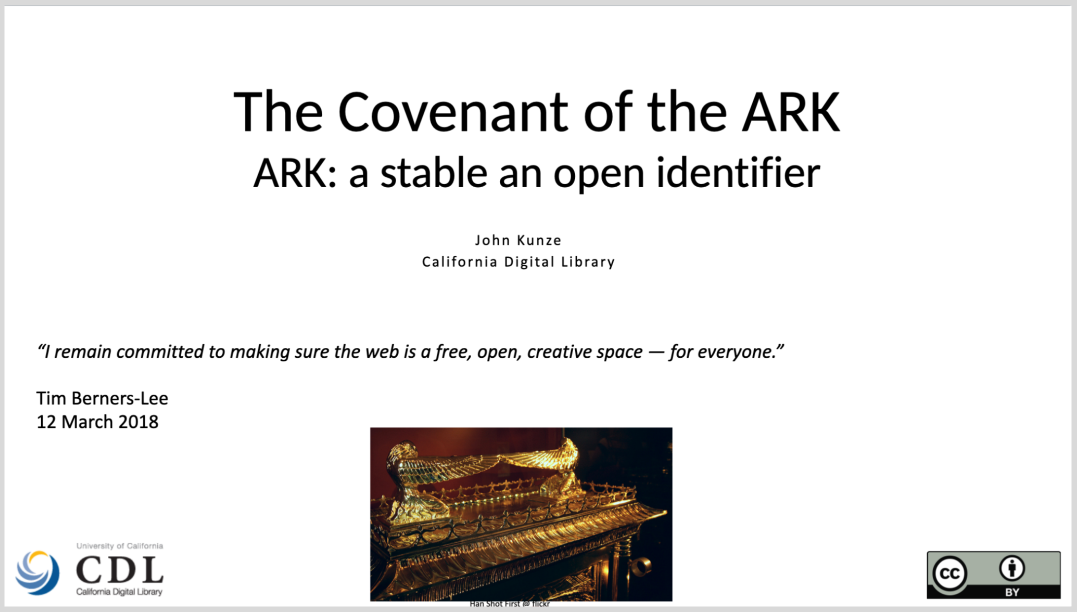
The Covenant of the ARK: a stable and open identifier

A brief history of open information access tools on the Internet and factors leading to the need to create the Archival Resource Key (ARK) persistent identifier (PID) scheme.

This is the English transcription of John Kunze's keynote address to the March 21, 2018 ARK Summit at the National Library of France (Bibliothèque nationale de France) in Paris. Link to video.

Photo credit: Beatrice Lucchese



The Covenant of the ARK

ARK: a stable an open identifier

John Kunze
California Digital Library

"I remain committed to making sure the web is a free, open, creative space — for everyone."

Tim Berners-Lee
12 March 2018

University of California
CDL
California Digital Library
cc BY

Han Shot First @ flickr

Thank you for that introduction, Sebastien. Thank you also to the BnF staff for organizing this fantastic summit, and for translating my speech. You don't want to hear what translate.google.com produced.

I am delighted to see so much interest in ARK identifiers. I would like to tell you about their history, their future, and the principles that shape them.

Daniel Parks and Pete Birkinshaw @ flickr

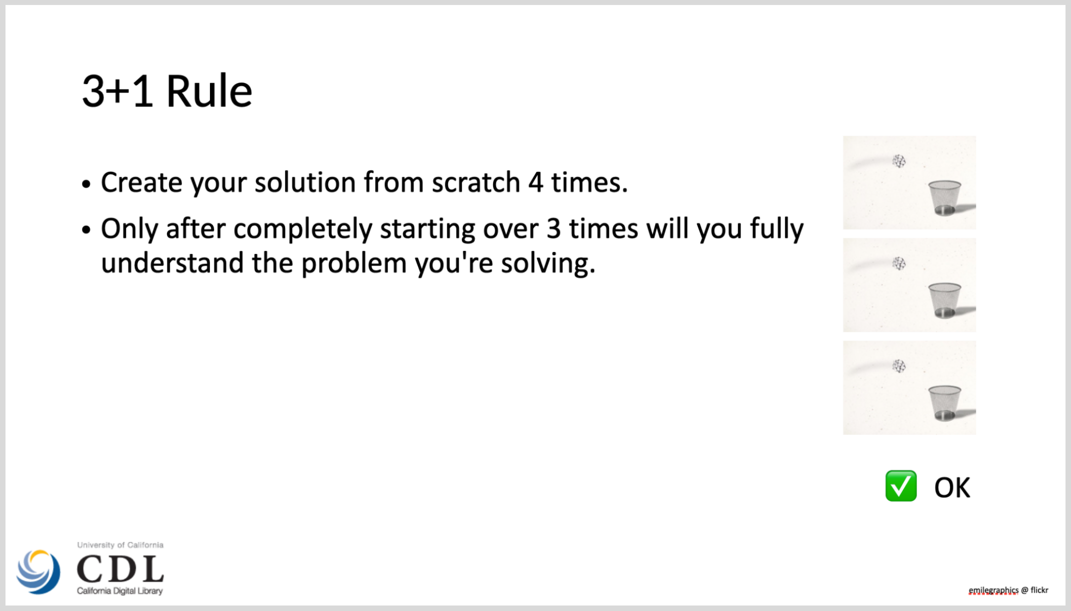
This photo is of Berkeley, California, where the ARK story begins. There is a well-known university in the foreground, with the city of San Francisco in the background.

Here is the computer science and mathematics building where I first met my first boss. He was a brilliant programmer, hired by the university after he hacked into the student database. It was so long ago that the attack was done with punched cards. His intent was not to cause damage but to correct an injustice. He risked his academic career just to change one of his class grades from A- to A. Fortunately for him, instead of punishing him the university offered him a job!

Fortunately for me, he took the risk of hiring me when I was a first year student with little computer experience. Berkeley was just starting to extend and enhance a new operating system called Unix.

Berkeley's system became "FreeBSD", the basis for Apple's Mac OS X. I'm very grateful that I got a chance to work on what was essentially "open source software", but this was 10 years before that term was invented. I also got to work with an amazing group of people, to meet my future wife, and to create software that runs today on every Mac computer in this room.

Berkeley gave me a strong regard for the principles of openness, sharing, and pragmatism. We gave away the fruits of our labors instead of hoarding them. Design principles favored the simplest solution that did the task. Each software tool should do one task well, but that task should be generalized to work in many different situations. Each tool should be able to cooperate with other tools and be combined with them to do more complex tasks.



3+1 Rule

- Create your solution from scratch 4 times.
- Only after completely starting over 3 times will you fully understand the problem you're solving.

✅ OK

University of California
CDL
California Digital Library

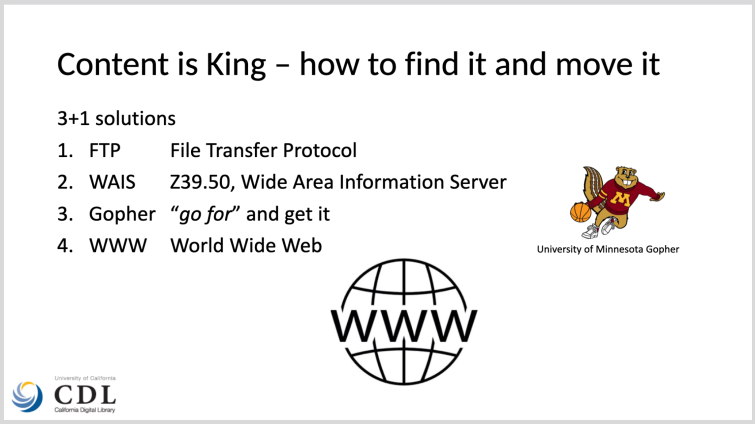
emilegraphics @ flickr

The birth of ARK identifiers relates to those values, as well as to an important lesson that my first boss taught me. I call it the 3+1 Rule. The rule says: write your solution “from scratch” 4 times; it is only after completely starting over 3 times that you will fully understand the problem you’re trying to solve. It took me years to really understand it, but I find it proven again and again, especially when it comes to internet identifiers.

When my class graduated, many people went to Silicon Valley to get high-paying jobs in the private sector. But I stayed in Berkeley because I enjoyed my public sector job at the university computer center. Software and information in the private sector at that time were not at all open. Moreover, Berkeley was a strong leader in creating advanced, practical network software.

I was able to propose and build a new networked information system for the Berkeley campus. The web did not exist yet, so this was a new concept for Berkeley.

My computer science background made me believe that the exciting work was in coding, but I learned that no matter how good my software was, it was never better than the content it served. Essentially, “Content is King”. This made me want to work more closely with people who managed open access, high-quality content, namely, the Library.

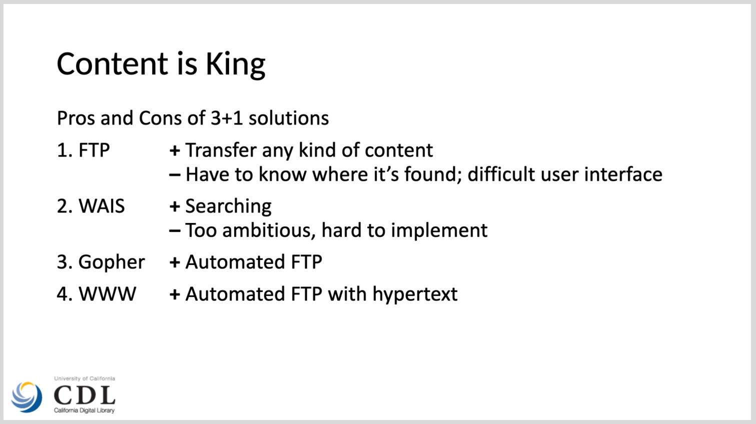


Content is King – how to find it and move it

3+1 solutions

1. FTP File Transfer Protocol
2. WAIS Z39.50, Wide Area Information Server
3. Gopher *"go for"* and get it
4. WWW World Wide Web

University of Minnesota Gopher
WWW



Content is King

Pros and Cons of 3+1 solutions

1. FTP + Transfer any kind of content
   – Have to know where it's found; difficult user interface
2. WAIS + Searching
   – Too ambitious, hard to implement
3. Gopher + Automated FTP
4. WWW + Automated FTP with hypertext

With a focus now on content, the main problem became finding and moving content from the computer where it was stored to the computer where the user would use it.

As you might suspect, there were 4 major solutions to this problem.

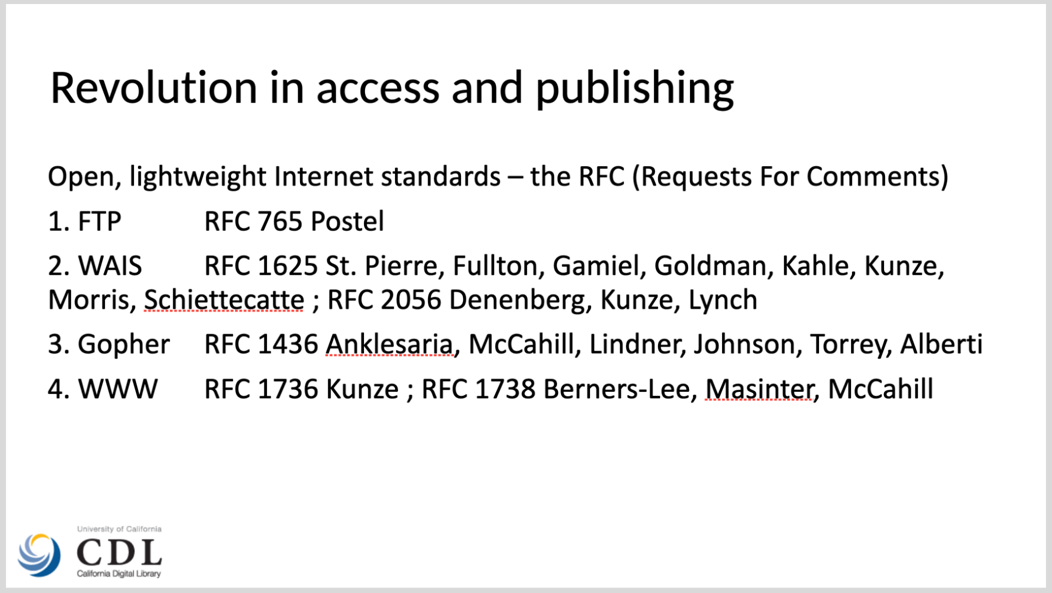
The first solution was FTP, which was very cool. But you needed to know where the content that you wanted was located, then enter a complex command in a terminal window. Only geeks loved this solution.

The second solution I encountered was the library-world solution. It assumed content transfer was easy, but that you first had to learn where the content you were interested in "lived". This was the "search" sub-problem. New to the Library world, I was impressed that this approach was the only one to have a standard, and I committed to it. The library standard had a difficult name, which was Z39.50, but a more well-known implementation was called WAIS. The standard was also complex and incomplete, and I learned through experience that standards development is much harder and slower than software development. In the end, finishing the standard and implementing "search" for the general internet (beyond the library world) proved to be too hard to finish.

The third solution was called "Gopher", a word that has several meanings. Gopher is the name of a rodent that is always busy moving food and earth through the tunnels that it digs. In English, it also means a person whose job is to "go for" and fetch things for other people. Finally, it is also the name of the sports mascot for the University of Minnesota, where the solution was created. This solution was very popular, because it gave each user a menu of choices, which took the pain out of FTP. Gopher automated FTP so that instead of entering complex commands, you just selected menu items. Menus were hierarchical and interconnected very much like the web today. There was no search capability, but many people thought Gopher was the final solution, but that changed when the 4th solution arrived.

I met Tim Berners-Lee for the first time when he came to Berkeley to speak about his new system, which he was calling the "world wide web". This 4th solution also lacked a search capability, but it compensated by adding hypertext. Instead of relying only on menus, like the

Gopher solution, the new web solution let you make better decisions about selecting content. Soon everyone realized that this was the best solution. Like any solution, it also changed our understanding of the problem. The problem we should have been solving was to automate transfer and use hypertext to aid content selection. A proper “search” system was a separate problem.



Revolution in access and publishing

Open, lightweight Internet standards – the RFC (Requests For Comments)

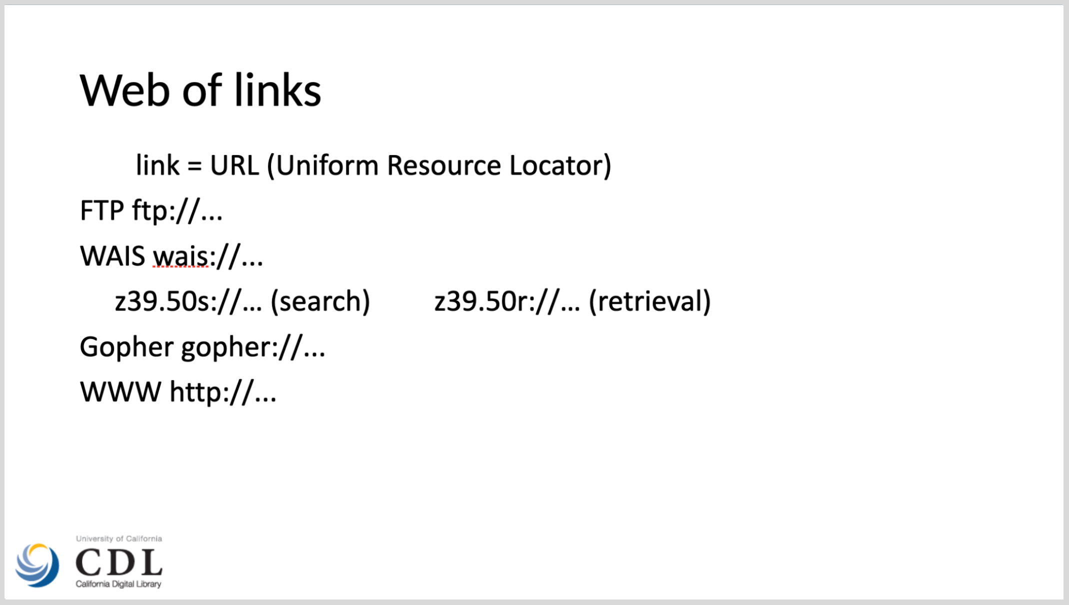
1. FTP RFC 765 Postel
2. WAIS RFC 1625 St. Pierre, Fullton, Gamiel, Goldman, Kahle, Kunze, Morris, Schiettecatte ; RFC 2056 Denenberg, Kunze, Lynch
3. Gopher RFC 1436 Anklesaria, McCahill, Lindner, Johnson, Torrey, Alberti
4. WWW RFC 1736 Kunze ; RFC 1738 Berners-Lee, Masinter, McCahill

University of California
CDL
California Digital Library

It was an exciting time. There was a revolution in information access and online publishing. Before, online information was more rare and hard to find. There was no free, instant access to almost any kind of content that we take for granted today.

But suddenly it all changed. It became easy for anyone to publish on the web, which began to grow into a vast body of information. The web was based on open information, powered by open source software, and specified by open standards. It grew out of a culture of sharing. That culture let people build more open content and software so that everyone could benefit.

At this point, I became involved with Internet standards, or “RFCs”, which were much more lightweight and open than Library standards. Anyone could propose a new Internet standard. You only had to convince the Internet to accept it. Each of the solutions had its own standards.



Web of links

link = URL (Uniform Resource Locator)

FTP ftp://...

WAIS wais://...

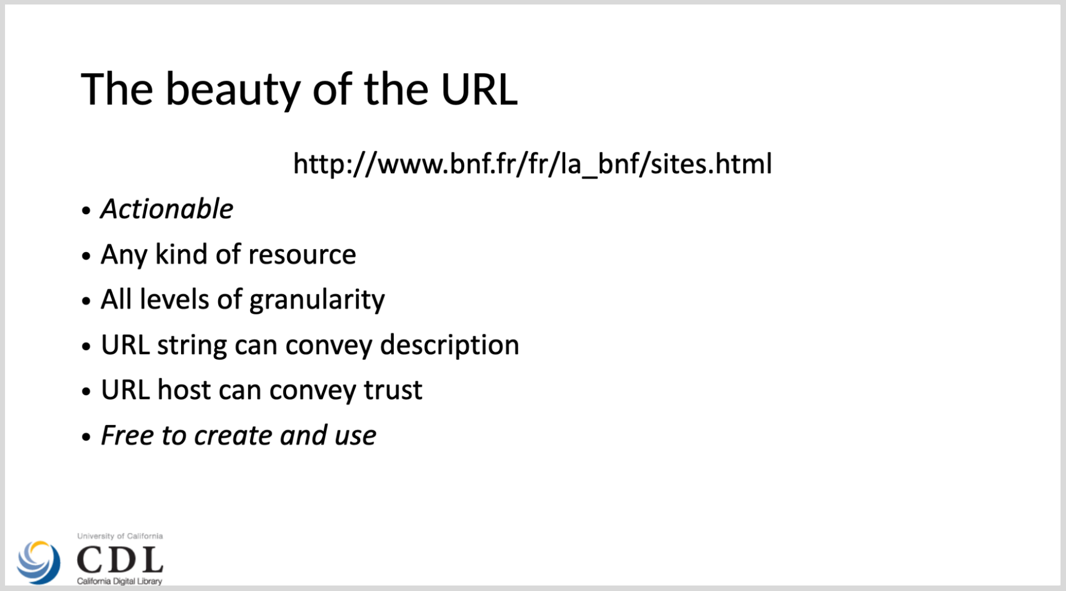
z39.50s://... (search)   z39.50r://... (retrieval)

Gopher gopher://...

WWW http://...

University of California
CDL
California Digital Library

At its core, the web is a “web of links”. Each solution has its own kind of link, such as “http” for the web, but only the web solution introduced one unified way to express all links. This was a generalized format, called the URL, or Uniform Resource Locator. A proper web browser understands all of these links. Eventually the native “http” link became dominant, and it is now ordinary and familiar.



The beauty of the URL

http://www.bnf.fr/fr/la_bnf/sites.html

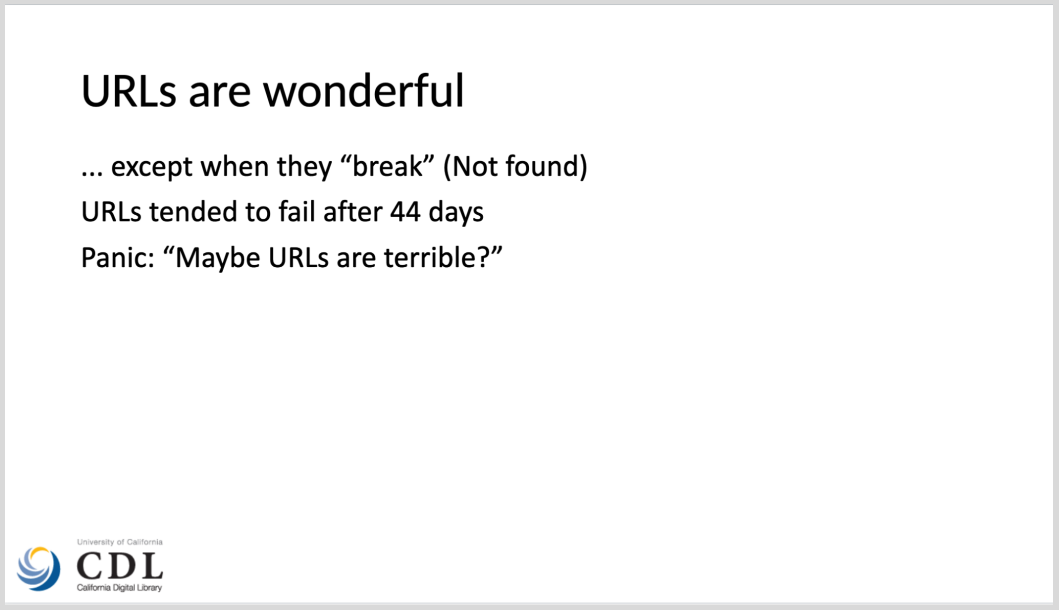
- *Actionable*
- Any kind of resource
- All levels of granularity
- URL string can convey description
- URL host can convey trust
- *Free to create and use*

University of California
CDL
California Digital Library

The URL is a beautiful, compact, 3-part package. It contains the kind of link, the address of a web server, and the address of content on that server. This makes the URL “actionable”, which distinguished it in the beginning, and even now, from most other identifiers. Modern software understands how to use a URL to transfer and display content automatically.

A URL can link to any kind of content and at any level of granularity. It can contain descriptive words to help you remember, enter, and verify them.

When the user reads the server address or hostname, that helps to understand whether the URL can be trusted. Finally, URLs are unusual in that you can create them without asking permission or paying a fee. These are wonderful properties for any identifier to have, and they taught important lessons in the design of the ARK as a persistent identifier. It is surprising to me that subsequent identifier systems have been proposed that downplay the importance of these features.



URLs are wonderful

... except when they "break" (Not found)

URLs tended to fail after 44 days

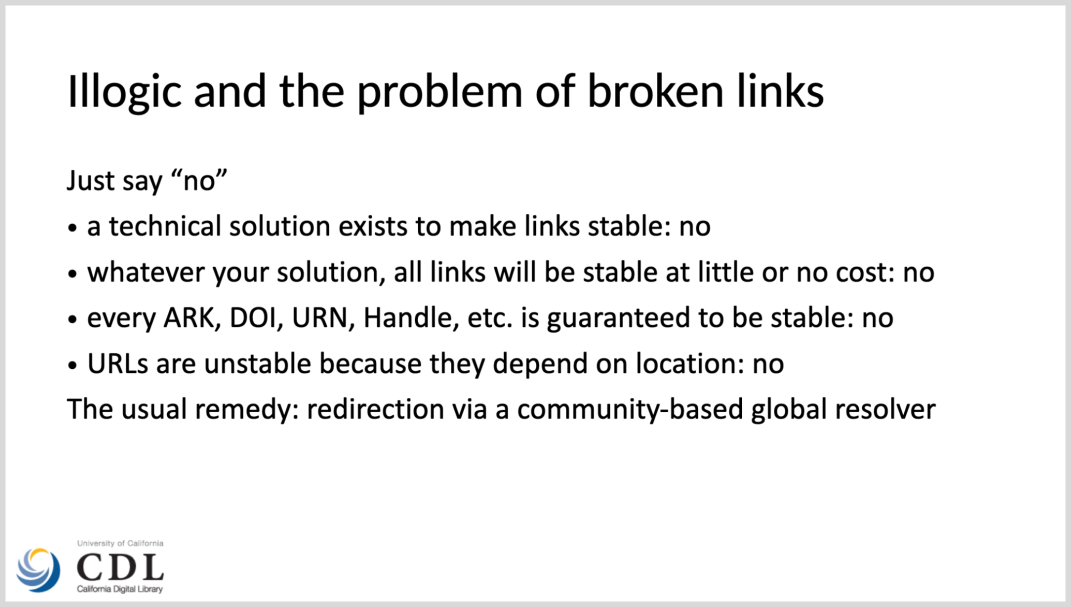
Panic: "Maybe URLs are terrible?"

University of California
CDL
California Digital Library

But there was a new problem. The links were not persistent enough. We love instant access to content, but we hate when the content disappears or changes. The web was still very young when people began to complain about how quickly URLs that used to work now led to "Not Found" errors.

According to one study, the average lifetime of a URL is 44 days. A URL often broke, for example, when content was moved or removed, or when a web server was renamed or removed from service. This was irritating for many, but a catastrophe for those preserving cultural and scholarly heritage, and for those building reliable interconnected web services.

There were other troubles. Non-technical troubles that Internet standards and technical experts were not good at addressing. Frustration and impatience was high in the standards groups. The URL RFC was taking much longer than expected. We all had jobs to return to. The URL itself was under attack and some worried that it was an incomplete solution.



Illogic and the problem of broken links

Just say “no”

- a technical solution exists to make links stable: no
- whatever your solution, all links will be stable at little or no cost: no
- every ARK, DOI, URN, Handle, etc. is guaranteed to be stable: no
- URLs are unstable because they depend on location: no

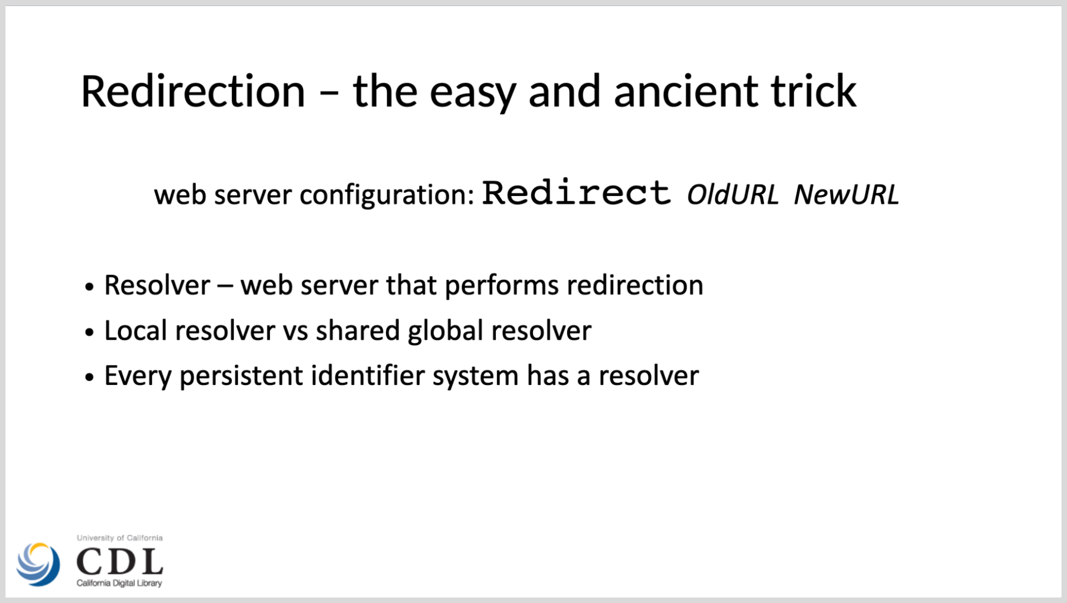
The usual remedy: redirection via a community-based global resolver

In a big hurry to solve the problem of broken links, a number of illogical ideas appeared and took root. Some of those ideas still pollute discussion today. There was the idea that a technical solution exists for broken links, when we now know it requires a socio-organizational solution.

Most people nowadays have also come to accept that persistence is purely about service commitment and deliberate management. There was also the idea that keeping URLs stable should be as easy and free as creating URLs. But large amounts of content are not meant to persist, especially when maintaining that content costs money.

People often think that persistent identifiers are guarantees. But ARK, DOI, URN, and Handle identifiers are never a guarantee. There are no guarantees in any software systems, but that doesn’t stop people from repeating nonsense. There are thousands of broken links in these categories.

Then there is the idea that URLs are not stable because they depend on location. Unfortunately, the URL acronym expands to contain the word “locator”, which strengthens the false equation with “location”. This flawed logic is based on URLs that broke because webmasters moved or removed files, but the real cause of breakage was that their organizations lacked commitment, skills, or awareness to repair those URLs by “forwarding” them.



Redirection – the easy and ancient trick

web server configuration: `Redirect` *OldURL NewURL*

- Resolver – web server that performs redirection
- Local resolver vs shared global resolver
- Every persistent identifier system has a resolver

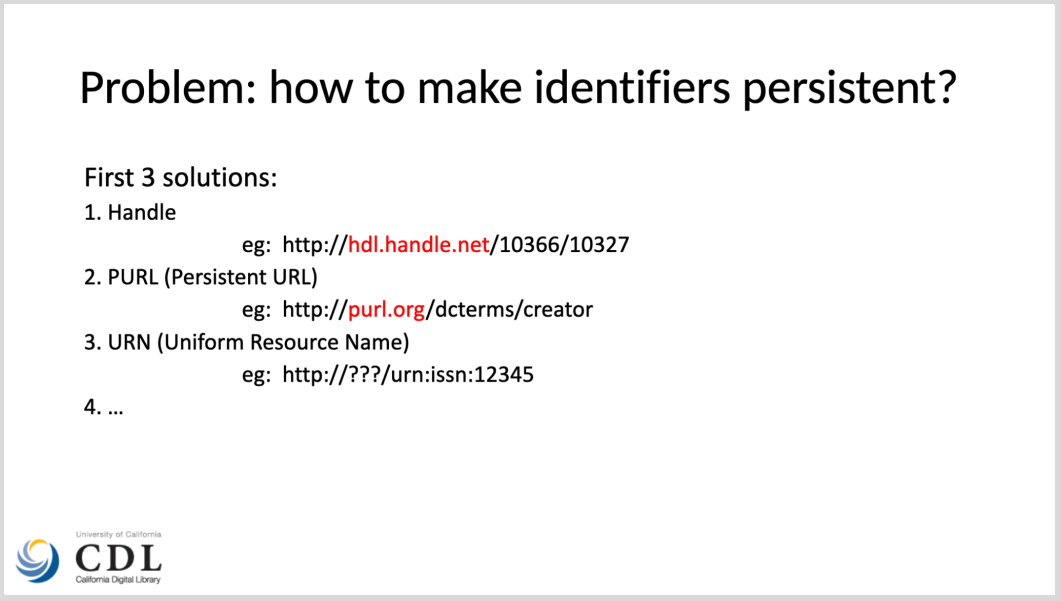
University of California CDL California Digital Library

Redirection is a technique for taking a request for an old URL and forwarding it to a new URL, similar to forwarding [snail] mail when you change your residential address. Redirection is a simple, ancient, and common practice on the web. Starting with the first web servers, to redirect an old URL to a new URL took a webmaster one operation. For URL shortening services, redirection is the main thing they do.

A web server that performs redirection is called a resolver. Persistent identifier systems always involve resolvers. The fundamental idea is that the content provider advertises an identifier that uses the resolver server address and not the content server address. Over time, the resolver's redirection table will be updated to forward the original URL to its current content server URL, whatever that might be. The content provider always has responsibility for keeping the current URL up-to-date in the resolver table.

Any content provider can be a local resolver by updating its own web server redirection table. This kind of local resolver works fine unless the server is shut down for good. To guard against such a case, typically a provider will join a group of providers agreeing to share one neutrally branded resolver for their URLs. That shared address keeps working even if one or a few providers leave the group. Effectively, this replaces a local resolver with a shared global resolver. It's a simple trick often used by consortia and persistent identifier systems.

So the new problem to solve was how to make identifiers persistent. The slide shows example identifiers and their resolvers in red.



Problem: how to make identifiers persistent?

First 3 solutions:

1. Handle
   eg: http://hdl.handle.net/10366/10327
2. PURL (Persistent URL)
   eg: http://purl.org/dcterms/creator
3. URN (Uniform Resource Name)
   eg: http://???/urn:issn:12345
4. …

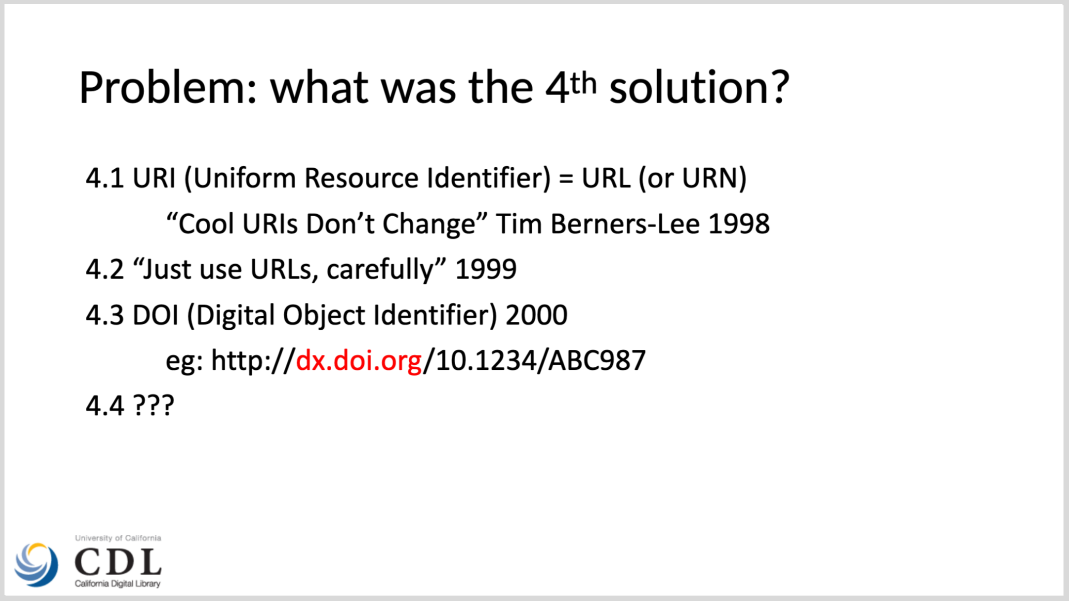
As you might expect, there were 3+1 solutions.

The first was the Handle system. Its two main strengths are to provide a redirection table and the shared resolver trick. These were quite simple things that a content provider might do for itself, but unfortunately the Handle system made it complicated, charged fees, and wanted to control all content access. It seemed like a step backward, and the Internet experts quickly rejected this solution, but it was later adopted by a group of publishers who were comfortable with fees and limited access. The DOI system, which I'll describe later, was built on top of the Handle system.

The second solution was PURL, which stands for Persistent URL. It provides a redirection table and a shared resolver. Unlike Handles, PURLs were open source and free of charge. PURLs did not surrender to the false logic that URLs are inherently unstable, and instead embraced URLs because they are actionable. The main problems with the PURL system were that you could not reliably detect if a given URL was a PURL, and the software and shared resolver were not maintained. Despite being quite popular, last year the Internet Archive had to rescue the service from dying.

The third solution was the URN, or Uniform Resource Name. It was the hope of the Internet experts that the URN standard would create a free, open, standards-compliant solution for the broken link problem. Unfortunately, the solution was rushed and incomplete. There has never been a full implementation or a shared resolver to make URNs actionable.

At this point, there were 3 solutions, but each one had logical and practical flaws. People advocated for URNs, Handles, and even DOIs, under the belief that URLs are inherently unstable. Ironically, those identifiers were useless compared to URLs. The web browsers never adopted those identifiers, so they were never actionable. Their persistence score was 0 because, in a sense, they were always broken. But eventually, the much maligned URL saved them. To make URNs, Handles, and DOIs actionable, users were officially instructed to embed them inside URLs.



Problem: what was the 4th solution?

4.1 URI (Uniform Resource Identifier) = URL (or URN)

"Cool URIs Don't Change" Tim Berners-Lee 1998

4.2 "Just use URLs, carefully" 1999

4.3 DOI (Digital Object Identifier) 2000

eg: http://dx.doi.org/10.1234/ABC987

4.4 ???

University of California CDL California Digital Library

With these unsatisfactory solutions, would a 4th solution emerge?

There was still so much faulty logic and confusion that we were more than one solution away from the best solution. As I see it, there were in fact 3+1 additional solutions before we got there.

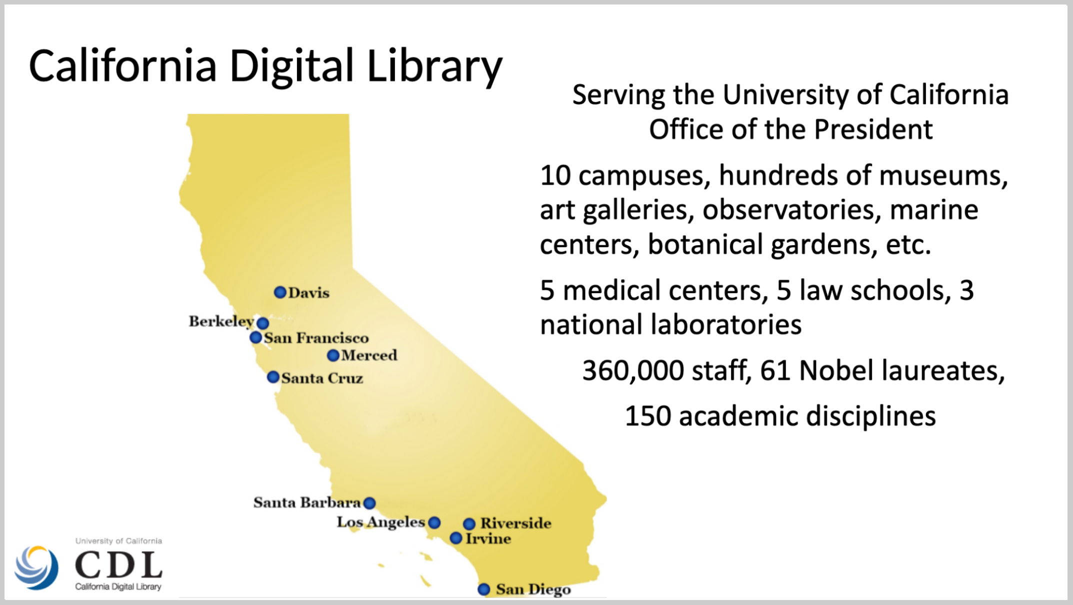
One lasting effect of the URN was to force creation of a new term, URI, or Uniform Resource Identifier. URI could mean either URL or URN, however, when people use the term, URIs, it is usually to talk about URLs.

Tim Berners-Lee never really accepted any of the previous solutions. In 1998 he wrote a short article to say that a URI, meaning a URL, could be perfectly stable if properly managed.

Around this time, I conducted a study of persistent identifier systems for the U.S. National Library of Medicine. I concluded that there appeared to be some risk and no real benefit to any of the proposed solutions, and I made almost the same recommendation: it is safer and cheaper to assign and manage your URLs and your server address carefully. That was the second solution.

The third solution was the DOI, which had the backing of publishers. The DOI system was expensive to use and not open source. On the other hand, it was the first time that people with real money and deep experience managing content had begun to deal with the socio-organizational problems. The Crossref group of publishers did an excellent job building tools, services, and community around persistence.

We now had 6 solutions to choose from. Each one had claims of persistence that were impossible to confirm because no one can actually predict the future. Compared to URLs, each solution brought cost, risk, and complexity but with unclear benefit.



California Digital Library

Serving the University of California Office of the President

10 campuses, hundreds of museums, art galleries, observatories, marine centers, botanical gardens, etc.

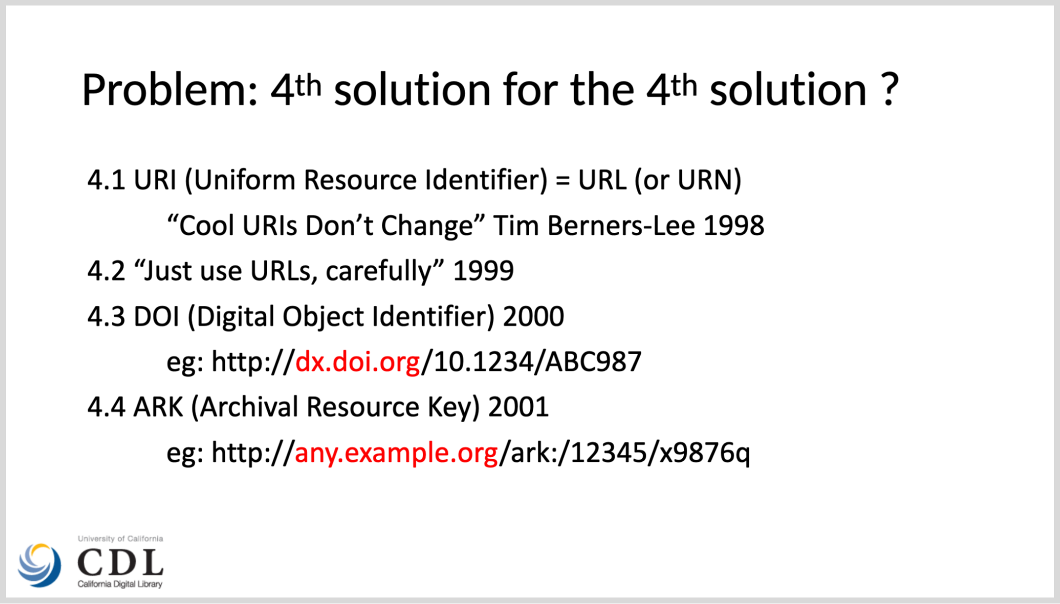
5 medical centers, 5 law schools, 3 national laboratories

360,000 staff, 61 Nobel laureates,

150 academic disciplines

I soon began to work at the California Digital Library, located in the large University of California system that includes Berkeley, San Francisco, and 8 other campuses. That also includes medical centers, law schools, and national laboratories, as well as hundreds of libraries, museums, galleries, and botanical gardens.

I thought I was finished with identifiers. It turns out that identifiers were not finished with me. The University of California serves over a quarter million people in 150 academic disciplines, and all of them wanted persistent access to scholarly content. Should not URLs, properly managed, suffice? The problem was that for many URLs, it was entirely appropriate for them to fade away and not persist, but there was no way to distinguish URLs intended to persist from other URLs.



Problem: 4th solution for the 4th solution ?

4.1 URI (Uniform Resource Identifier) = URL (or URN)

“Cool URIs Don’t Change” Tim Berners-Lee 1998

4.2 “Just use URLs, carefully” 1999

4.3 DOI (Digital Object Identifier) 2000

eg: http://dx.doi.org/10.1234/ABC987

4.4 ARK (Archival Resource Key) 2001

eg: http://any.example.org/ark:/12345/x9876q

University of California
CDL
California Digital Library

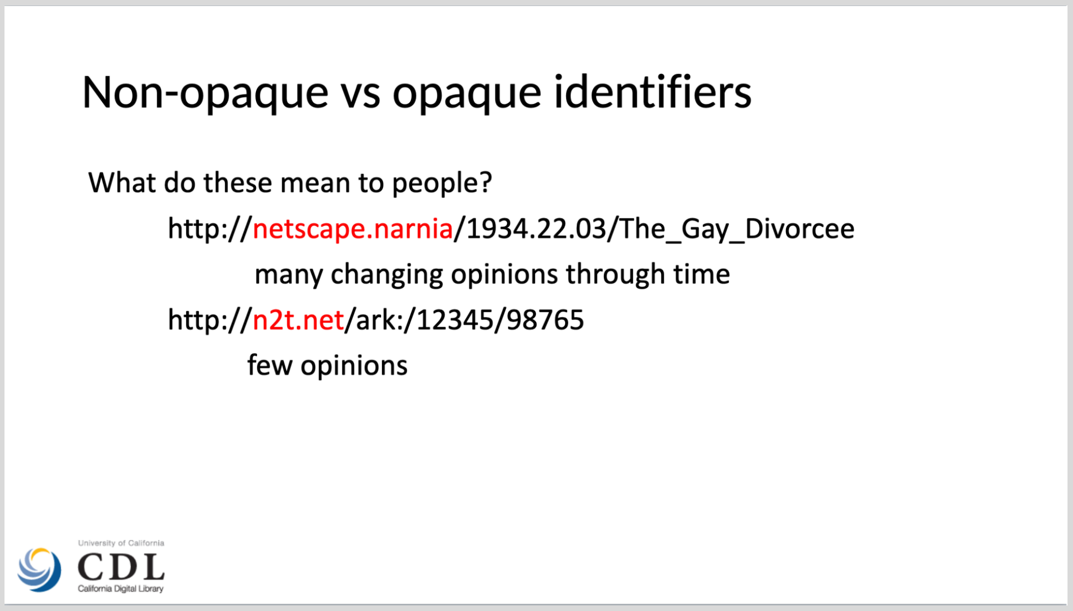
Perhaps a special label inside the URL could tell you whether it was meant, at some time, by some one, to persist? This was the beginning of the ARK identifier, which you can read from this internal label.

You may recall my stating that such a label could never be a guarantee, however, it could be a clue. What if one could interact with this ARK identifier to learn its current status, who the current provider was, and what commitment, if any, was being made today?

What if, furthermore, the ARK identifier were actionable from the start, and contained a globally unique part that could be based at your own local resolver, or a successor resolver from another organization, or a shared global resolver? And if it could be accessible by any future protocol?

And what if ARK identifiers were as easy and inexpensive to create and use as URLs? These questions informed the design of the ARK.

As I mentioned before, assigning an ARK, PURL, Handle, URN, or DOI is never a guarantee. It is merely a hope and a wish. There’s really no way to know if any identifier will be persistent. The ARK is meant to be a “talking identifier”. Ask it to tell you about itself. Its answers will give you the information you can use to make your own judgement.



Non-opaque vs opaque identifiers

What do these mean to people?

http://netscape.narnia/1934.22.03/The_Gay_Divorcee

many changing opinions through time

http://n2t.net/ark:/12345/98765

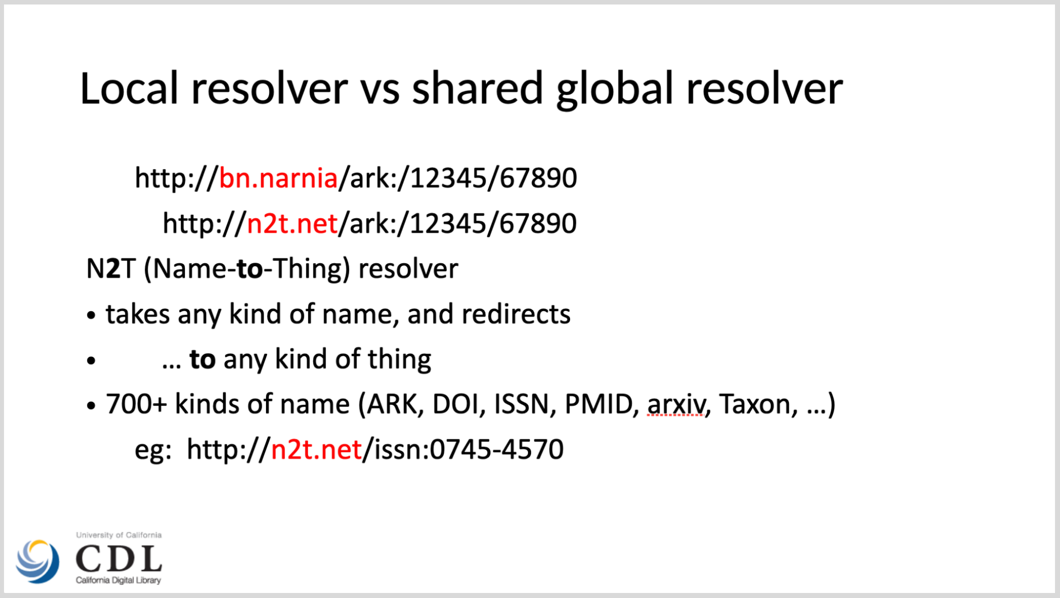
few opinions

University of California
CDL
California Digital Library

A while back I mentioned a feature of the URL that is actually controversial. This is the ability of the URL to describe the content it identifies. The first URL shown is an example. When words inside it are widely recognizable, they suggest or hint at meaning. That can be very useful when handling identifiers. Such a URL is really a tiny descriptive record, and it can give a pretty good idea of what the content was. The problem, however, is that it ages poorly. 80 years later the content may not have changed, but the meaning of the description inside the identifier often has changed.

Companies and even countries disappear. Date formats change. Natural language changes. This is called “semantic rot”, and is especially true for names of organizations found in server names. When your old words fight with their new meanings, people complain, and that makes it hard for your organization to keep those identifiers persistent. So one might be tempted to call this a “talking identifier”, but it’s really more of an endlessly repeating pre-recorded message, suggesting something that might have been true once upon a time. It can be very useful in the short term, but if persistence is important you should choose identifiers without widely recognizable meaning, in other words, opaque identifiers.

On this slide the first URL is non-opaque and the second URL is opaque. Note that there is no such thing as completely opaque identifiers because software needs to recognize some semantics in order to provide identifier services. You can always create non-opaque ARKs, but if persistence is your goal, you are strongly encouraged to create opaque ARKs.



Local resolver vs shared global resolver

http://bn.narnia/ark:/12345/67890

http://n2t.net/ark:/12345/67890

N**2**T (Name-**to**-Thing) resolver

- takes any kind of name, and redirects
- … **to** any kind of thing
- 700+ kinds of name (ARK, DOI, ISSN, PMID, arxiv, Taxon, …)

eg: http://n2t.net/issn:0745-4570

University of California CDL California Digital Library

The resolver server name in the ARK is a special case for semantics. It is a feature of ARKs that you can host them on your own server, or local resolver, instead of a global resolver.

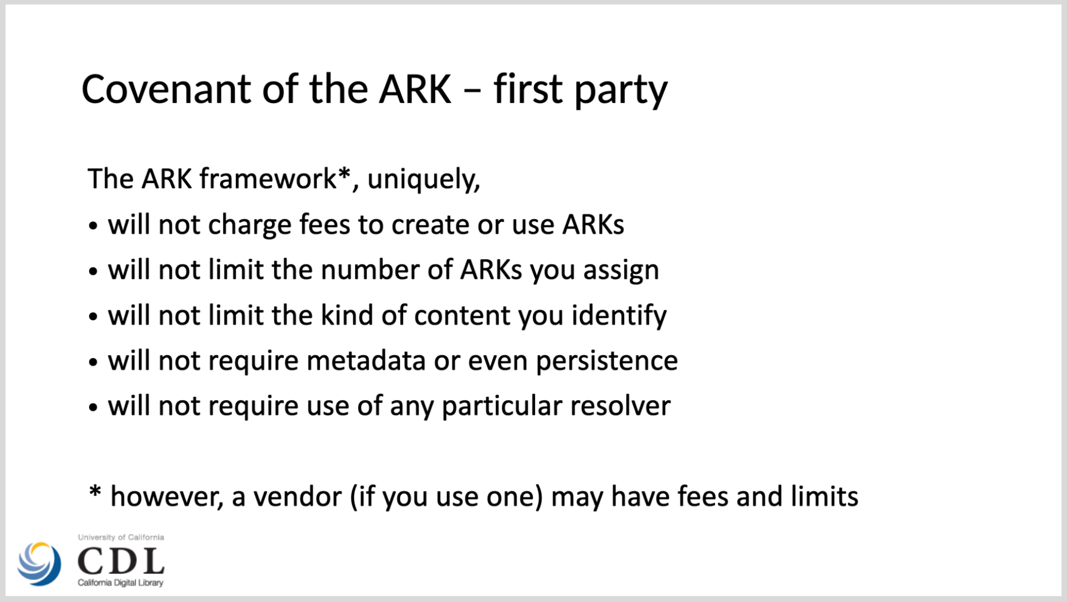
If you are a national library, your resolver name is likely to be stable, but even then there are exceptions. Many organizations that need identifiers are small and have unreliable funding sources. If they lost the server name used in their ARKs, it would be impossible to continue to serve them. But if the server name in the ARKs were a shared global ARK resolver, loss of their local server name would not by itself be a problem. A shared resolver can store redirection information for many organizations, and it can withstand the departure of one or a few organizations.

The N2T.net resolver is a shared global resolver run by the California Digital Library. Although we built it to resolve ARKs, we did not call it the “ARK resolver”. The reason is that we could not justify excluding other types of identifier. Such discrimination would violate basic principles of openness and would require an artificial rejection step in the software. So we gave it the generic name, N2T, because it resolves any kind of Name into any kind of Thing. We feel good that today it resolves over 700 kinds of identifiers. As far as we know, N2T is the only major resolver that is open to all identifiers.

N2T does redirection in two different ways. Let’s say the incoming identifier is the first ARK shown. First, if it has an individual record for that ARK, it forwards it to the target URL stored in that record. CDL has a system called EZID for making and managing long-term identifiers such as ARKs and DOIs. EZID stores all its identifier records in N2T.

For the second ARK shown, if N2T does not have a record for the ARK, it then looks for a record for the NAAN and uses organization-level redirection information. In this case, it causes the entire request to be repeated at the BnF, which actually does have an individual target for the ARK. So you may use N2T for resolution with or without running your own local resolver.

The global resolver that is now N2T.net has been around for about 10 years. Besides EZID, it accepts identifier records from the Internet Archive and from YAMZ.net (an open metadata dictionary). In a pilot collaboration with Crossref, N2T.net also successfully demonstrated resolution with 61 million ingested DOIs from Crossref. This was to explore whether N2T could provide failover for Handle system outages.



Covenant of the ARK – first party

The ARK framework*, uniquely,

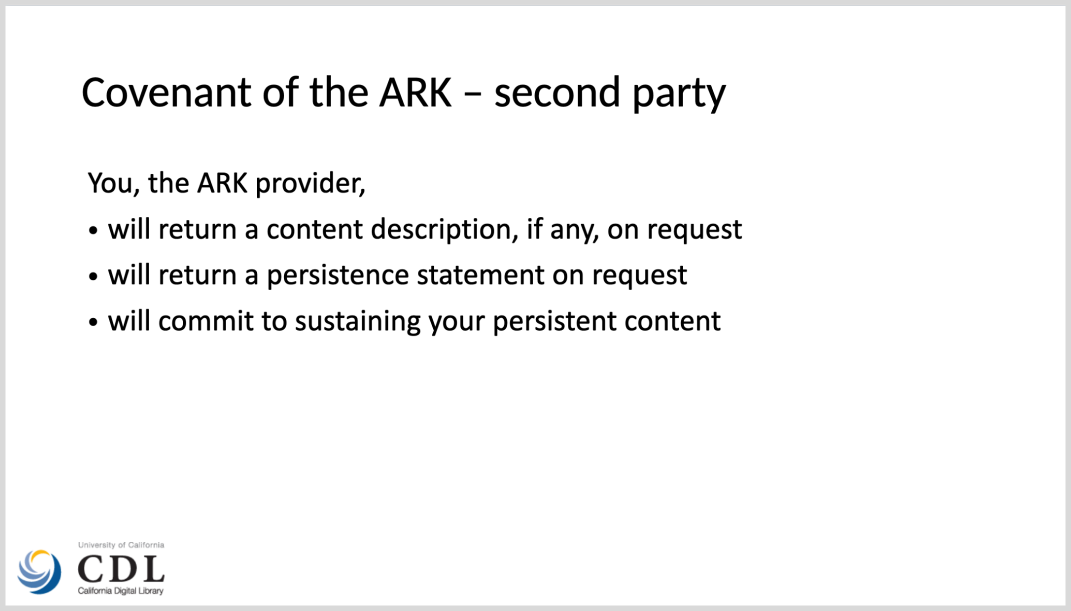
- will not charge fees to create or use ARKs
- will not limit the number of ARKs you assign
- will not limit the kind of content you identify
- will not require metadata or even persistence
- will not require use of any particular resolver

* however, a vendor (if you use one) may have fees and limits

The covenant of the ARK is about obligations between two parties: the ARK framework and the ARK provider. First are the obligations of the ARK framework, which makes an unusual set of promises.

First, it will not charge fees to create or use ARKs. There are no limits to the number of ARKs you assign or to the kind of content they can identify. Metadata and persistence are encouraged, but not required. From this perspective, ARKs are as free to use as URLs. Having said that, if you use a particular vendor, that vendor likely will have fees and limits. Finally, the ARK framework will not require use of any particular resolver.

We believe in open identifiers and discourage limiting access through a single point of control. If there were to be a single point of control, our highest priority would be to establish it as community-owned infrastructure with transparent governance and sustainability.

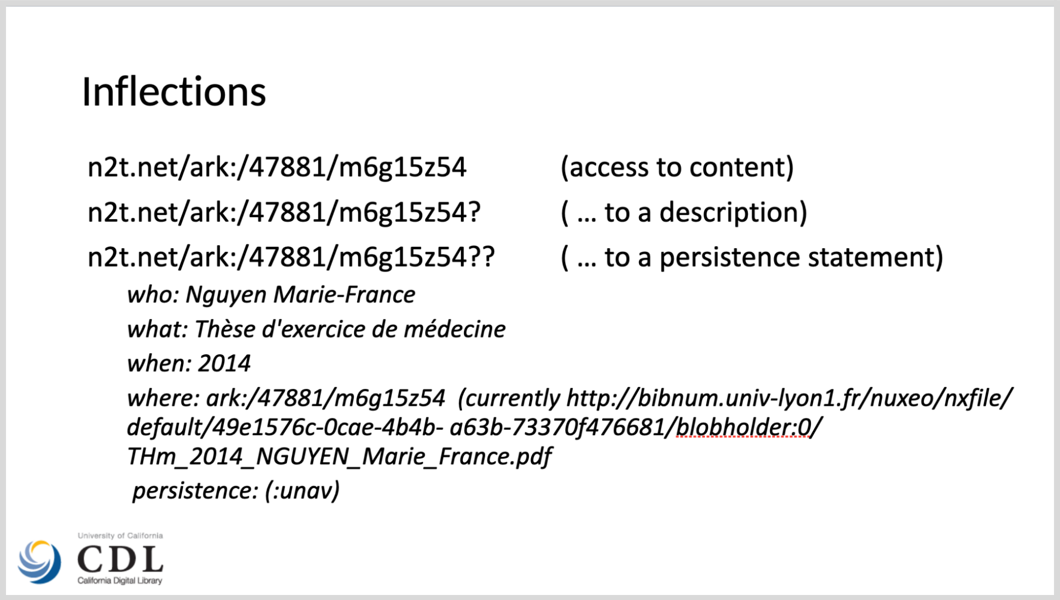


Covenant of the ARK – second party

You, the ARK provider,

- will return a content description, if any, on request
- will return a persistence statement on request
- will commit to sustaining your persistent content

Now we come to the obligations on you, the ARK provider, who is responsible for responding to an ARK request. Often the ARK provider is the same as the content provider. If you have a content description, or metadata, you will return it on request. You will be able to return your commitment, if any, to persistence. If you commit to persistence, you will be able to describe what kind of persistence you mean in regard to your content and your organization. The same applies to your resolver, if you have one.

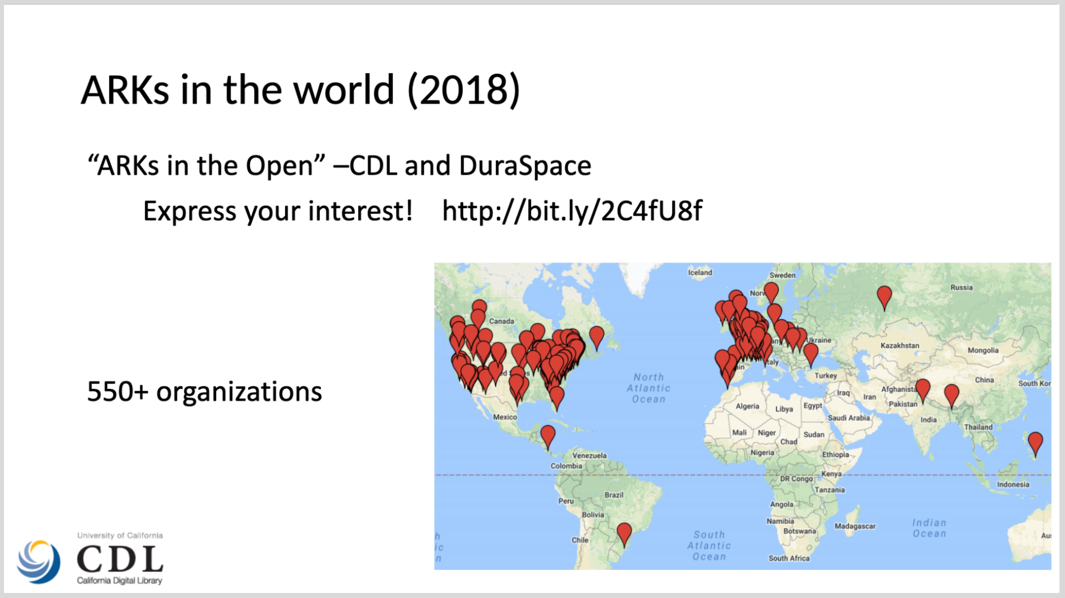


Inflections

n2t.net/ark:/47881/m6g15z54	(access to content)
n2t.net/ark:/47881/m6g15z54?	(… to a description)
n2t.net/ark:/47881/m6g15z54??	(… to a persistence statement)

who: Nguyen Marie-France
what: Thèse d'exercice de médecine
when: 2014
where: ark:/47881/m6g15z54 (currently http://bibnum.univ-lyon1.fr/nuxeo/nxfile/default/49e1576c-0cae-4b4b- a63b-73370f476681/blobholder:0/THm_2014_NGUYEN_Marie_France.pdf
persistence: (:unav)

What does it mean to request a content description or a persistence statement? To avoid creating two new identifiers we re-purpose the ARK but with inflections. Given an ARK like the one shown, you already expect it to give access to content. To see instead what the content is about, simply modify the original ARK by adding a question mark to the end. To see the persistence statement plus the content description, add two question marks. In the example shown, the persistence statement is unavailable.



ARKs in the world (2018)

“ARKs in the Open” –CDL and DuraSpace

Express your interest! http://bit.ly/2C4fU8f

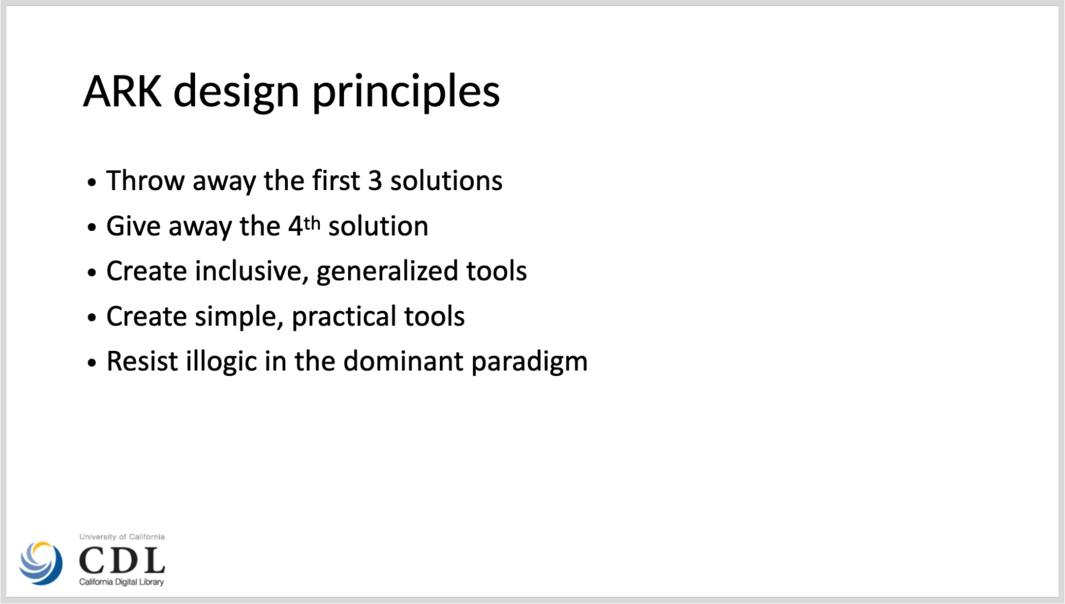
550+ organizations

University of California
CDL
California Digital Library

The ARK system began in 2001, and since then over 550 organizations across the world have registered to assign ARKs, as shown. I’ve been happy to see new registrations coming in at a steady rate of 2-3 per month, despite no promotion other than a presentation every couple of years.

But things may get interesting soon. This month CDL and DuraSpace announced a project called “ARKs in the Open”, designed to build an international community around ARKs as part of the scholarly communications ecosystem. DuraSpace is a not-for-profit organization that helps open source communities and technologies such as Fedora, DSpace, and VIVO. In the long term, we expect this project to help us establish N2T as community-owned infrastructure with transparent governance and sustainability.

We would welcome participation from anyone here. If you would like to be part of defining the vision and future for the ARKs in the Open project, please fill out the expression of interest form at the link shown.



ARK design principles

- Throw away the first 3 solutions
- Give away the 4th solution
- Create inclusive, generalized tools
- Create simple, practical tools
- Resist illogic in the dominant paradigm

The ARK identifier was designed according to values and principles I learned long ago in Berkeley. Construct multiple solutions. When you come upon the right solution you will have a new understanding of the problem. In this case the original problem was: create persistent identifiers. But the right problem was: create high-quality, “talking identifiers”, that can help users and providers to do the best job they can for persistence.

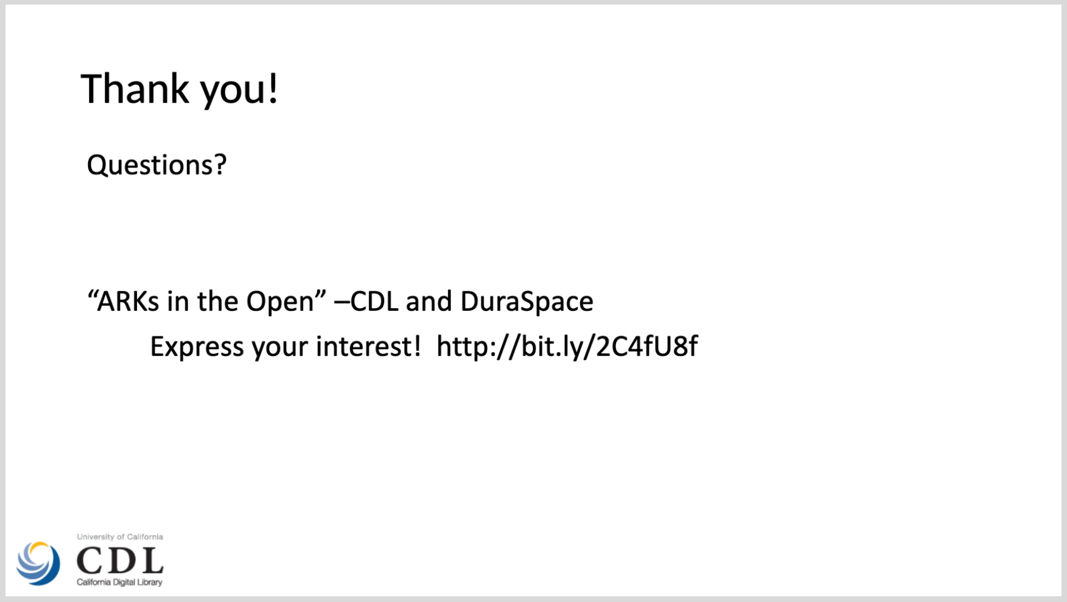
Solutions should be based on open access and open infrastructure. Give your tools away so that all can benefit.

Tools should be inclusive about content types. For example, ARKs identify anything – books, bones, statues, people, terminology, groups, and events. The University has 150 academic disciplines and scholars increasingly cross disciplinary boundaries.

Tools should be inclusive about identifier types too. Handles and DOIs have resolvers that only support Handles and DOIs, but tools that we built for ARKs also work for any kind of identifier. For example, the Noid software is used by many to create ARKs, but major Handle adopters also use Noid, and our N2T resolver supports 700 kinds of identifier. Why do other major resolvers only support one kind of identifier, excluding all others? When we were implementing inflections, an ARK concept, we could have prevented DOIs from benefitting, but that would artificially withhold a feature based on exclusionary principles, which we reject.

Tools should also be simple and do one task well. They can then be re-combined with other tools to do more complex tasks.

Finally, Berkeley is also known for protests. ARKs are a kind of protest on behalf of principles and reason. Sometimes it is important to resist the dominant paradigm in order to make change that benefits everyone.



Thank you!

Questions?

"ARKs in the Open" –CDL and DuraSpace

Express your interest! http://bit.ly/2C4fU8f

University of California
CDL
California Digital Library

--- END ---