Implementation document

DaCoPAn2

Helsinki, 7th May 2005

Software Engineering Project

UNIVERSITY OF HELSINKI
Department of Computer Science

Course

581260-4 Software Engineering Project (6 cr)

Project Group

Mikko Airaksinen
Tomi Korkki
Pauli Miettinen
Timo Tuominen
Mikko Väänänen

Customer

Markku Kojo

Project Masters

Juha Taina (Supervisor)
Marianne Korpela (Instructor)

Homepage

`http://www.cs.helsinki.fi/group/dacopan2`

Change Log

Version	Date	Modifications
0.1	28.04.2005	Initial version
0.2	03.05.2005	Beta version
1.0	05.05.2005	Final version

Contents

1 Introduction

This is the Implementation document of the DaCoPAn2 Software Engineering project at the Computer Science Department of the University of Helsinki. The DaCoPAn2 project is a follow-up project for the DaCoPAn project. Its main goal is to improve and expand the Animator subsystem produced by the DaCoPAn group.

This document summarizes the changes made to the design of the Animator during the implementation phase. All of these changes are relative to DaCoPAn2 Design document [1].

2 Message Sequence Chart

Albeit the Message Sequence Chart (MSC) had quite a lot of working code and our updates were supposed to be minor, its design was strongly changed during the implementation phase.

2.1 The meaning of `now time`

Probably the largest change affects the meaning of `now time` that Message Sequence Chart gets from Animation Sequence framework. `Now time` is handled differently based on whether or not MSC is in "play"-mode. When MSC is in "play"-mode, the `now time` is handled as if they were just interrupts from some sort of timer and they only advance the animation by some step. In the other hand, when MSC is not in "play"-mode, i.e. it is paused or stepping, `now time` means the "animation time", i.e. the time stamp within the animation. To work correctly, MSC sets the time of Animation Sequence framework differently when "Play" or "Pause" are selected. The class `MSCPanel` also has two new methods—`getRealTime():float` and `getAnimatorTime(time:float):float`—which return the animation time of current `now time` and "play"-mode timestamp for given time, respectively.

In order to make these changes to work correctly, the class `MainFrame` was also modified such that it changes the `now time` to be in correct mode whenever it is necessary.

2.2 Dropped units

The way to calculate the gradient of dropped units was refined during the implementation process. The gradient is calculated as follows:

> If nearest unit sent from same host is not dropped, use same gradient. Else if nearest unit sent from same host is sent after this unit and is dropped, but the unit sent before this unit is not dropped, use the gradient of that previously sent unit. If, however, this unit is first unit, i.e.

there is no previously sent packet, set gradient to be 0. Finally, if both of the nearest units are dropped or nearest unit is sent before this unit, use the same gradient as the last drawn dropped packet.

Because of implementation restrictions, the dropped packets are not drawn after their sent point is scrolled above the drawing area. To minimize this effect, the cross indicating that the packet is dropped, is drawn very near to the starting end of unit line. If user has decided to view some unit header information, the drawing point of dropped cross is pushed further such that the unit header information fits over the delay line.

These changes make it very hard to synchronize Message Sequence Chart and Unit Flow Orchestration when speaking about dropped packets. Thus that synchronization is not implemented—for more information about dropped units in Unit Flow orchestration, see section 3.

3 Unit Flow Orchestration

There were no designed changes to Unit Flow Orchestration (UFO). However, changes to Message Sequence Chart forced us to make some minor changes to UFO, too. First of all, it was synchronized to use same meaning of `now time` as MSC (see section 2.1). Another change had to do with the speed and dropping point of dropped units. As it was found out to be very hard to synchronize UFO and MSC in this case, synchronization was not implemented. Instead the dropped packets now advance in UFO channel with standard speed and disappear from channel right in the middle of it.

4 Time Sequence Chart

Changes to Time Sequence Chart (TSC) were minimal, but there were however two of them. First the microsecond scale was not implemented as it was found to be unnecessary and because it dropped the performance of software too much. Secondly user is allowed to choose from different flows of packets from a drop-down list. This was done because it was obvious that TSC cannot give a meaningful view from different flows having different sequence numbers simultaneously.

5 Time panel

No changes were designed for `TimePanel`. However, when the meaning of `now time` was changed for Message Sequence Chart (see section 2.1), also `TimePanel` class needed some minor adjustments. It was changed to show always the “animation

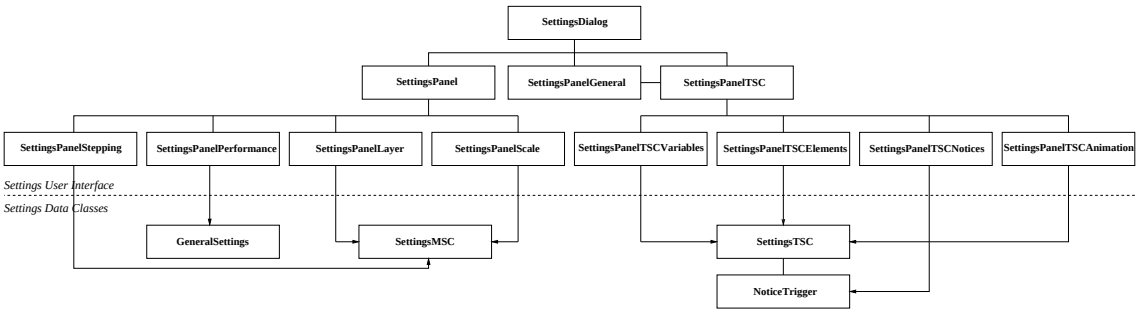


Figure 1: Settings panels class diagram

time". Also, when in Time Sequence Chart, the `TimePanel` shows only `-.-` as an animation time because Time Sequence Chart does not interact with it.

6 Settings

The settings related features were implemented mainly as designed, with mostly minor modifications. The largest changes had to do with making the MSC specific settings persistent so that they could be saved and stored independently of scenarios.

6.1 Settings User Interface Panels

A couple of new SettingsPanels were implemented compared to the design. All the settings related features were moved from `UserInterface` to `SettingsDialog` which is a `JDialog` window managing the different settings panels. Under it are the three main settings `JPanels` all containing more panels in `TabbedPanes`. An updated class diagram of the panels is in figure 1

6.2 Global MSC Settings

The MSC specific settings in the `SettingsMSC` class turned out to be more tightly related to the specific `DataView` instance of the currently loaded PEF than was first thought. The main problem was that SettingsMSC objects are always related to a single `Layer` of a DataView.

To be able to store all of the user chosen `VariableDefinition` arrays representing variables to be shown in the animation for all possible layers, class `GlobalSettingsMSC` was devised. It enables us to store the arrays with their `Protocol`, not Layer, as a key. It uses as values instances of the public inner class `VariableFields` where variables for the columns and center drawing area are stored.

However, since the Protocol class implements the `Identifiable` interface, it could not be serialized using `XStreamObjectSerializer` because it stores identifiable objects using their ids and restoring them requires that the current DataView is aware

of them, which is not always the case. This is why the `protocolId` is used for GlobalSettingsMSC. The class also stores the scenario- and layer-independent SettingsMSC object.

An updated class diagram for all the settings classes is in figure 2.

6.3 MSC timeScale

The formula to calculate the `timeScale` from selected `progressSpeed` is the inverse of the formula presented in Design document [1, p. 32–33]. Define

$$\begin{aligned} \text{last } y\text{-coordinate in layer} &= y \text{ pixels}, \\ \text{length of animation} &= l \text{ s}_\text{a}, \\ \text{wanted progress line speed} &= v \text{ pixels/s}_\text{r} \text{ and} \\ \text{time scale} &= x \text{ s}_\text{a}/\text{s}_\text{r}. \end{aligned}$$

Now the correct formula is

$$x = \frac{y}{lv}.$$

7 Miscellaneous changes

This section includes the relatively small changes made to different unmentioned classes.

7.1 The `CalcYCoord` class

The constructor of `CalcYCoord` now gets an object of type `DataView` instead of `List` as an reference to units. The correct constructor definition is thus `CalcYCoord(dataView:DataView, settings:SettingsMSC)`.

7.2 The `payload_size` variable for TCP

The animator now calculates the payload size for TCP segments and adds it as an unit header field variable. This calculation is done using method `getPayloadSize()` from class `TransferUnit`. Protocol TCP is identified by its id-number, witch is supposed to be “3”. Should PEF already has a variable with same name, it is overridden. This fix is to be toughed as an temporarily and Analyzer should be fixed as soon as possible to add the information from TCP pseudo header to PEF-files.

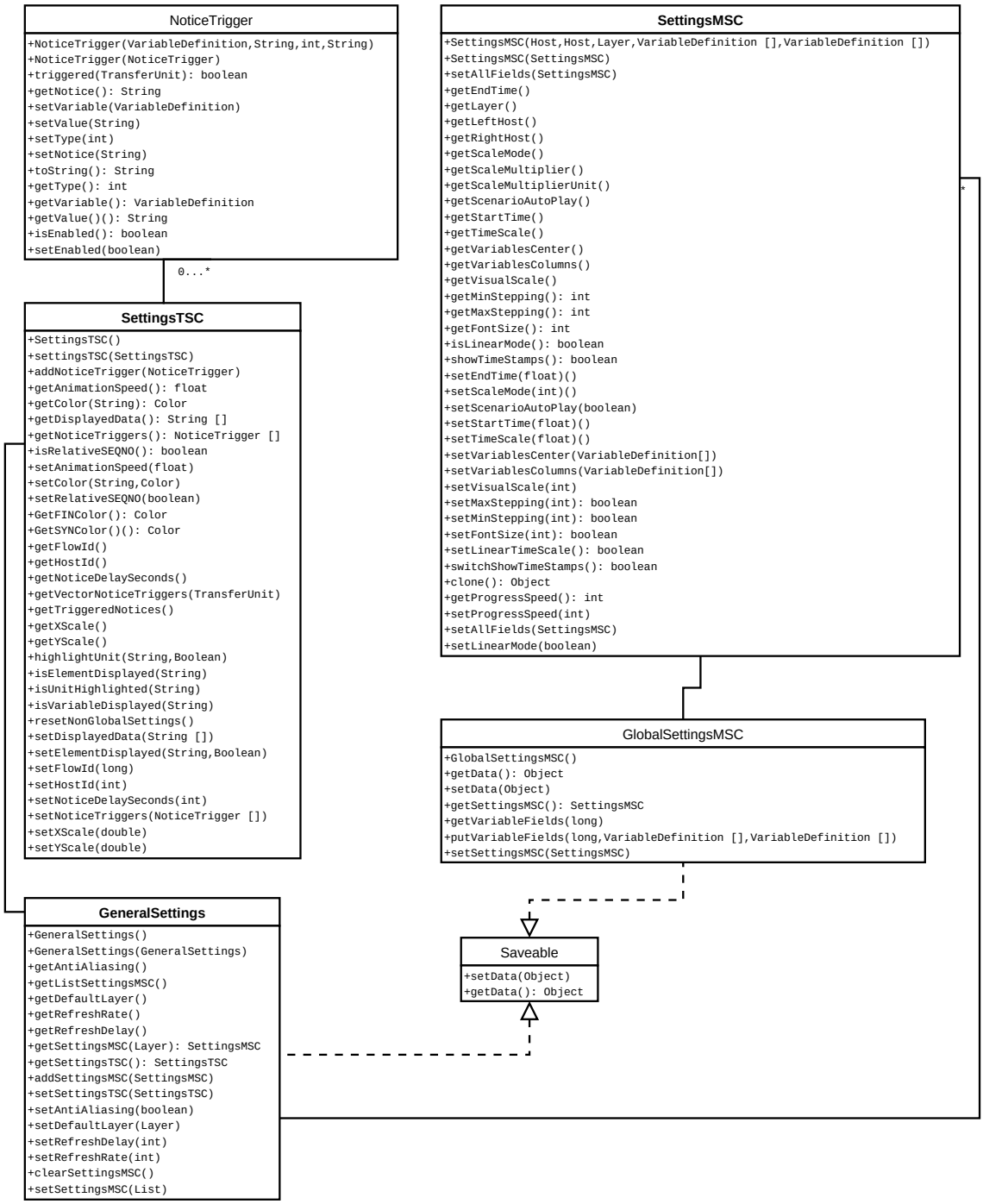


Figure 2: Settings class diagram

7.3 The `TransferUnit::getPayloadSize()` method

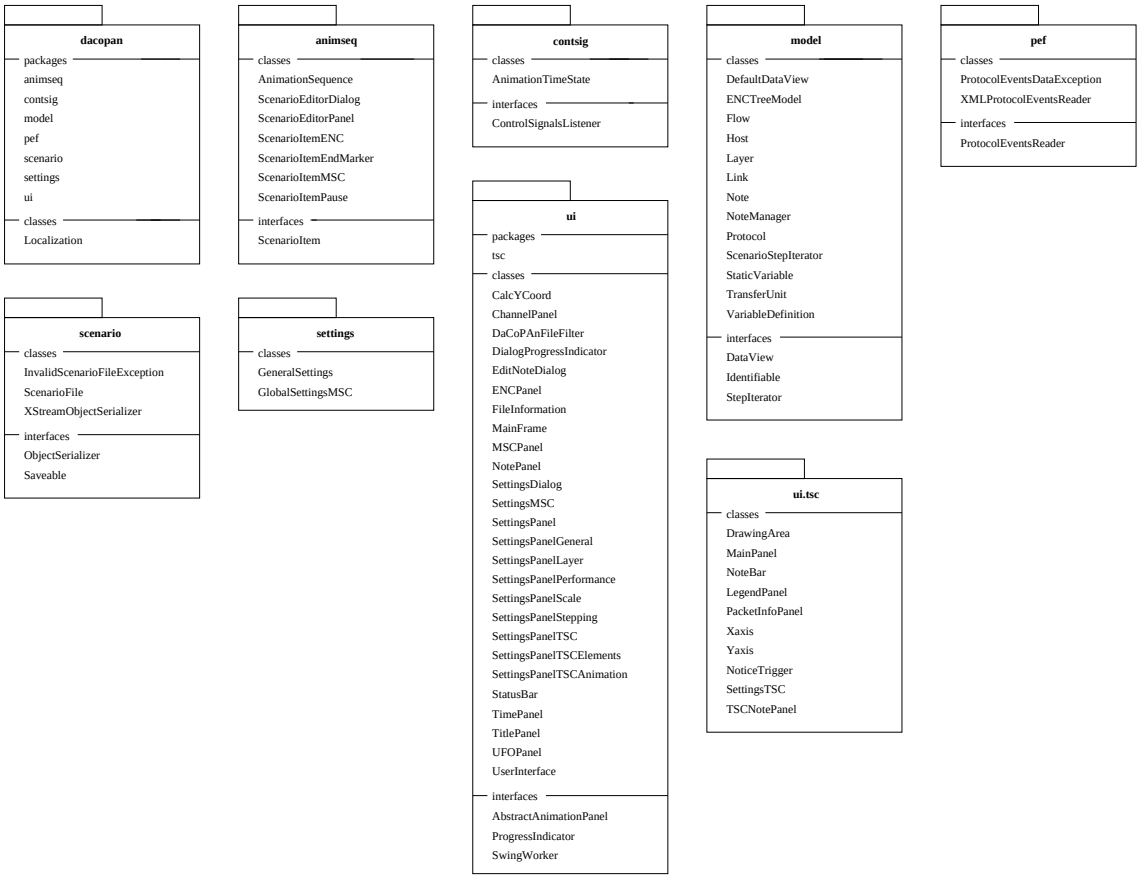
Method `getPayloadSize()` at class `TransferUnit` now uses value "5" as an "educated guess" for Internet Header Length, when this standard IP header field is missing from PEF file. This is to be considered as an temporarily fix and Analyzer should be fixed as soon as possible to add the IHL header field value to the PEF files.

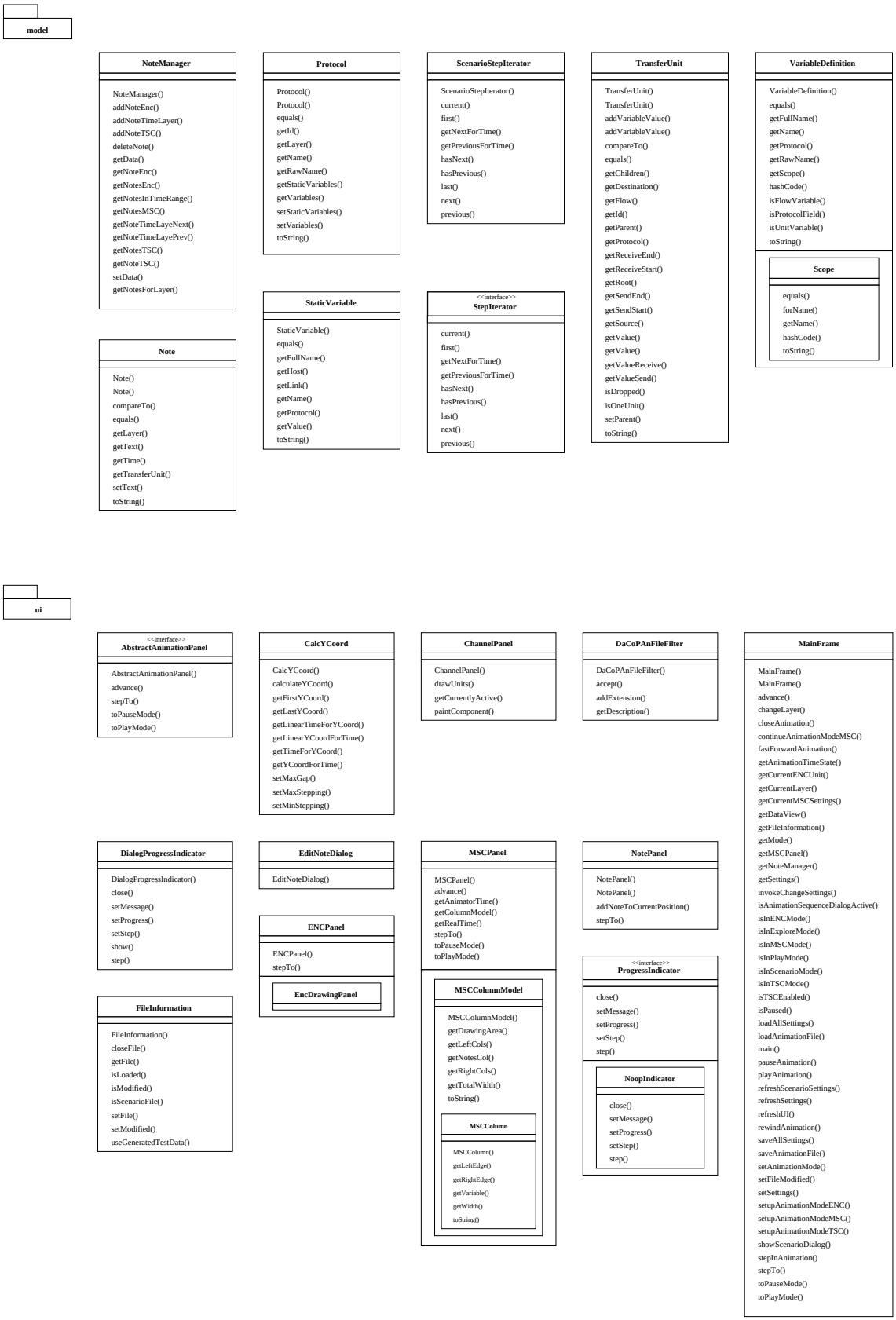
References

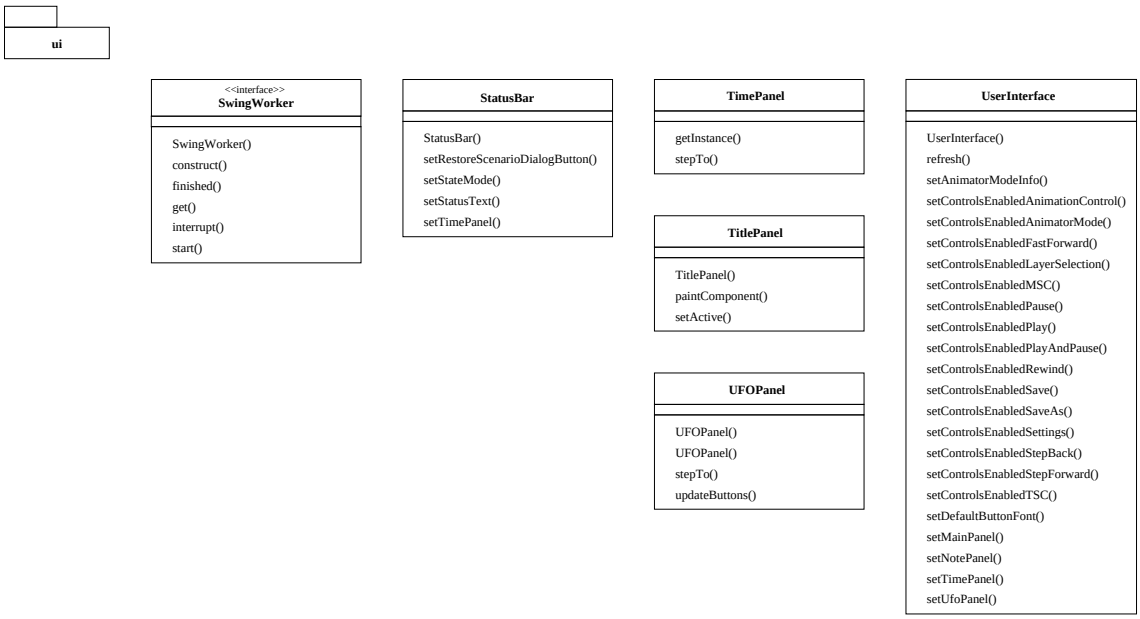
1 DaCoPAn2 Software Engineering Project, *Design document*. Relase 2.0. University of Helsinki, March 2005.

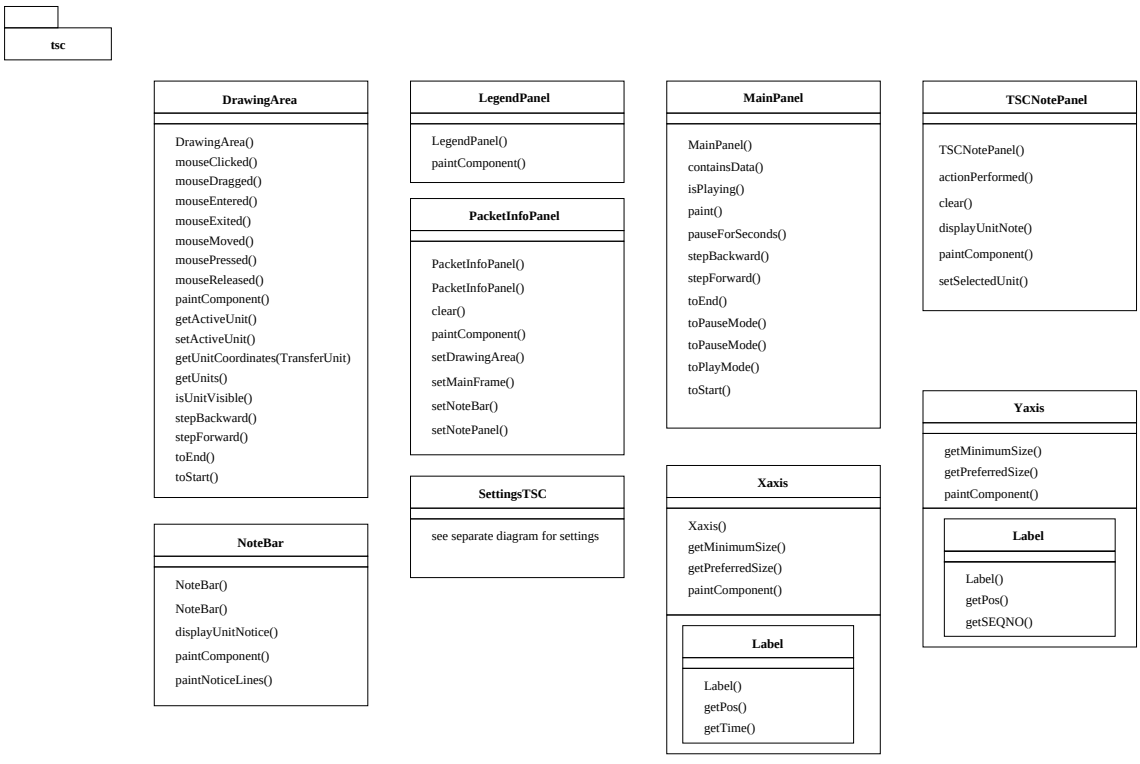
Appendix A. Packet-level class diagrams

The following are updated packet-level class diagrams for each individual packet where changes have been made to the diagrams in the appendix B of the design document. The diagrams for the packages `settings` and `ui.settings` in the design document are obsolete and the diagrams 1 and 2 should be consulted instead.




model
NoteManager
NoteManager()
addNoteEnc()
addNoteTimeLayer()
addNoteTSC()
deleteNote()
getData()
getNoteEnc()
getNotesEnc()
getNotesInTimeRange()
getNotesMSC()
getNoteTimeLayeNext()
getNoteTimeLayePrev()
getNotesTSC()
getNoteTSC()
setData()
getNotesForLayer()
Note
Note()
Note()
compareTo()
equals()
getLayer()
getText()
getTime()
getTransferUnit()
setText()
toString()
Protocol
Protocol()
Protocol()
equals()
getId()
getLayer()
getName()
getRawName()
getStaticVariables()
getVariables()
setStaticVariables()
setVariables()
toString()
StaticVariable
StaticVariable()
equals()
getFullName()
getHost()
getLink()
getName()
getProtocol()
getValue()
toString()
ScenarioStepIterator
ScenarioStepIterator()
current()
first()
getNextForTime()
getPreviousForTime()
hasNext()
hasPrevious()
last()
next()
previous()
<<interface>>
StepIterator
current()
first()
getNextForTime()
getPreviousForTime()
hasNext()
hasPrevious()
last()
next()
previous()
TransferUnit
TransferUnit()
TransferUnit()
addVariableValue()
addVariableValue()
compareTo()
equals()
getChildren()
getDestination()
getFlow()
getId()
getParent()
getProtocol()
getReceiveEnd()
getReceiveStart()
getRoot()
getSendEnd()
getSendStart()
getSource()
getValue()
getValue()
getValueReceive()
getValueSend()
isDropped()
isOneUnit()
setParent()
toString()
VariableDefinition
VariableDefinition()
equals()
getFullName()
getName()
getProtocol()
getRawName()
getScope()
hashCode()
isFlowVariable()
isProtocolField()
isUnitVariable()
toString()
Scope
equals()
forName()
getName()
hashCode()
toString()
ui
<<interface>>
AbstractAnimationPanel
AbstractAnimationPanel()
advance()
stepTo()
toPauseMode()
toPlayMode()
CalcYCoord
CalcYCoord()
calculateYCoord()
getFirstYCoord()
getLastYCoord()
getLinearTimeForYCoord()
getLinearYCoordForTime()
getTimeForYCoord()
getYCoordForTime()
setMaxGap()
setMaxStepping()
setMinStepping()
ChannelPanel
ChannelPanel()
drawUnits()
getCurrentlyActive()
paintComponent()
DaCoPAnFileFilter
DaCoPAnFileFilter()
accept()
addExtension()
getDescription()
MainFrame
MainFrame()
MainFrame()
advance()
changeLayer()
closeAnimation()
continueAnimationModeMSC()
fastForwardAnimation()
getAnimationTimeState()
getCurrentENCUnit()
getCurrentLayer()
getCurrentMSCSettings()
getDataView()
getFileInformation()
getMode()
getMSCPanel()
getNoteManager()
getSettings()
invokeChangeSettings()
isAnimationSequenceDialogActive()
isInENCMode()
isInExploreMode()
isInMSCMode()
isInPlayMode()
isInScenarioMode()
isInTSCMode()
isTSCEnabled()
isPaused()
loadAllSettings()
loadAnimationFile()
main()
pauseAnimation()
playAnimation()
refreshScenarioSettings()
refreshSettings()
refreshUI()
rewindAnimation()
saveAllSettings()
saveAnimationFile()
setAnimationMode()
setFileModified()
setSettings()
setupAnimationModeENC()
setupAnimationModeMSC()
setupAnimationModeTSC()
showScenarioDialog()
stepInAnimation()
stepTo()
toPauseMode()
toPlayMode()
DialogProgressIndicator
DialogProgressIndicator()
close()
setMessage()
setProgress()
setStep()
show()
step()
EditNoteDialog
EditNoteDialog()
MSCPanel
MSCPanel()
advance()
getAnimatorTime()
getColumnModel()
getRealTime()
stepTo()
toPauseMode()
toPlayMode()
MSCColumnModel
MSCColumnModel()
getDrawingArea()
getLeftCols()
getNotesCol()
getRightCols()
getTotalWidth()
toString()
MSCColumn
MSCColumn()
getLeftEdge()
getRightEdge()
getVariable()
getWidth()
toString()
NotePanel
NotePanel()
NotePanel()
addNoteToCurrentPosition()
stepTo()
<<interface>>
ProgressIndicator
close()
setMessage()
setProgress()
setStep()
step()
NoopIndicator
close()
setMessage()
setProgress()
setStep()
step()
ENCPanel
ENCPanel()
stepTo()
EncDrawingPanel
FileInformation
FileInformation()
closeFile()
getFile()
isLoaded()
isModified()
isScenarioFile()
setFile()
setModified()
useGeneratedTestData()


ui
<<interface>>
SwingWorker
SwingWorker()
construct()
finished()
get()
interrupt()
start()
StatusBar
StatusBar()
setRestoreScenarioDialogButton()
setStateMode()
setStatusText()
setTimePanel()
TimePanel
getInstance()
stepTo()
TitlePanel
TitlePanel()
paintComponent()
setActive()
UFOPanel
UFOPanel()
UFOPanel()
stepTo()
updateButtons()
UserInterface
UserInterface()
refresh()
setAnimatorModeInfo()
setControlsEnabledAnimationControl()
setControlsEnabledAnimatorMode()
setControlsEnabledFastForward()
setControlsEnabledLayerSelection()
setControlsEnabledMSC()
setControlsEnabledPause()
setControlsEnabledPlay()
setControlsEnabledPlayAndPause()
setControlsEnabledRewind()
setControlsEnabledSave()
setControlsEnabledSaveAs()
setControlsEnabledSettings()
setControlsEnabledStepBack()
setControlsEnabledStepForward()
setControlsEnabledTSC()
setDefaultButtonFont()
setMainPanel()
setNotePanel()
setTimePanel()
setUfoPanel()


tsc
DrawingArea
DrawingArea()
mouseClicked()
mouseDragged()
mouseEntered()
mouseExited()
mouseMoved()
mousePressed()
mouseReleased()
paintComponent()
getActiveUnit()
setActiveUnit()
getUnitCoordinates(TransferUnit)
getUnits()
isUnitVisible()
stepBackward()
stepForward()
toEnd()
toStart()
NoteBar
NoteBar()
NoteBar()
displayUnitNotice()
paintComponent()
paintNoticeLines()
LegendPanel
LegendPanel()
paintComponent()
PacketInfoPanel
PacketInfoPanel()
PacketInfoPanel()
clear()
paintComponent()
setDrawingArea()
setMainFrame()
setNoteBar()
setNotePanel()
SettingsTSC
see separate diagram for settings
MainPanel
MainPanel()
containsData()
isPlaying()
paint()
pauseForSeconds()
stepBackward()
stepForward()
toEnd()
toPauseMode()
toPauseMode()
toPlayMode()
toStart()
Xaxis
Xaxis()
getMinimumSize()
getPreferredSize()
paintComponent()
Label
Label()
getPos()
getTime()
TSCNotePanel
TSCNotePanel()
actionPerformed()
clear()
displayUnitNote()
paintComponent()
setSelectedUnit()
Yaxis
getMinimumSize()
getPreferredSize()
paintComponent()
Label
Label()
getPos()
getSEQNO()