Creating a "no cost" automated database tool for managing ILL requests

John Prentice | Library Manager, Australian and New Zealand College of Anaesthetists
jprentice@anzca.edu.au | orcid.org/0000-0002-3268-2099
Prior to joining the college, John was an implementation specialist at OCLC.

The project described below was the recipient of the HLA / Medical Director Digital Health Innovation Award for 2022. More information about the award is here.

Overview

Winner of the 2022 *HLA/MedicalDirector Digital Health Innovation Award*, ANZCA's ILLs MANAGER database was developed as a way of processing an increasing number of document delivery requests whilst at the same time automating many of the associated manual processes.

Background

In 2018, ANZCA undertook a project to replace its aging library management system (LMS). The project required us to not only update all our existing services but to completely re-catalogue and re-receive our print and journal collections due to the poor quality of the collection metadata in the preceding system. In order to properly tackle such a mammoth task, we began to look at ways we could free up significant amounts of staff time. Upon reviewing our existing set-up, the most obvious place to make time-savings was in the area of document delivery. At that point in time, the library was doing some 1400-1500 article requests per year, which accounted for some 50-60% of library staff resourcing. It was felt that if the processes associated with our document delivery service could be streamlined and/or partially automated in some way then this would free up the necessary staff time for both this and future projects.

Identifying the problem areas

A review of our existing document delivery processes showed a number of problematic areas:

1. Receiving requests: Even though we had recently instituted a document delivery request form, it was rarely being utilised by our users who found it easier to just copy and paste a citation into an email sent to the library, send a word document with a large number of article citations, or simply take a screenshot off their phone and send that via email. Due to copyright issues, this often involved us having to email back a copyright indemnification form before we could process the request. This process was resulting in clear delays and involved a lot of manual retyping during the search and request process.

2. Tracking requests: Once received, the communication process with our users was being handled entirely from within our library email inbox. This made tracking requests both time-consuming and complicated, especially where multiple requests were concerned. This often led to staff having to print up requests and keep the request sheets by their desk so they could manually annotate requests.
3. Searching for articles: Staff were having to copy and paste article title details for all searching being done as part of the requesting process. Not only was this time-consuming but with such a large volume of requests there were clear OH&S implications associated with all the copying, pasting, retyping and clicking involved as we searched for full-text across multiple systems.
4. Requesting articles: As with the searching above, there were similar issues with the copying, pasting and retyping of all the citation elements when completing request forms – predominantly in *GratisNet* and *LADD* (although we regularly request from several other sources as well).
5. Supplying articles: And finally, when it came to supplying articles, staff would have to hunt through a folder of requests to locate the correct email and then respond back to the requestor with the article attached.

In addition to the above, any solution that we potentially devised had to be created using our existing resources. There was simply no additional funding available to subscribe to any of the existing document delivery management tools on the market. Moreover, we were loath to use any service that would require integration into our existing systems in order to function effectively. We wanted a solution that could be used as a standalone, had portability and wouldn't make us any more dependent on IT than we already were.

Changing the way we receive requests

One "simple" solution to the request email situation, was to gradually remove the ability for our users to submit article requests by any method other than via a request form. Our direct email address was removed from the ILL request page and users who persisted in emailing us direct were politely yet consistently encouraged to use our *LibWizard*-based form. They were told that it was quicker for us to process requests when they arrived via the form as it ensured that we had all the necessary citation elements and was copyright compliant. At the same time, we updated the form so that it provided discrete citation elements for each request and also labelled fields in such a way that the form could also be used for book chapter requests. These updates would then be taken advantage of by our resulting database tool.

We then began the longer-term task of embedding the *LibWizard* form into our new discovery service, as well as our *BrowZine* service – allowing for one-click population of the request form. Once the former was in place, we ceased to accept any requests via standard email. Users were informed that due to copyright reasons we were no longer accepting requests via email and sent a permalink to the article citation in our

discovery service where they could then simply click a request button to populate the request form. By consistent application of this approach, we were able to encourage our users onto our new discovery service – where they could now check for full-text access prior to submitting any request – as well as ensuring we had a clearly-marked request email which could then be shepherded through our library mailbox through the application of *MS Outlook* rules.

Within a relatively short period of time, we were able to change the behaviour of our users, which was to have a big impact on how quickly we could then process their requests.

Automating the rest

The much bigger task was addressing the issue of tracking / searching / requesting / supplying of the actual requests.

For some time, we had been using an *MS Access* database designed by Van Duong and Barbara Slattery – whilst both were at Royal Melbourne Hospital – for tracking purposes. Citations were copied and pasted into the database field-by-field (another time-consuming task) with us then attempting to use this database as our "source of truth" whilst managing the rest of the request. The database had some nifty features: namely linked journals, supplying library and patron tables and the ability to quickly generate usage stats. However, the database also had some very clear limitations:

1. There was no easy way of getting requests into the database (everything had to be copied field-by-field for every single request)
2. There was no way of properly reviewing outstanding requests (the user typically clicked through the request detail screen till they found the correct record they were after)
3. Searching and requesting was still done by copying and pasting the article title into the relevant systems
4. Communication and supply were still done via our library email inbox
5. It lacked portability and usability as the database was split into 2 pieces (which often caused confusion for staff during operation)

On the positive side, use of the database did provide us with the standalone system we were seeking, and *MS Access* was still being – albeit somewhat reluctantly – supported by our IT department. Note: Plans previously announced by Microsoft to "retire" *MS Access* were quietly abandoned once it became clear that *MS Access* filled an important gap in the DBMS market. As of the time of writing, *MS Access* is still fully supported by Microsoft and there are no plans to retire *MS Access* in the foreseeable future.

As I have a background in computer programming (including a familiarity with *Visual Basic*) and knew *MS Access* relatively well, I was able to start redeveloping the core

RMH *MS Access* database functionality to add a suite of new features. Initially this included the following:

1. A switchboard that displayed a list of all outstanding requests and allowed for quick access to any request by simply double-clicking the Request ID
2. The ability to easily import pre-formatted email requests
3. The ability to undertake one-click search operations of our various full-text resources
4. The ability to initiate one-click *GratisNet* and *LADD* requests direct from the database
5. The ability to message the requestor direct from the database (and to supply the full-text)
6. Automated updates to systems checked, status and automated requesting notes
7. Integration of the multi-part database into a single unit
8. Full patron dump from our Admin system and the importing of patron names, IDs, and permanent email addresses into the database

Due to the fact that *MS Access* is part of the *MS Office* suite of products and that *MS Office* is in almost universal use, we were able to take advantage of *MS Access*'s inbuilt tools and reference, thus allowing the database to create and send pre-formatted emails using the operators own email system but from within the database itself. Requests could be quickly acknowledged and all correspondence with the requestor could be managed from the database, and details of that correspondence recorded via an extensive Notes field.

This ability to message users direct from within the database was integral to our being able to finally divorce our requesting from our library inbox. Once ingested into the database, the original request emails could simply be "filed away" and need not be referred to again throughout the rest of requesting process. This provided critical time savings, ensured we truly had a centralised "system of truth" and made tracking our requests relatively easy. Staff were then able to focus on actually sourcing requests, rather than the mechanics of managing the requests.

Similarly, the ability to search across an array of web-based services and databases using "one-click" operations effectively eliminated all the copying and pasting that had occurred previously and significantly sped up the process of searching (and requesting).

Whilst these enhancements freed up enough staff time to move forward with our new library systems implementation, there were many short-comings we wanted to address – not the least of which was the database's inability to effectively handle diacritics. Thus began an extended period of additional development which led to the following features being added:

1. An updated switchboard that now included a summary of outstanding book chapter/book loan requests and provided additional request details (like urgency)
2. Import functionality for structured request emails from *Ovid* databases (*Medline* and *Embase*) and quasi-formatted request emails from services such as *Read by QxMD*
3. An extended suite of email templates to cover all possible communication scenarios with the requestor
4. Additional requesting and supply options for *Subito* and NLM *Relais*
5. Standard handling of diacritics (which often prevented searching and form population due to their use in names, and which meant that for a long period this information was still being copied and pasted into the request screens for *GratisNet* and *LADD*)
6. The ability to do a "one-click" *Google* search
7. The addition of *PubMed* and *Google Books* API lookup features that allowed for the repopulation of incomplete/inaccurate article and book citations
8. A full ingest of the NLM journal catalogue which was integrated with the *GratisNet* journal listing to provide an exhaustive list of journals
9. "One-click" DOI and *LibKey* lookups for potential full-text supply
10. The ability to send an URL for supply purposes when an article was available from an accessible subscription database (as in the "one-click" DOI and *LibKey* examples above)
11. Updates to the statistics reporting and display options

Further changes were instituted following the receiving of the 2022 *HLA/MedicalDirector Digital Health Innovation Award*. In addition to the API lookup and diacritics-handling features mentioned above, the back-end of the database was extensively reworked to remove many of the "hard-wired" ANZCA-specific settings and to make these configurable options via an extensive settings menu. Where previously the database was locked to *GratisNet* and *LADD*, the database can now be configured for any document delivery and ILL service allowing for OpenURL population of requests. A full user guide was also written, and a micro site created from which the database and guide could be downloaded. This work was all done with a view to making the ILLs database freely available to other health libraries.

Regrettably, one issue that we have as yet been unable to successfully resolve is the correct population of *GratisNet* requests. As the *GratisNet* form is not OpenURL compliant, we have only been able to achieve partial population of the form during requesting. Currently, there is a workaround in place that transfers all the "missing" elements into the DOI field where the user can then drag-and-drop them into the correct request form fields. Whilst this isn't ideal, it provides a viable workaround whilst we work on potential solutions at the vendor end.

The impacts

The impacts on our turnaround times were immediate and resulted in long-term time-savings for both our users and for staff.

The average turnaround time for articles has dropped from 2-3 days in 2018 to under 1 day for the majority of 2022. This means that the bulk of our supply is essentially "same day". In fact, since implementation, the major factor affecting article turnaround times has been the staff training necessary when we replace team members. As a result, we have received consistent praise from our users as regards our turnaround times, and the service has become a "lifeline" for our users after we lost a major journal at the beginning of 2022. In addition, the statistics functionality embedded in the database allows us to not only quickly report on volume and turnaround times but our "top 10" most requested journals. This can then be fed into collection management decisions.

More crucially, the actual time spent by staff on undertaking requesting-related operations has dropped dramatically. On average, staff now spend 1-2 hours per day undertaking the same volume of requests that used to take them 4-5 hours. This has had a significant impact on the types of duties being undertaken by library staff. By reducing the time spent on repetitive tasks and making more efficient use of their time, staff have been freed up to the point that they are now able to participate in a range of projects and endeavours. We have been able to address cataloguing and donations backlogs, now have more time for training, and staff are enjoying a much more varied work experience in general.

Closing thoughts

Whilst some may see *MS Access* as somewhat dated technology, it has the virtue of being essentially standalone and functionally rich. This has allowed us to create a durable solution that has significantly sped up document delivery turnaround times and dramatically increased operational efficiency, and it has done so without any additional outlay in subscription costs and IT resourcing. These latter points were significant considerations for us, as we felt that creating a "no-cost" easy-to-use solution would also be attractive to other health libraries such as ourselves with small staffs and limited resources.

Download your own copy

A free copy of the ILLs database and accompanying user guide can be downloaded from the ANZCA library website: https://libguides.anzca.edu.au/illsdatabase. The user guide covers both configuration and usage.

ANZCA Library Manager, John Prentice, during a live demonstration of the ILLs database at the 2022 HLA Conference in Westmead.