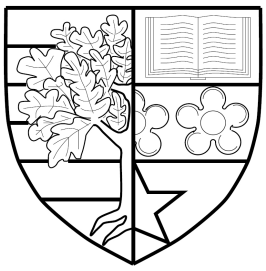
FROM FORMAL SPECIFICATION TO FULL PROOF: A STEPWISE METHOD

by

Lavinia Burski



Submitted for the degree of
Doctor of Philosophy

Department of Computer Science
School of Mathematical and Computer Sciences
Heriot-Watt University

March 2016

Abstract

Write the abstract here.

Contents

Acronyms

ZCGa Z Core Grammatical aspect.

ZDRa Z Document Rhetorical aspect.

Chapter 1

Conclusion and Future Work

In this chapter we discuss the current development of MathLang framework for Z specifications (ZMathLang) and it's future works. We also conclude a comparisson between ZMathLang framework to other ststems. Finally in section we give add concluding thoughts to this thesis.

1.1 ZMathLang Current and Future Developments

1.1.1 Current Developmets

The research on ZMathLang was started in 2013 and provides a novice approach to translating Formal specification to theorem provers. With this approach the gradual translation of the formal specification document is made via "aspects". Each aspect checks for a different type of correctness of the formal specification and output different products in order to analyse the system. Moreover, the annotation of the formal specification document should not require any expertise skills in the language of the targatted theorem prover. The only expertise needed for the annotations include the expertise of the formal specification document.

The ground basis of the MathLang framework for mathematics (MathLang) framework were studied by Maarek, Retel, Laamar and various other master and undergraduate students under the supervision of F.Kamareddine and J.B. Wells. This thesis presents the ground basis of the ZMathLang framework which uses the

methodology of the MathLang framework. The ZMathLang framework has taken the idea of breaking up the translation path from a document into a theorem prover and taking it through a grammar correctness checker, a rhetorical correctness checker, a skeleton into a proof. All the theory and implementation of the ZMathLang aspects have been developed and described in this thesis.

1.1.1.1 Other Developments

An extension to ZMathLang has started being developed by Fellar [2], [1] which takes the concept of ZMathLang and adds object orientatedness to it. With this, ZMathLang has the potential to translate not only Z specifications but object-Z specifications as well.

This thesis presents a very basic user interface to use with ZMathLang. Further developments on the user interface has been expanded during an internship by Mihaylova [4], [3]. The expansion on the user interface allows users to load and write their specifications. As well as going through each of the correctness checks, viewing the various graphs and skeletons all in one screen.

1.1.2 Future Developments

The future developments of ZMathLang have been discussed occasionally between students and supervisors during meetings. This section puts together and summarises these ideas and presents them to the reader in order to provide a general idea of future developments.

1.1.2.1 Automisation of the annotation

At present, the user needs to annotate their formal specification by hand using LaTeX commands before being check by the various correctness checkers. This sometimes can be a time-consuming task especially if the user isn't familar to LaTeX syntax. An advancement on this would be if the user would be able to visually see the Z specification as schema boxes (such as the compiled version of LaTeX) and then drag and highlight using mouse and buttons to annotate the specification with Z Core

Grammatical aspect (ZCGa) colours and Z Document Rhetorical aspect (ZDRa) instances. This idea could be done in a similar way to the annotations done in the original MathLang. Another way to ease the users input is if the labels would automatically label what user input. For example if the user labelled the variable '$v?$' as a term then all other variables '$v?$' would also be labelled a term automatically. This way the user wouldn't need to repeat the labels they have already done. This would drastically increse the workload for the user especially on very large specifications.

1.1.2.2 Extension to more complex proof obligations

The proof obligations described in this thesis are properties to check the consistancy of the specification. The current proof obligations for Z specifications are to give a flavour of what kind of properties to prove about the system and to ease the user in proving these properties. As mentioned before proof obligations for formal specifications is indeed a research subject in it's own right and more complex proof obligations can be developed to work alongside the ZMathLang framework. These proof obligation can come into the General Proof Skeleton aspect (Gpsa) part of the translation and follow through to the complete proof. If there are hint's or simple proof tactics to prove these properties then they can also be added to step 6 which would allow the user to get an idea of how to finish of the proofs.

1.1.2.3 Any formal specification to any theorem prover

This thesis describes how the ZMathLang framework can translate a Z specification into the theorem prover Isabelle. However, there are many other theorem provers which are prefered by certain users and ultimatly the ZMathLang framework should be able to translate from the Gpsa into a theorem prover of the users choice and not just be restricted to Isabelle. In this case steps 1 to 4 would be the same, regardless of which theorem prover the user wishes to translate to. The change would be made in step 5 when creating a skeleton of the specification in the chosen theorem prover. Other theorem provers which ZMathLang could transate to would be Mizar/HOL-Z/ProofPower-Z/Coq etc.

There are many other formal languages to write specifications in which could be another idea for future research. ZMathLang currently parses through Z specifcations however, further research could be done for ZMathLang to work on any formal language such as alloy, event B, UML or VDM. Investigation on whether the grammatical categories in the ZCGa or instances in the ZDRa would need adapting. Otherwise the current annotations would be suitable for any formal notation and only the implementation would need to be changed.

1.1.2.4 Informal specifications

A final future idea would be to combine parts of MathLang which handles mathematical documents written in part mathematics and part english and to translate informal specifications into theorem provers. With this idea, perhaps a TSa aspect would need to be adapted for informal specifications. So that a system specification written completely in english could be checked for ZCGa, ZDRa and ultimatly translated fully into a theorem prover.

1.2 Related Works

1.3 Conclusion

write conclusion chapter

Find year for atelier b proof obligation user manual

Bibliography

[1] D. Fellar, F. Kamareddine, and L. Burski. Using mathlang to check the correctness of specifications in object-z. In E. Venturino, H. M. Srivastava, M. Resch, V. Gupta, and V. Singh, editors, *In Modern Mathematical Methods and High Performance Computing in Science and Technology*, Ghaziabad, India, 2016. M3HPCST, Springer Proceedings in Mathematics and Statistics.

[2] D. Feller. Using mathlang to check the correctness of specification in object-z. 2015.

[3] M. Mihaylova. Zmathlang user interface internship report. 2015.

[4] M. Mihaylova. Zmathlang user interface user manual. 2015.