A Specification Construction Unit-based editor for Z

Hiroshi Ishikawa
Department of Information Processing Engineering
Fukuyama University
1 Gakuen-cho, Fukuyama-shi, Hiroshima 729-0292, Japan
h-ishika@fuip.fukuyama-u.ac.jp

Abstract

This paper proposes a formal specification supporting tool. It is designed on the basis of the specification reusability and the convenience to construct specification repository. The tool, called ZEditor, is designed for supporting the Z specification language. Two remarkable features of ZEditor are as follows. Firstly, we focus attention on some Z specification construction units so that ZEditor provides several specified input screens for them. Specifications described by ZEditor can be saved as the \LaTeX\ format or the XML one from each input screen. In each input screen, syntax and type checking of specifications can be also available. Secondly, the author can construct and edit an appropriate specification from some specification construction units which passed syntax and type checking. ZEditor will make it possible to describe specifications easily, to reuse correct specification construction units, and to support constructing specification repository.

1. Introduction

Describing correct specifications, reusing specifications, and constructing specifications repository are significant issues on software development to product safety and high quality software systems.

In order to deal with such issues, we propose a formal specification supporting tool designed for the Z specification language[3]. Our tool, called ZEditor[1], is now under development, however, some features have been implemented in Java. Several features of ZEditor are designed but under construction.

There are several tools for supporting the Z specification language, however, ZEditor has two remarkable features as follows: several specified input screens corresponding to \(Z\) specification construction units and an appropriate specification constructing feature from specification construction units.

These features will make it possible to

- describe specifications with ease
- reuse correct specification construction units
- construct specification repository with ease.

The remainder of the paper is organized as follows: Section 2 briefly introduces the technical background of \(Z\) notation and XML. Section 3 provides the design and current status of our tool. Section 4 and 5 mentions related work and conclusion respectively.

2. \(Z\) and XML

\(Z\) is a state-oriented formal specification language based on set theory and predicate logic. The specification written in \(Z\) has some construction units (Basic Type, Type Definition, Axiom, Generic Constant, Schema, and so on).

XML (extensible Markup Language) is a currently popular language for creating self-describing documents. Using XML, the author of a document can design a tag structure to suite the needs of the document. XML schema and DTD (Data Type Definition) are used to define a standard exchange format for \(Z\) specifications. Several XML schema and DTD for \(Z\) specification are already proposed. In this paper, we used the XML schema and DTD defined in [4].

3. ZEditor

The tool we propose, called ZEditor, allows creating and editing of \(Z\) specifications, saving the documents as \LaTeX\ format of XML format, checking their syntax and types. We focus attention on some \(Z\) specification construction units, therefore, ZEditor provides several specified input screens, for example, Basic Type, Axiom, Generic Constant, Schema, and so on. Specifications described by our editor are used not only as units for constructing an appropriate specification but also as input data for syntax and type checking facilities. Some non-standard symbol are used to describe \(Z\) specification. To clear the problem, ZEditor provide a support facility to input \LaTeX\ commands corresponded such symbols. Moreover, the author can construct and edit an appropriate specification from some specification construction units which passed syntax and type checking.

In consideration of multi-platform performance and the use of the technology of XML, ZEditor is implemented in Java.

The remainder of this section is mentioned what we have done and what we will do about our tool.

3.1. What we have done

ZEditor has several specified input screens corresponded to \(Z\) specification construction units. We have already provided four screens, that is, Basic Type, Axiom, Generic Constant, and Schema. Moreover, it provides a support facility to input \LaTeX\ commands corresponded mathematical symbols. In our tool, each specification construction unit is usually
represented by standard symbols. The author needs a non-standard symbol, however, he/she uses \LaTeX command corresponding to it. In that case, the support facility is useful.

Figure 1 is a snapshot for schema input screen. The author has to select the type of schema (state definition, initial state definition, or operator definition). Schema’s name, pre-condition (Decl-Part), and post-condition (Pred-Part) are described in the corresponding text areas.

The input data is saved as the \LaTeX format or the XML one. Specification in these file formats are used for input data for syntax and type checking, creating high quality document and browsing on the web.

The facilities for syntax and type checking are designed but under construction.

3.2. What we will do
One of the remarkable features of ZEditor is to construct and edit an appropriate specification from some specification construction units which are described and done syntax and type checking. It seems troublesome to describe a specification in such way, however, this method is based on the policy of the reusability and we can obtain the correct specifications easily.

Figure 2 shows the notion of the feature. Some specification construction units are described and saved in files (left side). The author can construct and edit an appropriate specification from them. Each selected unit is displayed in a small window and gathered in a special window called workspace window (right side). Therefore, an appropriate specification is displayed in the workspace window as gathering small windows (in another word, gathering units). Type checking for the specification in the workspace window is available to click the button labeled "type check". ZEditor can also save the specification in the \LaTeX or the XML format.

4. Related Work
There are several tools to support the Z specification, the notion and the design of our tool is motivated by ZTC (Z type checker)\cite{2} and ZML (Z family XML web environment and UML projection facilities)\cite{4}.

ZTC is one of well-known tools to check Z specification with \LaTeX as an input format. Specifications described by our tool can be written with \LaTeX to use ZTC and to create high quality documents using \LaTeX system.

ZML is designed how formal object modelling techniques facilities the XML application development and how XML technology helps formal/visual software design process. This approach, in particular, the design of XML schema and DTD for Z is of some help for our tool.

5. Conclusions
We have proposed a Z specification supporting tool. It is designed on the basis of the specification reusability and the convenience to construct specification repository.

The current status of our tool, some specified input screens, a supporting facility, a saving facility with file format have been implemented in Java. Some features mentioned in Section 3.2 are now under construction. We will implement them in the near future.

References