

#7041 [TP-J560XDn]XJMF communication between controller and UW

Change log

Rev.	Date	Author	Details
1	2021/12/20	GCS	Created

Target System

[TP-J560XDn]V3.??JXDxxx

Note for diagrams

Note for colors in diagram
Updated source code
New source code
Old source code
Deleted source code

200. JetDrive

The behaviors of 201 to 208 below work only when the key below is 1.

[File name] PrinterDescriptor.ini

[Section name] OPTION

[Key name] UW_CONNECT_FUNCTION

The default value of the above key is 0.

201. Check the startup of the HTTP communication service program.

1. Description

If UWandRW_Receiver.exe is not started when the controller is started, the following warning message dialog is displayed.

(Ja)前後装置の通信サービスプログラムが起動していません。

(En) The communication service program of the front and rear devices has not started.

2. Solution

- Add resource into strings_UnwinderManager.ini file to display UW status warning dialog

Resource\English\strings_UnwinderManager.ini

```
[MSG]
IDS_NOTIFY_RECEIVER_STATUS = The communication service program of the front and rear devices
has not started.
```

Resource\Japanese\strings_UnwinderManager.ini

```
[MSG]
IDS_NOTIFY_RECEIVER_STATUS = 前後装置の通信サービスプログラムが起動していません。
```

- Add class CDataIF inherits from CMakeComposeUnwinderData.
- In function CDataIF::PIM_InitSystem(), check UW_CONNECT_FUNCTION if 1 then create a thread to run plugin main process.
- In function CDataIF::PIM_ExitSystem(), signal thread to exit.

UnwinderManager\Data_IF.h

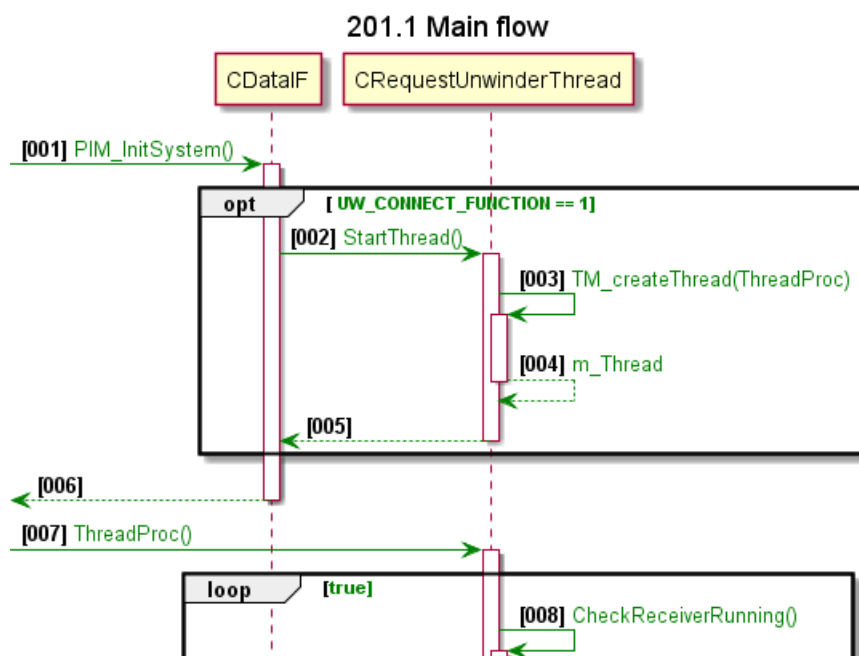
```
class CDataIF : public CBaseDataIF,
               public CMakeComposeUnwinderData
{
public:
    void Initialize();
    void Finalize();
    virtual BOOL PIM_InitSystem();
    virtual BOOL PIM_ExitSystem();
    ...
}
```

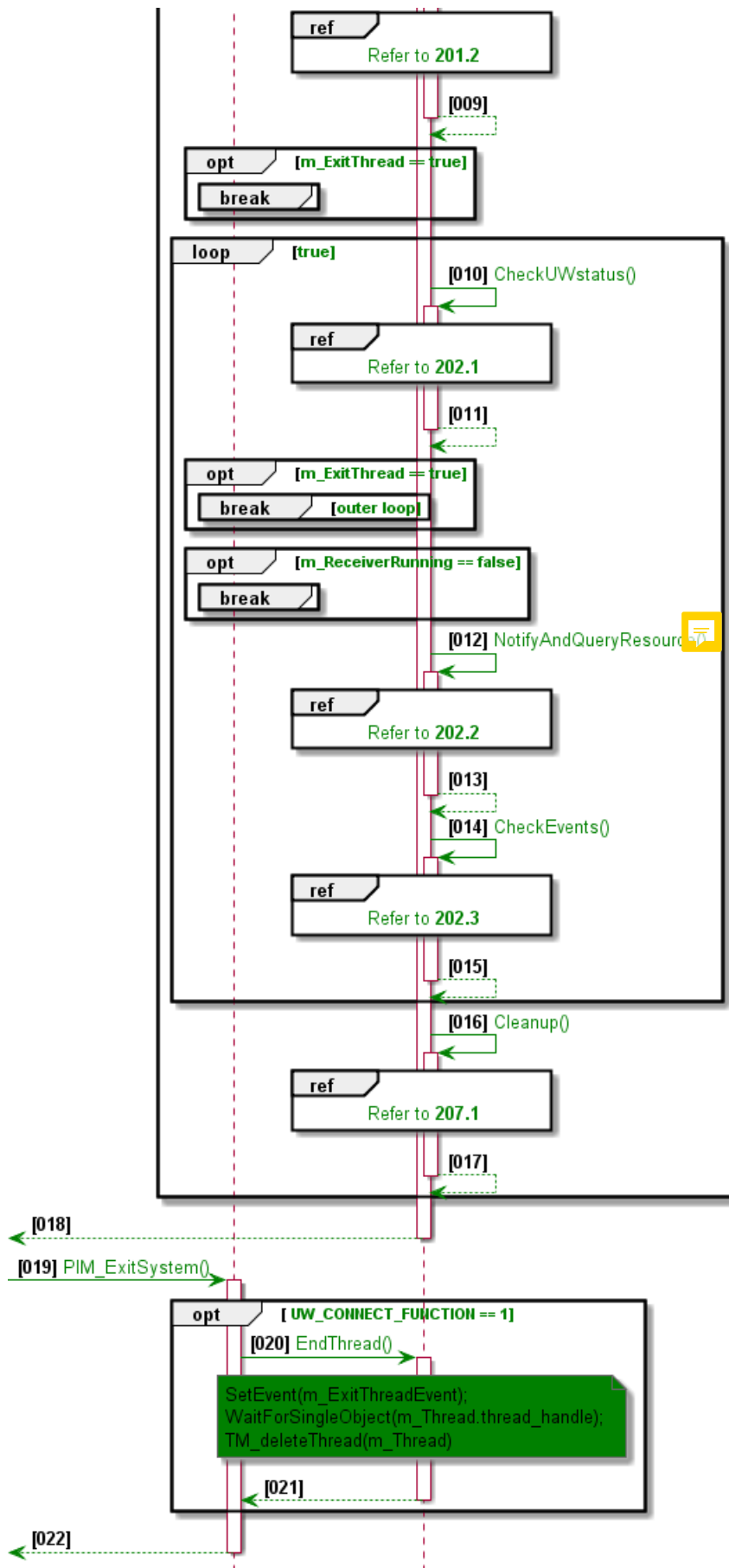
- Add class CRequestUnwinderThread() to run main process.
- At first, check for process “UWandRW_Receiver.exe” running, if not then display warning dialog.
- Add loop to wait until “UWandRW_Receiver.exe” running to continue process.

UnwinderManager\RequestUnwinderThread.h

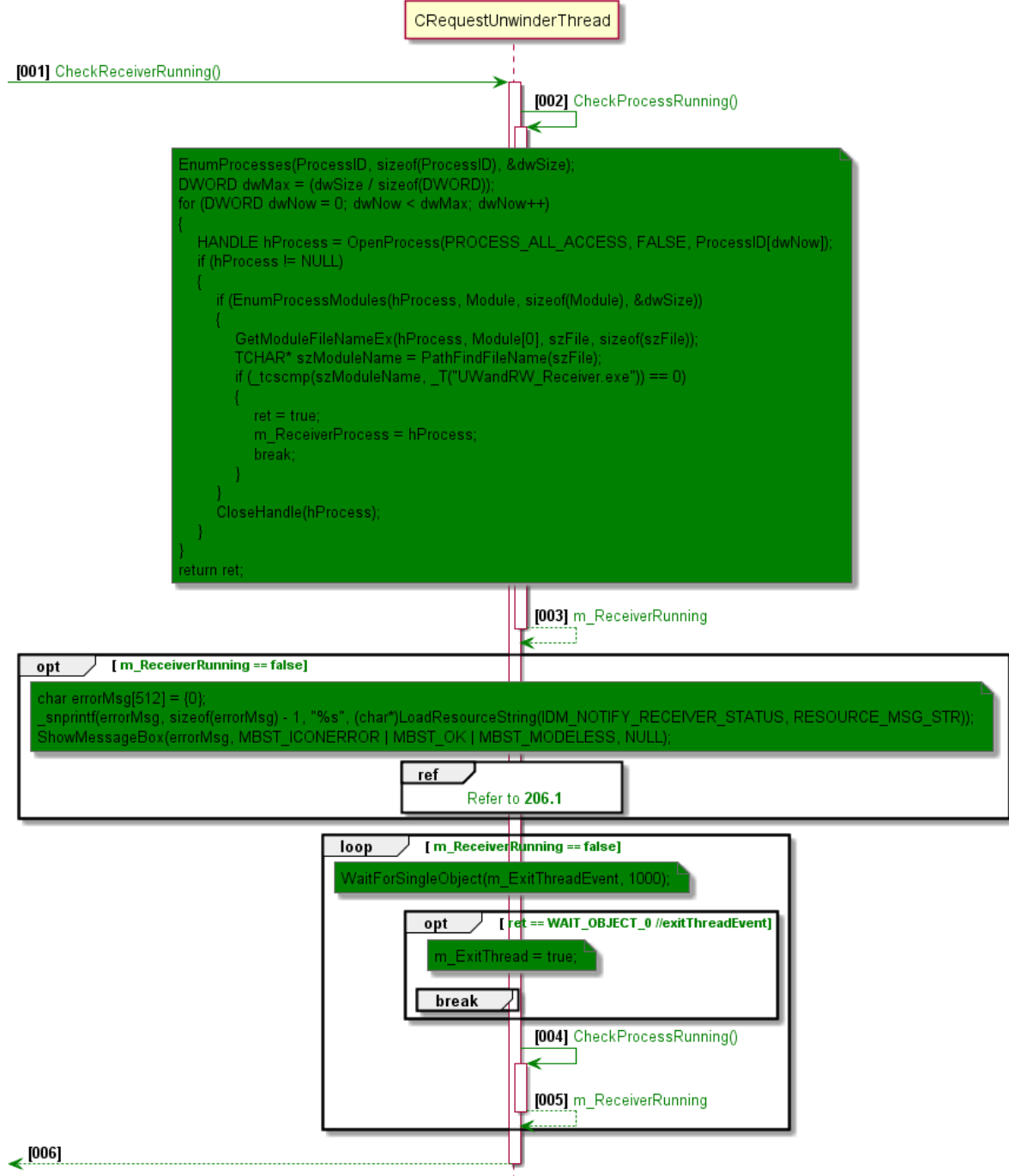
```
class CRequestUnwinderThread
{
public:
    CRequestUnwinderThread();
    virtual ~CRequestUnwinderThread();
    void Initialize(CMakeComposeUnwinderData* unwinderData);
    void Finalize();
    void StartThread();
    void EndThread();
    ...
private:
    void ThreadProc();
    void CheckReceiverRunning();
    CRequestUnwinder m_RequestUnwinder;
    ST_THREAD_INFO m_Thread;
    HANDLE m_ExitThreadEvent;
    HANDLE m_ReceiverProcess;
    bool m_ExitThread;
    bool m_ReceiverRunning;
    ...
}
```

3. Detail implementation





201.2 Check Receiver is running



202. Communication channel registration for UW

1. Description

The controller registers the communication channel in order to acquire information from the UW. Specify the URL when registering the channel, and notify the information to that URL.

Save the response channel ID in the TP-UW_Communication.ini file.

The following two communication channels are used, and channel registration assumes that UW is running.

A. Condition monitoring channel (Channel for the controller to get the status of UW from UW)

The channel registration timing is when the controller is started.

B. Paper information notification channel (Channel for the controller to obtain the remaining amount of paper, roll diameter, and paper thickness from UW)

The channel registration timing is set immediately after the print condition information is set in the UW from the controller and the setting result response is received from the UW.

The timing of setting the print conditions will be described in 204 below.

2. Solution

- Condition monitoring channel:
 - In function CRequestUnwinderThread::ThreadProc(), call to RequestQueryStatus().
 - If not success, update UW status (Refer to 206) and continue to call to RequestQueryStatus() until success.
 - If success, start the thread to receive signal status from UW. (refer 205.1)
 - When there is no signal status from UW (post message WM_USER_UW_NO_STATUS from timer), end the thread and repeat the process. (refer to 205.1)
- Paper information notification channel:
 - Add function CRequestUnwinderThread::NotifyAndQueryResource() and call in CRequestUnwinderThread::ThreadProc().
 - In CRequestUnwinderThread::NotifyAndQueryResource(), call to RequestQueryResource() after RequestCommandResource() (Refer to 203).
 - When there is an event (post message WM_USER_NOTIFY_QUERY_RESOURCE from other plugins), register again.

UnwinderManager\RequestUnwinderThread.h

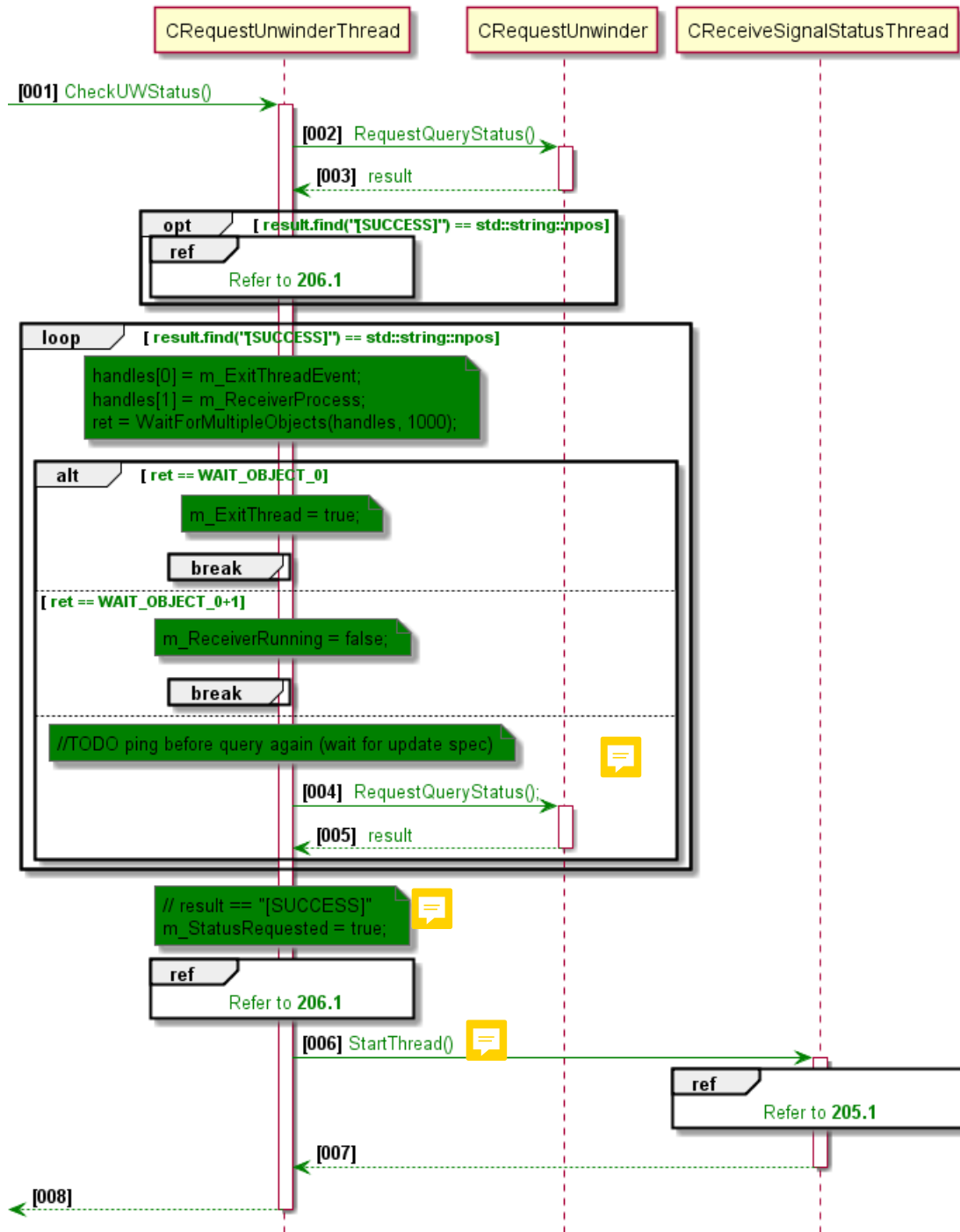
```
class CRequestUnwinderThread
{
private:
...
void CheckUWStatus();
void NotifyAndQueryResource(const std::string& inSectionId = "");
void CheckEvents();
...
bool m_StatusRequested;
bool m_ResouceRequested;
...
}
```

UnwinderManager\RequestUnwinderThread.cpp

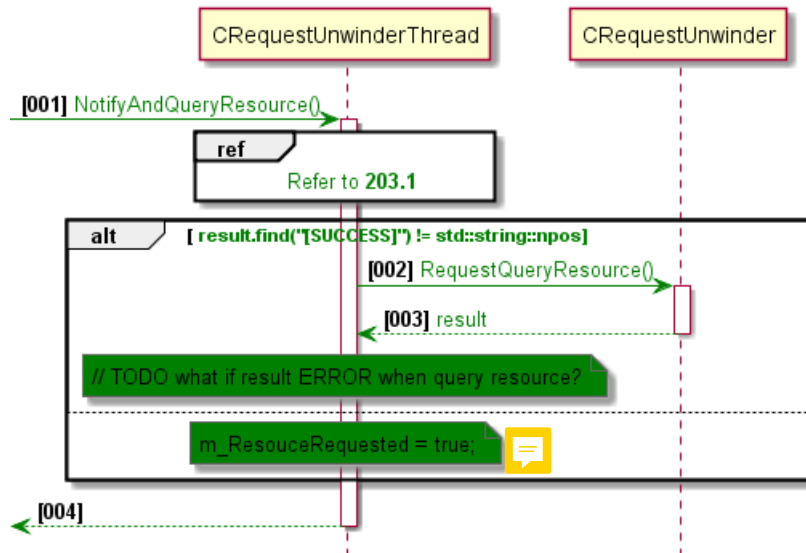
```
#define WM_USER_UW_NO_STATUS WM_USER+100
#define WM_USER_NOTIFY_QUERY_RESOURCE WM_USER+101
```

3. Detail implementation

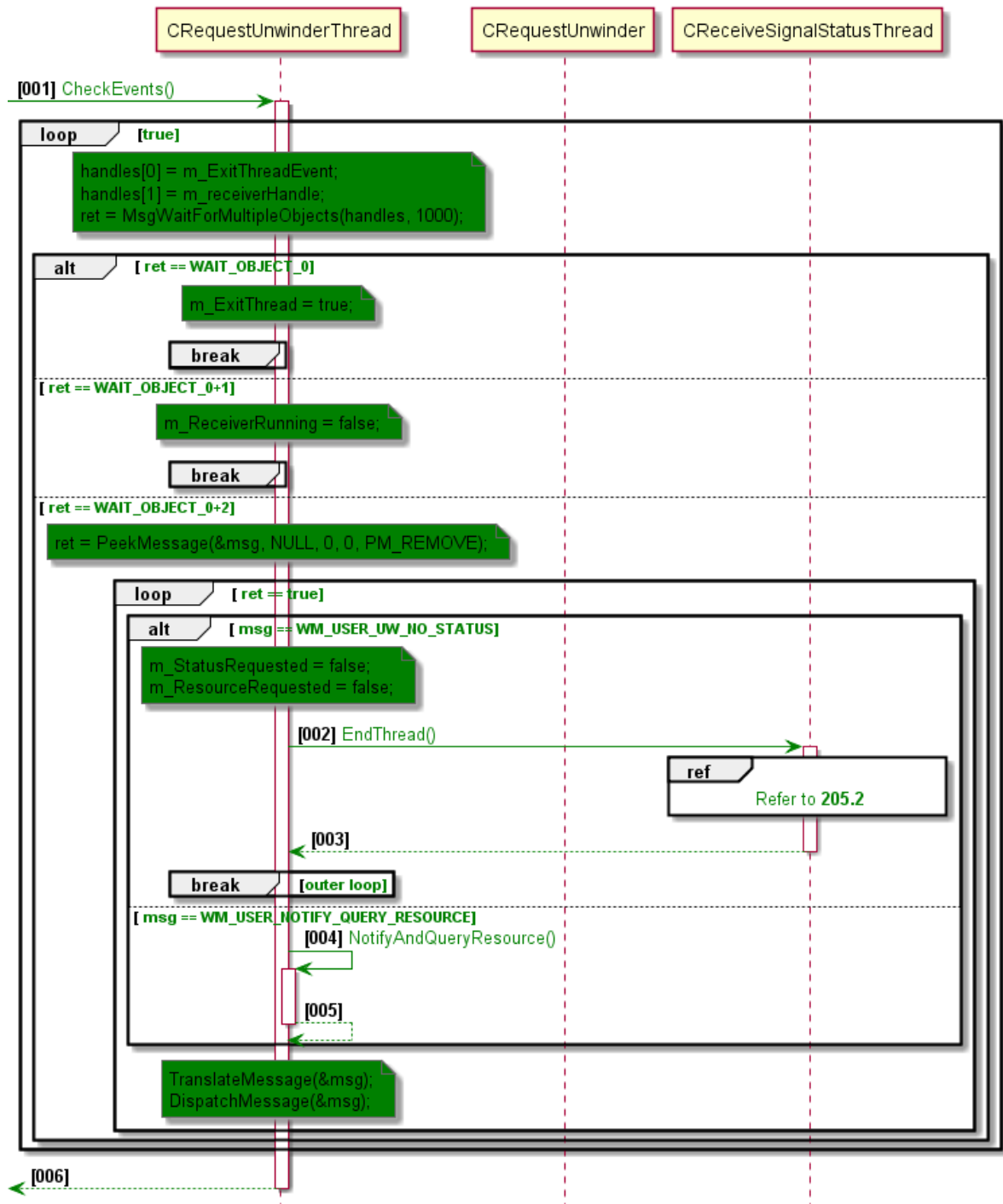
202.1 Register condition monitoring channel



202.2 Register paper information notification channel



202.3 Register channel when event happens



203. Notify UW of print condition information.

1. Description

The print conditions to be notified are as bellow.

- Print condition name(DescriptiveName)
- Media name(MediaType)
- Paper width(Dimension X point)
- Paper remaining amount(Dimension Y point)
- Paper thickness(Thickness)
- Tension(scr:UWTension)
- Print speed(scr:MaxRunSpeed)

※For tension and speed, use the calculation result using the formula described at the time of additional update.

Notify UW of printing conditions at the following timing.

1. When controller is started.(Current print condition)
2. When switching the current print condition(Current print condition)
3. When changing the current print condition setting(Current print condition)
4. When the job is running (Print conditions during job execution)

Regarding 4, in the case of continuous job printing, the content of the print conditions of the first job is notified.

2. Solution

- In CRequestUnwinderThread::NotifyAndQueryResource():
 - Depend on sectionId empty (current print condition) or not empty (job print condition), get the necessary information.
 - call to RequestCommandResource().
- Case 1 When controller is started:
Call to NotifyAndQueryResource() in CRequestUnwinderThread::ThreadProc(). (Refer to 202)
- Add plugin callbacks functions.

UnwinderManager\Plugin_IF.h

```
PLUGIN_MODULE_API bool _UnwinderManager_GetCallbacks(struct SUnwinderManager_Callbacks*
outCallbacks);
```

Common\UnwinderManager_Callbacks.h

```
typedef void (*_OnStartJobPrinting)(const std::string& inSectionId);
typedef void (*_OnSetCurrentPrintCondition)();
typedef void (*_OnUpdateCurrentPrintCondition)();
struct SUnwinderManager_Callbacks
{
    //Version 1
    DWORD StructVersion;
    _OnSetCurrentPrintCondition OnSetCurrentPrintCondition;
    _OnUpdateCurrentPrintCondition OnUpdateCurrentPrintCondition;
    _OnStartJobPrinting OnStartJobPrinting;
}
```

Common\UnwinderManager_Callbacks.h

```
class CUnwinderManager_OP : public CUnwinderManager
{
public:
    ...
    void OnSetCurrentPrintCondition();
    void OnUpdateCurrentPrintCondition();
    void OnStartJobPrinting(const std::string& inSectionId);
    ...
}
```

UnwinderManager\Data_IF.h

```
class CDataIF
{
public:
    ...
    void OnSetCurrentPrintCondition();
    void OnUpdateCurrentPrintCondition();
    void OnStartJobPrinting(const std::string& inSectionId);
    ...
}
```

- Add function CRequestUnwinderThread::MsgNotifyAndQueryResource() to notify main thread of the event.

UnwinderManager\RequestUnwinderThread.h

```

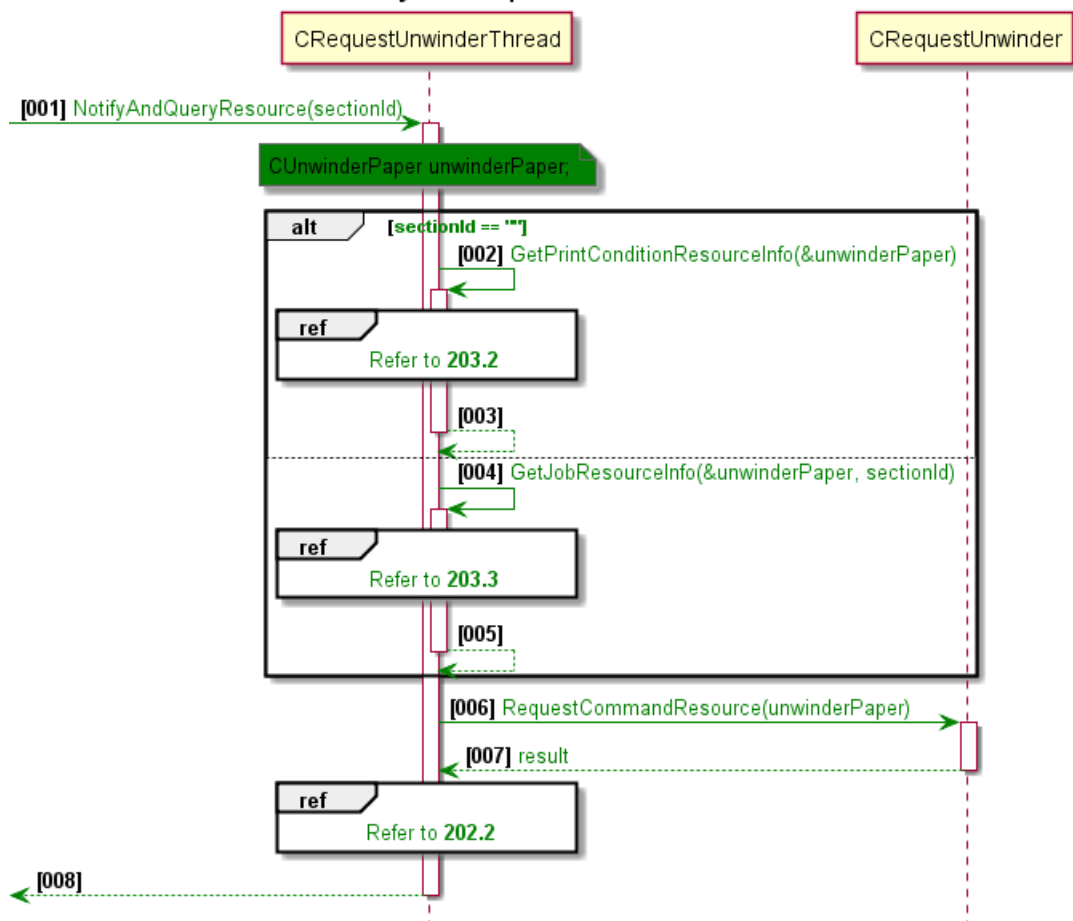
class CRequestUnwinderThread
{
public:
    ...
    void MsgNotifyAndQueryResource(const std::string& inSectionId = "");
    ...
}

```

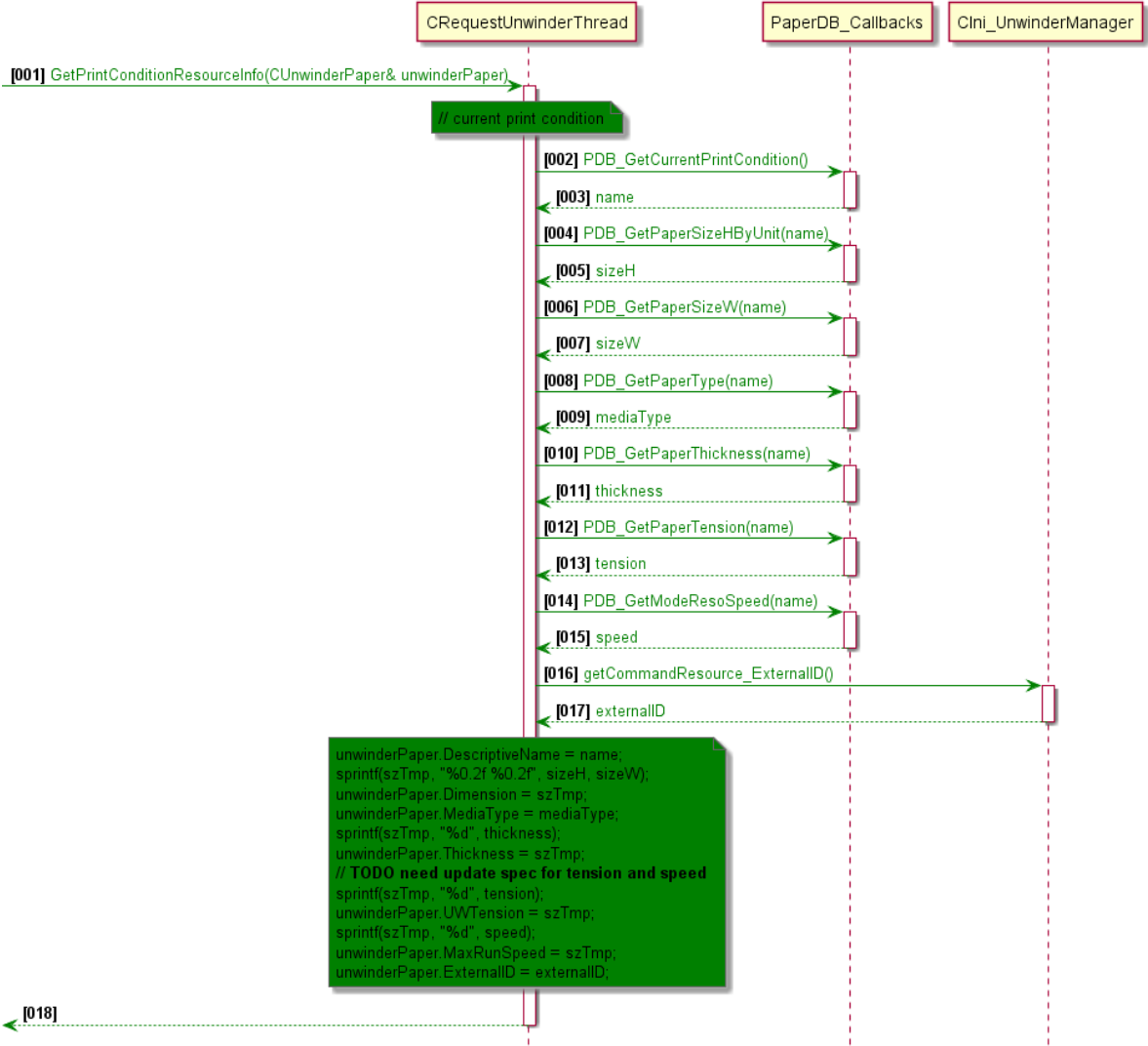
- In each callback function, call to CRequestUnwinderThread::MsgNotifyAndQueryResource().
- Case 2 When switching the current print condition:
In plugin PrintConditionGUI: call to
SUnwinderManager_Callbacks::OnSetCurrentPrintCondition() in
CDataIF::SetCurrentPrintCondition()
- Case 3 When changing the current print condition setting:
In plugin PrintConditionGUI: call to
SUnwinderManager_Callbacks::OnUpdateCurrentPrintCondition() in
CDataIF::SaveCurrentPrintCondition()
- Case 4 When the job is running:
In plugin JobPrintSequence: call to SUnwinderManager_Callbacks::_OnStartJobPrinting() in
PrintProcess::PrintProcessing()

3. Detail implementation

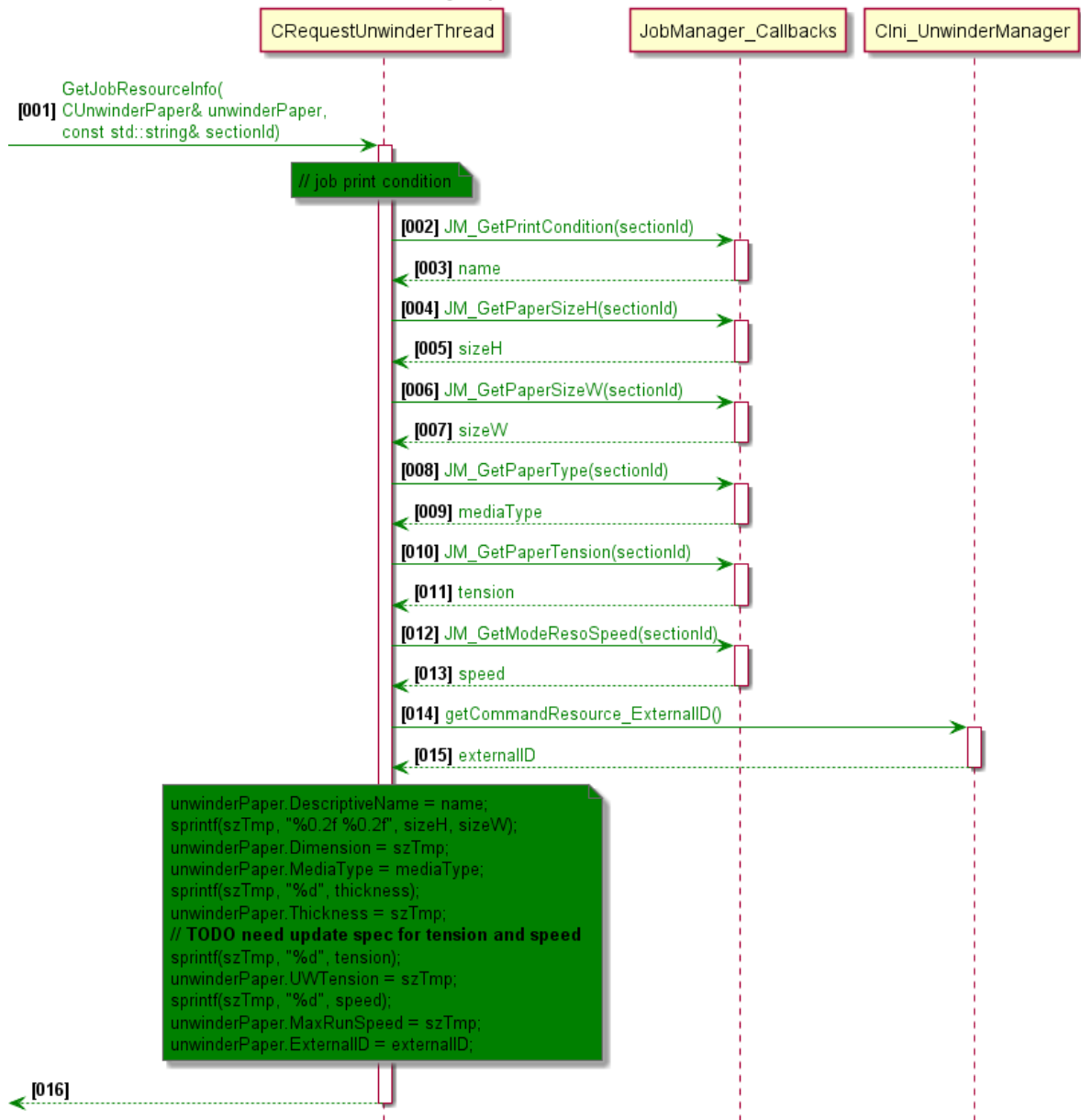
203.1 Notify UW of print condition information



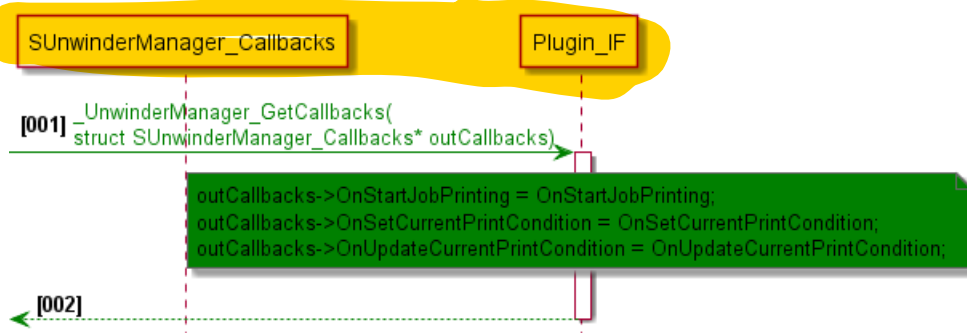
203.2 Get current print condition information



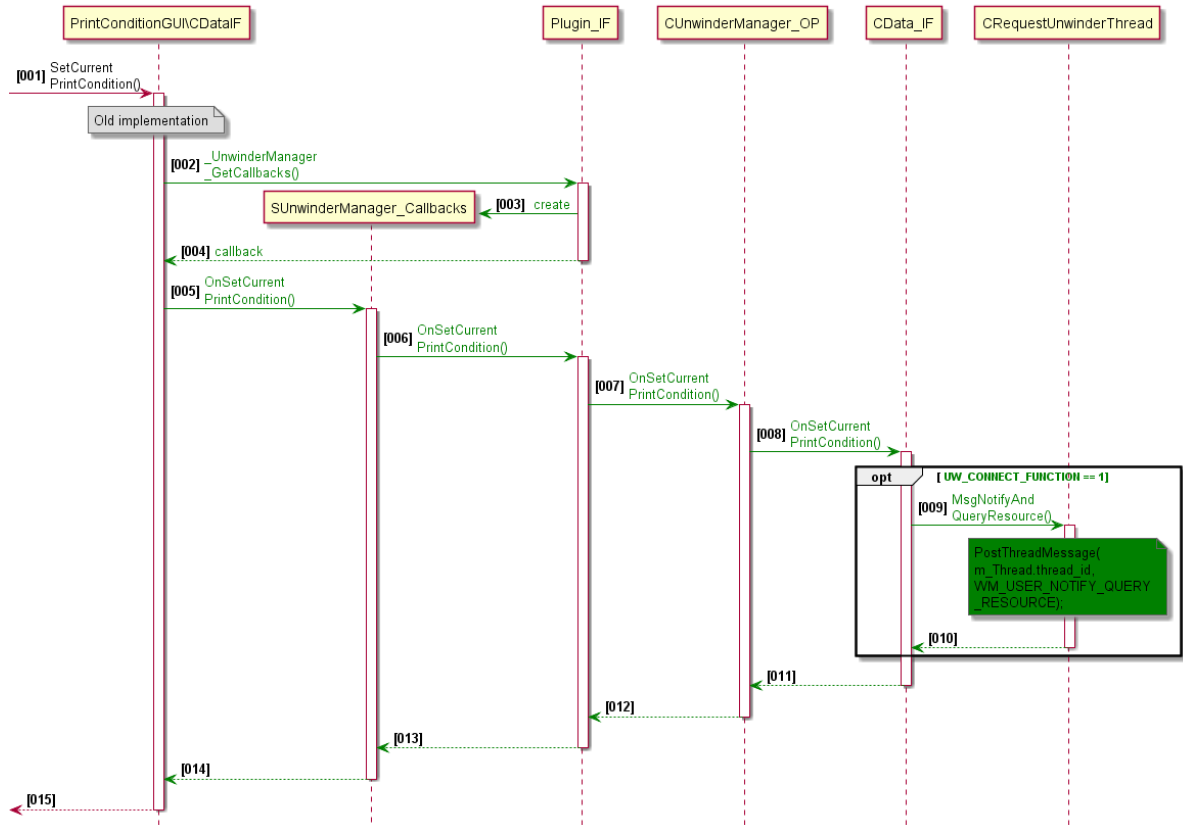
203.3 Get job print condition information



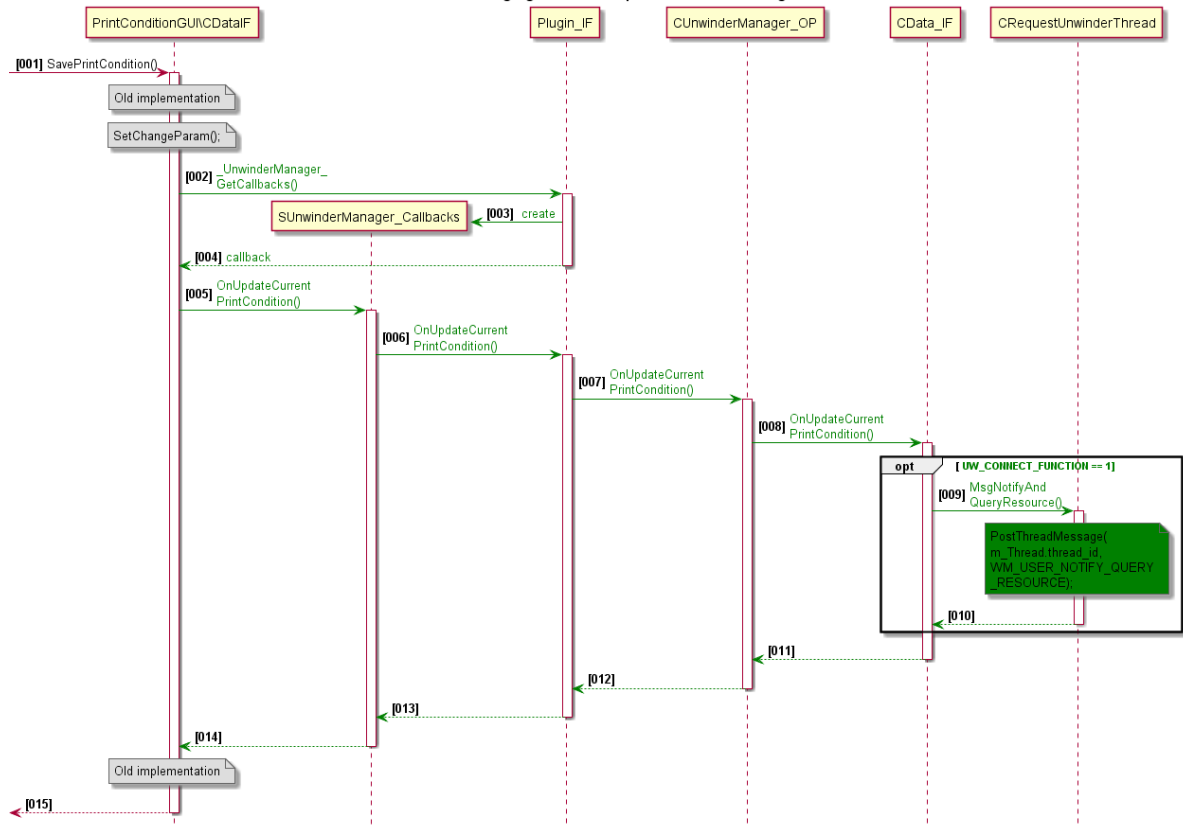
203.4 Set callback functions

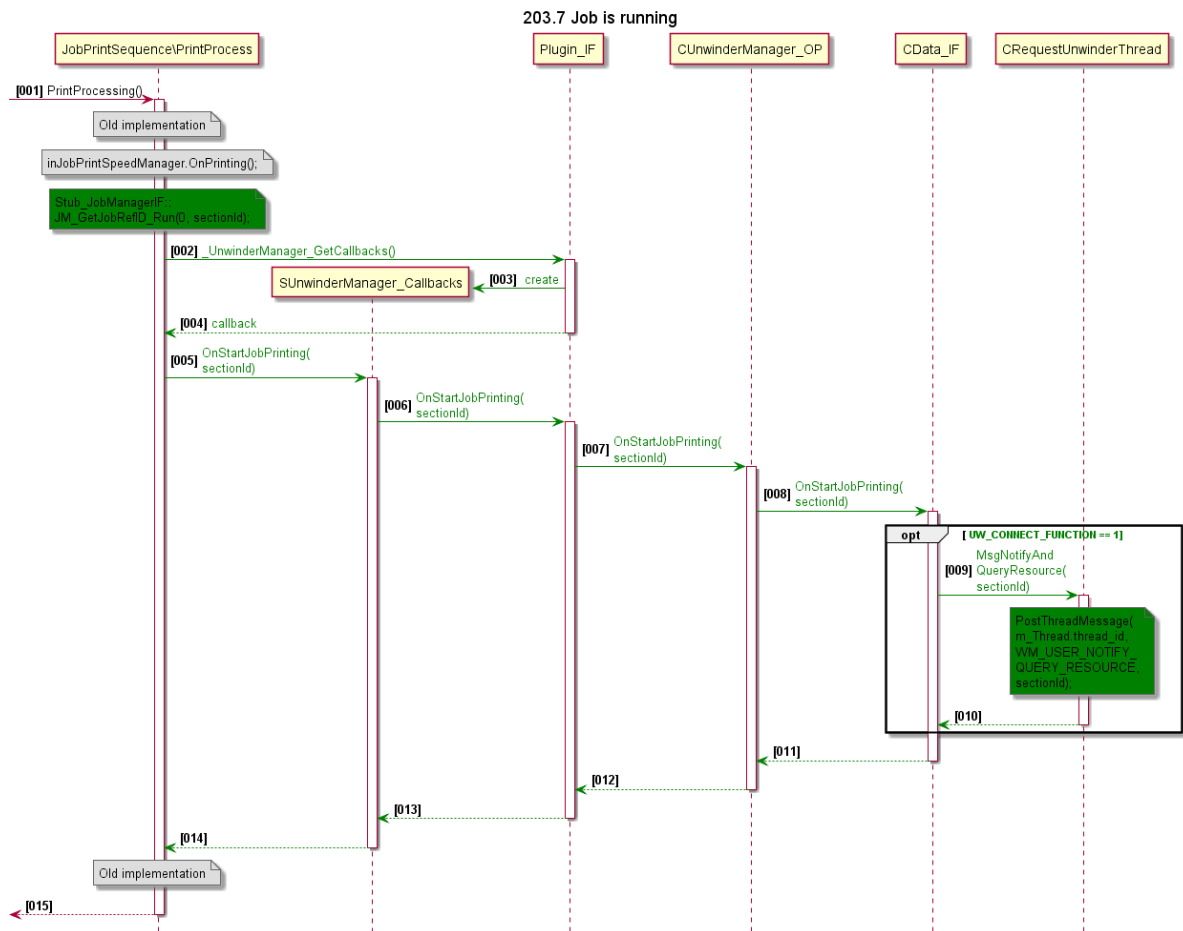


203.5 Switching the current print condition



203.6 Changing the current print condition setting





205. Reflect paper information notified from UW (Paper thickness, Roll diameter, Paper remaining amount)

1. Description

205-1. Management of roll diameter and remaining amount of paper

Save the roll diameter and remaining amount of paper in the TP-UW_Communication.ini file.

205-2. Notification timing of paper information from UW to the controller

After the transfer instruction by the controller, UW notifies the paper information at the interval specified at the time of registration of the paper information notification.

If no interval is specified, the paper information will be notified at the interval specified by UW.

However, if the UW is equipped with a paper thickness gauge and the paper thickness changes, the UW will promptly notify the controller.

The controller promptly reflects the paper information notified by UW.

The paper thickness will be reflected in the applicable printing conditions.

Update the TP-UW_Communication.ini file for the remaining amount of paper and roll diameter.

205-3. Reflect paper thickness information in printing condition.

The UW paper thickness is reflected in the paper thickness information of the current print condition or the print condition during job execution.

205-4. UW / RW icon and paper remaining amount display (※Additional update planned)

UW / RW display is added to the left and right of the printer icon to visually express the roll diameter and remaining amount (in meters).

When the remaining amount approaches the warning threshold, change the winding core from white to yellow to red.

Also, consider the UW / RW icon when the roll changer is installed.

2. Solution

- Create a class for receiving thread of the signal status information from UW:

ReceiveSignalStatusThread.h

```
class CReceiveSignalStatusThread
{
public:
    CReceiveSignalStatusThread();
    virtual ~CReceiveSignalStatusThread();
    // Start a thread
    void StartThread();
    void EndThread();
private:
    static UINT __stdcall ThreadFunction( void* pData );
    ST_THREAD_INFO m_Thread;
    HANDLE m_ExitThreadEvent;
};
```

- In function CRequestUnwinderThread::CheckUWStatus(), when receive the response from UW successfully, start a thread for receiving the signal status information.
- Create a class for receiving the signal status info and processing for the received info:

ReceiveSignalStatus.h

```
class CReceiveSignalStatus
{
public:
    CReceiveSignalStatus(CWnd* pWnd);
    virtual ~CReceiveSignalStatus();
    // Receive signal status info from UW
    void ReceiveSignalStatusInfo()
    // Receive signal status info notified from UWandRW_Receiver
    BOOL ReceiveInfo();
    // setting for timeout timer of SignalStatus(STATUS)
    void SetTimerStatusReceive();
    // setting for timeout timer of SignalStatus(PAPER)
    void SetTimerPaperReceive();
    // stop timeout timer of SignalStatus(STATUS)
    void KillTimerStatusReceive();
    // stop timeout timer of SignalStatus(PAPER)
    void KillTimerPaperReceive();
private:
    // pipe reading
    BOOL ReadData( HANDLE inPipe, char* outData, DWORD inSize );
    // analyze signal status info notified from UWandRW_Receiver
    BOOL AnalyzeData( std::string& inXmldata );
    // call UWandRW_Parse_Xml.exe to parse the receiving info
    std::string ExecuteParseXml( std::string& inSignalData );
    // check whether the data returned from UWandRW_Parse_Xml.exe is valid or not
    BOOL CheckPickupData( const std::string& inData );
    // extract data from the parsed info
    std::string SelectPickupData( const std::string& inScrData,
        const std::string& inSelectName );
    // processing when the status info is received
    BOOL ReceiveStatusInfo( const std::string& inStatus );
    // processing when the paper info is received
    BOOL ReceivePaperInfo( const std::string& inDescriptiveName,
        const std::string& inDimension,
        const std::string& inMediaType,
        const std::string& inRollDiameter,
        const std::string& inThickness );
    CWnd* m_pWnd;
    HANDLE m_ExitThreadEvent;
    HANDLE m_IOException;
```

- In function CReceiveSignalStatus::ReceivePaperInfo:
 - Stop the current timeout timer and start a new one.
 - Save the paper thickness into the print condition by using callback functions from PaperDB or JobManager.
 - Set the remaining amount of paper and roll diameter into TP-UW_Communication.ini file by new created functions: Ini_TPUWComunication::SetRollDiameter and Ini_TPUWComunication::SetPaperRemainingAmount
- In StatusBar plugin, create a new class CctlUWandRW for displaying of UW/RW icon and paper remaining amount:

```

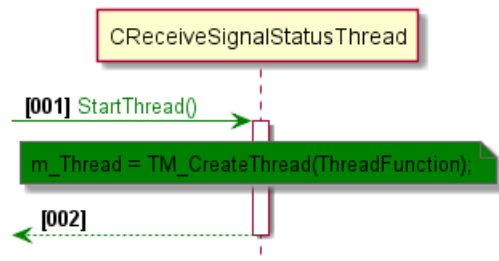
class CCtlUWandRW : public CBaseCtl
{
public:
    CCtlUWandRW();
    virtual ~CCtlUWandRW();
    virtual void OnUpdateState();
    virtual void OnUpdateValue();
    void SetPersonalInfo(long inPrinterID)
protected:
    virtual void OnSetAttribute();
private:
    long m_printerID;
};

```

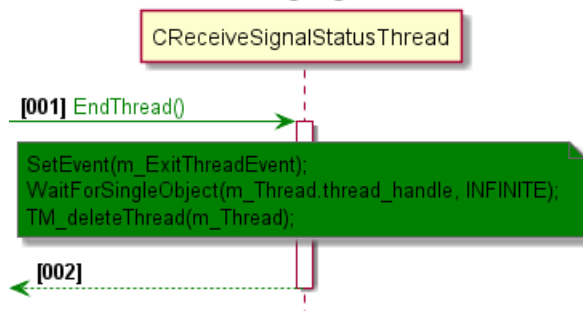
- In function CCtlUWandRW::OnUpdateValue, check the remaining amount. If it approach to the warning threshold then update displaying of the winding core (Waiting for answering of QnAs)

3. Detail implementation

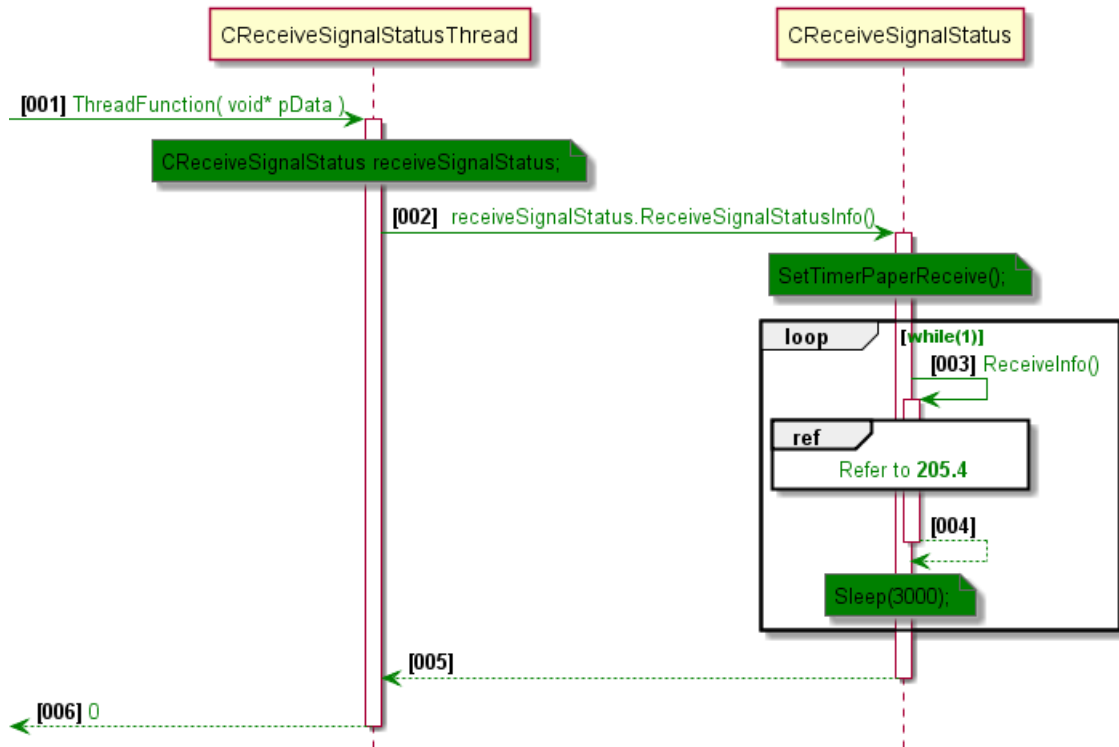
205.1 Start receiving signal status thread



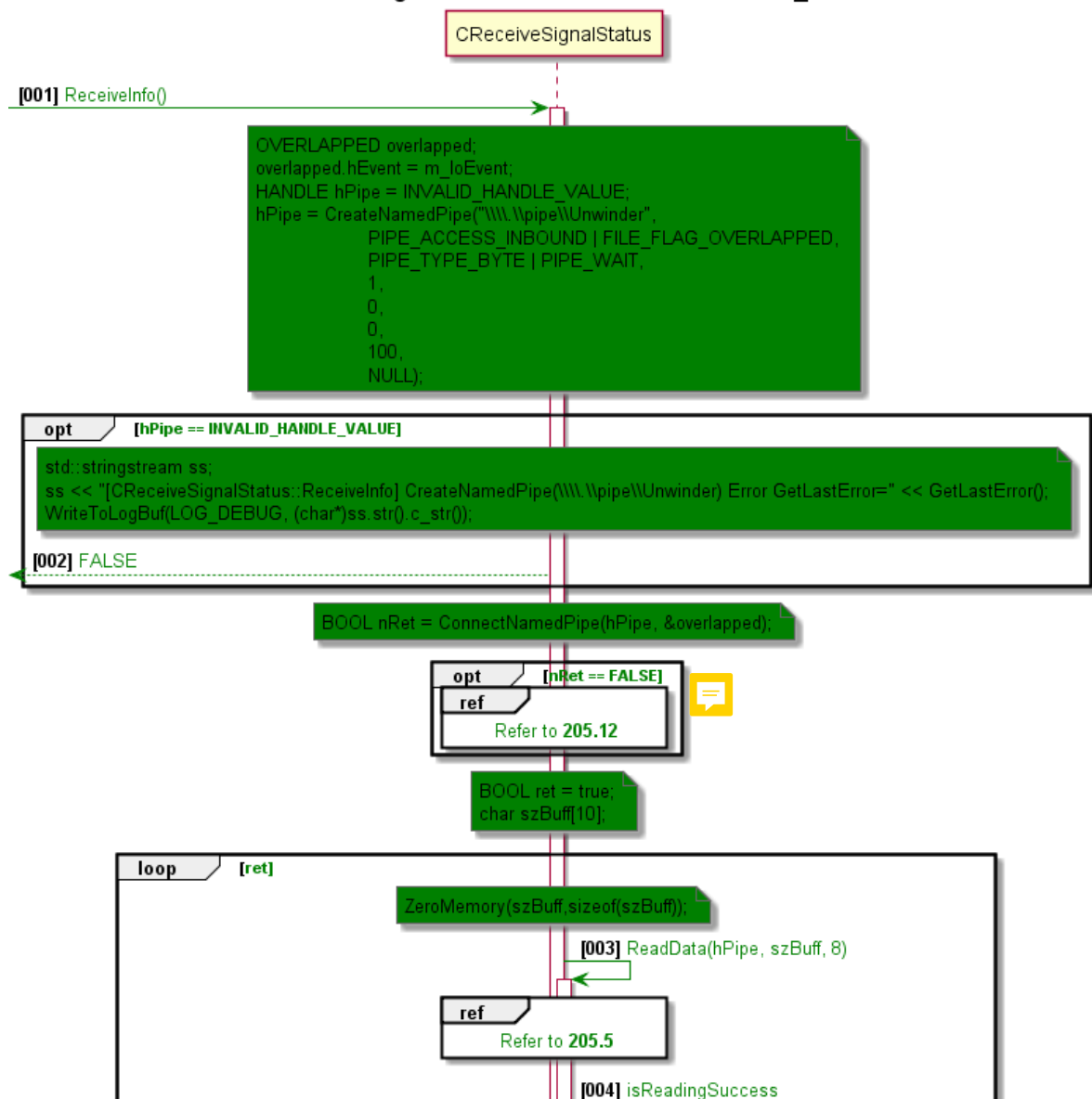
205.2 End receiving signal status thread

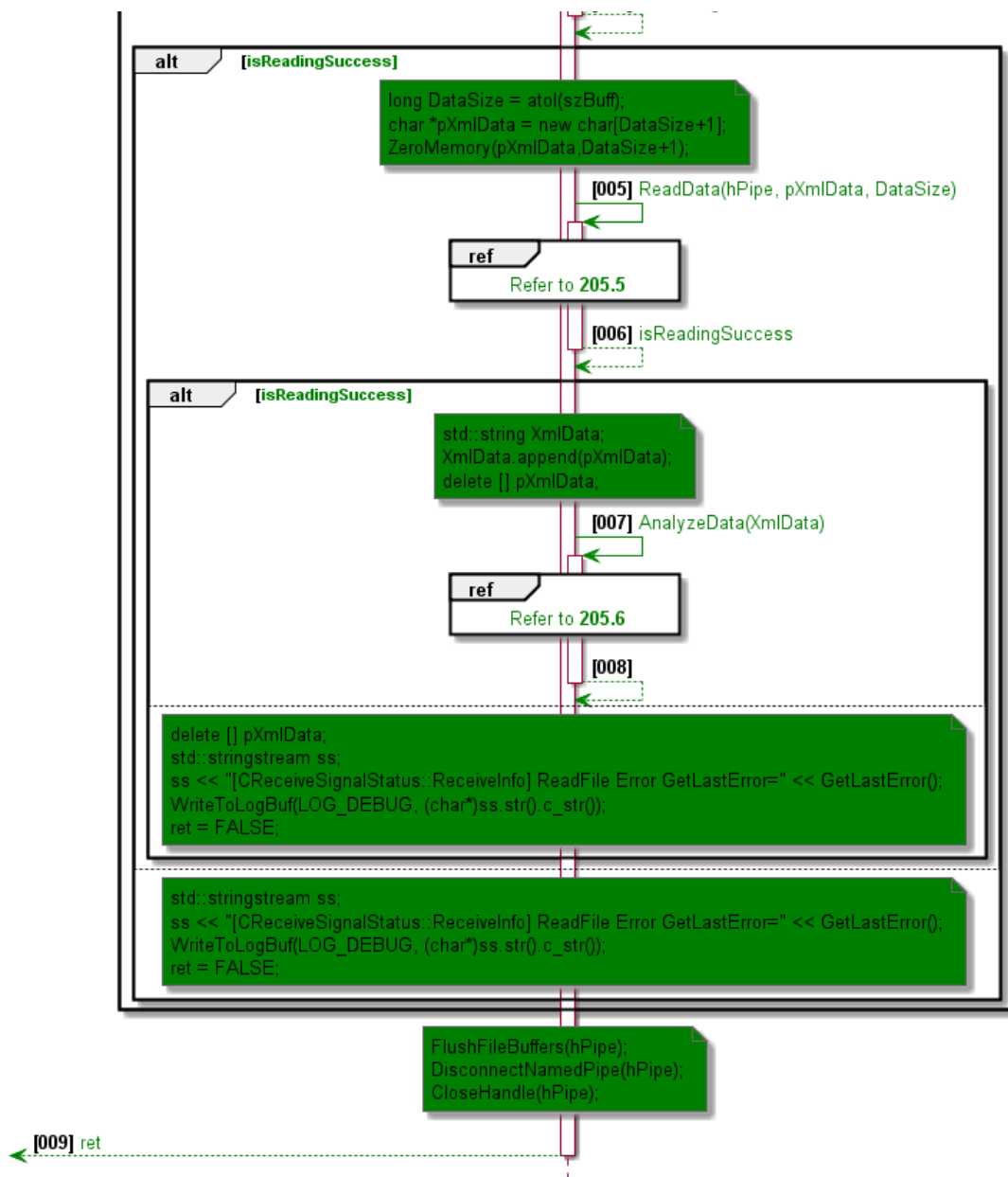


205.3 Receiving thread

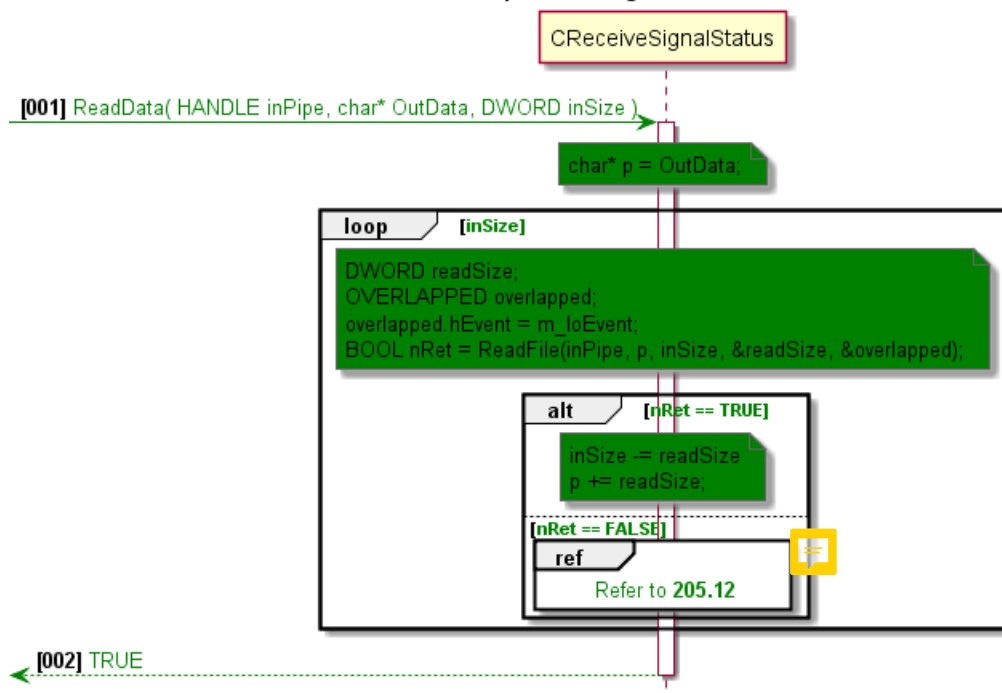


205.4 Receive signal status notified from UWandRW_Receiver





205.5 Pipe reading

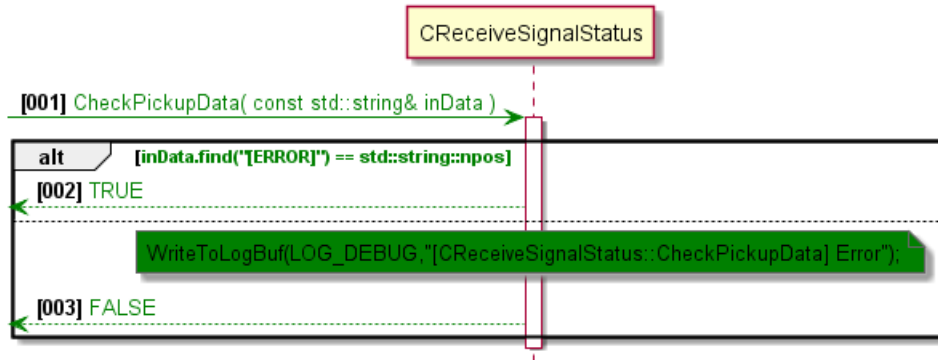


205.6 Analyze signal status info notified from UWandRW_Receiver

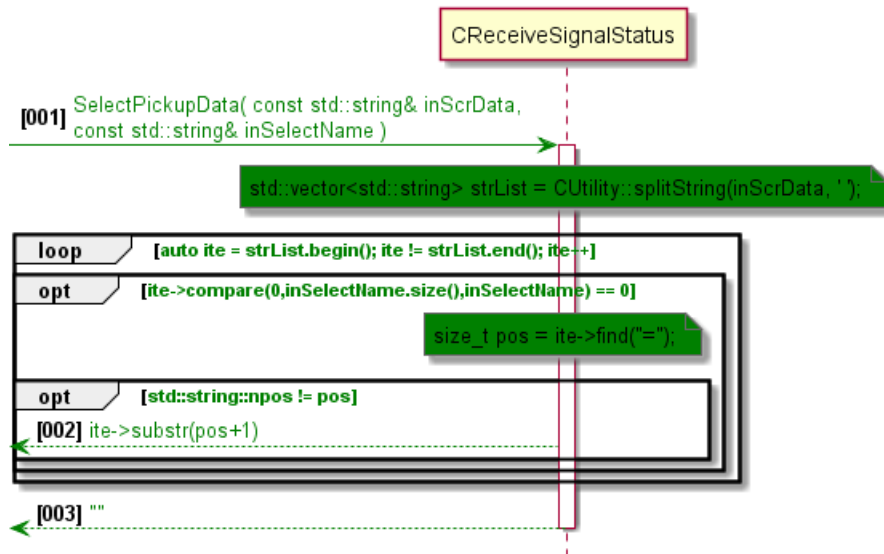




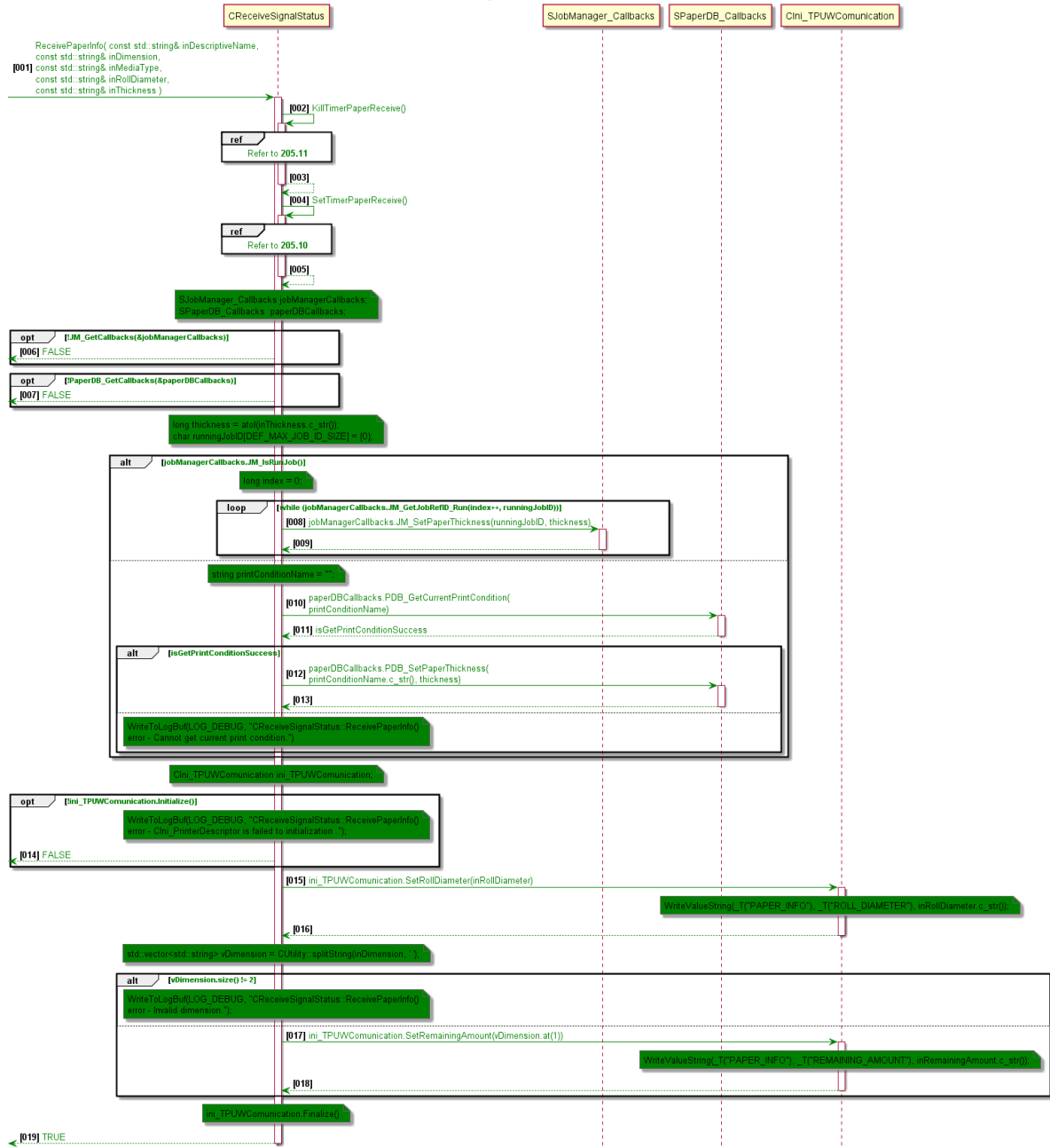
205.7 Check whether the data returned from UWandRW_Parse_Xml.exe is valid or not



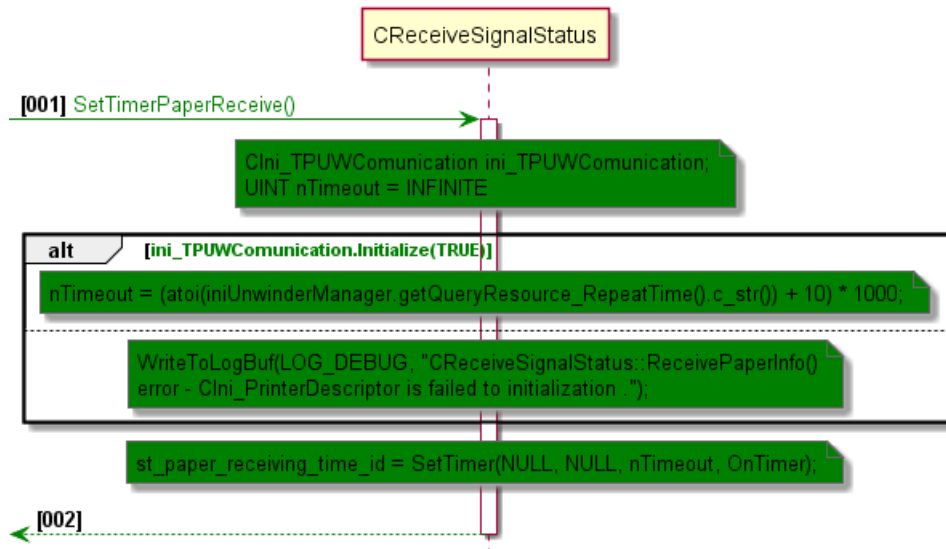
205.8 Extract data from the parsed info



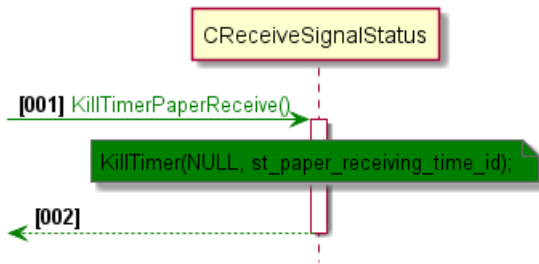
205.9 Processing when the paper info is received



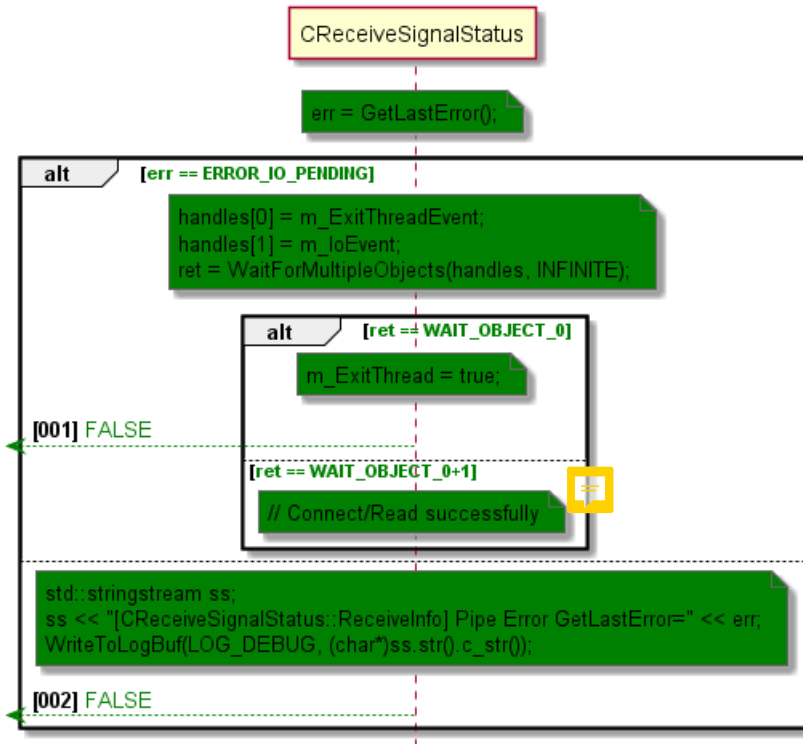
205.10 Setting for timeout timer of SignalStatus(PAPER)



205.11 Stop timeout timer of SignalStatus(PAPER)



205.12 Asynchronous pipe connecting/reading



206. Processing according to UW status

1. Description

If there is no response from the UW for the notification channel opening result after the controller registers the status monitoring channel, it is determined that the UW has not started.

Or, if there is no status notification from the UW at the interval specified by the controller when registering the status monitoring channel, it is determined that the UW has not started.

206-1. If UW is not started when the controller is started

- Display a warning icon on the status bar.
- (Ja) UWが起動していません。UWを起動してください。
- (En) A communication error with UW has occurred.
- Display the translucent UW icon.
- (※ These icons will be added and updated in relation to 205-4)

206-2. When UW ends while the controller is starting

- The following warning message dialog is displayed.
- (Ja) UWとの通信エラーが発生しました。
- (En) A communication error with UW has occurred.
- Display the translucent UW icon.
- (※ These icons will be added and updated in relation to 205-4)

206-3. When UW starts while the controller is starting

- Set the print condition information of the current print condition and register the channel for paper information notification.
 - Cancel the 206-1 warning icon display.
 - Switch from the translucent UW icon of the UW icon to the normal UW icon.
- (※ These icon will be added and updated in relation to 205-4)

2. Solution

- In file ControlErr.ini, add a warning message for the UW status.

Resource\English\ControlErr.ini

```
20004001 = A communication error with UW has occurred.[C=0][E=4]
```

Resource\Japanese\ControlErr.ini

```
20004001 = UW との通信エラーが発生しました。[C=0][E=4]
```

- In PressErrManagerDef_OP.h file, add new value in enum DEF_PRINTER_WARNING to display the warning icon.

Src\Common\PressErrManagerDef_OP.h

```
//before
enum DEF_PRINTER_WARNING{
    // ...
    DEF_PRINTER_WARNING_COMMUNICATION_DISORDER,
    DEF_PRINTER_WARNING_BARCODE_COLLATION_MISMATCH,
}

// after
enum DEF_PRINTER_WARNING{
    // ...
    DEF_PRINTER_WARNING_COMMUNICATION_DISORDER,
    DEF_PRINTER_WARNING_BARCODE_COLLATION_MISMATCH,
    DEF_PRINTER_WARNING_UW_STATUS
}
```

- Add resource into strings_UnwinderManager.ini file to display UW status warning dialog

Resource\English\strings_UnwinderManager.ini

```
[MSG]
IDS_NOTIFY_UW_STATUS = A communication error with UW has occurred.
```

Resource\Japanese\strings_UnwinderManager.ini

```
[MSG]
IDS_NOTIFY_UW_STATUS = UW との通信エラーが発生しました。
```

- In class CDataIF, create member variables and methods to get, set and handle display the UW status.

Src\UnwinderManagerDataIF.h


```
class CDataIF : public CBaseDataIF,
public CMakeComposeUnwinderData
{
public:
    /...

    //methods
    void SetUWStatus(bool inStatus);
    bool GetUWStatus();
    void UpdateDisplayUWStatus();

protected:

    // members
    BOOL m_isDED;
    bool m_UWStatus;
    bool m_displayWarningDialogEnable;
}
```

- In method CDataIF::UpdateDisplayUWStatus(), update display the warning about UW status.

- In method `CRequestUnwinderThread::CheckReceiverRunning()`, if `UWandRW_Receiver.exe` is not run, set UW status to off.
- In method `CRequestUnwinderThread::CheckUWStatus()`, if receive the response from UW successfully, set UW status is on, else, status is off and update display of the UW status.
- In method `CReceiveSignalStatus::ReceiveStatusInfo`:
 - Stop the current timeout timer and start a new one.
 - Set the UW status into `UnwinderManager_work.ini` file.
 - Set the UW status to `m_UWStatus` variable of class `CDataIF`
 - Update displaying UW status.
- In method `OnTimer(UINT_PTR nIDEvent)`:
 - In case `TIMER_STATUS_RECEIVE`, when the timer for receiving the UW status timeout, set display warning message, dialog warning message and the translucent UW icon.
 - Stop the current timer.
 - call `CRequestUnwinderThread::MsgNotifyAndQueryResource()` 
- Add method `CRequestUnwinderThread::MsgNotifyAndQueryResource()` to notify main thread.

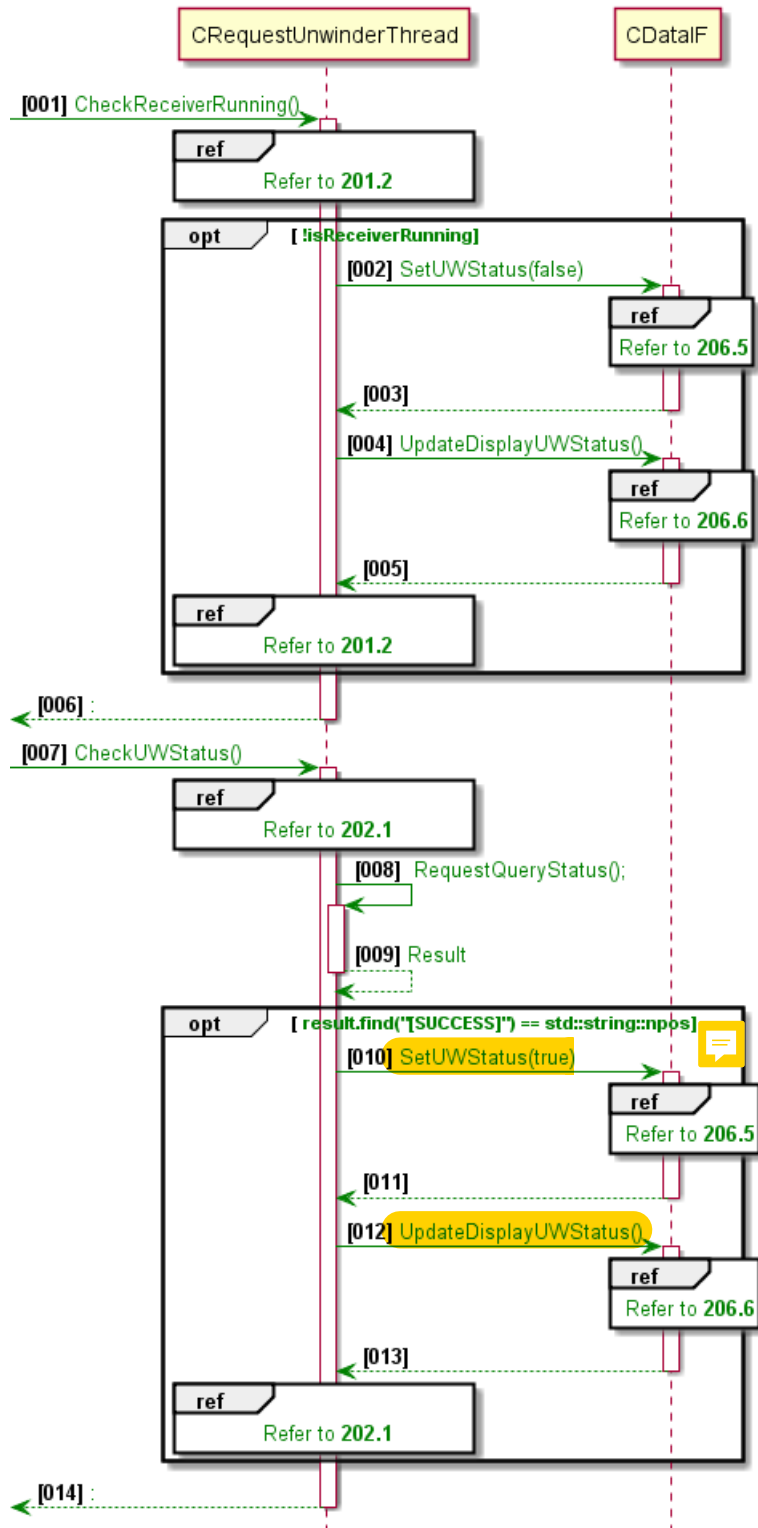
UnwinderManager\RequestUnwinderThread.h

```
class CRequestUnwinderThread
{
public:
...
void MsgNotifyAndQueryResource(const std::string& inSectionId = "");
...
}
```

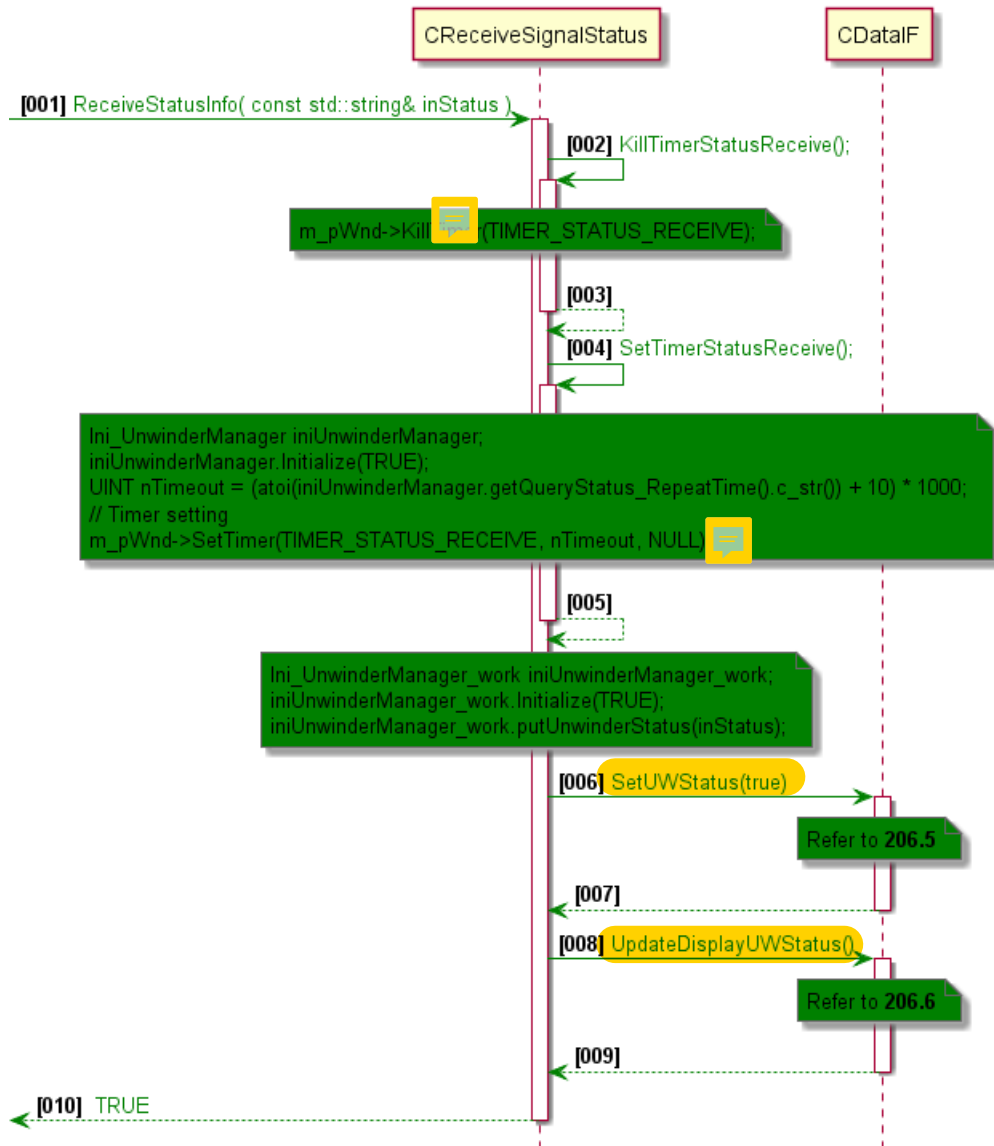
- In class `CctlUWandRW` of plugin `StatusBar`:
 - In method `OnUpdateValue()`, get UW status from plugin `UnwinderManager` and display the normal UW icon if UW status is on, else, display the translucent UW icon.

3. Detail implementation

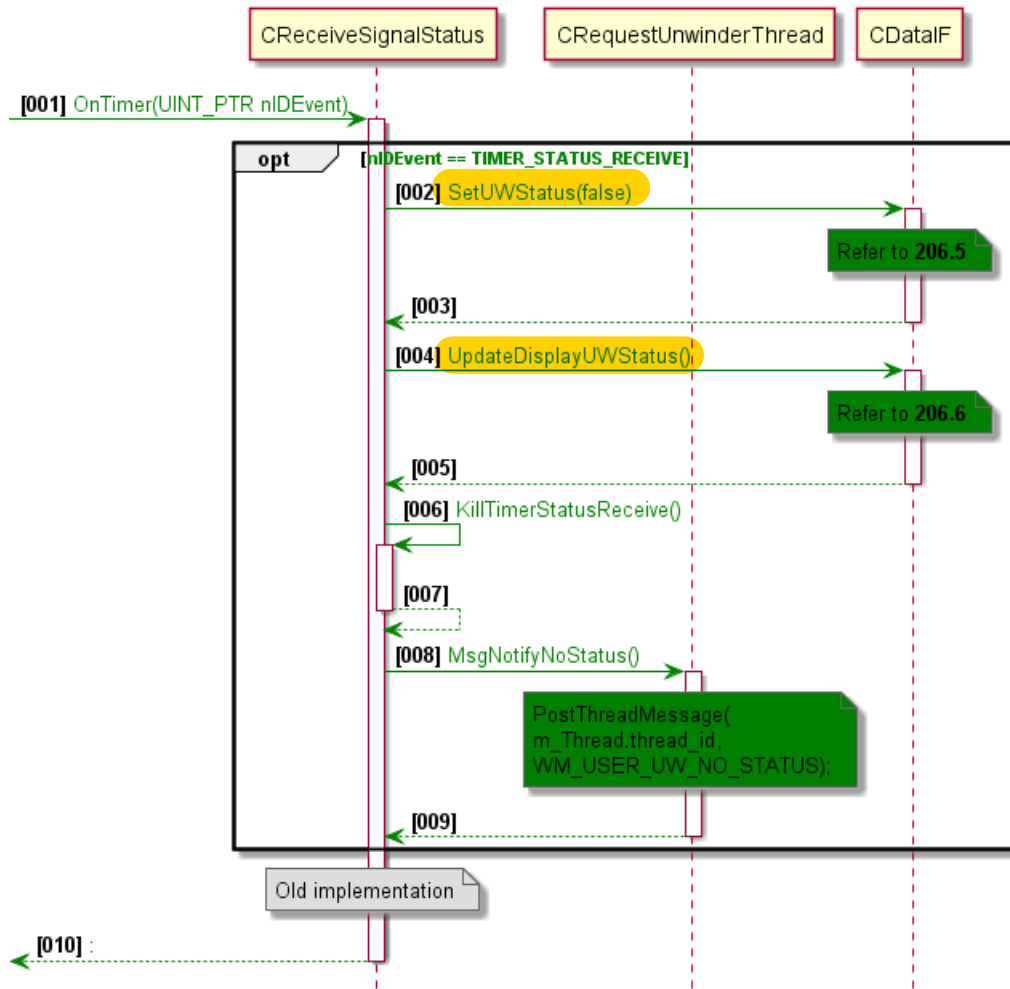
206.1 Check UW status when starting control



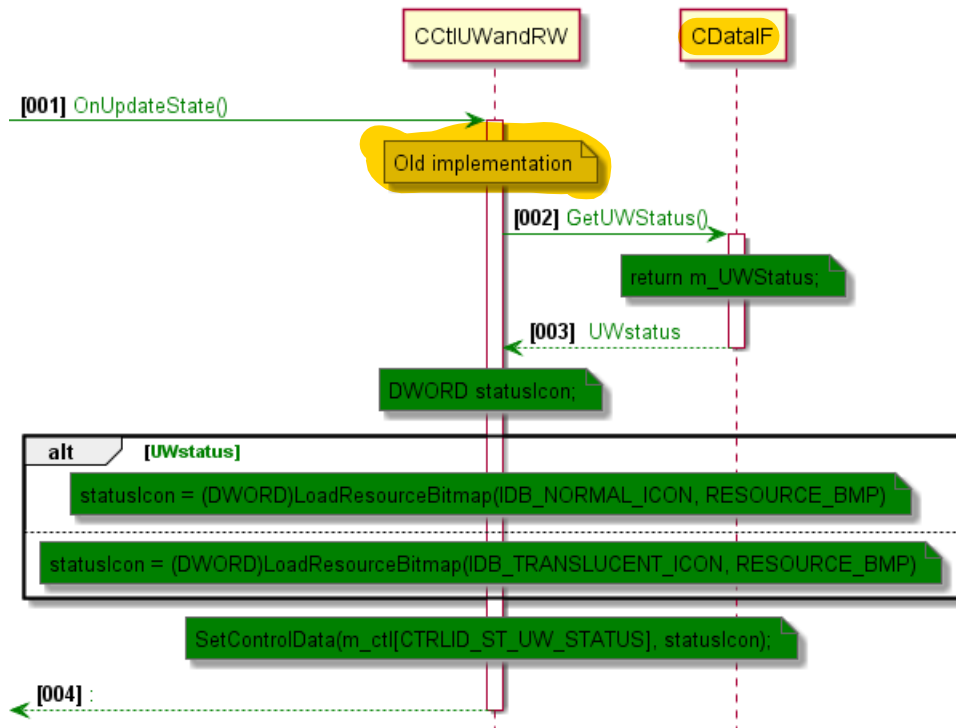
206.2 Receive status info



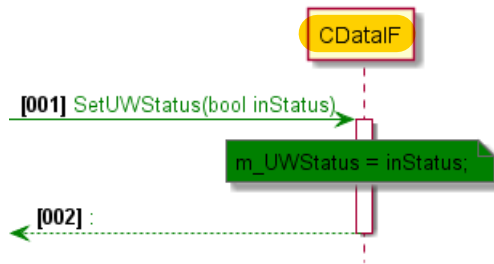
206.3 Handle when receive UW status is timeout



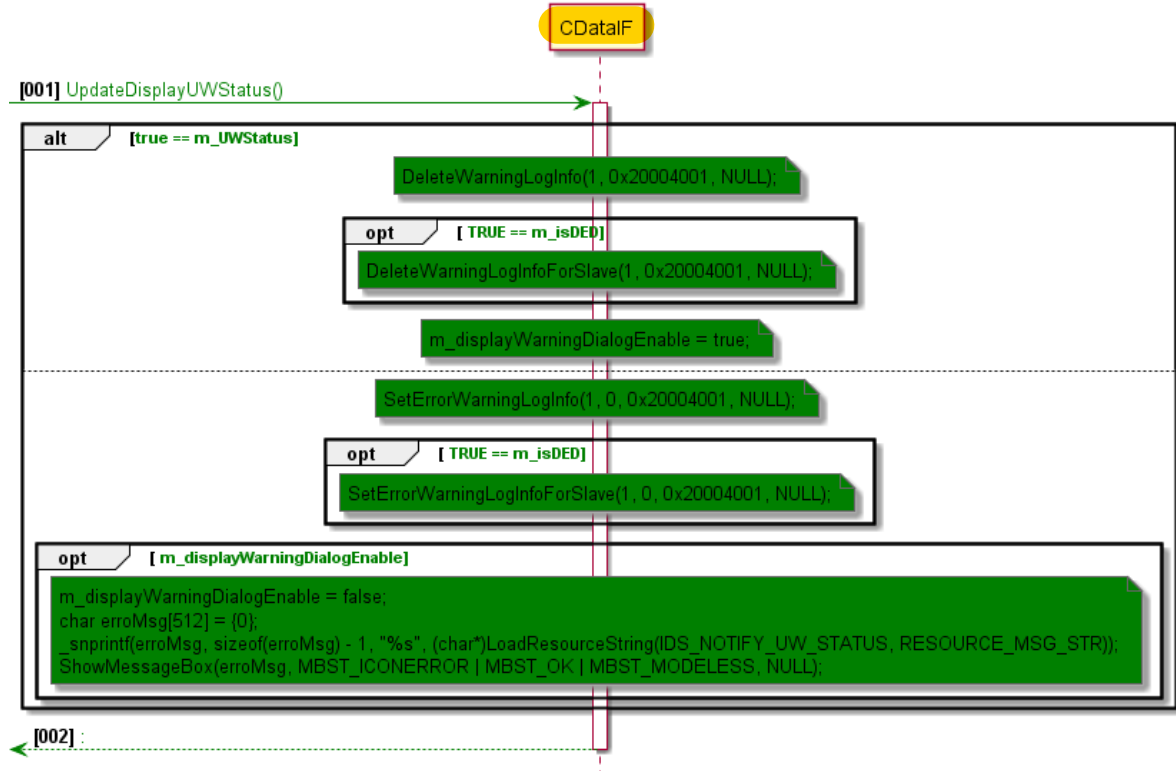
206.4 Set state of UW icon



206.5 Set UW status



206.6 Update display the UW status



207. Delete channel

1. Description

Delete channels that are no longer needed.

The communication channel is deleted by the channel ID notified in the response at the time of channel registration when the service of UWandRW_Receiver.exe is terminated.

2. Solution

- In CRequestUnwinderThread::ThreadProc(), wait for process "UWandRW_Receiver.exe" to end or m_ExitThread set. (see 201.1 Main flow)
- Call to RequestStopPersChParams() for channel which has been registered.
- If m_ExitThread not set yet, repeat the main loop. (see 201.1 Main flow)

3. Detail implementation

207.1 Delete channels

