Engineering Information Management Tools by Example

Michael Nebeling, Matthias Geel and Moira C. Norrie
Department of Computer Science, ETH Zurich
CH-8092 Zurich, Switzerland
{nebeling,geel,norrie}@inf.ethz.ch

ABSTRACT

While there are many established methodologies for information systems development, designing by example has not been formally explored and applied previously. Our work is also motivated by the desire to explore interface-driven development techniques that could complement existing approaches such as model-driven engineering with the goal of reducing the need for modelling and reengineering of existing applications and interfaces, while still supporting the development task. We explore the example-based technique for rapid development of powerful and flexible information management tools based on the example of Adobe Photoshop Lightroom, a system that was originally designed to support the workflow of digital photographers in a flexible way. We analyse experiments in which two new systems—one for managing collections of research papers and another for software project management—were developed based on the Lightroom paradigm. We derive a conceptual framework for *engineering by example* and assess the method by comparing it to traditional model-driven engineering.

Categories and Subject Descriptors

H.5.2 [**Information Interfaces and Presentation**]: User Interfaces—*Theory and methods*

General Terms

Design, Human Factors

Keywords

Engineering by example, Lightroom paradigm

1. INTRODUCTION

Designing by example is an established technique used by designers and now studied and promoted by a number of researchers in the HCI community [9, 16]. Building on our own experiences as well as those of others is clearly more efficient than starting each time from scratch. This has been recognised within the information systems and software engineering communities, leading to the widespread adoption of design patterns [6] at both the conceptual and implementation levels as well as the concept of software product lines [3]. However, the focus is often on the reuse of software artefacts, which tends to promote conformity rather than novelty, while it has been shown that the use of examples can encourage creativity if used in the right way [9]. First of all, designers often use examples of similar products or systems not to simply copy them, but rather to analyse them with a view to thinking about what they could change. Second, they deliberately search out examples from other domains and use analogical reasoning to transfer concepts and ideas to the current problem being addressed. Although example-driven design may often be applied implicitly, the potential as a systematic development method is so far unexplored.

We set out to study example-driven design and investigate how it could be formalised to support information systems engineering, as well as how it would compare to other, well-established techniques such as model-driven development. While this technique has its origins in the software engineering discipline, it has previously been extended to web application development [2] and user interface engineering [18]. The common approach is to start from a model that is a rather abstract description of the artefact to be built and, from that, generate or systematically derive the application or user interface. In this sense, it is opposed to designing by example where an existing interface is the starting point.

In a first step, our investigations have focused on the design of new information management tools. Applications designed to support the workflow of photographers are among the most advanced personal information management tools available today. Not only do they offer a wide range of features for organising both the physical and logical storage of data, but they also provide rich functionality for processing and publishing that data in various ways. Adobe Photoshop Lightroom[1] is a prime example of such a tool, designed to offer a modular, task-based environment (Figure 1) to support the workflow of professional and serious amateur photographers in a flexible way [13]. We therefore decided to look at how the information management paradigm used in Lightroom could be applied to other domains. Our approach was to start with the management of research publications and ask a student intern to build a first prototype for managing and querying publications based on the Lightroom paradigm [7]. This prototype focused on the user interaction borrowing key concepts from Lightroom's

AVI'14, May 27 - 29 2014, Como, Italy.
Copyright © 2014 ACM 978-1-4503-2775-6/14/05...$15.00.
http://dx.doi.org/10.1145/2598153.2598164

[1]http://www.adobe.com/products/photoshoplightroom

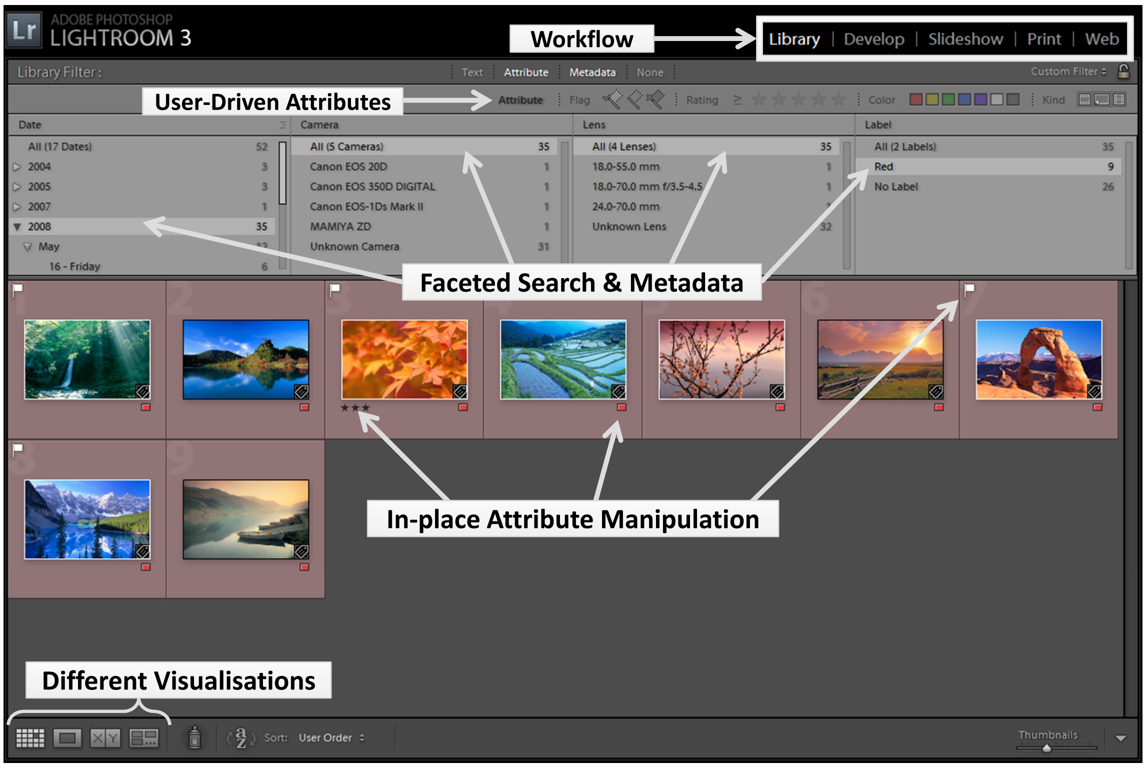

ADOBE PHOTOSHOP LIGHTROOM 3
Workflow
Library | Develop | Slideshow | Print | Web
User-Driven Attributes
Faceted Search & Metadata
In-place Attribute Manipulation
Different Visualisations

Figure 1: Adobe Photoshop Lightroom

user interface and very quickly showed promising results using the example-based technique.

In this paper, we focus on follow-up experiments in which another group of students were asked to build two additional prototypes. The first again targets publication management, but also includes database functionality and support for research tasks such as performing a literature review where users may want to mark and organise publications to track papers read, annotated and selected for reference. The second prototype applies the Lightroom paradigm to software project management and offers tools for managing software projects in terms of tasks and milestones. In both cases, the students were asked to use Lightroom as an example, but since collaboration is something central to the new domains, but rarer in photographer workflow, they were also asked to think beyond Lightroom and include features such as support for sharing.

This paper makes two primary contributions. First, it studies Lightroom as an example of an information management tool with the potential to generalise it to other domains. Second, it analyses the method used for developing the new prototypes inspired from Lightroom and presents a conceptual framework for using design-by-example as a user-driven development approach for information systems.

We start in the next section with a presentation of the main features of Lightroom along with a description of how it was designed and why we considered it a good example for our investigations. We then present the experiment and the resulting prototypes, followed by an analysis and discussion of the method in terms of development effort and the quality of the prototypes, as well as elaborating on our experience with, and the potential of, example-driven design.

2. LIGHTROOM PARADIGM

In 2002, Adobe started exploring ideas for a new product designed to support the entire workflow of digital photographers [13]. Previously, the focus had been on providing tools for image *processing* rather than *management*. The team at Adobe realised that little was known about the details of how photographers work and therefore they carried out an initial study based on interviews with different types of photographers. The results revealed that photographers were "patching together a series of applications to reach their goals" and there was a large variation in the time spent using Photoshop as well as the features used.

The Adobe team felt that a complete workflow solution could not only improve the efficiency of photographers, but also provide a "more consistent creative environment". They set about defining the requirements for a modular and task-based environment by getting photographers to participate in a qualitiative card sort exercise where they selected, sorted, grouped and labelled tasks. Although there was a wide variation in the initial task groups, after combining and decomposing them, there was a fair degree of consistency among photographers in how they grouped and ordered tasks. At the same time, features which appeared in many groups were identified as candidates for global status. They also identified commonly used image processing features that could be integrated into the new tool. This would mean that only those photographers requiring advanced processing features would have to open the Photoshop main application.

Adobe Photoshop Lightroom is the tool that emerged from this project and some of the main features are labelled in the screenshot in Figure 1. To support the management of thousands of images, it adopts features promoted in research such as automatic metadata extraction [10], advanced tagging systems [4] and dynamic, faceted search [15].

The main screen has a panel on each side which is made visible as and when required. The left panel shows the organisation of images into physical folders and logical collections which can be managed manually or correspond to pre-defined queries, i.e. smart collections. The bottom panel provides a horizontal, scrollable, strip of thumbnails of the current folder or collection. The right panel provides access to functions associated with a specific module. (Note that these panels are hidden in Figure 1). The top panel with the logo and modules can also be hidden to extend the image view area. Note that the faceted search interface is effectively another panel contained within the central pane that can be activated when required. The combination of the modules together with the panels allows the interface to be adapted to the current task.

In contrast to many special-purpose information management tools, Lightroom aims to support the entire workflow of photographers. Unlike other image processing tools, Lightroom focuses on the tasks and processing steps that are typically carried out by photographers. Its user interface is organised into the following modules:

- **Library** – supports the task of importing images with flexible controls over how and where they are stored as well as tasks that involve organising and selecting them.
- **Develop** – offers an extensive range of image processing operations, while basic ones for white balance and exposure adjustments are also available as "quick develop" features in the library module.
- **Slideshow/Print/Web** – support the tasks of publishing finalised photographs in various ways, including on popular Web 2.0 sites such as Facebook and Flickr.

Another distinguishing feature from many traditional desktop applications is that Lightroom does not follow the basic hierarchical model of organising resources, but instead provides many ways of organising images into different kinds of logical collections and collection sets. There are also several ways in which images can be marked to aid organisation and search. In addition to tagging, they can be rated, flagged and labelled with colours. By providing alternative ways of marking images, photographers are free to use these as they wish to support their workflow. For example, while some photographers rely on basic flagging to "pick" images to be developed [11], others rate images and then start by processing only the 5-star images using colour labels to mark the stages of image processing [5]. Selecting the best of many similar photographs is supported by comparison views, which can also be used to compare images before and after processing.

It is beyond the scope of this paper to describe all the functionality of the Lightroom tool. We have highlighted the main features that provide photographers with advanced, but flexible, ways of organising large libraries of images and their work. Lightroom is interesting to be studied as an example of a general information management tool both from the user interface perspective and the way in which it manages the data and metadata. We believe that the flexible support for the organisation of large data collections and its powerful data management tools, as well as the holistic and task-centred approach taken in its design, are the key reasons behind the success of the tool and enthusiastic responses from professional photographers, e.g. [11]. While one approach would be to adopt the same approach and design method for developing other kinds of information management tools from scratch, we wanted to investigate whether using Lightroom as an example could lead to richer information management tools in other domains such as reference management or for managing software projects. We therefore decided to set students the task of designing tools for these domains based on the Lightroom paradigm in order to see whether, within a short period of time, this approach could lead to tools that could at least match, if not improve on, existing tools in terms of the features supported and how it supports them. The next sections report on the experiment and the resulting tools.

3. EXPERIMENT

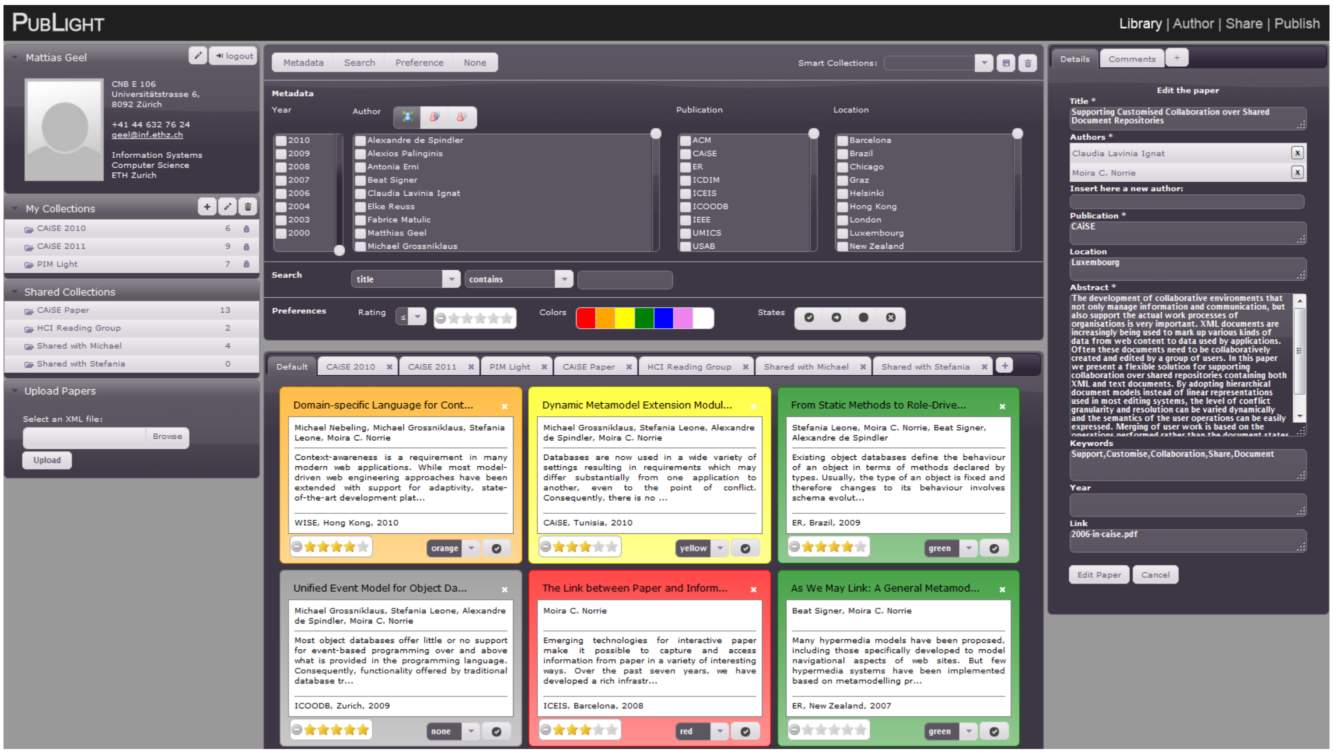
Our experiment took place as part of an Information Systems Laboratory course at ETH Zurich, which is a semester-long, project-based Master's course on information systems engineering. The goal of this course is for students to gain experience of working with state-of-the-art methods and technologies used in the development of information systems. Four students with programming experience participated in the experiment and formed groups of two students each. In a first step, they were introduced to Lightroom and the most interesting features similar to the overview in the previous section. One group was set the task of designing and developing a reference management tool based on Lightroom, while the other students were asked to devise a system that would assist in the management of software projects, again using Lightroom as the guiding example. Hence, the groups were asked to develop new prototypes for different domains.

The workload per student was 10 hours per week for a total of 13 weeks, using the first week to develop a project schedule and set up the development environment and the last week to prepare a presentation and give a demo of the resulting prototypes. Each group had weekly meetings with the supervisor to discuss the progress, but worked independently. Although the projects were not tightly specified in terms of what was to be delivered in order that we could assess the design-by-example approach, the students were still closely supervised in terms of defining the common steps and general principles as well as the experiment setup.

In the following, we describe the results in terms of the two systems, PubLight and ProLight that resulted from the experiment before moving on with our analysis of the method and formalising the approach. Both systems follow the modular, task-oriented environment of Lightroom for browsing, managing and sharing the information assets with the goal of supporting the research and project management workflows, respectively. We will first discuss which Lightroom concepts were considered for adoption and how they translated to the new domains. We then describe how tasks common to the research workflow, such as literature review and authoring, as well as those related to software projects, are supported in a flexible way based on the Lightroom paradigm.

3.1 Reference Management System

As already mentioned, Lightroom provides modules to address the different tasks of photographer workflow. Sim-



Mattias Geel

CNB E 106
Universitätstrasse 6,
8092 Zürich

+41 44 632 76 24
geel@inf.ethz.ch

Information Systems
Computer Science
ETH Zurich

My Collections

CAiSE 2010	6
CAiSE 2011	9
PIM Light	7

Shared Collections

CAiSE Paper	13
HCI Reading Group	2
Shared with Michael	4
Shared with Stefania	0

Upload Papers

Select an XML file:

Metadata

Year	Author	Publication	Location
2010	Alexandre de Spindler	ACM	Barcelona
2009	Alexios Palinginis	CAiSE	Brazil
2008	Antonia Erni	ER	Chicago
2007	Beat Signer	ICDIM	Graz
2006	Claudia Lavinia Ignat	ICEIS	Helsinki
2004	Elke Reuss	ICOODB	Hong Kong
2003	Fabrice Matulic	IEEE	London
2000	Matthias Geel	UMICS	Luxembourg
	Michael Grossniklaus	USAB	New Zealand

Search: title, contains

Preferences: Rating ≤ ☆☆☆☆☆ Colors 🟥🟧🟨🟩🟦🟪⬜ States

Default | CAiSE 2010 | CAiSE 2011 | PIM Light | CAiSE Paper | HCI Reading Group | Shared with Michael | Shared with Stefania

Domain-specific Language for Cont...
Michael Nebeling, Michael Grossniklaus, Stefania Leone, Moira C. Norrie
Context-awareness is a requirement in many modern web applications. While most model-driven web engineering approaches have been extended with support for adaptivity, state-of-the-art development plat...
WISE, Hong Kong, 2010
★★★★☆ orange

Dynamic Metamodel Extension Modul...
Michael Grossniklaus, Stefania Leone, Alexandre de Spindler, Moira C. Norrie
Databases are now used in a wide variety of settings resulting in requirements which may differ substantially from one application to another, even to the point of conflict. Consequently, there is no ...
CAiSE, Tunisia, 2010
★★★☆☆ yellow

From Static Methods to Role-Drive...
Stefania Leone, Moira C. Norrie, Beat Signer, Alexandre de Spindler
Existing object databases define the behaviour of an object in terms of methods declared by types. Usually, the type of an object is fixed and therefore changes to its behaviour involves schema evolut...
ER, Brazil, 2009
★★★★☆ green

Unified Event Model for Object Da...
Michael Grossniklaus, Stefania Leone, Alexandre de Spindler, Moira C. Norrie
Most object databases offer little or no support for event-based programming over and above what is provided in the programming language. Consequently, functionality offered by traditional database tr...
ICOODB, Zurich, 2009
★★★★★ none

The Link between Paper and Inform...
Moira C. Norrie
Emerging technologies for interactive paper make it possible to capture and access information from paper in a variety of interesting ways. Over the past seven years, we have developed a rich infrastr...
ICEIS, Barcelona, 2008
★★★☆☆ red

As We May Link: A General Metamod...
Beat Signer, Moira C. Norrie
Many hypermedia models have been proposed, including those specifically developed to model navigational aspects of web sites. But few hypermedia systems have been implemented based on metamodelling pr...
ER, New Zealand, 2007
☆☆☆☆☆ green

Edit the paper

Title *: Supporting Customised Collaboration over Shared Document Repositories

Authors *: Claudia Lavinia Ignat; Moira C. Norrie

Insert here a new author:

Publication *: CAiSE

Location: Luxembourg

Abstract *: The development of collaborative environments that not only manage information and communication, but also support the actual work processes of organisations is very important. XML documents are increasingly being used to mark up various kinds of data from web content to data used by applications. Often these documents need to be collaboratively created and edited by a group of users. In this paper we present a flexible solution for supporting collaboration over shared repositories containing both XML and text documents. By adopting hierarchical document models instead of linear representations used in most editing systems, the level of conflict granularity and resolution can be varied dynamically and the semantics of the user operations can be easily expressed. Merging of user work is based on the operations performed rather than the document states

Keywords: Support,Customise,Collaboration,Share,Document

Year:

Link: 2006-in-caise.pdf

Figure 2: Library Module in PubLight

ilarly, PubLight provides different modules which are inspired from Lightroom but were transformed to the new domain.

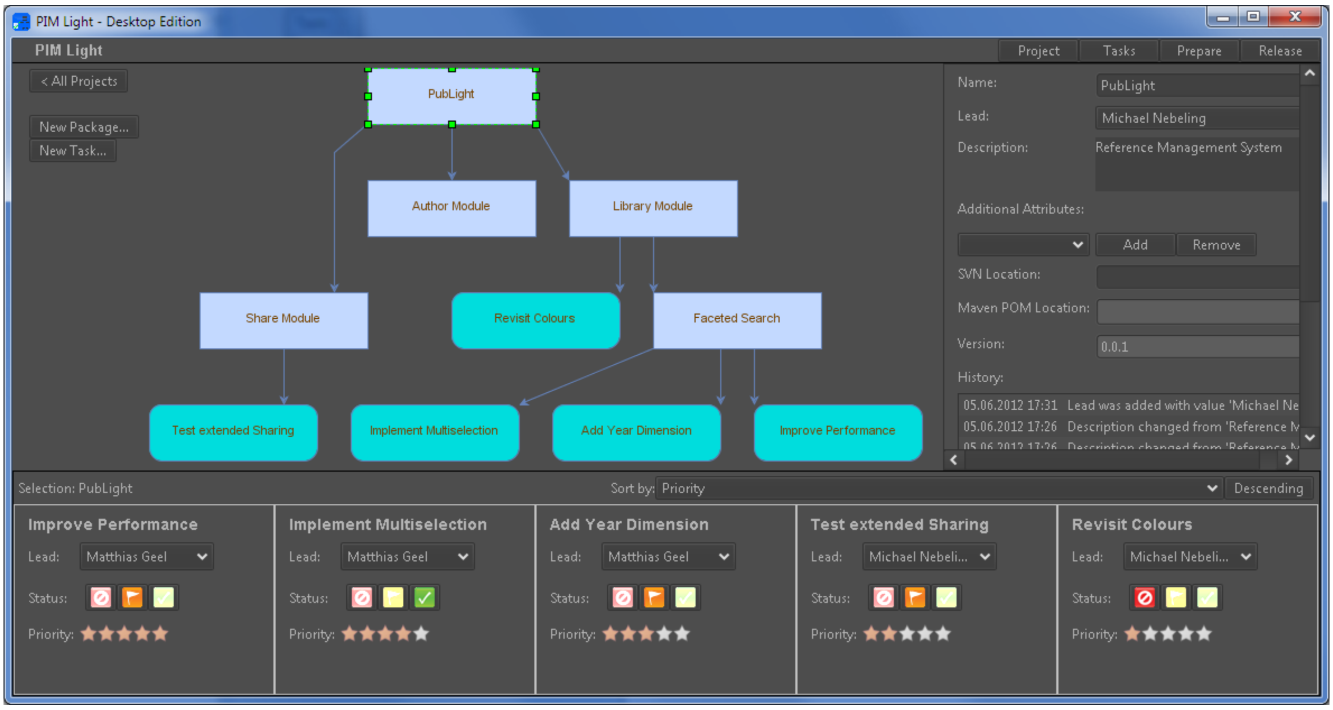
- **Library** – supports the browsing and management of publications.
- **Author** – provides simple tools for maintaining different versions of research publications, e.g. drafts, submissions and camera-ready, along with a review and discussion interface allowing co-authors to provide comments on different versions of a paper.
- **Share/Publish** – provides means for sharing parts of the publication library in the form of references or reading lists with other users, as well as allowing partial or complete export of the library to BibTeX or XML-based formats such as RSS and the integration of a public browsing interface with existing web sites.

Figure 2 shows the library module. On the left, there is the author panel with author information, collection management and import tools. The search panel and library view is shown in the central pane with details of selected publications in the context-sensitive panel on the right.

Similar to Lightroom, the tool offers a faceted search interface to filter according to a combination of criteria and users can colour, rate or flag items in various ways to help manage and search publications. For example, a user might colour papers according to topic and use flags to indicate which they have read. Our system also adopts some of the ideas of smart collections from Lightroom, but currently these are limited to saved searches rather than pre-defined queries, and need to be explicitly initiated by users.

For the faceted search interface, it was important to consider the types of queries a user might want to perform and how they could potentially differ from those supported in Lightroom. For example, since publications often have multiple authors, PubLight provides three different search modes for the *Author* facet. In the first mode, publications written by any of the selected authors are displayed. The second mode shows only those that were co-authored by the selected authors. The last mode allows publications to be filtered by first author only. The search can be further constrained by specifying terms for full-text search as well as conditions for user-defined ratings or colours.

Another important difference when moving to research publications relates to how collections are visualised. In the case of images, it is obvious to use thumbnails created from the images. While it would be possible to generate thumbnails for PDF documents, they would not reveal much information and in many cases would look very similar unless prominent teaser images were used. We therefore instead decided with the students to create summaries based on the publication's metadata such as title, authors, abstract and publication venue. The grid layout of summaries which may be coloured, rated, flagged etc. provides a visual means of organising publications. Yet most existing reference management systems rely on textual lists with relatively poor visual cues to the various attributes used to manage them and limited support for direct manipulation. In contrast, PubLight enhances reference thumbnails with small, in-place edit controls which enable users to directly manipulate the user-defined attributes such as colours or ratings, as well as managing the order of publications in the collection. The latter could, for example, be used to guide the writing process by deciding the order in which to present related work.



Name: PubLight
Lead: Michael Nebeling
Description: Reference Management System
Additional Attributes:
SVN Location:
Maven POM Location:
Version: 0.0.1
History:
05.06.2012 17:31 Lead was added with value 'Michael Ne
05.06.2012 17:26 Description changed from 'Reference M
05.06.2012 17:26 Description changed from 'Reference M

Selection: PubLight
Sort by: Priority

Improve Performance
Lead: Matthias Geel
Status:
Priority:

Implement Multiselection
Lead: Matthias Geel
Status:
Priority:

Add Year Dimension
Lead: Matthias Geel
Status:
Priority:

Test extended Sharing
Lead: Michael Nebeli...
Status:
Priority:

Revisit Colours
Lead: Michael Nebeli...
Status:
Priority:

Figure 3: Project Module in ProLight

In addition, it is possible to maintain the state as part of the authoring process for each paper using the in-place control in the bottom right of each entry. Our prototype discerns *draft*, *submitted*, *accepted* and *rejected* as four possible states in the publishing workflow.

3.2 Project Management System

The second prototype, ProLight, is similar to PubLight in that it also maps the core concepts of Lightroom to the new domain. In the case of ProLight, the students developed the following core modules:

- **Project** – supports the organisation and management of software projects.
- **Tasks** – provides different tools for managing a project in terms of packages and tasks.
- **Prepare/Release** – provides means for preparing software releases once milestones have been completed and supports the actual release process by integrating with Maven.

ProLight supports the management of software projects in terms of packages and tasks and distinguishes two types of projects. One option is that a ProLight project is based on Maven—a popular software project management tool[2] that can manage a project's build, reporting and documentation based on the central project object model (POM). In this case, the project can be imported from existing Maven POM files in the form of a new package. Alternatively, projects can be designed from scratch without the Maven backend. Figure 3 shows the project management view of ProLight with the packages and tasks defined for the project in terms of a graph (left-top) and ordered list (bottom) as well as a detailed view (right). For each task, a lead person, status and priority can be defined and it is also possible to define dependencies between tasks so that tasks can only be marked as finished when all dependent tasks have been done. Next to the default attributes, each task can also be associated with additional, project or task-specific, metadata attributes, which allows for flexible management of different kinds of projects and software domains. Like PubLight, ProLight also puts an emphasis on direct manipulation and in-place editing of attributes. In addition, ProLight automatically generates a history of all changes related to tasks and projects as a way of informing multiple collaborators and documenting the progress.

The planning of software projects can further be refined in the tasks view shown in Figure 4. Here, tasks can be assigned to the current package and put in order via drag-n-drop. Similar to PubLight, the faceted search interface on the left allows multiple filters to be combined. Note that any project-specific attributes are automatically available in the faceted search interface and can be used for filtering. Based on the data and metadata, it is then possible to check for open tasks and to prepare new releases. In the case of POM-file backed projects, ProLight automatically initiates the respective Maven prepare and release commands on the console and archives the build report.

[2]http://maven.apache.org

4. CONCEPTUAL FRAMEWORK

Our experiment raises the question of whether it is always best to start by designing the model in interactive systems design. In previous years of our course, we followed a more traditional approach by initially focusing the students' attention on the model of the artefact to be created. In this project, we explored the benefits of starting from an existing system, in this case Lightroom, and how it could be used as a guiding example.

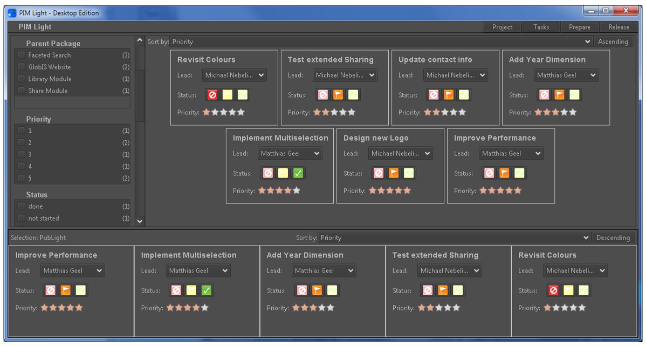


Figure 4: Tasks Module in ProLight

Figure 5 contrasts the example-based technique to the one described by Calvary et al. [1] which is common to most model-based user interface approaches. They distinguish different user interface abstraction levels starting from the model and an *abstract user interface* based on the concepts and tasks to be supported in the interface. The next level is the *concrete user interface* as an intermediate representation of the platform-dependent interface that is, however, not operational. The transition to the last level typically involves model transformation and code generation to produce the *final user interface* for a given context.

The suggested design processes typically unfold along a four-step, top-down approach starting with domain concepts and task modelling, followed by subsequent transformation steps from abstract to concrete and the final user interface. Some approaches support automatic transformation between some of the levels, while others rely more or less on the designer to perform the mapping manually. The authoring of model-based user interfaces has been the subject of extensive research, where one widely studied approach is MARIA [19]. One of the key benefits of having a more abstract description of the user interface in the form of a model is that it can be used to generate different user interfaces for different contexts of use. However, little attention has been devoted to the use of examples and how they could form an integral part of the development method.

In contrast to this vertical design process, our experiment based on the design-by-example technique mainly consisted of three steps.

1. **Extraction** In the first step, both teams studied Lightroom and identified the key activities that should also be at the core of the new information management tools extracting relevant features from the *example user interface*.

2. **Adaptation** In the second step, the respective features from the example user interface were transferred to the new domains.

3. **Reformation** In the third step, the new, *example-based user interface* was created based on a reformation and refinement of the features adapted for the new domain.

In our experiment, the first step resulted in the definition of three core activities: search and organisation of image collections using the *Library*, processing of photos in the *Develop* step, and sharing and publishing in the *Slideshow/*

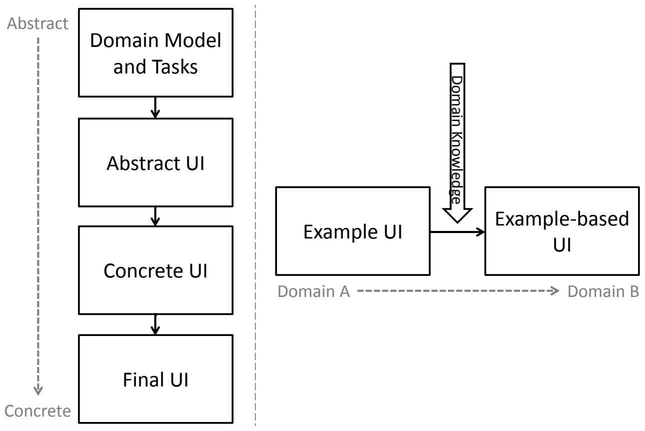


Figure 5: Model-based vs. Example-based User Interfaces

Print/Web views for the different output channels. The second step adapted these activities for the reference management and software project management domains. In the third step, the new *Library*, *Author* and *Share/Publish* as well as the *Project*, *Tasks* and *Prepare/Release* views were created and refined, respectively. However, in contrast to the model-based approach, there was no formal reengineering of Lightroom and also no formal modelling of the new domain in terms of concepts and tasks involved. Rather, the process was iterative in nature with students focussing on individual features, extracting them one by one and then transforming each of them to the new domains, gradually producing the *example-based user interface*.

5. DISCUSSION

The fresh example-based approach resulted in a very fast development cycle as well as producing rich functionality even though the team sizes were relatively small and manpower more limited than in previous course projects. Although both prototypes were implemented in fewer than 13 weeks (with an average of 10 hours per week per student), the teams produced two fully working systems with rich information management tools and basic support for collaboration among multiple users. Note that, given that the two student teams had only limited knowledge of the scientific publishing workflow and management of larger software projects, we provided them with information about the typical process steps that are involved based on which they then formed a possible feature catalogue for both systems. Therefore, the actual domain engineering was not a task carried out by the students. Rather, the focus was on investigating how the key concepts of Lightroom could be transferred to the new domains. In both projects, the students selected the main concept from the domain model, i.e. publication in the reference management system and task in the project management system, respectively. In each case, they developed thumbnail and detail views together with faceted search interfaces based on domain-specific and user-driven attributes to browse and manage the data collections.

The most obvious difference when comparing PubLight to established reference management systems such as Mendeley[3] or Papers[4] is the visual and direct manipulation in-

[3]http://www.mendeley.com
[4]http://www.mekentosj.com/papers

Approach	**Extraction**	**Adaptation**	**Reformation**
Bricolage [14]	Automatic support—limited to style and format based on CSS properties.	Automatic support—restricted to design; requires prior training of models to identify similarities between web pages.	No specific support—requires manual editing of generated web page.
WebML [2]	No specific support—manual extraction possible, but also requires example to be modelled in WebML.	No specific support—but generally possible based on separate models for content, navigation and presentation.	No specific support—requires manual editing of generated web site.
MARIA [19]	Automatic support—tools for reverse engineering, but not all details can be captured in the logical descriptions.	Not supported—adaptation to user, platform and device rather than different domains.	No specific support—requires manual editing of generated final user interface.
UsiXML [17]	Automatic support—based on graph transformation.	Automatic support—based on transformation of task models.	No specific support—requires manual editing of generated final user interface.

Table 1: Comparison of various approaches and tools regarding their support for designing by example

terface since existing tools tend to visualise collections of publications as lists of titles. Studies on the organisation of paper documents have shown the importance of visual cues [20] and the colour labelling offered in PubLight together with the summaries can help users recognise and categorise publications according to various criteria. Other noteworthy features are the grouping of tasks into modules to provide a clearer separation of workflow stages and the flexible means of organising publications into collections as well as within collections. PubLight offers simple yet powerful ways of searching for publications based on the concepts and ideas used in Lightroom while most existing systems tend to offer pre-defined filters or require users to construct complex queries.

Previously, we conducted a survey among researchers to find out about the reference management tools they use and how they use them for different scenarios such as literature review and the actual paper writing process [8]. The analysis clearly showed that existing tools tend to take a *data-oriented* rather than *task-oriented* view and support only parts of the research workflow. Further, although some of them offer a lot of functionality, the search process, interface and organisational capabilities tend to be more limited than those offered by tools for photographers and hence the prototypes that we developed based on the Lightroom paradigm. Interestingly, although the authoring of research papers is a very collaborative process, the survey indicated that users are either not aware of the support for sharing in reference management systems or do not make use of it. This could indicate that existing reference management systems do not support the entire research workflow in an integrated way and that the actual writing process is usually performed outside of the tool with the sharing of references tending to occur at the level of merging and editing BibTeX files. We see potential for PubLight to make a difference here. While formal user evaluations remain to be done, we have already begun using PubLight within our research group and in fact found it useful for compiling and jointly deciding on the references used in this paper.

We expect similar advantages for ProLight when compared to traditional project management tools that typically use list and table views or, in some cases, timelines to visualise the task schedule. One exception is Mylin[5] which is tightly integrated with the Eclipse development environment and itself uses a task-focused interface developed in previous research [12]. An early prototype of Mylin was shown to reduce information overload and make multi-tasking easier for developers by monitoring programmer activity to maintain the “task context” that automatically adapts the workspace and links all relevant artefacts to the task at hand. While task-oriented design of interactive applications have been the subject of an extensive body of research (e.g. [18, 19]), the core principles are often not carried through to and reflected at the final user interface level. Both PubLight and ProLight distinguish themselves by dividing the workflow into related tasks and organising the interfaces accordingly.

Starting from Lightroom’s interface and thinking about how the core information management tools provided by Lightroom could be translated to the new domain, not only helped the students to quickly come up with a first prototype while guiding the general design, but also helped them to think about the information concepts and how the interactions could be supported based on the example of Lightroom. Even for the new features of their prototypes, which were not part of Lightroom due to the focus on photographer workflow, they often revisited the example to get inspiration for how the new aspects could be incorporated and realised following the Lightroom paradigm. In particular, this reformation is a crucial step in designing by example. In order to take this work to the next level, it is therefore necessary to explore ways of facilitating extraction, adaptation and reformation.

We have started to experiment with several existing approaches as the basis for developing a new tool that can generate applications for different domains based on examples such as Lightroom. As summarised in Table 1, we have considered different solutions ranging from Bricolage [14] for transforming one web site into the design of another, over WebML [2] that can be used for modelling data-intensive web sites to MARIA [19] and UsiXML [17] for modelling and generating complex user interfaces in a platform-independent way. Analysing them in terms of their support for the core design-by-example tasks as identified in this paper revealed several strengths and shortcomings of each approach. We are currently in the process of developing a new tool that supports all three of them able to generate simpler versions of PubLight and ProLight. While this works fairly well for personal information management because there is usually a single main entity of interest, e.g. publication and task,

[5]http://www.eclipse.org/mylyn

we aim to support more complex data models and advanced workflows. The goal is to allow non-technical end-users—who are, however, domain experts—to develop their own information systems based on existing ones by further abstracting from the underlying models and instead providing direct manipulation tools.

6. CONCLUSION

Although our initial investigation on the design-by-example technique is still limited to only a few domains, the results in terms of speed of development and innovation show great promise. Based on the experiment presented in this paper, we have studied the design-by-example technique and identified the three key steps involved in the process, *extraction*, *adaptation* and *reformation*. We have demonstrated how the example-based technique was applied to the development of richer information management tools focusing on Adobe Photoshop Lightroom as an example and how the most promising features were transferred to other domains.

Our future work will focus on designing new development tools that specifically support these three design steps. We also plan additional studies on developers as well as detailed user evaluations of the prototypes created using design-by-example to further formalise and improve the technique. Future work would also need to, not only investigate the merits of the technique further, but also explore ways in which good examples could be identified and shared within the information systems community given that the choice of example would be pivotal to the success of a project.

Acknowledgements

We thank Julien Ribon, Alessandro Zala, Patrick Bänziger and Lucas Braun for participating in the experiment. This work was supported by the Swiss National Science Foundation under research grant 200020_134983.

7. REFERENCES

[1] G. Calvary, J. Coutaz, D. Thevenin, Q. Limbourg, L. Bouillon, and J. Vanderdonckt. A Unifying Reference Framework for Multi-Target User Interfaces. *IWC*, 15, 2003.

[2] S. Ceri, P. Fraternali, A. Bongio, M. Brambilla, S. Comai, and M. Matera. *Designing Data-Intensive Web Applications*. Morgan Kaufmann Publishers Inc., 2002.

[3] P. Clements and L. Northrop. *Software Product Lines*. Addison-Wesley, 2001.

[4] E. Cutrell, D. Robbins, S. Dumais, and R. Sarin. Fast, Flexible Filtering with Phlat. In *Proc. CHI*, 2006.

[5] M. Evening. *The Adobe Photoshop Lightroom 3 Book: The Complete Guide for Photographers*. Adobe, 2010.

[6] E. Gamma, R. Helm, R. Johnson, and J. Vlissides. *Design patterns: elements of reusable object-oriented software*. Addison-Wesley Professional, 1995.

[7] M. Geel, M. Nebeling, S. Leone, and M. C. Norrie. Advanced Management of Research Publications based on the Lightroom Paradigm. In *Proc. CAiSE Forum*, 2011.

[8] M. Geel, M. Nebeling, and M. C. Norrie. PubLight: Managing Publications using a Task-oriented Approach. In *Proc. TPDL*, 2012.

[9] S. Herring, C.-C. Chang, J. Krantzler, and B. Bailey. Getting Inspired! Understanding How and Why Examples are Used in Creative Design Practice. In *Proc. CHI*, 2009.

[10] D. Karger, K. Bakshi, D. Huynh, D. Quan, and V. Sinha. Haystack: A General-Purpose Information Management Tool for End Users Based on Semistructured Data. In *Proc. CIDR*, 2005.

[11] S. Kelby. *The Adobe Photoshop Lightroom 3 Book for Digital Photographers*. New Riders, 2010.

[12] M. Kersten and G. C. Murphy. Using Task Context to Improve Programmer Productivity. In *Proc. SIGSOFT*, 2006.

[13] G. Kim. Early Strategies in Context: Adobe Photoshop Lightroom. In *Proc. CHI Extended Abstracts)*, 2007.

[14] R. Kumar, J. O. Talton, S. Ahmad, and S. R. Klemmer. Bricolage: Example-Based Retargeting for Web Design. In *Proc. CHI*, 2011.

[15] B. Lee, G. Smith, G. Robertson, M. Czerwinski, and D. Tan. FacetLens: Exposing Trends and Relationships to Support Sensemaking within Faceted Datasets. In *Proc. CHI*, 2009.

[16] B. Lee, S. Srivastava, R. Kumar, R. Brafman, and S. Klemmer. Designing with Interactive Example Galleries. In *Proc. CHI*, 2010.

[17] Q. Limbourg and J. Vanderdonckt. *Multipath Transformational Development of User Interfaces with Graph Transformations*. Human-Computer Interaction Series. Springer, 2009.

[18] F. Paternó. *Model-based Design and Evaluation of Interactive Applications*. Springer, 2000.

[19] F. Paternò, C. Santoro, and L. Spano. MARIA: A Universal, Declarative, Multiple Abstraction-Level Language for Service-Oriented Applications in Ubiquitous Environments. *TOCHI*, 16(4), 2009.

[20] A. Sellen and R. Harper. *The Myth of the Paperless Office*. MIT Press, 2002.