1. Index

1. Index ............................................................................................................... 2
2. Revision history .............................................................................................. 6
3. Thank you! ...................................................................................................... 7
    3.1. Support .................................................................................................... 7
    3.2. Support FTP-Server ............................................................................... 7
4. Quick overview version 3.1 or later ................................................................. 8
    4.1. Steps to change to Version 3.1 or later .................................................. 8
    4.2. Python Environment ............................................................................. 8
        4.2.1 Converting script files...................................................................... 10
    4.3. Script import changes ........................................................................... 10
    4.4. Debugger support .................................................................................. 11
    4.5. Command-line interface ....................................................................... 11
    4.6. Make / Check Automation ..................................................................... 11
    4.7. GCOM_LOG.txt .................................................................................... 11
    4.8. Project Directory _Working .................................................................... 11
    4.9. Directory _Recycler ................................................................................ 11
    4.10. Sample Script Files ............................................................................. 11
    4.11. Sample EPICS Files ............................................................................ 11
5. Glossary ......................................................................................................... 12
6. Available Software Editions .......................................................................... 14
    6.1. Light-Edition .......................................................................................... 14
    6.2. Standard-Edition ................................................................................... 14
    6.3. Premium-Edition .................................................................................... 14
    6.4. Retrieving the supported software edition ............................................ 15
7. Requirements ................................................................................................. 16
8. Software components ................................................................................... 17
    8.1. The software structure .......................................................................... 17
    8.2. A typical test network setup ................................................................... 18
    8.3. Python Script language ......................................................................... 19
    8.4. Interesting Links to Python websites ..................................................... 19
        8.4.1 www.python.org ............................................................................. 19
        8.4.2 www.eclipse.org .......................................................................... 19
        8.4.3 http://pydev.sourceforge.net/ ....................................................... 19
        8.4.4 http://www.fabioz.com/pydev/ ....................................................... 19
        8.4.5 http://pychecker.sourceforge.net/ ............................................... 19
        8.4.6 http://mmm-experts.com/download/PyScripter-setup.exe ............... 19
    8.5. Python literature .................................................................................... 20
        8.5.1 Core Python programming ............................................................. 20
        8.5.2 Learning Python ............................................................................. 20
        8.5.3 Python Pocket Reference ................................................................ 20
        8.5.4 Python 2 ........................................................................................ 20
        8.5.5 Python GE-PACKT ........................................................................ 20
9. Wireshark Integration .................................................................................... 21
    9.1. Settings .................................................................................................. 21
10. Setup ............................................................................................................ 23
10.1. Requirements .............................................................................................................. 23
10.1.1 Operating system ..................................................................................................... 23
10.2. Setup procedure ......................................................................................................... 23
10.3. Updates ...................................................................................................................... 24
10.4. Updating using the Patchfile ..................................................................................... 24
10.5. Registering Office/Word2003 ..................................................................................... 25
  10.5.1 Step 1 ..................................................................................................................... 25
  10.5.2 Step 2 ..................................................................................................................... 26
  10.5.3 Step 3 ..................................................................................................................... 27
  10.5.4 Step 4 ..................................................................................................................... 28
  10.5.5 Step 5 ..................................................................................................................... 30
  10.5.6 Step 6 ..................................................................................................................... 31
11. General information about the BACnet Test Framework ............................................. 32
  11.1. The associated user .................................................................................................. 32
  11.2. System history ......................................................................................................... 32
  11.3. Project history ......................................................................................................... 32
  11.4. Telegram buffer for execution ................................................................................. 32
  11.5. Testing Efforts ......................................................................................................... 32
12. Working with the BTF .................................................................................................... 33
  12.1. Starting the Software ............................................................................................... 33
  12.2. Menu Project/Change .............................................................................................. 34
    12.2.1 Remove ............................................................................................................... 34
    12.2.2 New ..................................................................................................................... 35
    12.2.3 Open .................................................................................................................... 35
    12.2.4 Copy .................................................................................................................... 35
    12.2.5 Import ................................................................................................................. 37
  12.3. Menu Project/Rename ............................................................................................. 38
  12.4. Menu Project/Reset ............................................................................................... 38
  12.5. Menu Project/Export ............................................................................................. 38
  12.6. The System Administration ..................................................................................... 40
    12.6.1 System overview ............................................................................................... 40
    12.6.2 Settings ............................................................................................................... 44
    12.6.3 Testscript Management ...................................................................................... 47
    12.6.4 History ................................................................................................................. 51
    12.6.5 Licensing ............................................................................................................. 52
  12.7. Quit .......................................................................................................................... 54
  12.8. Results ..................................................................................................................... 54
    12.8.1 XML & Schema ................................................................................................. 54
    12.8.2 Word .................................................................................................................. 55
    12.8.3 Info ..................................................................................................................... 56
13. The main dialog .............................................................................................................. 57
14. The test tasks ................................................................................................................... 58
  14.1. Project Properties .................................................................................................... 59
    14.1.1 Information ....................................................................................................... 59
    14.1.2 Data ................................................................................................................... 60
    14.1.3 Notes .................................................................................................................. 63
    14.1.4 History ............................................................................................................... 64
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>15.2.1</td>
<td>Test Plan Control</td>
<td>88</td>
</tr>
<tr>
<td>15.2.2</td>
<td>The Output Window</td>
<td>88</td>
</tr>
<tr>
<td>15.3</td>
<td>Symbols used in test plan execution</td>
<td>90</td>
</tr>
<tr>
<td>15.4</td>
<td>Test interaction</td>
<td>92</td>
</tr>
<tr>
<td>16.</td>
<td>Debugging Scripts</td>
<td>93</td>
</tr>
<tr>
<td>17.</td>
<td>Command-line Interface</td>
<td>95</td>
</tr>
<tr>
<td>17.1</td>
<td>The command-line arguments</td>
<td>95</td>
</tr>
<tr>
<td>17.1.1</td>
<td>Language</td>
<td>95</td>
</tr>
<tr>
<td>17.1.2</td>
<td>The Remote Control File (XML)</td>
<td>95</td>
</tr>
<tr>
<td>17.1.3</td>
<td>Return Values</td>
<td>96</td>
</tr>
<tr>
<td>17.2</td>
<td>Sample Batchfile</td>
<td>97</td>
</tr>
<tr>
<td>18.</td>
<td>Make-Automation</td>
<td>99</td>
</tr>
<tr>
<td>18.1</td>
<td>Introduction</td>
<td>99</td>
</tr>
<tr>
<td>18.2</td>
<td>Writing a Make Script</td>
<td>99</td>
</tr>
<tr>
<td>18.3</td>
<td>Types of messages</td>
<td>100</td>
</tr>
<tr>
<td>18.3.1</td>
<td>Make without type</td>
<td>100</td>
</tr>
<tr>
<td>18.3.2</td>
<td>Make input</td>
<td>100</td>
</tr>
<tr>
<td>18.3.3</td>
<td>Make input with tolerance</td>
<td>100</td>
</tr>
<tr>
<td>18.3.4</td>
<td>Make choice</td>
<td>100</td>
</tr>
<tr>
<td>18.3.5</td>
<td>Make choice with ref_val</td>
<td>101</td>
</tr>
<tr>
<td>18.4</td>
<td>Importing a Make Script</td>
<td>101</td>
</tr>
<tr>
<td>18.5</td>
<td>Assigning a Make Script</td>
<td>102</td>
</tr>
<tr>
<td>18.6</td>
<td>Enabling/Disabling the Make automation</td>
<td>102</td>
</tr>
<tr>
<td>18.7</td>
<td>Flag Uninterrupted Execution</td>
<td>103</td>
</tr>
<tr>
<td>18.8</td>
<td>Execution Flowchart</td>
<td>104</td>
</tr>
<tr>
<td>19.</td>
<td>Reading and Writing proprietary data</td>
<td>105</td>
</tr>
<tr>
<td>19.1</td>
<td>Proprietary ReadProperty Example</td>
<td>105</td>
</tr>
<tr>
<td>19.2</td>
<td>Proprietary WriteProperty Example</td>
<td>106</td>
</tr>
<tr>
<td>20.</td>
<td>The recommended network setting</td>
<td>108</td>
</tr>
<tr>
<td>20.1</td>
<td>Setting up the BACnet Protocol-Stack</td>
<td>110</td>
</tr>
<tr>
<td>20.2</td>
<td>Error starting the test plan</td>
<td>112</td>
</tr>
<tr>
<td>21.</td>
<td>The process identifier for events and alarms</td>
<td>113</td>
</tr>
<tr>
<td>22.</td>
<td>Conventions for usage of wildcards</td>
<td>114</td>
</tr>
</tbody>
</table>
23. Sample EPICS-file for IUT ................................................................. 119
24. Sample EPICS-file for TD ............................................................... 129
25. Sample EPICS-file for OD ............................................................... 140

BACnet is a registered trademark of American Society of Heating and Refrigerating Engineers (ASHRAE).
Windows is a registered trademark of Microsoft Corporation.
The MBS logo is a registered trademark of MBS GmbH.
## 2. Revision history

<table>
<thead>
<tr>
<th>Version</th>
<th>Date</th>
<th>Author</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.4.0.0</td>
<td>14.05.2007</td>
<td>Frank Schubert</td>
<td>Initial Version</td>
</tr>
<tr>
<td>1.5.0.0</td>
<td>19.07.2007</td>
<td>Frank Schubert</td>
<td>Added support for tshark</td>
</tr>
<tr>
<td>1.5.0.0</td>
<td>20.07.2007</td>
<td>Frank Schubert</td>
<td>Extended chapter 7 (tshark support)</td>
</tr>
<tr>
<td>1.5.0.1</td>
<td>22.07.2007</td>
<td>Frank Schubert</td>
<td>Extended chapter 16-20</td>
</tr>
<tr>
<td>1.6.0.1</td>
<td>03.08.2007</td>
<td>Frank Schubert</td>
<td>Added Requirement for WORD/Office Professional 2003</td>
</tr>
<tr>
<td>1.7.0.0</td>
<td>15.08.2007</td>
<td>Horst Hannappel</td>
<td>Added description of software versions</td>
</tr>
<tr>
<td>1.7.0.0</td>
<td>15.08.2007</td>
<td>Horst Hannappel</td>
<td>Removed field &quot;NumPages&quot; from foot-line (does not work properly with pdf-output)</td>
</tr>
<tr>
<td>1.7.0.1</td>
<td>26.08.2007</td>
<td>Horst Hannappel</td>
<td>Added important hint for TD / OD device instance numbers</td>
</tr>
<tr>
<td>1.8.0.0</td>
<td>19.09.2007</td>
<td>Frank Schubert</td>
<td>Corrected FD entry in Default bacstacd.ini</td>
</tr>
<tr>
<td>1.8.0.0</td>
<td>19.09.2007</td>
<td>Frank Schubert</td>
<td>Added process-identifier for OD-device</td>
</tr>
<tr>
<td>3.1</td>
<td>02.06.2009</td>
<td>Frank Schubert</td>
<td>Added descriptions for new functions:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- License files</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- Lock test</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- Program start changed (Project selection)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- Enhanced csv/xml export</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- Debugger-Integration</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- Automation of Make/Check Statements</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- Script-handling of user-scripts as Python source-code</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- Script-handling within the project (no global pool anymore)</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- BTF remote start and remote control</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- New log-file GCOM_LOG.txt</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- New directory “_WORKING”</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- New directory _RECYCLER</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- Updated Glossary</td>
</tr>
<tr>
<td>3.2</td>
<td>02.08.2009</td>
<td>Frank Schubert</td>
<td>Updated _Working directory</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Updated script-import</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Updated licensing</td>
</tr>
<tr>
<td>3.2</td>
<td>12.09.2009</td>
<td>Frank Schubert</td>
<td>Added new literature</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Added hint to 3.1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Changed the order of chapter 3 and 4</td>
</tr>
</tbody>
</table>

© 2006-2009 MBS GmbH, Krefeld
All rights reserved.
3. Thank you!

Thank you for purchasing the BACnet Test Framework. This software allows comfortable setup of BACnet test projects, performing automated BACnet tests for BACnet server-devices as well as BACnet client-devices and supports printouts of test-reports directly from the database.

The BACnet Test Framework is designed to perform script-based BACnet-tests for conformance testing according to the BTL-test-plans.

You may add your own scripts and use the BACnet Test Framework for your QA-tasks (standard and premium version).

The BACnet Test Frameworks includes the test scripts used in the current BTL test-plan as a factory default database.

Continuous maintenance will be provided and updates may be retrieved from MBS in the future.

**We wish you a successful testing with the BACnet Test Framework!**

MBS GmbH, Krefeld

3.1. Support

The official support E-Mail address for BTF-support-issues is:

**support-btf@mbs-software.de**

Telephone support is provided Monday to Friday, 9:00 – 12:00 and 13:00 – 16:00 CET / CEST excluding public holidays.

Telephone:  +49 / 21 51 / 72 94 – 0  
Telefax:  +49 / 21 51 / 72 94 – 50

**We are looking forward to your queries, don't hesitate to contact us.**

3.2. Support FTP-Server

New releases and beta versions are provided on a FTP-Server. The access information is provided by a newsletter which is sent to all registered customers or interested parties. Please let us know, if you are interested in being added to the list of newsletter-participants.

Premium customers receive the password information for the script source-codes by a separate newsletter.
4. Quick overview version 3.1 or later

This chapter provides an overview of changes from earlier versions to version 3.1 or later versions. Please refer to the release notes for details.

4.1. Steps to change to Version 3.1 or later

Due to the debugger support and changes to the environment all prolog-, epilog, user- used in former BTF versions have to be converted using a converter tool (see below) and have to be imported again in every project, where these scripts may be used.

The import dialog is now located in the section “Other tests” in the test-plan configurator.

4.2. Python Environment

The environment variable «PYTHONPATH» is no longer used to determine Python.exe or dictionaries / libraries. The environment is now set within the scripts itself.

If you use Standalone scripts (without using the BTF GUI), simply import “BTFEnvironment.py” to set the path according to your installation.

The associated Python code to set the environment is shown below, even though we strongly recommend to simply include the settings files rather than setting the environment in every single source file.
Content of BTFEnvironment.py (if installed on drive C:)

```python
import sys
sys.path.append(r'C:\Programme\MBS\BTF\bin\framework')
sys.path.append(r'C:\Programme\MBS\BTF\bin\framework\lib')
sys.path.append(r'C:\Programme\MBS\BTF\bin\framework\internal')
sys.path.append(r'C:\Programme\MBS\BTF\bin\framework\lib\bacnet')
sys.path.append(r'C:\Programme\MBS\BTF\bin\framework\lib\bacdb')
sys.path.append(r'C:\Programme\MBS\BTF\bin\framework\lib\guicom')
```

As you can see from this code, the structure of the internal BTF dictionaries slightly changed.
4.2.1 Converting script files
To convert existing script files from earlier BTF versions to version 3.1 or later, use the converter tool included in BTF since version 3.1.

Execute converter.pyc from the directory “<BTF-Installation>\bin\framework\tools” to convert the scripts.

The original script files remain unchanged, while the converted files are stored in a sub-directory.

This picture shows the converter GUI.

4.3. Script import changes
Scripts like prolog-, epilog-, user- and make-scripts are now stored locally within the project. The former system pool for those scripts was removed from the software.

Script files are now stored as source code. This is required to allow debugging and allows easier updates when using future Python versions.
4.4. Debugger support
BTF now supports debugging the test execution, especially to verify and debug prolog-, epilog-, user- and make-scripts. At the moment BTF supports the freeware-tool PyScripter. Refer to the chapter “Debugging Scripts” later in this document.

4.5. Command-line interface
The command-line interface was enhanced to allow full remote control over BTF. Refer to the chapter “Command-line Interface” later in this document.

4.6. Make / Check Automation
The automation of Make- and Check-statements was included in BTF version 3. This allows executing so-called make-scripts instead of user-interaction anytime a Make- or Check-statement appears in the test. Automating those statements normally require a second access to the IUT, e.g. using telnet, FTP, or proprietary tools.

4.7. GCOM_LOG.txt
The interaction between the script interpreter and the GUI is now logged in a new logfile “GCOM_LOG.txt” in the local settings directory.

4.8. Project Directory _Working
The directory <BTF-Installation>\Projects\_Working is used to temporarily store all files related to the current test project and test-plan. E.g. the prolog- and epilog files, user-scripts and make-scripts are stored here. This is the directory, where the files are provided for debugging, too.

4.9. Directory _Recycler
If you removed a project from the database, a copy of the exported project is stored in the directory <BTF-Installation>_Recycler. This project may be imported again later or permanently removed from the hard-disk.

4.10. Sample Script Files
In the directory <BTF-Installation>\Documentation\SampleScripts BTF now contains a selection of sample script files. These should help getting started with developing user defined scripts for BTF.

4.11. Sample EPICS Files
In the directory <BTF-Installation>\Documentation\SampleEpics BTF now contains a selection of sample EPICS files. These are the same as provided on the BTF FTP-Server and may be used as templates when starting creating BTF projects.
5. Glossary

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>API</td>
<td>Application Programming Interface</td>
</tr>
<tr>
<td>BACnet</td>
<td>Building Automation and Control Network, communication protocol for building automation and energy services</td>
</tr>
<tr>
<td>BI</td>
<td>BACnet International, an organization defining BTL test plans and running the BTL.</td>
</tr>
<tr>
<td>BTF</td>
<td>BACnet Test Framework, this software</td>
</tr>
<tr>
<td>BTL</td>
<td>BACnet Test Laboratory, organization within BI</td>
</tr>
<tr>
<td>BTL-WG</td>
<td>BACnet Test Laboratory Working Group, organization within BI specifying the BTL test-plans</td>
</tr>
<tr>
<td>BTL test plan</td>
<td>Test specification developed by the BTL working group of BI</td>
</tr>
<tr>
<td>B-OWS</td>
<td>BACnet Operator Workstation</td>
</tr>
<tr>
<td>EPICS</td>
<td>Electronic Protocol Implementation Conformance Statement, described in the norm ISO 16484-6</td>
</tr>
<tr>
<td>Epilog</td>
<td>A script executed after the associated test script. An epilog is designated to cleanup the settings after performing a test.</td>
</tr>
<tr>
<td>IUT</td>
<td>Implementation under test, device to be tested</td>
</tr>
<tr>
<td>Make-script</td>
<td>A script performing (non-BACnet) interaction with the IUT to allow automated MAKE-statements (automated user-interaction with the BTF) and CHECK-statements (automated verification of test-steps).</td>
</tr>
<tr>
<td>NIC</td>
<td>Network Interface Card</td>
</tr>
<tr>
<td>OD</td>
<td>Other device (another device taking some action in certain tests, e.g. externally acknowledging alarms, this device is simulated within BTF using a second NIC)</td>
</tr>
<tr>
<td>PICS</td>
<td>Protocol Implementation Conformance Statement</td>
</tr>
<tr>
<td>Prolog</td>
<td>A script executed prior to the associated test script. A prolog is designated to setup the configuration requirements for a test.</td>
</tr>
<tr>
<td>TD</td>
<td>Testing device (the BACnet Test Framework)</td>
</tr>
</tbody>
</table>
User-test
A script performing user-specific tests possibly including the BTF-BACnet-API
6. Available Software Editions

This chapter describes the differences of the different software editions.

6.1. Light-Edition

The Light-Edition of the BACnet Test Framework is designated for pre-testing to prepare for a BTL-test. The factory database contains all test scripts used for the current BTL-tests in compiled form.

Prolog-Scripts and Epilog-Scripts may be programmed in Python and imported into the BTF system database.

Contrary to the Standard and the Premium Editions, the Light-Edition does not allow performing user-scripts. Additionally the report-format using Microsoft Word is limited to the factory-template and cannot be changed.

6.2. Standard-Edition

The Standard-Edition offers all features of the Light-Edition and is designated for pre-testing to prepare for a BTL-test as well as using the BACnet Test Framework as a Quality Assurance tool.

In addition to run the BTL-test-scripts, the standard version allows running user-scripts to perform tests beyond BTL-tests.

The Microsoft Word report-template may be changed according to the requirements of the QA-tests or to match the corporate design.

6.3. Premium-Edition

The Premium-Edition offers all features of the standard version. In addition, all BTL-test-scripts are provided as source-codes (requires an NDA=Non-Disclosure-Agreement).

Standard- and Premium-Edition are identical in software, they differ in packaging only. The premium edition contains a separate CDROM containing the script source-codes.
6.4. Retrieving the supported software edition

The supported software edition may be retrieved starting the system administration dialog.

BTF = Standard or Premium Edition
Light = Light Edition
7. Requirements

The BACnet Test Framework is a software designed for automated tests of BACnet server devices as well as BACnet client devices on conformance to the norm ISO 16484-5 (ANSI/ASHRAE 135-2001 and 135-2004).

Further versions of the BACnet-standard and ISO-Norm 16484 may be available as database updates.

As the Data-Link-Layer BACnet/IP is supported according to the specifications ANSI/ASHRAE 135-1995 Annex-J.

For MS/TP testing an additional hardware box is available upon request.

The software is approved for the operating system Microsoft Windows-XP Professional Service Pack 2 or 3. Other Microsoft operating systems maybe supported in principle, but other versions are not officially supported and not yet tested.

**Windows Vista / Windows 7 is not yet supported and will not work properly.**

An IBM compatible PC with network interface card NIC is required and if needed a printer for test result printouts (not included in the prices).

Minimum recommended: Pentium IV, 2 GHz, 1024 MB RAM, 80 GB hard disk, 19" monitor (resolution 1280x1024), network card Ethernet 10/100Mbit/s, printer.

A wide-screen monitor (16:9 or 16:10 format) with a resolution of up to 1920x1080 or 1920x1200 is recommended. A second monitor is recommended to run Wireshark or other analyzer tools while testing.

A significant difference in execution speed may appear when using too less memory, even though the software works with 256MB memory, at least 1.024MB is recommended to achieve a reasonable execution speed.

For OD-Tests (OD=Other Device) a second NIC (network interface card) is required. If you plan to run other BACnet-tools on the same PC on the test network, a third network interface card is suggested.

For printouts of the test documentation and test results in WORD-Format, WORD2003 Professional or Office 2003 Professional Service Pack 1 software is required.

The software is protected by a hardware key (dongle) against unauthorized Usage. A free USB port is necessary for this purpose.
8. Software components

This chapter describes the software components of the BACnet Test Framework.

8.1. The software structure

This picture shows the software structure of the BACnet Test Framework.

- MS-SQL-Server 7
- .Net Graphical User Interface
- XML / MS-Word 2003 Report Generator
- GUI-Com API Port: 11151
- Python Script Language
- BACnet API (370 functions)
- BACnet protocol stack system service (BacStack Daemon)
- BACnet Test network
- More BACnet Testtools
- IUT
8.2. **A typical test network setup**

This picture shows a typical and recommended setup for a test network.

1. Windows PC running the BACnet Test Framework, additional installation on this PC may be a network sniffer like Wireshark.

2. First and primary network interface connection from BTF to the IUT running the TD.

3. Second network interface connection from BTF to the IUT running the OD.

4. Main network connection, provided by either a network hub (typically 10 Mbit/s) or a managed switch providing a monitor port.

5. Second PC running any kind of BACnet-Client-Software, e.g. the BACnet OPC-Server available from MBS, VTS or a B-OWS.

6. IUT = Implementation under Test, the device to be tested on conformance to BACnet.
8.3. Python Script language

A central function within the BACnet Test Framework (the scripting engine) is based on the popular script language Python, see www.python.org. Python is an object-oriented script language which allows powerful script programming.

The integrated API with 370 functions provides direct access to BACnet.

The script language and the API allow programming of user-scripts, prolog script-files, epilog script-files and make (check) script files.

All included test scripts are programmed using Python and are available as source code when purchasing the premium version of the BACnet Test Framework.

Free software tools for programming Python and a lot of excellent books to learn Python are available.

8.4. Interesting Links to Python websites

8.4.1 www.python.org
Main website of the Python community. Provides excellent help documents to learn Python and offers basic articles and samples.

8.4.2 www.eclipse.org
Excellent IDE for different programming languages.

8.4.3 http://pydev.sourceforge.net/
Python plug-in for Eclipse.

8.4.4 http://www.fabioz.com/pydev/
Extended Python Development tool. Commercial product.

8.4.5 http://pychecker.sourceforge.net/
Syntax checker for Python, helps to find bugs in script code.

8.4.6 http://mmm-experts.com/download/PyScripter-setup.exe
Excellent Python editor, freeware, this is the recommended debugger-tool for the BACnet Test Framework.
8.5. Python literature

The following list is a collection of interesting literature for Python.

8.5.1 Core Python programming
Excellent introduction into Python, easy to read and to understand.
Wesly J. Chun
Prentice Hall
Language: English
ISBN: 0-13-226993-7
Price: $49.99

8.5.2 Learning Python
Mark Lutz and David Ascher
O'Reilly
Language: English
Price: $39.99

8.5.3 Python Pocket Reference
Mark Lutz
O'Reilly
Language: English
Price: $9.95

8.5.4 Python 2
German language only, Einführung und Referenz der objektorientierten Skriptsprache
Martin von Löwis und Nils Fischbeck
Language: German
Addison-Wesley
ISBN: 3-8273-1691-X
DM: 89.90 (unfortunately not available anymore)

8.5.5 Python GE-PACKT
German language only, Deutschsprachige Pocket-Referenz
Michael Weigend
Language: German
ISBN-10: 3-8266-1659-6
3.aktualisierte Auflage 2006

EUR: 15.-
9. Wireshark Integration

Wireshark is a popular free network sniffer with a BACnet plug-in (formerly known as Ethereal). The BACnet Test Framework allows controlling external sniffers to store the network logs according to the execution of test scripts. Please refer to www.wireshark.org for more information about this software.

The capture software “tshark.exe” as part of the Wireshark-distribution may be externally controlled to collect log-data on the network.

9.1. Settings

The tshark-command-line and it’s arguments can be configured in the system administration.

<table>
<thead>
<tr>
<th>Category</th>
<th>Wireshark Configuration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Filter String</td>
<td>udp and port 47808</td>
</tr>
<tr>
<td>LogPath</td>
<td>%ProgramFiles%\Wireshark\tshark.exe</td>
</tr>
<tr>
<td>NicNumbers</td>
<td>[2,'TD']</td>
</tr>
<tr>
<td>StartupTime</td>
<td>5</td>
</tr>
</tbody>
</table>

Default-settings:

**Filter String:** This file contains a BACnet-capture filter “udp and port 47808” by default, you may add or remove other capture filters to your local BTF installation. Please refer to the Wireshark documentation how to setup capture filters.

**Test-plan separation**

BTF can broadcast an "unconfirmed text message" which contains the name of the current test. This makes it easy to locate the interesting places in the packet log of a large test-plan. To activate this feature simply insert the prefix "verbose: " in the settings for "capture filter string".

Example

"verbose: udp and port 47808"

This feature is not active by default because it is not yet clear if every IUT safely ignores such messages.

**Log-Path:** Log-files are stored for each test-script separately. The log-files are stored in the project directory

“C:\Programme\MBS\BTF\Projects\Projectname”

**NIC-Numbers [(2,TD)]:** The number in brackets represents the number of the network interface card, a list of numbers for all network interface cards installed
on your machine may be obtained using the command-line: "tshark –D". The name in brackets is for identification of the interface only and has no further meaning.

You may define multiple cards to capture, example network card no. 2 and 3:

\[ (2, 'TD'), (3, 'OD') \]

This setting must exactly match the format as shown (in brackets), otherwise Python will fail retrieving the dictionary.

**Startup Time in seconds**: This value defines the time to wait for tshark to start the Capture. Default is 5 seconds, which is sufficient on most PCs. If packets at the beginning of log-files are missing, you need to increase the value.
10. Setup

This chapter describes the setup procedure of the BACnet Test Framework.

10.1. Requirements

The BACnet Test Framework requires the following basic installation and software requirements.

10.1.1 Operating system

At the moment the officially supported OS is Microsoft Windows XP Professional Service Pack 2 or 3. We recommend that you update to the latest version of OS patches, please refer to http://windowsupdate.microsoft.com for updates.

10.2. Setup procedure

Insert the CDROM in your CDROM drive, if the auto-start option is enabled, the setup will start automatically otherwise execute “Install.exe” manually.
10.3. Updates

Updates are provided as full installation archives (approx. 200MB) or as patch-files only (approx. 7MB). Updating a basic installation simply requires running the patch-file only without any need to install the full version again.

10.4. Updating using the Patchfile

If you are using the Patchfile for updating BTF, please make sure that no instance of BTF is running (shut down a running instance) and check that the SQL-Server is running (if you have stopped the SQL-Server before). You may check the SQL-Server status by launching the SQL-Server Services Manager.
10.5. Registering Office/Word2003

The .Net assembly support must be registered for Office/Word2003 after the first installation. This process is described in six easy steps.

10.5.1 Step 1

Open the control panel “Start/Programs/Control panel” and choose “Software”.

Select Microsoft Office Professional Edition and choose “edit”.

![Software panel](image)
10.5.2 Step 2
Choose “Add or remove Features”.

Add/Remove Features
10.5.3 Step 3

Enable the extended settings for applications.

Choose “continue” to step to the next screen.
10.5.4 Step 4
Open the Microsoft Office Word tab and select ".Net programming support".

Check, if .Net programming support is already registered. If the dialog looks like this

![Microsoft Office Word dialog]

the programming support was already activated, so you can end the control panel now and skip steps 5 and 6.

If .Net programming support is marked with a red cross, you need to activate the support now.
Select .Net programming support and choose “start from desktop”.

![Image of Microsoft Office 2003 Setup dialog box showing options for starting applications and tools]

- Start from Desktop
- Vom Arbeitsplatz starten
- Alles vom Arbeitsplatz starten
- Bei der ersten Verwendung installiert
- Nicht verfügbar

Beschreibung

Nötigter Speicherplatz auf C: 1056 KB
Verfügbare Speicherplatz auf C: 28 GB
10.5.5  **Step 5**
Check, if the .Net support is activated in the dialog.
10.5.6 Step 6

Finish the dialog and leave the control panel.
11. General information about the BACnet Test Framework

This chapter covers the general concept of the BACnet Test Framework.

11.1. The associated user

The user is taken from the windows system login and is stored in the project database anytime the project is changed or new projects are created. You may view the user access and the actions taken by the users anytime in the project or system history.

11.2. System history

The actions taken in the system administration are logged in a global history, which may be viewed within the system administration dialogs.

11.3. Project history

The actions taken in the project are logged in a local project history, which may be viewed in the project settings dialogs within the test project.

11.4. Telegram buffer for execution

It is very important to know, that telegrams are expected to be already sent, when prompting for an interaction. The telegrams are stored internally in a buffer, so that they will not be lost. This prevents from running into timing problems while performing the tests.

11.5. Testing Efforts

The most effort in pre-testing is required to setup a test database according to the IUT.

Prior to setting up a test project with BTF we therefore strongly recommend to refer to the BTL-document “BTL Getting started”, “BTL Testing Guide” and other documents contained in the test documentation package available from the test laboratory or from the BACnet International (www.bacnetinternational.org).

Once you have successfully setup the test project, it’s easy to run the same tests as often as it may take to check for all potential conformance problems.
12. Working with the BTF
This chapter explains the functions of the software step by step.

12.1. Starting the Software
The start screen of the BACnet Test Framework shows an empty main dialog.

The program functions can be accessed by the menus “Project”, “Results” and “Info”.

![Start screen of BACnet Test Framework](image-url)
12.2. Menu Project/Change

This menu item opens the project list.

This dialog allows access to the BTF projects.
Possible actions in this dialog:

12.2.1 Remove

This button removes the selected project. The project in fact is not physically removed, it will be stored in the internal trash bin and may be restored or removed permanently within the system administration dialogs.
12.2.2   New
This button creates a new test project. In the following dialog you may set the unique name of your project and select the associated test standard from the list of available BACnet standards in the list.

Define the unique name and select the BACnet standard and test-plan here.

The new project is accessible in the project list.

12.2.3   Open
This button opens the selected test project.

12.2.4   Copy
This button allows to copy the selected test project and store a new copy of it as a new project. This is useful to clone an existing project.
A dialog appears displaying the database reorganization.

![Reorganizing the databases!]

Please wait …

A new project with the name “Copy from <original name>” will be created and is available in the project list.

The project may be renamed later.

![Project list]

In this example a copy of the project “My First Project” was created with the name “Copy from My First Project”.
12.2.5 Import

This button allows importing a previously exported project file into the BTF. This function is useful when exchanging a pretest project and share it with the test laboratory. The file extension of exported files is ".BTF".

This dialog allows importing existing BTF export files.

The file extension for exported projects is ".btf", so we strongly recommend to compress the files using WinZip or similar programs when sending the project export file by E-Mail. The file size shrinks to a minimum using these tools.
12.3. **Menu Project/Rename**

This function allows to change the name of the project. Please note that the name of the project within the BTF test database shall be unique.

![Rename project dialog box](image)

12.4. **Menu Project/Reset**

This function allows to reset all results and the project history from the project database. The test plan definitions and private parameters are not affected by this function.

![Resetting project dialog box](image)

12.5. **Menu Project/Export**

This menu item allows exporting a test project. The export file (extension ".btf") contains all parts of the database used within the test project, all test plans, all prolog-, epilog and user-scripts and all imported EPICS-files (IUT, TD, OD).

The export function is useful, when exchanging pre-tested projects with the test laboratory.
A file save dialog allows storage of the file to any supported file media.

Please note that the project specific files in the project directory are not included in the export and must be copied/saved separately!
12.6. The System Administration

The following dialogs describe how to setup general settings and to perform imports as well as viewing the system history.

Possible actions in this dialog:

12.6.1 System overview

This dialog allows access to the system information. The different pages may be accessed with the tab-buttons.

12.6.1.1 Databases

This page shows the supported test databases (test standards) as well as a list of current projects showing the projects' database. The “Save” Button allows saving the system database, which may be used as a template database later on.
12.6.1.2. System

This page shows the version numbers of the different .Net assemblies used in the graphical user interface.
12.6.1.3. Vendor-IDs

This page shows the list of BACnet vendors, taken from http://www.bacnet.org/VendorID/BACnet%20Vendor%20IDs.html. The “Import” button allows importing an updated list using a comma-separated .csv file containing the vendor-ID and the vendor-name.
12.6.1.4. Trash bin

This page shows all projects formerly moved to the trash bin using “Remove” in the main page.

Using “Empty trash bin” permanently removes a project from the database.
12.6.2 Settings

This page allows adjusting different settings for the BACnet Test Framework.

Modifying the settings:

Click on the value, you like to change, enter the appropriate value and leave the field using the TAB-key or by selecting another value. In case the original content was modified, the buttons “Save” and “Discard Changes” will be activated.

Choose “Save” to store the settings within the database or “Discard Changes” without changing the settings. If you modified the settings but not saved them, you will be prompted by the software.
12.6.2.1. **Category Application Settings (read-only)**
This category is used to show the software application settings and program paths. Some changes have been writable in former versions, but now all settings are read-only.

12.6.2.1.1. **Project Counter**
This counter is internally used to store the number of the last test project created by the user.

12.6.2.1.2. **Project Database Preset**
The referenced template database is shown here.

12.6.2.1.3. **Main Program Directory**
This information shows the installation path of the application.

12.6.2.1.4. **SQL-Server Data Directory**
This directory refers to the database project path within the BTF installation.

12.6.2.1.5. **Project Path**
This is the directory where project related files are stored (e.g. Wireshark log-files).

12.6.2.2. **Category Customer Data**
In this page you may setup your own address information. This database fields will be used for the test documentation as XML or WORD-file.

You may setup the following data:

- Name
- Street and Number
- ZIP
- Country
- Telephone
- Telefax
- E-Mail
- Website
- Testlab Manager Name
- Testlab Manager Firstname
12.6.2.3. Category Python Settings

12.6.2.3.1. Timeout for Client tests in seconds
This value defines the timeout for the communication between the GUI and the Python interpreter. The default value should be 300 seconds.

If certain tests will require a larger timeout, e.g. a backup/restore procedure, you need to adjust the setting here!

12.6.2.3.2. Timeout for Python communication in seconds
This value is internally used within Python to check for a timeout in communication. Normally there are no changes required, except you are asked by the MBS-support to change this value.

12.6.2.3.3. Debugger
This value refers to the path and filename of the debugger (e.g. PyScripter). Refer to the chapter “Debugging Scripts” later in this document.

12.6.2.3.4. Python Port
This value defines the TCP port used for the communication between the GUI and the Python interpreter. This value should not be modified unless you are asked by the MBS-support to change it.

The test scripts are designed to use the TCP port 11151, if you change this value, the included test scripts will no longer run properly! This function is reserved for future use.

12.6.2.4. Category Wireshark Settings
See separate chapter Wireshark Configuration earlier in this document.
12.6.3 Testscript Management

This page allows access to the test script database and the group management.

The two main pages may be selected by the TAB button.

12.6.3.1. Overview

This page provides an overview about the supported test scripts within the system database.

You may switch between the different standards within the database using the Combobox “BACnet Standard”.

Each database contains Standard tests (the tests provided within the factory database).
12.6.3.1.1. Standard Tests
This page shows the test scripts included in the factory database. The numbers are taken from the BTL test plan.

12.6.3.2. Group Management

The BACnet Test Framework includes pre-defined groups based on the different parts of the BTL test plan.

Own groups may be defined using this page.

Choose “New Group” to add a group to the database.

Enter the unique name for the group.
Select the group in the right window, select the test scripts you want to add to the group in the right window. You may select multiple scripts using the Shift-key or CTRL-key while selecting.

Click on the “Add Script” button in the middle of the page.

The selected scripts appear in the selected group in the right window.
Choose “Save Changes” to store the group settings.
12.6.4 History

This page shows the global system history. The history contains all actions taken by the BTF users on the system level.

On the project level another local history is available within each test project.
12.6.5 Licensing

This page shows the status of the BTF license. Free updates may be obtained from the BTF support. If the software maintenance contract is not renewed, BTF versions earlier than the expiration date will still run, but updates later than the date of renewal will not work anymore.

Licence keys are sent by E-Mail, both the hardware-key (dongle) and a valid licence file are required to run BTF. The process of Updates or Upgrades is made easier, because the license file contains all information about the version and the BTF-edition, while the dongle generally enables running BTF.

To support the dongle, a driver for the “Aladdin Hardlock” dongle is installed in the BTF-installation process.

Please note that both the dongle and the licence-file have the same reference number. If your company owns multiple licences, always make sure that dongle and licence file match exactly.
12.6.5.1. Import

License files for updates can be imported using the import function, the file-extension is *.lic.
12.7. Quit
Quits the application. Since all data is stored within the database, there are no files to store here. Unlike file-oriented applications, you will never be prompted to save your data when terminating the application.

12.8. Results

12.8.1 XML & Schema
Stores all test data, results and a copy of the EPICS files as an XML file and a schema file.

The example below shows a copy of an empty test project, test projects containing EPICS files for IUT, TD and OD, test plans and other configuration data are quite larger of course.

```xml
<?xml version="1.0" encoding="UTF-8" standalone="yes" ?>
  <BTF_TestResultRoot xmlns:msdata="urn:schemas-microsoft-com:xml-msdata"
      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
      xsi:noNamespaceSchemaLocation="BTF_TestResult.xsd">
    ...<LabData>
    ...<AddressOfLab>
      ...<Name>MBS GmbH</Name>
      ...<Street>Römerstraße 15</Street>
      ...<Postal>D-47809</Postal>
      ...<Town>Krefeld</Town>
      ...<Country>Deutschland</Country>
      ...<Phone>+49 / 21 51 / 72 94-0</Phone>
      ...<Fax>+49 / 21 51 / 72 94-50</Fax>
      ...<Mail>info@mbs-software.de</Mail>
      ...<Web>www.mbs-software.de</Web>
    </AddressOfLab>
    ...<HeadOfLab>
      ...<Surname>Mustermann</Surname>
      ...<Prename>Max</Prename>
    </HeadOfLab>
    <LabData>
      <ProjectData Name="My First Project" TestStandard="ANSI/ASHRAE 135-2004 BTL-Testplan 4.0" Creator="frs" TimeOfCreation="2007-05-12T11:20:14.9530000+02:00" />
    </LabData>
  </BTF_TestResultRoot>
```

The associated XML Schema file (extension ".XSD") allows access to the XML structure.
12.8.2  Word

 Starts Microsoft Word 2003 and opens the associated test report template file (extension .dot).

 After starting Word, the template file takes a while to store all test data within the report. After the hour glass disappears, the test report may be processed using Word and the text processing functions.

 In the standard version and premium version the report template may be adjusted to your own format. This function requires the professional version of Microsoft Word.
12.8.3 Info

The info screen shows information about the program release, the license number and the purchased software edition.
13. The main dialog

After creating a new project or opening an existing project, the main dialog of the BACnet Test Framework is shown.

1. The Program menu
   The program menu provides access to project functions, printout and program information.

2. The property page
   The property page provides detailed information about the selected test task.

3. The test task buttons
   The test task buttons provide access to the most common program functions for testing.

4. The information / execution screen
   This screen shows details of the selected test task.
14. The test tasks

The test tasks may be selected from the button stack in the bottom left part of the main screen.

The associated properties for each task are shown in the upper left part of the main screen, the associated information/execution screen is shown in the right half of the main screen.

According to the selected task, the contents of the properties page and information/execution screen may vary.
14.1. Project Properties
This page allows access to the general setup of the test project.

14.1.1 Information
The picture shows the content of the information/execution screen in information mode. The provided information is read-only.
14.1.2  Data
This page provides the information about the IUT, the different pages may be selected by selecting the tab button Data, Contact1 or Contact2.

This data is used when creating a XML or WORD report.

14.1.2.1.  MAC-Address IUT
The MAC-Address is to be entered in the hex-notation, e.g. ‘A8982032BAC0’, if the IUT is tested using a router which provides access to another network than the TD/OD network, you may add the network number separated with a comma, e.g. ‘2,BF’, which reads BACnet network number: 2, MAC-Address: BFhex = 191.

If adding BACnet/IP addresses, the software provides a helper for editing the IP-address as decimal values, choose the button . The helper dialog appears, which allows editing the 4 octets of the IP-address and the two octets for the UDP-port as decimal values.
14.1.2.2. **Clock resolution**
This value in milliseconds specifies the resolution of the IUT’s internal clock. The value is used for tests which set a time in the IUT, wait a certain period and compare the two timestamps (e.g. in Schedule-tests). If the IUT supports a timer resolution in seconds, the value is 1000 milliseconds (default). If your IUT supports another resolution, enter the appropriate value here.

14.1.2.3. **Don't use Make-Scripts**
This checkbox disables the MAKE/CHECK automation. If set, no Make-Scripts are executed. This is useful for BTF projects which are used for QA and for BTL conformance tests. Normally the project runs automatically using MAKE-automation, but in the BTL laboratory the Make-statement may be executed manually.
14.1.2.4. Contact 1 / Contact 2

The contact information 1 and 2 may be used to associate personal contact information to the test device, e.g. you may add the responsible software engineer under contact 1 and the responsible commercial manager under contact 2. Adding data here is optional, but may be a great help to have the address information available while testing. The data may be used in reports, too.
14.1.3  Notes

Notes and remarks may be added to the test project at any time.
The order of notes may be changed by selecting the associated headline.

The order direction (alphabetical upwards/downwards) may be changed by selecting the same headline twice.

The direction is displayed by the direction of the arrow.

14.1.4 History
The history shows all actions taken in the current project.

A double-click on a history content opens the detail dialog.
Category: EPICS Standard consistency tests
Date: 12.05.2007  Time: 20:32:11
User: fts
Process: EPICS Standard Consistency Test cancelled

Test results:
Test (a): passed
Test (b): failed
Test (c): failed
Test (d): passed
Test (e): failed
14.2. Importing EPICS files

The handling of importing the IUT, TD and OD EPICS is the same, so the procedure is explained using an IUT in the next pictures.

The IUT EPICS document is used to reference the initial values of the IUT’s database. The specification of EPICS files can be found in the standard ANSI/ASHRAE 135.1-2007 or ISO 16484-6. The format is not explained in this document. Reference EPICS-files are provided on the BTF support FTP-Server as template files.

Choose “Import” to select the IUT EPICS-file.
Select the IUT EPICS file and choose “Open”.

The EPICS files is scanned and imported into the test database. A copy of the EPICS is stored in the project directory.
After importing the EPICS the dialog displays the success by a green symbol.

In case of syntax errors within the EPICS file, the errors are shown in the dialog. Like other tools like compilers or text-scanner, correcting a single entry may cause follow-up errors in the next import session.
14.2.1 Very important note for TD and OD EPICS files

**Very important note:**

The device-instance number of the TD-device must be greater than the device-instance number of the OD. Otherwise, broadcast messages are sent using the OD network interface card and some tests may fail.

The TD and OD EPICS files are not allowed to contain wildcards, otherwise the simulation will not work. Property values may be modified in any kind of script files.

14.2.2 Displaying the contents of the EPICS

You may step through the EPICS contents anytime to lookup certain contents. The object tree of the device is shown in the property page in the upper left part, the associated contents of the selected object is shown in the right part of the page.

Please note, that modifying the EPICS here is prohibited! This is not a technical issue, it would have been quite easy to allow modification here. The reason for
the decision, not to allow modifications of EPICS files are based on the requirements for certification. An EPICS is a manufacturers’ self-declaration, which may not be modified within the test software.

14.2.2.1. Removing an EPICS

EPICS files may be removed from a test project by selecting the “Remove” button.

Please note that at least an IUT EPICS and a TD EPICS must be imported, a few tests require an additional OD.
14.3. EPICS Consistency Test

This page starts the EPICS consistency test described in ANSI/ASHRAE135.1 and ISO 16484-6 chapter 5.

The “Start” button starts the consistency test, the results may be viewed in the output window.
14.4. Conformance Tests

This page allows performing the conformance tests.

The procedure of testing conformance to BACnet is described in the next chapter.

14.5. Hiding the test task buttons

You may hide or show the test task buttons by selecting the arrow button in the test task button stack.
15. Testing conformance to BACnet

This chapter describes the process of testing the conformance to BACnet.

This process is divided into two sub-processes, the test plan configuration and the test plan execution.

15.1. Test plan configuration

This process defines the test scripts to be executed with the test plan, the parameters for the single tests and check for the test dependencies (if necessary).

You may create, edit or remove test plans here.

Please note, that editing a large test plan using “Edit” may take a while loading it in the configuration window.
15.1.1 CSV-export

Clicking the button exports all test plans to a comma-separated csv file.
15.1.2 Creating a new test plan

A new test plan may be created choosing “New” in the property window, while the Conformance Test mode is selected in the test task button stack.

In the following test plan configuration dialog you may edit the test plan.
15.1.3  Test Plan Configuration
This page allows editing the test plan to be executed later.

15.1.3.1.  The Test Plan Configuration Menus

1  Save: Saves the current test plan to the database.
2  Save as: Saves the current test plan prompting for a new name.
3  Rename Test Plan: Prompts for a new name of an existing test plan.
4  Check Test Plan: Performs a check, if all required parameters are defined and the dependencies of the current test plan are correct.
5  Don't use OD: The use of the OD (Other Device) for the test plan may be enabled or disabled here (toggle the button).
6  Close: Leaves the test plan configuration window, asking for saving the changes.
### 15.1.3.2. The properties page

The properties page of the test plan configuration window provides four different selections:

**Standard tests:** This is the list of all factory-scripts, which cover the requirement of the current BTL test plan. The names are taken from the chapter of the norm or from the BTL test plan.

The checkbox allows selection of tests which match the EPICS only.

![Screencap of the properties page with a list of standard tests]
**User Tests:** If you have imported user scripts to your database, you may select them here.

**Other Scripts:** This page is designated to select or manage other scripts like Prologscripts, Epilogscripts or Makescripts.

This function has changed in Version 3 of BTF. All scripts are now included in the project, there is no global pool for scripts anymore. Refer to the chapter “Important changes in BTF Version 3” earlier in this document.
Groups: The factory database contains pre-defined groups of tests according to the different chapters of the BTL test plan, which are ready to be used.
**Properties:** This page provides the list of Private Parameters for the selected test. Additionally a Prolog- and/or Epilog Script may be added to the test script here.

If you have imported Prolog-, Epilog- or Make script-files, you may assign them to the test script(s) here.

These changes apply to test scripts which have been added to the test plan only. Refer to the next chapter how to add tests to the test plan.
15.1.4 Adding scripts to the test plan

Adding test scripts to the test plan is provided by a drag-and-drop function. Select the test scripts you want to add in the page on the left (Shift-key and Control-key may be used for the selection as defined in Microsoft Windows).

Drag the mouse on one of the selected tests using the left mouse key and drop them over the test plan window.

The selected test scripts appear in the test plan window now.

Please note that selecting a large number of tests may take some time before they appear in the test plan window due to the necessary database access.
15.1.4.1. The test plan window

The test plan window contains the current list of test scripts to be executed in the test plan.

The list is a multi-select list, this allows easy configuration for Private Parameters which appear in all selected tests.

Single tests may be checked to be not executed in the test plan execution, but remain within the test plan.

Locked tests will not be executed unless unlocked, the result is locked.

The buttons on the right are used to move the selected test script(s) up or down or to remove them from the test plan.

The sequence number shows the order of the tests.
Right mouse click in the test plan window open a context menu:

1. **Edit dependencies**: Opens the dialog to edit the dependencies of the selected test script(s), see below.
2. **Group**: Groups the selected Test Scripts, a dialog prompts for the name of the new group.
3. **Collapse Group**: Collapses the current group.
4. **Expand Group**: Expands the current group.
5. **Script active (execute)**: Sets the execute flag for the selected test script(s).
6. **Script inactive (don't execute)**: Removes the execute flag of the selected test script(s).
7. **Toggle the execution flags**: Toggles the execute flag of all scripts in the test plan.
8. **Lock test**: If set the test will not be executed unless unlocked, the result will be locked.
9. **Unlock test**: The test will be executed again, the result will be overridden.
15.1.4.2. Edit Dependencies
The dependencies of the test scripts may be edited here, this means the list of test which have to pass prior to executing the dependant test itself.

The left window shows the list of dependencies, the right window shows the test script which fulfills the dependency. In the example above, no dependency was provided, so the test plan check would fail.

Use the function “Reload Missing Scripts Dependencies” to automatically load the dependent tests and assign them to the test script.

Please note, that the BTL test plan does not specify any dependencies while ANSI/ASHRAE 135.1 does. The common agreement in BTL-WG is to not use dependencies.

15.1.4.3. The details window

The Test goal, Test concept and Configuration requirements (if existing) are shown for the selected test. The dependencies windows shows dependencies to other tests (if existing).
15.2. Test plan execution

Once you defined a test plan, you may want to execute it.

The buttons below the test script window are used to control the execution of test plans.

**Uninterrupted execution**: This flag is used in Make-automation only. If the automation of a Make or Check statement using a Make script fails, the execution of further tests is normally stopped. If the flag is set, the respective test will be marked as failed, but subsequent tests will still be performed.

Start the “Play” Button to execute the selected test plan.
The execution of the test plan start and the results are shown in the Output Window. After execution, some tests may be marked with a red sign (failed) others may be marked with a green sign (passed).

Right click on a test opens a context menu

Your may change the test result manually, e.g. a failed test may be set to passed, due to a clarification of the norm or a passed test may be set to failed due to a reset after test execution.
When changing the test result, you may add a comment here.

Please note, the original test result is kept within the database, so it may be restored anytime.

15.2.1 Test Plan Control

You may set a single test active or inactive, restart all tests or restart the failed tests here. This allows changing the test script execution without editing the test plan in the configuration window.

If you restarted some tests, the log-window will contain the results of the tests restarted only. To see all results of the full test plan, please select the test-plan again.

15.2.2 The Output Window
The Output Window shows the test results.
Some test results may be large depending on the contents, double-clicking on the row opens a details window.

15.2.2.1. Copy Results to Clipboard

Right click on a row, opens a short context menu providing a copy to clipboard function, so you may copy all test output to the clipboard to process them in other applications.
15.3. Symbols used in test plan execution

The following symbols are used within the test plan execution:

- This icon is used, when a test was not (yet) performed or was stopped due to an internal problem.

- This icon is used, when a test failed.

- This icon is used, when a test is skipped and is not set to be executed.

- This icon is used, when a test result was manually overridden with the result “failed”.

- This icon is used, when a test result was manually overridden with the result “passed”.

- This icon is used, when a test was not executed. Tests may be skipped manually by the operator or automatically, when dependent tests previously failed.

- This icon is used, when a test successfully passed.

- This icon is used, when a test is locked.
Test Execution Control

The following symbols are used in the Test Execution Control

- **csv/xml export:** Allows to export the test-plan to a csv-file or xml-file.

- **Debugger:** Starts the test-plan in the debugger. Refer to the chapter “Debugging Scripts” later in this document.

- **Interrupt execution:** Interrupts the current test plan execution (suspend).

- **Single Step:** Allows performing the steps waiting for a confirmation for executing each step.

- **Test Step:** Jumps to the next test step within the test script.

- **Step Back:** Steps back to the last test step.

- **Next Test Script:** Jumps to the next script.

- **Execute:** Starts the test plan execution

- **Break:** Pauses the test plan execution.

- **Stop:** Stops the test plan execution
15.4. Test interaction

While executing the test scripts, a user-interaction may be displayed (depending on the test).

Client tests but also most of the server tests require an interaction to inform the user about the test requirements or the prompt for certain values.

The interaction window remains on top of all windows, this is useful, e.g. when interacting with a protocol sniffer or other tools on the same machine.

Prompting for values:

There are different dialogs and communications for the interaction, the picture shows an example of checking if a certain condition matched.

Please note, that all factory test scripts are designed that they expect the required telegram sent prior to confirming the dialog interaction. This allows executing the test steps without running into timing problems.
16. Debugging Scripts

In Version 3.1 a debugger support was added to BTF. The recommended debugging tool is the freeware PyScripter (refer to http://www.mmm-experts.com/Downloads.aspx?ProductId=4 for details).

The debugger path and filename is specified in the system administration.

<table>
<thead>
<tr>
<th>Category: Python-settings</th>
</tr>
</thead>
<tbody>
<tr>
<td>Timeout for Clienttests in seconds</td>
</tr>
<tr>
<td>Timeout for Python communication in seconds</td>
</tr>
<tr>
<td>Debugger</td>
</tr>
<tr>
<td>Python-Port</td>
</tr>
</tbody>
</table>

To start a test plan in the debugger, choose the debugger button from the test plan execution panel.

The debugger is called with a list of Python script files including the file Testplan.py

We recommend PyScripter version 1.7.2 (official release), in higher versions we noticed, that only the last file passed in the command-line is shown in PyScripter. As an alternative you may load the file Testplan.py manually.
The program PyScripter is started and all related files including prolog-, epilog-, user- and make-scripts are loaded into the debugger GUI.

The most important part to start debugging is the section at the end of the file Testplan.py, where the tests within the test plan are called (example marked in yellow below).

```python
bac_handl = PyBacNet( 4194302, None, 'A89820DEBAC0', 22201 )
pybacapi.SetBACnetCharstringEncodingTypeAndCodepage(pybacapi.STRING_ANSI_X34)
bac_handl.construct_iut_objects_from_propFeatures_dict(global_dict['IUT']['PropertyFeatures'],global_dict['IUT']['Restrictions'])
bac_handl.construct_objects_from_propFeatures_dict(global_dict['TD']['PropertyFeatures'])
test = test_7_3_1_1( bac_handl, gcom, global_dict, [(3,0),True], prolog = prolog_clear_outofservice_priorityarray.prolog_clear_outofservice_priorityarray, epilog = None, segno = 0, make_scripts = [], autonomic = False)
test = test_7_3_1_8( bac_handl, gcom, global_dict, [(3,0),True,1,1], prolog = None, epilog = None, segno = 4, make_scripts = [], autonomic = False)
```

Breakpoints may be set to the single test calls or directly in the prolog- or epilog-files. Like in other debuggers PyScripter supports single step operations, watch windows for variables, etc.

Please note to always execute the file “Testplan.py” all other files like prolog, etc. are called from this major script.

Please note that BTF has a general timeout (default: 5 minutes), after this period, the GUI terminates the execution. If necessary increase the value for this timeout in the system administration.
17. Command-line Interface

BTF provides a powerful command-line interface to remote control the software. This is useful, if BTF is about to be integrated in other QA-tools or test frameworks or you want to perform test-plan execution as a batch process. When called in the batch mode, the flag "uninterrupted execution" is automatically set for the test plans to be executed.

17.1. The command-line arguments

This chapter describes the command-line arguments passed to the file BTF.exe.

17.1.1 Language

The default language of BTF is German (we are a German company). To switch to English language, use the argument "-en". If BTF is installed in English language, this argument is automatically added to the desktop shortcut.

17.1.2 The Remote Control File (XML)

The batch mode is controlled using an XML file passed to BTF using the command-line argument "-f <path- and filename>".

Example XML-file, which contains all possible call options:

```xml
<?xml version = '1.0' encoding = 'UTF-8'?>
<btf_command_sequence>
  <options protocol="logging.txt" base_path="C:\BTF"/>
  <!--The attribute protocol defines the filename to store the log-output (normally shown in the log-window)-->  </options>
  <execute_test project="Example 1" testplan="Testplan 1.1"/>
  <!--The attribute testinstances defines the sequence numbers of the tests to be executed.-->
  <execute_test project="Example 1" testplan="Testplan 1.2" testinstances="1,3"/>
  <export_result project="Example 1" output_file="C:\BTF\Example 1.xml" result_type="xml="/>
  <export_project project="Example 1" file_name="C:\BTF\Example 1.btf"/>
  <execute_test project="Example 2" testplan="Testplan 2.1" override_iut_mac="0AFA5942BAC0" override_iut_id="50"/>
  <export_result project="Example 2" output_file="c:\BTF\Example 2.xml" result_type="xml="/>
  <export_project project="Example 2" file_name="c:\BTF\Example 2.btf"/>
  <execute_test project="Example 3" file_name="Testplan.csv"/>
</btf_command_sequence>
```
17.1.2.1. The Tag Structure

The XML tag structure is defined as follows:

- **btf_command_sequence**: Defines the call structure.
  - options: Defines the log-file.
    - protocol =<filename>
    - base_path =<Path to store the log-file>
  - execute_test: Defines the testplan (and tests) to be executed.
    - project = <name of the BTF project>
    - testplan = <name of the testplan to be executed>
    - testinstances [opt.] = <list of sequence numbers of tests to be executed>
    - override_iut_mac [opt.] = assigns a new MAC-Address of the IUT
    - override_iut_id [opt.] = assigns a new Device-ID of the IUT

- **export_result**: Defines the result output.
  - project = <name of the BTF project>
  - output_file = <path and name of the output file>
  - result_type = <xml> (at the moment only XML-export is supported)

- **export_project**: Defines the project export.
  - project = <name of the BTF project to be exported>
  - file_name = <path and name of the export file>

- **export_testplan**: Defines the test-plan export.
  - project = <name of the BTF project to be exported>
  - file_name = <path and name of the test-plan export file>

  *the extension “.csv” exports the test-plan as a comma-separated file, the extension “.xml” exports the test-plan as xml-file*

17.1.3 Return Values

BTF provides execution feedback using the following return values:

- **0**: Execution successful: All tests successfully passed.
- **1**: Error executing test: This error code is returned, if at least one test did not pass.
- **2**: Wrong command-line argument(s): This error code is returned, if wrong arguments are passed to BTF.exe.
- **3**: License Error: This error code is returned if no valid BTF license was found or the USB-dongle was missing.
- **4**: File-Error: This error code is returned, if the File I/O to the XML-file did not succeed.
- **5**: Syntax-Error(XML): This error code is returned, if the XML files contained an invalid syntax or unknown command.
- **6**: Execution error: This error code is returned, if at least one command in the XML file could not be executed.
- **7**: BTF already running: This error code is returned, if an instance of BTF is already running (BTF is a single instance software).
The log-file output of BTF.exe may be re-directed into a text log-file (see example batch call below).

17.2. Sample Batchfile

The following example shows how to remote control BTF and retrieve the possible error-codes.

```bash
rem ----------------------------------------------------
rem command_sample.bat
rem shows an example of the BTF command-line interface
rem 02.06.2009
rem MBS GmbH, Krefeld
rem ----------------------------------------------------
@echo off
"C:\Programme\MBS\BTF\bin\BTF.exe" -f "C:\btf-batch\commands.xml" > btflog.txt
@if "%ERRORLEVEL%" == "0" goto good
@if "%ERRORLEVEL%" == "1" goto badtest
@if "%ERRORLEVEL%" == "2" goto badarg
@if "%ERRORLEVEL%" == "3" goto badlicense
@if "%ERRORLEVEL%" == "4" goto badfile
@if "%ERRORLEVEL%" == "5" goto badxmlsyntax
@if "%ERRORLEVEL%" == "6" goto badcmd
@if "%ERRORLEVEL%" == "7" goto alreadyrunning

:fail
    echo Execution failed
    echo return value = %ERRORLEVEL%
    type btflog.txt
    goto end

:badtest
    echo At least one test failed
    echo return value = %ERRORLEVEL%
    goto end

:badarg
    echo failure in command-line arguments
    echo return value = %ERRORLEVEL%
    goto end

:badlicense
    echo failure in license
    echo return value = %ERRORLEVEL%
    goto end

:badfile
    echo failure file access
    echo return value = %ERRORLEVEL%
    goto end

:badxmlsyntax
    echo failure in xml syntax of command-file
    echo return value = %ERRORLEVEL%
    goto end

:badcmd
    echo failure to execute command
```
echo return value = %ERRORLEVEL%
goto end

:alreadyrunning
    echo BTF is already running
    echo return value = %ERRORLEVEL%
goto end

:good
    echo Execution Succeeded
    echo return value = %ERRORLEVEL%
goto end

:end
del btflog.txt
pause

Please note to replace the path “Programme” by “Program Files” if running an English windows version.
18. Make-Automation
This chapter describes the Make automation engine and contains examples of scripts providing responses to Make or Check statements.

This feature was added in BTF version 3.1. Example Python code for these scripts are part of the distribution (directory <BTF installation>\Documentation\SampleScripts).

18.1. Introduction
While the prolog-scripts are designated to setup the configuration requirements and the epilog-scripts are designated to perform a cleanup after a test run, the make scripts are implemented to respond to so-called Make- and Check-statements in tests.

A Make-statement requires an interaction by the tester, normally a manual step e.g. “MAKE the configuration in the IUT different, such that the Database_Revision property increments”.

A Check-statement requires an interaction by the tester to check, if a certain action was taken in the IUT, e.g. “CHECK Did the IUT perform a COLDSTART reboot?”.

This manual interaction can be automated by Make-scripts, which allow to take over the interaction. Normally this requires a second access mostly proprietary to the IUT by calling proprietary tools, accessing the IUT by using telnet or FTP, etc.

An Excel spreadsheet file containing a list of all Make- and Check-statements of the current BTL test plan is provided on the BTF FTP-Server.

18.2. Writing a Make Script
A make-script is based on the same call-structure as prolog-scripts and epilog-scripts. The main part beyond the class elements like self.current_testname, self.private_list, etc. is the dictionary “make_dict” passed to all make-scripts. This dictionary contains all necessary information to identify the type of the Make- or Check-statement.

The necessary elements to uniquely identify the statement are:

self.current_testname: This is important to identify the test itself, if a make-script is assigned to a number of different tests.

msg_num: Unique message number within the test, can be obtained by calling “self.make_dict["msg"]”
18.3. Types of messages

Five different types of messages are currently defined in the GUI-interface. The different types of GUI-communication are additionally documented in the GUI-Com documentation available on the BTF support FTP-Server.

18.3.1 Make without type

This message is sent to simply return an OK-message to the GUI after successful Make or Check.

A typical call (may be used in user-scripts, too) is shown here:

```
ret_val = gcom.commit(7, {'msg':102, 'descr':'do MAKE without type'})
```

The return code (in the make-script) is shown here:

```python
# MAKE Simple OK acknowledge
if not self.make_dict.has_key('type):
    return {'answer':'1'}
```

18.3.2 Make input

This message is sent to request a value from the IUT.

A typical call (may be used in user-scripts, too) is shown here:

```
ret_val = gcom.commit(7, {'msg':108, 'descr':'MAKE input', 'type':'input', 'value':''})
```

The return code (in the make-script) is shown here:

```python
elif self.make_dict['type'] == 'input':
    # with value only
    if not self.make_dict.has_key('tolerance'):
        value = '123'
        return {'value': value}
```

18.3.3 Make input with tolerance

This message is sent to request a value from the IUT including a tolerance.

A typical call (may be used in user-scripts, too) is shown here:

```
ret_val = gcom.commit(7, {'msg':110, 'descr':'MAKE input with tolerance', 'type':'input', 'value':'', 'tolerance':''})
```

The return code (in the make-script) is shown here (marked in yellow):

```python
elif self.make_dict['type'] == 'input':
    # with value only
    if not self.make_dict.has_key('tolerance'):
        value = '123'
        return {'value': value}
    # with value and tolerance
    else:
        value = '123'
        tolerance = '0.1'
        return {'value':value, 'tolerance':tolerance}
```

18.3.4 Make choice

This message is sent to request a Yes or No, if a certain condition in the IUT matches.

A typical call (may be used in user-scripts, too) is shown here:

```
ret_val = gcom.commit(7, {'msg':104, 'descr':'MAKE choice', 'type':'choice'})
```
The return code (in the make-script) is shown here:

```python
elif self.make_dict['type'] == 'choice':
    # Choice YES/NO
    condition = True  # dummy value, this should reflect the real condition
    if condition is False:
        return {'answer': '0'}  # NO
    if condition is True:
        return {'answer': '1'}  # YES
```

### 18.3.5 Make choice with ref_val

This message is sent to request a Yes or No, if a certain condition in the IUT matches the value references by ref_val.

A typical call (may be used in user-scripts, too) is shown here:

```python
ret_val = gcom.commit(7, {'msg': 106, 'descr': 'MAKE choice with ref_val', 'type': 'choice', 'ref_val': '123'})
```

The return code (in the make-script) is shown here (ref_val part marked in yellow):

```python
elif self.make_dict['type'] == 'choice':
    # Choice YES/NO
    if self.make_dict.has_key('ref_val'):
        # with additional reference value
        # this demo code just sends back the ref_val to the GUI, normally
        # the value would be used to compare to the value in the IUT
        self.gcom.commit(3, {'step': 15, 'descr': 'dummy MAKE make_dict -> ref_val:' + str(self.make_dict['ref_val'])})
    condition = True  # dummy value for testing purposes
    if condition is False:
        return {'answer': '0'}  # NO
    if condition is True:
        return {'answer': '1'}  # YES
```

### 18.4 Importing a Make Script

Make-scripts are imported into the BTF project the same way as prolog- or epilog-files.

Select Make to open the file selector to import the make-script.
18.5. Assigning a Make Script

Make-scripts are assigned to tests in the test plan configurator. Select the Make-script the same way as prolog- or epilog-files.

18.6. Enabling/Disabling the Make automation

The project property page now contains a flag, which enables / disables execution of make-scripts. Check or uncheck the flag to decide whether to execute make-scripts or not. This function is useful, when using the same project for conformance tests in a BTL-Laboratory as well using it for QA-processes.
18.7. Flag Uninterrupted Execution

This flag is located left from the test execution panel.

If this flag is set, an error returned from the Make automation engine (e.g. problems running the Make-script or performing/retrieving the desired action from the IUT) result in the test to fail and the execution to continue with the next test without showing the Make/Check dialog. This allows complete automation without any user-interaction. If BTF is called using the remote call interface, this flag is automatically set.

If the flag is not set, errors in executing the make-script result in stopping the test plan execution and displaying the Make/Check dialog, requesting the desired user-action.
18.8. Execution Flowchart

The following picture shows the test plan execution flowchart.
19. Reading and Writing proprietary data

This chapter shows how to use the BTF Raw-Interface to form proprietary telegrams and send or receive them to or from the BACnet network.

19.1. Proprietary ReadProperty Example

The following example shows a standalone test performing a proprietary ReadProperty (Device-Object 123, Prop-ID: 701).

```python
def raw_test():
    objType = objType_SN['device']  # todo: define the object_type here
    instNum = 123                   # todo: define the IUT instance number here
    propID = 701                    # todo: define your proprietary property here
    Fail_Time = 3
    invoke_id = 0
    routerNeeded = False
    routerAddress = "C0A8A1B2BAC0"

    try:
        if routerNeeded:
            bac_handl = PyBacNet( 2001, None,'2,09',instNum) # IUT over router
        else:
            bac_handl = PyBacNet( 2001, None,"A8982002BAC0",instNum) # IUT directly w/o router
    except Exception, s:
        print s
        return

    APDU_read_prop = bacnet_apdu()
    APDU_read_prop.set_pdu_type_and_seg_mor_sa(pdu_type_SN['confirmed_request'])
    APDU_read_prop.set_max_segs_max_resp(16,1472) #16 segments, max_apdu_size=1476
    APDU_read_prop.set_invoke_id(invoke_id)
    APDU_read_prop.set_service_choice(conf_service_choice_SN['readProperty'])
    SERVICE_REQ_read_prop = bacnet_apdu_service_request().make_service_request_read_property(objType, instNum, propID)
    APDU_read_prop.set_service_request(SERVICE_REQ_read_prop)

    # router address on my net
    if routerNeeded:
        NPDU_read_prop = bacnet_npdu(APDU_read_prop.get_apdu_as_hex_string())
        npdu_address = bac_handl.iut_pybac_address.convert_to_bacnet_address()
        bac_handl.h_send_raw_frame_from_TD(npdu_address, hex_npdu = NPDU_read_prop.get_npdu_as_hex_string())
    else:
        router_pybac_address = PyBacAddress(bac_handl.iut_pybac_address)
        bac_handl.h_send_raw_frame_from_TD(router_pybac_address.convert_to_bacnet_address(), hex_npdu = hex_npdu)

    count = bac_handl.h_get_raw_count()  # todo: define the IUT instance number here
    print s
```

---

19. Reading and Writing proprietary data

This chapter shows how to use the BTF Raw-Interface to form proprietary telegrams and send or receive them to or from the BACnet network.

19.1. Proprietary ReadProperty Example

The following example shows a standalone test performing a proprietary ReadProperty (Device-Object 123, Prop-ID: 701).
if count == 0:
    raise Exception, 'No response from device'
for index in range(count):
    raw_frame = bac_handl.h_get_raw_frame(index)
    npdu_dict = bacnet_npdu().decode_npdu(raw_frame.pszPythonHexDataStringNpdu)
    if npdu_dict['hex_apdu'] == None: continue  # if network_layer msg
    apdu_dict = bacnet_apdu().decode_apdu(npdu_dict['hex_apdu'])
    if apdu_dict['pdu_type'] != pdu_type_SN['complexACK']:
        continue
    if apdu_dict['service_choice'] !=
        conf_service_choice_SN['readProperty']:
        continue
    if apdu_dict['invoke_id'] != invoke_id: continue
    break
else:
    raise Exception, 'No response from device'
finally:
    raise Exception, 'No response from device'

print 'NPDU', npdu_dict
print 'APDU', apdu_dict

obj, inst, prop, index, request_str =
    bacnet_apdu_service_request().decode_service_request_read_property(apdu_dict['hex_service_request'], True)
print obj, inst, prop, index, request_str

#~BEGIN PRIVLIST SCHEMA~
private_list_description = {
    1:{'name':'RouterNeeded','type':'BOOLEAN','default':False,'description':'If the UIT is only reachable via BACnet router, this has to be set to True.'},
    2:{'name':'MAC Address','type':'OctetString','description':'MAC address of the router which routes to network of IUT.'},
}
#~END PRIVLIST SCHEMA~

class user_12345(OOBasisTest):
    priv_list_descr = private_list_description
    scriptname = '''user_12345 RAW write proprietary property '''
    purpose = '''''
    configreq = '''''
    concept = '''''

    def init_run_test(self,bac_handl, gcom, global_dict, private_list):
        #~-------------------------------------------
        routerNeeded, routerAddress = private_list
        objType = objType_SN['device']
        instNum = 89
        propID = propID_SN['Description']
        #=>
        SERVICE_REQ_write_prop = "0C0140000119553E91003F"
        Fail_Time = 3
        invoke_id = 0

19.2. Proprietary WriteProperty Example

The following example shows a usertest performing a proprietary WriteProperty request. This example can be found in the BTF distribution in the documentation section.

from basis_test import *

#~BEGIN PRIVLIST SCHEMA~
private_list_description = {
    1:{'name':'RouterNeeded','type':'BOOLEAN','default':False,'description':'If the UIT is only reachable via BACnet router, this has to be set to True.'},
    2:{'name':'MAC Address','type':'OctetString','description':'MAC address of the router which routes to network of IUT.'},
}
#~END PRIVLIST SCHEMA~

class user_12345(OOBasisTest):
    priv_list_descr = private_list_description
    scriptname = '''user_12345 RAW write proprietary property '''
    purpose = '''''
    configreq = '''''
    concept = '''''

    def init_run_test(self,bac_handl, gcom, global_dict, private_list):
        #~-------------------------------------------
        routerNeeded, routerAddress = private_list
        objType = objType_SN['device']
        instNum = 89
        propID = propID_SN['Description']
        #=>
        SERVICE_REQ_write_prop = "0C0140000119553E91003F"
        Fail_Time = 3
        invoke_id = 0
APDU_write_prop = bacnet_apdu()
APDU_write_prop.set_pdu_type_and_seg_mor_sa(pdu_type_SN['confirmed_request'])
APDU_write_prop.set_max_segs_max_resp(16,1472) #16 segments, max_apdu_size 1476
APDU_write_prop.set_invoke_id(invoke_id)
APDU_write_prop.set_service_choice(conf_service_choice_SN['writeProperty'])

APDU_write_prop.set_service_request(SERVICE_REQ_write_prop)
NPDU_write_prop = bacnet_npdu(APDU_write_prop.get_apdu_as_hex_string())
if routerNeeded:
    NPDU_write_prop.set_network_layer_protocol_control_info(bit2 = 1, bit5 = 1)
    router_pybac_address = PyBacAddress(routerAddress, bac_handl.td_pybac_address.net)  # router address on my net
else:
    NPDU_write_prop.set_network_layer_protocol_control_info(bit2 = 1)
    router_pybac_address = None
    hex_npdu = NPDU_write_prop.get_npdu_as_hex_string()

input_list = [hex_npdu, routerNeeded, router_pybac_address, Fail_Time, invoke_id]
self.test_body(bac_handl, gcom, input_list)

#----------------------------------------------------------------------------------
def test_body(self,bac_handl, gcom, input_list):
    hex_npdu, routerNeeded, router_pybac_address, Fail_Time, invoke_id = input_list
    gcom.commit(2, {'objType':'/','instNum': '/'})
    gcom.commit(3, {'step':101, 'descr': 'Send WP Request to IUT'})
    bac_handl.h_start_raw_frame_queue( queue_size = 2000)
    if routerNeeded:
        bac_handl.h_send_raw_frame_from_TD(hex_npdu, Destination = router_pybac_address.convert_to_bacnet_address())
    else:
        bac_handl.h_send_raw_frame_to_iut(hex_npdu)
    time.sleep(Fail_Time)
    gcom.commit(3, {'step':102, 'descr': 'Receive SimpleACK from IUT'})
    try:
        count = bac_handl.h_get_raw_count()
        if count == 0:
            raise Exception, 'No response from device'
        for index in range(count):
            raw_frame = bac_handl.h_get_raw_frame(index)
            npdu_dict = bacnet_npdu().decode_npdu(raw_frame.pszPythonHexDataStringNpdu)
            if npdu_dict['hex_apdu'] == None: continue    # if network_layer msg
            apdu_dict = bacnet_apdu().decode_apdu(npdu_dict['hex_apdu'])
            if apdu_dict['pdu_type'] != pdu_type_SN['simpleACK']: continue
            if apdu_dict['service_choice'] != conf_service_choice_SN['writeProperty'] :continue
            if apdu_dict['invoke_id'] != invoke_id: continue
            break
        else:
            raise Exception, 'No response from device'
    finally:
        bac_handl.h_free_raw_frames()
    gcom.commit(4, {'status':1})
#----------------------------------------------------------------------------------
20. The recommended network setting

1. Windows PC running the BACnet Test Framework, additional installation on this PC may be a network sniffer like Wireshark.

2. First and primary network interface connection from BTF to the IUT running the TD.
   - **IP-Address:** 168.152.32.1
   - **Subnetmask:** 255.255.255.0
   - **Default Gateway:** no Default Gateway

3. Second network interface connection from BTF to the IUT running the OD.
   - **IP-Address:** 168.152.33.1
   - **Subnetmask:** 255.255.255.0
   - **Default Gateway:** no Default Gateway
4 Main network connection, provided by either a network hub (typically 10 Mbit/s) or a managed switch providing a monitor port.

5 Second PC running any kind of BACnet-Client-Software, e.g. the BACnet OPC-Server available from MBS, VTS or a B-OWS
   - **IP-Address:** 168.152.32.254
   - **Subnetmask:** 255.255.255.0
   - **Default Gateway:** 168.152.32.1

6 IUT = Implementation under Test, the device to be tested on conformance to BACnet.
   - **IP-Address:** 168.152.32.Vendor-ID
   - **Subnetmask:** 255.255.255.0
   - **Default Gateway:** 168.152.32.1

In this configuration it is very important, that the IUT will be set using the IUT’s IP-Address as the Default-Gateway, Windows will route between the two networks 168.152.32.x and 168.152.33.x.
20.1. Setting up the BACnet Protocol-Stack

The configuration file “C:\Programme\MBS\BTF\bin\BacStack\bacstackd.ini” contains the setup for the BACnet Protocol Stack.

The factory default file contains the following settings and is configured to run two network cards.

```
# bacstac.ini
# BacStack configuration file for the BACnet Test Framework
# MBS GmbH
# (C) 2007
# www.mbs-software.de
#
# defines the general settings for this file
#
[general] # general settings for this file
cnt_data_link = 2 # number of data-link-layers defined in this file
max_msg_req = 100 # message queue between physical layer and application

# defines the physical network connection for BACnet/IP for TD
[data_link 1] # first data-link-layer
port_id = 1 # port-no. 1
type = ip # physical BACnet/IP network
lan_name = 168.152.32.1 # IP-address of the TD (the BACnet Test Framework)
net_number = 1 # BACnet network-no. for the physical network
udp_port = 47808 # BACnet-port
max_msg_req = 50 # message queue between the physical layers

# defines the FD for the TD-Layer
#
# Please note that only BBMD or FD may be defined!
#
# If both is configured, the stack will not run properly.

#foreign_dev_addr = 168.152.32.254:47808 # ip/port address of BBMD to register to
#foreign_dev_reg_timeout = 10 # re-register-time in seconds

# defines the physical network connection for BACnet/IP for OD
[data_link 2] # second data-link-layer
port_id = 2 # port-no. 2
type = ip # physical BACnet/IP network
lan_name = 168.152.33.1 # IP-address of the OD (simulated device by the BACnet Test Framework)
net_number = 2 # BACnet network-no. for the physical network
```
udp_port = 47808  # BACnet-port
max_msg_req = 50   # message queue between the physical layers

#########################################################################
# BBMD/FD support
#
# Please note that only BBMD or FD may be defined!
#
# If both is configured, the stack will not run properly.
#
#########################################################################

#########################################################################
# defines the BBMD
#########################################################################

# [bbmd]  # BBMD configuration
# twohop   = YES  # one or two hop message for own BBMD
# cnt_bbmd = 1  # number of external BBMD entries
# addr_0  = 168.152.32.254:47808  # ip/port address of first external BBMD
# mask_0  = 255.255.255.255  # distribution mask of first external BBMD

Please follow the rules shown as comments for setting the network settings and the BBMD/FD support.

If you only have one network card and you do not want to run OD-tests, you may set the cnt_data_link in the general section to 1.
20.2. Error starting the test plan

If you receive an error-message as shown, below, please check the BACnet Protocol Stack setting.

You may additionally try to start the service manually. Start the service "BacStack Daemon" by double-clicking on it.

Please don’t forget to stop the service afterwards. If the start fails, an appropriate error message explaining the failure reason will be displayed.
21. The process identifier for events and alarms

In alarm- and event tests the process identifier in the TD-device is set to the value: 100. If you need to set the entry for the Recipient List Property of a notification class object, you must use process-identifier 100.

The other device OD uses process-identifier 101!

Sample Python Code to register in NotificationClass set in the variable “InstNumNotifClass”:

```
self.action_register_in_notification_class( int( InstNumNotifClass ) )
```
### 22. Conventions for usage of wildcards

Abstract: Most test scripts require values to compare the original EPICS content with the actual property value. For this, a convention was taken, where property values may contain wildcard values (if required) and property values which must contain valid property values to test against.

The following table shows, where wildcards are allowed and where not.

Please note, that this table applies to the IUT EPICS only. The TD and OD EPICS files shall not contain any wildcards (these are designated to simulate server devices, so wildcards are not allowed there).

<table>
<thead>
<tr>
<th>Property</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>accepted-modes</td>
<td>y single questionmark allowed as property-value</td>
</tr>
<tr>
<td>acked-transitions</td>
<td>n no questionmark allowed due to usage for tests</td>
</tr>
<tr>
<td>ack-required</td>
<td>- property-identifier used only for services</td>
</tr>
<tr>
<td>action</td>
<td></td>
</tr>
<tr>
<td>action-text</td>
<td></td>
</tr>
<tr>
<td>active-text</td>
<td></td>
</tr>
<tr>
<td>active-vt-sessions</td>
<td>y</td>
</tr>
<tr>
<td>active-cov-subscriptions</td>
<td>y</td>
</tr>
<tr>
<td>adjust-value</td>
<td>n</td>
</tr>
<tr>
<td>alarm-value</td>
<td>n</td>
</tr>
<tr>
<td>alarm-values</td>
<td>n</td>
</tr>
<tr>
<td>all</td>
<td>-</td>
</tr>
<tr>
<td>all-write-successful</td>
<td>y</td>
</tr>
<tr>
<td>apdu-segment-timeout</td>
<td>n</td>
</tr>
<tr>
<td>apdu-timeout</td>
<td>n</td>
</tr>
<tr>
<td>application-software-version</td>
<td>n</td>
</tr>
<tr>
<td>archive</td>
<td>y</td>
</tr>
<tr>
<td>attempted-samples</td>
<td>y</td>
</tr>
<tr>
<td>auto-slave-discovery</td>
<td>n</td>
</tr>
<tr>
<td>average-value</td>
<td>y</td>
</tr>
<tr>
<td>backup-failure-timeout</td>
<td>n</td>
</tr>
<tr>
<td>bias</td>
<td>n</td>
</tr>
<tr>
<td>buffer-size</td>
<td>n</td>
</tr>
<tr>
<td>change-of-state-count</td>
<td>y</td>
</tr>
<tr>
<td>Property</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>------------------------------------------</td>
</tr>
<tr>
<td>change-of-state-time</td>
<td></td>
</tr>
<tr>
<td>client-cov-increment</td>
<td>the property in this place was deleted</td>
</tr>
<tr>
<td>configuration-files</td>
<td></td>
</tr>
<tr>
<td>controlled-variable-reference</td>
<td></td>
</tr>
<tr>
<td>controlled-variable-units</td>
<td></td>
</tr>
<tr>
<td>controlled-variable-value</td>
<td></td>
</tr>
<tr>
<td>count</td>
<td></td>
</tr>
<tr>
<td>count-before-change</td>
<td></td>
</tr>
<tr>
<td>count-change-time</td>
<td></td>
</tr>
<tr>
<td>cov-increment</td>
<td></td>
</tr>
<tr>
<td>cov-period</td>
<td></td>
</tr>
<tr>
<td>cov-resubscription-interval</td>
<td></td>
</tr>
<tr>
<td>current-notify-time</td>
<td>This property was deleted in version 1</td>
</tr>
<tr>
<td>database-revision</td>
<td>revision 3.</td>
</tr>
<tr>
<td>date-list</td>
<td></td>
</tr>
<tr>
<td>daylight-savings-status</td>
<td></td>
</tr>
<tr>
<td>deadband</td>
<td></td>
</tr>
<tr>
<td>derivative-constant</td>
<td></td>
</tr>
<tr>
<td>derivative-constant-units</td>
<td></td>
</tr>
<tr>
<td>description</td>
<td></td>
</tr>
<tr>
<td>description-of-halt</td>
<td></td>
</tr>
<tr>
<td>device-address-binding</td>
<td></td>
</tr>
<tr>
<td>device-type</td>
<td></td>
</tr>
<tr>
<td>direct-reading</td>
<td></td>
</tr>
<tr>
<td>effective-period</td>
<td>asterisk (X'2A') * allowed as wildcard</td>
</tr>
<tr>
<td>elapsed-active-time</td>
<td></td>
</tr>
<tr>
<td>error-limit</td>
<td></td>
</tr>
<tr>
<td>event-enable</td>
<td></td>
</tr>
<tr>
<td>event-state</td>
<td></td>
</tr>
<tr>
<td>event-time-stamps</td>
<td></td>
</tr>
<tr>
<td>event-type</td>
<td></td>
</tr>
<tr>
<td>event-parameters</td>
<td>renamed from previous version</td>
</tr>
<tr>
<td>exception-schedule</td>
<td></td>
</tr>
<tr>
<td>fault-values</td>
<td></td>
</tr>
<tr>
<td>feedback-value</td>
<td></td>
</tr>
<tr>
<td>file-access-method</td>
<td></td>
</tr>
<tr>
<td>file-size</td>
<td></td>
</tr>
<tr>
<td>file-type</td>
<td></td>
</tr>
<tr>
<td>firmware-revision</td>
<td></td>
</tr>
<tr>
<td>high-limit</td>
<td></td>
</tr>
<tr>
<td>inactive-text</td>
<td></td>
</tr>
<tr>
<td>Property</td>
<td>In</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>----</td>
</tr>
<tr>
<td>in-process</td>
<td>y</td>
</tr>
<tr>
<td>input-reference</td>
<td>n</td>
</tr>
<tr>
<td>instance-of</td>
<td>n</td>
</tr>
<tr>
<td>integral-constant</td>
<td>n</td>
</tr>
<tr>
<td>integral-constant-units</td>
<td>n</td>
</tr>
<tr>
<td>issue-confirmed-notifications</td>
<td>n</td>
</tr>
<tr>
<td>last-notify-record</td>
<td>y</td>
</tr>
<tr>
<td>last-restore-time</td>
<td>y</td>
</tr>
<tr>
<td>life-safety-alarm-values</td>
<td>n</td>
</tr>
<tr>
<td>limit-enable</td>
<td>n</td>
</tr>
<tr>
<td>limit-monitoring-interval</td>
<td>n</td>
</tr>
<tr>
<td>list-of-group-members</td>
<td>n</td>
</tr>
<tr>
<td>list-of-object-property-references</td>
<td>n</td>
</tr>
<tr>
<td>list-of-session-keys</td>
<td>y</td>
</tr>
<tr>
<td>local-date</td>
<td>y</td>
</tr>
<tr>
<td>local-time</td>
<td>y</td>
</tr>
<tr>
<td>location</td>
<td>n</td>
</tr>
<tr>
<td>log-buffer</td>
<td>y</td>
</tr>
<tr>
<td>log-device-object-property</td>
<td>n</td>
</tr>
<tr>
<td>log-enable</td>
<td>y</td>
</tr>
<tr>
<td>log-interval</td>
<td>n</td>
</tr>
<tr>
<td>logging-object</td>
<td>n</td>
</tr>
<tr>
<td>logging-record</td>
<td>y</td>
</tr>
<tr>
<td>low-limit</td>
<td>n</td>
</tr>
<tr>
<td>maintenance-required</td>
<td>y</td>
</tr>
<tr>
<td>manipulated-variable-reference</td>
<td>n</td>
</tr>
<tr>
<td>manual-slave-address-binding</td>
<td>n</td>
</tr>
<tr>
<td>maximum-output</td>
<td>n</td>
</tr>
<tr>
<td>maximum-value</td>
<td>y</td>
</tr>
<tr>
<td>maximum-value-timestamp</td>
<td>y</td>
</tr>
<tr>
<td>max-apdu-length-accepted</td>
<td>n</td>
</tr>
<tr>
<td>max-info-frames</td>
<td>n</td>
</tr>
<tr>
<td>max-master</td>
<td>n</td>
</tr>
<tr>
<td>max-pres-value</td>
<td>n</td>
</tr>
<tr>
<td>max-segments-accepted</td>
<td>n</td>
</tr>
<tr>
<td>member-of</td>
<td>n</td>
</tr>
<tr>
<td>minimum-off-time</td>
<td>n</td>
</tr>
<tr>
<td>minimum-on-time</td>
<td>n</td>
</tr>
<tr>
<td>minimum-output</td>
<td>n</td>
</tr>
<tr>
<td>minimum-value</td>
<td>y</td>
</tr>
<tr>
<td>minimum-value-timestamp</td>
<td>y</td>
</tr>
<tr>
<td>min-pres-value</td>
<td>n</td>
</tr>
<tr>
<td>Property</td>
<td>Required</td>
</tr>
<tr>
<td>----------------------------------------------</td>
<td>----------</td>
</tr>
<tr>
<td>mode</td>
<td>y</td>
</tr>
<tr>
<td>model-name</td>
<td>n</td>
</tr>
<tr>
<td>modification-date</td>
<td>y</td>
</tr>
<tr>
<td>notification-class</td>
<td>n</td>
</tr>
<tr>
<td>notification-threshold</td>
<td>n</td>
</tr>
<tr>
<td>notify-type</td>
<td>n</td>
</tr>
<tr>
<td>number-of-APDU-retries</td>
<td>n</td>
</tr>
<tr>
<td>number-of-states</td>
<td>n</td>
</tr>
<tr>
<td>object-identifier</td>
<td>n</td>
</tr>
<tr>
<td>object-list</td>
<td>n</td>
</tr>
<tr>
<td>object-name</td>
<td>n</td>
</tr>
<tr>
<td>object-property-reference</td>
<td>n</td>
</tr>
<tr>
<td>object-type</td>
<td>n</td>
</tr>
<tr>
<td>operation-expected</td>
<td>y</td>
</tr>
<tr>
<td>out-of-service</td>
<td>y</td>
</tr>
<tr>
<td>output-units</td>
<td>n</td>
</tr>
<tr>
<td>polarity</td>
<td>n</td>
</tr>
<tr>
<td>prescale</td>
<td>n</td>
</tr>
<tr>
<td>present-value</td>
<td>y</td>
</tr>
<tr>
<td>previous-notify-time</td>
<td>y</td>
</tr>
<tr>
<td>priority</td>
<td>n</td>
</tr>
<tr>
<td>pulse-rate</td>
<td>y</td>
</tr>
<tr>
<td>priority-array</td>
<td>y</td>
</tr>
<tr>
<td>priority-for-writing</td>
<td>n</td>
</tr>
<tr>
<td>process-identifier</td>
<td>n</td>
</tr>
<tr>
<td>profile-name</td>
<td>y</td>
</tr>
<tr>
<td>program-change</td>
<td>y</td>
</tr>
<tr>
<td>program-location</td>
<td>y</td>
</tr>
<tr>
<td>program-state</td>
<td>y</td>
</tr>
<tr>
<td>proportional-constant</td>
<td>n</td>
</tr>
<tr>
<td>proportional-constant-units</td>
<td>n</td>
</tr>
<tr>
<td>protocol-conformance-class</td>
<td>n</td>
</tr>
<tr>
<td>protocol-object-types-supported</td>
<td>n</td>
</tr>
<tr>
<td>protocol-revision</td>
<td>n</td>
</tr>
<tr>
<td>protocol-services-supported</td>
<td>n</td>
</tr>
<tr>
<td>protocol-version</td>
<td>n</td>
</tr>
<tr>
<td>read-only</td>
<td>n</td>
</tr>
<tr>
<td>reason-for-halt</td>
<td>y</td>
</tr>
<tr>
<td>recipient</td>
<td>y</td>
</tr>
<tr>
<td>recipient-list</td>
<td>y</td>
</tr>
<tr>
<td>records-since-notification</td>
<td>y</td>
</tr>
<tr>
<td>Feature</td>
<td>Value</td>
</tr>
<tr>
<td>------------------</td>
<td>-------</td>
</tr>
<tr>
<td>record-count</td>
<td>y</td>
</tr>
<tr>
<td>reliability</td>
<td>y</td>
</tr>
<tr>
<td>relinquish-default</td>
<td>y</td>
</tr>
<tr>
<td>resolution</td>
<td>n</td>
</tr>
<tr>
<td>scale</td>
<td>n</td>
</tr>
<tr>
<td>scale-factor</td>
<td>n</td>
</tr>
<tr>
<td>schedule-default</td>
<td>n</td>
</tr>
<tr>
<td>segmentation-supported</td>
<td>n</td>
</tr>
<tr>
<td>setpoint</td>
<td>y</td>
</tr>
<tr>
<td>setpoint-reference</td>
<td>n</td>
</tr>
<tr>
<td>slave-address-binding</td>
<td>y</td>
</tr>
<tr>
<td>setting</td>
<td>y</td>
</tr>
<tr>
<td>silenced</td>
<td>n</td>
</tr>
<tr>
<td>start-time</td>
<td>n</td>
</tr>
<tr>
<td>state-text</td>
<td>y</td>
</tr>
<tr>
<td>status-flags</td>
<td>n</td>
</tr>
<tr>
<td>stop-time</td>
<td>n</td>
</tr>
<tr>
<td>stop-when-full</td>
<td>y</td>
</tr>
<tr>
<td>system-status</td>
<td>y</td>
</tr>
<tr>
<td>time-delay</td>
<td>n</td>
</tr>
<tr>
<td>time-of-active-time-reset</td>
<td>y</td>
</tr>
<tr>
<td>time-of-state-count-reset</td>
<td>n</td>
</tr>
<tr>
<td>time-synchronization-recipients</td>
<td>y</td>
</tr>
<tr>
<td>total-record-count</td>
<td>n</td>
</tr>
<tr>
<td>tracking-value</td>
<td>y</td>
</tr>
<tr>
<td>units</td>
<td>n</td>
</tr>
<tr>
<td>update-interval</td>
<td>n</td>
</tr>
<tr>
<td>update-time</td>
<td>y</td>
</tr>
<tr>
<td>utc-offset</td>
<td>y</td>
</tr>
<tr>
<td>valid-samples</td>
<td>y</td>
</tr>
<tr>
<td>value-before-change</td>
<td>y</td>
</tr>
<tr>
<td>value-set</td>
<td>y</td>
</tr>
<tr>
<td>value-change-time</td>
<td>y</td>
</tr>
<tr>
<td>variance-value</td>
<td>y</td>
</tr>
<tr>
<td>vendor-identifier</td>
<td>n</td>
</tr>
<tr>
<td>vendor-name</td>
<td>n</td>
</tr>
<tr>
<td>vt-classes-supported</td>
<td>n</td>
</tr>
<tr>
<td>weekly-schedule</td>
<td>n</td>
</tr>
<tr>
<td>window-interval</td>
<td>n</td>
</tr>
<tr>
<td>window-samples</td>
<td>n</td>
</tr>
<tr>
<td>zone-members</td>
<td>n</td>
</tr>
</tbody>
</table>
23. Sample EPICS-file for IUT

This chapter shows a sample EPICS file for the IUT.

>>>>>> The file starts after this line-----------------------------------------------

PIICS 0

BACnet Protocol Implementation Conformance Statement

Vendor Name: "TODO: Add Vendor Name here"
Product Name: "TODO: Add Product Name here"
Product Model Number: "TODO: Add Product Model Number here"
Product Description: "TODO: Add Description here"

BIBBs Supported:
{
  DS-RP-A
  DS-RP-B
  DS-RPM-A
  DS-RPM-B
  DS-WP-A
  DS-WP-B
  DS-WPM-B
  DS-COV-A
  DS-COV-B
  DS-COVU-A
  DS-COVU-B
  DS-COV-P
  AE-N-I-B
  AE-N-E-B
  AE-ACK-B
  AE-ASUM-B
  AE-ESUM-B
  AE-INFO-B
  SCHED-I-B
  SCHED-E-B
  T-VMT-I-B
  T-VMT-E-B
  T-ATR-B
  DM-DDB-A
  DM-DDB-B
  DM-DOB-A
  DM-DOB-B
  DM-DCC-B
  DM-TS-A
  DM-TS-B
  DM-UTC-A
  DM-UTC-B
  DM-RD-B
  DM-RR-B
  DM-LM-B
  DM-OCD-B
  NM-CE-A
  NM-CE-B
  NM-RC-A
  NM-RC-B
}

BACnet Standard Application Services Supported:
{
  AtomicReadFile  Execute
  AtomicWriteFile  Execute
  AcknowledgeAlarm  Execute
  ConfirmedCOVNotification  Initiate Execute
  ConfirmedEventNotification  Initiate
  GetEnrollmentSummary  Execute
  GetEventInformation  Execute
}
GetAlarmSummary    Execute
SubscribeCOV       Initiate Execute
SubscribeCOVProperty Execute
AddListElement     Execute
RemoveListElement  Execute
CreateObject       Execute
DeleteObject       Execute
ReadProperty       Initiate Execute
ReadPropertyMultiple Initiate Execute
WriteProperty      Initiate Execute
WritePropertyMultiple Execute
DeviceCommunicationControl Execute
ReinitializeDevice Execute
I-Am               Initiate Execute
I-Have             Initiate Execute
UnconfirmedCOVNotification Initiate Execute
UnconfirmedEventNotification Initiate
TimeSynchronization Initiate Execute
UTCTimeSynchronization Initiate Execute
Who-Has            Initiate Execute
Who-Is             Initiate Execute
ReadRange          Execute

Standard Object Types Supported:
{  
  Accumulator
  Analog Input
  Analog Output
  Analog Value
  Averaging
  Binary Input
  Binary Output
  Binary Value
  Calendar
  Command
  Device
  Event Enrollment
  File
  Group
  Loop
  Multi-state Input
  Multi-state Output
  Multi-state Value
  Notification Class
  Program
  Pulse Converter
  Schedule
  Trend Log
}

Data Link Layer Option:
{  
  BACnet/IP, 'DIX' Ethernet
  -- Point-To-Point. Modem, Baud rate(s): 115.2k
}

Character Sets Supported:
{  
  ANSI X3.4
}

Special Functionality:
{  
  Maximum APDU size in octets: 1476
  Segmented Requests Supported, window size: 16
  Segmented Responses Supported, window size: 16
  BACnet/IP BBMD
}

Default Property Value Restrictions:
{  
unsigned-integer: <minimum: 0; maximum: 4294967295>
signed-integer: <minimum: -2147483647; maximum: 2147483647>
real: <minimum: -3.40282347E38; maximum: 3.40282347E38; resolution: 0.01>
double: <minimum: 2.2250738585072016E-38; maximum: 1.7976931348623157E38; resolution: 0.0001>
date: <minimum: 01-January-1970; maximum: 31-December-2038>
octet-string: <maximum length string: 256>
character-string: <maximum length string: 64>
list: <maximum length list: 10>
variable-length-array: <maximum length array: 10>
}

Fail Times:
{
  Notification Fail Time: 30
  Internal Processing Fail Time: 2
  Minimum ON/OFF Time: 2
  Schedule Evaluation Fail Time: 2
  External Command Fail Time: 5
  Program Object State Change Fail Time: 5
  Acknowledgement Fail Time: 2
}

List of Objects in test device:
{
  
  {  
      object-identifier: (accumulator, 1)
      object-name: "Accumulator 1"
      object-type: accumulator
      present-value: ?
      description: "Accumulator"
      device-type: "Switch I3"
      status-flags: {FALSE, FALSE, FALSE, FALSE}
      event-state: normal
      reliability: NO-FAULT-DETECTED
      out-of-service: False W
      scale: 1.0
      units: KILOWATT_HOURS
      prescale: (1,1)
      max-pres-value: 10000
      value-change-time : ?
      value-before-change : 0
      value-set : 0 W
      pulse-rate : 1
      high-limit : 1000
      low-limit : 0
      limit-monitoring-interval : 10
      notification-class : 1
      time-delay : 0
      limit-enable: {TRUE, TRUE} W
      event-enable: {TRUE, TRUE, TRUE} W
      acked-transitions: {FALSE, FALSE, FALSE}
      notify-type: ALARM
      event-time-stamps: ?
    },

    {  
      object-identifier: (analog-input, 1)
      object-name: "Analog Input 1"
      object-type: analog-input
      present-value: ?
      description: "Analog Input"
      device-type: "DeviceType"
      status-flags: {FALSE, FALSE, FALSE, FALSE}
      event-state: normal
      reliability: NO-FAULT-DETECTED
      out-of-service: False W
      units: PERCENT
      min-pres-value: 0.0
      max-pres-value: 100.0
      resolution: 0.1
      COV-increment: 12.5
    }
  
}
time-delay: 2  
notification-class: 1  
high-limit: 80.0  
low-limit: 20.0  
deadbend: 1.0  
limit-enable: (TRUE, TRUE) W  
event-enable: (TRUE, TRUE, TRUE) W  
acked-transitions: (FALSE, FALSE, FALSE)  
notify-type: ALARM  
event-time-stamps: ?  
},  
{  
object-identifier: (analog-output, 1)  
object-name: "Analog Output 1"  
object-type: analog-output  
present-value: ? W  
description: "Analog Output"  
device-type: "Device Type"  
status-flags: (FALSE, FALSE, FALSE, FALSE)  
event-state: normal  
reliability: NO-FAULT-DETECTED  
out-of-service: False W  
units: PERCENT  
min-pres-value: 0.0  
max-pres-value: 100.0  
resolution: 0.1  
priority-array: {NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL}  
relinquish-default: 50.0  
COV-increment: 12.5  
time-delay: 2  
notification-class: 1  
high-limit: 80.0  
low-limit: 20.0  
deadbend: 1.0  
limit-enable: (TRUE, TRUE) W  
event-enable: (TRUE, TRUE, TRUE) W  
acked-transitions: (FALSE, FALSE, FALSE)  
notify-type: ALARM  
event-time-stamps: ?  
},  
{  
object-identifier: (analog-value, 1)  
object-name: "Analog Value 1"  
object-type: analog-value  
present-value: ? W  
description: "Analog Value"  
status-flags: (F, F, F, F)  
event-state: normal  
reliability: NO-FAULT-DETECTED  
out-of-service: False W  
units: PERCENT  
min-pres-value: 0.0  
max-pres-value: 100.0  
resolution: 0.1  
priority-array: {NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL}  
relinquish-default: 50.0  
COV-increment: 12.5  
time-delay: 2 W  
notification-class: 1  
high-limit: 80.0  
low-limit: 20.0  
deadbend: 1.0  
limit-enable: (TRUE, TRUE) W  
event-enable: (TRUE, FALSE, TRUE) W  
acked-transitions: (TRUE, FALSE, TRUE)  
notify-type: EVENT  
event-time-stamps: ?  
},  
{  
object-identifier: (averaging, 1)  
object-name: "Averaging 1"  
object-type: AVERAGING
minimum-value: ?
minimum-value-timestamp: ?
average-value: ?
maximum-value: ?
maximum-value-timestamp: ?
description: "Averaging"
attempted-samples: ? W
valid-samples: ?
object-property-reference: {analog-input, 1}, present-value
window-interval: 10 W
window-samples: 10 W

{
  object-identifier: (binary-input, 1)
  object-name: "Binary Input 1"
  object-type: binary-input
  present-value: ?
  description: "Binary Input"
  device-type: "Device Type"
  status-flags: [FALSE, FALSE, FALSE, FALSE]
  event-state: NORMAL
  reliability: NO-FAULT-DETECTED
  out-of-service: FALSE W
  polarity: NORMAL W
  inactive-text: "Off"
  active-text: "On"
  change-of-state-time: ? W
  change-of-state-count: ?
  time-of-state-count-reset: ?
  elapsed-active-time: ? W
  time-of-active-time-reset: ?
  time-delay: 2
  notification-class: 1
  alarm-value: ACTIVE
  event-enable: [TRUE, FALSE, TRUE] W
  acked-transitions: [FALSE, TRUE, TRUE]
  notify-type: EVENT
  event-time-stamps: ?
}

{
  object-identifier: (binary-output, 1)
  object-name: "Binary Output 1"
  object-type: BINARY-OUTPUT
  present-value: ? W
  description: "Binary Output"
  device-type: "Device Type"
  status-flags: [FALSE, FALSE, FALSE, FALSE]
  event-state: NORMAL
  reliability: NO-FAULT-DETECTED
  out-of-service: FALSE W
  polarity: NORMAL W
  inactive-text: "Off"
  active-text: "On"
  change-of-state-time: ? W
  change-of-state-count: ?
  time-of-state-count-reset: ?
  elapsed-active-time: ? W
  time-of-active-time-reset: ?
  minimum-off-time: 10
  minimum-on-time: 10
  priority-array: [NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL]
  relinquish-default: INACTIVE
  time-delay: 2
  notification-class: 1
  feedback-value: INACTIVE
  event-enable: [TRUE, FALSE, TRUE] W
  acked-transitions: [FALSE, TRUE, TRUE]
  notify-type: EVENT
  event-time-stamps: ?
}
{ 
  object-identifier: (binary-value, 1)
  object-name: "Binary Value 1"
  object-type: BINARY-VALUE
  present-value: \( ? \) W
  description: "Binary Value"
  status-flags: \{FALSE,FALSE,FALSE,FALSE\}
  event-state: NORMAL
  reliability: NO-FAULT-DETECTED
  out-of-service: FALSE W
  inactive-text: "Off"
  active-text: "On"
  change-of-state-time: \( ? \) W
  change-of-state-count: \?
  time-of-state-count-reset: \?
  elapsed-active-time: \( ? \) W
  time-of-active-time-reset: \?
  minimum-off-time: 10
  minimum-on-time: 10
  priority-array: \{NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL\}
  relinquish-default: INACTIVE
  time-delay: 2
  notification-class: 1
  alarm-value: ACTIVE
  event-enable: \{TRUE, FALSE, TRUE\} W
  acked-transitions: \{FALSE, TRUE, TRUE\}
  notify-type: EVENT
  event-time-stamps: ?
},

{ 
  object-identifier: (calendar,1)
  object-name: "Calendar 1"
  object-type: CALENDAR
  description: "Calendar"
  present-value: True
},

{ 
  object-identifier: (command,1)
  object-name: "Command Object 1"
  object-type : COMMAND
  description : "Command"
  present-value : ? W
  in-process : FALSE
  all-writes-successful : TRUE
  action : (((((device,91),(Analog-Output,1),present-value,0,65.400000,12,2,TRUE,TRUE),((device,91),(Binary-Output,1),present-value,0,INACTIVE,12,2,TRUE,TRUE))),((((device,91),(Analog-Output,1),present-value,0,20.500000,12,2,TRUE,TRUE),((device,91),(Binary-Output,1),present-value,0,ACTIVE,12,2,TRUE,TRUE))))
  action-text : ("Unoccupied","Occupied")
},

{ 
  object-identifier: (device, 1)
  object-name: "Device 1"
  object-type: device
  system-status: OPERATIONAL
  vendor-name: "Vendor"
  vendor-identifier: 0
  model-name: "Modelname"
  firmware-revision: "Firmware Revision"
  application-software-version: "Application Software Version"
  location: "Testlabor"
  description: "IUT for Conformance Tests WSPLab Stuttgart"
  protocol-version: 1
  protocol-revision: 4
  protocol-object-types-supported: \( (T, T, T, T, T, T, T, T, T, T, T, T, T, T, T, T, T, T, T, T, T, T, T, T) \)
  object-list:
  \{ 
    (accumulator, 1),
  \}
(analog-input, 1),
(analog-output, 1),
(analog-value, 1),
(averaging, 1),
(binary-input, 1),
(binary-output, 1),
(binary-value, 1),
(calendar, 1),
(command, 1),
(device, 1),
(event-enrollment, 1),
(group, 1),
(loop, 1),
(multi-state-input, 1),
(multi-state-output, 1),
(multi-state-value, 1),
(notification-class, 1),
(program, 1),
(pulse-converter, 1),
(schedule, 1),
(trend-log, 1)
}
max-APDU-length-accepted: 1476
segmentation-supported: segmented-both
max-segments-accepted: 16
local-time: ?
local-date: ?
utc-offset: -60
daylight-savings-status: ?
time-synchronization-recipients: () W
apdu-segment-timeout: 2000
apdu-timeout: 3000
number-of-APDU-retries: 5
device-address-binding: ?
database-revision: 4711
configuration-files: {(file, 1), (file, 2), (file, 3)}
last-restore-time: ?
backup-failure-timeout: 300 W
active-cov-subscriptions: ?
},
{  
object-identifier: (event-enrollment, 1)
object-name: "Zone1_Alarm"
object-type: EVENT_ENROLLMENT
description: "Zone 1 Alarms"
event-type: OUT_OF_RANGE
notify-type: ALARM
event-parameters: {30, 65.0, 85.0, 0.25}
object-property-reference: {(analog-input, 1), present-value}
event-state: HIGH_LIMIT
event-enable: [TRUE, TRUE, TRUE]
acked-transitions: [FALSE, TRUE, TRUE]
notification-class: 1
},
{  
object-identifier: (group, 1)
object-name: "Group object 1"
object-type: GROUP
description: "Group"
list-of-group-members: (  
  { (analog-output, 1), (present-value, status-flags) },
  { (analog-value, 1), (present-value, status-flags) } ) W
present-value: ?
},
{  
object-identifier: (loop, 1)
object-name: "LOOP 1"
object-type: LOOP
present-value: ?
description: "LOOP"
status-flags: (? , ?, ? , ?)
event-state: NORMAL
reliability: NO_FAULT_DETECTED
out-of-service: FALSE
update-interval: 1
output-units: Percent
manipulated-variable-reference: {(analog-value, 2), present-value}
controlled-variable-reference: {(analog-input, 2), present-value}
controlled-variable-value: ?
controlled-variable-units: Percent
setpoint-reference: {((analog-value, 3), present-value)}
setpoint: ?
action: REVERSE
proportional-constant: 23.4
proportional-constant-units: Percent
integral-constant: 27.9
integral-constant-units: Percent
derivative-constant: 50.34
derivative-constant-units: Percent
bias: 0.4
maximum-output: 100.0
minimum-output: 0.0
priority-for-writing: 16
COV-increment: 1.0
time-delay: 0
notification-class: 1
event-enable: {TRUE, TRUE, TRUE}
acked-transitions: {?, ?, ?}
notify-type: ALARM
event-time-stamps: ?

object-identifier: (multi-state-input, 1)
object-name: "Multi State Input 1"
object-type: MULTI-STATE-INPUT
present-value: ?
description: "Multi State Input"
device-type: "Device Type"
status-flags: {FALSE,FALSE,FALSE,FALSE}

notification-class: 1
alarm-values: (2)

object-identifier: (multi-state-output, 1)
object-name: "Multi State Output 1"
object-type: MULTI-STATE-OUTPUT
present-value: ?
description: "Multi State Output"
device-type: "Device Type"
status-flags: {FALSE,FALSE,FALSE,FALSE}

notification-class: 1

feedback-value: ?
event-enable: {TRUE, TRUE, TRUE} W
acked-transitions: {FALSE, FALSE, FALSE}
notify-type: ALARM
event-time-stamps: ?

{ object-identifier: (multi-state-value, 1)
  object-name: "Multi State Value 1"
  object-type: MULTI-STATE-VALUE
  present-value: ? W
  description: "Multi State Value"
  status-flags: [FALSE, FALSE, FALSE, FALSE]
  event-state: NORMAL
  reliability: NO_FAULT_DETECTED
  out-of-service: FALSE W
  number-of-states: 3
  state-text: ["Off", "On_Low", "On_High"]
  priority-array: [NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL]
  relinquish-default: 1
  time-delay: 2
  notification-class: 1
  alarm-values: (2)
  fault-values: (3)
  event-enable: {TRUE, TRUE, TRUE} W
  acked-transitions: {FALSE, FALSE, FALSE}
  notify-type: ALARM
  event-time-stamps: ?
}

{ object-identifier: (notification-class, 1)
  object-name: "Notification Class 1"
  object-type: NOTIFICATION-CLASS
  description: "Notification Class"
  notification-class: 1
  priority: {1, 3, 3}
  ack-required: {FALSE, FALSE, FALSE}
  recipient-list: () W
}

{ object-identifier: (program, 1)
  object-name: "PCD Program"
  object-type: PROGRAM
  program-state: RUNNING
  program-change: READY W
  status-flags: [FALSE, FALSE, FALSE, FALSE]
  out-of-service: FALSE
}

{ object-identifier: (pulse-converter, 1)
  object-name: "Pulse Converter 1"
  object-type: PULSE-CONVERTER
  description: "Pulse Converter"
  present-value: ? W
  event-state: NORMAL
  status-flags: [?,?,?,?]
  reliability: ?
  out-of-service: FALSE W
  units: KILOWATT_HOURS
  scale-factor: 12.5
  adjust-value: 0.0 W
  count: 0
  count-before-change: 1
  cov-increment: 1.0
  cov-period: 3600
  notification-class: 1
  time-delay: 0
  high-limit: 90
  low-limit: 5
  deadband: 2
limit-enable: {TRUE, TRUE} W
event-enable: {TRUE, TRUE, TRUE} W
acked-transitions: {FALSE, FALSE, FALSE}
notify-type: ALARM
event-time-stamps: ?
update-time: ?

{  
  object-identifier: (schedule, 1)
  object-name: "Schedule 1"
  object-type: SCHEDULE
  present-value: ?
  description: "Schedule"
  effective-period: {([5-September-1995],[10-June-2009])}
  weekly-schedule: {  
    {(*::*:00,0.5),(*:*:10,10.5),(*:*:20,20.5),(*:*:30,30.5),(*:*:40,40.5)},
    {(*::*:00,0.5),(*:*:10,10.5),(*:*:20,20.5),(*:*:30,30.5),(*:*:40,40.5)},
    {(*::*:00,0.5),(*:*:10,10.5),(*:*:20,20.5),(*:*:30,30.5),(*:*:40,40.5)},
    {(*::*:00,0.5),(*:*:10,10.5),(*:*:20,20.5),(*:*:30,30.5),(*:*:40,40.5)},
    {(*::*:00,0.5),(*:*:10,10.5),(*:*:20,20.5),(*:*:30,30.5),(*:*:40,40.5)},
    {(*::*:00,0.5),(*:*:10,10.5),(*:*:20,20.5),(*:*:30,30.5),(*:*:40,40.5)}
  }

  exception-schedule: {  
    {([23-November-1995],[0:00,12.1]),10},
    {([calendar, 1],[0:00,13.2]),11},
    {([5-March-1996],[7-March-1996]),([9:00,14.3],[14:00,15.4]),6}
  }

  schedule-default: 34.7
  list-of-object-property-references: {([analog-value, 1],present-value,,(device,1))}
  priority-for-writing: 12
  status-flags: {FALSE,FALSE,FALSE,FALSE,FALSE}
  reliability: NO_FAULT_DETECTED
  out-of-service: FALSE W
}

{  
  object-identifier: (trend-log, 1)
  object-name: "Trendlog 1"
  object-type: trend-log
  description: "Trendlog"
  log-enable: TRUE W
  start-time: {(*-*-*), *:*:*.*} W
  stop-time: {(*-*-*), *:*:*.*} W
  log-device-object-property: {([binary-input,0],present-value,,(device,1))} W
  log-interval: 0 W
  cov-resubscription-interval: 3600
  client-cov-increment: 1.0
  stop-when-full: FALSE
  buffer-size: 1000
  log-buffer: ?
  record-count: 0 W
  total-record-count: ?
  notification-threshold: 10
  records-since-notification: ?
  last-notify-record: 0
  event-state: ?
  notification-class: 1
  event-enable: {TRUE, TRUE, TRUE} W
  acked-transitions: {FALSE, FALSE, FALSE, FALSE}
  notify-type: ALARM
  event-time-stamps: ?
}

End of BACnet Protocol Implementation Conformance Statement

-- end of file --
24. Sample EPICS-file for TD

This chapter shows a sample EPICS file for the TD.

>>> The file starts after this line-----------------------------------------------------

PICS 0
BACnet Protocol Implementation Conformance Statement

Vendor Name: "MBS GmbH"
Product Name: "Simulated Server for TD"
Product Model Number: "Any Model Number"
Product Description: "Simulator only"

BIBBs Supported:
{
  DS-RP-A
  DS-RP-B
  DS-RPM-A
  DS-RPM-B
  DS-WP-A
  DS-WP-B
  DS-WPM-B
  DS-COV-A
  DS-COV-B
  DS-COVU-A
  DS-COVU-B
  AE-N-I-B
  AE-ACK-B
  AE-ASUM-B
  AE-ESUM-B
  AE-INFO-B
  SCHED-I-B
  SCHED-E-B
  DM-DDB-A
  DM-DDB-B
  DM-DOB-A
  DM-DOB-B
  DM-DCC-B
  DM-TS-A
  DM-TS-B
  DM-UTC-A
  DM-UTC-B
  DM-RD-B
  DM-LM-B
  DM-OCD-B
  NM-CE-A
  NM-CE-B
}

BACnet Standard Application Services Supported:
{
  AcknowledgeAlarm   Execute
  ConfirmedCOVNotification Initiate Execute
  ConfirmedEventNotification Initiate Execute
  GetEnrollmentSummary   Execute
  GetEventInformation    Execute
  GetAlarmSummary        Execute
  SubscribeCOV          Initiate Execute
  AddListElement        Execute
  RemoveListElement     Execute
  CreateObject          Execute
  DeleteObject          Execute
  ReadProperty          Initiate Execute
  ReadPropertyMultiple  Initiate Execute
  WriteProperty         Initiate Execute
  WritePropertyMultiple Execute

Roemerstrasse 15
D-47809 Krefeld
GmbH
Fax: +49 / 21 51 / 72 94 - 50
e-mail: info@mbs-software.de
Telephone: +49 / 21 51 / 72 94 - 0
Internet: http://www.mbs-software.de

129
DeviceCommunicationControl       Execute
ReinitializeDevice               Execute
I-Am                             Initiate Execute
I-Have                           Initiate Execute
UnconfirmedCOVNotification       Initiate Execute
UnconfirmedEventNotification     Initiate Execute
TimeSynchronization             Initiate Execute
UTCTimeSynchronization          Initiate Execute
Who-Has                          Initiate Execute
Who-Is                           Initiate Execute
ReadRange                        Execute

Standard Object Types Supported:
{ Accumulator
Analog Input
Analog Output
Analog Value
Averaging
Binary Input
Binary Output
Binary Value
Calendar
Command
Device
Event Enrollment
File
Group
Loop
Multi-state Input
Multi-state Output
Multi-state Value
Notification Class
Program
Pulse Converter
Schedule
Trend Log

Data Link Layer Option:
{ BACnet/IP, 'DIX' Ethernet

Character Sets Supported:
{ ANSI X3.4

Special Functionality:
{ Maximum APDU size in octets: 1476
Segmented Requests Supported, window size: 16
Segmented Responses Supported, window size: 16

Default Property Value Restrictions:
{ unsigned-integer; <minimum: 0; maximum: 4294967295>
signed-integer: <minimum: -2147483647; maximum: 2147483647>
decl: <minimum: -3.40282347E38; maximum: 3.40282347E38; resolution: 0.01>
double: <minimum: 2.2250738585072016E-38; maximum: 1.7976931348623157E38; resolution: 0.0001>
date: <minimum: 01-January-1970; maximum: 31-December-2038>
octet-string: <maximum length string: 256>
character-string: <maximum length string: 64>
list: <maximum length list: 10>
variable-length-array: <maximum length array: 10>

Fail Times:
{  
Notification Fail Time: 30  
Internal Processing Fail Time: 2  
Minimum ON/OFF Time: 2  
Schedule Evaluation Fail Time: 2  
External Command Fail Time: 5  
Program Object State Change Fail Time: 5  
Acknowledgement Fail Time: 2  
}  

-- please note:  
-- in BTF, the TD device must have a device-ID greater than the OD device-ID  

List of Objects in test device: 
{
  
  
  
  
  object-identifier: (device, 4194302)  
  object-name: "Dummyserver for TD"  
  object-type: device  
  system-status: OPERATIONAL  
  vendor-name: "MBS GmbH"  
  vendor-identifier: 50  
  model-name: "Dummyserver Simulation"  
  firmware-revision: "No Firmware"  
  application-software-version: "No Application Software Version"  
  location: "Testlab"  
  description: "TD for Conformance Tests WSPLab Stuttgart"  
  protocol-version: 1  
  protocol-revision: 4  
T, T, F, T, T)  
  object-list:  
  {
    (device, 4194302),  
    (accumulator, 1),  
    (analog-input, 1),  
    (analog-output, 1),  
    (analog-value, 1),  
    (averaging, 1),  
    (binary-input, 1),  
    (binary-output, 1),  
    (binary-value, 1),  
    (calendar, 1),  
    (command, 1),  
    (event-enrolment, 1),  
    (group, 1),  
    (loop, 1),  
    (multi-state-input, 1),  
    (multi-state-output, 1),  
    (multi-state-value, 1),  
    (notification-class, 1),  
    (program, 1),  
    (pulse-converter, 1),  
    (schedule, 1),  
    (trend-log, 1)  
  }  
  max-APDU-length-accepted: 1476  
  segmentation-supported: segmented-both  
  max-segments-accepted: 16  
  local-time: 12:15:22.99  
  local-date: (Monday,30-April-2007)  
  utc-offset: -60  
  daylight-savings-status: FALSE  
  apdu-segment-timeout: 2000  
  apdu-timeout: 3000  
  number-of-APDU-retries: 5  
  device-address-binding: ()  
  database-revision: 1  
  active-cov-subscriptions: ()  
}
{ 
  object-identifier: (accumulator, 1) 
  object-name: "Accumulator 1" 
  object-type: accumulator 
  present-value: 1 
  description: "Accumulator" 
  device-type: "counter" 
  status-flags: {FALSE, FALSE, FALSE, FALSE} 
  event-state: normal 
  reliability: NO-FAULT-DETECTED 
  out-of-service: False 
  scale: 1.0 
  units: KILOWATT_HOURS 
  prescale: (1,1) 
  max-pres-value: 10000 
  value-before-change: 0 
  value-set: 0 W 
  pulse-rate: 1 
  high-limit: 1000 
  low-limit: 0 
  limit-monitoring-interval: 10 
  notification-class: 1 
  time-delay: 0 
  limit-enable: {TRUE, TRUE, TRUE} W 
  event-enable: {TRUE, TRUE, TRUE} W 
  acked-transitions: {FALSE, FALSE, FALSE} 
  notify-type: ALARM 
  event-time-stamps: {0}0,0,0,0 
} 
{ 
  object-identifier: (analog-input, 1) 
  object-name: "Analog Input 1" 
  object-type: analog-input 
  present-value: 47.3 
  description: "Analog Input" 
  device-type: "Device Type" 
  status-flags: {FALSE, FALSE, FALSE, FALSE} 
  event-state: normal 
  reliability: NO-FAULT-DETECTED 
  out-of-service: False 
  units: PERCENT 
  min-pres-value: 0.0 
  max-pres-value: 100.0 
  resolution: 0.1 
  COV-increment: 1.0 
  time-delay: 2 W 
  notification-class: 1 
  high-limit: 80.0 
  low-limit: 20.0 
  deadband: 1.0 W <minimum: 0; maximum: 60; resolution: 0.1> 
  limit-enable: {TRUE, TRUE, TRUE} W 
  event-enable: {TRUE, TRUE, TRUE} W 
  acked-transitions: {FALSE, FALSE, FALSE} 
  notify-type: ALARM 
  event-time-stamps: {0}0,0,0,0 
} 
{ 
  object-identifier: (analog-output, 1) 
  object-name: "Analog Output 1" 
  object-type: analog-output 
  present-value: 22.5 W 
  description: "Analog Output" 
  device-type: "Device Type" 
  status-flags: {FALSE, FALSE, FALSE, FALSE} 
  event-state: normal 
  reliability: NO-FAULT-DETECTED 
  out-of-service: False 
  units: PERCENT 
  min-pres-value: 0.0 
  max-pres-value: 100.0
resolution: 0.1
priority-array: {NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL}

relinquish-default: 50.0
COV-increment: 1.5
time-delay: 2
notification-class: 1
high-limit: 80.0
low-limit: 20.0
deadband: 1.0
limit-enable: {TRUE, TRUE, TRUE} W
event-enable: {TRUE, TRUE, TRUE} W
acked-transitions: {FALSE, FALSE, FALSE}
notify-type: ALARM
event-time-stamps: ([1][0],[1][0],[1][0])

object-identifier: (analog-value, 1)
object-name: "Analog Value 1"
object-type: analog-value
present-value: 17.6 W
description: "Analog Value"
status-flags: (F, F, F, F)
event-state: normal
reliability: NO-FAULT-DETECTED
out-of-service: False W
units: PERCENT
priority-array: {NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL}

relinquish-default: 50.0
COV-increment: 1.5
time-delay: 2
notification-class: 1
high-limit: 80.0
low-limit: 20.0
deadband: 1.0
limit-enable: {TRUE, TRUE, TRUE} W
event-enable: {TRUE, FALSE, TRUE} W
acked-transitions: {TRUE, TRUE, TRUE}
notify-type: EVENT
event-time-stamps: ([1][0],[1][0],[1][0])

object-identifier: (averaging, 1)
object-name: "Averaging 1"
object-type: AVERAGING
minimum-value: 10.4
average-value: 56.7
maximum-value: 68.7
maximum-value-timestamp: {(24-10-2006), 23:22:17.68}
description: "Averaging"
attempted-samples: 55 W
valid-samples: 55
object-property-reference: {(analog-input, 1), present-value}
window-interval: 10 W
window-samples: 10 W

object-identifier: (binary-input, 1)
object-name: "Binary Input 1"
object-type: binary-input
present-value: inactive
description: "Binary Input"
device-type: "Device Type"
status-flags: {FALSE, FALSE, FALSE, FALSE}
event-state: NORMAL
reliability: NO-FAULT-DETECTED
out-of-service: FALSE W
polarity: NORMAL W
inactive-text: "Off"
{ 
    object-identifier: (binary-output, 1)
    object-name: "Binary Output 1"
    object-type: BINARY-OUTPUT
    present-value: inactive W
    description: "Binary Output"
    device-type: "Device Type"
    status-flags: {FALSE,FALSE,FALSE,FALSE}
    event-state: NORMAL
    reliability: NO-FAULT-DETECTED
    out-of-service: FALSE W
    polarity: NORMAL W
    inactive-text: "Off"
    active-text: "On"
    change-of-state-count: 66
    elapsed-active-time: 34 W
    minimum-off-time: 10
    minimum-on-time: 10
    priority-array: {NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL}
    relinquish-default: INACTIVE
    time-delay: 2
    notification-class: 1
    feedback-value: INACTIVE
    event-enable: {TRUE, FALSE, TRUE} W
    acknowledged-transitions: {FALSE, TRUE, TRUE}
    notify-type: EVENT
    event-time-stamps: ([1][0],[1][0],[1][0])
}

{ 
    object-identifier: (binary-value, 1)
    object-name: "Binary Value 1"
    object-type: BINARY-VALUE
    present-value: inactive W
    description: "Binary Value"
    status-flags: {FALSE,FALSE,FALSE,FALSE}
    event-state: NORMAL
    reliability: NO-FAULT-DETECTED
    out-of-service: FALSE W
    inactive-text: "Off"
    active-text: "On"
    change-of-state-count: 23
    elapsed-active-time: 234 W
    minimum-off-time: 10
    minimum-on-time: 10
    priority-array: {NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL}
    relinquish-default: INACTIVE
    time-delay: 2
    notification-class: 1
    feedback-value: INACTIVE
    alarm-value: ACTIVE
event-enable: {TRUE, FALSE, TRUE} W
acked-transitions: {FALSE, TRUE, TRUE}
notify-type: EVENT
event-time-stamps: ([1]0,[1]0,[1]0)
}

object-identifier: (calendar,1)
object-name: "Calendar 1"
object-type: CALENDAR
description: "Calendar"
present-value: True
}

object-identifier: (command,1)
object-name: "Command Object 1"
object-type : COMMAND
description : "Command"
present-value : 0 W
in-process : FALSE
all-writes-successful : TRUE
action :

(((,(binary-output,1),present-value,,active,8,5,F,F),(,(analog-output,0),present-value,,80,8,5,F,F)),
 (((,(binary-output,0),present-value,active,8,5,T,F),(device,4711),(accumulator,4711),prescale,,NULL,8,5,T,F),(,(analog-output,0),present-value,20,8,5,T,F)))),
 (((,(binary-output,0),present-value,active,8,5,F,F),(device,4711),(accumulator,4711),prescale,,NULL,8,5,F,F),(,(binary-output,0),present-value,active,8,5,F,F)));


object-identifier: (event-enrollment, 1)
object-name: "Zone1_Alarm"
object-type: EVENT_ENROLLMENT
description: "Zone 1 Alarms"
event-type: OUT_OF_RANGE
notify-type: ALARM
event-parameters: {30, 65.0, 85.0, 0.25}
object-property-reference: {(analog-input, 1), present-value}
event-state: HIGH_LIMIT
event-enable: {TRUE, TRUE, TRUE}
acked-transitions: {FALSE, TRUE, TRUE}
notification-class: 1
}

object-identifier: (group, 1)
object-name: "Group object 1"
object-type: GROUP
description: "Group"
list-of-group-members: (}

BACnet Test Framework Software Manual

GmbH  Telephone:  +49 / 21 51 / 72 94 - 0
Roemerstrasse 15  Fax:  +49 / 21 51 / 72 94 - 50
D-47809 Krefeld  e-mail: info@mbs-software.de
Internet: http://www.mbs-software.de

135
present-value : {
  [(analog-output,1),{(status-flags,(F,F,F,F)),(present-value,21.6))},
  [(binary-value,1),{(description,"Binary Value"),(present-value,inactive)}] }

{ object-identifier: (loop, 1)
  object-name: "LOOP 1"
  object-type: LOOP
  present-value: 2.3
  description: "LOOP"
  status-flags: (False,False,False,False)
  event-state: NORMAL
  reliability: NO_FAULT_DETECTED
  out-of-service: FALSE W
  update-interval: 1
  output-units: Percent W
  manipulated-variable-reference: {{analog-value, 2), present-value} W
  controlled-variable-reference: {{analog-input, 2), present-value} W
  controlled-variable-value: 5.4
  controlled-variable-units: Percent W
  setpoint-reference: {{(analog-value, 3), present-value} W
  setpoint: 2.3
  action: REVERSE
  proportional-constant: 23.4 W
  proportional-constant-units: Percent W
  integral-constant: 27.9 W
  integral-constant-units: Percent W
  derivative-constant: 50.34 W
  derivative-constant-units: Percent W
  bias: 0.4 W
  maximum-output: 100.0 W
  minimum-output: 0.0 W
  priority-for-writing: 16 W
  COV-increment: 1.0
  time-delay: 0
  notification-class: 1
  error-limit: 1.0
  event-enable: {TRUE, TRUE, TRUE}
  acked-transitions: {True, True, True}
  notify-type: ALARM
  event-time-stamps: ([1]0,[1]0,[1]0)
}

{ object-identifier: (multi-state-input, 1)
  object-name: "Multi State Input 1"
  object-type: MULTI-STATE-INPUT
  present-value: 1
  description: "Multi State Input"
  device-type: "Device Type"
  status-flags: (FALSE, FALSE, FALSE, FALSE)
  event-state: NORMAL
  reliability: NO_FAULT_DETECTED
  out-of-service: FALSE W
  number-of-states: 3
  state-text: ["Off","On_Low","On_High"]
  time-delay: 2
  notification-class: 1
  alarm-values: (2)
  fault-values: (3)
  event-enable: {TRUE, TRUE, TRUE} W
  acked-transitions: {FALSE, FALSE, FALSE, FALSE}
  notify-type: ALARM
  event-time-stamps: ([1]0,[1]0,[1]0)
}

{ object-identifier: (multi-state-output,1)
  object-name: "Multi State Output 1"
  object-type: MULTI-STATE-OUTPUT
  present-value: 1 W
  description: "Multi State Output"
device-type: "Device Type"
status-flags: {FALSE,FALSE,FALSE,FALSE}
event-state: NORMAL
reliability: NO-FAULT-DETECTED
out-of-service: FALSE
number-of-states: 3
state-text: {"Off","On_Low","On_High"}
priority-array: {NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL}
relinquish-default: 1
time-delay: 2
notification-class: 1
feedback-value: 1
event-enable: {TRUE, TRUE, TRUE}
acked-transitions: {FALSE, FALSE, FALSE}
notify-type: ALARM
event-time-stamps: ([1][0],[1][0],[1][0])

{} object-identifier: (multi-state-value, 1)
object-name: "Multi State Value 1"
object-type: MULTI-STATE-VALUE
present-value: 1
description: "Multi State Value"
status-flags: {FALSE,FALSE,FALSE,FALSE}
event-state: NORMAL
reliability: NO_FAULT_DETECTED
out-of-service: FALSE
number-of-states: 3
state-text: {"Off","On_Low","On_High"}
priority-array: {NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL, NULL}
relinquish-default: 1
time-delay: 2
notification-class: 1
alarm-values: (2)
fault-values: (3)
event-enable: {TRUE, TRUE, TRUE}
acked-transitions: {FALSE, FALSE, FALSE}
notify-type: ALARM
event-time-stamps: ([1][0],[1][0],[1][0])

{} object-identifier: (notification-class, 1)
object-name: "Notification Class 1"
object-type: NOTIFICATION-CLASS
description: "Notification Class"
notification-class: 1
priority: {1, 2, 3}
ack-required: {FALSE,FALSE,FALSE}
recipient-list: ()

{} object-identifier: (program, 1)
object-name: "PCD Program"
object-type: PROGRAM
program-state: RUNNING
program-change: READY
status-flags: {FALSE,FALSE,FALSE,FALSE}
out-of-service: FALSE

{} object-identifier: (pulse-converter, 1)
object-name: "Pulse Converter 1"
object-type: PULSE-CONVERTER
description: "Pulse Converter"
present-value: 567.89
status-flags: {FALSE,FALSE,FALSE,FALSE}
reliability: NO_FAULT_DETECTED
out-of-service: FALSE
units: KILOWATT_HOURS
scale-factor : 12.5
adjust-value : 0.0 W
count : 0
count-before-change : 1
cov-increment : 1.0
cov-period : 3600
notification-class : 1
time-delay: 0
high-limit: 90
low-limit: 5
deadbond: 2
limit-enable: {TRUE, TRUE} W
event-enable: {TRUE, TRUE, TRUE} W
acked-transitions: {FALSE, FALSE, FALSE}
notify-type: ALARM
event-time-stamps: (10,10,10)

object-identifier: (schedule, 1)
object-name: "Schedule 1"
object-type: SCHEDULE
present-value: 0.5
description: "Schedule"
effective-period: ((5-September-1995),(10-June-2009))
weekly-schedule: {
  (**:*:00,0.5),(*:*:10,10.5),(*:*:20,20.5),(*:*:30,30.5),(*:*:40,40.5)),
  (**:*:00,0.5),(*:*:10,10.5),(*:*:20,20.5),(*:*:30,30.5),(*:*:40,40.5)),
  (**:*:00,0.5),(*:*:10,10.5),(*:*:20,20.5),(*:*:30,30.5),(*:*:40,40.5)),
  (**:*:00,0.5),(*:*:10,10.5),(*:*:20,20.5),(*:*:30,30.5),(*:*:40,40.5)),
  (**:*:00,0.5),(*:*:10,10.5),(*:*:20,20.5),(*:*:30,30.5),(*:*:40,40.5)),
  (**:*:00,0.5),(*:*:10,10.5),(*:*:20,20.5),(*:*:30,30.5),(*:*:40,40.5))
}

object-identifier: (trend-log, 1)
object-name: "Trendlog 1"
object-type: trend-log
description: "Trendlog"
log-enable: TRUE W
start-time: {(*- *-*), *:*:*.*} W
stop-time: {(*- *-*), *:*:*.*} W
log-device-object-property: ((binary-input,0),present-value,,(device,1)) W
log-interval: 0 W
cov-resubscription-interval: 3600
client-cov-increment: 1.0
stop-when-full: FALSE
buffer-size: 1000
log-buffer: ()
record-count: 0 W
total-record-count: 1776
notification-threshold: 10
records-since-notification: 32
last-notify-record: 0
event-state: NORMAL
notification-class: 1
event-enable: {TRUE, TRUE, TRUE} W
acked-transitions: {FALSE, FALSE, FALSE}
notify-type: ALARM
event-time-stamps: (10,10,10)
End of BACnet Protocol Implementation Conformance Statement

-- end of file --
25. Sample EPICS-file for OD

This chapter shows a sample EPICS file for the OD.

>>>>>> The file starts after this line------------------------------------------

PICS 0
BACnet Protocol Implementation Conformance Statement

Vendor Name: "MBS GmbH"
Product Name: "Simulated Server for OD"
Product Model Number: "Any Model Number"
Product Description: "Simulator only"

BIBBs Supported:
{
    DS-RP-A
    DS-RP-B
    DS-RPM-A
    DS-RPM-B
    DS-WP-A
    DS-WP-B
    DS-WPM-B
    DS-COV-A
    DS-COV-B
    DS-COVU-A
    DS-COVU-B
    AE-N-I-B
    AE-ACK-B
    AE-INFO-B
    DM-DD6-A
    DM-DD6-B
    DM-DOB-A
    DM-DOB-B
    DM-DCC-B
    DM-TS-A
    DM-TS-B
    DM-UTC-A
    DM-UTC-B
    DM-RD-B
}

BACnet Standard Application Services Supported:
{
    AcknowledgeAlarm Execute
    ConfirmedCOVNotification Initiate Execute
    ConfirmedEventNotification Initiate Execute
    GetEventInformation Execute
    SubscribeCOV Initiate Execute
    ReadProperty Initiate Execute
    ReadPropertyMultiple Initiate Execute
    WriteProperty Initiate Execute
    WritePropertyMultiple Execute
    DeviceCommunicationControl Execute
    ReinitializeDevice Execute
    I-Am Initiate Execute
    I-Have Initiate Execute
    UnconfirmedCOVNotification Initiate Execute
    UnconfirmedEventNotification Initiate Execute
    TimeSynchronization Initiate Execute
    UTCTimeSynchronization Initiate Execute
    Who-Has Initiate Execute
    Who-Is Initiate Execute
}

Standard Object Types Supported:
{

}
Analog Input
Device
Notification Class

Data Link Layer Option:
{  
  BACnet/IP, 'DIX' Ethernet
}

Character Sets Supported:
{  
  ANSI X3.4
}

Special Functionality:
{  
  Maximum APDU size in octets: 1476  
  Segmented Requests Supported, window size: 16  
  Segmented Responses Supported, window size: 16
}

Default Property Value Restrictions:
{  
  unsigned-integer: <minimum: 0; maximum: 4294967295>  
  signed-integer: <minimum: -2147483647; maximum: 2147483647>  
  real: <minimum: -3.40282347E38; maximum: 3.40282347E38; resolution: 0.01>  
  double: <minimum: 2.2250738585072016E-38; maximum: 1.7976931348623157E38; resolution: 0.0001>  
  date: <minimum: 01-January-1970; maximum: 31-December-2038>  
  octet-string: <maximum length string: 256>  
  character-string: <maximum length string: 64>  
  list: <maximum length list: 10>  
  variable-length-array: <maximum length array: 10>
}

Fail Times:
{  
  Notification Fail Time: 60  
  Internal Processing Fail Time: 2  
  Minimum ON/OFF Time: 2  
  Schedule Evaluation Fail Time: 2  
  External Command Fail Time: 5  
  Program Object State Change Fail Time: 5  
  Acknowledgement Fail Time: 2
}

List of Objects in test device:
{  
  {  
    object-identifier: (device, 4194301)  
    object-name: "Dummyserver for OD"  
    object-type: device  
    system-status: OPERATIONAL  
    vendor-name: "MBS GmbH"  
    vendor-identifier: 50  
    model-name: "Dummyserver Simulation"  
    firmware-revision: "No Firmware"  
    application-software-version: "No Application Software Version"  
    location: "Testlabor"  
    description: "OD for Conformance Tests WSPLab Stuttgart"  
    protocol-version: 1  
    protocol-revision: 4  
    object-list:  
      {  
        (device, 4194301)  
      }  
    max-APDU-length-accepted: 1476  
    segmentation-supported: segmented-both  
    max-segments-accepted: 16
  }
}
local-time: 12:15:22.99
local-date: (Monday,30-April-2007)
utc-offset: -60
daylight-savings-status: FALSE
apdu-segment-timeout: 2000
apdu-timeout: 3000
number-of-APDU-retries: 5
device-address-binding: ()
database-revision: 1
active-cov-subscriptions: ()

End of BACnet Protocol Implementation Conformance Statement

-- end of file --