Geosci. Model Dev. Discuss., 1, S45–S48, 2008
www.geosci-model-dev-discuss.net/1/S45/2008/

Geoscientific
Model Development
Discussions

Interactive comment on "Spud 1.0: generalising and automating the user interfaces of scientific computer models" *by* D. A. Ham et al.

Anonymous Referee #1

Received and published: 31 July 2008

General comments

The paper describes well the benefits of Spud what comes to the automated building of a graphical user interface (GUI) for scientific simulation software. This is very well laid out and hence matches the scope of GMD. This is a useful paper.

Nevertheless, with myself not being that familiar with formal languages but rather being placed within the numerical model developer fraction, I miss a few deeper explanations. What I would like to have explained in a clearer way is how the system – including all files and program components starting from definition of the grammar file over Diamond up to the simulation code interacting with libspud – is built up. Preferably this could be done in a graphical way.

Especially the part of the interaction with the (most likely already existing) scientific code for me in my opinion is not discussed in a sufficient way. From the paper I understand that this in the end is done by libspud. From the added manual I found documentation on a Fortran/C/C^{++} interface. From that I conclude that the simulation software has to be adapted and linked to this library. What you call *in-memory representation* I understand as introduction of a new data storage scheme within the code. Lots of scientific software already have certain data storage schemes built in, like for instance the compact row storage (CRS) scheme, that are optimized with respect to the applied numerical schemes. What I would like to have included are some lines that address the effort and techniques involved in adapting existing code in order to play together with Spud/libspud. In this context I completely agree that adding input parameters is really easy on the Spud-side of life. Nevertheless, with respect to your more general claim in the abstract: "*This is combined with an automated graphical user interface which guides users to create valid model inputs based on the grammar provided, and a generic options reading module which minimises the development cost of adding model options.*" I though do not immediately see what benefits there are for introducing new parameters within the code. Could you please, also with respect to the similar claim made in **9 Conclusions**, express how the effort of altering the simulation code is positively affected by using Spud?

Finally, I was not able to find any hint under which licensing scheme Spud itself or the components used within fall. I know you have a `LINCENSE` file in the supplement within the sub directory `diamond` . Nevertheless, as I understood, this is explicitly demanded to appear in the main paper (`http://www.geoscientific-model-development.net/gmd_journal_white_paper.pdf`)

Specific comments

Abstract

"*... generic option reading module*" is this equivalent to the library libspud?

1 Introduction

You are using the terms *parameter files* and *input files* and later on (in **2 Problem description languages**, **3 Components of the Spud system** as well as in **4 Languages specification in Spud**) also *options file* and *problem description files*. To me the meanings are not immediately apparent. Are these synonyms? If not, please explain (perhaps within a graph, as suggested before).

4 Languages specification in Spud

4.2. Comments and documentation

With myself being guilty of adding tons of undocumented features into the codes I am contributing to and from the user response to that, I claim that comments in GUI's are beneficial (and - as you correctly claim - better than nothing) but not to all extend represent a replacement of a proper documentation in form of a printable manual. So, I would draw a clearer line between *comments* and *documentation*

8 Applicability to other models

"*It has been emphasised throughout this paper that Spud is model-independent with only the schema varying between models.*" Really only the schema, or is there perhaps also something to be done on the code side (see question concerning internal data storage schemes before)?

Figures

To me **Fig. 3** seems to be an in size enhanced representation of the left frame in **Fig 2** and hence redundant. Also, those two figures seem to be (at least on my screen and my printout) of quite coarse resolution (72 dpi screen-shots?). If it does not impose too much difficulty, a somewhat higher resolution would make it far more readable also for people with not so good eyes (including myself). And, as mentioned twice before, a general graph introducing the whole concept of Spud would be necessary to improve the paper.

Supplement

It would be good to include a toy simulation model (just a few lines of Fortan or C$^{(++)}$

code) and a primitive schema file in order to demonstrate how it works. This might be a useful blueprint for people (like me) to get started with Spud. I found some subdirectory under `diamond` as well as `src` named `tests` in the supplement. Is this what in principle I am searching for?
