Contract Specifications for
"Display Software"
June 4, 2009

1 General Requirements

1.1 Documentation standards

All changes to existing functions and all new functions must be accompanied by documentation formatted to the standard found in Schnute et al. (2008), which describes a complete user's guide (hereinafter UG) for the R package **PBSmodelling**. Specifically, Sections 5 and 6 of the UG must be updated to reflect software modifications, and code must reflect the written documentation. Additionally, R's object documentation files (`*.Rd`) must be revised to more adequately describe the details of all functions to appear in Sections 5+6 of the UG.

Schnute, J.T., Couture-Beil, A., Haigh, R., and Egeli, A. 2008. PBS Modelling 2.00: user's guide revised from Canadian Technical Report of Fisheries and Aquatic Science 2674: v +146 p. Last updated October 23, 2008. (See also `PBSmodelling-Updates.pdf` for functions not currently documented in the UG.)

1.2 Set up SVN system for `PBSmodelling`

As **PBSmodelling** will be altered by an offsite contractor, the latter will set up a functional SVN (subversion numbering) system for **PBSmodelling** on the Google Code site: http://code.google.com/p/pbs-software/.

1.3 Documentation describing '`presentTalk`'

Produce a separate, detailed developer's guide to document the structure and function of the display software called '`presentTalk`'. This guide will ultimately be added to `DeveloperGuide.pdf` that appears in the R package document directory `...\library\PBSmodelling\doc`.

1.4 Offsite visit with contract administrators

The successful contract applicant is required to meet with the contract administrators (at an offsite location to be mutually determined) prior to contract completion. The purpose of this meeting is to demonstrate the functionality of the display software and project management functions, and their integration into **PBSmodelling**. Periodic telephone conversations will likely be necessary in addition to the offsite visit.

2 Support for Lectures and Workshops

Section 6 of the UG introduces the concept of software that facilitates the presentation of lectures and workshops entirely from the R environment. This software, called `presentTalk` and detailed in Appendix B, offers five distinct tag lines that combined in a *talk description file* run the R presentation.

2.1 Write the code for '`presentTalk`'

The current code for `presentTalk` is convoluted and buggy. This stems in part from earlier design specifications that contemplated only moving forward through the talk. Once the specifications changed to allow backward movement also, we quickly realized that the design was flawed.

Jon Schnute believes that coding can be a lot cleaner if we implement an object-oriented approach. In R, this is achieved through S4 classes. Depending on the class attributes of an object, algorithms can be designed to execute procedures that remain consistent for each class. According to Chambers (2008, Chapter 9) this leads to more trustworthy software.

Ideally, it should be straightforward to add new tags and/or modify their behaviour. Also, make suggestions for changing the specs if that seems desirable. (We need not ensure backward compatibility for this function, which is relatively new to **PBSmodelling**.)

Chambers, J.M. (2008) Software for Data Analysis: Programming with R. Statistics and Computing series, Chambers, J., Härdle, W., and Hand, D. (eds.) Springer Science+Business Media, New York, NY. 498 pp.

3 Functions for Project Management

Review and revise the functions that appear in Section 5 of the UG, including hidden functions upon which they rely. Whenever possible, all functions should have the ability to be called from source code (independent from GUIs) as well as smart enough to be called from GUIs.

3.1 Project options

Section 5.1 of the UG outlines the functions that currently exist for controlling project options. Some of these functions have been re-written in the R package **PBSadmb** (see 4.1). The contract holder needs to rationalize the functionality of these various codes and produce a single suite of functions that can be used in any package.

At the heart of project options will be a single control file (`PBSoptions.txt`) using PBS formatting outlined in Section 3 of the UG. The **PBSmodelling** function `readList` converts this ASCII file into a single list object in R. Every project should contain a control file; each one could be different from all the others, or they could all be the same.

There should be some thought given to additional control files within one project. For instance, `PBSoptions.txt` might occur alongside `ADopts.txt` (part of **PBSadmb**) with redundant information. Rules on precedence will need to be formulated.

3.2 Project management utilities

Section 5.2 of the UG outlines functions that facilitate project management. These functions often deal with common prefixes and suffixes. Again, where possible all functions should have the dual ability to run from source code as well as GUIs. For instance, the function `findPrefix` cannot always be run in source code as it tries to set GUI values. Hence, another

function `getPrefix` was added later to run from source code. There should only be one function that can work in both environments.

In addition to `getPrefix`, there are a number of new project management functions outlined in the vignette `PBSmodelling-Updates.pdf`, located in the **PBSmodelling** package overview. These too need to be rationalized with functions in Section 5 of the UG and with functions in **PBSadmb**.

4 Miscellaneous Items

4.1 Package compatibility

Check that all packages in the PBS series remain compatible. Specifically, check those packages like **PBSadmb** (at http://code.google.com/p/pbs-software/) that rely directly on **PBSmodelling**. Mention has already been made in (3) about reducing code redundancy and merging functions to offer capability in both source code and GUI environments.

Distinguish functions that have broad applicability (and belong in **PBSmodelling**) and those that are specialized (and should remain in local packages).

4.2 Rubbish removal

Flag files and folders in **PBSmodelling** that are leftovers from package evolution and have no apparent utility. For instance, do we still need the file `LoadCSetup.pdf` in the directory `...\library\PBSmodelling\doc` ?

5 Deliverables and Milestones

Recommended sole source: Alex Couture-Beil, 2545 Napier St., Vancouver, BC, V5K 2W4

Contract period: 2009-06-04 to 2009-09-04.

Location of work: recommended sole source (off-site).

At a cost of $30/h, the proposed work would take an estimated 200 hours to complete, for a contract value of $6000.