MG-SOFT Corporation

YANG Explorer 2015
Professional Edition

USER MANUAL

(Document Version: 3.3.2)

Document published on Friday, 16-January-2015

In order to improve the design or performance characteristics, MG-SOFT reserves the right to make changes in this document or in the software without notice.

No part of this document may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the express written permission of MG-SOFT Corporation. Permission to print one copy is hereby granted if your only means of access is electronic.

Depending on your license, certain functions described in this document may not be available in the version of the software that you are currently using.

Screenshots used in this document may slightly differ from those on your display.

MG-SOFT may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. The furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

TABLE OF CONTENTS

TABLE OF FIGURES

1 INTRODUCTION

This document contains instructions for completing basic operations in MG-SOFT YANG Explorer Professional Edition application. Majority of instructions are provided on a step-by-step basis, which should help the reader start using the software effectively.

It is supposed that you are familiar with using a graphical computer environment, such as choosing a main menu command or a pop-up command, selecting items, closing windows and dialog boxes, etc.

All program commands in this manual are written in bold and italic letters. Individual commands in combinations of commands are separated by the "/" character. For example:
Edit / Preferences – which means: click the "Edit" entry in the menu bar and select the "Preferences" command from the "Edit" menu.

All hyperlinks in text are marked with blue colored letters, e.g., Starting YANG Explorer. Clicking a hyperlink opens the page, which the hyperlink points to.

The content of this guide is listed in the Table of Contents.

1.1 Product Description

MG-SOFT YANG Explorer Professional Edition is special application that lets you load NETCONF YANG or YIN modules and explore their structure and properties in the intuitive graphical user interface.

The software can load any standard or vendor-specific YANG or YIN module and display its contents in a visual manner, where module elements are represented as nodes of different types in a hierarchical tree structure. By expanding the YANG tree and selecting a node in the YANG Tree panel on the left side, its properties and description are displayed in the Node Properties tab in the right side of the main window. In addition, the YANG Source File view and the Textual Data Tree view are also available and let you quickly view the section of the YANG code that defines the selected node/subtree, or the textual tree representation (commonly used in IETF NETMOD publications) of the selected data tree branch, respectively.

YANG Explorer implements a wealth of features that let you effectively explore references between YANG statements within one or more YANG/YIN modules. While exploring YANG modules, it is vital to be able to quickly navigate between a reference of a definition and the definition itself, for example, to be able to go from an "if-feature" statement to the definition of a matching "feature" statement. YANG Explorer now offers such navigation whenever a reference exists. The software also provides a reverse functionality that lets you quickly find all usages of a definition in the loaded YANG/YIN modules, for example, to find all "if-features" that reference a specific "feature" definition. This is supported for "typedef", "grouping", "identity", "feature" and "extension" statements. 'Leafrefs' (YANG statements with a "leafref" type) are also an important concept in YANG, and the software now offers a quick way to find all 'leafrefs' and their target schema nodes in a module. Furthermore, you can also select any “leaf” or “leaf-list” node in the YANG tree and search for ‘leafrefs’ that reference it by using the “Find Referencing Leafrefs” functionality.

MG-SOFT YANG Explorer can visualize the selected YANG modules in form of a UML (Unified Modeling Language) class diagram. UML class diagram displays different types of YANG nodes as classes, with their attributes, methods and relationships between the classes (e.g., dependency, composition, inheritance, etc.). You can explore the UML objects and their relationships either within the scope of a single or several associated YANG modules. Furthermore, YANG Explorer lets you focus on the desired elements of YANG modules by specifying excludes (excluding specific, user-selected classes) and by setting various display options (to toggle displaying certain types of classes globally, etc.). You can zoom in and out of the UML class diagram, pan the diagram, reposition classes and connections in the diagram, print the diagram on a printer, and save it as an image file for external use.

Last but not least, MG-SOFT YANG Explorer incorporates the RFC 6110-compliant NETCONF Content Editor and Validator tool. This XML document editor with syntax coloring and auto-complete feature lets you easily compose any type of NETCONF XML document and validate it using the DSDL schemas, which are automatically generated from the selected YANG modules in the background. The tool contains templates for composing typical NETCONF document types, such as RPC requests (get, get-config, edit-config,…), entire configuration datastores, RPC replies, etc. By default, the XML documents you are editing are being automatically validated in real-time. The NETCONF Content Editor window displays easy-to-understand error and warning messages if any inconsistencies are detected in the validated document. This way, one can quickly fix all syntax and semantic inconsistencies before saving them to XML file format.

MG-SOFT YANG Explorer is a Java^{TM} application that can be installed and used on Windows, Linux and Mac OS X operating systems with Java Runtime Environment version 6.0 (a.k.a. JRE 1.6) or later installed.

2 INSTALLING YANG EXPLORER PROFESSIONAL EDITION

This section presents the basic system requirements your computer has to meet to install and use MG-SOFT YANG Explorer Professional Edition, and it describes the installing procedures for MG-SOFT YANG Explorer on Windows, Linux, Mac OS X operating systems.

2.1 Requirements

MG-SOFT YANG Explorer is a Java(TM) application that can be installed and used on Windows, Linux and Mac OS X operating systems with **Java Runtime Environment version 6.0 (a.k.a. JRE 1.6) or later** installed.

Additionally, administrator/root user privileges are required to install the software.

2.1.1 Windows Operating System

The Windows version of MG-SOFT YANG Explorer has been tested on the following 32-bit and 64-bit Microsoft Windows operating systems:

- Windows XP,
- Windows Server 2003,
- Windows Vista,
- Windows Server 2008,
- Windows 7,
- Windows Server 2012,
- Windows 8.x.

2.1.2 Linux Operating System

The Linux version of MG-SOFT YANG Explorer requires on the following 32-bit and 64-bit Linux distributions running on Intel x86/x86_64 architecture:

- Red Hat Enterprise Linux 5 or newer,
- Fedora Core 8 or newer,
- SUSE 11 or newer,
- Ubuntu 9 or newer,
- Slackware 13 or newer.

For the most recent information about the supported distributions, please refer to the release notes (READ_ME.TXT) of the current software release.

2.1.3 Mac OS X Operating System

The Mac OS X version of MG-SOFT YANG Explorer has been successfully tested by MG-SOFT on:

- Mac OS X v10.6.x Snow Leopard (Intel)
- Mac OS X v10.7.x Lion (Intel)
- Mac OS X v10.8.x Mountain Lion (Intel)
- Mac OS X v10.9.x Mavericks (Intel)
- Mac OS X v10.10.x Yosemite (Intel)

2.2 Installing YANG Explorer

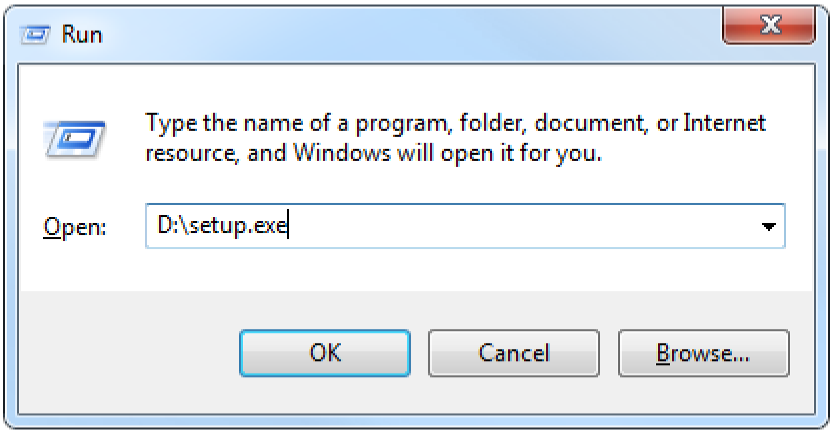
Before installing YANG Explorer Professional Edition on your computer, please make sure your computer meets the system requirements described in the Requirements section.

2.2.1 Windows Operating System

1. Put the MG-SOFT YANG Explorer Professional Edition CD into your CD or DVD drive.

Note: To install the software on Windows, you need to have administrative privileges.

2. Click the ***Start*** button and select the ***Run*** command.
3. The Run dialog box appears (Figure 1).


Run
Type the name of a program, folder, document, or Internet resource, and Windows will open it for you.
Open:
OK
Cancel
Browse...

Open: D:\setup.exe

Figure 1: Run dialog box

4. Into the ***Open*** input line, type `D:\setup` and click the ***OK*** button.

> **Note:** D is the letter assigned to the CD or DVD drive. If your CD or DVD has a different letter, type that one instead of D.

5. Follow the instructions displayed on the screen.

Once the installation is complete, you can start MG-SOFT YANG Explorer program.

2.2.2 Linux Operating System

Before the installation, please close all running MG-SOFT applications and uninstall any previous version of MG-SOFT YANG Explorer Professional Edition

1. Put the MG-SOFT YANG Explorer Professional Edition CD into your CD-ROM drive and mount the CD.
2. The software comes in three different software packages (rpm, deb and tgz). Depending on your Linux distribution, run one of the following commands in a Terminal window to install the software:

a) Linux distributions with the **RPM** package manager (RHEL, Fedora, SUSE, Mandriva, etc):

- On a 32-bit (i386) Linux distribution with the RPM package manager, install the 32-bit RPM package:

```
# rpm -ivh mgYangExplorer_2015-X.X-X.i386.rpm
```

- On a 64-bit (x86_64) Linux distribution with the RPM package manager, install the 64-bit RPM package, as follows:

```
# rpm -ivh mgYangExplorer_2015-X.X-X.x86_64.rpm
```

b) Linux distributions with the **DPKG** package manager (Debian, Ubuntu, etc.):

- On a 32-bit (i386) Linux distribution with the DPKG package manager, install the 32-bit DEB package:

```
# dpkg -i mgYangExplorer-2015_X.X-X_i386.deb
```

- On a 64-bit (x86_64/amd64) Linux distribution with the DPKG package manager, install the 64-bit DEB package, as follows:

```
# dpkg -i mgYangExplorer-2015_X.X-X_x86_64.deb
```

c) Linux distributions with the **installpkg** package manager (e.g., Slackware):

- On a 32-bit (i386) Linux distribution with the installpkg package manager, install the 32-bit TGZ package:

```
# installpkg mgYangExplorer_2015-X.X-i386-X.tgz
```

- On a 64-bit (x86_64) Linux distribution with the installpkg package manager, install the 64-bit TGZ package:

```
# installpkg mgYangExplorer_2015-X.X-x86_64-X.tgz
```

If you have KDE or GNOME Environment installed on your machine, the installation will add an entry to the K Menu or Gnome Menu respectively.

Once the installation is complete, you can start MG-SOFT YANG Explorer program.

2.2.3 Mac OS X Operating System

1. Double-click the MG-SOFT YANG Explorer disk image file (`.dmg`) that you have downloaded from MG-SOFT's Website or obtained on a removable medium.

   Tip: Use **Finder** to navigate to the DMG file if it is not located on your desktop.

2. The contents of the double-clicked disk image displays in a Finder window. MG-SOFT YANG Explorer virtual drive appears on the desktop. Drag&drop the "MG-SOFT YANG Explorer.app" to the "Applications" folder within the MG-SOFT YANG Explorer virtual drive.

Once the installation is complete, you can start MG-SOFT YANG Explorer for Mac OS X from the Finder.

3 STARTING YANG EXPLORER PROFESSIONAL EDITION

3.1 Starting YANG Explorer

3.1.1 Windows Operating System

1. In Windows operating systems, select the ***Start*** / ***Programs*** / ***MG-SOFT YANG Explorer*** / ***YANG Explorer*** command from the Windows taskbar.
2. The YANG Explorer desktop appears and you can start using the software. Please refer to the Applying License Key section for instructions on how to apply your license.

3.1.2 Linux Operating System

The easiest way to start YANG Explorer under Linux operating system is to use the start menu. The start menu can be displayed from the desktop taskbar.

1. If you have the KDE or GNOME desktop environment installed, display the ***K/Gnome*** start menu by clicking the button in the left corner of your taskbar.
2. To start YANG Explorer, search for and use the ***MG-SOFT YANG Explorer*** / ***YANG Explorer*** command.
3. The YANG Explorer desktop appears and you can start using the software. Please refer to the Applying License Key section for instructions on how to apply your license.

> **Tip:** To start the software from the command prompt, open the Terminal window, CD to the directory where the "mgYangExplorer.jar" file is (/usr/local/mg-soft/mgyangexplorer/java/) and run the following command:

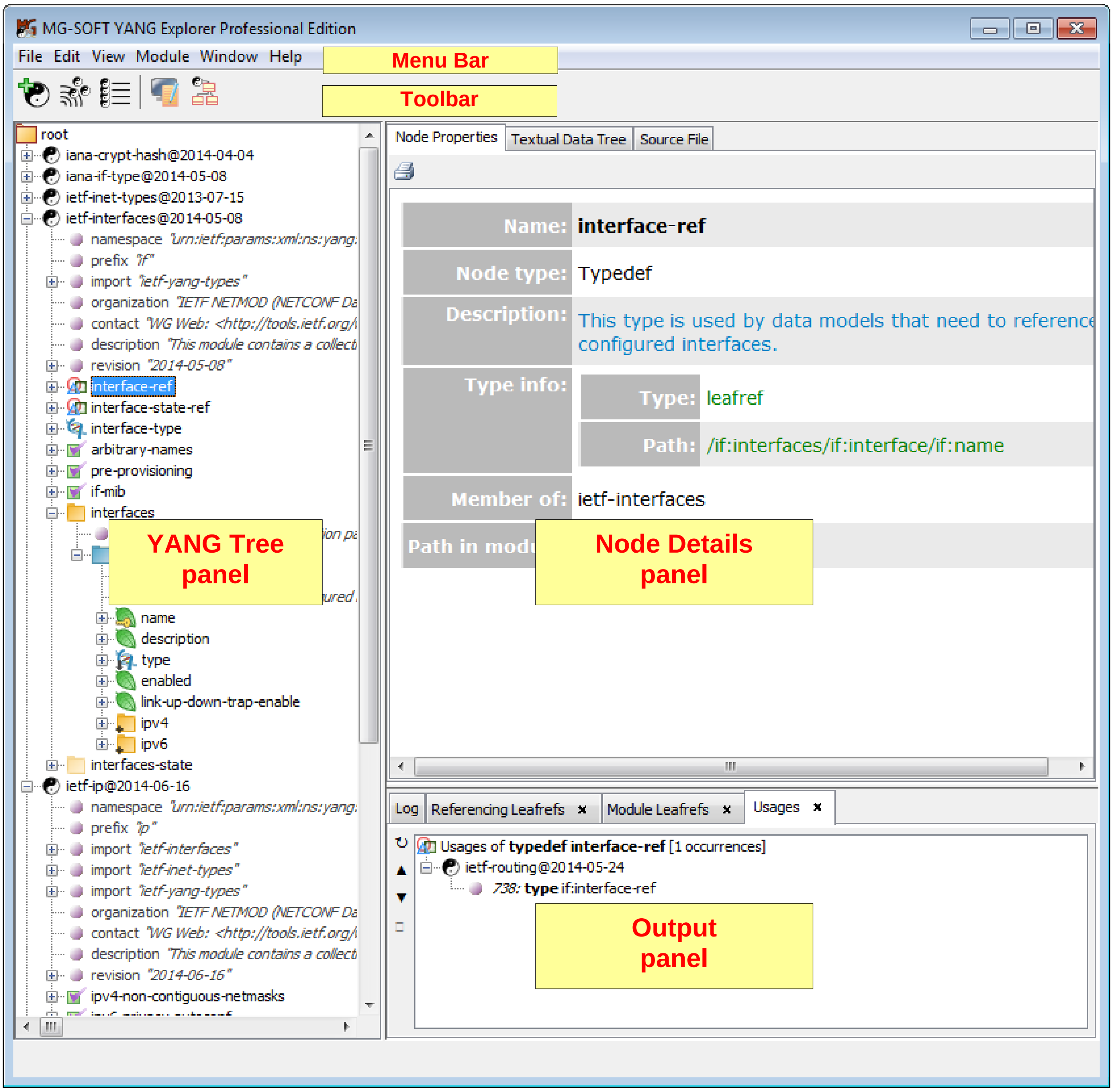
```
# java -jar mgYangExplorer.jar
```

3.1.3 Mac OS X Operating System

1. Open the **Finder** and select the ***Applications*** entry in the panel on the left.
2. Select and double-click the "MG-SOFT YANG Explorer.app" icon to start the YANG Explorer application.
3. The YANG Explorer desktop appears and you can start using the software. Please refer to the Applying License Key section for instructions on how to apply your license.

3.2 YANG Explorer Desktop

The YANG Explorer desktop is composed of typical graphical user interface components, like the title bar, menu bar, toolbar and the working area.



Name:	**interface-ref**
Node type:	Typedef
Description:	This type is used by data models that need to reference configured interfaces.
Type info:	Type: leafref Path: /if:interfaces/if:interface/if:name
Member of:	ietf-interfaces
Path in modu	

Usages of **typedef interface-ref** [1 occurrences]
ietf-routing@2014-05-24
738: **type** if:interface-ref

Figure 2: YANG Explorer desktop

The following window panels form the working area:

- **YANG Tree** (left panel)
  Displays the hierarchical tree structure of the loaded YANG or YIN modules and submodules, where module elements (statements/substaments) are graphically represented as nodes in a hierarchical tree that can be expanded and collapsed.

- **Node Details** (right panel)
  The Node Details panel contains 3 tabs that display different views of the node/subtree selected in the YANG tree panel, as follows:
  - **Node Properties** tab
    Displays the properties of the node selected in the YANG Tree panel in form of a table.
  - **Textual Data Tree** tab
    Displays a simplified text-based data tree diagram (commonly used in IETF NETMOD publications) of the node/subtree selected in the YANG Tree panel.
  - **Source File** tab
    Displays the section of the YANG code in the respective YANG source file that defines the selected node/subtree.

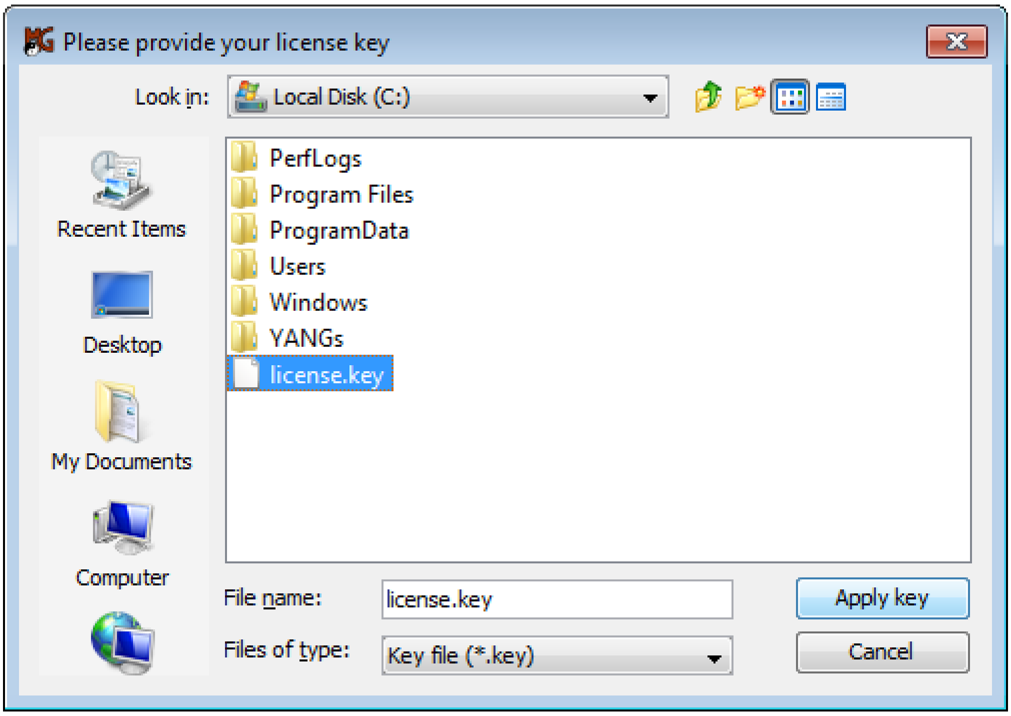
- **Output** (bottom panel)
  The Output panel contains the **Log** tab, which displays application log records that indicate module compilation, loading and validation progress and error messages. In addition to the Log tab, this panel can also display additional tabs that are created when specific functions are used in YANG Explorer, e.g., the **Usages** tab appears after selecting the ***Find Usages*** command, the **Module Leafrefs** tab appears after choosing the ***Find Leafrefs*** command etc. All these commands and the corresponding tabs are described in the respective subsections of the Exploring Cross-References in YANG Modules section.

The above listed window panels are arranged side-by-side in the main window. Window panels can be resized by dragging their borders. Additional windows and dialog boxes can be opened from the program menu, toolbar and the pop-up (context) menu.

4 APPLYING LICENSE KEY

To use MG-SOFT YANG Explorer without limitations, you need to apply a valid license.key file to the software, as follows:

1. Select the ***Help / Apply License Key*** command from the main menu or click the ***Apply License*** toolbar button (the latter is displayed only when the software is run without a valid license key file in place).
2. A dialog box appears that lest you select and apply your license key (Figure 3).



Look in: Local Disk (C:)

File name: license.key

Files of type: Key file (*.key)

Figure 3: Selecting and applying the license.key file

3. Navigate to the drive and folder containing your `license.key` file for MG-SOFT YANG Explorer Professional Edition, select the `license.key` file and click the ***Apply License*** button in the license key selection dialog box.
4. The software will copy the selected `license.key` file to the proper location in order for YANG Explorer to read it and unlock its features after a restart.
5. Exit YANG Explorer by choosing the ***File / Exit*** command and restart it, as described in the Starting YANG Explorer section. Now, the selected license should be applied and you can start using the software without licensing restrictions.

Tip: You can check if the license key has been properly applied by verifying if the About YANG Explorer dialog box (accessible via the ***Help / About*** command) displays your license details correctly.

5 NAVIGATING AND EXPLORING YANG TREE

When YANG Explorer is started for the first time, it automatically loads all standard YANG modules that are included in the distribution and graphically displays loaded modules in the YANG Tree panel in the left portion of the main window. Additional, vendor-specific YANG or YIN modules can be loaded by the user.

> **YANG** is a data modeling language for the Network Configuration Protocol (NETCONF).
>
> **YANG module** defines a hierarchy of data that can be used for NETCONF-based operations, including configuration, state data, Remote Procedure Calls, and notifications. Typically, a YANG module defines a tree of data elements that represent the configuration and runtime status of a particular network element managed via NETCONF. A YANG module is normally stored in a file with the `.yang` extension.
>
> YANG modules can be translated into an equivalent XML syntax called **YIN** (YANG Independent Notation), allowing applications using XML parsers to operate on the models. The conversion from YANG to YIN is lossless. Typically, a YIN module is stored in a file with the `.yin` extension.
>
> **Submodules** are partial modules that contribute definitions to a module. A module may include zero or more submodules, but each submodule may belong to only one module.

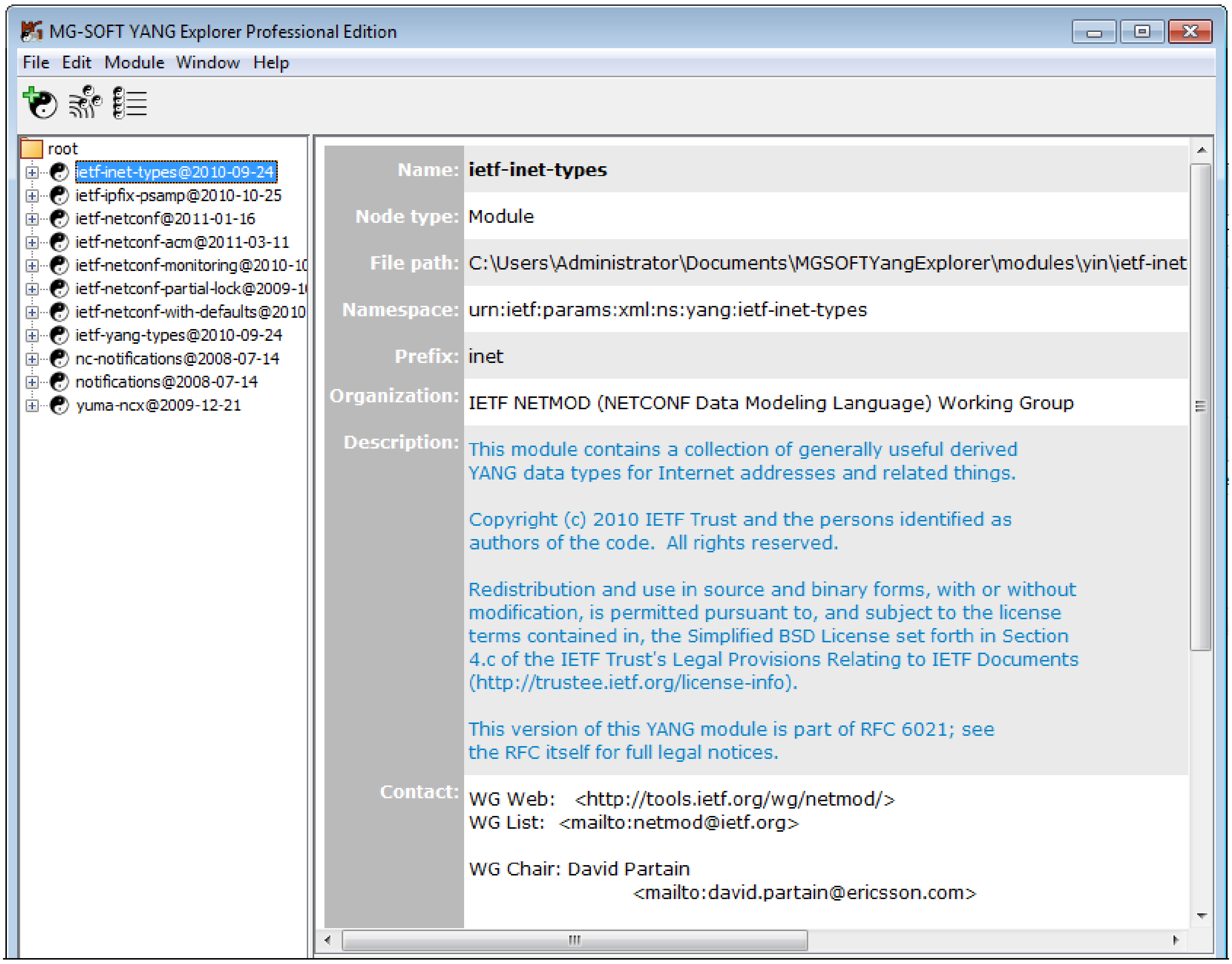
Loaded YIN and YANG modules are hierarchically organized and represented in the tree structure, containing nodes of different types. You can expand and view the tree structure in the YANG Tree panel (Figure 7) in the main window, as well as view the YANG properties of any selected node, as described in this section.

5.1 Viewing Node Properties

YANG Explorer lets you view the properties of any YANG Tree node as it is defined in YANG or YIN definition module. Node properties are displayed in the Node Properties window panel.

5.1.1 To View the Properties of a Module or a Submodule Node

1. In the YANG Tree panel, select a module node (☯), or submodule (☯) node, whose properties you want to view.
2. Te Node Properties window panel on the right side of the main window displays all the properties of the selected module or submodule as defined in the corresponding YANG or YIN definition file, e.g., name, node type, file path, namespace, prefix, imports, description, etc. (Figure 4). Note that some of the properties are displayed only if the selected module or submodule has these properties (e.g., imports, includes, contact, revision, description, etc.).


MG-SOFT YANG Explorer Professional Edition
File Edit Module Window Help
root
ietf-inet-types@2010-09-24
ietf-ipfix-psamp@2010-10-25
ietf-netconf@2011-01-16
ietf-netconf-acm@2011-03-11
ietf-netconf-monitoring@2010-10
ietf-netconf-partial-lock@2009-1
ietf-netconf-with-defaults@2010
ietf-yang-types@2010-09-24
nc-notifications@2008-07-14
notifications@2008-07-14
yuma-ncx@2009-12-21

Name: ietf-inet-types

Node type: Module

File path: C:\Users\Administrator\Documents\MGSOFTYangExplorer\modules\yin\ietf-inet

Namespace: urn:ietf:params:xml:ns:yang:ietf-inet-types

Prefix: inet

Organization: IETF NETMOD (NETCONF Data Modeling Language) Working Group

Description:

```
This module contains a collection of generally useful derived
YANG data types for Internet addresses and related things.

Copyright (c) 2010 IETF Trust and the persons identified as
authors of the code.  All rights reserved.

Redistribution and use in source and binary forms, with or without
modification, is permitted pursuant to, and subject to the license
terms contained in, the Simplified BSD License set forth in Section
4.c of the IETF Trust's Legal Provisions Relating to IETF Documents
(http://trustee.ietf.org/license-info).

This version of this YANG module is part of RFC 6021; see
the RFC itself for full legal notices.
```

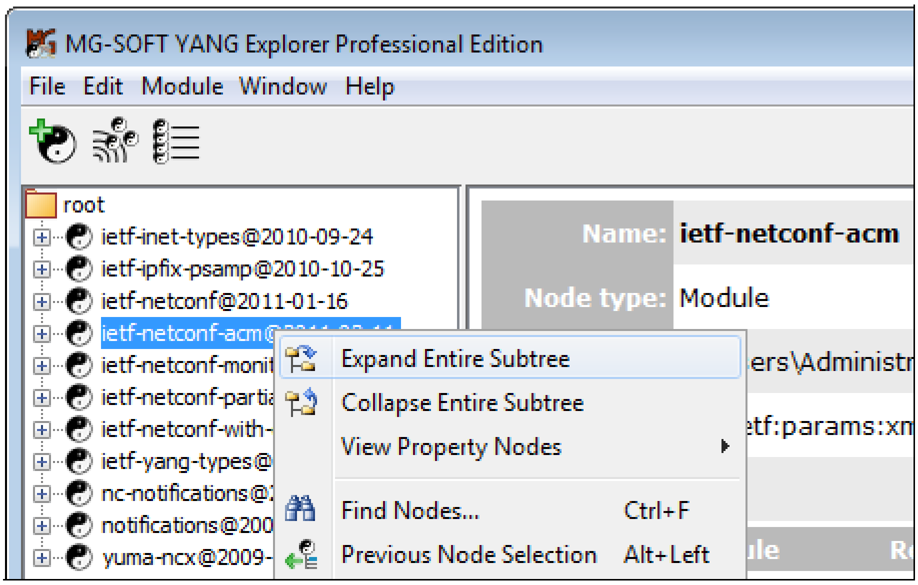
Contact:

```
WG Web:   <http://tools.ietf.org/wg/netmod/>
WG List:  <mailto:netmod@ietf.org>

WG Chair: David Partain
          <mailto:david.partain@ericsson.com>
```

Figure 4: Viewing the properties of the selected YANG module (ietf-inet-types)

5.1.2 To Expand the YANG Tree and Select a Node

1. In the YANG Tree panel, select the root node (🗀 root) if you want to expand the hierarchical tree structure of all the loaded modules, or a module (☯), or submodule (☯) node to expand and display the hierarchical tree structure of that (sub)module only.
2. Right-click the selected node to display the mouse context (pop-up) menu and select the ***Expand Entire Subtree*** pop-up command (Figure 5).



Name: ietf-netconf-acm

Node type: Module

Figure 5: Expanding the subtree of a module in the YANG Tree panel

3. The hierarchy of the expanded subtree is displayed in the YANG Tree panel (Figure 6).

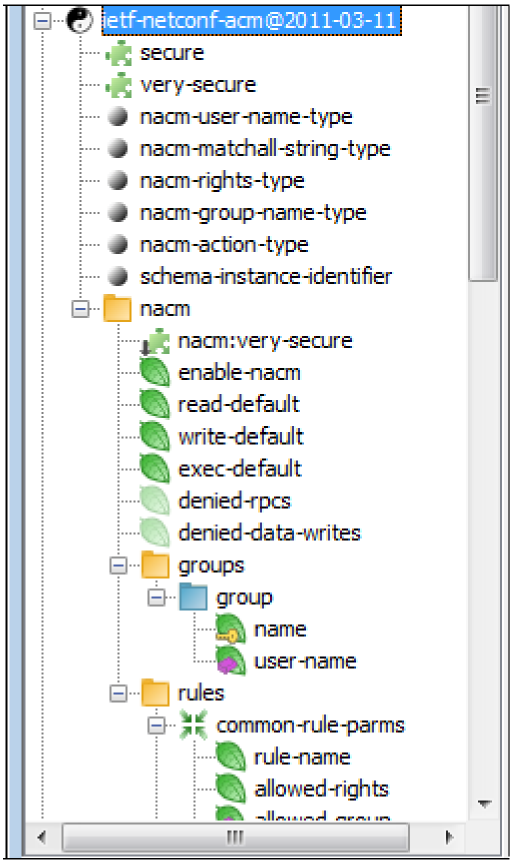
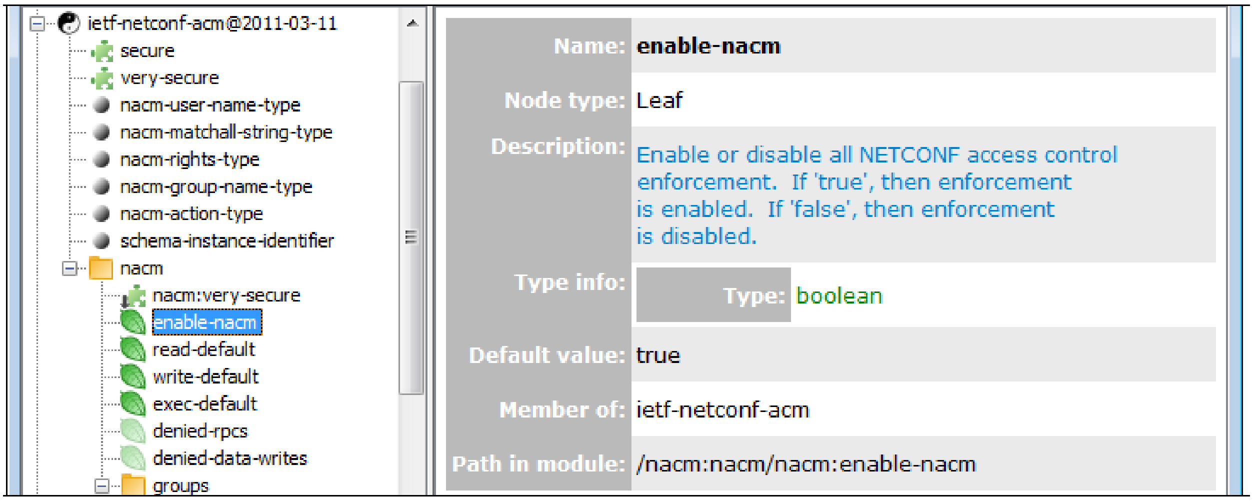


Figure 6: Expanded subtree of a YANG module

4. Select any node in the YANG tree hierarchy to view its properties in the Node Properties panel on the right side of the main window (Figure 7).



Name: enable-nacm

Node type: Leaf

Description: Enable or disable all NETCONF access control enforcement. If 'true', then enforcement is enabled. If 'false', then enforcement is disabled.

Type info: Type: boolean

Default value: true

Member of: ietf-netconf-acm

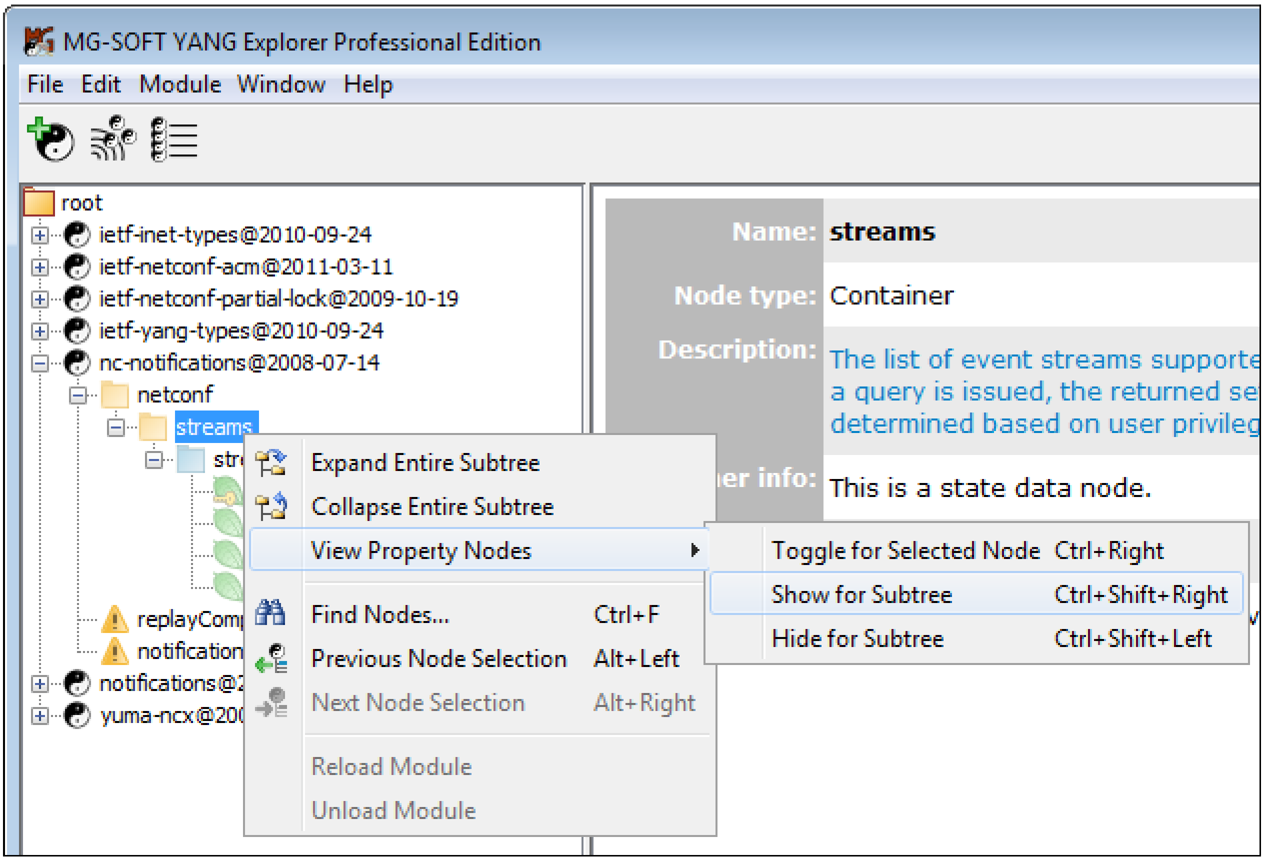
Path in module: /nacm:nacm/nacm:enable-nacm

Figure 7: Viewing the properties of a selected leaf node

5.1.3 To View the Property Sub-Nodes

The properties of any node can be displayed as sub-nodes (●) in the YANG Tree window panel. These property sub-nodes are not displayed by default.

1. Right-click a node and select the ***View Property Nodes / Show for Subtree*** pop-up command (Figure 8) to display the property sub-nodes for all nodes in the selected subtree.


MG-SOFT YANG Explorer Professional Edition
File Edit Module Window Help
root
ietf-inet-types@2010-09-24
ietf-netconf-acm@2011-03-11
ietf-netconf-partial-lock@2009-10-19
ietf-yang-types@2010-09-24
nc-notifications@2008-07-14
netconf
streams
str
replayComp
notification
notifications@2
yuma-ncx@200
Expand Entire Subtree
Collapse Entire Subtree
View Property Nodes
Find Nodes... Ctrl+F
Previous Node Selection Alt+Left
Next Node Selection Alt+Right
Reload Module
Unload Module
Toggle for Selected Node Ctrl+Right
Show for Subtree Ctrl+Shift+Right
Hide for Subtree Ctrl+Shift+Left

Name: streams

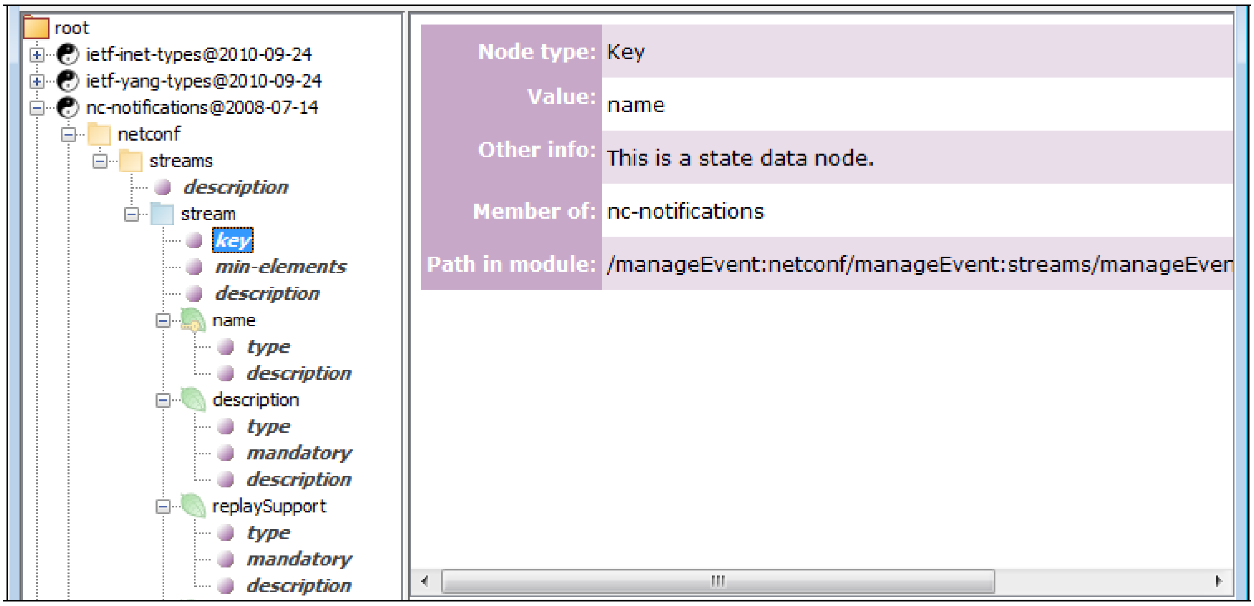
Node type: Container

Description: The list of event streams supporte a query is issued, the returned se determined based on user privileg

er info: This is a state data node.

Figure 8: Selecting the View Property Nodes for Subtree option in the YANG Tree panel

2. The property sub-nodes (●) are displayed for all nodes in the selected subtree (Figure 9).



Node type:	Key
Value:	name
Other info:	This is a state data node.
Member of:	nc-notifications
Path in module:	/manageEvent:netconf/manageEvent:streams/manageEven

Figure 9: Viewing the properties of a property sub-node

3. To view the properties of a sub-node, select it in the YANG Tree panel and view its properties in the Node Properties panel in the right section of the main window. Sub-node properties are displayed in pink-colored table (Figure 9).
4. To hide the properties sub-nodes of the selected node only, right-click the node and choose the ***View Property Nodes / Toggle for Selected Node*** pop-up command.
5. To hide the properties sub-nodes for all nodes in the selected subtree, right-click the subtree node and choose the ***View Property Nodes / Hide for Subtree*** pop-up command.

5.1.4 Different Types of YANG Tree Nodes and Sub-Nodes

Node icons representing different types of YANG statements

MG-SOFT YANG Explorer uses the following **node icons** to present different types of YANG statements in the YANG Tree panel:

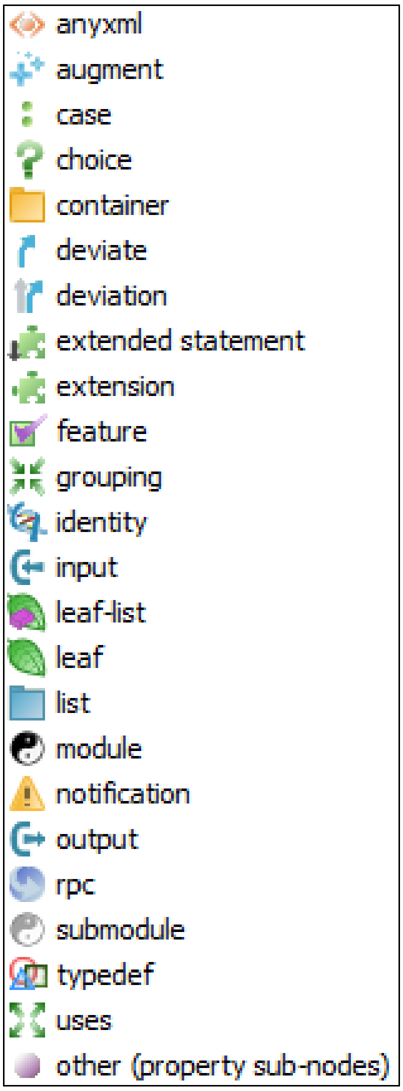


Figure 10: Node icons in the YANG Tree panel representing different types of YANG statements

For a description of YANG statements, please refer to the YANG specification: RFC 6020.

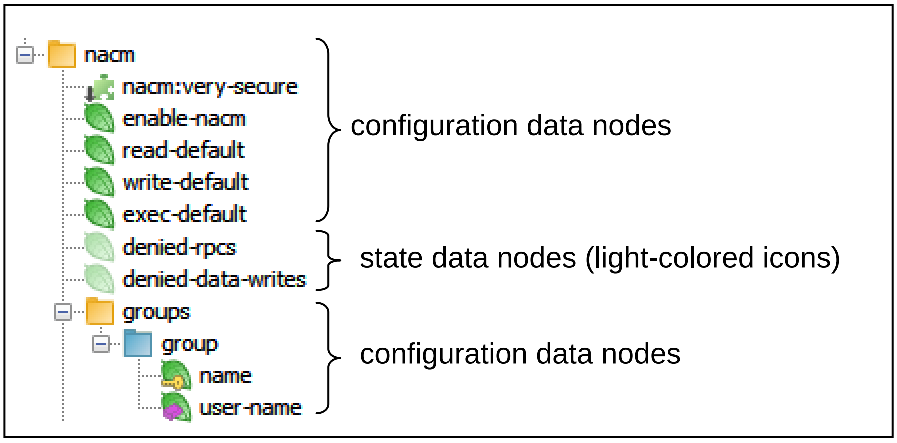
In addition, the following **overlay symbols** are displayed on some of the node icons listed above to depict special 'expanded' nodes in the YANG tree that represent either a usage of a reusable statement (e.g., `uses` of a `grouping`), or nodes that originate from other statements (e.g., `augment`, `extension`,...):

✚	node originating from an `augment` statement (e.g., , , ,...)
⬇	node originating from **use** of a `grouping` or `extension` (e.g., , ,...)
⬇ ✚	node originating from **use** of a `grouping` in an `augment` statement (e.g., ,...)
●	'leafref' node (`leaf` or `leaf-list` node with the *`leafref`* type property) (e.g., ,)
🔑	`leaf` node that is a `key` of a `list` (i.e., ,)

About configuration and state data nodes

Note: In contrast to **configuration data** nodes, which are depicted with normal-colored icons, the **state data** nodes are depicted with light-colored (translucent) icons in the YANG Tree panel (Figure 11). This principle lets you quickly distinguish between configuration (e.g., read-write) and state data (read-only) nodes in the YANG tree.

For example:



configuration data nodes

state data nodes (light-colored icons)

configuration data nodes

Figure 11: Example of configuration and state data nodes displayed in the YANG Tree panel

> RFC 4741, section 1.3: The information that can be retrieved from a running system is separated into two classes, **configuration data** and **state data**. Configuration data is the set of writable data that is required to transform a system from its initial default state into its current state. State data is the additional data on a system that is not configuration data such as read-only status information and collected statistics.

5.2 Searching for Nodes

YANG Explorer lets you search for nodes and sub-nodes, whose argument contains a user-specified text string. For main YANG nodes, this argument is node **name**, while for sub-nodes the argument can be any text (e.g., the argument of a *`description`* property sub-node is the entire description text).

Furthermore, the software lets you search for and find all nodes and sub-nodes of a certain type. To do this, select the desired type (e.g., a `container` or `leaf` or `description` or `config`, or `base`, etc.) from the ***Node type*** drop down list and repeatedly press the ***Find Next*** button in the Find Nodes dialog box to “walk” through nodes of selected type.

Optionally, you can combine both search conditions to find a (sub)node of a certain type whose argument contains a user-specified string.

Search is performed on all the loaded YANG and YIN modules and one can start the search operation from the desired node in the YANG Tree panel.

To find a YANG tree node (start searching from the root node):

1. Right-click the **root** node in the YANG Tree panel and select the ***Find Nodes*** pop-up command (Figure 12). The **Find Nodes** dialog box appears (Figure 13).

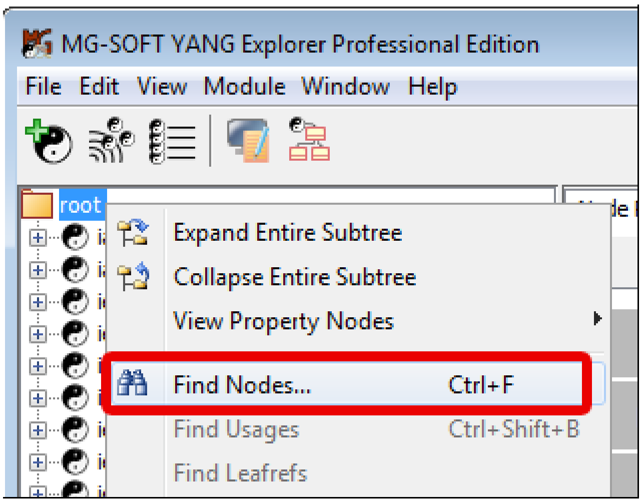


Figure 12: Selecting the Find Nodes command in the YANG Tree panel

2. Into the ***Find what*** input line in the Find Nodes dialog box, enter the name or part of the name of the node you are looking for.

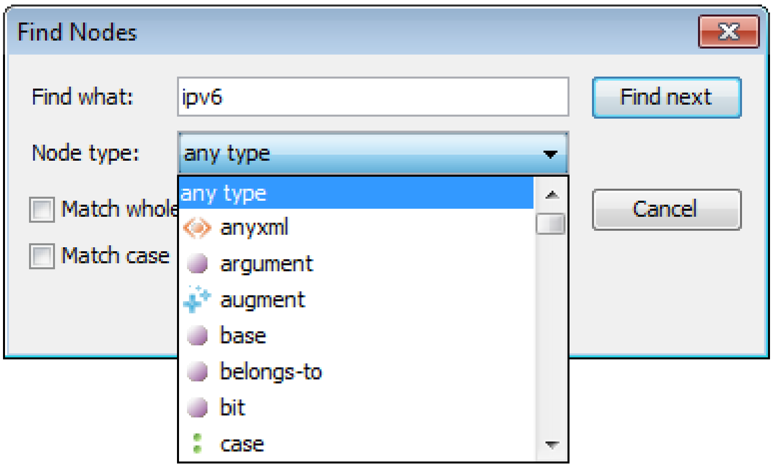
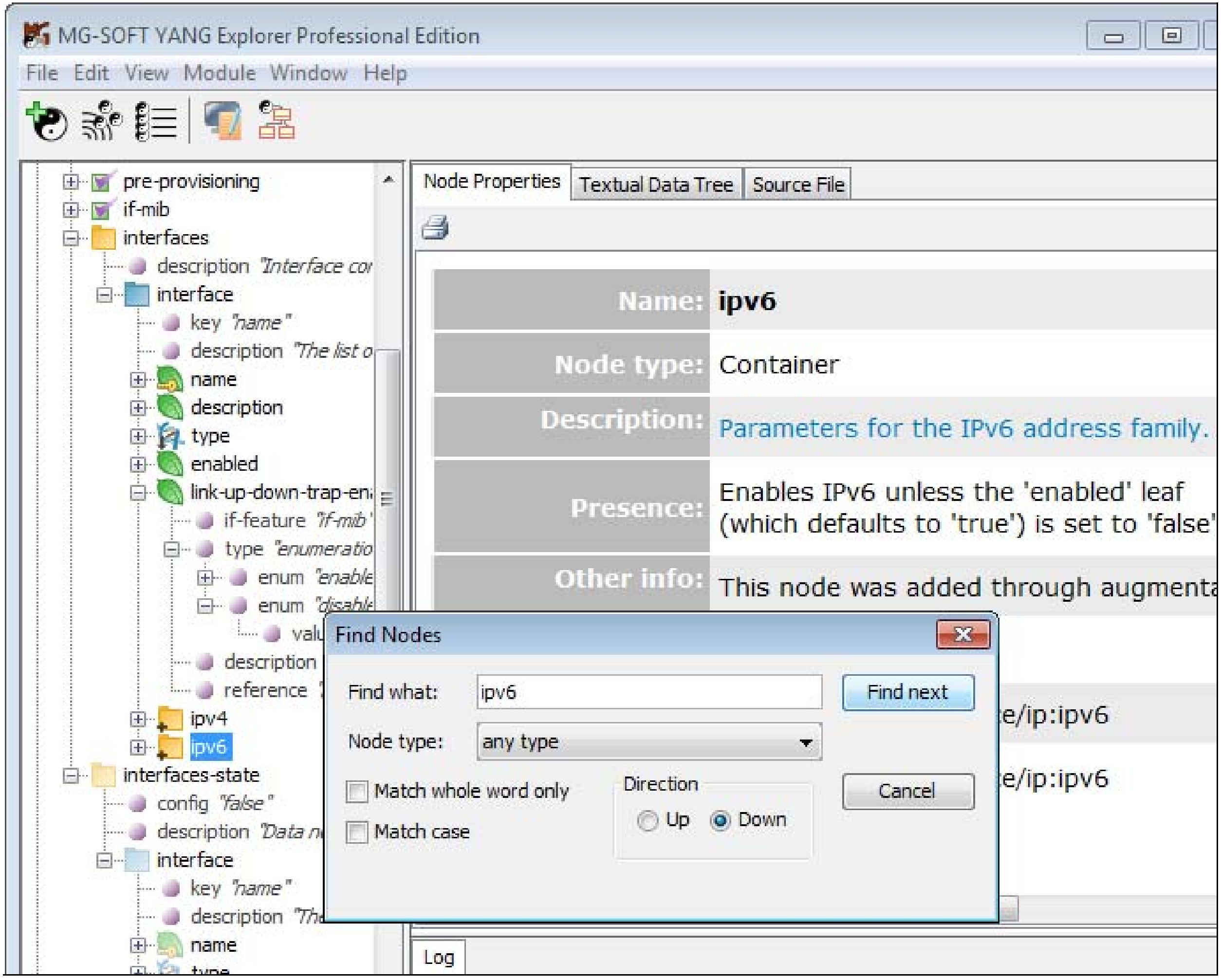


Figure 13: Specifying the search options in the Find Nodes dialog box

3. From the ***Node type*** drop-down list, optionally select the type of the node or sub-node (●) you are looking for. If the `any type` option is selected, YANG Explorer will search all types of nodes and sub-nodes and find the first one from the selected node (root) whose argument contains the string specified in the ***Find what*** input line.
4. Select the ***Down*** radio button in the **Direction** frame (see Figure 14) to enable searching in the direction downward from the selected node.
5. Optionally, select the ***Match case*** (it makes search case sensitive) and ***Match whole word only*** (it finds only occurrences that are whole word, not part of a larger word) checkboxes if they are applicable to your search.
6. Click the ***Find next*** button.

7. YANG Explorer starts searching for the matching node or sub-node in all loaded modules from the root downwards. If a matching node is found, the tree structure from the root to the matching node is expanded and the node is selected in the YANG Tree panel (Figure 14).



Name: ipv6

Node type: Container

Description: Parameters for the IPv6 address family.

Presence: Enables IPv6 unless the 'enabled' leaf (which defaults to 'true') is set to 'false'

Other info: This node was added through augmenta

e/ip:ipv6

e/ip:ipv6

Find what: ipv6

Node type: any type

□ Match whole word only

□ Match case

Direction ○ Up ● Down

Figure 14: The found node is selected in the YANG Tree panel

8. If you would like to continue the search, click the ***Find next*** button again or press the ***F3*** keyboard key to search for the next node whose argument matches the search criteria. The ***F3*** key lets you find the next matching node even after closing the Find Nodes dialog box.

5.3 Exploring Cross-References in YANG Modules

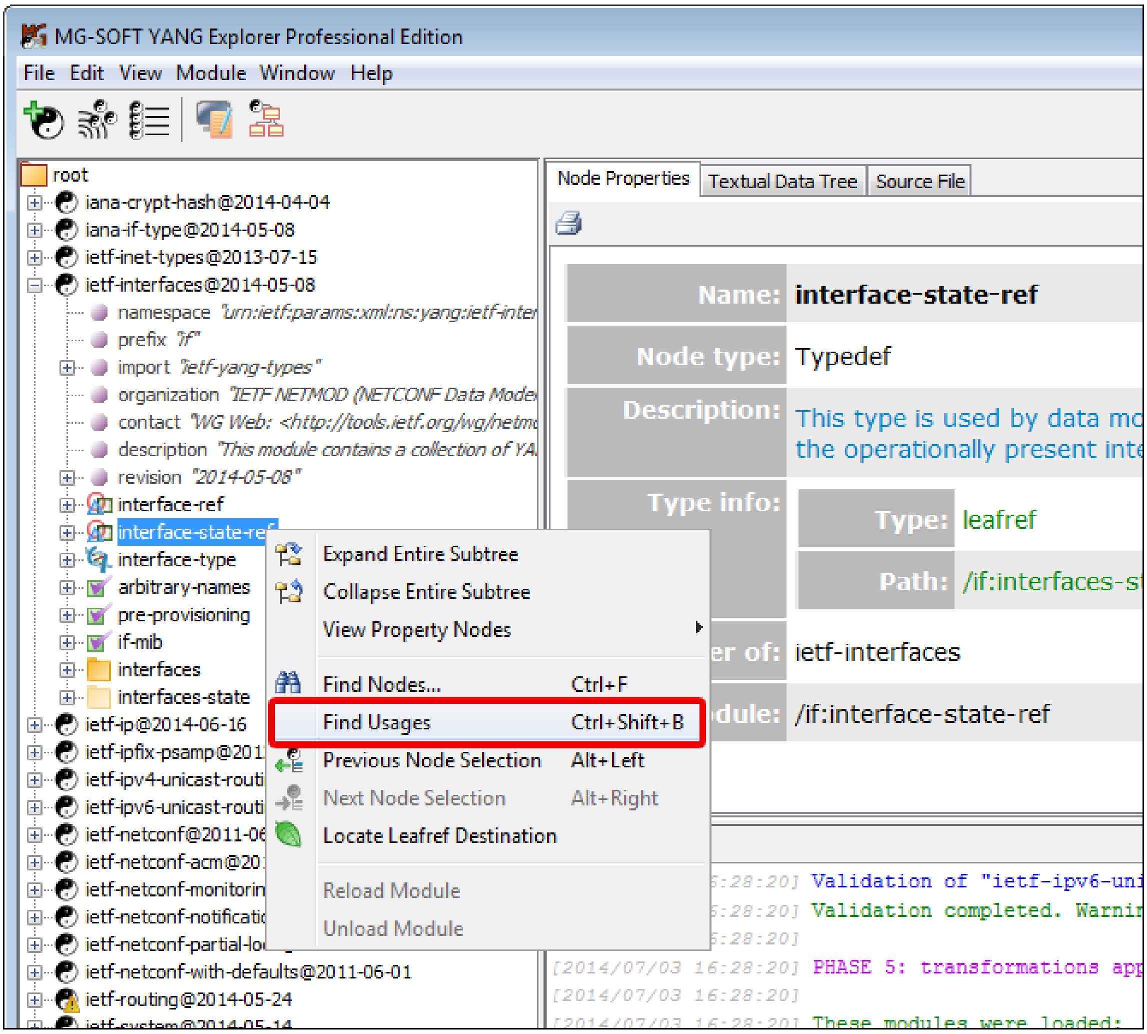
YANG Explorer implements a wealth of features that let you effectively explore references between YANG statements within one or more YANG/YIN modules, as explained in this section.

5.3.1 Using the “Find Usages” Feature

YANG Explorer incorporates the **Find Usages** command that lets you select a node in the YANG Tree that represents a reusable definition and quickly find all occurrences in the currently loaded YANG modules and submodules where this reusable definition is used. Node types that represent reusable definitions are:

- **`grouping`**,
- **`typedef`**,
- **`extension`**,
- **`feature`**,
- **`identity`**.

1. In YANG Tree panel, select a reusable node, e.g., a `typedef` node, right-click it and choose the ***Find Usages*** command from the pop-up menu (Figure 15).



Name: interface-state-ref

Node type: Typedef

Description: This type is used by data mo the operationally present inte

Type info: Type: leafref Path: /if:interfaces-s

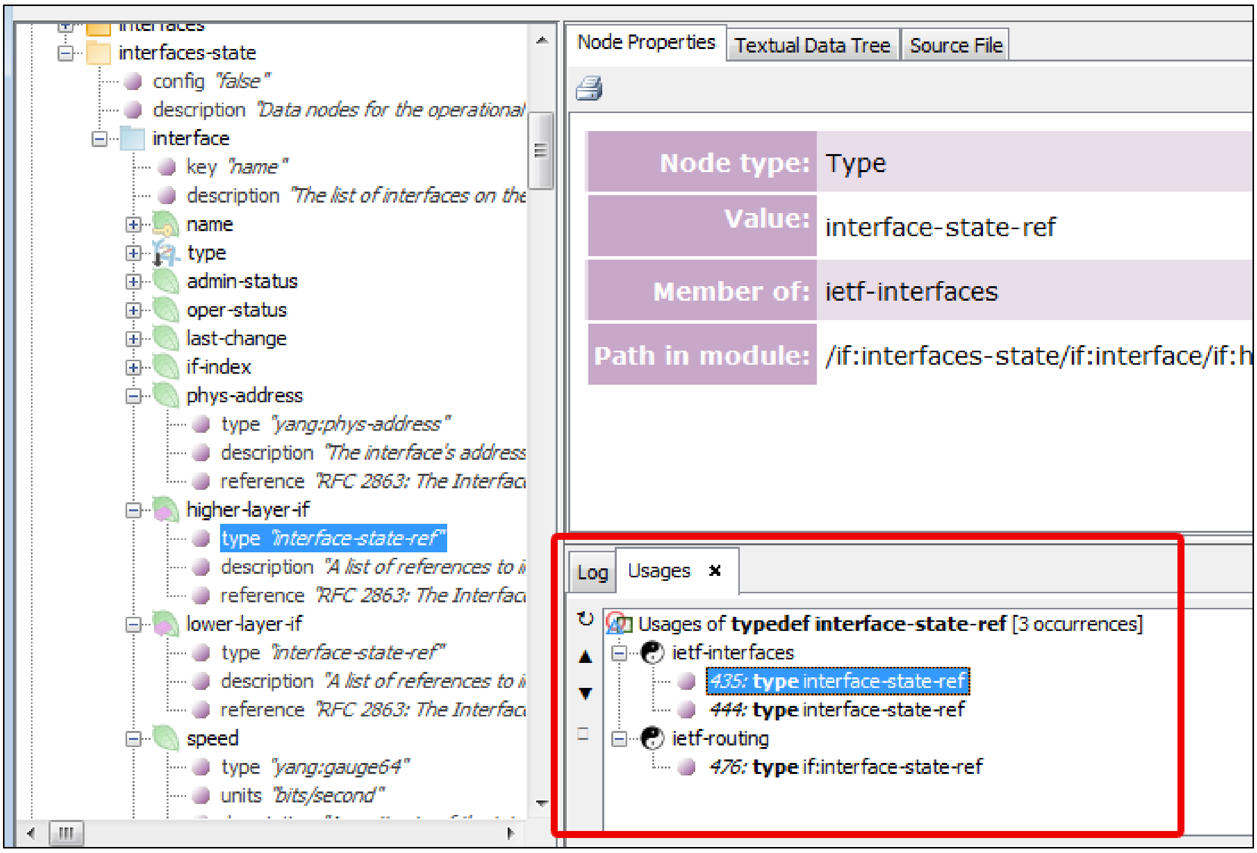
er of: ietf-interfaces

dule: /if:interface-state-ref

```
5:28:20] Validation of "ietf-ipv6-uni
5:28:20] Validation completed. Warnin
5:28:20]
[2014/07/03 16:28:20] PHASE 5: transformations app
[2014/07/03 16:28:20]
[2014/07/03 16:28:20] These modules were loaded:
```

Figure 15: Selecting the *Find Usages* command

2. A new **Usages** tab appears in the Output panel at the bottom of the main window. The Usages tab lists all occurrences in the loaded modules and submodules, where the reusable node is used (Figure 16).


interfaces
interfaces-state
config "false"
description "Data nodes for the operational
interface
key "name"
description "The list of interfaces on the
name
type
admin-status
oper-status
last-change
if-index
phys-address
type "yang:phys-address"
description "The interface's address
reference "RFC 2863: The Interfac
higher-layer-if
type "interface-state-ref"
description "A list of references to i
reference "RFC 2863: The Interfac
lower-layer-if
type "interface-state-ref"
description "A list of references to i
reference "RFC 2863: The Interfac
speed
type "yang:gauge64"
units "bits/second"
Node Properties
Textual Data Tree
Source File
Log
Usages

Node type: Type

Value: interface-state-ref

Member of: ietf-interfaces

Path in module: /if:interfaces-state/if:interface/if:h

Usages of **typedef interface-state-ref** [3 occurrences]
- ietf-interfaces
  - *435:* **type** interface-state-ref
  - *444:* **type** interface-state-ref
- ietf-routing
  - *476:* **type** if:interface-state-ref

Figure 16: Inspecting the usage of the selected reusable element

3. Double-click a usage entry in the **Usages** tab to quickly display the node in the (sub)module in the YANG Tree panel, where the reusable definition is being used (Figure 16).

4. Switch to the **Source File** tab in the Node Details panel to view the line of code in the YANG (sub)module, where the reusable definition is used.

5. To quickly go back to the previous location (node) in the YANG Tree panel, right-click the current node and choose the ***Previous Node Selection*** or press the ***ALT+🡐*** keyboard keys.

5.3.2 Using the "Go to Target Node" Feature

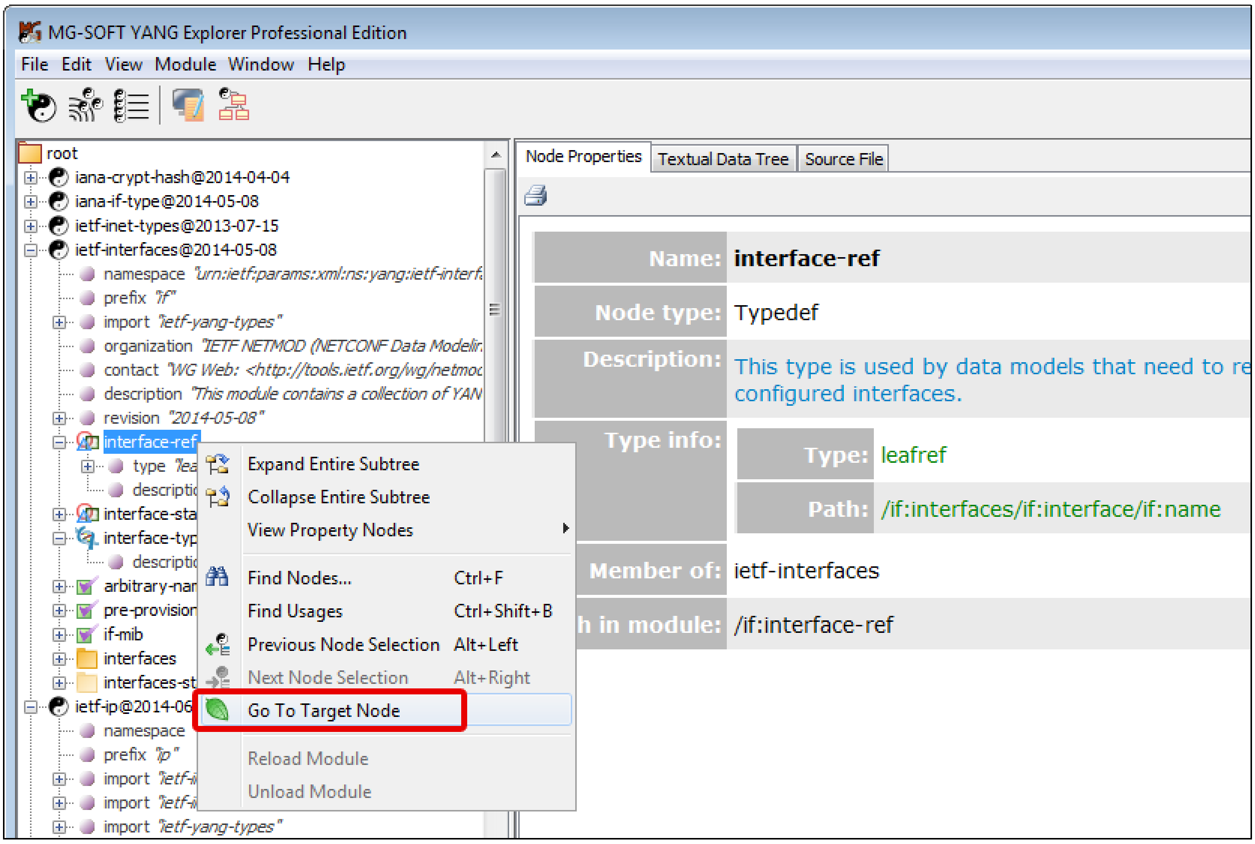
YANG Explorer features the **Go to Target Node** function that lets you select a certain type of node that references or amends other, target node and quickly locate the target node in the currently loaded YANG modules and submodules.

Unlike the **Find Usages** function, which is used for finding and exploring 1:N references in YANG (sub)modules, the **Go to Target Node** function is used for examining the 1:1 source-target references.

The **Go to Target Node** function can be used on the following types of nodes:

- **`augment`**,
- **`leaf`** or **`leaf-list`**, whose type is ***`leafref`***,
- **`typedef`**, whose type is ***`leafref`***,
- **`deviation`**.

1. In YANG Tree panel, select the appropriate node, e.g., a `typedef` node whose type is ***`leafref`***, right-click it and choose the ***Go to Target Node*** command from the pop-up menu (Figure 15).



Name: interface-ref

Node type: Typedef

Description: This type is used by data models that need to re configured interfaces.

Type info: Type: leafref

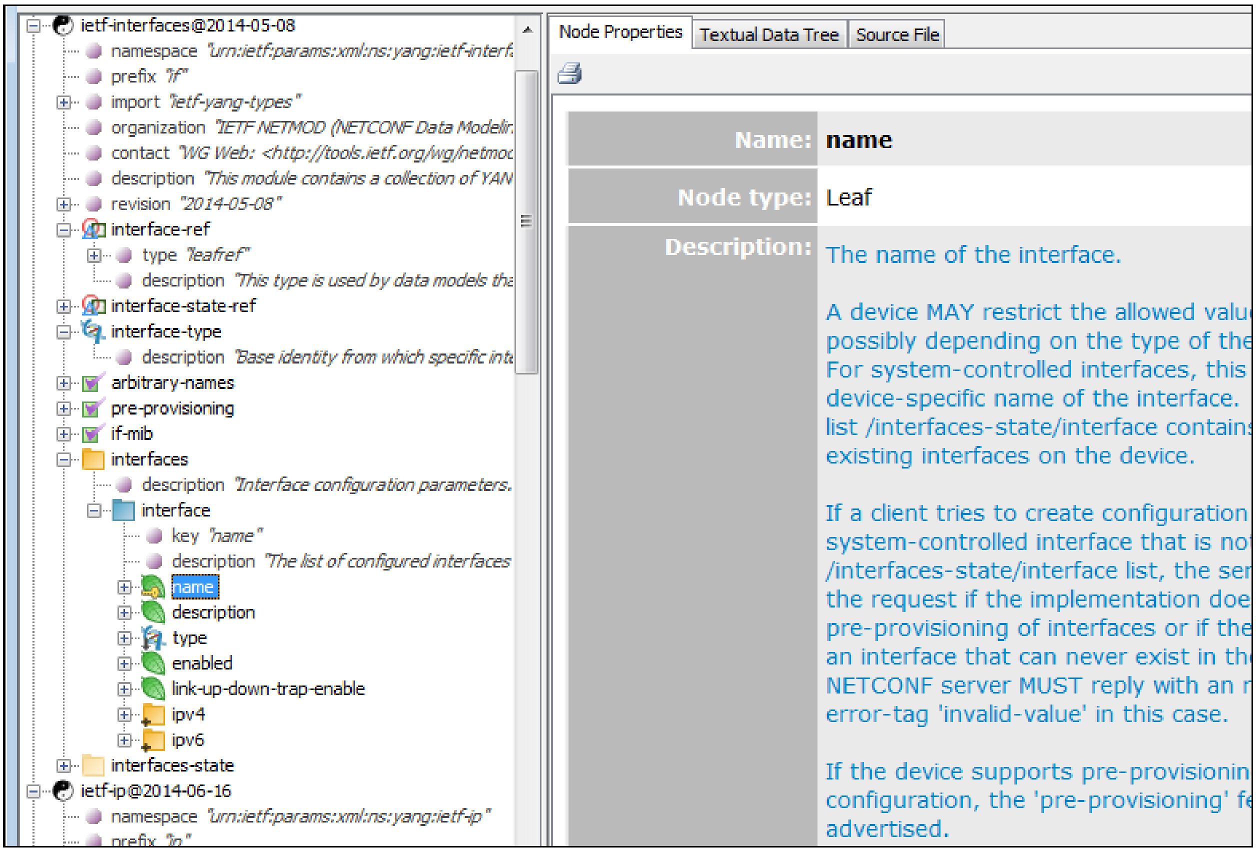
Path: /if:interfaces/if:interface/if:name

Member of: ietf-interfaces

h in module: /if:interface-ref

Figure 17: Selecting the *Go to Target Node* command on a “leafref typedef” node

2. The node that represents the respective target node (e.g., a `leaf` node that the given *`leafref path`* points to) becomes selected in the YANG Tree panel, so you can view its properties, textual data tree diagram or source code, depending on the tab selected in the Node Details panel (Figure 18).



ietf-interfaces@2014-05-08
namespace "urn:ietf:params:xml:ns:yang:ietf-interf
prefix "if"
import "ietf-yang-types"
organization "IETF NETMOD (NETCONF Data Modelin
contact "WG Web: <http://tools.ietf.org/wg/netmoc
description "This module contains a collection of YAN
revision "2014-05-08"
interface-ref
type "leafref"
description "This type is used by data models tha
interface-state-ref
interface-type
description "Base identity from which specific inte
arbitrary-names
pre-provisioning
if-mib
interfaces
description "Interface configuration parameters.
interface
key "name"
description "The list of configured interfaces
name
description
type
enabled
link-up-down-trap-enable
ipv4
ipv6
interfaces-state
ietf-ip@2014-06-16
namespace "urn:ietf:params:xml:ns:yang:ietf-ip"
prefix "ip"

Name: name

Node type: Leaf

Description: The name of the interface.

A device MAY restrict the allowed valu
possibly depending on the type of the
For system-controlled interfaces, this
device-specific name of the interface.
list /interfaces-state/interface contain
existing interfaces on the device.

If a client tries to create configuration
system-controlled interface that is no
/interfaces-state/interface list, the ser
the request if the implementation doe
pre-provisioning of interfaces or if the
an interface that can never exist in th
NETCONF server MUST reply with an r
error-tag 'invalid-value' in this case.

If the device supports pre-provisionin
configuration, the 'pre-provisioning' fe
advertised.

Figure 18: Viewing the resulting node of the *Go to Target Node* command

3. To quickly go back to the previous location (node) in the YANG Tree panel, right-click the current node and choose the ***Previous Node Selection*** or press the ***ALT+←*** keyboard keys.

5.3.3 Using the “Go to Definition” Feature

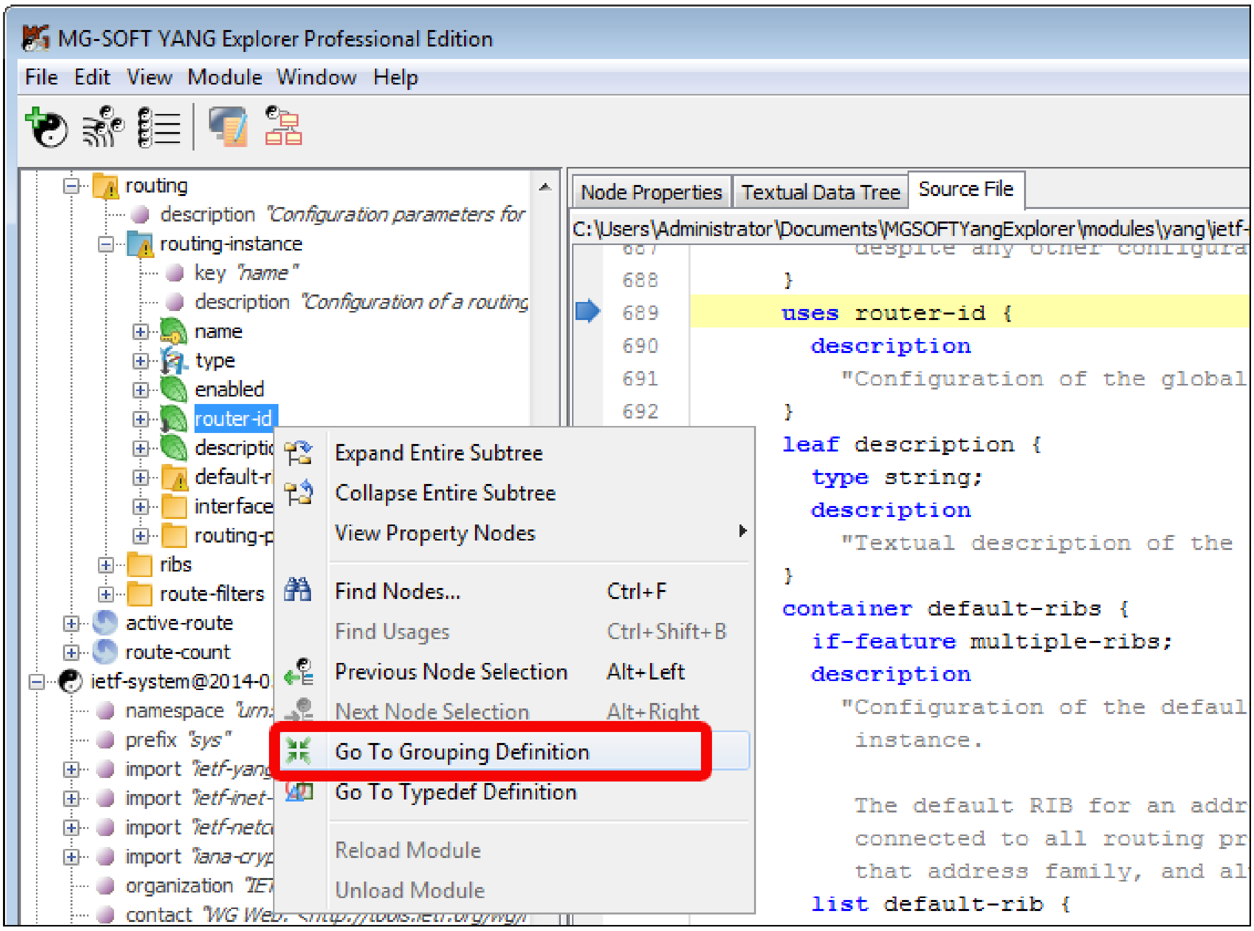
YANG Explorer incorporates the **Go to (X) Definition** function that lets you select a node that represents a reused definition (Y) and quickly find the node in the YANG Tree that represents the definition of the respective reusable code (X). For example, one can select the node representing the expanded *uses ‘xyz’* statement and use the *Go to Grouping Definition* command to find the node representing the *grouping ‘xyz’* statement. The nodes types (Y) on which the **Go to (X) Definition** command can be used are the following:

- expanded **uses** **(Go to Grouping Definition)**
- **leaf** or **leaf-list**, whose type is ***typedef*** **(Go to Typedef Definition)**,
- **typedef**, whose type is ***typedef*** **(Go to Typedef Definition)**,
- **leaf** or **leaf-list** with a ***base*** property **(Go to Identity Definition)**,
- **typedef** with a ***base*** property **(Go to Identity Definition)**,
- **if-feature** **(Go to Feature Definition)**,
- used **extension** **(Go to Extension Definition),**
- expanded **augment** **(Go to Augment Definition).**

The **Go to Definition** can be seen as a reverse function of the **Find Usages** function and the **Go to Target Node** function (for augments).

This section describes the basic principle of using the **Go to Definition** functionality in YANG Explorer.

1. In YANG Tree panel, select a reused node, e.g., a node that represents an 'expanded' **`uses`** statement, right-click it and choose the ***Go to (Grouping) Definition*** command from the pop-up menu (Figure 19).

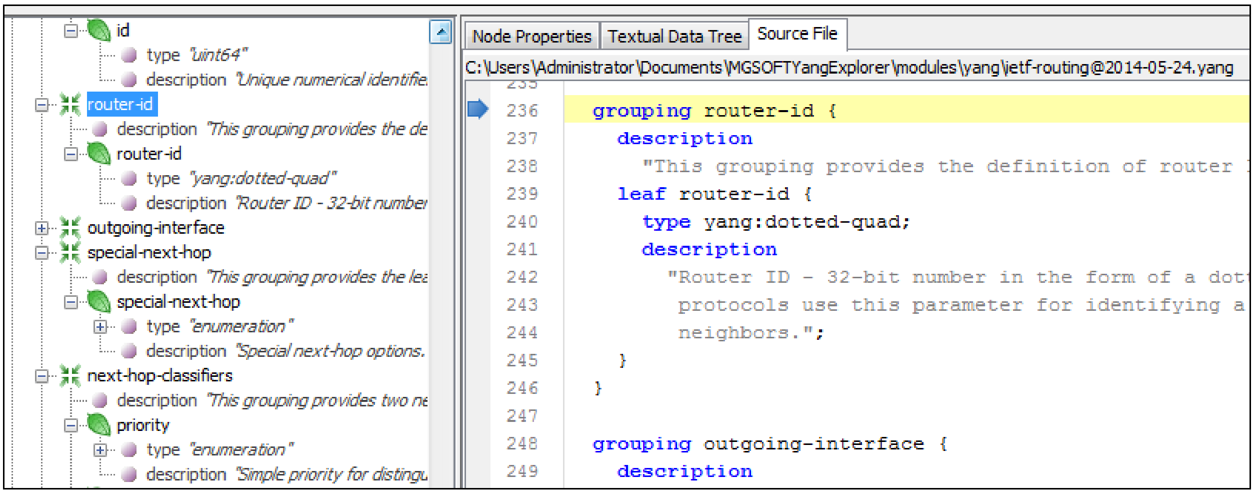


```
   despite any other configura
}
uses router-id {
  description
    "Configuration of the global
}
leaf description {
  type string;
  description
    "Textual description of the
}
container default-ribs {
  if-feature multiple-ribs;
  description
    "Configuration of the defaul
     instance.

     The default RIB for an addr
     connected to all routing pr
     that address family, and al
  list default-rib {
```

Figure 19: Selecting the *Go to Grouping Definition* command on an *expanded uses* node

2. YANG Tree displays the node that represents the respective source node (e.g., a `grouping` node that is used by the given expanded `uses` node) (Figure 20).

3. To quickly go back to the previous location (node) in the YANG Tree panel, right-click the current node and choose the ***Previous Node Selection*** or press the ***ALT+🡐*** keyboard keys.



```
id
    type "uint64"
    description "Unique numerical identifie
router-id
    description "This grouping provides the de
    router-id
        type "yang:dotted-quad"
        description "Router ID - 32-bit number
outgoing-interface
special-next-hop
    description "This grouping provides the lea
    special-next-hop
        type "enumeration"
        description "Special next-hop options.
next-hop-classifiers
    description "This grouping provides two ne
    priority
        type "enumeration"
        description "Simple priority for distingu
```

C:\Users\Administrator\Documents\MGSOFTYangExplorer\modules\yang\ietf-routing@2014-05-24.yang

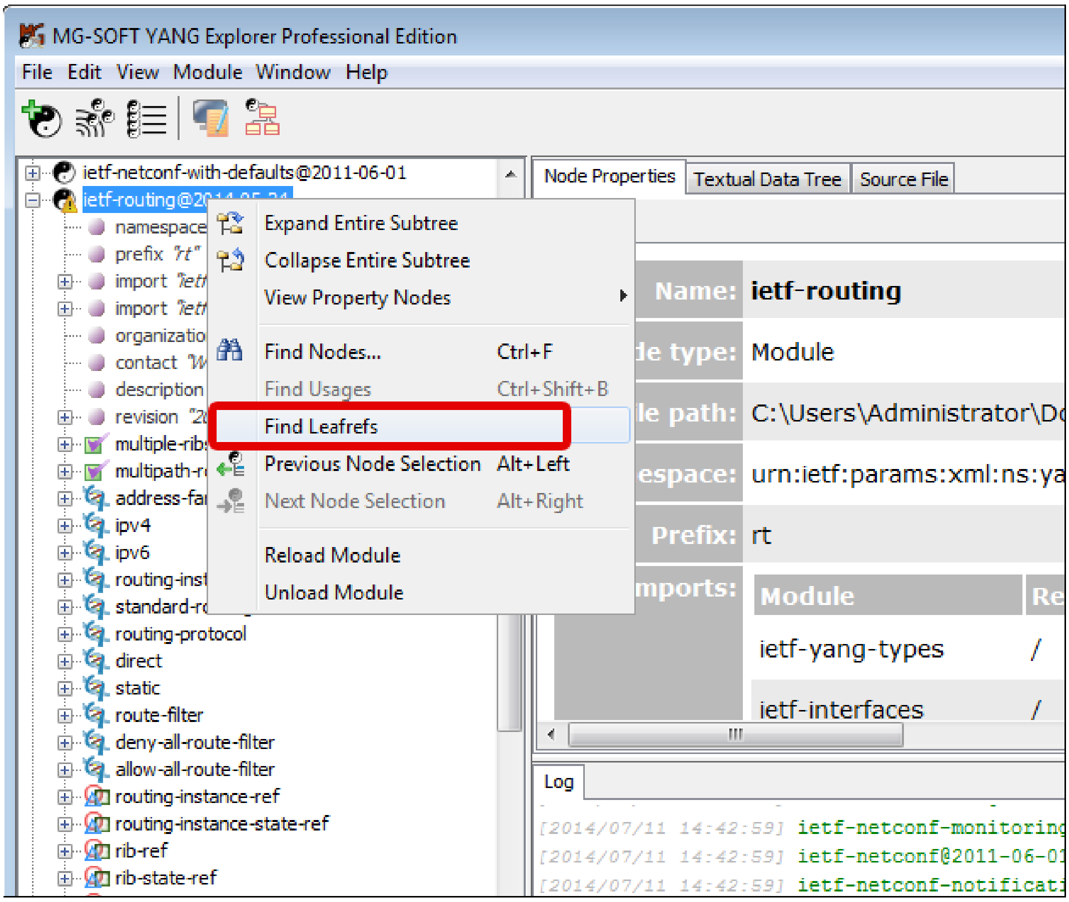
```
  grouping router-id {
    description
      "This grouping provides the definition of router 
    leaf router-id {
      type yang:dotted-quad;
      description
        "Router ID - 32-bit number in the form of a dot
         protocols use this parameter for identifying a
         neighbors.";
    }
  }

  grouping outgoing-interface {
    description
```

Figure 20: The *Go to Grouping Definition* command has found the relevant *grouping* node (and its definition)

5.3.4 Finding all Leafref Nodes in a YANG Module

YANG Explorer lets you easily find all “leafref nodes” in the given YANG module or submodule. A “leafref node” is a `leaf`, `leaf-list` or a `typedef` node with the *`leafref`* type property.

1. In YANG Tree panel, select a module node (☯) or submodule (☯) node, right-click it and choose the ***Find Leafrefs*** command from the pop-up menu (Figure 21).
2. YANG Explorer searches the entire YANG module or submodule for nodes with the *`leafref`* type property and displays results in a new **Module Leafrefs** tab that appears in the Output panel at the bottom of the main window (Figure 22).



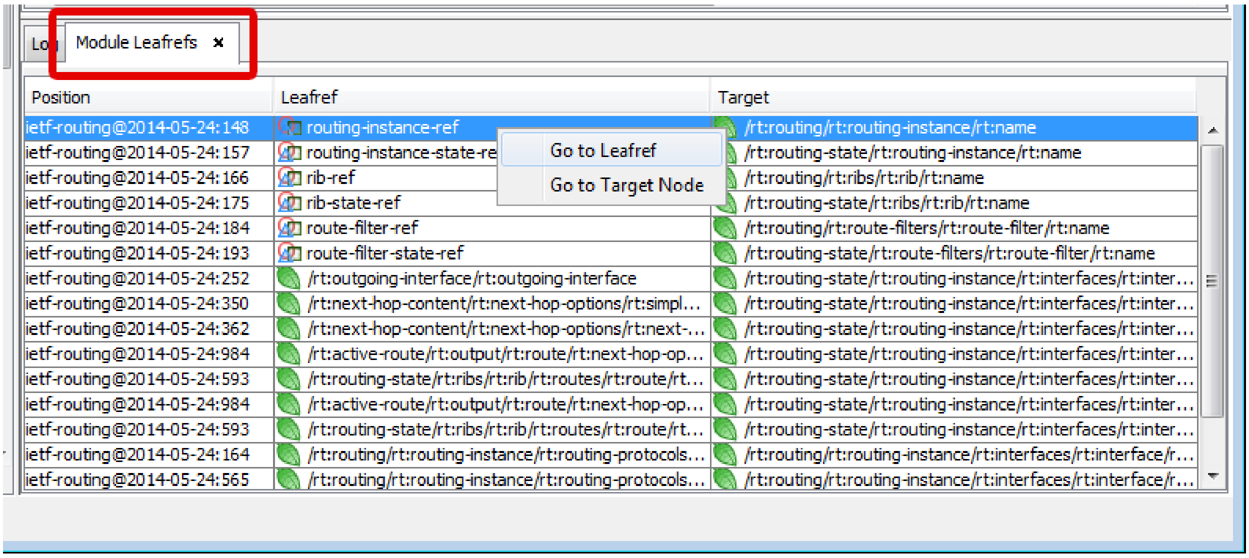
Name: ietf-routing
de type: Module
le path: C:\Users\Administrator\D
espace: urn:ietf:params:xml:ns:ya
Prefix: rt
mports:

Module	Re
ietf-yang-types	/
ietf-interfaces	/

```
[2014/07/11 14:42:59] ietf-netconf-monitoring
[2014/07/11 14:42:59] ietf-netconf@2011-06-01
[2014/07/11 14:42:59] ietf-netconf-notificati
```

Figure 21: The *Find Leafrefs* command lets you find all nodes with the *leafref* type in the given module

3. The **Module Leafrefs** tab displays a list of all nodes with the type of `leafref` in the given YANG module or submodule (Figure 22). The target of each listed "leafref node" is also displayed in the **Module Leafrefs** tab. Target is the `leaf` or `leaf-list` node, which the *`leafref path`* property points to.



Position	Leafref	Target
ietf-routing@2014-05-24:148	routing-instance-ref	/rt:routing/rt:routing-instance/rt:name
ietf-routing@2014-05-24:157	routing-instance-state-re	/rt:routing-state/rt:routing-instance/rt:name
ietf-routing@2014-05-24:166	rib-ref	/rt:routing/rt:ribs/rt:rib/rt:name
ietf-routing@2014-05-24:175	rib-state-ref	/rt:routing-state/rt:ribs/rt:rib/rt:name
ietf-routing@2014-05-24:184	route-filter-ref	/rt:routing/rt:route-filters/rt:route-filter/rt:name
ietf-routing@2014-05-24:193	route-filter-state-ref	/rt:routing-state/rt:route-filters/rt:route-filter/rt:name
ietf-routing@2014-05-24:252	/rt:outgoing-interface/rt:outgoing-interface	/rt:routing-state/rt:routing-instance/rt:interfaces/rt:inter...
ietf-routing@2014-05-24:350	/rt:next-hop-content/rt:next-hop-options/rt:simpl...	/rt:routing-state/rt:routing-instance/rt:interfaces/rt:inter...
ietf-routing@2014-05-24:362	/rt:next-hop-content/rt:next-hop-options/rt:next-...	/rt:routing-state/rt:routing-instance/rt:interfaces/rt:inter...
ietf-routing@2014-05-24:984	/rt:active-route/rt:output/rt:route/rt:next-hop-op...	/rt:routing-state/rt:routing-instance/rt:interfaces/rt:inter...
ietf-routing@2014-05-24:593	/rt:routing-state/rt:ribs/rt:rib/rt:routes/rt:route/rt...	/rt:routing-state/rt:routing-instance/rt:interfaces/rt:inter...
ietf-routing@2014-05-24:984	/rt:active-route/rt:output/rt:route/rt:next-hop-op...	/rt:routing-state/rt:routing-instance/rt:interfaces/rt:inter...
ietf-routing@2014-05-24:593	/rt:routing-state/rt:ribs/rt:rib/rt:routes/rt:route/rt...	/rt:routing-state/rt:routing-instance/rt:interfaces/rt:inter...
ietf-routing@2014-05-24:164	/rt:routing/rt:routing-instance/rt:routing-protocols...	/rt:routing/rt:routing-instance/rt:interfaces/rt:interface/r...
ietf-routing@2014-05-24:565	/rt:routing/rt:routing-instance/rt:routing-protocols...	/rt:routing/rt:routing-instance/rt:interfaces/rt:interface/r...

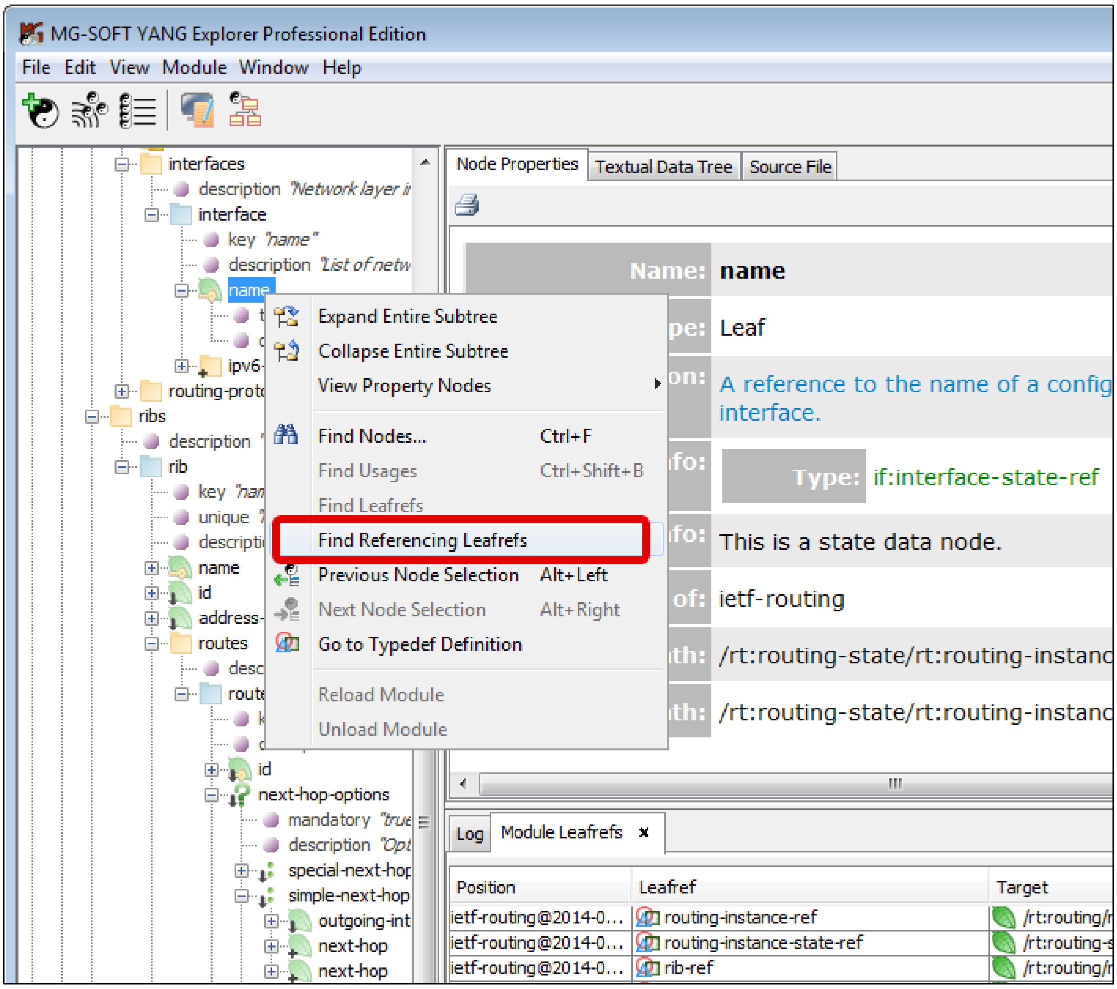
Figure 22: The list of found "*leafrefs*" displayed in the Module Leafrefs tab in the Output window

4. Right click an entry in the **Module Leafrefs** tab and choose the ***Go to Leafref*** pop-up command to quickly find and select the given “leafref node” in the YANG Tree panel.
5. Right click an entry in the **Module Leafrefs** tab and choose the ***Go to Target Node*** pop-up command to quickly find and select the node in the YANG Tree panel that the `leafref path` of the given “leafref node” points to .
6. To quickly go back to the previous location (node) in the YANG Tree panel, right-click the current node and choose the ***Previous Node Selection*** or press the ***ALT+🡐*** keyboard keys.

5.3.5 Finding all Leafref References of a Leaf or Leaf-List Node

YANG Explorer lets you select a **`leaf`** or a **`leaf-list`** node in the YANG tree and quickly find all nodes that reference it via the *`leafref`* statement, as described in this section.

1. In the YANG Tree panel, select a `leaf` or a `leaf-list` node (e.g., a node that you have previously found with the ***Find Leafrefs:Go to Target*** function), right-click it and choose the ***Find Referencing Leafrefs*** command from the pop-up menu (Figure 23).

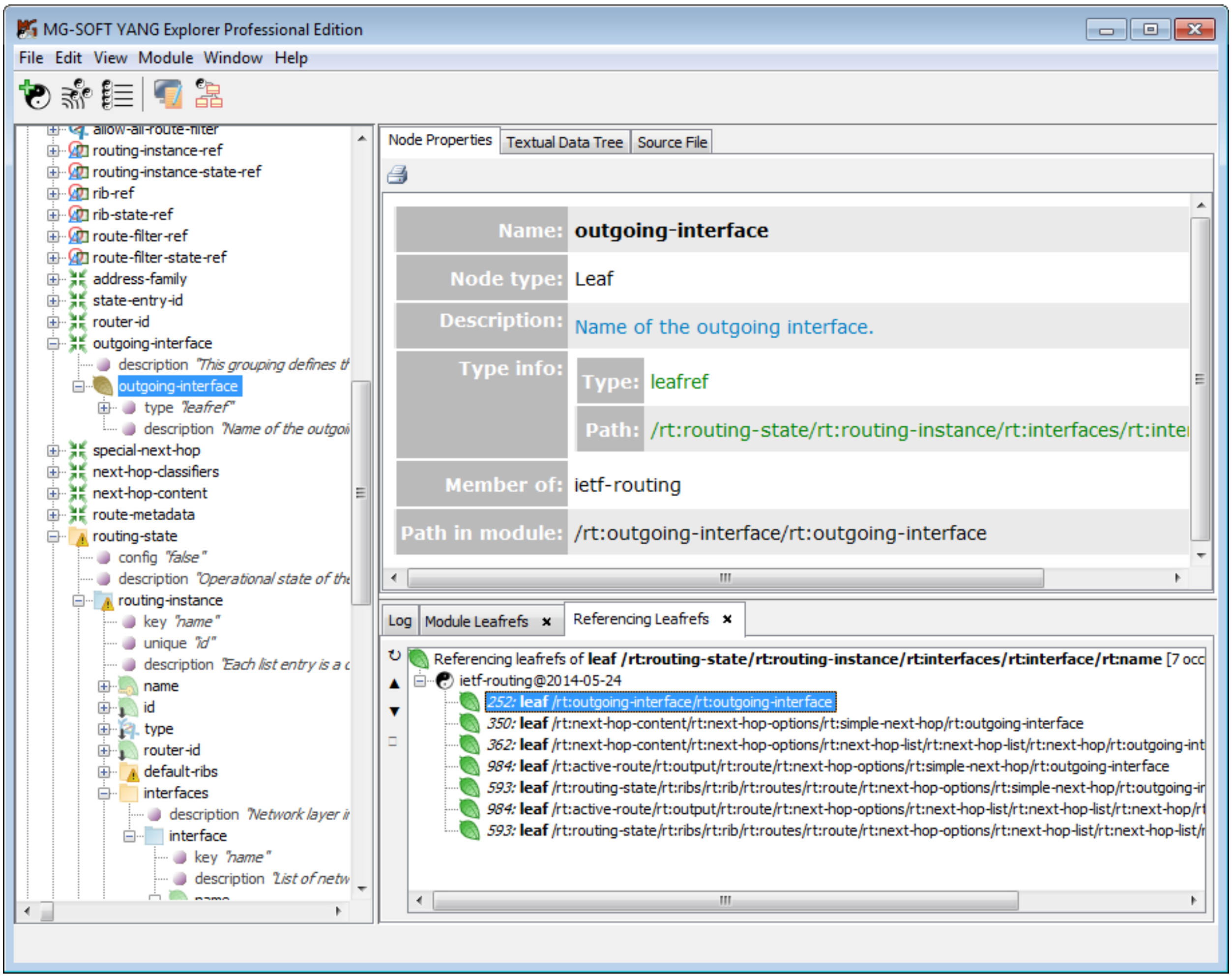


Name: name
pe: Leaf
on: A reference to the name of a config interface.
fo: Type: if:interface-state-ref
fo: This is a state data node.
of: ietf-routing
th: /rt:routing-state/rt:routing-instanc
th: /rt:routing-state/rt:routing-instanc

Position	Leafref	Target
ietf-routing@2014-0...	routing-instance-ref	/rt:routing/r
ietf-routing@2014-0...	routing-instance-state-ref	/rt:routing-s
ietf-routing@2014-0...	rib-ref	/rt:routing/r

Figure 23: Selecting the *Find Referencing Leafrefs* command on a *leaf* node

2. YANG Explorer searches through all currently loaded YANG modules or submodules for nodes with the *`leafref path`* property that points to the `leaf` or `leaf-list` node selected in step 1, and displays the results in a new **Referencing Leafrefs** tab that appears in the Output panel at the bottom of the main window (Figure 24).



```
allow-all-route-filter
routing-instance-ref
routing-instance-state-ref
rib-ref
rib-state-ref
route-filter-ref
route-filter-state-ref
address-family
state-entry-id
router-id
outgoing-interface
    description "This grouping defines t
    outgoing-interface
        type "leafref"
        description "Name of the outgoi
special-next-hop
next-hop-classifiers
next-hop-content
route-metadata
routing-state
    config "false"
    description "Operational state of the
    routing-instance
        key "name"
        unique "id"
        description "Each list entry is a c
        name
        id
        type
        router-id
        default-ribs
        interfaces
            description "Network layer i
            interface
                key "name"
                description "List of netw
```

Name: **outgoing-interface**

Node type: Leaf

Description: Name of the outgoing interface.

Type info: Type: leafref

Path: /rt:routing-state/rt:routing-instance/rt:interfaces/rt:inte

Member of: ietf-routing

Path in module: /rt:outgoing-interface/rt:outgoing-interface

```
Referencing leafrefs of leaf /rt:routing-state/rt:routing-instance/rt:interfaces/rt:interface/rt:name [7 occ
ietf-routing@2014-05-24
    252: leaf /rt:outgoing-interface/rt:outgoing-interface
    350: leaf /rt:next-hop-content/rt:next-hop-options/rt:simple-next-hop/rt:outgoing-interface
    362: leaf /rt:next-hop-content/rt:next-hop-options/rt:next-hop-list/rt:next-hop-list/rt:next-hop/rt:outgoing-int
    984: leaf /rt:active-route/rt:output/rt:route/rt:next-hop-options/rt:simple-next-hop/rt:outgoing-interface
    593: leaf /rt:routing-state/rt:ribs/rt:rib/rt:routes/rt:route/rt:next-hop-options/rt:simple-next-hop/rt:outgoing-ir
    984: leaf /rt:active-route/rt:output/rt:route/rt:next-hop-options/rt:next-hop-list/rt:next-hop-list/rt:next-hop/rt
    593: leaf /rt:routing-state/rt:ribs/rt:rib/rt:routes/rt:route/rt:next-hop-options/rt:next-hop-list/rt:next-hop-list/r
```

Figure 24: Exploring the *referencing leafrefs*

3. Double-click a referencing “leafref” entry in the **Referencing Leafrefs** tab to quickly display the “leafref node” in the YANG Tree panel, which points to the `leaf` or `leaf-list` node selected in step 1.

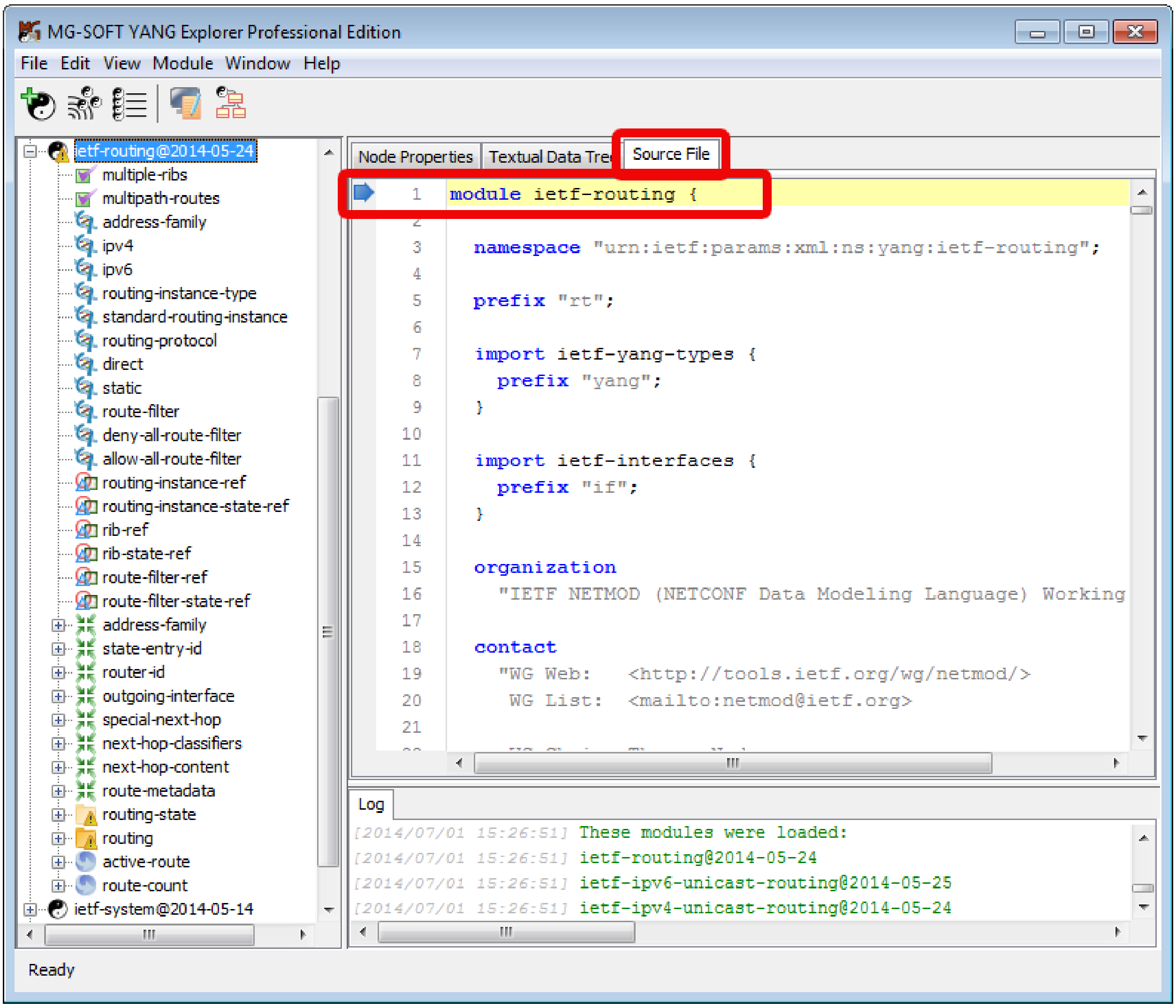
4. Switch to the **Source File** tab in the Node Details panel to view the line of code in the YANG (sub)module that defined the referencing “leafref node”.

5. To quickly go back to the previous location (node) in the YANG Tree panel, right-click the current node and choose the ***Previous Node Selection*** or press the ***ALT+🡐*** keyboard keys.

5.4 Viewing YANG Source File

The **Source File** tab in the Node Details panel displays the YANG source file (i.e., YANG code) that defines the node/subtree currently selected in the YANG Tree.

1. In the YANG Tree panel, select a module (☯) or submodule (☯) node whose source file you would like to view and click the **Source File** tab in the in the Node Details panel.
2. The respective YANG source file that defines the selected module or submodule is displayed in the **Source File** tab, with the blue arrow marker (➡) pointing to the line where the given module/submodule definition starts (Figure 25).

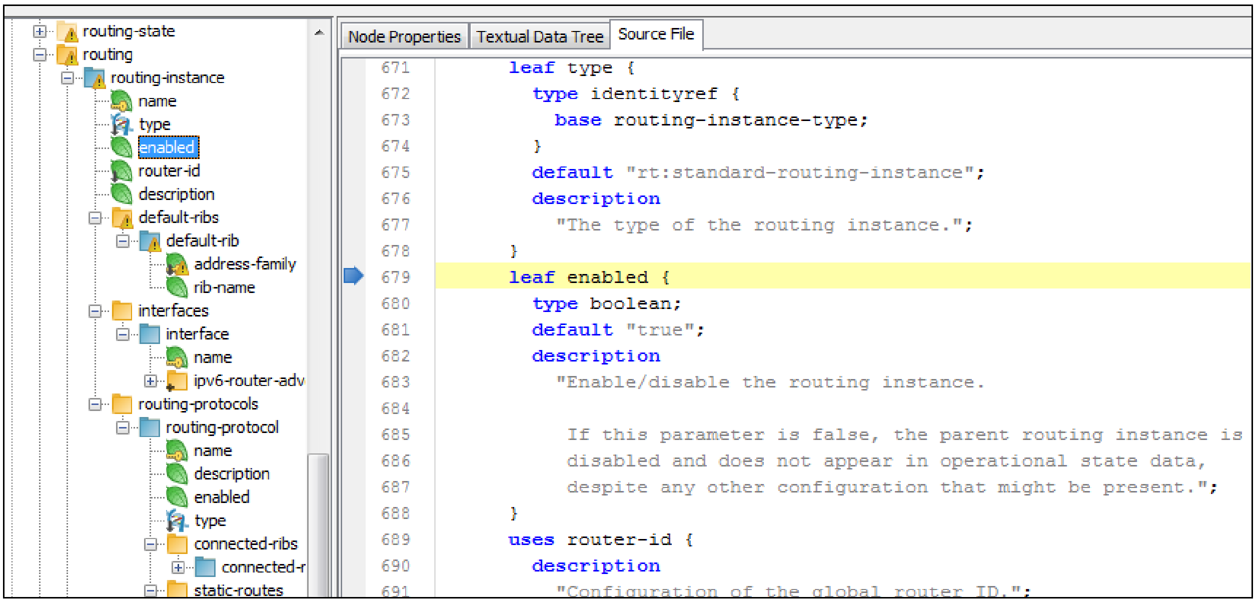


```
module ietf-routing {

  namespace "urn:ietf:params:xml:ns:yang:ietf-routing";

  prefix "rt";

  import ietf-yang-types {
    prefix "yang";
  }

  import ietf-interfaces {
    prefix "if";
  }

  organization
    "IETF NETMOD (NETCONF Data Modeling Language) Working

  contact
    "WG Web:   <http://tools.ietf.org/wg/netmod/>
     WG List:  <mailto:netmod@ietf.org>
```

```
[2014/07/01 15:26:51] These modules were loaded:
[2014/07/01 15:26:51] ietf-routing@2014-05-24
[2014/07/01 15:26:51] ietf-ipv6-unicast-routing@2014-05-25
[2014/07/01 15:26:51] ietf-ipv4-unicast-routing@2014-05-24
```

Figure 25: Viewing the definition of a selected module node in the Source File tab

3. Click any node or sub-node in the YANG Tree panel to view the corresponding section of the YANG source file that defines it (Figure 26).
4. Use the scrollbar to view other sections of the given YANG source file or click other (sub)nodes in the YANG Tree panel.



```
    leaf type {
      type identityref {
        base routing-instance-type;
      }
      default "rt:standard-routing-instance";
      description
        "The type of the routing instance.";
    }
    leaf enabled {
      type boolean;
      default "true";
      description
        "Enable/disable the routing instance.

         If this parameter is false, the parent routing instance is
         disabled and does not appear in operational state data,
         despite any other configuration that might be present.";
    }
    uses router-id {
      description
        "Configuration of the global router ID.";
```

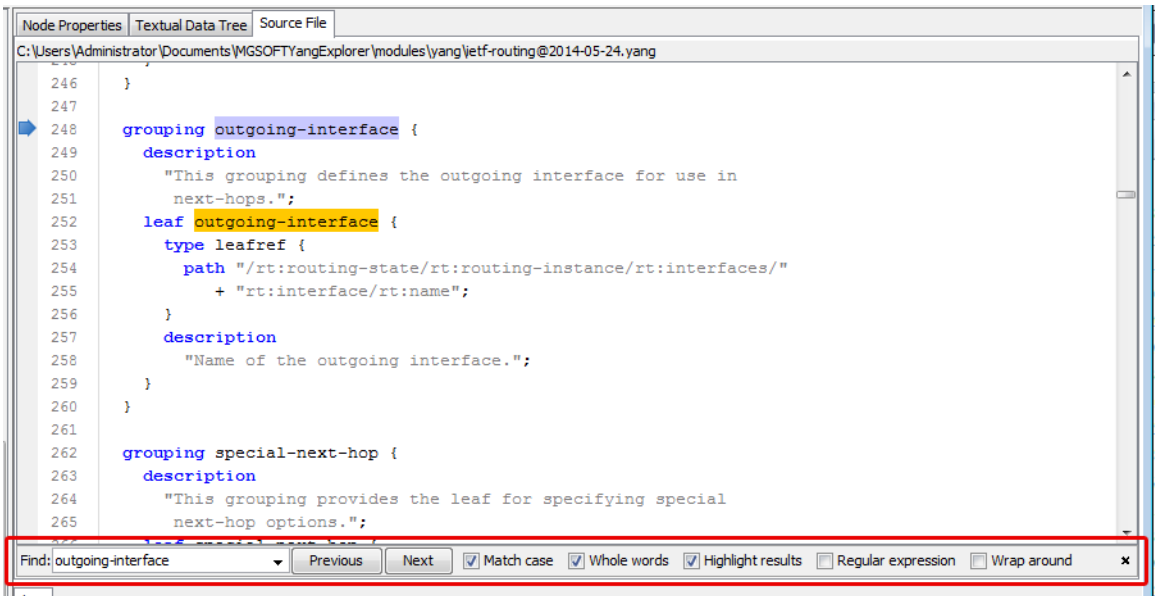
Figure 26: Viewing the definition of a selected leaf node

YANG Explorer automatically applies YANG syntax coloring when presenting a YANG file in the Source File tab (YANG keywords are displayed in blue, arguments in black, descriptions in gray and comments in green letters).

5.4.1 Finding Text in Source Files

This section describes how to use the convenient Find toolbar to search for a user-specified text string in the YANG file displayed in the **Source File** tab in the Node Details panel.

1. Right-click inside the in the **Source File** tab in the Node Details panel and choose the ***Find*** pop-up command or press the ***CTRL+F*** keyboard keys to display the **Find** toolbar at the bottom of the Source File tab (Figure 27).



C:\Users\Administrator\Documents\MGSOFTYangExplorer\modules\yang\ietf-routing@2014-05-24.yang

```
  }

  grouping outgoing-interface {
    description
      "This grouping defines the outgoing interface for use in
       next-hops.";
    leaf outgoing-interface {
      type leafref {
        path "/rt:routing-state/rt:routing-instance/rt:interfaces/"
           + "rt:interface/rt:name";
      }
      description
        "Name of the outgoing interface.";
    }
  }

  grouping special-next-hop {
    description
      "This grouping provides the leaf for specifying special
       next-hop options.";
    leaf special-next-hop {
```

Find: outgoing-interface Previous Next ☑ Match case ☑ Whole words ☑ Highlight results ☐ Regular expression ☐ Wrap around ×

Figure 27: The Find toolbar at the bottom of the Source File view

2. Into the ***Find*** input line enter one or more characters or words you are searching for. In addition, you can select the following search options in the Find toolbar:

   - Check the **Match case** checkbox to make the search case sensitive, meaning that search operation distinguishes between uppercase and lowercase letters. If this option is enabled, the search will find only those strings in which the capitalization matches the one used in the search query (e.g., `Current` will find `Current`, but not `current`).

   - Check the **Whole words** checkbox to find only those strings that are whole words and not part of a larger word (e.g., `link` will find `link Down`, but not `linkDown`).

   - Check the **Highlight results** checkbox to highlight found text in orange color. This option is enabled by default.

   - Check the **Regular expressions** checkbox to enable searching by using the advanced search capabilities offered by the Java regular expressions (for more information about using the regular expressions, please visit the following site: http://www.regular-expressions.info/java.html).

   - Check the **Wrap around** checkbox to enable wrap-around search, i.e., the program will search for the text you specify beginning at the current cursor position, and will continue past the end, to the beginning of the document back to the current cursor position. If this option is disabled, the search begins at the current cursor position and ends at the end of the document. This option is enabled by default.

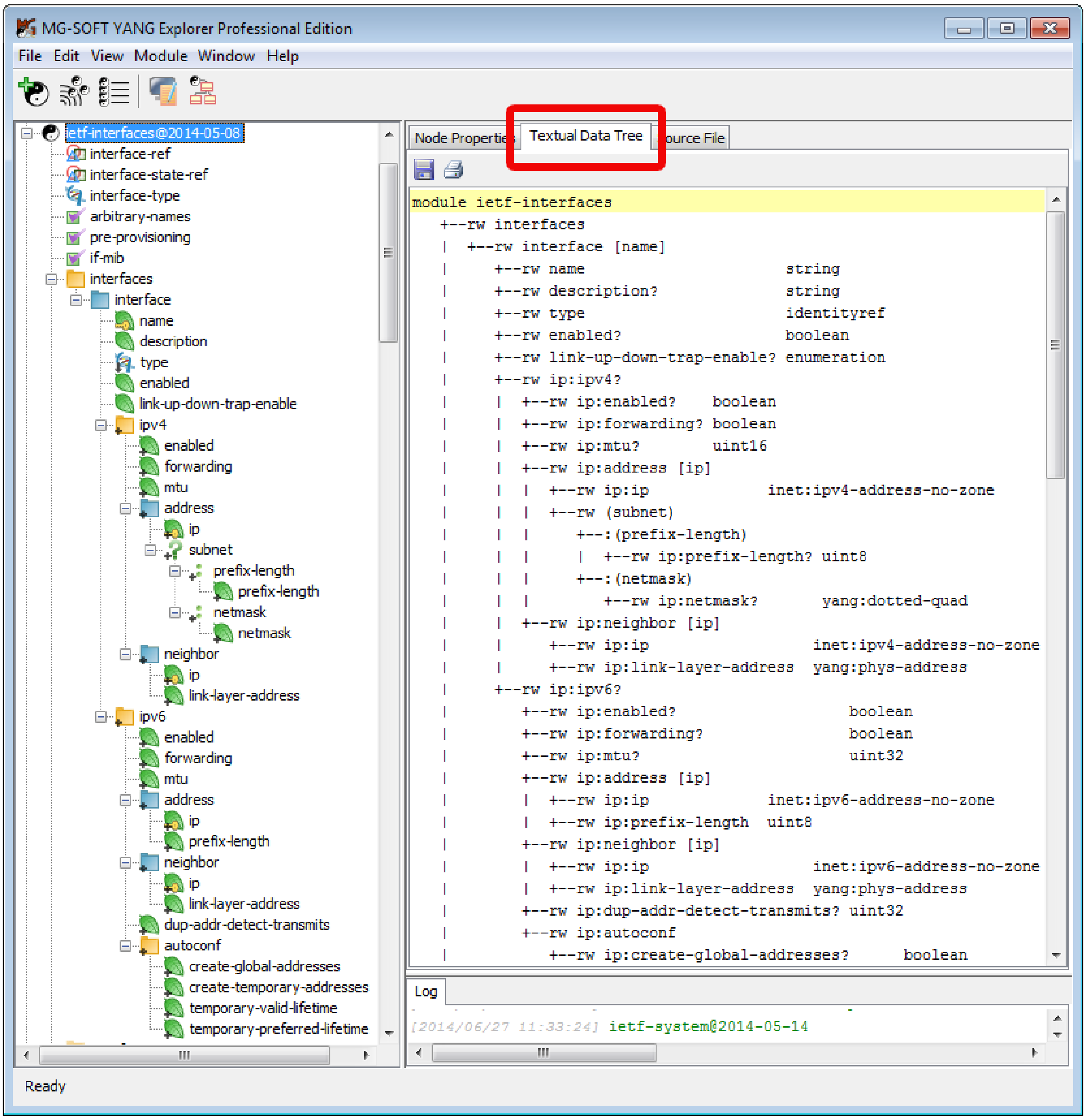
3. The software searches the document for the text string that matches the search pattern. By default, the **matches** (all occurrences of the found text) are **highlighted** in orange color (Figure 27).
4. Click the ***Next*** or the ***Previous*** button in the **Find** toolbar to jump to the next or preceding occurrence of the found text, respectively.

5. To stop the find operation, delete the search string from the ***Find*** input line or click the ***Close*** (✖) button in the right section of the Find toolbar (this also hides the toolbar).

5.5 Viewing Textual Data Tree Diagram

The **Textual Data Tree** tab in the Node Details panel displays a simplified text-based data tree diagram* of the node/subtree selected in the YANG Tree panel.

If a module node (☯) or submodule (☯) node is selected in the YANG Tree panel, the complete configuration and operational state data tree of the selected module or submodule is displayed in the Textual Data Tree tab, using ASCII characters only (e.g.Figure 28).



```
module ietf-interfaces
   +--rw interfaces
   |  +--rw interface [name]
   |     +--rw name                      string
   |     +--rw description?              string
   |     +--rw type                      identityref
   |     +--rw enabled?                  boolean
   |     +--rw link-up-down-trap-enable? enumeration
   |     +--rw ip:ipv4?
   |     |  +--rw ip:enabled?    boolean
   |     |  +--rw ip:forwarding? boolean
   |     |  +--rw ip:mtu?        uint16
   |     |  +--rw ip:address [ip]
   |     |  |  +--rw ip:ip             inet:ipv4-address-no-zone
   |     |  |  +--rw (subnet)
   |     |  |     +--:(prefix-length)
   |     |  |     |  +--rw ip:prefix-length? uint8
   |     |  |     +--:(netmask)
   |     |  |        +--rw ip:netmask?       yang:dotted-quad
   |     |  +--rw ip:neighbor [ip]
   |     |     +--rw ip:ip                  inet:ipv4-address-no-zone
   |     |     +--rw ip:link-layer-address  yang:phys-address
   |     +--rw ip:ipv6?
   |        +--rw ip:enabled?                   boolean
   |        +--rw ip:forwarding?                boolean
   |        +--rw ip:mtu?                       uint32
   |        +--rw ip:address [ip]
   |        |  +--rw ip:ip             inet:ipv6-address-no-zone
   |        |  +--rw ip:prefix-length  uint8
   |        +--rw ip:neighbor [ip]
   |        |  +--rw ip:ip                  inet:ipv6-address-no-zone
   |        |  +--rw ip:link-layer-address  yang:phys-address
   |        +--rw ip:dup-addr-detect-transmits? uint32
   |        +--rw ip:autoconf
   |           +--rw ip:create-global-addresses?      boolean
```

```
[2014/06/27 11:33:24] ietf-system@2014-05-14
```

Figure 28: Viewing the textual data tree diagram of a YANG module

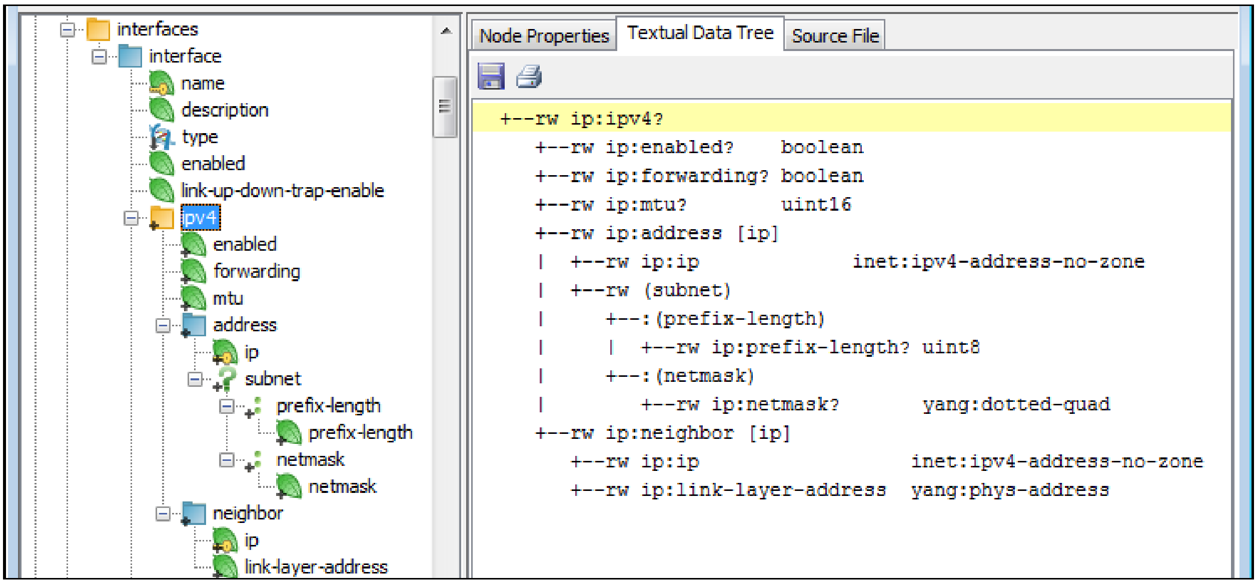
* This format is commonly used in IETF NETMOD (NETCONF Data Modeling Language) Working Group mailing lists, Internet-Draft and RFC documents.

The meaning of the symbols in textual data tree diagrams is as follows:

- Abbreviations before data node names: "**rw**" means configuration (read-write), "**ro**" state data (read-only), "**n**" denotes a "notification", and "**x**" stands for "rpc".
- Data type of every "leaf" node is shown near the right end of the corresponding line (e.g., **string**, **uint32**, **boolean**, etc.).
- Brackets "**[**" and "**]**" enclose list keys.
- Curly braces "**{**" and "**}**" contain names of optional features that make the corresponding node conditional.

- Symbols after data node names: "**?**" means an optional node and "*****" denotes a "list" or "leaf-list" node.
- Parentheses "`(`" and "`)`" enclose “choice” and “case” nodes, and “case” nodes are also marked with a colon ("**:**").
- Ellipsis ("**...**") stands for contents of subtrees that are not shown.

Click any data node (`container`, `list`, `leaf-list`, `leaf`, `anyxml`) in the YANG Tree panel to view the corresponding textual data tree in the Textual Data Tree tab in the Node Details window panel (Figure 29).



```
+--rw ip:ipv4?
   +--rw ip:enabled?    boolean
   +--rw ip:forwarding? boolean
   +--rw ip:mtu?        uint16
   +--rw ip:address [ip]
   |  +--rw ip:ip             inet:ipv4-address-no-zone
   |  +--rw (subnet)
   |     +--:(prefix-length)
   |     |  +--rw ip:prefix-length? uint8
   |     +--:(netmask)
   |        +--rw ip:netmask?       yang:dotted-quad
   +--rw ip:neighbor [ip]
      +--rw ip:ip                 inet:ipv4-address-no-zone
      +--rw ip:link-layer-address yang:phys-address
```

Figure 29: Viewing the textual data tree diagram of a container node

To copy as text the contents of the Textual Data Tree tab, select it, right-click it and choose the ***Copy*** command from the context (pop-up) menu.

To save the contents of the Textual Data Tree tab to a plain ASCII text file (.txt), click the ***Save As*** ( ) button in the Textual Data Tree tab toolbar and specify the location and name of the file to be saved in the standard Save As dialog box.

To send the contents of the Textual Data Tree tab to a printer, click the ***Print*** ( ) button in the Textual Data Tree tab toolbar and select the desired printer (or PDF writer driver, for example) from the standard Print dialog box.

6 LOADING YANG AND YIN MODULES IN YANG EXPLORER

YANG Explorer supports loading NETCONF modules in YANG and YIN format.

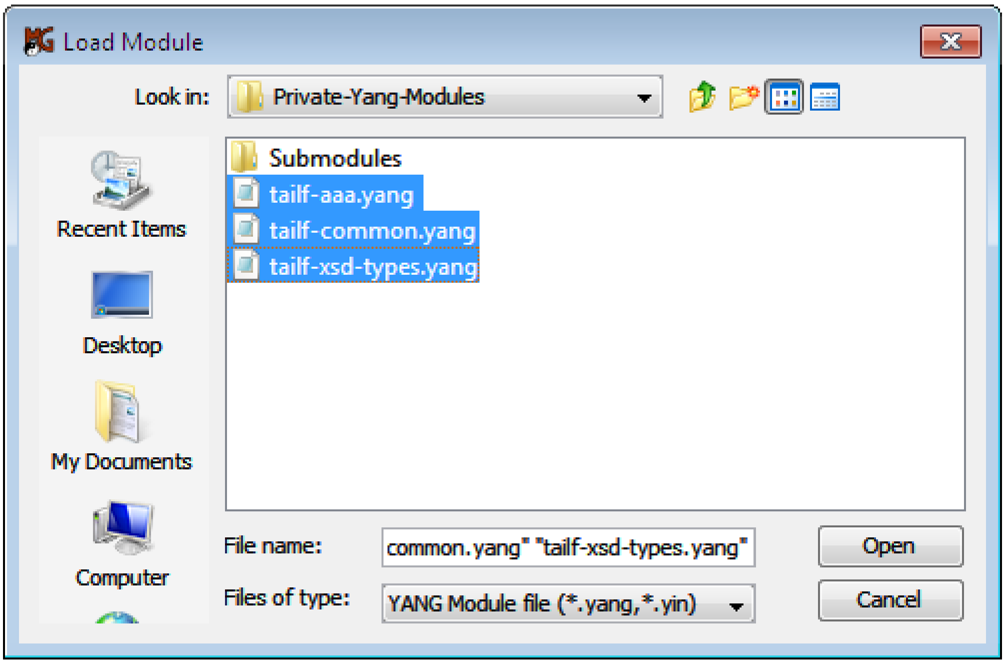
When YANG Explorer is started for the first time, it automatically loads all standard YANG modules that are included in the distribution and graphically displays loaded modules in the YANG Tree panel in the left portion of the main window. Additional, vendor-specific YANG or YIN modules can be loaded by the user and existing modules can be unloaded.

The loading of private modules in YANG Explorer is an important step that will provide you with a clear overview of the node hierarchy and node attributes representing the configuration and state data implemented in the managed NETCONF device.

6.1 Loading YANG and YIN Modules

To load a YANG or YIN module (and all dependent modules it imports and includes):

1. Copy the YANG or YIN module(s) you received from the vendor of the NETCONF device to a local folder of your choice.
2. In YANG Explorer, select the ***Module / Load Module*** command from the main menu.
3. The Load Module dialog box appears (Figure 30). In this dialog box, navigate to the folder containing the private YANG or YIN module(s), select one or more modules (use the **CTRL** or **SHIFT** keyboard key to select more than one module), and click the ***Open*** button.



Look in: Private-Yang-Modules

Submodules
tailf-aaa.yang
tailf-common.yang
tailf-xsd-types.yang

File name: common.yang" "tailf-xsd-types.yang"

Files of type: YANG Module file (*.yang,*.yin)

Figure 30: Selecting the YANG modules to load into YANG Explorer

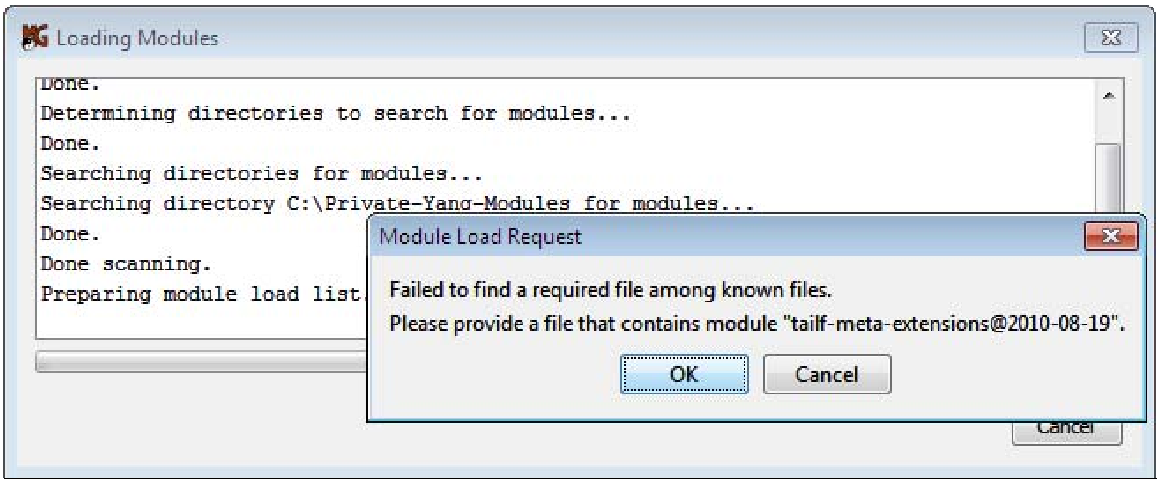
Tip: To select more than one file to load, hold down the ***Ctrl*** key on the keyboard and click the desired module files.

4. YANG Explorer first starts scanning the current folder for files that contain valid YANG and YIN modules to build a list of known modules, i.e., for all modules it detects, the module name, its revision and full path of the file defining this module is stored in the

program cache (known modules list) for future use. When done, YANG Explorer stars parsing the selected YANG or YIN module(s) and checking their consistency. While parsing the module(s), the progress and error messages are being logged in the Log window panel at the bottom of the main window. During this process, every module is also checked for dependencies, i.e., the modules it imports and submodules it includes (this is done recursively for all referenced modules).

- If any dependency is found that is not "known" to YANG Explorer, the Module Load Request dialog box appears (Figure 31) prompting you to specify the location of the file that defines the referenced (sub)module.

> **Note:** "Known modules" are those modules for which information in the program cache already exists, i.e., the standard modules that ship with YANG Explorer and those private modules that have already been loaded in YANG Explorer.

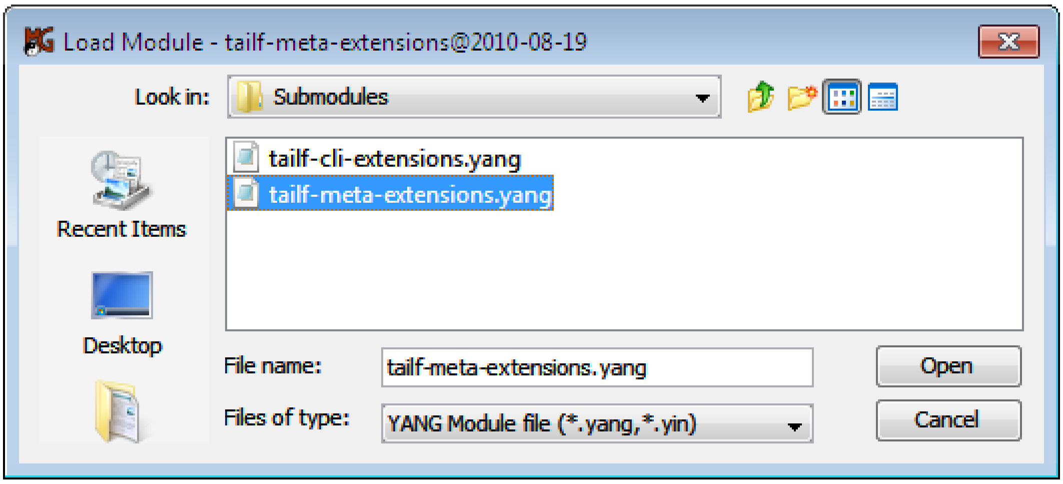


```
Done.
Determining directories to search for modules...
Done.
Searching directories for modules...
Searching directory C:\Private-Yang-Modules for modules...
Done.
Done scanning.
Preparing module load list.
```

Failed to find a required file among known files.
Please provide a file that contains module "tailf-meta-extensions@2010-08-19".

Figure 31: YANG Explorer prompts you to provide the location of the required module

- Click the ***OK*** button in the Module Load Request dialog box to close it and display the Load Module dialog box. In the Load Module dialog box, navigate to the file containing the definition of the required module, select it, and click the ***Open*** button (Figure 32).



Look in: Submodules

tailf-cli-extensions.yang
tailf-meta-extensions.yang

File name: tailf-meta-extensions.yang

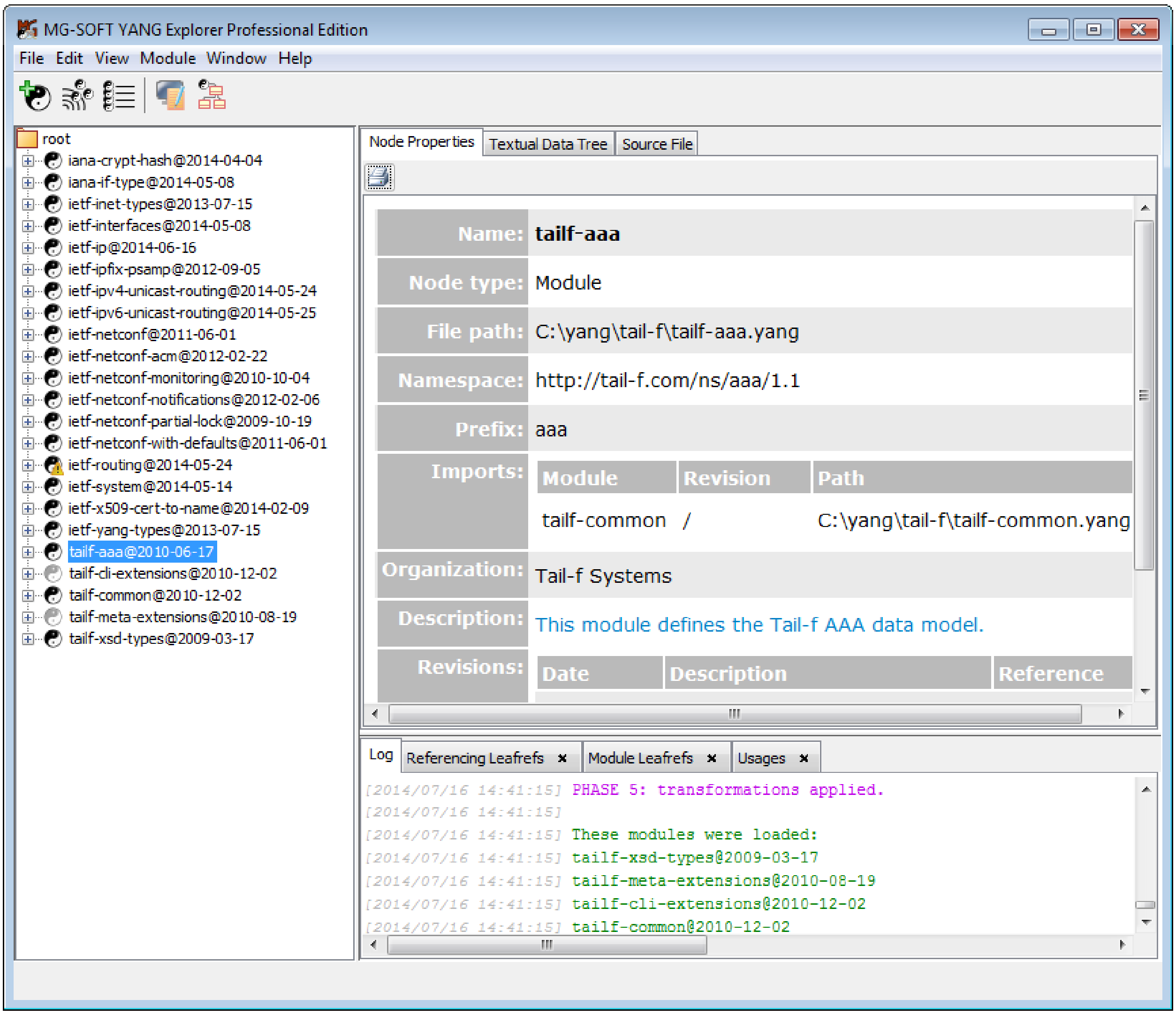
Files of type: YANG Module file (*.yang,*.yin)

Figure 32: Loading a YANG module

- Again, the specified folder is first scanned for files that contain valid YANG and YIN modules to build a list of known modules, then the selected module is validated and loaded.
- If you are prompted for any other missing module, repeat the above procedure to specify its location.

> **Note:** Once a module has been loaded from disk, this module and all other modules from the same folder (and optionally all its subfolders) are "registered" and can later be loaded from the **Known Modules** dialog box.

5. After the selected module(s) have been successfully parsed and validated (no syntax or semantic errors were found), the module(s) are loaded and displayed in the YANG Tree panel in the left portion of the main window (Figure 33). You can expand the loaded modules to view their tree structure and the properties of nodes, as described in the next section.


MG-SOFT YANG Explorer Professional Edition
File Edit View Module Window Help
root
iana-crypt-hash@2014-04-04
iana-if-type@2014-05-08
ietf-inet-types@2013-07-15
ietf-interfaces@2014-05-08
ietf-ip@2014-06-16
ietf-ipfix-psamp@2012-09-05
ietf-ipv4-unicast-routing@2014-05-24
ietf-ipv6-unicast-routing@2014-05-25
ietf-netconf@2011-06-01
ietf-netconf-acm@2012-02-22
ietf-netconf-monitoring@2010-10-04
ietf-netconf-notifications@2012-02-06
ietf-netconf-partial-lock@2009-10-19
ietf-netconf-with-defaults@2011-06-01
ietf-routing@2014-05-24
ietf-system@2014-05-14
ietf-x509-cert-to-name@2014-02-09
ietf-yang-types@2013-07-15
tailf-aaa@2010-06-17
tailf-cli-extensions@2010-12-02
tailf-common@2010-12-02
tailf-meta-extensions@2010-08-19
tailf-xsd-types@2009-03-17
Node Properties
Textual Data Tree
Source File
Log
Referencing Leafrefs
Module Leafrefs
Usages

Name:	**tailf-aaa**
Node type:	Module
File path:	C:\yang\tail-f\tailf-aaa.yang
Namespace:	http://tail-f.com/ns/aaa/1.1
Prefix:	aaa
Imports:	Module: tailf-common Revision: / Path: C:\yang\tail-f\tailf-common.yang
Organization:	Tail-f Systems
Description:	This module defines the Tail-f AAA data model.
Revisions:	Date / Description / Reference

```
[2014/07/16 14:41:15] PHASE 5: transformations applied.
[2014/07/16 14:41:15]
[2014/07/16 14:41:15] These modules were loaded:
[2014/07/16 14:41:15] tailf-xsd-types@2009-03-17
[2014/07/16 14:41:15] tailf-meta-extensions@2010-08-19
[2014/07/16 14:41:15] tailf-cli-extensions@2010-12-02
[2014/07/16 14:41:15] tailf-common@2010-12-02
```

Figure 33: Newly loaded modules displayed in the YANG Tree window panel

The location of the standard YANG modules bundled with YANG Explorer depends on the operating system used:

Windows: My Documents\MGSOFTYangExplorer\modules\

Windows Vista, Windows 7, Windows Server 2008:
C:\Users\[username]\Documents\MGSOFTYangExplorer\modules\yang

Windows XP, Server 2003:
C:\Documents and Settings\[username]\My Documents\ MGSOFTYangExplorer\modules\yang

Linux: ~/Documents/MGSOFTYangExplorer/modules/
/home/[username]/Documents/MGSOFTYangExplorer/modules/yang/

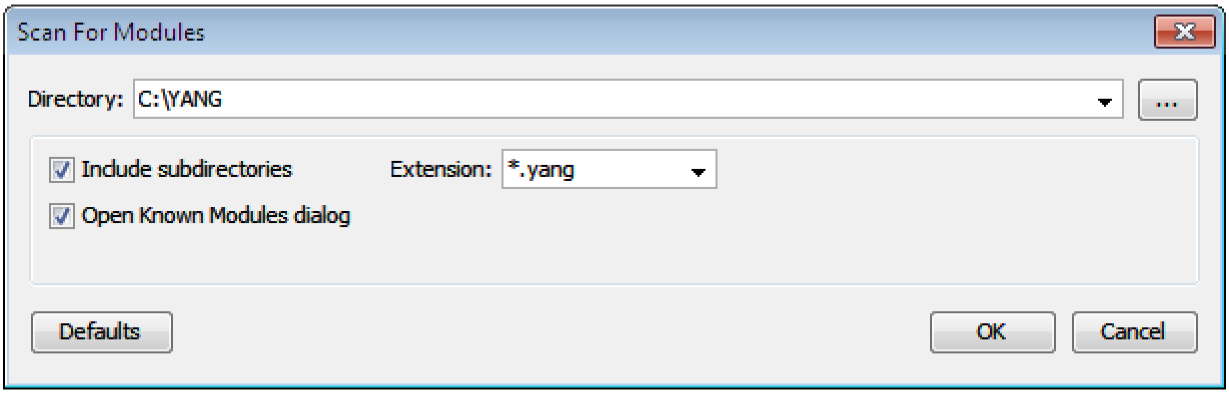
Mac OS X: ~/Documents/MGSOFTYangExplorer/modules/
/Users/[username]/Documents/MGSOFTYangExplorer/modules/yang

6.2 Scanning Folders for YANG and YIN Modules

YANG Explorer incorporates a convenient functionality that lets you scan selected folders (and optionally also subfolders) for files that contain YANG or YIN modules and automatically “register” them for use with YANG Explorer. Registered files are added to the known modules cache and appear in the Known Modules dialog box, from where they can be loaded into YANG Explorer.

To scan a folder for YANG or YIN module files:

1. In YANG Explorer, select the ***Module / Scan for Modules*** command from the main menu. The Scan for Modules dialog box appears (Figure 34).



Directory: C:\YANG

☑ Include subdirectories

Extension: *.yang

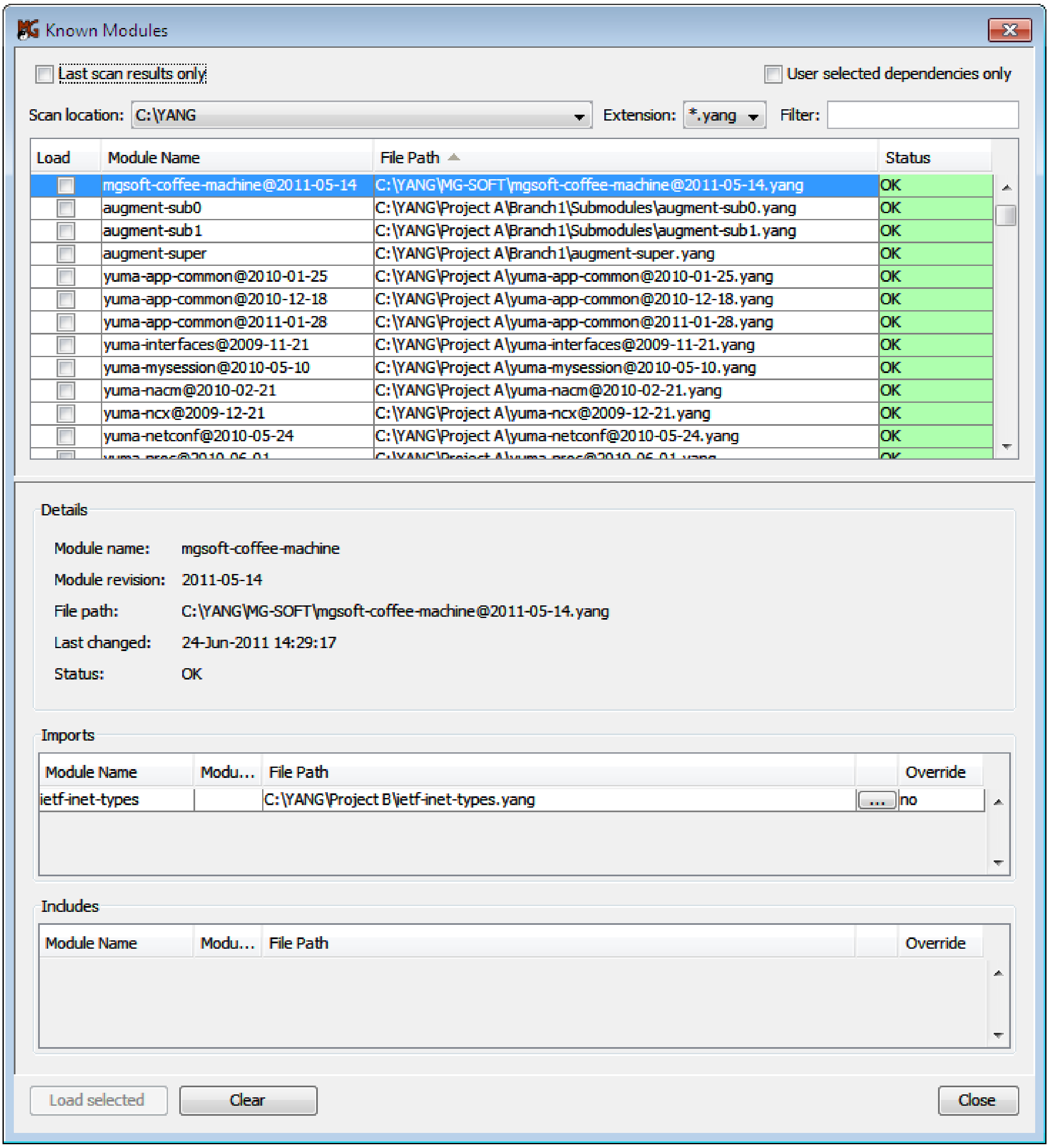
☑ Open Known Modules dialog

Figure 34: The Scan for Modules dialog box

2. In the Scan for Modules dialog box, specify the following:
   - In the ***Directory*** drop-down list, enter the full path of the folder containing (private) YANG or YIN modules,
   - Check the ***Include subdirectories*** checkbox if you want to scan also all the subfolders of the specified folder (directory).

- ❑ In the ***Extension*** drop-down list, select the file mask (e.g., *.yang, *yin, all) to be used for finding the YANG and YIN modules. Only files with the selected extension(s) will be taken into account when scanning for modules.
- ❑ Check the ***Open Known Modules dialog*** checkbox if you want to view the results of the scan operation in the **Known Modules** dialog box.
- ❑ Click the ***Defaults*** button to revert the settings in this dialog box to the default values.

3. Click the ***OK*** button to close the Scan for Modules dialog box and start scanning the selected folder. YANG Explorer scans the specified folder (and its subfolder if selected so) for files that contain valid YANG and YIN modules to build a list of known modules, i.e., for all modules it detects, the module name, its revision and full path of the file defining this module is stored in the program cache (known modules list) for future use. During the scan operation, every module is also checked for dependencies, i.e., YANG Explorer verifies if all the module(s) it imports and submodule(s) it includes are available. The progress of the scan operation is displayed in the Module scan dialog box.
4. When the scan operation finishes, the results are displayed in the Known Modules dialog box (Figure 35):
   - ❑ Select a module in the list of known module files in the upper panel, to view the module details (location, last modified date, etc.) in the middle section, and the module files it imports and submodules files it includes in the lower panel.
   - ❑ The status column in the list of known modules indicates the **OK** status if all dependencies (imported module files and included submodule files) for the given module are available. If the status of a module is not **OK**, you can specify the location of the missing (sub)module as described in this section.
5. To view only the new modules that have been found in the last scan operation, select the ***Last scan results only*** checkbox in the Known Modules dialog box.
6. To filter results by file extension, i.e., to view only modules saved in files with a specific filename extension, select the desired extension (e.g., *.yang, *.yin) form the ***Extension*** drop-down list in the Known Modules dialog box.
7. To filter results by text, i.e., to view only those lines that contain a specific text string, enter the desired text string into the ***Filter*** input line in the Known Modules dialog box.
8. To view the entire known modules cache (all the registered modules), select the **All** entry from the ***Scan location*** drop-down list in the Known Modules dialog box.
9. To load one or more modules (and their dependencies), check the ***Load*** checkbox of the modules you want to load in the list of known module files and click the ***Load Selected*** button at the bottom of the Known Modules dialog box (Figure 35).



□ Last scan results only

□ User selected dependencies only

Scan location: C:\YANG

Extension: *.yang

Filter:

Load	Module Name	File Path	Status
□	mgsoft-coffee-machine@2011-05-14	C:\YANG\MG-SOFT\mgsoft-coffee-machine@2011-05-14.yang	OK
□	augment-sub0	C:\YANG\Project A\Branch1\Submodules\augment-sub0.yang	OK
□	augment-sub1	C:\YANG\Project A\Branch1\Submodules\augment-sub1.yang	OK
□	augment-super	C:\YANG\Project A\Branch1\augment-super.yang	OK
□	yuma-app-common@2010-01-25	C:\YANG\Project A\yuma-app-common@2010-01-25.yang	OK
□	yuma-app-common@2010-12-18	C:\YANG\Project A\yuma-app-common@2010-12-18.yang	OK
□	yuma-app-common@2011-01-28	C:\YANG\Project A\yuma-app-common@2011-01-28.yang	OK
□	yuma-interfaces@2009-11-21	C:\YANG\Project A\yuma-interfaces@2009-11-21.yang	OK
□	yuma-mysession@2010-05-10	C:\YANG\Project A\yuma-mysession@2010-05-10.yang	OK
□	yuma-nacm@2010-02-21	C:\YANG\Project A\yuma-nacm@2010-02-21.yang	OK
□	yuma-ncx@2009-12-21	C:\YANG\Project A\yuma-ncx@2009-12-21.yang	OK
□	yuma-netconf@2010-05-24	C:\YANG\Project A\yuma-netconf@2010-05-24.yang	OK
□	yuma-proc@2010-06-01	C:\YANG\Project A\yuma-proc@2010-06-01.yang	OK

Details

Module name: mgsoft-coffee-machine

Module revision: 2011-05-14

File path: C:\YANG\MG-SOFT\mgsoft-coffee-machine@2011-05-14.yang

Last changed: 24-Jun-2011 14:29:17

Status: OK

Imports

Module Name	Modu...	File Path		Override
ietf-inet-types		C:\YANG\Project B\ietf-inet-types.yang	...	no

Includes

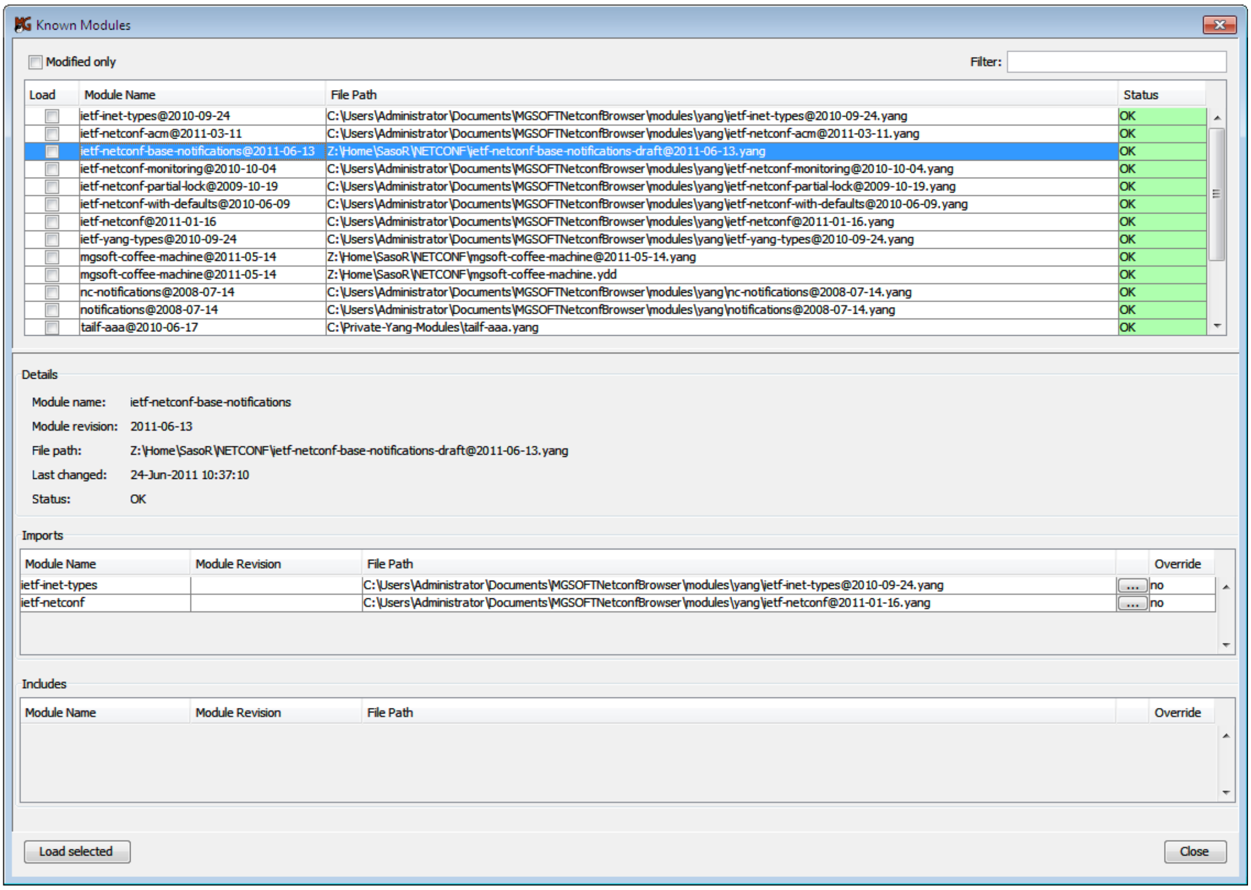
Module Name	Modu...	File Path		Override

Figure 35: The Known Modules dialog box listing YANG modules found in the scanned folder tree

6.3 Loading Known YANG and YIN Modules

Once a folder has been scanned for files that contain YANG or YIN modules, all module files from that folder (and optionally from all its subfolders) are “registered” and can later be loaded from the Known Modules dialog box.

The Known Modules dialog box also lets you view and configure dependencies for each module, that is, the module(s) it imports and the submodule(s) it includes.



Load	Module Name	File Path	Status
□	ietf-inet-types@2010-09-24	C:\Users\Administrator\Documents\MGSOFTNetconfBrowser\modules\yang\ietf-inet-types@2010-09-24.yang	OK
□	ietf-netconf-acm@2011-03-11	C:\Users\Administrator\Documents\MGSOFTNetconfBrowser\modules\yang\ietf-netconf-acm@2011-03-11.yang	OK
□	ietf-netconf-base-notifications@2011-06-13	Z:\Home\SasoR\NETCONF\ietf-netconf-base-notifications-draft@2011-06-13.yang	OK
□	ietf-netconf-monitoring@2010-10-04	C:\Users\Administrator\Documents\MGSOFTNetconfBrowser\modules\yang\ietf-netconf-monitoring@2010-10-04.yang	OK
□	ietf-netconf-partial-lock@2009-10-19	C:\Users\Administrator\Documents\MGSOFTNetconfBrowser\modules\yang\ietf-netconf-partial-lock@2009-10-19.yang	OK
□	ietf-netconf-with-defaults@2010-06-09	C:\Users\Administrator\Documents\MGSOFTNetconfBrowser\modules\yang\ietf-netconf-with-defaults@2010-06-09.yang	OK
□	ietf-netconf@2011-01-16	C:\Users\Administrator\Documents\MGSOFTNetconfBrowser\modules\yang\ietf-netconf@2011-01-16.yang	OK
□	ietf-yang-types@2010-09-24	C:\Users\Administrator\Documents\MGSOFTNetconfBrowser\modules\yang\ietf-yang-types@2010-09-24.yang	OK
□	mgsoft-coffee-machine@2011-05-14	Z:\Home\SasoR\NETCONF\mgsoft-coffee-machine@2011-05-14.yang	OK
□	mgsoft-coffee-machine@2011-05-14	Z:\Home\SasoR\NETCONF\mgsoft-coffee-machine.ydd	OK
□	nc-notifications@2008-07-14	C:\Users\Administrator\Documents\MGSOFTNetconfBrowser\modules\yang\nc-notifications@2008-07-14.yang	OK
□	notifications@2008-07-14	C:\Users\Administrator\Documents\MGSOFTNetconfBrowser\modules\yang\notifications@2008-07-14.yang	OK
□	tailf-aaa@2010-06-17	C:\Private-Yang-Modules\tailf-aaa.yang	OK

Details

Module name: ietf-netconf-base-notifications
Module revision: 2011-06-13
File path: Z:\Home\SasoR\NETCONF\ietf-netconf-base-notifications-draft@2011-06-13.yang
Last changed: 24-Jun-2011 10:37:10
Status: OK

Imports

Module Name	Module Revision	File Path		Override
ietf-inet-types		C:\Users\Administrator\Documents\MGSOFTNetconfBrowser\modules\yang\ietf-inet-types@2010-09-24.yang	...	no
ietf-netconf		C:\Users\Administrator\Documents\MGSOFTNetconfBrowser\modules\yang\ietf-netconf@2011-01-16.yang	...	no

Includes

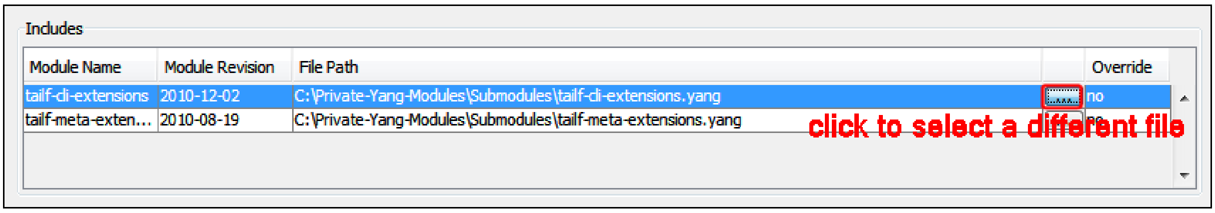
Module Name	Module Revision	File Path		Override

Figure 36: The Known Modules dialog box (viewing details of selected module)

To load a YANG or YIN module (and all dependent modules it imports and includes) from the Known Modules dialog box:

1. In YANG Explorer, select the ***Module / Known Modules*** command from the main menu.
2. The Known Modules dialog box appears, listing all YANG and YIN modules that are currently known to YANG Explorer (Figure 36).
   - ❑ Select a module in the list of known module files in the upper panel, to view the module details (location, last modified date, etc.) in the middle section, and the module files it imports and submodules files it includes in the lower panel.

- The status column in the list of known modules indicates the **OK** status if all dependencies (imported module files and included submodule files) for the given module are available.
- If the status of a module is not **OK**, or if the status is **OK**, but you would like to change the path of the files that will be imported or included by the module, click the ***Browse …*** button next to the Override column in the list of Imports or Includes (Figure 37)

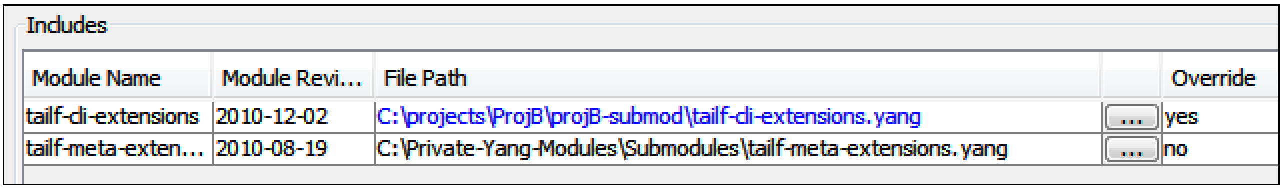


Includes

Module Name	Module Revision	File Path		Override
taif-di-extensions	2010-12-02	C:\Private-Yang-Modules\Submodules\tailf-di-extensions.yang	…	no
tailf-meta-exten…	2010-08-19	C:\Private-Yang-Modules\Submodules\tailf-meta-extensions.yang	…	no

Figure 37: Specifying a different include file in the Known Modules dialog box

- In the Edit Override dialog box that appears, select the ***Browse…*** entry and click the ***OK*** button to open the Load Module dialog box (Figure 30) that lets you browse the file system and select a different file containing the definition of the given submodule.

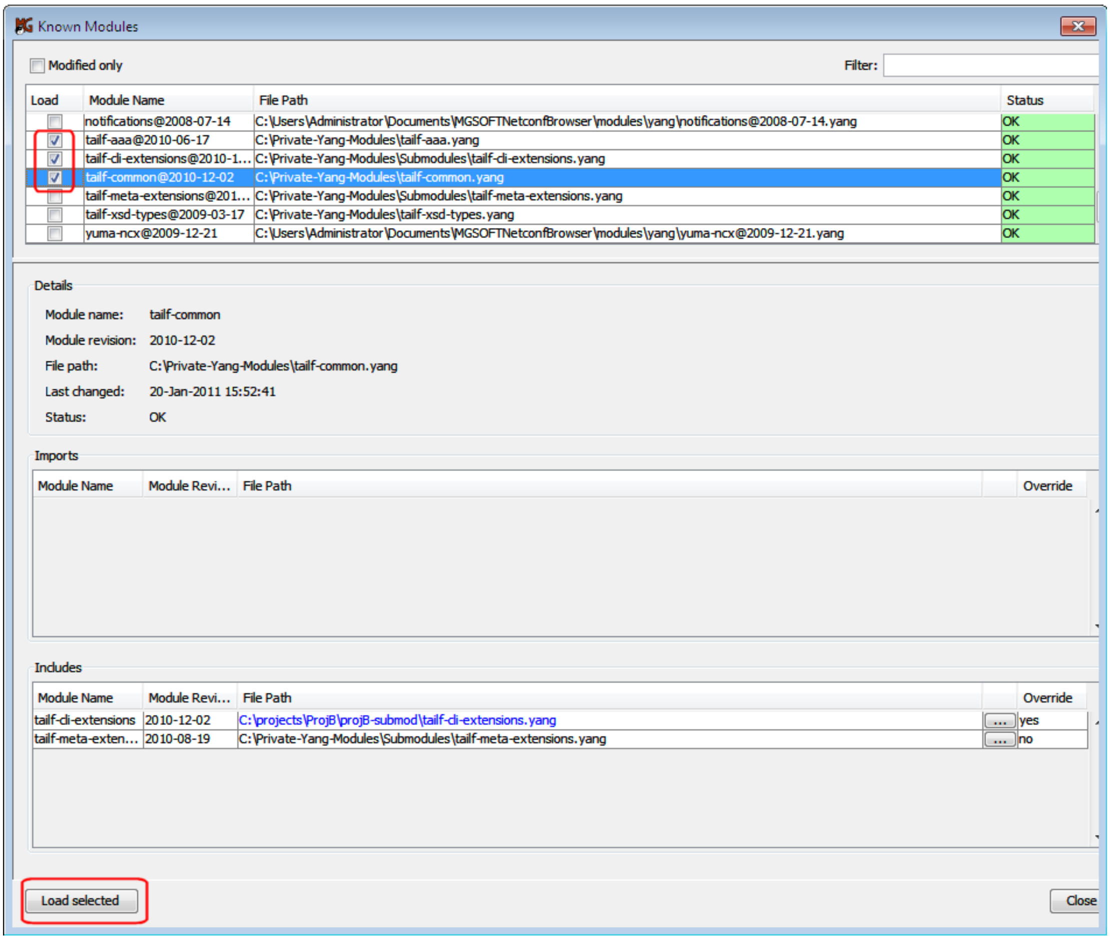


Includes

Module Name	Module Revi…	File Path		Override
tailf-di-extensions	2010-12-02	C:\projects\ProjB\projB-submod\tailf-di-extensions.yang	…	yes
tailf-meta-exten…	2010-08-19	C:\Private-Yang-Modules\Submodules\tailf-meta-extensions.yang	…	no

Figure 38: User overridden include submodule is displayed in blue

3. Check the ***Load*** checkbox of the modules (and their dependencies) you want to load in the list of known module files and click the ***Load Selected*** button at the bottom of the Known Modules dialog box (Figure 39).



□ Modified only

Filter:

Load	Module Name	File Path	Status
□	notifications@2008-07-14	C:\Users\Administrator\Documents\MGSOFTNetconfBrowser\modules\yang\notifications@2008-07-14.yang	OK
☑	tailf-aaa@2010-06-17	C:\Private-Yang-Modules\tailf-aaa.yang	OK
☑	tailf-di-extensions@2010-1...	C:\Private-Yang-Modules\Submodules\tailf-di-extensions.yang	OK
☑	tailf-common@2010-12-02	C:\Private-Yang-Modules\tailf-common.yang	OK
□	tailf-meta-extensions@201...	C:\Private-Yang-Modules\Submodules\tailf-meta-extensions.yang	OK
□	tailf-xsd-types@2009-03-17	C:\Private-Yang-Modules\tailf-xsd-types.yang	OK
□	yuma-ncx@2009-12-21	C:\Users\Administrator\Documents\MGSOFTNetconfBrowser\modules\yang\yuma-ncx@2009-12-21.yang	OK

Details

Module name: tailf-common

Module revision: 2010-12-02

File path: C:\Private-Yang-Modules\tailf-common.yang

Last changed: 20-Jan-2011 15:52:41

Status: OK

Imports

Module Name	Module Revi...	File Path	Override

Includes

Module Name	Module Revi...	File Path		Override
tailf-di-extensions	2010-12-02	C:\projects\ProjB\projB-submod\tailf-di-extensions.yang	...	yes
tailf-meta-exten...	2010-08-19	C:\Private-Yang-Modules\Submodules\tailf-meta-extensions.yang	...	no

Figure 39: Loading selected YANG modules

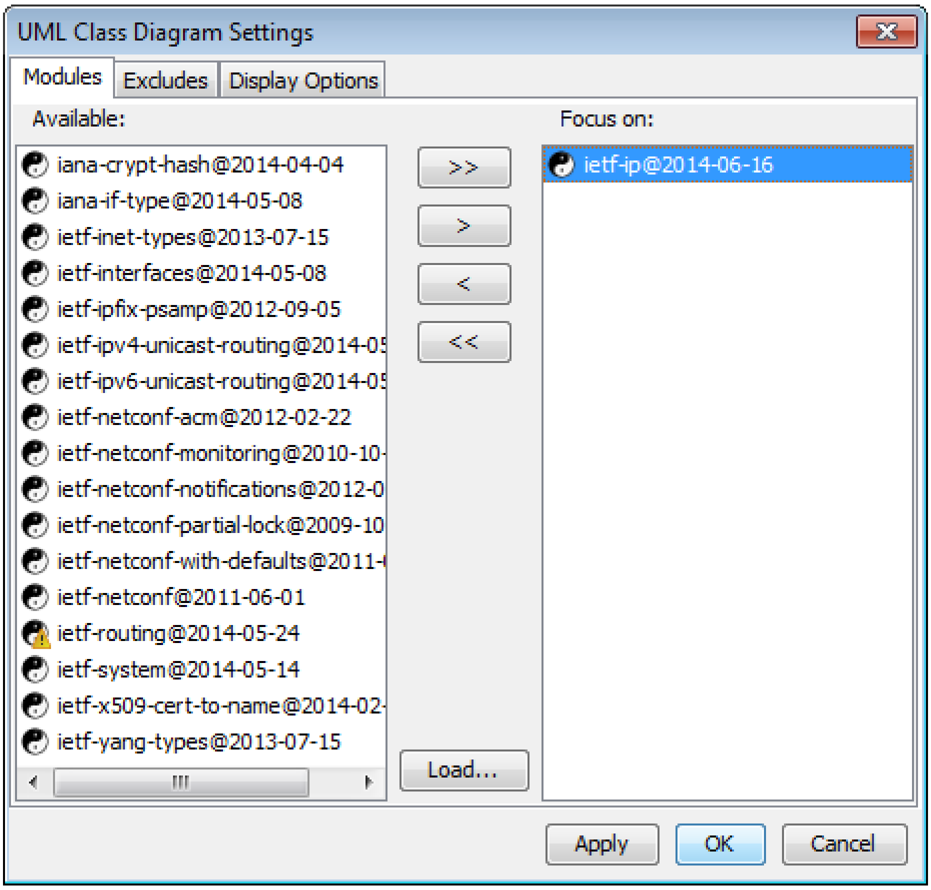
4. Known Modules dialog box closes and the selected modules load into the YANG tree panel in the main window (unless modules with the same names and revisions are already loaded – in such case you need to unload the currently loaded modules first and then repeat the loading procedure).

7 VIEWING UML CLASS DIAGRAM OF YANG MODULES

MG-SOFT YANG Explorer can present selected YANG module(s) in form of a UML (Unified Modeling Language) class diagram. UML class diagram displays different types of YANG nodes as different types of classes, with their attributes, methods and relationships between the objects (e.g., dependence, composition, inheritance, etc.). UML class diagram can be seen as a standard way of visualizing the YANG data model defined in one or more YANG modules.

This section describes how to view the UML class diagram of one or more YANG modules in the UML Class Diagram window. Furthermore, the procedure of printing and saving the UML class diagram to a bitmap image file is described.

1. In YANG Explorer, select the ***View / UML Class Diagram*** command from the main menu or click the ***UML Class Diagram*** toolbar button ( ).
2. The **UML Class Diagram** window appears (Figure 41). When you open the UML Class Diagram window for the first time, it displays no content. Otherwise, it displays the UML class diagram of the YANG module(s) selected in the previous session.
3. Click the ***UML Class Diagram Settings*** toolbar button ( ) to open the corresponding dialog box (Figure 40) to configure which YANG module(s) you want to include into UML diagram and optionally set other UML class diagram settings (e.g., excludes, display options).



Available:

- iana-crypt-hash@2014-04-04
- iana-if-type@2014-05-08
- ietf-inet-types@2013-07-15
- ietf-interfaces@2014-05-08
- ietf-ipfix-psamp@2012-09-05
- ietf-ipv4-unicast-routing@2014-05
- ietf-ipv6-unicast-routing@2014-05
- ietf-netconf-acm@2012-02-22
- ietf-netconf-monitoring@2010-10-
- ietf-netconf-notifications@2012-0
- ietf-netconf-partial-lock@2009-10
- ietf-netconf-with-defaults@2011-
- ietf-netconf@2011-06-01
- ietf-routing@2014-05-24
- ietf-system@2014-05-14
- ietf-x509-cert-to-name@2014-02-
- ietf-yang-types@2013-07-15

Focus on:

- ietf-ip@2014-06-16

Figure 40: Selecting the YANG module for UML class diagram

4. In the **UML Class Diagram Settings** dialog box, in the Modules tab (Figure 40), in the ***Available*** list on the left hand side, select the YANG module, whose UML class diagram you would like to view, and click the right-arrow button (>) to select the module (move it to the list on the right hand side).

> **Tip:** If the YANG/YIN module you would like to include into UML diagram is not currently loaded in YANG Explorer, you can click the ***Load*** button in the **UML Class Diagram Settings** dialog box and load the required module file directly from disk.

5. The UML class diagram of the selected module is displayed in the **UML Class Diagram** window (Figure 41). For more information on UML classes, their connections/relationships, packages, annotations, etc., please refer to the section Understanding the YANG Explorer UML Class Diagram.

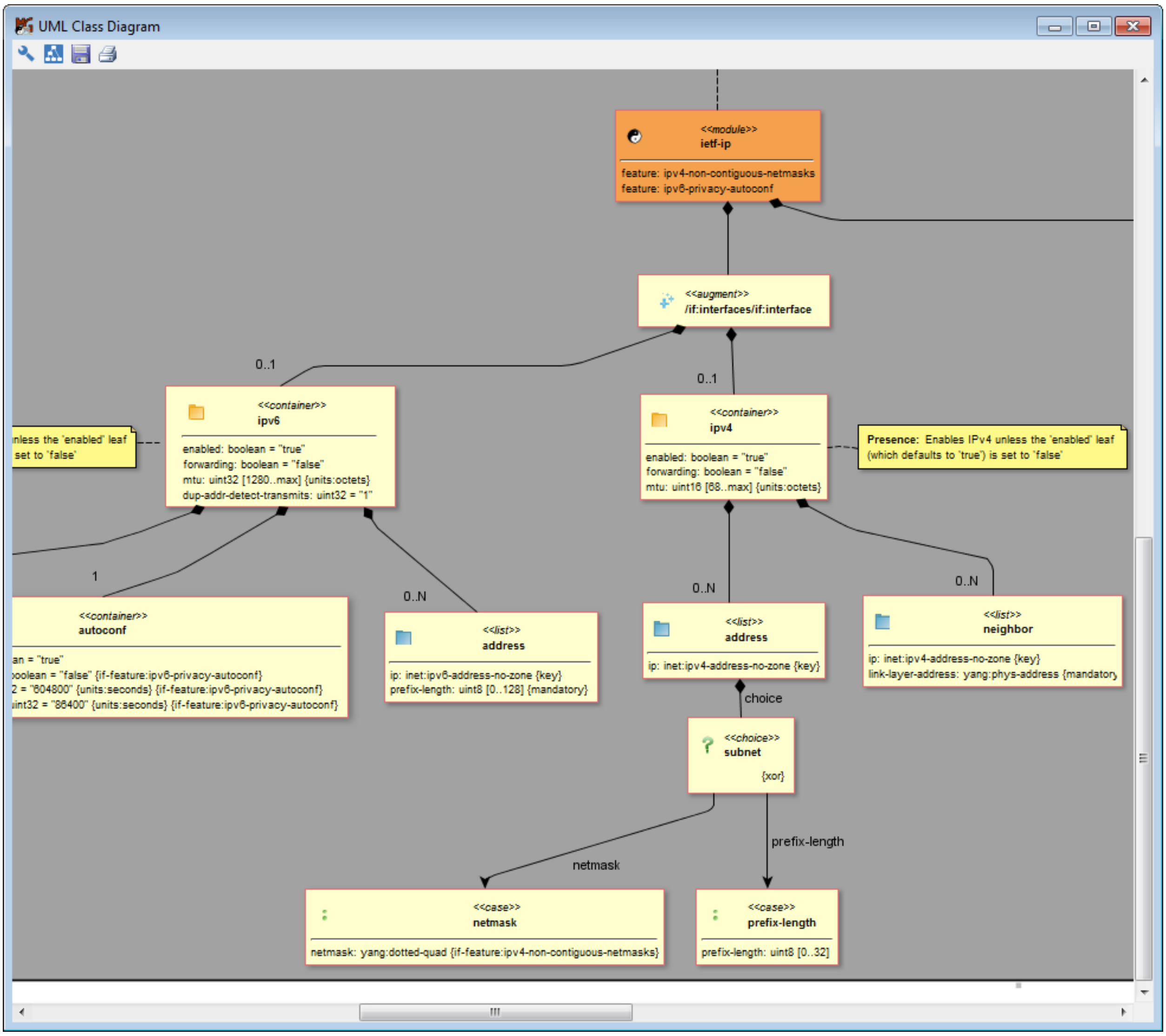

UML Class Diagram
<<module>>
ietf-ip
feature: ipv4-non-contiguous-netmasks
feature: ipv6-privacy-autoconf
<<augment>>
/if:interfaces/if:interface
0..1
0..1
<<container>>
ipv6
enabled: boolean = "true"
forwarding: boolean = "false"
mtu: uint32 [1280..max] {units:octets}
dup-addr-detect-transmits: uint32 = "1"
nless the 'enabled' leaf
set to 'false'
<<container>>
ipv4
enabled: boolean = "true"
forwarding: boolean = "false"
mtu: uint16 [68..max] {units:octets}
Presence: Enables IPv4 unless the 'enabled' leaf (which defaults to 'true') is set to 'false'
1
0..N
0..N
0..N
<<container>>
autoconf
an = "true"
oolean = "false" {if-feature:ipv6-privacy-autoconf}
2 = "604800" {units:seconds} {if-feature:ipv6-privacy-autoconf}
int32 = "86400" {units:seconds} {if-feature:ipv6-privacy-autoconf}
<<list>>
address
ip: inet:ipv6-address-no-zone {key}
prefix-length: uint8 [0..128] {mandatory}
<<list>>
address
ip: inet:ipv4-address-no-zone {key}
<<list>>
neighbor
ip: inet:ipv4-address-no-zone {key}
link-layer-address: yang:phys-address {mandatory
choice
<<choice>>
subnet
{xor}
netmask
prefix-length
<<case>>
netmask
netmask: yang:dotted-quad {if-feature:ipv4-non-contiguous-netmasks}
<<case>>
prefix-length
prefix-length: uint8 [0..32]

Figure 41: Viewing a portion of the UML class diagram of the *ietf-ip* YANG module

6. Use the mouse wheel to zoom in and out of the UML class diagram.

7. Hover the mouse pointer over a class box, to view the description of the class (if it has one) in a tooltip.
8. Right-click and drag the diagram to pan it inside the **UML Class Diagram** window while exploring the connections/relationships between classes and packages.
9. You can select and reposition individual elements (classes, connections, notes, packages, etc.), however, note that the diagram will be redrawn to match the original appearance if you make and apply any changes in the **UML Class Diagram Settings** dialog box.
10. To explore the connections between classes of two or more associated YANG modules, open the **UML Class Diagram Settings** dialog box (Figure 40) and include the associated (e.g., imported) module(s) into UML diagram (Figure 42).

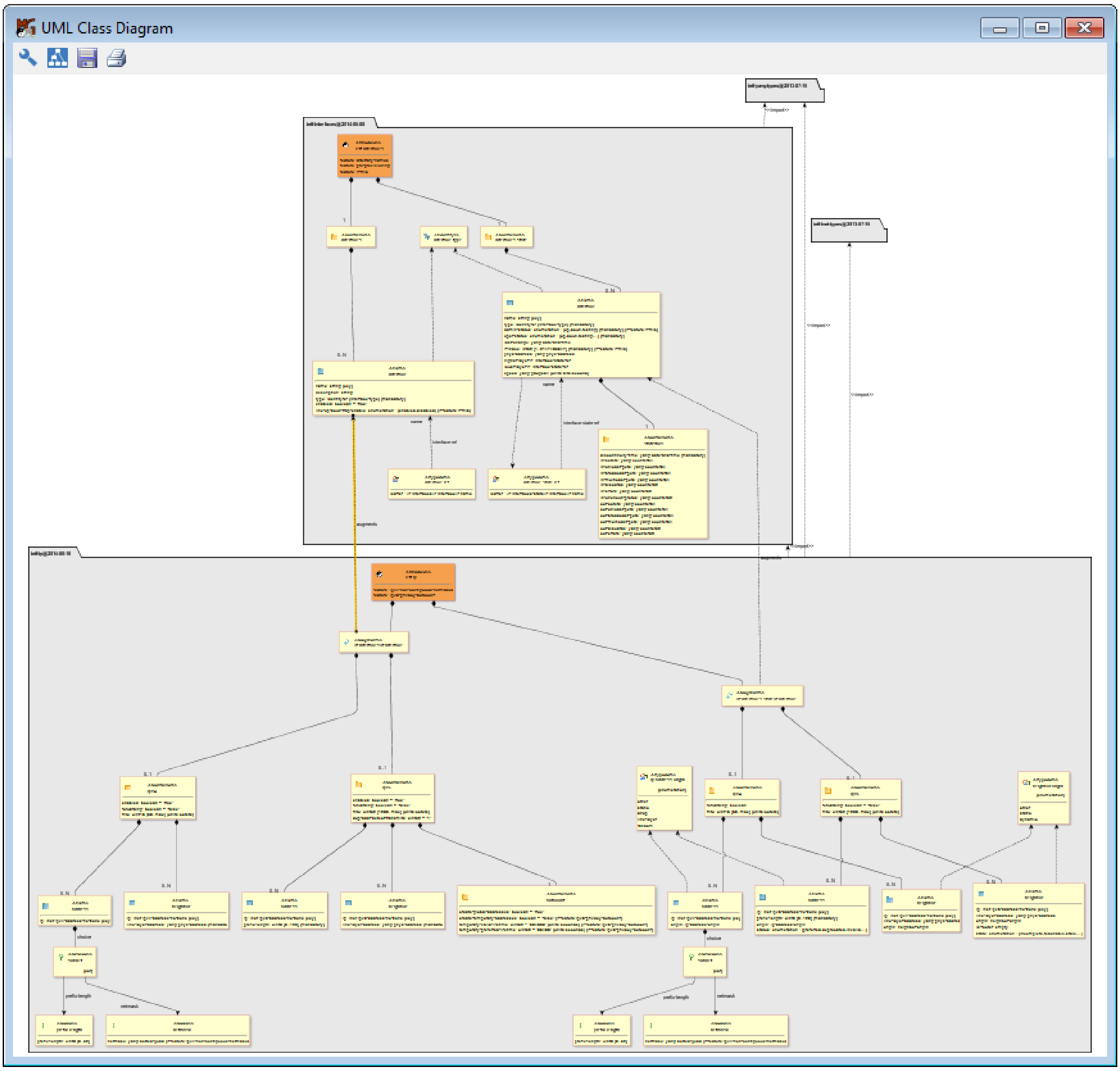
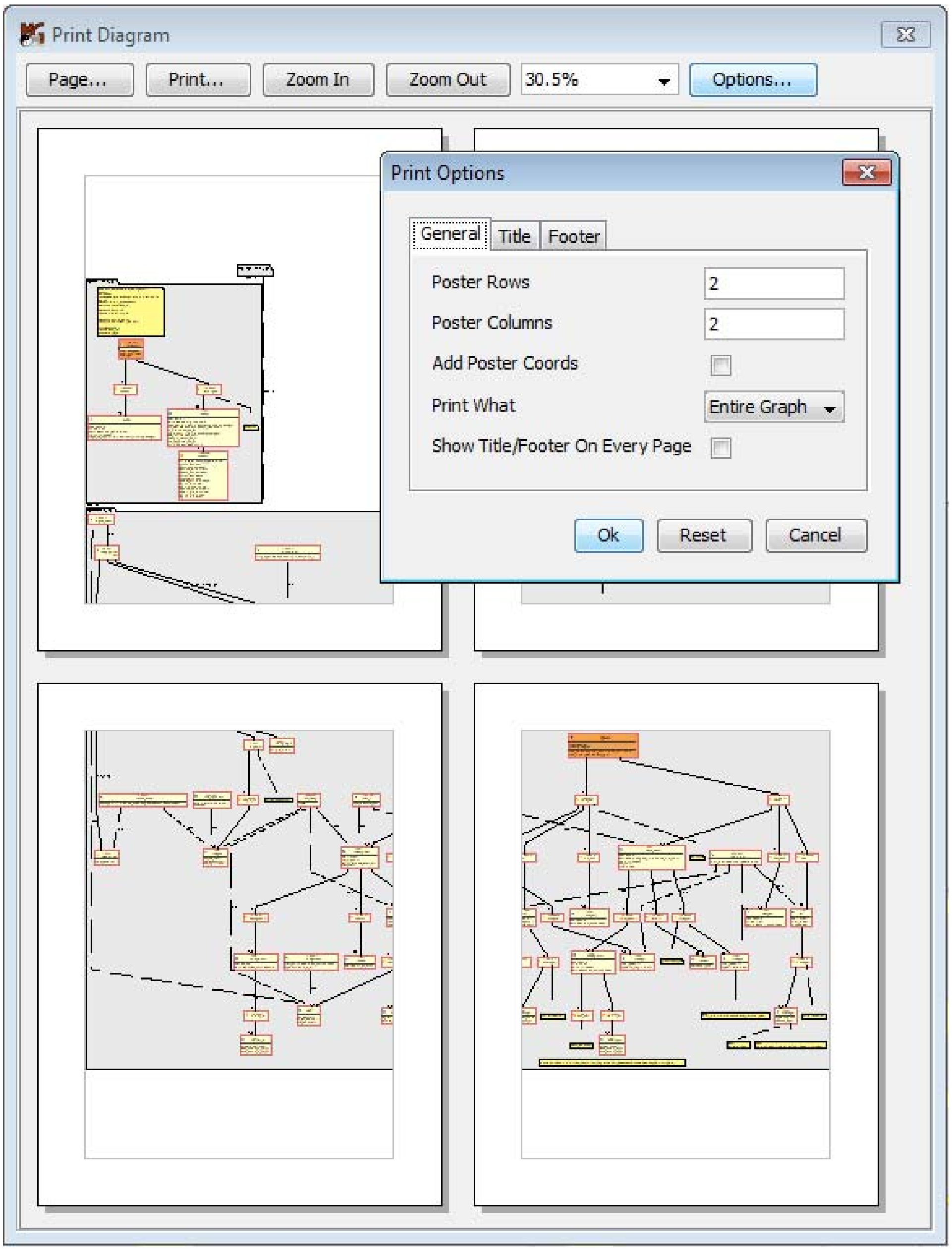


Figure 42: Viewing the relationships between classes of two modules in the UML class diagram

11. To print the UML class diagram to a printer, click the ***Print*** button (🖨) in the UML Class Diagram window toolbar.
12. The Print Diagram window appears displaying a print preview. If you would like to print the diagram as a poster spread across two or more sheets of paper or include a custom title or tooter into print, click the ***Options…*** button in the Print Diagram window and set the desired print options (Figure 43)



Poster Rows: 2
Poster Columns: 2
Add Poster Coords: □
Print What: Entire Graph
Show Title/Footer On Every Page: □

Figure 43: Setting the print options for printing UML class diagram as a 2x2 poster

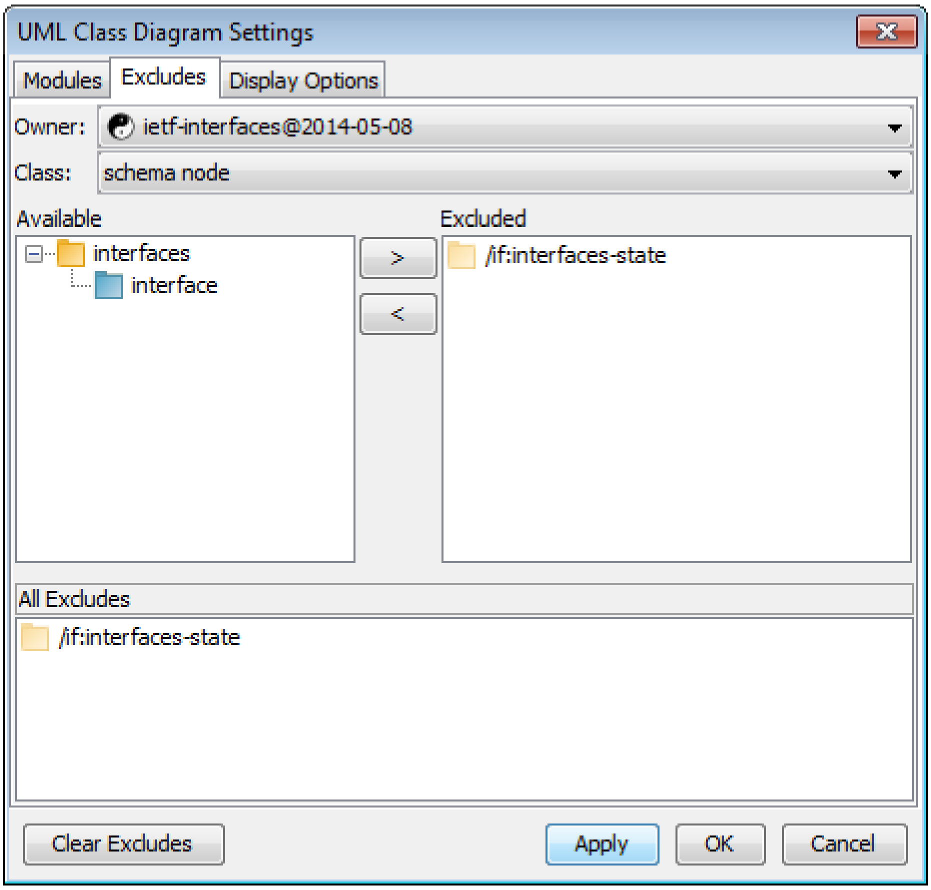
13. To save the UML class diagram to a bitmap image file (i.e., .jpg, .png or .gif format), click the ***Save As*** button (💾) in the UML Class Diagram window toolbar and in the standard Save As dialog box that appears, select the location, specify the name and the extension (i.e., .jpg, .png or .gif) of the image file to be created and click the ***Save Graph to Image*** button to write the image file to disk.

7.1 Filtering UML Class Diagram

By default, UML Class Diagram displays all classes of the selected YANG module(s). It also displays class annotations as UML comments and class descriptions in tooltips.

However, YANG Explorer lets you control which elements of the selected YANG module(s) will be shown in the UML class diagram and which not. This can be achieved by configuring the **excludes** (by excluding specific, user-selected classes) and by setting the **display options** (to enable/disable displaying certain types of classes globally (e.g., `typedef`, `leafref`, `identity`, etc.), select whether to display class packages, annotations, descriptions,…). Excludes and display options can be configured in the **UML Class Diagram Settings** dialog box, on the **Excludes** tab and **Display Options** tab.

1. To exclude certain elements from the UML class diagram, click the 🔧 toolbar button to open the **UML Class Diagram Settings** dialog box, switch to the **Excludes** tab and select the types of nodes and their instances you want to exclude from the diagram (Figure 44).



Owner: ietf-interfaces@2014-05-08

Class: schema node

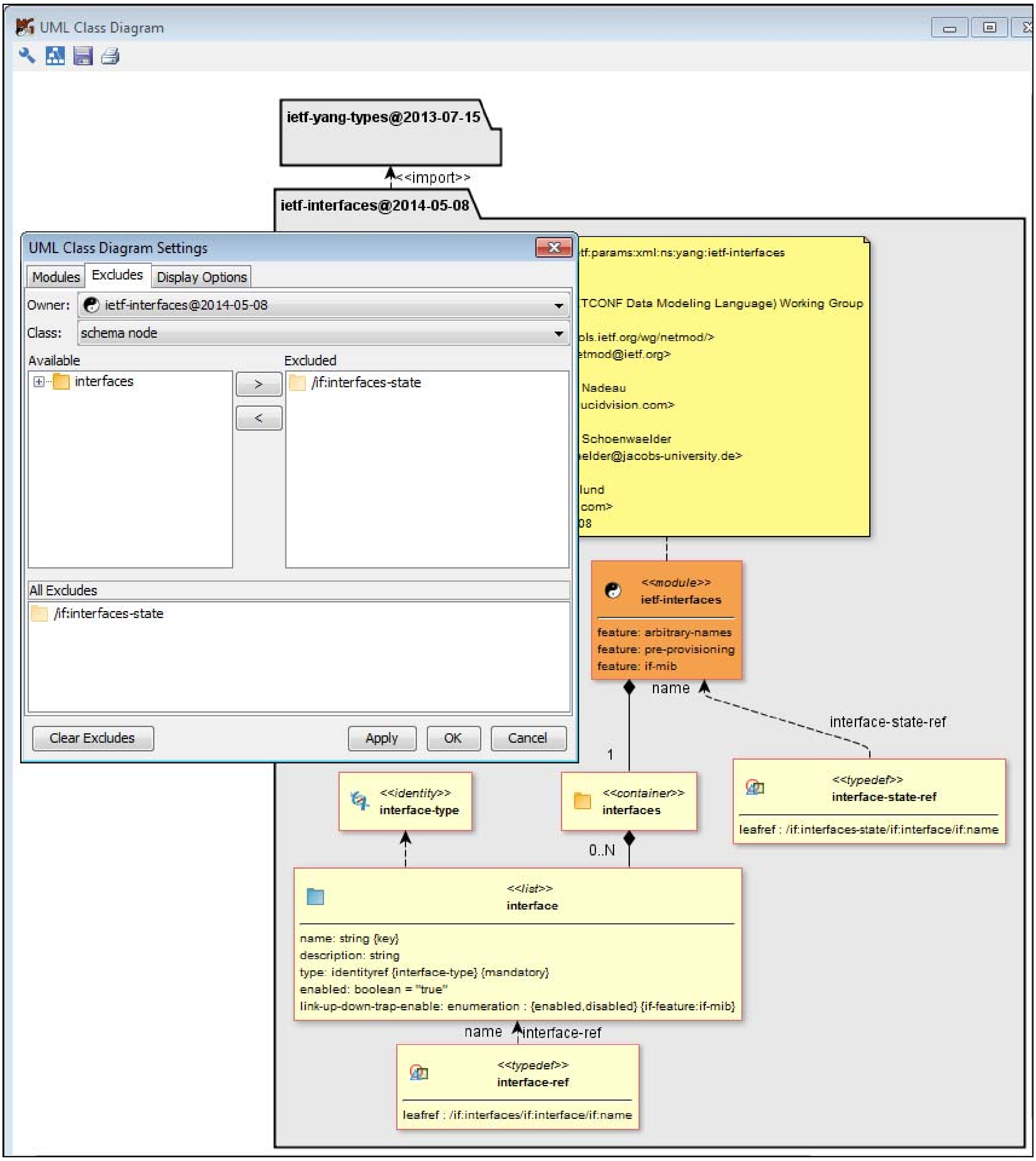
Excluded: /if:interfaces-state

All Excludes: /if:interfaces-state

Figure 44: Excluding elements (e.g., schema node classes) from UML class diagram

Example: The `ietf-interfaces` YANG module contains the `interfaces` and `interfaces-state` data subtrees. If we want to focus only on the configuration portion of the data tree (i.e. `interfaces` subtree), we can exclude the state data portion of the module. To do that, select the **`ietf-interfaces@*`** module from the

Owner drop-down list in the **Excludes** tab of the **UML Class Diagram Settings** dialog box. Then, select the ***schema node*** entry from ***Class*** drop-down list to display the top-level schema nodes that are represented as classes (i.e., `interfaces` and `interfaces-state`) in the ***Available*** list on the left hand side. Select the **`interfaces-state`** node in the ***Available*** list and click the right-arrow button (>) to move it to the ***Excluded*** list in the right portion of the dialog box and click the ***Apply*** button or the ***OK*** button to apply the changes and close the dialog box (Figure 45).



UML Class Diagram Settings

Owner: ietf-interfaces@2014-05-08

Class: schema node

Available: interfaces

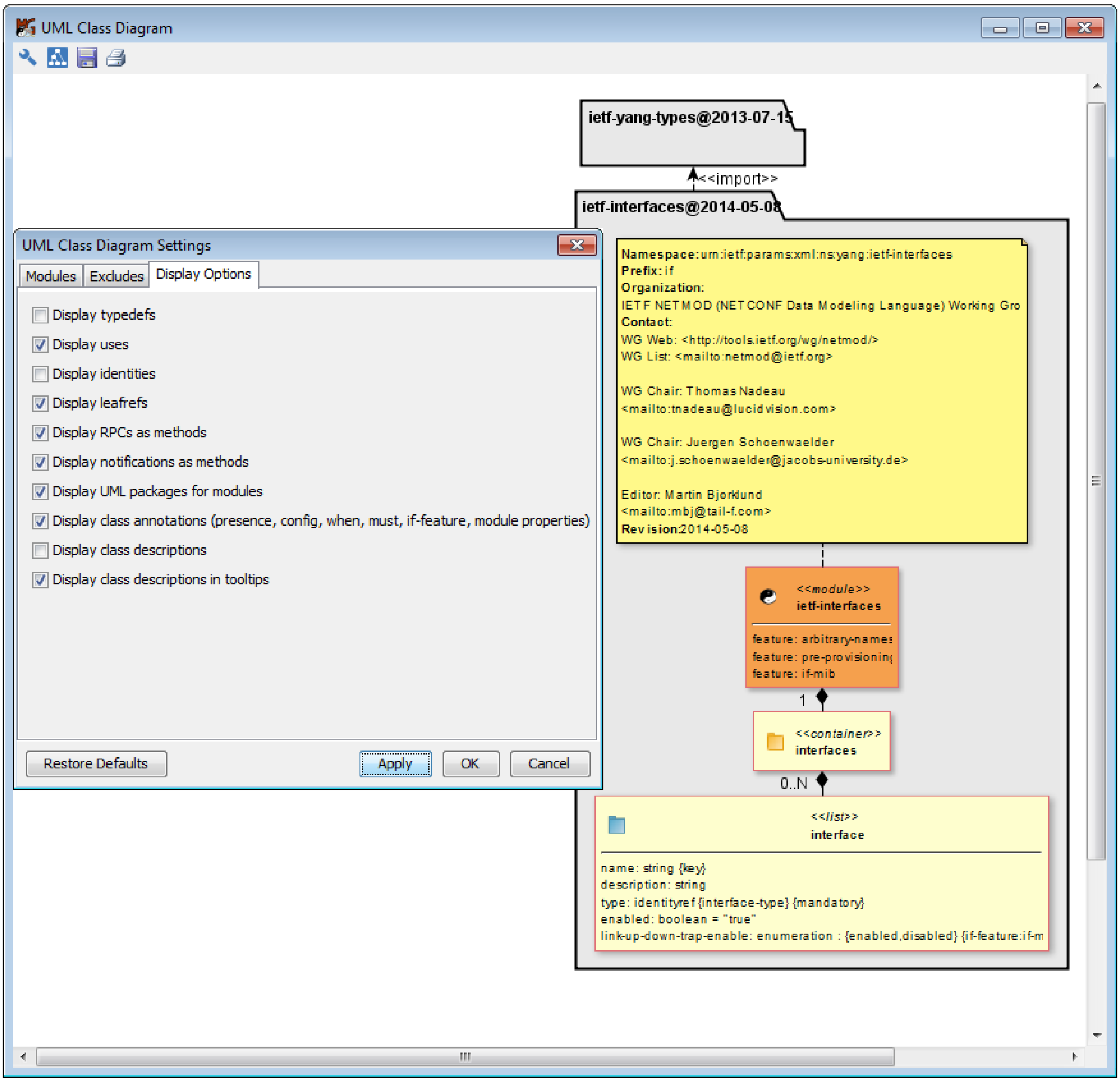
Excluded: /if:interfaces-state

All Excludes: /if:interfaces-state

Figure 45: Example of UML class diagram (*ietf-interfaces)* without the state data classes

2. To limit the elements of the YANG module(s) that will be displayed in the UML class diagram, click the 🔧 toolbar button to open the **UML Class Diagram Settings** dialog box, switch to the **Display Options** tab.

   Example: The `ietf-interfaces` YANG module contains certain `typedef` statements and an `identity` statement (Figure 45). If we want to view only the configuration data tree, we can disable displaying the `typedef` and `identity` classes. To do that, uncheck the ***Display typedefs*** and ***Display identities*** checkboxes in the **Display Options** tab of the **UML Class Diagram Settings** dialog box (Figure 46).



UML Class Diagram Settings

Modules | Excludes | Display Options

☐ Display typedefs
☑ Display uses
☐ Display identities
☑ Display leafrefs
☑ Display RPCs as methods
☑ Display notifications as methods
☑ Display UML packages for modules
☑ Display class annotations (presence, config, when, must, if-feature, module properties)
☐ Display class descriptions
☑ Display class descriptions in tooltips

Namespace: urn:ietf:params:xml:ns:yang:ietf-interfaces
Prefix: if
Organization:
IETF NETMOD (NETCONF Data Modeling Language) Working Gro
Contact:
WG Web: <http://tools.ietf.org/wg/netmod/>
WG List: <mailto:netmod@ietf.org>

WG Chair: Thomas Nadeau
<mailto:tnadeau@lucidvision.com>

WG Chair: Juergen Schoenwaelder
<mailto:j.schoenwaelder@jacobs-university.de>

Editor: Martin Bjorklund
<mailto:mbj@tail-f.com>
Revision:2014-05-08

name: string {key}
description: string
type: identityref {interface-type} {mandatory}
enabled: boolean = "true"
link-up-down-trap-enable: enumeration : {enabled,disabled} {if-feature:if-m

Figure 46: UML class diagram (*ietf-interfaces)* without the state data, *typedef* and *identitity* classes

3. Click the ***Apply*** button in the **UML Class Diagram Settings** dialog box to apply the changes without closing the dialog box or the ***OK*** button to apply the changes and close the dialog box.

7.2 Understanding the YANG Explorer UML Class Diagram

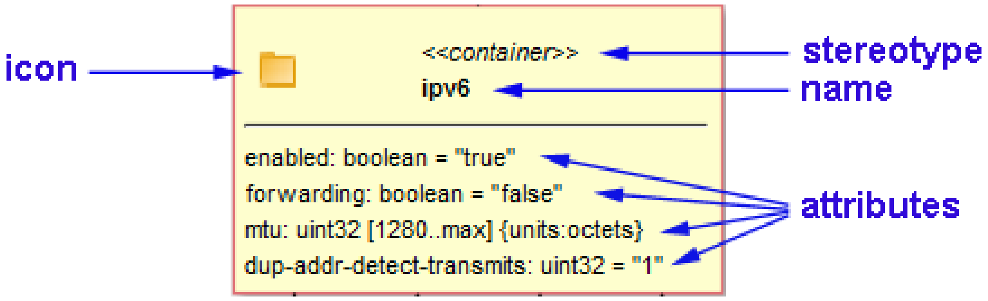
YANG Explorer can display loaded modules in form of a UML Class diagram (http://www.omg.org/spec/UML/2.4.1), a structure diagram which shows the structure of the designed system at the level of classes and shows their features, constraints and relationships.

7.2.1 Classes

The following YANG statements are represented as classes in the UML class diagram: `module`, `submodule`, `identity`, `typedef`, `grouping`, `container`, `list`, `choice`, `case`, `notification`, `rpc`, `input`, `output`, `augment`, and `deviation`.

Other statements are either omitted for brevity, appear as attributes or methods of a class, or are represented as annotations.

Each class has a name, an icon and a stereotype included in << >>, identifying the statement it represents. Many classes have attributes (e.g., container with leaf child nodes, etc.) or methods (notification), which, if present, are displayed below the separator line in the class box (Figure 47).


<<container>>
ipv6

enabled: boolean = "true"
forwarding: boolean = "false"
mtu: uint32 [1280..max] {units:octets}
dup-addr-detect-transmits: uint32 = "1"

Figure 47: Example of a *container* class

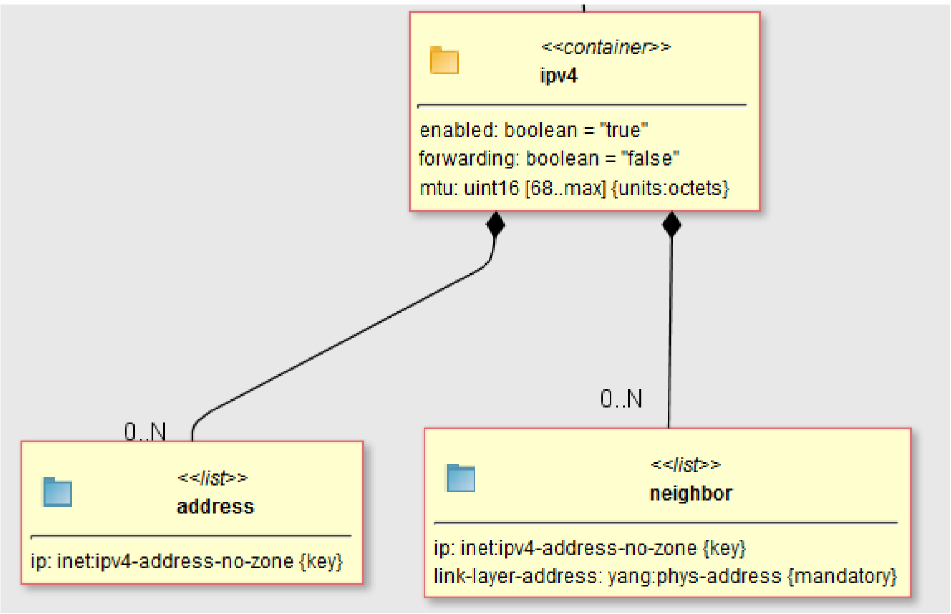
Note: In YANG Explorer you can select which types of classes will be displayed in the UML diagram, as well as whether to display the packages, annotation and descriptions. This can be configured in the **UML Class Diagram Settings** dialog box, **Misc.** tab.

7.2.2 Relationships

A relationship between two classes is represented with a directed line and can be one of the following:

- ———◆ **composition** (is part of),
- - - - - - - -➤ **dependency** (depends on),
- ———▷ **generalization** (inherits from),
- ———➤ **navigable association** (has alternative).

Composition represents a tight coupling relationship between classes and is read as **"is a part of"** in the direction of the arrow. An example composition relationship is one connecting a top-level `container` class and the class representing a `module`.



<<container>>
ipv4
enabled: boolean = "true"
forwarding: boolean = "false"
mtu: uint16 [68..max] {units:octets}
0..N
0..N
<<list>>
address
ip: inet:ipv4-address-no-zone {key}
<<list>>
neighbor
ip: inet:ipv4-address-no-zone {key}
link-layer-address: yang:phys-address {mandatory}

Figure 48: Example of a *composition (is part of) relationship* between UML classes

In the example above (Figure 48), the lower `address` and `neighbor` list classes have a composition relationship with the `ipv4` container class (e.g., `address` class *is a part of* `ipv4` class, `neighbor` class *is a part of* `ipv4` class).

A **dependency** represents relationship between two classes which means that one class requires another for its complete definition. It is read as **"depends on"** in the direction of the arrow. An example of this would be a relationship between a `list` class and a `typedef` class, if the `list` has a child that uses the `typedef` as its type.

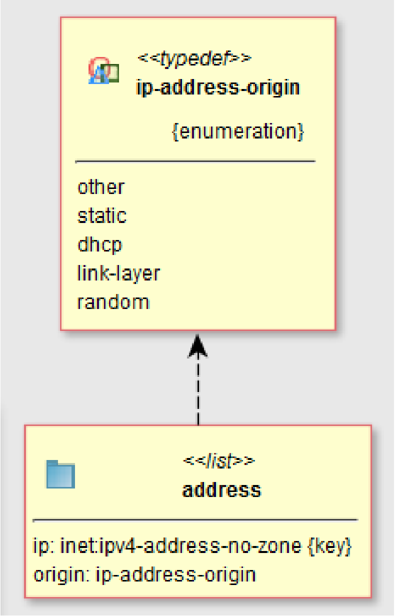


Figure 49: Example of a *dependency (depends on) relationship* between UML classes

Generalization is read as **"inherits from"** in the direction of the arrow and is only used to represent inheritance between classes that represent `identities` or `typedefs`.

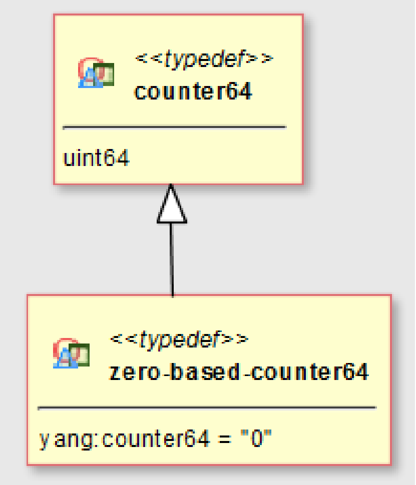
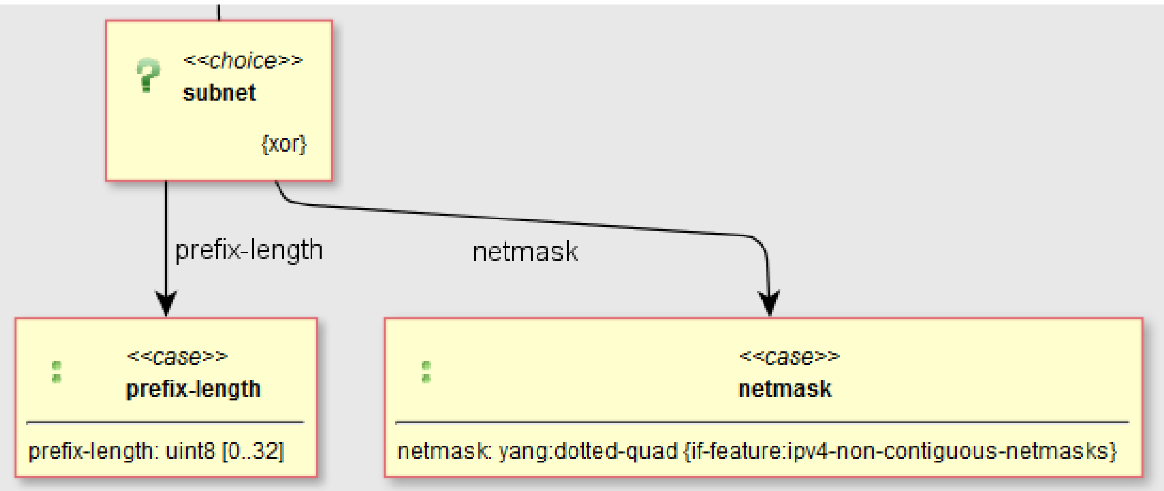


Figure 50: Example of a *generalization (inherits from) relationship* between UML classes

Classes representing `choice` statements are connected to their child `case` classes using a navigable **association** relationship (i.e., **"has alternative"**), since composition does not express the relationship properly. It means that the class instances are linked in some semantic sense.



<<choice>>
subnet
{xor}
prefix-length
netmask
<<case>>
prefix-length
prefix-length: uint8 [0..32]
<<case>>
netmask
netmask: yang:dotted-quad {if-feature:ipv4-non-contiguous-netmasks}

Figure 51: Example of a navigable *association relationship* between UML classes

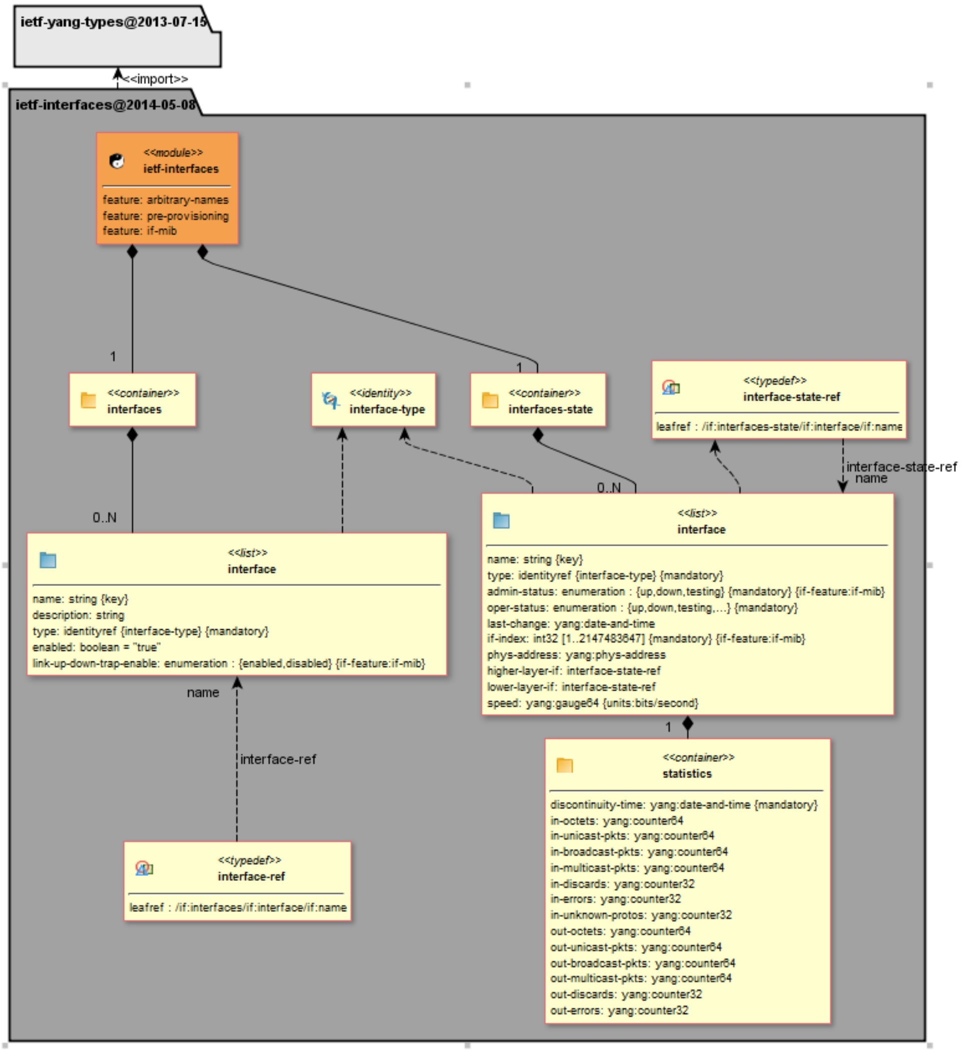
Some composition relationships express **cardinality** of the classes in the relationship using text values, such as '0..N' (zero or many), or by a single text value, such as '1' (exactly one). An example of displayed cardinalities '0..N' can be seen in Figure 48.

Some dependency relationships may be **labeled**, for example, `leafrefs` will have the target node name at the end of the directed connection and the name of the `leafref` at the center of it.

7.2.3 Packages

Classes may be grouped into a package, which represents a single namespace. Therefore, a package is always named after the module/submodule it was generated from.

Two packages may also be in a relationship, which is either a **package import** (represented with a dashed line with normal arrow head and an **`<<import>>`** stereotype) or **package composition** (line with a circle containing a plus sign head). The former is mapped from **import** statements and the latter from **include** statements.

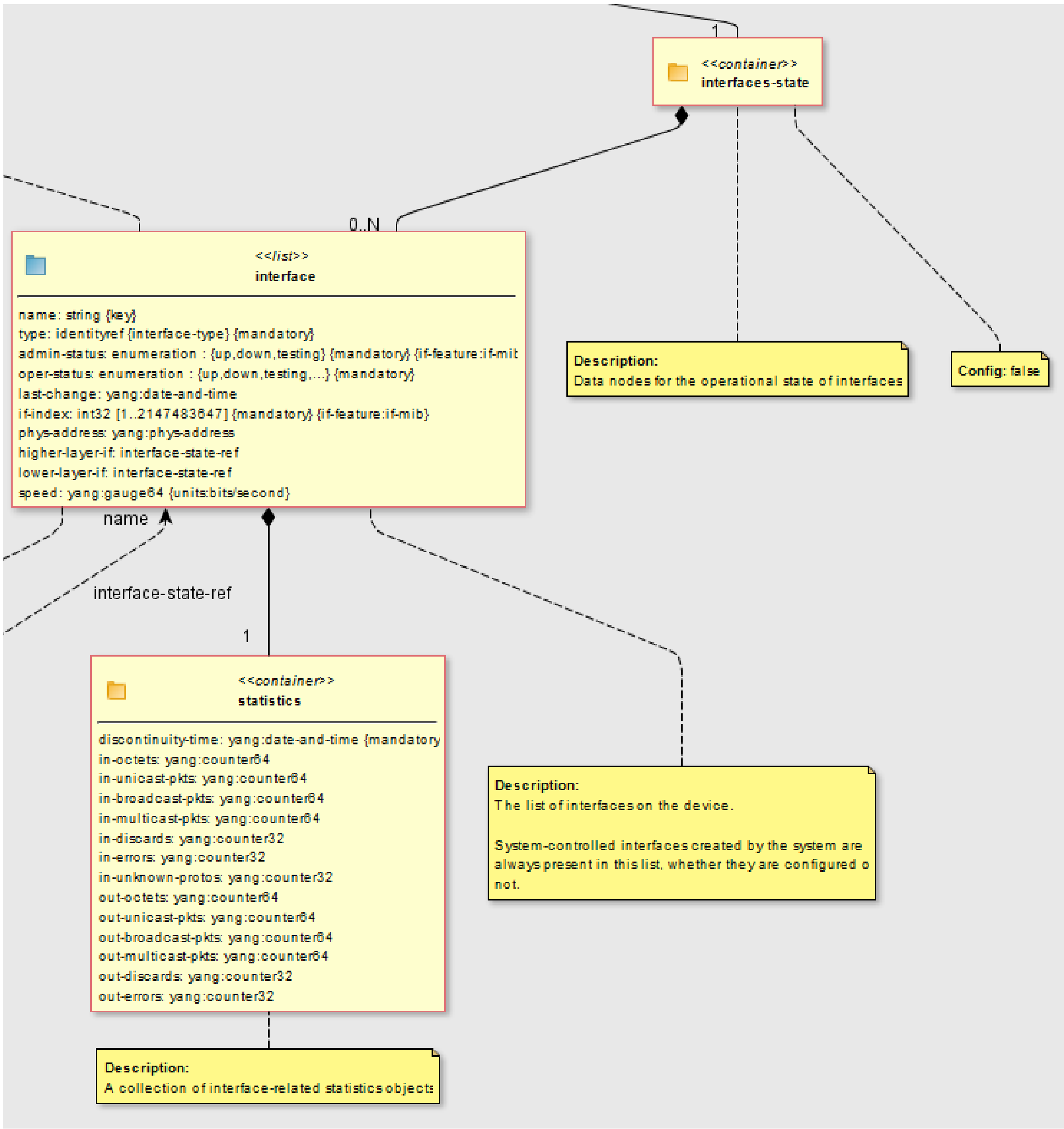


ietf-yang-types@2013-07-15
<<import>>
ietf-interfaces@2014-05-08
<<module>>
ietf-interfaces
feature: arbitrary-names
feature: pre-provisioning
feature: if-mib
1
1
<<container>>
interfaces
<<identity>>
interface-type
<<container>>
interfaces-state
<<typedef>>
interface-state-ref
leafref : /if:interfaces-state/if:interface/if:name
interface-state-ref
name
0..N
0..N
<<list>>
interface
name: string {key}
description: string
type: identityref {interface-type} {mandatory}
enabled: boolean = "true"
link-up-down-trap-enable: enumeration : {enabled,disabled} {if-feature:if-mib}
<<list>>
interface
name: string {key}
type: identityref {interface-type} {mandatory}
admin-status: enumeration : {up,down,testing} {mandatory} {if-feature:if-mib}
oper-status: enumeration : {up,down,testing,...} {mandatory}
last-change: yang:date-and-time
if-index: int32 [1..2147483647] {mandatory} {if-feature:if-mib}
phys-address: yang:phys-address
higher-layer-if: interface-state-ref
lower-layer-if: interface-state-ref
speed: yang:gauge64 {units:bits/second}
name
interface-ref
1
<<container>>
statistics
discontinuity-time: yang:date-and-time {mandatory}
in-octets: yang:counter64
in-unicast-pkts: yang:counter64
in-broadcast-pkts: yang:counter64
in-multicast-pkts: yang:counter64
in-discards: yang:counter32
in-errors: yang:counter32
in-unknown-protos: yang:counter32
out-octets: yang:counter64
out-unicast-pkts: yang:counter64
out-broadcast-pkts: yang:counter64
out-multicast-pkts: yang:counter64
out-discards: yang:counter32
out-errors: yang:counter32
<<typedef>>
interface-ref
leafref : /if:interfaces/if:interface/if:name

Figure 52: Example of a *package* containing all classes of a module

7.2.4 Annotations

Besides classes, the diagram may also contain annotations in form of a note (an UML comment). An annotation is associated with a class via a dashed line without arrows. An annotation may contain a description of the class or a YANG property, such as `if-features`, `presence`, `config`, and `when/must` constraints.


1
<<container>>
interfaces-state
0..N
<<list>>
interface
name: string {key}
type: identityref {interface-type} {mandatory}
admin-status: enumeration : {up,down,testing} {mandatory} {if-feature:if-mit
oper-status: enumeration : {up,down,testing,...} {mandatory}
last-change: yang:date-and-time
if-index: int32 [1..2147483647] {mandatory} {if-feature:if-mib}
phys-address: yang:phys-address
higher-layer-if: interface-state-ref
lower-layer-if: interface-state-ref
speed: yang:gauge64 {units:bits/second}
Config: false
name
interface-state-ref
1
<<container>>
statistics
discontinuity-time: yang:date-and-time {mandatory
in-octets: yang:counter64
in-unicast-pkts: yang:counter64
in-broadcast-pkts: yang:counter64
in-multicast-pkts: yang:counter64
in-discards: yang:counter32
in-errors: yang:counter32
in-unknown-protos: yang:counter32
out-octets: yang:counter64
out-unicast-pkts: yang:counter64
out-broadcast-pkts: yang:counter64
out-multicast-pkts: yang:counter64
out-discards: yang:counter32
out-errors: yang:counter32

Description:
Data nodes for the operational state of interfaces

Description:
The list of interfaces on the device.

System-controlled interfaces created by the system are always present in this list, whether they are configured o not.

Description:
A collection of interface-related statistics objects

Figure 53: Viewing annotations and descriptions (UML comments shown in dark yellow notes)

8 USING NETCONF CONTENT EDITOR

MG-SOFT YANG Explorer includes an XML editor and validator for NETCONF content that complies with the RFC 6110 specification. This feature allows you to edit or validate new and existing content according to Document Schema Definition Languages (DSDL), which are automatically generated from selected YANG modules in the background. Edited and validated NETCONF documents can be saved to XML files, which can be loaded into MG-SOFT NetConf Browser, for example, to send documents to the connected NETCONF server.

1. To start editing, load the desired YANG or YIN modules into YANG Explorer and select the ***Edit / Edit NETCONF Content*** command. This will open the NETCONF Content Editor window (Figure 54), which contains the following components:

   1) Menu bar

   2) Toolbar

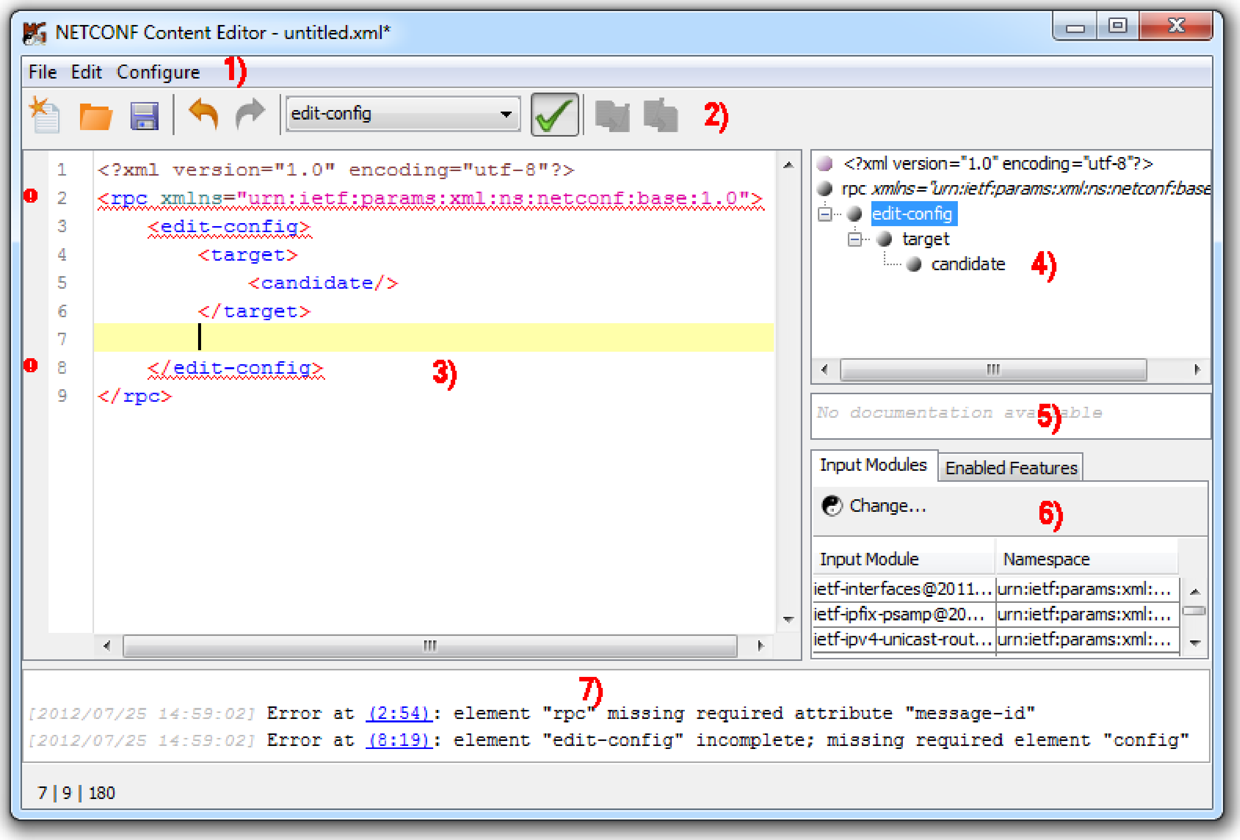
   3) NETCONF content XML editor panel – XML document editor with syntax coloring and autocomplete feature

   4) NETCONF content Tree editor panel - represents the currently edited XML document in a tree form and lets you edit the tree

   5) Informational text area containing description of the currently selected tree node (if the corresponding YANG module provides it)

   6) Configuration tabs, which let you refine the current schema (YANG modules, features)

   7) Notification area displaying the validation errors



```
<?xml version="1.0" encoding="utf-8"?>
<rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
    <edit-config>
        <target>
            <candidate/>
        </target>

    </edit-config>
</rpc>
```

<?xml version="1.0" encoding="utf-8"?>
- rpc *xmlns="urn:ietf:params:xml:ns:netconf:base*
  - edit-config
    - target
      - candidate

No documentation available

Input Module	Namespace
ietf-interfaces@2011...	urn:ietf:params:xml:...
ietf-ipfix-psamp@20...	urn:ietf:params:xml:...
ietf-ipv4-unicast-rout...	urn:ietf:params:xml:...

```
[2012/07/25 14:59:02] Error at (2:54): element "rpc" missing required attribute "message-id"
[2012/07/25 14:59:02] Error at (8:19): element "edit-config" incomplete; missing required element "config"
```

7 | 9 | 180

Figure 54: NETCONF Content Editor window

2. After opening the NETCONF Content Editor window, select the type of content you wish to edit or validate by selecting it from the ***Content type*** drop-down list in the toolbar. You can choose between these content types:
   - **`data`**: Use this content type to edit/validate entire datastores – XML instance files, which contain both state and configuration data,
   - **`config`**: Use this content type when you wish to edit/validate configuration data only,
   - **`get`**: Use his content type when you wish to compose an XML document for a NETCONF get request. This content type lets you add get operation specific elements in addition to being able to edit a configuration. For example, it will let you specify a NETCONF filter element.
   - **`get-config`**: Use his content type when you wish to compose an XML document for a NETCONF get-config request. This content type lets you add get-config specific elements in addition to being able to edit a configuration. For example, it will let you specify a NETCONF filter element.
   - **`edit-config`**: Use his content type when you wish to compose an XML document for a NETCONF edit-config request. This content type differs from “config” in that it lets you add edit-config specific elements and attributes in addition to being able to edit a configuration. For example, it will let you specify a NETCONF operation attribute for each element of the config subtree.
   - **`rpc`**: Allows you to create RPC operation requests based on RPC definitions available in the input modules and also lets you send a valid document as an RPC request, the same way this is possible with the `get`, `get-config` and `edit-config` content type,
   - **`notification`**: Lets you validate or create examples of notifications defined in your input modules,
   - **`get-reply`**: Enables you to validate or create examples of possible rpc-reply messages defined by your input modules, which would be created on server with a NETCONF get operation,
   - **`get-config-reply`**: Enables you to validate or create examples of possible rpc-reply messages defined by your input modules, which would be created on server with a NETCONF get-config operation,
   - **`rpc-reply`**: Enables you to validate or create examples of possible rpc-reply messages defined by your input modules, which would be created on server with any NETCONF RPC operation request.
3. Once you have chosen a content type, you can refine the schema behind it via the Configuration tabs. There are currently two ways to refine the schema:
   - by selecting **input modules** for the schema generation algorithm among all loaded YANG/YIN modules, and
   - by selecting enabled **features**, defined by the input modules.

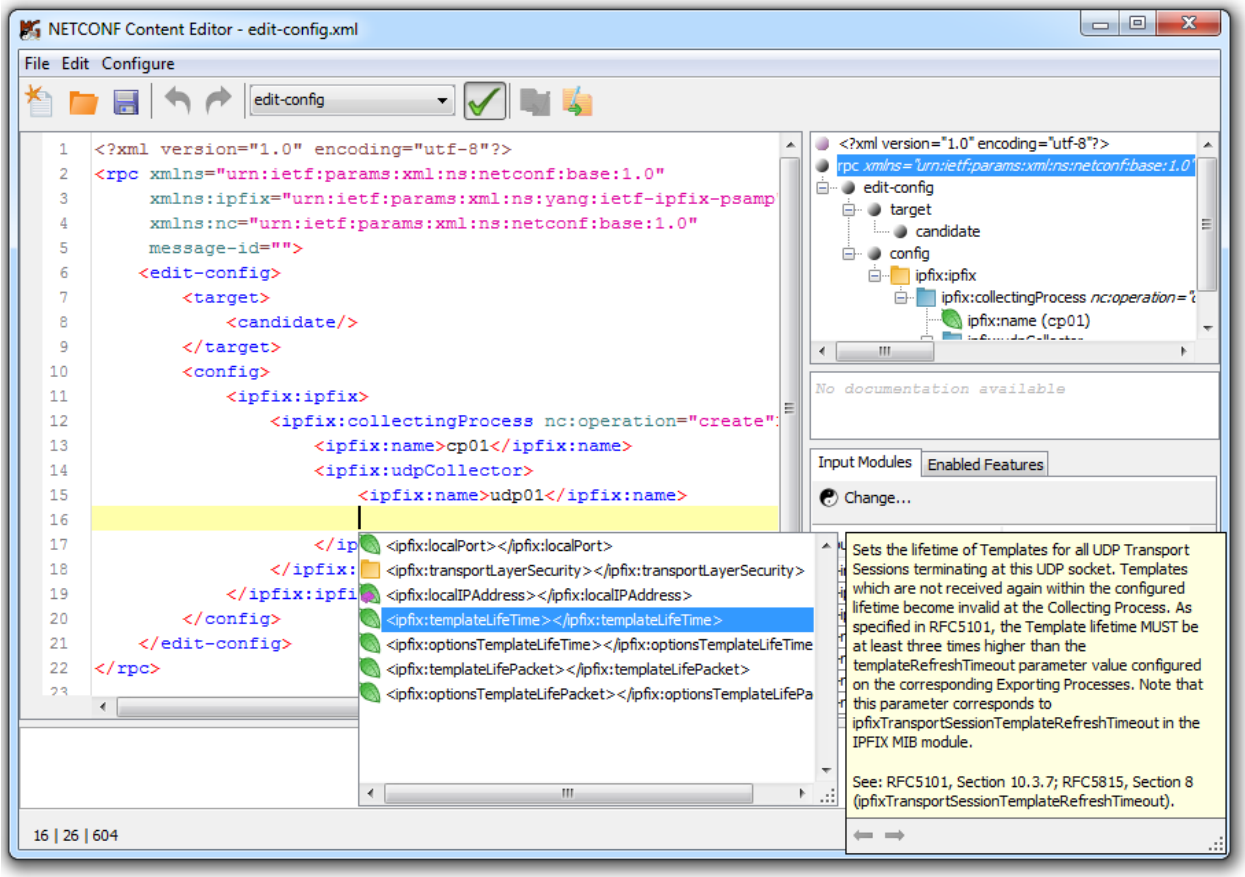
   To include or exclude a YANG/YIN module, select the ***Input Modules*** tab and click the ***Change***... button below it to open a dialog which will let you select the input modules. Similarly, you can enable or disable features in the ***Enabled Features*** tab (provided that the selected input modules define features).

Each time you make a change to the settings above, the schema that is used to validate your document may have to be re-generated. A progress bar dialog will appear each time this occurs, temporarily preventing you from editing the document. The main purpose of the schema is to enable document validation. NETCONF Content Editor, also uses it for the autocomplete feature that is available when writing XML documents.

Note: Actually, three different schemas are automatically generated in the background – a Relax NG schema, a DSRL schema (Document Schema Renaming Language schema) and an ISO Schematron schema – but for simplicity reasons we refer to them as if it were a single schema. (the schema files are generated in the following folder: $USER_HOME/`.mgyangexplorer/schemas`).

4. Edit your document in the NETCONF content editor. When you start writing an XML tag, the editor will assist you with the **autocomplete** feature, which displays a list of all possible elements defined by the schema. The autocomplete drop-down list of choices appears in the NETCONF content editor panel when you type in the "<" character or when you press the ***CTRL+Space*** keyboard keys (when the cursor is placed where completions are possible). Select an item in the autocomplete drop-down list to view its description (from YANG module) in a tooltip next to the selected item (Figure 55). Press the ***Enter*** key to insert the selected item into the NETCONF content editor panel.

Note: Autocomplete is provided for XML elements, attributes and their values. Note that completions for attribute and element values will only be provided if a set of possible values can be determined for the current attribute/element – for example this is possible if the element is specified by a YANG leaf statement of type enumeration, bits or similar. The autocomplete feature is XML namespace aware.



```
<?xml version="1.0" encoding="utf-8"?>
<rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
     xmlns:ipfix="urn:ietf:params:xml:ns:yang:ietf-ipfix-psamp
     xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0"
     message-id="">
    <edit-config>
        <target>
            <candidate/>
        </target>
        <config>
            <ipfix:ipfix>
                <ipfix:collectingProcess nc:operation="create"
                    <ipfix:name>cp01</ipfix:name>
                    <ipfix:udpCollector>
                        <ipfix:name>udp01</ipfix:name>

                    </ip
                </ipfix:
            </ipfix:ipfi
        </config>
    </edit-config>
</rpc>
```

```
<ipfix:localPort></ipfix:localPort>
<ipfix:transportLayerSecurity></ipfix:transportLayerSecurity>
<ipfix:localIPAddress></ipfix:localIPAddress>
<ipfix:templateLifeTime></ipfix:templateLifeTime>
<ipfix:optionsTemplateLifeTime></ipfix:optionsTemplateLifeTime
<ipfix:templateLifePacket></ipfix:templateLifePacket>
<ipfix:optionsTemplateLifePacket></ipfix:optionsTemplateLifePa
```

```
<?xml version="1.0" encoding="utf-8"?>
rpc xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"
  edit-config
    target
      candidate
    config
      ipfix:ipfix
        ipfix:collectingProcess nc:operation="
          ipfix:name (cp01)
```

No documentation available

Sets the lifetime of Templates for all UDP Transport Sessions terminating at this UDP socket. Templates which are not received again within the configured lifetime become invalid at the Collecting Process. As specified in RFC5101, the Template lifetime MUST be at least three times higher than the templateRefreshTimeout parameter value configured on the corresponding Exporting Processes. Note that this parameter corresponds to ipfixTransportSessionTemplateRefreshTimeout in the IPFIX MIB module.

See: RFC5101, Section 10.3.7; RFC5815, Section 8 (ipfixTransportSessionTemplateRefreshTimeout).

Figure 55: Using the autocomplete feature in the NETCONF Content Editor window

5. As you edit the XML document, its tree representation in the NETCONF content tree panel on the right side changes accordingly. The NETCONF content tree panel represents your document's structure and may be used to quickly navigate large documents by double-clicking the tree nodes. It also gives you some information on individual nodes – node type in YANG terms and its YANG description if available.

6. As you edit the document, the validation error/warning messages appear in the notification area at the bottom of the window (Figure 54). You can locate the source of most errors by clicking links in the messages that appear in the notification area. The editor will also indicate error sources by underlining erroneous elements and by showing error icons in front of the relevant lines in the content editor panel. If you hover your mouse over an underlined element, a tooltip with the error message appears.

   The editor automatically validates the document as you edit it. The validation includes several steps:

   - Ensures that the document is well-formed (XML syntax compliant),
   - Checks if the document is valid according to the current RelaxNG schema (element/attribute structure must be as defined by input YANG modules),
   - Creates a copy of the current document in the background and fills in the missing default values using DSRL (prerequisite for the final step),
   - Ensures that the copy of the document satisfies all semantic constraints specified by the input modules, such as XPath expressions from the YANG *when* statements (ISO Schematron).

   You can disable the validation at any time by clicking the ***Validation Enabled*** (✓) toggle button in the toolbar.

7. Depending on the currently selected content type you may use additional features of the editor, for example:

   - When the `config` or `edit-config` content type is selected, you can use the corresponding toolbar buttons (/), to convert the document between the two types, i.e., between an edit-config operation with a <config> element and a config datastore part – note that this may result in loss of certain information (e.g., NETCONF operation attributes within an edit-config's <config> element will be discarded if the content is converted to the `config` type, etc.).
   - Besides the standard text editing features such as ***Find***, ***Replace*** and file operations (***Save***, ***Open***, ***New***), the editor also offers XML pretty-printing capabilities (***Edit / Format XML***), which will transform the entire document into a form that is easily readable.