UNION INTERNATIONALE DES TÉLÉCOMMUNICATIONS

UIT-T T.176

SECTEUR DE LA NORMALISATION
DES TÉLÉCOMMUNICATIONS
DE L'UIT

(02/98)

SÉRIE T: TERMINAUX DES SERVICES TÉLÉMATIQUES

Interface de programmation d'application pour la commande et le contrôle de support de stockage numérique

Recommandation UIT-T T.176

(Antérieurement Recommandation du CCITT)

RECOMMANDATIONS UIT-T DE LA SÉRIE T

TERMINAUX DES SERVICES TÉLÉMATIQUES

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

RECOMMANDATION UIT-T T.176

INTERFACE DE PROGRAMMATION D'APPLICATION POUR LA COMMANDE ET LE CONTROLE DE SUPPORT DE STOCKAGE NUMERIQUE

Résumé

La présente Recommandation donne la spécification de l'interface de programmation d'application (API, *application programming interface*) pour la commande et le contrôle de support de stockage numérique (DSM-CC, *digital storage media command and control*) à utiliser dans les applications multimédia de base.

Source

La Recommandation UIT-T T.176, élaborée par la Commission d'études 16 (1997-2000) de l'UIT-T, a été approuvée le 6 février 1998 selon la procédure définie dans la Résolution n° 1 de la CMNT.

AVANT-PROPOS

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

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

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

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

NOTE

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

DROITS DE PROPRIÉTÉ INTELLECTUELLE

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

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

TABLE DES MATIÈRES

Recommandation T.176

INTERFACE DE PROGRAMMATION D'APPLICATION POUR LA COMMANDE ET LE CONTROLE DE SUPPORT DE STOCKAGE NUMERIQUE

(Genève, 1998)

1 Domaine d'application

La présente Recommandation donne la spécification de l'interface de programmation d'application (API, *application programming interface*) pour la commande et le contrôle de média de stockage numérique (DSM-CC, *digital storage media command and control*) à utiliser dans les applications multimédia de base. La présente Recommandation s'applique aux systèmes DAVIC conformes.

2 Références normatives

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

[1] ETS 300 777-3, *Terminal equipment (TE); End-to-end protocols for multimedia information retrieval services; Part 3: Application Programmable Interface (API) for MHEG-5*.

[2] ISO/CEI DIS 13818-6, *Technologies de l'information – Codage générique des images animées et des informations sonores associées – Partie 6: Extensions pour DSM-CC*.

[3] ISO/CEI 13522-5: 1997, *Technologies de l'information – Codage de l'information multimédia et hypermédia – Partie 5: Support pour applications interactives de niveau fondamental*.

[4] ISO/CEI DIS 13522-6, *Technologies de l'information – Codage de l'information multimédia et hypermédia – Partie 6: Support pour les applications interactives améliorées*.

[5] ETS 300 777-1, *Terminal Equipment (TE); End-to-end protocols for multimedia information retrieval services; Part 1: Coding of multimedia and hypermedia information for basic multimedia applications (MHEG-5).*

3 Définitions et abréviations

3.1 Définitions

Dans la présente Recommandation, on utilise les définitions données dans l'ISO/CEI DIS 13818-6 [2].

La présente Recommandation définit les termes suivants:

3.1.1 interface de programmation d'application (API, *application programmable interface*): frontière à travers laquelle une application logicielle utilise des éléments de langages de programmation pour invoquer des services logiciels. Ces éléments peuvent comprendre des procédures ou des opérations, des objets de données partagés et la résolution d'identificateurs.

3.1.2 application locale: sous-ensemble logiciel qui fait partie de l'application (de télécommunication) et qui tourne sur l'équipement considéré.

3.2 Abréviations

La présente Recommandation utilise les abréviations suivantes:

API	interface de programmation d'application (*application programming interface*)
ASN.1	notation de syntaxe abstraite numéro un (*abstract syntax notation one*)
DAVIC	consortium DAVIC (*digital audio visual council*)
DSM-CC	commande et contrôle de média de stockage numérique (*digital storage media command and control*)
MHEG	groupe d'experts pour le codage de l'information multimédia et hypermédia (*multimedia and hypermedia information coding experts group*)
SI	information de service (*service information*)
STU	dispositif d'adaptation multimédia (*set top unit*)
VM	machine virtuelle (*virtual machine*)

4 Aperçu général

Le paragraphe qui suit permet de positionner l'interface API définie dans la présente Recommandation dans le cadre général des spécifications DAVIC.

4.1 Format d'échange d'application DAVIC

Pour distribuer des informations multimédia à des dispositifs STU dans un environnement d'interfonctionnement, les applications doivent utiliser le format d'échange MHEG-5 de type forme finale, tel qu'il est défini dans l'ISO/CEI 13522-5 [3]. Le codage et la notation ASN.1, tels qu'ils sont définis dans la Norme ETS 300 777-1 [4], doivent être utilisés pour l'échange d'objets MHEG-5. Ce format donne la définition de la sémantique et du codage des objets multimédia et hypermédia.

Pour distribuer un code de programme à des dispositifs STU dans un environnement d'interfonctionnement, les applications doivent utiliser la classe MHEG-5 `InterchangedProgram` (programme échangé) pour encapsuler le code de machine virtuelle Java[1], conformément à la sémantique et au codage définis dans l'ISO/CEI DIS 13522-6 [5]. Les classes de machine virtuelle Java sont appelées à partir d'objets MHEG-5 au moyen des actions élémentaires MHEG-5 `Call` et `Fork`.

L'unité d'échange de code de machine virtuelle Java est une classe de machine virtuelle Java. De telles classes doivent être codées conformément à la définition donnée dans la section relative au format de la classe `File` (*class file format*) de la spécification de la machine virtuelle Java (*Java virtual machine specification*). Une classe Java encapsule des données et des méthodes qui sont composées de séquences d'instructions. L'ensemble des instructions est défini dans la section relative à l'ensemble des instructions de machine virtuelle Java (*Java virtual machine instruction set*) de la spécification de cette machine (*Java virtual machine specification*).

---

[1] Java est une marque ou une marque déposée de Sun Microsystems, Inc.

4.2 Ensemble essentiel d'interfaces API Java

L'ensemble d'interfaces API correspondant aux modules énumérés ci-après est utilisé par le code de machine virtuelle Java dans les spécifications DAVIC 1.1 [1] afin d'exprimer l'accès aux fonctions de bases du dispositif STU dans un environnement d'interfonctionnement:

– le module `java.lang`;

– le module `java.util`;

– le module `iso.mheg5`;

– le module `davic.dsmccuu`;

– le module `etsi.si`.

NOTE 1 – La spécification de la machine virtuelle Java fournit des mécanismes souples permettant d'appeler des fonctions externes dont l'interface est définie comme étant un module Java. La spécification DAVIC 1.1 ne comprend qu'un ensemble essentiel minimal de modules nécessaires pour que le code de machine virtuelle Java soit utile dans un environnement DAVIC. D'autres modules Java devraient être normalisés ultérieurement.

NOTE 2 – Bien qu'à proprement parler il ne soit pas nécessaire pour la qualité de fonctionnement voulue de l'environnement de machine virtuelle, le module `java.io` fait partie des classes Java fondamentales. Il est prévu d'ajouter le module `java.io` à l'ensemble essentiel DAVIC d'interfaces API Java conjointement avec une spécification appropriée de sa sémantique dans un environnement DAVIC.

Le module `java.lang`, tel qu'il est défini dans le document relatif aux interfaces API Java (*Java API documentation*), est constitué de l'ensemble minimal de classes de machine virtuelle Java nécessaires pour faire tourner le code de machine virtuelle Java, ces classes prenant en charge les fonctionnalités suivantes: types de données de base, objet, opérations mathématiques, sécurité, gestion des chemins, manipulation des chaînes, traitement des anomalies.

Le module `java.util`, tel qu'il est défini dans le document relatif aux interfaces API Java (*Java API documentation*), est constitué de classes de machine virtuelle Java prenant en charge un certain nombre d'utilitaires communs à tous les programmes de machine virtuelle Java.

Le module `iso.mheg5`, tel qu'il est défini dans la Norme ETS 300 777-3 [1], fournit le code de machine virtuelle Java avec accès aux objets de présentation et d'interaction multimédia MHEG-5 et avec manipulation de ces objets, c'est-à-dire avec accès aux attributs dynamiques des objets MHEG-5 et avec invocation d'actions élémentaires sur les objets MHEG-5.

Le module `davic.dsmccuu` et le module `davic.CosNaming` associé permettent au code de machine virtuelle Java d'utiliser les objets interface utilisateur-utilisateur DSM-CC pour l'accès aux données de réseau.

Le module `davic.dsmccuu` et le module `davic.CosNaming` associé permettent de donner accès à un sous-ensemble de l'interface API utilisateur-utilisateur DSM-CC définie dans l'ISO/CEI DIS 13818-6. Ce sous-ensemble comprend:

– les opérations `list` (listage) et `resolve` (résolution) associées à l'interface abstraite `NamingContext`;

– les opérations `close` (fermeture) et `destroy` (destruction) associées à l'interface abstraite `Base`;

– les opérations `next_one` (passage au suivant) et `next_n` (passage aux N suivants) associées à l'interface instanciable `BindingIterator`;

– les opérations `open` (ouverture) et `close` (fermeture) associées à l'interface instanciable `Directory`;

– les opérations `read` (lecture) et `write` (écriture) ainsi que l'attribut en lecture seule `ContentSize` (dimension de contenu) associés à l'interface `File`;

– les opérations `attach` (rattachement) et `detach` (détachement) associées à l'interface instanciable `Session`;

– l'interface instanciable `SessionGateway`.

Le module `etsi.si` permet au code de machine virtuelle Java d'accéder aux informations transmises dans le flux d'informations de service DAVIC.

5 Module davic.CosNaming

5.1 Classe davic.CosNaming.NameComponent

```
package davic.CosNaming;

public class NameComponent {
        public String id;
        public String kind;
}
```

5.2 Classe davic.CosNaming.Binding

```
package davic.CosNaming;

public class Binding {
        // constant declarations for the "binding_type" attribute
        public static final short nobject = 0;
        public static final short ncontext = 1;

        public NameComponent[] binding_name;
        public int binding_type;
}
```

5.3 Anomalie davic.CosNaming.NotFound

```
package davic.CosNaming;

public class NotFound extends Exception{
        // constant declarations for the "why" attribute
        public static final short missing_node = 0;
        public static final short not_context = 1;
        public static final short not_object = 2;

        public int why;
        public NameComponent[] rest_of_name;
}
```

5.4 Anomalie davic.CosNaming.CannotProceed

```
package davic.CosNaming;

public class CannotProceed extends Exception{
        public NamingContext cxt;
        public NameComponent[] rest_of_name;
}
```

5.5 Anomalie davic.CosNaming.InvalidName

```
package davic.CosNaming;

public class InvalidName extends Exception{
}
```

5.6 Classe davic.CosNaming.BindingIterator

```
package davic.CosNaming;

public class BindingIterator {
        public boolean next_one(
                Binding b
        )
        {
        // actual code shall be inserted here
        return true;
        }

        public void next_n(
                int how_many,
                Binding[] bl
        )
        {
        // actual code shall be inserted here
        }

        public void destroy()
        {
        // actual code shall be inserted here
        }
}
```

5.7 Interface davic.CosNaming.NamingContext

```
package davic.CosNaming;

public interface NamingContext {
        public void list(
                int how_many,
                Binding[] bl,
                BindingIterator bi
        );

        public Object resolve(
                NameComponent[] n
        )throws NotFound, CannotProceed, InvalidName;
}
```

6 Module davic.dsmccuu

6.1 Classe davic.dsmccuu.Step

```
package davic.dsmccuu;

import davic.CosNaming.*;
```

```
public class Step {
        public NameComponent name;
        public boolean process;
}
```

6.2 Anomalie davic.dsmccuu.SERVICE_XFR

```
package davic.dsmccuu;

import davic.CosNaming.*;

public class SERVICE_XFR extends Exception {
        // service location
        public byte[] serviceDomain;
        public NameComponent[] pathName;
        public byte[] initialContext;
}
```

6.3 Anomalie davic.dsmccuu.dsmccuuException

```
package davic.dsmccuu;

public class dsmccuuException extends Exception {
        public short minor;
        public short completed;
}
```

6.4 Anomalie davic.dsmccuu.INV_OFFSET

```
package davic.dsmccuu;

public class INV_OFFSET extends dsmccuuException {
}
```

6.5 Anomalie davic.dsmccuu.INV_SIZE

```
package davic.dsmccuu;

public class INV_SIZE extends dsmccuuException {
}
```

6.6 Anomalie davic.dsmccuu.READ_LOCKED

```
package davic.dsmccuu;

public class READ_LOCKED extends dsmccuuException {
}
```

6.7 Anomalie davic.dsmccuu.WRITE_LOCKED

```
package davic.dsmccuu;

public class WRITE_LOCKED extends dsmccuuException {
}
```

6.8 Anomalie davic.dsmccuu.OPEN_LIMIT

```
package davic.dsmccuu;

public class OPEN_LIMIT extends dsmccuuException {
}
```

6.9 Anomalie davic.dsmccuu.NO_AUTH

```
package davic.dsmccuu;

public class NO_AUTH extends dsmccuuException {
        public byte[] authData;
}
```

6.10 Anomalie davic.dsmccuu.UNK_USER

```
package davic.dsmccuu;

public class UNK_USER extends dsmccuuException {
}
```

6.11 Anomalie davic.dsmccuu.BAD_COMPAT_INFO

```
package davic.dsmccuu;

public class BAD_COMPAT_INFO extends dsmccuuException {
}
```

6.12 Anomalie davic.dsmccuu.NO_RESUME

```
package davic.dsmccuu;

public class NO_RESUME extends dsmccuuException {
}
```

6.13 Anomalie davic.dsmccuu.NO_SUSPEND

```
package davic.dsmccuu;

public class NO_SUSPEND extends dsmccuuException {
}
```

6.14 Interface davic.dsmccuu.Base

```
package davic.dsmccuu;

interface Base {
        public void close();

        public void destroy();
}
```

6.15 Classe davic.dsmccuu.File

```
package davic.dsmccuu;

public class File implements Base {
        // Base.close implementation
```

```
        public void close()
        {
        // actual code shall be inserted here
        }

        // Base.destroy implementation
        public void destroy()
        {
        // actual code shall be inserted here
        }

        public int[] getContentSize()
        {
        // actual code shall be inserted here
        return null;
        }

        public void read(
                int[] aOffset,
                int aSize,
                boolean aReliable,
                byte[] rData
        ) throws
                INV_OFFSET, INV_SIZE, READ_LOCKED
        {
        // actual code shall be inserted here
        }

        public void write(
                int[] aOffset,
                int aSize,
                byte[] rData
        ) throws
                INV_OFFSET, INV_SIZE, WRITE_LOCKED
        {
        // actual code shall be inserted here
        }
}
```

6.16 Classe davic.dsmccuu.Directory

```
package davic.dsmccuu;

import davic.CosNaming.*;

public class Directory implements NamingContext {
        // NamingContext.list implementation
        public void list(
                int how_many,
                Binding[] bl,
                BindingIterator bi
        )
        {
        // actual code shall be inserted here
        }

        // NamingContext.resolve implementation
        public Object resolve(
                NameComponent[] n
        )throws NotFound, CannotProceed, InvalidName
```

```
        {
        // actual code shall be inserted here
        return null;
        }

        public void open(
                char aPathType,
                Step[] rPathStep,
                Object[] resolvedRefs
        ) throws
                OPEN_LIMIT, NO_AUTH, UNK_USER, SERVICE_XFR,
                NotFound, CannotProceed, InvalidName
        {
        // actual code shall be inserted here
        }

        public void close()
        {
        // actual code shall be inserted here
        }
}
```

6.17 Interface davic.dsmccuu.SessionI

```
package davic.dsmccuu;

import davic.CosNaming.*;

interface SessionI {
        public void attach(
                byte[] serviceDomain,
                NameComponent[] pathName,
                byte[] userContext,
                Object[] resolvedRefs
        ) throws
                OPEN_LIMIT, NO_AUTH, UNK_USER, SERVICE_XFR, BAD_COMPAT_INFO, NO_RESUME,
                NotFound, CannotProceed, InvalidName;

        public void detach(
                boolean aSuspend,
                byte[] savedContext
        ) throws
                NO_SUSPEND;
}
```

6.18 Classe davic.dsmccuu.Session

```
package davic.dsmccuu;

public class Session implements SessionI {
        // SessionI.attach implementation
        public void attach(
                byte[] serviceDomain,
                NameComponent[] pathName,
                byte[] userContext,
                Object[] resolvedRefs
        ) throws
                OPEN_LIMIT, NO_AUTH, UNK_USER, SERVICE_XFR, BAD_COMPAT_INFO, NO_RESUME,
                NotFound, CannotProceed, InvalidName
        {
```

```
        // actual code shall be inserted here
        }

        // SessionI.detach implementation
        public void detach(
                boolean aSuspend,
                byte[] savedContext
        ) throws
                NO_SUSPEND
        {
        // actual code shall be inserted here
        }
}
```

6.19 Classe davic.dsmccuu.SessionGateway

```
package davic.dsmccuu;

public class SessionGateway extends Directory implements SessionI {
        // SessionI.attach implementation
        public void attach(
                byte[] serviceDomain,
                NameComponent[] pathName,
                byte[] userContext,
                Object[] resolvedRefs
        ) throws
                OPEN_LIMIT, NO_AUTH, UNK_USER, SERVICE_XFR, BAD_COMPAT_INFO, NO_RESUME,
                NotFound, CannotProceed, InvalidName
        {
        // actual code shall be inserted here
        }

        // SessionI.detach implementation
        public void detach(
                boolean aSuspend,
                byte[] savedContext
        ) throws
                NO_SUSPEND
        {
        // actual code shall be inserted here
        }
}
```

SERIES DES RECOMMANDATIONS UIT-T

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