TENeT 権利価値取引 API 仕様書

TEF950-S001-01.00.00/ja

2005 年 3 月
T-Engine フォーラム

TENeT 権利価値取引 API 仕様書

規定範囲 (Scope)

本文書は，TENeT 仕様 IC カードを用いて，電子的な権利価値の閲覧・作成などの操作や，売買・交換などの取引を行うための Java API 仕様である TENeT 権利価値取引 API を定義する．

参照規定 (Normative References)

e^{2}TP メッセージング API:

> e^{2}TP メッセージ仕様書, T-Engine Forum, 2005.
> Gosling, J., Joy, B., Steele, G., and Bracha, G.: The Java Language Specification, Sun Microsystems, 2000.

VTS API:

> TENeT メッセージ仕様書, T-Engine Forum, 2005.
> Gosling, J., Joy, B., Steele, G., and Bracha, G.: The Java Language Specification, Sun Microsystems, 2000.

はじめに

TENeT 権利価値取引 API (org.t_engine.tenet.vts) は，TENeT 仕様 IC カードを用いて，電子的な権利価値の操作や取引を行うための Java API である．

本 API が提供する機能のうち，代表的なものを以下に示す．

- IC カードに対する認証 (Participant#getAgent())
- フォルダの一覧取得 (Agent#getFolders())
- フォルダの生成・削除 (Agent#createFolder()/deleteFolder())
- (フォルダに格納された) 権利価値の一覧取得 (Folder#iterator())
- 権利価値の生成 (Folder#createVoucher())
- 権利価値の削除 (StoredVoucher#remove())
- 権利価値の譲渡・交換 (StoredVoucher#transfer()/exchange())
- 中断した取引の復旧 (Session#recover())

上記で示すように，TENeT 仕様 IC カードが提供する機能のほとんど (カード内容のバックアップ/復元など一部の運用機能を除く) は，本 API のみを用いて利用することができる．すなわち，本 API を用いることにより，アプリケーション開発者は e^{2}TP メッセージの送受を直接意識することなく， TENeT 仕様 IC カードが提供する各種機能を利用することができる．

本 API において，権利価値の取引 (譲渡・交換) は非同期に行われる．すなわち，StoredVoucher#transfer()/exchange() の呼出は，その結果を待つことなく直ちに return される．その後，取引の進行に伴って取引当事者の双方の Agent に登録されたリスナ

(see Agent#setListener()) が逐次呼び出される.

取引を実行するための手順を以下に示す. ここで, 取引の提案側 AP を A, 取引の受諾側 AP を B と記載する. また, A に登録されたリスナを lis_A, B に登録されたリスナを lis_B とする.

1. A が取引を開始する (StoredVoucher#exchange()/transfer())
2. lis_B に, 取引の提案が通知される (ReceptionListener#offered())
3. B が取引提案に合意する (Session#agree())
4. lis_A に, 提案への合意が通知される (ReceptionListener#agreed())
5. A が合意内容を承諾する (Session#confirm())
6. lis_A/lis_B の双方に取引の完了が通知される (ReceptionListener#committed())

上記は取引が正常に行われた場合について説明したが, 通信路の障害や取引相手が通知を無視することなどにより, 取引が途中で中断されることがある. この場合, Session インタフェースが提供する recover() を用いて, 取引を中止 (取引前の状態に戻る) もしくは強制完了させることができる (取引の復旧). 中止と完了のどちらになるかは, 取引の進行状況に依存する.

recover() による復旧処理は, (取引の開始と同様に) 非同期呼び出しにより行われるため, 復旧処理の結果の通知はリスナの呼び出しにより行われる. 取引が中止された場合は登録されたリスナの aborted() が, 取引が完了した場合は committed() がそれぞれ呼び出される (ReceptionListener#aborted()/committed()). いずれの場合でも, 取引の公平性は保証される. すなわち, 取引は正常に完了されるか開始前の状態に戻されるかのいずれかになり, 取引によって相手から受け取るべき権利価値を受け取ることなしに, 交換対象とした自らの権利価値を失うことはない.

復旧処理は, 調停サーバとの通信を伴う場合がある. 調停サーバとの通信なしに復旧できるか否かは, Session インタフェースが提供する isCancelable() を用いて判定することができる. この返値が真の場合, 調停サーバと通信することなしに復旧を行うことができる. このとき, 復旧結果は常に中止となる.

復旧に調停サーバとの通信が必要な場合 (isCancelable() の返値が偽の場合), recover() は調停サーバに対して復旧のための調停依頼を送信し, 調停サーバにより判断された (完了か中止かの) 調停結果を取得する. 復旧結果は調停結果に従う. ここで利用される調停サーバは, Agent#setArbiter() によって指定されたサーバである. 調停サーバとの通信および復旧処理は recover() の中で自動的に行われるため, AP が通信手順などを意識する必要はない.

以下に, 本 API の利用サンプルを示す.

```
// 1. 初期化

 ParticipantRepository repository = TENeTParticipantRepository.getInstance();
 // Internet Draft (draft-ietf-trade-voucher-vtsapi-06) 互換の,
 // ParticipantRepository を生成する.

// 2. IC カードへの認証

 Agent agent = repository.lookup(cardID).getAgent(myAuthHandler);
 // 権利価値操作のためのエントリポイントを取得する. ここで cardID
 // は IC カードの eTRON ID, myAuthHandler は AuthHandler インタフェー
 // スを実装するクラスのインスタンスとする.

// 3. フォルダの取得
```

```
 Folder folder = agent.getFolders().get(folderName);
 // フォルダの取得. folderName はフォルダ名とする.

// 4. 権利価値の生成

 folder.createVoucher(contents, 1, FileAcl.TransferAccess);
 // 譲渡可能であり, contents を内容とする権利価値を 1 つ作成する.

// 5. フォルダ内の権利価値一覧の取得

 Iterator i = folder.iterator();
 // Folder は Set を継承しているので, 保持している権利価値をコレク
 // ション関連のメソッドを用いて扱うことができる.

 while (i.hasNest()) {
   Voucher v = (Voucher)(i.next());
   System.out.println(v.getIssuer().toString() + " " + v.getCount());
 }

 // folder に格納された全ての権利価値の発行者と個数を出力する.

// 6. 譲渡 (権利価値の送付)

 Participant partner = repository.lookup(partnerID);
 // 譲渡の相手を特定する. partnerID は相手の eTRON ID とする.

 StoredVoucher v1 = (StoredVoucher)(folder.iterator().next());
 // 譲渡で相手に渡す権利価値 v1 を特定する. ここでは, folder に格納
 // されている最初の権利価値 (実際はこのような指定はせず, 権利価値
 // の一覧から利用者に選ばせるなどする).

 agent.setListener(listener);
 // 譲渡や交換が中断, 終了した際に呼び出されるリスナを登録する. こ
 // こで, listener は ReceptionListener インタフェースを実装するク
 // ラスのインスタンスとする.

 Session s = v1.select(3).transfer(partner);
 // 譲渡の実行. 格納されている v1 のうち 3 つを partner に送付する.
 // たとえば, v1 がもともと 10 個格納されていたら, 手元には 7 個の
 // v1 が残ることになる. 返値 s は譲渡セッションを特定するための ID
 // である. 譲渡が完了したら s を引数として listener.committed() が,
 // 譲渡が中止されたら listener.aborted() がそれぞれ呼び出される.
 // なお, なんらかの理由により譲渡が中途で進行不能となった場合は,
 // listener.suspended() が呼び出される.

// 7. 売買 (権利価値の交換)

 Session s = v1.select(3).exchange(partner, v2condition);
 // 交換の実行. 格納されている v1 のうち 3 つと, partner が保持する
 // v2 とを公平に交換する. ここで v2condition は交換で相手からもらう
 // 権利価値 v2 の条件を指定する. 交換が完了もしくは中止された時,
 // 譲渡の際と同様に listener.committed(), もしくは listener.aborted()
 // が呼び出される.

// 8. IC カードからのログアウト (非認証モーhへの移行)

 agent.changeAuthMode(Null, AuthHandler.None);
 // 非認証モードへ遷移する場合, AuthHandler を指定する必要はない.
```

org.t_engine.tenet.vts

Interface Agent

All Superinterfaces:
Participant

---

public interface **Agent**
extends Participant

TENeT仕様のICカードを操作するためのエントリポイントを提供するインタフェースである。
ICカードを表わすParticipantに対してParticipant#getAgent()メソッドを呼び出すことにより取得する。Agentには、権限モードがあり、所有者モードの場合にのみICカードの操作を行うことができる。
ICカードに格納された権利価値に対して交換および譲渡を行う場合、調停サーバの登録が必須である。また、ICカードに格納された権利価値に対して行った交換または譲渡の結果通知などは、本インタフェースで登録するReceptionListenerインタフェースのメソッドで適宜通知される。このため、権利価値の交換または譲渡を行う場合には、setListener()メソッドによりReceptionListenerの登録が必須である。

See Also:
Participant, Folder, ReceptionListener, AuthHandler

Methods

createFolder

```
public Folder createFolder(java.lang.String name,
                           int acl)
                    throws VTSInternalException,
                           VTSAccessViolationException,
                           VTSMemoryOverflowException,
                           VTSSmartcardNotFoundException,
                           VTSParameterException
```

生成するフォルダの名称nameと生成するフォルダのアクセス権aclを持つフォルダをICカード内に生成する。生成するフォルダのアクセス権aclの指定値については、FolderAclインタフェースが提供する値を使用すること。 複数のアクセス権を付与する場合は、FolderAclインタフェースの定数フィールド値を論理和で連結して指定することができる（例えば、権利価値作成権限と権利価値転送権限を付与する場合は、createFolder(name, FolderAcl.CreateAccess | FolderAcl.TransferAccess)などとすることができる)。

Parameters:
name - 生成するフォルダの名称
acl - 生成するフォルダのアクセス権

Returns:
生成されたフォルダ

Exceptions:
VTSInternalException - 内部的なエラーが発生した
VTSAccessViolationException - 処理に必要なアクセス権をAgentが得ていない
VTSMemoryOverflowException - ICカードの空き容量が不足している
VTSSmartcardNotFoundException - ICカードアクセスに対する応答がない(一般的にはICカードが存在しない場合に発生する)
VTSParameterException - 生成するフォルダの名称nameにnullが設定されたか、または同一名のフォルダが存在する、または生成するフォルダのアクセス権aclが正しくない

(continued on next page)

(continued from last page)

deleteFolder

```
public void deleteFolder(java.lang.String name,
                         boolean mode)
                  throws VTSInternalException,
                         VTSAccessViolationException,
                         VTSObjectNotFoundException,
                         VTSSmartcardNotFoundException,
                         VTSParameterException
```

削除対象フォルダの名称nameのフォルダを削除モードmodeの指定に従って、ICカード内より削除する。 削除モードmodeがtrueの場合、指定されたフォルダ内に権利価値が存在しても削除を行う。 削除モードmodeがfalseの場合に、指定されたフォルダ内に権利価値が存在したときはVTSAccessViolationExceptionをスローする。 また、削除モードmodeに関わらずフォルダ内に交換中の権利価値が存在する場合はVTSAccessViolationExceptionをスローする。

Parameters:

name - 削除対象フォルダの名称
mode - 削除モード

Exceptions:

VTSInternalException - 内部的なエラーが発生した
VTSAccessViolationException - 処理に必要なアクセス権をAgentが得ていない、または削除モードmodeがfalseの場合にフォルダ内に権利価値が存在する、またはフォルダ内に交換中の権利価値が存在する
VTSObjectNotFoundException - 操作対象のフォルダが存在しない
VTSSmartcardNotFoundException - ICカードアクセスに対する応答がない(一般的にはICカードが存在しない場合に発生する)
VTSParameterException - 削除対称フォルダの名称nameにnullが設定された

---

getFolders

```
public org.t_engine.tenet.util.Map getFolders()
                                  throws VTSInternalException,
                                         VTSAccessViolationException,
                                         VTSSmartcardNotFoundException
```

ICカードに格納されているフォルダの一覧をフォルダ名(String)をキーとしたフォルダ(Folder)のMapとして返却する。

Returns:

フォルダ名(String)をキーとしたフォルダ(Folder)のMap

Exceptions:

VTSInternalException - 内部的なエラーが発生した
VTSAccessViolationException - 処理に必要なアクセス権をAgentが得ていない
VTSSmartcardNotFoundException - ICカードアクセスに対する応答がない(一般的にはICカードが存在しない場合に発生する)

---

getSessions

```
public org.t_engine.tenet.util.Map getSessions()
                                   throws VTSInternalException,
                                          VTSAccessViolationException,
                                          VTSSmartcardNotFoundException,
                                          VTSUnsupportedMessageException
```

ICカードに格納されている交換および譲渡中のセッションの一覧をセッションID(String)をキーとしたセッション(Session)のMapとして返却する。

Returns:

セッションID(String)をキーとしたセッション(Session)のMap

Exceptions:

VTSInternalException - 内部的なエラーが発生した
VTSAccessViolationException - 処理に必要なアクセス権をAgentが得ていない

(continued on next page)

(continued from last page)

VTSSmartcardNotFoundException - ICカードアクセスに対する応答がない(一般的にはICカードが存在しない場合に発生する)
VTSUnsupportedMessageException - ICカードがこの処理をサポートしていない

---

getListener

```
public ReceptionListener getListener()
```

setListener()メソッドにより登録したReceptionListenerを返却する。 登録されていない場合はnullを返却する。

Returns:
登録されているReceptionListener

---

setListener

```
public void setListener(ReceptionListener listener)
```

交換および譲渡などの処理結果や確認情報を通知するためのリスナlistenerを登録する。 登録されたリスナは、交換/譲渡要求受信時、交換/譲渡応答受信時、交換/譲渡完了通知受信時、交換/譲渡中断通知受信時および、交換/譲渡異常検出通知受信時などに呼び出される。 このため、権利価値の交換および譲渡を行う場合には、本メソッドにてリスナの登録が必須である。 nullを指定すると、リスナの登録を解除する。

Parameters:
listener - 交換および譲渡などの処理結果や確認情報を通知するためのリスナ

---

getAuthMode

```
public int getAuthMode()
```

現在の権限モードを返却する。 返却する権限モードは、AuthHandlerインタフェースが提供する値が使用される。

Returns:
現在の権限モード

---

changeAuthMode

```
public boolean changeAuthMode(AuthHandler verifier,
                              int mode)
                       throws VTSInternalException,
                              VTSAccessViolationException,
                              VTSSmartcardNotFoundException,
                              VTSParameterException
```

変更する権限モードmodeの状態にAgentの権限モードを変更する。 変更する権限モードmodeで指定する値は、AuthHandlerインタフェースが提供する値を指定すること。 変更する権限モードmodeが所有者モードの場合、認証アルゴリズを搭載したインタフェースverifierのAuthHandler#authentication()メソッドが呼び返され、認証操作が行われる。 変更する権限モードmodeが非認証モードの場合は、認証アルゴリズムを搭載したインタフェースverifierは使用されないため、nullの指定が可能である。 なお、認証アルゴリズムを搭載したインタフェースverifierについては、AP側で実装を定義する必要がある。 現在の状態が所有者モードであるときにICカードアクセスを終了する場合は、本メソッドを使用し非認証モードに変更してから終了することを推奨する。

Parameters:
verifier - 認証アルゴリズムを搭載したインタフェース
mode - 変更する権限モード

Returns:
成功の場合trueを返却

Exceptions:
VTSInternalException - 内部的なエラーが発生した
VTSAccessViolationException - 認証に失敗した

(continued on next page)

(continued from last page)

VTSSmartcardNotFoundException - ICカードアクセスに対する応答がない(一般的にはICカードが存在しない場合に発生する)
VTSParameterException - 権限モードmodeが所有者モードのときに、認証アルゴリズムを搭載したインタフェースverifierにnullが設定された、もしくは権限モードmodeが正しくない

---

getHolder

```
public Participant getHolder()
```

ICカードを表わすParticipantを返却する。

Returns:
ICカードを表わすParticipant

---

getArbiter

```
public Participant getArbiter()
```

Agentに現在登録されている調停サーバを返却する。 登録されていない場合はnullを返却する。

Returns:
Agentに現在登録されている調停サーバ

---

setArbiter

```
public void setArbiter(Participant arbiter)
                throws VTSParameterException
```

交換および譲渡で用いる調停サーバを登録する。

Parameters:
arbiter - 調停サーバ

Exceptions:
VTSParameterException - 調停サーバarbiterにnullが設定された

org.t_engine.tenet.vts

Interface AuthHandler

public interface **AuthHandler**

権限モード移行時に認証処理を行う機能を実装する為のインタフェースである。
このインタフェースは、権限モードの指定値も定義する。

See Also:
Agent

Fields

None

```
public static final int None
```

非認証モード

Owner

```
public static final int Owner
```

所有者モード

Methods

getAuthentication

```
public byte[] getAuthentication(byte[] challenge)
                         throws VTSParameterException,
                                VTSInternalException
```

権限モードを所有者モードとしてAgentを取得または、Agentの権限モード変更をする際に認証情報を生成するために呼び出される。このインタフェースを実装したクラスにおいて、認証情報生成用のアルゴリズムを実装する必要がある。

Parameters:
challenge - 認証用のチャレンジ情報

Returns:
認証情報

Exceptions:
VTSParameterException - 認証のチャレンジ情報challengeがnullまたはサイズが正しくない
VTSInternalException - 認証情報生成時に内部的なエラーが発生した

org.t_engine.tenet.vts

Interface FileAcl

public interface **FileAcl**

権利価値に対するアクセス権の保持を行うインタフェースである。
アクセス権は、権利価値を生成した発行者以外に対して適用される。このため、発行者はアクセス権の有無に関わらずすべての操作が可能である。このインタフェースは、アクセス権の指定値も定義する。

See Also:
StoredVoucher

Fields

CopyAccess

```
public static final int CopyAccess
```

複製権限

TransferAccess

```
public static final int TransferAccess
```

転送権限

Methods

isDuplicatable

```
public boolean isDuplicatable()
```

権利価値の複製が可能な場合はtrueを返却する。

Returns:
権利価値の複製が可能な場合はtrue

isTransferable

```
public boolean isTransferable()
```

権利価値の転送が可能な場合はtrueを返却する。

Returns:
権利価値の転送が可能な場合はtrue

org.t_engine.tenet.vts

Interface Folder

public interface **Folder**
extends org.t_engine.tenet.util.Set

ICカードに格納されているフォルダを操作するためのエントリポイントを提供するインタフェースである。
Agent#getFolders()メソッドを呼び出すことにより取得する。Folderは、StoredVoucherの集合であり、自身は、Setインタフェースを継承する。

See Also:
Agent, FolderAcl

Methods

getName

```
public java.lang.String getName()
```

このフォルダのフォルダ名を返却する。

Returns:
このフォルダのフォルダ名

createVoucher

```
public StoredVoucher createVoucher(byte[] promise,
                                   int num,
                                   int acl)
                            throws VTSInternalException,
                                   VTSAccessViolationException,
                                   VTSObjectNotFoundException,
                                   VTSMemoryOverflowException,
                                   VTSMaximumNumberException,
                                   VTSSmartcardNotFoundException,
                                   VTSParameterException
```

生成対象の権利価値の内容promise、生成対象の権利価値の個数num、生成対象の権利価値のアクセス権aclおよび、Folderを持つICカードのeTRONIDを発行者として、ICカード内に権利価値を生成する。 生成対象の権利価値のアクセス権aclの指定値については、FileAclインタフェースが提供する値を使用すること。 複数のアクセス権を付与する場合は、FileAclインタフェースの定数フィールド値を論理和で連結して指定することができる(例えば、複製権限と転送権限を付与する場合は、createVoucher(promise, num, FileAcl.CopyAccess | FileAcl.TransferAccess)などとすることができる)。

Parameters:
promise - 生成対象の権利価値の内容
num - 生成対象の権利価値の個数
acl - 生成対象の権利価値のアクセス権

Returns:
ICカードに格納された権利価値

Exceptions:
VTSInternalException - 内部的なエラーが発生した
VTSAccessViolationException - 処理に必要なアクセス権をAgentが得ていない
VTSObjectNotFoundException - 操作対象のフォルダが存在しない
VTSMemoryOverflowException - ICカードの空き容量が不足している、または権利価値のサイズが格納可能最大値を超えている
VTSMaximumNumberException - 権利価値の個数が最大数を超えている

(continued from last page)

VTSSmartcardNotFoundException - ICカードに対する応答がない(一般的にはICカードが存在しない場合に発生する)
VTSParameterException - 生成対象の権利価値の内容promiseがnull、または生成対象の権利価値の個数numが0以下、または生成対象の権利価値のアクセス権aclが正しくない

---

getIdentifier

```
public java.lang.String getIdentifier()
```

このフォルダのフォルダIDを返却する。

Returns:

このフォルダのフォルダID

---

getFolderAcl

```
public FolderAcl getFolderAcl()
```

このフォルダのアクセス権を返却する。

Returns:

このフォルダのアクセス権

---

iterator

```
public org.t_engine.tenet.util.Iterator iterator()
```

このフォルダが保持する権利価値のコレクションを返却する。 ICカードに対する応答がない場合や処理に必要なアクセス権が存在しない等の理由により権利価値の取得に失敗した場合はnullを返却する。 このフォルダ内に権利価値が存在しない場合は空のコレクションを返却する。

Returns:

Iterator 権利価値のコレクション、権利価値取得に失敗した場合はnull

---

iterator

```
public org.t_engine.tenet.util.Iterator iterator(VoucherCondition condition)
```

権利価値抽出条件conditionを満たす権利価値のコレクションを返却する。 ICカードに対する応答がない場合や処理に必要なアクセス権が存在しない等の理由により権利価値の取得に失敗した場合はnullを返却する。 権利価値抽出条件conditionを満たす権利価値が存在しない場合は空のコレクションを返却する。

Parameters:

condition - 権利価値抽出条件

Returns:

Iterator 権利価値のコレクション、権利価値取得に失敗した場合はnull

org.t_engine.tenet.vts

Interface FolderAcl

public interface **FolderAcl**

フォルダに対するアクセス権の保持を行うインタフェースである。
アクセス権は、所有者以外に対して適用される。このため、所有者はアクセス権の有無に関わらずすべての操作が可能である。このインタフェースは、アクセス権の指定値も定義する。

See Also:
Folder

Fields

ReadAccess

```
public static final int ReadAccess
```

フォルダリード権限

CreateAccess

```
public static final int CreateAccess
```

権利価値作成権限

TransferAccess

```
public static final int TransferAccess
```

権利価値転送権限

Methods

isReadable

```
public boolean isReadable()
```

フォルダの読み込みが可能な場合はtrueを返却する。

Returns:
フォルダの読み込みが可能な場合はtrue

isCreateable

```
public boolean isCreateable()
```

フォルダ内に権利価値が作成可能な場合はtrueを返却する。

Returns:
フォルダ内に権利価値が作成可能な場合はtrue

isTransferable

```
public boolean isTransferable()
```

フォルダ内の権利価値が転送可能な場合はtrueを返却する。

(continued from last page)

Returns:

フォルダ内の権利価値が転送可能な場合はtrue

org.t_engine.tenet.vts

Interface Participant

All Subinterfaces:
Agent

---

public interface **Participant**

eTRONIDを持つ主体を表わすインタフェースである。
TENeTICカードや調停サーバおよび権利価値の取引相手となるアプリケーションなどのeTRONIDが割り当てられた主体を表象する。ParticipantRepository#lookup()メソッドを呼び出すことにより取得する。

See Also:
ParticipantRepository, Agent, AuthHandler

Methods

getIdentifier

```
public java.lang.String getIdentifier()
```

このParticipantのeTRONIDを返却する。

Returns:
このParticipantのeTRONID

---

getAgent

```
public Agent getAgent(AuthHandler verifier,
                      int mode)
               throws VTSInternalException,
                      VTSAccessViolationException,
                      VTSSmartcardNotFoundException,
                      VTSParameterException
```

Participantに対するアクセスを行うためのAgentを生成/返却する。 ICカードを表わす(ICカードのeTRONIDを持つ)Participantに対してのみ実行が可能である。 権限モードmodeで指定する値は、AuthHandlerインタフェースが提供する値を指定すること。 権限モードmodeが所有者モードの場合、認証アルゴリズを搭載したインタフェースverifierのAuthHandler#authentication()メソッドが呼び返され、認証操作が行われる。 権限モードmodeが非認証モードの場合は、認証アルゴリズムを搭載したインタフェースverifierは使用されないため、nullの指定が可能である。 なお、認証アルゴリズムを搭載したインタフェースverifierについては、AP側で実装を定義する必要がある。 権限モードmodeに所有者モードを指定しAgentを取得した場合にICカードアクセスを終了する場合は、 Agent#changeAuthMode()メソッドを使用し非認証モードに変更してから終了することを推奨する。

Parameters:
verifier - 認証アルゴリズム搭載したインタフェース
mode - 権限モード

Returns:
Participantに対するアクセスを行うためのAgent

Exceptions:
VTSInternalException - 内部的なエラーが発生した
VTSAccessViolationException - 認証に失敗した
VTSSmartcardNotFoundException - ICカードを表わすParticipantでないかまたはeTRONIDを払い出すことのできるTENeTICカードが見つからない

(continued from last page)

VTSParameterException - 権限モードmodeが所有者モードのときに、認証アルゴリズムを搭載したインタフェースverifierにnullが設定された、もしくは権限モードmodeが正しくない

org.t_engine.tenet.vts

Interface ParticipantRepository

public interface **ParticipantRepository**

Participantの生成/管理をするインタフェースである。
本APIを使用するためには、本インタフェースを実装するクラスのインスタンスを取得する必要がある。

See Also:
Participant

Methods

lookup

```
public Participant lookup(java.lang.String eTronID)
                   throws VTSParameterException
```

生成するParticipantのeTRONIDを表わすParticipantを生成し、返却する。

Parameters:
eTronID - 生成するParticipantのeTRONID

Returns:
生成するParticipantのeTRONIDを表すParticipant

Exceptions:
VTSParameterException - 生成するParticipantのeTRONIDにnullが設定された

org.t_engine.tenet.vts

Interface ReceptionListener

public interface **ReceptionListener**

ICカードに格納された権利価値に対して行った交換または譲渡の結果通知などを受けるためのリスナを構成するインタフェースである。
権利価値の交換または譲渡を行う場合には、このインタフェースを継承した実装が必須であり、Agentに対して登録する必要がある。

See Also:
Agent, Session

Methods

offered

```
public void offered(Session offer,
                    byte[] receivingVoucherCondition)
```

交換要求が受信されたことを通知する。 交換要求に対するセッションofferおよび相手から要求された権利価値の交換条件receivingVoucherConditionについて確認を行い、 Session#agree()メソッドを使用することで提示された要求に合意する。 提示された要求に合意しない場合は、Session#reject()メソッドを使用する。この場合、相手にはsuspended()メソッドにより通知される。

Parameters:
offer - 交換要求に対するセッション
receivingVoucherCondition - 相手から要求された権利価値の交換条件

agreed

```
public void agreed(Session session)
```

譲渡要求に対する応答が受信されたことを通知する。 譲渡応答に対するセッションsessionについて確認を行い、Session#confirm()メソッドを使用することで譲渡を確定する。 譲渡応答に対して拒否を行う場合はSession#reject()メソッドを使用する。この場合、相手にはsuspended()メソッドにより通知される。

Parameters:
session - 譲渡応答に対するセッション

agreed

```
public void agreed(Session session,
                   Voucher receivingVoucher)
```

交換要求に対する応答が受信されたことを通知する。 交換応答に対するセッションsessionおよび相手側から指定された交換対象の権利価値receivingVoucherについて確認を行い、 Session#confirm()メソッドを使用することで交換を確定する。 交換応答に対して拒否を行う場合はSession#reject()メソッドを使用する。この場合、相手にはsuspended()メソッドにより通知される。

Parameters:
session - 交換応答に対するセッション
receivingVoucher - 相手側から指定された交換対象の権利価値

committed

```
public void committed(Session session)
```

(continued from last page)

交換および譲渡に対するセッションsessionについて、交換および譲渡が完了したことを通知する。 この通知を受信したことにより、交換した権利価値の使用が可能となる。

Parameters:

session - 交換および譲渡に対するセッション

---

aborted

```
public void aborted(Session session,
                    int causeCode)
```

交換および譲渡に対するセッションsessionについて、交換および譲渡が取り消されたことを通知する。 このメソッドが呼び出された場合は、sessionで表される交換および譲渡がキャンセルまたは調停サーバより中止されたことを表わす。 取り消し理由は、引数で指定された取り消し原因コードcauseCodeで判別可能とする。 取り消し原因コードにはSessionインタフェースが定義する定数フィールド値が指定される。

Parameters:

session - 交換および譲渡に対するセッション
causeCode - 取り消し原因コード

---

suspended

```
public void suspended(Session session,
                      int causeCode)
```

交換および譲渡の異常検出を通知する。 交換および譲渡に対するセッションsessionについて、異常を検出したことを通知する。 異常検出理由は、引数で指定された異常検出原因コードcauseCodeで判別可能とする。 異常検出原因コードにはSessionインタフェースが定義する定数フィールド値が指定される。 また、異常検出原因コードが"拒否"を表す場合、Session#getRejectReason()メソッドを呼び出すことで拒否理由を取得することが可能である。 本メソッドが呼び出された場合は、Session#recover()メソッドを呼び出し、交換を復旧させる必要がある。

Parameters:

session - 交換および譲渡に対するセッション
causeCode - 異常検出原因コード

org.t_engine.tenet.vts

Interface Session

public interface **Session**

交換および譲渡の情報を保持するインタフェースである。また、このインタフェースは交換や譲渡に対する操作(合意や確定、復旧など)も提供する。
ICカードに格納されている権利価値について交換および譲渡を行う場合に生成される。このインタフェースは交換および譲渡の処理が正常、異常問わずに完了するまで一意となるIDを持つ。
交換および譲渡の復旧を行う場合はrecover()メソッドを、交換および譲渡の要求をされた(提案を受けた)側において、合意を行う場合はagree()メソッドを、交換および譲渡を要求(提案)した側において、確定を行う場合はconfirm()メソッドを呼び出す必要がある。また、要求(提案)した側・された側いずれにおいても拒否を行う場合はreject()メソッドを呼び出す必要がある。

See Also:
Voucher, Participant

Fields

None

`public static final int` **`None`**

中断通知または異常検出通知の発生原因が「中断や異常検出が発生していない」を表す。

AbortedByRecovery

`public static final int` **`AbortedByRecovery`**

中断通知または異常検出通知の発生原因が「調停サーバにより交換が取り消された」を表す。

ExchangeRejected

`public static final int` **`ExchangeRejected`**

中断通知または異常検出通知の発生原因が「交換および譲渡の要求(提案)した側に拒否された」を表す。

AccessViolation

`public static final int` **`AccessViolation`**

中断通知または異常検出通知の発生原因が「処理に必要なアクセス権をAgentが得ていない」を表す。

ObjectNotFound

`public static final int` **`ObjectNotFound`**

中断通知または異常検出通知の発生原因が「フォルダまたは権利価値が存在しない」を表す。

IllegalParameters

`public static final int` **`IllegalParameters`**

中断通知または異常検出通知の発生原因が「指定した権利価値では交換を確定することができません」を表す。

(continued from last page)

MaximumNumberExceeded

```
public static final int MaximumNumberExceeded
```

中断通知または異常検出通知の発生原因が「権利価値個数が不足している」を表す。

MemoryOverflow

```
public static final int MemoryOverflow
```

中断通知または異常検出通知の発生原因が「ICカード内に交換情報を格納する領域が不足している」を表す。

IncompatibleStatus

```
public static final int IncompatibleStatus
```

中断通知または異常検出通知の発生原因が「交換処理が存在しないまたは状態が不正のため続行不能」を表す。

InternalError

```
public static final int InternalError
```

中断通知または異常検出通知の発生原因が「内部的なエラーが発生した」を表す。

Methods

isOriginator

```
public boolean isOriginator()
                     throws VTSInternalException,
                            VTSAccessViolationException,
                            VTSObjectNotFoundException,
                            VTSSmartcardNotFoundException,
                            VTSUnsupportedMessageException
```

このセッションが交換および譲渡の要求(提案)した側であればtrueを返却する。

Returns:

このセッションが交換および譲渡の要求(提案)した側であればtrue

Exceptions:

`VTSInternalException` - 内部的なエラーが発生した
`VTSAccessViolationException` - 処理に必要なアクセス権をAgentが得ていない
`VTSObjectNotFoundException` - 操作対象の交換情報がすでにICカードに存在しない
`VTSSmartcardNotFoundException` - ICカードアクセスに対する応答がない(一般的にはICカードが存在しない場合に発生する)
`VTSUnsupportedMessageException` - ICカードがこの処理をサポートしていない

---

isRecipient

```
public boolean isRecipient()
                    throws VTSInternalException,
                           VTSAccessViolationException,
                           VTSObjectNotFoundException,
                           VTSSmartcardNotFoundException,
                           VTSUnsupportedMessageException
```

このセッションが交換および譲渡を要求された(提案を受けた)側であればtrueを返却する。

Returns:

このセッションが交換および譲渡を要求された(提案を受けた)側であればtrue

Exceptions:

(continued from last page)

VTSInternalException - 内部的なエラーが発生した
VTSAccessViolationException - 処理に必要なアクセス権をAgentが得ていない
VTSObjectNotFoundException - 操作対象の交換情報がすでにICカードに存在しない
VTSSmartcardNotFoundException - ICカードアクセスに対する応答がない(一般的にはICカードが存在しない場合に発生する)
VTSUnsupportedMessageException - ICカードがこの処理をサポートしていない

---

agree

```
public void agree(Folder incomingFolder,
                  StoredVoucher sendingVoucher)
           throws VTSInternalException,
                  VTSParameterException,
                  VTSIncompatibleStatusException
```

このセッションが表す交換および譲渡に対して合意を行う。 ReceptionListener#offerd()メソッドにより通知された交換および譲渡要求に対して合意を行う場合に呼び出し、 相手から渡される権利価値の格納場所と、自分が渡す権利価値を指定する。

Parameters:

incomingFolder - 交換および譲渡により送信される権利価値を格納するフォルダ
sendingVoucher - 自分が渡す権利価値

Exceptions:

VTSInternalException - 内部的なエラーが発生した
VTSParameterException - 譲渡される権利価値を格納するフォルダincomingFolderにnullが設定された
VTSIncompatibleStatusException - セッションの内部情報が不正で譲渡および交換処理が実行できなかった

---

confirm

```
public void confirm(Folder incomingFolder)
            throws VTSInternalException,
                   VTSParameterException,
                   VTSIncompatibleStatusException
```

このセッションが表す交換および譲渡の確定を行う。 ReceptionListener#agreed()メソッドにより通知された交換および譲渡の応答に対して確定を行う場合に呼び出し、 相手から渡される権利価値の格納場所を指定する。相手から渡される権利価値が存在しない場合は、 交換により送信される権利価値を格納するフォルダincomingFolderにnullを指定すること。

Parameters:

incomingFolder - 交換により送信される権利価値を格納するフォルダ

Exceptions:

VTSInternalException - 内部的なエラーが発生した
VTSParameterException - 相手から渡される権利価値が存在する時に交換により送信される権利価値を格納するフォルダincomingFolderにnullが設定された
VTSIncompatibleStatusException - セッションの内部情報が不正で譲渡および交換処理が実行できなかった

---

reject

```
public void reject(java.lang.String rejectReason)
           throws VTSInternalException
```

このセッションが表す交換および譲渡に対して拒否を行う。 ReceptionListener#offerd()またはagreed()メソッドにより通知された交換および譲渡の応答に対して拒否を行う場合に呼び出し、拒否の理由を指定する。 交換および譲渡の相手に対してはReceptionListener#suspended()メソッドにより通知される。 通知された相手はSession#getRejectReason()メソッドにより拒否の理由rejectReasonを取得することができる。 拒否の理由がない場合はnullを指定すること。

Parameters:

rejectReason - 拒否の理由

(continued from last page)

Exceptions:
VTSInternalException - 内部的なエラーが発生した

---

recover

```
public void recover()
            throws VTSInternalException
```

このセッションが表す交換および譲渡に対してキャンセルまたは調停サーバに復旧依頼を行う。 キャンセルが行われるか調停サーバへ復旧依頼が行われるかはICカードが保持しているセッションの状態に依存する。 復旧依頼に対しての完了通知、中断通知および異常検出通知を受けるため、ReceptionListenerの登録が必須である。

Exceptions:
VTSInternalException - 内部的なエラーが発生した

---

getIdentifier

```
public java.lang.String getIdentifier()
```

このセッションのIDを返却する。

Returns:
このセッションのID

---

getPartner

```
public Participant getPartner()
```

このセッションが表す交換および譲渡の相手を返却する。 このセッションが交換および譲渡の要求(提案)した側の場合、交換および譲渡を要求された(提案を受けた)側のParticipantを返却する。 このセッションが交換および譲渡を要求された(提案を受けた)側の場合、交換および譲渡の要求(提案)した側のParticipantを返却する。 存在しない場合はnullを返却する(交換が中断されすでにICカードに管理されていない場合など)。

Returns:
交換および譲渡対象の相手

---

getSendingVoucher

```
public Voucher getSendingVoucher()
```

このセッションが表す交換および譲渡における送信対象の権利価値を返却する。 存在しない場合はnullを返却する(譲渡のときなど)。

Returns:
このセッションが表す交換および譲渡における送信対象の権利価値

---

getReceivingVoucher

```
public Voucher getReceivingVoucher()
```

このセッションが表す交換および譲渡における受信対象の権利価値を返却する。 存在しない場合はnullを返却する(譲渡のときなど)。

Returns:
このセッションが表す交換および譲渡における受信対象の権利価値

(continued from last page)

isCancelable

```
public boolean isCancelable()
                     throws VTSInternalException,
                            VTSAccessViolationException,
                            VTSSmartcardNotFoundException,
                            VTSUnsupportedMessageException
```

このセッションがキャンセル可能な場合はtrueを返却する。

Returns:

このセッションがキャンセル可能な場合はtrue

Exceptions:

VTSInternalException - 内部的なエラーが発生した
VTSAccessViolationException - 処理に必要なアクセス権をAgentが得ていない
VTSSmartcardNotFoundException - ICカードアクセスに対する応答がない(一般的にはICカードが存在しない場合に発生する)
VTSUnsupportedMessageException - ICカードがこの処理をサポートしていない

---

isRecoverable

```
public boolean isRecoverable()
                      throws VTSInternalException,
                             VTSAccessViolationException,
                             VTSSmartcardNotFoundException,
                             VTSUnsupportedMessageException
```

このセッションが復旧可能な場合はtrueを返却する。

Returns:

このセッションが復旧可能な場合はtrue

Exceptions:

VTSInternalException - 内部的なエラーが発生した
VTSAccessViolationException - 処理に必要なアクセス権をAgentが得ていない
VTSSmartcardNotFoundException - ICカードアクセスに対する応答がない(一般的にはICカードが存在しない場合に発生する)
VTSUnsupportedMessageException - ICカードがこの処理をサポートしていない

---

getRejectReason

```
public java.lang.String getRejectReason()
```

このセッションが異常検出通知(拒否)を受けた際の拒否理由を返却する。 拒否理由は、交換および譲渡の相手により拒否が行われた際に指定された拒否理由である。 拒否理由が存在しない場合は0バイトの文字列を返却する。

Returns:

このセッションが異常検出通知(拒否)を受けた際の拒否理由

org.t_engine.tenet.vts

Interface StoredVoucher

All Superinterfaces:
Voucher

---

public interface **StoredVoucher**
extends Voucher

ICカードに格納されている権利価値を操作するためのエントリポイントを提供するインタフェースである。
このインタフェースはICカードに格納されている権利価値を表す。Folder#createVoucher()メソッドおよび、Folder#iterator()メソッドにて抽出することにより取得する。

See Also:
Folder, Voucher

Methods

transfer

```
public Session transfer(Participant destination)
                 throws VTSInternalException,
                        VTSParameterException
```

転送相手destinationに対してこの権利価値を譲渡する。 譲渡要求に対しての譲渡応答、完了通知および異常検出通知を受けるため、ReceptionListenerの登録が必須である。 また、転送相手からの応答が返ってこない場合、返値であるセッションのSession#recover()メソッドを呼び出すことにより、譲渡要求を復旧することができる。

Parameters:
destination - 転送相手

Returns:
譲渡要求に対するセッション

Exceptions:
VTSInternalException - 内部的なエラーが発生した
VTSParameterException - 転送相手destinationにnullが設定された

---

exchange

```
public Session exchange(Participant destination,
                        byte[] receivingVoucherCondition)
                 throws VTSInternalException,
                        VTSParameterException
```

交換相手destinationに対してこの権利価値と交換条件receivingVoucherに対応する権利価値を交換する。 交換要求に対しての交換応答、完了通知および異常検出通知を受けるため、ReceptionListenerの登録が必須である。 また、交換相手側からの応答が返ってこない場合、本メソッドの返値であるセッションのSession#recover()メソッドを呼び出すことにより、交換要求を復旧することができる。 交換条件がない場合は、交換条件receivingVoucherConditionにnullを指定する。

Parameters:
destination - 交換相手
receivingVoucherCondition - 交換条件

Returns:

(continued from last page)

交換要求に対するセッション

Exceptions:

VTSInternalException - 内部的なエラーが発生した、またはAgentにarbiterが設定されていない
VTSParameterException - 交換相手destinationにnullが設定された

---

move

```
public StoredVoucher move(Folder folder)
                  throws VTSInternalException,
                         VTSAccessViolationException,
                         VTSObjectNotFoundException,
                         VTSMemoryOverflowException,
                         VTSMaximumNumberException,
                         VTSParameterException,
                         VTSSmartcardNotFoundException
```

この権利価値を移動先のフォルダfolderに移動する。

Parameters:

folder - 移動先のフォルダ

Returns:

移動後の権利価値

Exceptions:

VTSInternalException - 内部的なエラーが発生した
VTSAccessViolationException - 処理に必要なアクセス権をAgentが得ていない
VTSObjectNotFoundException - 操作対象の権利価値またはフォルダが存在しない
VTSMemoryOverflowException - ICカードの空き容量が不足している、または権利価値のサイズが格納可能最大値を超えている
VTSMaximumNumberException - 移動元の権利価値の個数が不足している、または移動先の権利価値の個数が最大数を超える
VTSParameterException - 移動先のフォルダfolderにnullが設定された、または移動先に移動元と同じフォルダが設定された
VTSSmartcardNotFoundException - ICカードアクセスに対する応答がない(一般的にはICカードが存在しない場合に発生する)

---

copy

```
public StoredVoucher copy(Folder folder)
                  throws VTSInternalException,
                         VTSAccessViolationException,
                         VTSObjectNotFoundException,
                         VTSMemoryOverflowException,
                         VTSMaximumNumberException,
                         VTSParameterException,
                         VTSSmartcardNotFoundException
```

この権利価値を複製先のフォルダfolderへ複製する。

Parameters:

folder - 複製先のフォルダ

Returns:

複製後の権利価値

Exceptions:

VTSInternalException - 内部的なエラーが発生した
VTSAccessViolationException - 処理に必要なアクセス権をAgentが得ていない、またはこの権利価値を複製する権限がない
VTSObjectNotFoundException - 操作対象の権利価値またはフォルダが存在しない
VTSMemoryOverflowException - ICカードの空き容量が不足している、または権利価値のサイズが格納可能最大値を超えている

(continued from last page)

VTSMaximumNumberException - 移動元の権利価値の個数が不足している、または移動先の権利価値の個数が最大数を超える
VTSParameterException - 移動先のフォルダfolderにnullが設定された、または移動先に移動元と同じフォルダが設定された
VTSSmartcardNotFoundException - ICカードアクセスに対する応答がない(一般的にはICカードが存在しない場合に発生する)

---

remove

```
public void remove(int num)
            throws VTSInternalException,
                   VTSAccessViolationException,
                   VTSObjectNotFoundException,
                   VTSMaximumNumberException,
                   VTSParameterException,
                   VTSSmartcardNotFoundException
```

この権利価値を削除する。

Parameters:

num - 削除する個数

Exceptions:

VTSInternalException - 内部的なエラーが発生した
VTSAccessViolationException - 処理に必要なアクセス権をAgentが得ていない
VTSObjectNotFoundException - 操作対象の権利価値またはフォルダが存在しない
VTSMaximumNumberException - 削除する個数numに存在する個数より大きい値が設定された
VTSParameterException - 削除する個数numに0未満の値が設定された
VTSSmartcardNotFoundException - ICカードアクセスに対する応答がない(一般的にはICカードが存在しない場合に発生する)

---

getFolder

```
public Folder getFolder()
```

この権利価値が格納されているフォルダを返却する。

Returns:

この権利価値が格納されているフォルダ

---

getIdentifier

```
public java.lang.String getIdentifier()
```

この権利価値の識別子を返却する。

Returns:

この権利価値の識別子

---

select

```
public StoredVoucher select(int num)
                    throws VTSParameterException,
                           VTSInternalException
```

選択個数num分の権利価値を返却する。 選択個数numが0以下や現在の保持数より大きい場合はVTSParameterExceptionをスローする。

Parameters:

num - 選択個数

Returns:

選択個数num分の権利価値

(continued from last page)

Exceptions:

`VTSInternalException` - 内部的なエラーが発生した

`VTSParameterException` - 選択個数numに0以下、または保持数より大きい値が設定された

org.t_engine.tenet.vts

Interface Voucher

All Subinterfaces:
StoredVoucher

---

public interface **Voucher**

権利価値を表わすインタフェースである。
このインタフェースは権利価値を構成する要素を保持する。

See Also:
Participant, StoredVoucher

Methods

getIssuer

```
public Participant getIssuer()
```

この権利価値の発行者を返却する。

Returns:
この権利価値の発行者

---

getPromise

```
public byte[] getPromise()
```

この権利価値の内容を返却する。

Returns:
この権利価値の内容

---

getCount

```
public int getCount()
```

この権利価値の個数を返却する。

Returns:
この権利価値の個数

---

getFileAcl

```
public FileAcl getFileAcl()
```

この権利価値のアクセス権を返却する。

Returns:
この権利価値のアクセス権

org.t_engine.tenet.vts

Interface VoucherCondition

public interface **VoucherCondition**

権利価値を検索するための検索インタフェースを提供するインタフェースである。
このインタフェースを実装したクラスをFolder#iterator()メソッドのパラメータに指定することで、権利価値の検索(フィルタリング)が可能である。

See Also:
Voucher, Participant

Methods

matches

```
public boolean matches(Voucher object)
```

比較対象の権利価値objectがこの交換条件に合う場合はtrueを返却する。 比較対象の権利価値objectは、通常、ICカードに格納されている権利価値(StoredVoucher)を指定される。

Parameters:
object - 比較対象の権利価値

Returns:
比較対象の権利価値objectがこの交換条件に合う場合はtrue

org.t_engine.tenet.vts

Class VTSAccessViolationException

```
java.lang.Object
   |-java.lang.Throwable
      |-java.lang.Exception
         |-org.t_engine.tenet.vts.VTSException
            |-org.t_engine.tenet.vts.VTSAccessViolationException
```

public class **VTSAccessViolationException**
extends VTSException

**指定された権利価値またはフォルダへのアクセス権がないため、ICカードが処理を中断した場合にスローされる。
認証に失敗したときにもスローされる。**

Constructors

VTSAccessViolationException

```
public VTSAccessViolationException(java.lang.String string)
```

詳細メッセージstringを持つVTSAccessViolationExceptionを構築する。

Parameters:
`string` - **詳細メッセージ**

org.t_engine.tenet.vts

Class VTSException

```
java.lang.Object
   |-java.lang.Throwable
      |-java.lang.Exception
         |-org.t_engine.tenet.vts.VTSException
```

Direct Known Subclasses:
VTSAccessViolationException, VTSIncompatibleStatusException, VTSInternalException, VTSMaximumNumberException, VTSMemoryOverflowException, VTSMessageSizeOverflowException, VTSObjectNotFoundException, VTSParameterException, VTSSmartcardNotFoundException, VTSUnsupportedMessageException

---

public class **VTSException**
extends java.lang.Exception

vtsで例外が発生した際にスローされる例外の基底クラスである。

Constructors

VTSException

```
public VTSException(java.lang.String string)
```

詳細メッセージstringを持つVTSExceptionを構築する。

Parameters:
`string` - 詳細メッセージ

org.t_engine.tenet.vts

Class VTSIncompatibleStatusException

```
java.lang.Object
   |-java.lang.Throwable
      |-java.lang.Exception
         |-org.t_engine.tenet.vts.VTSException
            |-org.t_engine.tenet.vts.VTSIncompatibleStatusException
```

public class **VTSIncompatibleStatusException**
extends VTSException

指定された交換および譲渡が見つからない場合、または交換および譲渡の状態が不正な場合にスローされる。

Constructors

VTSIncompatibleStatusException

```
public VTSIncompatibleStatusException(java.lang.String string)
```

詳細メッセージstringを持つVTSIncompatibleStatusExceptionを構築する。

Parameters:
`string` - **詳細メッセージ**

org.t_engine.tenet.vts

Class VTSInternalException

```
java.lang.Object
  ┗-java.lang.Throwable
      ┗-java.lang.Exception
          ┗-org.t_engine.tenet.vts.VTSException
              ┗-org.t_engine.tenet.vts.VTSInternalException
```

public class **VTSInternalException**
extends VTSException

vtsで内部的なエラーが発生した場合にスローされる。
例えばICカードが書き込み異常を通知してきた場合などにスローされる。

Constructors

VTSInternalException

```
public VTSInternalException(java.lang.String string)
```

詳細メッセージstringを持つVTSInternalExceptionを構築する。

Parameters:
string - 詳細メッセージ

org.t_engine.tenet.vts

Class VTSMaximumNumberException

```
java.lang.Object
    |-java.lang.Throwable
        |-java.lang.Exception
            |-org.t_engine.tenet.vts.VTSException
                |-org.t_engine.tenet.vts.VTSMaximumNumberException
```

public class **VTSMaximumNumberException**
extends VTSException

権利価値の個数などが上限／下限値を超える処理が要求されたため、ICカードが処理を中断した場合にスローされる。
例えば、権利価値を移動すると移動先の権利価値個数が格納限度数を超える場合などにスローされる。

Constructors

VTSMaximumNumberException

`public VTSMaximumNumberException(java.lang.String string)`

詳細メッセージstringを持つVTSMaximumNumberExceptionを構築する。

Parameters:
`string` - **詳細メッセージ**

org.t_engine.tenet.vts

Class VTSMemoryOverflowException

```
java.lang.Object
    ┗-java.lang.Throwable
        ┗-java.lang.Exception
            ┗-org.t_engine.tenet.vts.VTSException
                ┗-org.t_engine.tenet.vts.VTSMemoryOverflowException
```

public class **VTSMemoryOverflowException**
extends VTSException

**領域に関する上限／下限を超える処理が要求されたため、ICカードが処理を中断した場合にスローされる。
例えば、作成しようとしている権利価値のサイズが格納最大値を超えてる場合や、領域が不足しているためこれ以上フォルダが作成できない場合などにスローされる。**

Constructors

VTSMemoryOverflowException

`public VTSMemoryOverflowException(java.lang.String string)`

詳細メッセージstringを持つVTSMemoryOverflowExceptionを構築する。

Parameters:
`string` - 詳細メッセージ

org.t_engine.tenet.vts

Class VTSMessageSizeOverflowException

```
java.lang.Object
  ┗-java.lang.Throwable
      ┗-java.lang.Exception
          ┗-org.t_engine.tenet.vts.VTSException
              ┗-org.t_engine.tenet.vts.VTSMessageSizeOverflowException
```

public class **VTSMessageSizeOverflowException**
extends VTSException

**ICカードが送受信可能な応答サイズを超えた処理が要求されたため、処理を中断した場合にスローされる。
例えば、権利価値一覧の取得結果が送受信可能な応答サイズを超える場合にスローされる。**

Constructors

VTSMessageSizeOverflowException

```
public VTSMessageSizeOverflowException(java.lang.String string)
```

詳細メッセージstringを持つVTSMessageSizeOverflowExceptionを構築する。

Parameters:
`string` - **詳細メッセージ**

org.t_engine.tenet.vts

Class VTSObjectNotFoundException

```
java.lang.Object
   |-java.lang.Throwable
      |-java.lang.Exception
         |-org.t_engine.tenet.vts.VTSException
            |-org.t_engine.tenet.vts.VTSObjectNotFoundException
```

public class **VTSObjectNotFoundException**
extends VTSException

操作対象の権利価値またはフォルダが存在しないため、ICカードが処理を中断した場合にスローされる。

Constructors

VTSObjectNotFoundException

```
public VTSObjectNotFoundException(java.lang.String string)
```

詳細メッセージstringを持つVTSObjectNotFoundExceptionを構築する。

Parameters:
`string` - **詳細メッセージ**

org.t_engine.tenet.vts

Class VTSParameterException

```
java.lang.Object
   |-java.lang.Throwable
      |-java.lang.Exception
         |-org.t_engine.tenet.vts.VTSException
            |-org.t_engine.tenet.vts.VTSParameterException
```

public class **VTSParameterException**
extends VTSException

メソッドの引数が不正な場合にスローされる。
例えばnullが許容されていないパラメータにnullを指定した場合などにスローされる。

Constructors

VTSParameterException

```
public VTSParameterException(java.lang.String string)
```

詳細メッセージstringを持つVTSParameterExceptionを構築する。

Parameters:
string - 詳細メッセージ

org.t_engine.tenet.vts

Class VTSSmartcardNotFoundException

```
java.lang.Object
   |-java.lang.Throwable
      |-java.lang.Exception
         |-org.t_engine.tenet.vts.VTSException
            |-org.t_engine.tenet.vts.VTSSmartcardNotFoundException
```

public class **VTSSmartcardNotFoundException**
extends VTSException

**ICカードアクセスに対する応答を受信する前にタイムアウトした場合にスローされる。
一般的にはICカードが存在しない状態でICカードアクセスした場合にスローされる。**

Constructors

VTSSmartcardNotFoundException

```
public VTSSmartcardNotFoundException(java.lang.String string)
```

詳細メッセージstringを持つVTSCardNotFoundExceptionを構築する。

Parameters:
`string` - **詳細メッセージ**

org.t_engine.tenet.vts

Class VTSUnsupportedMessageException

```
java.lang.Object
  ┗-java.lang.Throwable
      ┗-java.lang.Exception
          ┗-org.t_engine.tenet.vts.VTSException
              ┗-org.t_engine.tenet.vts.VTSUnsupportedMessageException
```

public class **VTSUnsupportedMessageException**
extends VTSException

交換系機能をサポートしないICカードを用いて、交換系機能の操作を行った場合にスローされる。

Constructors

VTSUnsupportedMessageException

```
public VTSUnsupportedMessageException(java.lang.String string)
```

詳細メッセージstringを持つVTSUnsupportedMessageExceptionを構築する。

Parameters:
`string` - **詳細メッセージ**

Index

A

aborted 18
AbortedByRecovery 19
AccessViolation 19
agree 21
agreed 17

C

changeAuthMode 6
committed 17
confirm 21
copy 25
CopyAccess 9
CreateAccess 12
createFolder 4
createVoucher 10

D

deleteFolder 4

E

exchange 24
ExchangeRejected 19

G

getAgent 14
getArbiter 7
getAuthentication 8
getAuthMode 6
getCount 28
getFileAcl 28
getFolder 26
getFolderAcl 11
getFolders 5
getHolder 7
getIdentifier 11, 14, 22, 26
getIssuer 28
getListener 6
getName 10
getPartner 22
getPromise 28
getReceivingVoucher 22
getRejectReason 23
getSendingVoucher 22
getSessions 5

I

IllegalParameters 19
IncompatibleStatus 20
InternalError 20
isCancelable 22
isCreateable 12
isDuplicatable 9
isOriginator 20
isReadable 12
isRecipient 20
isRecoverable 23
isTransferable 9, 12
iterator 11

L

lookup 16

M

matches 29
MaximumNumberExceeded 19
MemoryOverflow 20
move 25

N

None 8, 19

O

ObjectNotFound 19
offered 17
Owner 8

R

ReadAccess 12
recover 22
reject 21
remove 26

S

select 26
setArbiter 7
setListener 6
suspended 18

T

transfer 24
TransferAccess 9, 12

V

VTSAccessViolationException 30
VTSException 31
VTSIncompatibleStatusException 32
VTSInternalException 33
VTSMaximumNumberException 34
VTSMemoryOverflowException 35
VTSMessageSizeOverflowException 36
VTSObjectNotFoundException 37
VTSParameterException 38
VTSSmartcardNotFoundException 39
VTSUnsupportedMessageException 40

Appendix org.t_engine.util.* について

権利価値取引 API, e^{2}TP メッセージング API は, いずれも集合および連想配列を扱うためのインタフェース, およびイテレータを扱うためのインタフェースとして, org.t_engine.util.Set, org.t_engine.util.Map, org.t_engine.util.Iterator の 3 種類のインタフェースを利用している.

これらのインタフェースは, それぞれ java.util パッケージに存在する同名のインタフェース (java.util.Set, java.util.Map, java.util.Iterator) と, 以下の例外を除いて同一のインタフェースを提供するものとする.

すなわち, java.util.Collection が引数もしくは返値として現れるメソッドについては, 代わりに org.t_engine.util.Collection を用いるものとする.
org.t_engine.util.Collection インタフェースは, java.util.Collection インタフェースと同一のインタフェースを提供する.

これは, J2ME CLDC 環境では JCF (Java Collections Framework) が利用できないことによる対処である.