

Open Code at IPAC
David A. Imel, Manager
Caltech/IPAC

15 November 2017

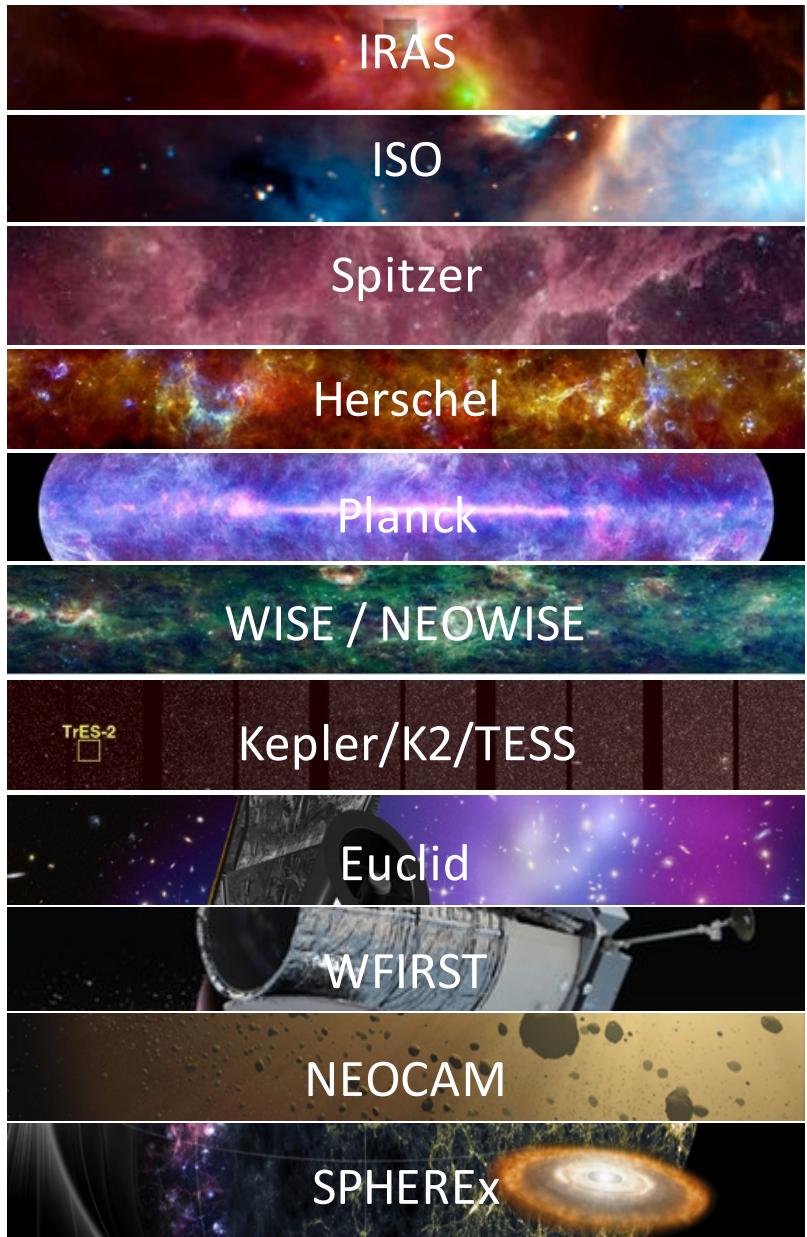
1. Introduction to IPAC
2. IPAC Code
3. IPAC Code Release Process
4. Considerations for Open Code



IPAC: Caltech Astrophysics Science Center

- Science Center functions for NASA missions since 1985
- Data centers and archives for major projects such as Great Observatories and all-sky surveys, including ground-based telescopes.
- Supports NASA, NSF and privately funded projects
- Award-winning media, outreach and education support
- Vibrant research environment and staff

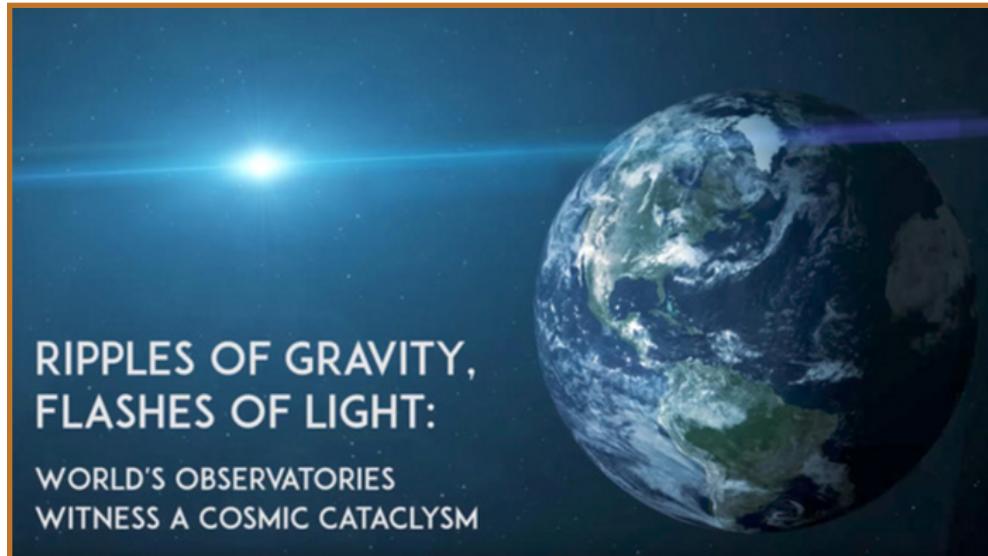
<i>Cosmology and galaxy evolution</i>	<i>Asteroids and the solar system</i>
<i>Exoplanets</i>	<i>Infrared-submillimeter astrophysics.</i>





IPAC Astronomy 2017

- Earth-Size planets: Spitzer finds seven Earth-size planets
- Gravitational Waves: IPAC astronomers, archives, and communications teams participate in synchronous EM observations.
- IPAC scientist shows how to point JWST to potential exoplanet systems.





Code Categories at IPAC

“Code” at IPAC covers a broad range of content.

- Data processing pipeline software for NASA Missions and Ground-Observatories
 - Data transfer and ingest
 - Calibration
 - Data product generation
- Observation planning and scheduling
- Archive software
 - Database ingest, search, and access
 - Fileservice
 - Web User Interface
 - Programmatic interfaces
 - Analysis software
- Research analysis software
- Websites (including javascript, CSS); e.g. proposal submission and management, helpdesk
- Internal datacenter systems; e.g. ansible playbooks
- Configuration files for all of the above

```
1
2
3   - hosts: bte
4     remote_user: centos
5     become_method: sudo
6     become: true
7     become_user: root
8     tasks:
9
10    - name: "Get Jenkins repo for Yum"
11      get_url:
12        dest=/etc/yum.repos.d/jenkins.repo
13        url=http://pkg.jenkins.io/redhat/jenkins.repo
14
15    #
16    # Had to disable gpg key check, because the key wasn't coming with the repo,
17    # and this command was failing all the time. Would rather get the key with
18    # the repo, but use this for now
19    #
20
21    - name: "install jenkins"
22      yum: name=jenkins state=present disable_gpg_check=yes
23
24    - name: "start jenkins service"
25      service: name=jenkins state=started
26
27    # Tried firewalld for this, but requires separate install, and google says unstable
28    # These commands aren't "idempotent", of course, so this could be wasting time...
29
30    - name: "start firewall service"
31      service: name=firewalld state=started
32
33    - command: /usr/bin/firewall-cmd --add-port=8080/tcp --permanent --zone=public
34    - command: /usr/bin/firewall-cmd --add-service=http --permanent --zone=public
35    - command: /usr/bin/firewall-cmd --reload
36
37    - command: sudo cat /var/lib/jenkins/secrets/initialAdminPassword
38      register: out
39      ignore_errors: yes
40
41    - debug: var=out.stdout_lines
```

Ansible playbook for
configuring a server



Code Release and Open Source

*While IPAC has always had released software,
IPAC is now also doing open-source development.*

NExSci
NASA Exoplanet Science Institute

SERVING THE EXOPLANET SCIENCE COMMUNITY

NExSci Interferometry Software

The NASA Exoplanet Science Institute (NExSci) provides software for use in the observation planning and data reduction for supported missions and facilities. This software was developed for direct support of the Keck Interferometer (KI) and Palomar Testbed Interferometer (PTI); however, it might also be useful to other investigators.

Some of the packages available here may be subject to copyright restrictions. Accept responsibility for compliance with any applicable restrictions and see the "Copyright" links below for more information regarding the release.

Support for the software discussed on this page is provided on a best effort basis.

V2Calib Links (PTI/KI visibility amplitude calib)

- DOWNLOAD V2CALIB
- ADDITIONAL INFORMATION
- COPYRIGHT
- VERSION INFORMATION

getCal Links

- DOWNLOAD GETCAL
- ADDITIONAL INFORMATION
- COPYRIGHT
- VERSION INFORMATION

Hipparcos Catalog Links (required for getCal)

- DOWNLOAD HIPPARCOS
- ADDITIONAL INFORMATION
- LINK TO HIPPARCOS MISSION

Visibility Modeling Tool Links - (part of the getCal software)

- DOWNLOAD VMT
- USER GUIDE (PDF FILE)
- ADDITIONAL INFORMATION
- COPYRIGHT
- VERSION INFORMATION

If NASA Exoplanet Science Institute software or services were helpful in your research, the following acknowledgement would be appreciated:

"This work has made use of services produced by the NASA Exoplanet Science Institute at the California Institute of Technology."

We are always interested in how our software is utilized and would appreciate receiving references to published works at the help desk page.

Interferometry analysis and observation planning code, available to the public for more than a decade.

json5_parser
Component for IRSA image reconstruction application

Montage
Image Mosaics for Astronomers

firefly_widgets
Jupyter widgets for Firefly

Fellowship-Software
Software used at NExSci to manage fellowship selection

planck_tol_utils
Component for IRSA image reconstruction application

Kepler-Discoveries
Utility scripts using the NASA Kepler Mission Discoveries website

CFOP
PHP/MySQL web database application which is designed to facilitate collaboration among astronomers in ground-based telescope observations of planetary candidates identified by the Kepler space telescope.

Caltech IPAC Software

Collection of Caltech IPAC Public Software Repositories

Caltech, Pasadena, CA | https://www.ipac.caltech.edu | support@ipac.caltech.edu

firefly
Java | Updated 23 hours ago

tabulator
C++ | Updated 21 days ago

firefly_client
Python API for Firefly

libtinyhtm
C++ | Updated 29 days ago

libhires
Component for IRSA image reconstruction application

libadql
IRSA Parser for VOADL queries

json5_parser
Component for IRSA image reconstruction application

Montage
Image Mosaics for Astronomers

firefly_widgets
Jupyter widgets for Firefly

Fellowship-Software
Software used at NExSci to manage fellowship selection

planck_tol_utils
Component for IRSA image reconstruction application

Kepler-Discoveries
Utility scripts using the NASA Exoplanet Archive API in support of the Kepler Mission Discoveries website

CFOP
PHP/MySQL web database application which is designed to facilitate collaboration among astronomers in ground-based telescope observations of planetary candidates identified by the Kepler space telescope.

Spitzer, Herschel observation planning code cleared for public release and shared with collaborators.

IPAC released and open-source code on Github.com: 13 public repositories.



IPAC Open Source: Montage

Montage code has been available since 2005

M

Montage

An Astronomical Image Mosaic Engine

NASA Space Act Award Winner 2006

[Web Service](#) [Download](#) [Documentation](#) [Publications](#) [Gallery](#) [Community](#) [Support](#)

What is Montage?

Montage is a toolkit for assembling [Flexible Image Transport System \(FITS\)](#) images into custom mosaics. Key features for end users are:

- Accuracy:** Preserves spatial and calibration fidelity of input images
- Portability:** Runs on all common Linux/Unix platforms
- Scalability:** Runs on desktops, clusters and computational grids
- Availability:** Open source code and user documentation available for download
- Generality:** Supports all [World Coordinate System \(WCS\)](#) projections and common coordinate systems

Performance

Performance: Processes 40 million pixels in up to 32 minutes on 128 nodes on a Linux cluster

Flexibility

Flexibility: Independent engines for analyzing the geometry of images on the sky; [re-projecting images](#); [rectifying background emission to a common level](#); [co-adding images](#)

Convenience

Convenience: Tools for managing and manipulating large image files

News

February 9, 2017
Four new tutorials on using Montage have been added to the [Documentation](#) page.

December 21, 2016
Version 5 released. See [Release of Montage Version 5.0](#) on the home page or the [Download](#) page.

August 16, 2016
We've posted a new tutorial for Montage novices called [Getting Started: Creating Your First Mosaic](#).

November 16, 2015
We have released our [YouTube](#) channel that has four videos of image cubes created with Montage, including the GALFA movie shown at the ADASS conference in Sydney in October. Also, Montage is now on Twitter as [@montage_mosaics](#).

September 30, 2015
Version 4 Released. See [Release of Montage Version 4.0](#) on the home page.

February 18, 2015
Mac users should see the note on the left regarding patching the software.

July 23, 2014
The [Montage license](#) has been changed to a [BSD 3-Clause License](#), which permits unlimited redistribution of Montage code for any purpose as long as its copyright notices and the license's disclaimers of warranty are included.

January 19, 2011
Montage now has a published [Wikipedia article](#).

December 15, 2010
[Montage version 3.3 released!](#)
Plus, new [C-shell scripts](#) contributed by Colin Aspin and new [publications](#) on using

We have released four new tutorials, as follows:

- [Image stretching in Montage](#)
- [Creating coverage maps in Montage](#)
- [Using HEALPix with Montage](#)
- [View Your Images In The WWT With Montage](#)

You may also access these from the Tutorials and Scripts section of [Documentation](#) page.

Release of Montage Version 5.0

Version 5.0 is a major new release of Montage. The distribution is available from GitHub at <https://github.com/Caltech-IPAC/Montage> and from the Montage web page at <http://montage.ipac.caltech.edu/docs/download.html>. Version 5.0 includes a new reprojection module, `mProjectQL`, that is suitable for creating images for visualization; `sup`, `sup`, `cor`, `cor`

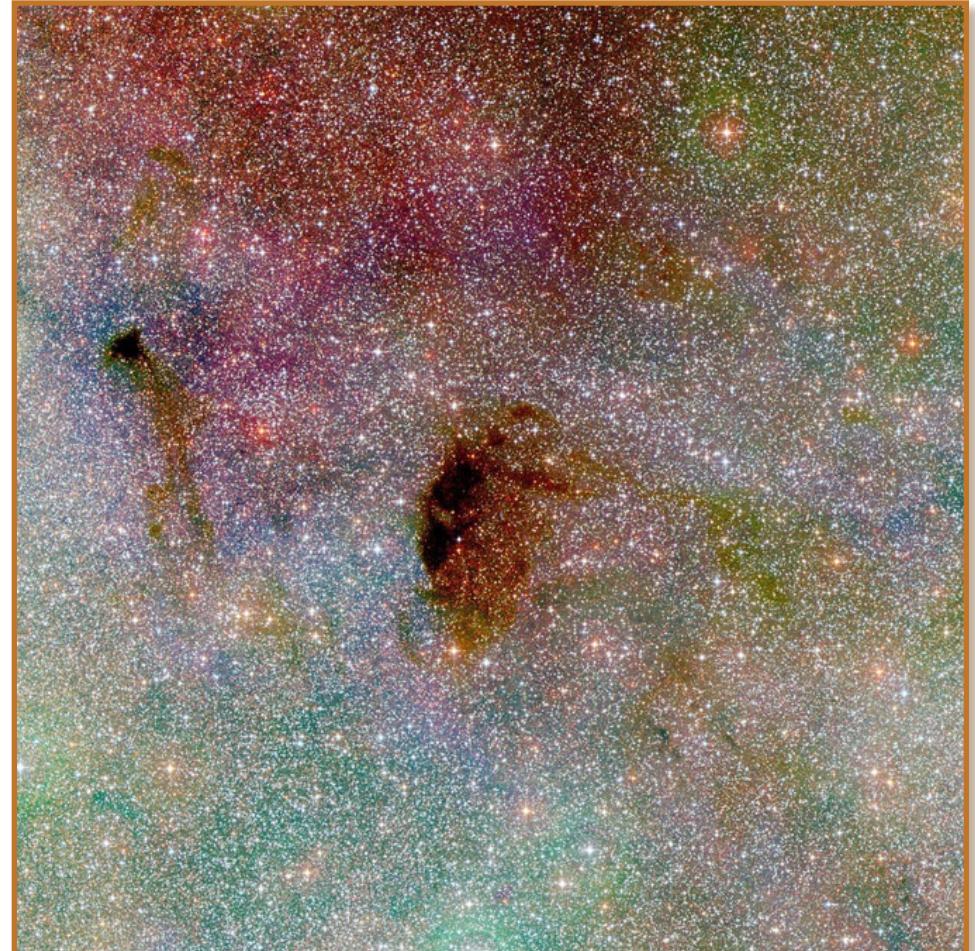
<http://montage.ipac.caltech.edu>

Montage is written in ANSI-compliant C and intended for use on all common Unix-based platforms. It was tested formally on RedHat Enterprise Linux Server 5.9 and on Mac OS X 10.11.

In detail:

- `mProjectQL` uses the Lanczos image interpolation scheme to provide higher performance in reprojection at the expense of conservation of flux; we recommend `mProjectQL` primarily for creating images for visualization rather than for science analysis. It offers a speed-up of approximately x20 over `mProject`. It can be invoked in the executive modules `mProjExec` and `mExec` by using the `-q` switch.
- Support for HEALPix and TOAST: Montage treats HEALPix and TOAST as if they were spherical projections that can be processed with the existing reprojection routines

Open Code at IPAC / imel



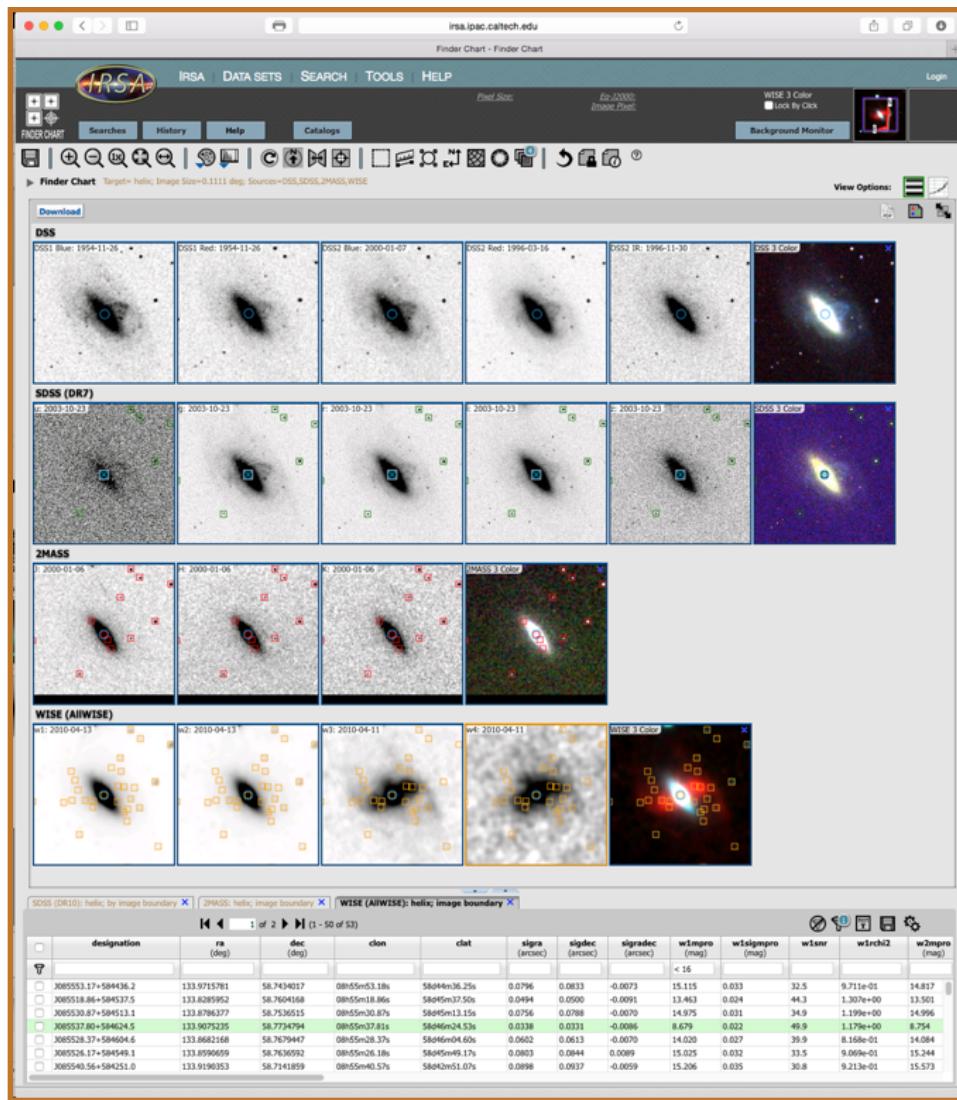
This image mosaic centered on the dark cloud Barnard 92 preserves the positional accuracy and intensities of a total of 92 input images, and is an example of the types of images astronomers use in their research.

Image credit: Montage.



IPAC Open Source: Firefly

IPAC's primary User Interface Code is now open source software.



Firefly is used by both the Infrared Science Archive (IRSA) to display and analyze 2MASS, WISE, Spitzer, Planck, Herschel, and many other datasets, as well as by the Large Synoptic Survey Telescope (LSST) Science Platform. Firefly is about ~200K lines of code.

October 6, 2017 – November 6, 2017 Period: 1 month ▾

Overview

19 Active Pull Requests 1 Active Issue

15	Merged Pull Requests	4	Proposed Pull Requests	0	Closed Issues	1	New Issue
----	----------------------	---	------------------------	---	---------------	---	-----------

Excluding merges, 8 authors have pushed 45 commits to dev and 51 commits to all branches. On dev, 277 files have changed and there have been 6,756 additions and 2,947 deletions.



15 Pull requests merged by 7 people

- Merged #487 DM-12248 Chart options as a dialog 5 hours ago
- Merged #489 IRSA-975: NED searching seems not to work 4 days ago
- Merged #486 IRSA-810: Replace left hand options links to top left hand tabs (FinderChart) 6 days ago
- Merged #484 DM-11856: framework for client properties defined at build time 11 days ago
- Merged #483 IRSA-811: Download PDF goes directly to background monitor 13 days ago
- Merged #479 IRSA-403 Remove Reset button next to the manually input box in the Se... 17 days ago



IPAC Process for Code Release

All Software created at Caltech is Caltech Intellectual Property

- IPAC is committed to sharing code under the right conditions.
- IPAC must follow Caltech policy on Intellectual Property
- Process when request is received (internal or external) for releasing code:
 1. Evaluate support required and resources available to support the release: both initial and ongoing.
 2. Obtain approval from Caltech Office of Technology Transfer
 3. **Conduct Code Release Review**
- Publish code (typically via Github.com) after successful review.

Public release of software inevitably means responding to questions, bugs, and suggestions for new functionality.

Author	Labels	Projects	Milestones	Assignee	Sort
erlcamandel					
astrofrog					
bfontaine					
lincastro					
olebole					
siodmik					
cdlga					



Code Release Review Criterion: Security

Beware git revision history and comments!

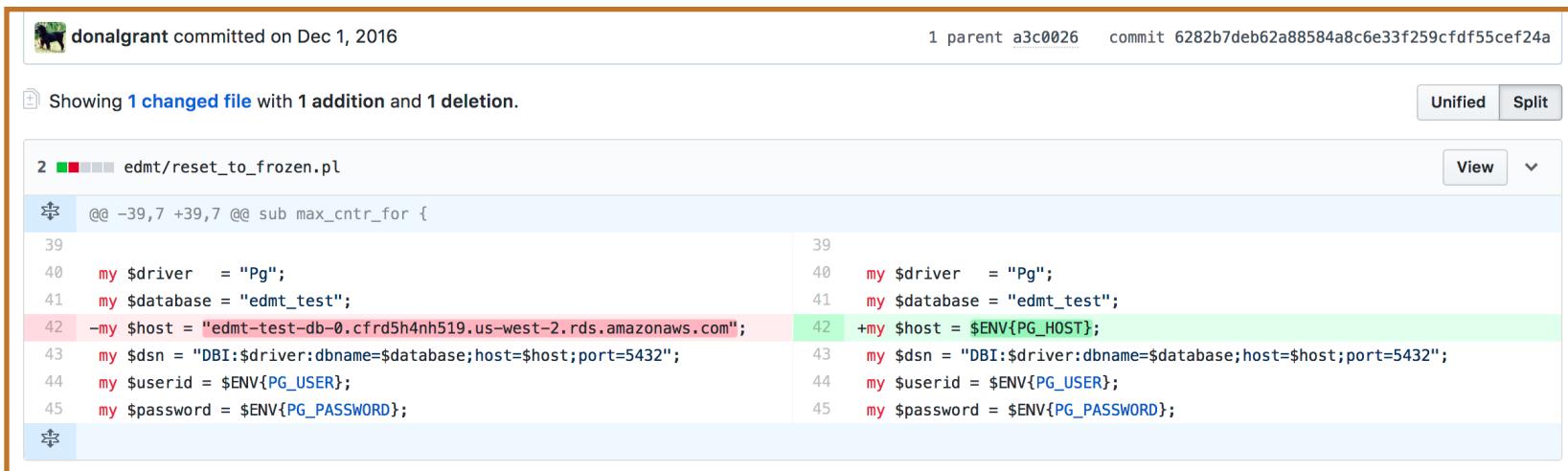
- Does the code expose internal network or server configurations?

Example from pre-released software:

```
// example: http://bacchus.ipac.caltech.edu:5000/wise/pass1/i3om_cdd/{coadd_id}/ {band}/{product}
```

➤ comment with instructions reveals internal server and filesystems

- Does the code reveal interfaces in a way that makes the system easier to hack?
- Are all inputs taint-checked?
- Passwords or other sensitive info in code or revision history?



donalgrant committed on Dec 1, 2016

1 parent a3c0026 commit 6282b7deb62a88584a8c6e33f259cfdf55cef24a

Showing 1 changed file with 1 addition and 1 deletion.

edmt/reset_to_frozen.pl

@@ -39,7 +39,7 @@ sub max_cntr_for {	39
40 my \$driver = "Pg";	40 my \$driver = "Pg";
41 my \$database = "edmt_test";	41 my \$database = "edmt_test";
42 -my \$host = "edmt-test-db-0.cfrd5h4nh519.us-west-2.rds.amazonaws.com";	42 +my \$host = \$ENV{PG_HOST};
43 my \$dsn = "DBI:\$driver:dbname=\$database;host=\$host;port=5432";	43 my \$dsn = "DBI:\$driver:dbname=\$database;host=\$host;port=5432";
44 my \$userid = \$ENV{PG_USER};	44 my \$userid = \$ENV{PG_USER};
45 my \$password = \$ENV{PG_PASSWORD};	45 my \$password = \$ENV{PG_PASSWORD};

If a password or some other sensitive piece of information was once part of the code, it will be there forever in the git revision history, unless careful measures are taken to excise all references to those revisions. Just scanning the current code will be insufficient to detect this issue.



Code Release Review Criterion: Legal

Beware of mixed license types in embedded software packages

1. Are the Caltech Office of Technology Transfer requirements met?
(Approval, license files / copyright statements in each source file.)
2. Are the constraints of any licenses of embedded modules met?

Software licenses and rights granted in context of the copyright according to Mark Webbink . ^[1] Expanded by freeware and sublicensing.							
Rights granted	Public domain	Non-protective FOSS license (e.g. BSD license)	Protective FOSS license (e.g. GPL)	Freeware/Shareware/Freemium	Proprietary license	Trade secret	
Copyright retained	No	Yes	Yes	Yes	Yes	Yes	Yes
Right to perform	Yes	Yes	Yes	Yes	Yes	Yes	No
Right to display	Yes	Yes	Yes	Yes	Yes	Yes	No
Right to copy	Yes	Yes	Yes	Often	No	No	No
Right to modify	Yes	Yes	Yes	No	No	No	No
Right to distribute	Yes	Yes, under same license	Yes, under same license	Often	No	No	No
Right to sublicense	Yes	Yes	No	No	No	No	No
Example software	SQLite, ImageJ	Apache web server, ToyBox	Linux kernel, GIMP	Irfanview, Winamp	Windows, Half-Life 2	Server-side World of Warcraft	

Table illustrating different software license types, from Wikipedia



Open Source Licenses

- IPAC typically releases software under the BSD-3 license:
 - Allows redistribution
 - Retains copyright
 - Limits liability
- Another popular license is the GPL “Copyleft” license:
 - More restrictive than BSD-3
 - Requires that all derivative work retain the same license as the original.
 - Guarantees perpetual open access to derivative work.
- Public domain:
 - May not be an adequate approach for private institutions (liability) and sponsored organizations (citations).

```
1 Copyright © <2014>, California Institute of Technology, developed for the
2 Spitzer Science Center and the NASA/IPAC Infrared Science Archive (IRSA) with
3 support from NASA, and for Large Synoptic Survey Telescope (LSST) with
4 support from NSF.
5
6 All rights reserved.
7 Redistribution and use in source and binary forms, with or without
8 modification, are permitted provided that the following conditions are met:
9
10 * Redistributions of source code must retain the above copyright notice,
11   this list of conditions and the following disclaimer.
12 * Redistributions in binary form must reproduce the above copyright notice,
13   this list of conditions and the following disclaimer in the documentation
14   and/or other materials provided with the distribution.
15 * Neither the name of the California Institute of Technology (Caltech) nor
16   the names of its contributors may be used to endorse or promote products
17   derived from this software without specific prior written permission.
18
19 THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND
20 ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED
21 WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
22 DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE
23 FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
24 DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR
25 SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER
26 CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY,
27 OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
28 OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
```

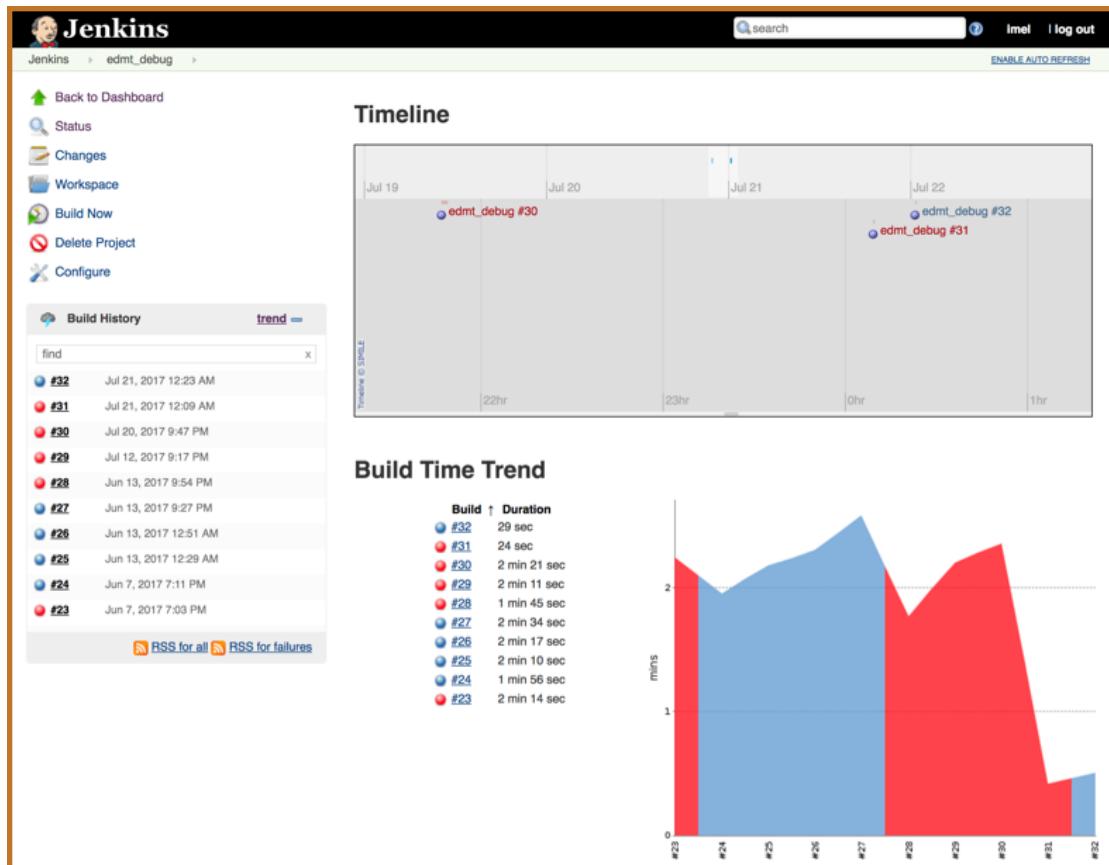
BSD-3 License



Code Release Review Criterion: Functionality and Builds

Does it work out of the box?

1. If the system is not already in operations, what testing has been done to validate that the system works as advertised?
2. Demonstrate a successful build using a fresh clone from Github.com repository into a system where the code has not previously been used.



Some IPAC software systems use automated continuous integration, where a build is automatically executed either periodically or whenever a change is pushed to the software repository.



Code Release Review Criterion: Style and Professionalism

Will this code embarrass anyone?

1. Any inappropriate comments in the code?
2. Does the code have commented-out components that serve no purpose?
3. Typos?
4. Any other stylistic issues that are so significant as to be embarrassing to the organization or the sponsor?

```
/*
public Position( double          lon,
                 double          lat,
                 CoordinateSys  coordSystem)
                 throws IllegalArgumentException {

    /* the following line is about as clear as mud, what it is doing
    // is passing an epoch of 1950 when the coordinate system is
    // b1950, an epoch of 2000 when J2000 and epoch to 2000 in
    // all other cases. In all other cases the epoch is meaningless
    // so it is just a placeholder
    this(lon,lat,null,coordSystem,
         coordSystem.equals(CoordinateSys.EQ_B1950) ? EPOCH1950 : EPOCH2000);
}
```



Code Release Review Criterion: Documentation

Superfluous content can be as bad as missing documentation.

1. Is the associated documentation current, accurate, complete, organized, and professional?
2. Is there any documentation or material related to the system which does not need to be included and should be removed? Some examples we've encountered:
 - Word documents with revision histories still embedded.
 - Early design documents which no longer accurately reflect the system.

Even when complete documentation exists, it can take time to make sure that each item is appropriate for public release. In effect, the documentation itself must be regression tested for the software system.



Case Study: The Spitzer Pipeline

Spitzer IRAC pipeline source code was released.

<http://irsa.ipac.caltech.edu/data/SPITZER/docs/irac/calibrationfiles/pipeline/>

IRAC Pipeline Source Code

The IRAC pipeline files consist of c codes and associated name lists. There is little documentation about the functions that each code performs, and there is no wrapper that would run the whole pipeline from beginning to end. If you intend to use any of this code, please be extremely careful. The Spitzer Science Center does not assume responsibility for any errors resulting from the use of these scripts by individual Spitzer data users.

[Download Source Code \(Tar File, 2.6 MB\)](#)

5.1.13 FOWLINEARIZE (detector linearization)

Like most detectors, the IRAC arrays are non-linear near full-well capacity. The number of incident photons, rather than becoming increasingly small as the levels above half full-well (typically 20,000–30,000 DN in the raw data), the processing the raw data are linearized on a pixel-by-pixel basis using a model verified in flight. The software module that does this is called FOWLINEARIZE to each pixel based on the number of DN, the frame time, and the linearity solution, i.e., we model the detector response as

$$DN_{obs} = kmt - Ak^2t^2$$

The linearization solution to the above quadratic model is:

$$DN = \frac{-1 + \sqrt{1 - 4L\alpha DN_{obs}}}{2L\alpha}$$

where

$$L = \frac{\alpha}{n(w+n)^2} \left[\left(\sum_{i=w+1}^{n+w} i^2 - \sum_1^n i^2 \right) - 2(1 - \frac{t_d}{t_c})n(n+w) \right],$$

and $\alpha = \frac{A}{m^2}$, n is the Fowler number, and w is the wait period. The above formula is for the case of no multi-sampling. Note that the time between frames is t_d and the time between observations is t_c .

Algorithm descriptions

For the cubic model, the solution is derived via a numerical inversion.

Corresponding Code

```

for (p=0;p<Frames; p++)
  for (j=0; j<LenY; j++)
    for (i=0; i<LenX; i++)
    {
      Inp1 = p*LenX*LenY + i*LenY + j;
      I_Index_Inp2c1 = i*LenY + j; /* plane 1 lincal */
      I_Index_Inp2c2 = LenX*LenY + i*LenY + j; /* plane 2 lincal */
      Out1 = p*LenX*LenY + i*LenY + j;

      /* compute delay time in msec (td) depending on whether sub-array or full-
array Clock_Period was specified. */

      tc = FOMP_Const->Clock_Readout;

      if (tc == 200.0)
        tdij = (16.8*(256-(i+1)) + 1180
          + 10*(int)((i+1)-1)/4
          + 648*((i+1)-1))*E-3;
      else if (tc == 100)
        tdij = (16.8*(248-(i+1)) + 1160
          + 10*(int)((i+1)+1)/4
          + 108*((i+1)+1));
      else
        tdij = (16.8*(248-(i+1)) + 1160
          + 10*(int)((i+1)+1)/4
          + 108*((i+1)+1));

      /* output from LINCAL is in m^2 and defined mt + At^2, hence need -1* */
      /* (image2[I_Index_Inp2c1]) *
      ((s2-s1)-2*(1-(tdij/tc)) * FOMP_FITS->afowl
      * (FOMP_FITS->a_fowl + FOMP_FITS->awa_it))
      (FOMP_FITS->a_fowl+pow((FOMP_FITS->afowl
      + FOMP_FITS->awa_it),2))

      /* if problems from LINCAL (mask) then don't linearize, otherwise do: */

      if (! (strcmp(FOMP_Fnames->CP_Filename_FITS_Image_CMsk, "") == 0))
      {
        if ((CMask[I_Index_Inp2c1] & FOMP_Const->CMaskFatal) > 0)
        {
          DP_Output_Array[Out1] = Image1[Inp1];
        }
      }
    }
  }
}

```

- 1998 design decisions—driven by requirements and costs—meant that pipeline code is not buildable outside of Spitzer environment (specific connections to Informix database, Spitzer filesystems)
- Recognized as an issue after launch but no resources for complete infrastructure change to make exportable modules for all pipelines
- Algorithms for all instrument pipeline modules are documented in the Instrument Handbooks
- Mosaicking and Source extraction pipelines, and multiple other tools, released as user-buildable code
- IRAC pipeline source code released with **warnings** (2638 files, 490k lines of code)



Considerations with Wider Code Release

Enforcing public code release could impact the cost of NASA missions.

Capability:

- Public release can provide volunteer external review and validation of algorithms
- Open source development may lead to external bug catches and algorithm improvements.
- In some cases, projects may require applications where no open-code alternative to a commercial package exists, and the cost of developing a new open-code solution is prohibitive.

Intellectual Property:

- Typically the implementing institution owns rights to the code. Caltech Office of Technology Transfer has been very helpful with code release.
- Can probably be addressed by contract with NASA.

Security:

- ITAR and export-compliance restrictions. Source code related to covered technologies is explicitly covered.
- Exposing source code can increase the risk of a successful hack attempt. It may also provide broader feedback on security vulnerability.
- Some config files or parameter files probably cannot be released because of security concerns.

Licensing:

- Will need a guiding principal on software license. For example, BSD-3.
- Software components with more restrictive licenses cannot be released under less restrictive licenses.
- Some restrictions on use (misuse) probably needed.

Cost:

- Most projects are not costed with public code-release as part of the estimate:
 - additional review activities.
 - development and documentation costs may be higher for publicly-released code.
 - additional support is required to respond to queries about released source code. This cannot be avoided without damaging project and organization reputations.
- Code may be used for cases not covered by original requirements and testing. Release notes must give bounds on support.
- Need to find a support-model for released code after project termination. What happens when code no longer builds on any currently-supported compiler version?