Evaluating Advanced Search Interfaces using Established Information-Seeking Models

Max L. Wilson
School of Electronics and Computer Science
University of Southampton, UK
mlw05r@ecs.soton.ac.uk

m.c. schraefel
School of Electronics and Computer Science
University of Southampton, UK
mc@ecs.soton.ac.uk

Ryen W. White
Microsoft Research
One Microsoft Way
Redmond, WA 98052
ryenw@microsoft.com

ABSTRACT

When users have poorly defined or complex goals, keyword searching may not provide sufficient support. Subsequently, more advanced systems are being developed to support richer modes of search. This paper presents a formative framework for evaluating advancing search systems, which are providing more versatile environments that become increasingly hard to compare. This is done by quantifying their strengths and weaknesses in supporting user tactics and varying user conditions. This framework combines established models of users, user needs, and user behaviours to achieve this. The framework is applied to evaluate three advanced search interfaces and shows promising results.

Categories and Subject Descriptors

H.3.4 [**Information Storage and Retrieval**]: Systems and Software—*Performance evaluation*; D.2.8 [**Software Engineering**]: Metrics—*complexity measures, performance measures*; D.2.2 [**Software Engineering**]: Design Tools and Techniques—*User Interfaces*

Keywords

Search, Browse, Exploratory, Advanced, Evaluation, Comparison, HCI, Faceted

1. INTRODUCTION

Keyword search has become the default standard for exploring the Web. However, while this approach is powerful, it does not support users well when they have poorly defined goals or complex questions, have insufficient pre-search knowledge, or may be using a system with poorly defined or unpredictable indexing [13]. To support these situations, richer modes of search, such as Faceted Browsing [6], are being developed. While these more interactive models of search provide increasingly versatile combinations of functions, the challenge is not to simply add more features but to combine them to produce synergetic designs[10]. To evaluate rich support for search, metrics need to consider strategies of information-seeking behaviour so we may understand how well, or not, they are endorsed by a design.

In this paper, we describe a novel formative application of established models of information-seeking behaviour to evaluate the performance of three faceted browsers. We use the lessons learnt from Information-Seeking Research to specifically quantify the support for user needs and the tactics they may employ to meet them. By using this as a measure for support, the strengths and weaknesses of design can be identified to motivate potential redesign that will improve the search experience. This approach is used to produce a measure for comparing systems that may be otherwise difficult to compare.

In particular, we first hypothesise that by applying this combination of models, we can quantify the strengths and weaknesses of three interface designs in terms of their support for established search tactics. Second, we hypothesise that these strengths and weaknesses can be attributed to different search conditions to identify support for particular types of user. To test these, we first present related work including established models of user search behaviour. A combination of these are included in an evaluation framework, which is applied to three example faceted browsers. We conclude with the results, implications and future developments to convert the approach towards a reusable framework for evaluating rich search environments.

2. MODELS OF INFORMATION SEEKING

The recent interactive, cognitive and relevance revolutions in Information Retrieval (IR) literature [11] have spawned an interest in the human element in search dating back to the mid-1970s. Recent research [8, 10] continues to recognise the need for better understanding the Human-Computer Interaction (HCI) side of IR. The aim is not only produce interfaces that include a collection of features but develop effective user interfaces that specifically support Human Computer Information Retrieval (HCIR) and support more effective information-seeking[13]. In the following section, we summarise some of the research produced by these revolutions that purposefully consider the support for user oriented search and can be used to produce metrics to quantify interface evaluations.

2.1 Exploratory Search

The term Exploratory Search (ES) has been coined for de-

SIGIR '07 Amsterdam, Netherlands

scribing IR that incorporates actions such as investigating and learning, which requires larger user involvement [13]. Marchionini points out that, although there are projects that claim to support ES, including the faceted browsers discussed below, ES covers a broad range of activities: these include comparison, aggregation, and evaluation and do not simply require looking up answers [10].

2.2 Stratified Models

Saracevic's stratified approach to IR encompasses this human effort by presenting a layered model that focuses on both the user and the system having equal constraints on search. There are multiple levels of interaction and processing on both the user and system side [12]. Notably a system involves hardware, processing and data structures. For a user, their interaction involves cognitive, affective and situational levels, which represent their interpretation, motivation, and their requirements respectively. Both the system and user aspects were later extended into much greater detail by Bates[2], who identified additional levels that interact and affect each other. The key conclusion drawn from stratified models is that even if a great searching algorithm is implemented, or an intuitive user interface is designed, poor indexing or hardware can ruin their collective usefulness. Similarly, a user's interpretation of results may seriously affect their success in achieving their goals with the system.

2.3 Episodic Models

While Saracevic's model can be used to evaluate the concurrently running layers of an IR system to identify at what level support for user search is constrained, Belkin *et al.* [3] have produced an episodic model to define and understand the flow in scenarios of human-system interactions: these flow definitions are called "scripts". However, to do this, Belkin *et al.* first highlights four binary dimensions that define 16 unique Information-Seeking Strategies (ISS). They have calculated separate scripts for each of these 16 ISS conditions, which allow for switching between them. The dimensions are *Method*, *Goal*, *Mode* and *Resource* and in combination produce sixteen conditions shown in Table 1.

ISS	Method	Goal	Mode	Resource
1	Scan	Learn	Recognize	Information
2	Scan	Learn	Recognize	Meta-Information
3	Scan	Learn	Specify	Information
4	Scan	Learn	Specify	Meta-Information
5	Scan	Select	Recognize	Information
6	Scan	Select	Recognize	Meta-Information
7	Scan	Select	Specify	Information
8	Scan	Select	Specify	Meta-Information
9	Search	Learn	Recognize	Information
10	Search	Learn	Recognize	Meta-Information
11	Search	Learn	Specify	Information
12	Search	Learn	Specify	Meta-Information
13	Search	Select	Recognize	Information
14	Search	Select	Recognize	Meta-Information
15	Search	Select	Specify	Information
16	Search	Select	Specify	Meta-Information

Table 1: Information Seeking Strategies (cf. Belkin *et al.* 1995)

Method describes whether a user is either searching for an information object, or scanning a set of information objects. This is easily differentiated by finding a specific paper in order to get its reference details, or by searching for an possible paper, which may not exist, that can be used to support a point. *Goal* describes whether a user is learning about something or selecting something. Using the bibliographic example differentiates these as researching a topic, or finding a reference. *Mode* is between recognising and specifying something. One might remember that there was a useful publication at SIGIR2005 and so is trying to identify it in the proceedings, or may have known the author, title and year and has typed them into the ACM Portal. *Resource* is between wanting information items or meta data about an information item. Usually, with a bibliographic repository users are trying to find specific papers, but it is possible that the user is trying to find out first what workshops existed in a conference so that they can better define a search query at a later point in time.

For example, Google is best used for ISS15, where the user is searching (*Method*) to select (*Goal*) by specifying (*Mode*) attributes of a specific information object (*Resource*). Subsequently it least supports users who are scanning (*Method*) to learn (*Goal*) by recognising (*Method*) some meta data about an information object (*Goal*): this is ISS2. Faceted browsing tries to support users by presenting all the meta-information to the user in advance and letting them choose. Conversely, this best supports ISS2, but may poorly support ISS15: useful meta-data can be embedded in long lists and it may require more effort to find them than to simply type them into a search box.

Belkin *et al.* realised that these dimensions were not completely exhaustive and so extended and expanded upon these four dimensions with much greater detail in a later publication[5]. However, the new model goes into more detail than is easily coverable and reusable within the constraints of this paper: the initial four dimensions are expressive enough to classify seeking behaviours here.

2.4 User Search Activities

In 1990, Bates described in detail two models for both the different levels of search activities and the different levels of system automation [1]. First she identifies 5 levels of system ranging from complete user action to complete system automation. This is then combined with four levels of search activities: Move, Tactic, Stratagem and Strategy. The first of these is a single action performed by the user, either physically or mentally: mental actions may be deciding or reading. A tactic is a combination of moves and there are endless combinations of moves that can be used to support a tactic, which depends on system implementations. She defines 32 specific information search tactics. Stratagems are a larger combination of both individual moves and tactics: some examples include performing a citation search or following a footnote. Strategies are again higher and involve a combination of moves, tactics and stratagems: this might be finding relevant work for a paper and depends heavily on what the user is currently working on.

Bates suggests that by combining these, a system that supports fully automated strategy is one that reads a user's mind and knows what she is doing and that most systems provide no automatic support for strategies, allowing complete human control. A system that has no automation for moves is a pile of unsorted paper. The work on scent above is trying to automate and support stratagems and most software systems automate moves at least.

We suggest that strategies and even stratagems may define the use of a search interface and the choice will be based

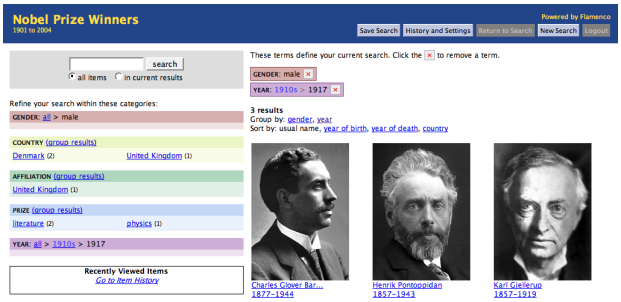

Figure 1: The Flamenco Interface on the Nobel Prize Winners Dataset

upon the tactics it supports and the current conditions for the user, such as Belkin's ISSs. Thus we propose a combination of Bates' lower levels of search activity and Belkin's ISS model to produce a theoretical metric evaluation of search interfaces.

3. FACETED BROWSERS

Faceted Browsers are an example class of Exploratory Search Interfaces (ESIs), which present meta-data attributes as a series of selectable categorised options. Typically the meta-data is created around a series of Target Objects (TOs), which represent the core information being sought. In a bibliographic repository, TOs would be the literature being sought. Through modelling a domain in a faceted structure, direct manipulation can be used to construct queries. Thus, when a user is not clear on appropriate terminology or the meta-data is unpredictable, they do not have to estimate search terms, but can make selections to build their query. Through this extra support faceted browsers can be considered a type of ESI. Below, we discuss three example faceted browsers developed in academia and motivated by improving access to information. Other faceted browsers exist. Endeca [1] is a commercial faceted browser that is not publicly accessible for research purposes. More recently, facet has been developed to use faceted browsing for supporting information architecture and evolution [7].

3.1 Flamenco

Flamenco[2], shown in Figure 1, supports both keyword search and faceted browsing, accounting for both those who know their target and those who don't have much knowledge about the domain. The initial display shows all the possible facets in two columns, with vertical scroll as necessary. By entering a search query or selecting an item in one of the facets, the user is moved away from the initial view to one where all the facets are listed vertically down the left column, with the search box remaining at the top left. A breadcrumb is located at the top right, which presents the path of selections made by a user. A search term acts as a domain filter and the search results (displayed in the remaining space at the bottom right) may still be browsed using the facets. If the search term can be matched to particular items in the facets, these are presented to the user above the breadcrumb.

[1]http://www.endeca.com - Endeca
[2]http://flamenco.berkeley.edu/ - Flamenco Home

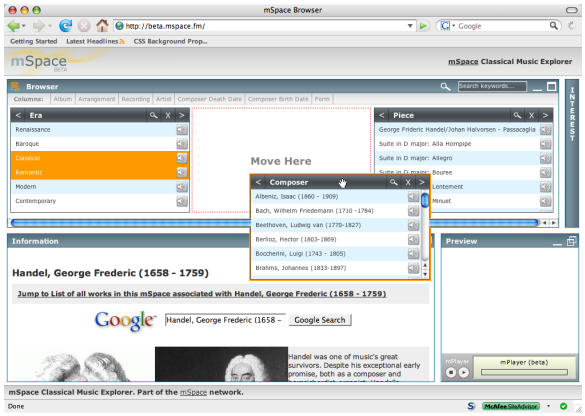

Figure 2: The mSpace Interface on the Classical Music Dataset

When a selection is made in a facet, the sub-categories within the facet are shown and a per-facet breadcrumb displays the selection made. If there are no sub-items, the facet is effectively minimised (facet representations grow endlessly with the number of options within it). If facets are hierarchical, results are automatically clustered into the sub-categories of the latest selection. The user may optionally group the results by any other facet through a single interaction provided by the presence of a new link along side of each facet name. Any potential option for selection is accompanied by numeric volume indicators (NVIs [15]), to estimate the number of TOs that can be reached by its selection.

When TO selections are made, the user is moved away from the faceted browser display to one that shows a summary of the data associated with their choice. From here, the user is given options to return to the faceted browser: extra facet selections can be made to expand or further narrow their constraints and view similar objects. Users may also reset the interface by pressing the 'New Search' button.

3.2 mSpace

mSpace[3], shown in Figure 2, also supports both keyword search and faceted selections. Normally, the user is presented with four panels: Facet Browser, Interests, Information, and Preview Cues. These panels remain persistently available throughout the subsequent interactions with mSpace, using a zooming interface technique.

The facet browser holds a series of active columns, and optional columns are listed in the header above. Within a facet, scrolling can be reduced using a keyword filter. The number and order of active facets can be changed freely, and horizontal scrolling is used if necessary. Unlike Flamenco, users may make multiple selections within any facet, and the results combined using logical 'OR'. NVIs are also presented. While most faceted browsers allow the reordering of facets for aesthetic purposes, the order of the facets in mSpace matters and forms a hierarchy from left to right. By reordering the columns, the users can construct a path towards TOs using meta-data they are comfortable with. Other facets can be simply removed from the active set to avoid confusion. The majority of objects can be 'dragged and dropped' to further support direct manipulation. Fi-

[3]http://mspace.fm/ - mSpace

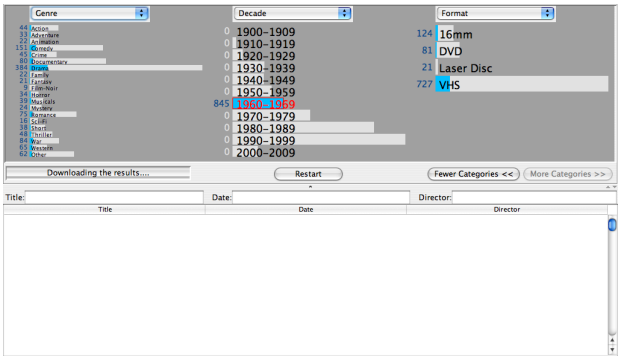

Figure 3: The RB++ Interface on the UNC Movie Catalogue

nally, the ordering of columns also represents an in-place breadcrumb of search decisions.

The second panel, at the top right, is an interest box that allows users to save and keep objects, much like a favourites or bookmark list at the top of a browser. Uniquely, this saving system allows the regeneration of the path taken to find an item, which can be anything in the columns and not just TOs. The bottom left panel is an information panel which can provide information about the last selection. If this is a TO, then a user will expect to find the details of the TO here: this means that when viewing a specific object, a user has not lost their browsing path. However, in the classical music demo, users can also see information about any facet item, such as composers or eras. This box can also be used for search results, through either the keyword search at the top-right or as constrained by the selections in the columns.

The bottom right panel is a preview cue panel, which presents a space on the screen for giving example TOs. One unique attribute of this preview cue panel is that it can be used to display example TOs for anything in the facets. In the classical music demo, users can hear examples of classical music from each era, composer, arrangement, etc. This is triggered by icons that appear at the right hand side of each item in the facet contents.

3.3 Relation Browser

The relation browser, named RB++[4] and shown in Figure 3, currently presents all the facets and their contents persistently: these facets are listed across the top of the UI and grow/shrink to fit on the screen. Users can reorder the columns for aesthetic reasons, using a drop down list that formulates as both a mechanism for changing the facet and also for displaying its label. There is no breadcrumb visualisation. NVIs are represented as an in-place bar graph. The population of the bar represents both the number of achievable TOs from making that selection and, uniquely, the number of total TOs in the dataset. The exact NVI value is represented to the left of each label. Hovering over items in each facet previews the affects of the selection on each of these NVIs and is made persistent by clicking.

By pressing the *search* button, results are displayed in the lower half of the screen, where items can be filtered, sorted and individually selected. Once the search results are displayed, the previous selections above are transformed into a label representing the selections, much like a breadcrumb but without temporal order. The facet browser is also transformed to represent the subset of TOs that had been previously achieved through facet selection. Thus NVIs represent the number of TOs in the new subset. Any subsequent facet selections automatically filter the search results. Upon selection in the results, the TO is displayed in a new window.

[4]http://idl.ils.unc.edu/rave/ - Interactive Design Laboratory Presents RAVE

4. EVALUATION FRAMEWORK

In the following comparison, we employ two models of information seeking: The ISS conditions from Belkin's episodic model [3] and the levels of search activities presented by Bates[1]: Moves, Tactics, Stratagems and Strategies. These are both combined to represented various levels of Saracevic's stratified analysis of users [12]. In particular, the Moves of Bates' model are used to quantify the Tactics she later defines. At this stage, Stratagems and Strategies are ignored as their level of definition requires further work. Finally, while Belkin *et al.* used these ISS conditions to motivate the design of an IR system, here we combine the conditions with metrics to quantify the support provided for users by different implementations. The framework is applied in the following six stages.

4.1 Application Procedure

Stage 1: Feature Identification. First, the interface features and their interactions must be identified. For example, mSpace has a set of features including: browser columns, a collection space, a preview player and an information panel. The features of each design should be incorporated.

Stage 2: Measuring Support for Tactics. Each interface feature is addressed one at a time, for each design. For the current feature of the current design, the moves required to support each tactic are counted. This produces a series of tables, one for each design, where tactics are listed across the top and the interface features down the side. The count of moves is noted in the appropriate cross section between feature and tactic. No support by a feature for a tactic counts as 0. Four moves for the user to use the feature in support of a tactic counts counts as 4. Repeat and Optional moves are ignored. For example, selecting multiple items involves choosing and selecting 2+ items, selecting 3+ is considered a repeat move of selecting 2 items. Optional moves include scrolling: a desired item may be the first or last item. The optimum situation is that it is one of the items that is visible without scrolling.

Stage 3: Summarising Metrics. As no support is represented by zero, support in a single move is represented by 1 and support in ten moves by 10, all values above 0 must be inverted. Thus a feature that supports a tactic well approaches the value of 1 and a poor support approaches 0. These inverted metrics can then be summed by feature and by tactic. This calculates the support provided by a feature for all tactics and the support provided for a tactic across all features, respectively.

Stage 4: Feature Strength Analysis. A graph can be produced including the summed values for each feature in each design. An example can be seen in the following section. Strong features will produce tall bars, and a comparison of user effort can indicate a strong feature design.

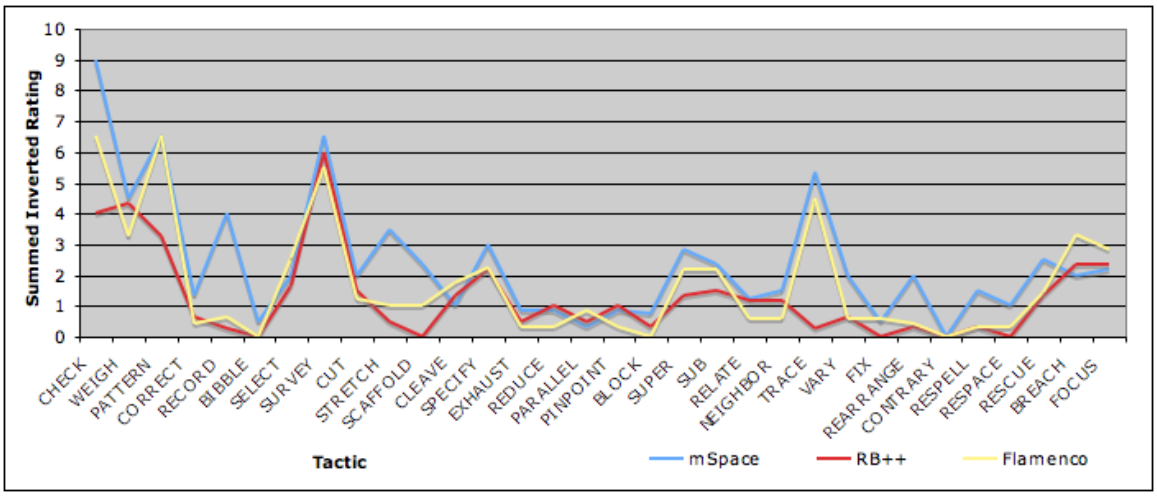

Figure 4: Graph Showing the Summed Inverted Metrics of each Tactic for the Three Browsers. Strong support for a Tactic is shown by a high metric value.

Stage 5: Tactic Support Analysis. A graph can be produced including the summed values for each tactic in each design. Again, tall bars indicate strong support for a tactic. This comparison may identify tactics which may require improved support through redesign.

Stage 6: User Conditions Analysis. Each tactic supports particular ends of Belkin's dimensions of user conditions. CHECK, a tactic for users checking their decisions so far, supports users who are trying to Learn as their *Goal*. The support for a tactic by a design is added to the total support for a dimension. Then for each of the sixteen conditions, the sum of the total support values are calculated. This value for each condition can be graphed showing the difference in support for different user conditions.

As the framework above, like the Key-Stroke model [4], assumes optimal user interaction, user studies have not been used to test its application. In the remainder of this section we describe the results discovered when applying the framework to the three faceted browsers above.

4.2 Support for Tactics

A number of observations can be drawn from Figure 4. First, each interface has a high peak for SURVEY. This is an expected peak when evaluating faceted interfaces because the user is presented with optional selections at each stage. This peak would not be so visible in keyword only interfaces. We now continue by investigating significant differences between interfaces in this graph.

The first tactic, CHECK, has different levels of support in all three interfaces: this tactic is to see what actions have made to corroborate them with the current aims. In RB++, although previous selections are highlighted in the interface, no representation of order is given and so a lower support for checking ones actions is provided. In Flamenco, this feedback is given in a breadcrumb, and is visible when navigating through the facets. To view a TO in Flamenco, the user is moved to a new page with a summary of that object. Thus, before the user can view the breadcrumb, they must first return to search: this requires two moves. In mSpace, breadcrumbs are embedded into the ordered facets. As mSpace is a focus+context browser, the user can view the facets and their previous actions at all times, including when viewing a TO. This leads to a higher peak for mSpace and then Flamenco in Figure 4.

There is a significant peak for the mSpace interface, which supports the RECORD tactic. The mSpace interface includes a within-browser collection space that can store any object in the facets. Although any state reached in Flamenco and mSpace can be saved using the parent application[5], and pages displaying TOs in all three interfaces can be saved in this way, a single double-click move can store facet items in the Interest panel of the mSpace browser at any point: even when viewing a TO it can be saved with by double-clicking or dragging the item into the box.

There is also a significant peak over the STRETCH and SCAFFOLD tactics for the mSpace browser. STRETCH, reusing objects in unintended ways, is highly supported because of the explicit ordering of facets. The reordering of facets allows users to see the effects of meta-data on other meta-data: this reordering involves a single dragging action. SCAFFOLD, finding quick paths to TOs, is highly supported, because selecting preview cue objects will brings up not only information about its TO, but can also be used to see its position in the facets. Users may recover a path used to find items in the Interest panel by dragging it onto the columns or double clicking the item, displaying a quick jump to a previous path.

It may be noted that mSpace is specifically higher over all of the Term Tactics (SUPER to RESPACE). It may also be noted that no interface supported CONTRARY, an antonym of a selection. After investigation, these higher ratings are supported mainly by a combination of features. While it is easy in Flamenco to use the SUPER tactic, by simply removing an item from the breadcrumb, users of mSpace have two options: they may simply identify and click on a different item, or they may reorder the columns so that a selection is placed higher up the temporary hierarchy. The former of these two is not achievable in Flamenco, as alternatives of a selection are hidden and the exact selection is only displayed in the breadcrumb. The RELATE and

[5]Usually an Web Browser such as Firefox or IE

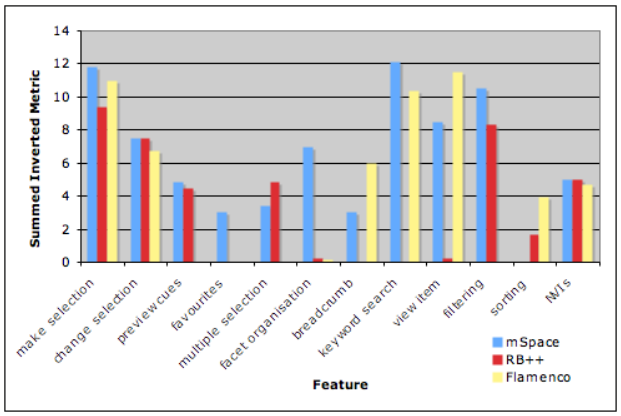

Figure 5: Graph Showing the Summed Inverted Metrics of each Feature for the Three Browsers. Strong support is indicated by a high metric value.

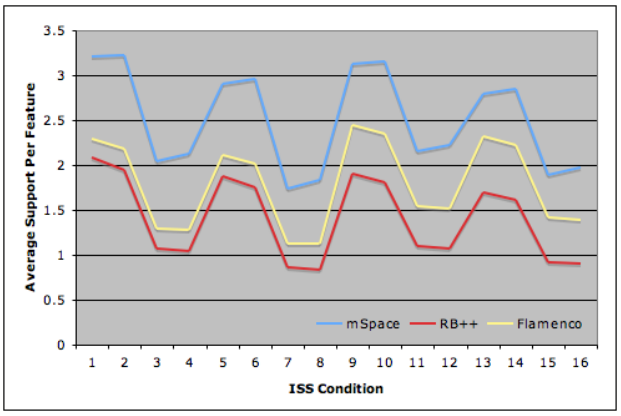

Figure 6: Graph Showing the Normalised Support for each ISS Condition by Faceted Browser

NEIGHBOR tactics are also poorly supported in Flamenco due to the aforementioned four step process to change a selection. REARRANGE is also well supported by mSpace due to the ease in reordering facets. Finally, tactics like RESPELL are well supported by mSpace because changes to misspellings and unrecognised words in the keyword search are suggested and can be applied by a single click.

Finally, SCAFFOLD and TRACE are both poorly supported by RB++ as the facet columns are used for two purposes: making facet selections and, once TOs have been listed, filtering TOs. The selections made before TOs are listed are hidden. It is a unique feature that this separation exists, as making facet selections are by nature filtering the TO list and most browsers merge these conditions.

4.3 Support of Features

Figure 5 shows the significant contribution of different interface features. Certain elements of the previous discussion can be seen here clearly. Flamenco's four steps to change a selection are reflected in the slight drop of their bar. It may also be noted that Flamenco has no preview cue, and thus the bar is absent from the graph. The ease of multiple selection in RB++ is also clearly shown. One feature to compare is 'View Item'. RB++ has a significant drop in support here, as the implementation has a significant separation between TOs and Browser. TO Pages may be simply launched from the browser, but there are no ways in which the user can interact with the browser when viewing them. The only option is to return to the browser. In Flamenco and mSpace, users can make further selections from the TO page that force automatic interactions with the facets: this is most direct in mSpace where the facets are always present.

mSpace has no sorting function, which is shown clearly on the graph, but is well supported by RB++ and Flamenco. In Flamenco, a user is able to group the results by any of the facets in the system and provides the strongest implementation of a sorting method. However, Flamenco does not support filtering. In mSpace, user can filter long lists of items in facets to jump quickly to selections. RB++ also provides the filtering of TOs by reusing the facets for filter selections: this support is only for TOs and presents weaker support for the interface. The in-browser collection space in the mSpace interface clearly provides support for the interface but is also unique to mSpace.

5. DISCUSSION

Two observations can be drawn from these graphs. First, there is better support for search in mSpace due to the wider number of implemented features. Second, each interface has strong features that prevail in the group. These two points are considered in greater detail below.

It is clear from Figure 4 that there are few tactics that are not best supported by the mSpace browser. This is explained from Figure 5 by the number of strong contending features: there are comparably high mSpace bars for almost every feature. This is arguably representative of the focus+context design, which aims to present as many options and features to the user as possible and at all times.

In line with the second observation, however, there are clearly some features of each interface that have stronger implementations of the three browsers. For example, multiple selection is easiest in RB++, yet keyword search is missing from RB++ and the implementation is strongest in mSpace. One feature missing from the mSpace implementation is the ability to sort items. The strongest implementation of this is the ability to group the results by any facet, as seen in Flamenco.

5.1 Supporting User Conditions

One question that was considered at the start of this paper is when a browser well supports user intention: under what user conditions does a browser provide good support. By attributing each Tactic to support one of Belkin's four dimensions of search, such as supporting learn (*Goal*) or meta-information (*Resource*), the support for each ISS condition can be quantified using the Summed Inverted Metrics as before: this is shown in Table 2 and in Figure 6.

The pattern that is seen almost identically for each interface in Figure 6 is indicative of the mapping between Bates' tactics and the pattern of ISS conditions defined by Belkin *et al.* Predictably, as was shown in Figure 4, there are three distinct lines, showing that mSpace provides the widest support for search. This height difference does not show us new information. Instead what should be drawn from the graph is hidden within this pattern and shown in the differences in peaks and troughs for each interface condition. Quite clearly

ISS	Tactics	mSpace		RB++		Flamenco	
1	CHECK, WEIGH, RECORD, SURVEY, EXHAUST, PARALLEL, SUPER, RELATE, NEIGHBOUR, RESCUE, BREACH	35.25	3.20	22.92	2.08	25.20	2.29
2	CHECK, WEIGH, RECORD, SURVEY, STRETCH, EXHAUST, PARALLEL, SUPER, RELATE, NEIGHBOUR, RESCUE, BREACH	38.75	3.23	23.42	1.95	26.20	2.18
3	CHECK, CORRECT, RECORD, CUT, SPECIFY, CLEAVE, EXHAUST, PARALLEL, BLOCK, SUPER, RELATE, NEIGHBOUR, REARRANGE, CONTRARY, RESPELL, RESPACE, RESCUE, BREACH	36.83	2.05	19.25	1.07	23.23	1.29
4	CHECK, CORRECT, RECORD, CUT, STRETCH, SPECIFY, CLEAVE, EXHAUST, PARALLEL, BLOCK, SUPER, RELATE, NEIGHBOUR, REARRANGE, CONTRARY, RESPELL, RESPACE, RESCUE, BREACH	40.33	2.12	19.75	1.04	24.23	1.28
5	WEIGH, RECORD, SELECT, SURVEY, SCAFFOLD, EXHASUT, PARALLEL, SUPER, RESCUE, BREACH	29.00	2.90	18.75	1.88	21.12	2.11
6	WEIGH, RECORD, SELECT, SURVEY, STRETCH, SCAFFOLD, EXHAUST, PARALLEL, SUPER, RESCUE, BREACH	32.5	2.95	19.25	1.75	22.12	2.01
7	CORRECT, RECORD, SELECT, CUT, SCAFFOLD, SPECIFY, CLEAVE, EXHAUST, PARALLEL, BLOCK, SUPER, REARRANGE, CONTRARY, RESPELL, RESPACE, RESCUE, BREACH	29.42	1.73	14.58	0.86	19.15	1.12
8	CORRECT, RECORD, SELECT, CUT, STRETCH, SCAFFOLD, SPECIFY, CLEAVE, EXHAUST, PARALLEL, BLOCK, SUPER, REARRANGE, CONTRARY, RESPELL, RESPACE, RESCUE, BREACH	32.92	1.83	15.08	0.84	20.15	1.12
9	CHECK, WEIGH, PATTERN, BIBBLE, SURVEY, REDUCE, PINPOINT, SUB, RELATE, NEIGHBOUR, TRACE, VARY, FIX, FOCUS	43.75	3.13	26.67	1.90	34.25	2.45
10	CHECK, WEIGH, PATTERN, BIBBLE, SURVEY, STRETCH, REDUCE, PINPOINT, SUB, RELATE, NEIGHBOUR, TRACE, VARY, FIX, FOCUS	47.25	3.15	27.17	1.81	35.25	2.35
11	CHECK, PATTERN, CORRECT, BIBBLE, CUT, SPECIFY, CLEAVE, REDUCE, PINPOINT, BLOCK, SUB, RELATE, NEIGHBOUR, TRACE, VARY, FIX, REARRANGE, CONTRARY, RESPELL, RESPACE, FOCUS	45.33	2.16	23.00	1.10	32.28	1.54
12	CHECK, PATTERN, CORRECT, BIBBLE, CUT, STRETCH, SPECIFY, CLEAVE, REDUCE, PINPOINT, BLOCK, SUB, RELATE, NEIGHBOUR, TRACE, VARY, FIX, REARRANGE, CONTRARY, RESPELL, RESPACE, FOCUS	48.83	2.22	23.5	1.07	33.28	1.51
13	WEIGH, PATTERN, BIBBLE, SELECT, SURVEY, SCAFFOLD, REDUCE, PINPOINT, SUB, TRACE, VARY, FIX, FOCUS	36.33	2.79	22.00	1.69	30.17	2.32
14	WEIGH, PATTERN, BIBBLE, SELECT, SURVEY, STRETCH, SCAFFOLD, REDUCE, PINPOINT, SUB, TRACE, VARY, FIX, FOCUS	39.83	2.84	22.50	1.61	31.17	2.23
15	PATTERN, CORRECT, BIBBLE, SELECT, CUT, SCAFFOLD, SPECIFY, CLEAVE, REDUCE, PINPOINT, BLOCK, SUB, TRACE, VARY, FIX, REARRANGE, CONTRARY, RESPELL, RESPACE, FOCUS	37.92	1.90	18.33	0.92	28.20	1.41
16	PATTERN, CORRECT, BIBBLE, SELECT, CUT, STRETCH, SCAFFOLD, SPECIFY, CLEAVE, REDUCE, PINPOINT, BLOCK, SUB, TRACE, VARY, FIX, REARRANGE, CONTRARY, RESPELL, RESPACE, FOCUS	41.42	1.97	18.83	0.90	29.20	1.39

Table 2: Table Showing Bates' Tactics for each of Belkin's ISS Conditions With Standard and Normalised Scores for each Interface

the graphs rise and fall in alternating pairs. This represents the alternation between recognise and specify (*Mode*) and is perhaps a predictable outcome for faceted browsers. By including more lessons learnt from the information seeking work on keyword search, such as relevance feedback, we might see a balance between these two conditions. Within each of these alternating pairs, the mSpace line marginally increases where the others fall. This indicates an increased support for meta-information (*Resource*). Considering individual browser lines, while RB++ and Flamenco follow a similar pattern for the first 8 ISS conditions, Flamenco notably improves this gap in the final 8 conditions. These two halves are made unique by the *Method* dimension and indicates that Flamenco provides better support for search, which is defined by having a known TO to exist: this might be knowing that an academic paper exists and just trying to find it. This significant increase, also sharper than mSpace, may be present due to the better support for making further selections and the lower support for changing selections.

The final pattern we draw from Figure 6 is shown every four conditions and is controlled by Belkin's *Goal* dimension. The Learn aspect of this dimension is shown by height differences between ISS1-4 and ISS5-8, and again between ISS9-12 and ISS13-16. This is characterised by the ability to see options in faceted browsers. The persistence of these options shown throughout to the user of mSpace is highlighted by the exaggerated difference in the first and third troughs compared to the second and fourth.

5.2 Implications for Design

There is a difference between unintentional and intentional design. Keyword search is clearly stronger in mSpace than Flamenco, but this is not part of intentional design. Subsequently, a feature upgrade could improve both by adding some information-seeking theory, such as relevance feedback. Other features may be intrinsic to a design: Facet Organisation in mSpace is one of these. Facet organisation has little use, but is possible, in both RB++ and Flamenco and is mainly used to bring popular facets to the forefront. However, the ability to order columns in mSpace supports a number of different Tactics. Multiple selection is also supported by both mSpace and RB++, but not supported in the Flamenco design. However Flamenco has purposely supported faster selections towards a TO and adding multiple selection would slow this down. In order to provide this feature, making a normal selection would require more Moves.

First, it is clear from Figure 5 that keyword search should still be integrated into faceted browsers to support users in both methods. These can be optimised by including the enhancements detailed by information-seeking theory. Second, while supporting a user in selection making is important for users who are confident of their target, optimising the ability to change selections and make multiple selections is important for users who are searching for a potentially relevant

but unknown object. Third, we suggest that representing the temporary hierarchies to the user is important for keeping track of user actions and understanding the effects of facets on each other. Although facets can be used in any order, the spatial ordering of them has been shown to support a number of search tactics. Fourth, the interoperability of viewing information pages with the browsing of facets is a key element in maintaining the search context. Fifth, the sorting and filtering of lists is an important part interacting with data. Sixth, the collection of information during search is important, especially when users are trying to locate relevant information rather than specific information. Finally, previewing the affects of actions is important for making decisions in search and should be shown as soon as possible.

6. CONCLUSIONS

In this paper, we have made two contributions. First we have presented the application of a combination of models to evaluate three faceted browsers. Although these models have been designed to encompass elements of user search, applied in combination they can be used to identify the strengths and weakness of a browser. Second, by applying this evaluation to three interfaces we have then been able to quantify these strengths and weaknesses over: the support for tactics, the support provided by interface features, and the support for sixteen unique user conditions.

Both moves and well defined tactics, from the model of strategic search interaction defined by Bates [1], have been used to quantify the support for each tactic provided by the features of three faceted browsers: mSpace, Flamenco and RB++. These metrics have first been summed by tactic to show which of Bates' tactics are particularly supported by a browser. Second, by summing the metrics by feature, we can show the support provided by its implementation. Identifying weak features can promote changes and advances in implementation to support more tactics or reduce the moves required to achieve each tactic. Finally, by summarising and normalising these metrics into Belkin's model of Information-Seeking Strategies [3], we have identified particular strengths and weaknesses of the three faceted browsers in different search conditions.

The goal of our approach is to enhance the design phase of a system before expensive and complex user studies are employed to assess versatile systems that are increasingly difficult to compare. Evaluation in IR has focused on designing experiments that are: *insightful*, to assess the attributes, on which they focus, successfully; *affordable*, in respect to the cost of creating and running experiments; *repeatable*, so that others can build on results; and *explainable*, to guide subsequent improvements [9]. Our evaluation framework adheres to these four principles, and through simulating core user interactions, allows for refinements made to designs during their formative development. In some respects, our approach mirrors that of White *et al.* [14], but is more theoretically grounded.

Through our evaluation framework, we create a performance benchmark that is essential to drive advancements in search system development. The formative approach above uses established models that have been previously validated with user studies. We intend to further validate this combined application of models with the logs of our own user studies. Further, we intend to extend the framework to model more of Saracevic's stratified analysis of users [12]. For example, we wish to measure cognitive load, as we expect this to be a large factor in creating synergetic designs. Finally, we wish to formalise stratagems in our framework, so that Marchionini's suggested exploratory search activities can be analysed [10]. We anticipate that with such developments, this hybrid IR/HCI approach will be a useful design tool for the development of exploratory search interfaces. Further, it may be used to compare systems between experimental sites, and perhaps even bring search interaction back to the Text Retrieval Conference.

7. REFERENCES

[1] M. J. Bates. Where should the person stop and the information search interface start? *Inf. Process. Manage.*, 26(5):575–591, 1990.

[2] M. J. Bates. The cascade of interactions in the digital library interface. *Inf. Process. Manage.*, 38(3):381–400, 2002.

[3] N. Belkin, C. Cool, A. Stein, and U. Thiel. Cases, scripts, and information-seeking strategies: On the design of interactive information retrieval systems. *Expert Systems with Applications*, 9(3):379–395, 1995.

[4] S. K. Card, T. P. Moran, and A. Newell. The keystroke-level model for user performance time with interactive systems. *Commun. ACM*, 23(7):396–410, 1980.

[5] C. Cool and N. J. Belkin. A classification of interactions with information. In *Proc. CoLIS 4*, pages 1–15, 2002.

[6] M. A. Hearst. Next generation web search: Setting our sites. *IEEE Data Engineering Bulletin: Special Issue on Next Generation Web Search*, 2000.

[7] M. Hildebreand, J. van Ossenbruggen, and L. Hardman. /facet: A browser for heterogeneous semantic web repositories. In *Proc. ISWC*, 2006.

[8] S. Kriewel. Finding and using strategies for search situations in digital libraries. *Bulletin IEEE TCDL*, 2(2), 2006.

[9] B. Liu and D. W. Oard. One-sided measures for evaluating ranked retrieval effectiveness with spontaneous conversational speech. In *Proc. ACM SIGIR*, pages 673–674, New York, NY, USA, 2006. ACM Press.

[10] G. Marchionini. Exploratory search: from finding to understanding. *Commun. ACM*, 49(4):41–46, 2006.

[11] S. E. Robertson and M. M. Hancock-Beaulieu. On the evaluation of ir systems. *Inf. Process. Manage.*, 28(4):457–466, 1992.

[12] T. Saracevic. The stratified mode of information retrieval interactino: Extension and applications. In *Proc. Info. Science*, volume 34, pages 313–327, 1997.

[13] R. W. White, B. Kules, S. M. Drucker, and m.c. schraefel. Introduction. *Commun. ACM*, 49(4):36–39, 2006.

[14] R. W. White, I. Ruthven, J. M. Jose, and C. J. V. Rijsbergen. Evaluating implicit feedback models using searcher simulations. *ACM Trans. Inf. Syst.*, 23(3):325–361, 2005.

[15] M. L. Wilson and m.c. schraefel. mspace: What do numbers and totals mean in a flexible semantic browser. In *Proc. SWUI*, 2006.