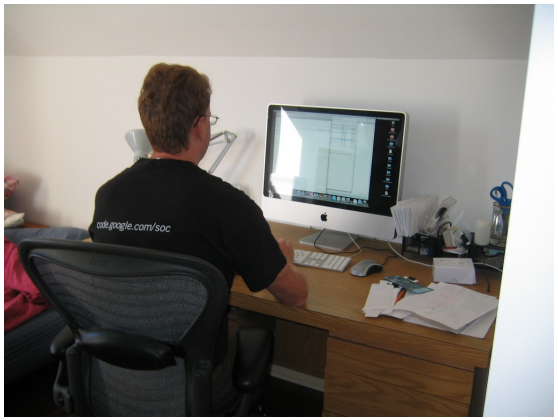
Open Source and the Spatial Web

Paul Ramsey
Boundless
pramsey@boundlessgeo.com

paul ramsey

My name is Paul Ramsey

OpenGeo Suite

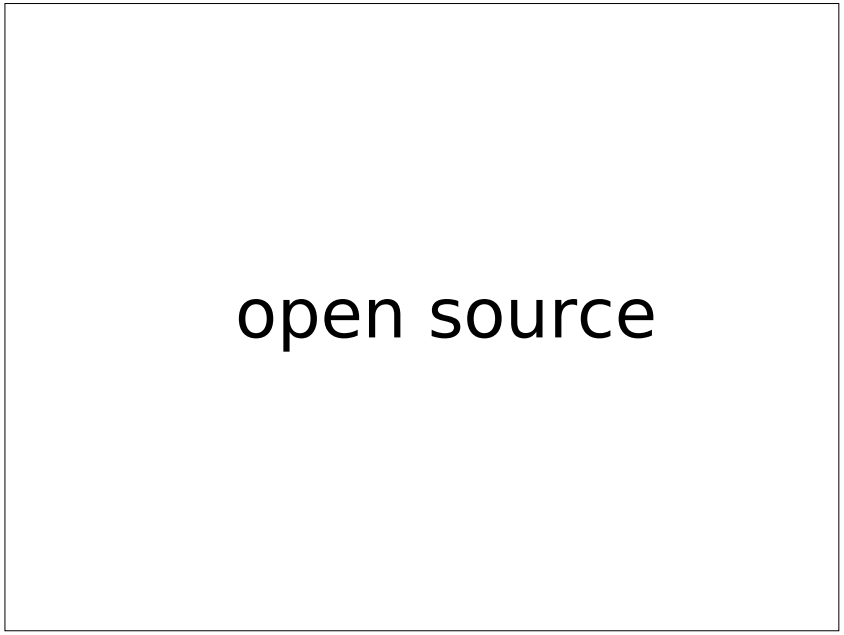


I'm the product manager for the OpenGeo Suite from Boundless, which is commercially supported open source software for building spatial web apps

I'm also a long time open source developer.

I founded the PostGIS open source spatial database project 13 years ago, and have been actively involved in it every since.

Today I'm going to talk about three big users of open source web tools,
<x> the City of New York,
<x> the Portland regional transit authority TriMet,
<x> and the Federal Communications Commission.
Three levels of government (municipal, regional, federal),
who all built high traffic, public facing
web services and applications on an open source platform.



First, open source.
Open source deserves some words of explanation, because it's not widely understood outside of the technology arena
(and often not even there).

Open source software is software that is distributed with three basic conditions:

the recipients are free to share the software with others
<x> the recipients are free to modify the software if they wish
<x> the reciptents are free to share their modified software with others

to m

Because all open source software comes with these special freedoms,
it is sometimes interchangeably called "free software",

but the reference is not to price it's to the three freedoms.

Open source is a relatively new concept,

richard stallman

1985

it was "invented" from a software perspective by Richard Stallman in the 1980s.

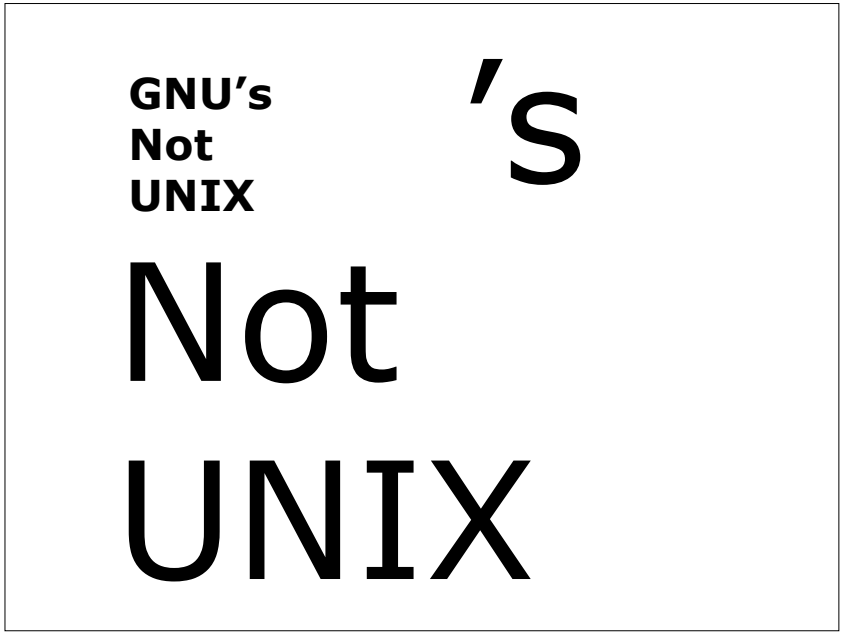
But it’s not a new idea,
it's worth noting, that the three freedoms are also the basis of Western science:

this is science

discover
publish
verify results
publish

when you discover an interesting scientific result you
<x> publish, to share your results with others,
<x> who verify them, advance them, improve them,
<x> and in turn share their results with others.

GNU’s
Not
UNIX



Stallman and his early followers in the 80s built the first substantial collection of open source software for what they called
<x> the GNU operating system (GNU is a recursive acronym that stands for
<x> "GNU's not UNIX", see the recursion?)

GNU's Not UNIX

's

Not
UNIX

's

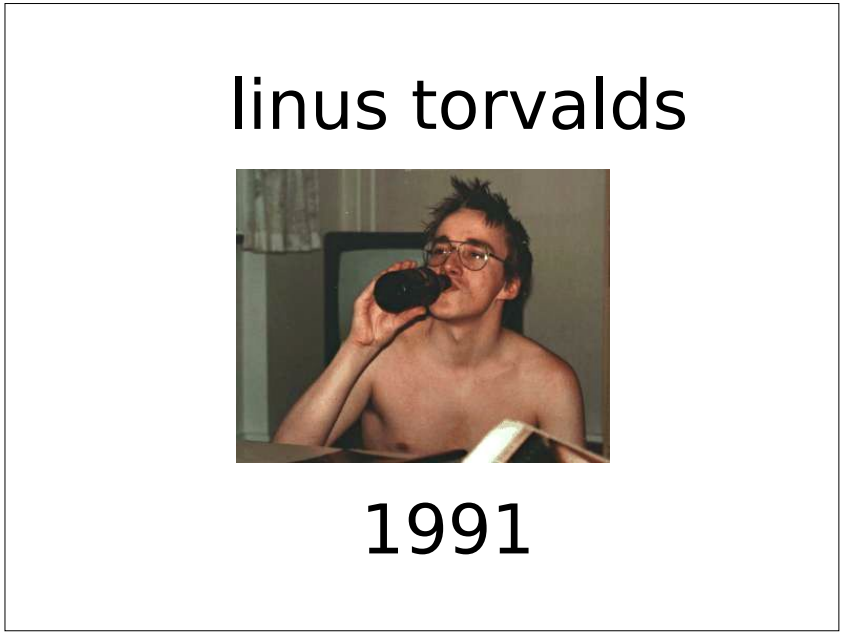
Not UNIX

They built all the components of an operating system. Standard libraries, compilers, shells, code editors, everything except the bit that talked to the underlying hardware, the "kernel".

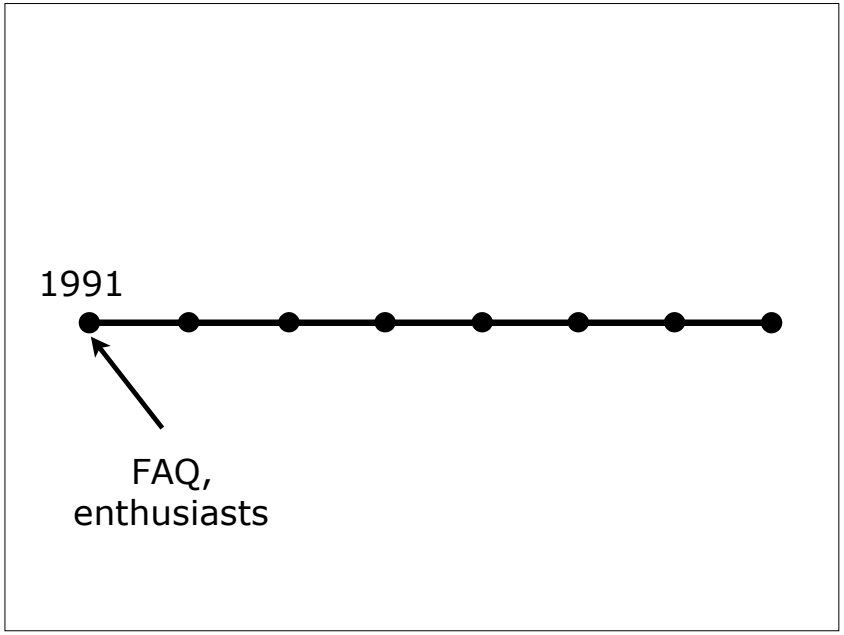
For his work on GNU and open source, Stallman received a Macarthur genius award.

But, I think he'd probably give it back to trade places with this next guy.

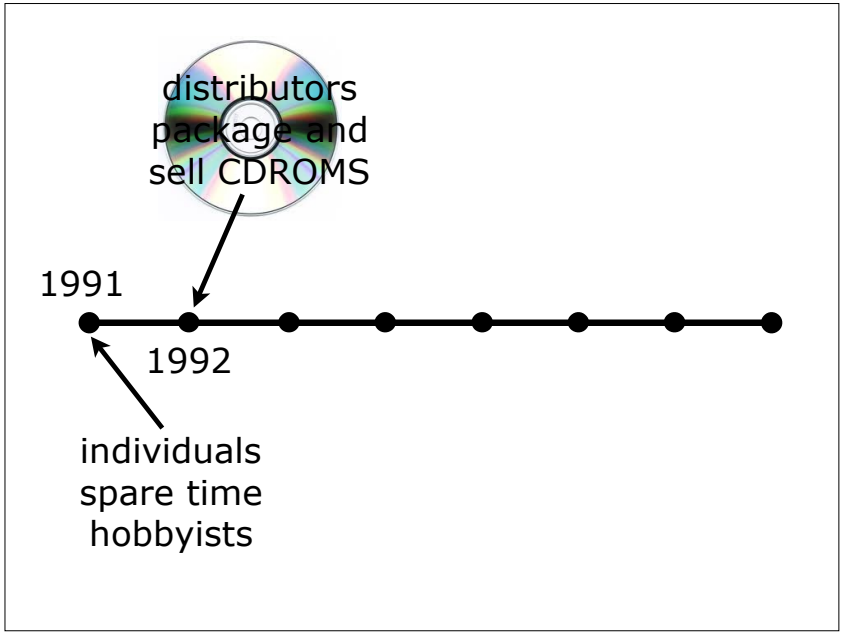
In 1991, frustrated by the lack of a UNIX operating system for his fancy new intel 386 computer (bought with christmas money from his grandmother)



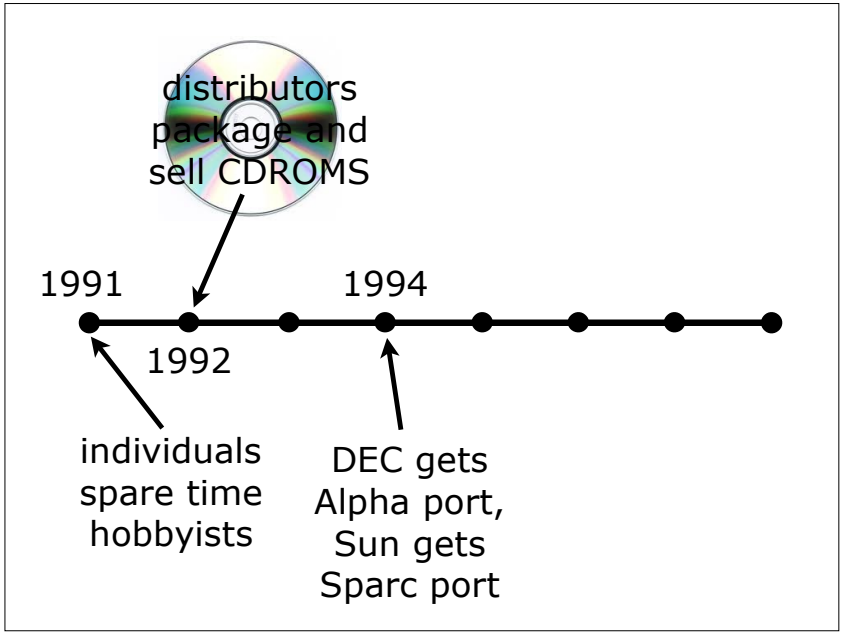
Linus Torvalds wrote a kernel for the GNU operating system, and posted it online.
It became an overnight success,
bringing the GNU UNIX platform to
inexpensive intel hardware perfect for individual developers.



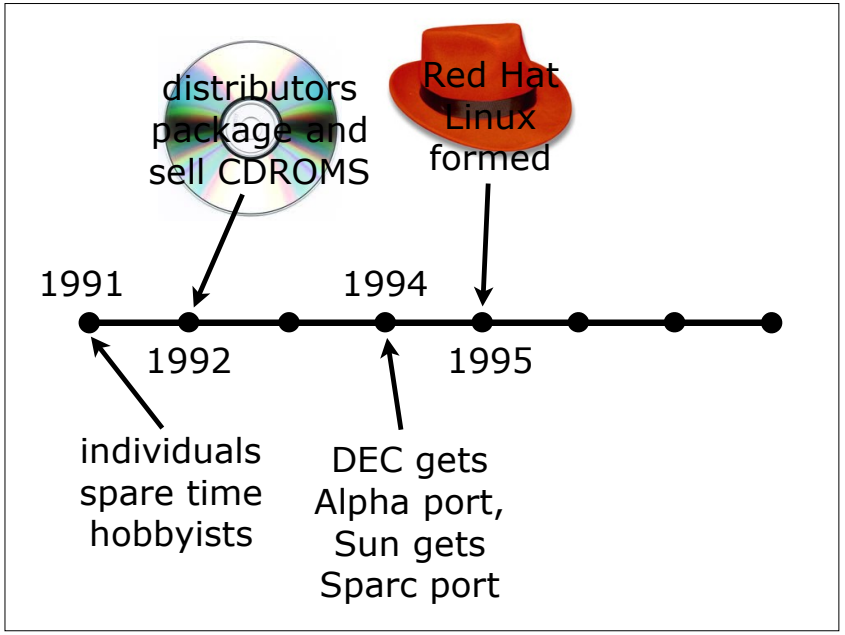
In four months, it is so popular an F.A.Q. has been written to handle the common questions.



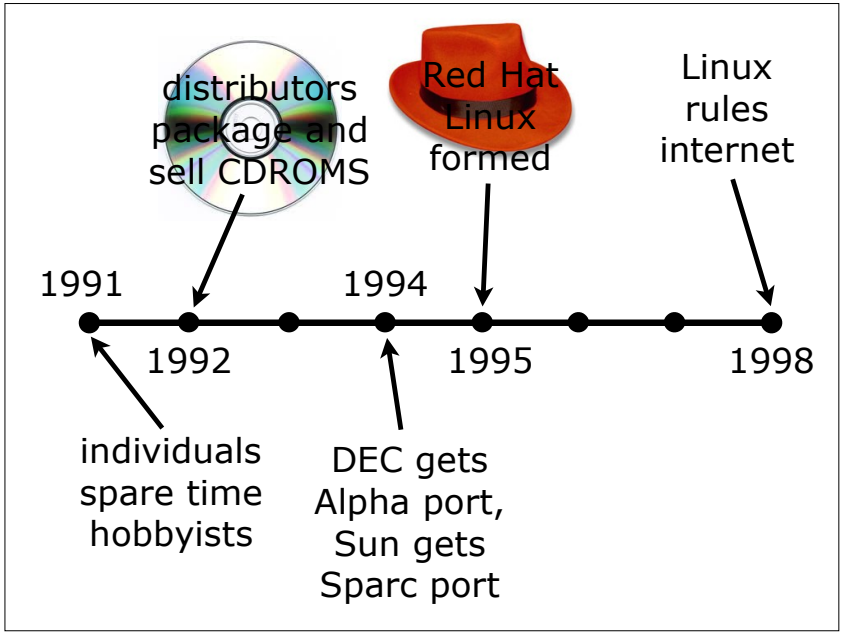
In a year, packagers were selling Linux DVDs that combine the Linux kernel with the GNU software tools.



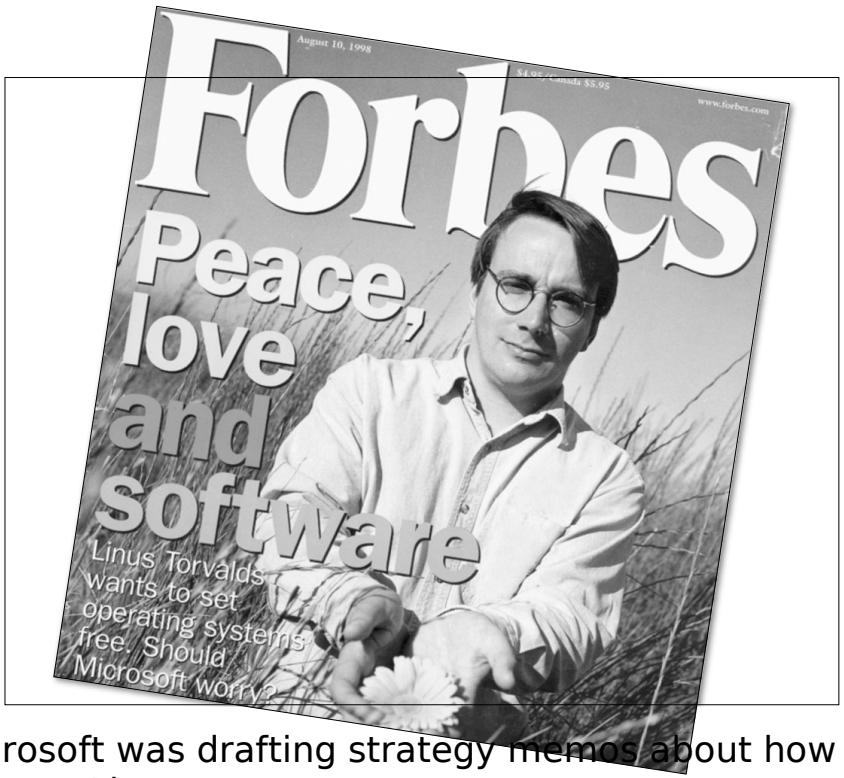
In three years, big UNIX operators like DEC and Sun were sponsoring ports of Linux to their hardware.



In four years, Red Hat had been founded, a future $2B open source concern.



By 1998 the internet explosion was in full swing powered by hundreds of thousands of commodity servers running Linux.



Microsoft was drafting strategy memos about how to counter Linux,
and Linus Torvalds is featured on the front page of Forbes magazine

That's seven years from prototype code to industry realignment.

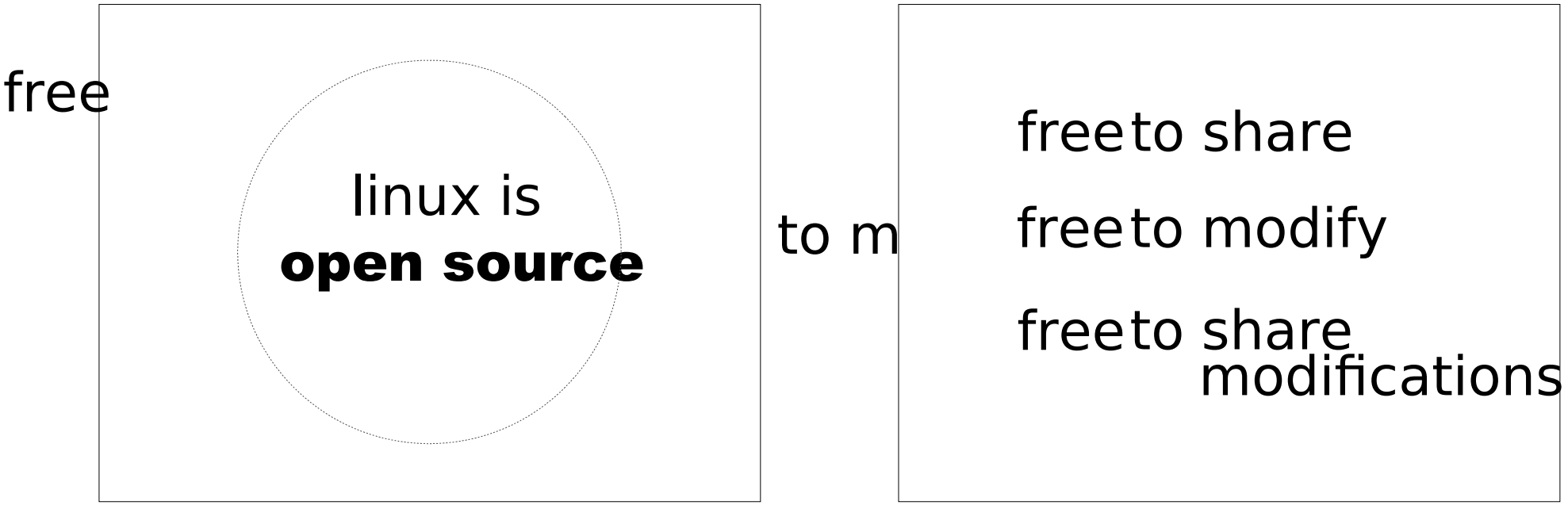
All without a marketing department, without licensing, without sales,
without ownership or control.

not just
linus
torvalds

Oviously, Linus didn't achieve all that alone.

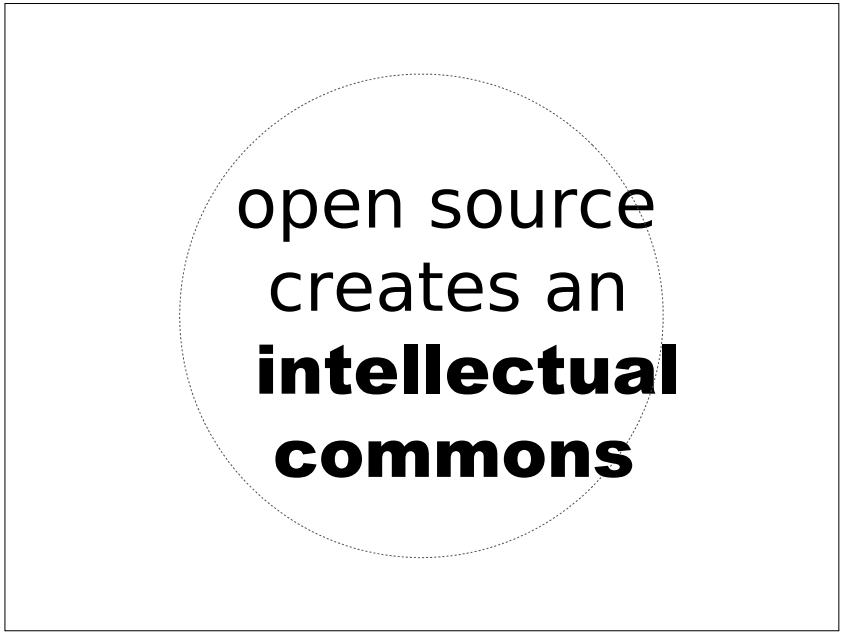
He achieved it working with ...

a huge community of like-minded developers who shared his goals.
But why would they share their work with Linus?



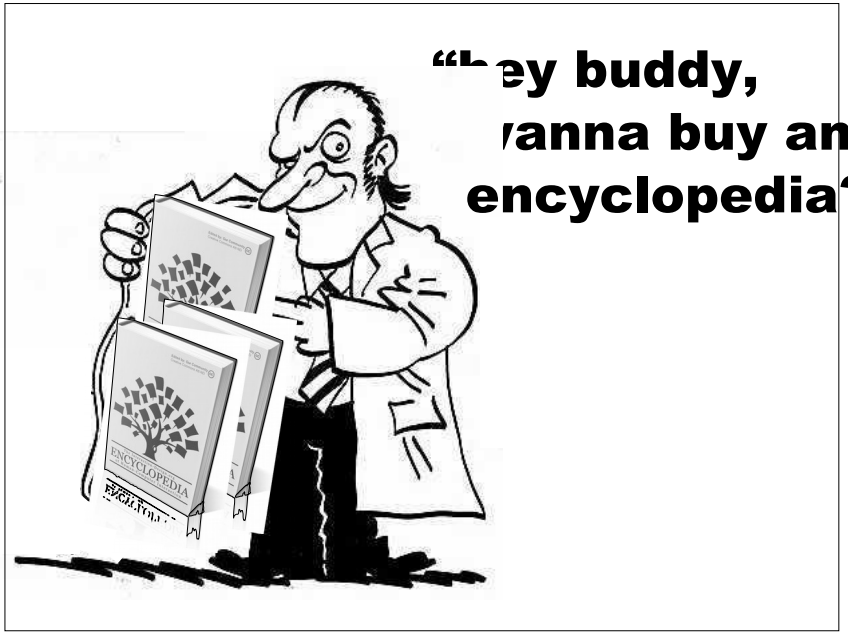
Because Linus released Linux under an open source license.

And the three freedoms meant that developers knew that any work they added to Linux would always be available to them. Linus couldn't just walk away with it.



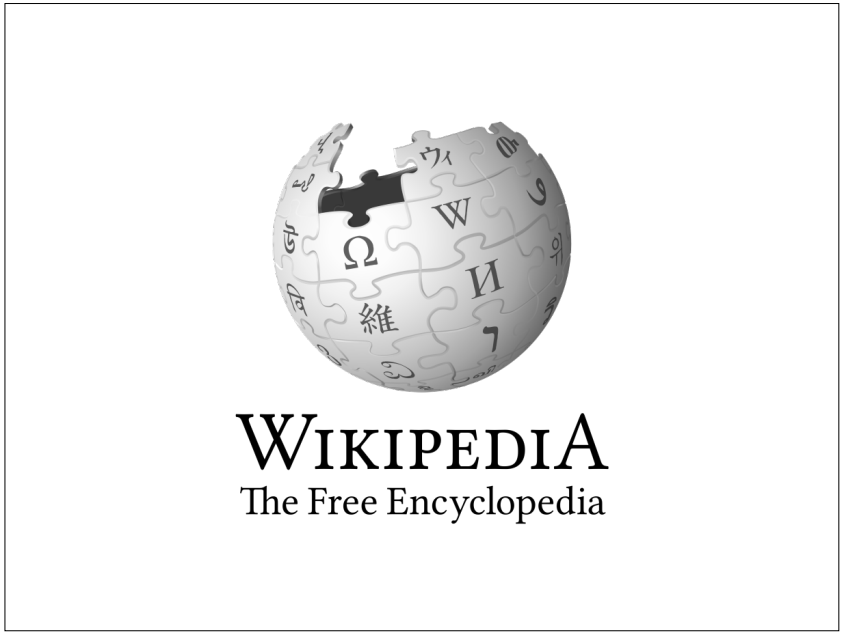
Open source sets up "intellectual commons" that can never be fenced off or alienated.
People are willing to contribute to a commons in ways they are not willing to contribute to a private intellectual pool.
People pull weeds in the community garden, they don't do it in my front yard. (Which is a shame, I think.)
Open source software is not an isolated example of this dynamic. The arrival of the internet has caused numerous "intellectual commons" to spring up.

Last year, after a couple hundred years of publishing, the Encyclopdiea Britannica announced that they wouldn't be printing encyclopedias anymore.

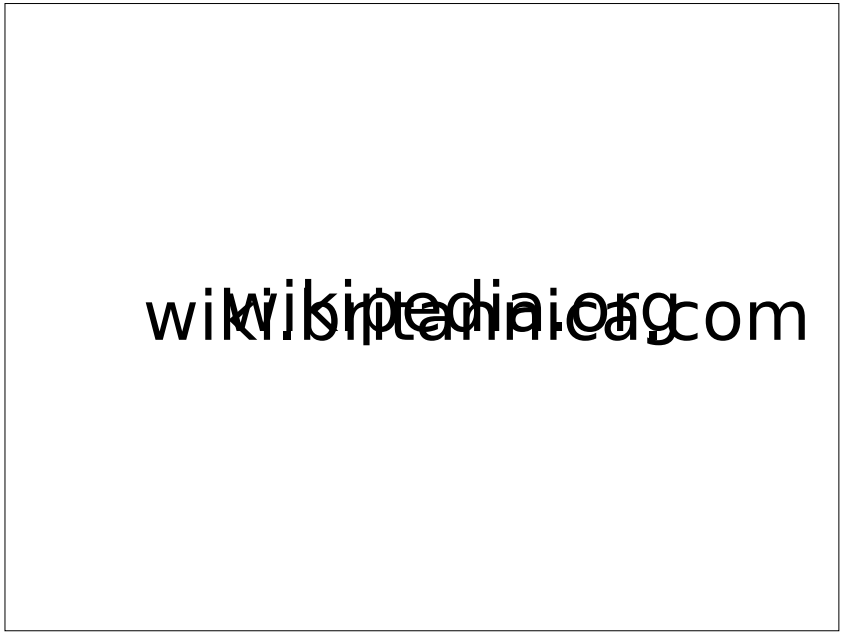


Not because they'd been put out of business by encyclopedia theft, or encyclopedia copying, or anything like that.
They've just been outcompeted. And not by the World Book, or National Geographic.

By a decentralized community of writers working together to build an intellectual commons around factual knowledge.

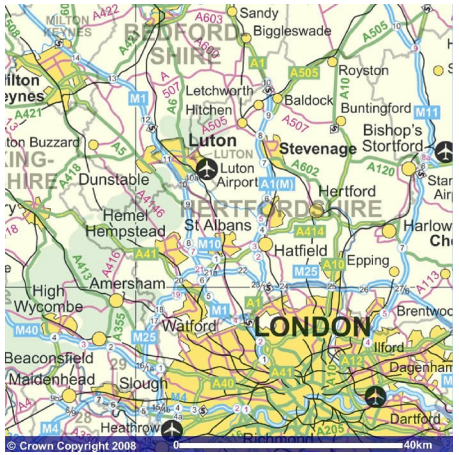


By Wikipedia. An online intellectual commons.
It's not like Britannica didn't see it coming,
but they couldn't stop it.



They couldn't set up wiki.britannica.com, because people wouldn't contribute to it.
People don't like giving their work to other people for free.
<x> But they give it to Wikipedia, why?

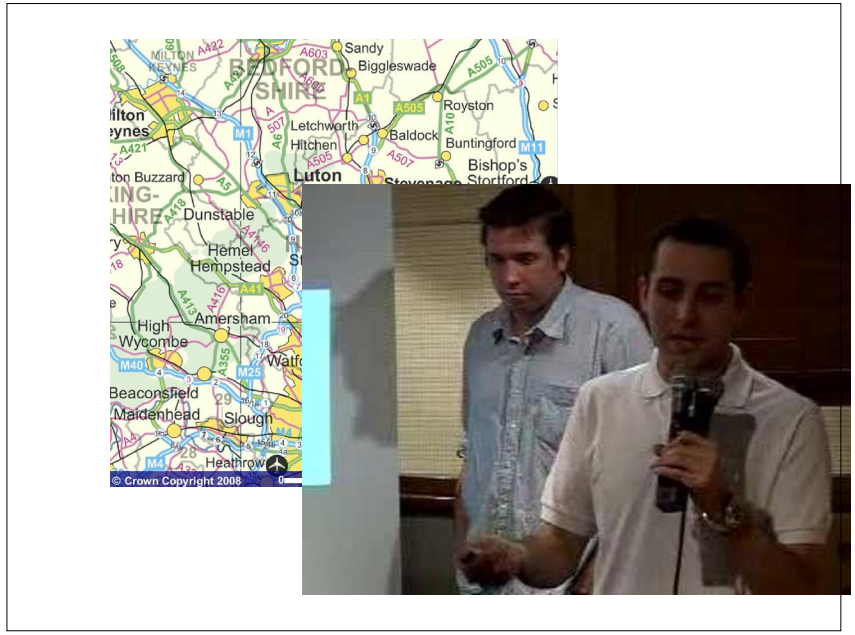
creative commons



Look at the license Wikipedia uses for its content, the "Creative Commons" license.
It's basically an open source license.
You're free to share, to modify, and to share modifications of Wikipedia content.

How about an example from our own field...

Almost all geographic data in the UK is collecfted by the Ordnance Survey, and they sell it. If you don't have money, you can't have it.



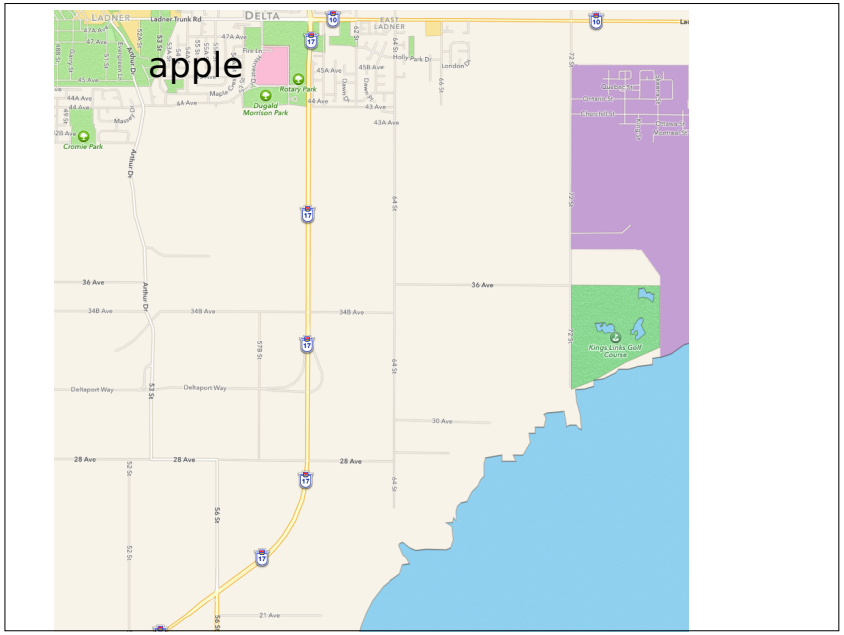
A couple British university students got annoyed at that.
So the students started an intellectual commons for map data, and called it Open Street Map.

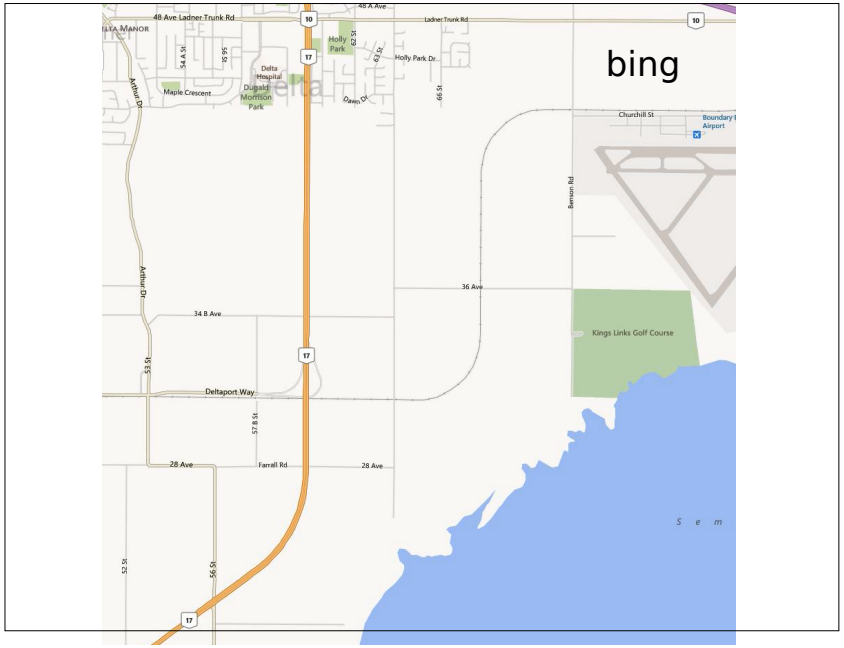
It started off small, but it grew quickly because it filled an important need. First in places like the UK, Germany and India, where map access was expensive or illegal.

Then in places like Canada, where it provides a more up-to-date view of the world than institutional mapping.

I recently had the pleasure of getting lost on the South Fraser Connector after getting off the ferry.

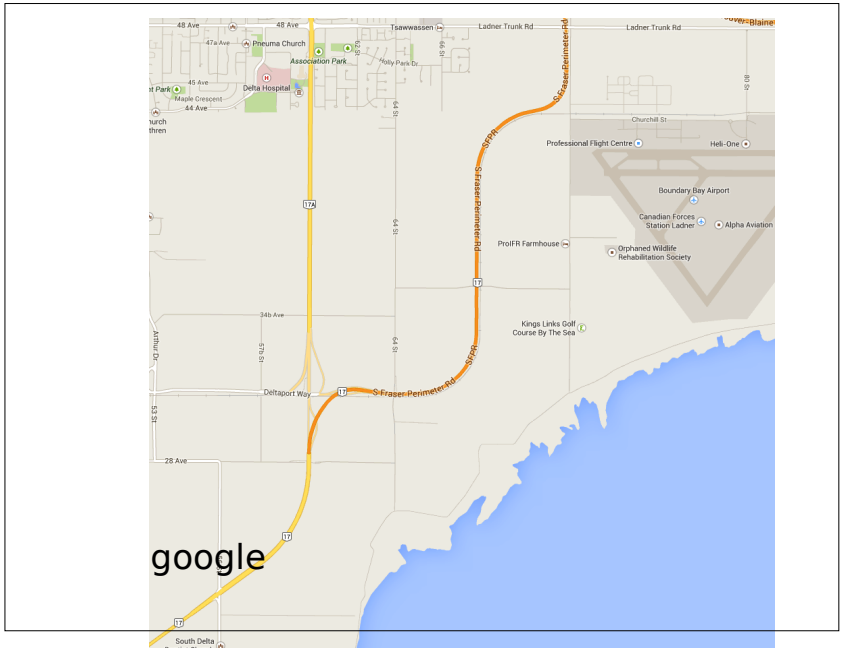
This is a multi-billion dollar infrastructure addition that has been underway for years, it’s not some new culdesac.



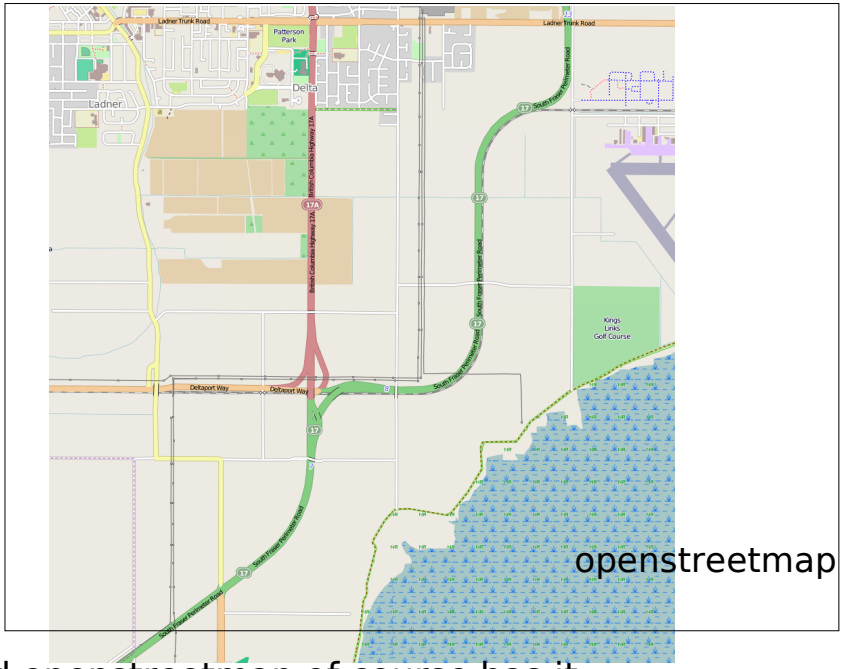


Here's the view of the connector from Apple (it doesn't even exist)

And Bing Maps (doesn’t exist there either)



Google’s got it (though some of the lines seem tentatively colored)



And openstreetmap of course has it.

So, open source is just the "software" aspect of a new way of building knowledge,

where communities of interest collaborate over the internet
and build things that previously required large institutions to create

it's like magic, but it's not,
it's the combination of,

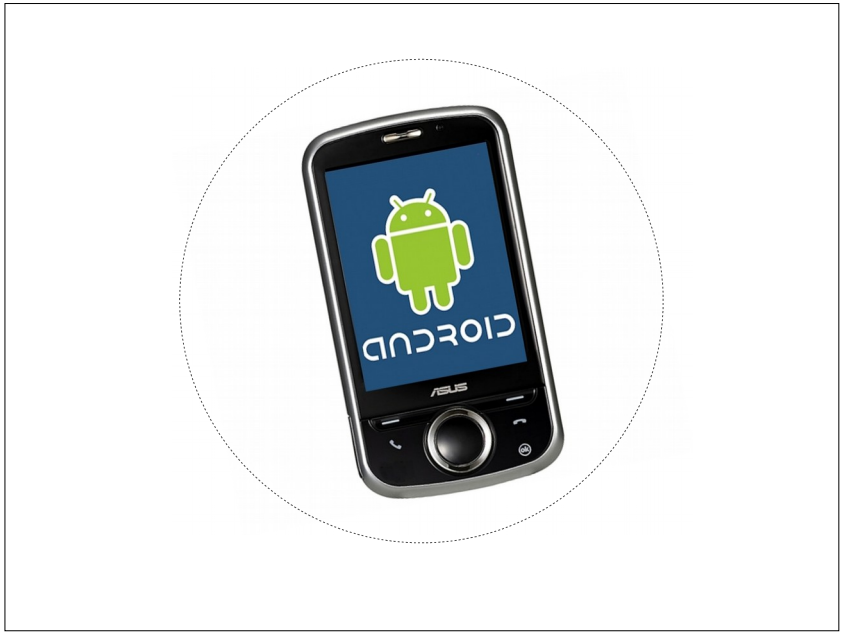
shared interests
rules for sharing
communications tech

open source
is the
new normal

shared interests in the work products
<x> rules for sharing that put everyone on an equal footing (like open source licenses or creative commons)
<x> and internet communications technology to join everyone together.

Open source style collaboration is not an aberration, it's not a new fad, it's the new normal.

Certainly open source SOFTWARE is now completely ubiquitous,
to the extent that I have to actually point it out. You got up this morning,



- Turn on Android smart phone? Open source. (OS kernel of iPhone is also open source.)

- Searched Google? Wouldn't exist without open source. Hundreds of thousands of Linux servers.

- Checked Facebook? Same story. Both using existing OSS tech, and releasing new OSS tech.

- Packets traversed your wireless router. Open source.

- Adjusted Nest thermostat? Open source.

- Put dinner in the crock pot? Open source.

#1
we go with
what we know

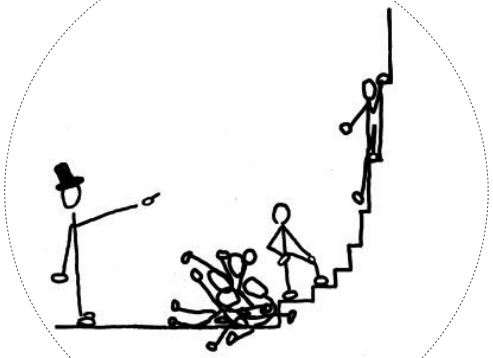
- Watched a movie on the plane? Open source.

So, in the internet service and embedded device spaces in particular,
open source is ubiquitous.

How about in our world? In the municipal, regional geo-world?
Less ubiquitous, for sure. But why is that?

Well,
First of all,
People go with what they know, that's normal.
And what people know is something they first bought 15 years ago
and have just kept on buying,
usually ESRI, sometimes something else.

#2



open source
learning curve

#3

no marketing department

Second,
Open source can have an intimidating learning curve.
(In fact, one of our corporate goals as Boundless is to flatten that curve,
as our CEO says “our job is to reduce the cost of free software")

And third,
Open source has no marketing department.
So proprietary stories dominate our marketplace of ideas,

Which means people don't always talk about web mapping in a generic sense,

“you need something that does ____”

“you need the Whurligizmo™”

they don’t say
"you need these capabilities", or
“you need something that does THIS”
they adopt a shorthand

and start talking about capabilities
in terms of proprietary products or product analogues

But architecturally, you'll get better designs if you model your system in terms of generic capabilities, and then figure out what product mix fulfills your needs.

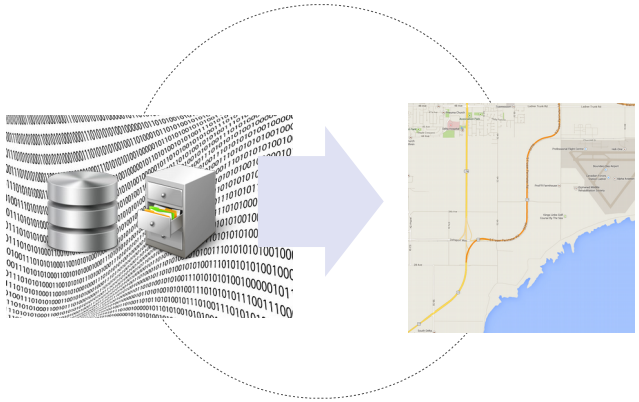
the
parts of a
web map

Here are the functional parts of a web map.
These are common to every web map,
whether it be Google Maps or Bing,
an open source solution or an ESRI one.

raw data storage

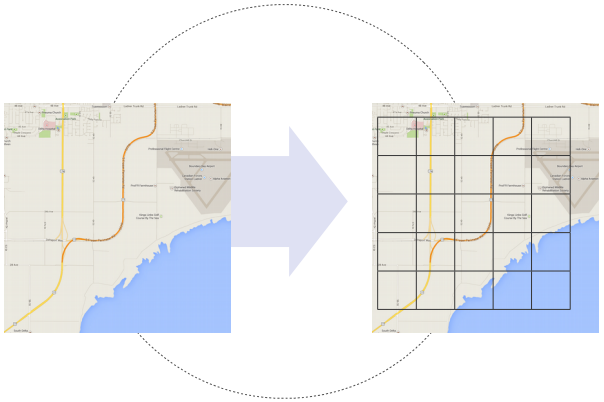
Somewhere back on a server there will be some raw data,
vectors and imagery, it might be in files or in a database for easy random access.

rendering service



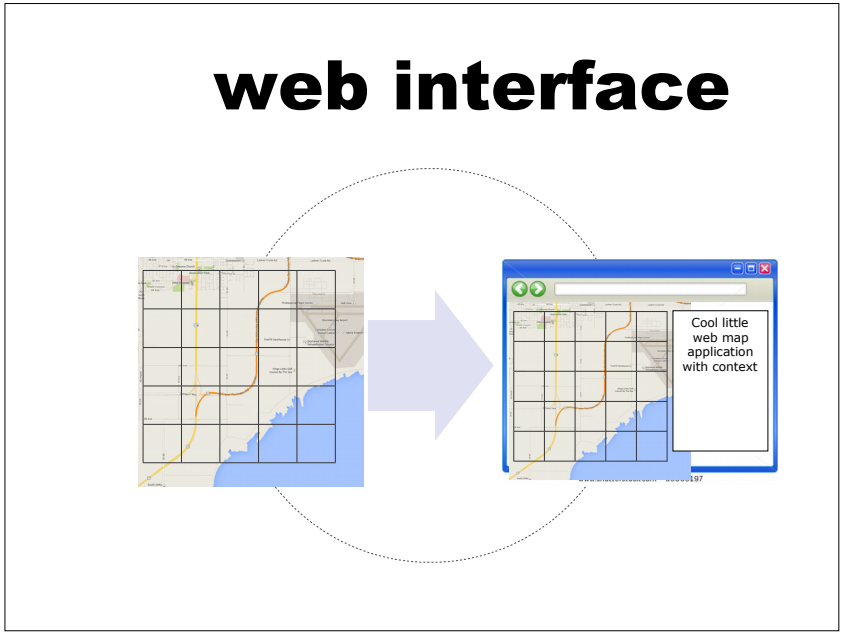
That raw data gets converted to cartographic output by a map server. So raw data goes in, and colorful images come out.

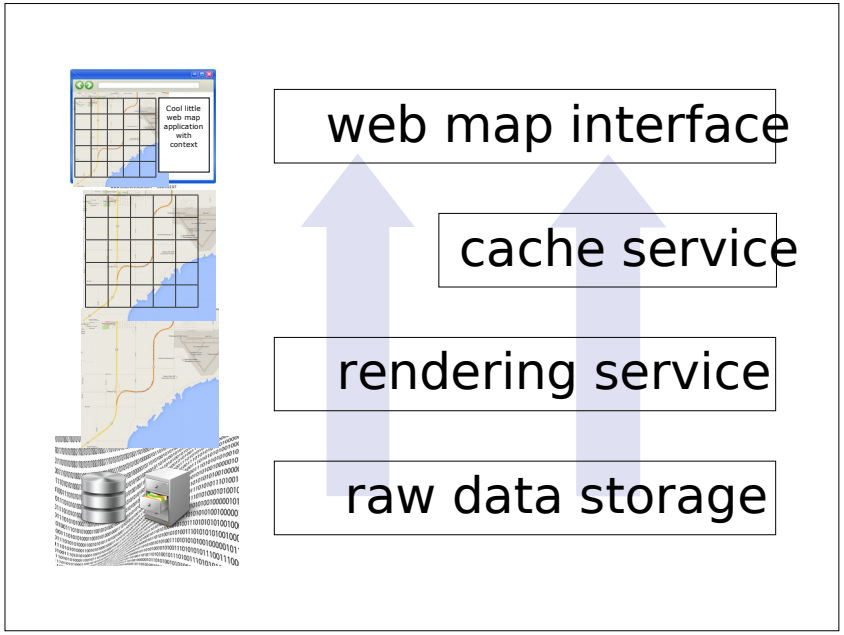
caching service



In order to speed up web maps,
the cartographic output is often cut up into "tiles" and stored in a "tile cache".

So far all the software and information I've described is server-side.
Running in your server room or in a cloud somewhere.



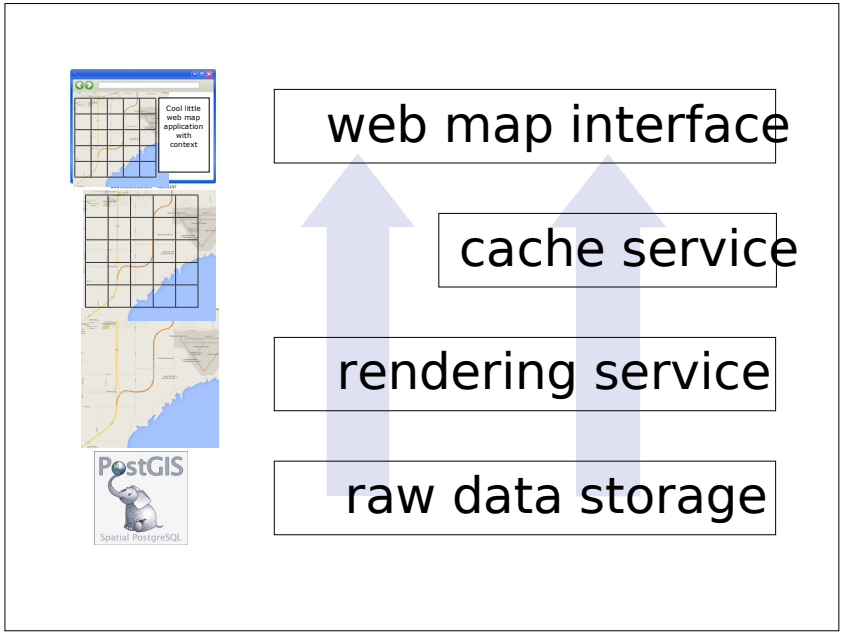


On the client side is a web page,
within which is a rectangular piece of active map.
In the Google and Bing and open source world the active map is a piece of JavaScript,
but in older systems it's sometimes Flash or Silverlight.

The map either talks directly to the map server to get live rendered data, or to the tile cache to get static maps,
and layers them up into a zoomable map view.

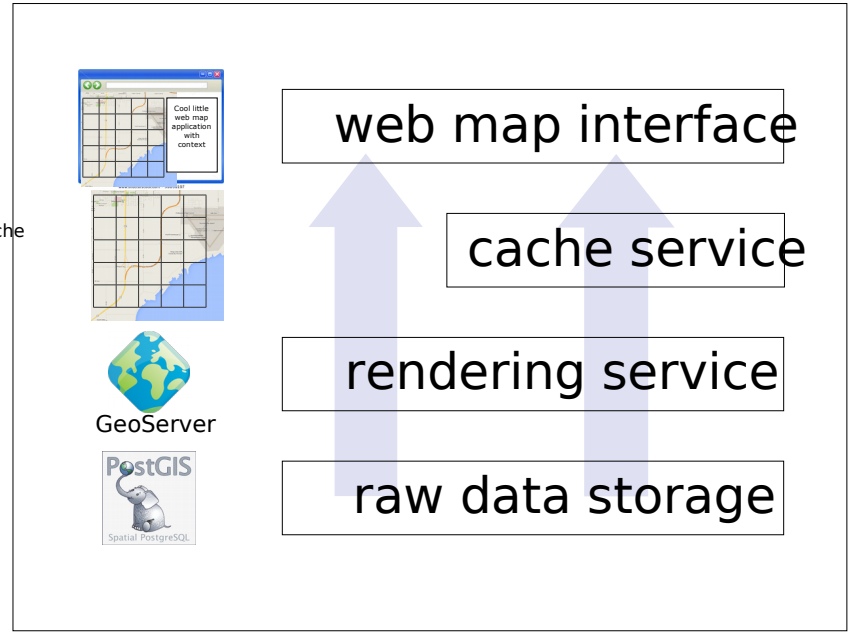
So those are the application layers:
<x> raw data storage
<x> rendering service
<x> caching service
<x> web interface

And there are open source tools that can fulfill each of these roles.
And each of these tools is included in OpenGeo Suite

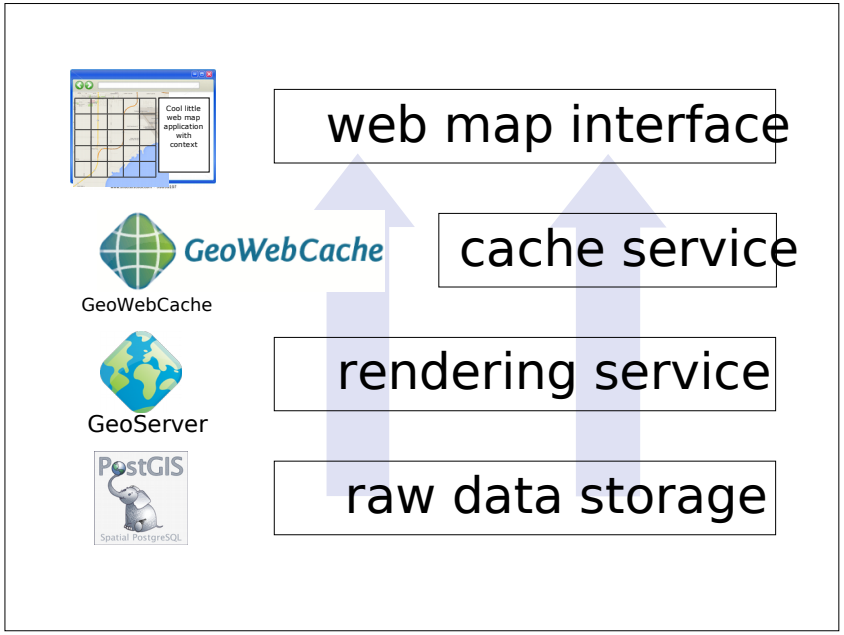


PostGIS / PostgreSQL open source spatial database is a great place for raw data storage. It widely supported by both open source applications and ESRI.

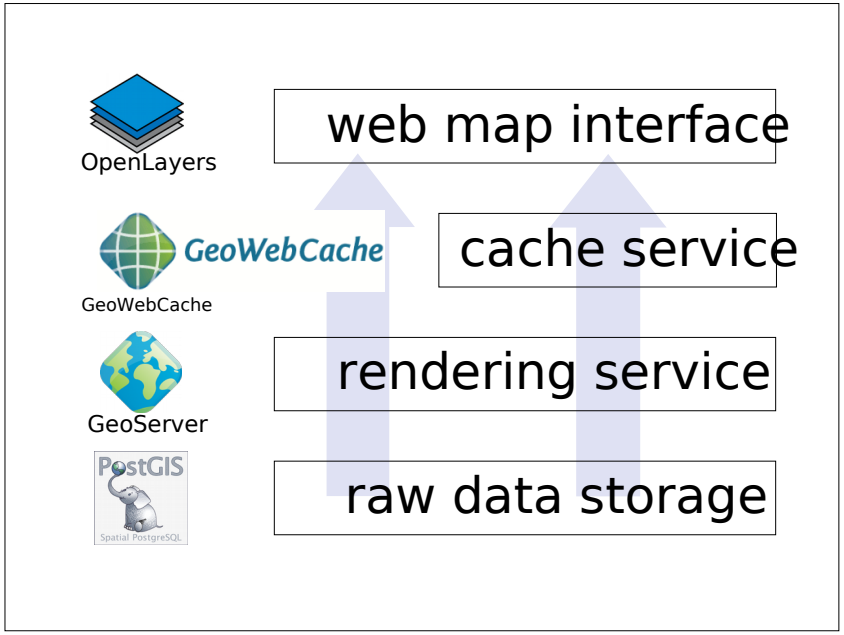
GeoWebCache



GeoServer provides a rendering service, and other web services for data access. It supports both support map and feature access, web editing operations, KML, and GeoJSON.



GeoWebCache provides a tile caching service, capable of seeding on demand, which is useful to save space and avoid time intensive seeding.

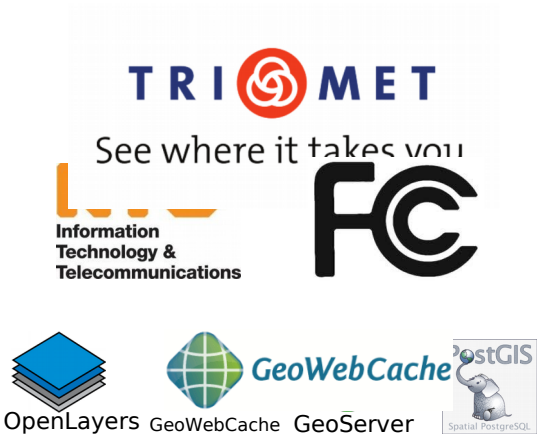


OpenLayers is the web map interface that supports a very wide range of web browsers. It supports multiple tile sources (Google, Bing, MapQuest, OpenStreetMap, GeoServer, MapServer, ArcGIS Server, etc, etc), it supports vector data and web editing, point clustering, pop-ups, all the usual bits and pieces.

is that all?

Am I leaving anything out?
Oh, yes, I very much am.

There's way more options for rendering than GeoServer,
and more caching option than GeoWebCache,
and more web interfaces than OpenLayers.
But comparing and contrasting them all would take hours.

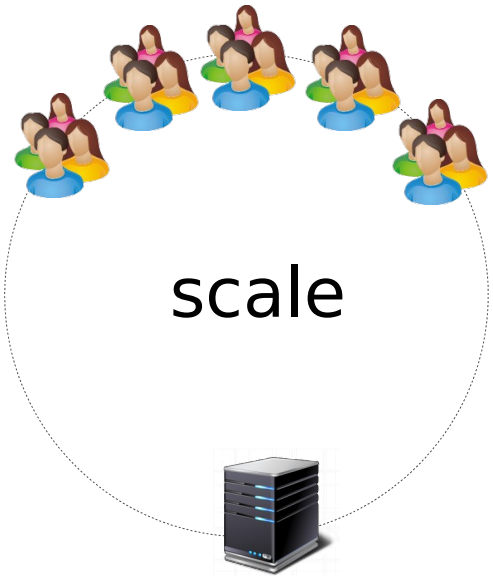


And the organizations I'm going to talk about,
TriMet, DoITT and the FCC
that built spatial web sites using open source,
all used this particular open source stack: PostGIS,
GeoServer, GeoWebCache, OpenLayers.

integrations

real-time data

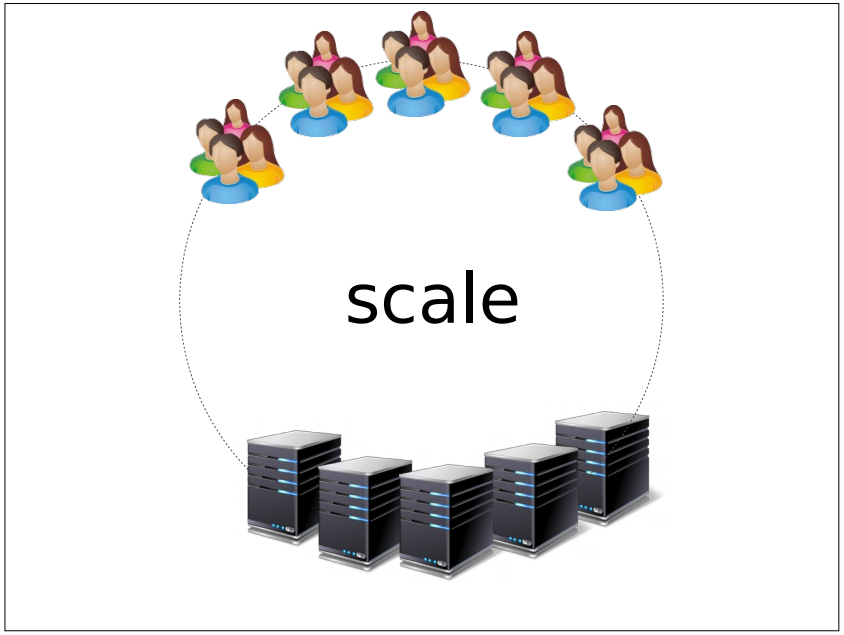
public facing



Beyond the software they used, the sites these organizations built all have some commonalities, which help explain their use of open source

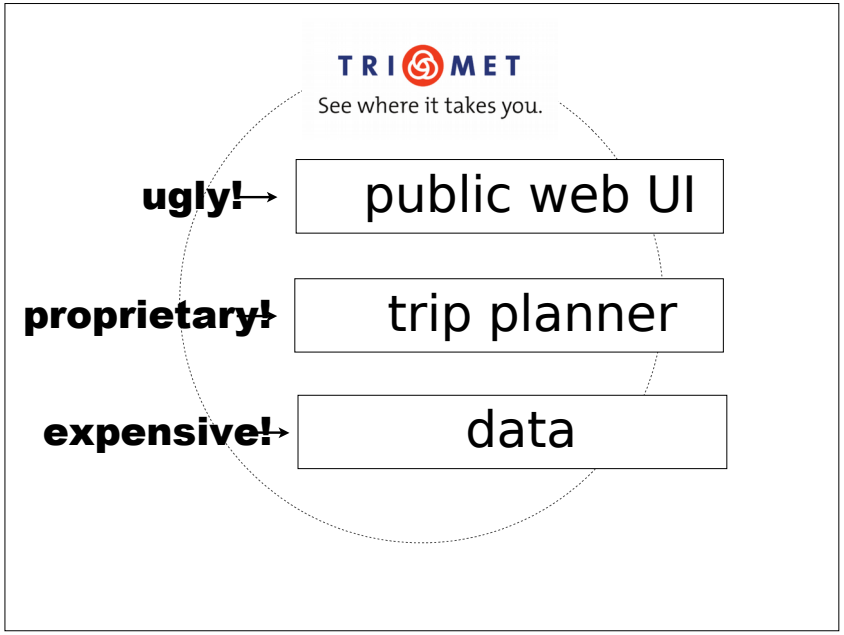
<x> all integrate multiple systems
<x> all publishing real-time data, with high public salience
<x> all publicly facing, and very high traffic because of that desirable data

Publicly facing sites with lots of traffic and constantly changing data generate issues of SCALE, which are best solved through



horizontal replication of servers.
That's cheap with open source software, very costly with proprietary software.
(Hopefully the parallel to Google and Facebook and their hundreds of thousands of servers is clear.)

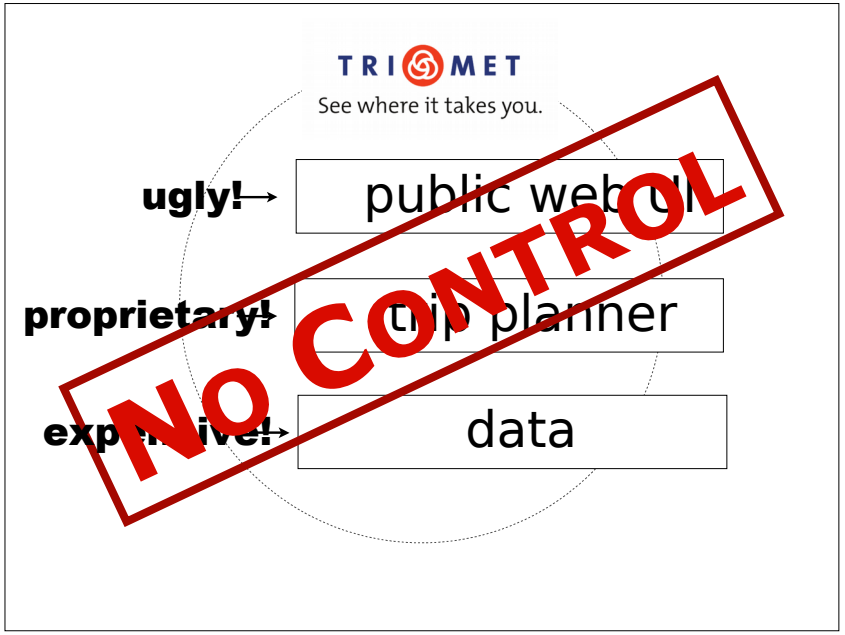
TriMet is the regional public transit authority for the Portland Metro area. They handle the busses and trains. They are the single biggest "open" story I know of, thanks to the work of their GIS Manager Bibiana Mchugh.



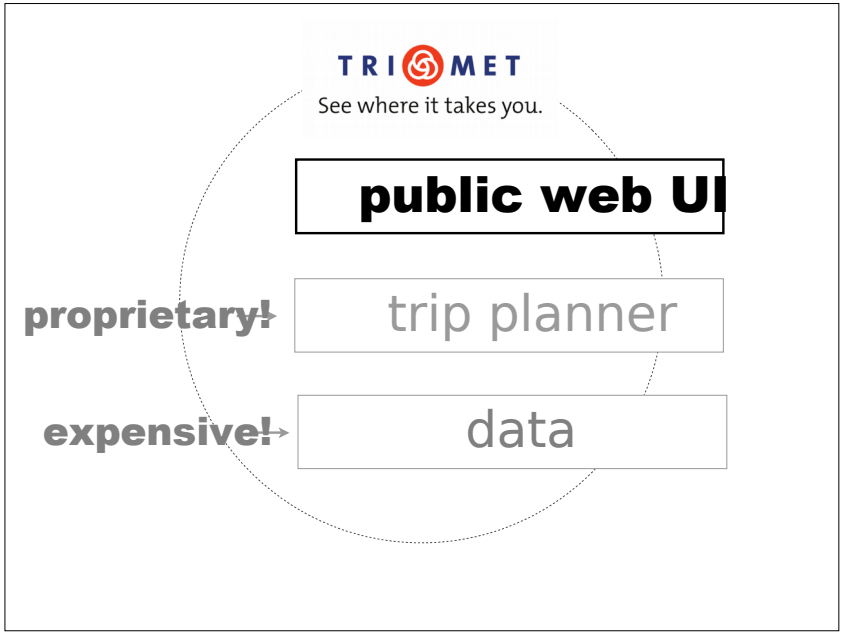
When Mchugh started, their systems included
a proprietary trip planning engine,
with a very ugly web interface,
running on expensive commercial map data.

The first order problem was that
the public trip planning interface
was awful and the vendor was slow to improve it.

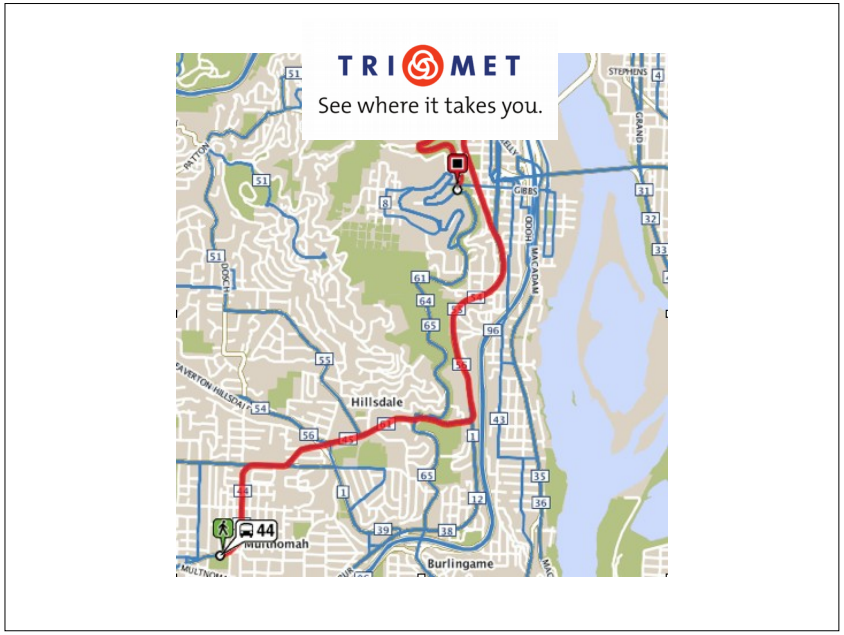
But really the problem was that the
whole system was brittle because they didn't really
control any part of it.



Someone else controlled the data,
someone else controlled the trip planner,
someone else controlled the web UI,
so changes got made when only SOMEONE ELSE
.. felt like doing it.

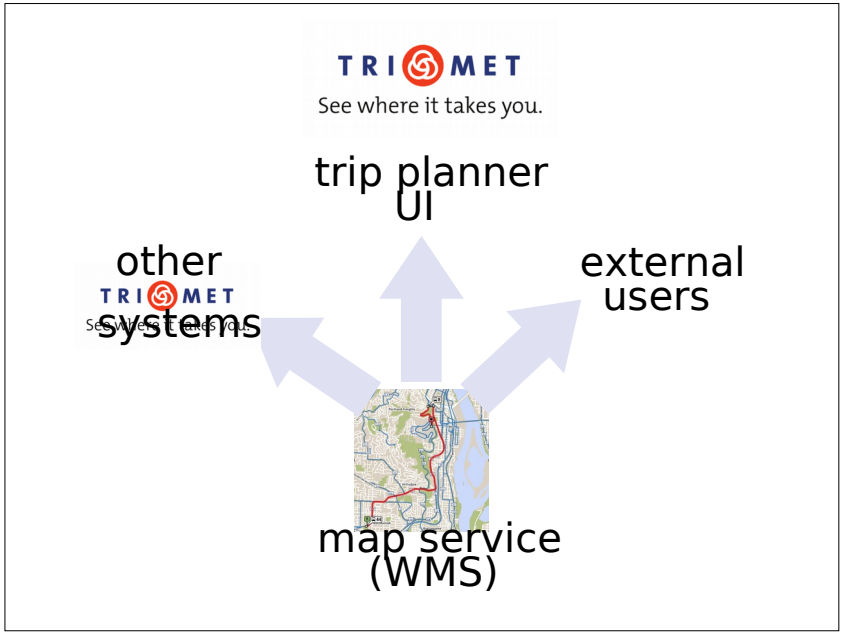


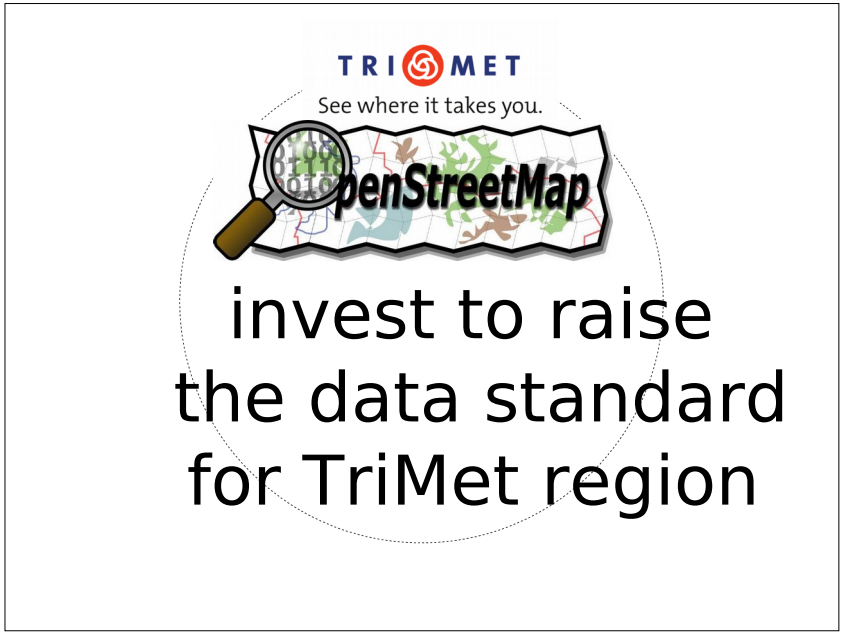
First they took control of the public face of the trip planner,
the web UI.
they built out their own web experience,
used open source database, rendering and user interface tools
and only called the just called the trip planner for routing.



They ended up with a much prettier user facing experience.
But they didn't stop there.

In architecting the trip planner, they used open standards for web mapping,
so that the map services could be mixed into other applications.

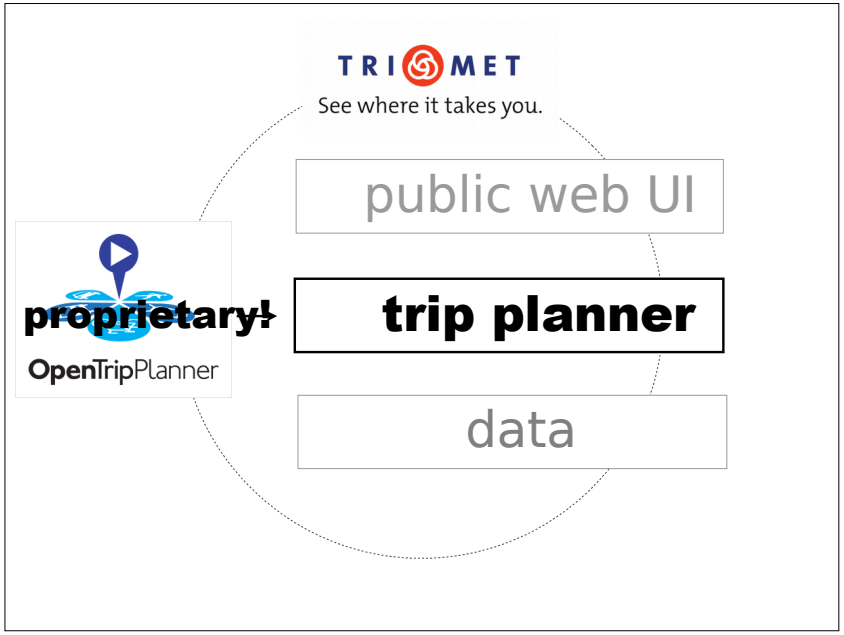




This ended up being beneficial for the rest of TriMet, because they could make use of the transit maps in their own web mapping applications.
Since the services were public, they were also used by other organizations in the Portland area as well. Many wheels were not reinvented as a result of the open standards architecture.

Then they took control of the data.
They hired interns to update the OpenStreetMap data in the Portland TriMet region
to get to a quality standard suitable for their map.

Once the map was up to snuff, they cancelled their commercial data subscription.

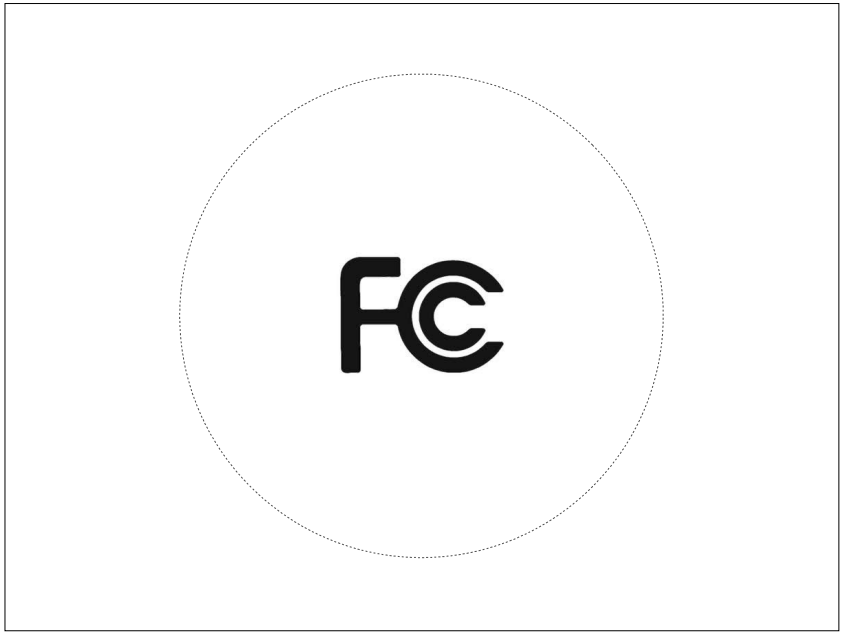


Finally, they took control of the trip planning algorithm itself.
This was the riskiest part, but it paid off.
They contracted with the Open Planning Project to polish up an open source routing engine to the standard needed to do transit trip planning for them.

<x> Then they moved all their trip planning requests to the open source engine.

TriMet executed a full "open triple play":
<x> open standards architecture, allowing service reuse across their enterprise (and indeed their region)
<x> open data use, initially investing in a common data pool, and now reaping the rewards in free crowd-sourced updates
<x> open source use, both building with existing open source tools, and investing in new ones (and again reaping benefits as other organizations also invest in the OpenTripPlanner software)

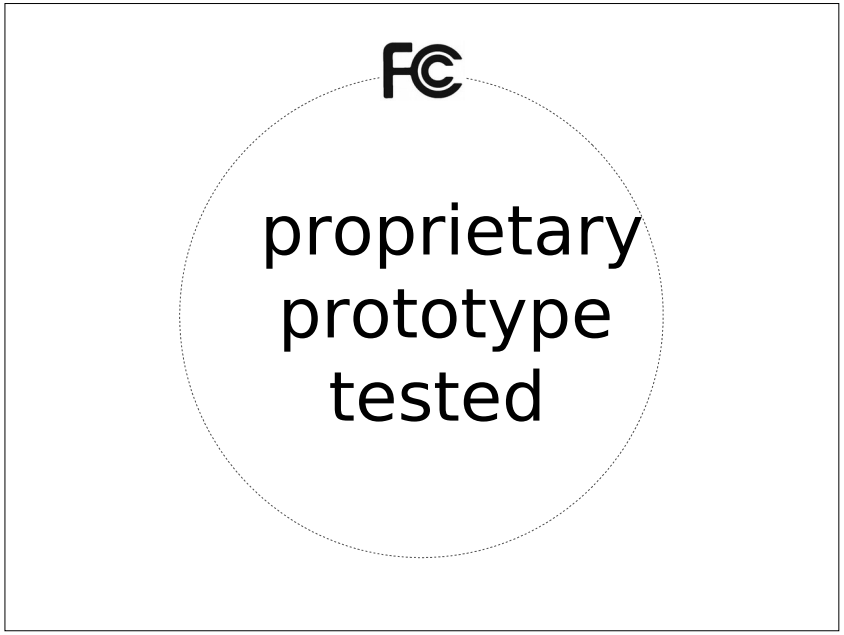
you can see it all in action at TriMet.org
the trip planner, the live maps, the data on the maps
everything is open

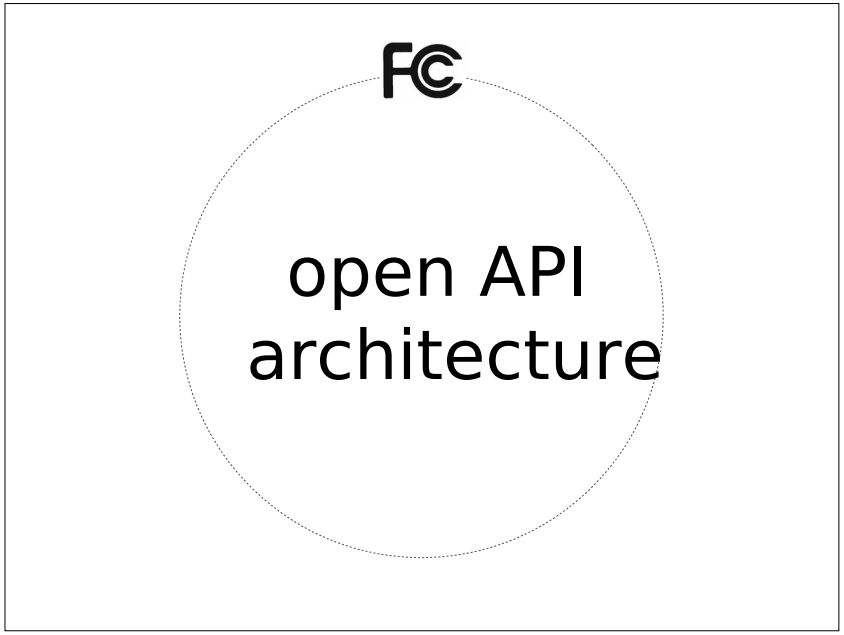


The Federal Communications Commission is a regulatory body, which means they both enforce rules and gather data.

In 2008, the Congress passed the Broadband Data Improvement Act that directed them to gather data about access to broadband internet in the USA, and then to publish it on an online map.

So the FCC had a **legal requirement** to build a web map of broadband access.
You think it's when the boss gives you tight deadline.
Congress wrote this web map and delivery data **into the law**.





Since the map was going to go live on a date set in law, they knew they only had one chance to get it right.

So they built a prototype, using the standard proprietary tools, then they load tested it.

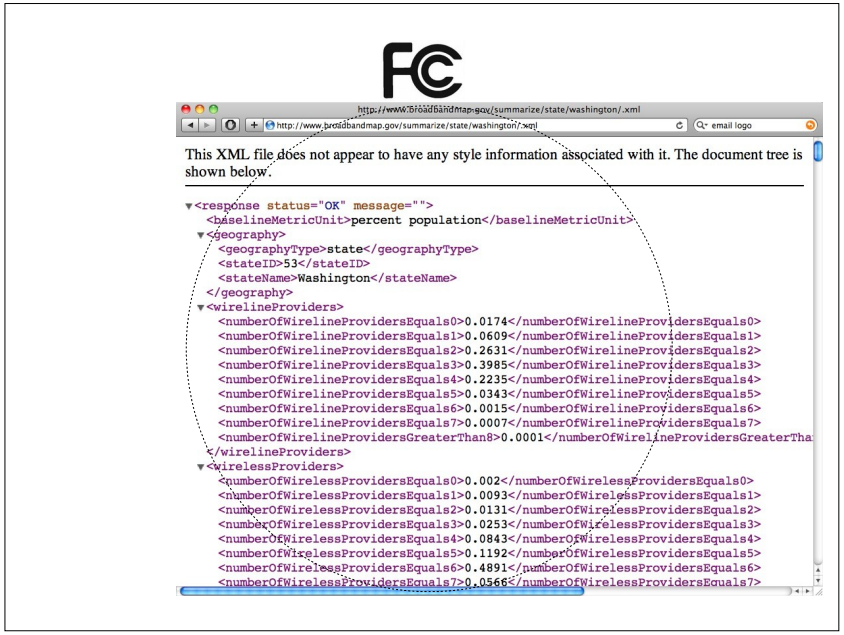
Guessing at the load they were going to get, they knew they could not scale up the proprietary technology far enough.

So they started again using open source.

And here I want to point out that the FCC went beyond open source in their solution.

They recognized that their broadband data was going to be widely of interest,
and that their particular map view was not going to be the way
everyone wanted to consume the data.

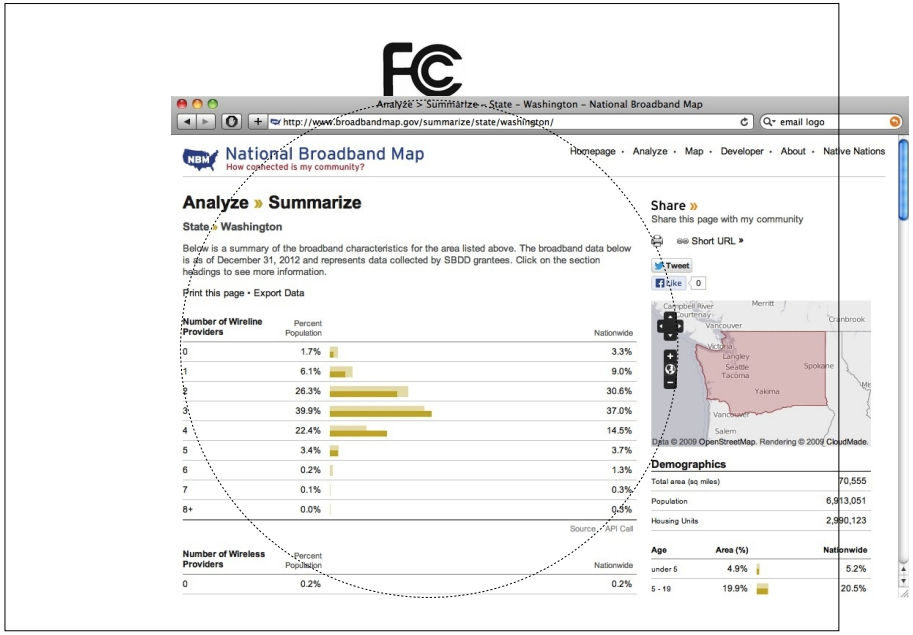
So they built their site in two halves.



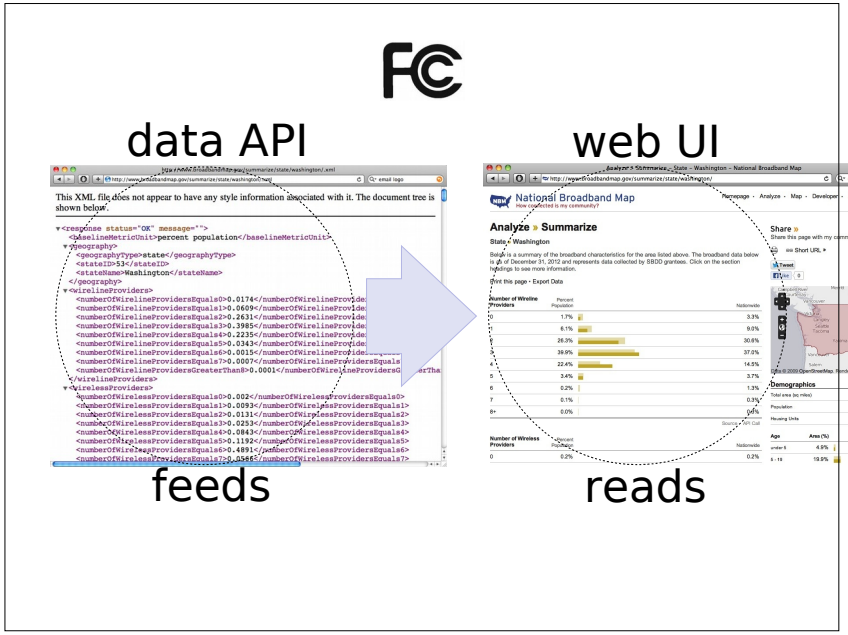
The first half, the data tier, was a data services half.

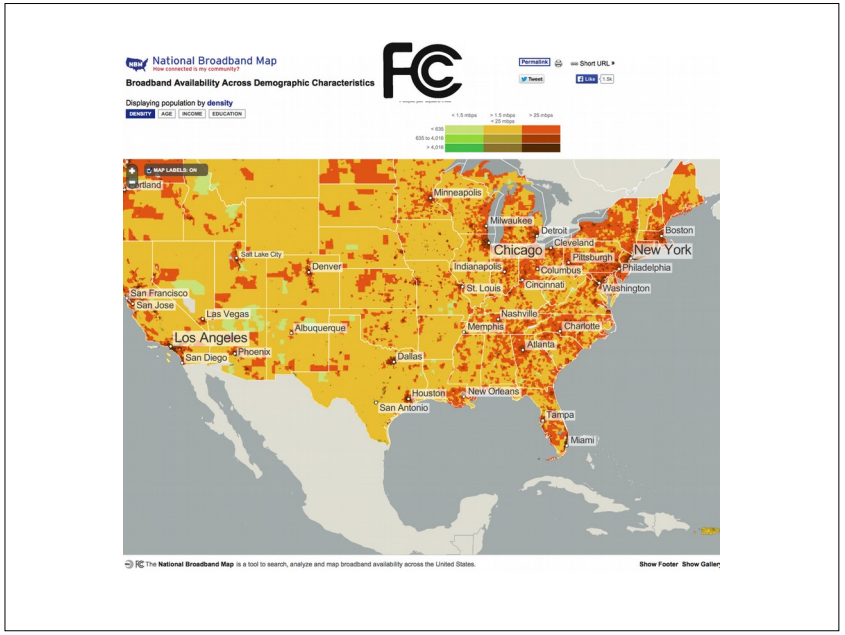
All the information in the site was published using computer readable formats,
and web services APIs.

So when they went live with the broadband map,
they would also be able to simultaneously go live with an open data site.



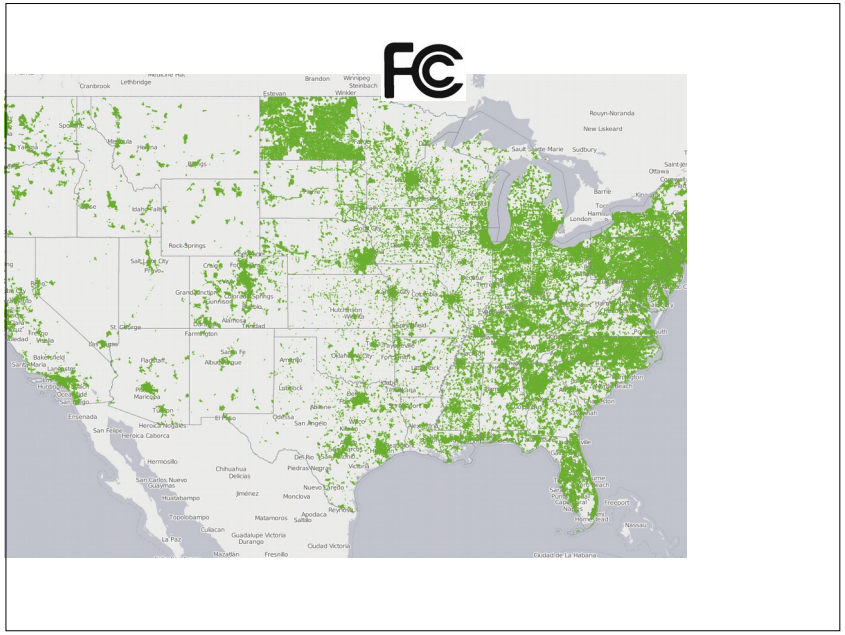
The second half, the presentation tier (which included the map),
just consumed the data services, and presented a human-viewable version of the data.
Maps and charts and tables and so on.





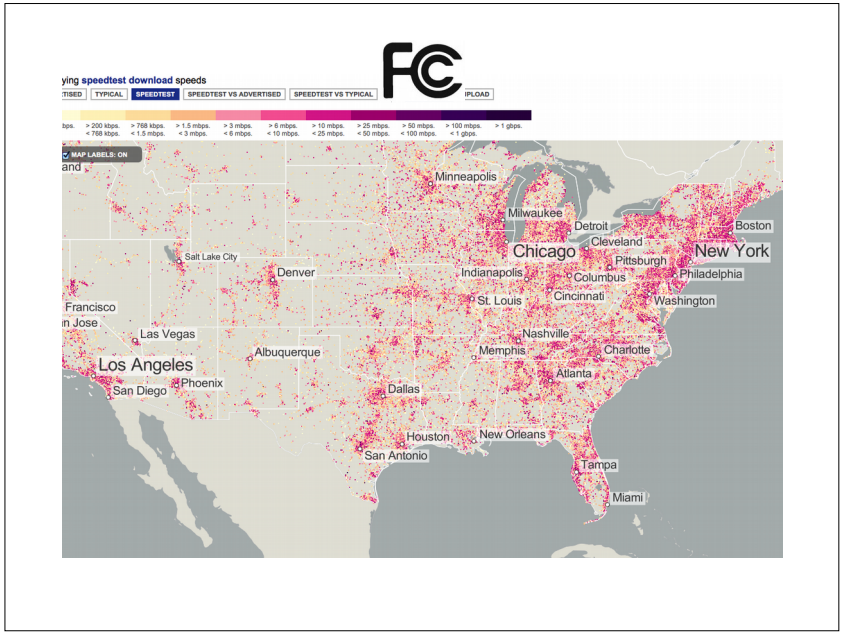
Because the presentation tier entirely depended on the data tier,
they were stress testing and acceptance testing their data tier
in the process of building their presentation tier.

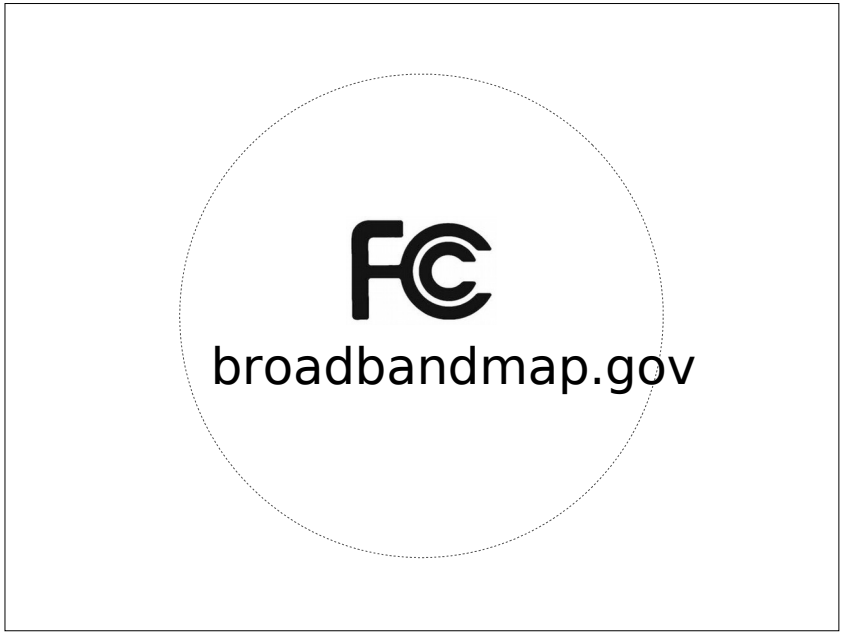
The FCC site went live in 2011, and promptly exceeded traffic projections by a factor of two, necessitating a quick doubling of server capacity over the weekend (unfortunately it was not cloud-deployed). But the extra capacity had no license implications, it was just a horizontal scaling problem.



The FCC site went live in 2011, and promptly exceeded traffic projections by a factor of two, necessitating a quick doubling of server capacity over the weekend (unfortunately it was not cloud-deployed). But the extra capacity had no license implications, it was just a horizontal scaling problem.

The FCC site went live in 2011, and promptly exceeded traffic projections by a factor of two, necessitating a quick doubling of server capacity over the weekend (unfortunately it was not cloud-deployed). But the extra capacity had no license implications, it was just a horizontal scaling problem.





The FCC site went live in 2011,
and promptly exceeded traffic projections by a factor of two,
necessitating a quick doubling of server capacity over the weekend
But the extra capacity had no license implications,
it was just a horizontal scaling problem.

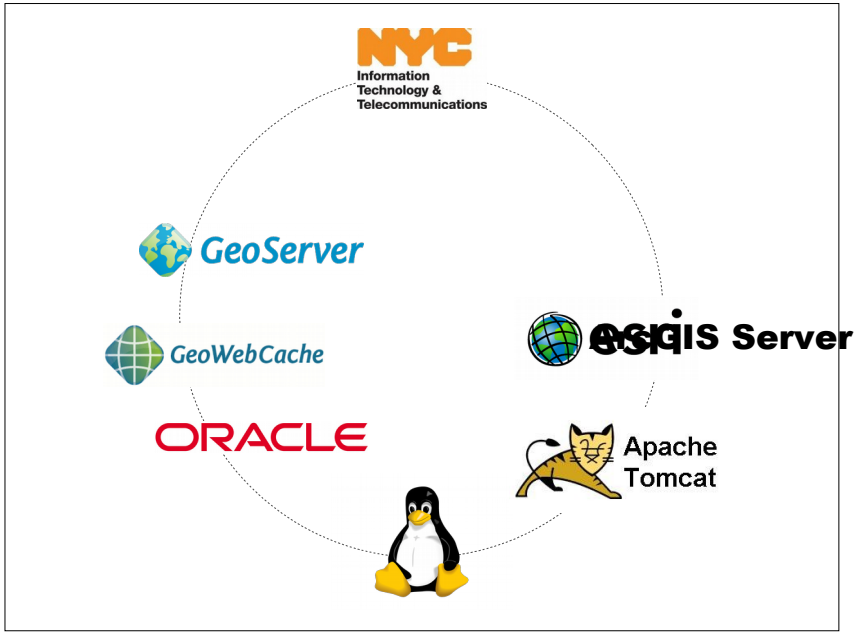
The site is still online, being updated with new network speed data all the time.
Everything at broadbandmap.gov is powered by open source.

The City of New York is a really big place.
It's bigger than most state or provincial governments.
So calling it a "municipal" use case it a stretch.
They have substantial IT resources and skills to build with.

The "geo" group inside the ITT department supports a huge number of applications, over 50, with 3-4 under development at any time and a staff of 3 developers.

They initially got exposed to open source the old fashioned way:
their IT group would only support Solaris servers.
ArcIMS ran OK on Solaris,

but it was going away, and ArcGIS Server under Solaris was very unreliable.
The immovable object met the unstoppable force.
What to do?



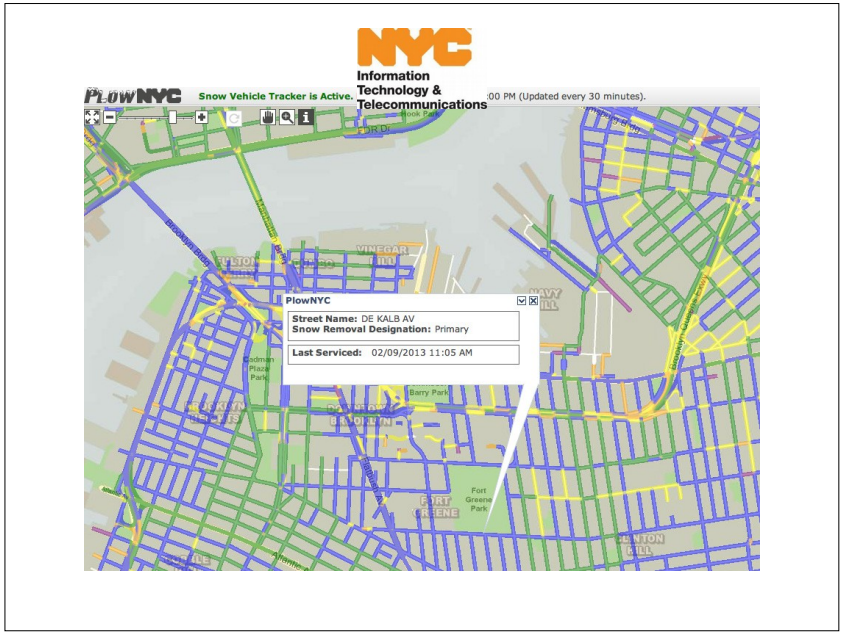
So they explored other options, and found GeoServer and GeoWebCache,
and became quite happy with them.
(I heard a similar story from an IBM consultant who was told by his boss that he could solve the client's mapping problem
any way he liked...
As long as the solution used DB2 and AIX.
So he ended up compiling open source mapping tools on AIX.)

They still run a hybrid shop, however.
For some apps they generate tiles from ArcGIS, and publish them inside open source web maps. For other apps they use an open source stack on top of Oracle.

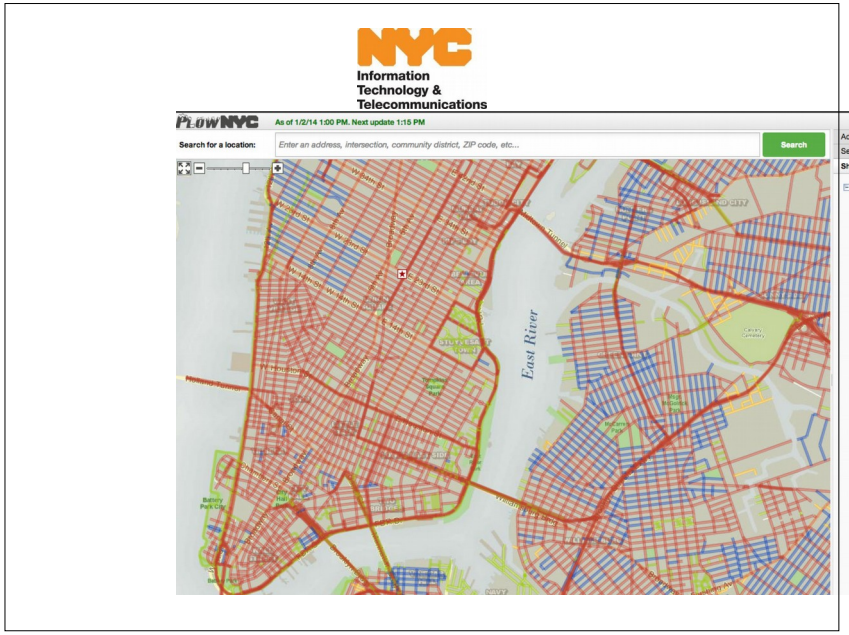
Open source isn’t a 100% either/or proposition,
it’s an and/also proposition.

On December 26, of 2010, New York was hit by a huge snow storm,
and the City got a bit of a black eye for its response.

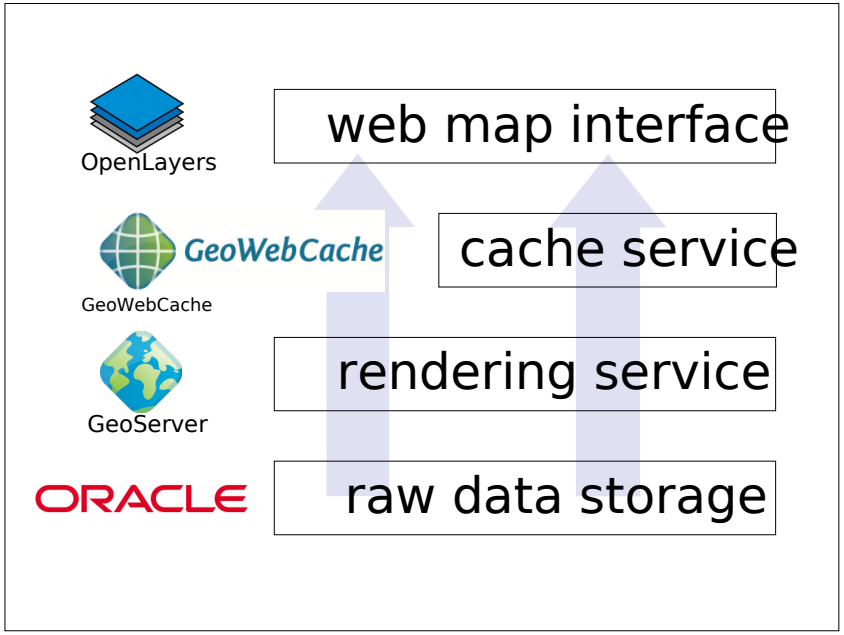
Once of the initiatives the Mayor's office came up with out of that experience
was transparency about services, in particular snow plowing.
They asked for a public map of all streets that had been plowed,
where plows were, and what streets would be plowed next.

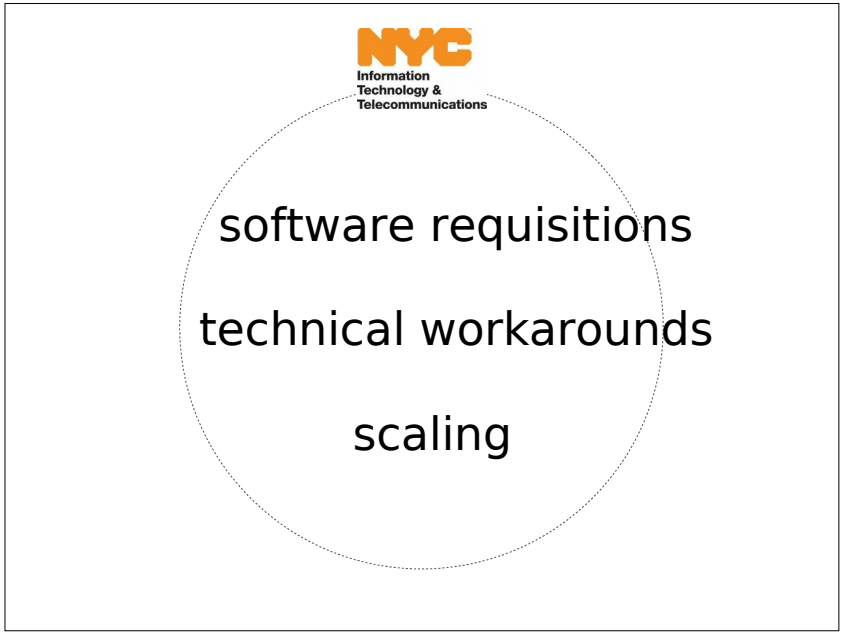


So DoITT built the PlowNYC sit,
using an open source software stack
(linux, tomcat, apache, geoserver, gwc, openlayers)
over an Oracle database.



During a snow event it services 100s of thousands of unique users.
It's been up for three years,
and is now also being internally in the command center to monitor plow activity.



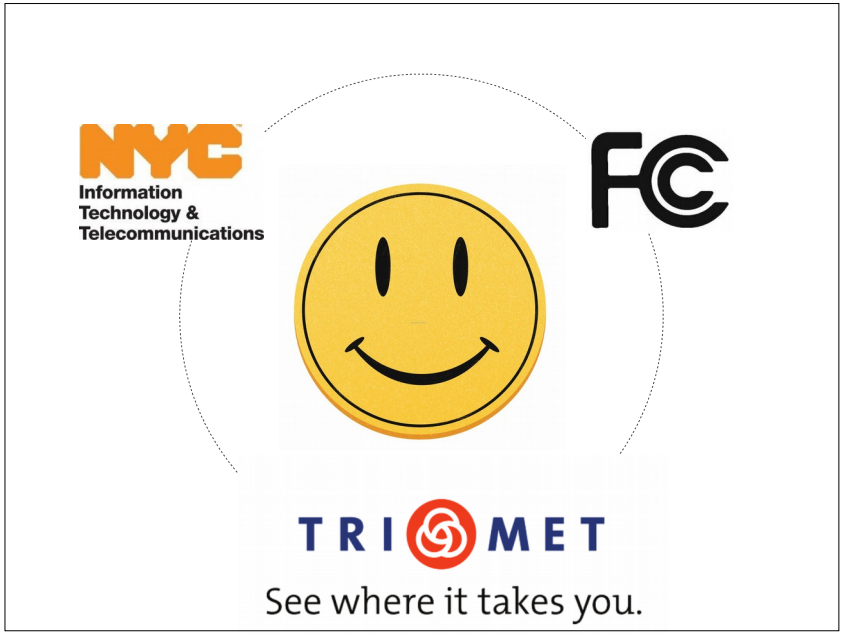


Technically, there is nothing unique about the solution: it's a database, a render farm, a caching tier and a web UI.
They could build it on proprietary software too.
In fact the City of Chicago has built a similar plow status application using proprietary.

But the procedural barriers to success would have been much higher:
software requisitions and approvals would have taken more time;
technical issues that were solved very quickly with open source would have had to be worked around or fixes waited for;
scaling would have involved license restrictions and more requisitions.

New York City is still a hybrid shop,
but more and more they are building on pure open source,
for faster development velocity.

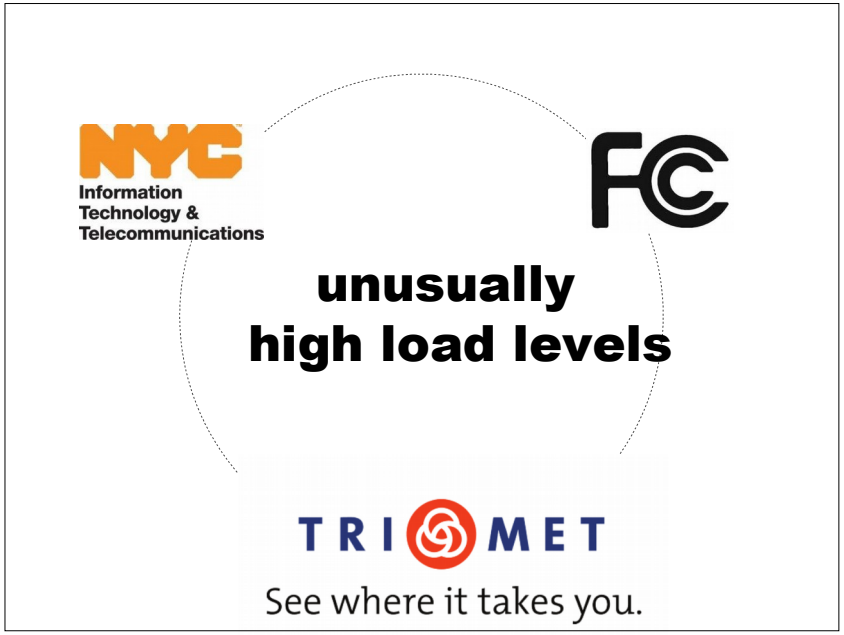
You can see the snow status app online now,
and see it under load when it snows back East.



So,
Three levels of government (municipal, regional, federal),
three high traffic, public facing web services,
three happy organizations.
Open source has been good to them.

But I’m not saying
open source is a panacea.
It's not for every organization.

All three of those organizations had very capable developers on staff or contract.



All three had abnormally high requirements for traffic on their sites.
(And all three of them actually exceeded their load predictions when they went live.)

At this point,
all three of them have been using open source for a while.

I think all three would say they were nervous at the start,
and they experienced the steep learning curve,
but are happy with the journey they've gone on.

If you're thinking about exploring open source,
<x> start small, with a single application, and give it a shot.
<x> Give your developers the space to learn and experiment and try different approaches.
<x> Remember than open source doesn't have a marketing department,
self-motivation is a key requirement.

You also have a unique opportunity this fall, to learn,
one that won't be repeated for a long time,
because FOSS4G, the international open source
geospatial conference,
is just down south in Portland.

(In case you're thinking "oh, we'll catch it next time",
the previous four locations were Sydney, Barcelona,
Denver, and Nottingham.
Next year it will be on the other side of an ocean.
Take advantage of the opportunity this year.)

Thanks very much, please ask me questions, now, or
afterwards,...

Boundless	boundlessgeo.com
TRIMET See where it takes you.	trimet.org
FCC	broadbandmap.gov
NYC Information Technology & Telecommunications	maps.nyc.gov/snow