ACCEPTING SOFTWARE INTO GRASS

Source Directory Definitions, Responsibilities, and Programming Standards

1. INTRODUCTION

Perhaps the most important quality of GRASS is its flexibility in support of research and development of both fundamental GIS processes and GIS applications. The GRASS community contains applications users who expect a given set of GIS functionalities to operate flawlessly. It also contains system developers who are very interested in programs which, though possibly rough and not quite foolproof, provide new algorithms and approaches to solving research problems. To accommodate this full spectrum of users GRASS software is distributed in several different categories reviewed below. Each category is associated with a different set of guidelines for acceptance. These guidelines are discussed.

2. MAIN (src)

2.1. Definition

1. Stable - no one is actively developing the code
2. Survival quality is proven - the program has existed as an ALPHA program for at least one year. All known bugs have been fixed.
3. Passes all of the criteria described for ALPHA programs.
4. OGI supported - OGI can make modifications without conflicting with original development group.

2.2. Discussion

Programs accepted into main become the primary responsibility of the Office of GRASS Integration (OGI). That office will facilitate the management of bug fixes, documentation rewrites and clarifications, and system-wide upgrades. It is therefore highly important that programs in this category be of generally high quality. Significant experience has shown that GRASS software only gets sufficient testing through months (and sometimes years) of use.

Note that, by the very definition of "main" code (stable), any OGI modifications will be limited to bug fixes or extremely minor modifications that do not require any extensive testing.

ONLY CODE THAT HAS SURVIVED A YEAR OF EXISTENCE AT THE ALPHA LEVEL IS ELIGIBLE FOR INCLUSION AT THE MAIN LEVEL.**

3. ALPHA (src.alpha)

3.1. Definition

1. Software has broad potential application. That is, the program does not meet the needs of simply one site, agency, or application. Such programs will be assigned to the CONTRIB software.
2. Meets GRASS programming and documentation standards (see PROGRAMMING STANDARDS section below).
3. Has been verified useful and operative by three independent user sites.
4. Development site supported. OGI will facilitate communication between the user community and the original development site.

3.2. Discussion

SRC.ALPHA will always be the most exciting category for those interested in the latest and greatest software. In the process of assembling releases, OGI will put most of their effort into this area. OGI's role will involve the following activities:

1. Maintain information on the status of a program being considered for inclusion in the next release (including status of below actions).
2. Ensure that software meets all of the standards outlined below in PROGRAMMING STANDARDS.
3. Coordinate the testing of the software by at least three different interested users working on three different UNIX environments. Every attempt will be made to ensure timely testing.
4. Coordinate communication between user community bugs, comments, ideas, and upgrades to the development sites.

Software included under this category will have been identified by OGI (and the GRASS Inter-Agency Coordinating Committee or GIACC) as being potentially included under main. OGI and the GIACC will also have the responsibility of determining what software has a broad user group potential (see src.contrib below).

** A number of SCS programs placed into CONTRIB moments before the 4.0 release must also be considered for inclusion within the 4.1 MAIN code.

4. CONTRIB (src.contrib)

4.1. Definition

1. Known to work for programmers
2. Accompanied by appropriate documentation
3. Proper naming
4. Not necessarily well tested
5. Potentially useful GRASS program
6. Development site supported
7. May still be undergoing somewhat major development

4.2. Discussion

A large amount of software gets developed within GRASS, but for numerous reasons is never polished off sufficiently to be taken over by OGI. Much of this code is nevertheless very valuable to sites that develop software. Such sites have the opportunity to peruse the SRC.CONTRIB areas for code that might decrease a programming effort by weeks or months.

OGI will apply minimal standards to code in src.contrib. The program names must not conflict with names used elsewhere. There must be some sense that the code may have value to groups in the GRASS user community.

5. RELATED (src.related)

5.1. Definition

GRASS is a set of geo-processing modules which can be used in conjunction with other free or public domain packages. Some of these systems are appropriate for redistribution with GRASS.

Related packages normally have a user community of their own and may follow programming guidelines and documentation standards established by the organization supporting the package. RELATED software, as such, is not supported by OGI.

6. GARDEN (src.garden)

6.1. Definition

This section consists of software which has been developed to link GRASS to other (RELATED) software packages.

Although software is placed in this directory based on what it is (link to related), the

same standards, requirements, and breakdowns apply as for any other software placed into src.main or src.alpha. The differentiation for these programs will be inherent only in the Gmakefiles, which place the compiled versions into different directories based on the main vs. alpha judgement.

Note that both src.related and src.garden programs must inherently be left as separate categories (not integrated under src.alpha or src.main), since the related programs are developed and maintained by separate organizations, and the garden software (although the link may be accomplished by developing either a GRASS command line interface or a shell script interface) is only useful in relationship to the related code.

7. SCRIPTS (src/scripts/shells)

7.1. Definition

Shell scripts written to run with GRASS. These are meant to be not only useful, but instructional. GRASS was to be a programming language in the spirit of UNIX as a programming language. Scripts, especially those linked with XGEN, bring out the true usefulness of GRASS.

8. PROGRAMMING STANDARDS

Every GRASS program distributed in the SRC.ALPHA and MAIN categories must not only be verified through testing in the user community, but must meet strict programming requirements. These requirements will generally be applied by OGI personnel (Shapiro, et.al.). Requirements involve:

1. Programs do not directly open and manipulate GRASS data bases.
2. Programs utilize GRASS subroutine libraries to the greatest extent possible. Often there exist subroutines which not only save the writing of pages of code, but they also have been proven to function on a broad array of UNIX environments.
3. Reserved subroutines are not used. Some GRASS subroutines are reserved for use by GRASS libraries and must not be discovered and used by the programmer.
4. Programs must provide a command-line version which uses the standard GRASS command line parsing routines. This is essential for facilitating a standard user interface.
5. Software must be organized with appropriate directories, subdirectories, Gmakefiles, etc.
6. Standardized reference manual documentation must exist.
7. Programs must not utilize files outside of the GRASS data base structure. In particular, temporary files must be created and accessed through the standard GRASS subroutines.

8. Software must always provide more than enough space for user input. For example, a future legal map name might look something like this:

   soils@PERMANENT:1-5:10:30-120

   The programmer never expecting any name longer than 14 letters will leave code susceptible to trouble.
9. Programs must compile into the appropriate place.
10. Program names must follow GRASS naming conventions.
11. All C programming must follow the guidelines for standard K&R C as described in the book "The C Programming Language" by Brian W. Kernighan and Dennis M. Ritchie (Prentice-Hall, Inc.). Modules requiring an ansi-C compiler will not be accepted.

For further details on standards, the reader is referred to the GRASS Programming Manual.

9. TRACKING CODE: PROGRAMMER TO MAIN

To further illuminate the process by which programs become part of the main distribution we will track an imaginary contribution from submission by the programmer through the acceptance into the main code.

1. Submission
   The program is submitted to OGI.
2. Code Review
   Time: less than one month. OGI, or its designate, reviews the software code checking for such problems as those outlined in the previous section. Code with problems is returned to the original programmer or site.
3. Software Review
   Time: less than two months. OGI submits the software to three different reviewing sites. Such sites are chosen based on their potential interest in the software, their timely availability, and their available hardware. To ensure good reviews, reviewers shall remain anonymous. Simultaneously the software is made available on the GRASS distribution FTP site. Programs rejected by the reviewers will not be processed further. Code with problems is returned to the programmer for repair and/or response. Repaired code is resubmitted to OGI.
4. Alpha Phase
   Recognizing that public domain software receives thorough testing only in general operation, the first general distribution of the new program will be as part of the ALPHA code in GRASS**. During this time errors and problems reported back to OGI will be referred to the original programmer who will then be expected to

---

**Not to be confused with the alpha test phase of a GRASS release during which time GRASS, as a whole, is evaluated. There is insufficient time during a GRASS alpha or beta phase to thoroughly test all new software.

make appropriate repairs available via anonymous ftp.

5. Main Phase

   Assuming all bugs discovered during the ALPHA phase have been fixed, the code is moved into MAIN in a subsequent release of the GRASS software.

Abandoned code - Occasionally code will be abandoned by the original programmer during the process detailed above. Such code, at the discretion of OGI, will be either discarded or placed into the CONTRIB section of GRASS software. From there any site, including OGI, will have the option of furthering the development of the orphaned code.