PERSONALISING THE EDITORIAL MIX FOR A DIGITAL NEWSPAPER USING CONSTRAINT PROGRAMMING

MICHAEL LUNØE

Department of Informatics and Mathematical Modelling
Technical University of Denmark

August 2012

SUPERVISORS:
Michael Kai Petersen
Carsten Witt

[August 2012 – Technical University of Denmark]

ABSTRACT

Short summary of the contents...

CONTENTS

ACRONYMS

API Application Programming Interface: A specification intended to be used as an interface by software components to communicate with each other.

CP Constraint Programming: Constraint programming is a programming paradigm wherein relations between variables are stated in the form of constraints.

CSP Constraint Satisfaction Problem: Mathematical problems defined as a set of objects whose state must satisfy a number of constraints or limitations.

COP Constraint Optimisation Problem: Can be defined as a regular constraint satisfaction problem in which constraints are weighted and the goal is to find a solution maximizing the weight of satisfied constraints.

RSS Really Simple Syndication: A family of web feed formats used to publish frequently updated works—such as blog entries, news headlines, audio, and video—in a standardized format.

Part I

EXPLORING THE FIELD

Text for part 1.

1 CREATING THE EDITORIAL MIX

This chapter introduces the editorial mix of a digital newspaper and which parameters to account for when composing the newspaper. It is afterwards discussed which of these parameters are suited for personalisation, and how this can be done. The proposed approach is then presented with its pros and cons, which results in a list of contributions, that this project has to the field of personalisation.

What am I trying to achieve? Broad introduction, why is the following theory necessary

1.1 THE EDITORIAL MIX PROBLEM

In the conventional newspapers the editors job is to compose an intriguing front page that offers the contents of the sections that might interest the individual user. His challenge is to accommodate the needs of the newspapers segment of readers, divide the articles into sections, with a nice reading flow and attractive illustrations, and hand-pick articles to go on the front page. In 1965 [Haskins, 1965] defines the editorial mix problem as finding the least number of items to obtain maximum audience coverage, but with adaptive web sites it is possible to obtain a single user's preferences and accommodate them. Therefore it is possible to redefine the problem to:

> *Finding the composition of articles that provides the best satisfaction of the individual user preferences.*

An important part of the editorial mix is also that each piece of the mix needs to be interesting in it self, as stated by [Tidwell, 2010] in her definition of the editorial mix pattern.

[Perkowitz and Etzioni, 2000] decomposes the problem of synthesising an adapted page into several subproblems[1] :

[1] *In the subproblems stated here, "hyperlink" has been replaced by "item" to appreciate them more generally, rather than the original specific sense.*

- What is the content (that is, set of items) of the index page?
- Does it have a coherent topic? What should its title be?
- How are the items on the page ordered?
- How are the items labelled?
- Is the page consistent with the site's overall graphical style?
- Is it appropriate to add the page to the site? If so, where?

Some efforts have been made to digitally calculate similarities between articles and based on a current article suggest similar reading material or use collaborate filtering to suggest articles based on other users' reading behaviour. Some papers propose a composition of articles from user picked RSS-feeds, which can, e.g. in the case of Google Reader, be divided into sections. This comes close to conventional newspapers, but there is no ordering of the flow of articles. The ordering, flow and choice of relevant articles is from here on referred to as using *relational* features for creating the editorial mix.

The solution for some digital newspapers is still to have an editor to create their coherent composed digital newspaper, like the New York Times or Wired Magazine. Flipboard, on the other hand, compose their editorial mix of articles from feeds and divide their pages into three (or more rarely four or five) articles[1] with excerpts and images, much like conventional newspapers front pages. How they choose their composition is kept a business secret, but it does seem to vary a lot, see Figure 1.

[1] *Flipboard includes specialised layouts with more articles per page for Twitter.*

Maybe put in a reference to a conventional newspaper.

It is hard to say if there is a control behind the placement of content other than the choice of featured and non-featured articles, but this is actually an example of a computationally composed newspaper. The placement and amount of room given for an article is from here on referred to as using *spacial* features for creating the editorial mix.

Figure 1: A screenshot of composition of three articles in Flipboard, with different subjects, i.e. world crime, world finance and technology news.

Finally, subjects of articles have more relevance at some points in time than others, and editors choose the amount of time stories should be available in, where RSS-readers just displays the newest articles first, which are not always the most relevant. This selection of articles within a chosen time frame is from here on referred to as using *temporal* features for creating the editorial mix.

One thing that is vastly different from newspapers to RSS-readers is the ability to deliver personalised content, in that a user can choose which RSS-feeds to follow, whereas readers of newspapers need to navigate it in order to find interesting articles. Also, where newspapers have quality assurance of its content, RSS-readers have a seemingly unlimited amount of articles.

1.2 PERSONALISING A DIGITAL NEWSPAPER

[Bush, 1945] describes a collective memory library machine that can be indexed, called the memex. Items in the library are linked together forming personal association trails. This is the early conception of the hypertext media that would later be-

come the World Wide Web and later personalised web applications. With many respects that is what this project tries to achieve; i.e. link information in the form of articles together and present them in personalised trials defined by the user. As opposed to [Bush, 1945] proposed manual linking, it is now possible to, e.g. classify and compute similarity automatically, which greatly aids the process.

User preferences are very diverse, and it is therefore hard to accommodate every individual in a single solution. A digital solution must be bound to a specific domain, but must also be open for novel use.

> *"Web personalization is defined as any action that adapts the information or services provided by a Web site to the needs of a particular user or a set of users, taking advantage of the knowledge gained from the users' navigational behaviour and individual interests, in combination with the content and the structure of the Web site."*
>
> – [Eirinaki and Vazirgiannis, 2003]

The three categories of the editorial mix can be described in the sense of personalisation as well. Accommodating individual user preferences based on *spacial personalisation* is achieved by a placement of articles, *temporal personalisation* by selecting articles of higher news value based on their relevance time frame and, finally, *relational personalisation* by selecting articles that provides more value based on their respective and collaborate topics. Temporal personalisation is also obtained by letting user preferences have a life time and decrease the preference influence on which articles to select as time passes. This is referred to as personalising using a "temporal user model", whereas the former is referred to as personalising "temporal user preferences".

All three categories are related, as they each provide some value to the editorial mix; a different spacial placement of a specific article can, e.g. provide a different composition of the editorial mix and therefore a different relational value to the user, which also means that the user will discover articles at different times and therefore also provide different temporal value.

This paper seeks a more general approach to solving personalisation problems, and tries to establish the contributions of Constraint Programming to this field.

1.3 CONTRIBUTIONS OF CONSTRAINT PROGRAMMING

As a declarative programming language, Constraint Programming (CP) offers means for describing the problem to be solved using constraints and a general purpose constraint solver. Once the general purpose solver is set up, the constraints can be defined to model the problem to be solved, but does not necessarily make it easy. However, the problem definition can easily be extended and modified afterwards.

> *"Ordinary people generally aren't interested (and rightly so) in low-level programming details – they just want to express the problem in some reasonably congenial way and let the system get on with solving the problem. [...] Having to deal only (or mostly) with the logic component simplifies many things for the programmer. First, (the logic component of) a declarative program is generally easier to write and to understand than a corresponding imperative program. Second, a declarative program is also easier to reason about and to transform, as much current research in functional and logic programming shows."*
>
> – [Lloyd, 1994]

The editorial mix, and personalisation problems in general, consists of a series of requirements on what should be shown to the user. These requirements changes according to the individual user, often just by an adjustment of a general requirement. Because CP is a language for modelling requirements, it can be a great contribution to this field. If requirements are modelled as logic constraints, the changes to the individual user could be done by adjusting variables and parameters of the given values to fit the individual user.

To be able to use CP to solve our personalisation problems, it is necessary to define which problems CP works with. Constraint Satisfaction Problems (CSPs) and Constraint Optimisation Problems (COPs) are the two types of problems CP can be used to

solve. The following descriptions of CSPs and COPs have been modified to fit personalisation problems from the original definitions provided by [Russell and Norvig, 2003] and [Apt, 2006].

A Constraint Satisfaction Problem is defined by the 4-tuple $(\mathcal{V}, \mathcal{X}, \mathcal{D}, \mathcal{C})$, where $\mathcal{V}$ is the set of values, $\mathcal{X}$ is the set of variables, $\mathcal{D}$ is the corresponding set of domains and $\mathcal{C}$ is the set of constraints on the variables.

Each variable has a corresponding domain and each domain has sub-domains corresponding to the sub-variable, or attribute, of each variable.A value has a set of attributes, that may extend that of a variable, but if a variable is assigned, the variable's attributes should reflect that of its assigned value[1]. Unlike [Russell and Norvig, 2003] and [Apt, 2006] a constraint is here defined as a function on specific variables returning a boolean value. Therefore the tuple of u values with w attributes, n variables with m attributes and p constraints, where the ith constraint is defined on s_i variables, can be expanded to:

[1] *Others do not consider representation of values because they in their case only consist of simple integer, real or boolean values.*

$$\left(\begin{array}{ll} \mathcal{V}: & \left\{ v_1 : \begin{pmatrix} v_1.a_1 \\ \vdots \\ v_1.a_w \end{pmatrix}, \ldots, v_u : \begin{pmatrix} v_u.a_1 \\ \vdots \\ v_u.a_w \end{pmatrix} \right\}, \\ \mathcal{X}: & \left\{ x_1 : \begin{pmatrix} x_1.a_1 \\ \vdots \\ x_1.a_m \end{pmatrix}, \ldots, x_n : \begin{pmatrix} x_n.a_1 \\ \vdots \\ x_n.a_m \end{pmatrix} \right\}, \\ \mathcal{D}: & \left\{ d_1 : \begin{pmatrix} d_1.a_1 \\ \vdots \\ d_1.a_m \end{pmatrix}, \ldots, d_n : \begin{pmatrix} d_n.a_1 \\ \vdots \\ d_n.a_m \end{pmatrix} \right\}, \\ \mathcal{C}: & \left\{ \begin{array}{c} c_1 : func(x_{(1,1)}, \ldots, x_{(1,s_1)}) \to \mathbb{B}, \\ \vdots \\ c_p : func(x_{(p,1)}, \ldots, x_{(p,s_p)}) \to \mathbb{B} \end{array} \right\} \end{array} \right) \tag{1}$$

Where $\mathbb{B}$ is either `true` or `false`.

A CSP is a special case of a Constraint Optimisation Problem (COP) and a COP is defined by the 5-tuple $(\mathcal{V}, \mathcal{X}, \mathcal{D}, \mathcal{C}, \mathcal{O})$, where the three first elements are defined as in a CSP and $\mathcal{O}$

is a set objective (or cost) functions on variables, that determines the quality of a current state. The set of objective functions can be described with the same structure as constraints in CSPs and can be expanded as follows with q constraints, where the ith function is defined on t_i variables.

$$\mathcal{O} : \left\{ \begin{array}{c} o_1 : func(x_{(1,1)}, \ldots, x_{(1,t_1)}) \rightarrow \mathbb{R}, \\ \vdots \\ o_q : func(x_{(q,1)}, \ldots, x_{(q,t_q)}) \rightarrow \mathbb{R} \end{array} \right\}$$

Where $\mathbb{R}$ is the set of real numbers.

Satisfaction (or regular) constraints are also called hard constraints and objective functions are called soft constraints because a solution can be found if all hard constraints are satisfied, whereas and optimal assignment is enough to satisfy objective functions.

Finally, a constraint on a single variable is called an unary constraint, a constraint on two variables is called a binary constraint and a constraint on three or more variables is called a global constraint.

1.4 PROBLEM DESCRIPTION

Tilføj editorial mix til de første linjer af problemformulering så det er skåret ud i pap at problemet vel primært er hvordan man kan personalisere the editorial mix og dernæst hvorvidt constraint programming kunne være en løsning.

Motivation: Since you have a goal, there must be some problem that you are trying to solve. Explain this problem.

MAIN HYPOTHESIS SUGGESTION 1 Is it possible to use Constraint Programming to solve the editorial mix problem.

This project is a feasibility study of the implementation of CP in the field of personalisation. The criteria of success is whether it is possible to successfully implement general techniques of personalisation using CP and to make personalisation more accessible with the aid of CP. Therefore the project will be divided into two main areas; i.e. an assessment of the use of CP in the context of personalisation and a direct application of this in the form of a personal digital newspaper, where CP is used to personalise the content and composition of a digital newspaper.

Many techniques for personalising digital solutions already exists, but the role of CP within this domain has not been determined. This project seeks to explore CP as a tool to make the personalisation of digital solutions.

1.4.1 *Personalisation Challenges*

In an attempt to introduce a personal editorial mix in the digital newspaper, the report will try to analyse which preferences the users will have with respect to the content, composition and the time frame of relevant articles. It will describe the search for articles to fit the user needs as an Constraint Optimisation Problem and try to solve it. What makes a newspaper is not only the accumulated content of its articles, but the arrangement of them. "Which articles should go where" is just as important, and the placement of articles in the newspaper should therefore go through an equal solving process.

1.4.2 *Algorithmic Challenges*

To be able to use and assess CP in the context of personalisation a full understanding must be acquired. Features that can be solved using CP will be modelled as a COP and solved. Furthermore, because the problem has a fixed budget for finding a solution, its algorithmic complexity will be analysed. The findings will be concluded in an evaluation of the applicability of CP to personalisation problems.

2 PERSPECTIVES AND INSPIRATIONS

Web personalisation is by [Mobasher, 2007] divided into phases of data collection and preprocessing, pattern discovery and evaluation, and applying the discovered knowledge in real-time to mediate between the user and the Web. There have been many suggestions on how to tackle these different processes of creating the interactive personalised digital newspaper. [de Buenaga Rodríguez *et al.*, 2004] proposes a strictly stochastic approach to dynamic personalisation obtained by characterisation of content and user's interests. Both implicit and explicit relevance feedback[1] is used to refine the user models. Stochastic approaches have the advantage of being effective, but often solves a very specific problem. Also, these approaches tend to get very complex in order to deliver promising results. Some cope with this by introducing logic to the problem like it is done in [Nilsson, 1999] with a spacial approach. In this project it is possible to benefit from the structure of the logic approach of CP and the effectiveness of a stochastic approach by introducing preference constraints with an objective function.

Reference for stochastic approaches?

Many uses the approach of computing the tf-idf similarity with a cosine function as the distance function, as it is done in [Díaz and Gervs, 2005] to apply relational personalisation. It is based on a set of keywords extracted from the news items and a set of training documents. This constitutes the initial approach for computing similarity in this project. However, [Abuzir and Vandamme, 2002] argues that semantic knowledge is more substantial than keywords.

Classification techniques can be applied in order to ease the task of selecting relevant articles and determine their mutual

1 Implicit is when the (unaware) user's behaviour is recorded to determine relevance and explicit is where the user is aware of the action of giving the feedback.

relationships. [de Buenaga Rodríguez *et al.*, 2004] uses a library of documents to train a categorisation algorithm and the users are then asked to select categories of which they have interest. [Abuzir and Vandamme, 2002], on the other hand uses a thesaurus of hierarchically, and to the task specifically, structured terms to index news articles. Results of the indexing are thereafter mapped with user profiles to select the relevant articles. In stead of using predefined root terms as the basis for a classification, WordNet can be used to obtain semantic knowledge for a document. WordNet is a large lexical database of English words and their relationships in the form of different graphs. [Bouras and Tsogkas, 2010] presents an algorithm for enriching articles using WordNet's hypernym-graphs. WordNet also contains similarity functions between words. These functions will later on constitute the next step for computing similarity in this project.

[Díaz and Gervs, 2005] does present the means of combining the use of categories and keywords, but this approach demands predefined categories, which must be kept updated in order to follow semantic changes to the field. The time limitations and prioritisation of this project did not allow for a thesaurus to be obtained to aid the classification and will therefore not be introduced to the solution. One, could also argue that semantic assumptions are made, when categories are predefined, which could lead to some false classification. Whereas the structure of [Bouras and Tsogkas, 2010]'s algorithm bases its semantic structure only on words from the article and the general semantic (and more neutral) structure that constitutes the basis for WordNet.

[Claypool *et al.*, 1999] presents a combination of content-based and collaborate filters to predict interest in articles based on a user profile.

[Claypool *et al.*, 1999] presents a front page design and available sections. A section can appear as their front page. relevance feedback

their editorial mix consists of ordering the articles by most predicted interest first.

[Díaz and Gervs, 2005] along with both a short- and long-term representation of the user models. Furthermore, a global user profile, to get the process of generating the user model started.

[Centeno *et al.*, 1999] incorporates temporal personalisation in that it is possible to ask for articles in the newspaper based on a specific period, but also by incorporate ageing of user interests.

[Esteban *et al.*, 2001] incorporates temporal features of their personalisation of a digital newspaper based on Yahoo! Spain. Automatic Categorisation of news items, long- and short-term user models.

In the explored literature users shows much interest in being able to turn pages as it is done in a regular newspaper. [Ihlström *et al.*, 2004] describes this as "open, turn pages, chose article, read and return".

[Díaz and Gervs, 2005] proposes the use of collaborate filtering to handle the problem of converging, which is what will happen if no non-personalised articles are introduced, but this still only concerns articles that are within the area of the users interest. If e.g. a user has not shown interest in politics, the news of Barack Obama becoming the President of USA will never be included in the newspaper. Instead a ratio between personalised and general articles will solve this issue, and since it is not within everyones interest to receive general news, this ratio should be adjustable.

Users navigate the newspaper using sections and headlines as the main entry points [Ihlström *et al.*, 2004] and these should therefore be kept in the digital version.

Users express that these should be put into menu [Ovesson and Wikström, 2005].

[Díaz and Gervs, 2005] proposes personalised excerpts from the articles to further ease the navigation.

There exist many different examples of preference modelling using CP, and [Abidi and Chong, 2004] describes a factual information system to find personal information, e.g. about healthcare. They present two constraints: (1) select only

information-objects that correspond to the user-model and; (2) the content of the retained information-items do not contradict each other. This is an example of an editorial mix in that it incorporates the relational features, i.e. both between user and articles, and articles in between. However, they do not take into account the spacial part of their editorial mix, nor do they take into account any temporal features of the information needed. It is of cause notable that the user needs for a strictly factual information system are different than from a newspaper.

Another application of preference modelling using CP is [Vossen, 2005]. He proposes a CP approach to automatic playlist generation, which very much relates to what this project attempts to achieve. A playlist can be perceived as a, in this case, personal mix of songs. He presents constraints to exclude songs with certain attributes, to model that certain songs should be similar to each other or a user preference, to describe preference about the number of songs from a specific artist and the well-known `all-diff` constraint. These can be directly translated to the editorial mix of a newspaper, where the songs are articles and artists could be a specific author or content provider.

som afslutning på kapitel 2 kunne du opsummere hvilke features du mener er mest relevante så det ikke bare bliver en lang liste

You might also want to combine related work (3) and method (4) since it is easier to explain your own work in relation to the work it is built upon. Just make sure that your own contribution is not hidden. The reader will want to know what you did!

This paper proposes a Constraint Programming (CP) approach to solve the editorial mix problem. The problem will be expressed as a Constraint Optimisation Problem and solved using local search for CP. Furthermore, this paper proposes a keyword based solution in combination with a comparison of entities for determining the relevance for an article to a user defined topic and similarity between articles.

Er der argumenteret for at bruge keywords?

personalised topics in stead of the generic definition of the terms. Maybe what the crowd thinks is technology is not what you deem as technology

3 FEATURES TO BE PERSONALISED

This section analyses the user preferences and identifies which features should be modelled as personalisation constraints. It also seeks to define the default setting that consists the general model for a good editorial mix of a digital newspaper.

The focus of automatically generating the editorial mix introduces temporal, spacial and relational circumstances about the composition. But before these can be modelled as constraints it is necessary to look at the user needs of the application.

Also some prerequisites must be stated in order to set the focus of the problem, but also for the potential user to relate to the product. The research done by [Ihlström *et al.*, 2004] and [Ovesson and Wikström, 2005] determines the preferable size of the digital newspaper to be 14.732×20.828cm $\sim$ size A5, which reflects the size of the iPad. Therefore this project will target the iPad as its primary device. The hand-held device also introduces mobility, which is also of great preference to the potential users.

3.1 USER NEEDS

This section will define the user needs for the application. A full description of personas, scenarios and business case is found in appendix A on page 69.

Because there are so many that reads an online newspaper of some kind, the user group is very large, e.g. a whole 17% is smallest group of [Eurostat, 2012]'s distribution of individuals using the Internet for reading and downloading online newspapers and news magazines in 2011, divided into educational background. The focus of the project on the iPad, does not reduce the problem much as more and more users of tablet computers emerges. This means that the application must accom-

modate many user needs, and it is therefore rewarding to focus on few, but very general needs to outline the objective of success. This is done in the following.

From the scenarios it is clear that the users needs an overview of the content and that it should be presented in a nice layout. This can be done through the front page, where the most interesting stories should be found. In addition only headlines, images and excerpts could be displayed. To familiarise it with conventional newspapers the application could build on a paged interface, from the front page (page 0) through sections until it reaches the back page and to keep the overview page numbers could be used. It seems that both general and personal content is needed in order to satisfy users needs for information. However, it seems that the users may not necessary look for it. The amount of general versus personalised news is hard to define, but the user can be provided with functionality to adjust it. Also, reviews and opinionated articles could be of interest and maybe even puzzles and cartoons. In all cases it seems that the articles in the newspaper should be fresh and if the user makes changes in the personal settings, the newspaper should instantly update.

From the scenarios it is seen that users want to be social with their personal newspaper. This could be implemented both through a community in the application with comments on articles, but also by the possibility of sharing through common social networks. Notifications can be added to provide the user with the functionality of getting information on when there is a comment on a article that the user have already shown interest in or if new articles have arrived on a subject the user follows.

Finally, it seems to make sense that the users provide their preferences about sections in keywords, which could also be gathered using relevance feedback. These models of the users can later be used to sell user behaviour patterns and very targeted ads.

Based on the personas, scenarios and business case the following general user needs have been derived:

- Get an easy overview of the content of the newspaper

- Easily navigate between articles with few touch-friendly interactions
- Read articles presented in a nice and digestible layout
- Read relevant articles based on user defined topics

Furthermore, the editorial mix might not emerge as a need that users are aware of, but the rules of the editorial mix are established to better accommodate user needs in a composition of articles. In order to work with it in the application this is added to the user needs and the rules of the editorial mix are derived in the bottom of this section.

- Read articles in a composition based on the editorial mix

These user needs can be interpreted and fulfilled in many ways, e.g. relevant articles is very individual to each user and the way to apply these preferences on articles needs to be determined. Therefore no design choices are made before the problem have been analysed closer.

3.2 USE CASES

This section presents use cases based on the user needs from the previous section.

Use case #1	Get an easy overview of the content of the newspaper
Description	It is possible to get an overview of the articles in the newspaper
Actors	User, web server
Scenario	1. The user opens the application 2. The application sends a request to the web server to fetch articles 3. From the articles the personal newspaper is composed 4. The user is presented an overview of articles contained in the newspaper
Extension	3a. There are not enough personal articles The newspaper is composed of less strict preferences or the newspaper is composed only from available articles

Table 1: Use case 1

Notice that no specific assumption about the design was made in these use cases, but in stead merely that some components, like menus, headlines and a representation of the user interests are present.

There are many ways to achieve the presented use cases, but the next section will draw from the presented use cases and the explored literature to derive requirements.

3.3 REQUIREMENTS

In the explored literature, scenarios and presented use cases expresses some non-functional requirements. These are described in this section.

User needs states the requirement of having a clear overview of the content and as stated in [Ihlström *et al.*, 2004], this includes a clear marking of the beginning and the end of the articles and sections. This is obtained by both having a summery of the

Use case #2	Easily navigate between articles with few touch-friendly interactions
Description	It is possible to browse and navigate the articles in the application using few touch and conventional interactions
Actors	User
Scenario	1. The user gets an overview of the content of the newspaper as suggested in use case 1 2. The user navigates between articles using accessible menus of topic sections, headlines of articles and excerpts from articles 3. The user finds an article to read
Extension	3a. The user does not find an article to read The user searches for an article using the search bar

Table 2: Use case 2

Use case #3	Read articles presented in a nice and digestible layout
Description	It is possible to get articles displayed in a layout that serves the purpose of reading
Actors	User
Scenario	1. The user navigates the articles as suggested in use case 2 2. The user chooses an article to read 3. The chosen article is presented in a layout that easy to read

Table 3: Use case 3

Use case #4	Read relevant articles based on user defined topics
Description	It is for the user to read relevant articles of topics based on data gathered about user interests
Actors	User, web server
Scenario	1. The user uses the application regularly as suggested in use cases 1 to 3 2. The system gathers data about user interests 3. The system composes a newspaper of articles from the web server based on user interests
Extension	3a. The system does not have enough data on the user The system composes a newspaper of articles from a general model of a good newspaper

Table 4: Use case 4

Use case #5	Read articles in a composition based on the editorial mix
Description	It is possible for the user to read articles in a composition based on rules of the editorial mix
Actors	User
Scenario	1. The user uses the application as suggested in use case 1 to 4 2. Every composition of articles provided by the system follows rules of the editorial mix

Table 5: Use case 5

most interesting articles on the front page and by having a list of headlines in each section.

From the user needs it is also required that the system should be easily navigated and as stated by [Ovesson and Wikström, 2005], this should be through clickable sections, headlines and through paging, or as the users from [Ihlström *et al.*, 2004] describes it; "open, turn pages, chose article, read and return". [Ihlström *et al.*, 2004] also states that the newspaper indexing is the most effective "navigational" tool in newspapers and headlines are the main entry points to text, which means that these should be very central in application.

The layout, typography and design should be familiar to what is found in conventional newspapers, as stated by [Ihlström *et al.*, 2004] and [Åkesson *et al.*, 2005]. This is achieved by choosing a structure that resembles that of a newspaper and displaying content in balanced columns. In appendix A.4.2.1 on page 77 is found calculations on how many columns should be used on the iPad and on desktop computers based on conventional newspapers. The result of the calculations is that there should be 2 columns in portrait mode and 3 columns in landscape and on desktop computers of 1200px. However, it only requires a screen of 1320px before 4 columns would be optimal. But as the project targets the iPad screen size only 2 and 3 columns will be considered from here on.

The contents of the newspaper should consist of both personal and general news according to the scenarios. Furthermore, the typography of the application should also resemble that of a conventional newspaper. It should contain a good ratio of both graphical and textual content and should, when possible, supply multimedia content. The conclusions made from the empirical data in [Ihlström *et al.*, 2004], about exploring which features to bring from conventional to digital newspapers, was that valuation and position of the news was important. More importantly that the reader should be guided through the digital newspaper. It is, however, crucial to consider that their empirical basis is not very large. They have a qualitative selection of respondents from newspapers that have, recent to its execution, become dedicated to the project. Moreover, they have chosen 16 open questions for the respondents to answer, which

should provide some sort of basis for their conclusions. In this project it is chosen to use them as guidelines, but the choices made on this basis must be verified[1] .

[1] *Earlier statements from this paper have been backed by additional sources.*

That the reader should be guided through the newspaper using valuation of the items does, however, fall in line with the editorial mix, which also have been used in conventional newspapers for a long time. This suggests control of the temporal, spacial and relational values of individual articles and between them. Also, using the Gestalt principles [Tidwell, 2010] suggests providing a visual hierarchy so the user can see the relative importance of the page elements and the relationship among them.

Some technical requirements have also been gathered from the explored literature.

[Díaz and Gervs, 2005] suggests to use article excerpts in addition to the navigation using clearly marked sections and article headlines, and that these should be personalised. The results from [Ovesson and Wikström, 2005] suggests that the menu with clickable sections should be placed on the left side of the screen, but they could have been biased as it was already placed there in the tested prototype. In, addition, they found this as a good choice as they recognised it from the web. A menu in the top of the page would therefore also be in line with their findings, as it is a general pattern of the web [Tidwell, 2010]. In addition, it would take up less space in the view, leaving more room for the general purpose of the application, namely reading. The menu items will work well as the user defines the content of them. Furthermore, it would aid the user to relate more to these divisions if it is possible for him to name them himself.

Furthermore, it should be possible for the user to get an overview of the headlines contained in a section. This could be done by just having a list of the headlines, or by using the overview plus detail pattern presented by [Tidwell, 2010].

Many articles discuss different ways of represent the user's interests. It seems, however, that both [Díaz and Gervs, 2005] and [Billsus and Pazzani, 2000] generate good results with a

dynamic short-term user model in combination with a static long-term user profile.

Rewrite to fit list. Maybe not divide into functional and non-functional.

Finally, the implementation of a community in the application should be done with the possibility of sharing the story on different social networks, but could also include comments on articles, as suggested in the scenarios.

These requirements can be summed up in the following list:

- Summery of few most relevant articles on the front page
- Clear section and article headlines and personalised article excerpts to ease navigation
- Menu of section, that is always visible
- Paged navigation through sections
- Overview of article headlines
- Personal and general news
- Layout, typography and design should be familiar to a newspaper
- Visual hierarchy
- Balanced columns should be divided into screen size chunks
- Serve multimedia content
- The reader should be guided through the newspaper using valuation of the items in terms of categories of the editorial mix
- Combination of long-term and short term interest model of the user
- Incorporate community and social networking

3.4 THE EDITORIAL MIX

The task at hand is to decompose the spacial, temporal and relational features of the composition of articles that provides the best satisfaction of the individual user preferences into constraints. This section analyses the existing literature on reading behaviour of conventional and digital newspaper and derives constraints to compose the editorial mix of.

The reading behaviour of conventional newspapers differs from of them read digitally. In the experiments done in [Holmqvist *et al.*, 2003] it is concluded that the net paper[1] readers read stories thematically close to their own specific profession or interests. So it is important to provide this setting for the reader. The readers also used the front page as a provider of main entry points. And finally, the readers "claim to scan more in order to find the two or three stories they will read in the net paper" which is explained by the poorer chances of links catching reader interest. However, it could be possible to attract the reading behaviour from conventional newspapers onto digital platforms - it is certainly interesting to see an equal analysis of the reading behaviour of tablet computers, which calls for more in-depth reading. If a digital platform are to attract more in-depth reading it requires some flow in the presentation of the articles, i.e. it requires an editorial mix, so the readers do not feel like they have left the main trail [Holmqvist *et al.*, 2003].

[1] *Newspapers on the Internet.*

To attract reading behaviour of conventional newspaper it is worth while understanding readers expectations of these. [Holsanova *et al.*, 2006] confirms a summery of reading behaviour assumptions from [Kress and others, 1999] on conventional newspapers using eye-tracking measurements:

- Readers prefer the most general information at the top and the most specific information at the bottom of the semiotic space.
- Readers look for the most important information in the centre of the page and less important information on the periphery.
- Readers look for paratexts[1] .

[1] *[Genette, 1997] defines paratext as those productions accompanying a text, such as an author's name, a title, a preface, or illustrations.*

And, two are not confirmed, but not declined either:

- Readers look for graphically salient elements; however, it is important to bear in mind that 'what is made salient is culturally determined' [Kress and others, 1999].
- Readers follow elements connected to each other by framing devices such as lines and arrows.

That the most important information should be in the centre of the page will be hard to attract on digital platforms because of the limited space. Because of the screen size only one or two, and in some cases three, articles are shown at a time and the user will have to scroll to see the next items. However, the relation between adjacent articles can still be controlled, so a featured[1] article should be adjacent to some smaller, but still very relevant articles. This will hopefully attract the same behaviour, but of cause needs to be confirmed.

[1] *A featured article means providing it with more space than others, a central position and it is often accompanied with graphically salient elements.*

Also, a central position is hard to obtain, as many articles will be listed below each other, so a central position is here deemed to be higher than its relevant non-featured articles.

Based on the user study and the explored literature on reading behaviour the editorial mix problem can be divided in two; (1) the front page and (2) the sections.

1 The purpose of the front page is to draw attention and provide an intriguing overview of the whole newspaper. This is done by using many images and providing headlines and excerpts of the most relevant articles of the newspaper. The most relevant article should be featured in the centre with a selection of a little less, but still very relevant articles adjacent to it. A visual hierarchy should be provided so the user can see the relative importance of the page elements and the relationship among them. The front page should, if available, provide interesting articles from all sections as main entry points.

2 The purpose of each section is to provide a flow of articles relevant to a, by the user provided, topic that keeps the user interested and invites for in-depth reading. More general articles should be placed in the top of the screen and more specific at the bottom, with a featured main article in the centre. Fram-

ing and lines should guide the user to what is related and a graphical .

These descriptions can be decomposed into the following constraints.

GENERAL CONSTRAINTS

- A featured articles should be allowed to take up more space
- A featured articles should be accompanied by an image
- A featured should have a central position
- A non-featured article should take up less space
- A featured article should be adjacent to non-featured articles
- All articles should be different

FRONT PAGE CONSTRAINTS

- Every article should have a very high level of relevance to at least one of the section topics
- Most or every non-featured article should be accompanied by an image

SECTION CONSTRAINTS

- Every article should have a high level of relevance to its containing section topic
- A section should contain an article if the front page contains the article and its relevance to this section is highest
- Articles should be grouped into subjects
- The section should contain a balanced weight between graphical and textual content
- Images should be spread evenly in the section

Part II

USING WHAT WE HAVE LEARNT

Text for part 2.

4 PROPOSED DESIGN

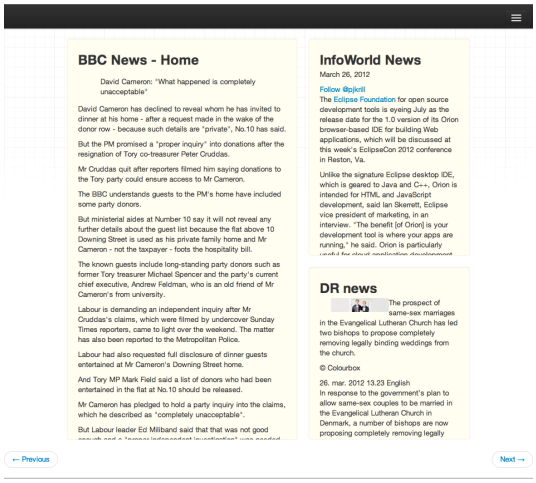
This section describes the different choices in the interface for a digital newspaper and how the application can provide the user with an personal editorial mix.

Before it is possible express the editorial mix problem formally the interface of the application must be described, because the constraints will have to make some assumptions about the application structure. The two following sections will describe the layout and typography and the interaction with the application, respectively.

4.1 LAYOUT AND TYPOGRAPHY

After the definition of the initial requirements was done the first prototype was developed. Its main features followed the requirements on turning pages, choosing an article, read it and returning to the overview of articles, see Figure 2a. And after a small preliminary survey it became clear, that a column-based layout showing full articles would be more attractive, and would provide a better opportunity to explore the editorial mix. With a column-based layout the digital newspaper would have more resemblance to conventional newspapers and therefore it was possible to apply some of the same principles of the editorial mix.

Figure 2 shows three iterations of the prototype design, which were based on the derived requirements. The third iteration of the prototype was used as the foundation for user tests. The prototype consisted of the basic navigation between topic categories, i.e. sections, and articles. Navigational choices was made in order to present the general idea of the framework, but where more crucial choices on its uses have not been made jet. This was also to encourage the test subjects to talk about



(a) Initial prototype layout with adjustable ratios between articles and a paged interface of each section.

(b) Second iteration of the prototype with an "endless" layout. Sections are placed beneath each other.

(c) Third iteration of the prototype with a column-based and "endless" layout. Sections are placed beneath each other.

(d) Third iteration of the prototype with a column-based and "endless" layout. Sections are placed beneath each other.

Figure 2: The figure shows three iterations of the prototype layout.

what uses they would have of the presented framework. However, they were also asked about the navigational structure and indeed some changes had to be done. A specification of the test can be found in Table 6.

Table 6: Test Specification

Test subjects	The test was conducted on a total of 7 test subjects of ages between 21-29, and of different sex and occupation.
Participants	Each test was done with 1 test conductor and 1 test subject.
Materials	An iPad with the application running and a computer to write notes on the test subject's statements and propositions.
Description	The test subject was presented with the prototype layout seen in Figure 2c and 2d. The test was conducted as an informal qualitative talk with a basis in the test subject's interests in such a product. Transcripts from each test can be found at `http://lestrade.imm.dtu.dk/~s062596/data/test-transcripts.zip` and a summery of the results in section A.5 on page 79.

The main points from the user test was that the newspaper should provide an overview of its contents, that it should be easy to navigate relevant new and archived articles, that the users wanted to provide relevance feedback on articles and indication of the relevance of the article. Moreover, the newspaper should provide a good balance between imagery and textual content, that white space in between articles is not a problem and that an article should be read screen by screen, even if it means dividing text into a new set of columns. It was pointed out that some problems may arise if there is not room enough in the top menu, e.g. when in portrait mode and, as the test subjects specifically requested, this could be solved by introducing a carousel-like arrow buttons to scroll the tabs, or even just using touch interactions. Moreover, the user should be able to read the newspaper screen by screen, meaning that trailing text should be put into a new set of columns whenever it exceeds the screen, see Figure 3 and 4.

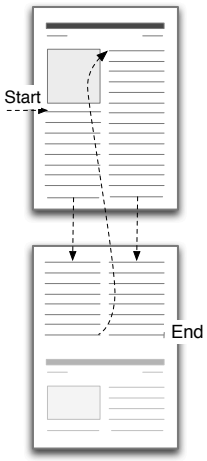


Figure 3: Reading pattern where the user has to scroll in order to see the full length of the column.

One user in particular expressed that the application had solved the problems that `http://nyhederne.tv2.dk/`[1] has and given the additional features it would provide a readable layout, easy navigation and a good overview of its content.

[1] *The website of a Danish news channel.*

Figure 5: The figure shows mockups of the layout in landscape and portrait mode, respectively.

[1] *This is also known as the black sheep problem.*

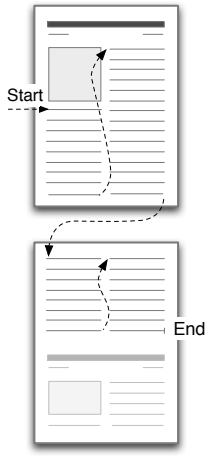


Figure 4: Reading pattern where the user can finish reading a whole page before scrolling to read the next.

No user expressed the need for any general news as presumed in the scenarios, only personalised news was of preference to the test subjects. They argued that if they wanted news of some kind, they would just create a section for it. Even so [Díaz and Gervs, 2005] argues that common sense dictates some "breaking" articles to be universally interesting and that the problem of some user may not receive them[1] could be solved be collaborate filtering. Collaborate filtering is a good way apply wisdom of the crowd to the application, which might make it stronger as the number users grows. Finally according to the user tests, it should be chosen from which period the articles should come from, as opposed to what was extracted from the scenarios and respondents from [Ihlström *et al.*, 2004] which suggests that the paper should be continuously updated.

Based on the user feedback a new design was developed. It is seen in Figure 5.

The top menu from the prototypes is kept, but arrows are added to solve the problem of overflow if the items gets too numerous. That the menu presents the items along side each other supports the fact that sections visually lie beside each other in the navigational space, i.e. the user can flick (see Figure 6) the

section and an animated transition slides the section out and the next in.

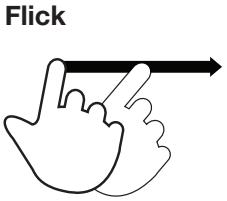


Figure 6: Touch gesture: Quickly brush surface with fingertip.

The top menu bar is given a dark colour to provide some visual contrast from content to functionality. In the new layout articles are shown in full and images are maximised. Rules of readability determines, as opposed to that on print, that small point size text work better with a sans-serif font [Tidwell, 2010]. The neutral Helvetica has therefore been chosen as the body text, whereas the article headlines are the most important thing on the page, and therefore are supplied with the largest typeface. To separate them further from the body text a serif font has been used. The section headers are assigned with a medium size, but wide, font because the user needs to navigate using this headline. However, the user already knows the name of the section (he has probably given it himself) and does therefore not necessarily need to read it - just recognise it. On this background the section headline is placed in a bar and supplied with the same light colour as the background. This makes them more neutral, but still easy to navigate using the bar. To relate the paratext with the section they are supplied with the same colour as the section bar. This should let the user associate the article with the topic of the section. Furthermore, a the line that was on top of the articles (see Figure 2d) is moved to be beside it. This is visually indicate when an article starts and ends. This will also aid to understand that the article continues if the article columns should be divided into screen sizes. Again the same colour as the section bar is used to set both the visual and contextual frame.

In the prototype the menu consisted of all the settings in a modal panel, but the test subjects wanted a division between visual tools, e.g. changing size of the font or colour scheme, and the content settings, i.e. the control of what each section should contain. The former is moved into a side menu with a button to access the latter, which kept in a modal panel. This way the handy visual tools are only one interaction away, whilst the more complex settings are hidden away with two interactions. And the overview of article headlines in the section could be done as in Sublime Text 2[1] , see Figure 7

[1] *Sublime Text 2 is a text editor for coding.*

Figure 7: The figure shows the overview plus detail function in Sublime Text 2.

This overview could be placed in the side menu and show a larger (and readable) scale of headlines and the user should be able to see the images further down in the newspaper.

4.2 INTERACTIONS

In Figure 8 is the navigational of the application structure outlined.

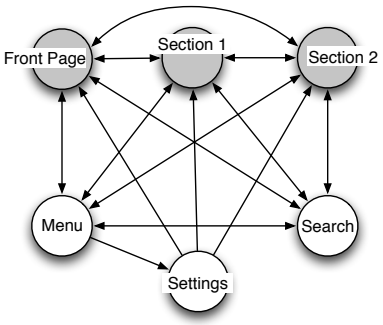


Figure 8: The figure shows the navigational structure of the application.

The three sections are holds the contents of the newspaper, which can be extended with additional sections through the settings menu. The sections lay along side each other in the navigational space and it is possible to navigate to sections beside the current through the flick gesture, but through the top menu it is possible to go directly to any of the sections. Because

the top menu is visible at all times except when in the settings menu, it is possible to go directly to any of the sections and do a search anywhere from the application, except of cause from the settings menu. This means that the application is very interconnected and since the settings menu is meant to be used rarely the extra navigational step does not matter. The settings menu should only be used the first time the user opens the application to adjust the basic settings and then afterwards only to correct if the application does not comply with the user preferences, or when the super user wants to adjust the settings. In a perfect world the user would never have to open the settings menu to adjust anything, only observe while the application learns the user interests and delivers what is expected.

When the user is at first presented with the application he should have as a direct path as possible leading to actually reading articles, which is of main user needs. He is presented with a form to make choices about the contents of the newspaper. The application provides the possibility for choosing whether the front page should be visible or not. This functionality is given to the user that would rather just have his sections and no front page. After this the user can choose the topic for the first section from a list of predefined topics. If the topic he is looking for is not in the list he can choose to fill out some keywords to cover his interests. After this he can provide it with a name for the section and choose to add another section or save his user profile. It is also possible for him to choose how many articles he would like in each section, including the front page. Figure 9 on the next page shows a mockup of the settings menu.

To be able to solve the editorial mix problem using CP it must be expressed as a COP and the presented constraints must be translated to logical constraints. This is done in the following section.

4.3 PROBLEM REPRESENTATION

The division of rules between the front page and the section can be kept in the problem representation, which will, as we will see later, be the source for the possibility of incorporating lazy loading of each section. This section will therefore present

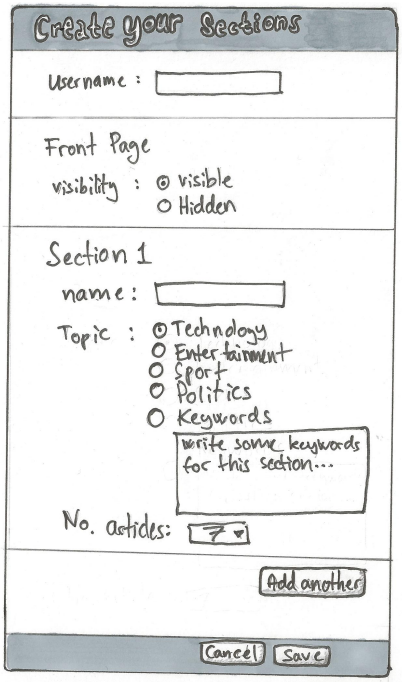


Figure 9: The figure shows mockup of the form that constitutes the settings menu.

a general problem specification for a section, which can be used in every section and on the front page, with varying constraints. Also, the front page will through this section and the following be referred to as a section, and specifically as section 0.

The set of values, $\mathcal{V}$, from equation 1 on page 8 in the problem is represented by a library of currently available articles and the set of variables, $\mathcal{X}$, is represented by the available positions in the section. Each variable can then be assigned a value in the form of a specific article and when a complete assignment satisfies all constraints, a solution has been found. An article consists of a set of attributes, e.g. a date, the number of words in the article and a number indicating a relevance. Constraints are defined on variables and through them bound to their specific places in the section.

In the following the constraints presented in section 3.4 on page 28 will be formulated as logical constraints divided into general constraints for all sections and specific constraints for the front page and other sections, respectively. Constraints defined on variables with letters, e.g. x_a and x_b, means that the constraint is defined for every combination of variables from the problem. Hard constraints returns whether it is satisfied and preference constraints returns a violation, where 0 means not violated.

GENERAL UNARY CONSTRAINTS

$$\left\{\begin{array}{rcl} \texttt{time-frame}(x_a) & \rightarrow & x_a.date >= today - 7, \\ \texttt{featured-space}(x_a) & \rightarrow & x_a.featured = \texttt{false} \vee \\ & & x_a.columns = 2 \vee \\ & & x_a.columns = 3, \\ \texttt{nonfeatured-space}(x_a) & \rightarrow & x_a.featured = \texttt{true} \vee \\ & & x_a.columns = 1 \vee \\ & & x_a.columns = 2, \\ \texttt{featured-image}(x_a) & \rightarrow & x_a.featured = \texttt{false} \vee \\ & & x_a.has\_image = \texttt{false} \end{array}\right\} \quad (2)$$

Where n is the number of positions and therefore variables, in the section and *today* is variable that holds the current date. The `time-frame` constraint is set to include articles from a week ago, but can of cause be adjusted. As the layout is defined in 2 and 3 columns the `featured-space` constraint is satisfied only when the article fills out 2 or three 3 columns. Likewise with the `nonfeatured-space` which is only satisfied with articles that fills 1 or 2 columns. `featured-image` is a constraint, to control that a featured article should have an image. These three constraints could be used as they are, but could also function as assignments along with a calculation of the relevance to base the rest of the constriants on, i.e. to determine that a featured article is an article with many words, has an image and has a lot of relevance; otherwise it is not.

GENERAL BINARY CONSTRAINTS

$$\left\{\begin{array}{lcl}
\texttt{featured-adj}(x_a, x_b) & \rightarrow & \texttt{not adjacent}(x_a, x_b) \vee \\
 & & (x_a.featured = \texttt{false} \wedge \\
 & & x_b.featured = \texttt{false}) \vee \\
 & & x_a.featured = \texttt{true} \wedge \\
 & & x_b.featured = \texttt{true}, \\
\texttt{featured-pos}(x_a, x_b) & \rightarrow & \texttt{not adjacent}(x_a, x_b) \vee \\
 & & (x_a.featured = \texttt{false} \wedge \\
 & & x_b.featured = \texttt{false}) \vee \\
 & & (x_a.featured = \texttt{true} \wedge \\
 & & x_a.position > x_b.position) \vee \\
 & & (x_b.featured = \texttt{true} \wedge \\
 & & x_b.position > x_a.position), \\
\texttt{adj-subj}(x_a, x_b) & \rightarrow & \textbf{if}(\texttt{not adjacent}(x_a, x_b)) \\
 & & \textbf{return } \texttt{max}(0, \\
 & & 64 \cdot \texttt{similarity}(x_a, x_b)^2 \\
 & & -64 \cdot \texttt{similarity}(x_a, x_b) \\
 & & +15.36) \\
 & & \textbf{else return } 0
\end{array}\right\} \quad (3)$$

Where `max(number, number)` is a function that returns the maximum of the given numbers and `similarity`(*variable*, *variable*) is a function that returns the mutual similarity between the values of two given variables. These general binary constraints control most of the editorial mix, because they express that to featured articles should not be placed adjacent to each other, that featured articles should be placed higher than its adjacent non-featured articles and that adjacent articles should have the same subject. The latter is a preference constraint because some uncertainty is introduced when controlling that two articles should have the same subject. That articles have the same subject can approximately be determined by how similar they are to each other, i.e. if the articles are much too similar they could be articles on the same story, and if they differ too much, they may be on a different subject. However, in between should roughly determine that the articles are on the same or a similar subject, which is what is needed. In the `adj-subj` constraint this is done by the returning a violence based on the parabolic function seen in Figure 10.

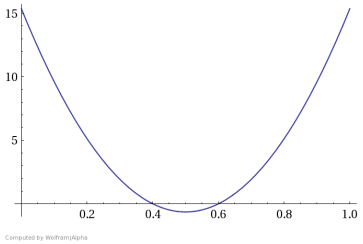


Figure 10: Plot of $64x^2 - 64x + 15.36$.

If the similarity of the two articles are below 0.6 or above 0.8, than its distance to this point will determine its violence by the parabolic function. These numbers are of cause adjustable and will vary according to the selected similarity function, but are given to show a more expressive example.

GENERAL GLOBAL CONSTRAINTS

$$\left\{ \texttt{all-diff}(x_1, \ldots, x_n) \rightarrow \bigwedge_{i=1,\ldots,n} \bigwedge_{j=1,\ldots,n} \texttt{similarity}(x_i, x_j) < 0.9 \right\} \tag{4}$$

The final of the general function determines that every article must be different. In the constraint this is expressed by checking if the similarity between the articles is too high, it could also be done by just checking the id, but this might introduce more of the same story, but from different content providers.

FRONT PAGE CONSTRAINTS

$$\left\{ \begin{array}{lcl} \texttt{main-stories}(x_a) & \rightarrow & \bigvee_{i=1,\ldots,g} \texttt{relevance}(x_a, i) >= 0.75, \\ \texttt{nonfeatured-image}(x_a) & \rightarrow & \texttt{if}(x_a.\mathit{has\_image})\ \texttt{return}\ 0 \\ & & \quad \texttt{else return}\ 1 \end{array} \right\} \tag{5}$$

Where g is number of sections and $\texttt{relevance}(\mathit{variable}, \mathit{section\ number})$ returns the relevance of the given variable in the given section number. The former constraint expresses the need for only articles of high relevance on the front page and the latter that nonfeatured articles preferably also should have images on the front page.

SECTION CONSTRAINTS

$$\left\{\begin{array}{rcl} \texttt{topic}(x_a) & \rightarrow & \texttt{relevance}(x_a, k) >= 0.65, \\ \texttt{fp-article}(x_1, \ldots, x_n) & \rightarrow & \bigvee_{i=1,\ldots,n} \bigwedge_{j=1,\ldots,m} x_i.id = a_j.id, \\ \texttt{image}(x_t, x_{t+1}, x_{t+2}, x_{t+3}) & \rightarrow & \texttt{if}(\bigvee_{i=t,\ldots,t+3} x_i.has\_image) \\ & & \texttt{return } 0 \texttt{ else return } 1 \end{array}\right\} \tag{6}$$

Where k is the current section number, $a_1, \ldots, a_m$ is a list of articles from the front page that should be contained in this section and x_t are variables where t fulfils the equation $(t-1)\%3 = 0$. In other words, it is defined for every fourth variable. The three section constraints controls that articles should be relevant to the topic, that this section contains all necessary articles from the front page and that at least every fourth article should should hold an images.

These constraints are more or less simply translated from the presented textual constraints into logical functions and luckily only few contains multiple or every variable in the problem. It is in many cases profitable to choose a different representation of the problem if constraints builds on multiple variables. This can be done e.g by a tree decomposition or a reduction of the constraints to binary constraints. To compose the tree decomposition, first the constraint graph[1] needs to be considered. In this problem a complete graph of all variables will emerge, as it contains constraints that holds every variable. The tree decomposition is done by dividing the problem into subproblems and viewing them as "mega-variables" with their set of solutions as their respective domains, so the outermost problem becomes a tree. This works well if no subproblem is too large [Russell and Norvig, 2003], but for a complete graph of n variables the tree decomposition will become a problem of $n-1$ subproblems each with $n-1$ variables, which is not very efficient. As each subproblem can be solved independently the decomposition could be done continuously until small enough problems emerge. However, the time is of cause dependent on the branching factor of the tree, so this is not efficient either. Reduction of constraints to binary constraints can be done by introducing new variables, with constraints that defines the relation to this and old variables in the new problem.

[1] *A constraint graph is a graph where vertices are variables of the problem and their links are the constraints that binds them.*

[Russell and Norvig, 2003] however argues that it is possible to design special-purpose inference algorithms that only handles global constraints and in practise this can be done by counting the number of variables the constraint is defined for and then deciding how to handle it.

4.4 CHOICE OF ALGORITHM

Several algorithms exists to solve CSPs and many of them can be converted to work on COPs. A depth-first search using BACKTRACKING can e.g. be used. The search continues with a descendant spanning the whole tree, which is independently viewed as a new CSP. The search backtracks to the parent node, whenever an empty domain is encountered. The algorithm stops with the first assignment that satisfies the CSP if a single solution or inconsistency is sought. Because no information, other than that given in the problem formulation, is used to solve the problem it is a uniformed search and therefore not expected to perform very well [Russell and Norvig, 2003, p. 73].

With the use of heuristics the search can be improved. An example of this is the BRANCH AND BOUND search which work on COPs, but can handle CSPs as well. The branch and bound heuristic bounds the search by an objective function, that returns the current best assignment. States after a worse assignment is encountered are therefore not considered. Another example of a heuristic function is the *most-constrained-variable* (MCV) heuristic. This function always selects the variable that appears in the largest number of constraints, to be assigned next. When used with backtracking, the performance can be greatly improved [Apt, 2006, pp. 337].

Constraint Propagation is another type of heuristic. Where the branch and bound heuristic is concerned with which variable to select, constraint propagation is concerned with the implications of the assignment of a variable. This means that if the assignment of a variable removes the possibility of some values to others in the solution, the values are removed from these domains. An example of this is *arc consistency*. If every constraint is represented by an arc, like in the constraint graph, but directed, arc consistency is when there for every value exists some

value that it is consistent with. Arc consistency can be applied multiple times to obtain *path consistency* and until no inconsistency is left, which is the property of the MAC (Maintaining Arc Consistency) algorithm.

FORWARD CHECKING uses propagation whenever a variable is assigned to a value to detect inconsistency. It does, however, not detect for new inconsistencies after the removal of values. The worst-case time for MAC is $O(n^2d^3)$ for n arcs and d values in a problem [Russell and Norvig, 2003, p. 146]. Even so will, MAC, still not find every inconsistency [Apt, 2006, p. 150] and [Russell and Norvig, 2003, p. 147].

Finally there is the MIN-CONFLICTS algorithm. It is the result of the application of *local search* to CSPs. It uses the min-conflicts heuristic, which chooses the value that results in a minimum number of conflicts. The algorithm is shown in Figure 11.

```
MIN-CONFLICTS(csp, max_steps) returns a solution or failure
    inputs:
        csp, a constraint satisfaction problem
        max_steps, the number of steps allowed before giving up
    current ← an initial complete assignment for csp
    for i = 1 to max_steps do
        if current is not a solution for csp
            return current
        var ← a randomly chosen,
              conflicted variable from VARIABLES[csp]
        value ← the value v,
                that minimises CONFLICTS(var, v, current, csp)
        set var = value in current
    return failure
```

Figure 11: The MIN-CONFLICTS algorithm for solving CSPs by local search. The initial state may be chosen randomly or by a greedy assignment process that choses a minimal conflict value for each variable in turn. The CONFLICTS function counts the number of constraints violated by a particular value, given the rest of the current assignment [Russell and Norvig, 2003, p. 151].

Local search are algorithms that do not care about which path they take to a solution, just that they find one. The

MIN-CONFLICTS algorithm operates by moving to neighbours from a current state using the *min-conflicts* heuristic, i.e. selecting the value that results in a minimum number of conflicts with other variables. It can be applied to both CSPs and COPs. In the latter the techniques of *hill climbing* and *simulated annealing* can be used to improve the search [Vossen, 2005]. Local search has proved very effective in solving many CSPs and COPs. For the MIN-CONFLICTS algorithm to work an initial complete assignment is needed, so it can operate on a current state. The efficiency is depending on this initial state [Russell and Norvig, 2003].

In [Russell and Norvig, 2003, p. 143] a thorough survey on commonly used algorithms efficiency on commonly known CSPs is conducted and the results from the n-Queens problem is listed in table 7.

Table 7: Comparison of various CSP algorithms on the n-Queens problem. The algorithms from left to right, are simple backtracking, backtracking with most constrained variable (MCV) heuristic, forward checking, forward checking with MCV, and MIN-CONFLICTS local search. Listed in each cell is the median number of consistency checks (over five runs), required to solve all n-Queens problems for n from 2 to 50; note that all entries are in thousands (K). Numbers in parentheses mean that no answer was found in allotted number of checks [Russell and Norvig, 2003, p. 143].

Problem	Backtracking	BT+MCV	Forward Checking
n-Queens	(> 40,000K)	13,500K	(>40,000K)

Problem	FC+MCV	Min-Conflicts
n-Queens	817K	4K

Table 7 shows that the MIN-CONFLICTS local search is by far the most efficient algorithm for solving the n-Queens problem. The initial assignment of the editorial mix problem can be randomly chosen or by a greedy algorithm and the neighbour states can be generated by selecting a new value for a variable or swapping the values of two. The conflicts function could return the number hard constraints violated plus the sum of violation of the preference constraints [Apt, 2006, pp. 372][Russell and Norvig, 2003, p. 150]. It is afterwards up to the solution check, to check if all hard constraints are satisfied and if an optimal solution has been found for the preference

constraints. In choosing a variable MCV can be used to guide the search than the proposed random conflicted choice.

Because the local search techniques can be applied to the MIN-CONFLICTS algorithm and because of its iterative nature this algorithm is chosen to solve the problem. The property that the algorithm starts by an initial complete assignment and refines the it, makes it attractive for the purpose of composing the newspaper and solving personalisation problems in general. Either it returns an optimal solution or it the returns the current best assignment if there is no iterations left.

4.5 APPLICATION STRUCTURE

After the description of the interface, the problem and choice in algorithm it is possible to discuss how the client side should handle user requests and the web server should handle the requests from the client side. In Figure 12 is shown a use case diagram of how the client side and web server handles use cases.

In the diagram and extra use case has been added to show how the system should handle the initial topic selection by the user. The user is able to select relevant topic categories to get an easy start with the application and his choices are thereafter saved to the user profile for later use. The user can thereafter get an overview of the content from the front page or read articles from a selected topic category, both in a nice readable layout and in a composition based on the editorial mix. Moreover, it is possible to an overview of the articles within a section from a the list of headlines from articles contained within it.

The next chapter will present the technical specification of the implementation, which constitutes the product of this project.

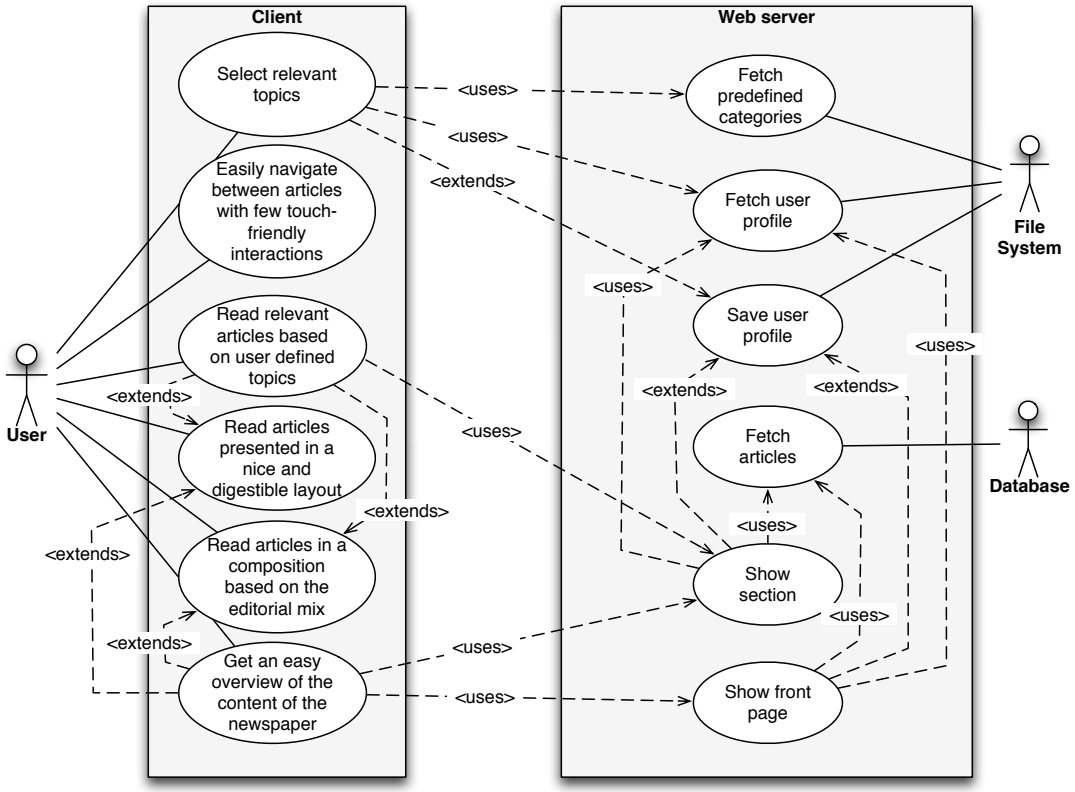


Figure 12: The figure shows the use cases of the system and how the client side handles them and how the web server acts to accomplish requests from the client side.

5 TECHNICAL SPECIFICATION

This section describes a technical specification of a direct implementation of the design choices made in the previous chapter in terms of client side and server side tasks.

The proposed design presented in the previous chapter has accounted for the presented requirements and user needs, but there are some requirements that will not be implemented due to prioritisation and time limits.

The in chapter 4 presented application incorporates the user model, but this chapter will only present how to collect the necessary user data and use it in the application – and not implement it. It will, however, describe which meta data is needed and how to acquire it.

The social aspects of the system is of cause an important part of it and in terms of the business case, social media are easy channels for awareness. [Tidwell, 2010] even states her editorial mix pattern as a social media pattern. Nonetheless, these will not be implemented in the presented application as it does not contribute with new knowledge to field. The gathered news articles to used in this project contained both images and videos, but only images will be considered here. It is however easy to implement the support for videos as the same space allocation principles applies.

Also, the personalised summaries have already been very well explored in the paper [Díaz and Gervs, 2005] and this project will not try to compete with it, so only the first few sentences will constitute the excerpts from articles to be used on the front page.

The categories used to compose sections of is by no means a full list and they have not been verified, but they are some of the most recurring in popular news sites and are used just in

order to proof the concept is possible. Thus, their definitions are not comprehensive either.

Finally, only a selection of the full list of the editorial mix constraints, presented in the proposed design, will be implemented. Furthermore, before the user test was done the implementation had already started leading to the implementation of a fairly complex layout constraint to minimise white space. It turned out that some white space was actually a user preference, but the implementation of the constraint will be presented nonetheless, as it shows a good example of what is possible with the system.

5.1 SERVER FOR ACQUIRING AND MINING DATA FOR PERSONALISATION

The [Dub, 2012] proposes 15 meta data elements for documents, i.e. Title, Creator, Subject, Description, Publisher, Contributor, Date, Type, Format, Identifier, Source, Language, Relation, Coverage and Rights. The articles in this project have been acquired using the [Readability API, 2012] to scrape web pages of articles given by links from RSS-feeds. This way it is possible to get the full article. Under normal circumstances, these would probably be supplied by a content provider, say through an agreement with newspaper companies or social networks. The Readability API provides several meta data elements for the parsed articles; a domain, a title, a article URL, a URL for a lead image, the author, an article excerpt, a word count and a date of publish. This satisfies many of our needs, but there are still some very crucial calculations to be done, i.e. the article relevance according to user topics and similarity between articles.

The analysis of article relevance and similarity to other articles will use the same two types analysis, namely a document using the WordNet and entity comparison using Open Calais. These will be presented in the following two sections.

Er der argumenteret for at bruge keywords?

5.1.1 *WordNet Enriching Articles*

WordNet is a large lexical database of English words and their relationships in the form of different graphs. WordNet is based on *synsets*, which is a set of synonyms which describes different meanings of the same word. WordNet has hyperonymy and hyponymy graph for noun synsets, which is based on the ISA relation between words. A *hypernym* relation is a generalisation, e.g. a hypernym for a bed is a piece of furniture; and a *hyponym* relation is a specification, e.g. a hyponym for a bed is a bunkbed. For nouns there also exists the meronymy graph, which is a graph describing the part-whole relation; a chair e.g. has a back, a seat and legs. Also, parts are inherited by superordinates, e.g. if a chair has legs, then an armchair has legs as well. Furthermore, WordNet has a graph describing elaboration, i.e. *troponyms*, of verbs synsets, adjectives organised in terms of antonymy and adverbs which can be be described in terms of adjectives. The synsets, the hypernym and hyponym graph and the troponym graph of WordNet are the most interesting, because they describe different meaning of a given word, whereas the others describes relation to other words.

The initial approach involved computing the tf-idf similarity between articles and the topics and articles in between using the Python libraries for this [Bird *et al.*, 2009]. This approach works on a bow (bag-of-words) with key words and weights representing a single item. The weight is computed by the number of occurrences in the provided text and a cosine distance determines the similarity. However, Python also provides an interface for working with WordNet. This opens the door to a more in-depth analysis of the obtained news articles. [Bouras and Tsogkas, 2010] presents an algorithm for enriching articles using WordNet's hypernym graph. A subgraph of WordNets hypernym graph is generated by the top 20% frequent keywords of an article and weighted by:

$$W(d, f) = 2 \cdot \frac{1}{1 + e^{-0.125(d^3 \frac{f}{TW})}} - 0.5$$

Where d stands for the node's depth in the graph (starting from root and moving downwards), f is the frequency of appearance of the node to the multiple graph paths and TW is the total

number of words used to generate the hypernym graph. An example of such a hypernym graph is seen in Figure 13.

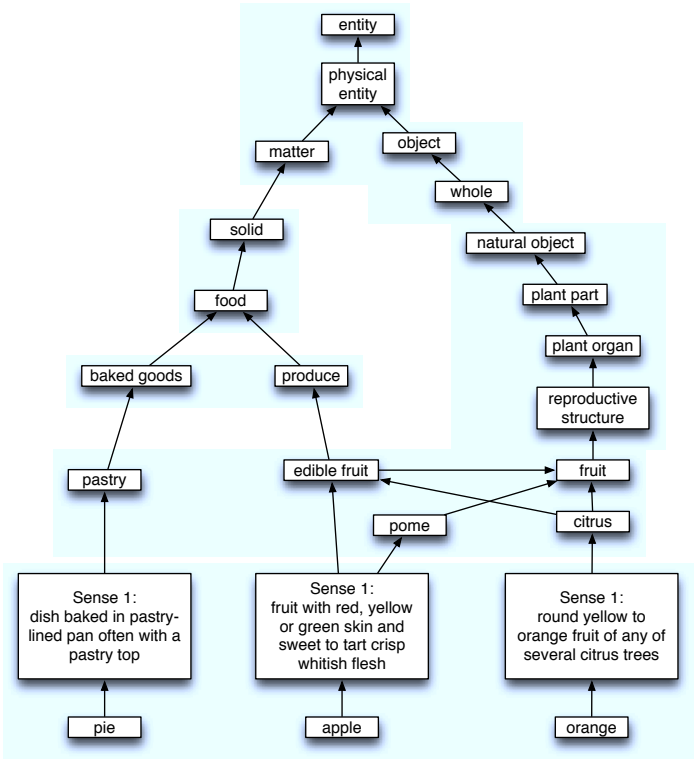


Figure 13: The figure shows an example of a hypernym graph that could be generated by the WordNet Enrich algorithm.

To be able to work with hypernyms, words from articles must be converted to synsets. For each word there exists a synset for each use of the word, with the most frequently used first. Every synset is included at this point, but in a later stage this could be further focused by only using the top n. An analysis on how many percent of the words

WordNet distinguishes among Types (common nouns) and Instances (specific persons, countries and geographic entities). (http://wordnet.princeton.edu/)

level is only a part Python implementation of WordNet

5.1.2 *Meta Data from Open Calais*

5.1.3 *Computing Similarity*

- Storing Data

5.2 CLIENT FOR COMPOSING THE EDITORIAL MIX AND PRESENTING THE ARTICLES

- Dynamic Page (created from form)

css: conditional styling

From the description of the editorial mix it is possible to model
General requirements:

5.2.1 *Interface*

In Figure 14 is shown a sequence diagram of what the system does in order to display the front page (or a section), when the user opens the application.

When the application is opened, or the application is changed to display a section, a background worker is initialised to compose its containing mix of articles. Of cause, if the mix of articles in a section, or the front page, have already been computed, it should not have to recompute it. The background worker needs to get both the user preferences of the chosen topic category and articles that potentially fit the user preferences. While the worker computes the editorial mix it sends messages to the user interface about the progress, to provide feedback to the user. When it finishes the user interface is asked to display the articles.

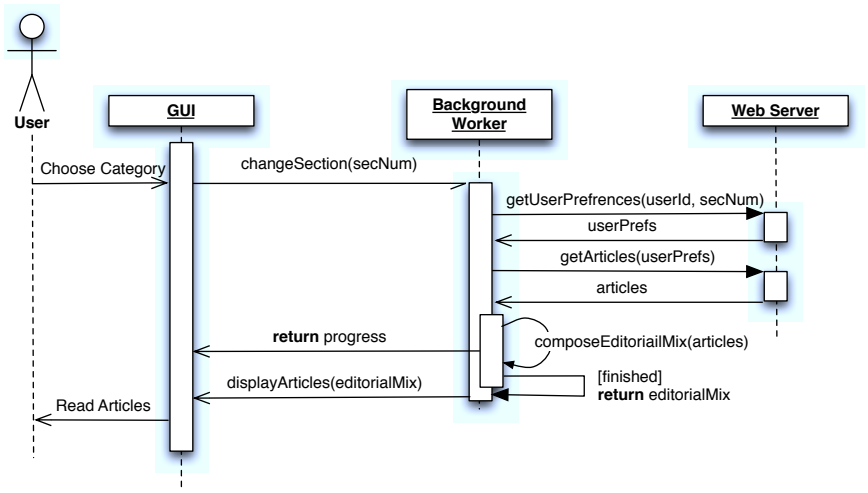


Figure 14: A sequence diagram from when the user chooses a topic category until reading articles. `secNum` is a section number, (front page is section 0), `userId` the user id, `userPrefs` the user preferences on a given section and `articles` is a library of articles to compose the editorial mix of.

5.2.2 *Constraint Personalisation Library*

5.2.2.1 *Data Structure*

5.2.2.2 *General Purpose Solver*

5.2.3 *Backend worker.js*

- Lazy Loading Sections - Knowledge Passing

worker i tråd for sig selv, men samme proces

worker.js to create a background worker to perform the constriant programming.

Model-View-Control using backbone.js

Paging: single page web apps + manipulation the browser history https://developer.mozilla.org/en/DOM/Manipulating_the_browser_history

Preference ordering of hard constraints or division between preference constraints and hard constraints.

Assignment from library in stead of arbitrary assignment? The latter is a more hypothetical approach. Providing the library as a constraint, where each variable assignment must have a unique combination from one of the possibilities of the constraint. (sim,breaking,chars,date,sections?,columns):list The former introduces an implicit constraint in that the general purpose solver can only choose from the library, thus can only choose a combination of values that exists.

Ranges can be optimised in space by converting them to integer ranges. This can be done by setting min = 0 and max = (b-a)/gap.

Furthermore each subdomain should be able to be represented by a set of ranges and atomic values. Propagating through values causes many iterations and a whole range may be discarded by looking at its maximum and minimum value. However, if the range holds a potential valid value (solution to a variable) it can be divided into smaller ranges and their minimum and maximum values may be examined. This divide-and-conquer technique may continue until the search reaches atomic values (determined by the gap value of the range). If some atomic values and ranges seems to fulfil the constraints they should be returned. And the subdomain now consists of both ranges and atomic values.

Optimal/promising fixed budget computation

The library could take any combination of constraints and then organise them into conjunctions of disjunctions, with the constraints taking fewer values first.

In the implementation this is done by hand, so the program takes conjunctions of disjunctions of constraints organised with constraints that takes fewer values first. Constraint weighing could also help organising the disjunctions and furthermore lead the search to concentrate on variables that is bound by these constraints. (p. 222 AIRussel).

Constraints should point to specific variables, this makes it somewhat rigid/ineffective because I have to write at global constraint that accounts for everything (ineffective in propagation – might also be a problem if it does not show progress in

changing values, i.e. it is a hard constraints and not returning a cost of the set of values.) or divide it into smaller constraints separated by an 'or' (v). The latter is ineffective because there would should be a combination of constraints accounting for every situation, e.g. if the first variable is satisfying an unary constraint, the next say 3 variables (if the problem holds 4 variables) could satisfy three unary constraints, an unary and a binary (two combinations exists) or a constraint that takes three variables. This grows fast with the number of constraints.

Does it make sense that a continuous range cannot have specific values removed? Should it be possible for it be divided into subranges if the user decides to remove a range of values in between its domain of [min;max]?

Pool of workers to compute sections and send results to another worker to handle front page articles. Or, a lazy load approach where the front page is computed and then sections are computed with hard constraints that manages placement of articles within the given sections, e.g. if an article from the front page has the potential to be placed in only one section the constraint should state this, but it would demand cross-worker-constraints to handle if an article from the front page has the potential to be placed in more section (i.e. `xor`). color

Reducing constraints to binary constraints.

Implement visual difference between featured articles and non-featured articles.

Part III

LEARNING WHAT WE HAVE USED

Text for part 3.

6 DISCUSSION

Apple vs. apple

7 EVALUATION OF THE SOLUTION

Evaluate using the in [Diaz *et al.*, 2001] and [Esteban *et al.*, 2000] presented method. Their categorisation based on few keywords (5 as the lowest) to represent a category resulted in poor evaluation, this gives a good motivation for including more keywords and using Wordnet to enrich the set of keywords.

It could be interesting to evaluate the precision and recall points based on: news items per section, news items per category, maximum number of news items per message required by the user, general relevance of the contents of a given day for a given user, etc. as [Diaz *et al.*, 2001] proposes.

[Diaz *et al.*, 2001] also raise the problem of precision in finding news within based on a single day. This can hopefully be handled by having the user specify in which period of time he wants news and maybe notify the user that solutions might be inaccurate if a limited period of time is chosen, or just limit the user to specify 24 hours as a minimal value.

7.0.4 *Initial Test*

NOT ONLY 5 test persons [Nielsen, 2012]. This has been discarded by many.

7.0.5 *Result*

7.1 TEST

7.1.1 *Layout*

classify items as first and second articles and use layout to distinguish them.

columns constraints, white space and odd placement of articles.

We follow a strict vertical structure, but there is a lot of work to be done with the horizontal structure

In order to come closer to understanding what newspapers does it could be interesting to analyse their component structure, e.g. using [Liu *et al.*, 2001] algorithm.

At an early stage the paging of a section was discarded, because a preliminary test (ask around) showed that users wanted to scroll down to see the full section. This was also necessary if a full-length articles were to be shown, but the reading behaviour analysis suggests that articles should be divided into chunks of subjects, which may be better to visualise using pages. This way the featured article could be shown in a longer length excerpt and stories on the same subject could surround it with only headlines, images and short excerpts shown. If the user then wants to read one of the articles in full length he can select it and the full article could be displayed, using the full size of the screen.

7.1.2 *Content*

The system can be used for automatic classification of articles. Of cause, then a sufficient list of categories and their definitions must be used. This can either be retrieved by the list of

Google News categories[1] and a Wordnet enriched list of key words from Google News list of suggested keywords[2] or by the root terms presented in [Abuzir and Vandamme, 2002]. These can later on be refined by information retrieved by the user behaviour in the system and manually removal of false negatives? However, also more advanced techniques of text classification could be used in later stages of the system, like one presented in [Esteban *et al.*, 2000].

Maybe find a better example of text classification.

Use of automatic generation of personal item summaries [Díaz and Gervs, 2005]

Use geotargeting to supply local news.

Use a thesaurus and predefined root terms as in [Abuzir and Vandamme, 2002] which improves classification; semantic knowledge is more general than keywords.

scattered ads [Ovesson and Wikström, 2005]

7.1.3 *Functionality*

Order a print copy of the newspaper

The development of the Internet from a distributor of information to a library of digital applications has deeply integrated the users in every step of an applications lifetime. It has become harder to distinguish between super users and developers, applications are branched and modified according to every need and authors can therefore no longer predict which use his application can be to another user – nor should he have to.

1 http://support.google.com/webmasters/bin/answer.py?hl=en&answer=42993.

2 http://support.google.com/news/publisher/bin/answer.py?hl=en&answer=116037.

7.1.4 *Improvements*

[Centeno *et al.*, 1999] suggest virtual communities, or individuals with common interests.

Weights on key words should be adjusted by a strength (or an uncertainty) of prediction as it is proposed in [Claypool *et al.*, 1999].

Count the number of sources have included an articles that are very similar to find the breaking factor.

How to handle that a user is not presented with an already read article? [Billsus and Pazzani, 2000] presents the nearest neighbour (NN) algorithm approach, using a tf-idf similarity, to determine whether the story is already known, i.e. the similarity to the NN read story is above a given threshold. This could be solved by keeping a library of read items (this can be done along with the tracking of which article is in focus) and then match new items against this banned list and down prioritise them if their similarity is too high. This could be done with the same polynomial function as used between articles.

"Distinguishing between short-term and long-term models has several desirable qualities in domains with temporal characteristics (Chiu and Webb, 1998)." [Billsus and Pazzani, 2000].

What things were even better or a little worse than expected regarding the methods you used to solve your problems. How could your project be improved by further work.

8 CONCLUSION

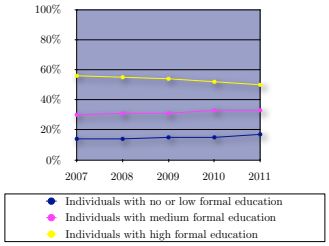
Part IV

APPENDIX

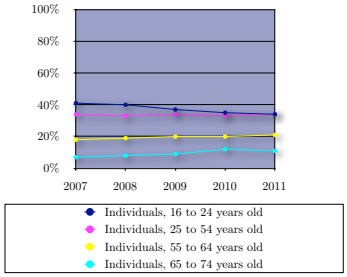
A USER NEEDS

This section will define the user needs for the application.

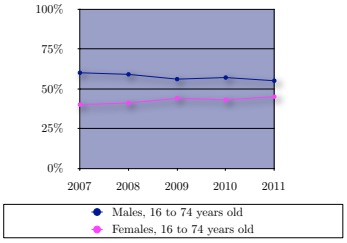
A.1 PERSONAS



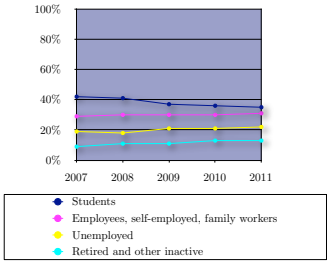
(a) Initial prototype layout with adjustable ratios between articles and a paged interface of each section.



(b) Second iteration of the prototype with an "endless" layout. Sections are placed beneath each other.



(c) Third iteration of the prototype with a column-based and "endless" layout. Sections are placed beneath each other.



(d) Third iteration of the prototype with a column-based and "endless" layout. Sections are placed beneath each other.

Figure 15: Eurostat: Individuals using the Internet for reading / downloading online newspapers / news magazines

In Figure 15 is shown the basis for the division into the following user groups:

The user groups provide the basis for the following personas.

Table 8: Eurostats distribution of individuals using the Internet for reading and downloading online newspapers and news magazines.

Description	**%**
student	35
employees, self-employed, family workers	31
unemployed	22
retired or other inactive	13
Description	**%**
high formal education	50
medium formal education	33
no or low formal education	17
Description	**%**
male	55
female	45
Description	**%**
16-24 of age	34
25-54 of age	34
55-64 of age	21
65-74 of age	11

A.1.1 *Thomas: student medium formal education male of age 21*

Thomas is 21 and a student at the Technical University of Denmark to be a bachelor of engineering in software. He is very interested in soccer and is therefore always updated on sports news. He reads about it online, newspapers and talks about it with friends. With big events he even likes to post it on Facebook. As a soon-to-be software engineer he has a natural thirst for news about technology, and he mainly reads these at home at the dormitory. wired.com, newz.dk, engadget.com, facebook.com computer, Samsung Galaxy Tab

A.1.2 *Laura: employed high formal education female of age 39*

Laura is 39 and is employed as a key account manager. She likes to be updated on strategies and economical status of rivalling companies. She is also very interested in politics and likes to discuss this subject with her friends. She reads economical news and likes to be updated on the run. b.dk, borsen.dk, twitter.com iPhone, iPad

A.1.3 *Marie: unemployed no or low formal education female of age 61*

Marie is 61 and a currently unemployed housekeeper. She spends her day looking for a job and taking care of her pet cat until her husband comes home. She mostly looks for the gossip sections or news about crime or big disasters. She also spends some time reading through the travelling guides as she dreams of going away with her husband. ekstrabladet.dk, bt.dk, nyhederne.tv2.dk computer, Lenovo IdeaPad A1

A.1.4 *Carl: retired or other inactive high formal education male of age 69*

Carl is a retired professor in psychology. He likes to discuss human behaviour and relation with his acquaintances and is very interested in cultural events. Therefore he often seeks the cultural sections and discussion fora to see what is going on. politiken.dk, aok.dk, dr.dk computer, iPad

A.2 SCENARIOS

A.2.1 *Thomas*

Thomas comes home after a day at the study, picks up his tablet computer and opens Editor from the desktop. Editor opens and shows him the front page where all the headlines stories are displayed. The main story is about a new version of the Android OS that has been released today and presses it to read more. The story opens in a full window display with quality images to match the articles. He reads the first section and feels satisfied with the amount of information, but wants to share the information on Facebook, so he clicks share button and writes a comment and posts it on his Facebook wall. He closes the article and returns to the front page. He sees a top story below the main story about Mr. Mærsk Mc-Kinney Møller who has died. It is not a story that falls into his key interests, but as the news is big he is satisfied that he got informed about it. Thomas feels like reading more about technology so he opens the menu and chooses the "Tech" section he has installed in the application. The section opens with a head line and a page number to let him know where in his paper he has navigated to and finds an article about a new multicore CPU technology. He has never been interested in CPU technology before, but finds this technology interesting after reading about it, so he opens the application settings and types in keywords about the technology under his "Tech" section to keep him updated about it. He also adjusts the ratio between general and personal news, to be less personal as he feels like he needs to broaden his horizon a bit with respect to news. He closes the settings menu

and Editor immediately starts updating the articles. Some new articles about CPU technology has been included amongst the articles in the "Tech" section after paging through the section and reading some of the most interesting articles he closes the application.

It could be nice if the key words of a story could be or is already highlighted, so he can click it and add it to his positive or negative list.

A.2.2 *Laura*

Laura is on the train on her way to a business meeting this morning and pulls out her tablet and sees she has one notification from Editor. She opens Editor to get updated on todays news. The front page is displayed and there are headlines from different top articles and a notification is shown in the corner. She presses the notification and the pages turns to show her the article, which opens in full screen. After reading it she wants to see todays headlines, so she presses the back button to return to the paper and presses the return to front page button and the paper turns pages to reach the front page. She scans the page to see if there is any big news about her rivalling companies. There is no breaking news, so she just turns the page to browse the content of todays paper. As she browses the "Politics" section of her paper she finds an article about the Prime Minister introducing a new bill about a toll ring around the capitol city. She chooses the article and it is shown in full screen. As she reaches the bottom of the article she sees the comments about it where her friends and most others are against it. She decides to join the discussion and posts a comment on the article wall. She also sees one of her friends has not commented on the article wall and decides to share the article with her as she thinks she would agree with her opinion. She presses the share button and chooses the Editor logo. A list of her friends is shown, some of them who has already read the article is greyed out, but the one she was looking for is not. So she chooses her and a notification is sent to her.

A.2.3 *Marie*

It is morning and Marie wants to check the news with her coffee in the couch, so she opens Editor from her tablet to get updated. The front page is displayed with a collection of stories as highlights of the content of the paper. It mainly contains stories about celebrities and a big disaster that has happened in japan, but there is also a story about a big political change, that she does not find interesting. So she goes to the settings menu and types in "politics" to add to her negative list. She also adjusts the personal/general news ratio to contain only personal news as she wants only news that is directed to her. She returns to the front page which is now free of political stories. Her newspaper contains many images and videos as she has set her graphical/textual content ratio more towards graphical content.

A.2.4 *Carl*

Sunday morning Carl wakes up, puts over the kettle to make a cup of coffee. While he waits for the water to boil he picks up his iPad and opens Editor to check the news. The front page opens with headlines from the different sections. There is a review article about a new show in the theatre. Carl presses the article and the system turns pages to the "Cultural" section of his newspaper and opens the article in full screen. Because the show gets good ratings he decides to order some tickets to him and his wife, which he does using the devices browser. After this he reopens Editor which opens in a display of the same article, as he left it. Carl pours his coffee and turn to the back page with the some crossword puzzles and some cartoons. He presses a puzzle that looks funny and it is displayed in a full page, where he can solve it. When he is done he returns to the page and chooses his regular cartoon to read.

A.3 BUSINESS CASE

A.3.1 *Need*

User value: personal quality and up-to-date stories enriched with quality images. This means that content providers should be chosen/verified. Same navigation as actual newspapers, but faster and with endless more content. Instantly up-to-date. Adaptive layout. Adjustable to individual user.

A.3.2 *Approach*

Personalised content in an editorial mix.

Constraint Programming: fast computation - good for optimal solutions, describes the generic solution in stead of how to solve or find it, easy to tailor the problem definition of the solution and adjust it and even let users make the adjustments - transparency.

Content providers can get to know their readers preferences better and improve the provided content.

A.3.3 *Benefit Per Cost*

Revenue flow: Content providers are paid. Income from advertisers (scattered [Ovesson and Wikström, 2005, p. 6-7]) and users. Income from selling user behaviour patterns and precise targeted commercials.

A.3.4 *Competition*

FlipBoard, Wired magazine, Zite and app with actual editors affiliated.

A.4 REQUIREMENTS

The above scenarios, user needs and business case led to the following requirements.

A.4.1 *Non-functional Requirements*

- "the clear overview of content, including a beginning and an end, the ease of use, typography and design" [Ihlström *et al.*, 2004, p. 7]
- both general and personal news (collaborate filtering solves that some news are not received, but are universally interesting [Díaz and Gervs, 2005])
- familiarity in design from printed paper [Ihlström *et al.*, 2004, p. 7]
- Design and layout from printed newspaper [Åkesson *et al.*, 2005]
- both images and videos - test
- a good ratio of graphical and textual - test
- front page should give a good overview of the content - test
- "news valuation, e.g. positioning of lead story" [Ihlström *et al.*, 2004, p. 7]
- mobility [Ihlström *et al.*, 2004, p. 7]
- continuous updates [Ihlström *et al.*, 2004, p. 7]
- "easy and intuitive navigation" [Ihlström *et al.*, 2004, p. 7]
- add video and sound [Ihlström *et al.*, 2004, p. 7]
- incorporate social community and social networks

A.4.2 *Functional Requirements*

A.4.2.1 *Calculations on number of columns*

iPad screen size:

$197 \times 148mm$ or $1024 \times 768px$

International Herald Tribue (the global edition of the new york times): $\frac{398mm}{6} = 66.333333333mm$

Børsen (uses both 5 and 6 columns): $\frac{285mm}{5} = 57mm$ and $\frac{285mm}{6} = 47.5mm$

Information (5 columns 4 on the back): $\frac{285mm}{5} = 57mm$ (back $\frac{285mm}{4} = 71.25mm$)

Guardian (5 columns): $\frac{314mm}{5} = 62.8mm$

Politiken (6 columns): $\frac{392mm}{6} = 65.33mm$

Berlingske (4 columns): $\frac{285mm}{4} = 71.25mm$

Average on the most regular columns:

$\frac{66.3+57+57+62.8+65.3+71.3}{6} = 63.283333333mm$, i.e. 3.11 columns in landscape and 2.34 in portrait.

(Average on every column width:

$\frac{66.3+57+47.5+57+71.25+62.8+65.3+71.3}{8} = 62.30625mm$, i.e. 3.16 columns in landscape and 2.38 in portraint.)

$1200px$ screen:

$\frac{1200 \cdot \frac{197}{1024}}{63.28} = 3.65$ columns, where 197/1024 is px to mm ratio and 63.28 is the thinnest column width

column size in px:

$63.28 \cdot \frac{1024}{197} = 329px$

$62.31 \cdot \frac{1024}{197} = 324px$

average $= 326px$

- 2 columns in portrait and 3 in landscape
- "open, turn pages, chose article, read and return" [Ihlström *et al.*, 2004, p. 6]
- section headlines [Ovesson and Wikström, 2005, p. 6-7]
- article headlines
- article summaries / extracts [Díaz and Gervs, 2005]
- menu w. section headlines [Ovesson and Wikström, 2005, p. 8]
- page numbers [Ovesson and Wikström, 2005, p. 6-7]
- press "like" or key word based user profile (mark self or highlighted? right click to add): positive + negative list (keywords+categories [Abuzir and Vandamme, 2002], [Díaz and Gervs, 2005] and [de Buenaga Rodríguez *et al.*, 2004])
- full screen display of article
- organise into personalised sections
- opens in front page view (summery of newspaper 8 articles) [Ovesson and Wikström, 2005, p. 8]
- adjust variables
- share directly (grey out the ones who have read it)
- comment
- see friends comments
- "The presentation schema – headline, abstract, and text, together with a relevance value with respect to the user

profile – rates the highest in terms of user satisfaction, and yet it is not the most frequent." [Diaz *et al.*, 2001]

- ability to search [Ihlström *et al.*, 2004, p. 7]
- Landscape + portrait [Ovesson and Wikström, 2005, p. 6-7]
- touch screen interaction [Ovesson and Wikström, 2005, p. 6-7]
- Functionality from online newspaper [Åkesson *et al.*, 2005]
- Name of columnist [de Buenaga Rodríguez *et al.*, 2004, p. 4]
- Transparency of implicit relevance feedback (see/modify current weights of categories) [de Buenaga Rodríguez *et al.*, 2004, p. 7]
- dynamic short-term + static long-term user profile [Abuzir and Vandamme, 2002], [Díaz and Gervs, 2005] and [de Buenaga Rodríguez *et al.*, 2004]
- relevance feedback [Abuzir and Vandamme, 2002], [Díaz and Gervs, 2005] and [de Buenaga Rodríguez *et al.*, 2004]

A.5 TEST RESULTS

This section sums up the test results in an unordered list.

- Touch friendly interaction
- Tools for changing the layout, like changing the font size and colour scheme
- The front page should give an overview
- View whole menu all the time

- Give suggestions to similar articles, i.e. more on this story, subject and topic
- List overview of headlines in top of sections
- Search within relevant articles. Search bar should be visible at all times.
- Indication of similarity on articles
- Archive possibility
- User feedback notated with "relevant" and "irrelevant"
- White space besides articles is not a problem
- General layout corrections
- Better visual division between articles
- More and larger images
- No need for general news, personalised news is enough
- Choose categories with topics to get started
- Ability to choose period show articles from
- Ability to choose when the newspaper should be generated, e.g. on Fridays to be read in the weekend
- Dividing columns into screens
- Social community implementation
- Visualisation of user behaviour
- Possibility to use it for research
- Get articles from magazine and newspaper subscriptions, e.g. by adding them to a specific section
- It has solved the problems that http://nyhederne.tv2.dk/[1] has, i.e. confusing layout, no overview and hard to

[1] *The website of a Danish news channel.*

navigate

BIBLIOGRAPHY

[Abidi and Chong, 2004] Syed Sibte Raza Abidi and Yong Han Chong. Constraint satisfaction methods for information personalization. Technical report, Dalhousie University, Canada and Universiti Sains Malaysia, Malaysia, 2004.

[Abuzir and Vandamme, 2002] Yousef Abuzir and Fernand Vandamme. E-newspaper classification and distribution based on user profiles and thesaurus. Technical report, University of Ghent, 2002.

[Apt, 2006] Krzysztof R. Apt. *Principles of Constraint Programming*. Cambrige University Press, 2006.

[Billsus and Pazzani, 2000] Daniel Billsus and Michael J. Pazzani. User modeling for adaptive news access. *User modeling and user-adapted interaction*, pages 147–180, 2000.

[Bird *et al.*, 2009] Steven Bird, Ewan Klein, and Edward Loper. *Natural Language Processing with Python*. O'Reilly Media, Inc., 2009.

[Bouras and Tsogkas, 2010] Christos Bouras and Vassilis Tsogkas. W-kmeans: Clustering news articles using wordnet. *Knowledge-Based and Intelligent Information and Engineering Systems*, pages 379–388, 2010.

[Bush, 1945] Vannevar Bush. As we may think. *The Atlantic Monthly*, 1945.

[Centeno *et al.*, 1999] Vicente Luque Centeno, Mª Carmen Fernández Panadero, Carlos Delgado Kloos, Andrés Marín López, and Carlos García Rubio. Personalizing your electronic newspaper. Technical report, Universidad Carlos III de Madrid, 1999.

[Claypool *et al.*, 1999] Mark Claypool, Anuja Gokhale, Tim Miranda, Pavel Murnikov, Dmitry Netes, and Matthew Sartin.

Combining content-based and collaborate filters in an online newspaper. Technical report, Worchester Polytechnic Institute, Massachusetts, 1999.

[de Buenaga Rodríguez *et al.*, 2004] Manuel de Buenaga Rodríguez, Manuel J. Maña López, Alberto Díaz Esteban, and Pablo Gervás Gómez-Navarro. A user model based on content analysis for the intelligent personalization of a news service. Technical report, Halmstad University, 2004.

[Diaz *et al.*, 2001] Alberto Diaz, Pablo Gervas, Antonio Garcia, and Inmaculada Chacon. Sections, categories and keywords as interest specification tools for personalised news services. *Online Information Review, OIR*, 2001.

[Dub, 2012] Dublin Core Metadata Initiative. *Dublin Core Metadata Element Set, Version 1.1*, 2012.

[Díaz and Gervs, 2005] Alberto Díaz and Pablo Gervs. Personalisation in news delivery systems: Item summarization and multi-tier item selection using relevance feedback. Technical report, Universidad Complutense Madrid, 2005.

[Eirinaki and Vazirgiannis, 2003] Magdalini Eirinaki and Michalis Vazirgiannis. Web mining for web personalization. Technical report, Athens University of Economics and Business, 2003.

[Esteban *et al.*, 2000] Alberto Esteban, Pablo Gómez-Navarro, and Antonio Jiménez. Evaluating a user-model based personalisation architecture for digital news services. In *Research and Advanced Technology for Digital Libraries*, pages 259–268. Springer, 2000.

[Esteban *et al.*, 2001] Alberto Díaz Esteban, Manuel J. Maña López, Manuel de Buenaga Rodríguez, Jose María Gómez Hidalgo, and Pablo Gervás Gómez-Navarro. Using linear classifiers in the integration of user modeling and text content analysis in the personalization of a web-based spanish news service. Technical report, Universidad Europea de Madrid, Universidad de Vigo and Universidad Complutense de Madrid, Spain, 2001.

[Eurostat, 2012] Eurostat. http://epp.eurostat.ec.europa.eu/tgm/table.do?tab=table&init=1&language=en&pcode=tin00097&plugin=0, 2012.

[Genette, 1997] G. Genette. *Paratexts: Thresholds of interpretation*. Cambridge Univ Pr, 1997.

[Haskins, 1965] Jack B. Haskins. The editorial mix: One solution to a magazine editor's dilemma. Technical report, Journalism & Mass Communication Quarterly, 1965.

[Holmqvist *et al.*, 2003] K. Holmqvist, J. Holsanova, M. Barthelson, and D. Lundqvist. Reading or scanning? a study of newspaper and net paper reading. *Mind*, 2(3):4, 2003.

[Holsanova *et al.*, 2006] J. Holsanova, H. Rahm, and K. Holmqvist. Entry points and reading paths on newspaper spreads: comparing a semiotic analysis with eye-tracking measurements. *Visual communication*, pages 65–93, 2006.

[Ihlström *et al.*, 2004] Carina Ihlström, Maria Åkesson, and Stig Nordqvist. From print to web to e-paper - the challenge of designing the e-newspaper. Technical report, Halmstad University, 2004.

[Kress and others, 1999] G. Kress et al. *Reading images*. Routledge, 1999.

[Liu *et al.*, 2001] Fei Liu, Yupin Luo, and Masataka Yoshikawaf Dongcheng Hu. A new component based algorithm for newspaper layout analysis. Technical report, Tsinghua University, Beijing, 2001.

[Lloyd, 1994] JW Lloyd. Practical advantages of declarative programming. Technical report, University of Bristol, 1994.

[Mobasher, 2007] Bamshad Mobasher. Data mining for web personalization. In *The Adaptive Web*, Lecture Notes in Computer Science, pages 90–135. Springer Berlin / Heidelberg, 2007.

[Nielsen, 2012] Jakob Nielsen. How many test users in a usability study? http://www.useit.com/alertbox/number-of-test-users.html, 2012.

[Nilsson, 1999] Jørgen Fischer Nilsson. A conceptual space logic. Technical report, Department of Information Technology Technical University of Denmark, 1999.

[Ovesson and Wikström, 2005] Fredrik Ovesson and Kristin Wikström. From visions to specification - using user designed mock-ups for envisioning user requirements for the future e-newspaper. Technical report, Halmstad University, 2005.

[Perkowitz and Etzioni, 2000] Mike Perkowitz and Oren Etzioni. Adaptive web sites. *Commun. ACM*, pages 152–158, 2000.

[Readability API, 2012] Readability api. http://www.readability.com/developers/api, 2012.

[Russell and Norvig, 2003] Stuart Russell and Peter Norvig. *Artificial Intelligence: A Mordern Approah*. Prentice Hall, 2nd edition, 2003.

[Tidwell, 2010] J. Tidwell. *Designing Interfaces*. O'Reilly Media, Inc., second edition, 2010.

[Vossen, 2005] M.P.H. Vossen. Local search for automatic playlist generation. Master's thesis, Technische Universiteit Eindhoven, 2005.

[Åkesson *et al.*, 2005] Maria Åkesson, Carina Ihlström, and Jesper Svensson. How would you like your e-newspaper? – converging the best from two worlds. Technical report, Halmstad University, 2005.