

ADVANCED SOFTWARE ENGINEERING

HUSACCT

User Manual

VALIDATE

BRAM STAAL - NATHAN RHIJNSBURGER - TIMOTHÉ  
BEEMSTER

RICK VAN DER STAAIJ - RONALD VAN  
BROECKHUIJSEN

# CONTENTS

Introduction .....	2
Terminology .....	3
History points.....	3
Rule types and violation types .....	3
Exception rules.....	3
Violation .....	3
Severity .....	3
Supported rules.....	4
Browse Violations .....	6
Filter dialog .....	8
code viewer.....	11
Exporting a report.....	12
Configuration .....	13
Configure severities .....	13
Configure severities per rule type and per violation type .....	15
Configure active violation types .....	16
Selection of the default rules.....	17

# INTRODUCTION

The validate component of the HUSACCT application is responsible for validating the defined rules against the analyzed dependencies. This manual will describe the features that the users can use of the GUI that the validation part of the application provides for the user.

The preconditions before this part of the application can be used are:

- The source code is analyzed;
- Logical modules are defined;
- The classes or packages of the source code are mapped to the defined logical modules;
- The logical defined architecture must contain rules.

## TERMINOLOGY

The most used terminology will be explained in this paragraph.

### HISTORY POINTS

When HUSACCT periodically is used, there might be the need of an overview of violations in the past. One point of the past is a history point.

### RULE TYPES AND VIOLATION TYPES

A rule is of a specific type (rule type). A rule can be applied on one or more modules (depending on the rule type).

A violation type is the type of the violation that can occur during the validation process. It is possible to filter on violation types in the define process (see for more information the documentation of the define component).

### EXCEPTION RULES

Some rule types can have exception rules which are mostly counterparts of the rule type. For example the exception rule of rule type 'Is Not Allowed To Use' is 'Is Allowed To Use'.

### VIOLATION

When a rule is exceeded during validation process, a violation is occurred.

### SEVERITY

Each rule and violation type have their own severity. If a rule has a lower severity than a violation type (or vice versa), always the highest severity will be taken as the severity.

## SUPPORTED RULES

The following rules are provided in the HUSACCT applications and can be validated and which exception are possible on the specific/explained rule. Every rule has violation types, the possible violation types will be given for each rule.

### Property rule types

#### Interface convention

Each class in the specified module must implement the specified interface. An exception on this rule will exclude certain classes or packages from this rule.

#### Façade convention

Each class in the specified module must only communicate to classes outside the module via the façade.

#### SuperClassInheritance convention

Each class in the specified module must extend the specified class. An exception on this rule will exclude certain classes or packages from this rule.

#### Naming convention

All the classes and/or packages in the specified module must meet to the specified *Regular Expression* (see table 1). The *regular expressions* are case-sensitive. An exception on this rule will exclude certain classes or packages from this rule.

Regex	Validates	
<b>*Friends*</b>	domain.locationbased.foursquare.History	false
	domain.locationbased.latitude.Friends	true
	infrastructure.socialmedia.locationbased.foursquare.FriendsDAO	true
	infrastructure.socialmedia.locationbased.foursquare.MyFriendsDAO	true
<b>*Account</b>	domain.locationbased.foursquare.MyAccount	true
	domain.locationbased.latitude.Map	false
	infrastructure.socialmedia.locationbased.foursquare.AccountDAO	false
<b>DAO *</b>	infrastructure.socialmedia.locationbased.foursquare.DAOFourSquare	true
	infrastructure.socialmedia.locationbased.foursquare.IMap	false
	domain.locationbased.foursquare.History	false

TABLE 1 OVERVIEW OF POSSIBLE REGEXES FOR THE NAMING CONVENTION RULE TYPE

The first regex enforces that a class/package must contain the word 'Friends' in the name.

The second regex enforces that a class/package must contain the word 'Friends' anywhere in the name.

The third regex enforces that a class/package must end with the word 'Account'.

The fourth regex enforces that all the classes and/or packages must start with the word 'DAO'.

#### Visibility convention

All the classes and/or packages in the specified module must have the specified visibility or lower.

The possible exceptions are:

- a possibility to exclude classes/packages;
- add another visibility convention for classes/packages that were specified in the main rule.

## Relation rule types

### Is allowed to use

This rule can only be defined as exception rule. All the classes/packages between two logical modules are allowed to have a dependency. This is normally the case, so it can only be defined as an exception rule.

### Is not allowed to use

All the classes/packages between two logical modules are not allowed to have a dependency with each other.

The possible exception is to define a 'is allowed to use' rule.

### Is only allowed to use

One defined logical module is only allowed to have dependencies with another defined logical module. The possible exception is to define a 'is allowed to use' rule.

### Is only module allowed to use

Of all defined logical modules, only one logical module may have dependencies with another specified logical module.

There are no possible exception rules.

### Must use

At least one class/package between two logical modules must have a dependency with each other. The possible exception is to define a 'is allowed to use' or a 'is not allowed to use' rule.

### Is not allowed to make back-call

This rule can only be applied on modules of type layer (see for more information the manual of the define component). A layer may not have dependency with layers that are defined with a higher hierarchal level then the layer on which the rule is applied to.

The possible exception is to define a 'is allowed to use' rule.

### Is not allowed to make skip-call

This rule can only be applied on modules of type layer (see for more information the manual of the define component). A layer may not have dependency with layers that are defined with a lower hierarchal level then the layer on which the rule is applied to, except for the layer that is directly under the layer on which the rule is applied to.

The possible exception is to define a 'is allowed to use' rule.

## BROWSE VIOLATIONS

After defining the logical architecture, mapping scanned source code to the logical architecture and defining rules in the logical architecture the possibility to check if the defined architecture has violations is available.

This can be done in the *browse violations screen* (figure 1). The goal of this screen is to validate the project and to see if any violations have occurred. Additional features are:

- sorting;
- filtering;
- creating a history point.

The screen can be opened through “menu -> Validate -> Validate now” or by pressing the “validate now” icon (🚩) in the toolbar

**Note: When storing the workspace the violations won't be stored. The violation history points are experimental and unwanted behavior can occur.**

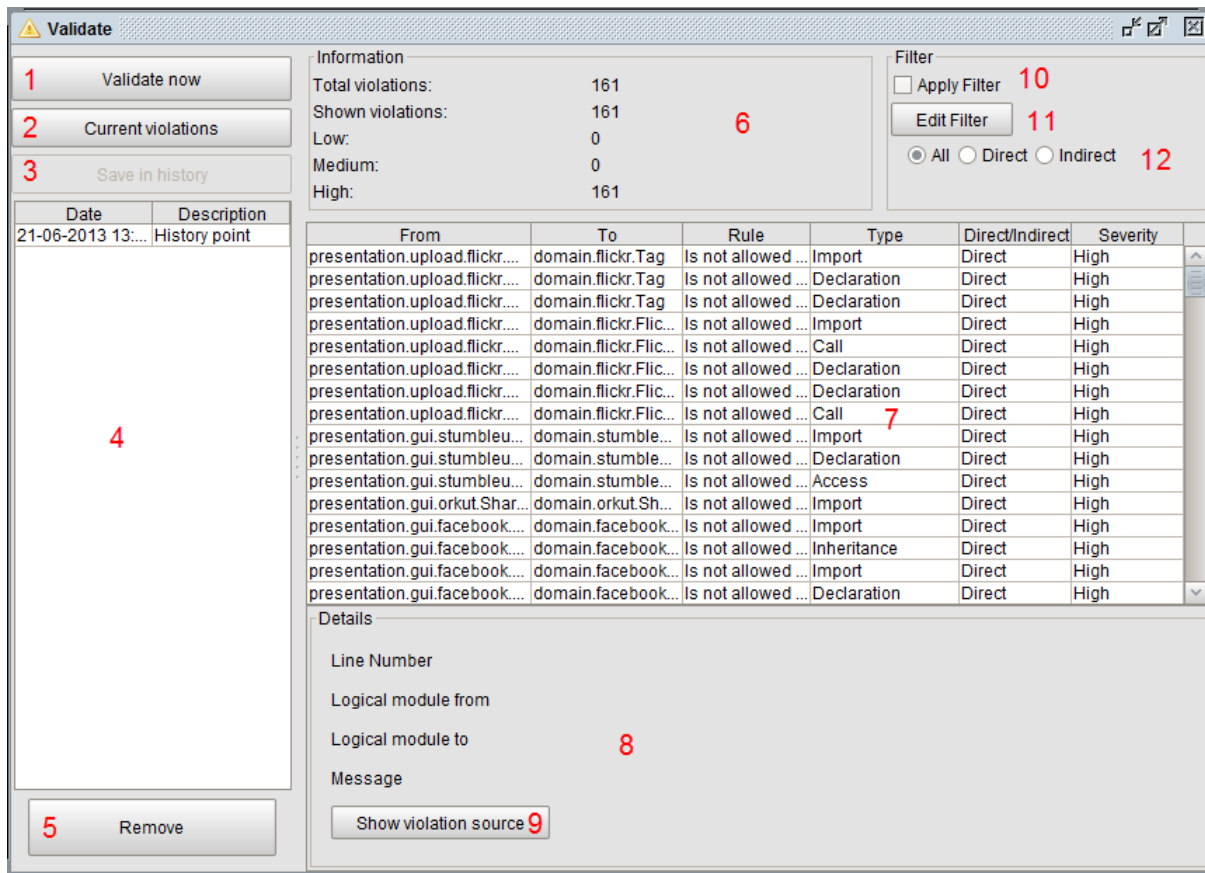


FIGURE 1 BROWSE VIOLATIONS SCREEN

Explanation of the components of figure 1:

1. When this button is pressed, the system will check if there are violations on the defined rules.
2. When this button is pressed, the last known violations will be shown to the user. The violations will be shown in area 7.

3. When this button is pressed, the system creates a history point of the last known violations. When the violations are saved the system will disable the button. For saving the violations you need to add a description of the violations.
4. This table contains all the created/available history points. After clicking on a row, the system will show the violations that are saved in that point. The violations will be shown in area 7.
5. When this button is pressed, the system will remove the selected history point in area 4.
6. This panel shows information about the amount of violations. It shows the total violations, the violations shown this moment and the amount of violations per severity.
7. This table shows all the violations. The violations can be sorted by clicking on the headers, e.g. 'From'. If you want more information about a violation select the row of the specific violation. The information will be shown in area 8.
  - From: the 'from' path from which the dependency/violation starts. Contains the physical path of the source file where the violations occurs;
  - To: the 'to' path where the dependency/violations ends. Contains the physical path of the source file which is responsible for the violation;
  - Rule: the name of the defined rule;
  - Type: the dependency type of the violation;
  - Direct/Indirect: the kind of dependency type of the violation. Direct or indirect;
  - Severity: the severity of the violation.
8. This panel shows information about the selected of the selection violation in area 7.
  - Line number: the line number in the source code where this violation occurs;
  - Logical module from: the name of the logical module where the physical class of the 'Source' column of area 7 is mapped into;
  - Logical module to: the name of the logical module where the physical class of the 'Target' column of area 7 is mapped into;
  - Message: a textual description of the defined rule.
9. With this button the code viewer can be used to see the exact piece of code which is responsible for the violation. (see 3.2)
10. With this checkbox the filter can be turned on or off.
11. When this button is pressed it will open the Filter dialog (see 3.1).
12. This is an on screen filter to switch between all kinds of dependencies, only direct dependencies or only indirect dependencies. These only work when the checkbox of area 9 is selected.



## FILTER DIALOG

The violations table can also be filtered. This screen provides an easy to use way to filter the violations. During filtering can be chosen to show or hide the selected values. There are two tabs in this screen. In the first tab only the rule types and violation types are shown. In the second tab there is a possibility to filter on the source path of the violations.

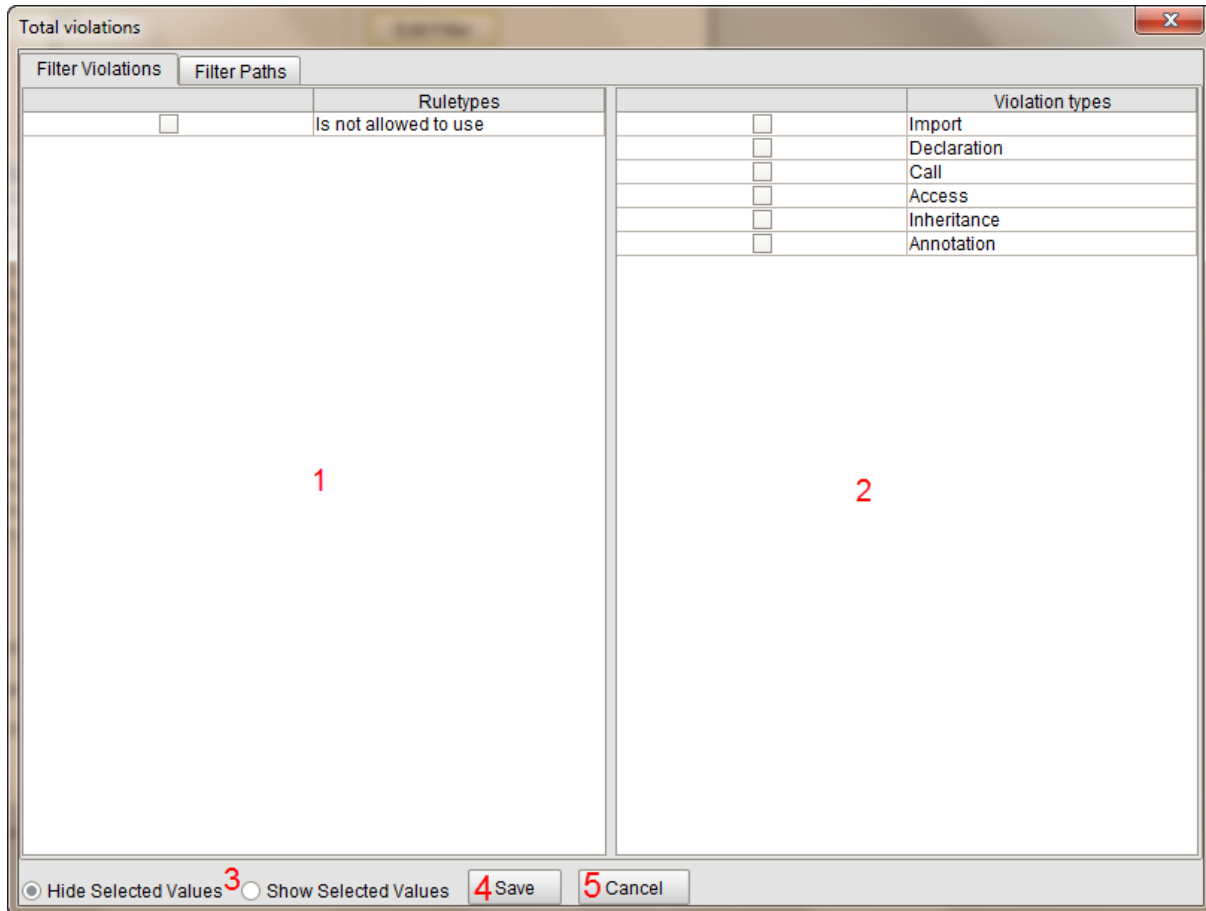
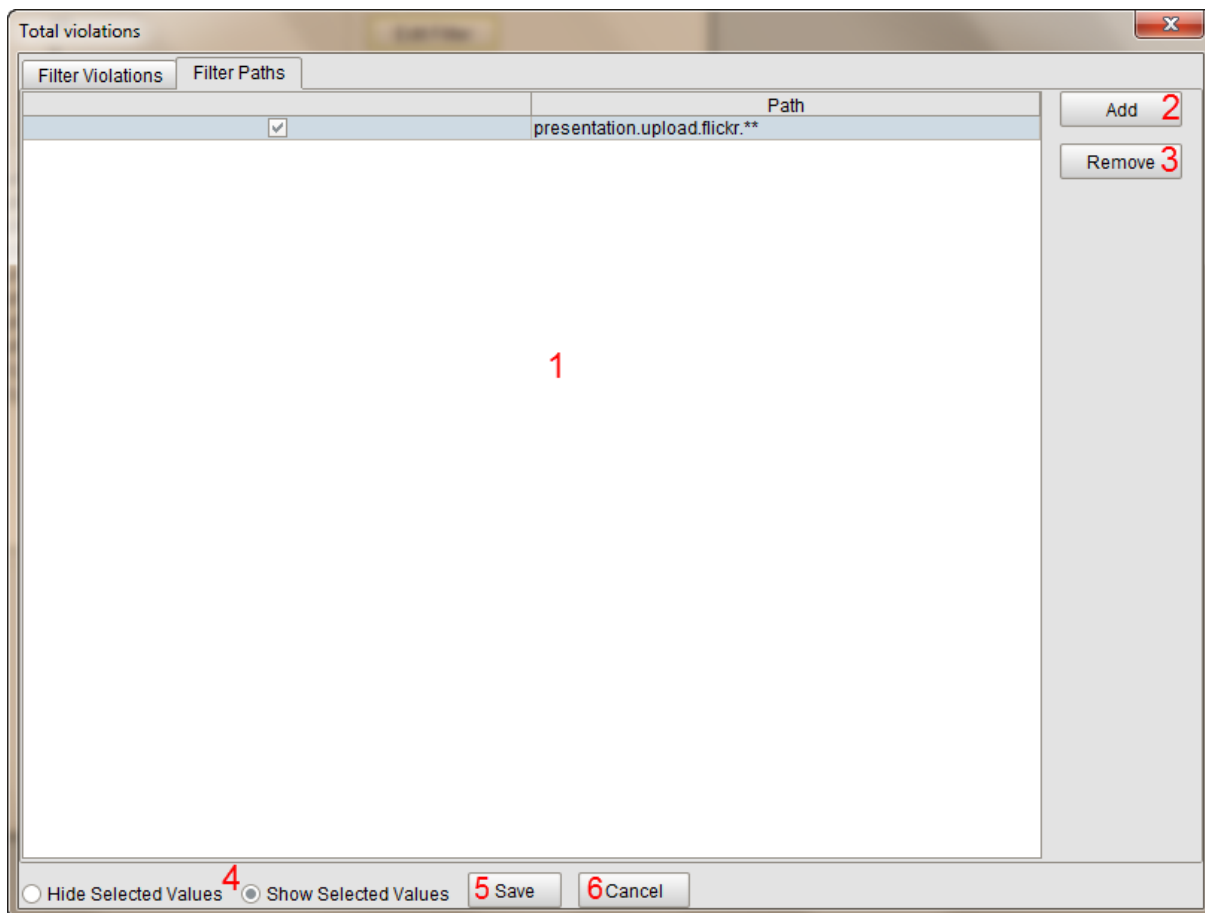


FIGURE 2 FILTER DIALOG TAB FOR FILTERING RULE- AND/OR VIOLATION TYPES

1. In this table the rule types that will be filtered can be selected.
2. In this table the violation types that will be filtered can be selected.
3. The option to choose if the filtered values must be shown or must be hidden.
4. When this button is pressed, the violations will be filtered.
5. When this button is pressed, the screen will be closed.



*FIGURE 3 FILTER DIALOG TAB FOR FILTERING CLASSPATHS*

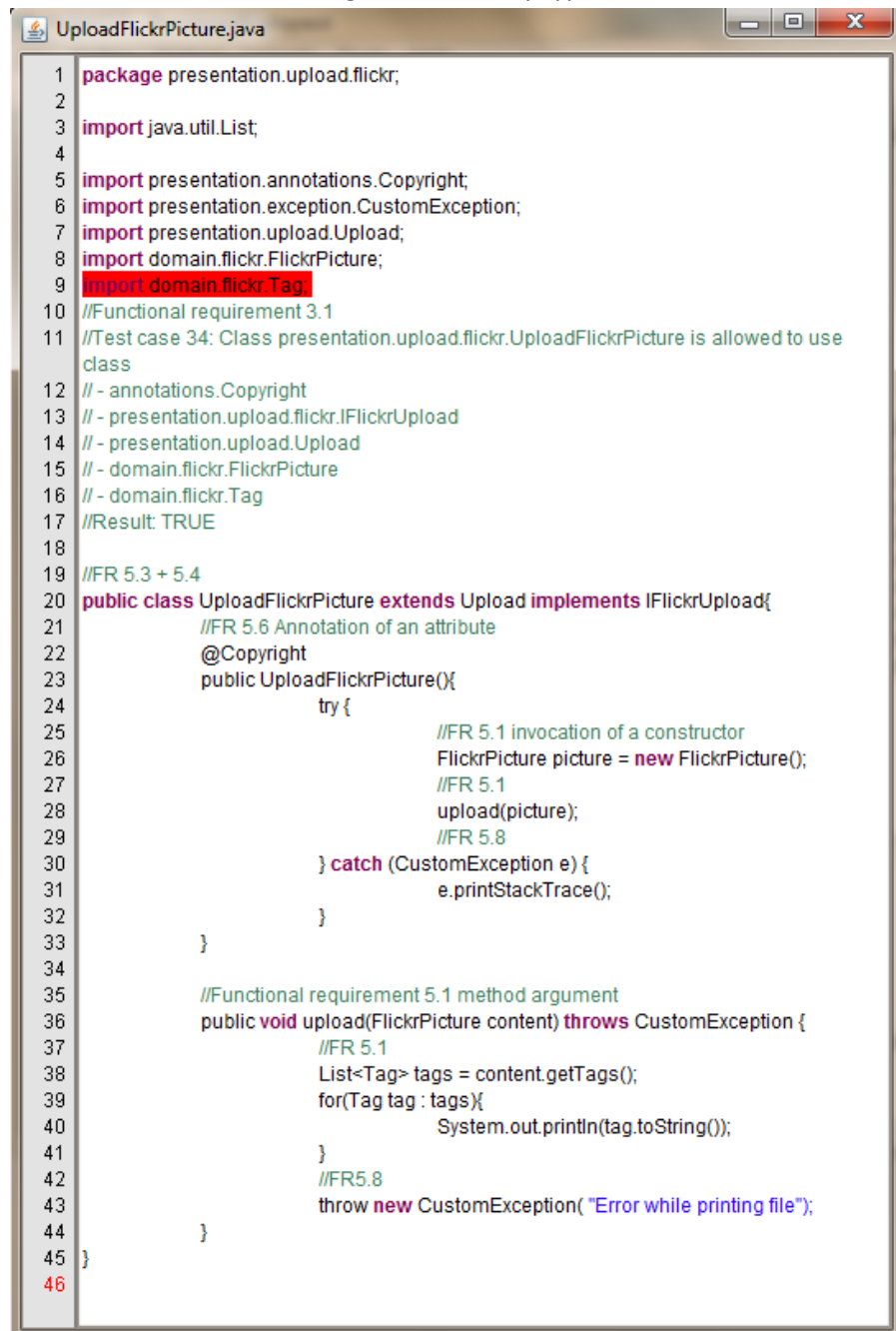
1. This table shows all the paths are added to filter. The physical paths can be filtered with a regex; the possibilities will be explained in table 2.
2. When this button is pressed, the system will add an empty field to area 1.
3. When this button is pressed, the system will remove the selected row from area 1.
4. The option to choose if the filtered values must be shown or must be hidden.
5. When this button is pressed, the violations will be filtered.
6. When this button is pressed, the screen will be closed.

Your Input Example	Path Input Example	Output Example
<b>java.St*</b>	java.String java.StringBuffer java.String.Fake com.class.String	java.String java.StringBuffer
<b>java.St**</b>	java.String java.StringBuffer java.String.Fake com.class.String	java.String java.StringBuffer java.String.Fake
<b>*.String</b>	java.String java.StringBuffer java.String.Fake com.class.String	java.String
<b>**String</b>	java.String java.StringBuffer java.String.Fake com.class.String	java.String com.class.String
<b>**Stri**</b>	java.String java.StringBuffer java.String.Fake com.class.String	java.String java.StringBuffer java.String.Fake com.class.String

*TABLE 2 OVERVIEW OF POSSIBLE REGEXES FOR FILTERING*

## CODE VIEWER

To see the exact piece of code which is responsible for the violation, the code viewer can be used. The highlighted line of code is colored according to the severity type of the violation.



```
1 package presentation.upload.flickr;
2
3 import java.util.List;
4
5 import presentation.annotations.Copyright;
6 import presentation.exception.CustomException;
7 import presentation.upload.Upload;
8 import domain.flickr.FlickrPicture;
9 import domain.flickr.Tag;
10 //Functional requirement 3.1
11 //Test case 34: Class presentation.upload.flickr.UploadFlickrPicture is allowed to use
12 //class
13 // - annotations.Copyright
14 // - presentation.upload.flickr.IFlickrUpload
15 // - presentation.upload.Upload
16 // - domain.flickr.FlickrPicture
17 // - domain.flickr.Tag
18 //Result: TRUE
19
20 //FR 5.3 + 5.4
21 public class UploadFlickrPicture extends Upload implements IFlickrUpload{
22     //FR 5.6 Annotation of an attribute
23     @Copyright
24     public UploadFlickrPicture(){
25         try {
26             //FR 5.1 invocation of a constructor
27             FlickrPicture picture = new FlickrPicture();
28             //FR 5.1
29             upload(picture);
30             //FR 5.8
31         } catch (CustomException e) {
32             e.printStackTrace();
33         }
34     }
35
36     //Functional requirement 5.1 method argument
37     public void upload(FlickrPicture content) throws CustomException {
38         //FR 5.1
39         List<Tag> tags = content.getTags();
40         for(Tag tag : tags){
41             System.out.println(tag.toString());
42         }
43         //FR5.8
44         throw new CustomException( "Error while printing file");
45     }
46 }
```

FIGURE 4 CODE VIEWER

## EXPORTING A REPORT

There also is the possibility to export the violations. Currently the violations can be exported to the following file types:

- XML
- HTML
- PDF

You can open this screen at “menu -> Validate -> Violation Report”.

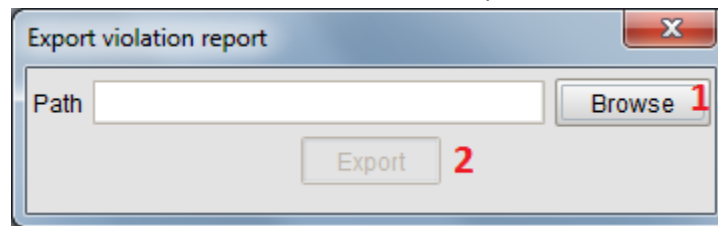


FIGURE 5 EXPORT REPORT BROWSE DIALOG

1. When this button is pressed, it opens an export dialog. (see figure 5)
2. When this button is pressed, the report will be exported.

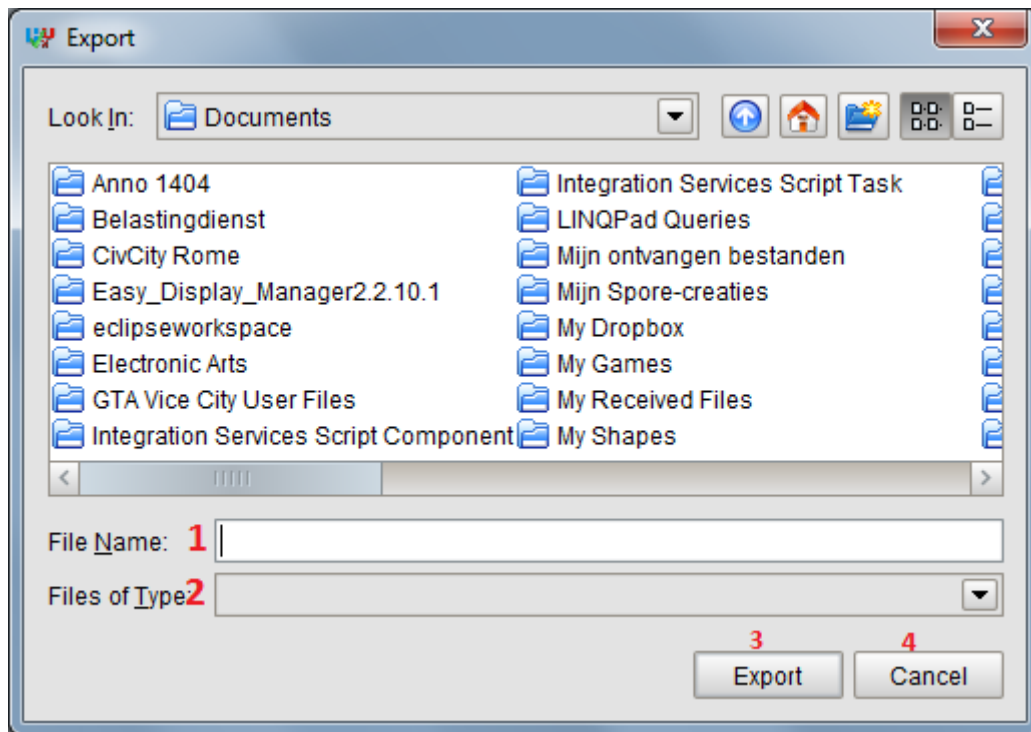


FIGURE 6 EXPORT REPORT FILE DIALOG

1. Select the file name and directory where the file must be saved.
2. Select the type of the file.
3. When this button is pressed, the path will be set in figure 4.
4. When this button is pressed, the screen will be closed.

## CONFIGURATION

The validate component provides also the possibility to configure the following:

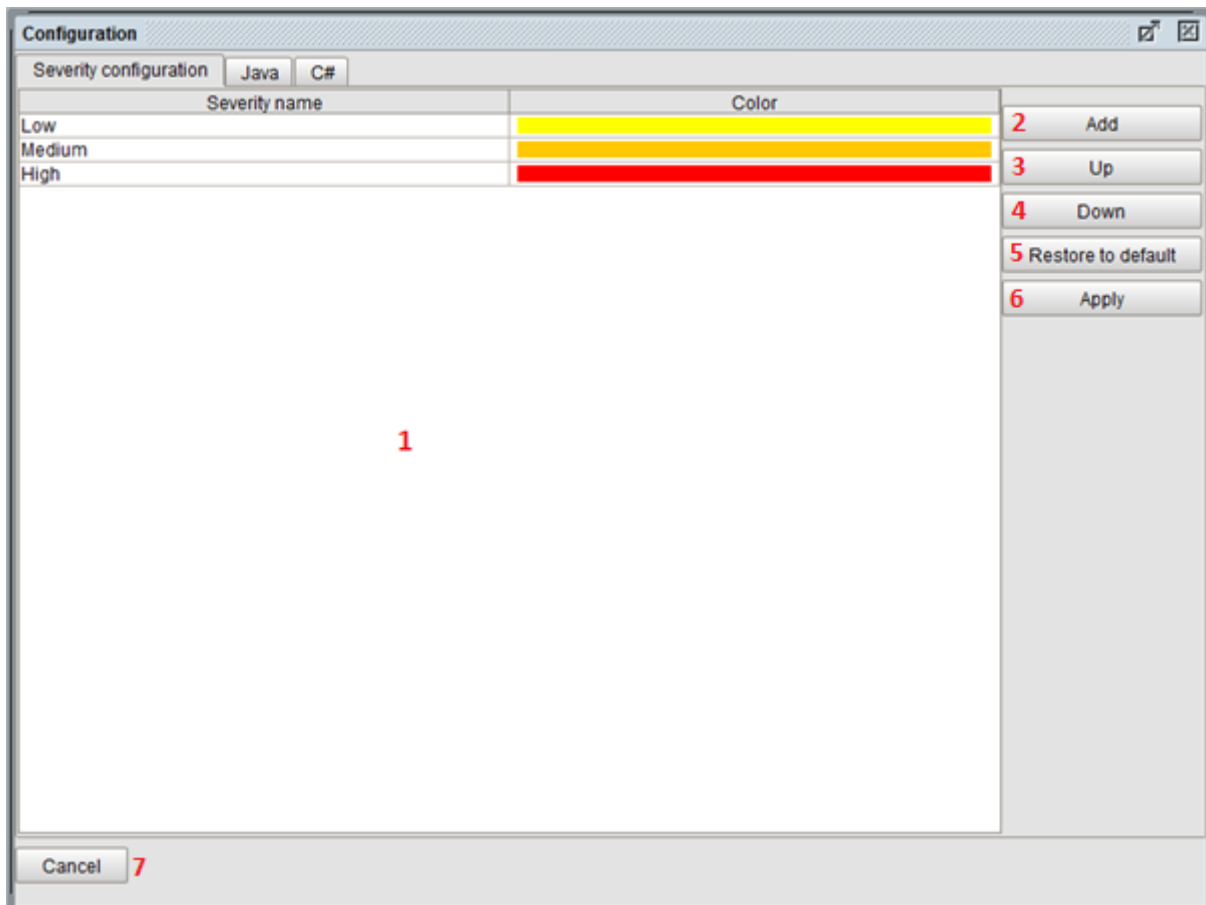
- Severities; adding new severities and change existing severities.
- Severity per rule type; change the severity for a rule.
- Severity per violation type; change the severity for a violation type.
- Active violation types; which violation types should be enabled by default in the filter when adding new rules in the define component. (For more information about filtering in rule types see the manual of the define component.)

The changes that are made are not directly reflected in the GUI. For example when severities are changed in the configuration, there must be revalidated to reflect the changes that are made in the configuration.

The screen is available once a workspace has been created. The configuration screen is available through “menu -> Validate -> Configuration”.

## CONFIGURE SEVERITIES

There are two or more tabs on the screen. The first is to create and edit severities. For every supported programming language a tab will be shown. Inside each of ‘programming language’ tab will contain three other tabs. The first two tabs are for setting the severity per rule/ violation types. The last tab is to set the active violation types.



*FIGURE 7 CONFIGURATION TAB FOR CHANGING AND/OR ADDING SEVERITIES*

1. This table lists all the severities. It is possible to change the name and/or color of a severity. The color can be changed by clicking on the color bar and a color picker dialog will pop up. The severities are in order from lowest to highest.
2. When this button is pressed, an empty severity will be added on the lowest place.
3. When this button is pressed, the selected severity will be moved up in the list.
4. When this button is pressed, the selected severity will be moved down in the list.
5. When this button is pressed, all the severities will be restored to the default severities.
6. When this button is pressed, all the changes in this tab will be saved.
7. When this button is pressed, the screen close without saving.

## CONFIGURE SEVERITIES PER RULE TYPE AND PER VIOLATION TYPE

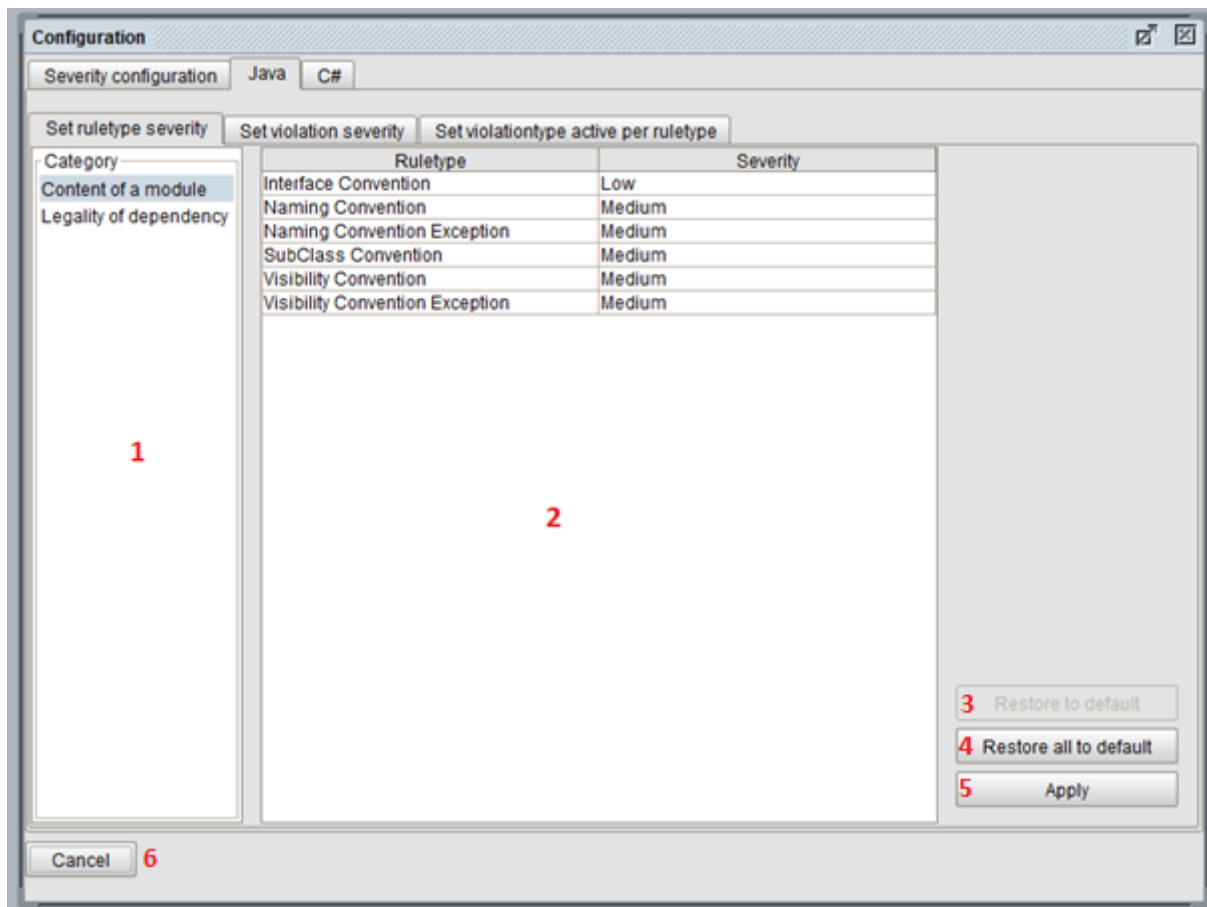


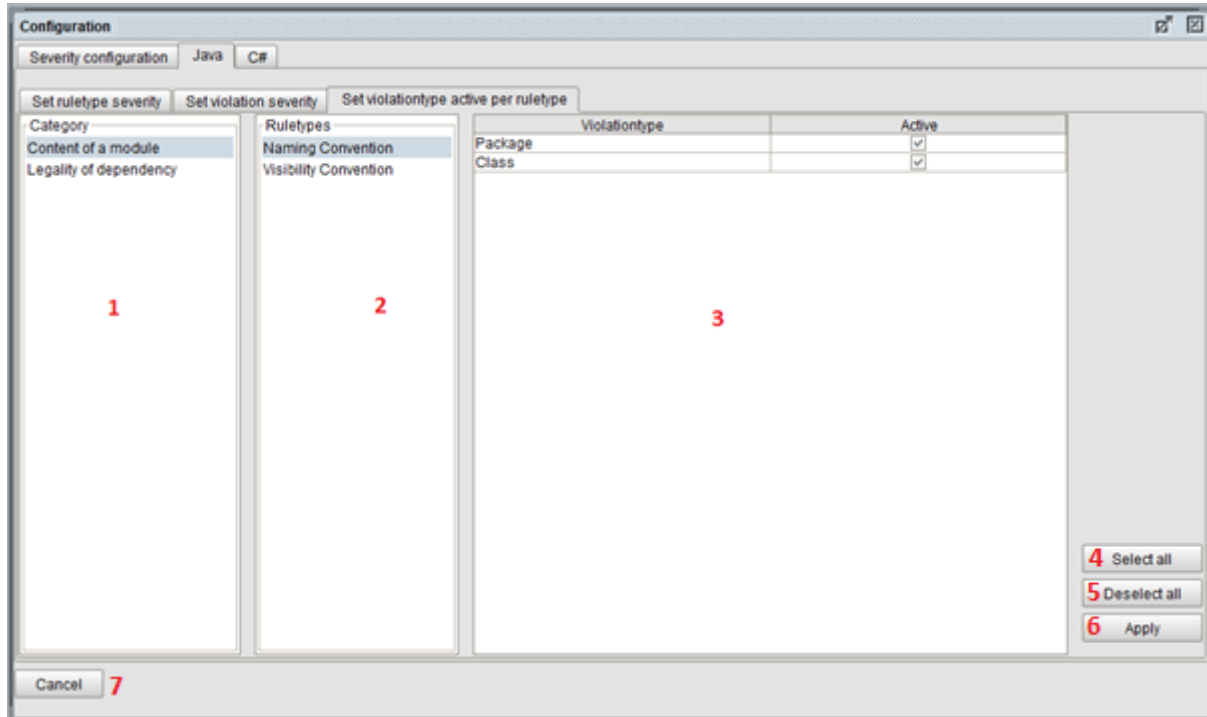
FIGURE 8 CONFIGURATION TAB FOR ASSIGNING SEVERITIES TO A RULE-/VIOLATION TYPE

This screen applies for the first two tabs.

1. List of categories.
2. Table with rule types or violation types. Changed by selecting a different category in area 1. You can change the severity by clicking on the severity name.
3. When this button is pressed, the selected rule type will be restored to its default severity level.
4. When this button is pressed, all the rule/ violation types will be restored to their default value.
5. When this button is pressed, all the changes in this tab will be saved.
6. When this button is pressed, the screen close without saving.



## CONFIGURE ACTIVE VIOLATION TYPES



*FIGUUR 8 CONFIGURATION TAB FOR SETTING THE ACTIVE VIOLATIONTYPES PER RULE TYPE*

1. List of categories.
2. List of rule types. Changes when you select another category in area 1
3. Table of active violation types. Changes when you select another rule type in area 2.
4. When this button is pressed, selects all the active violation types.
5. When this button is pressed, deselect all the active violation types
6. When this button is pressed, all the changes in this tab will be saved.
7. When this button is pressed, the screen close without saving

## SELECTION OF THE DEFAULT RULES

Within the configuration, you can define what rules should be added by default, to a new component in the software architecture definition. Opening the “Default rules” tab, will lead you to a list of allowed rules per component. Select a component to set or unset the default rules in the component.

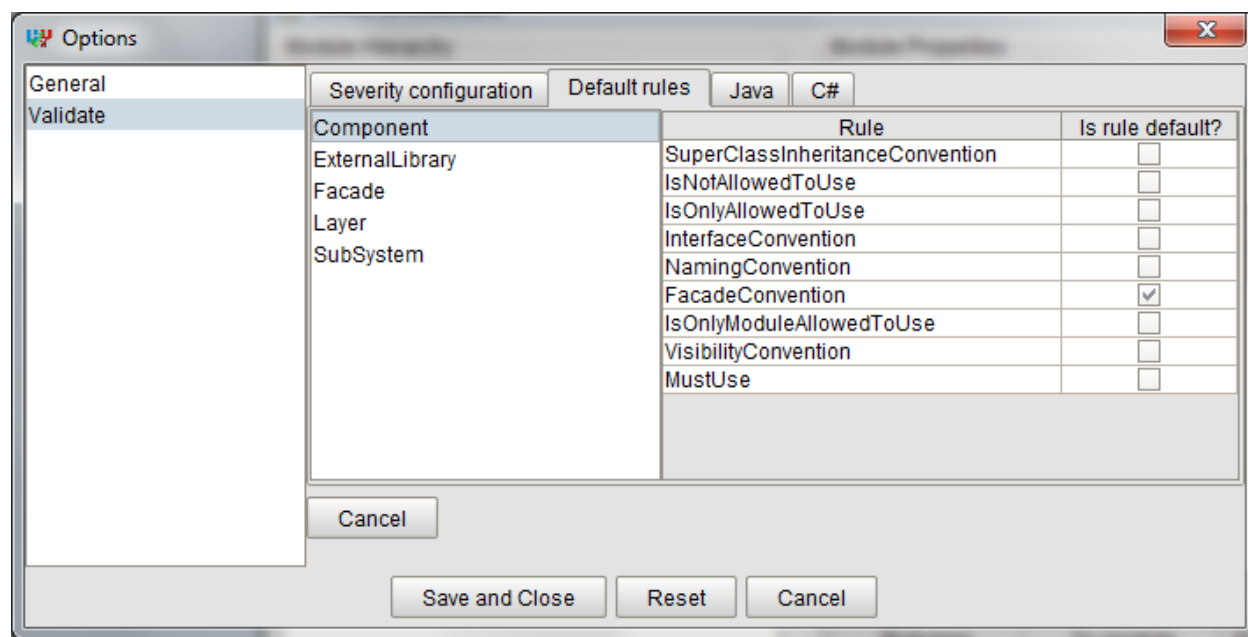


FIGURE 9: SELECTION OF THE DEFAULT RULES

There are a few rules selected by default. These can be deselected if required. These settings are instantly saved, so no save and close are required.