

Designing the Structured Search Experience: Rethinking the Query-Builder Paradigm

Tony Russell-Rose

GOLDSMITHS, UNIVERSITY OF LONDON

Farhad Shokraneh

UNIVERSITY OF NOTTINGHAM

Volume 3, Issue 1, 2020

DOI: <http://dx.doi.org/10.3998/weave.12535642.0003.102> [<http://dx.doi.org/10.3998/weave.12535642.0003.102>]

 [<http://creativecommons.org/licenses/by/4.0/>]



[[javascript:window.print\(\)](javascript:window.print())]

 [https://www.altmetric.com/details.php?domain=quod.lib.umich.edu&citation_id=79765025]

Abstract

Knowledge workers such as healthcare information professionals, legal researchers, and librarians need to create and execute search strategies that are comprehensive, transparent, and reproducible. The traditional solution is to use command-line query builders offered by proprietary database vendors. However, these are based on a paradigm that dates from the days when users could access databases only via text-based terminals and command-line syntax. In this paper, we present a new approach in which users express concepts as objects on a visual canvas and manipulate them to articulate relationships. This offers a more intuitive user experience (UX) that eliminates many sources of error, makes the query semantics more transparent, and offers new ways to collaborate, share, and optimize search strategies and best practices.

1. Introduction

According to a landmark McKinsey report, knowledge workers spend as much as 19% of their working week searching for and gathering information (Chui et al., 2012). Whether they eventually find what they are looking for or stop and make a suboptimal decision, either outcome can come at a high cost. Healthcare information professionals, for example, perform painstaking and meticulous searching of literature sources as the foundation of evidence-based medicine. However, systematic literature reviews can take years to complete (Lefebvre et al., 2008), and new research

findings may be published in the interim, leading to a lack of currency and potential for inaccuracy (Shojania et al., 2007).

Likewise, patent agents rely on accurate prior-art search as the foundation of their due diligence process, yet frequent infringement suits result from the later discovery of prior art which the original search missed (Gibbs, 2006). And media-monitoring organizations routinely manage thousands of Boolean expressions consisting of hundreds of search terms, leading to significant challenges in maintenance, editing, and debugging (Pazer, 2013).

What these professions have in common is a need to develop search strategies that are comprehensive, transparent, and reproducible (Russell-Rose et al., 2019). The traditional solution to this problem is to use command-line query builders such as that in Figure 1.

PubMed Advanced Search Builder

YouTube Tutorial

Use the builder below to create your search

[Edit](#) [Clear](#)

Builder

All Fields [Show index list](#)

AND All Fields [Show index list](#)

Search or [Add to history](#)

History [Download history](#) [Clear history](#)

Search	Add to builder	Query	Items found	Time
#7	Add	Search (#2) AND #6	15621	09:34:31
#6	Add	Search (bcc) OR basal cell carcinoma	29447	09:34:06
#3	Add	Search scc	18924	09:33:35
#2	Add	Search (cancer) AND skin	196105	09:33:13

[/w/weave/images/12535642.0003.102-00000001.png]

Figure 1.

A traditional form-based query builder (PubMed).

This approach requires the user to build their query by entering keywords, operators, and ontology terms on a line-by-line basis and combine them by constructing Boolean expressions that reference the appropriate line numbers from their search history. The output consists of an interlinked sequence of Boolean strings which in combination express the composite semantics of their information need (fig. 2). These search strategies can extend to many lines and multiple pages, and they often represent the combined efforts of many individuals.

```

1 "Aspergillus"[MeSH]
2 "Aspergillosis"[MeSH]
3 "Pulmonary Aspergillosis"[MeSH]
4 aspergill*[tiab]
5 fungal infection[tw]
6 (invasive[tiab] AND fungal[tiab])
7 1 OR 2 OR 3 OR 4 OR 5 OR 6
8 "Serology"[MeSH]
9 Serology"[MeSH]
10 (serology[tiab] OR serodiagnosis[tiab] OR serologic[tiab])
11 8 OR 9 OR 10
12 "Immunoassay"[MeSH]
13 (immunoassay[tiab] OR immunoassays[tiab])
14 (immuno assay[tiab] OR immuno assays[tiab])
15 (ELISA[tiab] OR ELISAs[tiab] OR EIA[tiab] OR EIAs[tiab])
16 immunosorbent[tiab]
17 12 OR 13 OR 14 OR 15 OR 16
18 Platelina[tw]
19 "Mannans"[MeSH]
20 galactomannan[tw]
21 18 OR 19 OR 20
22 11 OR 17 OR 21
23 7 AND 22

```

[\[/w/weave/images/12535642.0003.102-00000002.png\]](#)

Figure 2.

An example Boolean search strategy (Leeflang et al., 2015).

The use of query builders such as that of Figure 1 is ubiquitous within the library and information science community and has served as the primary means for expressing structured searches since the development of the first online databases. However, from a UX perspective, the approach suffers from a number of fundamental shortcomings. In this paper, we examine those shortcomings, review their implications from a design perspective, and offer a new approach based on an alternative paradigm.

In Section 2, we discuss the weaknesses of the current paradigm and review their consequences for structured searching. In Section 3, we describe alternative approaches and key design insights and principles. In Section 4, we propose a new approach which builds on related work and extends it to address a number of challenges. We conclude in Section 5 with a summary and opportunities for further work.

2. The UX of Structured Searching

2.1 Boolean Strings

The use of command-line query builders currently serves as the de facto paradigm for all manner of advanced search experiences. However, despite their ubiquity, the practice of using Boolean strings to express complex information needs suffers from a number of shortcomings.

First, Boolean strings are an ineffective vehicle for communicating structure. The use of parentheses as delimiters may be commonplace in programming languages and other machine readable media, but when intended for human interpretation they rely on additional physical cues such as indentation. In the absence of such visual signals, parentheses can become lost in an undifferentiated sequence of alphanumeric characters, and trying to interpret the meaning and structure of such expressions can be overwhelming (Russell-Rose & Chamberlain, 2016).

Second, Boolean strings do not scale well. As users add terms to a Boolean string, it grows uniformly in length. This may be manageable for a handful of terms, but as the number grows to double figures and beyond, transparency degrades. A common solution to the analogous problem in software development is to offer some form of abstraction, so that lower-level details become progressively hidden, revealing higher-level structure. But Boolean strings in their native textual form offer no such facility.

Third, Boolean strings are very error-prone: even if the query builder automatically checks syntax, it is still possible to place parentheses incorrectly, which can inadvertently change the semantics of the whole expression. This gives rise to errors whose effects may not become apparent until long after the initial search is completed.

To mitigate these issues, many professionals rely on previous examples of best practice. Healthcare information professionals, for example, draw on repositories such as MedTerm Assist (Saleh et al., 2014), Search Strategy Library (Gulhane, 2019), or other sources of ready-made search blocks (de Jonge & Lein, 2015). These resources provide an archive of search strategies that users can reuse via copy and paste. However, the library and information science community has not universally adopted them, and the volume of content in such collections remains relatively modest.

Many information professionals use word-processing tools such as Google Docs or Microsoft Word as a platform for developing search strategies. However, there are many reasons document-centric tools such as these are not suitable for search-syntax development:

- Auto-correction can undermine truncation and corrupt truncated formats.
- Spell checking can obfuscate important differences between British and American English and can create unwanted duplicates.
- Auto-conversion of straight quotations (""") into curly quotations (""") can cause errors in some platforms such as Ovid SP.
- Copying and pasting text fragments between different word-processing tools can lead to loss of non-print characters.

In summary, the practice of manipulating search strategies as text strings compromises their ability to function as transparent, reproducible, executable artifacts.

2.2 Command-Line Query Builders

As mentioned above, the command-line query builder has served as the default for structured searching since the first online databases. However, errors and inefficiencies often compromise their output. In their study of sixty-three MEDLINE strategies, Sampson and McGowan detected at

least one error in over 90 percent of these, including spelling errors, truncation errors, logical operator error, incorrect query line references, and redundancy without rationale (2006).

In command-line query builders, users build search strategies incrementally as a set of discrete expressions they reference by line number and combine using various operators. This type of approach offers the benefit that strategies can be built using techniques such as successive fractions or building blocks (Booth, 2008). It also allows the searcher to review the number of results returned at each step and to refine their expressions accordingly.

However, it is questionable whether line numbers are the most effective way to organize a series of interconnected logical expressions. If the goal is to provide a principled mechanism for structured searching, it seems odd to expect the user to rely on something as arbitrary as a line number. This is the conceptual equivalent of the GOTO statement used in first generation BASIC, and this approach was discredited many decades ago (Dijkstra, 1968).

We speculate on how much support is offered for constructing expressions that are syntactically correct and semantically transparent. A well-designed query language could, for example, support concepts such as:

- **Encapsulation:** whereby a single component contains data and functions. Arguably, the command-line approach supports this, but a lack of facility for naming reusable elements compromises it.
- **Abstraction:** whereby a collection of lower-level instances generates a higher-level concept. For example, in Figure 2, lines one to seven may be interpreted as the ‘problem’ element of a PICO search (Shokraneh, 2016). However, it is difficult to manipulate this sub-element as a discrete component, thus compromising its effective reuse.

There is a further issue with the command-line paradigm. As Figure 2 illustrates, the output of such query builders is a set of procedural commands that are combined to express the declarative semantics of which documents to retrieve. This raises the question: if the goal is to express the declarative semantics of an information need, then why force the user to think procedurally? One explanation may be that the command-line approach reflects the days when searchers sent text-based commands to remote—and expensive—subscription-based resources. In this context, the continued adoption of command-line query builders reflects a historical requirement rather than any enduring quality of the search experience.

3. Related Work

In this section, we review a number of original and influential alternative approaches to search-query formulation and discuss their contribution to contemporary UX concepts and principles.

The application of data visualization to search-query formulation can offer significant benefits, such as fewer zero-hit queries, improved query comprehension, and better support for exploration of an unfamiliar database (Goldberg & Gajendar, 2008). Anick et al. (1989) is an early example of such an approach. They developed a system that could parse natural language queries and represent them as movable tiles on a two-dimensional canvas. The user could rearrange the tiles to reformulate the expression and to activate or deactivate alternative elements to modify the query.

In addition, the system offered support for integration with thesauri, and it displayed the number of hits in the lower left corner of each tile.

In subsequent work, Fishkin and Stone (1995) investigated applying direct manipulation techniques to database query formulation using a system of “lenses” to refine and filter the data. Users could combine lenses by stacking them and applying a suitable operator or combine them to create compound lenses, supporting the encapsulation of complex queries. Jones (1998) proposed an influential approach in which concepts are expressed using a Venn diagram notation combined with integrated query result previews. Users could formulate queries by overlapping objects within the workspace to create intersections and disjunctions, and they could select subsets to achieve a further refined set of results.

Yi et al. developed a system based around a “dust and magnet” (2005, p 239) metaphor, in which users could represent dimensions of interest within the data as magnets on a visual canvas. The effect of the “magnetic forces” on individual “data particles” reflected the relationships between points in the data. More recently, Nitsche and Nürnberger (2013) developed a system based around a radial interface in which users could integrate and manipulate queries and results. The concept used a pseudo-desktop metaphor in which objects of interest clustered toward the center. Query objects could be entered directly onto this canvas, and their proximity to the center and to other objects was a relevance cue, influencing the selection and position of search results.

In our work, we adopt and extend many of the design principles and insights embodied in this work:

- Users can formulate and manipulate Boolean expressions as objects on a canvas.
- Users can include or exclude individual query elements.
- Users can create complex queries via nested aggregate structures.
- Interaction and animation communicate meaning and structure.
- Real-time feedback is fundamental to effective query optimization.

4. An Alternative Approach

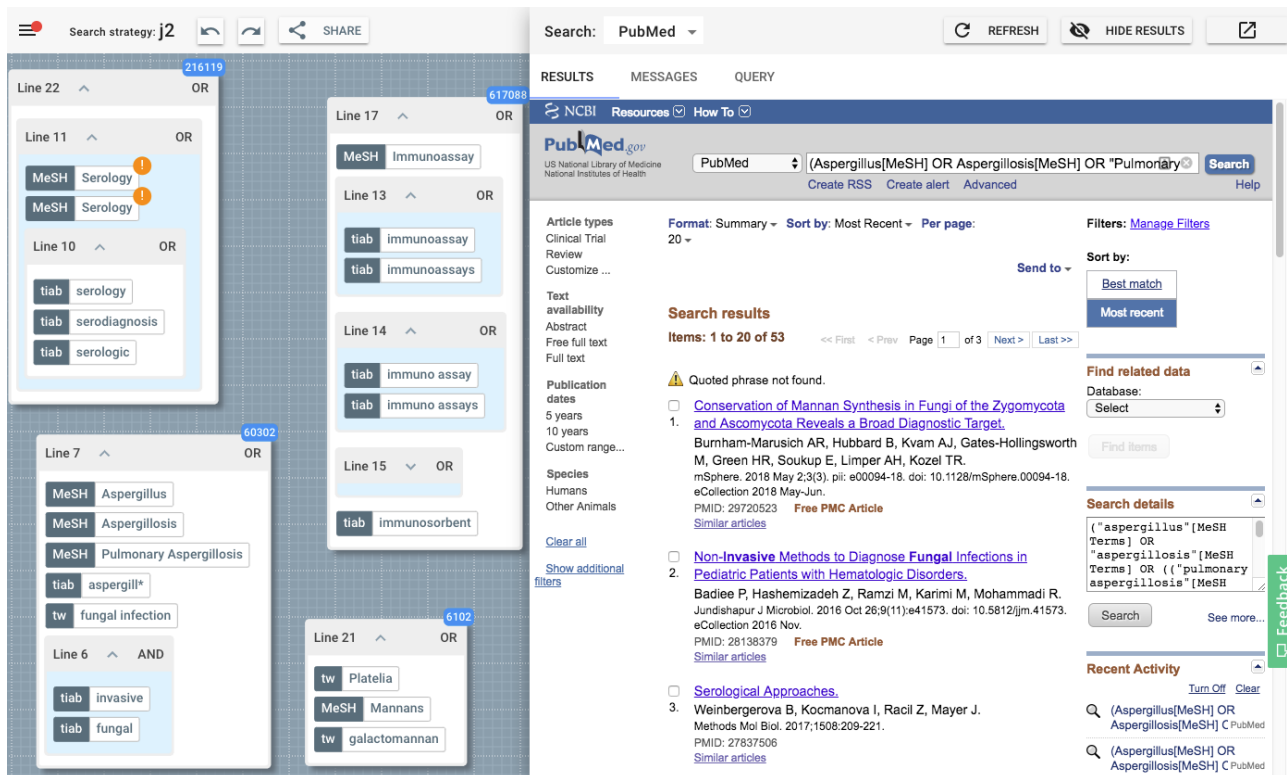
We propose an alternative solution to the problem of search-query formulation. Our approach, which we call 2Dsearch (<https://www.2dsearch.com> [<https://www.2dsearch.com>]), is the product of a research and development program combining novel techniques from the fields of data visualization, artificial intelligence, and natural language processing. Innovate UK awarded us multiple grants for research and development, and through this work we have been able to launch the tool as a free web application. Our team is small (“Team,” n.d.), and we rely on the enthusiasm of key individuals to develop and maintain the platform. We remain committed to working closely with the scientific community and publishing as much of our work as possible in open-access scholarly publications.

The name “2Dsearch” reflects a fundamental change of approach. Instead of a one-dimensional search box, concepts display as objects on a two-dimensional canvas. Users manipulate these objects using drag-and-drop in order to express relationships between them. This visual approach has the potential to eliminate many sources of syntactic error, helps to make the query semantics

transparent, and offers an open-access platform for sharing reproducible search strategies and best practices.

4.1 Building Search Strategies

At the heart of 2Dsearch is a graphical editor which allows the user to formulate search strategies using a visual framework. Concepts can be simple keywords or attribute:value pairs representing controlled vocabulary terms (e.g. MeSH terms) or database-specific search operators (e.g. field codes and other commands). Users can combine them using Boolean—and other—operators to form higher-level groups and then iteratively nest them to create complex expressions.



<http://w/weave/images/12535642.0003.102-00000003.png>

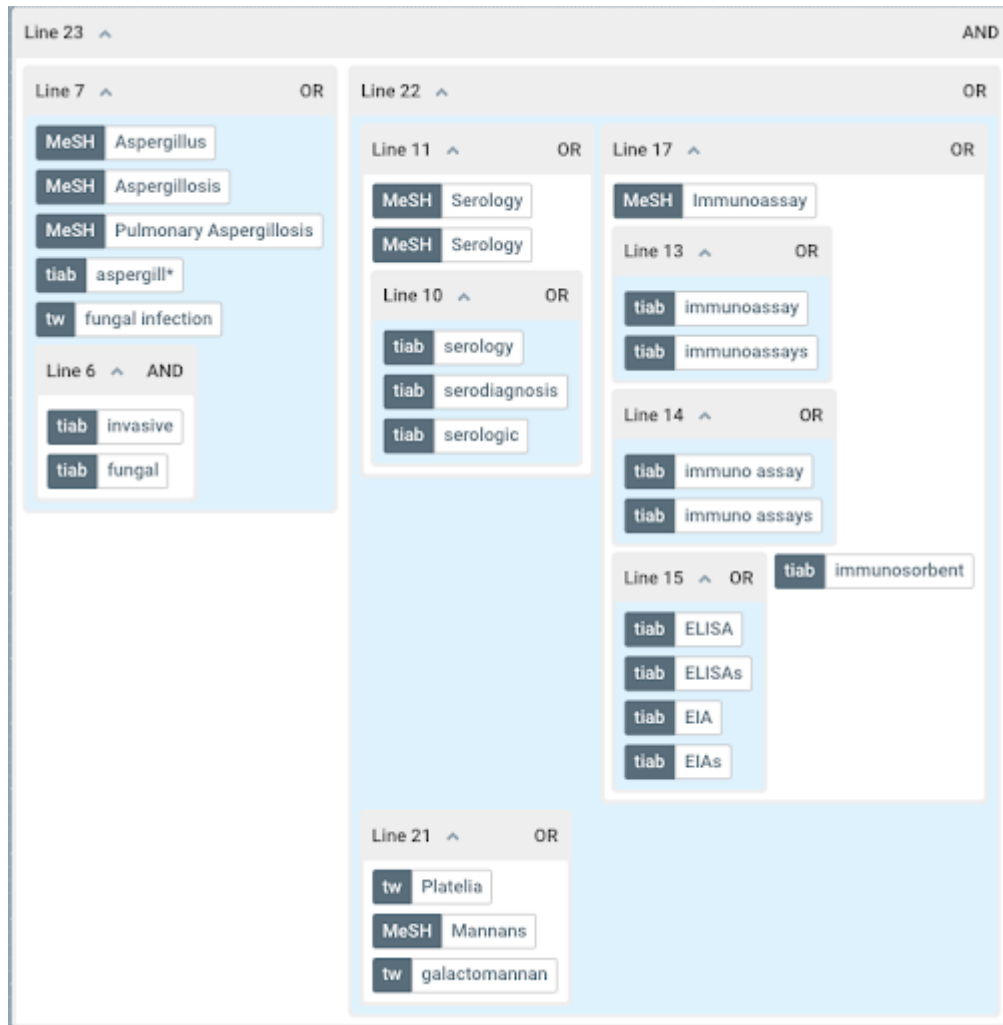
Figure 3.

The 2Dsearch application showing the query canvas (left) and the results pane (right).

The application consists of two panes (fig. 3): a query canvas on the left and a search results pane on the right, both of which users can resize; they may also detach the results in a separate window. The canvas includes an overview widget which allows the user to navigate to elements outside the current viewport. Adopting design cues from Google's Material Design (<https://material.io> [<https://material.io>]) language, a sliding menu on the left provides file import, file output, and other options. Complementing this is a navigation bar which provides support for document-level functions such as naming and sharing search strategies.

Although 2Dsearch supports the creation of complete strategies from a blank canvas, its function and value are most readily understood by reference to a sample text-based search strategy, such as that in Figure 2. A trained professional may be able to mentally parse the sequence of commands

and interpret the general approach, but without associated documentation it is difficult to understand exactly what the searcher intended, let alone modify, debug, or reuse this strategy.



[\[/w/weave/images/12535642.0003.102-00000004.png\]](http://w/weave/images/12535642.0003.102-00000004.png)

Figure 4.

Visualizing a text-based search strategy.

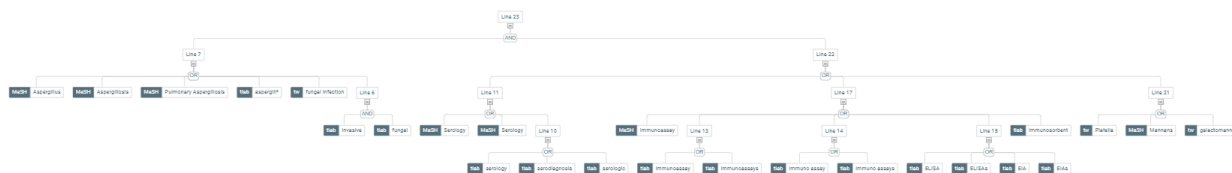
However, opening this strategy using 2Dsearch makes its structure much more apparent (fig. 4). The overall expression consists of a conjunction of two disjunctions (Lines 7 and 22), the first of which articulates variations on the fungal infection concept, while the second contains nested disjunctions to capture the diagnostic test (serology) and associated procedures. Evidently, the line numbers themselves are arbitrary in this context, but by displaying them as nested blocks with transparent structure, 2Dsearch offers support for abstraction in which users can hide lower-level details and reveal higher-level structure. Moreover, it is now possible to give meaningful names to such blocks, so they can be saved and reused as modular components.

Although visualization of search strategies in this manner offers immediate utility, the true value of the approach is in the interaction design. For example, to edit the expression, the user can move terms from one block to another and create new groups simply by combining terms. They can also cut, copy, delete, and lasso multiple objects. If they want to understand the effect of one block in isolation, they can execute it individually or use its hit count, as shown in blue ellipses in Figure 3. Conversely, if they want to remove one element from consideration, they can temporarily disable it.

The effects of each change display in real time in the adjacent search results pane, which allows users to rapidly optimize their search queries.

As we discussed in Section 2, crafting syntactically correct search expressions can be an error-prone and tedious process. Line numbers, parentheses, square brackets, punctuation, whitespace characters, and Boolean operators all have the potential for errors. However, a visual representation can delegate to lower-level system functions the task of generating syntactically correct expressions. Transforming logical structure into a visual layout provides a more direct mapping between the underlying semantics and physical appearance, and it provides a more intuitive experience for users wishing to experiment with different approaches.

The view in Figure 4 (which we call the Nested View) is just one of many. Another way to understand the hierarchy embodied in complex searches is to apply the metaphor of the family tree. The Tree View represents the search as a visual hierarchy, with the root node at the top and each level below represented as successive generations of children (fig. 5).

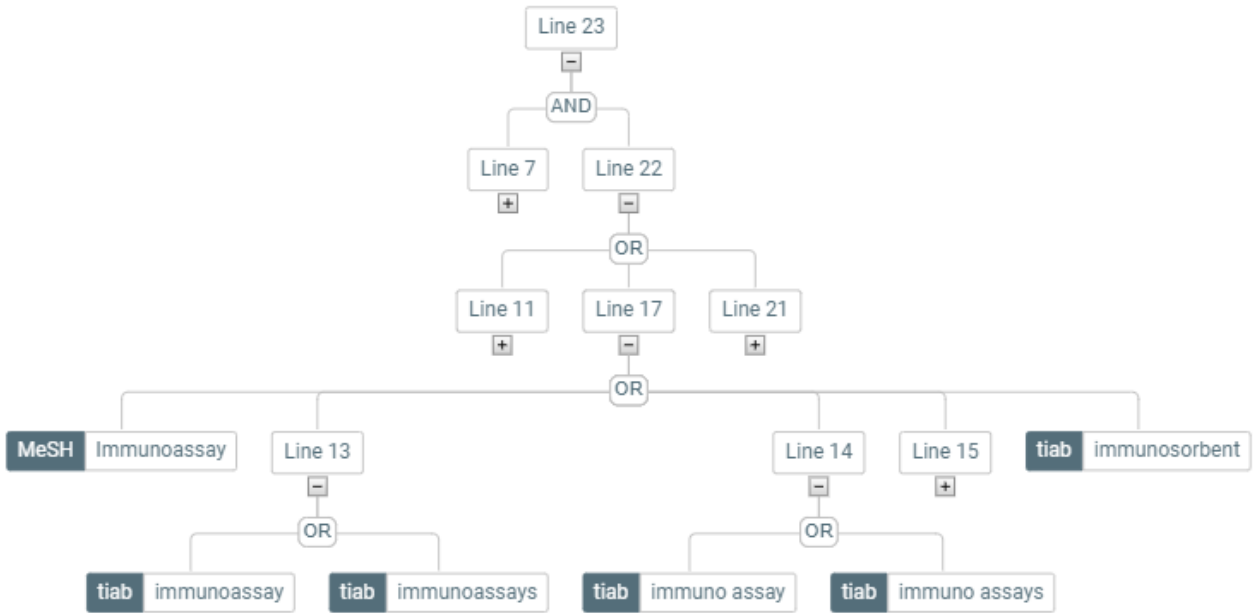


[\[/w/weave/images/12535642.0003.102-00000005.png\]](#)

Figure 5.

The Tree View.

This example displays the search in its entirety, shrunk to fit across the page. But it's easy to zoom and reveal just the higher levels or to close branches and focus on one particular region of the tree (fig. 6).



[\[/w/weave/images/12535642.0003.102-00000006.png\]](#)

Figure 6.

The Tree View, partially expanded.

Boolean strings may be inefficient and error-prone, but they do offer one key benefit: one can read them left to right. Of course, this attribute may reflect nothing more than the inertia of decades of convention, but there remains something enduring about being able to read searches in this way. We developed a third visualization, the Inline View to support this principle. Like the Nested View, the Inline View maps conceptual hierarchy in a manner that aligns groups along a common midline, facilitating a natural left-to-right reading (fig. 7).



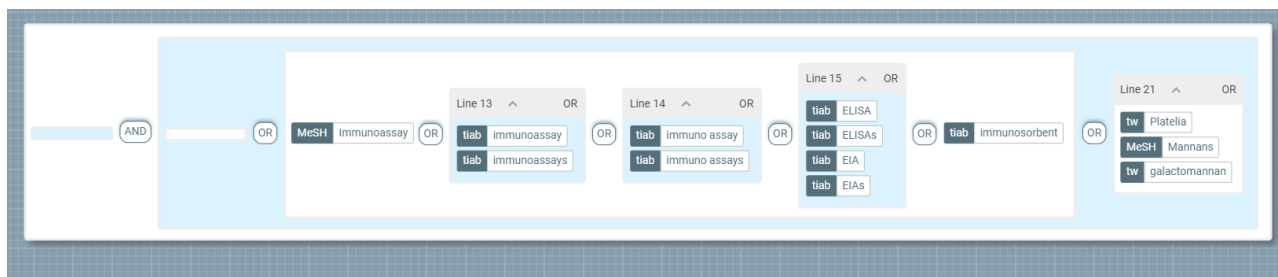
[\[/w/weave/images/12535642.0003.102-00000007.png\]](#)

Figure 7.

The Inline View.

Notice that in this view we elevate the operators to same level as content items, so they appear in sequence. This means we can also reduce some of the chrome around groups, leading to a cleaner

layout. Again, we've shrunk the figure to fit across a single page, but it is easy to zoom to display only the higher levels or to close branches to focus on one particular region of the search (fig. 8).



[\[/w/weave/images/12535642.0003.102-00000008.png\]](#)

Figure 8.

The Inline View, partially expanded.

4.2 Optimizing Search Strategies

2Dsearch functions as a meta-search engine, so is in principle agnostic of any particular search technology or platform. In practice however, to execute a query and retrieve results the semantics of the canvas content must be mapped to the syntax of the underlying database. 2Dsearch achieves this via a set of adapters for search platforms such as Bing, Google, PubMed, Google Scholar, Epistemonikos, and TRIP Database. Users select these in the interface via a drop-down control.

In developing a search strategy, information professionals will often revise and create alternative versions to find the best combination of search blocks and to understand the contribution of each to the overall result set. With 2Dsearch, it is possible to disable or enable individual blocks and dynamically investigate their contribution to the results, eliminating the need to create or manage alternate versions.


It is common for healthcare information professionals to want to search more than one database, particularly when undertaking a systematic literature review (Russell-Rose & Chamberlain, 2017). [\[https://bit.ly/2UU38kT\]](https://bit.ly/2UU38kT). This requires a process of translation of the search strategy to match the syntax of each database. For a relatively simple query this may not be a major undertaking, although the user still has to understand which elements are platform-specific, identify the closest equivalent in the other database, and manually edit their query—all of which is laborious and time consuming (Bramer et al., 2017). 2Dsearch invokes the PolyGlot Search Translator (<http://sr-accelerator.com/#/polyglot>) which provides automated translations on the Query tab of the result pane (fig. 9).

Search: PubMed REFRESH HIDE RESULTS


RESULTS MESSAGES **QUERY**

Query string

(Aspergillus[MeSH] OR Aspergillosis[MeSH] OR "Pulmonary Aspergillosis"[MeSH] OR aspergill*[tiab] OR "fungal infection"[tw] OR (invasive[tiab] AND fungal[tiab])) AND ((Serology[MeSH] OR Serology[MeSH] OR (serology[tiab] OR serodiagnosis[tiab] OR serologic[tiab])) OR (Immunoassay[MeSH] OR (immunoassay[tiab] OR immunoassays[tiab] OR ("immuno assay"[tiab] OR "immuno assays"[tiab]) OR (ELISA[tiab] OR ELISAs[tiab] OR EIA[tiab] OR EIAs[tiab]) OR immunosorbent[tiab]) OR (Platelia[tw] OR Mannans[MeSH] OR galactomannan[tw]))



Ovid Medline / Ovid Embase	▼
Cochrane Library	▼
Embase	▼
Web of Science	▼
CINAHL	▲
<p>((MH "Aspergillus+") OR (MH "Aspergillosis+") OR (MH "Pulmonary Aspergillosis+") OR TI aspergill* OR AB aspergill* OR "fungal infection" OR (TI invasive OR AB invasive AND TI fungal OR AB fungal)) AND (((MH "Serology+") OR (MH "Serology+") OR (TI serology OR AB serology OR TI serodiagnosis OR AB serodiagnosis OR TI serologic OR AB serologic)) OR ((MH "Immunoassay+") OR (TI immunoassay OR AB immunoassay OR TI immunoassays OR AB immunoassays) OR (TI "immuno assay" OR AB "immuno assay" OR TI "immuno assays" OR AB "immuno assays") OR (TI ELISA OR AB ELISA OR TI ELISAs OR AB ELISAs OR TI EIA OR AB EIA OR TI EIAs OR AB EIAs) OR TI immunosorbent OR AB immunosorbent) OR (Platelia OR (MH "Mannans+") OR galactomannan))</p>	
PsycInfo	▼
Scopus	▼
Lexical Tree (JSON)	▼
Lexical Tree (Human Readable)	▼

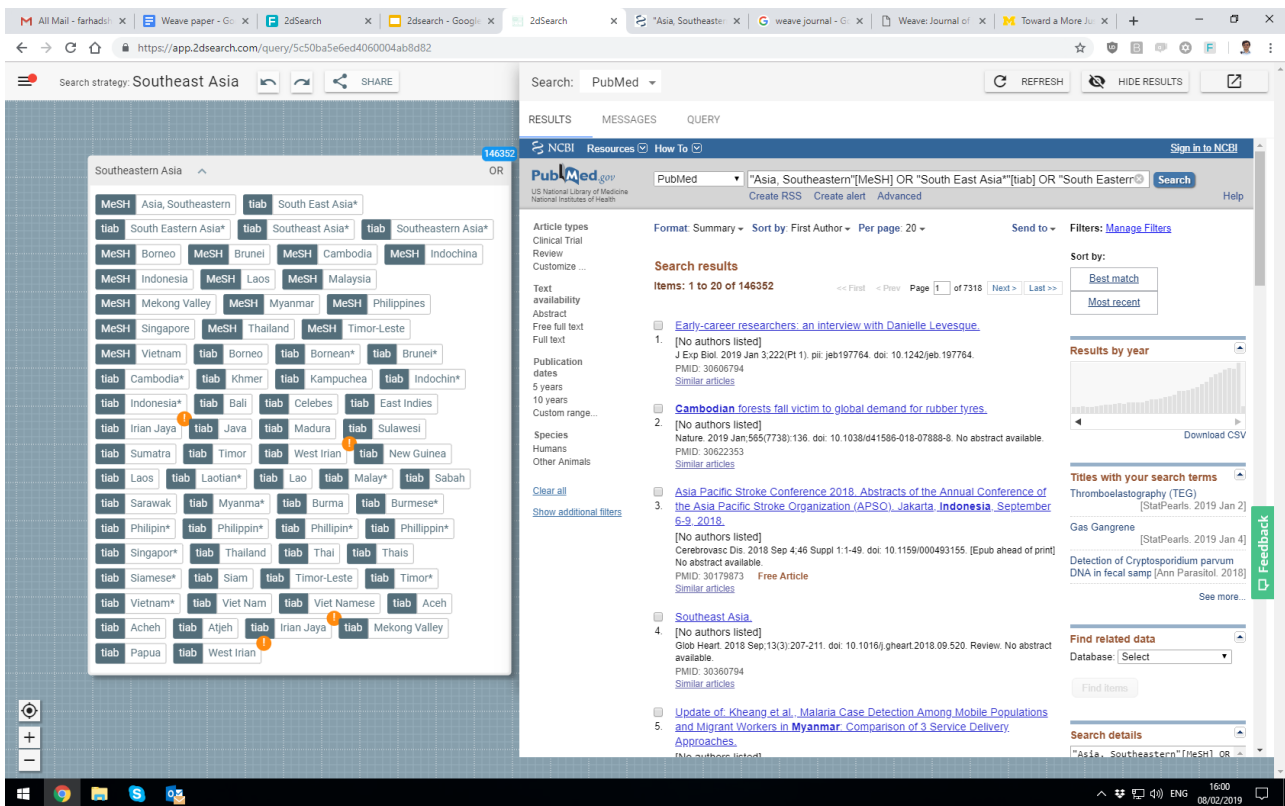


[\[/w/weave/images/12535642.0003.102-00000009.png\]](#)

Figure 9.

Automated translations of search syntax.

Although the primary purpose of 2Dsearch is to embody new approaches to structured searching, we recognize that much content exists in legacy form. For this reason, 2Dsearch provides support for importing traditional text-based strategies and rendering them on the canvas as editable, executable objects. In so doing, 2Dsearch identifies errors in the form of redundant structure, e.g. spurious brackets or duplicate elements (fig. 10).

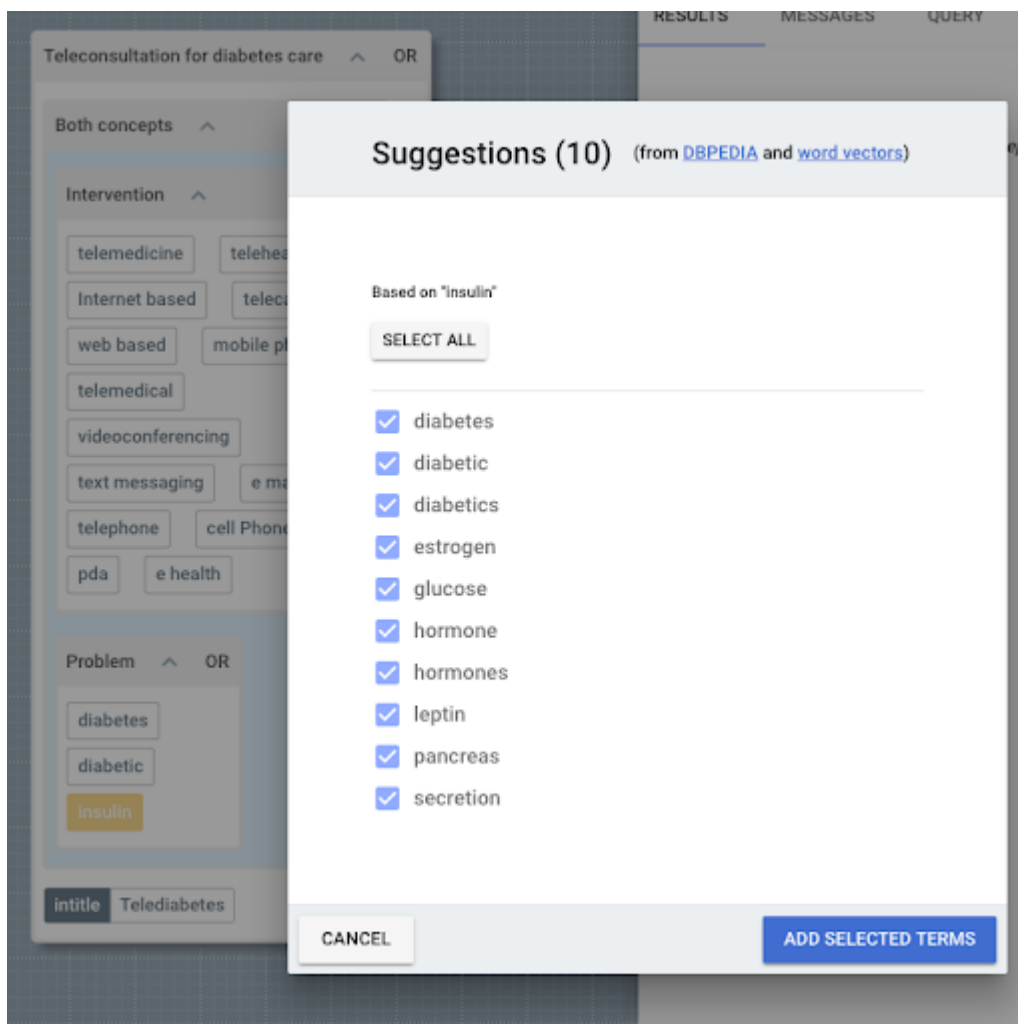


[/w/weave/images/12535642.0003.102-00000010.png]

Figure 10.

2Dsearch identifies duplicates in imported search strings and labels them with exclamation marks.

A further challenge in search strategy development is identifying appropriate keywords and controlled vocabulary terms. 2Dsearch provides support for this via a natural language processing (NLP) services API (fig. 11), which uses a variety of machine-learning techniques and manually curated resources (e.g. MeSH, DBPEDIA) to provide interactive query suggestions (Russell-Rose & Gooch, 2018).



[\[/w/weave/images/12535642.0003.102-00000011.png\]](http://w/weave/images/12535642.0003.102-00000011.png)

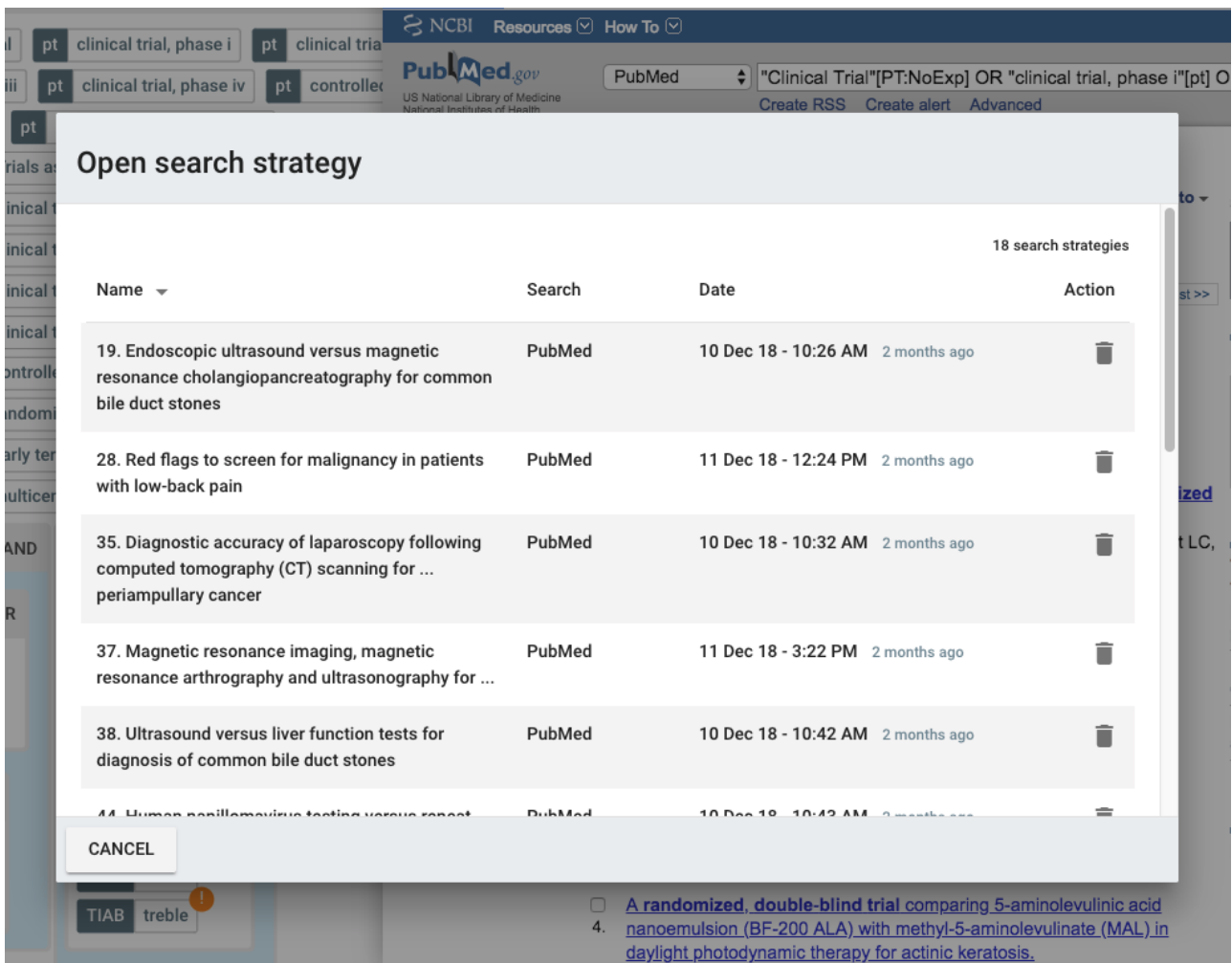
Figure 11.

Interactive query suggestions.

4.3 Working with Existing Search Strategies

Formulating effective search strategies can involve hours of effort, and consequently many information professionals routinely save their work as documents within their local file system. There have been various initiatives to provide a central repository for such artifacts, such as MedTerm Assist (Saleh et al., 2014), and various forums for reviewing them, such as the PRESS forum (Lefebvre et al., 2019). However, these repositories are not universally supported and are often closed to non-members. Moreover, they typically manipulate content as unstructured text strings or using word-processing tools which can result in the undesirable transformations we discussed earlier.

By contrast, users can freely distribute searches in 2Dsearch as hyperlinks generated using the Share button (see fig. 10). In this way, 2Dsearch offers an open-access, online platform for saving, sharing, and executing search strategies as validated, reproducible artifacts (fig. 12).



<https://weave/images/12535642.0003.102-00000012.png>

Figure 12.

Users can save searches in a public or private repository for archival purposes or peer review.

5. Summary and Further Work

In this paper we have reviewed conventional approaches to structured searching and identified a number of shortcomings. We have reviewed alternative paradigms and identified key design insights and principles that help mitigate these shortcomings. In 2Dsearch, we have applied and extended these insights to develop an alternative approach to traditional command-line query builders. In particular, we have developed a framework for search-strategy formulation in which users express queries by combining objects on a two-dimensional canvas. Transforming logical structure into visual layout provides a more direct mapping between the underlying semantics and physical appearance. This helps to eliminate syntax errors, makes the query semantics more transparent, and offers new ways to optimize, share, and reproduce search strategies. The platform currently provides adapters for Google, Google Scholar, Bing, PubMed, Epistemonikos, and Trip Database. In due course, we will provide other adapters, but our next objective is to engage in a formal, user-centric evaluation, particularly in comparison to traditional query builders. We are currently engaging with the library and information science community in this regard and welcome feedback. We also invite subject-matter experts to work with us in building repositories of best practice examples and templates.

Adopting a database-agnostic approach presents challenges, but it also offers the prospect of a universal framework in which users can articulate information needs in a generic manner and

delegate to platform-specific adapters the task of mapping to the syntax of an underlying database. If that transpires to be a practicable proposition, then such a development could have profound implications for the way in which people teach, learn, and apply professional search skills.

References

- Anick, P. G., Brennan, J. D., Flynn, R. A., Hanssen, D. R., Alvey, B., & Robbins, J. M. (1989). A direct manipulation interface for Boolean information retrieval via natural language query. *Proceedings of the 13th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM. <https://doi.org/10.1145/96749.98015> [<https://doi.org/10.1145/96749.98015>]
- Booth, A. (2008). Unpacking your literature search toolbox: On search styles and tactics. *Health Information and Libraries Journal*, 25(4), 313–317. <https://doi.org/10.1111/j.1471-1842.2008.00825.x> [<https://doi.org/10.1111/j.1471-1842.2008.00825.x>]
- Bramer, W. M., Rethlefsen, M. L., Kleijnen, J., & Franco, O. H. (2017). Optimal database combinations for literature searches in systematic reviews: A prospective exploratory study. *Systematic Reviews*, 6(1), 245. <https://doi.org/10.1186/s13643-017-0644-y> [<https://doi.org/10.1186/s13643-017-0644-y>]
- de Jonge, G., & Lein, R. K. (2015). Sharing literature search blocks: Status and ideas for a cooperative solution. *Journal of EAHIL*, 11(3), 11–14. [<http://f1000.com/work/bibliography/6394815>]
- Chui, M., Manyika, J., Bughin, J., Dobbs, R., Roxburgh, C., Sarrazin, H., Sands, G., & Westergren, M. (2012). *The social economy: Unlocking value and productivity through social technologies*. McKinsey Global Institute. <https://www.mckinsey.com/industries/technology-media-and-telecommunications/our-insights/the-social-economy> [<https://www.mckinsey.com/industries/technology-media-and-telecommunications/our-insights/the-social-economy>]
- Dijkstra, E. (1968). Go to statement considered harmful. *Communications of the ACM*, 11, 147–148.
- Fishkin, K., & Stone, M. C. (1995). Enhanced dynamic queries via movable filters. *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM. <https://doi.org/10.1145/223904.223960> [<https://doi.org/10.1145/223904.223960>]
- Gibbs, A. (2006). *Heuristic Boolean patent search: Comparative patent search quality/cost evaluation super Boolean vs. legacy Boolean search engines*. Patent Cafe.
- Goldberg, J. H., & Gajendar, U. N. (2008). Graphical condition builder for facilitating database queries.
- Gulhane, L. (2019). Search strategy library: Testing and evaluating a resource for sharing literature searches and blocks. *European Association for Health Information and Libraries Workshop*, Presentation conducted at the meeting of European Association for Health Information and Libraries, Basel.

- Jones, S. (1998). Graphical query specification and dynamic result previews for a digital library. *Proceedings of the 11th Annual ACM Symposium on User Interface Software and Technology*. ACM. <https://doi.org/10.1145/288392.288595> [<https://doi.org/10.1145/288392.288595>]
- Leeflang, M. M., Debets-Ossenkopp, Y. J., Wang, J., Visser, C. E., Scholten, R. J., Hooft, L., Bijlmer, H. A., Reitsma, J. B., Zhang, M., Bossuyt, P. M., & Vandenbroucke-Grauls, C. M. (2015). Galactomannan detection for invasive aspergillosis in immunocompromised patients. *Cochrane Database of Systematic Reviews*. <https://doi.org/10.1002/14651858.CD007394.pub2> [<https://doi.org/10.1002/14651858.CD007394.pub2>]
- Lefebvre, C., Manheimer, E., & Glanville, J. (2008). Searching for studies. In J.P.T. Higgins & S. Green (Eds.), *Cochrane Handbook for Systematic Reviews of Interventions* (pp. 95–150). [<http://f1000.com/work/bibliography/6367663>]. Cochrane.
- Lefebvre, C., McGowan, J., Salzwedel, D., & Sampson, M. (2019). PRESSforum: The peer review portal for librarians. <http://pressforum.pbworks.com> [<http://pressforum.pbworks.com>]
- Nitsche, M., & Nürnberger, A. (2013). QUEST: Querying complex information by direct manipulation. In S. Yamamoto (Ed.), *Human interface and the management of information. information and interaction design* (Vol. 8016, pp. 240–249). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-39209-2_28 [https://doi.org/10.1007/978-3-642-39209-2_28]
- Pazer, J. W. (2013). The importance of the Boolean search query in social media monitoring tools. <https://www.dragon360.com/wp-content/uploads/2013/08/social-media-monitoring-tools-boolean-search-query.pdf> [<https://www.dragon360.com/wp-content/uploads/2013/08/social-media-monitoring-tools-boolean-search-query.pdf>]
- Russell-Rose, T., & Chamberlain, J. (2016). Searching for talent: The information retrieval challenges of recruitment professionals. *Business Information Review*, 33(1), 40–48. [<http://f1000.com/work/bibliography/6378257>] <https://doi.org/10.1177%2F0266382116631849> [<https://doi.org/10.1177%2F0266382116631849>]
- Russell-Rose, T., & Chamberlain, J. (2017). Expert search strategies: The information retrieval practices of healthcare information professionals. *JMIR Medical Informatics*, 5(4), e33. <https://doi.org/10.2196/medinform.7680> [<https://doi.org/10.2196/medinform.7680>]
- Russell-Rose, T., Chamberlain, J., & Shokraneh, F. (2019). A visual approach to query formulation for systematic search. *Proceedings of the 2019 Conference on Human Information Interaction & Retrieval*. <https://doi.org/10.1145/3295750.3298919> [<https://doi.org/10.1145/3295750.3298919>]
- Russell-Rose, T., & Gooch, P. (2018). 2dSearch: A visual approach to search strategy formulation. *Proceedings of the First Biennial Conference on Design of Experimental Search & Information Retrieval Systems*. <http://ceur-ws.org/Vol-2167/paper8.pdf> [<http://ceur-ws.org/Vol-2167/paper8.pdf>]

- Saleh, A. A., Ratajeski, M. A., & Ladue, J. (2014). Development of a web-based repository for sharing biomedical terminology from systematic review searches: A case study. *Medical Reference Services Quarterly*, 33(2), 167–178. <https://doi.org/10.1080/02763869.2014.897518>
[<https://doi.org/10.1080/02763869.2014.897518>]
- Sampson, M., & McGowan, J. (2006). Errors in search strategies were identified by type and frequency. *Journal of Clinical Epidemiology*, 59(10), 1057–1063.
<https://doi.org/10.1016/j.jclinepi.2006.01.007> [<https://doi.org/10.1016/j.jclinepi.2006.01.007>]
- Shojania, K. G., Sampson, M., Ansari, M. T., Ji, J., Doucette, S., & Moher, D. (2007). How quickly do systematic reviews go out of date? A survival analysis. *Annals of Internal Medicine*, 147(4), 224–233. <https://doi.org/10.7326/0003-4819-147-4-200708210-00179>.
[<https://doi.org/10.7326/0003-4819-147-4-200708210-00179>]
- Shokraneh, F. (2016). PICO framework: Two decades of variation and application. *Evidence Live*.
<https://10.13140/RG.2.2.35019.23841> [<https://10.13140/RG.2.2.35019.23841>]
- Team. (n.d.) 2Dsearch. Retrieved Feb 18, 2020, from <https://www.2dsearch.com/our-team>
[<https://www.2dsearch.com/our-team>]
- Yi, J. S., Melton, R., Stasko, J., & Jacko, J. A. (2005). Dust & magnet: Multivariate information visualization using a magnet metaphor. *Information Visualization*, 4(4), 239–256.
<https://doi.org/10.1057/palgrave.ivs.9500099> [<https://doi.org/10.1057/palgrave.ivs.9500099>]

Hosted by [Michigan Publishing](#), a division of the [University of Michigan Library](#).

ISSN 2333-3316



[About](#)

[Editorial Board](#)

[Editorial Philosophy](#)

[Contact](#)

[Log in](#)

Follow us:

[Twitter](#)

[Email](#)

[RSS](#)