

Open Knowledge: Promises and Challenges

Jonathan Gray, Rufus Pollock, Jo Walsh, The Open Knowledge Foundation [1]

Abstract

Open Knowledge is material that others are free to access, re-use and redistribute (see opendefinition.org). We are just beginning to witness the great potential of what can be done with it.

Increasing the visibility and discoverability of open resources is crucial if we are to encourage innovative re-combination and re-use. Hence the importance of open metadata for open knowledge.

Componentization – or the atomization of a given resource into 'packages' – has greatly contributed towards the ease with which software developers are able to re-use and build upon each other's work. We argue that this kind of approach is becoming significantly more important in knowledge development.

This paper will discuss some of the Open Knowledge Foundation's work in these areas – with an emphasis on Public Domain Works and the Comprehensive Knowledge Archive Network (CKAN).

Introduction

The Open Knowledge Foundation is a not-for-profit organisation founded in 2004 with the aim of protecting and promoting open knowledge in all its forms. By 'knowledge' we mean any text, data, image, multimedia and so on. By 'open' we mean free for anyone to access, re-use and re-distribute. (For more details see opendefinition.org.)

Our work can broadly be broken down into promoting the idea of open knowledge, doing research and policy work, developing various open knowledge projects and tools for open knowledge.

Promoting the idea of open knowledge

We have created the Open Knowledge Definition to provide a clear set of conditions for openness in relation to knowledge. This provides a common thread to material that is made available under different liberal licenses (such as Creative Commons Attribution and Attribution-Sharealike, the GNU Free Documentation License...), material in which rights have been waived (CC-Zero, the Public Domain Declaration License...), material that is in the public domain, and so on.

Our 'open knowledge' and 'open data' web buttons are intended to publicise 'openness' regardless of the legal basis of this. We have also drafted an Open Service Definition to fulfil the same function in relation to Software as a Service (SaaS).

We aim to act as a hub and partner for the community of users and producers of open knowledge – facilitating discussion through our mailing lists, forums and annual conferences.

Research and policy work

We produce material on legal, economic, and domain-specific issues relevant to open knowledge at UK, EU and international levels. [2]

Open knowledge projects

We help to initiate and maintain specific 'open knowledge projects':

- ∞ Open Shakespeare is a complete collection of Shakespeare's works with ancillary information, a concordance and an annotation tool; [3]
- ∞ Open Economics is a data store for economic data, plus a visualisation tool; [4]
- ∞ Open Text Book is a registry of textbooks that are fully open; [5]
- ∞ Public Domain Works is a registry of artistic works that are in the public domain – last year it merged with the Open Library. [6]

Open knowledge tools

Our KForge project is an open source system for managing software and knowledge projects – integrating tools such as a versioned storage system, a wiki, a tracker and a blog with the systems own facilities for projects, users and permissions. [7] We also run a free service called KnowledgeForge which runs on the Kforge software and currently houses a variety of open knowledge projects – from British parliamentary data to the works of Ivo of Chartres. [8]

The Comprehensive Knowledge Archive Network (CKAN) is a registry of open knowledge packages – we shall return to this later.

Databases of metadata and metadata for databases

Openness means cheaper and better access to knowledge, as well encouraging richer ecologies of sharing and participation. For example, the Dbpedia project extracts structured information from wikipedia articles to allow complex querying. The W3C community project Linking Open Data is working hard towards inter-linking various open datasets. An increasing number of projects such as Gapminder, Swivel, and Manyeyes seek to socialise the process of visualising and analysing (open) datasets. The 'Principle of Many Minds', which we often allude to, states that 'the most interesting thing to be done with your material will be thought of by someone else'.

In order to encourage this kind of collaboration it is essential that open knowledge resources are as visible and as easily discoverable as possible. Having more and better metadata is one way to facilitate this.

Much metadata is of the kind we find in library card catalogues. For our Public Domain Works project we wanted to build up a large registry of metadata for artistic works – and then to use this metadata to determine which of these works are in the public domain, and hence open. Unfortunately we found that a lot of the material we were interested in was closed and prohibitively expensive. We were luckily enough to be donated several databases from the BBC and from private enthusiasts.

Last year the project merged into the Open Library project – the brainchild of Brewster Kahle, who also founded the Internet Archive – which aims to provide a very large versioned database of bibliographic data. [9] They've had some donations of data from libraries in the US. We're keen to work with them, and any other interested parties, to create a series of 'public domain calculators' which could be used to determine whether a

given work is in or out of copyright in a given jurisdiction. While this constitutes a significant development in this area, unfortunately most bibliographic data is proprietary and cannot be re-used or built upon by the technical community.

As well as metadata for specific works, we can also have metadata for large collections of knowledge resources. We believe that this is integral in order to support greater re-use and re-combination of knowledge resources.

Componentisation and Open Knowledge

Collaborative production and distribution of data is gradually progressing towards the level of sophistication displayed in software. Data licensing is important to this progression, but is often over-examined. Instead we believe the crucial development is componentization. By focusing on the packaging and distribution of data in a shared context, one can resolve issues of rights, reportback, attribution and competition. Looking across different domains for "spike solutions", we see componentisation of data at the core of common concern.

For those familiar with the Debian distribution system for Linux, the initial ideal is of "a debian of data". Through the 'apt' package management engine, when one installs a piece of software, all the libraries and other programs which it needs to run are walked through and downloaded with it. The packaging system helps one 'divide and conquer' the problems of organising and conceptualising highly complex systems. The effort of a few makes re-use easier for many; sets of related packages are managed in social synchrony between existing software producers.

Code got there first

In the early days of software there was little arms-length reuse of code because there was little packaging. Hardware was so expensive, and so limited, that it made sense for all software to be bespoke and little effort to be put into building libraries or packages. Only gradually did the modern complex, though still crude, system develop. These days, to package is to propagate, and to be discoverable in a package repository, is critical to utility. What makes distribution of data the same; what makes it different?

The size of the data set with which one is dealing changes the terms of the debate. Genome analysis or Earth Observation data stretches to petabytes. Updates to massive banks of vectors or of imagery impact many tiny changes across petabytes. At this volume of data it helps to establish a sphere of concern - distributing the analysis and processing across many sets of users, in small slices.

Cross-maintenance across different data sets - rebuilding aggregated updates - becomes more important. Having cleanly defined edges, something like a "knowledge API", or many APIs, is envisaged. Each domain has a set of small, concrete common information models. To distribute a data package is to distribute a reusable information model with it -- to offer as much automated assistance in reusing and recombining information as is possible.

Licensing clarity is important because without it one is not allowed to recombine data

sources (though there is still a large gap between being allowed and being able). Code got a long way with the legal issues, and differently flavoured Free Software Definitions have gained a good consensus. The state of 'open' data is more uncertain, especially looking at the different ways of asserting the right to access and to reuse data in different legislative regions. Open data practise should demonstrate value, utility, thus it becomes a natural choice, and not an imposition. The Open Knowledge Definition is an effort to describe the properties of truly open data.

Knowledge and Data 'APIs'

Open knowledge research projects are carried out in an atmosphere of "fierce collaborative competition". The Human Genome Analysis project was a shining example: slices of source data were partitioned out to a network of institutions. Near-to-realtime information about the analysis results led to the redirection of resources and support to centres which were performing better. In the context of open media, people are also "competing to aggregate", to compile not mere volume but more cross-connectedness into indexes and repositories of common knowledge.

Progress on the parts is easier to perceive than on the whole. In the parts, the provenance is clear -- who updated data when and why, and how it was improved. The touchstones are to improve reusability, accuracy, and currency of data. Working with subsets of datasets, in the absence of significant hardware or bandwidth barriers, anyone can start to carry out and contribute analysis from home. Knowledge is given back into a publically available research space, becoming easier to build on the work of others. The more people who access and analyse data, the more value it has to everybody.

As open source software has shown so well, "openness" is complementary to commercial concerns, not counter to them. As the GPL encourages commercial re-use of code, open knowledge is of benefit to commercial activity. Providing a "reference system" and a common interface, more "added value" applications are built on a base layer. The ability to monitor and report in near to realtime on the basis of package development can be useful to more than the "funded community"; it provides real validation of a working (or non-working) business model.

What Do We Mean by Componentization?

Componentization is the process of atomizing (breaking down) resources into separate reusable packages that can be easily recombined.

Componentization is the most important feature of (open) knowledge development as well as the one which is, at present, least advanced. If you look at the way software has evolved it now highly componentized into packages/libraries. Doing this allows one to 'divide and conquer' the organizational and conceptual problems of highly complex systems. Even more importantly it allows for greatly increased levels of reuse.

The power and significance of componentization really comes home to one when using a package manager (e.g. apt-get for debian) on a modern operating system. A request to install a single given package can result in the automatic discovery and installation of all packages on which that one depends. The result may be a list of tens -- or even hundreds --

of packages in a graphic demonstration of the way in which computer programs have been broken down into interdependent components.

Atomization

Atomization denotes the breaking down of a resource such as a piece of software or collection of data into smaller parts (though the word atomic connotes irreducibility it is never clear what the exact irreducible, or optimal, size for a given part is). For example a given software application may be divided up into several components or libraries. Atomization can happen on many levels.

At a very low level when writing software we break things down into functions and classes, into different files (modules) and even group together different files. Similarly when creating a dataset in a database we divide things into columns, tables, and groups of inter-related tables.

But such divisions are only visible to the members of that specific project. Anyone else has to get the entire application or entire database to use one particular part of it. Furthermore anyone working on any given part of one of the application or database needs to be aware of, and interact with, anyone else working on it -- decentralization is impossible or extremely limited.

Thus, atomization at such a low level is not what we are really concerned with, instead it is with atomization into *Packages*:

Packaging

By packaging we mean the process by which a resource is made reusable by the addition of an external interface. The package is therefore the logical unit of distribution and reuse and it is only with packaging that the full power of atomization's "divide and conquer" comes into play -- without it there is still tight coupling between different parts of a resource.

Developing packages is a non-trivial exercise precisely because developing good *stable* interfaces (usually in the form of a code or knowledge API) is hard. One way to manage this need to provide stability but still remain flexible in terms of future development is to employ versioning. By versioning the package and providing 'releases' those who reuse the packaged resource can stay using a specific (and stable) release while development and changes are made in the 'trunk' and become available in later releases. This practice of versioning and releasing is already ubiquitous in software development -- so ubiquitous it is practically taken for granted -- but is almost unknown in the area of open knowledge.

Componentization for Knowledge

We are currently at a point where, with projects such as wikipedia, we have powerful examples of the first three principles in action but little or none on the fourth.

In the early days of software there was also little arms-length reuse because there was little

packaging. Hardware was so expensive, and so limited, that it made sense for all software to be bespoke and little effort to be put into building libraries or packages. Only gradually did the modern complex, though still crude, system develop.

The same evolution can be expected for knowledge. At present knowledge development displays very little componentization but as the underlying pool of raw, 'unpacked', information continues to increase there will be increasing emphasis on componentization and reuse it supports. (One can conceptualize this as a question of interface vs. the content. Currently 90% of effort goes into the content and 10% goes into the interface. With components this will change to 90% on the interface 10% on the content).

The change to a componentized architecture will be complex but, once achieved, will revolutionize the production and development of open knowledge.

The Comprehensive Knowledge Archive Network (CKAN)

Our CKAN project aims to encourage and support the emergence of a culture where knowledge packages can be easily discovered and plugged together as is currently possible with software. Named after software archives such as CPAN for Perl, CTAN for TeX, CRAN for R and so on, it is a registry for knowledge resources.

It is currently in beta and consists of a versioned database of metadata for large datasets and substantial collections of knowledge resources – 'from genes to geodata, sonnets to statistics'. It gives the 'lowest common denominator' of metadata for its packages: author, id, license, user-generated tags, and links. We plan to add support for domain specific metadata. We are also planning to make provision for the automated installation of knowledge packages.

Conclusion

We hope to have indicated the importance of open metadata and componentization for open knowledge, and the importance of openness for a vibrant knowledge society.

We look forward to working with COMMUNIA members towards strengthening and increasing the visibility of the digital public domain in Europe.

Notes and references

[1] Substantial parts of this paper are based on *Componentization and Open Data*, delivered by Rufus Pollock and Jo Walsh at XTech 2007.

[2] See <http://www.okfn.org/wiki/Research>

[3] <http://www.openshakespeare.org/>

[4] <http://www.openeconomics.net/>

[5] <http://www.opentextbook.org/>

[6] <http://www.publicdomainworks.net/> and <http://www.openlibrary.org/>

[7] <http://www.kforgeproject.com/>

[8] <http://www.knowledgeforge.net/>

This article is licensed under a [Creative Commons Attribution 3.0 license](https://creativecommons.org/licenses/by/3.0/).