|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **ETSI TC SCP Meeting #65**  **Venice, Italy, 28-29 August 2014** | | | | | | | | | | | | | | ***SCP(14)000227r1*** | | | | | | | |
| **ETSI TC SCP TEC Meeting #53**  **Sophia Antipolis, France, 7-11 July 2014** | | | | | | | | | | | | | | ***SCPTEC(14)000118r4*** | | | | | | | |
| *TC SCP CR-Form-v1.8.0* | | | | | | | | | | | | | | | | | | | | | | |
| **CHANGE REQUEST** | | | | | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | | | | | | | |
| ⌘ | **102 705** | | | | **CR** | **042** | | | ⌘ rev |  | | | ⌘ Current version: | | | | | **11.0.0** | | ⌘ | | |
|  | | | | | | | | | | | | | | | | | | | | | | |
| *For* [***HE******LP***](http://www.3gpp.org/3G_Specs/CRs.htm#_blank)*on using this form, see bottom of this page or look at the pop-up text over the* ⌘ *symbols.* | | | | | | | | | | | | | | | | | | | | | | |
|  | | | | | | | | | | | | | | | | | | | | | | |
| ***Proposed change affects:*** ⌘ | | | | | smart card | | **X** | terminal | | | |  | server / network entity | | | |  | |  | |  | |
|  | | | | | | | | | | | | | | | | | | | | | | |
| ***Title:*** ⌘ | | Add reader RF gate registry READER STATUS functionality | | | | | | | | | | | | | | | | | | | | |
|  | |  | | | | | | | | | | | | | | | | | | | | |
| ***Source:*** ⌘ | | SCP TEC | | | | | | | | | | | | | | | | | | | | |
|  | |  | | | | | | | | | | | | | | | | | | | | |
| ***Work item name:*** ⌘ | | TEI 11 | | | | | | | | | |  | | ***Date:*** ⌘ | | 11/07/2014 | | | | | | |
|  | |  | | | |  | | | | | | | |  | |  | | | | | | |
| ***Category:*** ⌘ | | **F** |  | | | | | | | | | | | ***Release:*** ⌘ | | REL-11 | | | | | | |
|  | | *Use one of the following categories:* ***F*** *(essential correction)* ***A*** *(corresponds to a correction in an earlier release)* ***B*** *(Addition of feature),* ***C*** *(Functional modification of feature)* ***D*** *(Editorial modification)*  Detailed explanations of the above categories can be found in 3GPP TR 21.900. | | | | | | | | | | | | | Use one of the following releases: REL-6 (Release 6) REL-7 (Release 7)  REL-8 (Release 8)  REL-9 (Release 9)  REL-10 (Release 10)  REL-11 (Release 11) | | | | | | | |
| TC SCP use of the status "Accepted" for change requests:   * At the TC SCP working group level, the status "Accepted" means that a change request is accepted for presentation to TC SCP Plenary. * At the TC SCP Plenary level, the status "Accepted" means that a change request is accepted for implementation in the targeted specification. | | | | | | | | | | | | | | | | | | | | | | |
|  | |  | | | | | | | | | | | | | | | | | | | | |
| ***Reason for change:*** ⌘ | | | SCP(12)000124 and SCP(12)000272 introduce new reader RF registry entries in 102 622 not accessible from Applets using the API defined in TS 102 705 | | | | | | | | | | | | | | | | | | | |
|  | | |  | | | | | | | | | | | | | | | | | | | |
| ***Summary of change:*** ⌘ | | | Define functionality to allow an Applet to enable Reader Status events and be called by HCI Framework upon reception of HCI event EVT\_READER\_STATUS. | | | | | | | | | | | | | | | | | | | |
|  | | |  | | | | | | | | | | | | | | | | | | | |
| ***Consequences if*** ⌘ ***not accepted:*** | | | EVT\_READER\_STATUS information not accessible for Applets using TS 102 705 | | | | | | | | | | | | | | | | | | | |
|  | | |  | | | | | | | | | | | | | | | | | | | |
| ***New tag value defined within the CR?*** | | |  | If ticked, add document number of related CR to TS 101 220: | | | | | | | | | |  | | | | | | | | |
|  | | |  | | | | | | | | | | | | | | | | | | | |
| ***Clauses affected:*** ⌘ | | | 4.3  102705\_Annex\_A\_Java\uicc\hci\services\readermode\ReaderListener.java  102705\_Annex\_A\_Java\uicc\hci\services\readermode\ReaderMessage.java  102705\_Annex\_A\_Java\uicc\hci\framework\HCIService.java | | | | | | | | | | | | | | | | | | | |
|  | | |  | | | | | | | | | | | | | | | | | | | |
| ***Other specs*** ⌘ | | |  | Other core specifications ⌘ | | | | | | |  | | | | | | | | | | | |
| ***Affected:*** | | |  | Test specifications | | | | | | |  | | | | | | | | | | | |
|  | | |  |  | | | | | | |  | | | | | | | | | | | |
| ***Other comments:*** ⌘ | | |  | | | | | | | | | | | | | | | | | | | |

## […]

## 4.3 Reader Mode

The functionality to support the reader mode is provided in the package *uicc.hci.services.reader.* In reader mode the communication technologies defined by the contactless platform for reader mode [4] are supported. The Contactless Framework shall bind the services defined in *uicc.hci.services.reader* to the corresponding resources (e.g. gates and pipes) defined by the contactless platform for reader mode [4].

An Applet has to be in the selectable state (according to the Java Card™ specification [9], [10] and [11]) to act as a contactless Applet in reader mode.

Reader mode Applets shall follow the extended lifecycle model that is defined in "GlobalPlatform Amendment C" [8] for contactless Applets in card emulation mode (i.e. following Application Availability States and the related transition rules).

Per RF technology there shall be only one reader mode Applet in the state ACTIVATED (according to "GlobalPlatform Amendment C" [8]) at any time.

The installation parameters for contactless reader mode applications are specified in TS 102 226 [7].

When the state of a reader mode Applet changes to lifecycle ACTIVATED (according to "GlobalPlatform Amendment C" [8]) the Contactless Framework shall ensure that the HCI gates and pipes are setup for the RF technologies that are supported by the reader mode Applet.

The procedures for receiving and sending messages over the contactless interface and the procedures for notifications about the reader status are described in the following clauses.

### 4.3.1 Receiving and sending messages over the contactless interface

To be able to receive and send messages over the contactless interface in reader mode the Applet shall activate the *ReaderListener.EVENT\_TARGET\_DISCOVERED*. An Applet shall only be able to activate this event or to use the restartReadermodeProcedure method if it is in lifecycle state ACTIVATED. To release the CLF control at the end of a transaction an Applet shall deactivate the *ReaderListener.EVENT\_TARGET\_DISCOVERED*.

When an Applet lifecycle state changes from ACTIVATED to DEACTIVATED the Contactless Framework shall enforce that the *ReaderListener.EVENT\_TARGET\_DISCOVERED* is deactivated.

The Contactless Framework shall request the reader mode control on the CLF by sending the HCI events EVT\_READER\_REQUESTED and EVT\_END\_OPERATION according to the state of the reader mode Applet. The EVT\_READER\_REQUESTED shall be sent by the Contactless Framework if an Applet instance activates the event *ReaderListener.EVENT\_TARGET\_DISCOVERED* and no other Applet instance has the event activated, i.e. it shall not be sent if the Contactless Framework has earlier sent an EVT\_READER\_REQUESTED due to the request from another Applet instance, which was not yet ended by an EVT\_END\_OPERATION. The HCI event EVT\_END\_OPERATION shall be sent to the CLF when an Applet instance or the Contactless Framework deactivates the event *ReaderListener.EVENT\_TARGET\_DISCOVERED*. The Contactless Framework shall resend the EVT\_READER\_REQUESTED to the CLF if another Applet instance exists with the *ReaderListener.EVENT\_TARGET\_DISCOVERED* event activated.

The Contactless Framework shall inform the Applet instance which has activated the *ReaderListener.EVENT\_TARGET\_DISCOVERED* when a target is discovered on one of the RF technologies the Applet instance is registered to with its installation parameters as specified in TS 102 226 [7].

The Contactless Framework shall ensure that the *ReaderListener.EVENT\_TARGET\_DISCOVERED* is deactivated for all Applets when access to the interface is disabled on the UICC level.

### 4.3.2 Receiving notifications about reader status

To be able to receive CLF reader status notifications the Applet shall activate the event *ReaderListener.EVENT\_READER\_STATUS*. An Applet shall only be able to activate this event if it is in lifecycle state ACTIVATED. To release the CLF reader status notifications an Applet shall deactivate the event *ReaderListener.EVENT\_READER\_STATUS*.

When the Applet lifecycle state changes from ACTIVATED to DEACTIVATED the Contactless Framework shall enforce that the event *ReaderListener.EVENT\_READER\_STATUS* is deactivated.

The Contactless Framework shall request the CLF reader status notification on the CLF by sending the HCI commands ANY\_SET\_PARAMETER(STATUS\_EVENT\_EN) with the respective value to the HCI registry of the Reader Gate (s) of the RF technologies for which the Applet instance is registered according to its installation parameters, according to rules below:

* HCI ANY\_SET\_PARAMETER(STATUS\_EVENT\_EN) command with the value 1 shall be sent when an Applet instance activates the event *ReaderListener.EVENT\_READER\_STATUS*.
* HCI ANY\_SET\_PARAMETER(STATUS\_EVENT\_EN) command with the value 0 shall be sent when an Applet instance or the Contactless Framework deactivates the event *ReaderListener.EVENT\_READER\_STATUS*.

The Contactless Framework shall notify the Applet instance which has activated the event *ReaderListener.EVENT\_READER\_STATUS* as described above when the reader status has changed for the respective RF technology.

The Contactless Framework shall ensure that the event *ReaderListener.EVENT\_READER\_STATUS* is deactivated for all Applet instances when access to the interface is disabled on the UICC level.

# public interface ReaderListener extends HCIListener {

# […]

# /\*\*

# \* This value is used to notify the Applet when the HCI event EVT\_TARGET\_DISCOVERD

# \* has been received from the CLF.

# \* This constant can be used in HCIService.activateEvent() and HCIService.deactivateEvent().

# \* \*/

# public final static byte EVENT\_TARGET\_DISCOVERED = (byte) 0x50;

# /\*\*

# \* This value is used to notify the Applet when the HCI event EVT\_READER\_STATUS

# \* has been received from the CLF.

# \* This constant can be used in HCIService.activateEvent() and HCIService.deactivateEvent().

# \* \*/

# public final static byte EVENT\_READER\_STATUS = (byte) 0x51;

# }

# package uicc.hci.services.readermode;

# […]

# /\*\*

# \* The ReaderMessage offers specialized methods for generating

# \* Contactless Reader commands or events.

# \*\*/

# public interface ReaderMessage extends HCIMessage { // ------------ Type A parameters

# […]

# /\*\*

# \* when there are several targets in the field irrespective

# \* of the RF technology

# \*/

# public final static byte MULTIPLE\_TARGET\_STATUS = (byte) 0x03;

# /\*\*

# \* This constant can be used to test the Type A availability bit

# \* (b1) in the READER\_STATUS data of the HCI event EVT\_READER\_STATUS

# \*/

# public final static byte READER\_STATUS\_TYPE\_A\_ENABLE = (byte) 0x01;

# /\*\*

# \* This constant can be used to test the Type B availability bit

# \* (b2) in the READER\_STATUS data of the HCI event EVT\_READER\_STATUS

# \*/

# public final static byte READER\_STATUS\_TYPE\_B\_ENABLE = (byte) 0x02;

# }

# package uicc.hci.framework;

# [...]

# /\*\*

public interface HCIService {

[...]

/\*\*

\* This method is used to activate an event.

[...]

\* <li>for ReaderService:

\* <ul>

\* <li><code>ReaderListener.EVENT\_GET\_PARAMETER\_RESPONSE</code>,

\* <li><code>ReaderListener.EVENT\_WRITE\_EXCHANGE\_DATA\_RESPONSE</code>,

\* <li><code>ReaderListener.EVENT\_TARGET\_DISCOVERED</code>.

\* <li><code>ReaderListener.EVENT\_READER\_STATUS</code>.

\* </ul>

[...]

\*/

public void activateEvent(byte event) throws HCIException;

/\*\*

\* This method is used to deactivate an event from the list of activated events.

[...]

\* <li>for ReaderService:

\* <ul>

\* <li><code>ReaderListener.EVENT\_GET\_PARAMETER\_RESPONSE</code>,

\* <li><code>ReaderListener.EVENT\_WRITE\_EXCHANGE\_DATA\_RESPONSE</code>,

\* <li><code>ReaderListener.EVENT\_TARGET\_DISCOVERED</code>.

\* <li><code>ReaderListener.EVENT\_READER\_STATUS</code>.

\* </ul>

[...]

\*/

public void deactivateEvent(byte event) throws HCIException;

/\*\*

\* return the activation state of an event.

[...]

\* <li>for ReaderService:

\* <ul>

\* <li><code>ReaderListener.EVENT\_GET\_PARAMETER\_RESPONSE</code>,

\* <li><code>ReaderListener.EVENT\_WRITE\_EXCHANGE\_DATA\_RESPONSE</code>,

\* <li><code>ReaderListener.EVENT\_TARGET\_DISCOVERED</code>.

\* <li><code>ReaderListener.EVENT\_READER\_STATUS</code>.

\* </ul>

[...]

\*/

public boolean getEventNotificationStatus(byte event) throws HCIException;

}