NTNU
Norwegian University of Science and Technology

Newspaper on e-paper with WiFi transfer

The newspaper of the future

Håkon Rørvik Aune

Master of Science in Communication Technology

Submission date:	June 2008
Supervisor:	Yuming Jiang, ITEM
Co-supervisor:	Lene Maria Myhre, Trådløse Trondheim
	Marvin Bredal Lillehaug, Under Dusken

Norwegian University of Science and Technology
Department of Telematics

Problem Description

The task consists of assessing the technical adaptions that need to be done to distribute future newspapers to e-paper devices over wireless networks such as Wireless Trondheim's.

A demonstration of such a solution shall be made. The student newspaper Under Dusken supplies the newspaper content, and distribution to the e-paper devices shall be done over the network of Wireless Trondheim.

In cooperation with Under Dusken a user test-group shall be established. This group shall evaluate the different properties of use such as content, user friendliness and functionality. The results from the test-group shall be included in the final report.

This master thesis is based on work done in a fall project at ITEM, NTNU in 2007, and shall further assess the potential in wireless distribution of newspapers to e-paper devices.

Assignment given: 15. January 2008
Supervisor: Yuming Jiang, ITEM

Abstract

An e-newspaper is the result of newspaper content adapted to electronic paper. Electronic paper is a display technology which has many favourable attributes. It is a passive display technology which means that it can have the appearence of paper due to its high contrast, it is readable like paper due to only changing its pixels when the image changes and also uses very little power making it suitable for mobile uses.

Making an e-newspaper service has many challenges, the areas focused on in this thesis are namely layout and distribution. There is great differences between an online newspaper and a paper newspaper, and the question is which direction is suitable for an e-newspaper. Some of the factors that plays a role includes rates of updates, feel of quality, "charm" and of course personal preferences.

Distribution is split in two different approaches, pull or push. The latter most preferable because it gives the quickest updates in an automatic fashion. The reason for considering pull distribution is because it is more suited for devices that wishes to limit the time with an active air interface.

There was made a demonstration of such an e-newspaper service, with a standard template, online newspaper inspired, layout and pull distribution to accomodate the chosen e-paper device's poor battery life. The pull distribution was further user initiated to conserve even more power. The software made for this demonstration can be found in the accompanying archive file.

There was a test group, which evaluated the demonstration for a period of time. They did not completely agree with the layout choices, as they favoured a layout closer to the original paper source. That the download was user initiated was not a great problem though.

Preface

This is a Master's Thesis written by Håkon Rørvik Aune at the Institute of Telematics, NTNU. It was supervised by Lene Maria Myhre, Wireless Trondheim, and with faculty supervising by Professor Yuming Jiang.

I would like to thank my supervisor Lene Maria and professor Yuming for the help i received during the work with this thesis. I would further like to thank Marvin B. Lillehaug from Under Dusken for excellent assistance in getting content to the demonstration. Further i would like to thank Martine Moore, Morten Smedsrud, Nils Christian Roscher-Nielsen and Sveinung Sundfør Sivertsen for testing the demonstration made. I would also like to thank my office mates Erik Berg and Svein Tore Landsverk for an entertaining semester sharing writing this thesis.

A last thank to the people behind Ubuntu, LaTeX, Python and the open source community generally, these products certainly eased the work on this thesis.

Nothing is impossible.
-*Anonymous*

Contents

List of Figures

List of Tables

Part I

Introduction and theory

1 Introduction

Electronic paper is a display technology that is well suited for pleasent reading as well as portability. With the ever increasing coverage of wireless networks in urban areas, there are great opportunities for new services. The coupling of wireless capabilities with the electronic paper is one new platform that could host several new services. Newspapers delivered to such devices is the subject of this masters thesis, and is a continuation of work done in a Fall Project by the same student in 2007 at the institute of Telematics, NTNU.

1.1 Background

Electronic paper[1] is a technology that has been worked on since it was first conceived at the Xerox' Palo Alto Research Centre in the 1970's. It is first in the later years that this technique has been a viable replacement for paper in many situations. Compared to other popular display technologies the e-paper only uses power when the displayed image is changed, and does not rely on backlight. This gives it an appearence and reading experience that is close to ordinary ink on paper.

Wireless networks, in particular Wireless LANs, are gaining a more and more widespread use for every day. Wireless Trondheim is a good example of this, with its coverage of downtown Trondheim and plan for more. Many cities and towns are following suit, either for mainly research purposes such as Wireless Trondheim or for commercial reasons. Likewise, more and more devices gets some kind of wireless support. This increase in both wireless coverage and devices that can utilise it makes for great possibilities for new services.

[1]From this point, e-paper will be used instead of electronic paper for simplicity, and likewise for newspapers on e-paper – e-newspapers.

Due to the nature of the commercially available e-paper, a lot of the services that today flourish on the internet will not be able to do a transition to the e-paper format. The main reasons for this is the lack of colour and acceptable refresh/frame rates. Those services that base themselves mainly on the written word can on the other hand thrive on the e-paper medium. Newspapers are a prime example of something that could do well on e-paper.

1.2 Problems to be addressed

The problems that are to be adressed in this master thesis is threefold.

1.2.1 Solution for e-newspaper service

> *The task consists of assessing the technical adaptions that need to be done to distribute future newspapers to e-paper devices over wireless networks such as Wireless Trondheim's.*

This was understood as looking at different kinds of e-paper devices that are available and soon available, different kinds of content delivery to such devices and different kind of architectures that can be used for such an e-paper service. Lastly a design is sketched.

1.2.2 Demonstration

> *A demonstration of such a solution shall be made. The student newspaper Under Dusken supplies the newspaper content, and distribution to the e-paper devices shall be done over the network of Wireless Trondheim.*

The goal of the demonstration is explored, then a plan and design for the demonstration is devised. Then the process of the implementation is described.

1.2.3 Evaluation

> *In cooperation with Under Dusken a user test-group shall be established. This group shall evaluate the different properties of use such as content, user friendliness and functionality. The results from the test-group shall be included in the final report.*

The purpose of the test-group is explained and the properties they were asked to evaluate is described and reasoned for. The results from the evaluation is described and commented on.

1.3 Limitations of problem

In the case of an e-newspaper service, it was decided to limit the task to the e-newspaper itself and means of distribution. This means that topics such as security, economics and reliability are not touched upon neither in the discussion nor the demonstration.

1.4 Structure

This thesis is structured in three main parts, theory, implementation and evaluation.

The theory will introduce the e-paper technology, its properties and implications. Coming and existing devices that utilise this display technology follows, with special care about those equipped with wireless technologies. Different means of content delivery and distribution is then explored. Lastly different tools used in the thesis is briefly touched upon.

In the implementation part the possible implementation approaches will be surveyed and analysed. The chosen approach will be described and argumented for. The process of the implentation is also described

In the last part, the demonstration will be evaluated with help from a test group. There will also be focus on further work that can be done to bring e-newspapers closer to a realisation. Finally a conclusion will be presented.

In the following Appendixes there will among other be a description of the code that was used in the demo implementation, and which is also in the accompanying archive file.

2
Theory

In this chapter the necessary theory and background information will be presented. This amounts to a description of Wireless Trondheim and their network. A short introduction to e-paper and available devices that utilise this display technology. The content that will be used are described as well as tools and methods that can aid in transforming content to an e-newspaper, and distributing it.

2.1 Wireless Trondheim

Wireless Trondheim is a research and development project sponsored by local businesses and public institutions such as NTNU and the municipality of Trondheim. It's modus operandi is to provide Trondheim with wireless network coverage, and utilising this for research in areas such as services and networking. This wireless coverage also provides wireless access to Internet, which are available for the students and general populace – free for the former and a small fee for the rest.

In Table 2.1 some of the Key Features of Wireless Trondheim is listed, divided in the Access Network, Service Research & Development Lab and the Networking Lab.

2.2 Electronic paper

Electronic paper is a family of display technologies that has certain properties that makes them suitable for a number of applications, in this case reading on the go – mobility. Important characteristics include readability as print on paper, low power usage and portability.

E-paper displays are most commonly based on technologies that are bistable, which means that they stays in one state until current is applied. In other words the display will

Table 2.1: Key features Wireless Trondheim [3]

Access Network	Based on the Cisco WiFi platform Supports several ISPs (i.e. support for several SSIDs) Mobility is provided by layer2 mobility in the Cisco system High capacity - every access point has at least a 10 Mbit/s connection Location - the network can gather location information from its users Security - the network can be secured with the usual WPA/WPA2
Service Research & Development Lab	Business models for mobile telecom services Usage Patterns Service production and evaluation Telecom Protocols Radio interfaces
Networking Lab	"Open air" lab stations for real environments Each lab station features: "weather proof cabinet, power supply, high speed Ethernet switch, a host computer, a GPS receiver, mounting facilities for project specific instrumentation". Each lab station is connected to a dual ring of fiber for redundancy IPv6 compliance

only redraw when the image changes. The popular LCD technology is on the conterary monostable, which implies that they has to be refreshed to stay in a given state, 60 times a second (i.e. 60 Hz) is a usual refresh rate.[65]

2.2.1 Historical background

On a history note, this kind of technology was first conceived in 1974 at Xerox PARC (Palo Alto Research Center) and has been worked on by several parties since. At that time the application was intended for use in displays for another of Xerox' inventions, the Alto personal computer.

Due to Xerox' "obsession" with copier machines, the initial research was shelved and not worked with until 1989. At the end of the 1980's most people realised that the paperless society was not going to be a reality in a foreseeable future. The original inventor observed that people had obtained the habit of printing and discarding content instead of reading it on their computer screen. He then realised that Xerox' e-paper technology could be used to alleviate some of the paper being comsumed. The computer screen of that age had a screen refresh as much as current computer displays, and was as straining on the eyes.[23]

2.2.2 Different kinds of electronic paper

There are several different approaches that can be chosen to achieve displays with e-paper characteristics. Two of these originated at PARC, the Gyricon bead technology and electrocapillarity, the latter is in recent research known as electrowetting. Other

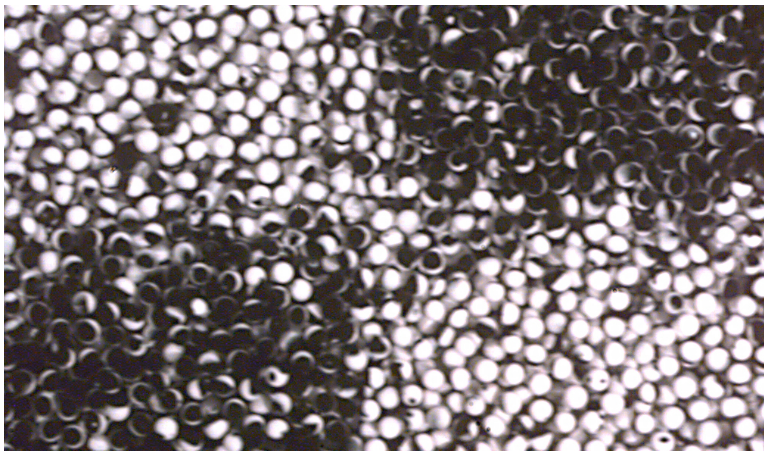


Figure 2.1: A close-up image of a Gyricon sheet. Notice that each bead has one black and white hemisphere.[9]

notably techniques are electrophoretics and cholesteric liquid crystals.

Gyricon

The Xerox PARC e-paper technology earlier mentioned got the name Gyricon. A Gyricon e-paper consists of millions of small beads contained in an oilfilled cavity between two transparent films. The beads are coloured with opposite colours on each hemisphere, and charged in such a fashion that when voltage is applied to the sheet the beads will rotate and show either hemisphere. With the right voltage patterns, image and text can be shown. This can be seen in Figure 2.1.[9]

Electrophoretics and electronic ink

Electrophoresis is a principle well suited for making e-paper displays. Electrophoresis happens when particles in a fluid moves because they are under influence of an electric field.[66]

Most of the e-paper devices that are commercially available uses electrophoretic displays. The E Ink company is currently the biggest manufactorer of such electrophoretic displays with their electronic ink.

This electronic ink consists of millions of small capsules, roughly the size of a human hair, which represent each pixel. These capsules are filled with fluid and white and black particles. The particles are oppositely charged, white particles got a positive charge while the black got a negative charge. Applying an electric field to a sheet of e-ink will bring particles with the opposite charge of the field to the top. Manipulating the electric field, would make it possible to create text and images. An illustration of this can be seen in Figure 2.2.[10]

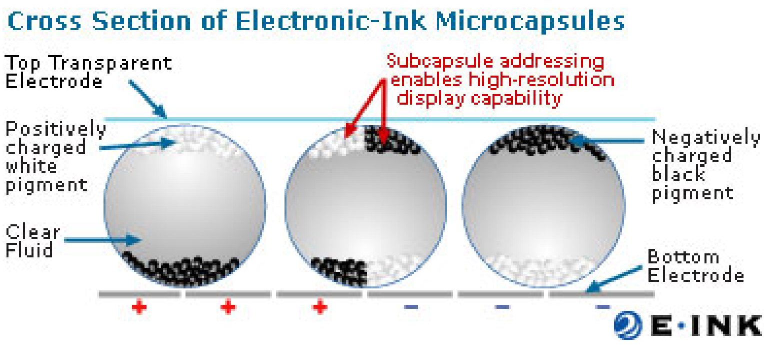

Cross Section of Electronic-Ink Microcapsules
Top Transparent Electrode
Subcapsule addressing enables high-resolution display capability
Positively charged white pigment
Clear Fluid
Negatively charged black pigment
Bottom Electrode
+
+
+
−
−
−
E·INK

Figure 2.2: An illustration of the microcapsules in electronic ink under the influence of different charges. [10]

Electrowetting

Using electrowetting as a technology was first considered at Xerox PARC along with Gyricon, the latter was easiest to implement at the time so electrowetting was put aside.[23]

Electrowetting is a technology that can possibly give e-paper with both colour and the speed required for video. It is based on the fact that different voltages gives different wettability for certain materials. The wettability of something tells how water-repellent something is.

To make a pixel in an e-paper display, a stack are made of water, coloured oil , and an electrode coated with a hydrophobic (water-repellent) material. When there is no voltage the oil covers the hydrophobic material and the pixel has the colour of the oil. When a voltage is applied the wattability is changed and the water moves the oil away from the hydrophobic material. With a white material beneath this stack and oil dyed black, grayscale can be achieved. See Figure 2.3.[26, 17]

Bistable Liquid Crystals

This is a technology that uses the same nematic liquid crystals as in ordinary LCD (Liquid Crystal Display) monitors. The liquid crystals used in LCD monitors are only monostable, and will return to a given state if it is not supplied with a voltage. Bistable liquid crystals will stay in one state until it is changed to its other state. When it has reached the other state it will be stable until the image changes.

Weak anchoring The e-paper company Nemoptic has devised an e-paper technology called BiNem, using weak anchoring and nematic liquid crystals. The nematic liquid crystals are placed between two layers with different anchoring properties. In ordinary LCD displays both have layers have strong anchoring properties, but in the BiNem technique one has weak anchoring properties. Applying a voltage will make the crystals break the anchoring, and using a controlled voltage will make the crystals settle in one of two possible states. Add a polariser at the top layer, and it is possible to show two

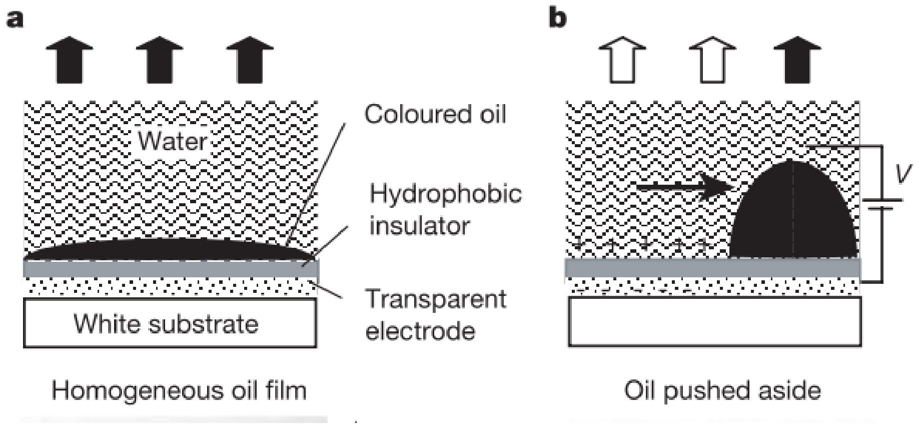


Figure 2.3: An illustration of the different stages of a pixel in an electrowetting setup. The left will show a pixel with the colour of the oil, while on the right a pixel with the colour of the underlaying substrate will show.[26]

different colours. See Figure 2.4 for an illustration of this. [4]

Cholesteric Liquid Crystals Liquid crystals in a cholesteric phase acts in a special way; the molecules in different layers orients at a slightly different angle relative to each other. It can be seen as a stack of many thin layers that rotate with respect to layers over and under. If no electric field the stack will have a selective reflection of light and appear bright. If an electric field is applied the stack will be "scattered" and the colour of the bottom layer will show.[6]

2.2.3 Coloured e-paper

To achieve colours, colour filters are one of the usual techniques. This requires the use of three sub pixels for each pixel to show the three base colours Red Green and Blue (RGB). Such use of subpixelation will both lower the resolution per area as well as lower the percieved brightness.

There are currently none e-paper devices readily available with colour, but the Fujitsu FLEPia is a device with colours that might release within a year. It is currently being tested at various companies in Japan and is further described in the coming sections.

2.2.4 Flexible e-paper

Flexible e-paper can make the e-paper technology more sought after, as it will increase the mobility by much. There is much easier to carry an e-paper device that can be rolled or folded to a small non-use size, compared to the more "bulky" e-paper devices of today.

Current driving techniques used for e-paper are the same as for ordinary computer displays (LCD). These are made with glass, as the production temperatures for silicon

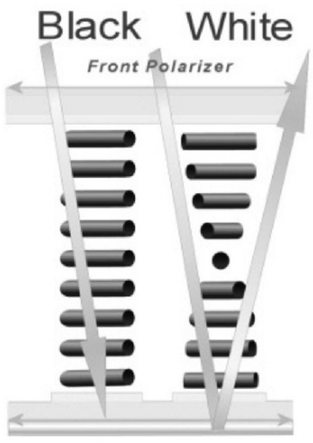


Figure 2.4: An illustration of how a BiNem pixel looks. On the left, the liquid crystals are in a uniform state, and not letting reflecting any light. On the right, the liquid crystals are twisted, and will reflect a white light.[4]

based semiconductors are fairly high. Organic electronics is a field of research for making semiconductors that requires lower production temperatures, enabling the use of flexible plastics instead of glass.

Plastic Logic [44] is currently preparing for large scale manufactoring of such organic electronics. There are already one device using this kind of displays, the Polymer Vision Readius, as will be presented in 2.3.2. Their techniques are further described in [30, 50].

2.3 E-paper devices

There are a number of different e-paper devices on the market. The differences in technology and features are on the other hand not that numerous. With one exception all available e-paper devices are of the "large PDA" design. This can be seen in <Fig 1 -X>. The exception is the Readius by Polymer Vision, which can be seen in <Fig X>.

In this section a presentation of current and soon to be launched e-paper readers will be made. This is a fairly young market with a lot of new contenders along with more established companies such as Sony. In general it is two kinds of operations, those who design and build their own devices, and those who buys in bulk from OEM (Original Equipment Manufactor) factories and rebrand the units and add their own software.

The e-paper displays are mostly supplied from one source it seems, E Ink. E Ink is the market leader on displays made with electrophoretic (electronic ink), and are used by most of the manufactorers of e-paper devices.

2.3.1 The available devices

Currently there are a number of devices on the market, here the most prominent of them are presented. An overview can be found in Table 2.2.

Table 2.2: An overview over available e-paper devices

Device	iLiad	Kindle	Cybook	Reader
Manufactor	iRex Tech.	Amazon	Bookeen	Sony
Display	8.1"	6"	6"	6"
Resolution	768 x 1024	600 x 800	600 x 800	600 x 800
Size (WxHxD mm)	155 x 216 x 16	134.5 x 190 x 19	188 x 118 x 8.5	175.6 x 123.6 x 13.8
Grayscale	16	4	4	8
OS	Linux(2.4)	Linux(2.6.10)	Linux(2.4.18)	Montevista Linux
Formats	PDF, XHTML,TXT, MOBI[a]	AZW[b], MOBI, TXT	MOBI, PALM[c], TXT, PDF, HTML	PDF, TXT, RTF, DOC, BBeB[d]
Connectivity	IEEE802.11b/g, Ethernet, USB	EVDO, USB	USB	USB
Expendability	USB[e], MMC, CF	SD	SD	MemoryStick Duo, SD
CPU	400MHz	?[f]	200MHz	200MHz
Storage[g]	128MB	180MB	64MB	200MB
Price	649EUR	399USD	?	?
Sources	[36, 69]	[63]	[64]	[70]

[a] Mobipocket format is an ebook format based on XHTML with DRM(Digital Rights Management) protection. [b] The Mobipocket format with a slightly different DRM scheme. [c] PalmDoc format [d] BroadBand eBook, Sonys proprietary format used in their Connect eBook store. [e] USB memorystick [f] Intel XScale CPU of either 200, 300 or 400 MHz. [g] Available onboard after the system has taken what it needs.

Amazon Kindle

The online bookstore Amazon is the manufactorer of the Kindle e-paper device. It got a 6 inch 800 x 600 e-paper display as most other similar devices. This E Ink powered display can show 4 shades of gray.

As the Kindle is manufactured by Amazon there is naturally a large selection of books and resources available for it. This content is delivererd in Amazon's own proprietary format from the Kindlestore[1] via Whispernet, a 3G equivalent EVDO[2] network in the US. The Kindlestore can be browsed from the device, and there is only need for a computer

[1]`http://www.amazon.com/gp/browse.html?node=133141011`

[2]EVDO (Evolution-Data Optimized), is a 3G standardization by the 3GPP2 group, the Asian and North American counterpart of 3GPP (3rd Generation Partnership Project), which currently works on the CDMA (Code division multiple access) family of mobile communications. [67]

to put content acquired from other sources onto the device. The Kindle is only available in the US, and if it will be available elsewhere is unknown at this time. [2, 39]

Bookeen Cybook

The Cybook series of e-paper readers are sold by the French company Bookeen. Their latest contribution, the Cybook Gen3 is one of the more lightweight e-paper units, 174 grams, while still keeping the fairly standard 6 inch display. This 6 inch screen is of 800 x 600 resolution and 4 levels of gray.

This model is not supplied with WiFi, but the OEM producer Netronix (see 2.3.2) which seems to be the producer of the Cybook, got a similar reader with WiFi in their catalogue.

Sony Reader

Sony was the first manufactor that launched an e-paper device with an actual e-paper display. Earlier products in this segment used LCDs and such for the display, and was marketed as e-book devices.

Their first device was the LIBRIe in 2004 primarily for the Japanese market, and in 2006 they introduced the Reader in the US. The Reader uses the same display as the former LIBRIe, a 6 inch e-paper display made with E Ink technology. The resolution is 800 x 600, and while the first edition only came with 4 levels of gray, the second edition is sporting 8 levels of gray with a new E Ink version.

It is not supplied with WiFi, and whether it is planned included in a new version is not known at the point of this writing. But it seems to be the trend for some of the upcoming (next generation) readers that are soon to be launched, so it would not be a surprise if one found its way to the market one day. Usually WiFi capability is only a chip and some code adaptions away.

iRex iLiad

The iRex iLiad is the only commercially available e-paper device on the market with support for WiFi. It is also equipped with the largest e-paper display of them all, a 8.1 inch display while the rest of the devices usually got one in the 6 inch range. It can also display the currently highest levels of gray, 16, while the usual is 4 or 8.

The iLiad is also equipped with a Wacom tablet[3], which enables the user to use a stylus to navigate, annotate and use the iLiad as a digital notebook.

See 3, where more information is provided, as this is the device that will be used for demonstration purposes.

[3]Wacom is the largest producer of tablets for graphical use. It is based on electromagnetic resonance, and a tablet consists of a grid of wires that makes a magnetic field that can be used to detect certain characteristics of the pen used (position, pressure, speed etc.)[60]

2.3.2 Soon to be released devices

A lot of the upcoming e-paper devices are equipped with the all important WiFi connectivity, here the most interesting devices of the imminent future are presented. A quick overview can be found in Table 2.3.

Table 2.3: An overview over soon to be available e-paper devices

Device	Hanlin V9c	EB-300	FLEPia	Readius
Manufactor	Jinke	Netronix	Fujitsu	Polymer Vision
Display	10"	9.7"	12"/8"	5"
Resolution	1200 x 825	1200 x 825	1024 x 768	?
Size (WxHxD mm)	255.8 x 173.5 x 14.3	255 x 195 x 14	210 x 304 x 12(A4) 158 x 240 x 12(A5)	115 x 57 x 21
Grayscale	?	4	8 or 4096 colours	16
OS	Wolf Linux	WinCE5.0	WinCE5.0	WinCE
Formats	PDF, DOC, TXT, HTML	PDF, RTF, TXT, HTML	?	?
Connectivity	IEEE802.11b/g, GPRS, CDMA, USB	IEEE802.11b/g, BlueTooth, USB	IEEE802.11b/g, USB	GSM/GPRS/ EDGE, UMTS/HSDPA, USB, BlueTooth
Expendability	SD	SD	SD	Micro SD
CPU	200MHz	?	?[f]	400MHz
Storage[a]	64MB	?	?	256MB
Price	?	?	?	?
Sources	[55, 54]	[31]	[40]	[57]

[a] Available onboard after the system has taken what it needs. [b] Intel XScale CPU of either 200, 300 or 400 MHz.

Netronix EB-300

Netronix seems to be an OEM for some other companies, notably the Bookeen Cybook Gen3. The strongest indication on this is that the EB-100, EB-200 and EB-210 of Netronix looks exactly like the Cybook Gen3. This is also supported by several discussions on enthusiast forums such as `MobileRead.com`[4].

The most interesting of Netronix' e-paper devices is the EB-300, a device with a 9.7 inch E Ink display, 1200x825 pixels and 4 levels of gray. Most importantly it is going to be equipped with 802.11g WiFi. Other features includes BlueTooth and a touch screen. Whether the touch screen needs a stylus like the Wacom enabled iRex iLiad or not is not known yet. They do also have some lesser sized devices that can be equipped with WiFi capability.[31]

Jinke Hanlin eReader v9c

Jinke is a Chinese manufactor of e-paper devices, and got 3 available readers with 6 inch displays and the usual features. The company do also have more coming up the pipeline, one of them more interesting as it got WiFi.

The v9 series of their Hanlin eReader is equipped with WiFi in the v9c model. In addition to WiFi, the v9c is also going to be equipped with CDMA and GPRS. The display is a 10 inch E Ink display[5] with a resolution of 1200x825 and 4 levels of grayscale. It runs on a linux distribution which seems to be developed by themselves, Wolf Linux. In addition to the usual PDF, DOC, HTML and TXT they also have their own DRM protected format WOLF.[55, 54, 53]

Polymer Vision Readius

This e-paper device is not going to be released with WiFi capability, but is interesting nonetheless. It is the first commerciable available device that is going to sport a flexible e-paper display. This e-paper display is 5 inches and capable of 16 shades of gray. As mentioned no WiFi, only mobile phone technology; GSM/GPRS/EDGE and UMTS/HSDPA aswell as BlueTooth.

Fujitsu FLEPIa

This device is currently the largest and also the only device made with a colour display. Currently it has only been sold as test units to different corporations for development and testing, and according to their announcement they are supposed to start selling it globally in larger volumes in the fiscal year of 2008.

The unit will run on version 5.0 of Windows CE, and will be equipped with IEEE802.11b/g WiFi. In addition to the usual buttons this device can be operated via the touch enabled display.

It comes with two different display sizes, 8 and 12 inches. Further it can also be delivered with two different colour depths, 8 and 4096 colours. The announcement is unclear on it, but this device probably uses colour filters to achieve colours and as both display sizes is delivered with a resolution of 1024 x 768, the largest display size is

[4]Pointer to an example discussion!!! be here!!

[5]most likely the same as the Netronix EB-300 in 2.3.2 as they got the same resolution and levels of grayscale.)

probably the only one with the greatest colour depth. It is also worth to mention that a display with 4096 colours needs 10 seconds to redraw while the 8 colour one can manage the same operation in 2.3 seconds. [40]

2.4 Content

"Content is king" is a common wording used in relation with most content providing services. There is usually a greater chance of success for a service if there is plentiful of content for users to consume. With less available content the users will most likely look for something else that can hold their attention longer.

This is also true for e-paper devices and services for it, available content is needed for it to succeed. Most of the e-paper devices these days are mostly used for reading books and other content the users has on their hands. This can be reports, operating manuals and technical articles.

In most cases the user would also want to adapt content for the reader if possible, as there is still no readers that are 1:1 with plain A4 paper. Most e-paper readers got display sizes comparable to something between A5 and A6, which would either require extensive scrolling or a zoom which renders the print tiny. But with the current technology an e-paper device with a display that could handle A4 in a satisfactory fashion would become of some size. Most of the current e-paper displays are made with rigid backplanes and thus cannot be collapsed to a size that is easier transportable. The curcuitry and battery also requires some space, making it all in all comparable to a large PDA.

E-paper devices of the future will most likely use displays made of flexible materials, and thus can be rolled or folded to a more convenient size. In [23] one of the pioneers of e-paper, Nick Sheridon, describes his vision of the future e-paper:

> I like to tell people that the holy grail of e-paper will be embodied as a cylindrical tube, about 1 centimeter in diameter and 15 to 20 centimeters long, that a person can comfortably carry in his or her pocket. The tube will contain a tightly rolled sheet of e-paper that can be spooled out of a split in the tube as a flat sheet, for reading, and stored again at the touch of a button. Information will be downloaded – there will be simply user interface – from an overhead satellite, a cell phone network, or an internal memory chip. This document reader will be used for e-mail, the Internet, books downloaded from a global digital library that is currently under construction, technical manuals, newspaper (perhaps in larger format), magazines, and so forth, anywhere on the planet. It will cost less than $100, and nearly everyone will have one!

2.4.1 Under Dusken

Content for demonstration purposes are provided by Under Dusken, the largest student newspaper in Trondheim. It has an ordinary paper edition as well as an online edition. They share most of the content with some exceptions such as notices, videos and such.

The paper edition is published every other week, which amounts to 8 editions each semester. The content is roughly divided in news, reports, commentaries, reviews and culture articles. Some of these categories also got some subcategories such as commentaries that can be divided in chronicles, editorials, commentaries and reader's letters.[15]

The online edition got an RSS (Really Simple Syndication) feed with some of the categories mentioned above. For the purpose of this demonstration a special feed was made with the ten newest articles in each category and subcategory. This amounted to about 90 articles. Differing from usual web feeds this feed got the entire article text in the feed instead of a link to the website. It also contained more information about the authors and illustrations. How this was utilised will be further explored later on in 7.1.1.

2.5 Content Delivery

When a service has its content provided for, considerations for the content delivery needs to be done. In the fall project [7], it was noted that there are two main approaches for content delivery, push or pull. With pull distribution of content the recipient initiates the delivery herself, pulls it to her. On push distribution the content delivery is started from the content source. This can either be by sending a notification to the recipient telling that there is content to be dowloaded at a given location, or sending all of it to the recipient in the first place. There are advantages and disadvantages with both of these approaches.

The content delivery phase can also be optimized by some further means. Caching is one of these means, while transcoding and adaption is another. In the following subsections the mentioned topics will be further explored. In the Discussion chapter these topics will be related to the task at hand.

It is assumed that a user, either can subscribe to one or more newspapers, or parts thereof. The alternative is that a user is not regulary subscribing but fetches one or more newspapers from a selection of many, on a whim or such. The last alternative is a combination of these, regular subscription on something and irregular fetching of something else.

The content it is assumed is of a kind that is prepared and processed in some kind. Most digital newspaper content these days is not required to be adapted in any way, but as the e-paper devices so far are inferior in many ways it is many benefits in alleviating the device of most processing required. An e-newspaper service is in this thesis envisaged as a centre point in the flow of newspaper content. Content is published to or acquired by the service, which processes and adapts it for the e-paper medium. An interesting comparison can be found in the web browser company Opera and their Mini web browser[5], which uses a proxy to adapt and preprocess content before sending it to the device.

2.5.1 Wireless Considerations

Compared to fixed lines and non-mobile hardware, mobile datacommunications and mobile devices are at a disadvantage. Mobile devices are generally less powerful than their stationary counterparts, and they rely on limited stored power. To get the most out of these limits on a mobile device, the device should not be counted on being able to do many hardware intensive operations.

Compared to fixed lines, a wireless link is capable of less throughput complemented with higher probabilities of both packet loss and disconnects. These are important factors that should be taken account for when planning content delivery.

2.5.2 Pull

With pull distribution, the sole responisbility of downloading the content lays on the user, or the users device. This means that all content is downloaded either when the user tells the device to download an e-newspaper, or further refined by having the device download at given times.

The latter technique of downloading at given times can be improved to work as precaching, providing a subscriber with “instant” content when they need it. This was further explored in the fall project [7]. This included looking on the problems with content delivery routing and mobility [37], precaching based on spatial knowledge [43], calendard based precaching [38] and caching based on broadcasting and a cache split between the network and the device [72].

2.5.3 Push

With push distribution, the distribution is initiated by the source. This can either be the source pushing the entire e-newspaper to the receiver, or sends a notificaion about e-newspapers that are ready to be downloaded.

Push distributing is most suited for publish subscribe kinds of distribution schemes, and was as with pull distribution explored in the fall project [7]. Topics looked at then included looking at existing publish subscribe systems, problems with publish subscribe in mobile environments and examples of systems for this such as [45] and [29]. Some of the more interesting aspects mentioned here about push and pull distribution will be explored a bit further down in 2.5.5.

2.5.4 Content Adaption and Transcoding

One way of optimising the transfer of content to mobile devices is to lessen the amount of content that needs to be transfered. Huge colour images is a waste to transfer to contemporary e-paper devices – small black and white displays has obvious limitations. If the device is actually capable of showing the image in some sort, it has still wasted unecessary storage capacity, processor cycles and bandwidth – all commodities that are scarce for a mobile device.

While the process of transcoding can happen both at the device and the network, it is the latter which is of interest. But where this is placed in the network do also have alternatives, namely at the content source or closer to the receiver. Having this at each content source would seem a bit overly redundant but would also distribute the load, as having it at fewer locations could mean bottlenecks at peak time.

more

2.5.5 Content Delivery Networking

Serving an area such as that of Wireless Trondheim can easily be done from one source, but if such a service catches on and are introduced to larger and more areas a single source is less optimal.

Content delivery networking encompasses different means to serve content effectively to larger and/or more widespread audiences. These techniques are usually supposed to be used for content of higher volume and more timely requirements such as video and other multimedia. But some of the ideas are interesting in an e-newspaper delivery perspective, and as the e-paper devices most likely evolve to support more kinds of content, these ideas can be more relevant. Thus a brief review should not be out of its place.

Subscription

For an e-newspaper service, the use of some kind of subscriptions is to an advantage for both the service provider and the user of the service. A subscription lets the provider offer the possibilities to tailor an e-newspaper to the wishes of the subscriber. This can either be in the form of the subscriber telling the service that she wants this and that from those newspapers, alternatively just subscribing to a preselected choice of content, or it could also be in the form of the subscriber saying that she wants newspaper content with such and such characteristics. This is called topic-based, channel-based and content-based subscription. [29]

The provider can then either push style send the subscibed content or notification to the subscriber, or the provider could let the subscriber set times she wants it either pushed to her or ready to be pulled down to the device.

In the litterature there are great interest in these publish/subscribe systems. Such publish/subscribe systems are asynchronous of nature, and uncouples the initial publisher and the receivers. This is as noted in [29] advantageous of many reasons; assuring of receival is not the publishers responsibility after it has arrived at the e-newspaper service. As well as giving an impression of multicasting this is advantagous in a mobile environment as it lets the publisher focus on publishing, and lets the e-newspaper service worry about getting the content to its receiver. Mobility adds some further problems when it comes to delivery, as will be touched upon in 2.5.5.

Caching

Caching can happen either at the last stages in the network before the customer or at the customers device. There are advantages and disadvantages with both of these as will be touched upon below.

In the network Caching in the network involves putting content a customer might want closer to the device, making the accesstimes shorter than to the source. This could be simple mirroring of content as many download services provide. A download service could have servers in Europe, eastern and western US and Asia serving the same files, giving users in each area optimal downloads.

In many cases there are really no need to mirror all content on all servers. This could lead to a waste of resources for content never used. Optimally there are only mirroring of content that subscribers in a service area wishes. [52] describes a scheme wich involves pushing content to those who requests it and to those whose profile indicates that they probably would request it later on. This profile is based on content downloaded by one user and which other users in the same area also acquired.

At the device This leads to the other kind of caching, at the device. This implies that a device will precache content the user will or might need in the future. If the user keeps a subscription it is an easy task for the device to download the content the user subscribes to, either push or pull fashion, to be ready when the user wants it. If the user do not have any subscriptions, but more of a random habit concerning the reading material, the task is more trickier to perform correct. A scheme as the one mentioned earlier by [52], could work to a degree but it needs to differ between content downloaded by subscriptions or randomly as this could skew the statistics. An interesting approach is proposed by [38], which uses a users daily schedual and interests as contextual information to cache content the user might find interesting. If there is no severe limitations on battery nor storage, it could be an option to cache whatever happens to be downloaded in the reception area of the device. If the content is news, it is likely that some of it might be of interest to the user of the device.

Difficulties regarding mobility

Following here are a few notes on the difficulties that mobility adds to the equation. To actually utilise some of the solutions described above, one has to take into account mobility. If a service relies on being able to push content to its subscribers they would suffer from the same problems as routing in mobile environments suffer from; reaching the recepient.

The challenge for pulling content to a device, either for precaching or user requests, is to find the closest cache or source. The device either needs to find and maintain this information itself, or let logic in the network guide it to the closest content source. The latter could be routers keeping this information in its routing tables.

For pushing content there is the trouble of knowing where the recipient is, and being able to send content to a device that is mobile and cabable of roaming; the same problems that solutions such as Mobile IP has to overcome. While Mobile IP might be more focused towards Voice over IP, where content dropped and content after a disconnect is of little consequence, content dropped for e-newspapers should be retransmitted to the device if possible and on disconnects content should be kept and queued until the device can receive it again.

Further, mobile users can make it hard to intelligently fill caches in the network with content that most likely will be required, as statistics are skewed by people “passing by“. The impact of users travelling through a cache’s service area can to some degree be dependent on the size of said area. A large area will have a higher density of regular users than a smaller – people mostly tend to move around in the same area, between home and work and home and shops. If these areas is all in the service area of one cache, there should be easier to predict content to cache.

2.6 Python

Python was chosen for doing most of the programming on the demonstration. The Python programming language is good at string operations and the kind of scripting needed for the demo, and there was also a wish from the author to learn more about this scripting language. The author was at the start of this thesis in the progress of learning more about Django[13], a Python based web framework similar to the more known Ruby on Rails framework. Django has some features which at that point seemed useful for the demonstration to be made, and thus it seemed sensible to continue with Python. A bit more information about Python and the Python tools used can be found in Appendix A

Part II

Implementation

3
iRex iLiad as the chosen device

The iLiad e-paper reader by iRex Technologies is currently the only reader available with WiFi, thus making the choice for device easy. It is also equipped with other technologies and features which makes it to one of the most advanced currently available. [35]

3.1 Operating System

The iLiad runs on Linux, kernel 2.4.19 to be precise, which makes it possible to port software made for other versions of Linux to the iLiad.

In the patch that brought the devices to version 2.9.5 iRex made the software on the device GPL compliant, in other words the iLiad got open source. Prior to this, with version 2.7.1 a developer package was introduced that would give root access on the device, and enabling users to execute and test third party programs and shell scripts for the iLiad. Acompanying the developer package, iRex also released a toolchain that would ease any attempts of porting and creation of software for the iLiad.

3.1.1 User Interface

The user interface consists of a number of buttons on the physical device, and in software the user interface mainly consists of a content lister which are used to browse files and start programs. See Figure 3.1 for an image of the iLiad with its various interface elements explained.

Physical

Starting in the top left corner of the iLiad as seen in Figure 3.1. Top button is for going one level up, next is for going to the topmost level, the device manager as also can be seen

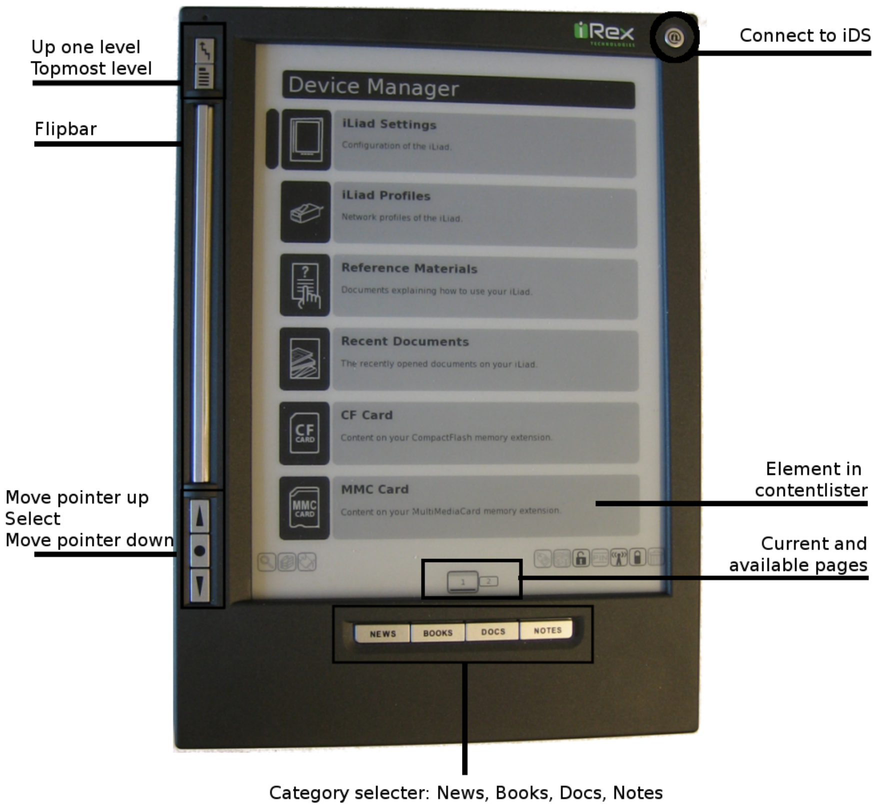


Figure 3.1: Image of the iRex iLiad with buttons explained. The image is taken in natural light, and as can be seen the display is a bit gray, but still fairly paper like.

in the image. Next is the long flipbar for going one page forward or backward (flipped to the left as a book page, it will go one page forward). At the bottom on the left side is the three navigational buttons for each view of the content manager. The pointer these buttons control is the black bar beside the iLiad settings seen in the image.

On the bottom there are four buttons for taking the user to one of the four main categories: news, books, docs and notes. When pressed the iLiad will open the content lister for that category.

In the top right corner is the button that on default will connect the iLiad to the iRex Delivery Service (iDS) for downloading content or updates.

There is also the possibility to use the accompanying stylus to select and manoeuver through the menus.

Software

The main element of the software user interface is the content lister. It does as the name tells, lists content. This content can either be content files such as books and newspaper or software that does things. To control how the content lister shows the entries, the system utilises XML files called manifest files. They control what happens when the entry is selected, but also what the entry shows, such as the icon, the headline and the description.

The content lister can be seen on the screen of the iLiad in Figure 3.1. There the topmost level of the iLiad is shown, the Device Manager. Here the settings reside as well as links to the different storage on the iLiad. For e.g. the newspaper section, the content lister acts in the same way, but with other contents. The settings and network software that comes with the iLiad are built as several pages; the user flips through the pages and makes changes and adds information as they see fit.

3.2 E-paper display

In September 2007 iRex released a second version, the iLiad v2. Apart from some minor design changes and a 2.11 software update, which also was available for the first version, the second version got an improved e-paper display. This improved display is made from E Ink's Vizplex$^{\text{TM}}$Imaging Film. Compared to the previous generations of E Ink displays, the rewrite speed is lower and the brightness i higher.

For testing purposes four of the iLiad version 2 was bought, and compared to the old version 1 the greatest difference is in the brightness. The new model is significantly brighter than the older one, but it is still some way to go before it is as bright as plain white paper as the picture in Figure 3.1 shows.

The typical redraw time on the new display is about 720 ms, and the peak is on 260 ms. This is about 40-50% lower than the previous generation. This is still fairly slow compared to other display technologies, and will have effect on what they can be used for. For the iLiad this means that consideration needs to be done when choosing software to develop or port. Software that easily can be divided in pages are most suitable until the e-paper gets less static.[12]

3.3 Performance

The iLiad is equipped with an Intel XScale 400 MHz processor and 64 MB of RAM.[36]

Following here is the authors impressions of the device, and its performance.

3.3.1 In use

When the iLiad is switched on, it needs to boot. This takes a while, something in the area of 30-40 seconds which might be a bit long. This would not be a big issue had it not been for the poor battery life of the iLiad, with use of wireless and the stylus the battery might last a day. It might thus be advisable to turn it off between bouts to preserve the battery, but a long boot up time can be detrimental to more spontaneous use of the device.

When it is first up and running the performs is ok, but it is still not lightning fast compared to other devices that are equipped with the same kind of XScale/ARM processor. Such devices amounts to smart phones and PDAs, which feels significantly more responsive than the iLiad. The limiting element is the redraw time on the display which is as noted earlier 720 ms. With maximum close to a second between each different view, the device can seem slow and unresponsive on some operations.

When it comes to opening books and e-newspapers it do use some time, but comparing it with smart phones might be a bit unfair as they rarely have to battle that kind of files and filesizes. When the content is opened it is fairly quick to navigate between pages, probably a bit faster than the speed one flips pages when reading normally.

3.4 iRex Delivery Service

The iRex Deliver Service (iDS) is a internet based content delivery system made for the iLiad. When a user has registered her iLiad she is in the possession of an iDS mailbox. Content sent to this mailbox can be downloaded to the iLiad at a later moment. This operation can be achieved by pressing the iLiad's iDS button, which will connect it to the iDS server to check for any content in the mailbox. [34]

4
Implementation goal

One of the goals of this thesis is to make a demonstration of a newspaper for an e-paper device, the iRex iLiad in this case. Making such a demo can be achieved with a few different approaches and design choices as will be explored in the subsequent sections.

4.1 A(n e-)newspaper

A usual newspaper these days, are either a traditional print on paper newspaper, or an online version residing on a website. When making a newspaper for the e-paper medium, choices needs to be taken for what to include.

A newspaper for an e-paper device can contain many elements, some necessary, others fairly optional. A newspaper does primarily need news, usually in the form of text, augmented with images and illustrations. Online versions of newspapers have brought this further by supplementing articles with videos, animations and interactive tests.

Articles in a print on paper newspaper is further divided into sections such as news, local, national and foreign. There are also sections for categories such as sports, economics, culture and many more.

Online newspapers can be enhanced with further features due to their excistense in the digital world. When reading an article in an online version of a newspaper the subscriber can easily follow hyperlinks to either background information or previous articles of that topic. Many newspapers also have enabled commenting of articles and other ways for the users to contribute to and discuss articles and content.

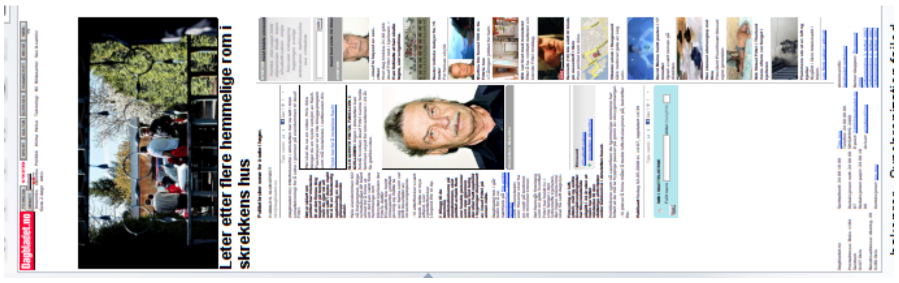


Figure 4.1: A screenshot of one article on `dagbladet.no`, it is flipped to the horizontal for space reasons.

4.1.1 Layout

Bringing most of the different features and elements of the mentioned online and print on paper newspapers is possible. The question is which layout direction to follow: online or print on paper style layout.

The most common sizes for e-paper displays are 6 and 8 inches, as noted in 2.2, with some devices with 10 inch displays coming in the near future[1]. A 10 inch e-paper display would roughly be of the same size as an A5 sheet of paper. An ordinary print on paper newspaper comes in several sizes from the large broadsheet to the small tabloid, a sheet of each kind is though much larger than A5. If the wish is to view a newspaper in its original form, zooming and panning would be needed to make the text on each page legible.

Online versions of newspapers are not bound to the sheetsize of the print on paper version. Usually an online newspaper is built with a main index page with headers and links to each of the articles. Each article is made of a single column with images and illustrations interspersed with the text, as can be seen in Figure 4.1.

One of the main uses of current e-paper devices is for reading books, and these devices are often dubbed as being an e-book reader. Such e-books are delivered in a pageable format which lets a user flip through the book as she would with an ordinary print on paper book.

4.2 Approach for demonstration

The greatest obstacle in the current e-paper technologies is the slow redraw times. This makes a strong argument for content to be pageable as e-books, and not contain any active elements, such as Flash animations which the e-paper would struggle with. This would also make the use of zooming and panning for navigating large sheets undesirable, as each of these actions would require a redraw of the display. Another undesirable effect of using zooming and panning is that it can make navigation on each sheet harder and

[1]The Fujitsu FLEPia with its 12 inch alternative is also coming soon, but if the redraw time is as high as it seems it will probably not see widespread use as a reading device.

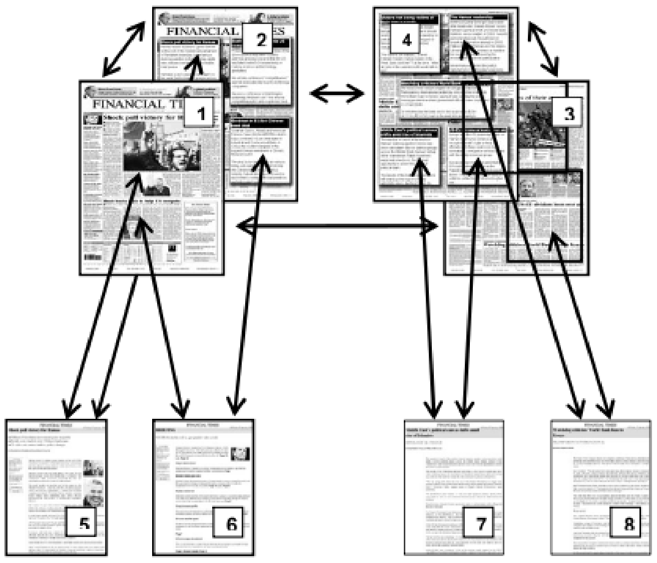


Figure 4.2: An alternative layout, with a composite of paper newspaper layout, with hyperlinks to "zoomed in" versions of each article with a more online kindof layout.[32]

unclear. This results in a few unsatisfactory elements related to the use of an original sized newspaper on an e-paper device. The alternative for keeping the newspaper layout and feel is to have it specially made for the smaller format by a layout artist.

Using a layout of an online edition would not be entirely without undesirabilities. A standard online newspaper is as noted mostly constructed as a single column of text and illustrations. The need to centre reading at an index page is also an unfortunate element, as a subscriber would get a significant amount of visits to this root page and its redrawing if the goal is to read an entire newspaper.

A combination of these would be a valid approach. Articles are divided into pages fitting the display size with the possibility to flip between different pages of an article as well as between subsequent articles. With an overall index page at the root of the newspaper to quickly jump to desirable sections or articles this would seem to be the best approach with the given hardware.

There is a last alternative which is slightly closer to print on paper, which is also described in the "How to make content" manual of the iLiad [32]. It is to use the actual newspaper as some sort of index. Each page is zoomed out so they will fit the e-paper display. The user can then flip through the newspaper, and when they see an article which looks interesting (the headliner and possibly the ingress are legible) they can click on it to bring the entire article into view and read it. This can be seen in Figure 4.2

4.2.1 Chosen approach

The goal for the demonstration is to make a simple demonstration that could work independently and not need much hands on work to be able to do its thing. It should also be easy to use for subscribers and users.

A newspapers sheet which is too big to fit the screen is not easy to use and making a special small version would require a layout artist to prepare it. Using an online style with a root index page is not satisfactory on a slow device as the iLiad. The last alternative mentioned above is promising in usability, but in these curcumstances would require more human interaction than desired; each page would need to be prepared as a photograph with mappings on each article, a task which would be hard to automate. This leaves the alternative which is a cross of paper and online newspapers; an index at the start linking to each article, the articles are divided in several pages and the user can flip pages from one end till the other.

5 Architectural considerations for an e-newspaper service

A newspaper service for the iLiad is most easily realised with some kind of client-server architecture. The content of a newspaper would then reside on one or more servers while the clients download this content. The main processing that needs to be done is the the converting of text, illustrations and such into an e-newspaper that fits the e-paper device it will be read on. This can either be done at the server or the client, the former resulting in a thin client when the latter results in a thicker client.

5.1 Content

Before continuing there would probably be in its place with an explanation on content in the context of this thesis. Content is what the newspapers consists of, this is most commonly articles, which again conains text and media such as images and illustrations. Content is also used as a description on the prepared e-newspapers a user or subscriber reads.

5.1.1 Subscription

What the server offers to the e-newspaper subscribers is a question to consider. Currently a subscriber of an ordinary paper newspaper will get the entire issue delivered on the door every morning. Most online newspapers offers some kind of web feeds as means to subscribe to their news. These feeds is currently made as XML documents that feed readers check repeatingly for updates. Each XML document is usually for a single section or category, and consists of an arbitrary number of entries. Each entry usually contains the title of the article, the ingress, authors, datestamp and a link to the rest of the article.

The exact details of this vary due to a number of different standards and versions of each standard, Atom1.0 and RSS2.0 are two popular standards.

When an online newspaper offers web feeds it usually have many alternatives available. This ranges from all news published on the site to separate feeds for sports, football, culture to even more narrow selections such as news concerning a single football team. The newspaper can even make software that interfaces with their content management system (CMS) and let users compose their own feeds based on a selection of all the tags that the articles are accompanied with.

An e-newspaper service should be able to atleast provide the same level of subscription options; the goal should be to deliver e-newspapers completely tailored in the way the subscriber wants, whenever she wants it.

5.1.2 Content updating

One goal of an e-newspaper service is to provide a subscriber with the possibility of having consistently up to date content on their devices. Depending on the kind of newspaper that provides the content, throughout a day it is very likely that new content is published in some way. This content could be either an entire new newspaper, a new article, or maybe only an update to an existing article. How this content is propageted to the subscriber is of some consequence, and there are two main alternatives for this. Either only the new content is sent to the subscriber or the content is processed and incorporated in what the subscriber already has, e.g. a new article is added to an existing e-newspaper and then the entire e-newspaper is propagated to the receiver. How difficult each of these are as well as how suiting they are, will be reflected on when both the thin and thick native client are described in the rest of the chapter.

5.2 Client-server with native thick client

Doing everything on the device with a native client is one obvious approach. With such an approach the client running on the device would be doing all the work, with the server tasked only with serving the textual and illustrative content. The client would download the content when the user wants it, then convert and assemble it to a newspaper and present it for the user. This is more or less what a web browser such as Opera and Firefox does on an ordinary computer.

What the server is responsible for can be stepped up, it could be tasked with transcoding and adapting the illustrations to a format suitable for the e-paper client. It will most likely be some years before there are a majority of e-paper devices with colour displays. The colour technology is still in its infancy as well as generall small interest in e-paper from consumers.

The last responsibility that could be put on the server is the management of mobility. Handover and that kind of mobility is assumed provided by the network, as in Wireless Trondheim. What the network cannot provide for is content that happens when the client is offline or gets disconnects. This is only of importance if the distribution technique used is of push character. For pull distribution this responisbility is on the client as the content will be ready on the server for downloading at all times.

Advantages of a thick client approach is that the user can tweak the local client to her own wishes, this could be elements such as font size and type, existense of images, what content to include and other elements of how an e-newspaper can be represented. The disadvantage is that tasks the server easily could do once, would be done many times by all the devices with this client. If the work is of a hardware intensive kind, the mobile device would take a greater toll on its battery doing it than a server running on cabled electricity. Lastly this could result in a fair bit of content being unnecessarily transferred to the device that will be scrapped or transcoded due to incapability with the device.

Regarding the earlier mentioned difference in downloading all or only new content, the client would only need the new content as it assembles the e-newspapers themselves.

With a native client most of the subscription process could probably be managed on the device. If the subscription process is performed from the device is dependant on whether content is pushed or pulled to the device. With pull distribution, the subscription is more a list of what the user wants the device to download at given times or when user initiated. For push distribution this would have the client editing a same kind of list, and then sending it to the server, and the server will provide the client with content of its wishes.

5.3 Client-server with thin client

With a thin client the server would be tasked with most of the work. The content would be assembled to a newspaper with text, illustrations and whatever else a newspaper for a subscriber might contain. When that is done, the client on the device would download it to the device, and be mostly tasked with simple book-keeping.

Advantages of following an approach like this is that the server will offload the device by doing a lot of the necessary processing before the content is downloaded. This processing will also be done once for every piece of content instead of once on every device that are used to read that piece of content. The device will also avoid downloading a lot of content the subscriber will not have any benefit of, such as colour images on a device with only grayscales display.

For this kind of approach, which mainly relies on receiving more or less finished products, the method of only receiving updated contents would be difficult as it is not meant to have that kind of functionality. It would be better off receiving the updated newspaper from the server.

Disadvantages of this is that the subscriber will get less options for customising the newspapers he is receiving. To prevent the server becoming to complicated, a certain level of customisability would probably need to be set. E.g. the newspaper service could let a subscriber choose which newspapers to subscribe too, and which sections of a newspaper, but not a custom layout for each kind of article and such micromanagement aspects.

5.4 Client-server with "medium" thin client

The approaches described above are fairly different, and at the end of each extreme. An approach which lies in the middle is possible, with possibly the best from each of the extremes. This would include a client that are as lightweight as possible, but with a modicum of necessary and beneficient functionality. If the server could provide the client with chunks of processed content, not as granular as for the thick native client or in a single piece as for the thin client. This could be as sections or articles of an e-newspaper. If the client could assemble this to a single updated e-newspaper based on the different pieces of content it was receiving. This makes it easier to update an e-newspaper throughout the day as news content trickle down to the device.

In addition it would need to keep some sort of archiving function, which for example could be saving the content chunks it received and in some way also be able to show these in their relevent e-newspapers. E-newspapers could e.g. appear for the user archived as one issue a day.

The difference compared to the thick native client is that the content is preprocessed when it comes to the device, and there does not need to be any concerns about the layout as it is all fixed for the device.

5.5 Relating to real world examples

The examples given above can be related to real world examples to give an impression on how they can be achieved. Starting with the thick native client, it could be supplied with XML files with all textual content and information on where to obtain images. This it would download itself. When all parts are obtained it would process and put it together in a format that a user could read. Alternatively it could download original HTML files from e.g. an online newspaper, and then process this to a pageable format with optionally keeping the layout from the original newspaper.

Continuing to the thin client, this could for example be the server assembling all the e-newspaper content to a single file such as a PDF. The client can then download this single file each time it is updated with content that it subscribes on.

Continuing to the last kind of client, which needs to be supplied with coarser content than the thick native client. This could be HTML files that already have been paginated by the server, that the client can use to update the current e-newspaper with when they are published and downloaded.

5.6 Summary

The thick native client can be seen as a client that takes raw content or content in a format not directly suited on the device. This content the client will turn into something that fits on the device. The thin client is more of a dedicated reader for a format, with a few builtin functionalities suited for newspapers. Lastly the client in the middle is a

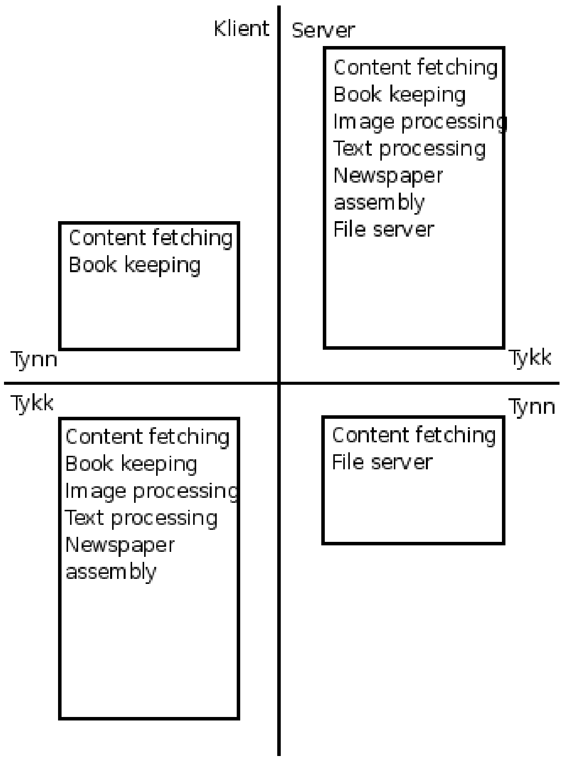


Figure 5.1: Illustration of thick and thin client and server.

more flexible variant of the thin client, with support for updating only the relevant parts of an e-newspaper.

In Figure 5.1, an illustration of a thick and thin client and server is provided.

6 Preferable approach for an e-newspaper service in Wireless Trondheim with the iRex iLiad

After some consideration, the preferable approach for an e-newspaper service in Wireless Trondheim with the iRex iLiad device is a thin client with a server responsible for most of the processing, perhaps some of the functionality mentioned for the medium thin client, and with pull distribution. This is also the chosen approach for the implementation of the demonstration.

6.1 Tools

The iLiad comes with a set of tools for making software for it, and can easily be expanded with more tools to open up the possibilities for different kinds of development, programs and functionality.

6.1.1 Writing or porting software

As the iLiad runs on Linux, there are already a plethora of existing software that can be adapted and made to run on the iLiad. As mentioned in 3.1 iRex has released a toolchain to ease this process. In most cases it is often enough with a recompiling of the source with the help of the mentioned toolchain. The greatest obstacle is the e-paper display which does not cooperate fully with what the X Window System(X11)[1] wants to. To get the device to draw when something happens onscreen, modified X11 liberaries needs to be used.[62]

[1]The X Window System is a windowing system that utilises a protocol named X. The X protocol is a client server model with the users terminal as the server, the server offers display services to programs.[71, 22]

To give some examples of what can be ported, the iLiad can be outfitted with the word processor Abiword[2], the web browser Mozilla Minomo[3] and the e-book reader FBReader[4]

When a program for a problem does not exist it must necessary be written. Most software used on the iLiad is written with C or C++, as much software for Linux is. There is one exception to this, the Java based MobiPocket reader, which is run with the help of a proprietary Java Virtual Machine (JVM). There is not much documentation available for this JVM and what libraries and functionality it might provide, making it harder to write third-party Java software to be used on the iLiad[5]. There has also been recompiled a Python interpreter for the iLiad with basic functionality, but there is not known of any "useful" applications made with this yet. [61]

6.1.2 Shell scripts and commands

As well as the ability to launch programs like those mentioned above, the iLiad can run shell scripts consisting of basic Unix commands and utilities. This is done with the help from BusyBox[6], an application that can provide a subset of these Unix functionalities in a small package suited for systems with embedded Linux such as the iLiad. The iLiad in a standard configuration is equipped with Busybox version 1.01, but an iLiad enthusiast at the MobileRead forums[8] has made an updated version for the iLiad with about twice as many commands available. The original and updated list of commands available at the iLiad can be found in Appendix B.1 and Appendix B.2.

6.2 Reasoning

There was several reasons for choosing this approach. One of the main reasons was that it was with respect to the available hardware and software, the far easier of the two main approaches. When making a demonstration there are certain elements and functionality of a fully fledged service that are too much work to add, even with that in mind there would still be far easier to follow the thin client approach.

6.2.1 Native thick client

As mentioned in 6.1.1 there exist both a Java Virtual Machine and a Python Interpreter for the iLiad, but not much documentation or working examples that can aid in making a native e-newspaper application. Thus the client software would most likely need to be made with C or C++. While the author is quite a novice in these programming

[2]Abiword: `http://www.abisource.com/`,
iLiad Abiword: `http://www.mobileread.com/forums/showthread.php?t=14953`

[3]Mozilla Minimo: `http://www.mozilla.org/projects/minimo/`,
iLiad Minimo: `http://www.mobileread.com/forums/showthread.php?t=15448`

[4]FBReader: `http://www.fbreader.org/about.php`,
iLiad FBReader: `http://www.mobileread.com/forums/showthread.php?t=20490`

[5]The company that created this JVM, Tao Group, have appearently gone into administration, which is unfortunate for the iLiad.[18]

[6]`http://busybox.net/`

languages, this is no real disadvantage with the right programmer behind the wheel. But in these circumstances this was considered a disadvantage as there would need to be put some unknown amount of time into acquiring the right C or C++ skills for programming a demonstration on an embedded Linux platform such as the iLiad.

Alternatively a program written for "plain" Linux could be ported to the iLiad, but again such a program would need to be written with C or C++ if there is to be any hope of porting it successfully. A slightly different approach would be to use already existing software, and modify it to get the right functionality and then port it. There have not been much luck in finding anything that fulfills enough functionality to bother with modifying it for the purpose of demonstration or a real life application later on. Late in the work with this thesis, the last statement changed to be a truth with certain limitations. `Feedbooks.com`, a web site that provides content for e-paper devices, mostly books but also the possibility to get excerpts of webfeeds and such, has lately released an early version of a piece of software they have made and ported to the iLiad, which lets a user browse and download content from their site. This could possibly act as as either inspiration or something to build on, given that it is to be released as open source.

But it is not only the programming deficiency of the author that is the reason for this approach not being chosen. The iLiad is not a greased lightning, and if it was tasked with doing to much, would probably give users a too slow experience. Though, the thick native client approach sketched in the previous chapter is probably a bit on the extreme side of things, and an approach with more done on the server is probably more viable, closer to the medium client-server approach described.

6.2.2 Thin client

The main reason for going the route of a thinner client was as mentioned it being easier and more doable than a native client. Most of the functionality can run on "normal" hardware with software written in more high level programming languages. Again, the programming languages chosen is of less consequence in a real life implementation, but when the author is to make a demonstration it is comforting to use languages and tools that there is familiarity and knowledge with. The software on the iLiad can with this approach be of the simpler kind, for most functionality some shell scripts can provide the necessary tools.

Looking away from any programming language considerations, there are other reasons for a thin client being suitable in these circumstances. The iLiad being no powerhouse promotes the notion of doing as much processing as possible server side. Not valid yet, but sometime in the future, when iRex' own iRex Delivery Service (iDS) opens up for more general usage, it can be a great asset for content delivery to the iLiad. The iDS would probably be of best use for a thin client approach than a thick client who would either need to interface heavily with the iDS or might even need more control over the download process.

6.2.3 Pull over Push

Battery and battery life is the main reason for choosing pull over push as means for distributing the e-newspapers. Push technology while more convenient for the device and its user would require the device to have its wireless interface always online. Previous experience with the iLiad shows that battery life is one of the areas that sorely needs improvements. Until the battery life with wireless enabled is at an acceptable level pull distribution is the best choice.

Many users of e-paper devices do not need to have the newest news brought to the device each time they are published. They need them either at preset times such as ready for the commute, or they need them brought to the device only when they sit down with the e-paper device to read news and clicks on the “Read newspaper” button. If the update process for the latter case is fairly quick there is no huge disadvantages for the user compared to have the device always updated.

6.3 Design and plan

After deciding on a main approach, a more detailed plan and design was devised for the implementation, but also with elements for a more complete solution. Since the implementation is for a demonstration, several aspects of a full fledged e-newspaper service needs to be toned down and simplifyed to make it doable. For completion it was decided to try to describe a complete e-newspaper service, but with focus on the demo implementation.

In Figure 6.1 an approximate model is shown of the different parts that the envisonaged e-newspaper service will contain.

6.3.1 Server side

The server is where most of the work will be done. The server needs to get or receive all textual and illustration content. Text might need to be encoded and escaped for further processing while images and illustrations should be in grayscale and at the right resolution to minimize file sizes. Then all the content needs to be assembled to articles with a given layout and then put together to sections and finally e-newspapers.

Subscription management

It is envisaged that the e-newspaper system provides for its users a subsription system. These subscriptions must be managed and controlled from server side, and the functionality that should at least be possible for subscribers are:

- Register subscriber
- Tailor subscription
  - Choose newspapers

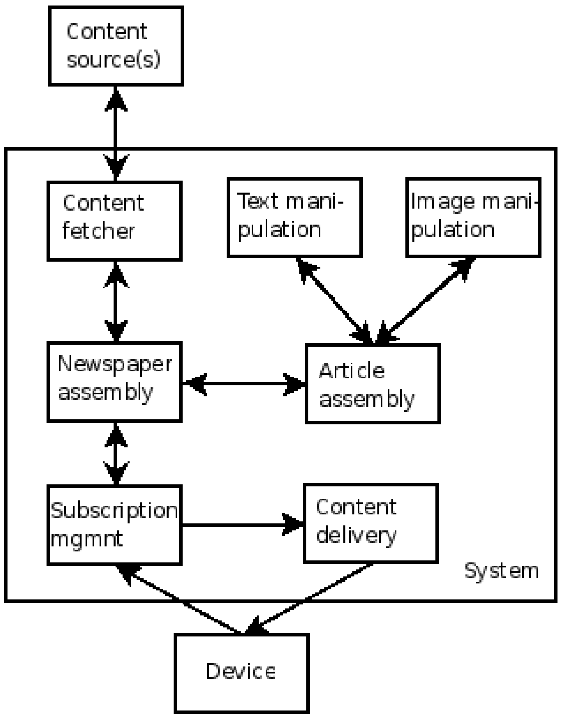


Figure 6.1: Model of the different parts and e-newspaper service might contain.

  - Choose sections of newspapers
  - Choose among optional layouts for newspapers (illustrations, no illustrations etc.)

- Tailor subscription delivery
  - Choose how often content are to be delivered/updated (instant/hourly/dayly etc.)
  - Choose method of update (push, pull etc.)
- Available archive of earlier content
- Subscription payment

These are all what the subscriber should be able to do. For the provider there are some additional functionality that are needed. This is among other:

- Suspend subscription
- Stop subscription
- Add content to list of available content

- Add layout options
- Billing

Demonstration For the demonstration it was chosen to not bother with this kind of subscriber functionality. With only one newspaper available and only a few testers it would be too much hassle to be worth it in the end.

Content delivery to the server

The content published by the content sources needs to find their way to the server that are to turn them into e-newspapers. The least amount of information for each article should be;

- Title of article
- Category or Section it belongs to
- Author of both the text and illustrations
- Ingress or introduction to the body text
- The body text
- The illustration

Optionally there could also be elements such as subtitle, caption for the illustration and noteworthy quotes. The body text should have some kind of structure with headers and paragraphs.

To transfer this between publisher and server, XML would probably be the best medium. There is easy to make an XML schema that includes the information mentioned above, and there are software that can read and understand XML documents for most kinds of contemporary programming languages.

Demonstration The student newspaper Under Dusken was chosen as the main content contributor for this project, and initially it was decided that they would provide the content as XML. XML as an approach was not followed through, as will be explained in 7. A schema and some test examples of such an approach was made as well as some prototype scripts to test it. These can be found in Appendix C.1 and C.2 for completion.

Without making some kind of custom system, web feeds are an option for getting content to the server. Web feeds, most commonly RSS or Atom, are specially formatted XML files that can contains most of the content mentioned above for XML. The reason for this being preferable over XML is that there already exists solutions for publising content to web, and it would be easier for Under Dusken to customise a special feed for this purpose based on something existing than making some custom system for outputting XML.

Text manipulation

Ideally the textual information entering the system is encoded with Unicode, and all characters are represented in a correct way. Unfortunately things can happen when several systems are to talk together, garbled characters such as "`&amp;aring;`"[7] can occur. If or when this happens there needs to be some way of restoring the original character.

The best is of course to prevent it in the first place, but sometimes it can happen. To restore the characters to their original, the most usual approach for this kind of problems is to have dictionaries of common characters and what their garbled counterparts is.

Demonstration There is great likelyhood for this happening in the implementation of the demonstration, and this is partially one of the reasons for choosing to make the server side with Python, a programming language that are strong on text manipulation.

Image manipulation

Images used in both paper and online newspapers are of a kind not suited for e-paper displays, and needs to be adapted to both grayscale and a size suited. Doing this will provide the e-paper device with images that do not contain information it cannot show, as well as not wasting any bandwidth in the transfer.

Demonstration The e-paper display of the iLiad can show 16 shades of grayscale. Before images are to be transferred to the iLiad or used in e-newspaper assembly, all images should be transcoded or adapted to an appropriate size and grayscaled. The iLiad does not have abundant amounts of storage capacity, and transferring images and illustrations that the receiving device can not show in its entirety (i.e. size or colour) is completely unneccessary and a waste of resources.

Article assembly

Regarding how each article should be put together, there is a few different ways to achieve this. Each article contains one or more pages of text and illustrations. In addition there should also be information about the author, the photographer or illustrator.

Formats The articles can be assembled either in a fixed file format, or in a reflowable file format. PDF is an example of the former, while HTML can be an example of the latter.

With HTML giving one file for each page of an article, it is an an obvious choice if the wish is to make the updating of the newspaper as lightweight as possible. Only new articles needs to be downloaded to the device, and is then slotted into the existing issue.

PDF files makes it easier to make one file out of the article or even newspaper without using archiving tricks ("zip-files"), as every element can be gathered in one file. The PDF

[7] which is the same as the nordic character "å"

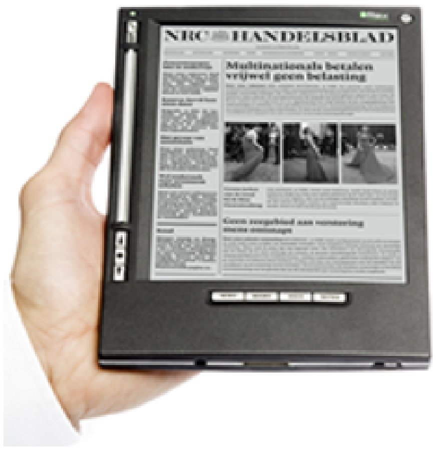


Figure 6.2: An illustration of what the NRC Handelblad looks like.[24]

file also allows the design to be more detailed and guaranteed, as everything is bound to the file and not the interpretation of the markup as is the matter with the HTML viewers.

Layout There are two main different approaches for layout in article assembly. Either the article are put together automatically by the system with the content it has been provided. The alternative is to have each article put together by, or at least monitored by, a skilled layout artist.

These different approaches are suited for different kind of content. Doing the layout with a human layout artist in the loop introduces a cost that might not be worth it for all kind of content. Much of the content provided by various content publishers are small pieces of content that are not much processed or thoroughly prepared. This kind is more suited for automatically assembly. Thoroughly prepared and high quality content might be worth the extra cost of assembling it with a custom layout.

A tradeoff is to use skilled layout artists in the work on the templates, to get highest quality possible. There is also the possibility of creating several layout templates, and having the algorithm choose the layout that fits best with a given set of parameters. The NRC Handelblad seems to have put some effort into their layour, which can be seen in Figure6.2.

Demonstration The iRex iLiad can among other formats display HTML and PDF files, which are the most interesting in this context. HTML files will require one HTML file for each page of an article if it is to be of a pageable configuration as was decided on in 4.2. A PDF file on the other hand can also provide such granularity with a PDF for each page, but due to the way PDF viewing works in the iLiad it is not desirable with a PDF file for each article page, nor for each article. The reason for this is that the PDF viewing program would need to close and open between each successive PDF file, an operation that takes some time. The HTML viewer uses a bit longer time than the PDF viewer on a single file, but it will act and respond almost instantly on hyperlink clicks.

Newspaper assembly

A newspaper consists of one or more sections, and each section contains one or more articles. As the newspaper can be split into several parts, the question is should the newspaper be delivered in its entirety with articles collected in sections and the sections bound together as a single entity or should it be delivered in parts?

Update method If each article could be delivered to the e-paper device singly, a lot of redundant transfers could be avoided. More network use than necessary is taxing on the battery for devices like the iLiad, which in the first place is not of very great capacity. Even though bandwidth is getting more and more abundant with every new technology, there should be no need to squander it away on transfering redundant information.

Delivering the entire newspaper in one file, requires less logic mainly server side. The client it is decided will only do the viewing and archiving of e-newspapers, and will mostly need to archive at least one issue a day. Server side there is no need to manage several files, and know and tell the device which files are of the current issue.

An important question to consider for this is how often and what kind of updates a subscriber wants. If the subscriber wants a newspaper a day, there is no benefit from having the newspaper consisting of several smaller files than one large one. If the subscriber wants to be updated more often there are more benefits to reap from smaller elements.

Assembling the entire newspaper for each time there is a new newspaper will be the easiest on both server side and client side. The downside is that this will introduce many redundant downloads as an entire newspaper is downloaded even if it contains some old articles. On the contrary to paper subscription newspapers that are delivered with all new content, an e-paper newspaper can due to its digital nature be updated many times during a day. But there are rarely such speed or quantity in the updates that would make an e-newspaper completely purged of old news on each update.

Manifest file The iLiad uses a type of manifest files for the organising of each folder. This file decides which files in a folder that belongs together, and how they are shown. Each manifest file is an XML file which contains two main parts, one about metadata and one about the indexing. The metadata is used to show a summary of the contents in the file browser, and contains information about the content such as title, date, type and format. The index part details how the content shall be represented in the iLiad, in which order the pages are to be shown, and what the iLiad shall show on the page indicator.[32]

Demonstration It was decided to use HTML files for the demonstration. This mainly due to the fact that there was a HTML templater readily at hand, but also because the possibility of updates that required less wireless transfer was tempting and most interesting. It was further decided to go with the automatical assembly of both articles and e-newspapers. The reason for this was that it would provide the possibility for faster updates with the chain being entirely digital from the moment the article was published. As Under Dusken publishes some conent that does not go into the paper version, this approach can easier catch the slack.

Delivery

Delivery is either push or pull. Either the device will initiate the downloads itself, or the server initiates by pushing the content to the device. This is more described in 2.5.3 and 2.5.2.

Push Instead of or in addition to pushing the content, the server could alternatively push content notifications, telling the client that new content is available and ready to be downloaded from a given location.

If the server is to push content or content notifications to the device, the device will need to have its wireless interfaces turned on all the time. The iLiad does not have a very good batterylife, and with the WiFi enabled at all times, significantly shorter. Thus it was decided to go for pull distribution, at least for the demonstration.

Pull With a pull scheme for downloading, the server will only need to have an area available that the client easily can download from. Every kind of web server can do this, and the public part of the network storage NTNU supplies for its students can do this easily. The NTNU servers proximity to the Wireless Trondheims network is also a positive factor.

Practical

For making the server side software in the demonstration, Python was chosen. The main reason for this was due to it being well suited for tasks that involves text processing as well as the author having familarity with this language from earlier endeavours. There also exist a plethora of modules and Python programs that makes programming with Python powerful.

Django The author was making some other projects with the Python based web framework Django at the same time as the thesis startup. As working with Django requires use of Python, it seemed as a good choice to continue with Python. Django do also possess some features that could come in handy, namely a templating system.

Django is a web framework made with Python that utilises a model-view-controller pattern. It consists of data models mapped to an relational database, regular expression URL dispatching, view mechanism to handle and process requests and a template system. It is also equipped with an administration tool for managing the data models. [28, 13, 14]

6.3.2 Client side

With a thin client setup the software on the device is mainly tasked with downloading new content and managing the old files. For the actual delivery the question is whether the device should manage the downloads itself, the user starting the downloads or finally the content being pushed by the server to the device as mentioned above in 6.3.1. Alternatively the server pushes content notifications to the client that acts upon it by downloading the content if it wants or needs it.

Delivery

System initiated The iLiad got a built in feature for automatically downloading updates and such via the iRex Delivery System (iDS) at fixed times. The problem is that this is until further exclusively for iRex' own updates and a few collaborating partners. This automatic download can also be enabled for automatically downloading from a computer on the same subnet via a Samba share or such. Using this feature should probably work as Wireless Trondheim resides on its own subnet. This feature was introduced in the last update to the iLiad, which brought it to version 1.12, but the usefullness of this was not discovered until well into the thesis. And as HTML files was initially chosen there would be need for some book keeping and organising that would needed to be run as well.

Cron and Crontab If one wants to achieve this functionality without using the built in, there are a few possible alternatives. The set of Unix commands and utilities the iLiad is shipped with is not large, and one of the utilities that are much used in the *nix world for scheduled work is missing. This is the `cron`[59] and `crontab`[58][8]utilities, which enables a task to be scheduled to run at specific times, from every minute to yearly and anything in between. As noted in 6.1.2, there exists an update to Busybox which brings more utilities to the iLiad, in this case `cron` and `crontab`. Unfortunately this update was not discovered and tried until the end of the work with the demonstration.

Sleep Cron as noted in the last paragraph is a daemon that can execute commands at given date and times. With no cron available in the stock iLiad the choice is to emulate this with `sleep`. With `sleep` a process is paused for a given amount of time, and with this the content downloading can be run, paused and run again in a loop fashion.

[8] `cron` is a daemon that every minute will wake up, check all stored `crontabs` for any scheduled tasks be executed, and will then execute those who are scheduled to run. It will also check its crontab directory for any changes and will reload those that have been changed. A `crontab` is a file that contains a users instructions for the `cron` daemon, i.e. which commands that should be run at given times.[59, 58]

First it was considered using this, but initially it was thought that this would prevent a user of the device to do nothing but downloading as only one “main” process can be run at one time in the user interface. Though, after some time writing and learning more about Bash scripts, it was discovered that processes easily can be put to run in the background. At this time, another approach was chosen. But such a process would need to be restarted each time the iLiad is turned on, the alternative is to perform the tricky business of adding it to the processes that are started on boot.

After some previous experience with an early iLiad version 1 and its poor batterylife in the prestudy project, having something that runs all the time in the background makes one a bit sceptical.

User initiated The easiest alternative to implement is a user initiated system, with the user starting the download each time she wants to check for an updated newspaper. This leaves the action of initiating the download to the user, which none of the others do. This adds a slightly negative aspect to it for the user, as she has to check for any updates herself. The faster this check for updates is, the better the experience will be. The device should not redownload any content if there is nothing new.

The iDS mentioned above can also be user initiated with a push of a button. The details of how it actually works are few, but iRex has promised that they will open it up at some time in the future, and letting people and businesses use it. Basically it works like this: Each registered user has a virtual mailbox that content can be sent to. When the iLiad connects to the iDS, it will check if there is any new items in the mailbox and download these.

Not using the iDS, the download functionality can be achieved with a download script using some of the Unix commands and utilities that Busybox brings to the iLiad. It is no real need to update the Busybox to the newest version, as the required functionality is present, mostly important the `wget`[42] command. `wget` is a network downloader, and can be used to download files from both HTTP, HTTPS and FTP locations. The version of `wget` supplied with Busybox on the original iLiad do miss some of the functionaily of the full fledged `wget` for standard *nix systems. One of the features that miss is the timestamping function which makes `wget` skip files that are of the same name and not newer than the local copy.

7 Implementation

With an overall design and plan as sketched in the previous section, 6.3, the actual implementation could commence. This ended up as not entirely smooth sailing, and a few changes of approach and method was needed for a few of the elements.

7.1 Server side

The server side is most laden with logic, and does most of the work by turning given content into a newspaper that can easily be transfered to and read on the iLiad.

The initial plan for the server side functionality did change some during implementation, this is described in the following subsections.

7.1.1 Content delivery to the server

Instead of using such a custom XML approach as described in 6.3.1, there was later decided to use a slightly customised RSS feed instead. The reasons for this was that it was easier for Under Dusken to realise, as they only had to make another feed in addition to those they already had, instead of making something completely new to interface with their content management system.

The implementation goal is a demonstration, thus easing some of the requirements. It is only one newspaper, and to make it less complex there are no customisation of the newspaper for users. The newspaper delivered is split in sections, with each section corresponding to a section in the online version. The online version do have most of the same sections as the paper version, with a few exceptions. These exceptions are mainly small notices and sections that might not suit the online format such as the parodic “Spitposten”.

Table 7.1: An overview of the (relational) database models used to create an e-newspaper

Author	Name
Illustration	Title Caption File Author
Category	Name
Article	Author Cateogry Title Subtitle Ingress Bodytext Illustration
E-newspaper	Issue Volume Articles

Initial plan: XML

The initial plan was to use XML as the input into the system. This could either be RSS style with the XML file residing on a web server to receiving it in an e-mail and loading it into the system manually. This is included here for completion as some thought and work went into it.

As described in 6.3.1, such a newspaper service as this needs some sort of management system, but most of the mentioned functionality would be too ambitious for a mere demonstration. For testing purposes a basic Django system was set up that could input XML files and save them to an SQLite database using the given models. These models correspond to the XML schema mentioned earlier. Schema and XML can be found in Appendix C, while the implemention of the models can be found in the file `models.py` in the code folder of the accompanying archive file. A summary of the models is provided in Table 7.1.

Loading the XML When one of the test XML files was loaded into the system, the XML would be processed by the application and could be read as a DOM[1] tree. With the information readily available it could be read and saved to the right places in the database.

[1]Document Object Model, it will present an XML document as a tree structure.[46]

Showing the articles and newspapers With newspaper content in the database, it is fairly easy to output this as HTML files with the builtin template system. Basically one sets up a basic HTML page, and then hooks and basic logic are placed to dictate how the content from the database shall be represented. Each webpage has a view that fetches the right content from the database, and can also do more advanced processing to it than the template logic, which is basically restricted to if/else and loops.

Getting the content to the iLiad As was earlier explained, it was decided to use HTML as the format for the articles and newspapers. Using CSS stylesheets on the HTML that are output from Djangos template engine, can achieve a good fit to the iLiad. The hardest task is to paginate content, as most web sites present each content element (articles etc.) on one page that can be scrolled. The only paginate support in Django is the pagination of list elements, i.e. 10 each page and such, akin to those produced from search results. How pagination was achieved will be explained later on. The switch to RSS happened at the same time as the experimenting with a CSS stylesheet to show a single page of HTML on the iLiad.

Using RSS to obtain the content

In the end, using XML was scrapped in favour of RSS as it made it all easier to automate both for publishing the content and obtaining it. An RSS document uses XML to express its document, and several parts of what was made to test XML could be reused to work with the data from the RSS documents.

It was made a custom RSS feed for this demonstration. The tree structure of it was as following:

- Articles
  - Article
    - Title
    - Author
    - Published
    - Summary
    - Text
    - Section
    - Imagelink
  - Article
    - Title
    - Etc.

Compared to standard RSS feeds for online newspapers, the difference is that they do usually not contain the body text, nor the imagelink and author. A copy of the RSS feed used can be found in the accompanying archive, an exerpt can be found at Appendix D.1.

Table 7.2: An overview of some of the character problems faced in the received text

æ	“`&amp;eacute;`”
	“`&amp;aelig;`”
	“`&eacute;`”
	“`&aelig;`”
ø	“`&amp;oslash;`”
	“`&oslash;`”,
å	“`&amp;aring;`”
	“`&aring;`”

Obtaining information not in RSS For some reason the body text for some of the articles in the RSS document lost some or all of their formatting, as well as suffering from unwanted escaping of special characters such as the nordic ones, “>”, and “<”. To remedy this, it was devised a solution that could go to the correct website and fetch the source of the page for a given article. This was then processed by `Beautiful Soup`[2], a Python HTML and XML parser that can tackle most kinds of websites. It will output a parse tree that makes it easy to collect the information one seeks.

To save the information for processing further down the chain, each article got saved as a Python dictionary[3]. Each dictionary was pickled, which in Python means that it was saved and serialized as a file, and could be loaded at a later time instead of repeating the content fetching. Alternatively it could be saved to a database for the same reasons. The reason for choosing the former was that it was easy to know if this article had been processed before, its existense would decide it.

7.1.2 Text manipulation

As noted in 7.1.1, some of the text that are obtained via the RSS feed contains text that needs to be corrected and manipulated into something that is legible and understandable for a reader.

Fortunately, text manipulation is one of Pythons stronger points, and it was easy to replace the escaped and garbled characters with ones of greater meaning. It was many different versions of the same characters, and the replace definition list grew long. See Table 7.2 for an example of some of the problems.

7.1.3 Image manipulation

Not all articles are accompanied by an image or illustration, but those who are needs the images transformed into a suitable format. The goal is to convert the images to

[2]`http://www.crummy.com/software/BeautifulSoup/`

[3]A Python dictionary can be seen as an unordered set of “key:value” pairs, see `http://docs.python.org/tut/node7.html` for more information.

16 shades of gray, and a size appropriate to the e-paper display. The resolution of the e-paper display of the iLiad is 1024 x 768 pixels and it was chose to resize the images to 350 pixels height and a width that preserves the aspect ratio of the image. This number was chosen as it would let an image cover about 1/4 of the total screen estate, anything more would be a bit overwhelming given that the display size is only 8.1 inches.

The tool used for this is the `Python Imaging Library`[4], which supports a plethora of image formats as well as operations on them.

7.1.4 Article and newspaper assembly

The assembly of the article and newspaper is probably amongst the most crucial processes in the creation of an e-newspaper. The result is what a subscriber spends most time with, and how the articles and the e-newspaper are presented are of great importance. The assembly of an article and newspaper can be done in roughly two ways. Either it is assembled by a (professional) layout artist, or it could be done automatically by appropriate software. As this demonstration mostly concentrates on an e-newspaper that are updated each time a new article are available, having a human in the pipeline would complicate it further and make the update process slower.

Ruling out any human interaction means that the assembly needs to be done by software in one way or the other. As was explained in the Design section 6.3, HTML was chosen as the medium for the article and newspaper. Django was mentioned as having among other functionalities a template engine.

Templating HTML

With both the textual and illustration content loaded and prepared it can be processed into the final HTML files. For this purpose, the templating engine of Django was chosen. It takes a Python dictionary, and will place the right content at the right places. The template files that was used can be found in Appendix G.

Pagesplitting Before the HTML files can be rendered by the templating engine, the body text needs to be split into several sub-texts, one for each page if the text is longer than what there is room for on one page. This was done by testing how many chars of body text there was room for in a HTML file with the e-newspaper CSS applied. This amounted to about 1 400 characters on the start page of an article, and about 2 500 characters on the subsequent pages.

The text was then split into parts with 1 400 characters in the first part, and 2 500 in the rest. This method is a bit crude, and does not take into account words or other elements that one might want to conserve in its entirety. HTML was as will be discussed in a moment left out of the final demonstration, and this method was not refined further.

[4]`http://www.pythonware.com/products/pil/`

Template syntax A part of the template can be found below in Listing 7.1. This part will render the HTML code that shows the illustration, ingress and authors of the text and illustration. The `{{ object.### }}` will be replace by what `###` points to in the `object` dictionary. As an example, `{{ object.author.name }}` will be replaced by the name of the author, and `{{ object.ingress|safe }}` will return the ingress of the article. The "`|safe`" part means that the `safe` filter is applied to the ingress string, which means that characters will not be escaped. The reason for this is that some strings might contain markup such as `<i>` for italic typesetting.

`{% if start %}` means that the part between `if` and `endif` will only be rendered if the `start` variable is True, which only happens when the page rendered is the first page of an article. For subsequent pages, the illustration, ingress and authors are not rendered.

Listing 7.1: Excerpt of template code that creates an article in HTML.

```
{% if start %}
   <DIV id="ill">
      <img src="{{␣object.illustration.enewsversion␣}}"
         alt="{{␣object.illustration␣}}">
      <DIV id="ingress">
         <p>
            <b>
            {{ object.ingress|safe }}
            </b>
         </p>
      </DIV>
      <DIV id="credits">
         <p>
            Tekst: {{ object.author.name }} <br/>
            Foto/Ill: {{ object.illustration.author.name }}
         </p>
      </DIV>
   </DIV>
{% endif %}
```

Manifest files and its trouble To show content made with HTML files properly on the iLiad, manifest files are used. This is as explained previously XML files that lets the iLiad know how files belong to each other. For a newspaper containing several HTML files, the manifest file should administer the files such that the pages of each article is in right order, and that the articles are in their right order.

Manifest generation To generate the manifest file, all articles and their filenames was first sorted on its category/section and with the given ordering all HTML files was given their page number and included in the manifest file for the e-newspaper. It was also created a number of start pages which contained all the sorted articles and links to the first page of each article.

If everything turns out as it should be, there will be one or more start pages with links to all sections or articles. Following these start pages, the articles should follow and it should be possible for a subscriber to flip through the entire e-newspaper from the start pages via the first article to the last article, as one can flip through a print on paper newspaper.

Problem The problem is that it is not possible to flip through the newspaper as first intended. All articles are ordered in the right way, and the page indicater shows and it is possible to jump between pages with the stylus. It seems that it is not possible to flip through several HTML files as it is possible with other formats, which also shows this pagebar. There was spent a fair amount of time to try to solve it, but no appearent solution showed itself.

Unfortunately the HTML and manifest combination was only lightly tested before spending time on a solution that relied on it, as the content manual[32] for the iLiad gave impression of HTML files and manifest files would be pageable. After a more thorough study of the manual it explains how it uses the manifest file to populate the page indicator and how it can be pressed to jump to the given page. It was thus assumed that the pagebar could be used for flipping in this situation as that is how it works for other files such as PDF files. But alas, it seems like this is not the case. If this is a bug or "working as intended" is not known at the moment.

The alternatives for newspaper assembly: PDF and mobipocket

Having no page flipping was considered as a dealbreaker in this instance. It is assumed that using the flipbar is crucial for operation of a reading device. Thus, it was needed to find a new format when the combination of HTML and the manifest file managed to disqualify themselves. As the iLiad do not support many formats, the choices was between PDF and Mobipocket[41].

Mobipocket The Mobipocket format was at the time of deciscion ruled out as it seemed to a proprietary DRM-only format, and thus not suitable to use for this demonstration. In hindsight it turns out that there is also a non-DRM version of the Mobipocket format, which would make it more desirable as the format contains some nice features. It is based on the Open eBook format by the Open eBook Forum of the International Digital Publishing Forum, and uses HTML as well as some XML at the centre for keeping things together. Based on HTML it is on the contrary to PDF a reflowable format, which makes it more adaptable to different display sizes. PDFs are fixed, and of that consequence it will fit the format it was made for better than anything smaller or larger.

PDF The PDF format was chosen before these facts about Mobipocket was discovered, and for the sake of the demonstration reflowable content is not that interesting as there is only one device to cater for, the iLiad. More interesting is the fact that Mobipocket files are delivered as one file, and can be compressed. This excludes the scheme for downloading only new HTML files to upgrade an e-newspaper issue, but compressing could alleviate this to some degree.

PDF is not an optimal format for e-newspapers, but as it was the only choice considered, it had to do. The negatives of PDF is that it cannot be easily reflowed, it is delivered as one static file and cannot be easily updated. The positives of PDF is that as a typesetting format, it can fully decide how the output shall look at a given paper size.

Assembling with PDF

Django, as had its template engine used for making the HTMLs, do have some PDF support with the help of the Python PDF library Reportlab. In the end the visual result the HTML and CSS could deliver was a bit lacking, as no article is alike there was hard to get a good result on the small space the iLiad provided. Thus it was deceided to skip the template engine of Django, and use Reportlab standalone to render and create the PDFs.

Reportlab is PDF creation library written in Python, and can create all sorts of PDFs either with low level tools to draw simple lines and other basic forms to higher level tools and libraries which can easier create complete pages.

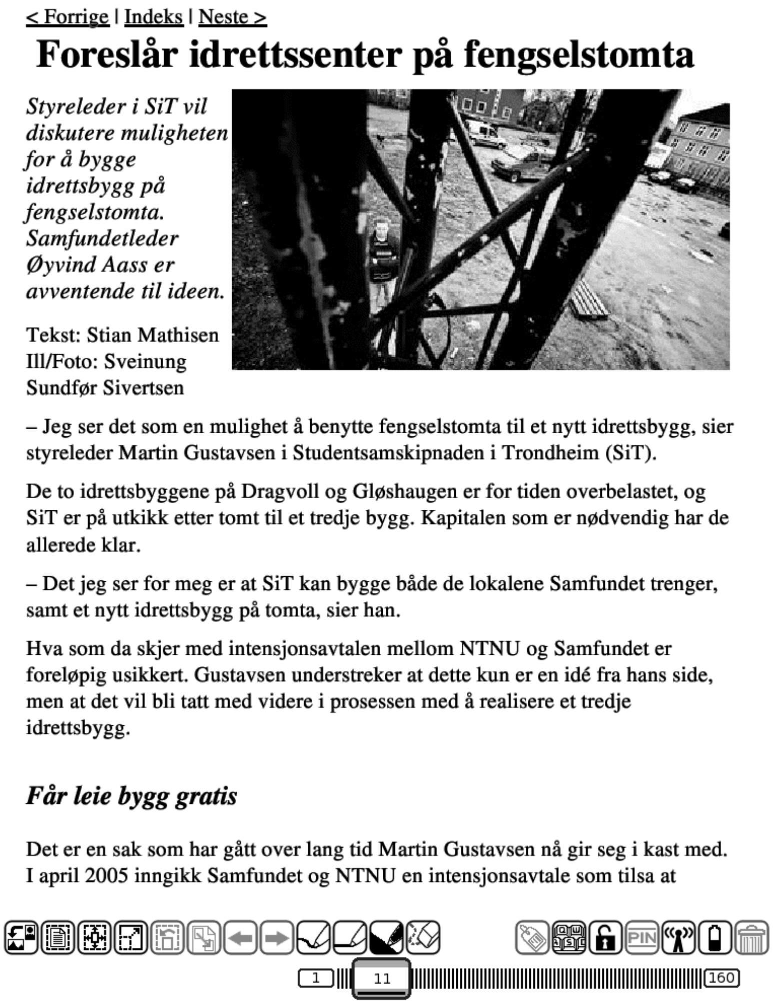
As all data for each article exists as a Python dictionary throughout the code, it was fairly easy to add the code that took care of the PDF creation, without the need for adapting the existing code. These PDF libraries can reflow text in a better and more correct way than the earlier mentioned brute splitting of text at a given character count. In addition to creating each article, there are created a index part in the beginning which lets the subscriber skip directly to a given article. Each article contains links to the next and previous article as well as to the index for easier navigation. The final result can be seen in Figure 7.1, while one of the e-newspapers made can be found in the acompanying archive file.

7.1.5 Delivery

With the finished e-newspaper as a PDF file, it only needs to be delivered to the iLiad. With the client doing the fetching itself, it only needs to be put on a place where it is easily reachable for the iLiad. The public area of the online storage that NTNU offers its students fits the bill, and its close proximity to Wireless Trondheims network makes it fit even better.

Since it is no support for running `cron`jobs on the student servers, it was decided to run it from another computer, and use `scp`[5] to transfer the finished e-newspaper to

[5]Secure Copy, uses SSH for data transfer and security. Use "`man scp`" in a *nix terminal for more information



< Forrige | Indeks | Neste >

Foreslår idrettssenter på fengselstomta

Styreleder i SiT vil diskutere muligheten for å bygge idrettsbygg på fengselstomta. Samfundetleder Øyvind Aass er avventende til ideen.

Tekst: Stian Mathisen
Ill/Foto: Sveinung Sundfør Sivertsen

– Jeg ser det som en mulighet å benytte fengselstomta til et nytt idrettsbygg, sier styreleder Martin Gustavsen i Studentsamskipnaden i Trondheim (SiT).

De to idrettsbyggene på Dragvoll og Gløshaugen er for tiden overbelastet, og SiT er på utkikk etter tomt til et tredje bygg. Kapitalen som er nødvendig har de allerede klar.

– Det jeg ser for meg er at SiT kan bygge både de lokalene Samfundet trenger, samt et nytt idrettsbygg på tomta, sier han.

Hva som da skjer med intensjonsavtalen mellom NTNU og Samfundet er foreløpig usikkert. Gustavsen understreker at dette kun er en idé fra hans side, men at det vil bli tatt med videre i prosessen med å realisere et tredje idrettsbygg.

Får leie bygg gratis

Det er en sak som har gått over lang tid Martin Gustavsen nå gir seg i kast med. I april 2005 inngikk Samfundet og NTNU en intensjonsavtale som tilsa at

Figure 7.1: Screenshot from the iLiad showing the first page of an article. A few more showing some other elements can be found in Appendix E

the server. To automate this, a Python script with `Pexpect`[6] was used to automatically transfer the content to the server with `scp`.

7.1.6 Summary of server process

First the provided RSS document is checked for any new content, if it is, these will be downloaded and saved. The data delivered to the parts further in the chain is Python dictionaries, which are "Pickled" for future references and to prevent the system from downloading articles more than once. Each article is bundled with its title, authors, ingress, bodytext, section and illustration. The ingress and bodytext will be purged of any garbled characters and escaped characters will be restored to their former selfs. Illustrations are resized and transformed into grayscale to better fit the receiving iLiad. Finally this is put together to a PDF and uploaded to a location where the iLiad can reach it.

7.2 Client

It was decided to go for a thin client approach, with only the bare minimum of functionality. As the iLiad only can support a subset of functionality compared to a normal PC, this turned out to be a wise choice. Getting the small set of desired functionality to work satisfactory on the iLiad required a fair bit of work.

7.2.1 Connectivity

The first obstacle was getting the iLiad connected to Internet via the Wireless Trondheim network. For the builtin network manager, it is not satisfied with a connection until it can reach one of iRex' own servers. The Wireless Trondheim network is not protected with WEP, WPA nor WPA2, but requires a username/password web authentication or Virtual Private Network (VPN) to your home network.

At this time there does not seem to exist any VPN software for the iLiad, so this approach is ruled out. There is not any easy way of using Username/Password to authenticate the iLiad either, without making some scripts or such to imitate the process of username/password authentication.

It was decided that registering the wireless MAC adresses of the iLiads would be the easiest way to gain access to the network. This would authenticate the iLiads based on their MAC adresses, and they would gain the same kind of connection as those who uses Wireless Trondheims SMS offer. The SMS offer is 10 NOK for 12 hours of 500kbit/s as can be purchased on [56].

[6]Pexpect is a Python implementation of Expect. It can be used to spawn child applications, control them and respond to expected output. Pexpect can be found at `http://www.noah.org/wiki/Pexpect`

7.2.2 Delivery

It was decided to make a Shell script that would download the necessary files. As mentioned above, the newspapers ended as being a single PDF file for each newspaper, and this PDF file is being updated throughout the day to contain a number of the freshest news items for each section.

The last version of the script used can be found in Appendix F. It is worth to note that this script is a bit different to the first script that worked, and the one initially used by the test group. After some revisions it ended up at what it is now, and there are still some elements that do not work completely as the author wants it to do. The revisions was primarily aimed at a bug that would overwrite the e-newspaper files with a blank file if there was no connection available when a download was tried.

Script elements

The overall task of the script is to check if there are any new files on the newspaper repository, if there are, it shall download it while keeping at least a copy of the current newspaper. This is achieved with the following commands and utilities:

- `wget` to download files
- `if` and `test`([,]) for flow control
- `mv, cp, rm` and `cd` for file control
- `echo` and > (redirect) for log files and creation of manifest files.

Script logic

The script should work as the following:

- First, the script will define all the important paths, i.e. current directory, where to save the newspaper, where to save the current newspaper and where to save the log.
- Then it connects to the network with SSID WirelessTrondheim. If that fails, it will quit and write to the log that it failed.
- When it is successfully connected it will try to download the content number for the newspaper. This number it will compare to the local, and if it is greater than the local, the newspaper on the server is newer. If it fails to download the content number file it will exit and write to log. If the current newspaper is the same as the server it will do nothing and exit.
- If the server newspaper is newer it will then download the newspaper from the server. If this fails it will exit with a write to the log. If the download is successful it will write to the log.

- Finally it will disconnect from the WirelessTrondheim accesspoint.

The iLiad do not do it exactly like this. How it compares numbers and handles the `if` functionality seems to be either broken, not working in a standard manner, or there are some fault not yet discovered in the code. This causes the iLiad to download files every time even if the newspaper server side is the same as local. Enabling debug mode and running it in `mrxvt`, a terminal emulator someone has ported to the iLiad, it seems like it cannot read the input from the content number file, and evaluates all the `if` tests as true, and performs the downloads. Why it is not capable of reading the file correctly has not yet been worked out.

Currently the download script will put the current newspaper in a “Previous” folder before downloading a new edition. Initially it was planned to have the 5 latest downloads there, but achieving it seemed a bit hard at that point with the functionality the stock iLiad was shipped with. It might be easier with the improved Busybox and its features. Now it deletes the current newspaper in the “Previous” folder before copying a new newspaper into the folder.

User interface

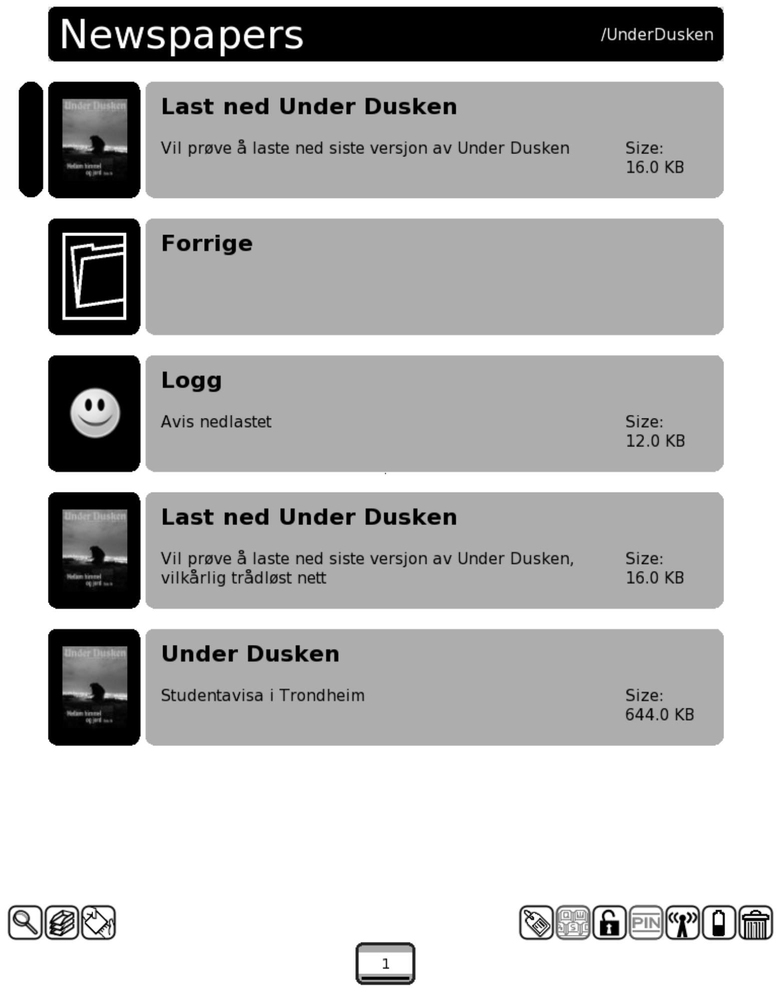
The user interface uses the content lister and is very simple. As can be seen in Figure 7.2 there is one button for downloading specificly via Wireless Trondheim network, one for downloading via any other network being available. There is further one button for viewing the e-newspaper itself. The button with a smileyface on it is the log; depending on the latest transfer tried it will show an appropriate smileyface. Lastly there is a “Forrige” button for the previous e-newspaper.

Performance

The newspapers tried delivered to the iLiad has been from about 200 Kbyte to 800 Kbyte, and the actual download of each file is comparable to what other WiFi devices can manage. The most timeconsuming part is the initialising and establishment of connection in the start. The upper limit of download is the 500 kbit/s that the connection provides, this because the iLiad is connected to the SMS pay category they provide, as described in 7.2.1.

7.3 Difficulties

Working with a task such as this is more or less bound to run into some difficulties and challenges. Here a few presented that either was not mentioned above or not mentioned thoroughly.


Newspapers /UnderDusken

Last ned Under Dusken
Vil prøve å laste ned siste versjon av Under Dusken Size: 16.0 KB

Forrige

Logg
Avis nedlastet Size: 12.0 KB

Last ned Under Dusken
Vil prøve å laste ned siste versjon av Under Dusken, vilkårlig trådløst nett Size: 16.0 KB

Under Dusken
Studentavisa i Trondheim Size: 644.0 KB

Figure 7.2: Screenshot from the iLiad showing the user interface. There is one button for downloading, viewing the e-newspaper and viewing the log.

7.4 The iLiad

The iLiad is a bit quirky and can lock itself and require several resets and reboots before it is usable again if one is not careful with what one tries to run on it. It is a bit lacking in some departments, the battery especially.

7.4.1 Reset and reflashing

If the iLiad somehow crashes and gets unresponsive and generally non-working, the iLiad can be reset and then rebooted. The reset will turn the device off, and let it reboot ordinary. This usually solves most crashes and the like, but a few times when experiencing with "beta" software the unit has needed several reset and reboot cycles to function properly again.

With the latest software update to the iLiad, the version 1.12 update, iRex made a reflash tool available to the public. This required the user to load a CF memory card with this tool, and some special button presses while booting with this card inserted. This would reflash the device to a stock version, factory reset one could say. Luckily this has not been needed, but it was comforting to have this option available should the device suffer from a hard crash or unwise tinkering with sensitive parts of its software; the developer access do give root access to the entire device.

7.4.2 Use of shell and Unix utilities

The only way to use the Unix utilities without writing scripts is to either use a terminal emulator on the device or use `SSH` to connect to it. In the case of the former there has been ported the `mrxvt`. By itself it works good, but coupled with the onscreen keyboard there are some problems that appear. There are particularly two elements that are making it a challenge to use it. First, the combination of slow e-paper and onscreen keyboard makes it likely to write wrong, particularly often in the middle of long commands. Secondly there is no `Ctrl` key, making it impossible to terminate any running processes. The entire program needs to be terminated, making it very inefficient at times.

As was noted above, there are some inconsistenses between what happens when a script is run in the content lister and when it is run via the `mrxvt`. Due to the slow operating of the `mrxvt` there has not been any success in finding what goes wrong.

When there was not satisfactory results from the use of `mrxvt`, it was hoped that using`SSH` would be more fruititious. There exist a ported version of `Dropbear` for the iLiad. But alas, there was no success in connecting to the iLiad from a laptop running Linux with the correct certificates installed.

7.4.3 Battery

During the theoretical preproject to this thesis, an early version 1 iRex iLiad was available for some initial testing. It was most likely from the batch made before they started the initial sales. The battery time of this was not very impressive, and at one point it managed to get in a state that would not even let the battery be charged. Some weeks of quiet rest in a file cabinet in addition to software updates restored its vigour, at least the ability to recharge. But the available battery time was not great at any point. It would easily discharge, especially if the WiFi was enabled.

The battery was improved with the second version of the iLiad, the version 2. The main improvements of this version was a better battery and a new version of the E Ink display. The old battery hava en estimated batterylife of ˜10 hours, while the new got ˜15 hours. This is lower than advertised by iRex. WiFi and stylus use will lower this further.[33]

The reason for mentioning this under "Difficulties" is that a low battery life limits the possible options to a certain degree. It is not that bad, but it puts a damper(sp?) on the dream of a device that is always on and receiving updates as they happen.

The iLiad have more features than the average e-paper device, but pays for it with more limited battery life. It will be interesting to see if the devices of the future with WiFi will manage any better on the battery life aspect.

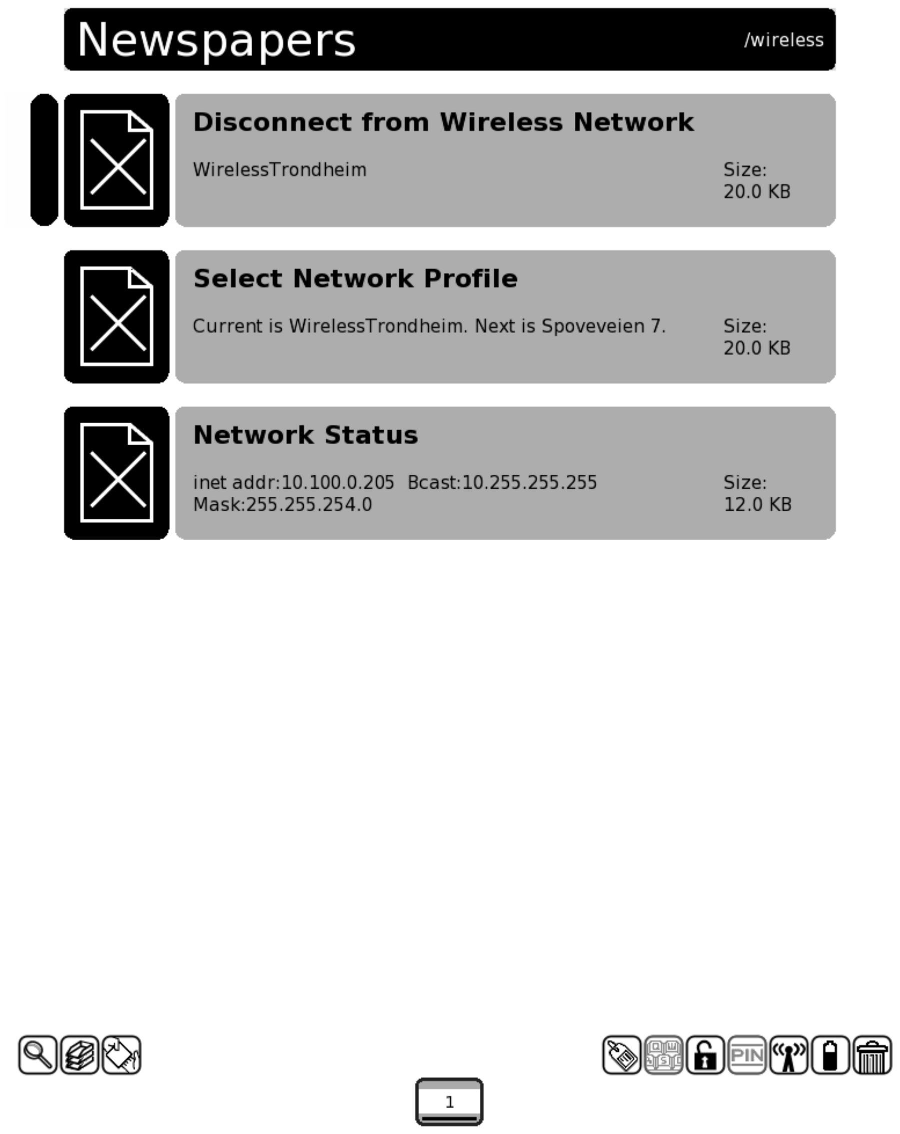
7.4.4 Wireless interface (WiFi)

The interface tools provided for connecting to a (wireless) network is not entirely optimal. For connecting to a wireless (and wired) network it will not get into a connected state if it cannot reach one of iRex servers. The iLiad can connect automatically to any existing unprotected wireless networks, but it is difficult to find the network it is connected to, the signal strength and other alternative wireless networks in the vicinity.

It is also possible to save a number of network profiles for those networks with some kind of protection. Currently WEP and WPA is the possible alternatives, which leaves out WPA2 that Eduroam uses and those security schemes that uses some sort of web login like Wireless Trondheim.

The user interface makes it hard to choose which network to connect to if there are alternatives. The only information it gives is through an icon of a radiotower that can show three states, which are most likely on, off and connected – there is not much information about this.

Luckily the wireless utility that lies at bottom and actually controls the wireless interface is fairly easy to use, and there has also been made a few wireless connectivity scripts by the users, which is fairly straight forward. The one that has been tested most on the iLiad, shows the network profile that has been used to connect with and the IP address that has been obtained. See Figure 7.3 for a screenshot of this from the iLiad.


Newspapers /wireless

Disconnect from Wireless Network
WirelessTrondheim Size: 20.0 KB

Select Network Profile
Current is WirelessTrondheim. Next is Spoveveien 7. Size: 20.0 KB

Network Status
inet addr:10.100.0.205 Bcast:10.255.255.255
Mask:255.255.254.0 Size: 12.0 KB

Figure 7.3: Screenshot of the Wireless script that easily enabled a user to connect to a given wireless network. As also can be seen there is feedback on the IP adresses obtained and such.

Part III

Evaluation

8 Evaluation

The demonstration was tested by a testgroup. This chapter will contain their thoughts about e-paper, e-newspapers and the demonstration that was made. The author will also contribute some thoughts about the demonstration, e-paper and newspapers on e-paper. There is a summary at the end.

8.1 Test group

To test the demo and e-paper, it was decided to try to make the test group as diverse as possible in relation with technical aptness. The test group was recruited from Under Dusken, the student newspaper that provided content to test. The members of Under Dusken consists of students from many fields of study in Trondheim, and the test group was fairly diverse in such respects – not only gadget fanatics. There was four e-paper devices at this thesis disposal, and four testers, A, B, C and D.

8.1.1 Topics and questions

There was made a selection of topics and questions that the test group should try to answer after some time of testing the iLiad and newspaper service loaded on it. These can be found in Table 8.1.

8.2 Results

Following here are a summary of what the test group answered on the questions in Table 8.1. The test was performed in norwegian, and the answers are translated with

Table 8.1: Topics and questions the test group covered

Topic	Question
E-paper	What do you think about e-paper in comparison to: • Ordinary print on paper and? • Computer displays (LCD etc.)? What do you think about the device: • Size and shape? • User Interface? • Use of Wireless (Wireless Trondheim)? • What to improve on the unit? • Does the device have a future? • Colour, Video capable or flexible as a next step? • Other observations
Newspaper Service	Under Dusken: • What do you think about the adaption to e-paper? • Anything you miss from either the paper or online version? General: • What do you think about reading newspapers on such a device? • What is the least amount of functionality you would need in an e-newspaper? • What is most important, tailored or always up to date content? • Would you sacrifice battery for the sake of always having up to date content?
E-paper Service	Wireless: • What kind of services that uses wireless networks would you like? • Other? School • Do you think e-paper is something that could be used in education?
Other	Anything else you would like to mention?

the authors best abilities to english. Each question is followed with a reasoning for that question, and some comments on the answers given.

8.2.1 E-paper compared to

E-paper as a new display technology competes with both traditional paper and the different kinds of computer displays. It was felt necessary to ask how they felt about the technology, was it good enough.

Ordinary paper

Paper is what e-paper primarily competes with, and ultimately can replace.

> The readability is perceived by all as good, one actually preferred this over ordinary paper, while others felt that it is still a bit to go; Tester B meant that while still nice to read on, paper have a better "three dimensional feel". While it is very good to read in bright sunshine it could be plagued by a bit of reflection from various sources compared to ordinary paper. Tester C meant that it could have an even higher resolution. Tester D finds that it is almost better than print on paper when reading in the sun (daylight?), there are still a little bit to go before the contrast and brightness of print on paper are reached.

Compared to ordinary there are still some work that needs to be done, as noted, brightness, resolution and contrast is the main areas. Compared to computer displays where the big focus is on achieving good black levels, the white is what is generally lacking on e-paper.

E Ink, the producer of the display in the iLiad, operates with 160 dots per inch (DPI) as noted in [11]. Comparing to print on paper technology this is not a big number, for example, according to [51] ordinary inkjet printers can be between 300 and 1200 DPI.

Computer displays

Computer displays can be used and performs well for many viewing tasks, but reading at length is not one of them. Ordinary computer displays are also less mobile due to higher requirements to power.

> Tester A felt it was better for reading text, more comfortable. Tester B said that it is a better intermediary of textual information, but when it comes to multimedia information and content it is lacking and "boring". Tester C thought it really lacked in resolution and that the grayscales was not pretty. It should be noted that this tester used the oldest version without the Vizplex technology, which was both less bright and had slower redraw time. He also notet that it was a shame that there sometimes was too visual what had been before the last redraw. Tester D noted that it was less tiring, and there are no problems in daylight. He also noted that the display was slow to update, and that there was a bit of display lag, where a weak trace of the previous text or image remained.

Current computer displays are all of the active kind with refresh rates in tens if not hundreds of times a second. This makes it tiring for the eyes compared to e-paper which only changes when there is a new image to show.

The reason for the noted traces of previous screens lays in the technology used. Each pixel consists, as noted in the theory section for E Ink, of many white and black particles which are differently charged. Sometimes not all pixels or particles moves to their places successfully.

Comparing an e-paper display of one size with a computer display of the same size, the resolution is more or less comparable. As an example, the popular Asus Eee subnotebooks are equipped with an LCD display of 8.9 inches and 1024 x 600 while the iRex iLiads e-paper display is of 8,1 inches and 1024 x 768. The driving technology used for both kinds of technologies are of the same kind.

8.2.2 What do you think about the device

The success of an service for the e-paper relies as much on the actual service as the device it runs on.

Size and shape

Size and shape of devices are important, too big it might be too cumbersome to carry and too small it might be hard to use.

> On this point the testers disagree to a point. Tester A thinks that the size was ok, but it could benefit from being thinner. Tester B thought it was an ok size to use, but too big to easily carry wherever one might go. Traditional newspapers might be bigger, but the benefit of being able to easily discard of it after use is a big plus for the old format. He thought that e-paper currently fits better for e-books, books are not bought on a whim and discarded in the same degree as newspapers are. Tester C agrees that it could be both thinner and lighter to ease the transport. He thought the size was ok, but would have preferred it to be something larger like an A4. Tester D finds the size of the device to be appropriate and the display to be big enough, most irritating was the fact that the pageflipper was on the left side.

Size is an important factor, and there are many opinions on what is the right size. For this kind of device, it is most important that it is easy to carry along – that it is no hassle to bring it with oneselves. As long as most e-paper displays are rigid, the limiting factor is the display. When flexible e-paper displays are more widely introduced, there is easier to make a wide range of display sizes that do not take great space. As one of the e-paper inventors mentioned earlier in 2.4, he envisages an e-paper device of the future to be a fairly small cylinder (1 cm diameter and 15-20 cm long) containing a roll of flexible e-paper.

User interface

Being able to actually use the device in a productive way is of importance.

> It was mostly agreed that most of the user interface was intuitive and understandable to use. Tester A thought the books and newspaper part was

> ok, but that the notetaking part was hard to understand how some parts worked. Tester B noted it was a tad boring and again more fitted for books than newspapers. Tester C thought that even though it was intuitive, it was a bit slow and difficult to know the state of the device. Tester D finds the user interface to be incredibly slow, ok organized but not all that intuitive.

The user interface consists of as explained in 3.1.1 a set of buttons, the stylus and a contest lister, which is what a user will navigate in. It works good for content, but maybe not optimally for integral parts of the operating system and user interface.

To indicate whether the device is processing or not, there is a small light that will flash green when it is working. This is not very legible in the same daylight which the display of the iLiad thrives in.

On the noted slowness, this is part to be blamed on the e-paper display. Even with the new improved Vizplex display of version 2, the redraw time is close to one second in worst case. This adds time to every process the device is performing. The other hardware in the iLiad does not feel all that "snappy" either as many operations takes some time; opening documents, opening settings, turning the device on.

Use of Wireless Trondheim

As one of the e-newspaper service's features is the ability to download newspapers via the wireless network of Wireless Trondheim, it is of interest to see if downloads is actually possible.

> Tester A did not manage to get it connected at all. Tester B did not try it in Wireless Trondheim network. Tester C managed to test, but noted that it was hard to know if there was any network at a location and if it was, its signalstrength. When it worked, it worked ok. Tester Ds also had some problems in achieving connection. When the connection worked, it worked satisfactory. Possible to be mobile (walk) when the device was updating was positive.

The iLiad was initially tested in a close to perfect environment with no problems at all with wireless connectivity. When it was tested in a more natural area in downtown Trondheim, the connectivity was ok when it managed to connect. But it was some occacions when it did not manage to connect properly, and one had to do a few tries before it could connect and properly download something. Note that this was not a thorough test, more a few arbitrary tests throughout downtown Trondheim to see if it actually worked.

The reason for why the device was unable to properly download sometimes is not known, and the device is not very good at showing information about its wireless state. Likely culprits is either the wireless hardware on the iLiad being not suited for the outdoor

[0]1-2 meters away from a Wireless Trondheim accesspoint, with low interference from other accesspoints.

use in a citywide wireless network and the interference that includes. It could also be that the software that actually connects the iLiad to wireless networks is suboptimal. As noted, there seemed to be no great problems when a download was correctly started and in progress, the problem seemed to be the initial connection. But without a more thorough investigation it is just speculation.

What to improve

No device is entirely perfect, and knowing functionality users might want is of interest, especially if one is to make a grander attempt at services with e-paper devices.

> Tester A thought the battery life was good, as she did not need to recharge. She did not manage to use it in wireless mode though. Tester B meant that colours, redraw speed and larger resolution are necessary improvements. Tester C meant that the device should be able to go to a non-power modus without resetting the display and shutting off the device. He also meant that the device was too slow for such a dedicated one purpose device. He also missed both a mail client as well as an internet browser. Tester D notes that better battery, faster display, colour display and a better user interface with possibilities for automatic up and downloads are the way to go.

The mentioned improvements to the display is something that will make it appearence when the technology has matured enough and the demand for e-paper is large enough to alleviate the costs for these advances in research and development.

Concerning the absense of a way to put the iLiad in a low-power or no-power state without turning it off and blanking the display, it could be many reasons for this. One approach could be to hibernate or suspend the device. The Linux kernel supports this kind of hibernation/suspend with either saving state to storage or RAM. The problem is that the Linux kernel used on the iLiad is version 2.4, while the desired functionality was added in 2.6 and later. There do exist software that can achieve suspend on earlier kernels than 2.4, such as TuxOnIce for Linux. [19, 1, 68]

The largest culprit in power consumption is probably the CPU, and other means of saving power is to clock it down to only a fraction of its operating speed, and turning off all other systems. Thus there are possibilities for power saving, which the iLiad would benefit from as the battery life of it is not very great.

Future

Do the testers think this kind of device, and newspaper service have a future.

> Tester A thinks that a device as this has the brightest future as an e-book device, easy to use, bring and comfortable to read on. Tester B is of the opinion that it certainly has a future when the size has reached a more optimal “carry-along” size, as well has achieved better performance and resolution. Tester C thinks such a device needs to be very cheap before he would bother

> with it, and would be suited for commuters if it could pre load the morning newspaper. Tester D believes that there are need for improvements in the technology and a lower price before it is something the public would bother with.

It is only a matter of time and money before the e-paper devices will be as cheap and transportable as the testers wish. But the time and money required for this will potentially only come if there is enough interest in it from the audience. Unfortunately most e-paper devices are quite pricey these days and aimed more for the enthusiast than the common man.

Colour, video or flexible

What kind of device would the testers like to have. Is colour, video or flexible materials important.

> Tester A thinks that a flexible e-paper would be fun, one that could be bended and folded in all directions. Tester B is uncertain, but thinks colour might be most interesting. Tester C do not think video is that important, and colour is more interesting if they also managed to up the resolution. If that does not happen, he thinks flexible e-paper is the most exiting. Tester D would prefer an e-paper device that are both flexible and with colours, he dont think video is necessary for this kind of device.

All these mentioned features are possible, and are being researched on. Electrowetting is a technique that can both enable colours and fast enough redraw times to be usable for videos. The reason for rigid e-paper thus far is that the backplanes that control the drawing on the displays are made on rigid materials due to its production methods, but there are extensive research performed on flexible electronics. One day an e-paper display that either can be folded or rolled and can show both colour and video are available.

Other observations

> Tester A suggests a waterproof e-paper as something that could be useful on travels. Tester C misses automatical zoom to prevent unused whitespace, and bookmarks for speedier navigation.

The wishes of Tester C is problems that better reader software can solve, and might even exist or being developed in some way. If not on the iLiad then for other similar devices.

8.2.3 Newspaper Service: Under Dusken

Adaption to e-paper

How did the testers find the adaption of Under Dusken to the e-paper device.

> Tester A found it nice if one wishes to read it in detail, no distractions in the text or advertisements. Tester B thinks it is not completely successful, more akin to first generation online newspapers. Understandebly enough he adds. Tester C did not find this service particularly good, too many old articles and the division in sections not quite good, which he also adds that Under Dusken is not quite good at either. Tester D finds that it is ok, but thinks that emulation of paper is the way to go, and that it should be closer to the paper version in appearance.

The approach chosen for the test implementation was to use a modified RSS feed to create the e-newspaper version of Under Dusken, which is reflected in what the testers responds. The reason for chosing this approach was that it was easiest to automate and implement. To make an adaption of the paper version for the iLiad would have required much more human interaction than the chosen approach. The PDF version of Under Dusken that they publish on their web site would not provide a good fit, and to provide a better experience for the reader would need to be adapted or even recreated from bottom by a layout artist. An alternative in the middle would be to create several well designed layouts which the text could be slotted into. There was also some content which did not make it to the e-paper version, such as small notices and fact boxes, due to them not existing on the web site and thus not in the system which creates the RSS feed either.

Misses from either paper or online version

What could have been better in adapting the newspaper to e-paper.

> Tester A finds that it misses some of its distinctiveness, it gets a bit too plain or undiverse. She notes that it could maybe fit better for other users. Tester B thinks that the greatest challenge for an e-newspaper is to find the right direction to focus on, either more towards those of online newspapers or that of paper newspapers. If not, the e-paper could end up as an impractical unwieldy/heavy newspaper or a laptop without backlight. Tester C wants better manoeuvrability, and to keep more of the layout from the original paper version. He means that the loss of fact boxes, sub stories and other binding elements makes it more boring. Tester D is of the opinion that the e-paper version should be Under Dusken paper version in a smaller package, downloadable from the net.

There is certainly merit in going for a closer representation of newspapers. When the e-newspaper was initially made, there was more focus on making it easy to use and fill with content. There could certainly have been taken greater care with bringing in more elements of the original newspaper.

8.2.4 Newspaper Service: General

Reading newspapers on e-paper

What do the testers think about reading newspapers on e-paper (after trying it with Under Dusken).

> Tester A feels that a lot of the charm disappears, and she still finds it prefereable to read a paper newspaper. Tester B finds it readable, but feels that the e-paper is still better suited for e-books, and for a newspaper would need more work on layout of articles and the newspaper. Tester C finds it ok, but misses a bit of the "feel" from reading a newspaper, more of a "drab newspaper service". Tester D says that an e-paper device opens the possibilities for many newspapers in a smaller package, but that the display might be a bit small, and with a poorer contrast compared to paper.

Again, there are two different paths that can be chosen for making newspapers for e-paper devices. The "sterile" approach of an e-newspaper which is some sort of aggregated feeds, always up to date and filled with whatever the subscriber wants, but possibly with the tradeoff of lesser quality on some of the content. The other is the more genuine and "charming" newspaper filled with higher quality content throughout and assembled by a skilled layout artist.

Needed functionality

What functionality do the testers find as necessary for an e-newspaper. This could be archiving, internal links, dictionaries and fact sheets.

> Tester A wants better notetaking software and a dictionary. Tester B wants more speed and userfriendlyness. His dream is something that acts and feels more like paper; thinner and foldable. Tester C wants more of a true adaption of the existing newspapers, a bit more than the only text adaption he finds the current Under Dusken adaption is now. Tester D is of the opinion that the most important is a good adaption of the paper version, anything else is of less importance. Archiving of all old editions is assumed.

Getting the basic functionality absolutely right is of great importance, either if one opts for closer to paper or online newspapers.

Tailored or up to date content

If the test group could decide, what would they like: content that is tailored to their tastes or content that is always updated whenever anything happens.

> Tester A values tailored content most, as news can be found most places. She believes the e-paper can work be as a book not requiring to be online at all

> times draining the battery. Tester B also feels that tailored is to prefer, unless e-paper has an ambition of replacing the laptop which he feels is more suited for the up to date content. The well edited and well made content found in weekly and monthly publications should fit well on the e-paper. Tester C would like some kind of tailored RSS feed functionality with a tailored selection of sections and downloaded whenever he wanted it at the device. Tester D thinks that there should both be possibilities for subscriptions with either often but small updates and rarer but larger updates.

Both of the approaches mentioned, tailored and always up to date, are both possible but involves different amount of logic needed, and can of course be combined. For tailored content there needs to be logic for bringing the content the subscriber wishes to the device of the subscriber. This could either be assembled to one piece server side and then delivered or delivered in several smaller pieces. The difficulty in delivering whenever something is updated is related to mobility and whether there are push or pull distribution. If the device cannot be reached, or cannot reach the server the content needs to be saved for next opportunity.

Batterylife or up to date content

Would the test group value battery life over rate of updates. E.g. the iLiad is updated when connected to a computer, whenever the user initiates it or always in the background – with increasing strain on the battery.

> Tester A finds that updating via the PC is enough, and the possibility of buying e-books online and transfer them to the device. Tester B would sacrifice batterylife for having up to date content. Tester C would like to choose this himself, and thinks self initiated wireless updating the best. For Tester D the best would be self initiated wireless updates; e.g. when he presses "check for content" the device does the rest of setting up connections and etc.

The difference in update method and battery life can be quite large. If the iLiad is solely updated via the PC the wireless interface is not used, and will not put additional drain on the battery. On the other side of the spectrum with the iLiad always having WiFi enabled to check for updates the drain on the battery will be significant larger. In the middle, as the testers seemed to prefer, self initiated checks for content with the interface off for the rest of the time. This could of course be initiated by the device with given intervals. As long as the check process is fast, there should not be much "discomfort" for any users.

8.2.5 E-paper service

Other kinds of wireless services

What kind of service for an e-paper device operating in a wireless network would the test group like to have.

> Tester A would like the possibility of watching movies on such a device. Tester B thinks that either the e-paper device needs to either go for the laptop replacement direction, or go more in the book direction with improved readability as a focus. Tester C would like some kind of feed reader, with high customasibility. Overall he feels that a device needs to have a high degree of customasibility so that it can cater to all kinds of users, news on the commute, weather reports when doing sports to books in the park. Tester D finds that newspapers, magazines and locationdependent maps are of interest, but multimedia is of less interest as he feels e-paper is primarily a paper replacement.

Until the e-paper technology speeds up the redraw speed it fits for services which provide fairly static content and service.

Use in education

Could the iLiad and similar e-paper devices have a future in the field of education.

> Tester A think it could fit well as a notebook, with a good notetaking program. But for reading she would still prefer paper books due to they being easier to underline in and getting an overview. Tester B thinks it could have good uses, but had some bad fortune with testing a few PDFs of course materials with the result of fonts blending. Tester C thinks that it takes too long time to navigate in books, and the notataking needs better precision and handwriting to computer letters functionality. Tester D would prefer an e-paper device over books if there is an impovement in user interface and speed, while higher resolution and response could promote it as a note taking device.

Reading and noting is a large part of what happens in education, and a device that can provide this can alleviate many a sore student back.

8.2.6 Other

Tester B had some other observations as well:

- To little friction between pen & paper.
- Missed tooltips in the interface
- Bad Mac support.
- Uncomfortable on/off switch
- Bad font support, at least for those documents tried.
- More comfortable when it is turned 90 degrees, interface symbols should follow though.

Tester D has some remarks on number of updates versus quality. Using online newspapers feeds as a source can provide speedy and many updates, but at the cost of being ugly and not thorougly prepared. He thinks that news for the e-paper format would be better in a newspaper format with the layout and structure this would bring. Fewer, as in once or twice a day, updates should not be a great problem.

8.3 Summary

There are a few main topics which was repeated a few times in the evaluation:

- A layout towards the traditional newspaper is most favourable.
- The resolution, redraw time and to some degree overall performance is below expected.
- User initiated download is good.
- It can be of importance to follow through with one direction, frequent and fresh content or less frequent and more processed content.
- Size of the iLiad is not optimal, it could be more portable. The test group is anticipating flexible and colour e-paper devices.

9 Discussion

In this section the problems at hand will be discussed with focus on the possible solutions and the chosen solutions. As can be recalled from the introduction (1) the main goal of this thesis is to prepare and perform a demonstration of newspapers on e-paper in a wireless environment such as Wireless Trondheims.

9.1 Experiences taken from the fall project

This thesis is based on work done in a fall project about e-newspaper, done by the same author at Institute for Telematics (ITEM). That project was mainly theoretical and was aimed at the

- Hardware. About e-paper displays, e-paper devices and the properties of these. What kind of applications would these be suited for.
- E-newspapers. What would be needed to bring newspapers to the e-paper format; layout options, content and additional features.
- Content delivery. What are the different options for content delivery to such e-paper devices. What can be done to improve and make this delivery and update process better.
- Overall e-newspaper service. How could such a service be built. What would it contain and what deciscions would need to be made.

The work done in the fall project has helped speed up some of the parts of the work done in this thesis. Mainly those parts concerned with e-paper and e-paper devices, as well as some of the aspects of an e-newspaper service.

Concerning e-paper and e-paper displays, there was fairly easy at the start of the thesis work to know what kind of devices was available, might soon be available and which devices that would provide the desired functionality. In the digital world the development and turnover happens at a much faster speed than anywhere else. This made it necessary to survey the e-paper market for the status past christmas. Some of the devices that was slated for early 2008 launch did not make it, and there was as well a few new contenders but not in time. The iLiad was the device that was best suited, and was also the device the author had available during the work on the fall project.

With the e-newspaper service there was several aspects of the work done at the fall project which was a great help. This includes the different kinds of layout that are feasible, previous attempts at e-newspapers, some initial thoughts on distribution and one possible way to create such a service.

9.2 The e-paper device

The e-paper device is the most crucial part of the demo, and the most crucial feature is WiFi connectivity; it should be able to connect to Wireless Trondheim's wireless networks. Unfortunately most of the e-paper devices available fails on this first criteria, and leaves the iRex iLiad as the *winner* on a mere walkover. That said, the iLiad is not a bad piece of equipment and would have been a strong contender had there been other units equipped with WiFi.

9.2.1 iRex iLiad

The iLiad e-paper reader by iRex Technologies is currently the only reader available with WiFi which as mentioned directly qualified it for the demonstration. The wireless is not without its problems though, there was experienced both shorter battery life and some wireless connection problems.

Battery

WiFi is a great benefit for the iLiad but also indirectly contributes to the the biggest weaknesses of the iLiad, as it places an additional stress on the existing battery. The battery can be considered as the greatest shortcomings of the iLiad; the battery life is very limited compared to the low power usage of e-paper displays. A low battery life is limiting on the use of devices like the iLiad which clearly is suited for mobile usage.

Wireless problems

Continuing with the wireless interface, there was some problems with making it work satisfactory all the time in the Wireless Trondheim (WT) network. It was tested briefly by the author as well as some of the test group. Unfortunately not all of the test group understood or missed the importance of also testing the iLiad in WT. When it was used in the WT network it occascionally would not download or connect.

From the authors part the testing was not very thorough, it was mostly bringing the iLiad on a few strolls through the coverage area, to test if either the demonstration worked, or if it managed to connect to or detect the wireless network. On the most thorough of these tests a laptop equipped with WiFi functionality was brought, and while the laptop always managed to connect the iLiad was not having an equally high success rate. The exact details of how the test group tested wireless is not known, but probably not any more thorough than the author. It was not asked of them either.

The packaged wireless manager of the iLiad do not indicate well what happens related to the wireless interface. The only feedback to the user is an icon with three states, which tells if it is on, off or connected. But it do not tell what network it might be connected too, signal strength or other alternative networks. This information is possible to view, but it is hidden behind some options. The test group noted the same things as the author, it was working perfectly fine when the wireless worked, but occascionly it would not work.

When the demonstration was made and tested in fairly ideal conditions, there was established a WT access point in the office the author worked in. The only (WiFi) interference was from an NTNU access point in the hallway outside the office. There was no trouble getting the iLiad to work flawlessly in this environment, and there was no wireless trouble. The trouble in the WT coverage downtown Trondheim can probably to some degree be related to interference from other sources operating in the same spectrum; the spectrum used by WiFi is by some nicknamed the garbage spectrum as it is an open spectrum, and all kinds of devices uses this. WiFi access points, microwave ovens, garage door-openers and so forth uses or operates in this spectrum. But there are to be noted that the thesis is not primarily about the suitedness of iRex iLiad in the WT network.

User experience

Disregarding the subpar battery and the slight problems with use of wireless in areas with heavier interference, the last problem that both the author and the test group noted with the device was that its general performance was somewhat poor on some areas. Some of it might be related with the e-paper; even with the improved E Ink Vizplex display the maximum redraw speed could be as high as 720 ms. Close to a second before a display have drawed feels slow compared to what most people are used to, even on mobile phones.

The iLiad is equipped with an 400 MHz, which is comparable to many smartphones [68]. Most of these phones do have a user experience that is rarely obstructed by lag of different kinds. This at least indicates that on similar tasks the iLiad should be comparable ignoring the screen lag. What the iLiad might be a bit slow on is the opening of books and e-newspapers of a large number of pages. But this is also tasks that smart phones rarely are used for. Of course some slowness might be attributed to poor programming, but the screen lag of the e-paper masks most of the lesser performance problems if there are any.

Compatibility

But there are not entirely negative elements associted with this device, it being based on Linux, and the developers opening most made it easy to make things work on the iLiad. This also brought various open source enthusiast to the iLiad, and the MobileRead forums[1] is full of many useful contributions which also made some of the work easier. Unfortunately not all of these gold nuggets was found or available from the start of the thesis work. To mention some of the tools that made it easier; the `Wireless` scripts which made it easier to connect the iLiad to wireless networks and get more information while connected, the `mrxvt` port which enabled "under the hood" access to the iLiad and the `Minimo` web browser which showed what was possible despite it clearly being beta grade software.

9.2.2 Reader software

Most e-paper devices available these days are on an overall note intended first and foremost as e-book devices, and all reader software the author has encountered are overall less suited for newspapers. They are mostly intended for end to end reading and not the more sporadic or random reading that occurs when many read their newspapers. What is missing is in general tools for easy access to different sections, and what each section consists of. As an example the PDF software on the iLiad do only consists of an page indicator, nothing else to indicate where a reader are or what the it might consist of. Elements that could be of interest is something to indicate what the article is named, which section. Further it would be of interest to be able to bring up a menu to show the available sections and articles. This can to some degree be incorporated into the files itselves, but a standard way of doing it would make it easier to include, and if it was at the content creators leisure it might be dropped.

If the newspaper wants to go for a more "natural" newspaper look, there should also be possibilities for this. The manifest solution for linking overview images of the original newspaper to the full content is interesting, but needs some work. There should be possibilities for flipping through several files linked by an manifest file, as was unfortunately encountered when creating an e-newspaper with HTML and manifest files.

9.3 E-newspaper service

To devise the e-newspaper service sketched in 6.3 several choices had to be made. The goal was a solution that could have been implemented with whatever devices and hardware that are or will be available within a year. The devices that are available within a year of this thesis is mostly similar to what are available now, but only with slightly larger displays and WiFi capability on most of them. Colour and/or flexible displays is probably some distance into the future; at the moment of this writing there exist one flexible and one colour e-paper devices, both of them out of reach for the purpose of

[1] `http://mobileread.com`

e-newspaper distributed over WiFi. The colour one, FLEPia, is currently being tested at different japanese technology companies and no news of it since the first press release in 2007, while the second, the Polymer Vision Readius is only equipped with mobile telephony wireless technologies (e.g. UMTS, GPRS).

For an e-newspaper service, it is probably the layout and the distribution which is the most important factors, which also can give a widely different result depending on what is chosen.

9.3.1 Layout

As earlier mentioned there are two main directions for the layout; one going more towards the online newspapers with one static layout and content automatically put in place, while the other approach is to go more towards the traditional paper newspaper with content put in place by a skilled layout artist.

There are pro's and con's for both of these approaches. For the demonstration made in this thesis it was decided to go towards the online layout of a few reasons. The reasons was that it was easier for Under Dusken to provide content; they did not need to make a custom version for the iLiad. It was easier to continually update the e-newspaper with new content, as the content is published. The e-enewspaper service could also automatically update and upload the current e-newspaper(s).

The disadvantage of that approach is that the e-newspaper gets more sterile and loses some of its "charm", as pointed out by the test group. The test group felt that an e-newspaper with a more newspaper look would not suffer from fewer updates, which is also what such a solution would get. The layout made was on purpose simple as the author is not very talented on that area, and it was felt that simple and not blatant wrong was the better approach.

The topic of optimal layout for e-newspapers is not an easy one, and the right answer depends on many factors, and what might feel right for one person might not feel good for another person. When e-paper devices becomes widespread in use and there are much more content available the content providers will most likely follow different paths for representing it. Publishers which aim at high quality content which is processed and polished through and through, but with a low frequency of publications will have much benefit from custom made e-paper versions. News agencies who lives and breeds on bringing the newest news to the audiences as fast as possible will have more use of automatical layout.

The challenge in making a good e-newspaper service will be to cater for all kinds of content in all kinds of forms. This would mean that the service needs to be able to get the content to the subscribers independently of the content either being fully prepared by the content provider or just slotted into a premade news compilation layout.

9.3.2 Distribution

Again two different main approaches; push or pull. For the demonstration pull was chosen first and foremost for simplicity and battery life. The iLiad needs all the battery it can muster and squandering battery away on idle listening on the wireless interface is not the way to go. Under Dusken which provided the content do not update that often; which would mean many hours with active wireless for no good reason. As pointed out by the test group, as long as the newspaper could wirelessly be updated when the subscriber wants to read, there is no great problem in user initiated updating. The only thing the subscriber needs to do is give the update button a quick tap, and in the case of the demonstration an eventual update would be ready in 10-30 seconds. The main disadvantage of pull distributing, congestion, is of no great consequence for a demonstration of this small scale. It is when there are thousands if not tens of thousands that poll the server that this is becoming a problem and push distribution or at least push notification is a valid alternative.

For the user the advantage of push distribution of content is that the content is there without the user needing to take action. The user would only need to give its device a quick look, and that would be enough for the user to know if there is anything to bother with on the device. For the server side it is as mentioned the lack of congestion problems when a massive amount of devices wants to check if the server has any news that is the greatest advantage. Server side there is even possible to control its own load when either sending content or notifications. Push distribution seems to be the trend in other fields of content distribution; push email is one example that are being more used. It is probably a question of time before push distribution is the only valid alternative. It is a key component in pervasive computing and the always online state which is touted as the future.

9.4 Demonstration

A fair part of the thesis was centered on the work with the demonstration. The purpose of this demonstration was to make a working e-newspaper that could be tested by a test group. The experiences harvested from the process of making the demonstration and the end result can be valuable in determining what is needed for making a fully fledged e-newspaper service.

Making a fully fledged e-newspaper service was never intended as it would require too large a set of functionality to be managable by one person, and having all that functionality work together would be a daunting job at best. A subset of this was chosen that would be most interesting given the e-paper medium and the task being given by Wireless Trondheim; make an e-newspaper and distribute it in some way wirelessly. There are many other topics that could equally be of interest with the given topic of e-newspapers. It is almost given that an e-newspaper service needs some kind of subscription possibilities, and coupling this with a number of publishers of content; there are publish subscribe systems to consider. And how would one perform billing of such things and actually make money on distributing e-newspapers.

As mentioned above, there was also choices to be made for both the layout of the e-newspaper as well as the distribution technique. The choices made was probably not optimal, but nonetheless a starting point for making an e-newspaper.

9.5 Evaluation

The last part of the thesis work was to evaluate the e-newspaper that was made, and the distribution for it. As described in 8.2 the test group consisted of people from Under Dusken with various backgrounds. They did also had a bit more experience and opinions of newspapers than the common man, enabling them to also come with reasonable feedback.

One of the more important pieces of knowledge obtained from the evaluation process, was that the newspaper layout and newspaper feel is not obsolete yet, and can definitely have a place in an e-newspaper service. While there was a longing for an "authentic" newspaper for the e-paper device, there was also interest in the direction that the thesis had chosen. There is certainly also belief in a future with "quick & dirty" news easily available on the iLiad.

That the update process is not 100% automatic did not seem to be of such importance, and being able to choose themselves when to download or update is interesting.

It is also necessary to note the fact that all of the test group was "newspaper people", and may not be a complete representative for what the common man or woman would like in an e-newspaper. But this was also on purpose as there was not going to be available a large amount of e-paper devices for a diverse testing group. With a background of making voluntary newspapers they had necessary newspaper know-how, but still they could relate to the common man.

9.6 Comparison with other attempts at e-newspapers

This thesis is not the first attempt at making a newspaper for e-paper devices. As far as the author has observed there has been mostly tried in Europe with the iLiad. Most of these are test projects that are scheduled to run for a limited time. The project in closest vicinity to Trondheim was done by Sundsvalls Tidning in 2006 in Sweden. They are planning to have another test sometime in 2008, and with possibly a commercial launch in 2009/10 [7]. In the rest of Europe, it is the Flemish newspaper De Tijd, the Dutch NRC Handelsblad and the French Les Echos and Le Monde.

De Tijd had a test with 200 subscribers in 2006, and they decided to not pursuit e-newspapers for the time being. [7].

Les Echo is touted as the first commercial contender to try their luck on e-newspapers. They have a subscription that covers two editions. These are updated every hour (0700 – 2100), and can be downloaded Wirelessly. As they cooperates with iRex and uses the iLiad, the update process is likely powered by iRex' iDS content delivery service. The newspaper is made as a PDF file, with index and overview pages at the beginning, and articles thereafter. Each article seems to have an illustration; headliner, introduction, illustration and then the body text seems to be the layout. The body text is of one single column.[16]

The NRC Handelsblad seems to have a similar cooperation with iRex. Wireless distribution of an e-newspaper edition that contains all elements of the paper edition.

The newspaper is distributed as a PDF file, with pretty much the same overall layout with index pages in the beginning and the articles following thereafter. The newspaper uses three columns of text, and illustrations are the width of one column.[25]

It is not known much about the Le Monde offer, but it is one of several newspapers in a test project by the French telecom operator Orange. This project uses the iLiad and a 3G card for distribution of the selected newspapers. It is not know much about the other newspapers either, besides that one of them is Les Echos. [27]

9.7 Further Work and Future of e-newspapers

There are many interesting aspects of e-newspapers and services for delivering them. Based on the experienes gathered from working with it and test group feedback, there are several subjects that certainly would benefit from more thorough research.

Layout for e-newspapers are a subject that there are still no conclusive answers to, and which would certainly be an interesting area to delve deeper into. This could also be coupled with a study in efficient ways for updating e-newspapers resident on a users device.

Other areas could include distribution. This thesis was done for Wireless Trondheim, and the focus was on WiFi. There are other wireless technologies out there, and in the imminent future; 3G and WiMax to mention some. A service such as e-newspapers does not need the same abundant resources as streaming audio and video, and "lesser" technologies might be of interest.

The e-newspaper service explained in this thesis consists of different parts that could easily be split into several smaller services. It would thus be interesting to take a look into service oriented architecture and web services to decide if that could be a good approach for a real world implementation of an e-newspaper service.

The e-paper has potential to replace most of the displays used today, atleast for mobile use. As the technology matures there will be available more features such as colour and video. It would be interesting to further look into how these could complement and contribute to e-newspapers as a medium.

Lastly it would be of interest to make a new or further improve the demonstration of an e-newspaper service. Within a year there are probably more available devices, and iRex might have opened their delivery network, iDS.

10 Conclusion

This masters thesis has explored the requirements and needs for an e-newspaper service and from this devised a demonstration of such an e-newspaper. This demonstration was tested and evaluated by a test group.

10.1 E-paper and devices

The thesis work started with surveying the available e-paper technology and devices using these. There was no great changes compared to what was found in the fall project on e-newspapers. There are still a majority of black and white displays available and it looks like it will be like this for yet some time.

The most important feature of an e-paper device for the sake of this thesis is the possibility of connecting to wireless networks, WiFi. There are still only one of these available, the iRex iLiad. There will *soon* be introduced more WiFi capable devices by different manufactorers, which hopefully will reduce the price and increase the innovation for what such devices can provide. Current e-paper devices, WiFi or no, are selling at a price level that few people beside enthusiasts will accept.

Besides the price, there are some areas that can increase the e-paper appeal. Colour capability, flexible displays or devices and video capability represents some of the work being done on e-paper. Colour is possible today, but the largest problem is too low resolution with the common method of RGB colour filters due to the subpixelation. Flexible displays will greatly reduce the physical size of the devices while increasing the possible display size. There are being done much work in the field of flexible electronics and there are already one commerciable available device, the Polymer Vision Readius. To enable video redraw speeds, the electrowetting and similar techniques are promising and hopefully soon mature enough for commercial application.

10.2 E-newspaper service

An e-newspaper service is a service that can provide a subscriber or user with a newspaper dedicated to an e-paper device. With content from the paper and online newspapers and the advantages of a portable digital medium. In this thesis a proposal for such a service was described aswell as a demonstration made.

On overall such a service was envisaged as middleware in the network, as an intermediary between content publishers and content subscribers. Content in some form are sent to the service, which processes it and matces it with the right subscribers.

10.2.1 Layout

There are many possibilities for how an e-newspaper are made, distributed and consumed. As was pointed out in the initial phases of the implementation and as well by the test group, there are mainly two directions for layout; towards the old paper version or towards the newer online versions. Each with their benefits and disadvantages; a paper approach keeps the charm, professionalism and quality but as an approach has the tradeoff of more work per e-newspaper and possibly less streamlined navigation. It will be more suited for those who specialise in weekly or monthly publications. The online "opponent" tend to become more sterile and with an aim for the quick and easy news might lose some of the craftmanship and quality of the paper approach. This is suited for those publishers that aim for pushing news around the clock.

For the demonstration it was chosen an approach closer to the online with an index page and the possibility to flip through the entire issue; it was more suited for frequent and automatic updates.

10.2.2 Distribution

Distribution is another area with two different possibilities; pull or push. Pull distribution requires the distribution to be initiated from the device, while push is initiated from the network. On the device pull distribution will potentially waste less power as the wireless interface do not need to be active all the time. The disadvantage of this is that the server will suffer more traffic and calls from devices wishing to either download or check for new content. The congestion problem can be alleviated some by increasing the user interactivity requirements; having the user initiate a check and download each time she wishes to read the newspaper will generate less traffic than a device regularly polling the server. This will also be more power saving than having a device polling the server often, i.e. every minute.

Push distribution on the other hand will draw more power on the device because of the need for always being connected to receive any notifications or content from the server. There will be less traffic in the network as only the necessary traffic needs to be transmitted. This approach do make it a bit complex for the network as mobility needs to be taken care of. It should know where to push content (notifications) when the device is online, queue content when the device is offline and keep content the device misses due to disconnects.

For the demonstration it was chosen a pull type distribution due to it being simpler mobility wise and it would be less straining on the iLiad's battery.

10.2.3 Evaluation

The results gathered from the test group can be of great use when working on e-newspapers in the future, and certain topics do certainly need some investigation. Topics that the test group had most opinions on was layout of an e-newspaper – the layout of traditional newspapers are not dead. Different distribution techniques was also interesting, and they did not see the immediate need for always on and always updated. The test group also felt that the e-paper had lots of potential that could do with fixing such as e-paper display resolution and redraw speed.

10.3 Future work and Future of e-paper

There are many exiting aspects for further work on the field of e-newspapers. As mentioned there could be done more work in the direction of distribution; WiFi, mobilephone technology(3G), Ad Hoc networks etc. There would also be of use to explore push or pull distribution and the different needs for user interaction and frequency of updates.

There would also be interesting with further explorations of layout for e-newspapers; can a paper or online direction thrive together or would there be need for one joint way. When the redraw speed of e-papers are getting close to current computer displays there is also the question of whether there are needs for dedicated e-newspapers.

Service Oriented Architectures (SOA) as an approach for making e-newspapers are interesting, and several aspects of the e-newspaper creation and distribution could be suited for a SOA.

Bibliography

[1] Userspace software suspend. http://suspend.sourceforge.net/intro.shtml, May 2008.

[2] Amazon.com. Frequently asked questions about amazon kindle. http://www.amazon.com/gp/help/customer/display.html?nodeId=200127480&, May 2008. Year/month is accessed date.

[3] Steinar H. Andresen, John Krogstie, and Thomas Jelle. Lab and research activities at wireless trondheim. In *Proceedings of ISWCS 2007*. IEEE ISWCS, Oct 2007.

[4] Jacques Angelï¿½ and Thierry Emeraud. Binem electronic paper. *Gekkan Display - Techno Times Japan*, pages 1–8, Oct 2006.

[5] Opera Software ASA. Opera mini features. http://www.operamini.com/features/, June 2008.

[6] Polymers & Liquid Crystals at Case Western Reserve University. Polymer stabilized cholesteric liquid crystals. http://plc.cwru.edu/tutorial/enhanced/files/pslc/apps/apps.htm, June 2008.

[7] Hï¿½kon Rï¿½rvik Aune. Electronic newspaper over wifi, Dec 2007.

[8] Adam Boeglin. Updating busybox. http://www.mobileread.com/forums/showthread.php?p=83469#post83469, Jul 2007.

[9] Xerox PARC (Palo Alto Research Center). Electronic reusable paper. http://www2.parc.com/hsl/projects/gyricon/, Sep 2007.

[10] E Ink Corporation. E ink technology. http://www.eink.com/technology/howitworks.html, Sep 2007.

[11] E Ink Corporation. High resolution displays. http://www.eink.com/products/matrix/High_Res.html, May 2008.

[12] E Ink Corporation. Vizplex$^{\text{TM}}$imaging film. http://www.eink.com/products/matrix/imaging_film.html, May 2008.

[13] Django. Django - the web framework for perfectionists with deadlines. http://www.djangoproject.com/, Apr 2008.

[14] Django. Django faq. `http://www.djangoproject.com/documentation/faq/`, May 2008.

[15] Under Dusken. Om oss (about us). `http://www.underdusken.no/omoss`, Apr 2008.

[16] Les Echos and iRex Technologies. Les echos first french electronic newspaper edition daily on irex iliad. `http://www.irextechnologies.com/files/20070907%20Press%20release%20LesEchos.pdf`, Sep 2007.

[17] Johan Feenstra and Rob Hayes. Electrowetting displays. `http://www.liquavista.com/documents/default.asp?CatID=6`, Jan 2006. (Whitepaper for Liquavista, Principles of Electrowetting).

[18] Kelly Fiveash. Tao group throws in the towel. `http://www.theregister.co.uk/2007/06/13/tao_group_administration/`, June 2007.

[19] TuxOnIce for Linux. Software suspend - features. `http://www.tuxonice.net/features`, May 2008.

[20] Python Software Foundation. About python. `http://python.org/about/`, Apr 2008.

[21] Python Software Foundation. What is python? executive summary. `http://www.python.org/doc/essays/blurb.html`, Apr 2008.

[22] X.Org Foundation. X.org wiki. `http://www.x.org/wiki/`, May 2008.

[23] Iddo Genuth. The future of electronic paper. `http://www.tfot.info/articles/1000/the-future-of-electronic-paper.html`, Oct 2007.

[24] NRC Handelblad. epaper. `http://epaper.nrc.nl/`, June 2008.

[25] NRC Handelsblad and iRex Technologies. Nrc handelsblad, first dutch newspaper on electronic paper. `http://www.irextechnologies.com/files/Press%20Release%20NRC%2003062008_english.pdf`, Mar 2008.

[26] R.A. Hayes and B.J. Feenstra. Video-speed electronic paper based on electrowetting. *Nature*, 425:383–385, Sep 2003.

[27] Ulf Petter HellstrÃ¸m. E-avisene kommer. `http://www.aftenposten.no/forbruker/digital/nyheter/internett/article2463042.ece`, June 2008.

[28] Adrian Holovaty and Jacob Kaplan-Moss. *The Definitive Guide to Django: Web Development Done Right*. Apress, 2007.

[29] Yongqiang Huang and Hector Garcia-Molina. Publish/subscribe in a mobile environment. *Wireless Networks*, 10(6):643–652, 2004.

[30] H. Edzer A. Huitema, Gerwin H. Gelinck, Erik van Veenendaal, Fred J. Touwslager, and Pieter J. G. van Lieshout. Roll-up active-matrix displays. In Hagen Klauk, editor, *Organic Electronics*, chapter 14. Wiley-VCH, 2006.

[31] Netronix Inc. Eb-300. `http://www.netronixinc.com/product_EB300.htm`, May 2008.

[32] iRex Technologies. *How to make content for your iLiad.* iRex Technologies BV, High Tech Campus 9, NL-5656 AE Eindhoven, 1.2 edition, Nov 2006.

[33] iRex Technologies. Battery life. `http://forum.irexnet.com/viewtopic.php?t=1022`, Mar 2007.

[34] iRex Technologies. Business solutions - digital publishing. `http://www.irextechnologies.com/business/solutions`, June 2008.

[35] iRex Technologies. Developer news. `http://developer.irexnet.com/news`, May 2008.

[36] iRex Technologies. Technical specifications, the iliad. `http://www.irextechnologies.com/products/specs`, May 2008.

[37] Hayat Kara and Christopher Edwards. A caching architecture for content delivery to mobile devices. In *Proceedings of the 29th EUROMICRO Conference 'New Waves in System Architecture'.* IEEE Computer Society, 2003.

[38] Andreas Komninos and Mark D. Dunlop. A calendar based internet content pre-caching agent for small computing devices. *Personal Ubiquitous Computing*, 2007.

[39] Steven Levy. Cover story: Technology - the future of reading. `http://www.newsweek.com/id/70983/page/1`, 2007.

[40] Fujitsu Frontech Limited. Fujitsu frontech starts limited sales of portable information terminal 'flepia'. `http://www.frontech.fujitsu.com/en/release/20070420.html`, Apr 2007.

[41] Mobipocket.com. Mobipocket developer center - creating content. `http://www.mobipocket.com/dev/article.asp?BaseFolder=prcgen`, May 2008.

[42] Hrvoje Niksic. *Wget - The non-interactive network downloader.* Free Software Foundation, Inc., Jun 2007.

[43] Michael J. O'Grady and Gregory M. P. O'Hare. Just-in-time multimedia distribution in a mobile computing environment. *IEEE MultiMedia*, pages 62–74, Oct-Dec 2004.

[44] Plastic Logic Limited. Plastic logic - process. `http://www.plasticlogic.com/process.php`, Jun 2008.

[45] Ivana Podnar, Manfred Hauswirth, and Mehdi Jazayeri. Mobile push: Delivering content to mobile users. In *Proceedings of the 22nd International Conference on Distributed Computing Systems Workshop*. IEEE Computer Society, 2002.

[46] Python Library Reference. *xml.dom – The Document Object Model API*. Python Software Foundation.

[47] ReportLab Inc. *ReportLab PDF Generation User Guide*, 2.1 edition.

[48] Leonard Richardson. *Beautiful Soup*. Crummy, `http://www.crummy.com/software/BeautifulSoup/`, June 2008.

[49] Secret Labs AB, `http://www.pythonware.com/library/pil/handbook/index.htm`. *Python Imaging Library Handbook*.

[50] Henning Sirringhaus, Christoph W. Sele, Timothy von Werne, and Catherine Ramsdale. Manufacturing of organic transistor circuits by solution-based printing. In Hagen Klauk, editor, *Organic Electronics*, chapter 12. Wiley-VCH, 2006.

[51] Oki Systems. Inkjet printers. `http://www.askoki.co.uk/encyclo/printertech/inkjet.asp`, May 2008.

[52] Sadhna Ahuja Tau Wu and Sudhir Dixit. Efficient mobile content delivery by exploiting user interest correlation. In *IEEE International Conference on Multimedia and Expo*, 2004.

[53] LTD Tianjin Jinke Electronics Co. About jinke. `http://www.jinke.com.cn/Compagesql/English/company/index.asp`, May 2008.

[54] LTD Tianjin Jinke Electronics Co. Hanlin ereader v9c. `http://www.jinke.com.cn/Compagesql/English/embedpro/prodetail.asp?id=35`, May 2008.

[55] LTD Tianjin Jinke Electronics Co. Product plan. `http://www.jinke.com.cn/Compagesql/English/embedpro/newpro.asp`, May 2008.

[56] Wireless Trondheim. How to connect. `http://wirelesstrondheim.no/sec.php?page=sec_connection&la=en`, May 2008.

[57] Polymer Vision. Readius. `http://www.polymervision.com/frameset.php?id=&page=`, May 2008.

[58] Paul Vixie. *crontab - tables for driving cron*, Jan 1994.

[59] Paul Vixie. *cron - daemon to execute scheduled commands (Vixie Cron)*, Oct 2006.

[60] Wacom. EMR®(electro-magnetic resonance) technology. `http://www.wacom-components.com/english/technology/emr.html`, May 2008.

[61] MobileRead Wiki. Iliad software. `http://wiki.mobileread.com/wiki/Iliad_Software`, May 2008.

[62] MobileRead Wiki. Iliad unofficial toolchain. `http://wiki.mobileread.com/wiki/Iliad_unofficial_toolchain`, May 2008.

[63] Wikipedia. Amazon kindle. `http://en.wikipedia.org/wiki/Amazon_Kindle`, May 2008.

[64] Wikipedia. Cybook gen3. `http://en.wikipedia.org/wiki/Cybook_Gen3`, May 2008.

[65] Wikipedia. Electronic paper. `http://en.wikipedia.org/wiki/Electronic_paper`, Apr 2008.

[66] Wikipedia. Electrophoresis. `http://en.wikipedia.org/wiki/Electrophoresis`, May 2008.

[67] Wikipedia. Evolution-data optimized. `http://en.wikipedia.org/wiki/Evdo`, May 2008.

[68] Wikipedia. Hibernate (os feature). `http://en.wikipedia.org/wiki/Hibernate_(OS_feature)`, May 2008.

[69] Wikipedia. iliad. `http://en.wikipedia.org/wiki/ILiad`, May 2008.

[70] Wikipedia. Sony reader. `http://en.wikipedia.org/wiki/Sony_Reader`, May 2008.

[71] Wikipedia. X window system. `http://en.wikipedia.org/wiki/X_Window_System`, May 2008.

[72] Tao Wu, Sadhna Ahuja, and Sudhir Dixit. Acme: A new mobile content delivery architecture. In Sudhir Dixit and Tao Wu, editors, *Content Networking In The Mobile Internet*, chapter 6. Wiley Interscience, 2004.

List of Abbreviations

3GPP	3rd Generation Partnership Project
3GPP2	3rd Generation Partnership Project 2
CDMA	Code Division Multiple Access
CF	Compact Flash
DPI	Dots Per Inch
DRM	Digital Rights Management
EDGE	Enhanced Data rates for GSM Evolution
EVDO	Evolution-Data Optimized
FTP	File Transfer Protocol
GNU	GNU's Not Unix
GPL	GNU Public License
GPRS	General Packet Radio Service
GSM	Global Systems for Mobile communications / Groupe SpÃ©cial Mobile
HSDPA	High-Speed Downlink Packet Access
HTML	HyperText Markup Language
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
HTTPS	HyperText Transfer Protocol over Secure Socket Layer
ISP	Internet Service Provider
JVM	Java Virtual Machine
LCD	Liquid Crystal Display
MB	MegaByte

MHz	MegaHertz
MMC	MultiMediaCard
NOK	Norwegian Kroner
PARC	Palo Alto Reseach Center
PDF	Portable Document Format
RAM	Random Access Memory
RTF	Rich Text Format
SD	Secure Digital card
SSID	Service Set Identifier
TXT	TeXT
UMTS	Universal Mobile Telecommunications System
USB	Universal Serial Bus
WiFi	WLAN products based on the IEEE 802.11 standards, promoted by the Wi-Fi alliance
XHTML	eXtensible HyperText Markup Language
XHTML	eXtensible HyperText Markup Language
XML	eXstensible Markup Language

Appendix

A Python

Python was chosen for the programming on the demonstration for this thesis. Here is a bit more information of Python and Python tools used are presented.

A.1 Python

Python is a high level dynamic programming language introduced to the world in 1991 by Guido van Rossum. It uses a readable non-bloated syntax which makes it easy to read. There is no compiling, speeding up the development process. It can be used for object oriented programming, but also for less extensive scripting. It is supplied with many data structures (lists, dictionaries, tuples, strings etc.) and dynamic typing and binding. It supports packages and modules, and there are Python versions available that can use C, .Net and Java packages. Python is supplied with a comprehensive standard library. [20, 21]

A.2 Beautiful Soup

Beautiful Soup is an open source HTML and XML parser made in Python. It creates a parse tree from the HTML and XML source it is being fed. The strength of Beautiful Soup is that the HTML and XML do not need to have a strict markup, it will attempt to make a parse tree from any HTML and XML-like code. The parse tree can be searched, navigated and modified. Everything is converted to and output in UTF-8 Unicode.

For the thesis demonstration work it helped with fetching content from the Under Dusken website when there was a possibility of better formatting there. [48]

A.3 ReportLab

ReportLab is an open source PDF library written in Python. It can be used to create all sorts of PDFs from all kinds of digital sources. There exist tools for both low level PDF creation such as individual drawing of shapes, but also the higher level tool called Platypus which can easier create complete pages. [47]

A.4 Python Image Library

Pythin Image Library is a library that adds a suite of image capabilities to the Python toolbox. The library supports many different image formats, and among the abilities are thumbnailing, format convertion, image printing, image displaying and image processing. The processing consists of colour conversion, spatial conversion and applying different image filters[49]

B
Busybox Unix utilities

The iRex iLiad uses Busybox to supply it with Unix commands and utilities. It is originally supplied with version 1.01, but one of the members at the MobileRead[1] forums, Adam B., has made an updated version for the iLiad with about twice as many commands and utilities. [8]

B.1 Busybox version 1.01 for the iRex iLiad

```
BusyBox v1.01 (2007.04.10-13:14+0000) multi-call binary

Usage:  busybox [function] [arguments]...
or:  [function] [arguments]...

BusyBox is a multi-call binary that combines many common Unix utilities
into a single executable.  Most people will create a link to busybox
for each function they wish to use and BusyBox will act like whatever
it was invoked as!

Currently defined functions:
[, ash, awk, basename, bunzip2, busybox, bzcat, cat, chgrp, chmod,
chown, chroot, clear, cp, cut, date, dc, dd, df, dirname, dmesg, du,
echo, egrep, env, expr, false, fbset, fdisk, fgrep, find, free, grep,
gunzip, gzip, head, hexdump, hostname, hwclock, id, ifconfig, ifdown,
ifup, insmod, ip, kill, killall, klogd, ln, logger, logname, losetup,
ls, lsmod, md5sum, mkdir, mkfifo, mknod, mktemp, modprobe, more, mount,
mv, nc, netstat, od, ping, pivot_root, printf, ps, pwd, readlink,
renice, reset, rm, rmdir, rmmod, route, run-parts, sed, seq, sh, sleep,
```

[1]http://www.mobileread.com/forums/

```
sort, start-stop-daemon, strings, stty, sync, syslogd, tail, tar,
tee, telnet, test, time, top, touch, tr, traceroute, true, tty, udhcpc,
umount, uname, uniq, unzip, uptime, vi, watch, wc, wget, which, who,
whoami, xargs, yes, zcat
```

B.2 Busybox version 1.72 for the iRex iLiad

```
BusyBox v1.7.2 (2007-10-30 12:45:14 EDT) multi-call binary
Copyright (C) 1998-2006 Erik Andersen, Rob Landley, and others.
Licensed under GPLv2.  See source distribution for full notice.

Usage:  busybox [function] [arguments]...
or:  [function] [arguments]...

BusyBox is a multi-call binary that combines many common Unix utilities
into a single executable.  Most people will create a link to busybox
for each function they wish to use and BusyBox will act like whatever
it was invoked as!

Currently defined functions:
[, [[, addgroup, adduser, adjtimex, ar, arp, arping, ash, awk, basename,
bunzip2, bzcat, cal, cat, catv, chattr, chgrp, chmod, chown, chpasswd,
chpst, chroot, chrt, chvt, cksum, clear, cmp, comm, cp, crond, crontab,
cryptpw, cut, date, dc, dd, deallocvt, delgroup, deluser, df, dhcprelay,
diff, dirname, dmesg, dnsd, dos2unix, dpkg, du, dumpkmap, dumpleases,
echo, ed, egrep, eject, env, envdir, envuidgid, ether-wake, expand,
expr, fakeidentd, false, fbset, fdflush, fdformat, fdisk, fgrep, find,
fold, free, freeramdisk, fsck, fsck.minix, ftpget, ftpput, fuser,
getopt, getty, grep, gunzip, gzip, halt, hdparm, head, hexdump, hostid,
hostname, httpd, hwclock, id, ifconfig, ifdown, ifup, inetd, init,
insmod, install, ip, ipaddr, ipcalc, ipcrm, ipcs, iplink, iproute,
iprule, iptunnel, kill, killall, killall5, klogd, last, length, less,
linux32, linux64, linuxrc, ln, loadfont, loadkmap, logger, login,
logname, logread, losetup, ls, lsattr, lsmod, lzmacat, makedevs, md5sum,
mdev, mesg, mkdir, mkfifo, mkfs.minix, mknod, mkswap, mktemp, modprobe,
more, mount, mountpoint, mt, mv, nameif, nc, netstat, nice, nmeter,
nohup, nslookup, od, openvt, passwd, patch, pidof, ping, ping6, pipe_progress,
pivot_root, poweroff, printenv, printf, ps, pscan, pwd, raidautorun,
rdate, readlink, readprofile, realpath, reboot, renice, reset, resize,
rm, rmdir, rmmod, route, run-parts, runlevel, runsv, runsvdir, rx,
sed, seq, setarch, setconsole, setkeycodes, setlogcons, setsid, setuidgid,
sh, sha1sum, slattach, sleep, softlimit, sort, split, start-stop-daemon,
stat, strings, stty, su, sulogin, sum, sv, svlogd, swapoff, swapon,
switch_root, sync, sysctl, syslogd, tail, tar, tcpsvd, tee, telnet,
telnetd, test, tftp, time, top, touch, tr, traceroute, true, tty,
```

`ttysize, udhcpc, udhcpd, udpsvd, umount, uname, uncompress, unexpand, uniq, unix2dos, unlzma, unzip, uptime, usleep, uudecode, uuencode, vconfig, vi, vlock, watch, watchdog, wc, wget, which, who, whoami, xargs, yes, zcat, zcip`

C
XML

C.1 XML schema for e-newspaper delivery

```
<!ELEMENT enewspaper (article+)>
<!ATTLIST enewspaper issue CDATA #REQUIRED>
<!ATTLIST enewspaper volume CDATA #REQUIRED>
<!ELEMENT article (author, ingress, body, illustration+)>
<!ATTLIST article title CDATA #REQUIRED>
<!ATTLIST article subtitle CDATA #REQUIRED>
<!ATTLIST article category CDATA #REQUIRED>
<!ELEMENT author EMPTY>
<!ATTLIST author text CDATA #REQUIRED>
<!ATTLIST author ill CDATA #REQUIRED>
<!ELEMENT ingress (#PCDATA)>
<!ELEMENT body (#PCDATA)>
<!ELEMENT illustration (#PCDATA)>
<!ATTLIST illustration caption CDATA #REQUIRED>
```

C.2 XML example

```
<?xml version="1.0"?>
<!DOCTYPE enewspaper SYSTEM "enewspaper.dtd">
<enewspaper issue='10' volume='15'>
   <article title="Lipsum"
            subtitle="Lorem ipsum dolor sit amet"
            category="Test">
      <author>
         <text>Ola Nordmann</text>
         <ill>Kari Nordmann</ill>
      </author>
      <ingress>
         Lorem ipsum dolor sit amet, consectetuer adipiscing elit.
         Morbi enim nibh, tristique quis, cursus et, ultricies id, eros.
      </ingress>
      <body>
         Lorem ipsum dolor sit amet, consectetuer adipiscing elit.
         Morbi enim nibh, tristique quis, cursus et, ultricies id, eros.
         Aliquam hendrerit dui vel metus. Praesent eget ligula eu eros
         sodales feugiat. Morbi pulvinar, dolor vitae iaculis laoreet,
         augue purus lacinia nisl, ullamcorper mattis ipsum purus sed dui.
         Aliquam varius tincidunt mi. Donec condimentum. Vestibulum faucibus
         libero a ante mollis facilisis. Cras quam. Morbi tempus risus in erat
         elementum fringilla. Etiam risus lorem, congue ac, tincidunt ac,
         iaculis vel, magna. Sed nec purus.
      </body>
      <illustration caption="Sed nec purus.">
         http://folk.ntnu.no/hakonror/bilder/carusobilder/CarusoLab%20127.jpg
      </illustration>
   </article>
</enewspaper>
```

D
RSS

Here is an exerpt of the RSS feed as it looked the 21st of April 2008. This one contains some of the character escape problems that decided to rear its ugly head, but far from all.

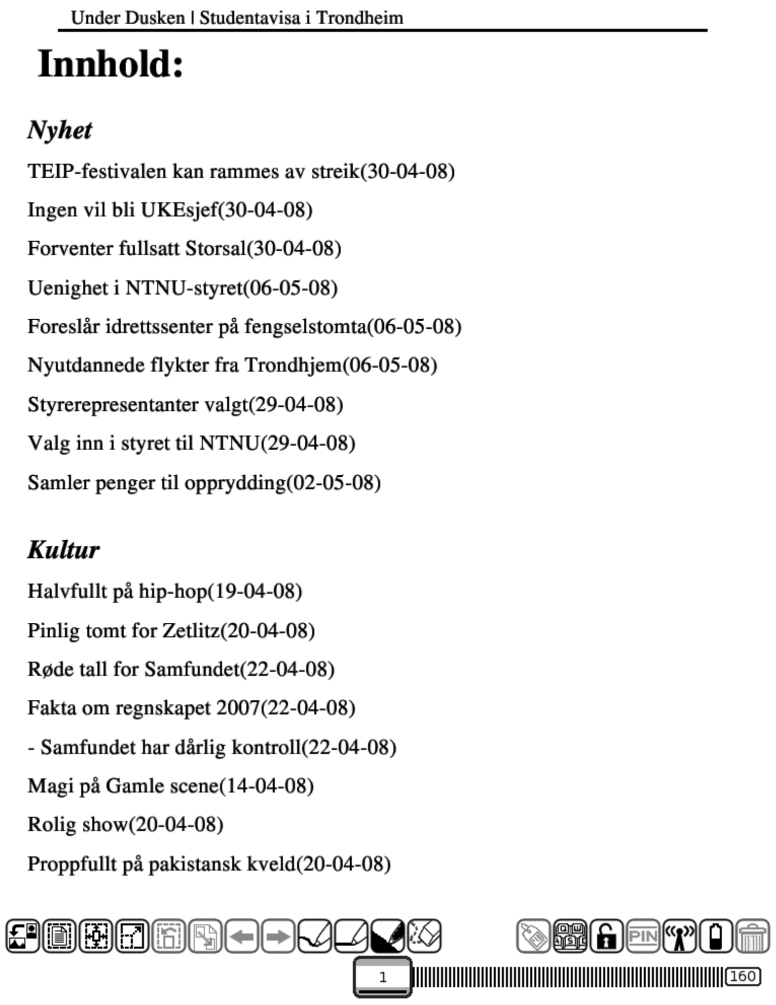
D.1 Article

```
<Article>
   <Title>Maskeradesalg i Rundhallen</Title>
   <Author>David Bach</Author>
   <Published>20:17 15-04-2008</Published>
   <Summary>
      Tirsdag holdt Studentersamfundets Interne Teater (SIT) kostymesalg.
      Kostymesjefen begrunner salget med  plassproblemer.
   </Summary>
   <Text>
      Pï¿½ spï¿½rsmï¿½l om hvorfor SIT valgte ï¿½ ha kostymesalg tirsdag svarer
      kostymesjef Kristine Wallï¿½e slik:? Vi har rett og slett ikke plass
      til dette noe annet sted. Derfor selger vi det.Hun forteller at SITs
      kostymelager fremdeles er stappfullt ? selv etter at tirsdagens
      salgsvarer er hentet ut og plassert pï¿½ bord og stativ rundt om i
      Rundhallen. Kostymesjefens forslag pï¿½ nytt lager er noe originalt.
      &lt;div&gt;? Jeg synes kanskje Strossa burde bli vï¿½rt nye kostymelager.
      Det hadde passet bra for lokalene, mener Wallï¿½e.&lt;/div&gt;
   </Text>
   <Section>Kultur</Section>
   <Imagelink>http://www.underdusken.no/image/1142210</Imagelink>
</Article>
```

E
Screenshots of e-newspaper on the iLiad

E.1 E-newspaper

Following here are a few screenshots showing how the e-newspaper turned out in the end. There are a complete PDF of an issue in the acompanying archive aswell as some of the earlier HTMLs.



Under Dusken | Studentavisa i Trondheim

Innhold:

Nyhet

TEIP-festivalen kan rammes av streik(30-04-08)

Ingen vil bli UKEsjef(30-04-08)

Forventer fullsatt Storsal(30-04-08)

Uenighet i NTNU-styret(06-05-08)

Foreslår idrettssenter på fengselstomta(06-05-08)

Nyutdannede flykter fra Trondhjem(06-05-08)

Styrerepresentanter valgt(29-04-08)

Valg inn i styret til NTNU(29-04-08)

Samler penger til opprydding(02-05-08)

Kultur

Halvfullt på hip-hop(19-04-08)

Pinlig tomt for Zetlitz(20-04-08)

Røde tall for Samfundet(22-04-08)

Fakta om regnskapet 2007(22-04-08)

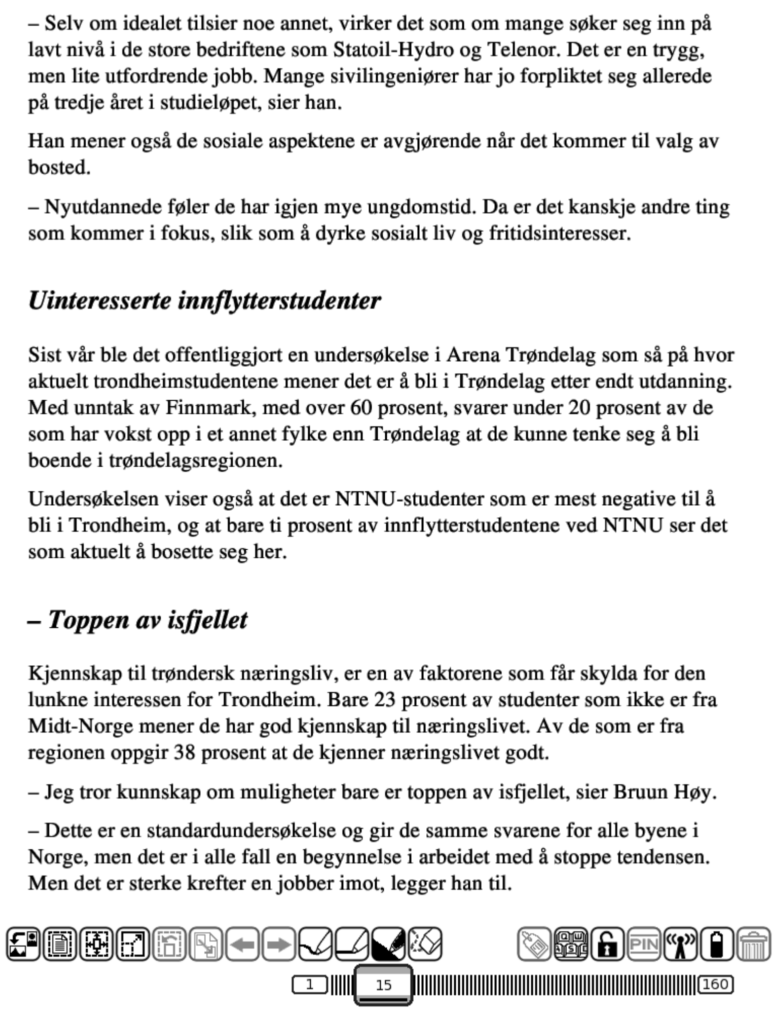
- Samfundet har dårlig kontroll(22-04-08)

Magi på Gamle scene(14-04-08)

Rolig show(20-04-08)

Proppfullt på pakistansk kveld(20-04-08)

Figure E.1: Screenshot from the iLiad showing a page of the index.



– Selv om idealet tilsier noe annet, virker det som om mange søker seg inn på lavt nivå i de store bedriftene som Statoil-Hydro og Telenor. Det er en trygg, men lite utfordrende jobb. Mange sivilingeniører har jo forpliktet seg allerede på tredje året i studieløpet, sier han.

Han mener også de sosiale aspektene er avgjørende når det kommer til valg av bosted.

– Nyutdannede føler de har igjen mye ungdomstid. Da er det kanskje andre ting som kommer i fokus, slik som å dyrke sosialt liv og fritidsinteresser.

Uinteresserte innflytterstudenter

Sist vår ble det offentliggjort en undersøkelse i Arena Trøndelag som så på hvor aktuelt trondheimstudentene mener det er å bli i Trøndelag etter endt utdanning. Med unntak av Finnmark, med over 60 prosent, svarer under 20 prosent av de som har vokst opp i et annet fylke enn Trøndelag at de kunne tenke seg å bli boende i trøndelagsregionen.

Undersøkelsen viser også at det er NTNU-studenter som er mest negative til å bli i Trondheim, og at bare ti prosent av innflytterstudentene ved NTNU ser det som aktuelt å bosette seg her.

– Toppen av isfjellet

Kjennskap til trøndersk næringsliv, er en av faktorene som får skylda for den lunkne interessen for Trondheim. Bare 23 prosent av studenter som ikke er fra Midt-Norge mener de har god kjennskap til næringslivet. Av de som er fra regionen oppgir 38 prosent at de kjenner næringslivet godt.

– Jeg tror kunnskap om muligheter bare er toppen av isfjellet, sier Bruun Høy.

– Dette er en standardundersøkelse og gir de samme svarene for alle byene i Norge, men det er i alle fall en begynnelse i arbeidet med å stoppe tendensen. Men det er sterke krefter en jobber imot, legger han til.

Figure E.2: Screenshot from the iLiad showing an n'th page of an article.

F Scripts used on the iLiad

F.1 Download script

This script is used for the download of content to the iLiad. It requires a developer access to the iLiad. There have been done some changes here for legibility, so it is recommended to use the download.sh file in the accompanying archive file.

```
#!/bin/sh

export WDIR=/mnt/free/newspapers/UnderDusken
export IDIR=/mnt/free/newspapers/UnderDusken/downloadUD
export ADIR=/mnt/free/newspapers/UnderDusken/Forrige
export SDIR=/mnt/free/newspapers/UnderDusken/log

date >> $SDIR/log.txt
echo Wireless trondheim downloader >> $SDIR/log.txt

/usr/bin/wireless.sh start WirelessTrondheim
if [ $? -ne 0 ]; then
   echo Failed to connect to wireless >> $SDIR/log.txt
   cp $IDIR/iconA.png $SDIR/icon.png
   ./manifester.sh "Logg" "Klarte ikke ï¿½ koble seg til Trï¿½dlï¿½se Trondheim"
                                      > $SDIR/manifest.xml
   exit 1
fi

i=ud
```

```
mv $IDIR/contents.txt $IDIR/old_contents.txt
wget -O $IDIR/contents.txt http://folk.ntnu.no/hakonror/iliad/pdf/contents.txt

# if not able to download, exit
if [ $? -ne 0 ]; then
   echo Failed to download contentfile >> $SDIR/log.txt
   cp $IDIR/iconA.png $SDIR/icon.png
   ./manifester.sh "Logg" "Klarte ikke ï¿½ laste ned innholdsfila"
                                       > $SDIR/manifest.xml
   exit 1
fi

loc=$(cat old_contents.txt)
serv=$(cat contents.txt)

echo loc: $loc >> $SDIR/log.txt
echo serv: $serv >> $SDIR/log.txt

if [ $serv -gt $loc ]; then
   rm -rf $ADIR/$i.pdf
   rm /tmp/$i.pdf
   mv $WDIR/$i.pdf $ADIR
   mkdir -p $WDIR/$i.pdf
   ./manifester.sh "Under Dusken" "Studentavisa i Trondheim"
                                       > $WDIR/$i.pdf/manifest.xml

   wget -O /tmp/$i.pdf http://folk.ntnu.no/hakonror/iliad/pdf/$i.pdf
   # if not able to download, exit
   if [ -f /tmp/$i.pdf ]; then
      echo Newspaper downloaded >> $SDIR/log.txt
      cp $IDIR/iconB.png $SDIR/icon.png
      ./manifester.sh "Logg" "Avis nedlastet"
                                       > $SDIR/manifest.xml
      mv /tmp/$i.pdf $WDIR/$i.pdf/$i.pdf
      cp $IDIR/icon.png $WDIR/$i.pdf/icon.png
      /usr/bin/wireless.sh stop
      exit 0
   fi
   echo Failed to download newspaper >> $SDIR/log.txt
   cp $IDIR/iconA.png $SDIR/icon.png
   ./manifester.sh "Logg" "Klarte ikke ï¿½ laste ned avisen"
                                       > $SDIR/manifest.xml
   /usr/bin/wireless.sh stop
   exit 1
fi
```

F.2 Manifest script

A script that takes a title and description as input, and echoes a manifest file with these embedded. Note that the script will probably not work as presented here due to the iLiad being particular about how the manifest file are written on file. The manifest file should preferable be on one line, but are presented here in a more legible format. Use the manifester.sh script in the accompanying archive file.

```
#!/bin/sh
# Echo a manifest file with given Title and Description

title="${1}"
description="${2}"

echo "<?xml version=\"1.0\" encoding=\"utf-8\"?>
         <package>
            <metadata>
               <dc-metadata>
                  <Title>$title</Title>
                  <Description>
                     $description
                  </Description>
                  <Date>2008-04-16T18:02:24</Date>
               </dc-metadata>
               <y-metadata>
                  <startpage>log.txt</startpage>
                  <image>icon.png</image>
                  <version>000</version>
               </y-metadata>
            </metadata>
            <last-location>
               <pagenumber>1</pagenumber>
            </last-location>
            <viewer-settings>
               <zoomfactor>zoomPage</zoomfactor>
               <rotation>0</rotation>
               <positionx>2</positionx>
               <positiony>0</positiony>
               <mode>page</mode>
            </viewer-settings>
         </package>"
```

G
HTML resources

Here, resources used for creating e-newspapers with HTML will be found.

G.1 HTML templates

To create articles as HTML, the Django template engine and such templates as the one below was used.

G.1.1 Base template

The overall base template for articles. The {% block #name# %} lets another template extend and change this. If it is not changed by a template that extends this base template, it will be printed as it is defined in the base template.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<meta name="keywords" type="standard"/>
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
   <head>
      {% block head %}
      <title>
         This be title
      </title>
      <link rel="Stylesheet" href="/enews/media/css/style2.css" type="text/css"/>
      {% endblock %}
      <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
   </head>
   <body>
      {% block content %}
         Innhold!
      {% endblock %}

      {% block footer %}
      <DIV id="footer">
         <a href="#">Forrige artikkel</a>
         <a href="#">Neste artikkel</a>
         <a href="#">Forside</a>
         <a href="#">Kategori</a>
      </DIV>
      {% endblock %}
   </body>
</html>
```

G.1.2 Article details

This template extends the base template of the previous section. It mainly replaces the content block of the base template. The {{ #name# }} is used to fetch information from the dictionary that is passed to the template engine renderer. For example, the {{ object.title }} will be replace by the string that "title" in the object dictionary points to. Likewise the {{ object.author.name }} will be replace by the name of the author of the article.

The {% if start %} is one of the limited possibilities for flow control with the template engine. In this case, it will only render the content between the `if` and `endif` if the passed variable start is `True`. It will only be set to `True` if the page rendered is the first page of an article, for subsequent pages the illustration and credits will thus not be rendered.

```
{% extends "article_base.html" %}

   {% block head %}
      <title>
         {{ object.title }}: Side {{ page }}
      </title>
         <link rel="Stylesheet" href="css/style2.css" type="text/css">
   {% endblock %}

   {% block content %}
      <DIV id="logo">
         <h2> Under Dusken - {{ object.issue }} | {{ volume }} </h2>
      </DIV>

      <DIV id="title">
      <h1>
         {{ object.title }}
      </h1>
   </DIV>

   {% if start %}
   <DIV id="ill">
      <img src="{{ object.illustration.enewsversion }}" alt="{{object.illustration }}">
      <DIV id="ingress">
         <p>
            <b>
            {{ object.ingress|safe }}
            </b>
         </p>
      </DIV>
      <DIV id="credits">
         <p>
            Tekst: {{ object.author.name }} <br/>
            Foto/Ill: {{ object.illustration.author.name }}
         </p>
      </DIV>
   </DIV>
   {% endif %}

   <DIV id="content">
      <p>
      {{ object.body|safe }}
      </p>
   </DIV>
{% endblock %}

{% block footer %}
   <DIV id="footer">
      <a href="#">Forrige artikkel</a>
      <a href="#">Neste artikkel</a>
      <a href="#">Forside</a>
      <a href="#">Kategori</a>
   </DIV>
{% endblock %}
```

H Archive

This is an overview over the files that follows this thesis in an archive file. It will be referenced throughout the text.

H.1 Overview

root

- Background Information
  - Fallproject 2007 - e-newspapers
- Evaluation
  - The questionaires from the evaluation of the demonstration.
- On the iLiad
  - Backup of the newspaper contents from 20. may[1] Use the scripts from this directory for any trials on the iLiad.
- Python_code
  - All the code that has been created for the demonstration. See H.2 for more information.

[1]Note that there was made a few extra e-newspapers to test the demonstration further.

H.2 Python Code

- pyscp.py – Code for uploading to the NTNU public areas via scp.
- sleeper.py – Code for running the newspaper process. It was originally intended that it should run with `cron`, but for some reason `cron` did not work at acceptable time granularities on the hardware available. This will check and create if necessary an e-newspaper every 20 minutes.
- RSSreader.py – Code for fetching article data from an RSS feed. Will also fetch bodytext with correct layout if not present in RSS feed.
- udpdf_simple.py – Code for creating an e-newspaper from the article data that RSSreader fetches. Can either create it as single pages or as one newspaper.

H.2.1 xml

Contains trial XML files for transfering e-newspaper data.

H.2.2 xml up

Folder for XML files that was trialed by the early e-newspaper creator. Do also contain an RSS sample.

H.2.3 pdf

Contains all the newest versions of the PDFs created aswell as the pickled (serialized) data for some of the articles.

H.2.4 iliad, iliad backup and iliad testing

Contains iLiad resources such as programs, contains backup of the latest that was running on the iLiad before end of project, and contains some more software that has not been tested yet. The difference on enhUnderDusken and UnderDusken found in this folder is that enhUnderDusken uses the download.sh and manifester.sh that are mentioned earlier in this appendix. The other uses the original download.sh which is a bit too big to be represented here in the appendix.

H.2.5 html

Contains some of the HTML and PDFs created with the earliest version of the e-newspaper software.

H.2.6 enews

- imagemagic.py – transforms and saves images to a more iLiad friendly format.
- index.py – creates the index at the start of the e-newspaper
- manifest.py – creates the manifest file for the iLiad (for an HTML e-newspaper)
- renderarticle.py – renders article pages in HTML

Templates

Consists of the HTML templates for creating HTML e-newspapers.

Enewspaper

Consists of the files specific for the creation of e-newspapers from XML to HTML.

- article.py – gets article info from XML files
- models.py – the database models are in this file. XML will be saved to a database here.
- newspaper.py – same as for article, just for newspapers
- views.py – prepares and renders the e-newspaper HTML pages.