Designing APIs and Designing Value for an Education Graph

Alan Dix

Talis and University of Birmingham

http://alandix.com/papers/apis-2012/

APIs (application programmer interfaces) are part of the fabric of Web interaction from ubiquitous Google maps to Facebook 'Like'. They have fostered the growth of many major web applications, enable niche mash-ups servicing the long-tail of web2.0 development, and enhance user experience in many sites. This paper uses a case study of an educational application to explore the area of API design. It finds that HCI issues, technical design and market considerations are not siloed concerns, but interact closely in the design of rich APIs that provide value and ease of development to developers and enhanced user experience for end-users.

API, Linked Open Data, designing value, web development

1. INTRODUCTION

Public APIs (application programmer interfaces) are now part of the fabric of Web interaction. Some are more behind the scenes, such as cloud data management; some are embedding APIs more apparent to the user, including the ubiquitous Google Maps and Facebook OpenGraph 'Like' buttons on websites. There are also an increasing number of API marketplaces (such as Mashape[1]) and API-only start-ups (e.g. RapLeaf[2]), which simply provide a web service API for others to apply to their own data, or mashup with yet more data,

Designing APIs is partly a technical problem, providing the right information in the right way. However, it is also a rich socio-technical problem, as the nature of APIs influences end-user experience, and the various stakeholders each need to realise appropriate value from their deployment and use.

This paper explores the issue of API design based on theoretical principles rooted in HCI, CSCW and economics, and enacted in the design of an experimental API for an 'education graph' of text books, articles and module reading lists. We find that HCI issues, technical architecture and business value are all crucial interrelated elements.

The rest of this introductory section argues why API design has an HCI as well as a technical dimension and also looks at the Talis Aspire system which is used as the case study for this paper. Section 2 explores some of the theoretical tools that inform API design, in particular issues of mutual value between stakeholders. This is then followed by descriptions of a practical-focused use-based approach where API use-cases are used to drive API design informed by the preceding theory. Finally we return to the broader canvas and discuss some of the general lessons learnt.

1.1 Why API Design is an HCI Issue

While the technical nature of API design is clear, the socio-technical aspects are maybe not so immediately obvious. It might seem that the HCI issues are all about the eventual applications that use the API. Indeed, one approach to API design would be to see the API as 'low-level details' and focus HCI expertise on the applications, on the principle that good usability or user experience design can be built on any base.

There are several reasons for rejecting this simplistic approach and addressing HCI concerns early in the API design process.

First, designing APIs without regard to eventual use is rather like the old (and now happily rare) approach of building a system and then asking the UI designer to "put an interface on it". The bits always leak out and the information and system architecture design of a system inevitably has an

[1] http://www.mashape.com/
[2] https://www.rapleaf.com/

impact on user experience and other aspects of usability. A classic example of this in the CSCW literature is Palen's account of the way calendar naming conventions gave an element of privacy in an otherwise open system at Sun (Palen, 1999).

As a more contemporary example, consider the Dropbox API. This has two kinds of application: (i) one kind has access to a single folder in Dropbox private to that application, (ii) the other kind has complete access (read and write) to the user's entire Dropbox file system. For some applications (i) is sufficient and is obviously a lot safer for the user, but limits the extent to which the application can share data with other applications, reducing usability. If an application wants to share data it must choose option (ii) with the attendant responsibility of having unrestricted access and the risk that knowledgeable users will refuse this.

The second reason is that some embedding APIs include user interface elements. For example, Google Maps is not only a tool for developers, allowing them to create rich map-based applications, but also provides many of the UI elements in these applications, such as scroll and zoom controls, and pop-ups for map pins. Similarly, Facebook OpenGraph provides 'Like' buttons to embed in external sites, and login functionality for them.

Third, the developers are users of the API. The ease of use of an API as a tool will have an impact on a developer's choice to adopt it and experience of using it. Furthermore, ease of use of the API as a tool frees the designer/developer to focus on the design of the application, making them more likely to provide a good end-user experience.

Finally, there is the big human picture of interactions between major stakeholders: end user, API provider, and the developer of applications or web pages using the API. This is about understanding the motivations and values of multiple parties and is a part of what Shneiderman (2011) has recently called 'macro-HCI'.

1.2 The Education Graph

Talis Aspire Campus Edition (Clarke, 2009) is a commercial reading list management system used by a significant proportion of UK Universities and some elsewhere, often branded as part of institutional VLEs (virtual learning environments). As part of the Talis corporate commitment to open linked data (Bizer, Heath, and Berners-Lee, 2009), Universities are given a substantial discount if they elect for non-personal data created in Aspire to be publicly available as open data, and the majority have chosen to do so.[3] Data including reading lists for modules, bibliographic information, etc., is available as RDF.

This open data has been aggregated, including extensive work to reconcile, different representations of the same resource. This forms the core of an educational graph (Heath et al., 2012), which currently includes reading list information for over 22,000 University courses.

The education graph is at the heart of a non-institutional web application, Talis Aspire Community Edition[4]. This allows users to browse and search the education graph; for example, from a textbook or article one can find on which courses it is recommended, and other resources that are often on the same reading lists. This is similar to Amazon recommendations, except that rather than being based on popular purchasing, this nascent education graph is based on traceable provenance and authoritative sources (to the extent that University lecturers are authoritative!).

This aggregated data is already available in a raw form as RDF[5], and it is the design of an experimental richer API for this aggregated dataset that is the topic of the case-study facet of this paper.

The reason for wanting to make the education graph more openly available is partly driven by the core Talis ethos, and partly by the desire to grow this education graph in terms of volume of resources and richness of content. This may include linking to additional author materials for textbooks, and intra- or cross-institutional social interactions around resources. In particular, enabling social learning is often regarded as a core element of modern education and an opportunity for social media (e.g., Twidale and Ruhleder, 2004; Evans, Kairam and Pirolli, 2010)[6].

This richer graph could be enabled entirely by additional tools and mechanisms within Aspire itself, and certainly various additional functions are planned. However, experience of many web applications, including Twitter and Facebook, is that open APIs allow innovation and

[3] e.g. http://readinglists.central-lancashire.ac.uk/lists/1D53ADAB-FA02-4475-2E73-43034A5343D2.rdf
[4] http://community.talisaspire.com/
[5] http://www.w3.org/RDF/
[6] Although the story of social learning is far from clear. For example, the authors of the last of these papers conclude that the answer to "Do your friends make you smarter?" is "yes" even though their evidence unequivocally shows that non-social learning out-performed social learning as measured by Bloom's taxonomy (Bloom, et al., 1956).

experimentation, beyond any that would be possible by a single team. Indeed the launch of Facebook Platform in 2007 coincided with a dramatic shift in its growth rate (Figure 1).

Adding a richer API for accessing and embedding Aspire data in third party pages and applications could have similar benefits, but also has to be justifiable commercially.

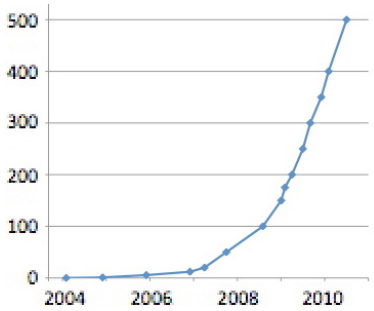

Figure 1: *Facebook Growth 2004–2010 – millions of users (data from Facebook, 2012)*

2. THEORY FOR API DESIGN

While we would not pretend to offer a complete theory *of* API design, there are substantial elements of existing theoretical and practical expertise in the literature, which can be brought to bear.

This theory is particularly important because APIs are 'generative artefacts', things that make other things, for which prior justification plays a stronger role than post-hoc evaluation (Dix, 2008). While one can, and we will, apply APIs to real projects, the resulting applications will owe much of their quality to the application or site designer using the API.

This does not mean empirical work is not critical for API design. Indeed, section 3 describes a very practical exemplar-driven approach as part of API design. It is just that exemplars are more to 'sample the space' of potential application scenarios. The danger in API design (and indeed the design of any generic tool/service) is that in attempting to create something for everything, it ends up being good for nothing. Samples/exemplars spread over the potential space of applications help one to design an API that is more likely to be fit for as yet unthought-of applications.

API Stakeholders

Identifying stakeholders is a core part of all socio-technical design approaches (e.g. Checkland, 1981). In API design this is in a way straightforward: there is the API provider as stakeholder, the developer/designer who is using the API to create an application, and the actual end-users (and other stakeholders) of the final application. However, there are complications to this. The data the API is providing may involve other stakeholders; for example, Aspire data has originally come from participating universities and academics creating reading lists in those universities. In addition, the stakeholders for the final application will vary from application to application. For the API provider they are at best a fairly abstract category. In the case of Aspire data it is likely that these end-users will include students or tutors, but there may be others, librarians, or people outside academia.

The concept of value design is crucial here (Dix et al., 2004, pp.159–160). We need to create value for each stakeholder in order for the API to succeed. Arguably the value for the end-user is the application developer's job, but the API developer needs to consider this, especially if the API includes the provision of embedded UI components. By enhancing the user experience, the API gives additional reasons for use by the developer; that is increasing developer value in addition to the intrinsic value of the functionality offered by the API.

In the case of Aspire the end-users for API developers are also likely to be direct users of Aspire. This can increase the potential for virtuous spirals of adoption where an existing user base encourages third-party developers and third-party applications encourage further user growth. This 'lattice of value' (Dix, 2001) is an example of a 'market ecology', mapping out the relationships between potential classes of users of a product or product family, and the patterns of mutual benefits between them (Dix et al., 2011).

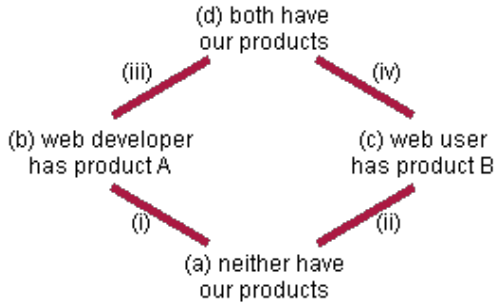

Figure 2: *Lattice of Value (from Dix, 2001). Steps (i) and (iv) require value for developers, steps (ii) and (iii) require value for the end-user. At least one of paths (i)–(iii) or (ii)–(iv) must exist for a product to experience self-generated growth.*

The Value of Networks

In principle there is 'value to spare' for the API developer and API provider due to so-called

'network externalities' or 'network effects' (Economides, 1996; Liebowitz and Margolis, 1998). This refers to the way that certain kinds of items, in particular digital ones, may vary in their value to one individual dependent on another individual's use. If one person is using a particular spreadsheet, then the value of that spreadsheet increases for the person's colleagues, as they may want to share data.

This has been quantified in Metcalfe's Law "*The value of a network goes up as the square of the number of users*" (Shapiro and Varian, 1999, p.184). The exact formula is a matter of debate (Briscoe, Odlyzko and Tilly, 2006), but there is little doubt that the value of social, communications and data networks increases faster than the number of people / nodes / data items[7].

For the education graph, this means that if an API user is in some way adding to the graph of resources (perhaps by creating additional lists, links between resources, new material such as reviews), then the final value of the network is greater than the sum of both the API provider's data and that of the API user – there should be spare value for both to be better off.

Asymmetric Benefits – Strong Gatekeepers

It is the potential for mutual benefit that is at the heart of arguments for the business and social benefits of networks (Benkler, 2006; Varnelis, 2008). However, while the total value may be higher, this is no guarantee that the added value will be spread evenly. This may fail in either direction.

The API or service provider may have considerable power as gatekeeper to users/customers or data. This power imbalance may be used to extract disproportionate amounts of the net value. For example, the Apple iTunes App Store clearly generates value for App developers and for Apple, but some App developers feel aggrieved at the size of Apple's cut[8]. This may lead to defection or non-adoption of APIs and services, and certainly to disaffection.

In the case of the App Store there is a very clear value (the App price) and the App provider may feel that this is being 'taken' by Apple. In more data-focused APIs this is less of a problem as the additional value generated by the API is often less visible and not seen as 'taken away'. For example, Google gain from having people click through to the full Google Maps web site from Google maps embedded in a web page. This value for Google in no way subtracts from the value the web page owner obtains from having the map. It could easily be that the value Google obtains compared to the web page owner is more than the 30% Apple cut on the App Store, but as it is not seen as being taken from the owner, the mutual benefit is more clear.

Avoiding Free-Riders

On the other hand, the API provider risks the application developer simply using the services of the API, but not in any way giving back to the API provider. For example, if a search engine provided a free search API it would be possible for third party sites to simply set up their own commercial search pages with advertisements and never credit or pass on profits to the original search engine.

This potential for free-riders has been identified in both CSCW and economics literature, although frequently associated with the misnamed "Tragedy of the Commons".[9]

As a data owner there are a number of options:

(i) Don't allow external access at all, but simply build everything you want into your own central site.
(ii) Charge for access and allow third-party users to brand as they like.
(iii) Use terms and conditions (T&C) to force third parties to use your data in ways that return value to you.
(iv) Use technical means to encourage/enforce preferred modes of use.
(v) Ensure that the value proposition for third parties mean they *want* to use your data in ways which give value back to you.

Going through these (i) sounds like a cop-out and would lose the advantages of API-led growth enabled by network effects, but it does not preclude more limited forms of external plug-ins and widgets, as was the case with early Facebook integration.

[7] See Hendler and Golbeck (2008) for discussion of this applied to Web 2.0 and semantic data.
[8] Despite this, developers continue to produce iOS apps, suggesting they are still getting more value than if they developed elsewhere.
[9] Misnamed because Hardin's original (1968) article uses a fable of common land being eroded through unrestrained self-interest. While this may be true for many kinds of systems, in fact actual common lands were (and where they still exist still are) well tended and were actually destroyed by enclosures and often eviction of life-long tenants. See Tierney (2009) and Dix (2011), for recent commentary on this.

Option (ii) is more or less what the existing Aspire Campus Edition does, a traditional charging model where it may be so well integrated into the institutional VLE that staff and students never realise they are using Talis software. This model is common in many 'freemium' based APIs, for example embed.ly[10], which allows site owners to create rich embedded content to sites such as Flickr and YouTube. It allows free use up to small volumes, then charges. The end-user will typically not be aware embed.ly is used on the site at all.

Option (iii) is also not uncommon, often combined with (ii). For example FreeFoto.com, an online photo bank, has paid-for options for certain kinds and volumes of print and web use, but also allows substantial free use of stock photographs so long as the photos are clearly acknowledged.[11] At an API level, the Amazon Product Advertising API (which, inter alia, allows access to book and other product details and images) includes the following clause in its T&C[12]:

> (d) You will link each use of Product Advertising Content to, and only to, the related Product detail page of the Amazon Site, and you will not link any Product Advertising Content to, or in conjunction with any Product Advertising Content direct traffic to, any page of a site other than the Amazon Site ..."

Many user-content sites including Flickr and YouTube adopt option (iv), having means to easily embed content in third-party sites (e.g. see Figure 3). These mechanisms make it more likely that web page authors and bloggers will use Flickr and YouTube material, thus increasing traffic to the main site. However, in addition they make it more likely that they will stick to the T&C in terms of correct attribution or 'game' the systems, for instance, by including very small, or out of the way, attribution.

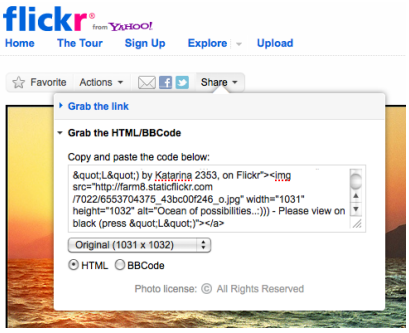

Figure 3: *Flickr embedding dialog)*[13]

The extreme of this is when the method of embedding is completely controlled by the originating site, either using an iframe or black-box JavaScript inclusions. For example, Snap Shots[14] are embedded using a single JavaScript file included in the target site's header, with a site-specific key. It would undoubtedly be possible to reverse engineer the JavaScript to obtain the raw API being used, but few are likely to do this when it is easier to use the API in its intended way.

Of course if the user of an API gets more value from using it in the intended way than subverting it (option (v)), then the need for complex T&C or technical solutions is reduced. For example, when a Flickr user is embedding their own Flickr photos in a blog or web page, they will want to send people back to the Flickr site to see their complete collection.

This does not obviate T&C and technical means entirely (for example, third parties using Flickr images), but does transform the task of API design from an adversarial to a collaborative endeavour.

3. IN PRACTICE

This section describes two use cases and associated prototypes that were used to drive the development of an experimental API. One is simple embedding in course pages; the other, BookNotes, a mini-app to demonstrate third-party use.

One of the authors is an academic and the usage cases were developed from his academic practice. This is a form of 'single person study' or single person design (Razak, 2008). Single person design is particularly powerful for addressing the 'long tail' of niche products (Dix, 2010). Here the intention is not to design these niche applications per se, but more that the API needs to be able to address such niche applications.

This individual-focused design does not obviate more traditional methods, and standard user-centred-design methods (e.g. user interviews, participatory groups, etc.) are being applied alongside.

3.1 Experimental Architecture

In order to create these prototypes, an experimental harness was required so as to experiment with an alternative API without interfering with the real running application. To do this a form of proxy API server was deployed that uses the live data currently available from Talis

[10] http:// embed.ly/
[11] http://www.freefoto.com/browse/99-05-0/Free-Use-Rules
[12] https://affiliate-program.amazon.com/gp/advertising/api/detail/agreement.html
[13] http://www.flickr.com/photos/jup3nep/6553704375/
[14] http://www.previewshots.com/

Aspire Community Edition. (The data behind most user viewable pages is available in JSON format through content negotiation (Fielding, et al., 1999)).

The academic's own web page and the BookNotes application then communicate with the proxy API. This middleware layer relays the original data, but with augmented functionality, for example, delivering content even when the resource is not listed in Aspire.

Figure 4 shows these main elements. Note that both academic and students will also interact directly with Aspire itself, but these interactions have been omitted to show the API mediated interactions more clearly. This is not unlike the architecture of a production environment, except that the proxy functionality would simply become part of the main system.

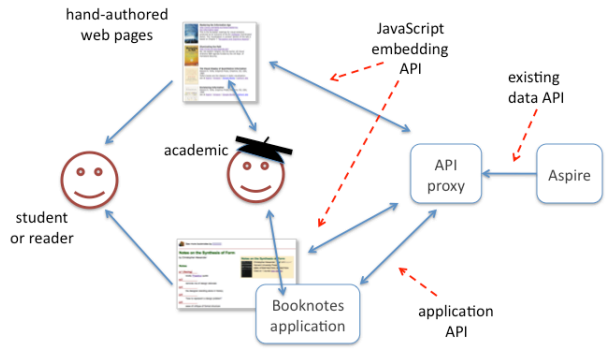

Figure 4: *Experimental architecture*

3.1 Embedding in Academic Web Pages

The first use case is when the academic edits his own web pages. Two example web pages giving resource lists for pages drove this.

The first (figure 5) started with a CSCW course page created with a pre-release version of a course page creation system hosted within Aspire. The author had used this for a previous version of a course, but it had limitations, as it was an early prototype. So, for a re-run of the course, the page was saved as pure HTML and then hand-edited to make it closer to how he would ideally like it. This was partly intended to feedback into future Aspire hosted resource pages, and partly to investigate author embedding of resources.

The second example was a completely bespoke page developed for a short tutorial on visualisation (figure 6).

The first example was trying to be as close as possible to the automatically created resource page, but in a number of places the academic modified book details and even cover images. This was partly because the data in Aspire was only as accurate as the reading list data in the Universities from whence it came, and partly because of preferences (e.g. shorter/longer versions of titles). In general, from both the academic's practice and other interviews, it is clear that academics like to be in control!

The second example highlighted several issues. This was a visually rich list, but the images were not always book covers; in particular, articles were often illustrated using a screenshot or other figure from the content of the article (rather like Facebook's option to choose an image for a web link). In addition nearly every resource link had some accompanying note or comment.

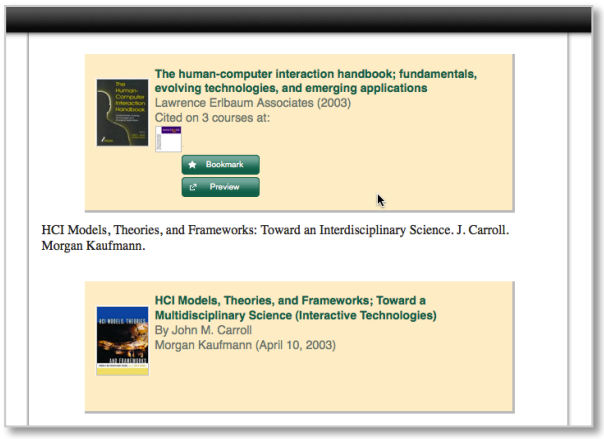

Figure 5: *Hand-edited modification of existing prototype resource page.* [15]

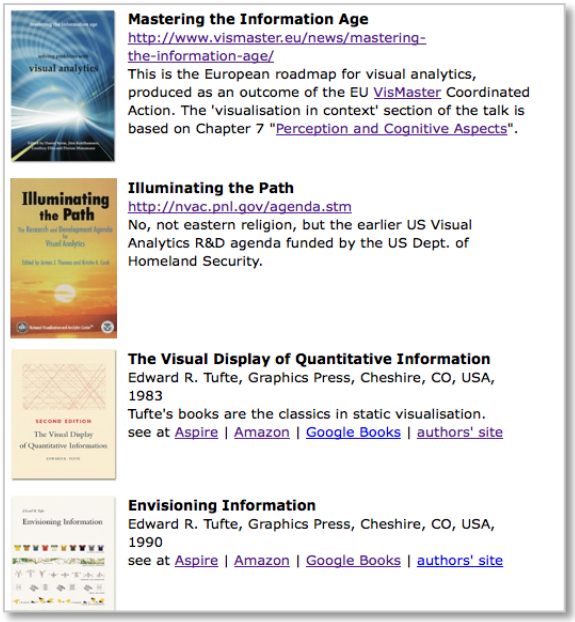

Figure 6: *Completely bespoke resource page*[16].

[15] http://alandix.com/academic/teaching/USI-Eindhoven/gen-2011/
[16] http://alandix.com/academic/teaching/USI-Eindhoven/gen-2011/

Based on this a JavaScript embedding API was created. A small script is included on the web page that scans for links to Aspire resources or tags that have been marked for expansion with a special class. These may simply be a link (Figure 8), but can also include elements marked with classed entities (see link with class `aspire-title` in Figure 9), similar to microformats[17] and an <img> tag. The span classes include 'title', 'author', etc. which are used to override the information derived from the Aspire data, or to include resources not in Aspire. The simple link is often sufficient, but the option to override it is essential to give the page author a feeling that it is ultimately their content.

```
<a
href='http://community.talisaspire.com/
resources/OarshcnC-_J3ip3WOlXJGw' >HCI
Models</a>
```

Figure 8: *Simple link for expansion*

Based on the observations from the second example. The image tag if present overrides the Aspire cover image, allowing alternative images of a book cover or other images to be used. Furthermore, the elements can include a "note' class that allows annotation (see last paragraph tag in Figure 9 and appearance in Figure 10).

```
<div class="acorn-infobox">
  <h2><a
href="http://community.talisaspire.com/
resources/gCR27a9hLAXcmtTsVrREkg "
class="aspire-title">
 Human-Computer Interaction Handbook
</a></h2>
  <p class="aspire-notes">See Chapter
16, Network-Based Interaction, pp. 331-
357.</p>
</div>
```

Figure 9: *More complex expansion with additional notes*

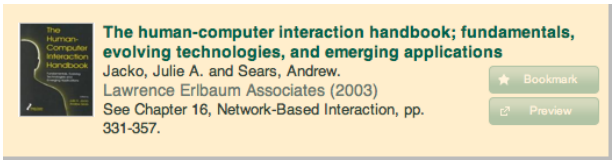

Figure 10: *Infobox on web page generated by the API form the HTML in Figure 9*

3.2 Third-Party Application

The second use case is where the academic does not edit pages directly but uses a third-party application to create content. For this a mini-application, BookNotes,[18] was developed, which accessed the experimental API, but was otherwise disconnected from the Aspire infrastructure. While developed alongside the development of the experimental API, it deliberately uses only access methods that would be expected to be available to a third-party developer.

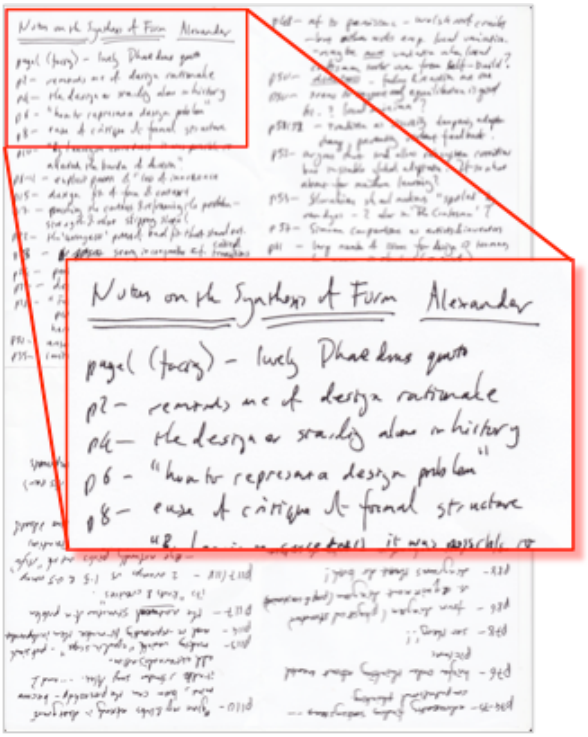

Figure 11: *Handwritten book notes*

BookNotes was inspired by the handwritten notes that the academic kept as bookmarks in his personal library. Inside each book is a folded sheet of A4 paper on which are written page-by-page notes. Figure 11 shows a scan of the notes taken from Alexander's "Notes on the Synthesis of Form" (Alexander, 1964). These are typically of the form 'page number - note", with occasional additional embellishments (e.g. "(facing)" on the first entry.

These hand-written notes are useful when one returns to the book, but cannot easily be searched (except by opening each book to look at the notes), nor shared with students. BookNotes was designed to enable the academic to either quickly transcribe these notes, or to enter them directly into a mobile device whilst reading (a paper copy of the book!)

To use BookNotes, the academic user simply types notes into a plain text file in a Dropbox folder using a minimal formatting (page number -- note). The user tells BookNotes where to find the file in Dropbox and then BookNotes reads and formats the file for the web. Figure 12 shows the raw file format. Because this is simply a Dropbox text file, it can be created using any Dropbox enabled text

[17] http://www.microformats.org
[18] anonymised url

editor, notably those available on iPhone, Android or iPad.

```
Notes on the Synthesis of Form
Christopher Alexander
url: http://community.talisaspire.com/r
esources/pqKNVk3qRn-gIGyLmtWMnQ


p.1 (facing) -- lovely [Phaedrus] quote

p.2 -- reminds me of design rationale

p.4 -- the designer standing alone in
history

p.6 -- "how to represent a design
problem"
```

Figure 12: *BookNotes entered by user*

Note how this is designed to mimic as closely as possible the original hand-written notes. Clearly to make this a useful application for other users, modification would be necessary as it is closely tuned to the original academic's style. However, this process of generalisation is not discussed further here as the critical point of this mini-application is to act as a sample point for the API.

Figures 13 and 14 show the formatted book notes. In particular, note the infobox populated with bibliographic data and links to reading lists in Aspire that include this book.

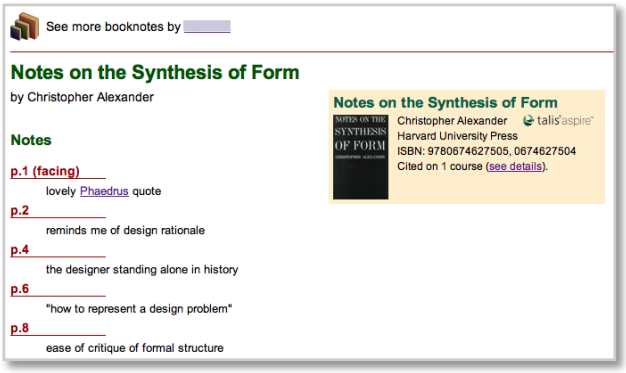

Figure 13: *BookNotes application*

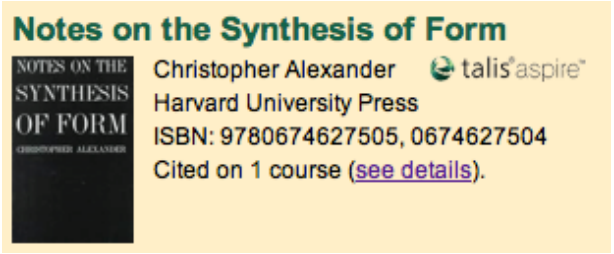

Figure 14: *Close-up of infobox from Figure 10*

This infobox is generated from the same kind of HTML mark-up and JavaScript API as used in the academic's own hand-edited HTML pages. Figure 15 shows the code in the HTML page that the application generates directly from the three header lines in Figure 12.

```
<div class="acorn-infobox"
     aspire-citeType="about"
     aspire-options="title-inline;">
  <h2><a
href="http://community.talisaspire.com/
resources/pqKNVk3qRn-gIGyLmtWMnQ"
class="acorn-title">
    Notes on the Synthesis of Form
</a></h2>
  <p class="aspire-authors">
    Christopher Alexander
  </p>
</div>
```

Figure 15: *HTML for bibliograohic*

In addition the link to Plato's "Phaedrus" on page 1 has become a live link with pop-up bibliographic details. Figure 16 shows the pop-up and Figure 17, the corresponding mark-up that the application needed to generate.

Figure 16: *Pop-up bibliographic information*

```
<p>
lovely                                  <a
href="http://community.talisaspire.com/
resources/v0YUd2bd-hUhuLc-cX17fg"
title="Phaedrus                  (Plato)"
class="bn_link">Phaedrus</a> quote
</p>
```

Figure 17: *Mark-up for pop-up in Figure 16*

Both infobox and popup are provided by the same JavaScript API as used in the author's hand edited pages, except that the BookNotes application also includes some meta information. Some of this is on per-entry basis. In particular, note the 'aspire-citeType' in Figure 15. This tells the API that "Notes on the Synthesis of Form " is the main reference for this page as opposed to "Phaedrus", which is merely a citation. In addition, global meta-data in the page header identifies the application itself and, if known, the Aspire id of the author of

the notes (Figure 18). The latter allows Aspire to create links back to the BookNotes application from the owner's home page in Aspire.

```
<script>
Aspire.setOptions({"appId":"booknotes",
"canonicalURL":"http://anonymised.url/"
,"pageOwner":"anon@anon.ac.uk"});
</script>
```

Figure 18: *BookNotes meta-information for Aspire API*

If desired, the application can access a JSON API directly to obtain the HTML for the infobox when the page is created. However, for the initial implementation this was left to be filled in dynamically by the JavaScript API. This simplified the implementation, allowing the creation of a complete, but highly functional mini-app in a couple of days, which is already being used by the academic.

4. DISCUSSION

In both of the above examples the academic is benefiting from richly formatted bibliographic details with minimal effort, including pop-ups where desired. Where full-text is available for an article or book (for example through Google Books), the page author/application can also choose to have a preview button included, which opens a light-box-style viewer without leaving the page. Furthermore, if the viewer is an Aspire user, they can be offered the option of bookmarking within Aspire.

However, the main benefits for Aspire and the additional benefits for the API user are happening behind the scenes. Simply by servicing the JavaScript API requests Aspire learns that a particular page is citing a learning resource. If the page owner is registered as an academic with Aspire then it also knows that it is an authoritative resource. Similarly, the meta-information that BookNotes includes allows Aspire to know that this is not just a page citing "Notes on the Synthesis of Form", but a page where this is the primary topic.

This information is essentially growing the education graph. The academic's hand-edited page acts very like a course reading list created within Aspire Campus Edition, and the BookNotes page becomes both a place where resources (such as Phaedrus) have been cited and also a page that is known to be 'about' the main book.

This information can then be reflected back in the main Aspire Community Edition web interface. We have already noted how the meta-information passed by the BookNotes application to the API can include the id of the page owner, allowing Aspire to include links back to the BookNotes from the owner's Aspire home page. In addition this adds provenance and, if the owner is a verified academic, or maybe the author of a resource, it can be used as a measure of the authority of the information. Similarly, for the hand-edited pages, the page owner can link these to their Aspire account, so that it becomes regarded as part of their authenticated resources.

Issues of reputation, authority and provenance are of course central in an academic setting. This is not to underestimate the potential of student-generated and other content, which offer complementary perspectives. For example, we may trust student-generated content more as an assessment of the readability and clarity of a text book, but academic content more for an assessment of accuracy, reliability and analytic depth.

This provenance information could therefore also allow future versions of Aspire to place these third-party resources alongside those obtained from institutional reading lists. Figure 19 shows a mock-up page generated from the API proxy (Figure 4). The page is identical to the normal Aspire Campus Edition page for the book[19], including a preview, and a list of courses using the book. However, in addition to the course that includes the book, it also lists the BookNotes referring to it.

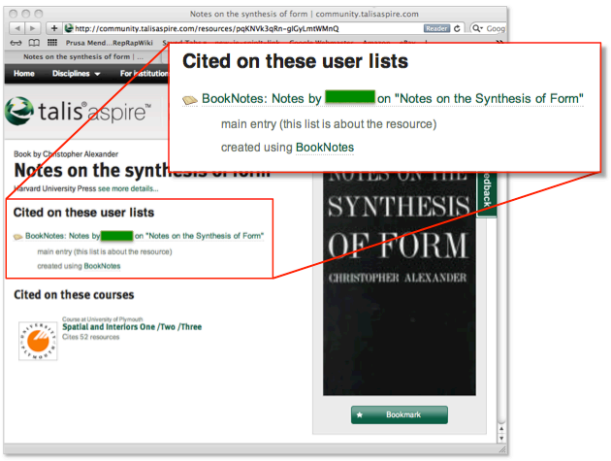

Figure 19: *Feeding back into the education graph (mock-up)*

In summary, the Aspire education graph grows through API use, giving value to the API provider, and the application and page authors gain value through being highlighted whenever students view the resources they cite, or for which they create additional information.

19 http://community.talisaspire.com/resources/pqKNVk3qRn-gIGyLmtWMnQ

5. CONCLUSIONS

This paper has looked at issues around the design of one facet of the API for the Talis Aspire family of education services. This has involved theoretical understanding of mutual value between the stakeholders, and also a practice-focused approach to developing use-driven API services. While the paper considers a particular case study, we hope the theoretical and methodological approaches are of more general applicability. Furthermore, the paper highlights the way HCI issues, technical design and market considerations are not siloed concerns, but interact closely in the design of rich APIs that provide value and ease of development to developers and enhanced user experience for end-users.

6. REFERENCES

Alexander, C. (1964). *Notes on the Synthesis of Form*. Harvard University Press

Benkler, Y. (2006) *The Wealth of Networks: How Social Production Transforms Markets and Freedom*. Yale University Press.

Bizer, C., Heath, T. and Berners-Lee, T. (2009). Linked data – the story so far. Int. J. Semantic Web Inf. Syst., 5(3):1–22, 2009.

Bloom, B., Englehart, M. Furst, E., Hill, W., & Krathwohl, D. (1956). Taxonomy of educational objectives: The classification of educational goals. Handbook I: Cognitive domain. Longmans, Green.

Briscoe, B., Odlyzko, A., and Tilly, B. (2006) Metcalfe's Law is Wrong. *IEEE Spectrum*, July 2006.

Checkland. P. (1981) *Systems Thinking*, Systems Practice. John Wiley, Chichester,

Clarke, C. (2009). A resource list management tool for undergraduate students based on linked open data principles. *Proc. of the 6th European Semantic Web Conference*, Heraklion, Greece, 2009.

Dix, A (2001). The Lattice of Value – Designing Products for Self-Growth. *eBulletin*. Nov. 2001. http://www.hiraeth.com/alan/ebulletin/lattice-of-value/

Dix, A., Finlay, J., Abowd, G., and Beale, R. (2004) *Human Computer Interaction*, Prentice Hall. (see also http://www.hcibook.com/e3/casestudy/search/)

Dix, A. (2008). Theoretical analysis and theory creation, Chapter 9 in *Research Methods for Human-Computer Interaction*, P. Cairns and A. Cox (eds). Cambridge University Press, pp.175–195.

Dix, A. (2010) Human-Computer Interaction: a stable discipline, a nascent science, and the growth of the long tail. *Interacting with Comp.*, 22(1):13-27.

Dix, A. (2011). The Real Tragedy of the Commons. 6th March 2011. http://alandix.com/blog/2011/03/06/the-real-tragedy-of-the-commons/

Dix, A., Beale, R., Shabir, N. and Leavesley, J. (2011). Anatomy of an Early Social Networking Site. *Proc. of HCI 2011*, BCS eWics.

Economides, N. (1996). The economics of networks. *International Journal of Industrial Organization*. 14(6):673–699

Evans, B., Kairam, S., and Pirolli. P. (2010). Do your friends make you smarter?: An analysis of social strategies in online information seeking. *Inf. Process. Manage.* 46, 6 (November 2010), 679-692. DOI=10.1016/j.ipm.2009.12.001

Facebook (2012). Company Timeline http://newsroom.fb.com/content/default.aspx?NewsAreaId=20 (retrieved 28/3/2012)

Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., and Berners-Lee, T.(1999). Section 12 Content Negotiation. *Hypertext Transfer Protocol -- HTTP/1.1*. RFC 2616, W3C http://www.w3.org/Protocols/rfc2616/rfc2616-sec12.html#sec12

Hardin, G. (1968). The Tragedy of the Commons, *Science*, 162(1968):1243-1248.

Heath, T., Singer, R., Shabir, N., Clarke, C. and Leavesley, J. (2012) "Assembling and Applying an Education Graph based on Learning Resources in Universities". *Proc. 2nd Intnl Workshop on Learning and Education with the Web of Data (LiLe2012)*, WWW2012, Lyon, France.

Hendler, J. and Golbeck, J. (2008). Metcalfe's law, Web 2.0, and the Semantic Web. *Web Semant.* 6(1):14-20. DOI=10.1016/j.websem.2007.11.008

Liebowitz, S. and Margolis, S. (1998) Network Externalities (Effects). Entry in *The New Palgraves Dictionary of Economics and the Law*, MacMillan.

Palen, L. (1999). Social, individual and technological issues for groupware calendar systems. *Proc. CHI '99*. ACM, 17-24. DOI=10.1145/302979.302982

Razak, F. (2008). *Single Person Study: Methodological Issues*. PhD Thesis. Lancaster University, UK. February 2008.

Shapiro, C. and Varian H. (1999). *Information rules: a strategic guide to the network economy*. Harvard Business Press,

Tierney, J. (2009) The Non-Tragedy of the Commons. *New York Times*, October 15, 2009,

Twidale, M.B. and Ruhleder, K. (2004). Over-the-Shoulder Learning in a Distance Education Environment. In C. Haythornthwaite & M.M. Kazmer (Eds.) *Learning, Culture and Community in Online Education: Research and Practice*. NY: Peter Lang. 177-194.

Shneiderman, B. (2011). Claiming Success, Charting the Future: Micro-HCI and Macro-HCI. *Interactions*, September + October 2011, 10–11.

Varnelis, K. (ed.) (2008). *Networked Publics*. MIT Press.