The directory ICD has to be generated using the following command:

Console

```
getTemplateForDirectory MODROOT_WS ICD
```

Makefile

The Makefile was generated with the previous command, but the following entries need to be modified / added:

Makefile

```
#
# Generate ACS Error System classes
#
ACSERRDEF := SystemErr

#
# IDL Files and flags
#
IDL_FILES = Types Console DataBase Instrument Scheduler Telescope TelescopeControl Camera Storage
IDL_TAO_FLAGS =
USER_IDL =

TypesStubs_LIBS = acscomponentStubs
ConsoleStubs_LIBS = acscomponentStubs SYSTEMErrStubs TypesStubs
DataBaseStubs_LIBS = acscomponentStubs SYSTEMErrStubs TypesStubs
InstrumentStubs_LIBS = acscomponentStubs SYSTEMErrStubs TypesStubs
SchedulerStubs_LIBS = acscomponentStubs SYSTEMErrStubs
TelescopeStubs_LIBS = acscomponentStubs SYSTEMErrStubs TypesStubs
TelescopeControlStubs_LIBS = baciStubs acscomponentStubs SYSTEMErrStubs TypesStubs
CameraStubs_LIBS = baciStubs acscomponentStubs SYSTEMErrStubs TypesStubs
StorageStubs_LIBS = acscomponent TypesStubs
```

IDL Files

The IDL files have to be placed in the ICD/idl directory:

Types.idl

```
#ifndef _TYPES_IDL_
#define _TYPES_IDL_

/*******************************************************************************
*    ACS Community - https://github.com/ACS-Community/ACS-Workshop
*
*    This library is free software; you can redistribute it and/or
*    modify it under the terms of the GNU Lesser General Public
*    License as published by the Free Software Foundation; either
*    version 2.1 of the License, or (at your option) any later version.
*
*    This library is distributed in the hope that it will be useful,
*    but WITHOUT ANY WARRANTY; without even the implied warranty of
*    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
*    Lesser General Public License for more details.
*
*    You should have received a copy of the GNU Lesser General Public
*    License along with this library; if not, write to the Free Software
*    Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307  USA
*/

#pragma prefix "acsws"
```

```
/** @file Types.idl
 *  IDL specification of Mount object for ACS Course
 *
 *  There are 4 different interfaces that show the implementation
 *  of a Mount component with increasing complexity.
 *  At every step we add new functionality, aligned with what
 *  is demonstrated in the course
 */

module TYPES
{
    // Image types
    typedef sequence<octet> ImageType;
    typedef sequence<ImageType> ImageList;

        // Coordinates type
        struct Position {
                double az;
                double el;
        };

        // Targets types
        struct Target {
                long     tid;
                Position coordinates;
                long   expTime;  /* seconds */
        };
        typedef sequence<Target>  TargetList;

        // Proposal types
        struct Proposal {
                long pid; /* proposal ID */
                TargetList targets;
                long status; /* 0 queued, 1 running, 2 ready */
        };
        typedef sequence<Proposal>  ProposalList;

        // RGB Configuration of the CCD
        struct RGB {
                long red;
                long green;
                long blue;
        };

};

#endif
```

Console.idl

```
#ifndef _CONSOLE_IDL_
#define _CONSOLE_IDL_

/*******************************************************************************
*    ACS Community - https://github.com/ACS-Community/ACS-Workshop
*
*    This library is free software; you can redistribute it and/or
*    modify it under the terms of the GNU Lesser General Public
*    License as published by the Free Software Foundation; either
*    version 2.1 of the License, or (at your option) any later version.
*
*    This library is distributed in the hope that it will be useful,
*    but WITHOUT ANY WARRANTY; without even the implied warranty of
*    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
*    Lesser General Public License for more details.
*
*    You should have received a copy of the GNU Lesser General Public
*    License along with this library; if not, write to the Free Software
*    Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307  USA
```

```
*/

#include <acscomponent.idl>
#include <Types.idl>
#include "SystemErr.idl"

#pragma prefix "acsws"


/**
 * @file Console.idl
 * SystemErr Console IDL file
 */

module CONSOLE_MODULE
{

        /** @interface Console
         * Operator's interface to set automatic and manual modes.
         * Grants manual access to low level components.
         */
        interface Console :  ACS::ACSComponent
        {
        /**
         * Set the automatic / manual mode for the operator. Raises an exception
         * if the automatic mode is asked twice.
         *
         *  @param mode if true then automatic mode otherwise manual mode.
         *  @return void
         */
        void setMode(in boolean mode)
                raises(SystemErr::AlreadyInAutomaticEx);

        /**
         * Get the current operator's mode.
         *
         *  @return current operator's mode
         */
        boolean getMode();

        /**
         * Set the camera on.
         *
         *  @return void
         */
        void cameraOn()
                raises (SystemErr::SystemInAutoModeEx);

        /**
         * Set the camera off.
         *
         *  @return void
         */
        void cameraOff()
                raises (SystemErr::SystemInAutoModeEx);

        /**
         * Move telescope in synchronous mode. Raises an exception if the
         * requested position is out of limits.
         *
         *  @coordinates az, el coordinates
         *  @return void
         */
        void moveTelescope(in TYPES::Position coordinates)
                raises(SystemErr::PositionOutOfLimitsEx, SystemErr::SystemInAutoModeEx);

        /**
         * Current telescope position.
         *
         *  @return Telescope position
         */
```

```
        TYPES::Position getTelescopePosition();

        /**
         * Get an image from the camera (from actual position of telescope).
         *
         *  @return Image from the camera
         */
        TYPES::ImageType getCameraImage()
                raises(SystemErr::SystemInAutoModeEx, SystemErr::CameraIsOffEx);

         /* Camera settings */

        /**
         * Set the RGB configuration of the camera
         * @param rgbConfig the TYPES::RGB configuration
         */
        void setRGB(in TYPES::RGB rgbConfig)
                raises(SystemErr::CameraIsOffEx);

        /**
         * Set the pixel bias configuration of the camera
         * @param bias the pixel bias configuration
         */
        void setPixelBias(in long bias)
                raises(SystemErr::CameraIsOffEx);

        /**
         * Set the reset level configuration of the camera
         * @param resetLevel the reset level configuration
         */
        void setResetLevel(in long resetLevel)
                raises(SystemErr::CameraIsOffEx);
        };
};

#endif
```

DataBase.idl

```
#ifndef _DATABASE_IDL_
#define _DATABASE_IDL_

/*******************************************************************************
*    ACS Community - https://github.com/ACS-Community/ACS-Workshop
*
*    This library is free software; you can redistribute it and/or
*    modify it under the terms of the GNU Lesser General Public
*    License as published by the Free Software Foundation; either
*    version 2.1 of the License, or (at your option) any later version.
*
*    This library is distributed in the hope that it will be useful,
*    but WITHOUT ANY WARRANTY; without even the implied warranty of
*    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
*    Lesser General Public License for more details.
*
*    You should have received a copy of the GNU Lesser General Public
*    License along with this library; if not, write to the Free Software
*    Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307  USA
*/

#include <acscomponent.idl>
#include <Types.idl>
#include "SystemErr.idl"

#pragma prefix "acsws"


/**
 * @file Database.idl
```

```
 * SystemErr Database IDL File
 */

module DATABASE_MODULE
{
        /** @interface Database
         *  Interface to get access to the UOS database
         */
        interface DataBase : ACS::ACSComponent
        {
                /**
                 * Stores a new Proposal.
                 *
                 * @param targets Target list composing this proposal.
                 * @return Assigned proposal ID (pid).
                 */
                long storeProposal (in TYPES::TargetList targets);


                const long STATUS_INITIAL_PROPOSAL = 0;
                const long STATUS_NO_SUCH_PROPOSAL = -999; // it is not recomended to use this constant. this
constant may be removed in future.

                /**
                 * Get the current proposal status for the given
                 * proposal.
                 *
                 * @param pid Proposal ID
                 * @return status
                 */
                long getProposalStatus(in long pid);

                /**
                 * Remove proposal.
                 *
                 * @param pid Proposal ID
                 */
                void removeProposal(in long pid);

                /**
                 * Retuns all images for a given proposal.
                 * Raises an exception if proposal has not been
                 * executed yet.
                 *
                 * @param pid Proposal ID
                 * @return Image list that belongs to this proposal
                 */
                TYPES::ImageList getProposalObservations(in long pid)
                        raises(SystemErr::ProposalNotYetReadyEx);

                /**
                 * Returns stored proposals which have not been executed yet.
                 *
                 * @return Proposals with queued status. If there are no
                 * pending proposals returns an empty list
                 */
                TYPES::ProposalList getProposals();

                /**
                 * Set the proposal status. Raises an exception if the change is not from
                 * queued(0) to running(1) or from running(1) to ready(2).
                 *
                 * @param pid Proposal ID
                 * @param tid target ID
                 * @return None
                 */
                void setProposalStatus(in long pid, in long status)
                        raises(SystemErr::InvalidProposalStatusTransitionEx);

                /**
                 * Stores an image for a given proposal and target. Raises an exception
```

```
                 * if an image has already been stored for the given
                 * tid and pid.
                 *
                 * @param pid Proposal ID
                 * @param tid target ID
                 * @return None
                 */
                void storeImage(in long pid,
                                in long tid,
                                in TYPES::ImageType image)
                        raises(SystemErr::ImageAlreadyStoredEx); // TODO raise also new exception
"ProposalDoesNotExist"
                /**
                 * Clean all the proposals
                 */
                void clean();
        };
};

#endif
```

Camera.idl

```
#ifndef _CAMERA_IDL_
#define _CAMERA_IDL_

/*******************************************************************************
*    ACS Community - https://github.com/ACS-Community/ACS-Workshop
*
*    This library is free software; you can redistribute it and/or
*    modify it under the terms of the GNU Lesser General Public
*    License as published by the Free Software Foundation; either
*    version 2.1 of the License, or (at your option) any later version.
*
*    This library is distributed in the hope that it will be useful,
*    but WITHOUT ANY WARRANTY; without even the implied warranty of
*    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
*    Lesser General Public License for more details.
*
*    You should have received a copy of the GNU Lesser General Public
*    License along with this library; if not, write to the Free Software
*    Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307  USA
*/

#include <baci.idl>
#include <Types.idl>

#pragma prefix "acsws"


module CAMERA_MODULE
{
        interface Camera : ACS::ACSComponent
        {

                TYPES::ImageType takeImage(in string exposureTime, in string iso);

        };
};

#endif
```

Instrument.idl

```
#ifndef _INSTRUMENT_IDL_
#define _INSTRUMENT_IDL_
```

```
/*******************************************************************************
*    ACS Community - https://github.com/ACS-Community/ACS-Workshop
*
*    This library is free software; you can redistribute it and/or
*    modify it under the terms of the GNU Lesser General Public
*    License as published by the Free Software Foundation; either
*    version 2.1 of the License, or (at your option) any later version.
*
*    This library is distributed in the hope that it will be useful,
*    but WITHOUT ANY WARRANTY; without even the implied warranty of
*    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
*    Lesser General Public License for more details.
*
*    You should have received a copy of the GNU Lesser General Public
*    License along with this library; if not, write to the Free Software
*    Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307  USA
*/

#include <acscomponent.idl>
#include <Types.idl>
#include "SystemErr.idl"

#pragma prefix "acsws"


/**
 * @file Instrument.idl
 * SystemErr Instrument IDL file
 */

module INSTRUMENT_MODULE
{
        /*****************************************************************/
        /** @interface Instrument
         *  This is the Webcam interface for the 50mm telescope.
         */
        interface Instrument : ACS::ACSComponent
        {
        /**
         * Turns the Instrument camera on.
         *
         * @return None
         */
        void cameraOn ();

        /**
         * Turns the Instrument off.
         * A NULL string as the target identifier indicates that no image file
         * should be saved.
         *
         * @todo Function should be refactored so that only one operation is performed.
         *
         * @return None
         */
        void cameraOff ();

        /**
         * Retrieve image from the Instrument. Raises an exception if the
         * camera is not on.
         *
         * @return array of longs containing the image pixels
         */
        TYPES::ImageType takeImage(in long exposureTime)
                raises(SystemErr::CameraIsOffEx);

        /* Camera settings */

        /**
         * Set the RGB configuration of the camera
         * @param rgbConfig the TYPES::RGB configuration
         */
```

```
        void setRGB(in TYPES::RGB rgbConfig)
                raises(SystemErr::CameraIsOffEx);

        /**
         * Set the pixel bias configuration of the camera
         * @param bias the pixel bias configuration
         */
        void setPixelBias(in long bias)
                raises(SystemErr::CameraIsOffEx);

        /**
         * Set the reset level configuration of the camera
         * @param resetLevel the reset level configuration
         */
        void setResetLevel(in long resetLevel)
                raises(SystemErr::CameraIsOffEx);
        };
};

#endif
```

Storage.idl

```
#ifndef _STORAGE_IDL_
#define _STORAGE_IDL_


/*******************************************************************************
*    ACS Community - https://github.com/ACS-Community/ACS-Workshop
*
*    This library is free software; you can redistribute it and/or
*    modify it under the terms of the GNU Lesser General Public
*    License as published by the Free Software Foundation; either
*    version 2.1 of the License, or (at your option) any later version.
*
*    This library is distributed in the hope that it will be useful,
*    but WITHOUT ANY WARRANTY; without even the implied warranty of
*    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
*    Lesser General Public License for more details.
*
*    You should have received a copy of the GNU Lesser General Public
*    License along with this library; if not, write to the Free Software
*    Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307  USA
*/

#include <acscomponent.idl>
#include <Types.idl>

#pragma prefix "acsws"

module STORAGE_MODULE
{
    /**
     * Special storage created for the observatory
     * The storage has been designed to be written only
     * once and read many times per observation
     */
    interface Storage :  ACS::ACSComponent
    {

        /**
         * @return the next valid ID to store a proposal
         */
        long getNextValidId();

        /**
         * Store in the Storage the completed observation
         * The number of Targets within proposal must match
         * the number of images
         */
        void storeObservation(in TYPES::Proposal prop, in TYPES::ImageList images);

        /**
         * Wipe out the storage
         */
        void clearAllData();

        /**
         * Retrieve a completed proposal from Storage
         */
        TYPES::ImageList getObservation(in long pid);
    };
};

#endif
```

Scheduler.idl

```
#ifndef _SCHEDULER_IDL_
#define _SCHEDULER_IDL_

/*******************************************************************************
*    ACS Community - https://github.com/ACS-Community/ACS-Workshop
*
*    This library is free software; you can redistribute it and/or
*    modify it under the terms of the GNU Lesser General Public
*    License as published by the Free Software Foundation; either
*    version 2.1 of the License, or (at your option) any later version.
*
*    This library is distributed in the hope that it will be useful,
*    but WITHOUT ANY WARRANTY; without even the implied warranty of
*    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
*    Lesser General Public License for more details.
*
*    You should have received a copy of the GNU Lesser General Public
*    License along with this library; if not, write to the Free Software
*    Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307  USA
*/

#include <acscomponent.idl>
#include "SystemErr.idl"

#pragma prefix "acsws"


/**
 * @file Scheduler.idl
 * SystemErr Scheduler IDL file
 */

module SCHEDULER_MODULE
{
        /** @interface Scheduler
         *  This is a simple scheduler for the SystemErr
         */
        interface Scheduler : ACS::ACSComponent
        {
        /**
         * Start the scheduler.
         * The scheduler will loop through all available proposals,
         * either until all proposals are done or until the stop method is called.
         * Raises an exception if called twice.
         * @return None
         */
        void start ()
                raises(SystemErr::SchedulerAlreadyRunningEx);

        /**
         * Stops the scheduler.
         * This will stop the scheduler from scheduling more proposals.
         *
         * It will not(!) break the ongoing observation, and will return only
         * when the running observation has finished.
         *
         * Raises an exception if called twice.
         * @return None
         */
        void stop ()
                raises(SystemErr::SchedulerAlreadyStoppedEx);

        /**
         * Returns the pid of the proposal currently under execution
         *
         * Raises exception if no proposal is executing.
         * @retun Proposal ID
         */
        long proposalUnderExecution()
                raises(SystemErr::NoProposalExecutingEx);
        };
```

```
};

#endif
```

Telescope.idl

```
#ifndef _TELESCOPE_IDL_
#define _TELESCOPE_IDL_

/*******************************************************************************
*    ACS Community - https://github.com/ACS-Community/ACS-Workshop
*
*    This library is free software; you can redistribute it and/or
*    modify it under the terms of the GNU Lesser General Public
*    License as published by the Free Software Foundation; either
*    version 2.1 of the License, or (at your option) any later version.
*
*    This library is distributed in the hope that it will be useful,
*    but WITHOUT ANY WARRANTY; without even the implied warranty of
*    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
*    Lesser General Public License for more details.
*
*    You should have received a copy of the GNU Lesser General Public
*    License along with this library; if not, write to the Free Software
*    Foundation, Inc., 59 Temple Place, Suite 330, Boston, MA 02111-1307  USA
*/

#include <acscomponent.idl>
#include <Types.idl>
#include "SystemErr.idl"

#pragma prefix "acsws"


/**
 * @file Telescope.idl
 * SystemErr Telescope IDL file
 */
module TELESCOPE_MODULE
{
        /** @interface Telescope
         * High level interface to communicate with the hardware
         * related component.
         */

        interface Telescope : ACS::ACSComponent
        {
        /**
         * Moves to the given position and takes an exposure
         * of length exposureTime (seconds). Raises and exception
         * if the requested position is out of limits.
         *
         * @param coordinates target coordinates
         * @param exposureTime exposure time of the current observation
         * @return Image
         *
         */
        TYPES::ImageType observe(in TYPES::Position coordinates, in long exposureTime)
                raises(SystemErr::PositionOutOfLimitsEx);

        /**
         * Commands the telescope to move to the given position. Raises
         * an exception if the requested position is out of limits.
         *
         * @param coordinates Requested telescope position
         * @return None
         */
        void moveTo(in TYPES::Position coordinates)
```

```
                raises(SystemErr::PositionOutOfLimitsEx);

        /**
         * Get the current telescope position.
         *
         * @return current telescope position
         */
        TYPES::Position getCurrentPosition();
        };
};

#endif
```

TelescopeControl.idl

```
#ifndef _H3E_IDL_
#define _H3E_IDL_

#include <baci.idl>

#pragma prefix "acsws"

module TELESCOPE_MODULE {

   /** @interface TelescopeControl
    * Defines the interface for controlling and monitoring a simple
    * telescope. This model considers a "quantum" of movement defined by
    * the calibration process for the discrete rotation sensors. The telescope
    * can't be operated until the calibration is done. A "calibrated" status bit
    * is kept, thus the commanded position is not going to be followed if this bit
    * is not set. This IDL was thinked for H3E project and for a lego model, but is used for
    * a more general purpose.
    * Please refer to <a href="https://csrg.inf.utfsm.cl/twiki4/bin/view/ACS/HardwareEndToEndExample">
    * CSRG H3E project Twiki site</a> for more information.
    */
   interface TelescopeControl : ACS::CharacteristicComponent {

      /**
       * Asynchronously sets the telescope to the specified position.
       * If the "calibrated" status bit is set, this method returns when the telescope
       * is at the commanded position, accepting an error defined by the calibration of
       * the rotation sensors for each axis. If not, the telescope is not going to move
       * and this method returns immediately.
       *
       * @param altitude        desired telescope's altitude (degrees)
       * @param azimut          desired telescope's azimut   (degrees)
       */
       void setTo (in double altitude, in double azimuth);


      /**
       * Asynchronously moves away the telescope, starting from actual position.
       * If the "calibrated" status bit is set, this method returns when the telescope is
       * positioned at the actual position plus the indicated values for each axis. If not,
       * the telescope does not move and this method returns immediately.
       * The indicated altitude and azimut offsets must be bigger than the minimal altitude
       * and azimut steps, defined by the calibration process, or the telescope is not going
       * to move.
       *
       * @param altOffset    desired altitude offset (degrees)
       * @param azOffset     desired azimut offset (degrees)
       */
      void offSet (in double altOffset, in double azOffset);


      /**
       * Moves the telescope to zenith position. It is the same that a "setTo(90,0)" call.
       */
      void zenith ();
```

```
      /**
       * Moves the telescope to parking position (implemmentation-dependant).
       */
      void park ();


      /**
       * Unsets the "calibrated" status bit. It is necessary to manually move the
       * telescope (e.g. when it needs to be calibrated).
       */
      void setUncalibrated   ();


      /**
       * Starts the calibration procedure for the conversion from motor rotation
       * to arc degrees. This procedure requires the telescope to be at zenith
       * position due to lack of touch sensors on the Lego MindStorms kit. Don't
       * forget to unset the "calibration" status bit when manually moving the telescope
       * or it is going to try to return to the last commanded position.
       */
      void calibrateEncoders  ();


      /** Indicates the last commanded telescope's altitude.
       */
      readonly attribute ACS::RWdouble commandedAltitude;   // devio, LegoCmdAltDevIO, CORBA::Double

      /** Indicates the last commanded telescope's azimut.
       */
      readonly attribute ACS::RWdouble commandedAzimuth;   // devio, LegoCmdAzDevIO, CORBA::Double

      /** Indicates the actual telescope's altitude.
       */
      readonly attribute ACS::ROdouble actualAltitude;   // devio, LegoAltDevIO, CORBA::Double

      /** Indicates the actual telescope's azimut.
       */
      readonly attribute ACS::ROdouble actualAzimuth;    // devio, LegoAzDevIO, CORBA::Double

      /** Indicates some telescope's status parameters
       */
      readonly attribute ACS::RWpattern status;    // devio, LegoStatusDevIO, ACS::pattern
   };

};

#endif
```

Error Definitions

The XML Error Definition file has to be placed in the ICD/idl directory:

SystemErr.xml

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<Type xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="Alma/ACSError" xsi:schemaLocation="Alma
/ACSError ACSError.xsd" name="SystemErr" type="900907" _prefix="acsws">
        <ErrorCode name="AlreadyInAutomatic"
                   shortDescription="Already in automatic mode"
                   description="Trying to set automatic mode, failed. It has already been set"/>
        <ErrorCode name="PositionOutOfLimits"
                   shortDescription="Position out of limits"
                   description="Command tries to move the telescope out of limits"/>
        <ErrorCode name="ProposalNotYetReady"
                   shortDescription="Proposal is not ready"
                   description="Proposal is not ready"/>
        <ErrorCode name="InvalidProposalStatusTransition"
                   shortDescription="Database: Invalid proposal status"
                   description="Trying to set an invalid status for the proposa."/>
        <ErrorCode name="ImageAlreadyStored"
                   shortDescription="Image already stored"
                   description="Image has already been stored in the database."/>
        <ErrorCode name="CameraIsOff"
                   shortDescription="camera is off"
                   description="Trying to take exposure with camera off."/>
        <ErrorCode name="SchedulerAlreadyRunning"
                   shortDescription="Scheduler is already running"
                   description="Trying to start scheduler, but is already has been started."/>
        <ErrorCode name="SchedulerAlreadyStopped"
                   shortDescription="Scheduler is already stopped"
                   description="Trying to stop scheduler, but is already has been stopped."/>
        <ErrorCode name="NoProposalExecuting"
                   shortDescription="No proposal is executing"
                   description="Trying to retrieve an executing proposal, but no proposal is executing."/>
        <ErrorCode name="SystemInAutoMode"
                   shortDescription="System is in automatic mode"
                   description="Trying to execute a command in console while the system is in automatic mode."/>
        <ErrorCode name="CannotOpenDevice"
                   shortDescription="Can't open device"
                   description="Can't open THE device."/>
</Type>
```