Hogeschool Utrecht

Architecture Graphics Manual

Architecture Graphics Team & Service

HUSACCT Team 2 – Graphics

Stan Verhoeckx
Dennis Bullee
Tim Bazuin
Ruwan Kellaert
Mark Wallenburg

HUSACCT Team 5 – Architecture Graphics

Tom de Bruijn
Johan van der Slikke
Guido van Tricht
Patrick van der Willik

Clients

Michiel Borkent (michiel.borkent@hu.nl)
Christian Köppe (christian.koppe@hu.nl)
Leo Pruijt (leo.pruijt@hu.nl)

Team mentor

Christian Köppe (christian.koppe@hu.nl)

Date

20-06-2013

Contents

Introduction

This document contains an overview of the different screens you will encounter while using HUSACCT (HU Software Architecture Compliance Checking Tool), along with an explanation of what the different parts do.

Before anything else, a glossary of used terms is provided. Subsequently, a description of the process to open diagrams within the application is given. Finally, the functionality in the main screen is described.

Terminology

The following terms are used within the HUSACCT application:

- *Dependency*
  - A relationship between layers, packages or classes.
- *Violation*
  - A violation of the defined architectural rules.
- *Zoom in*
  - Zooming in this context likely means going down into the architectural hierarchy. By zooming on a package, the package is unfolded and all the underlying packages and classes are shown.
- *Zoom out*
  - Step one level “up” in the architectural hierarchy.
- *Multizoom*
  - Zooming in on multiple packages/classes in order to view their connectedness.

Interactive Diagrams

The purpose of interactive diagrams is visualizing software architecture. The physical and logical diagrams are made based on the analysed code and the defined software architecture. These diagrams are a part of HUSACCT. Where other screens are primarily focused on entering data, these diagrams are primarily meant to display it.

Two types of diagrams are offered within HUSACCT. One based on the analysed software, which can be opened immediately after analysing an application and one based on the architecture as defined by the user. The analysed diagram shows the structure of the application on a physical level. This can be compared to a explorer type system where the dependencies between objects is also shown. This screen can be opened through the main menu of HUSACCT under “Analyse” -> “Analysed architecture diagram” as shown in figure 1.

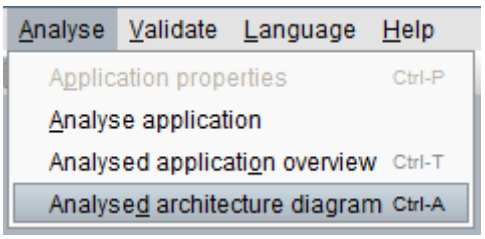

Figure 1 Opening the analysed architecture diagram

The defined diagram is based on user input. Before anything can be displayed, an architecture needs to be defined. Based on this information and the linked physical objects, a logical architectural diagram is shown. Using this diagram it is possible to compare the documented architecture to the actual architecture. This diagram supports the creation of logical layer models and component models. This diagram can be accessed as shown in figure 2.

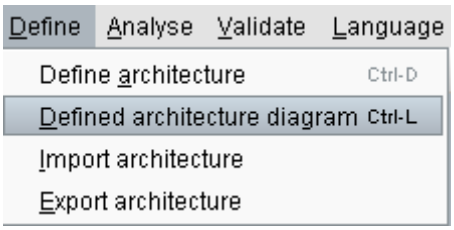

Figure 2 Opening the defined architecture diagram

The following examples use the Analysed architecture diagram as an example, but the options and functions shown can be applied to both diagrams as their interfaces are identical.

User Interface

Figure 3 shows the main screen of the graphics component. The numbers in the image correspond to the following chapter numbers

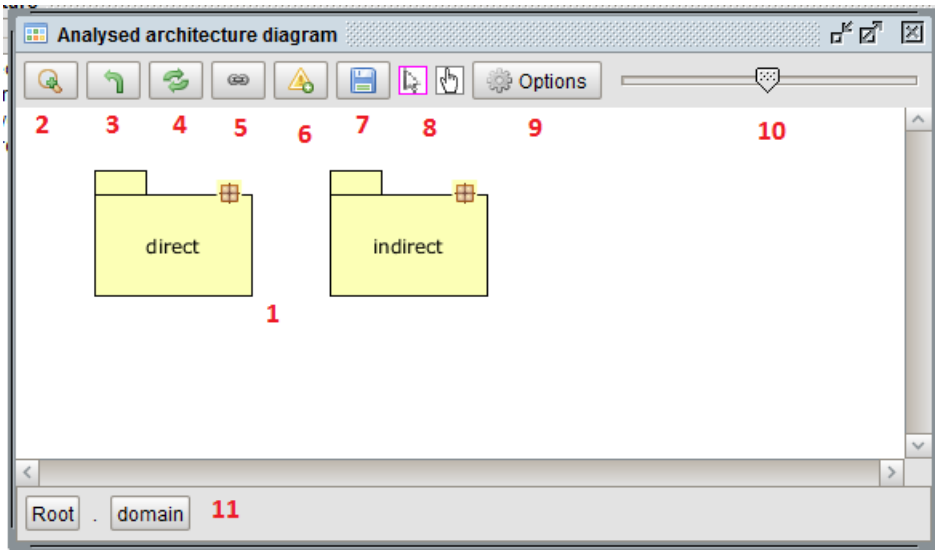

Figure 3 Graphics service main screen

1. Drawing surface

This surface is meant to display the diagram. The figures and lines that are involved with visualizing a software architecture are shown here. This area is interactive – modules can be dragged around to facilitate reader comprehension. Selecting figures and lines allows the end-user to access more detailed data about them. Double clicking on a figure zooms in on it.

2. Zoom

Left clicking this button zooms on the selected figures. Right clicking on this button displays a menu that offers the different zoom options: “Zoom with context” and “Zoom in”.
“Zoom with context” keeps the other diagrams displayed at the same level, while zooming in on the selected diagrams.
“Zoom in” only zooms in on the selected diagrams.

Figure 4 shows an example of two packages that have been zoomed in on at the same time.

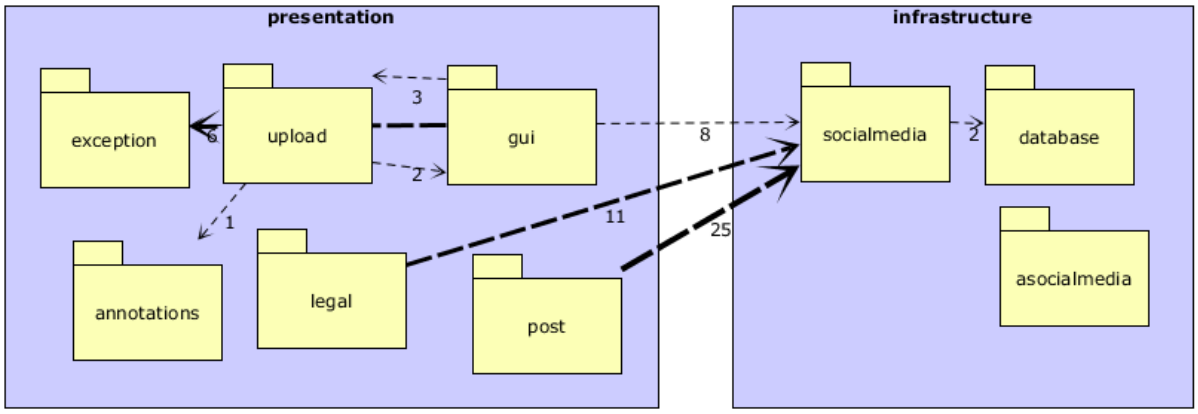

Figure 4 Multi zoom diagram.

3. Zoom out

This button steps up one level of abstraction in the diagram.

4. Refresh diagram

Clicking this button results in the diagram being refreshed from internal data. This is primarily useful when/if things have been changed using other screens within HUSACCT.

5. Toggle dependencies

This button toggles the dependency lines between figures in the diagram. Figure 5 shows an example of a diagram where the dependency lines have been turned off.

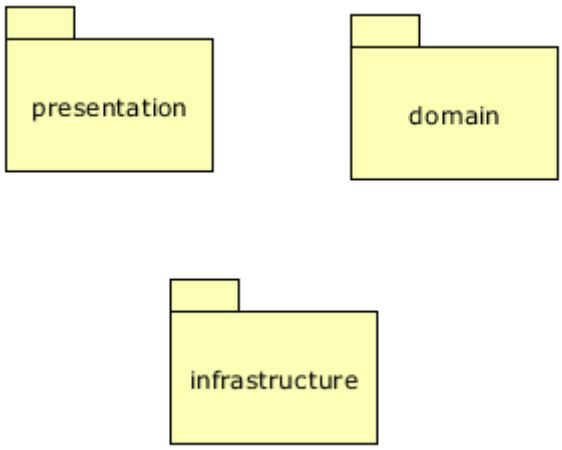

Figure 5 Dependency lines off

6. Toggle violations

By default the violations are not shown. These arrows, when drawn between figures, indicate violations of the defined architectural rules. These lines are not black in order to facilitate distinguishing between them and dependency lines. This colour can be changed using the validation configuration screen.

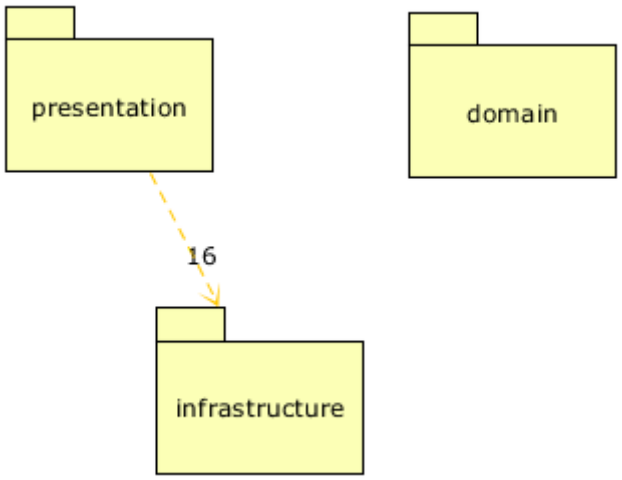

Figure 6 Violations enabled, Dependencies disabled

7. Export to image

This button opens a screen that will allow the end-user to store the displayed diagram as an image file, to be used separately from HUSACCT. The exported image will be in .png format.

8. Mouse tool selection

By default the selection tool is enabled. The currently selected item can be identified from the magenta border around it. The selection tool (the leftmost tool) allows for the selection of figures in the diagram and moving or zooming in on them.
The pan tool (the rightmost tool) allows for panning of the viewport without using the scroll bars on the sides of the diagram. If the diagram is zoomed out too far to be scrollable or too small to scroll, the pan tool does nothing.

9. Options menu

The options menu offers all the functionality of the smaller options bar above the architectural diagram currently opened, as well as some others. For the sake of brevity, only new functionality is explained here. Looking at Figure 7, options 1-7 and 9 have been explained in previous chapters. The other options will be explained in the following chapters.

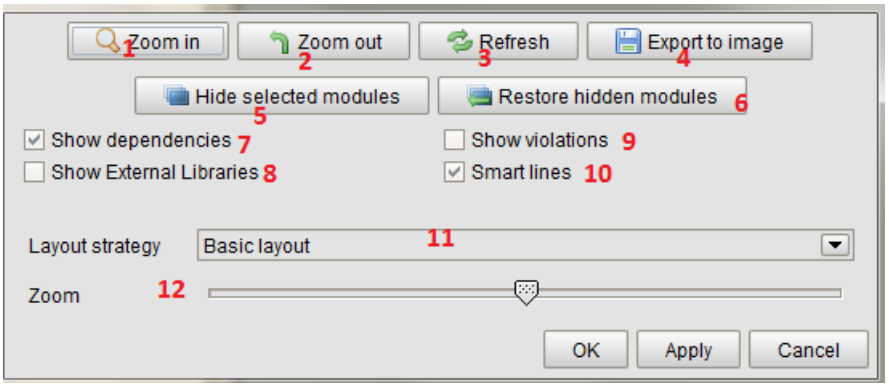

Figure 7 Options screen

8. Toggle external libraries

External libraries are referenced libraries outside of the source of the currently analysed project. These dependencies can be displayed or not, this option allows the end-user to control this.

10. Smart lines

This option condenses the many arrows that can arise from displaying a complex diagram into fewer arrows. To accomplish this, parallel arrows are condensed into a single arrow, whose thickness depends on the amount of lines it “contains”.

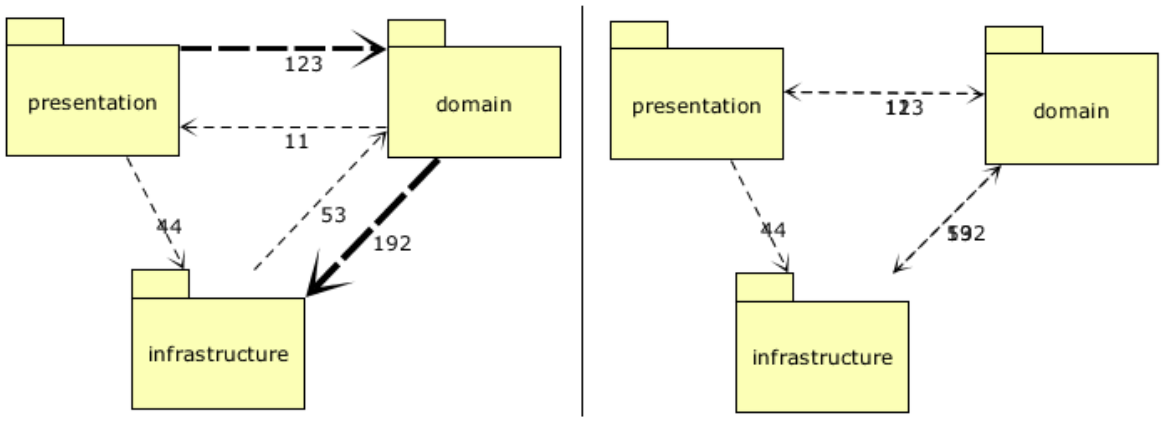

Figure 8 Left: Smart lines on. Right: Smart lines off. Some lines overlap in the rightmost diagram.

12. Layout strategy

A layout strategy describes a method of automatically ordering figures. This reduces initial visual clutter and may help with initial comprehension.

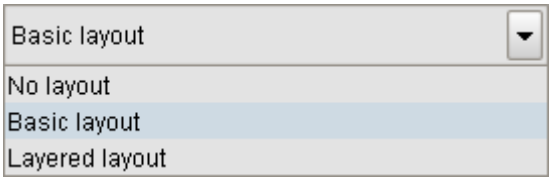

Figure 9 Layout strategies

- No layout
  - This is no layout. All figures are put in the top left of the screen.
- Basic layout
  - This is default. This layout strategy puts the figures in a grid with a width that is as big as the screen allows and a height that varies based on the amount of figures that go in it.
- Layered layout
  - Experimental layout strategy. This strategy takes the dependencies between figures into account and tries to draw a more sensible initial layout than the other strategies offer. This layout strategy is not in a stable state and is offered merely as a preview.

10. Zoom slider

This slider scales the diagram up or down.

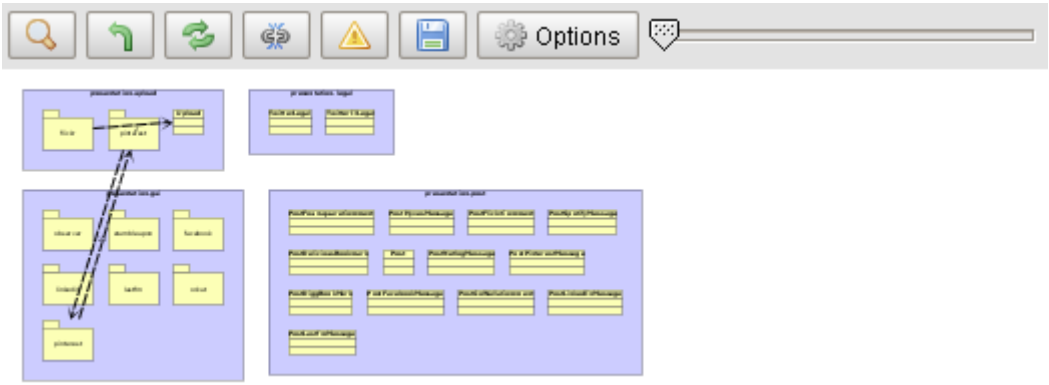

Figure 10 Scaled diagram

11. Location/"breadcrumb" bar

Shows the current location of the diagram and allows to see the path through which the end-user arrived there. Pressing one of these buttons zooms to that level.

Figure 11 Location bar as displayed when zooming on a single item

Figure 12 Location bar as displayed when zooming on multiple items