INTERNATIONAL CONFERENCE ON ENGINEERING DESIGN, ICED'09
24 - 27 AUGUST 2009, STANFORD UNIVERSITY, STANFORD, CA, USA

VISUAL BROWSING IN PRODUCT DEVELOPMENT PROCESSES

Raiko Eckstein[1], Andreas Henrich[1]
(1) University of Bamberg

ABSTRACT
In modern product development a wealth of knowledge is developed and stored in electronic form which leads to challenging retrieval tasks. Opposing to that, companies need to reduce development times and costs to stay competitive. Therefore, it is necessary to reuse existing knowledge in the company that comprises existing parts and components amongst others. This paper introduces an exploratory approach to support engineers in retrieval tasks in product development. We present a search engine prototype which employs data visualization techniques to expand the idea of browsing and faceted search. We propose the usage of parallel coordinates plots known from multi-dimensional data visualization as a method for issuing faceted search queries. Next to the higher expressiveness of the possible queries which can be stated, the introduced solution offers better visual insight about the artifacts from product development. Additionally, we introduce ways to influence rankings by user preference functions which help weighting the search criteria.

Keywords: Product Development, Information Retrieval, Visualization, Faceted Search

1 INTRODUCTION AND PROBLEM STATEMENT
In the field of product development much of the product knowledge is created and stored in digital form. In future projects design engineers are urged to reuse that knowledge to avoid (expensive) duplicate work. For instance, that knowledge comprises different product models, best practice documents, parts, and components. If companies manage to increase the level of reuse in the company, the development cycles can be shortened and the development costs can be lowered. Several problem areas make the retrieval for the sought parts difficult. A product is not described in a single document, but spread over many different document types which are often stored in several management systems (e.g. product data management, document management and project management tools). The engineer starting a retrieval task has to consult several systems to attain an overview over the data collection. Additionally, engineers face a variety of different, but connected artifact types during their work. An artifact type is defined as a certain type of a search result. The engineer not only wants to retrieve simple *documents*, but his information needs also center on *products*, *projects* and *persons / experts*. The importance of different artifact types and document types varies during product development processes. More specifically, the information need of an engineer depends on the task in the product development process he or she is working on. Some information needs of an engineer can be characterized as vague, which means that the user cannot easily spell out what he or she is looking for. A retrieval system should support the user in stating vague queries by allowing an exploratory browsing access to the data collection.
In this paper we introduce an approach aiming at supporting engineers in a better provision of information. Our research focuses on context-based information retrieval where user context is used to provide more precise search results which can lead to a higher degree of reuse of existing knowledge.
In [1], we introduced an integrated context model for the domain of product development and showed benefits of utilizing that additional information. The model comprises the following seven dimensions: *user*, *document*, *product*, *process*, *task*, *project*, and *company*. For instance, the *process dimension* describes the product development process with the process phases, documents, methods, and responsible roles. The documents in the process can be described more precisely with available information about the creation phase, revision phase, author(s), deployed methods, etc. Furthermore, the user's involvement in product development can be characterized more accurately. Information about his past and current projects as well as roles and tasks the user is responsible for, can be

provided. This information should be incorporated in document matching when the search results are determined.
The context model enables us to describe both the *user context* and the *document context* which later have to be matched to get search results for specific user information needs. The main problem area of this matching process lies in the situation-specific information needs which are partially described by contextual information. If a product developer is creating a 3D model of a product he might search for an applicable part he can reuse in his current task. A search engine now can return 3D-CAD documents which comprise the models the engineer can include in his task. But additionally, the user might need data sheets, test and simulation results for that part. Also general documents like guidelines, legal requirements and method descriptions might be beneficial for the engineer. Thus, the user's current context described by the process dimension does not precisely confine the information need of the user. The additional documents were created in a different context (e.g. later in the process, by a different role, different kind of project, etc.) and would not appear if we would enforce a strict matching of the user's context model and the retrieved documents' context models. Hence, a search tool for the engineering domain has to enable the use of context information but cannot be rigid in this respect.
After the gathering of this contextual knowledge the search mechanism evolves from a single-criteria matching problem (i.e. for instance comparing the textual document contents with the query) to a multi-criteria matching problem which introduces several difficulties concerning the processing of this additional information.
This includes the determination of weights for the different search criteria, a description of the dependencies between the criteria and the decision, whether a criterion acts as a simple filter or if it influences the artifact ranking in the search result. Different solutions are conceivable including methods of machine learning [2] and manual determination of the weights by experts. But since the researched domain and the scope of the search engine are quite broad, we claim that no applicable automatic definition of the weights can be achieved as the information needs in that special scenario are too diverse. We are of the opinion that only the user himself is able to be decisive about the search criteria. In order to support the user, the user interface has to be expressive and intuitive. The user should be able to define which contextual factors the search engine should take into account for processing the query and calculating the result ranking. Of course, this shifts much of the responsibility to the user and can lead to a bad acceptance if user training is omitted or insufficient.
Nevertheless, we think that this approach is feasible as the user needs this flexible and precise access to the available information including context information due to his vague information needs. That access can be subsumed under the broader notion of *exploratory search* which depicts a more interactive approach to find relevant information [3]. In [4], we introduced an *interactive retrieval model for complex search situations* as found in product development processes. The model combines approaches from different areas such as *faceted search*, *similarity search* and *information visualization*. The present paper extends [4] in two directions. First, we show how this model can be realized in a prototype for visually enhanced faceted browsing. Second, we introduce how a ranking functionality can be included in our faceted search by the statement of user preference functions.
The present paper is organized as follows. The next section covers related work for the topics covered by our search approach. In Section 3 we introduce our visual approach to faceted search and explain our search prototype incorporating a parallel coordinates plot. In Section 4 we address issues and problems of our approach and provide possible solutions. The paper concludes with a summary of the contributions of this publication.

2 RELATED WORK

Several research projects studied context enhanced provision of information in the domain of product development. The European integrated project VIVACE researches the retrieval of engineering knowledge from a repository using case-based reasoning systems [5]. Karnik et al. introduced the *Design Navigator System* which targets the management of design information including the search for design rationales for reuse in other projects by means of best practices. The search functionality supports browsing, design rationale search and geometry-based search amongst others. However, context information is not incorporated [6].
Our approach follows the search paradigm of *Browsing* which is defined by Marchionini and Shneiderman as "an exploratory, information-seeking strategy that depends on serendipity. It is

especially appropriate for ill-defined problems and for exploring new task domains." [7]. In contrast to pure information retrieval, browsing is characterized by the fact, that the user does not know what results he might find in the end. Usually, the underlying information (including metadata) is organized in classification schemes which are used to provide additional information and descriptions for documents. The user applies these schemes to navigate to the intended search results.

Marchionini distinguishes three different information seeking activities: *Lookup*, *Learn* and *Investigate.* They differ in complexity and cognitive efforts of how the retrieved information is handled [8]. The traditional *information lookup* is the most basic kind of search task and mainly used in fact retrieval which is also called *known-item search*. Assuming more complex information needs this approach is not feasible anymore. The more complex search tasks of *Learning* and *Investigating* are subsumed under the broader notion of *exploratory search*. In addition to the concept of browsing, exploratory search expands this paradigm with more interactive approaches to find the relevant information [3]. Exploratory search emerges as a new paradigm where the user is catered with tools which support *learning* and *investigating* to a certain degree [8]. As in the saying "The journey is the reward.", the user is acquiring knowledge along the way while he is searching.

Searches in mechanical engineering are often more complex than *lookup searches* as the engineer needs to look for applicable solutions he can use or adapt. He is looking for solution principles which he wants to incorporate into his current project. Therefore, the notion of *exploratory search* describes the way how the user retrieves information and lastly knowledge much better than classical fact retrieval. Engineers face different types of projects. In a *new design* they create a product which did not exist before. Nevertheless, engineers in this situation are searching for principles from other products or projects they can reuse. Having to deal with an *adaptive design* [9], engineers have to adapt an already existing product to new requirements. Search tasks for these kinds of projects include the search for solutions for those specific new requirements based on experiences made in past projects. Those solutions clearly exceed the notion of *known-item search*.

The *Faceted Search* paradigm is one embodiment of the combination of browsing and exploratory search. It enables the user to navigate along hierarchical faceted categories which contain the artifacts from the document collection [10]. Each artifact is assigned facet values for each facet. A facet describes an aspect of an artifact, e.g. the author, the file type, the creation phase, etc. Facets can be distinguished in single-value and multi-value facets. Whereas the former only allows a document to belong to one facet value, the latter allows multiple assignments of facet values for one facet. Usually, those categories have to be defined manually which differs from clustering approaches [11, 12] for search results. Hearst elaborates on the topic clustering vs. hierarchical faceted categories in [13] and concludes that the predictability of the categories approach is more accepted amongst users. The research of Hearst's group culminated in the open-source project Flamenco[1] (short for FLexible information Access using MEtadata in Novel COmbinations) [10, 14].

As the amount of information is growing tremendously nowadays, an intuitive way has to be found to access this information easily. Our approach combines the classical search paradigm with faceted search and some ideas from the domain of information visualization.

An interesting application of multi-dimensional data visualization is the concept of the *parallel coordinates plot* developed by Alfred Inselberg [15, 16]. Although, developed under the circumstances of non-existence of visualizations for multi-dimensional geometries, parallel coordinates (||-coords) later were incorporated in data analysis for multi-dimensional data [17, 18]. ||-coords project the multiple dimensions into the 2-dimensional (drawable) space. Inselberg achieved that by drawing the dimensions as axes of equal height parallel to each other. Therewith, it is possible to visualize an arbitrarily high number of dimensions. Usually, the axes are aligned on the horizontal line. For continuous values an axis usually has a linear scale, though a logarithmic scale is possible as well. So each dimension is represented in a parallel line. Each data record is plotted as a polyline intersecting all axes at the appropriate values. Applications of parallel coordinates plots can e.g. be found in the geographic domain [19] and in exploratory volume visualization [20].

[1] http://flamenco.berkeley.edu/

3 ||-COORDS AS A SEARCH INTERFACE

This section introduces our search engine prototype and discusses ways of interaction with the user interface. It is shown how to combine similarity search (e.g. query-by-example or keyword queries) and a customizable weighting of contextual information and other facets.

We expand the search paradigm of *faceted search* [21] in a visual way to enable users who cope with complex search tasks to state complex queries visually. For that, we employ a *parallel coordinates plot* and use this visualization as a query interface to display the facets and the according user selections. Additionally, our search prototype aims at making dependencies and correlations between different facets of an artifact visible.

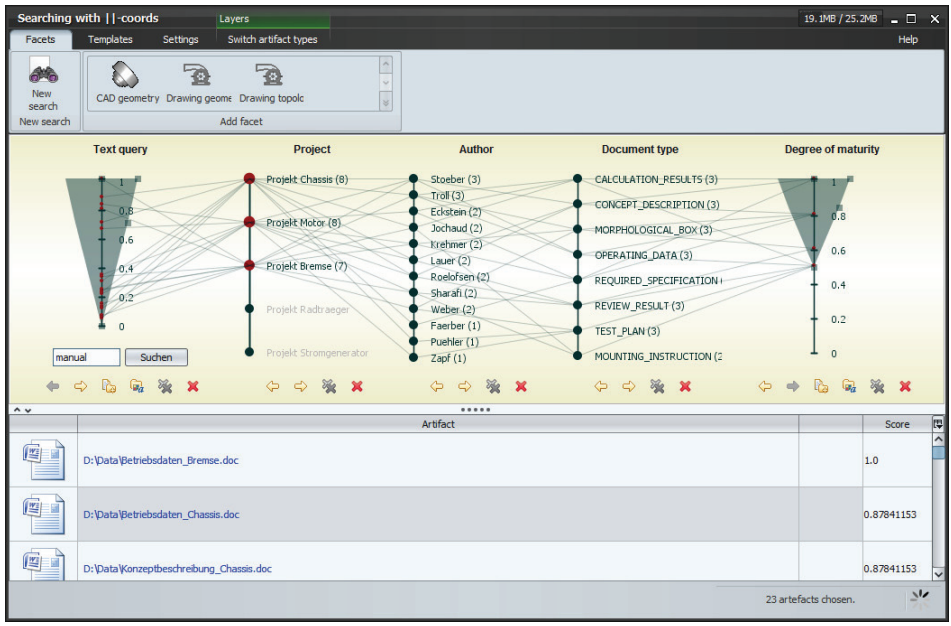

Figure 1. Example view of the search interface

Figure 1 shows a screenshot of our current graphical user interface search prototype after several selections were conducted. The user initially started with a textual *keyword query* for the term "manual" as he was interested in documents of that kind. As this query is not very specific the user decides that he needs to restrict the returned results by additional filter criteria and adds the facet *project* to the parallel coordinates plot and selects three projects (highlighted in red) in the user interface. To see who is responsible for the now remaining documents and of which type they are, the engineer adds the facets *author* and *document type* to the plot. The last addition to the plot is the facet *degree of maturity* as the user wants to restrict the search results to more mature documents. The user is applying a user preference function to weight more mature documents higher in the result list and filter out documents having a degree of maturity which is less than 50%. The upper area consists of a ribbon component (as found in newer versions of Microsoft Office) which controls the prototype. In the ribbon band *Facets* the user can choose the facets he wants to apply in his current search task. The constantly updated facet list ("Add facet") shows only applicable facets, which are appropriate according to the current selections. The area below contains the *parallel coordinates plot* [16] as the main part of the user interface. The visualization consists of a changeable number of parallel axes representing different facets. Each axis can be adjusted by a control panel situated below which enables swapping, removing and filtering of the facet. The bottom of the search interface shows the currently selected search results according to the chosen facets and their values in a list view. In future versions of the prototype that result list will be enriched by additional result representations of the document's facets, e.g. displaying the distribution of the *authors* as term cloud visualization [22].

The classic input field for text queries was omitted to allow a harmonization of the query statement. Text queries or more general *similarity searches* are considered as yet another document facet. The left axis in the screenshot shows a full text query for the term "manual" which can be entered below the appropriate axis. The specific facet values for this facet consist of the score, the text search component returned for each document. The score is the numerical representation for the degree of similarity of a document to the given query. A score of 0 represents no similarity whereas a higher score represents a higher similarity. The same approach works for arbitrary types of queries-by-example where the similarity score is shown on the axis.

3.1 Supported facet types

The retrieval model supports different kinds of facets according to the applicable level of measurement. On the one hand, that differentiation is necessary to represent the facet in the visualization and on the other hand for the way queries can be stated to express all filter and ranking conditions the specific type can offer. Initially, we distinguish three different types of facets: *nominal*, *ordinal* and *ratio*.

A *nominal scale* is characterized by facet values which cannot be put into an order, e.g. the facet *project* addressing the project in which a document was created. This fact raises a few questions. Though, no natural order exists, the facets still have to be aligned in an intuitive way for the user. As each facet value "carries" a count describing the number of artifacts which belong to that value, a decreasing order based on the counts can be used. Alternatively, a lexicographic order is possible. Of course, other configurable heuristics can be used for sorting.

In contrast to the nominal facet type, the *ordinal scale* includes a natural order of the facet values. But still, the distances between the facet values cannot be derived from the underlying semantics, because there is no meaningful definition of the distance between the facet values. For instance, an ordinal facet could be the project phase the document was created in as the phases can be ordered according to the underlying project model.

Another distinction for the two described facet types can be made: an artifact can be assigned multiple facet values of one facet. An example for a multi-value facet is the *author*. If a document is created by two authors these two are distinct values of the facet *author*. In contrast, a document belongs to the project where it was created, i.e. the selection of a certain project in the facets returns only those documents of this single project. We call this kind of a facet a single-value facet.

Ratio scale facets are characterized by the following key points. The values of such a facet can be ordered and they carry a value meaning. The degree of maturity of a document or a product can be described in a ratio scale.

3.2 Querying the Parallel Coordinates Plot

Facets in a faceted search engine usually have the function of filtering. When the user chooses a facet value, all artifacts with different values for that facet are masked out. With the parallel coordinates visualization we want to expand this notion and enable the disjunction (logical OR) and conjunction (logical AND) of facet values of the same facet. Although, some commercial websites like kayak.com and endless.com support the selection of multiple facet values per facet, the user has no possibility to assess how many results he will get, when making different selections. Additionally, dependencies in the results (in this example flights or shoes) due to the facet values are not comprehensible.

Parallel coordinates plots allow the visualization of single data records with multiple facet values. In contrary to a fixed set of categories, the facets in our prototype can be changed in their order and can be included in or excluded from the visualization and the query formulation.

The possible selections in our approach depend on the facet type. For *ordinal* or *nominal* facets both single-value and multi-value selections seem possible. Whereas the first equals to the classical way, the latter could incorporate different ways of combining the values. If we assume a single-value facet, the conjunction would lead to zero results. In that case the disjunction is the only way to combine several facet values. If we have a multi-value facet –e.g. the *part function* where a (mechanical) part or assembly implements several functions – the combination with the conjunction works and filters out documents describing parts which do not comply with the requirements.

For facets which can be described by a ratio scale a single value selection is possible but uncommon. The multi-value selection can be solved by allowing the user to "draw" an interval of the values he is

particularly interested in the axis. If the above mentioned selection possibilities are combined, a quite expressive and complete query can be stated.
For multi-selections, the user can simply select several values similar to file selections in file browsers. After the selection, all other facet values are masked out. If the user later decides he wants to specify more facet values of the multi-value facet, a filter dialog, which can be opened from the control panel of each facet, allows further adding and removing. Figure 1 shows a multiple selection for the facet *project* which restricts the search results to artifacts which belong to the projects "Chassis", "Motor" and "Bremse". This selection describes a disjunction between the facet values of the facet *project*, i.e. all documents are returned which belong to one of the selected projects. If the facet is multi-valued, i.e. documents can be assigned multiple values of one facet, a conjunction of the facet values is assumed and only those documents are returned as search results which comply to all selected facet values.
If the user chooses an interval for ratio facets, the axis can be displayed in two ways. First, the scaling of the axis can remain as it was before the selection to illustrate the minimum and maximum value of that facet for the complete data. Second, the axis could be "zoomed" which restricts the axis to the current minimum and maximum of the chosen interval. Therewith, the single facet values are better viewable because they can be arranged on the complete height of the axis. For our prototype we chose the first approach as default with the option to enable zoom.
If the user constrains the search result with facet selections, the result list at the bottom of the user interface is constantly updated so that only the selected artifacts are displayed (cf. Figure 1). The result list shows the file path of each search result and the score for the current search. Additionally, the user could be provided with a short textual preview (usually called snippet) if available. This snippet might be a textual snippet for text documents or a graphical preview of a CAD model for example. The user has the possibility to open each search result in the appropriate host application or system.
Since one main benefit of the search interface visualization is the attained insight and knowledge about the underlying search results, it is important to provide the user with a set of different operations which help to navigate through the artifacts [23].
The user can influence the presentation of the parallel coordinates in various ways in order to allow for a flexible assessment of the artifacts:

- The search tool provides the facility to add and remove facets to refine search queries in the parallel coordinates plot. This functionality is justified by the requirement that each information need affects different contextual factors which are represented as facets. The user can simply drag new facets from the ribbon component in the upper area of the prototype on the parallel coordinates plot. The facet is instantly added to the plot so that the user can assess the connections between the artifacts.
- New insights of the data in a parallel coordinates plot may only be discovered when facets are reordered [17], as this visualization especially offers deeper insights for two adjoint axes. The user can reorder the axes through the control panel below each facet (cf. Figure 1).
- The user can influence the sorting of the facet values within a facet axis according to the facet type.
- Strongly connected is the filtering of facet values. The facet values of artifacts, which do not match the current facet user query, are not shown in the parallel coordinates plot. If a facet selection is removed, the facet values, for which artifacts exist, are shown again.
- Finally, the user can inspect and select single artifacts and their facet values by hovering over the polylines in the plot or by selecting (multiple) entries in the search result list.

3.3 Dynamic facet provision

One important aspect of the faceted search paradigm is the prevention of empty result sets by providing the user only with those facet values which are valid for the current artifacts in the result set. The same has to be valid for the available facets which are offered to the user for further refinement of his search. For this requirement we are making two distinctions. On the one hand, the provisioning of facets for the different artifact types and on the other hand, dependencies between facets.
As mentioned above, our prototype supports different artifact types which may be beneficial for different information needs during product development. Our search engine distinguishes different layers. A layer is defined by a single artifact type and contains all indexed artifacts from this type. In our scenario, we initially distinguish the layers *document*, *product*, *project*, and *person*. This step is

necessary, since a mixture of different artifact types within the ranking of one search result is not possible assuming different similarity measures and would also be difficult to convey to the user.
Different types of connections between the different artifact types and the actual artifacts exist. The elements of one layer can be connected. In the product layer, *is-part-of* relations can be established between products and their components. The linkage between the layers defines the anchor points which enable the traversal between the layers. For example, *documents* are linked to the *project*, in which they were created. Our search paradigm allows advancing from one layer to another through those inter-layer connections. For example, a requirements document – found in the document layer – is connected to a project from the project layer. Additionally, the different projects consist of subprojects where different links might be propagated. For each project the user can switch easily to the person layer and find people working in that project. In our model, every layer can be connected to every other layer.
The available layers and connections are company-specific and have to be modeled as artifact hierarchies before the search engine can be deployed. These hierarchies are integrated into an artifact type schema by defining the intra-layer and inter-layer relationships.
During product development, engineers usually search on one layer and are navigating in one type of artifacts. Our retrieval model enables the user to switch between different layers (i.e. the different artifact types) based on interim search results. The prototype offers a contextual ribbon task (the green ribbon group with the task "Switch artifact types" in Figure 1) which is displayed when layer switches are allowed. The search engine decides on this question based on the current artifact type of the search result. If that artifact type contains links to other layers, these are traversable through the ribbon. The search engine then uses result artifacts and switches to the other layer by using the currently chosen facet selections. These can be adjusted later if necessary and therefore enable a roundtrip between the different layers.
An interesting use case for this layer switching lies in the examination of how successful development methods were used in projects. The search engine could be queried for project review documents. The respective result set can then be taken as an input for a search which retrieves the methods used in those projects, which traditionally would have implicated a manual review of the complete project documentation.
Additionally, we consider dependencies between facets. As mentioned above, an artifact type schema is defined for each artifact type. The dependencies are derived from artifact hierarchies which have to be created for the specific domain. We conducted a document analysis for the domain of product development in mechanical engineering to identify and describe the existing and necessary documents. Each document in the hierarchy was assigned metadata and context categories. All sub-documents inherit the categories from the corresponding parent document.
The available facets from which the user can choose during a search task depend on the previous facet selections. The search query is specialized by each new facet selection which offers the user more specific facets for his information need. For instance, if the user specifies that he wants to search for *products* and restricts the commodity group to *o-ring seals*, the facets *inner radius* and *outer radius* can be enabled. The update of the currently enabled facets is done instantly after a facet selection of the user.
We distinguish two different approaches how dependencies affect the visualization in the parallel coordinates plot. The first case describes that a facet is completely dependent on certain facet values of another facet. For instance, if the user chooses *CAD* as the *document type* to filter on, the facet *part function* should show up and allow the user to constrain his query by that criterion. In the second case the dependency is defined between facet values of one facet with facet values for another facet. For instance, we have a dependency between the facet *document type* and the facet *document format*. If the user chooses to filter on documents of type *CAD*, the facet *document format* should be restricted to *IGES*, *JT*, *STEP* … which are all specific CAD file formats. If contextual information is applied, dependencies exist e.g. between the role of the user in the current project and the current project phase. With this information the search engine knows what documents are relevant to the user's information need – at least to a certain degree.
The last approach generates some difficulties for the search interface. If more than one dependency is defined between two facets, the search engine may only offer those facets which are guaranteed to be applicable for all artifacts. For illustration we expand the above described example dependency with a dependency of *requirements documents* which are stored in "textual" file formats like *PDF*, *DOC*, etc.

If the user chooses both CAD and requirements documents, the inclusion of the facet *part function* would lead to a contradiction, as that facet is not defined on textual documents. In this situation the search engine would omit the facet *part function*. If the user only chooses CAD documents that facet would be enabled.

3.4 Combination of similarity search and faceted search

As already mentioned, we tried to harmonize the query formulation for the user. Therefore, we included similarity searches like keyword queries and queries-by-example in the parallel coordinates plot. These queries behave like any other ratio facet of the current artifact types, i.e. the similarity score represents the facet value for these specific *similarity facets* of a document.

Similar to attribute facets, similarity facets can be included during a retrieval task by adding them from the "Add Facet" list from the ribbon component. That adds an empty axis as a placeholder to the parallel coordinates plot and offers the user a possibility to specify the search criteria for that facet. Here, we distinguish two conceptually different types of queries. The user can enter a text query giving some keywords as seen in the facet *text query* in Figure 1 where we searched for the term "manual". The backend search engine then performs a search for these keywords according to the Vector Space Model (cf. [24] for further reading).

Alternatively, the user can conduct an example query in uploading an example drawing or CAD model he needs in his current task. This is helpful in identifying existing parts in the company's repositories. For similarity queries on CAD documents see [25]. The similarity can be determined by comparing different feature types. For instance, technical drawings can be compared according to the contained geometry or the contained topology. With our approach the user has the possibility to choose the similarity measures which are most appropriate in his current search situation. A combination of several measures of course is possible as well. With the different weighting per facet, the user is able to explicitly weight those measures according to his information need. Hereby, the selection of a single value of the similarity score does not seem very useful, but the selection of an interval of the scores is more appropriate.

3.5 Influencing the ranking with facets

In addition to the classic filter functionality of faceted search and the above explained similarity facets, our retrieval model provides the possibility to influence the ranking of artifacts by the statement of *user preference functions*. The result of a faceted search is usually only a set of artifacts as a simple filtering takes place. One possibility to combine faceted search and ranking is the statement of a similarity search beforehand (e.g. a keyword query) that returns an ordered ranking on which the faceting is based. Faceting comprises the determination of the facet values and the according facet counts, i.e. the number of artifacts which comply with the specific facet value, based on a result list. Another alternative would be that the user tells the search engine according to which facet a sorting should take place. In our retrieval system a more generic model is used. The user is able to set a function for each facet in his current query. Therewith, the user can simply express his preferences for specific facet values or for ranges of a facet. For instance, the user can rank artifacts higher which have a higher degree of maturity. Facets which can be parameterized by a preference function are called *ranking facets*.

The user can express his weighting choices by stating user preference functions f_i for each facet i graphically. These functions are applicable for all supported facet types but differ in their continuity. Whereas a continuous function can only be set for ratio facets, preferences for ordinal as well as nominal facet values can be stated in a discretized function as exemplified in the first axis in Figure 2. Artifacts conforming to one of the three facet values which are highlighted by a gray bar are ranked according to the length of the bar for each highlighted facet value, where a longer bar represents more weight for that value.

For ratio facets arbitrary complex user preference functions f_i could be defined according to preferences of the user. However, for easier comprehension the system should introduce some simple template functions which are needed frequently. The simplest application of these templates is the filtering by intervals as seen in the second axis in Figure 2 where only documents are included with a degree of maturity between 60% and 100%. The function in the third axis ranks more mature documents higher and excludes documents where the corresponding value lies below 50%. If the user

already knows the released documents (i.e. documents that are 100% mature) and wants to rank documents higher which will be finished soon, the function in the fourth axis can be helpful which puts an emphasis on documents with 80%.
The user can simply adjust these functions by resizing the gray area defining the function with the mouse. The search engine instantly updates the ranking according to the function.

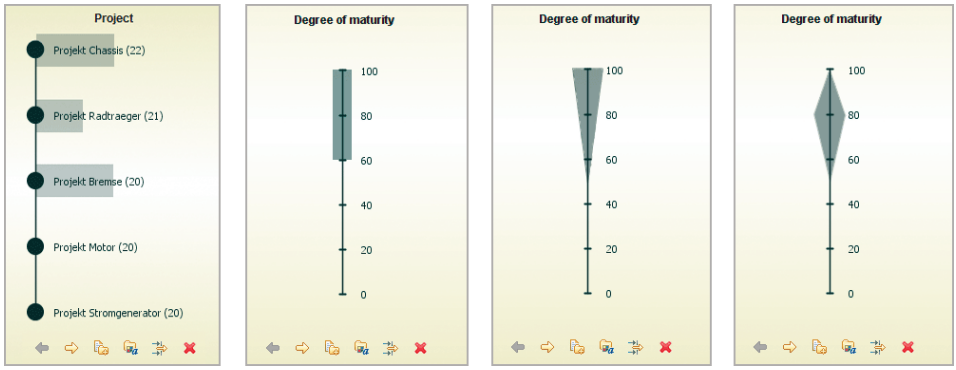

Figure 2. Example user preference functions

The retrieval system determines the overall score of an artifact a_j by calculating the weighted sum of the sub-scores for each facet according to:

$$score(a_j) = \sum_{i=1}^{n} \alpha_i \cdot f_i(x_{i,j}) \text{ with } \alpha_1 + \alpha_2 + \cdots + \alpha_n = 1$$

Here, α_i is the weight for the facet i, which can be set by the user via the control panel (3rd icon from the right) to influence the weighting of the functions for each facet. $f_i(x_{i,j}) \in [0,1]$ gives the preference of the user for the value $x_{i,j}$ of facet i for artifact a_j as graphically defined by the user.

3.6 User support with contextual information

As already mentioned, we examined the context of engineers and documents during product development in the mechanical engineering domain. This context information can be integrated reasonably in our search prototype. As stated, the complexity due to the variety of facets which are available for the artifacts in the portrayed domain complicates the queries. In [26], we showed ways to integrate a context-aware search engine into project planning portals. One outcome of the research cooperation in which these results were achieved is the concept of a *process navigator* which is a specific workflow management system for product developers. Systems alike create much context information about the user's current working situation. This information can and should be used in initializing the search interface.
For instance, the context model describes the construction process the engineers have to follow. The process model depicts tasks, methods and documents, which denote entry and exit criteria. With this knowledge the search engine can already exclude file types and formats which are not useful for the current information need of the user. This automatism can be applied in an implicit or explicit fashion depending on the user's experience. Whereas a more experienced user might demand more control over the final search query, the casual user might not be aware of the possibilities of the search engine and therefore would omit search criteria. In the latter case, the search engine could help providing better search results. To show the common user different possibilities to conduct his search an initial set of facets can be displayed with some pre-chosen facet values. The user of course may change these presets if they appear inappropriate for his current search scenario.
Based on the current user context, the initial search could be automatically augmented with the facet "creation phase" and the role of the user. Based on the context knowledge the search engine assumes that the current user has to create a document in the current phase and therefore searches for

documents from past projects which would help him in his current task. The user can also state his query more precisely in doing a keyword search which is augmented in the background.
In this situation two different scenarios have to be considered. Does the search engine display the facets with all preselected facet values or is an initial filtering done in the background? The latter case hides complexity but it can lead to a lack of clarity for the user. He might not understand how the current selection of facet values emerged.
Additionally, engineers can be supported during retrieval tasks by being offered search templates. These templates consist of a pre-defined set of facets with optional pre-selections of facet values which are specific for certain tasks in product development. The end-user can simply select the template for his current situation and choose the desired facet values. This helps to hide the higher complexity of the search interface in comparison to current search engines. Experts or power-users are able to save certain searches as "templates" for later reuse and can assign them to tasks in the process model.

4 VISUALIZATION PROBLEMS AND POSSIBLE SOLUTIONS

Several workshops with different groups of engineering students showed that our exploratory search approach can lead to an improved searching experience. The workshops covered the usability and the acceptance of the ||-coords search interface. But nevertheless, several issues were identified which have to be resolved before the rollout in the company.
When a search returns many search results, the ||-coords visualization has a high density of plotted lines which is called o*verplotting*. This can make the understanding and inference of dependencies between two variables complicated because other data records might overlay the dependency. Edsall introduces several enhancements to overcome these disadvantages of parallel coordinates plots [19].
One approach to minimize the clutter caused by a huge set of facet values can be achieved by *focusing* on certain facets. For continuous facets a range can be chosen which should be included in the view to filter out facet values which are of no interest for the current query. All other data records are masked out which results in an uncluttered view. Taking the notion of a *Hierarchical Faceted Search* [21] those facets can be classified in a hierarchy, e.g. for a facet which describes a geographic reference (e.g. continent → country → state → city). The user can then choose the level of detail which is necessary for his current query. Another example would be the hierarchy of date facets where the user could choose for instance from year → quarter → month → day.
These mainly manually determined hierarchic categories have to be distinguished from clustering methods [13] which usually produce results which are not perfectly determinable. The determination of the hierarchy has to be done in the back end search engine. The front end only needs to be aware of hierarchical facets.
Another interesting approach for highlighting data sets in the parallel coordinates plot is called *Brushing*. The user can simply draw a rectangle to select a set of records which are instantly highlighted with a striking color. Edsall calls the use case where only information for one record is desired *Strumming*. Similar to playing a guitar (the axes are the frets and the records are the strings) the user strums over a record and the string shows up and is highlighted in the whole plot.
To allow the user to quickly assess the distributions of facet values of a certain facet, we added the visual clue of a circle which varies in the diameter. A facet value which is assigned to more artifacts is depicted with a bigger value marker. This is achieved by a logarithmic function to constrain the maximum size of the marker (cf. Figure 1).

5 CONCLUSION / FUTURE WORK

In this publication we showed an approach how to employ the manifold (meta-) information which is created throughout product development processes to support engineers in stating complex queries in a visual way. Therewith, we enhanced the classical faceted search approach by giving the user more appropriate insights about the underlying data collection by means of parallel coordinates plots. We included ways to incorporate similarity searches specifically for the domain under research and showed how engineers can query the data collection with both text and example product models.
Additionally, we contributed a homogeneous method to combine faceted search and ranking approaches in utilizing simple, but yet powerful user preference functions to influence result artifact rankings.

The paper introduced a user-centric way of enabling a contextual search engine for the mechanical engineering domain. Although the search interface bears a higher complexity for the user, we are confident that this approach is feasible in a domain where search is a key point in leveraging reuse of existing knowledge and more specific, parts or components. The usability assessment and the anticipated more precise search results are a subject in the upcoming user tests.
Another field we want to delve into is collaborative elements which nowadays are strikingly labeled "Enterprise 2.0". Relevance feedback and user tracking can be facilitated for the presentation of the facets as already mentioned above. We also imagine collaborative filtering and recommender system mechanisms [27, 28]. If other users did similar searches in the past and the results proved to be helpful, the same results can be recommended to the current user.
Faceted search in general can be implemented quite efficiently [29]. We are aware of potential performance issues which are caused by the calculation of the polylines for the chosen facets and are working on performance improvements.

Contact: Raiko Eckstein
University of Bamberg
Chair of Media Informatics
Bamberg, Germany
Tel: +49 951 863 2884
Fax: +49 951 863 2852
Email: raiko.eckstein@uni-bamberg.de
URL: http://www.uni-bamberg.de/en/minf/team/eckstein/
Raiko Eckstein is a research assistant at the Chair of Media Informatics at the University of Bamberg and working in the Bavarian joint project FORFLOW. His main research topics are context-based visual information retrieval, data visualization and exploration in an enterprise context.

Contact: Andreas Henrich
University of Bamberg
Chair of Media Informatics
Bamberg, Germany
Tel: +49 951 863 2850
Fax: +49 951 863 2852
Email: andreas.henrich@uni-bamberg.de
URL: http://www.uni-bamberg.de/en/minf/team/henrich/
Andreas Henrich is a full Professor for Media Informatics at the University of Bamberg. His research interests are especially in information retrieval, data visualization and exploration, and e-Learning.

REFERENCES

[1] Eckstein R. and Henrich A. An integrated context model for the product development domain and its implications on design reuse. In *10th International design conference: DESIGN 2008*, 2008, pp. 761–768. .
[2] Mitchell T.M. *Machine Learning*, 2008 (McGraw Hill; McGraw-Hill WCB, Boston, Mass.).
[3] White R.W. and Drucker S.M. and Marchionini G. and Hearst M. et al. Exploratory search and HCI: designing and evaluating interfaces to support exploratory search interaction. In *CHI '07: CHI '07 extended abstracts on Human factors in computing systems*, 2007, pp. 2877-2880. (ACM, New York, NY, USA).
[4] Eckstein R. and Henrich A. Reaching the Boundaries of Context-Aware IR: Accepting Help from the User. In *Proceedings of the 2nd Int. Workshop on Adaptive Information Retrieval (AIR 2008)*, 2008 (BCS, London).
[5] Redon R. and Larsson A. and Leblond R. and Longueville B. VIVACE Context Based Search Platform. In *Modeling and Using Context, 6th International and Interdisciplinary Conference, CONTEXT 2007, Roskilde, Denmark, August 20-24, 2007, Proceedings*, 2007, pp. 397–410. (Springer, Berlin / Heidelberg).
[6] Karnik M.V. and Gupta S.K. and Anand D.K. and Valenta F.J. et al. Design Navigator system: A case study in improving product development through improved information management. In *Proceedings of IDET/CIE 2005: ASME 2005, Long Beach, California, USA*, 2005 .

[7] Marchionini G. and Shneiderman B. Finding Facts vs. Browsing Knowledge in Hypertext Systems, *Computer*, 1988, vol. 21, no. 1, pp. 70–80.
[8] Marchionini G. Exploratory search: from finding to understanding, *Commun. ACM*, 2006, vol. 49, no. 4, pp. 41-46.
[9] Pahl G. and Beitz W. and Feldhusen J. and Grote K.-H. et al. *Engineering design: A systematic approach* 3rd ed., 2007 (Springer, London).
[10] Yee K.-P. and Swearingen K. and Li K. and Hearst M. Faceted metadata for image search and browsing. In *CHI '03: Proceedings of the SIGCHI conference on Human factors in computing systems*, 2003, pp. 401-408. (ACM Press, New York, NY, USA).
[11] Duda R.O. and Hart P.E. and Stork D.G. *Pattern classification* 2nd ed., 2001 (Wiley, New York, NY).
[12] Zamir O. and Etzioni O. Grouper: A Dynamic Clustering Interface to Web Search Results, *Computer Networks*, 1999, vol. 31, no. 11-16, pp. 1361–1374.
[13] Hearst M.A. Clustering versus faceted categories for information exploration, *Commun. ACM*, 2006, vol. 49, no. 4, pp. 59-61.
[14] Hearst M. Next Generation Web Search: Setting Our Sites, *IEEE Data Engineering Bulletin, Special issue on Next Generation Web Search,* 2000, vol. 23, no. 3, pp. 38–48.
[15] Inselberg A. The plane with parallel coordinates, *The Visual Computer*, 1985, vol. Volume 1, Number 4, pp. 69–91.
[16] Inselberg A. and Dimsdale B. Parallel coordinates: a tool for visualizing multi-dimensional geometry. In *Visualization '90: Proceedings of the First IEEE Conference on Visualization, San Francisco, California, October 23 - 26, 1990*, 1990, pp. 361–378. (IEEE Computer Soc. Press, Los Alamitos, Calif.).
[17] Unwin A. and Theus M. and Hofmann H. *Graphics of Large Datasets - Visualizing a Million*, 2006 (Springer Science+Business Media, LLC).
[18] Wegman E.J. Hyperdimensional Data Analysis Using Parallel Coordinates, *Journal of the American Statistical Association*, 1990, vol. 85, no. 411, pp. 664-675.
[19] Edsall R.M. The parallel coordinate plot in action: design and use for geographic visualization, *Computational Statistics & Data Analysis*, 2003, vol. 43, pp. 605-619.
[20] Tory M. and Potts S. and Moller T. A Parallel Coordinates Style Interface for Exploratory Volume Visualization, *IEEE Transactions on Visualization and Computer Graphics*, 2005, vol. 11, no. 1, pp. 71–80.
[21] Hearst M. and Elliott A. and English J. and Sinha R. et al. Finding the flow in web site search, *Communications of the ACM*, 2002, vol. 45, no. 9, pp. 42–49.
[22] Halvey M.J. and Keane M.T. An assessment of tag presentation techniques. In *WWW '07: Proceedings of the 16th international conference on World Wide Web*, 2007, pp. 1313-1314. (ACM, New York, NY, USA).
[23] Siirtola H. Direct manipulation of parallel coordinates. In *4th. International Conference on Information Visualization (IV'00)*, 2000, pp. 373–378. .
[24] Baeza-Yates R. and Ribeiro-Neto B. *Modern information retrieval*, 2008 (Pearson Addison-Wesley, Harlow).
[25] Bustos B. and Keim D.A. and Saupe D. and Schreck T. et al. Feature-based similarity search in 3D object databases, *ACM Comput. Surv.*, 2005, vol. 37, no. 4, pp. 345–387.
[26] Eckstein R. and Henrich A. Integration possibilities of a context-based search engine into a project planning portal in the mechanical engineering domain. In *LWA 2007: Lernen, Wissen, Adaption, Halle, September 2007 ; Workshop Proceedings*, 2007 (Universität Halle, Halle).
[27] Resnick P. and Iacovou N. and Suchak M. and Bergstrom P. et al. GroupLens: an open architecture for collaborative filtering of netnews. In *CSCW '94: Proceedings of the 1994 ACM conference on Computer supported cooperative work*, 1994, pp. 175-186. (ACM, New York, NY, USA).
[28] Adomavicius G. and Tuzhilin A. Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions, *IEEE Transactions on Knowledge and Data Engineering*, 2005, vol. 17, no. 6, pp. 734–749.
[29] Ben-Yitzhak O. and Golbandi N. and Har'El N. and Lempel R. et al. Beyond basic faceted search. In *WSDM '08: Proceedings of the International Conference on Web Search and Web Data Mining*, 2008, pp. 33-44. (ACM, New York, NY, USA).