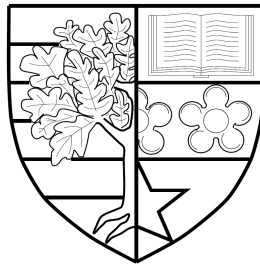


FROM FORMAL SPECIFICATION TO FULL PROOF:
A STEPWISE METHOD

by

Lavinia Burski



Submitted for the degree of
Doctor of Philosophy

DEPARTMENT OF COMPUTER SCIENCE
SCHOOL OF MATHEMATICAL AND COMPUTER SCIENCES
HERIOT-WATT UNIVERSITY

September 2018

The copyright in this thesis is owned by the author. Any quotation from the report or use of any of the information contained in it must acknowledge this report as the source of the quotation or information.

Contents

1	Interface	1
1.1	Inserting a specification	1
1.2	Checking ZCGa	4
1.3	Checking ZDRa	5
1.4	Skeletons	5
1.4.1	General Proof Skeleton	5
1.4.2	Isabelle Skeleton	6
1.5	Output messages which could occur	9
1.6	Conclusion	9
	Bibliography	11

List of Tables

1.1	Messages which could appear in the user interface and their meanings.	9
-----	---	---

List of Figures

1.1	Example of how to start the interface for the a toolkit for checking various degrees of correctness for Z specifications (ZMathLang) framework using the terminal.	1
1.2	This figure shows the steps to open an existing specification to be imported into the user interface.	2
1.3	This figure shows the pop up window that comes up when a user is asked which specification they wish to insert. In our example we would like to import a file named "fullexample.tex".	3
1.4	This figure shows an imported in the user interface which is shown in the main panel on the left.	4
1.5	An example of how to check the specification for ZCGa correctness (left) and the message which appears when the specification is ZCGa correct (right).	4
1.6	An example of how to check the specification for ZDRa correctness (left) and the message which appears when the specification is ZDRa correct.	5
1.7	This part of the user interface shows the steps to create a general proof skeleton from the users loaded specification.. . . .	6
1.8	A new file appears named skeleton in the same directory as the users specification. This is the automatically generated skeleton.	6
1.9	To create an Isabelle skeleton then ' <i>Isabelle Skeleton</i> ' should be selected from the GPSa sub menu on the user interface.	7

1.10	New buttons which appear on the right hand side of the interface if the user chooses to create a general proof skeleton or an Isabelle skeleton.	7
1.11	By clicking ‘ <i>Show Isabelle skeleton</i> ’ a box will be displayed in the user interface showing the isabelle skeleton.	8
1.12	This figure shows the steps to automatically fill in the Isabelle skeleton. The user can choose the ‘ <i>FillInIsaSkeleton</i> ’ from the GPSa sub menu.	8

makeglossaries HWThesis

Chapter 1

Interface

The interface was designed so that the steps to convert a Z specification to a fully proven specification would be easier to complete by the user. It made the ZMathLang process more user friendly then by just typing commands in a terminal. Full details for the user interface and all its functions can be found in Mihaylova's user guide [2]. This chapter only explains the process needed to translate a specification into a full proof, other steps such as writing the specification in the first place and outputting pdf using the interface can be found in the manual.

1.1 Inserting a specification

To use the ZMathLang framework through the interface the user can download the files from [1] then using a terminal run the interface program by typing `python Interface.py`, figure 1.1 shows an example of this.

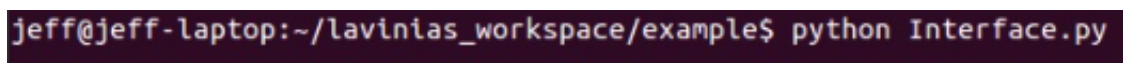
A terminal window with a dark background and light-colored text. The prompt is 'jeff@jeff-laptop:~/lavinias_workspace/example\$' and the command entered is 'python Interface.py'.

Figure 1.1: Example of how to start the interface for the ZMathLang framework using the terminal.

An example of the interface can be seen in figure 1.2. Depending on the operating system the interface may look slightly different however the main panels and buttons will be the same. The main menu bar is at the top left had side and the main panel is on the left. There is a messages panel on the right top side which displays any

messages when checking for correctness and converting to skeletons. To check a specification the user can click *file* then *open* from the main menu as shown in figure 1.2.

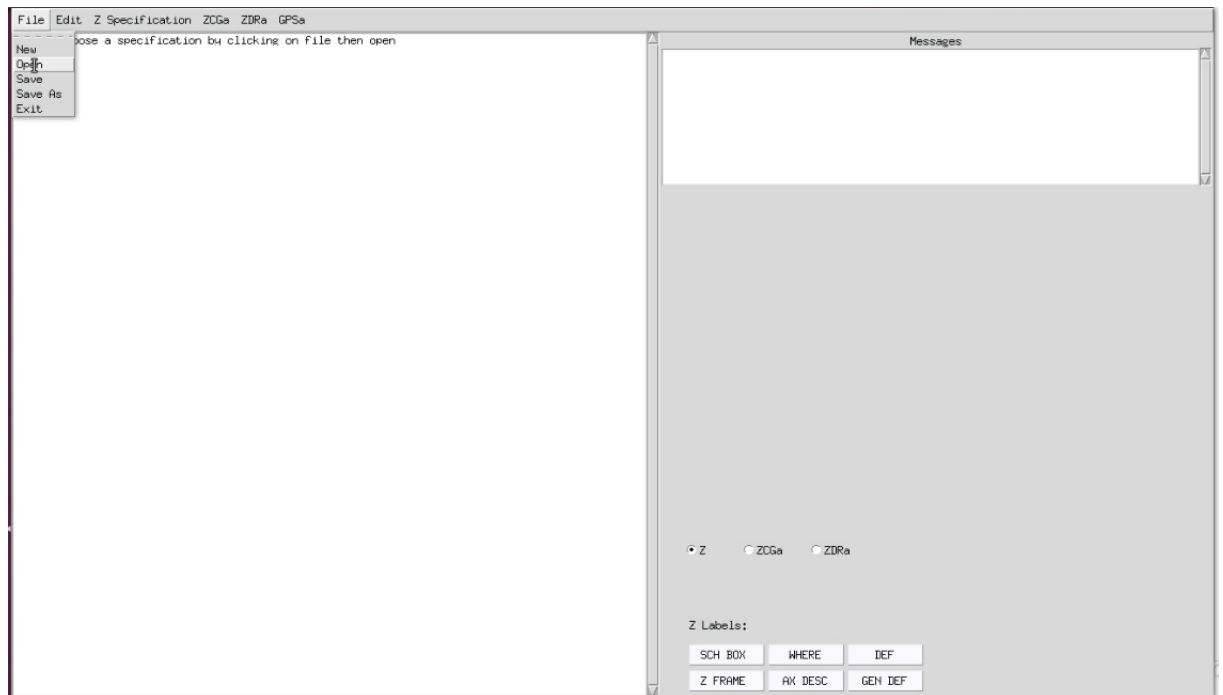


Figure 1.2: This figure shows the steps to open an existing specification to be imported into the user interface.

A pop up box appears asking the user to locate the specification they would like to translate. An example of the pop up box is shown in figure 1.3.

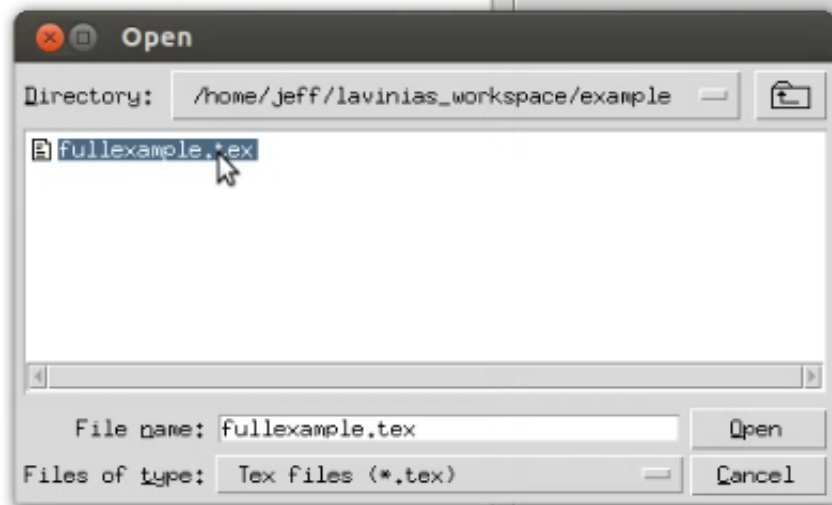


Figure 1.3: This figure shows the pop up window that comes up when a user is asked which specification they wish to insert. In our example we would like to import a file named "fullexample.tex".

Once a specification has been chosen then the specification should appear in the panel on the left hand side. Figure 1.4 shows an example of this. Note that no messages appear yet in the messages box.

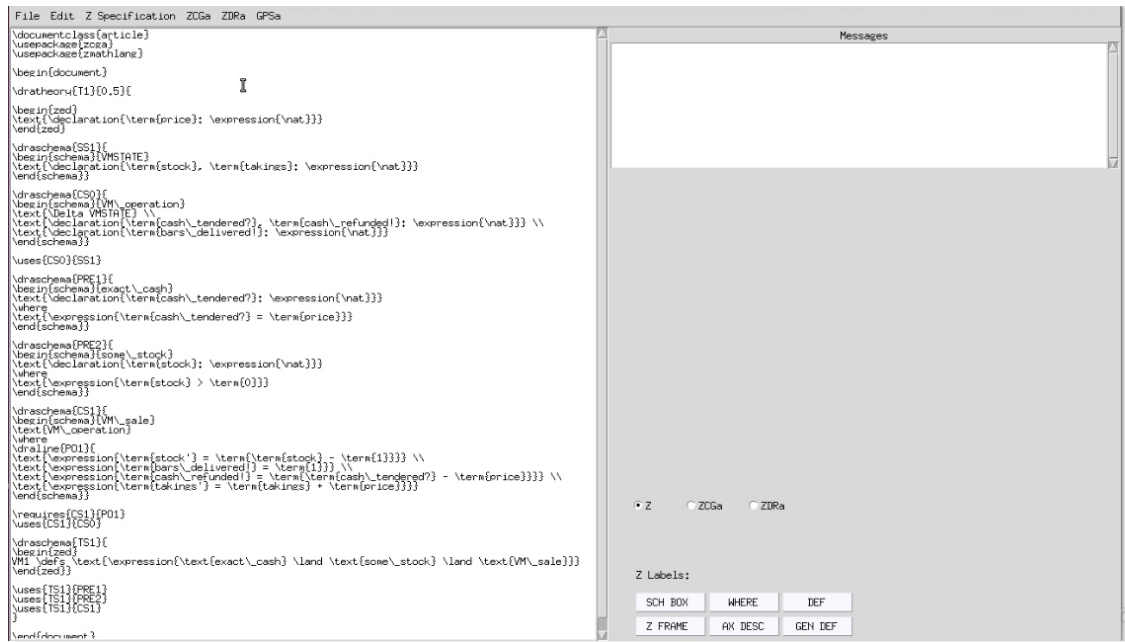


Figure 1.4: This figure shows an imported in the user interface which is shown in the main panel on the left.

1.2 Checking ZCGa

To then check for Z Core Grammatical aspect (ZCGa) correctness the specification loaded into the interface must be ZCGa annotated.



Figure 1.5: An example of how to check the specification for ZCGa correctness (left) and the message which appears when the specification is ZCGa correct (right).

To check for ZCGa correctness the user can click on the *zcga* button from the top menu and then click on *Zcga Check*. If the specification is grammatically correct then a message appears in the message box in the top right of the interface (n see figure 1.5).

1.3 Checking ZDRa

To check for Z Document Rhetorical aspect (ZDRa) correctness the specification loaded into the interface must be labelled with ZDRa annotations (this can be with or without ZCGa annotations).

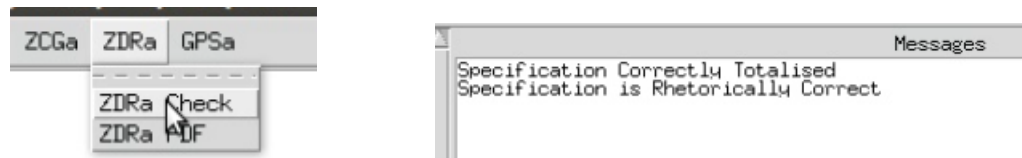


Figure 1.6: An example of how to check the specification for ZDRa correctness (left) and the message which appears when the specification is ZDRa correct.

To check for ZDRa correctness the user can click on the *ZDRa* button in the top menu of the interface and then on the *Zdra Check* button. If the specification is ZDRa correct and the specification has been correctly totalised then a message confirming this appears in the message box. Figure 1.6 shows both of these actions.

1.4 Skeletons

The user may also want to create a general proof skeleton, Isabelle skeleton and fill in the Isabelle skeleton using the Interface. This section explains how this may be done.

1.4.1 General Proof Skeleton

To create a general proof skeleton from the users ZDRa annotated and correct specification. The user will need to input their specification into the interface, check the specification for ZDRa correctness then click on the *GPSa* button in the top menu and choose *Proof Skeleton* from the drop down menu. An example of this is shown in figure 1.7.

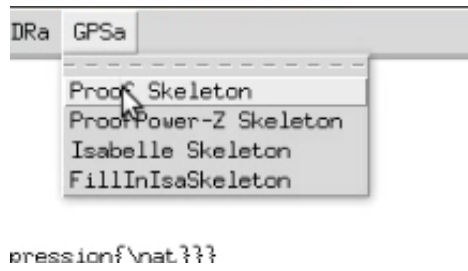


Figure 1.7: This part of the user interface shows the steps to create a general proof skeleton from the users loaded specification..

A new file should then appear in the same folder as your `Interface.py` program (see figure 1.8).



Figure 1.8: A new file appears named skeleton in the same directory as the users specification. This is the automatically generated skeleton.

The user may then open this file using an external text editor or they may view it in the interface itself. When creating a general proof skeleton a message appears in the Messages box saying **Skeleton Created**, a button also appears underneath the message box saying **Show Skeleton** (see figure 1.10). By clicking on this new button the general proof skeleton can be opened within the Interface.

1.4.2 Isabelle Skeleton

After creating a general proof skeleton the user may want to take the translation one step further and create an Isabelle proof skeleton. Again, to do this the specification must be labelled with ZDRa and be ZDRa correct. The user may then click on the *GPSa* button in the top menu and then the *Isabelle Skeleton* button in the drop down menu as shown in figure 1.9.

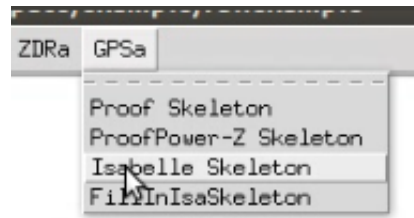


Figure 1.9: To create an Isabelle skeleton then '*Isabelle Skeleton*' should be selected from the GPSa sub menu on the user interface.

If all is correct a message saying **Isabelle Skeleton Created** and a new button should appear under the message box with the text **Show Isabelle Skeleton** as seen in figure 1.10. A new file will be produced and automatically saved in the same directory as the interface with a *.thy* extension.



Figure 1.10: New buttons which appear on the right hand side of the interface if the user chooses to create a general proof skeleton or an Isabelle skeleton.

The user may then open the Isabelle skeleton using an external text editor such as Jedit or Isabelle itself or they may choose to open the Isabelle Skeleton within the interface by clicking on the **Show Isabelle Skeleton Button**. An example of a pop-up window showing the Isabelle skeleton in the user interface can be seen in figure 1.11.

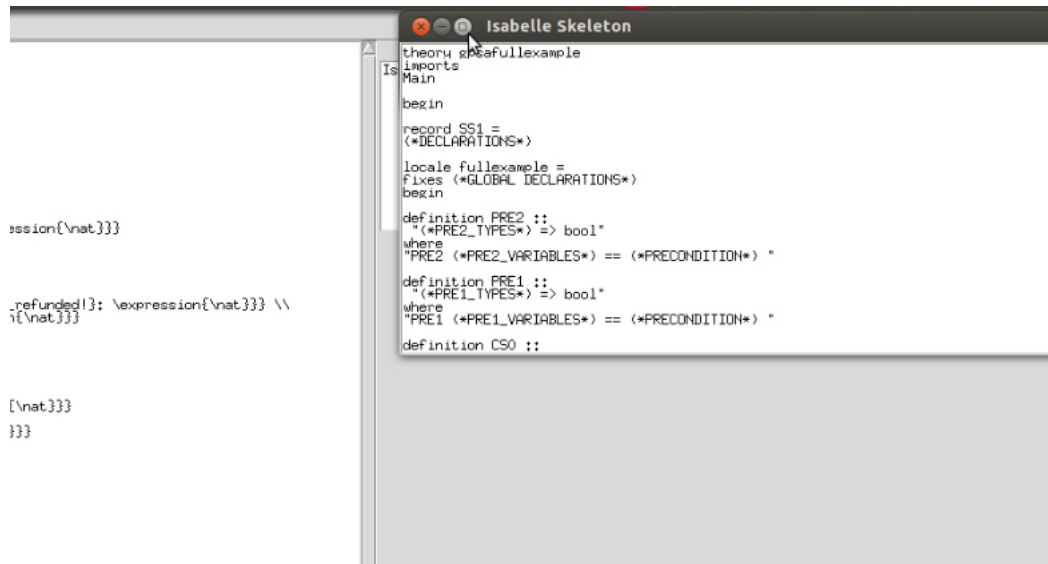


Figure 1.11: By clicking ‘*Show Isabelle skeleton*’ a box will be displayed in the user interface showing the isabelle skeleton.

The user may even wish to go as far as filling in the Isabelle skeleton. To do this the Isabelle skeleton will need to be created first and the specification loaded into the interface must also be annotated with ZCGa. The user can then click on GPSa in the top level menu and then *FillInIsaSkeleton* (see figure 1.12). If the skeleton is not in the same directory as the interface or has been renamed then the interface will ask for the user to locate the Isabelle skeleton in a similar way to opening a specification (figure 1.3).

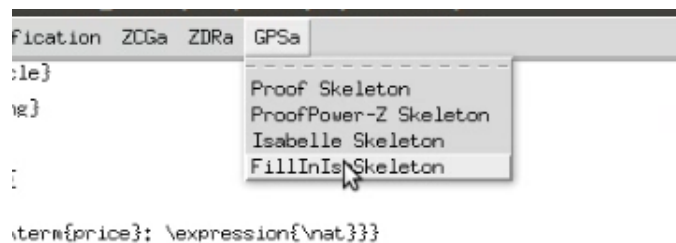


Figure 1.12: This figure shows the steps to automatically fill in the Isabelle skeleton. The user can choose the ‘*FillInIsaSkeleton*’ from the GPSa sub menu.

The user may then wish to open the filled-in isabelle skeleton externally or in the interface the same was as they opened the skeleton in the previous step.

1.5 Output messages which could occur

Sometimes there is an error in any of the actions previously described in this chapter. The message box not only tells the user what has been successful but it also gives the user information if there has been some error in the action they were trying. Table 1.1 shows all the possible messages which can appear in the message box along with an explanation about the message.

Text in message box	Explanation
Specification Correctly Totalised	All preconditions in the specification have a totalising condition
Warning! Specification not correctly totalised	Not all preconditions have an alternative output (skeleton can still be created) (<i>see chapter ??</i>)
Specification is Rhetorically Correct	Specification is ZDRa correct (<i>see chapter ??</i>)
Skeleton Created	General proof skeleton has been successfully created
Isabelle Skeleton Created	Isabelle skeleton has been successfully created
Isabelle Skeleton successfully filled in	Isabelle proof skeleton has been filled in using ZCGa text
Please convert specification into Isabelle Skeleton first	Covert the specification into an Isabelle skeleton first before filling it in (<i>see chapter ??</i>)
Please select your isabelle skeleton:	Please locate the Isabelle skeleton which you wish to be filled in (<i>see chapter ??</i>)
Please convert specification into GPSa first	Can not find the general proof skeleton (<i>see chapter ??</i>)
Loops in reasoning Can not create Skeleton	Specification is not ZDRa correct and the skeleton can not be created (<i>see chapter ??</i>)
Spec Grammatically Correct	The specification has passed ZCGa checker
Spec Grammatically Incorrect Number of errors: 2	The specification has failed ZCGa correctness and has 2 errors (<i>see chapter ??</i>)

Table 1.1: Messages which could appear in the user interface and their meanings.

1.6 Conclusion

This chapter has described how a user of the ZMathLang framework can use the implemented user interface to assist them with checking for grammatical and rhetorical correctness. The user interface also gives a clear and easy way to translate the specification into a general proof skeleton, Isabelle skeleton and filling in the Isabelle

skeleton without using the user unfriendly terminal. The interface also allows the user to view the documents automatically produced from the annotated specification.

Bibliography

- [1] L. Burski. Zmathlang. <http://www.macs.hw.ac.uk/~lb89/zmathlang/>, Jan 2016.
- [2] M. Mihaylova. ZMathLang User Interface User Manual. Intern User Manual, 2015.