Eckstein, Raiko; Henrich, Andreas; Weber, Nadine

LFRP-Search: Multi-Layer Ranked Visual Faceted Search : an Approach to Cope with Complex Search Situations

Date of secondary publication: 24.02.2025

Accepted Manuscript (Postprint), Conferenceobject

Persistent identifier: urn:nbn:de:bvb:473-irb-1066163

Primary publication

Eckstein, Raiko; Henrich, Andreas; Weber, Nadine (2010): LFRP-Search: Multi-Layer Ranked Visual Faceted Search : an Approach to Cope with Complex Search Situations, in: Sung Y. Shin, Sascha Ossowski, Michael Schumacher, u. a. (Ed.), SAC '10 : Proceedings of the 2010 ACM Symposium on Applied Computing, New York: ACM, pp. 1713–1717, doi: 10.1145/1774088.1774455.

LFRP-Search: Multi-Layer Ranked Visual Faceted Search – An Approach to Cope with Complex Search Situations

Raiko Eckstein
Chair for Media Informatics
University of Bamberg
Germany
raiko.eckstein@uni-bamberg.de

Andreas Henrich
Chair for Media Informatics
University of Bamberg
Germany
andreas.henrich@uni-bamberg.de

Nadine Weber
Chair for Media Informatics
University of Bamberg
Germany
nadine.weber@uni-bamberg.de

ABSTRACT

In enterprise search scenarios information needs are quite diverse. If users know exactly which items they are looking for, they need support in known-item searches. But often information needs are vague and unclear, which means that users cannot explicitly define the search criteria that specify their search request. In these cases, exploratory search approaches are necessary to support users in interactively refining their search queries. Our approach for a retrieval system for complex search situations can be characterized by its four constituent parts: The approach deals with the heterogeneity of potential target objects when performing a search considering multiple artifact layers (e. g. *projects*, *products*, *persons*, and *documents*). The basic search paradigm applied is faceted search which is well-suited for exploratory interactive retrieval. To cope with result sets of different granularity, we include ranking facilities based on facet values as well as Query-by-Example (QbE) functionalities. Users can easily state their priorities visually by using preference functions. Finally, parallel coordinates (PC) are used to visualize the characteristics and dependencies of (intermediate) results in order to provide users with a deeper understanding of the data under investigation. These four cornerstones of our approach are reflected in the acronym LFRP-search (Multi-Layer Faceted Search with Ranking using Parallel Coordinates). The main contribution of this paper is the semi-formal description of the query options.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval; H.5.2 [**Information Interfaces and Presentation**]: User Interfaces

Keywords

Information Retrieval, Retrieval Model, Multi-Criteria, Visualization, Faceted Search, Parallel Coordinates

1. INTRODUCTION

The ability of companies to access existing knowledge and information is crucial for commercial success. Studies in different engineering domains have shown that users often have difficulties in phrasing the right search query due to complex search situations. Users rather approached colleagues than using existing search functionalities as these human experts helped them to refine or rephrase their unclear information needs in a direct dialog [1].

In [6], Hawking outlines challenges and opportunities for the field of enterprise search. Amongst others, these are the effective ranking over heterogeneous collections, the exploitation of the search context within enterprises and the search for different artifact types. Our approach tries to tackle these problem areas. In this publication, we cover the consideration of different artifact types in the iterative process of refining the search query. We consider heterogeneous interlinked artifact types, like *projects*, *documents*, *products*, and *persons*. In particular, the latter topic gained research interest in the field of expert search [3]. Additionally, we cover ranking functionalities based on user preference functions which can be set for facet selections. Queries in the enterprise search domain often tend to be vague, i. e. users cannot completely specify what their information need is. In situations like this users should be supported in refining queries by an interactive exploratory search interface. LFRP-search provides a means to initially obtain an overview of the existing data landscape. In a next step, users can interactively add and remove search and ranking criteria and immediately assess the implications of these constraints on the search results.

In [5] we introduced the prototypical query interface and described the interactive query formulation employing a PC plot. The main contribution of this publication is the provision of the formalization of the query model which describes the available query options. The query model is described by our LFRP-search approach and realized in the user interface.

After an overview of relevant related work, we introduce our visual approach to interactive query refinement implemented in the LFRP-search framework. Then two different search use cases on multiple artifact layers are outlined and followed by a semi-formal specification of the query model.

2. RELATED WORK

Often, information needs are vague and cannot be specified in a concrete way. Information seeking approaches deal-

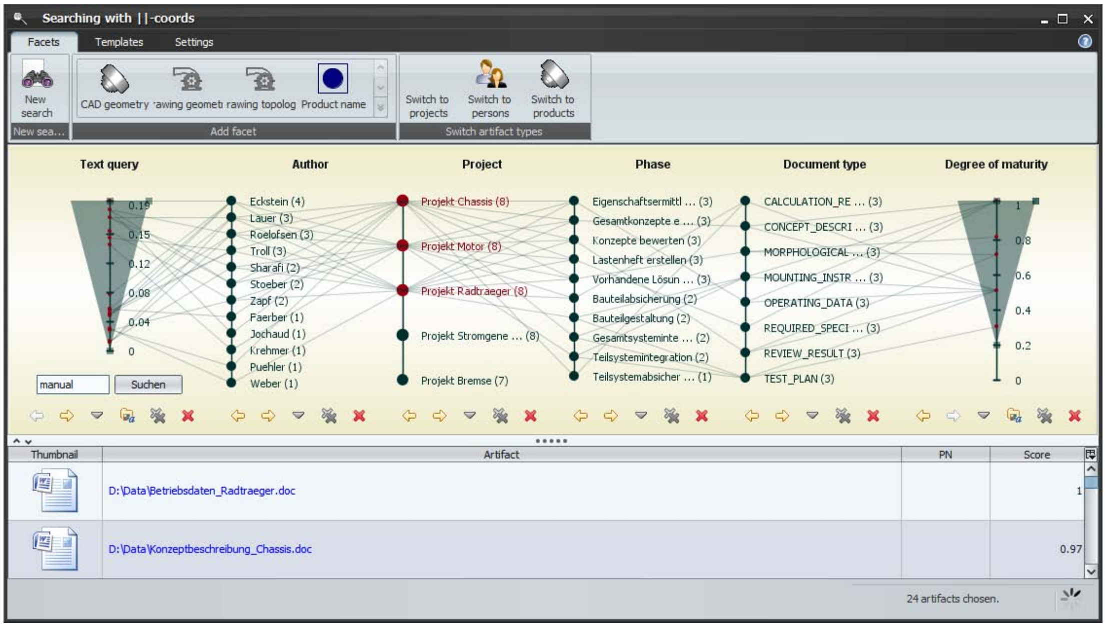

Figure 1: Screenshot of the user interface

ing with likewise information needs are subsumed under the notion *exploratory search* [8]. Users need to solve complex search problems in an interactive way, i. e. they need search tools which support them in pursuing different search strategies and helping them to interactively refine their queries.

Faceted search is a paradigm which supports exploratory searches. It permits the navigation along hierarchical faceted categories which contain the artifacts from the collection [9]. Each artifact is described by a set of facets which are assigned facet values. Examples are the *author* of a document or the *project* a product was developed in. The main advantage of the faceted search paradigm is the provision of dynamically generated *query previews* which help users to assess how many artifacts belong to a certain facet value. Facet values which are not available due to previous selections (i. e. no artifacts in the result set have these values) are not displayed to users preventing empty result sets in subsequent selections. In the academic domain several prototypical faceted search implementations have been developed. One of the first was the open-source project Flamenco (short for FLexible information Access using MEtadata in Novel COmbinations) [9].

Our search approach supports exploratory searches by offering visual insights about dependencies and relations between artifacts and their describing metadata. We employ the visualization technique of *PC* to visualize dependencies between facet values and the artifacts from the result ranking [7]. PC project multiple dimensions into the two-dimensional (drawable) space. Inselberg achieved this by drawing the dimensions as axes of equal height parallel to each other. Therewith, it is possible to visualize an arbitrary high number of dimensions.

We employ the well-known vector space model (VSM) [2] for textual QbE-queries. For feature-based similarity searching on 3D models, we refer to [4] which provides an overview of applicable approaches.

3. INTERACTIVE QUERY REFINEMENT

LFPR-search is a general approach to support knowledge workers in complex search situations. Our approach consists of both a semi-formal description of the supported query model, i. e. the supported search queries (cf. Section 5), and an intuitive user interface supporting knowledge workers by providing an interactive and flexible search tool enabling an interactive and incremental search query refinement. The graphical user interface was introduced in [5].

Figure 1 shows an example screenshot of our search engine user interface after several facets have been chosen and selections on these facets have been applied. The user interface is divided into three horizontal parts. The topmost part consists of a menu component which mainly enables users to add additional facets to the PC plot. The list of available facets is adjusted after each query refinement step made by the users. This *dynamic facet provision* relies on an available artifact type and facet schema which is company-specific. Roughly spoken, this schema defines the inheritance hierarchy together with the discriminating attributes (called *discriminating facets* in our context). As soon as a value for a discriminating facet is selected the additional facets of the corresponding subtype become available in the "Add facet" area.

The central part of the user interface contains the PC plot consisting of a dynamic number of vertical axes which represent the chosen facets with the currently visible facet values. The multi-dimensional faceted metadata is mapped to a two-dimensional area which can be charted and comprehended easily. Each axis can be adjusted by a control panel situated below the axis which enables swapping, removing and weighting of the corresponding facet. In addition, each artifact in the search result is plotted into the PC plot as a polyline intersecting all axes at the appropriate facet values. The connecting polylines of two adjacent axes can be used to determine relationships and correlations between facets and artifacts.

Users have the option to add facets to the PC plot in order to use these facets for further analysis of the search result. Moreover, they can use the axes to specify sub-queries. By selecting specific facet values of one axis users filter the current search result set to those artifacts which fulfill this constraint.

The lower part of the user interface simply shows the search results in a list view which can be sorted and allows users to open the search results in a host application or e. g. a document management system.

We harmonized the query statement by integrating the different search types in the PC plot. Users simply need to understand the notion of an axis in the plot referring to a sub-query for their information need which they combine with other axes.

4. THE LAYER APPROACH

One of the main goals of LFRP-search is the support for the retrieval of different artifact types like *documents*, *products*, *persons*, and *projects*. These artifacts are interconnected by different types of associations which should be harnessed by users while stating queries. Each of these artifact types can be part of a result set that can be restricted by filter conditions depending on the same or other artifact types. An artifact type corresponds to the notion of a layer. A layer is defined by a single artifact type and contains all indexed artifacts of this type.

Our model distinguishes intra-layer and inter-layer connections. Intra-layer connections are relationships which exist among the artifacts of one layer. For instance, *projects* can consist of sub-projects what is denoted by a *is-subproject-of* relationship. Additionally, we consider relationships that exist between the individual layers. For instance, *documents* are created, revised or needed in a certain *process phase* which itself is linked to a certain *project*. Documents are accessed and handled by a specific *role* which has a connection to *persons*. In each *project*, specific *products* are developed. With the help of these connections a transition between the layers can be enabled which will be detailed below.

We distinguish two different use cases in the layer concept. In the first use case, users search for artifacts on one layer by defining their search criteria as sketched in Section 3. To further narrow down the search results, users can apply facets stemming from directly connected layers. Considering the search on the *document* layer, users might decide that they need to filter by *product name*, which is a product facet. In the examined domain, documents and products are linked by a *describes* relationship. Our search engine aggregates facets of directly connected artifacts by automatically generating a nested query which takes the current artifacts from the search result—in our case documents—and retrieves all linked products. In the next step the facet *product name* is determined for this set of products. The displayed search result only contains documents which were filtered by facets stemming from a different layer.

The second use case covers the transition between layers of artifacts. Users filter the search results by facet selections similar to the first use case. In the next step, they want to change the artifact layer based on their current results. A practical example of this approach is a search in the document layer where users begin with a QbE-query consisting of a similar document. Based on the returned result users can do further filtering based on document facets. If they are satisfied by the result set, they might want to see all related products which are described by the documents in the result set. The search engine now determines the connected products according to the linkage stored in the index. Roughly spoken, LFRP-search takes the current result set on the origin layer, navigates along the specified relationships and builds the new result set from the destination objects on the target layer. Users can apply additional selections on the result set and filter further. Additionally, it is possible to return to the initial selection on the origin layer to widen or narrow the selections which are applied to the destination layer.

5. QUERY MODEL FORMALIZATION

This section covers the query model of our LFRP-search approach described in Sections 3 and 4. Formally, a query in the model consists of an arbitrary number of sub-queries. Each sub-query represents a facet selection for one of the axes in the PC. Due to the support of arbitrary filter criteria and different types of similarity searches, each sub-query has to adhere to a certain query semantics.

Attribute facets are aspects of an artifact which usually are determined during indexing time. Attribute facets can either be *single-valued* or *multi-valued*.

In addition, each facet is assigned a specific facet type which is based on the level of measurement that represents the scale of the possible values of this facet. We distinguish the three facet types *nominal*, *ordinal* and *ratio*. This distinction is necessary with respect to the visualization of the respective axis as well as to the expressiveness of the query.

A *nominal* scale is applicable for facet values for which no ordering can be defined, e. g. for the authors of a document. Facets with an *ordinal* scale can be ordered, but no distance meaning is supported between the facet values. If the values can be ordered and a distance between the facet values is defined a *ratio* scale type facet is given.

For the illustration of the possible queries we employ an SQL-like notation. We chose this approach because this notation of the supported queries also can be used in communication with power users.

5.1 Selections for One Artifact Type

Users can combine multiple facets in their queries. Each axis in the PC plot, for which a selection is made, represents a sub-query Q_i addressing the respective facet which is combined with the other sub-queries by a conjunction. This leads to the following SQL-like expression:

$$\begin{aligned}&\textbf{SELECT} * \textbf{FROM } \textit{layer}\\&\textbf{WHERE } Q_1 \textbf{ AND } Q_2 \textbf{ AND } \ldots \textbf{ AND } Q_n\end{aligned} \qquad (1)$$

Here, n equals the number of facet axes which are currently added to the plot and for which user selections are made. Facets in the plot without selections do not affect the query but support users in getting a better insight into the search results.

Each facet sub-query Q_i is specific for the facet type it is built upon. For *ordinal* or *nominal* facets both single-valued and multi-valued selections are possible. If we assume single-valued facets where only one value can be assigned to an artifact, the disjunction is the only way to combine several selected values for one facet. For multi-valued facets—e. g.

SELECT doc.DocType, doc.DegreeOfMaturity, doc.Author, product.ProductName **FROM** doc **JOIN** product **ON** doc.describesProduct = product.productId **WHERE** doc.DocType = "Requirements Specification" **AND** doc.DegreeOfMaturity > 0.8 **AND** doc.Author = "Smith" **AND** product.ProductName = "Engine"

Figure 2: Example query statement for facet selections from multiple artifact layers

the document facet *author*—the conjunction is applicable and filters out documents which do not comply with the requirements, i. e. only documents that were jointly created by these authors are returned. Applying a disjunction for several authors weakens this criterion and returns documents where at least one of these persons is amongst the authors.

The distinction from above results in three different cases. In the following statements, a sub-query Q_i describes selections on the facet F_i with values v_j.

For single-valued facets, the SQL-like expression for Q_i utilizing the disjunction can be formulated as:

$$Q_i \equiv F_i \textbf{ IN } (v_1, v_2, \ldots, v_n) \tag{2}$$

The values of multi-valued facets are set-valued which usually would demand that the according database tables are normalized. Instead, we assume set-valued entries in the database and introduce the query notation $v_j \in F_i$ that demands that the facet value v_j is included in the facet F_i. The SQL-like expression for Q_i can be formulated as:

$$Q_i \equiv (v_1 \in F_i) \textbf{ OR } (v_2 \in F_i) \textbf{ OR } \ldots \textbf{ OR } (v_n \in F_i) \tag{3}$$

For multi-valued facets additionally the conjunction is possible for Q_i according to:

$$Q_i \equiv (v_1 \in F_i) \textbf{ AND } (v_2 \in F_i) \textbf{ AND } \ldots \textbf{ AND } (v_n \in F_i) \tag{4}$$

Multi-valued selections on ratio scale facets can be realized by allowing users to choose one or more inclusive intervals of the values they are particularly interested in. The first axis in Figure 4 shows this approach where users restrict the search results to documents whose degree of maturity lies between 60% and 100%. A use case for multiple intervals is the comparison of products from different price brackets.

More formal an interval query Q_i could be written:

$$\begin{aligned} Q_i \equiv\ & F_i \textbf{ BETWEEN } v_1 \textbf{ AND } v_2 \textbf{ OR} \\ & F_i \textbf{ BETWEEN } v_3 \textbf{ AND } v_4 \textbf{ OR } \ldots \textbf{ OR} \\ & F_i \textbf{ BETWEEN } v_n \textbf{ AND } v_{n+1} \end{aligned} \tag{5}$$

Additionally, we provide users with *query previews* which consist of the number of artifacts the search engine would return if the users select the facet value in addition to already conducted selections.

The following statement shows how the query preview for the facet F_i with the value v_j is determined in an SQL-like notation. The facet sub-queries Q_2 to Q_n are the selections made prior to the current query preview calculation which have to be considered.

$$\begin{aligned} & \textbf{SELECT COUNT } (*) \textbf{ FROM } layer_i \textbf{ WHERE} \\ & F_i = v_j \textbf{ AND } Q_2 \textbf{ AND } \ldots \textbf{ AND } Q_n \end{aligned} \tag{6}$$

Without any selection in the PC plot, the sum of all query previews of a single-valued facet is the number of all artifacts in the considered $layer_i$. For multi-valued facets the sum is greater than or equal to the number of all artifacts of $layer_i$, since multiple facet values can be assigned to one artifact.

5.2 Selections in Multiple Artifact Layers

If multiple layers are considered in one query, the two use cases described in Section 4 have to be distinguished. In the first use case the query is composed of facet selections from directly connected artifact layers, but the returned search result only contains elements of the initial artifact layer. A layer is defined to contain all indexed elements of one artifact type and is identified by $layer_i$ in the following. Facets of a layer i are referred to as $F_{i,j}$ where j is used to distinguish different facets of an artifact type. In general, an inter-layer query can be notated according to the following statement.

$$\begin{aligned} & \textbf{SELECT } F_{i,j}, F_{i,j+1}, F_{k,l}, \ldots \textbf{ FROM } layer_i \\ & \textbf{JOIN } layer_k \textbf{ ON } F_{i,m} = F_{k,n} \textbf{ WHERE} \\ & Q_{i,j} \textbf{ AND } Q_{i,j+1} \textbf{ AND } Q_{k,l} \textbf{ AND } \ldots \end{aligned} \tag{7}$$

The three exemplary facet sub-queries $Q_{i,j}$ correspond to the facets of different artifact layers given in the **SELECT** clause. The index i denotes the number of the layer and j is the number of the facet on layer i.

The example query in Figure 2 comprises the document facets *document type*, *degree of maturity*, and *author* and the *product name* facet from the product layer. The query shows selections on both artifact layers. Additionally, the primary / foreign key relationship *describesProduct* between the document and the product layer is harnessed for the result determination.

The second use case consists of an initial query on one artifact layer whose search results serve as initial constraints for artifacts of a directly connected layer. Selections are conducted in two steps. The initial selection on the first layer is stated in the first step. The facet sub-queries $Q_{i,j}$ from layer i can be defined according to Section 5.1.

$$\begin{aligned} & \textbf{SELECT } F_{i,j}, F_{i,j+1}, F_{i,j+2} \textbf{ FROM } layer_i \\ & \textbf{WHERE } \ Q \end{aligned} \tag{8}$$

$$Q \equiv Q_{i,1} \textbf{ AND } Q_{i,2} \textbf{ AND } \ldots \textbf{ AND } Q_{i,n} \tag{9}$$

The actual layer switch happens in the second step where the relationship described by facet $F_{i,j+1}$ is used as a sub-selection. The two layers i and k are connected by a relationship defined by the facet $F_{i,j+1}$ from layer i and by the facet $F_{k,l}$ from layer k.

$$\begin{aligned} & \textbf{SELECT } * \textbf{ FROM } layer_k \textbf{ WHERE } F_{k,l} \textbf{ IN} \\ & (\textbf{SELECT } F_{i,j+1} \textbf{ FROM } layer_i \textbf{ WHERE } Q) \end{aligned} \tag{10}$$

Similar to the previous use case, the example shown in Figure 3 harnesses the relationship between the document and the product layer called *describesProduct*. In terms of relational databases the connection between the two layers would be represented by a primary / foreign key relationship. When further facet selections on $layer_k$ are added, these can be easily added in the outer WHERE-clause in statement (10).

1^{st} step: Filter on document layer:

SELECT doc.DocType, doc.DegreeOfMaturity, doc.Author **FROM** doc **WHERE** Q

Q $\equiv$ doc.DocType = "Requirements Specification" **AND** doc.DegreeOfMaturity > 0.8 **AND** doc.Author = "Smith"

2^{nd} step: Switch to product layer via the relationship describesProduct:

SELECT $*$ **FROM** product **WHERE** productId IN (**SELECT** doc.describesProduct **FROM** doc **WHERE** Q)

Figure 3: Example query statements for switching the artifact type

5.3 Influencing the Ranking

LFRP-search supports ranking functionality based on facet selections. Users are able to augment the query by the notion of *preference functions*. Users can "draw" functions onto each axis in the PC to prioritize artifacts different according to their facet values for a specific facet. The statement of these functions is completely optional for the users.

According to the SQL notation the artifact ranking is determined based on the **ORDER BY** clause. The order is conducted according to the score of an artifact a_j which is determined as follows:

$$score(a_j) = \sum_{i=1}^{n} \alpha_i \cdot f_i(x_{i,j}) \text{ with } \sum_{i=1}^{n} \alpha_i = 1 \qquad (11)$$

Here, α_i is the weight for the facet i, which can be set by the users via the control panel to influence the weighting of the functions for each facet (the default is $1/n$). By this means users can define higher or lower emphasis for the single ranking criteria. $f_i(x_{i,j}) \in [0, 1]$ gives the preference of the users for the value $x_{i,j}$ of facet i for artifact a_j as graphically defined by them.

The function f_i can be defined completely according to preferences of the users in a visual way. But for easier comprehension the system introduces some simple template functions which are needed more often like the triangular and quadrilateral shapes found in Figure 4. It should be noted that the described ranking functionality allows us to seamlessly integrate Query-by-Example and keyword-based search. As an example, the screenshot in Figure 1 shows a text query based on the VSM on the leftmost axis.

The combination of the above mentioned query selection options offers the users a toolset to formulate complex queries which can be stated interactively with the help of the user interface described above. Users do not have to formulate the complete query in the beginning, but can add new criteria after they assessed the interim search results or remove criteria if the filtering was too restrictive. Referring to the SQL statement (1) this corresponds to changing one of the sub-queries Q_i or removing the selections of a sub-query.

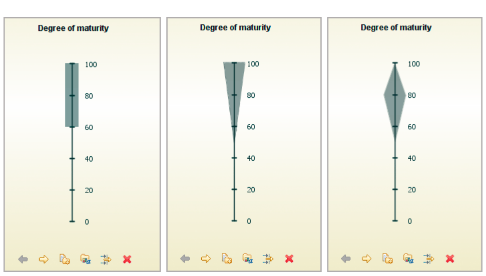

Figure 4: Example user preference functions

6. CONCLUSION AND FUTURE WORK

We have proposed LFRP-search as an interactive search solution for complex search scenarios. Our approach combines the established techniques of faceted search, ranking, and PC in a seamless and powerful tool for exploratory retrieval tasks. It has been developed in a research project with industrial partners from the engineering domain. Obviously, there are many open points for future work. We develop a concept distinguishing different user types. Experts can use the full functionality of the approach to build search templates for situations with a high potential for information needs. These are integrated into a process management solution and can be used by "normal" users for interactive searches. Finally, the efficient implementation of the framework and the useful distribution of the computations between frontend and backend are interesting aspects.

7. REFERENCES

[1] T. J. Allen. *Managing the flow of technology: Technology transfer and the dissemination of technological information within the R&D organization.* MIT Press, Cambridge, Mass., 7. print. edition, 1995.

[2] R. Baeza-Yates and B. Ribeiro-Neto. *Modern information retrieval.* Addison-Wesley, Harlow, 2008.

[3] P. Bailey, N. Craswell, A. P. de Vries, and I. Soboroff. Overview of the trec 2007 enterprise track. In *Proceedings of the 16th Text Retrieval Conference (TREC 2007)*, Gaithersburg, USA, 2007.

[4] B. Bustos, D. A. Keim, D. Saupe, T. Schreck, and D. V. Vranic. Feature-based similarity search in 3d object databases. *ACM Comput. Surv.*, 37(4):345–387, 2005.

[5] R. Eckstein and A. Henrich. Visual browsing in product development processes. In The Design Society, editor, *Proceedings of the 17th International Conference on Engineering Design*, pages 183–194, 2009.

[6] D. Hawking. Challenges in enterprise search. In *ADC '04: Proceedings of the 15th Australasian database conference*, pages 15–24, Darlinghurst, Australia, Australia, 2004. Australian Computer Society, Inc.

[7] A. Inselberg and B. Dimsdale. Parallel coordinates: a tool for visualizing multi-dimensional geometry. In A. E. Kaufman, editor, *Visualization '90*, pages 361–378. IEEE Computer Soc. Press, 1990.

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

[9] K.-P. Yee, K. Swearingen, K. Li, and M. Hearst. Faceted metadata for image search and browsing. In *CHI '03: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 401–408, New York, NY, USA, 2003. ACM Press.