

# A history, status report, and outlook of Proj.4

[@HOWARDBUTLER](#)



## Open Source LiDAR

---

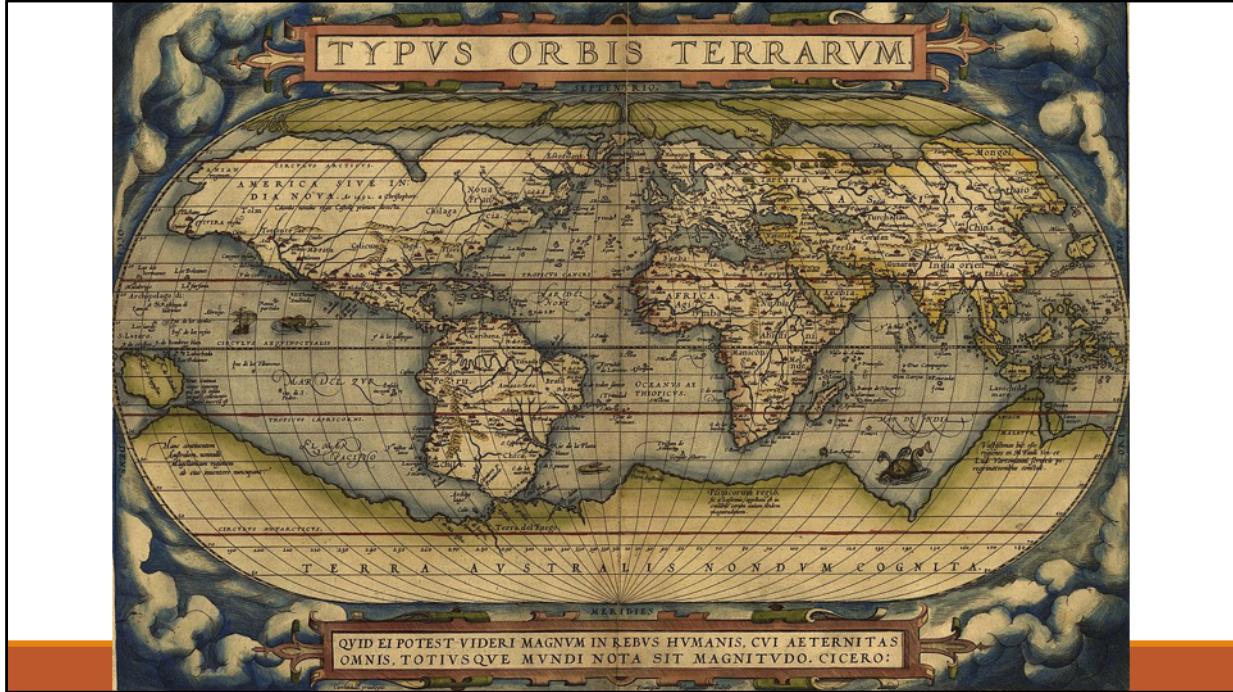




[https://en.wikipedia.org/wiki/Map\\_projection](https://en.wikipedia.org/wiki/Map_projection)

We in the geospatial software realm take a number of truisms for granted.

The ability to transform between coordinate systems is  
one of the most fundamental.

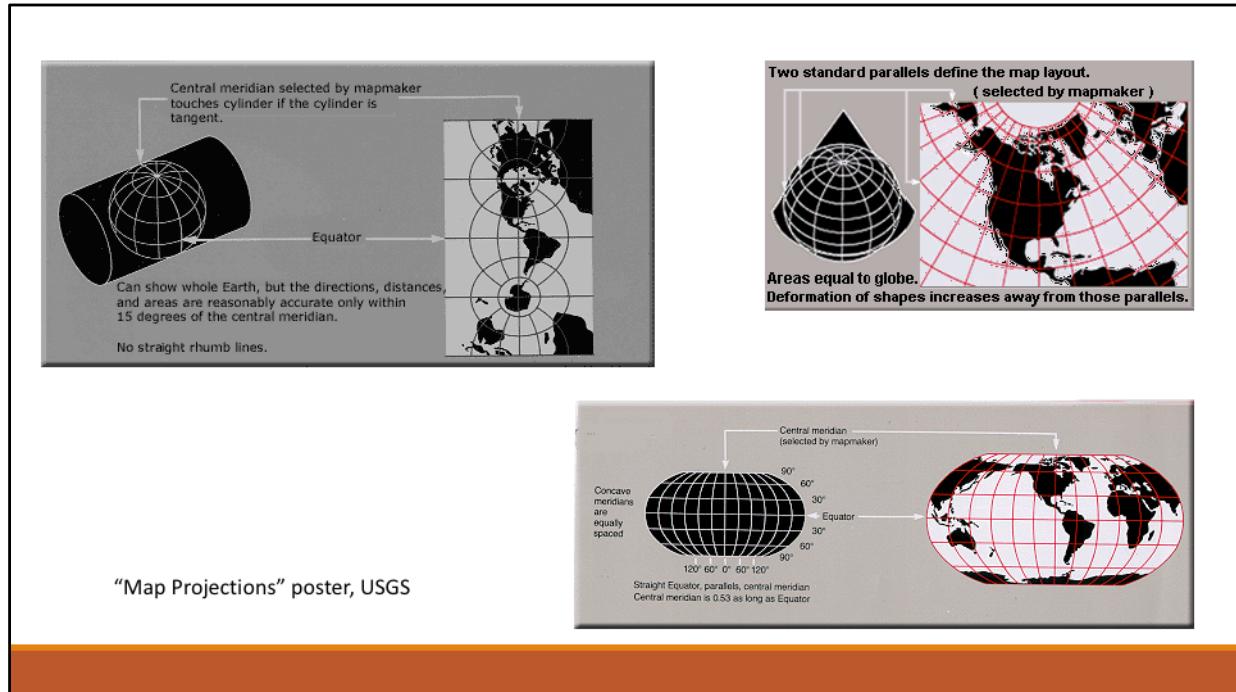


A primary challenge of geospatial software is  
how to project a drawing of stuff on a round sphere onto a flat plane.

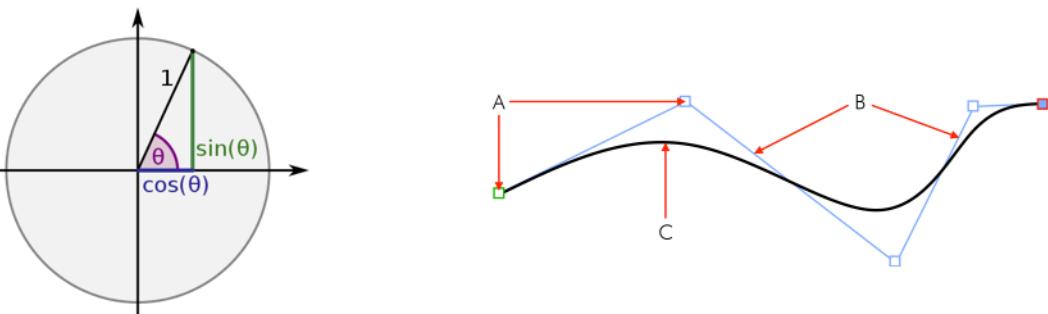
There are centuries of effort on the topic.



Some might argue that mapping software that does not concern itself with this challenge is not credible.



The ternary constraints of direction, distance, and shape  
loom over the mapmaker as they choose  
which to forgive in exchange for clear communication.



Cheap CPU cycles and efficient software  
were required to support the computation of  
fancy curves,  
obnoxious trigonometry, and  
iterative solutions required by software that does these projections.

# Jerry Evenden

- 1935 - 2016
- USGS
  - Coastal and Marine 1976-1993
- San Jose State and Colorado School of Mines
- Geophysicist



In this locus, a researcher named Jerry Evenden in the Marine Geology group of USGS at Woods Hole took up the challenge of implementing the intricate math colleagues such as Snyder and Robinson collated and created in their eponymous books.

The software he wrote, PROJ, has wide and lasting impact throughout the geospatial software industry.

Today we will look back on PROJ and note Jerry's recent passing in 2016 with a remembrance.

I will describe how PROJ evolved with Frank Warmerdam's effort to become a full fledged open source project that nearly embedded itself in every open source geospatial software in some way.

I will discuss recent developments new contributors such as Kristian Evers, Thomas Knudsen, and Even Rouault have made that will keep the project moving forward for years to come.

and I will also discuss where the project is evolving.

# Jerry Evenden

- 1935 - 2016
- USGS
  - Coastal and Marine 1976-1993
- San Jose State and Colorado School of Mines
- Geophysicist



Jerry knew he was going to be a scientist from age ten.

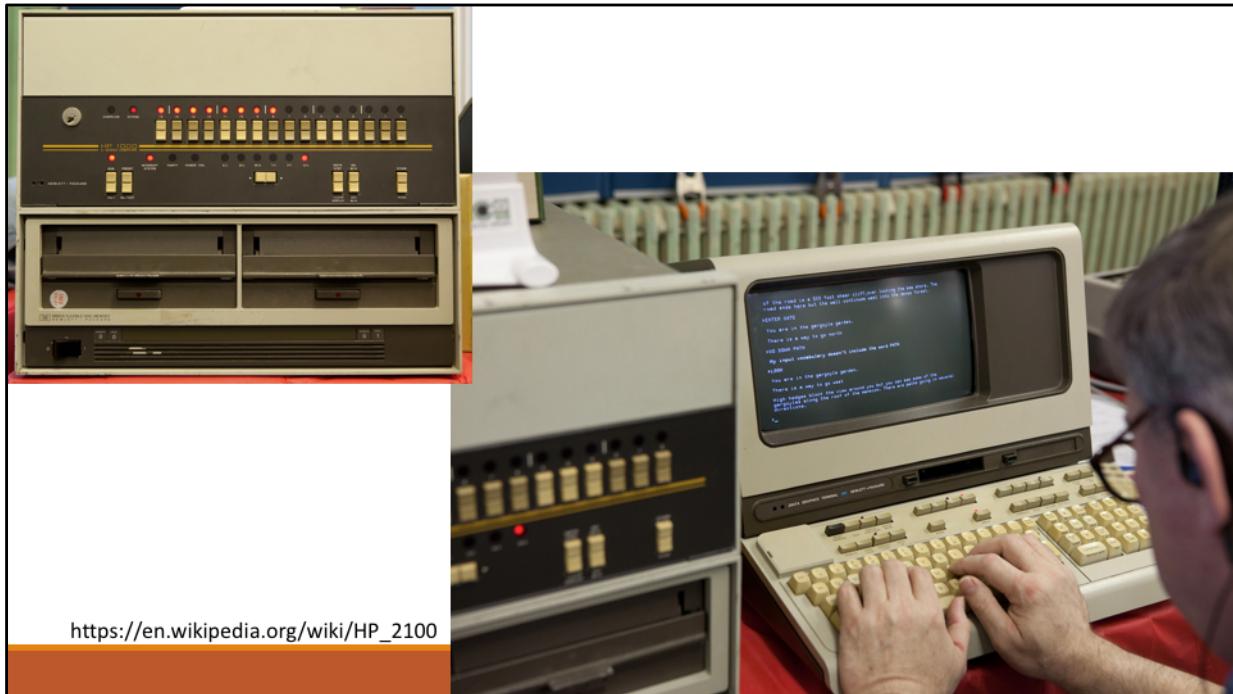
He wasn't an overly emotional person, and  
he prized logical, rational thought.

Jerry met his wife Phyllis in Washington DC  
where she was a park naturalist for the Park Service.

Phyllis mentioned they enjoyed  
going to the pistol range together in DC  
before they moved out to Denver,  
where Jerry took a job doing seismic anomaly fieldwork for the USGS.

In 1976, he was offered a transfer to  
Marine Geology in Woods Hole, Massachusetts.

It was there where he lived with his wife Phyllis of 51 years until his death last year in April 2016.



The offer to go to Woods Hole coincidentally timed with the era of mini and personal computers, and Jerry quickly started using them at USGS to solve problems.

The rapid miniaturization of computing revolutionized sensors too, and Jerry collaborated with many projects using them to map the geology of the ocean floors.

Making maps that legibly covered large areas with the results of their surveys was a challenge for Jerry and his collaborators, and he set about the task with whatever he could find.



Jerry requisitioned three unix computers he called Larry, Curly, and Moe to support his software development efforts.

He wrote a map rendering software called **MAPGEN** for vector rendering of maps, and he combined that with a typesetting software he wrote called **WOLF**, or Word Oriented Line Formatter, to build a LaTeX + Scalable Vector Graphics of its day.

In support of mapping projects he was doing at the time, he took John Snyder's book and started implementing projection transforms into what became the PROJ software library.

Jerry retired from USGS in 1993, and his software including PROJ lived online as a few tarballs on some FTP servers until Frank Warmerdam started to pick them up in the mid 90s



A problem with software that spans generations is  
it is a product of the software culture and problems of its time.

Software design fashions change

Software tooling changes.

Software development techniques change

and each of these changes at different time scales.

Each of these causes software to atrophy in different ways too.

When we say an API from an ancient open source software project isn't usable,  
we really mean we don't have a  
software cultural literacy to translate it to today



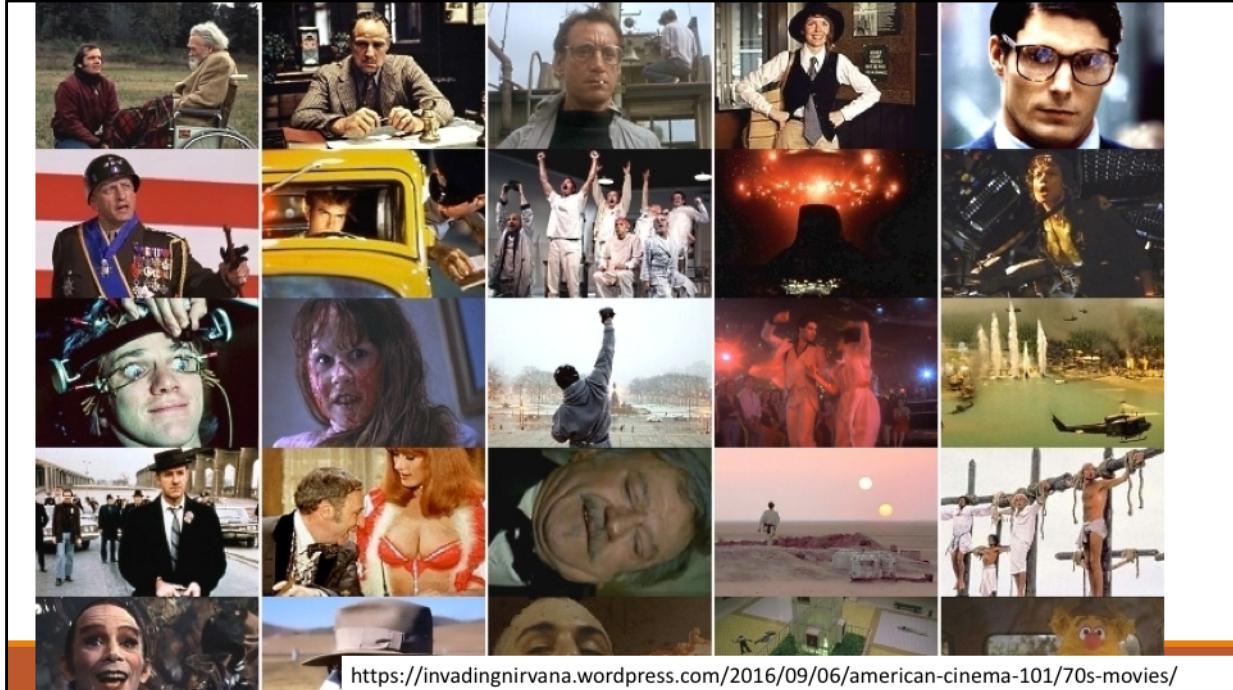
[https://en.wikipedia.org/wiki/Glenn\\_Miller\\_Orchestra](https://en.wikipedia.org/wiki/Glenn_Miller_Orchestra)

Like music from the 1940s,



<https://www.flickr.com/photos/128157542@N08/16037316102>

fashion from the 1950s,



<https://invadingnirvana.wordpress.com/2016/09/06/american-cinema-101/70s-movies/>

or movies from the 1970s,  
PROJ was software born of its culture.

That it is incidentally useful to us,  
here in the future,  
is a feature unique to open source software.

Open source software allows anyone  
wishing to give software attention  
the ability to keep it alive indefinitely.



Like an antique store record junky spinning 45s,  
all that is required is for someone with enough  
motivation,  
enthusiasm, and  
desire to keep  
the software alive simply  
by paying attention to it and  
taking care to keep it from atrophying.



The foundation of one software civilization is the bones of another.

Software eras are more like civilizations.

Long lasting software is like art, music, or culture  
that survives the transition from one civilization to another.

We might know what it does, and  
we might know who made it, but  
the context of how it fits with its peers, or  
why certain choices were made are often lost.

Even when something is carried forward  
to a new computing future, developers who come later  
end up trampling down APIs and  
wrapping and stacking abstraction over them  
to translate the disconnect.

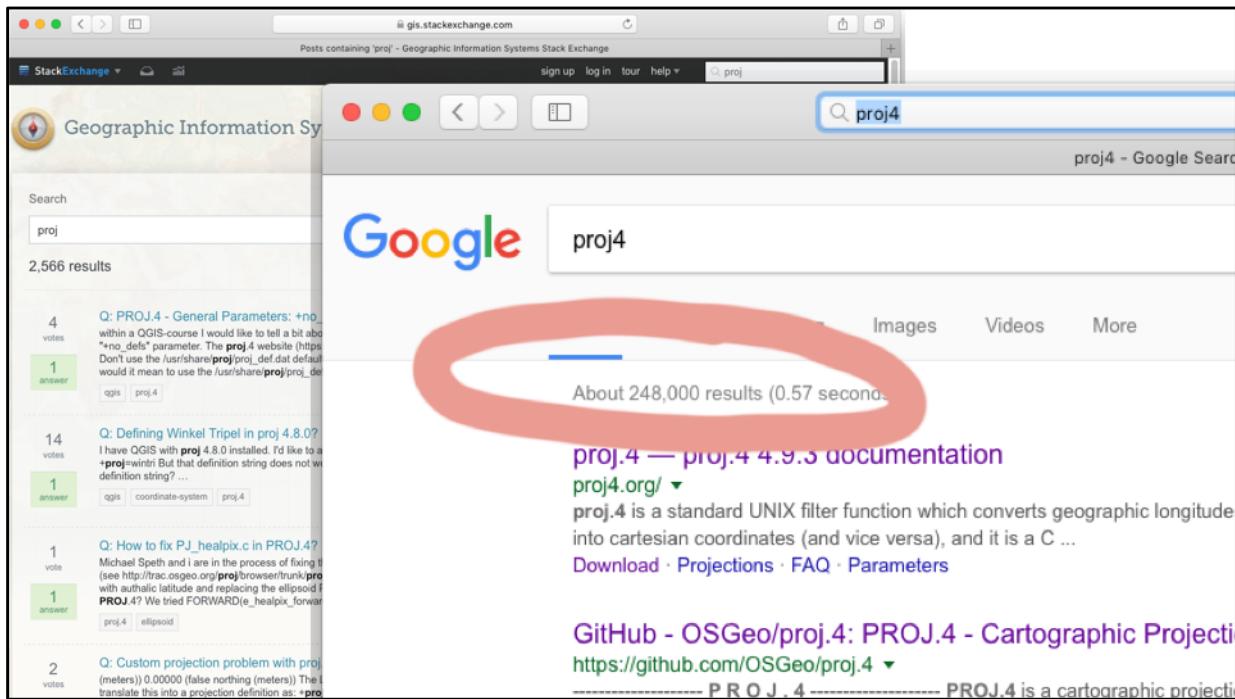
PROJ was written for another computing time.



and PROJ was written for computers that don't exist today.

Did those computers even have one tenth of the computing power of the phones in our pockets today?

PROJ's API is quite natural if you are  
culturally steeped in consuming 1990s  
opaque object C APIs,  
fume huffing macro compiler abuse,  
and messy include structure  
that was once organized to make things compile faster.



PROJ was written before there was StackOverflow,

before Google,

and before Usenet.

Programming in the 1980s and 90s  
wasn't the simple pasting of error messages  
into a search engine as it is now

The screenshot shows two terminal windows side-by-side. Both windows have a title bar 'proj.h [~/dev/git/proj4src] - VM3'.

**Left Terminal (proj.h):**

```

1 / 
2 * Project: PROJ.4-
3 * Purpose: Revised, experimental API for PROJ.4, intended as the foundation-
4 * for added geodetic functionality.
5 *
6 *
7 * The original proj API (defined in projects.h) has grown organically-
8 * over the years, but it has also grown somewhat messy.-
9 *
10 * The same has happened with the newer high level API (defined in-
11 * proj_apl.h): To support various historical objectives, proj_apl.h-
12 * contains a rather complex combination of conditional defines and-
13 * #includes. Probably for good (historical) reasons, which are not-
14 * always evident from today's perspective.-
15 *
16 * This is an evolving attempt at creating a re-rationalized API-
17 * with primary design goals focused on sanitizing the proj_apl.h-
18 * interface. All symbols exposed are being moved to the PROJ namespace,
19 * while all data types are being moved to the PJ namespace.-
20 *
21 * Please note that this API is "orthogonal" to the previous APIs:-.
22 * Apart from some inclusion guards, projects.h and proj_apl.h are not-
23 * touched - if you include proj.h, the projects and proj_apl-
24 * APIs should work as they always have.-
25 *
26 * A few implementation details:-.
27 *
28 * Apart from the restructuring efforts, I'm trying to eliminate three-
29 * proj_apl elements, which I have found especially confusing.-
30 *
31 * FIRST and foremost, I try to avoid projdefining away pointer-
32 * semantics. I agree that it can be occasionally useful, but I-
33 * prefer having the pointer nature of function arguments being-
34 * explicitly visible.-
35 *
36 * Hence, projCtx has been replaced by PJ_CONTEXT,-
37 * and projPT has been replaced by PJ.-.
38 *
39 * SECOND, I try to eliminate cases of information hiding implemented-
40 * by redefining data types to void pointers.-
41 *
42 * I prefer using a combination of forward declarations and projdef.-
43 * Hence:-.
44 *
45 * projdef void "projCtx";-
46 * Has been replaced by:-.
47 * projCtx_t PJ_CONTEXT;-
48 *
49 * projdef void "projPT";-
50 * Has been replaced by:-.
51 * projPT_t PJ;-
52 */

```

**Bottom Left Status:** NORMAL > +0 -0 -0 master <c/proj.h> cpp utf-8[unix] < 9% : 40: 1 <! trailing[283]

**Right Terminal (projects.h):**

```

1 / 
2 * Project: PROJ.4-
3 * Purpose: Primary (private) include file for PROJ.4 library.-
4 * Author: Gerald Knorr
5 *
6 ****
7 * Copyright (c) 2000, Frank Warmerdam
8 *
9 * Permission is hereby granted, free of charge, to any person obtaining a-
10 * copy of this software and associated documentation files (the "Software"),-
11 * to deal in the Software without restriction, including without limitation-
12 * the rights to use, copy, modify, merge, publish, distribute, sublicense,-
13 * and/or sell copies of the Software, and to permit persons to whom the-
14 * Software is furnished to do so, subject to the following conditions:-
15 *
16 * The above copyright notice and this permission notice shall be included-
17 * in all copies or substantial portions of the Software.-
18 *
19 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS-
20 * OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,-
21 * FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL-
22 * THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER-
23 * LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING-
24 * FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER-
25 * DEALINGS IN THE SOFTWARE.-
26 ****
27 */
28 /* General projections header file */
29 #ifndef PROJECTS_H_
30 #define PROJECTS_H_
31 /*
32 #ifdef MSC_VER
33 #ifndef _CRT_SECURE_NO_DEPRECATED
34 # define _CRT_SECURE_NO_DEPRECATED
35 #endif
36 #ifndef _CRT_NONSTDC_NO_DEPRECATED
37 # define _CRT_NONSTDC_NO_DEPRECATED
38 #endif
39 /* enable predefined math constants M_* for MS Visual Studio workaround */
40 #ifndef _USE_MATH_DEFINES
41 # define _USE_MATH_DEFINES
42 #endif
43 #endif
44 */
45 /* standard inclusions */
46 #include <math.h>
47 #include <stdio.h>
48 #include <stdlib.h>
49 #include <string.h>

```

**Bottom Right Status:** NORMAL > +0 -0 -0 master ..src/projects.h cpp utf-8[unix] < 0% : 1 <! trailing[283]

The C programming language,  
which PROJ is written in,  
was still somewhat new when Jerry started writing PROJ in it.

Unix and C were just about ten years old,  
and which tools would survive  
the test of time was still in doubt.

```

1 /> Project: PROJ-4-
2 Purpose: Revised, experimental API for PROJ-4, intended as the foundation-
3 for added geodetic functionality.
4
5
6
7 The original proj API (defined in projects.h) has grown organically-
8 over the years, but it has also grown somewhat messy.-
9
10 The same has happened with the newer high level API (defined in-
11 proj_apl.h): To support various historical objectives, proj_apl.h-
12 contains a rather complex combination of conditional defines and-
13 #ifdefs. Probably for good (historical) reasons, which are not-
14 always evident from today's perspective.-
15
16 This is an evolving attempt at creating a re-rationalized API-
17 with primary design goals focused on sanitizing the proj_apl.h-
18 interface. All symbols exposed are being moved to the PROJ namespace,-
19 while all data types are being moved to the PJ namespace.-
20
21 Please note that this API is "orthogonal" to the previous APIs:-.
22 Apart from some inclusion guards, projects.h and proj_apl.h are not-
23 touched - if you do include proj.h, the projects and proj_apl-
24 APIs should work as they always have.-
25
26 A few implementation details:-.
27
28 Apart from the rearranging efforts, I'm trying to eliminate three-
29 proj_apl elements, which I have found especially confusing.-
30
31 FIRST and foremost, I try to avoid projdefining away pointer-
32 semantics. I agree that it can be occasionally useful, but I-
33 prefer having the pointer nature of function arguments being-
34 explicitly visible.-
35
36 Hence, projCtx has been replaced by PJ_CONTEXT .-
37 and projP has been replaced by PJ .-
38
39 SECOND, I try to eliminate cases of information hiding implemented-
40 by redefining data types to void pointers.-
41
42 I prefer using a combination of forward declarations and projdefs.-
Hence:-.
43
44 projdef void *projCtx ;-
Has been replaced by:-.
45
46 projdef PJ_CONTEXT *projCtx ; Pj CONTEXT;-.

```

NORMAL > +0 -0 -0 master <c/proj.h> cpp utf-8[unix] < 9% : 40: 1 <! trailing

```

1 /> Project: PROJ-4-
2 Purpose: Primary (private) include file for PROJ-4 library.-
3
4
5
6 ****
7 Copyright (c) 2000, Frank Weller
8
9 * Permission is hereby granted, free of charge, to any person obtaining a-
10 * copy of this software and associated documentation files (the "Software"),-
11 * to deal in the Software without restriction, including without limitation-
12 * the rights to use, copy, modify, merge, publish, distribute, sublicense,-
13 * and/or sell copies of the Software, and to permit persons to whom the-
14 * Software is furnished to do so, subject to the following conditions:-
15 *
16 * The above copyright notice and this permission notice shall be included-
17 * in all copies or substantial portions of the Software.-
18 *
19 * THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS-
20 * OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,-
21 * FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL-
22 * THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER-
23 * LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING-
24 * FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER-
25 * DEALINGS IN THE SOFTWARE.-
26 ****
27
28 /* General projections header file */
29 #ifndef PROJECTS_H_
30 #define PROJECTS_H_
31
32 #ifdef _MSC_VER
33 #ifndef _CRT_SECURE_NO_DEPRECATED
34 # define _CRT_SECURE_NO_DEPRECATED
35 #endif
36 #ifndef _CRT_NONSTDC_NO_DEPRECATED
37 # define _CRT_NONSTDC_NO_DEPRECATED
38 #endif
39 /* enable predefined math constants M_* for MS Visual Studio workaround */
40 #ifndef _USE_MATH_DEFINES
41 # define _USE_MATH_DEFINES
42 #endif
43 #endif
44
45 /* standard inclusions */
46 #include <math.h>
47 #include <stdio.h>
48 #include <stdlib.h>
49 #include <string.h>

```

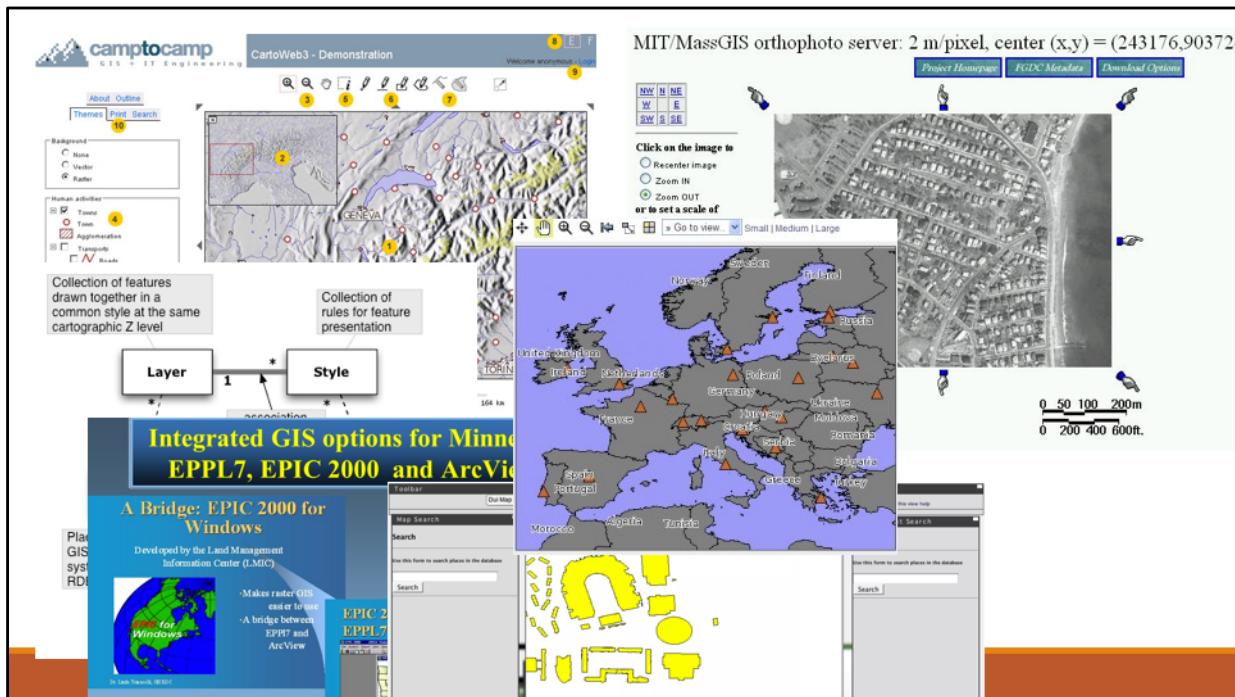
NORMAL > +0 -0 -0 master <src/projects.h> cpp utf-8[unix] < 0% : 1 <! trailing[283]

The C programming language helped smooth over many rough parts as a system language, but connection of tools like PROJ to others was still very challenging with C in the 1980s and 90s.

Assumed API norms that are common today weren't settled

Hardware constraints of memory, cpu, and storage meant program efficiency often overshadowed

ease of code reading, ease of integration, or ease of reuse.



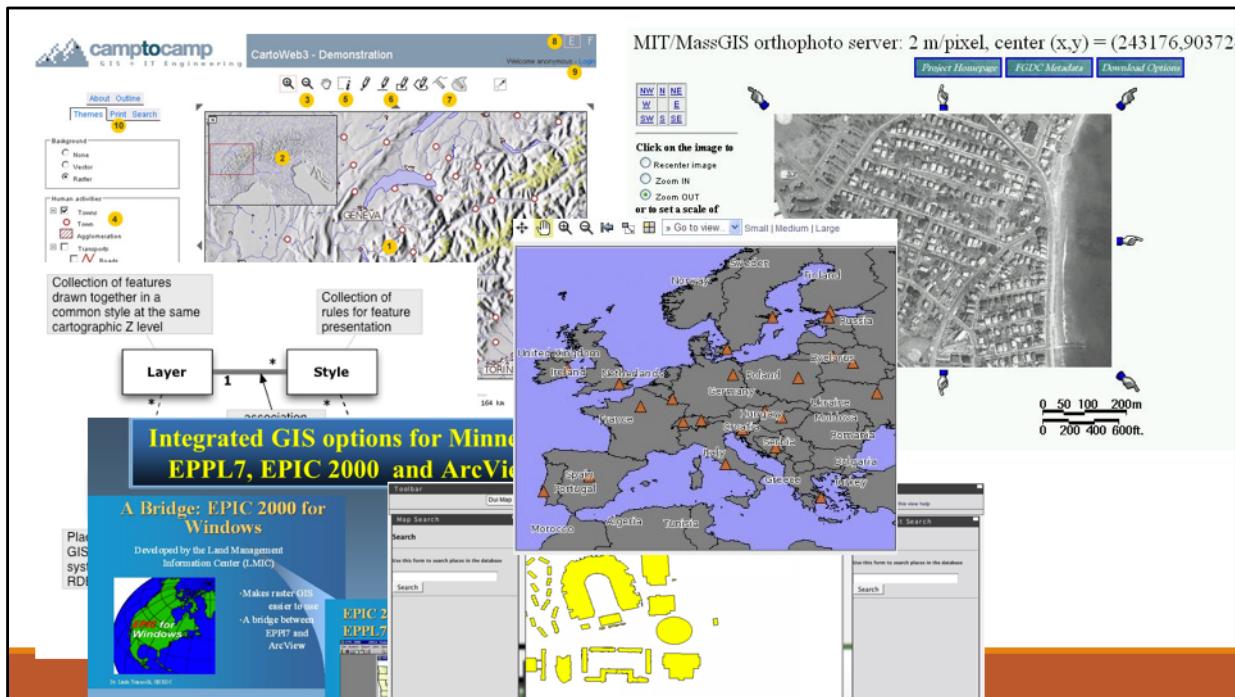
Successful open source software has a longevity problem.

Authors don't optimize today's creations  
with the expectation that they will  
survive the culture that produced them.

Here's a list of geospatial cultural dead ends  
I've contributed to or used in the past:

OpenEV.  
FWTools.  
OGDI.  
KaMap.  
CartoWeb.  
MIT OrthoServer.  
Zope Cartographic Objects

You might remember some of these names.  
It took me a while to come up with them actually,  
but I know I contributed many hours to more than a few of them.



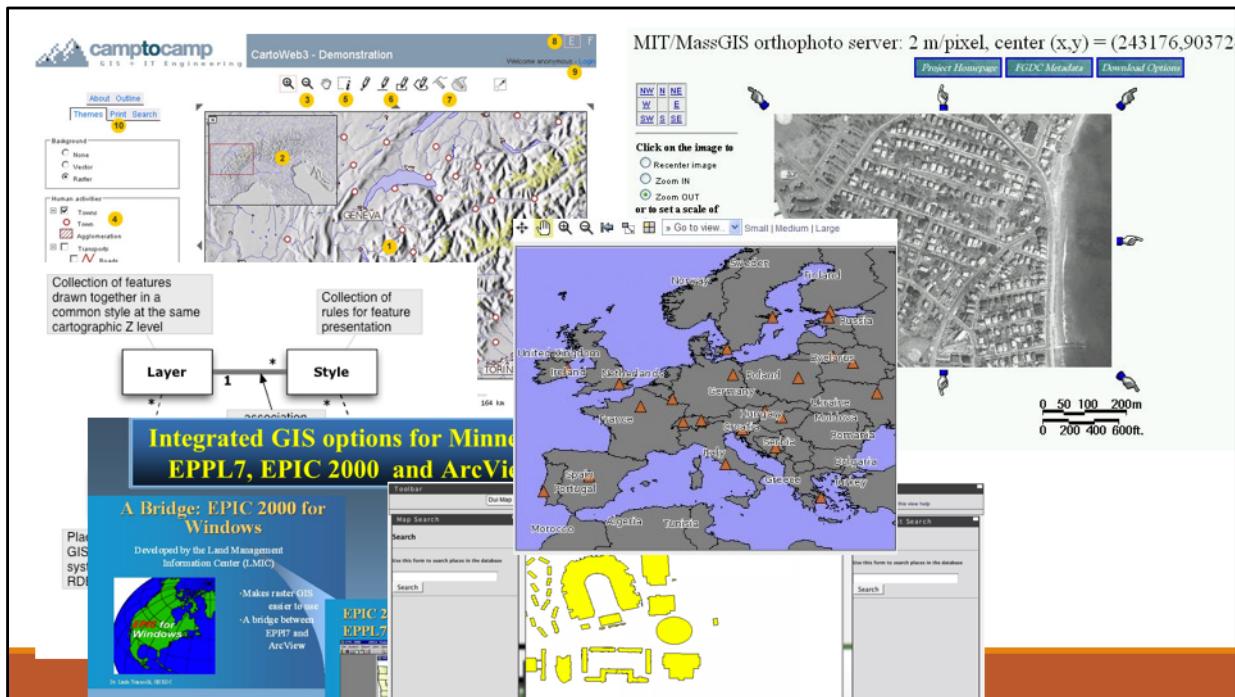
They're just gone to the ether,  
and no one pays any attention to them anymore.

The problems they solved might be no longer relevant

The problems they solved are done better solved by modern tools

The project's majordomo moved to more  
interesting problems, and no one has stepped into their place.

My own attic includes a bunch of ridiculously  
named stuff you've never heard of  
based on technologies you've forgotten:  
PySDE. Hobutools. PyTerra. libLAS.



Attention is life for open source software, and it is gone for each of these.

Without attention, these tools faded into dust with aspects of the problems they addressed being resolved in the context of a new development cultures.

PROJ is a software too critical to our geospatial ecosystem to let wither, even if that means we need to translate it into today's development culture.

PROJ, however, wasn't designed in 1983 for 2017's computing problems. It was designed in 1983 for 1983's problems.



Roger Andre

Frank Warmerdam

The funny thing is PROJ has already  
jumped this gap once before.

In 1998, nearly five years after Jerry retired from the USGS,  
Frank Warmerdam started using PROJ  
to provide coordinate system transformation  
for one of those now dead projects -- OGDI.

Frank had been using **another**  
USGS created coordinate system library  
that few probably remember -- GCTP, but  
he liked PROJ's description language  
and the open source GRASS software project  
had already integrated PROJ to give him a head start.



Roger Andre

Frank Warmerdam

It was a simple thing that made him choose PROJ  
-- GRASS had already integrated it --  
but it reverberated our entire industry as  
Frank used it as the basis for a software stack  
that soon included  
libgeotiff, shapelib, and ultimately GDAL.

When Frank started contributed, PROJ had  
a coordinate system definition language and  
many implemented transformations, but  
it was still missing a number of pieces to make it  
a complete transformation library.

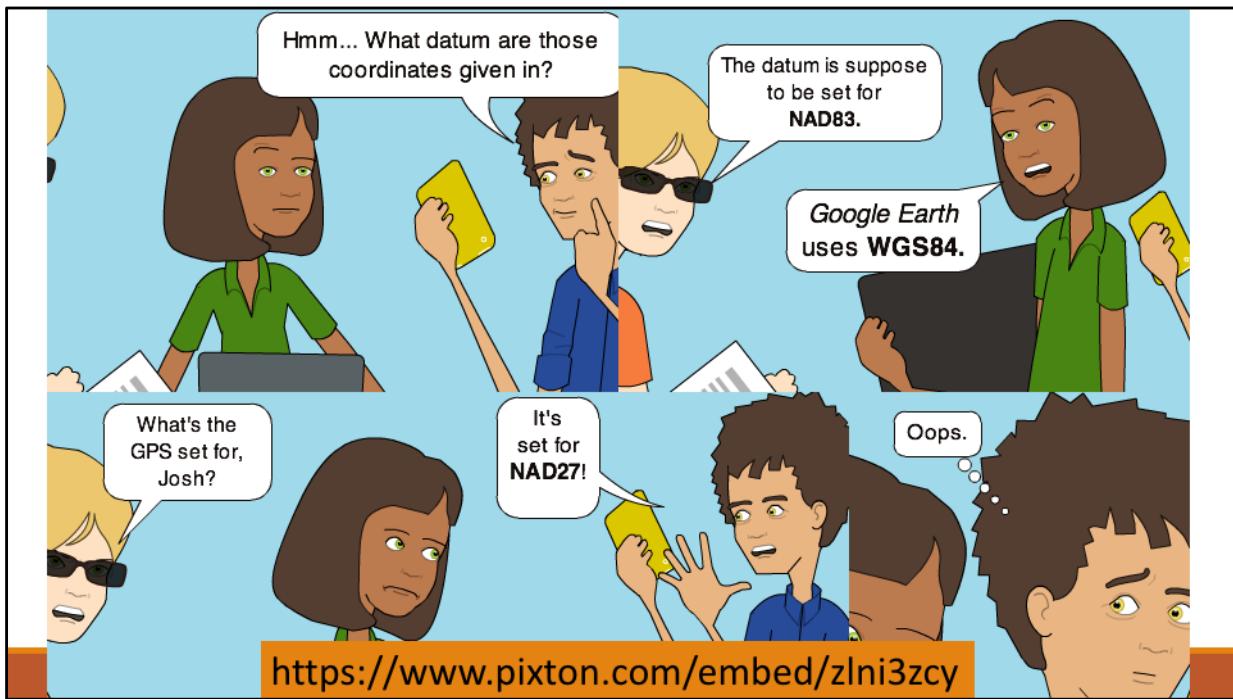


International  
Association  
of Oil & Gas  
Producers

*European Petroleum Survey Group*



It was missing the EPSG database,  
one with which we are now all so  
familiar to provide a dictionary of transform parameters.



It was missing the ability to transform between datums other than NAD83 and NAD27.

All source, data files and other contents of the PROJ.4 package are available under the following terms. Note that the PROJ 4.3 and earlier was "public domain" as is common with US government work, but apparently this is not a well defined legal term in many countries. I am placing everything under the following MIT style license because I believe it is effectively the same as public domain, allowing anyone to use the code as they wish, including making proprietary derivatives.

Though I have put my own name as copyright holder, I don't mean to imply I did the work. Essentially all work was done by Gerald Evenden.

-----  
Copyright (c) 2000, Frank Warmerdam

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

It was missing an open source software license, tests, and a revision control system -- cornerstone tenets of all open source projects.



- Thread safety
- API
- EPSG
- Pivot datum shifting
- License
- Mailing list
- Bug fixes

Frank built up PROJ's capabilities to include support for the EPSG database through distribution of CSV files in specially designated directories, and he enabled PROJ to apply datum transformations by pivoting through the WGS84 ellipsoid.

He coordinated the addition of modern features such as thread safety, error contexts, integration tests, and Makefiles and he assigned a true open source software license.

As Frank continued pushing the software forward, Jerry participated on PROJ's open source mailing list chiming in with knowledge, history, and opinions about the scope of the project.



- Thread safety
- API
- EPSG
- Pivot datum shifting
- License
- Mailing list
- Bug fixes

Jerry's original vision for PROJ did not include things like datum shifts, but the practical desires of software developers looking to include the capability from a single software library carried forward.

Throughout the 2000s, PROJ continued to improve in capability and reach, with its success culminating in other developers porting PROJ to different computing platforms such as Java, JavaScript, and .NET.

**PROJ.4 - Cartographic Projections Library**

This web page relates to the PROJ.4 Cartographic Projections library originally written by Gerald Evenden then of the USGS. The primary version of this web page can be found at <http://www.remotesensing.org/proj>.

## Download

The current development source is available by anonymous CVS. It can be checked out from server "pserver:anonymous@cvs.remotesensing.org:/cvsroot". For example:

```
% cvs -d :pserver:anonymous@cvs.remotesensing.org:/cvsroot login
Password: anonymous
% cvs -d :pserver:anonymous@cvs.remotesensing.org:/cvsroot co proj
...
```

The following files are available from the [proj ftp directory](#):

- [proj-4.4.5.tar.gz](#) or [proj-4.4.5.zip](#): Current source release.
- [proj-nad27-1.1.tar.gz](#): US and Canadian datum shift grids - untar in the nad directory before configuring to add NAD27/NAD83 datum conversion.
- [PROJ.4-4.4.1-1.i586.rpm](#): Binary distribution for Intel Linux (glibc 2.1).
- [PROJ.4 in a Debian Package](#).
- [PROJ.4 Ported to the Delphi \(Borland C++\) environment](#).
- [PROJ.4.3.3.tar.gz](#): The last PROJ.4.3 release produced by Gerald Evenden (classic PROJ.4).
- [from\\_kai](#): A snapshot of everything on the kai.er.usgs.gov/pub/PROJ.4 directory as of March/2000.

## Documentation

- [OF90-284.pdf \(2.7MB\)](#): The main users manual for PROJ; however, this dates from PROJ.3 - the addendums [PROJ.4.3.pdf \(1MB\)](#), [PROJ.4.3.12.pdf \(2MB\)](#), and [SWISS.pdf \(78KB\)](#) contain important additional information.

Frank kept PROJ chugging along until about 2011 when he answered a Google recruiter email. He eventually moved on to PlanetLabs, and has been busy there ever since bootstrapping their massive data processing backend.

Planet uses PROJ, GDAL and other open source software. Frank pioneered with the goal of capturing a 3m resolution satellite photo of the entire earth, every day. They are well on their way to achieving that too, with over 100 micro satellites now capturing data.

Frank's moving on caused PROJ languished a bit. The bug tracker continued to fill up with items, including security issues, but releases stagnated.

## Open Source LiDAR



My company, Hobu, Inc., writes LiDAR software called PDAL, which is very dependent upon the ability to transform between different coordinate systems.

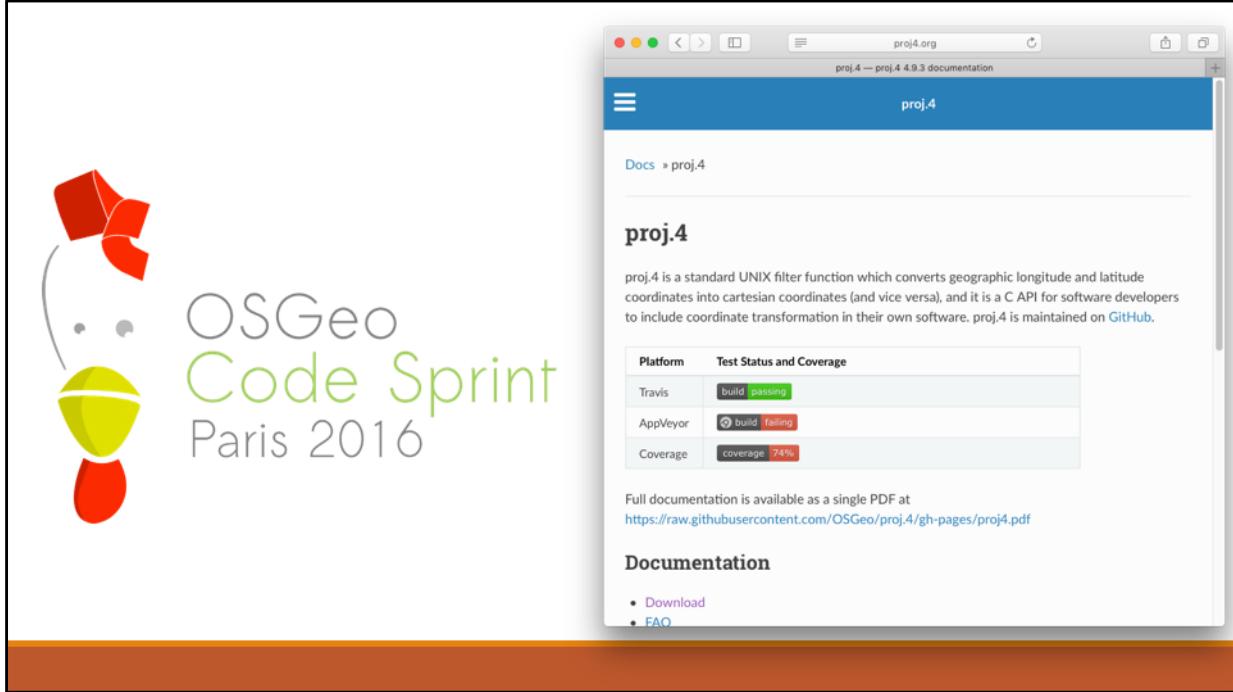
I took it upon myself to take on the task of applying and shepherding fixes in the bug repository and organizing an approximately annual release process.

I don't know very much about the intricacies of coordinate transformations, but I do know plenty about shepherding open source software, issuing releases, and keeping the train moving.

With that experience, I started to organize and issue roughly annual releases to collect and squash bugs, patch any security items, and include small new features.

After a few releases, it was clear to me that the documentation infrastructure for PROJ was terribly deficient.

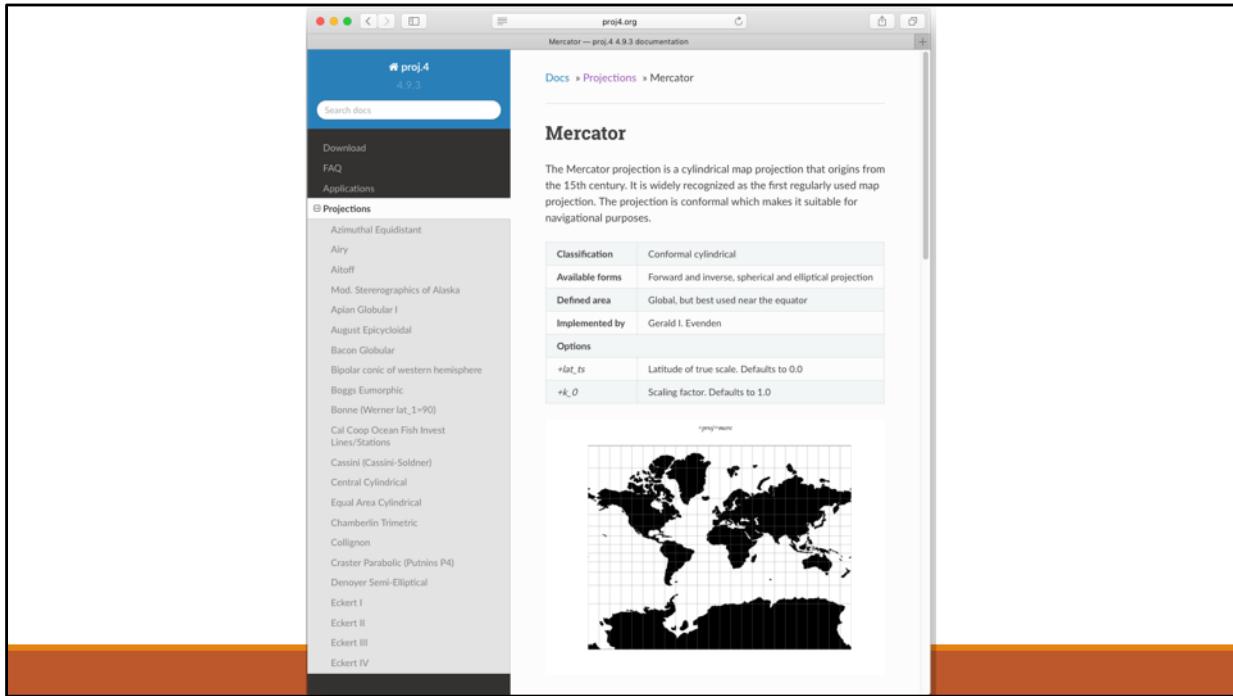
There were a couple of very old PDFs that Jerry had written while he was still working at USGS available online, bits and pieces of unorganized wiki pages spread across about three generations of project infrastructure, and some light source-tree documentation.



During the OSGeo Paris Code Sprint in 2016,  
I embarked on improving PROJ's infrastructure  
to allow the project to start to treat the documentation  
just like it was treating the code.

After the Paris effort, every commit to the repository  
now regenerates the entire project website,  
including a single PDF with its entire contents --  
approximately 135 pages long.

Other contributors such as  
Kristian Evers,  
Julien Moquet, and  
Elliot Sales de Andrade  
saw the website effort and started working  
to pull forward content and port those PDFs of Jerry's into



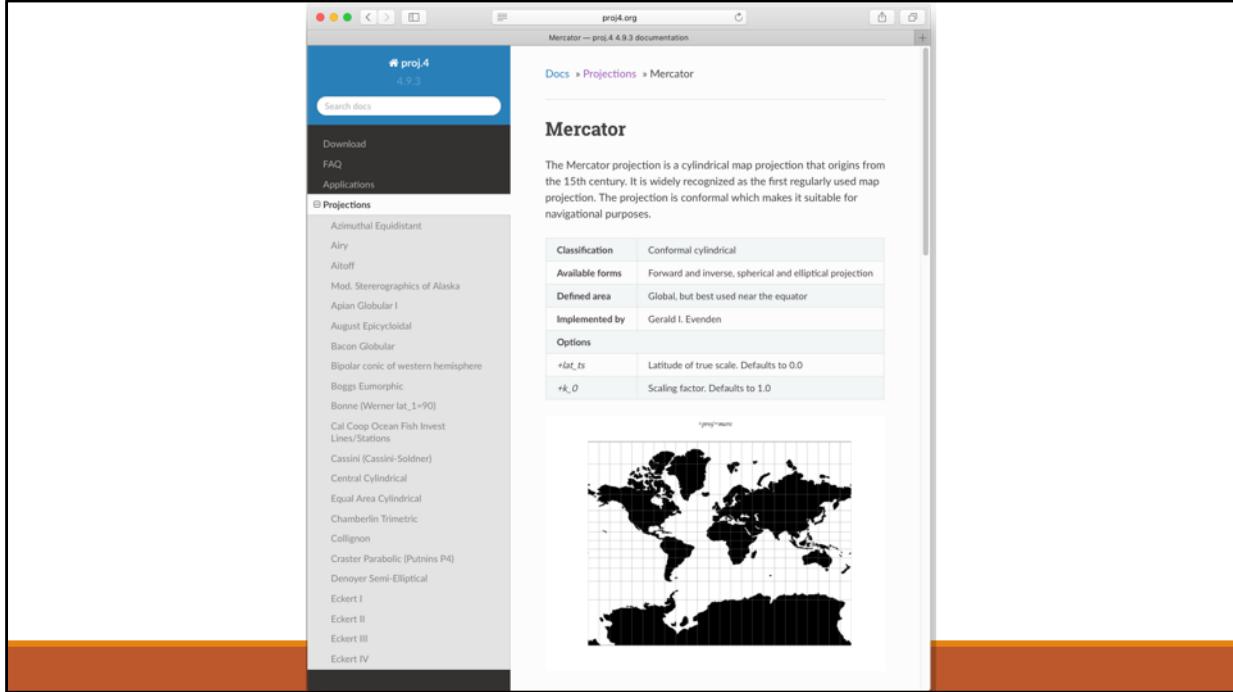
a corpus of supported projections,

clear equations for forward and inverse methods

a nice representative graphic showing what each projection might look like

and some notes on usage and caveats.

After implementing the Sphinx documentation system  
and coordinating a spiffy new URL for the website,  
proj4.org was enabled, and  
we worked to pull down all of the old,  
misleading, and out-of-date bits spread throughout the internet.



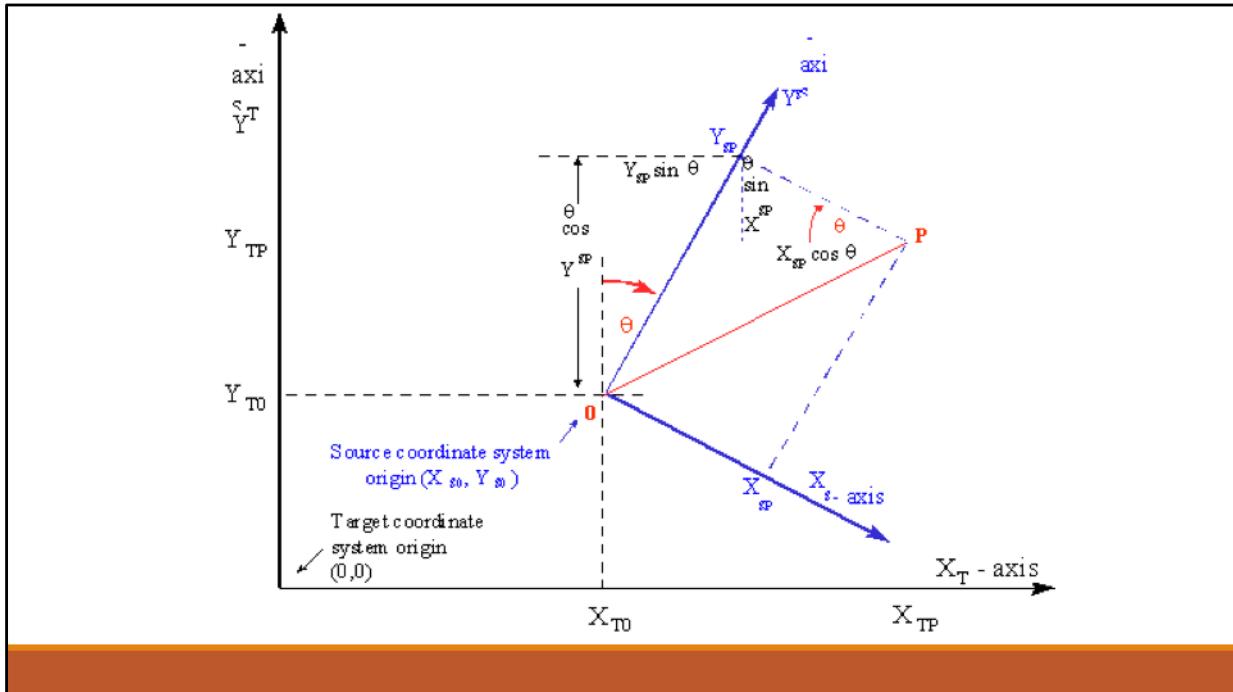
There's still plenty of work to do.

Most of the 130+ projection methods PROJ supports need their documentation completed.

Some of that information still resides in Jerry's old PDFs, and meticulous effort is required to verify the equations in those documents match the math in the actual software.

Even so, the new website is a dramatic improvement over the previous situation, and documentation infrastructure is critically important to a software project like PROJ with a multi-generational lifespan.

All it took to improve the situation was attention.



Seeing renewed interest in PROJ after the website refactoring,  
 Kristian Evers and Thomas Knudsen  
 of the Danish Agency for Data Supply and Efficiency,  
 which is kind of like the Danish equivalent of the US Geological Survey, embarked in  
 2016 on an effort  
 to bring a formalized approach to datum transformations to PROJ

*”\*A lot of luck\* to whoever wants to put together the computational part of the datum shift software.”*

**Jerry Evenden, 2000**

As Jerry said,

"A lot of luck to whoever want to put together the computational part of the datum shift software."

Thomas is a long-time contributor to PROJ, and his first contribution was in 1999.

It was a massive undertaking.

Thomas and Kristian started by porting a library called trlib,  
originally written by the Danish agency in Algol in 1961.

The end result is new versions of PROJ will support  
14-parameter, time-dependent shifting, and  
users will be able to create their own shifts  
using Thomas' generic transformation pipelines.

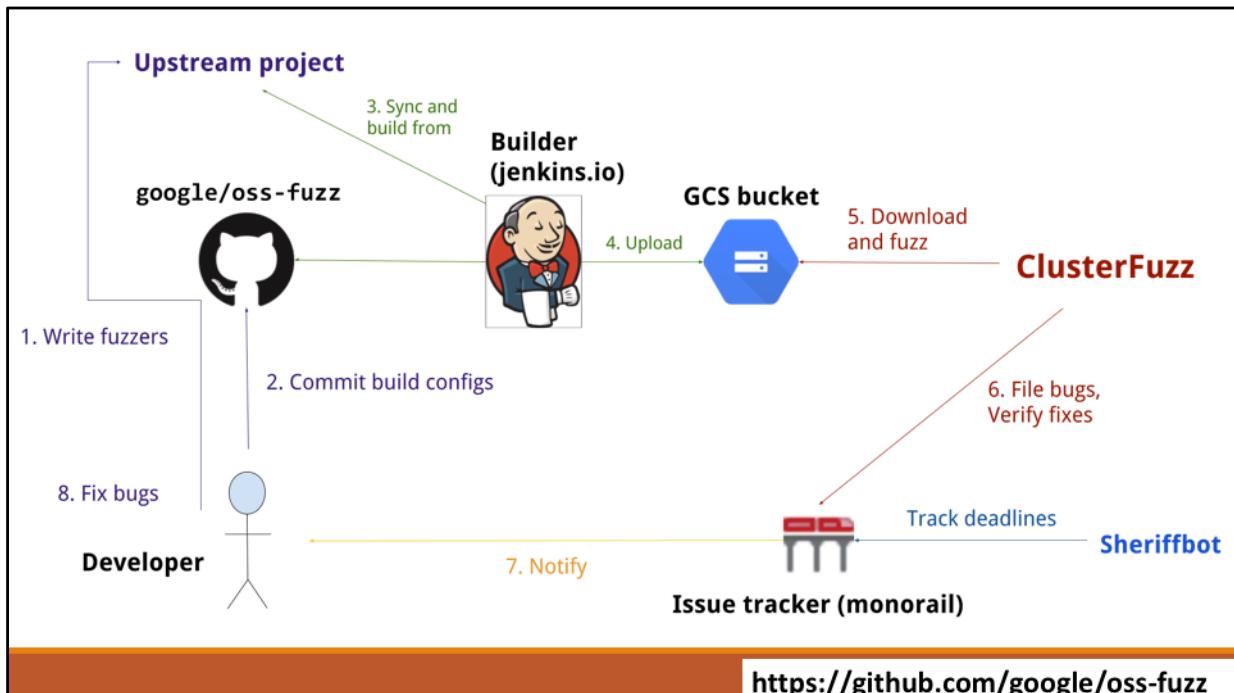
*”\*A lot of luck\* to whoever wants to put together the computational part of the datum shift software.”*

**Jerry Evenden, 2000**

Thomas and Kristian targeted PROJ for this contribution for a couple of slightly selfish reasons --

PROJ is the most widely used coordinate system transformation library, and their hope was wider availability of transformation pipelines will mean broader support of their precision needs flowing back toward them in commercial software.  
It's the virtuous circle of open source.

More attention begets more attention.



Even Rouault recently submitted PROJ to the Google Open Source Software Fuzz project. Fuzz clones the PROJ software repository every day, builds it, and runs tests designed to highlight

buffer overflows,  
uninitialized memory,  
and exploitable assertions.

When Fuzz finds something, it automatically creates a bug report in a private repository for developers to fix. After a month, that bug report becomes public.

Even has also been using this capability for improving the security hardening of GDAL project too.

For projects such as GDAL and PROJ, tools like Fuzz keep them relevant and safe in today's harsh computing environment. This is especially true in situations where code written in 1983 potentially processes random input over the internet.

## Thanks Jerry!



Jerry's initial development and Frank's maintenance made the library an indispensable tool in most geospatial people's toolkits, whether they use it explicitly from the command line, or embedded in software that hides it from them.

New contributions from people like Thomas and Kristian and Evan will keep it going for an unpredictably long future.

Thanks Jerry for letting all of us use PROJ, which has and will continue to impact the world by providing the math to compute where we are.