Sergei Bolotin et al.: Development of a New VLBI Data Analysis Software, IVS 2010 General Meeting Proceedings, p.197–201
http://ivscc.gsfc.nasa.gov/publications/gm2010/bolotin.pdf

Development of a New VLBI Data Analysis Software

Sergei Bolotin, John M. Gipson, Daniel S. MacMillan

NVI, Inc./NASA Goddard Space Flight Center
Contact author: Sergei Bolotin, e-mail: sergei.bolotin@nasa.gov

Abstract

We present an overview of a new VLBI analysis software under development at NASA GSFC. The new software will replace CALC/SOLVE and many related utility programs. It will have the capabilities of the current system as well as incorporate new models and data analysis techniques.

In this paper we give a conceptual overview of the new software. We formulate the main goals of the software. The software should be flexible and modular to implement models and estimation techniques that currently exist or will appear in future. On the other hand it should be reliable and possess production quality for processing standard VLBI sessions. Also, it needs to be capable of processing observations from a fully deployed network of VLBI2010 stations in a reasonable time. We describe the software development process and outline the software architecture.

1. Introduction

The current VLBI data analysis package, CALC/SOLVE, has served us for more than thirty years. It is the most widely used data analysis software in the geodetic/astrometry VLBI community. CALC/SOLVE software is used for data preparation as well as analysis of VLBI observations. Database output from CALC/SOLVE is input to other VLBI data analysis packages. The current CALC/SOLVE system consists of more than a hundred individual programs most of which are written mostly in FORTRAN90. Many significant results were obtained with CALC/SOLVE in various branches of astronomy, geodesy, and geophysics.

However, the software has certain limitations. CALC/SOLVE was developed as a set of utilities around the Mark III database handler, which was created at the end of the 1970s and then was frozen. Computer hardware limitations that were in effect at that time still constrain the database access library and procedures that use it. On the other hand, since CALC/SOLVE is a production system in the VLBI data flow (preparation of *Version 4* databases and *NGS* files), adding new models and features was usually done as patches over existing code or even as 'tricks' to keep the existing general structure of the software. As a result, it is hard to maintain and to extend the current CALC/SOLVE system. Moreover, the forthcoming VLBI2010 system will demand flexibility and performance that the current realization of CALC/SOLVE cannot satisfy.

Therefore, in 2007 the VLBI group at the NASA Goddard Space Flight Center started active work on developing a new VLBI data analysis system. With the help of the geodetic VLBI community new formats for data storage were developed. The NetCDF library was chosen for storing/retrieving of data, corresponding file specifications were determined, and in July 2009 the first conversion from Mark III database to NetCDF files was performed.

The process of software development started in August 2009. We selected the *evolutionary* model for the software production process [1]. This process consists of the following, overlapping-in-time activities: *Conceptualization*: formulating a statement of the problem and establishing core requirements; *Analysis*: developing a model of the system's behavior, describing the functionality

and performance of the system and necessary resources, producing scenarios of system's behavior; *Design*: creating the architecture for the evolving implementation and establishing common tactical policies, architectural planning, release planning; *Evolution*: growing and changing the implementation through successive refinement, leading to the production system; *Maintenance*: managing post-delivery evolution.

In this article we will discuss *Conceptualization*, a part of *Analysis*, and an upper level of *Design* activities.

2. Conceptualization: What Do We Want from a New Version?

Goals for developing a new version of the software. The VLBI data analysis software is a *tool* for studying Earth rotation, constructing Celestial and Terrestrial reference frames, and investigating various geophysical phenomena. Therefore, the following primary goals have to be achieved with the new software: 1) reliability; 2) ability to easily maintain and extend the software; 3) high performance, and 4) data integrity.

Since the output produced by the software is used by many scientists throughout the world, we must be confident in results obtained. The new software should behave reliably and provide reliable results. Secondly, it should be easy to add new models, new estimated parameters or even types of observables. Further improvement of the software should not force changing of its general design. The third main goal is performance. The software must be efficient enough to process observations from the VLBI2010 system and/or from other techniques of modern geodesy in a reasonable time. User efforts in processing observations should be minimal. It is assumed that data analysis is performed by a group of users. The results of the work of one user should be transparently available to the whole group and the software should provide data integrity in such group work.

In addition, nothing should preclude the following additional goals: combining VLBI with other types of space geodesy observations; implementing different approaches of estimating parameters; supporting efficient interaction with other software; and allowing simulation analysis.

Core Requirements. Based on these goals we determined the core requirements of the software. The first group of requirements concerns reliability. The software should have quality control facilities. This could be test suites that check all the essential procedures of the software, a multilevel logging system, and an error reporting mechanism. The new software should support versioning of input data and solutions. It should be able to reproduce the current CALC/SOLVE results. The software should protect users from making blunders and be capable of automated feedback to stations and correlators. We will pay special attention to data visualization. We will implement a modern graphical user interface with elaborated plotting system. Later we will add the ability to shift from one level of data to another one. For example, to check a specific point in an EOP series, the user could enter a session view, then drill down to some particular baseline, a scan, a group delay, or a fringe plot.

There are several requirements of the software development process. The software should be modular, flexible, and portable. The software development process should be performed with standard software development tools, be a team-oriented work, support multiple versions, and have a multilevel documentation system.

Performance issues can be divided into two parts: consumption of user time and computer time. To increase user performance, the software should be automated and one should be able to

use a script language to set up and run batch jobs. Also, the software should understand various formats of data and use standard network protocols for data transfer. CPU performance can be increased by switching from development to production mode. The part of the code that performs debug calculations and output could be turned off during compilation of a production mode of the software. A more significant increase in performance can be achieved by parallelization of calculations. We expect to use either OpenMP API or threads library on a multi-CPU computer. Alternatively, we could run it on a dedicated computer cluster using one of the standard network protocols.

To ensure data integrity in a multiuser environment the software should perform user administration with different user levels. There should be some file locking mechanism. In addition, the new software should monitor available system resources and resources consumed by each user.

3. Analysis: What Should the System Do?

Overview of the System. The software belongs to the data analysis type of software. It is not a time critical application. As with many similar data analysis softwares, it should read observations in predefined format(s), evaluate theoretical values corresponding to the observables, and with the Least Square Method estimate parameters that have influence on the observable values. In addition, it should perform "cleaning" of the observations: remove outliers, resolve anomalies inherent to a particular observing technique, reweight data, and so on.

There are no special design requirements for the hardware. The system should run on a conventional "personal computer" as well as on a dedicated workstation or even a hardware cluster.

Also, there is no specific requirement for the operating system (OS). We assume POSIX compliance; this could be one of the Linux distributions, Solaris, or FreeBSD. The software could be portable to these families of OS.

There are two prototypes which we will use in the development process. We have the current CALC/SOLVE system which performs the same tasks. We also have SteelBreeze, which could be used for prototyping the Object-Oriented Design and graphical user interface. In general, all time-tested algorithms and approaches realized in these prototypes should be used.

Data flow in the current CALC/SOLVE system is shown on the left in Figure 1. The right part of the figure displays the proposed data flow for the new software. The general structure of the data flow of the new software is kept the same, but there are some significant changes. The data files would be physically separated in two parts: observations and editables. This would prevent observations from unintentional modification by a user. The Mark III database handler format will be replaced by an open source NetCDF library. Several executables that are currently used in data preparation, `calc`, `pwxcb`, `xlog` and `dbcal`, will be replaced by one executable block. This executable block, *Editor of Observations*, is intended to interact with a user, while other blocks run mostly in a batch mode. On the other hand, to make the software structure modular, the block `solve` will be split into several independent parts for evaluating (O-C) and the partials, parameter estimation, and further handling of estimated parameters (e.g., writing reports, updating *a priori*). The format conversion filters and network transfer utilities are presented in a separate block.

Functionality of the System. Key function points, or abilities of the system to perform something useful for an end user, are the following: 1) Import observations into a local storage, check reliability of imported data and correct them if it is necessary and possible; 2) Import *a priori* data into a local storage, check them, export them in the appropriate formats; 3) Prepare

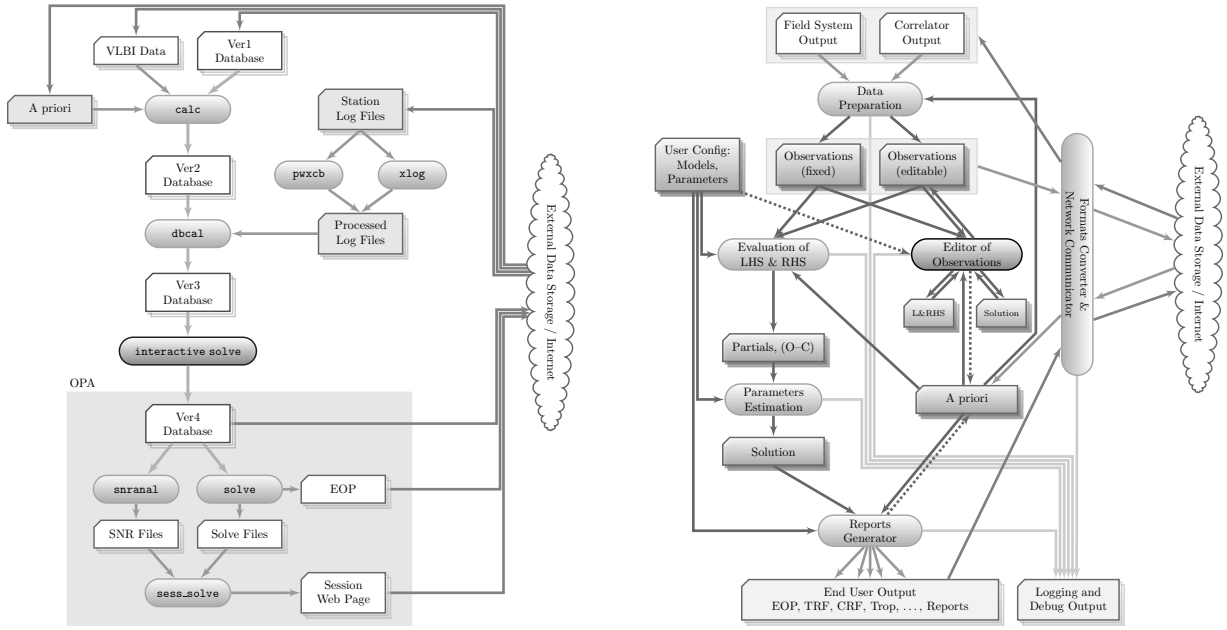

Figure 1. Data flow, current (left) and proposed (right).

newly imported observations for the analysis: check for ambiguities, outliers, clock breaks, new sources/stations, anomalous cable calibration data, perform reweighting, and so on; 4) Edit data; 5) Multi-session analysis: evaluate (O-C) and partial derivatives according to a predefined set of models; form equation systems; obtain a solution; 6) Generate reports and update files with *a priori* data; 7) Send reports to analysts and external data analysis centers; 8) Export checked and validated observations in appropriate formats.

There is another group of function points that are necessary to implement but are not interesting to an end user. They are called system function points. For the new VLBI data analysis software the system function points are: 1) report an error, 2) send and receive a file through a network connection, 3) parse a script for a batch run, 4) check data integrity, 5) monitor available resources (CPU, memory, hard disk space, network connections, open file descriptors), 6) lock a file before writing, unlock it after changes are finished.

4. Architecture Design: How Do We Develop the System?

Architecture Overview. According to the core requirements we performed a decomposition of the system into separate modules. Each module is a block of code that is loosely tied with other parts of the software. In Figure 2 the proposed system decomposition is shown. Each arrow represents a dependency which provides information (e.g., types, function calls, constants). Some blocks in the figure correspond to external software (FITS, NetCDF, LibZ, LibBz2, and Qt libraries) that will be used in the VLBI data analysis software.

Almost all parts of the software will be organized as a shared library, which will allow code reuse for various applications of the geodetic VLBI community.

Common Tactical Policies. Common software development policies are intended to stan-

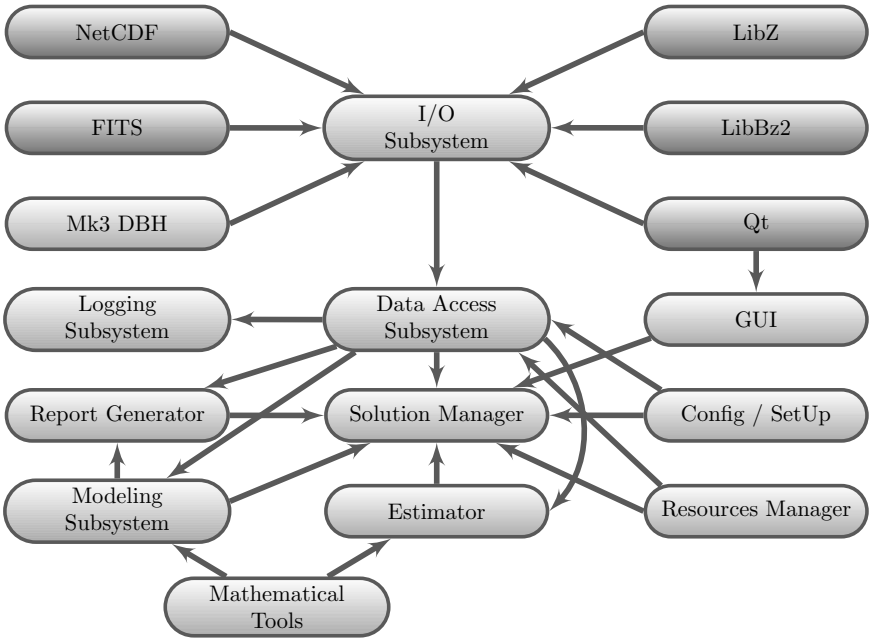

Figure 2. System decomposition.

dardize the development process inside a working group. The policies cover different aspects of the software building process. Using a common *programming style convention* increases readability of the source code for members of the development group. Applying a *test suite* for every essential part of the system assures software quality at the development stage and during distribution to external users. For the portability issues we will use *autogen/autoconf suites*, which are now standard de facto in the open source community. Proper use of these suites allows efficient porting software to a new operating system. Reference documentation will be generated automatically with the *Doxygen utility*. *A version control system* will support simultaneous access to different versions of the software. *Automatic backup* of the source tree will prevent hardware failure.

The Roadmap for future development. The current CALC/SOLVE system will be replaced with the new software piece by piece. We expect to have the first internal release of a replacement for interactive SOLVE by mid-2010. Initially, it will have the minimum functionality and later evolve into mature software that will be able to perform preliminary analysis and editing of new VLBI sessions. The first public release of this new software is planned for the end of 2010. In the spring of 2011 we are going to finish integration of current `pwxcb`, `xlog`, and `dbcal` functionality into the new software. Later, work on the replacement of multisession SOLVE and CALC will begin.

References

[1] G. Booch, Object-Oriented Analysis and Design with Applications. Addison-Wesley, 2^{nd} ed., 1994. ISBN 0-8053-5340-2.