D2, Adresse) Indication TC-INVOKE (D2, 1, Essai) Demande TC-INVOKE (D2, 2, 1, Sélection d'options, Classe = 1) Demande TC-CONTINUE (D2)
Indication TC-CONTINUE (D1) Indication TC-INVOKE (D1, 2, 1, Sélection d'options) Demande TC-RESULT-L (D1, 2, Options) Demande TC-CONTINUE (D1)	
	Indication TC-CONTINUE (D2) Indication TC-RESULT-L (D2, 2, Options) Demande TC-RESULT-L (D2, 1, Résultat de l'essai) Demande TC-END (D2)
Indication TC-END (D1, Normal) Indication TC-RESULT-L (D1, 1, Résultat de l'essai)	
Temps	
NOTE – D1 et D2 sont les références locales d'un même dialogue et correspondent aux identificateurs de transactions qui apparaissent dans les messages.	

N'importe quel groupement de composants est autorisé dans les messages d'un dialogue: le TCAP ne vérifie pas, par exemple, qu'un message terminant un dialogue ne contient pas des lancements d'opération de classe 1.

L'échange bidirectionnel de composants est adopté: si un utilisateur du TC veut introduire certaines restrictions, par exemple un fonctionnement en mode synchrone tel que défini pour les utilisateurs de ROSE, il lui appartiendra de définir lui-même les procédures nécessaires.

3.2.1.2 Echange de messages

La transmission de messages est effectuée avec la qualité de service des services des couches sous-jacentes: aucun contrôle de flux, aucun mécanisme de correction d'erreur n'est offert par le TCAP.

– La première primitive de gestion du dialogue d'un dialogue doit indiquer l'établissement d'un dialogue (TC-BEGIN). Les messages ultérieurs ne doivent pas être envoyés par l'extrémité origine du dialogue tant qu'un message, indiquant la continuation du dialogue, n'a pas été reçu dans la direction opposée.

– Si un utilisateur du TC essaie d'envoyer un grand nombre de messages dans un court laps de temps, aucun mécanisme de contrôle de flux du TCAP ne l'empêchera.

– La classe 1 du SCCP, garantissant le séquencement, peut être demandée en tant qu'option et indiquée par le paramètre qualité de service. Il est à noter que cette option peut ne pas être disponible de bout en bout en cas d'interfonctionnement avec un réseau qui ne la fournit pas.

3.2.1.3 Terminaison d'un dialogue

Le TCAP ne met aucune restriction sur la possibilité qu'a un utilisateur du TC de demander la terminaison d'un dialogue. Il s'ensuit que des messages peuvent être perdus si aucune précaution n'est prise par l'application concernant le moment où un dialogue peut être terminé.

En particulier, si le protocole d'application permet aux deux utilisateurs du TC d'envoyer des primitives TC-END à peu près au même moment, et si ces primitives provoquent la transmission de composants, il est vraisemblable que certains de ces composants (sinon tous) ne seront pas remis à leur utilisateur du TC de destination respectif.

C'est à l'application de définir, si nécessaire, ses propres règles concernant le droit de terminer un dialogue: le TCAP ne les vérifiera pas.

N'importe quel message reçu pour un dialogue terminé est détruit s'il demande la terminaison d'un dialogue, et tout message autre qu'un message de terminaison ou d'abandon entraîne l'abandon du dialogue à l'entité distante.

Il faut noter qu'un utilisateur du TC ne peut rejeter, au moyen d'une primitive de demande TC-U-REJECT, un composant quelconque reçu dans un message END. Il est important qu'une application puisse rejeter tout composant reçu ou soit avertie des rejets par l'extrémité distante, tous les composants doivent être alors inclus soit dans le message initial BEGIN, soit dans les messages CONTINUE suivants.

Le dialogue prend fin soit grâce à la méthode prédéterminée, soit en envoyant un message END ne contenant pas de composants ou de composants REJECT (le cas échéant).

Les différences entre les trois moyens de terminer un dialogue sont décrites ci-dessous.

Terminaison prédéterminée

Une application typique est l'accès à une base de données distribuée, où l'utilisateur demandeur (utilisateur du TC A) ignore où l'information qu'il cherche est localisée. L'utilisateur du TC A diffuse une demande à chaque localisation qui pourrait avoir l'information requise et recevra, éventuellement, une réponse de l'utilisateur du TC qui détient cette information. La terminaison prédéterminée évite que des messages, en provenance des autres destinations, ne soient envoyés pour dire: «je n'ai pas cette information». Seule la destination qui répond peut, si elle le souhaite, continuer le dialogue. Par convention, toutes les autres destinations termineront le dialogue localement: lorsque l'initiateur des demandes recevra une réponse, il terminera également localement les dialogues avec les destinations qui n'ont pas répondu. Cette convention est passée entre applications: le TCAP ne vérifiera pas qu'elle est respectée, ni qu'elle est mentionnée dans le protocole TCAP.

L'exemple E4 dans le Tableau 13 illustre cette situation dans les conditions suivantes: deux destinations B1 et B2; deux dialogues (D1, D2) et (D3, D4) sont engagés; B1 possède l'information demandée et décide de continuer le dialogue.

Une terminaison prédéterminée peut également être utilisée lorsque l'utilisateur du TC veut envoyer l'information sans attendre de réponse d'aucune sorte ultérieurement.

Terminaison de base

La réception d'une primitive TC-END d'un utilisateur du TC provoque la transmission de composants en suspens à l'extrémité distante. Le TCAP ne vérifie pas que tous les lancements d'opération ont reçu une réponse lorsque la terminaison du dialogue est demandée: aucune information n'est donnée à l'utilisateur du TC que des lancements d'opération n'ont pas reçu de résultat complet.

A l'extrémité de réception, le dialogue est considéré comme terminé lorsque tous les composants reçus, dans un message indiquant la terminaison, ont été remis à l'utilisateur du TC.

Exemple: le dialogue prend fin lorsque l'essai de l'exemple E1, Tableau 14, reçoit une réponse.

TABLEAU 13/Q.775

Utilisateur du TC A	Utilisateur du TC B1	Utilisateur du TC B2
Demande TC-INVOKE (D1, 1, Question) Demande TC-BEGIN (D1, Adresse) Demande TC-INVOKE (D3, 1, Question) Demande TC-BEGIN (D3, Adresse)	Indication TC-BEGIN (D2, Adresse) Indication TC-INVOKE (D2, 1, Question) Demande TC-RESULT-L (D2, 1, Réponse) Demande TC-CONTINUE (D2)	Indication TC-BEGIN (D4, Adresse) Indication TC-INVOKE (D4, 1, Question) B2 ne détient pas l'information recherchée Demande TC-END (D4, Local)
Indication TC-CONTINUE (D1) Indication TC-RESULT-L (D1, 1, Réponse) D1 continue D3 se termine localement Demande TC-END (D3, Local)		
Temps		

TABLEAU 14/Q.775

Utilisateur du TC A	Utilisateur du TC B
..... Indication TC-END (D1) Indication TC-RESULT-NL (D1, 1, P1) Indication TC-RESULT-NL (D1, 1, P2) Indication TC-RESULT-L (D1, 1, P3) Fin du dialogue pour A	..... Demande TC-RESULT-NL (D2, 1, P1) Demande TC-RESULT-NL (D2, 1, P2) Demande TC-RESULT-L (D2, 1, P3) Demande TC-END (D2, Normal) Fin du dialogue pour B
Temps	

Abandon par l'utilisateur du TC

L'abandon permet à un utilisateur du TC d'arrêter le dialogue à n'importe quel moment. L'abandon du service par l'utilisateur est un cas typique d'abandon du dialogue. Les principales différences entre un abandon et une terminaison normale sont:

- les composants pour lesquels la transmission est en cours ne sont pas envoyés à l'entité homologue;
- l'information utilisateur du TC entre entités homologues peut être transmise lorsque l'abandon est émis et remise à l'utilisateur du TC distant.

La séquence donnée au Tableau 15 montre un abandon par l'utilisateur dans l'exemple E2.

TABLEAU 15/Q.775

Utilisateur du TC A	Utilisateur du TC B
Demande TC-INVOKE (D1, 1, Essai, Classe = 1) Demande TC-BEGIN (D1, Adresse)	
	Indication TC-BEGIN (D2, Adresse) Indication TC-INVOKE (D2, 1, Essai) Demande TC-INVOKE (D2, 2, 1, Sélection des options, Classe = 1) Demande TC-CONTINUE (D2)
Indication TC-CONTINUE (D1) Indication TC-INVOKE (D1, 2, 1, Sélection d'options) Demande TC-U-ABORT (D1, Cause)	
	Indication TC-U-ABORT (D2, Cause)
Temps	

3.2.1.4 Situations anormales relatives aux messages

Ces situations sont considérées indépendamment des effets de ces événements dans la sous-couche composant.

Perte de message

Le TCAP n'offre aucune protection contre la perte de message. Trois cas sont identifiés:

1) Le message débute un nouveau dialogue: le dialogue n'existera qu'à l'extrémité origine, et aucun message ne sera accepté quelle que soit la direction. Eventuellement, un mécanisme, dépendant d'une réalisation à l'extrémité origine, terminera le dialogue.

2) Le message continue un dialogue existant: la perte n'est pas détectée. Le TCAP réagira (ou non) à la perte des composants contenus dans le message tel qu'indiqué en 2.4.1.

3) Le message termine un dialogue: le TCAP réagira éventuellement (par une chute de la temporisation comme indiqué en 2.4.1) si ce message contient une réponse à une opération de classe 1, sinon un mécanisme dépendant d'une réalisation peut terminer le dialogue à l'extrémité de destination.

Duplication de message

La duplication d'un message BEGIN provoque l'ouverture de deux transactions comme indiqué ci-dessous: chacune de ces transactions a son propre identificateur local et le même identificateur de destination. L'utilisateur du TC détecte éventuellement une situation anormale, et les deux dialogues sont abandonnés.

La séquence donnée au Tableau 16, illustre la duplication d'un message BEGIN dans l'exemple E2.

TABLEAU 16/Q.775

Utilisateur du TC A	Utilisateur du TC B
Demande TC-INVOKE (D1, 1, Essai, Classe = 1) Demande TC-BEGIN (D1, Adresse)	
Indication TC-CONTINUE (D1) Indication TC-INVOKE (D1, 2, 1, Sélection d'options)	Indication TC-BEGIN (D2, Adresse) Indication TC-INVOKE (D2, 1, Essai) Duplication de Begin: indication TC-BEGIN (D3, Adresse) Indication TC-INVOKE (D3, 1, Essai) Réponse au premier Begin demande TC-INVOKE (D2, 2, 1, Sélection d'options, Classe = 1) Demande TC-CONTINUE (D2) Réponse au second Begin demande TC-INVOKE (D3, 2, 1, Sélection d'options, Classe = 1) Demande TC-CONTINUE (D3)
Indication TC-CONTINUE (D1) Indication TC-INVOKE (D1, 2, 1, Sélection d'options) L'utilisateur du TC considère que ce lancement est anormal et peut le rejeter ou abandonner un des dialogues: demande TC-U-ABORT (D1, Cause)	
	Indication TC-U-ABORT (D3, Cause)
Temps	

A cet instant, il y a encore un dialogue (avec l'identificateur local D2) à l'extrémité B, mais aucun dialogue à l'extrémité A. L'utilisateur du TC B recevra une indication du TCAP lorsque la temporisation de l'opération 2 du dialogue D2 chutera faute de réponse (indication TC-L-CANCEL), et pourra alors décider d'abandonner D2. Il est à noter que la situation serait plus difficile à détecter si l'utilisateur du TC B n'avait pas lancé une opération de classe 1.

La duplication d'un message CONTINUE n'est pas détectée par le TCAP.

Lorsqu'un messsage END est dupliqué, le second message est reçu avec un identificateur qui ne correspond pas à un dialogue en cours: le TCAP détruit le message dupliqué.

Arrivée hors séquence de messages

Lorsque les messages hors séquence ne concernent ni l'établissement, ni la terminaison d'un dialogue, le mauvais séquencement n'est pas détecté par le TCAP mais peut entraîner un mauvais séquencement des composants, auquel le TCAP réagit comme indiqué en 2.4.3.

Lorsqu'un message indiquant la continuation d'un dialogue arrive après un message indiquant la terminaison de ce même dialogue, il n'est pas passé et provoque l'abandon du dialogue par le TCAP, l'utilisateur du TC détectera probablement la perte en recevant une indication de terminaison de dialogue prématurée. Si l'application a besoin de surmonter ce problème, un nouveau dialogue doit être établi.

Message erroné

Lorsqu'un message erroné est reçu, le TCAP réagit comme il est indiqué dans la Recommandation Q.774.

La séquence de primitives, lorsque le TCAP décide d'abandonner le dialogue suite à la réception d'un message erroné dans l'exemple E2, est indiquée dans le Tableau 17.

TABLEAU 17/Q.775

Utilisateur du TC A	Utilisateur du TC B
Demande TC-INVOKE (D1, 1, Essai, Classe = 1) Demande TC-BEGIN (D1, Adresse)	
	Indication TC-BEGIN (D2, Adresse) Indication TC-INVOKE (D2, 1, Essai) Demande TC-INVOKE (D2, 2, 1, Sélection d'options, Classe = 1) Demande TC-CONTINUE (D2)
Message erroné indication TC-P-ABORT (D1, Cause)	Indication TC-P-ABORT (D2, Cause)
Temps	

3.2.1.5 Relations entre la gestion du dialogue et la gestion des composants

La suite donne quelques directives sur le moment où la terminaison d'un dialogue peut être demandée; si elles ne sont pas respectées, le TCAP ne refusera pas la demande de terminaison du dialogue.

Les problèmes qui peuvent résulter de la collision de messages demandant la terminaison du dialogue ont été décrits précédemment.

Une terminaison normale ne devrait pas être demandée lorsque:

- il y a des lancements d'opération en cours pour le dialogue;
- le protocole d'application anticipe que les réponses transmises avec la demande de terminaison pourraient être rejetées.

De nombreuses applications pourraient ne pas définir les scénarios de rétablissement en cas de rejet d'une réponse. Cela justifie la transmission de résultats ou d'opérations de classe 4 dans un message indiquant la terminaison d'un dialogue. Les autres applications devraient, soit envisager l'utilisation d'un service réseau en mode connexion, soit terminer le dialogue par un message, ne contenant pas de composant, qui serait envoyé seulement lorsqu'une indication de rejet ne peut plus être reçue.

Il est recommandé de ne pas envoyer une opération pour laquelle une réponse est attendue dans un message END (l'utilisateur du TC n'émet pas de primitive de demande TC-END pour déclencher le composant associé). Il convient de faire observer qu'il ne s'agit pas d'une condition du protocole du TCAP ni de l'utilisateur du TC mais d'une directive judicieuse. Si l'utilisateur du TC choisit d'envoyer un composant de lancement pour une opération de classe 1, 2 ou 3 dans un message END, son choix n'a aucune implication pathologique pour les systèmes de protocole du TCAP homologues ni pour les utilisateurs de TC. La manière dont les composants produits à la suite de l'exécution de l'opération en l'absence de dialogue actif sont détruits est une question locale. A vrai dire, le fait que l'utilisateur du TC qui a lancé l'opération ait choisi de les inclure dans un message END signifie qu'il a décidé pour cet exemple de passer outre à la sémantique inhérente à l'opération et qu'il ne se soucie guère de savoir si l'opération a pu ou non être exécutée.

3.2.1.6 Questions d'adressage

L'utilisateur du TC qui débute un dialogue doit donner l'adresse de destination et l'adresse d'origine au TC. Il appartient à l'utilisateur du TC qui fournit l'adresse d'origine de la donner de manière que le TC distant puisse l'utiliser, sans procéder à une vérification, pour l'atteindre.

Quand un SCCP sans connexion assurant les capacités du TC est utilisé, toutes les options d'adressage qu'il offre peuvent être choisies.

Pendant l'instauration d'un dialogue, la combinaison des adresses peut être optimisée à la fois par l'utilisateur du TC B et par le TCAP.

Le paramètre du destinataire facultatif d'adresse dans la première primitive de demande TC-CONTINUE peut servir à modifier l'adresse qui devrait être utilisée pour acheminer les messages associés au dialogue. La destination véritable ne devrait pas être affectée par cette modification. S'il s'agit d'une nouvelle destination, il faudra mettre fin au dialogue et en commencer un nouveau vers la nouvelle destination.

Exemples de modification d'adresse qui n'affectent pas la destination véritable:

- l'adresse initiale était un titre global, l'adresse suivante est un PC et un SSN pour la même destination afin de permettre un acheminement optimal (qui ne traverse pas la limite du réseau);
- un titre global général est utilisé pour sélectionner une base de données à partir d'un ensemble de bases de données répétées. La base de données interrogée renvoie son propre titre global spécifique.

3.2.1.7 Qualité de service

Les primitives de demande de gestion du dialogue permettent à l'utilisateur du TC d'exiger une qualité de service donnée de la couche réseau. Si l'utilisateur du TC ne formule aucune demande particulière, le SCCP sélectionne des options par défaut (c'est-à-dire pas d'option retour, pas de mise en séquence).

Si le contrôle de séquence est demandé, l'utilisateur du TC est aussi chargé de fournir les informations permettant à la couche réseau d'identifier le flux des messages connexes qui doivent être remis en séquence.

Il est conseillé aux utilisateurs de TC de demander le contrôle de séquence lorsqu'ils utilisent les services TC-RESULT-NL.

3.2.2 Dialogue non structuré

Un message unidirectionnel contient soit des lancements d'opération uniquement de classe 4, soit des notifications d'erreurs de protocole dans ces lancements. Des composants multiples peuvent être transmis dans un message unidirectionnel, pourvu que la longueur maximale du message ne soit pas dépassée.

3.3 Service amélioré de gestion du dialogue

3.3.1 Vue d'ensemble

Etant donné que le nombre d'applications de signalisation augmente, il faudra pouvoir les différencier pendant une communication, notamment lorsque plusieurs d'entre elles se trouvent au même endroit dans un nœud de SS n° 7.

La possibilité de signaler au début du dialogue le protocole d'application (parmi plusieurs éventuellement) qui intervient dans l'échange ultérieur de messages est assurée par les fonctions facultatives et le protocole de portion de dialogue. La portion de dialogue facultative permet la négociation du contexte d'application et, à titre d'option supplémentaire, le transfert transparent des données d'utilisateur qui ne sont pas des composants. Ces dernières pourraient servir par exemple à acheminer des données d'initialisation, des versions de protocoles d'utilisateur, d'autres précisions du contexte d'application, des mots de passe, etc.

3.3.2 Utilisation du contexte d'application

L'utilisateur du TC qui commence un dialogue peut proposer un contexte d'application à son homologue en incluant un nom de contexte d'application dans la primitive de demande TC-BEGIN. Le contexte d'application se rapporte à l'ensemble d'éléments ASE et aux règles de coordination associées qui peuvent être nécessaires pendant le dialogue.

Si le nom du contexte d'application est acceptable, l'utilisateur du TC qui répond peut soit décider de poursuivre le dialogue, soit y mettre fin normalement. Il doit inclure le même nom de contexte dans la première (ou l'unique) primitive de demande de gestion du dialogue qu'il utilise sauf s'il demande une terminaison de dialogue prédéterminée.

Si le nom du contexte d'application n'est pas acceptable, l'utilisateur du TC peut choisir:

i) de détruire les composants reçus et d'émettre une primitive de demande TC-U-ABORT pour faire savoir qu'il refuse le dialogue. Il doit inclure un nom de contexte d'application dans cette primitive qui correspond soit à celui reçu, soit à un autre qui sera utilisé par l'initiateur du dialogue pour faire une nouvelle tentative. Le TC n'a pas de fonction type permettant à son utilisateur de proposer plus d'un nom de contexte d'application de rechange. Une procédure de ce type peut toutefois être définie en dehors du TC avec le paramètre d'information d'usager (voir 3.3.3);

ii) de poursuivre le dialogue en indiquant qu'il utilise un autre contexte d'application [par exemple un contexte qui n'utilise pas le(s) ASE(s) qu'il ne prend pas en charge] en incluant un autre nom de contexte dans la première primitive de demande TC-CONTINUE. Les composants reçus peuvent être conservés ou détruits selon l'accord préalablement conclu entre les utilisateurs de TC;

iii) de mettre fin au dialogue normalement en indiquant que la (les) réponse(s) qui figure(nt) dans le message END repose(nt) sur un autre contexte d'application [par exemple un contexte qui n'utilise pas l'(les) ASE qu'il ne prend pas en charge] en incluant un autre nom de contexte d'application dans la primitive de demande TC-END.

L'utilisateur du TC peut aussi donner un nom de contexte d'application lorsqu'il se sert d'un service UNI-TC. Dans ce cas, le nom du contexte permet à l'utilisateur du TC homologue d'interpréter les composants reçus.

Il importe de noter que l'information relative au contexte d'application qui est acheminée dans l'APDU de gestion du dialogue est un nom du type OBJECT IDENTIFIER (IDENTIFICATEUR D'OBJET). Ce nom fait référence à une spécification (document) dans laquelle le contexte d'application est décrit. Ce document peut renvoyer à d'autres spécifications où, par exemple, la syntaxe abstraite de certains protocoles d'application est décrite. Des spécifications de ce type peuvent être énoncées sous forme de notation formelle ou semi-formelle ou encore de texte clair.

La spécification des contextes d'application, leur sémantique, le choix d'une valeur d'identificateur d'objet et la diffusion de l'information à toutes les parties qui souhaitent communiquer sont autant de modalités qui permettent d'enregistrer un contexte d'application. Dans le cas où les contextes d'application sont enregistrés dans des Recommandations relatives à la signalisation du RNIS, un exemple de valeur type pourrait être {ccitt recommendation q xxx ac-name (y)} pour le contexte d'application y décrit dans la Recommandation q.xxx.

S'il est en principe possible de décrire le contexte d'application en détail et d'ajouter de nouveaux contextes d'application pour tenir compte de tous les cas qui peuvent se produire, il est sans doute plus facile de conserver ces spécifications si les contextes d'application ne sont pas trop nombreux. Supposons par exemple qu'un nom de contexte d'application donné se réfère à l'utilisation combinée des ASE A et B. Dans certains cas, il peut être nécessaire d'indiquer que seul un sous-ensemble des fonctions de A et/ou B sera utilisé pour une communication donnée. Au lieu d'enregistrer un nouveau nom de contexte d'application pour couvrir ce cas, la même information peut être acheminée dans une syntaxe acceptable par les deux parties dans le champ d'information d'usager des APDU AARQ et AARE.

3.3.3 Utilisation des données d'utilisateur

Les primitives de gestion du dialogue du TC permettent à l'utilisateur du TC de transférer des informations qui n'ont pas trait aux services de gestion des composants (c'est-à-dire qui ne reposent pas sur le paradigme de l'opération distante). Ces informations sont acheminées soit dans le champ d'information-utilisateur de la PDU de gestion du dialogue, soit directement dans la portion de dialogue une fois ce dernier établi.

Un service de ce type est nécessaire au moment de l'établissement du dialogue pour transmettre certaines données d'initialisation à l'homologue (précisions relatives au contexte d'application, données d'authentification, identification d'un sous-processus de destination, etc.)

Ce service peut aussi être utilisé pour négocier le contexte d'application quand l'utilisateur du TC refuse un dialogue (abandon par l'utilisateur pendant le dialogue avec une raison pour l'abandon, à savoir l'absence de prise en charge du contexte d'application), il peut insérer une liste de noms de contextes d'application de rechange dans le champ de

données d'utilisateur de la primitive de demande TC-U-ABORT. Ces noms sont alors acheminés dans le cadre des données d'utilisateur de l'unité de données de protocole de dialogue (ABRT). L'utilisateur du TC qui est à l'origine de la demande d'établissement du dialogue peut faire une nouvelle tentative avec l'un de ces contextes.

Pour utiliser ce service, les deux utilisateurs de TC devront définir la syntaxe et la sémantique de l'information à acheminer, s'il y en a une, dans chaque APDU de dialogue pour tous les contextes d'application. Comme le type de cette information d'utilisateur en ASN.1 est EXTERNAL, la syntaxe de l'information peut être écrite en utilisant la notation ASN.1 ou toute autre notation spécifique à l'utilisateur. La manière dont l'information est codée peut aussi être spécifique à l'utilisateur. Le type EXTERNAL permet d'intégrer la valeur de données à partir d'une syntaxe abstraite (dans ce cas une syntaxe spécifique à l'utilisateur) dans une autre (celle des APDU de dialogue).

La Recommandation X.208 définit le type EXTERNAL comme suit:

```
EXTERNAL ::= [UNIVERSAL 8] IMPLICIT SEQUENCE {
direct-reference                OBJECT IDENTIFIER OPTIONAL,
indirect-reference              INTEGER OPTIONAL,
data-value-descriptor           ObjectDescriptor OPTIONAL,
encoding                                CHOICE {
  single-ASN1-type              [0] ANY,
  octet-aligned                 [1] IMPLICIT OCTET STRING,
  arbitrary                             [2] IMPLICIT BIT STRING }}
```

Parmi les trois formes de dénotation qui permettent d'identifier le type et le codage de la valeur de données qui est contenue dans la construction EXTERNAL, les utilisateurs de TC doivent utiliser la dénotation directe. Le nom de la dénotation directe est essentiel pour identifier à la fois la syntaxe abstraite de la valeur de données et les règles de codage qui s'appliquent. La dénotation indirecte sert à identifier le contexte de présentation dont l'utilisation n'est pas, actuellement, prise en charge par le réseau sémaphore n° 7. En plus de la dénotation directe, l'utilisateur du TC peut aussi décrire implicitement la valeur de données dans une notation informelle en ayant recours au descripteur de la valeur de données.

Si la valeur de données externe correspond à un type d'ASN.1 unique et si les règles de codage de base servent à le coder, le choix de champ de «codage» importe peu. Si le codage de cette valeur de données qui a été choisi se traduit par un nombre entier d'octets, le codage «aligné-octets» ou «arbitraire» peut être utilisé. Dans le cas contraire, il convient de choisir l'option «arbitraire».

Comme le protocole permet à une SEQUENCE OF EXTERNAL d'être présente dans un champ d'information d'utilisateur facultatif des APDU de gestion du dialogue, les deux utilisateurs de TC ne sont pas limités, lorsqu'ils définissent un contexte d'application, à un nombre donné dans la séquence. (Si la segmentation n'est pas assurée par le SCCP, les utilisateurs de TC devront veiller à ce que la taille des messages du système de signalisation n° 7, qui est limitée, soit respectée).

3.3.4 Questions de compatibilité amont

Les fonctions et le protocole nouveaux décrits ci-dessus sont facultatifs et les procédures de protocole qui sont spécifiées dans les Recommandations Q.771 à Q.774 sont faciles à distinguer et à retirer de la documentation remise à l'utilisateur, des spécifications relatives aux mises en œuvre et à l'interface entre les réseaux qui prennent ces Recommandations comme base. Dans ce cas, il convient de se reporter aux messages du TCAP définis dans les Recommandations de 1988. Aucun réseau n'est obligé d'accepter ces nouvelles fonctions s'il n'offre pas des services qui les exigent.

Un nœud qui accepte la version du TCAP de 1988 ne comprendra pas les APDU associées au contexte d'application dû à ce nœud conforme à la présente Recommandation (version de 1992) et arrêtera donc la transaction en utilisant la cause d'abandon partiel «portion de transaction incorrecte». Si l'utilisateur du TC à un nœud qui prend en charge le TCAP conforme à la présente Recommandation (version de 1992) reçoit un message d'abandon avec la cause susmentionnée en réponse à un débat de dialogue utilisant l'information du contexte d'application, il doit l'interpréter, du moins dans les cas où un réseau prend en charge diverses mises en œuvre du TCAP d'après la présente Recommandation (1992) et les Recommandations de 1988 parce qu'il communique avec un nœud qui n'accepte que les Recommandations relatives au TCAP de 1988 et non en raison d'une véritable erreur de syntaxe (qui est très peu probable). L'utilisateur du TC peut donc essayer de retransmettre le message sans autres informations supplémentaires relatives au contexte d'application que celles qui sont essentielles à l'application.

Il est vraisemblable que pendant un certain temps, un réseau acceptera les mises en œuvre du TCAP conformes aux Recommandations de 1988 et à la présente Recommandation (1992) ainsi que des applications qui exigent ou non la prise en charge du mécanisme de contexte d'application. Ces possibilités ne relèvent pas des présentes normes mais ne doivent pas être oubliées au moment où les services sont déployés si l'on veut éviter de transmettre le message initial deux fois.

4 Directives pour l'élaboration des protocoles utilisateurs de TC

4.1 Introduction

La Recommandation Q.1400 décrit la manière dont les éléments de service d'application (ASE) (*application service element*), les contextes d'application (AC) (*application-context*) et les entités d'application (AE) (*application entity*) sont structurés et celle dont une AE est traitée dans le système de signalisation n° 7. Le présent paragraphe illustre cette architecture en tenant compte de la découpe fonctionnelle d'une application et décrit la manière dont les AE, les AC, les ASE, les opérations et les erreurs devraient être définis.

4.2 Découpe fonctionnelle dans une application

Un processus d'application (AP) (*application process*) de signalisation communique à travers une partie de son logiciel consacré exclusivement aux communications qui est appelée entité d'application (AE) (*application entity*). Une AE contient donc toutes les fonctions nécessaires pour que les divers AP puissent communiquer. A maintes reprises, on a constaté que pour un certain nombre d'applications, les fonctions de communication pouvaient être groupées dans des ensembles intégrés d'actions de manière que chacun de ces ensembles puisse être utilisé dans plusieurs AE. Un ensemble intégré d'actions de ce type qui peut être utilisé dans plusieurs AE est appelé un élément de service d'application (ASE) (*application service element*). Il va sans dire qu'il existe toujours certaines fonctions de communication spécifiques à une application qui ne peuvent servir qu'à répondre aux besoins de l'application donnée pour laquelle elles ont été définies. Le thème des AP, AE et ASE est développé dans la Recommandation Q.1400.

Ainsi, le TCAP peut être considéré comme un ASE, car il offre un moyen général à toutes les applications de signalisation de communiquer en utilisant le paradigme des opérations distantes sur un service de réseau sans connexion. Un ASE utilisateur du TC est constitué de définitions d'opérations distantes qui, ensemble, fournissent un protocole global de communication à une application de signalisation. La définition de l'ASE indique aussi quel utilisateur du TC homoloque peut lancer une opération, en précisant la nature de cette dernière, et dans quel ordre. Si l'un ou l'autre des utilisateurs du TC peut lancer n'importe quelle opération, on dit que l'ASE est symétrique. La manière dont les opérations sont définies et groupées est décrite en 4.5.

Du point de vue de l'utilisateur du TC, le mécanisme qui permet d'obtenir les services d'un ASE utilisateur du TC est le lancement des opérations de ce dernier. Chaque opération fournit une partie du service de l'ASE de manière intrinsèquement asymétrique car elle est lancée par un utilisateur du TC et exécutée par l'homologue distant. Les utilisateurs de TC ne sont toutefois pas toujours asymétriques (c'est-à-dire qu'un utilisateur ne se limite pas toujours à exécuter des opérations et un autre à les lancer) car chacun peut être en mesure de lancer et d'exécuter les mêmes opérations ou des opérations différentes. [En fait, l'interface du service du point de vue de l'utilisateur du TC (qui fera l'objet d'un complément d'étude) peut sembler très différente de celle fournie par le TC. Le lancement par exemple d'une opération de classe 1 peut être considéré par l'utilisateur du TC comme le lancement d'un service confirmé alors que dans la perspective de l'interface du service du TC, il résulte de deux services non confirmés, à savoir TC-INVOKE et TC-RESULT].

4.3 Comment spécifier une AE entité d'application et un contexte d'application

Un type d'AE représente un ensemble d'ASE. Quand deux AE communiquent, les interactions entre ces deux entités ainsi que les interactions entre les ASE dans une AE sont régies par les règles d'un contexte d'application.

Un ou plusieurs contextes d'application peuvent être associés à un type d'AE. Chacun représente l'utilisation de sous-ensembles différents de l'ensemble d'ASE inclus dans le type d'AE et/ou des règles de coordination différentes.

Le concepteur d'une application doit définir chaque type d'AE. Cette définition doit au moins contenir:

- une description générale des services fournis par la combinaison de deux AE homologues communiquant par un dialogue;
- une vue d'ensemble de la structure de l'AE;
- la liste des AC acceptés.

Le concepteur d'une application doit définir chaque AC pris en charge par le type d'AE. Cette définition doit au moins contenir:

- une description générale affirmant la description du type d'AE;
- une définition du protocole d'application complet entre AE homologues en:
  - i) identifiant chaque ASE utilisé par l'AC (parmi ceux qui constituent le type d'AE) et en indiquant quelles AE homologues initient le service;

ii) indiquant les règles de coordination utilisées entre ces ASE (par exemple concaténation des PDU d'ASE utilisateurs du TC différents, contraintes relatives à l'ordre dans lequel les opérations provenant d'ASE utilisateurs du TC différents peuvent être lancées, etc.) outre les règles qui font partie intégrante des spécifications relatives aux ASE;

iii) précisant la (les) syntaxe(s) abstraite(s) requise(s) par les ASE;

– les contraintes spéciales afin de garantir la compatibilité entre différentes versions d'AE homologues.

Un nom doit être donné à chaque contexte d'application. Ce nom est une valeur du type OBJECT IDENTIFIER (IDENTIFICATEUR D'OBJET) qui est acheminée (le cas échéant) en tant que valeur de l'élément d'information nom du contexte d'application dans la portion de dialogue.

La spécification officielle des contextes d'application appelle un complément d'étude.

NOTE – Les primitives qui assurent une interface de service standard pour l'accès à une AE en tant qu'objet composé seront étudiées ultérieurement.

4.4 Comment spécifier un ASE

La spécification d'un ASE utilisateur du TC doit fournir au moins:

– une description générale de l'ASE et de ses procédures;

– la liste des opérations assurées ainsi que les règles sur la séquence dans laquelle les opérations peuvent être lancées et l'extrémité (ou les deux) qui peut lancer les opérations en précisant leur nature;

– la description précise des procédures;

– les modalités d'interfonctionnement des différentes versions de protocole;

– la description des interactions entre l'ASE et le TC du point de vue des primitives du service du TC;

– les diagrammes SDL.

La spécification officielle des éléments du service d'application appelle un complément d'étude.

NOTE – Les primitives qui fournissent une interface de service standard pour l'accès des ASE au sein des AE seront étudiées ultérieurement.

4.5 Comment spécifier les opérations et les erreurs

4.5.1 Considérations générales

L'ensemble des opérations et des erreurs qui constitue une spécification du protocole d'utilisateur du sous-système d'application pour la gestion des transactions (TCAP) peut être décrit à l'aide d'un ou de plusieurs module(s) ASN.1. Le nombre de modules ASN.1 à utiliser est laissé à l'appréciation du concepteur de protocole et cette question est examinée plus en détail en 4.5.5.

La notation permettant de définir les opérations et les erreurs est basée sur le service MACRO ASN.1 défini dans la Recommandation X.208. OPERATION MACRO et ERROR MACRO sont les types de données associés respectivement à une opération et à une erreur.

Chaque opération (erreur) appartient à un type d'opération (type d'erreur) qui est dérivé du type OPERATION MACRO (ERROR MACRO).

Il convient d'attribuer un nom (une référence de type ASN.1 qui commence par une lettre majuscule) à chaque type d'opération ou type d'erreur.

Il convient d'attribuer un nom (une référence de valeur ASN.1 qui commence par une lettre minuscule) à chaque opération ou erreur.

La Recommandation X.219 précise que les valeurs d'un ensemble d'opérations et d'erreurs doivent être uniques dans une syntaxe abstraite. Dans le cas du sous-système TCAP, cela veut dire pour le moment qu'elles doivent être uniques dans le cadre d'un numéro de sous-systèmes ou d'un groupe de numéros de sous-systèmes connexes ou d'un contexte d'application.

La définition du type peut être associée à l'attribution d'une valeur ou effectuée en deux étapes, comme le montre l'exemple suivant:

– La spécification du type et l'attribution d'une valeur sont distinctes:

```
OperationTypeExample1 ::= OPERATION
ARGUMENT            ParameterType1
RESULT              ResultType1
ERRORS              { error1, error2 }

operationExample1 OperationTypeExample1 ::= localValue 1
```

– La spécification du type et l'attribution d'une valeur sont combinées:

```
operationExample1 ::= OPERATION
ARGUMENT            ParameterType1
RESULT              ResultType1
ERRORS              { error1, error2 }
::= localValue 1
```

On trouvera en 4.5.6 les raisons pour lesquelles il est plus pratique de combiner les spécifications selon le type et selon la valeur, ainsi que des détails sur l'utilisation de valeur globale ou locale.

4.5.2 Utilisation de la notation OPERATION MACRO

4.5.2.1 Utilisation de la notation de type

Un type d'opération est entièrement défini comme un exemple de type OPERATION MACRO complété par un commentaire ASN.1 indiquant la valeur de la temporisation associée.

Les paragraphes ci-après donnent des directives sur l'utilisation des diverses productions ASN.1 qui constituent la description OPERATION MACRO.

4.5.2.1.1 Spécification de l'argument d'opération

La production ASN.1 ci-après montre comment l'argument d'une opération doit être spécifié:

```
Parameter ::= ArgKeyword NamedType | empty
ArgKeyword ::= "PARAMETER" | "ARGUMENT"
```

Si des informations doivent être fournies au moment du lancement de l'opération, l'un des mots clés PARAMETER ou ARGUMENT doit être inséré et suivi par le type nommé qui correspond à la structure de données à fournir, sinon le mot clé ne sera pas présent dans la description d'opération.

Les deux mots clés sont autorisés aux fins de la compatibilité amont, les spécifications relatives à l'utilisateur du TC reposant sur des versions anciennes de TC. L'utilisation du mot clé PARAMETER est toutefois déconseillée pour définir des applications nouvelles.

4.5.2.1.2 Spécification de résultats positifs

Les productions ASN.1 ci-après montrent comment spécifier des opérations dont les résultats sont positifs.

```
Result ::= "RESULT" ResultType | empty

ResultType ::= NamedType | empty
```

Si l'information doit être renvoyée en tant que résultat de l'exécution d'une opération réussie, le mot clé RESULT doit être suivi du type nommé associé à la structure de données à émettre. Si aucune information ne doit être fournie mais que la classe d'opération indique que l'opération a réussi, le mot clé RESULT doit être présent dans la description d'opération mais l'alternative vide de la production résultat est utilisée. Si le mot clé RESULT n'est pas inclus dans une description d'opération, cela indique qu'il n'y a pas de notification de réussite (c'est-à-dire opération de la classe 2 ou 4).

4.5.2.1.3 Erreurs associées

Les productions ASN.1 ci-après montrent comment spécifier des opérations qui notifient un échec:

```
Errors ::= "ERRORS" "{" ErrorNames "}" | empty
ErrorNames ::= ErrorList | empty
ErrorList ::= Error | ErrorList "," Error
Error ::= value (ERROR) | type
```

Si l'opération notifie un échec, le mot clé ERRORS doit être inclus et suivi de la liste des erreurs associées, sinon le mot clé ne doit pas être présent. Les erreurs incluses dans la liste peuvent être indiquées soit à l'aide d'une référence de type ou d'une référence de valeur (c'est-à-dire un code d'erreur).

4.5.2.1.4 Spécification d'opérations corrélées

Les productions ASN.1 ci-après montrent comment spécifier des opérations corrélées:

```
LinkedOperations ::= "LINKED" "{" LinkedOperationNames "}" | empty
LinkedOperationNames ::= OperationList | empty
OperationList ::= Operation | OperationList "," Operation
Operation ::= value (OPERATION) | type
```

Si l'opération est l'opération mère d'un ensemble d'opérations corrélées, le mot clé LINKED doit être inclus suivi de la liste des opérations filles. Les opérations filles figurant sur la liste peuvent être indiquées soit à l'aide d'une référence de type, soit d'une référence de valeur (c'est-à-dire un code d'opération).

4.5.2.2 Utilisation de la notation de valeur

La notation de valeur pour une opération est soit la notation de la valeur d'un élément de type INTEGER ou la notation d'un élément de type OBJECT IDENTIFIER. Cela dépend si l'on a attribué à l'opération une valeur locale ou une valeur globale.

4.5.2.3 Spécification de temporisateurs

La valeur du temporisateur associé à un type d'opération doit être indiquée en tant que commentaire ASN.1 par rapport à la description du type d'opération MACRO ASN.1.

4.5.3 Utilisation de la notation ERROR MACRO

La notation du type pour une erreur est le mot clé ERROR suivi à titre facultatif du mot clé PARAMETER et du type nommé associé à l'information qui peut être envoyée en tant que paramètre d'erreur. Le mot clé PARAMETER ne doit pas être présent si aucune information n'est liée à la condition d'erreur.

La notation de la valeur pour une erreur est soit la notation de la valeur d'un élément du type INTEGER, soit la notation d'un élément du type OBJECT IDENTIFIER. Cela dépend si l'on a attribué à l'opération une valeur locale ou globale.

4.5.4 Exemples de descriptions d'opérations et d'erreurs

Le présent paragraphe explique une spécification de protocole qui traite des définitions des opérations et des erreurs associées pour un simple ASE utilisateur du TC. L'objet des opérations et des erreurs est brièvement décrit sous forme textuelle. Ensuite, les opérations et les erreurs, ainsi que les types de données connexes, sont officiellement décrits dans un module ASN.1.

L'exemple ci-après est fondé sur un dialogue du type libre appel fictif, entre un centre de commutation et une base de données libre appel.

4.5.4.1 Objet des opérations et des erreurs

4.5.4.1.1 Fournir des informations d'acheminement

Cette opération est lancée par un centre de commutation pour demander à une entité éloignée de fournir des informations d'acheminement afin d'établir un rappel pour un abonné. Les informations d'acheminement fournies peuvent être un numéro de destination et peuvent dépendre du numéro du demandeur et/ou du service de base requis. Dans le dernier cas, l'opération fille, obtenir le numéro du demandeur est lancée.

4.5.4.1.2 Obtenir le numéro du demandeur

Cette opération est lancée par un élément de réseau afin de demander à un centre de commutation de fournir le numéro du demandeur lié à l'établissement d'une communication.

4.5.4.1.3 Numéro appelé non valable

Cette erreur est renvoyée par un élément du réseau pour indiquer que le numéro appelé reçu n'est pas conforme au plan de numérotage utilisé.

4.5.4.1.4 Abonné non atteignable

Cette erreur est renvoyée par un élément de réseau pour indiquer qu'il y a actuellement aucune information d'acheminement disponible correspondant au numéro appelé.

4.5.4.1.5 Appel interdit

Cette erreur est renvoyée par un élément du réseau pour indiquer que l'appel ne peut pas être établi car le numéro du demandeur n'est pas compatible avec les conditions d'interdiction liées au numéro appelé.

4.5.4.1.6 Numéro du demandeur non disponible

Cette erreur est renvoyée par un centre de commutation pour indiquer que le numéro du demandeur ne peut pas être fourni.

4.5.4.1.7 Erreur de traitement

Cette erreur est renvoyée par un élément du réseau pour signaler une erreur de traitement.

4.5.4.2 Spécification ASN.1

Le module ASN.1 ci-après spécifie les opérations et les erreurs connexes ainsi que les types de données qui correspondent aux éléments de protocole décrits ci-dessus. Dans cet exemple, les opérations et les erreurs sont définies par une valeur utilisant une production «assignation de valeur» ASN.1.

```
TCAP-Examples { ccitt recommendation q 775 modules(2) examples(1) version1(1) }
DEFINITIONS ::=
BEGIN

IMPORTS OPERATION, ERROR
FROM TCAPMessages { ccitt recommendation q 773 modules(2) messages(1) version2(2) };

provideRoutingInformation                OPERATION
ARGUMENT                                 RequestArgument

RESULT                                   RoutingInformation

ERRORS                                   { invalidCalledNumber,
                                         subscriberNotReachable,
                                         callBarred,
                                         processingFailure }

LINKED                                   { getCallingPartyAddress }
-- temporisateur T-pi = 10 s
::= localValue 1

getCallingPartyAddress                   OPERATION
RESULT                                   CallingPartyAddress

ERRORS                                   { callingPartyAddressNotAvailable,
                                         processingFailure }
-- temporisateur T-gp = 5 s
::= localValue 2

invalidCalledNumber ERROR ::= localValue 1
subscriberNotReachable ERROR ::= localValue 2
calledBarred ERROR ::= localValue 3
callingPartyAddressNotAvailable ERROR ::= localValue 4
processingFailure ERROR ::= localValue 5
-- types de données

RequestArgument  ::= SEQUENCE {
calledNumber      IsdnNumber,
basicService      BasicServiceIndicator OPTIONAL
}

RoutingInformation ::= CHOICE {
  reroutingNumber       [0] IMPLICIT IsdnNumber,
  forwardedToNumber     [1] IMPLICIT IsdnNumber }

BasicServiceIndicator  ::= ENUMERATED {
speech (0),
unrestrictedDigital (1) }
```

```
CallingPartyAddress ::= IsdnNumber

IsdnNumber  ::= SEQUENCE {
typeOfAddress     TypeOfAddress,
digits            TelephonyString }

TypeOfAddress ::= ENUMERATED {
national (0),
international (1),
private (2) }

TelephonyString ::= IA5String   (FROM ("0"|"1"|"2"|"3"|"4"|"5"
|"6"|"7"|"8"|"9"|"*"|"#")) (SIZE (1..15))

END
```

4.5.5 Utilisation de modules ASN.1

Un module est une construction ASN.1 dans laquelle un concepteur de protocole réunit plusieurs définitions de types et de valeurs.

Théoriquement, l'ASN.1 n'impose aucune limitation du nombre de modules utilisés pour définir un protocole. Toutes les définitions peuvent être contenues dans un ou plusieurs module(s). Toutefois, si les définitions contenues dans un module sont nécessaires dans un autre (par exemple, l'erreur utilisée pour une opération est définie dans un autre module), la définition correspondante devient alors disponible en l'EXPORTant du module dans lequel elle est définie et en l'IMPORTant dans le module dans lequel elle est utilisée. Cela s'applique à tous les objets ASN.1, qu'ils soient définis selon le type ou la valeur.

Cette possibilité donne au concepteur d'application toute latitude de structurer les modules conformément aux besoins ou à des règles propres. Par exemple, un seul modèle peut contenir toutes les définitions, en particulier dans un environnement AE, ou ASE uniques. En revanche, il peut y avoir un seul module pour chaque définition d'un élément ASE, chaque module contenant toutes les opérations et les erreurs utilisées exclusivement par cet élément. A l'autre extrémité, toutes les opérations et erreurs peuvent être définies dans un module du registre central et exportées pour être utilisées dans les autres modules dans lesquels les éléments ASE sont définis.

4.5.6 Attribution et gestion des codes d'opération et d'erreur

4.5.6.1 Considérations générales

Les paragraphes 4.1 à 4.5.4 décrivent comment les entités d'applications (AE), les éléments de service d'application (ASE), les opérations et les erreurs peuvent être spécifiés. On examine également comment les éléments ASE utilisent les opérations et les erreurs, et comment les ASE proprement dits sont utilisés pour définir les contextes d'application entre deux entités d'application homologues.

Les AE et les ASE sont des outils de modélisation et de spécification commodes qui servent à élaborer des protocoles d'application. A la fin, pendant un dialogue entre deux utilisateurs de TC, tout protocole d'application reposant sur un TCAP est constitué par l'échange de valeurs de données pour les types d'opérations et d'erreurs identifiés respectivement par leur code d'opération ou d'erreur. La seule condition imposée dans le système de signalisation n° 7 (et ROSE) est la suivante: les codes d'opération et d'erreur doivent être uniques dans une syntaxe abstraite. Comme il n'existe pas actuellement de moyen de signaler explicitement la syntaxe abstraite à laquelle un code d'opération ou d'erreur appartient, le concepteur des applications doit veiller à ce que ces codes soient uniques dans le cadre d'un numéro de sous-système ou d'un contexte d'application. Dans le cas où le domaine des codes d'opération et d'erreur est un contexte d'application, il faut que le nom du contexte d'application soit acheminé à l'extrémité distante en utilisant le protocole de la portion de dialogue.

Les mécanismes d'attribution et de gestion des codes d'opération et d'erreur possibles sont nombreux, de même que les facteurs dont il faut tenir compte. La structure de l'AE/ASE et la réutilisation des opérations et des erreurs constituent deux éléments importants.

4.5.6.2 Importation et exportation d'opérations et d'erreurs

Comme n'importe quel autre type ASN.1, les opérations et les erreurs peuvent être exportées et importées entre modules ASN.1. Il est possible d'utiliser cette méthode lorsqu'il est nécessaire de définir une opération dont le type correspond à une opération existante, lorsque la valeur à attribuer à cette nouvelle opération est différente de celle qui est attribuée à l'opération existante (c'est-à-dire à des fins d'unicité). Cela est illustré par l'exemple ci-après, où l'identificateur d'objet 1 et l'identificateur d'objet 2 sont des identificateurs fictifs.

```
ExportingModule { objectIdentifer1 }  DEFINITIONS ::=
BEGIN
EXPORTS operation1, OperationTypeA, error1, ErrorTypeA;

IMPORTS OPERATION, ERROR FROM TCAPMessages
{ ccitt recommendation q 773 modules(2) messages(1) version2(2) };

operation1                  OPERATION
ARGUMENT                    ParameterType1
RESULT                      ResultType1
ERRORS                      { error1 }
::= localValue 1

OperationTypeA ::= OPERATION
ARGUMENT                    ParameterTypeA
RESULT                      ResultTypeA
ERRORS                      { ErrorTypeA }

operation2 OperationTypeA ::= localValue 2

error1  ERROR
PARAMETER DiagnosticType1
::= localValue 1

ErrorTypeA ::= ERROR
PARAMETER DiagnosticTypeA

error2 ErrorTypeA ::= localValue 2
-- Noter que ParameterType1, ResultType1, ParameterTypeA, ResultTypeA,
-- DiagnosticType1 et DiagnosticTypeA doivent être définis quelque part.
-- S'ils ne sont pas définis dans ce module, ils doivent être importés du module dans lequel ils sont définis.

END

ImportingModule { objectIdentifier2 }  DEFINITIONS ::=
BEGIN
IMPORTS OPERATION, ERROR FROM TCAPMessages
{ ccitt recommendation q 773 modules(2) messages(1) version2(2) };
operation1, OperationTypeA, error1, ErrorTypeA
FROM ExportingModule { objectIdentifier1 };

operation2 OPERATION
ARGUMENT ParameterTypeX -- doit être défini quelque part dans le module
::= localValue 2

error2 ERROR ::= localValue 2
-- Valeur 2 est déjà utilisée. Donc, valeur 3 est allouée aux objets importés.
operationA OperationTypeA ::= localValue 3
errorA ErrorTypeA ::= localValue 3

END
```

4.5.6.3 Incidence de la structure ASE/AE sur la gestion des codes d'opération et d'erreur

En ce qui concerne la structure des AE/ASE, les possibilités sont les suivantes:

Approche monolithique – Une entité d'application, un élément de service d'application

Au niveau du concept, c'est l'approche la plus simple. Le protocole d'application est défini par une entité d'application qui comprend seulement un élément ASE (en plus du TCAP). Toutes les opérations utilisées dans cet élément pourront être définies dans un module ASN.1, qui contient également la définition de l'ASE et de l'AE. A l'intérieur du protocole, toutes les opérations et les erreurs sont identifiées de façon univoque en recevant une valeur locale unique (nombre entier).

L'avantage de ce schéma est sa simplicité. Son inconvénient est qu'il ne permet pas d'identifier des blocs fonctionnels indépendants qui peuvent évoluer séparément dans la structure AE.

Une entité d'application comportant plusieurs éléments de service d'application (ASE)

Lorsqu'il définit un protocole d'application, le concepteur doit choisir de structurer l'entité d'application (et donc les AC) de façon qu'elle comporte deux ou plusieurs éléments ASE. Par exemple, on pourra décider de regrouper les éléments de protocole liés à l'authentification de l'usager dans un élément ASE distinct (qui pourrait être réutilisé dans un autre

protocole), et ceux qui sont associés à l'interrogation de la base de données dans un autre élément ASE. Cela peut faciliter la conception d'ensemble des systèmes, mais, lorsque tous les ASE constituants sont regroupés pour former l'entité d'application, il faudra s'assurer que l'on n'a pas attribué la même valeur aux différentes opérations/erreurs contenues dans les différents éléments ASE.

Utiliser la même opération/erreur dans deux éléments ASE différents à l'intérieur de la même entité d'application ne pose pas de problème. Si les valeurs attribuées à cette même opération sont les mêmes dans chaque cas, il y aura dans le protocole une seule opération/erreur associée à cette valeur. Si différentes valeurs sont attribuées, il apparaîtra dans le protocole qu'il y a deux opérations/erreurs différentes mais au niveau de la mise en œuvre de l'application, ces deux valeurs différentes provoqueront le lancement/identification de la même opération/erreur.

Toutefois, si à l'intérieur de cette même entité d'application, une opération définie dans un élément ASE reçoit la même valeur qu'une opération différente dans un autre élément ASE, cela posera de toute évidence un problème. Lorsqu'un élément ASE est utilisé dans une seule entité d'application, un schéma d'attribution de code simple permet de surmonter ce problème. Mais, lorsque le même élément ASE est utilisé dans plusieurs entités d'application, cette situation peut devenir difficile à gérer, et les seules approches sûres sont celles qui sont décrites sous i) à iii) ci-après.

i) Deux ou plusieurs protocoles partagent les mêmes valeurs d'opération/erreur locales.

   Lorsque les éléments ASE sont définis, les valeurs sont attribuées par le concepteur de protocole de manière à éviter toute collision. Une coordination s'impose pour définir les ASE. En d'autres termes, les ASE partagent la même syntaxe abstraite.

   Ce système présente un inconvénient dans le cas où l'un des ASE est utilisé dans plusieurs contextes (c'est-à-dire avec un ensemble différent d'ASE), car il est alors plus ou moins impossible d'éviter des problèmes de compatibilité de valeurs pour une combinaison.

ii) Assignation de valeurs globales (identificateurs d'objet) pour des opérations et des erreurs.

   Etant donné qu'il y a un seul identificateur d'objet dans le système de signalisation n° 7, il n'y a aucun risque de collision de valeurs quand un élément ASE est combiné à un autre.

   L'inconvénient de ce schéma est que l'identificateur d'objet, s'il est codé, est plus long qu'un nombre entier simple.

iii) Partage des opérations/erreurs en assignant des types et non des valeurs lors de la définition des opérations/erreurs.

   On suppose ici que le type des opérations et des erreurs a été défini indépendamment de la valeur assignée.

   Quand un concepteur de protocole définit un contexte d'application, il rassemble tous les types d'opérations et d'erreurs utilisés par les ASE requis et attribue des valeurs adaptées de manière à éviter toute collision.

   On peut considérer que ce faisant, le concepteur de protocole définit un nouvel ensemble d'ASE isomorphes qui diffèrent simplement des ASE existants par les valeurs de leurs opérations et erreurs.

4.5.6.4 Réutilisation des opérations et des erreurs

Indépendamment du nombre d'ASE inclus dans un protocole, il est parfois utile d'inclure une opération ou une erreur au moment de définir un nouvel ASE.

L'opération ou l'erreur peut être réutilisée de la manière suivante:

L'opération est importée dans l'un des modules définissant l'un des ASE. Cette importation n'est pas possible lorsque les valeurs sont différentes.

Ces blocs peuvent être identifiés si:

i) Il existe un registre central des opérations et des erreurs dont les valeurs sont prises dans une fourchette réservée qui n'est jamais utilisée par des opérations spécifiques d'ASE. Cette solution impose une contrainte aux ASE utilisateurs de TC qui risque de subsister dans un environnement plus vaste (c'est-à-dire si les opérations ou les erreurs des protocoles DSS1 ou ISO doivent être utilisées).

ii) Des valeurs globales ont été attribuées aux opérations et aux erreurs. Cette approche présente un inconvénient dans la mesure où une valeur globale nécessite davantage d'octets pour être codée qu'une valeur locale et doit être officiellement enregistrée dans un arbre d'identificateur d'objet.

iii) Le type d'opération ou d'erreur est importé dans l'un des modules définissant l'un des éléments ASE dans lesquels une valeur adaptée est attribuée. Cela suppose que le protocole d'exportation utilise la méthode à deux étapes pour définir les opérations et les erreurs ou que les types d'opération et d'erreur requis figurent dans un registre central.

iv) L'opération ou l'erreur est entièrement redéfinie bien qu'une partie de la définition d'origine (par exemple le type de l'argument) puisse être importée.

4.6 Spécifications de types de données

4.6.1 Considérations générales

Comme indiqué ci-dessus, le type d'information qui peut accompagner un lancement d'opération, à savoir la notification d'une réussite ou d'un échec, est spécifié en tant que type de données ASN.1. Cela vaut aussi pour l'information qui peut être échangée sous forme de données d'utilisateur de la portion de dialogue.

Ce type de données peut être du type intégré (par exemple, integer, boolean, null, octet, string, etc.) ou du type structuré (par exemple, sequence, sequence-of type, choice type, etc.). Il peut aussi être inspiré de ces types par un mécanisme de construction de sous-type (par exemple, contrainte de taille, gamme de valeurs) ou d'étiquetage.

4.6.2 Utilisation d'étiquettes

La notation ASN.1 fournit un mécanisme d'étiquetage qui permet de définir un type isomorphe à un type existant et qui ne diffère que par la présence de son étiquette.

Comme indiqué clairement dans la Recommandation X.208 (ASN.1), les étiquettes sont destinées à une utilisation en machine pour faciliter avant tout le processus de décodage.

Les étiquettes ne doivent pas être utilisées pour l'identification directe des éléments d'information, étant donné qu'elles sont vues du point de vue d'un processus d'application local. La manière dont ces éléments d'information sont identifiés localement est une question de mise en œuvre qui dépend de la conception du logiciel et du langage utilisé pour manipuler la représentation des données internes. A cet égard, il convient de noter que des étiquettes distinctes sont surtout nécessaires dans l'une des situations suivantes:

- les éléments d'information font partie d'un ensemble (non ordonné) (c'est-à-dire un type d'ensemble) et par conséquent leur position relative ne peut être utilisée pour différencier deux éléments d'information du même type (ayant donc la même étiquette);
- les éléments d'information font partie d'un ensemble ordonné (c'est-à-dire un type de séquence) mais la présence ou l'absence d'éléments facultatifs ne permet pas de faire une distinction entre la présence d'un élément facultatif et la présence d'un élément d'information du même type suivant immédiatement;
- le même type de base apparaît deux fois dans un type de choix.

Il y a quatre classes d'étiquettes. Outre la classe universal qui est utilisée pour identifier un type de base, trois classes sont définies et permettent la définition de types isomorphes à des fins de décodage.

- La *classe APPLICATION-WIDE* – Les étiquettes attribuées à cette classe peuvent servir à identifier directement la structure du type de données à décoder. Elles sont significatives dans une application et ne doivent pas être utilisées s'il y a risque d'incompatibilité entre les valeurs. La classe APPLICATION-WIDE ne devrait être utilisée que si l'application est un domaine «fermé» ou s'il existe un registre commun.
- La *classe CONTEXT-SPECIFIC* – Les étiquettes attribuées à cette classe ne sont significatives que dans un domaine défini. C'est pourquoi, le processus de décodage identifie la structure des données à décoder à la fois à partir de la valeur de l'étiquette et du contexte dans laquelle elle apparaît. On estime habituellement que le contexte est limité à la prochaine construction supérieure.
- La *classe PRIVATE* – Analogue à la classe APPLICATION-WIDE mais n'entre pas dans le cadre de la normalisation.

Il convient de noter que la classe CONTEXT-SPECIFIC est la seule (si elle est utilisée correctement) qui garantit qu'il n'y aura jamais d'incompatibilité entre les valeurs, lorsqu'il y a des importations/exportations d'unités de données entre modules ou protocoles.

4.6.3 Instances et types

Il est nécessaire de différencier clairement un type de données (c'est-à-dire les véritables éléments d'information acheminés dans un message ou une sous-structure). A des fins de spécification et de facilité de lecture, l'ASN.1 fournit une notation type nommé qui permet de qualifier une instance spécifique d'un type de données à l'aide d'un identificateur ASN.1.

Il convient de noter qu'il n'est pas nécessaire de définir un type de données par élément d'information. Lorsque deux éléments d'information sont équivalents du point de vue de la syntaxe, il est de toute évidence plus pratique de les représenter en tant que deux instances du même type de données, ou si cela est nécessaire aux fins de décodage, en tant qu'instances de deux types dérivés du même type de données grâce à l'étiquetage CONTEXT-SPECIFIC, et dont la définition apparaîtra donc seulement dans la définition de la construction supérieure (par exemple, les éléments étiquetés ne sont définis que dans le contexte spécifique de la construction supérieure).

4.6.4 Exportation et importation d'éléments d'information

Il se pourrait que des protocoles de signalisation fondés sur le sous-système TCAP aient à utiliser des éléments d'information définis dans d'autres spécifications de protocole de signalisation. Plutôt que de définir un nouvel élément d'information en partant de zéro, il serait préférable d'importer le type associé des spécifications dans lesquelles il a été défini pour la première fois. Des types de données peuvent être importés officieusement ou officiellement selon la façon dont le protocole d'exportation est spécifié.

- Le protocole d'exportation est spécifié à l'aide d'un module ASN.1, qui exporte les types de données requis: les types de données requis peuvent être importés officiellement dans l'un des modules qui définit le nouveau protocole.
- Le protocole d'exportation n'est pas spécifié à l'aide d'un module ASN.1: un moyen pratique est de définir un nouveau type de données isomorphe au type de chaîne d'octet et de spécifier officieusement une référence à la spécification dans laquelle la structure interne est définie (c'est-à-dire en utilisant un énoncé de commentaire).

4.7 Comment spécifier les syntaxes abstraites

Les spécifications relatives à l'ASE et à l'AC impliquent qu'il soit fait référence à une ou plusieurs syntaxes abstraites. Chaque syntaxe représente, au niveau de l'abstraction (c'est-à-dire indépendamment des techniques de codage) des ensembles de valeurs de données qui peuvent être échangées pendant la communication.

Il est actuellement inutile de donner explicitement un nom à la syntaxe abstraite formée par les messages de TC pour une application donnée car cette syntaxe abstraite est implicitement identifiée par le numéro de sous-système qui traite l'AE. Toutefois, la structure de l'information d'utilisateur acheminée dans la portion de dialogue doit être définie dans le cadre d'une ou de plusieurs autres syntaxes abstraites.

En conséquence, un concepteur de protocole qui souhaite avoir des informations d'utilisateur qui ne constituent pas des composants devant être acheminés par le TC doit commencer par définir une ou plusieurs syntaxes abstraites comprenant tous les types de données dont les valeurs peuvent être acheminées.

Il doit aussi donner un nom à chacune de ces syntaxes abstraites. Ce nom qui est une valeur du type OBJECT IDENTIFIER (IDENTIFICATEUR D'OBJET) servira de référence directe lorsque la valeur réelle sera acheminée dans le cadre d'une valeur construite du type EXTERNAL, comme il est spécifié dans la Recommandation Q.773.

Il n'existe pas actuellement de méthode officielle pour spécifier une syntaxe abstraite; toutefois lorsqu'une syntaxe de ce type peut être décrite en utilisant la notation ASN.1, la méthode la plus simple consiste à définir un type de choix élaboré à partir de tous les types de données qui forment la syntaxe abstraite.

Une syntaxe abstraite peut alors être définie officiellement par la phrase ci-après qui doit figurer dans les spécifications de protocole:

«L'ensemble des valeurs de données du type module X. Type A forme une syntaxe abstraite qui est identifiée par le nom de syntaxe abstraite suivant: <objectIdentifierValue>».

Dans la phrase précédente, le type A correspond au type de choix et le module X au module où il est défini.

Dans ce contexte, le nom de la syntaxe abstraite se réfère aussi implicitement aux règles de codage qu'il convient d'appliquer à la syntaxe abstraite. Ces règles de codage qui peuvent correspondre (mais pas nécessairement) à celles définies dans la Recommandation X.209, doivent être acceptées «*a priori*» par les utilisateurs de TC.

L'exemple ci-après illustre un module qui définit InitData comme un ensemble de trois unités de données de protocole formant une syntaxe abstraite utilisée lors de l'établissement du dialogue pour transférer soit la liste des unités fonctionnelles prises en charge, soit l'information d'authentification et qui définira les types d'ASN.1 nécessaires:

```
InitModule DEFINITIONS ::=
BEGIN

InitData ::= CHOICE {
functionalUnits [0] IMPLICIT FunctionalUnits,
authenticationInfo [1] IMPLICIT AuthenticationInfo }

FunctionalUnits ::= SEQUENCE OF FunctionalUnit
FunctionalUnit ::= ENUMERATED {
unit(1), unit2(2), unit3(3) }

AuthenticationInfo ::= SEQUENCE {
algorithm OBJECT IDENTIFIER,
signature OCTET STRING }
-- L'ensemble des valeurs de données du type InitModule-InitData forme
-- une syntaxe abstraite qui est identifiée par le nom de syntaxe
-- abstraite suivant: "<objectIdentifierValue>"

END
```

4.8 Règles de codage

La syntaxe concrète des messages du sous-système TCAP (c'est-à-dire le train de bits échangés entre sous-systèmes TCAP homologues en tant que données d'usager des messages SCCP) est établie en appliquant les règles de codage de base à la description de la syntaxe abstraite des messages du sous-système TCAP [y compris les éléments d'usager du TCAP à l'exception de ceux acheminés en tant que valeurs du type EXTERNAL (par exemple le champ d'information d'utilisateur d'une APDU de gestion de dialogue)]. Les règles de codage de base sont décrites dans la Recommandation X.209, certaines restrictions mineures sont signalées dans la Recommandation Q.773 en ce qui concerne le codage de la partie TCAP.

L'information d'utilisateur acheminée en tant que valeur du type EXTERNAL peut aussi (mais pas nécessairement) être codée conformément aux règles de codage de base. Dans ce cas, le nom de la syntaxe abstraite associé sert aussi de référence implicite aux règles de codage appliquées (voir 3.3.3).

Il faut noter que les règles de codage de base permettent plusieurs options, spécialement pour le codage des longueurs. Cela veut dire qu'une mise en œuvre doit être à même de décoder une unité de données indépendamment des options de codage choisies par l'entité expéditrice.

Imprimé en Suisse

Genève, 1994