“LyX Branches” – a case study in Open Source and development of teaching materials

Martin Vermeer

May 7, 2020, originally March 8, 2004

1 Describing the problem

1.1 Multilinguality in education

Finland is a bilingual nation. Our university, Helsinki University of Technology (HUT), is in reality multilingual and very international. The presence of many foreign students implies the need to be able to provide eduation, and educational materials, in English in addition to Finnish and Swedish.

I have on several occasions had to produce exam question sets both in Finnish and in Swedish, sometimes also in English. Also lecture note sets and other teaching materials are sometimes needed in more than one language: in 2001, I was lecturing for the first time the course “Methods of Navigation”, which belongs to the major “Positioning and Navigation”. Among the students were several graduate students, one of which was a foreigner unable to read Finnish. The lecture notes were however only available in Finnish.

In 2001 I solved the problem by providing the student with pages from an English textbook, which contained more or less the same things. I decided however there and then, that I wouldn’t be caught again with my pants down… in 2003, the next time I lectured this course, an English version of the lecture notes was available. And as it happened, there was again a foreign student.

The number of students without a command of Finnish is always small in my courses, but a solution has to be found. This situation will become only more urgent as European policies in support of facilitating international student exchange take hold, which will position us on an international market for attracting the best and brightest to Helsinki.

I applied for funding for this translation work from the HUT’s funds for teaching development, under the name *Verkkodesia* (a word play: *verkko* means “network” in Finnish; *desia* comes from *geodesia* == geodesy). Most of my applied-for funds were granted and the work could start.

1.2 The translation work

In spring 2002 I hired a student to translate the lecture notes into English. She had spent a training period in Canada and thus managed reasonably well in English, but by no means perfectly. I ended up making numerous corrections to the text, especially where professional terminology was concerned. Nevertheless, doing it in this way saved a substantial amount of my time.

This translation work was done using the LyX document processor, which is installed for use on the HUT’s local area network’s UNIX and Linux work stations. The student worked at home using her own home Windows PC and a Secure Shell as well as an "X server" (UNIX's graphic user interface) installed onto it under the HUT's campus licences for these packages.

In this work, the existing standard version of LyX was used with a little "kludge" (ugly, improvised solution) added to it. The basic idea was however great: there was to be a *single source document* containing both the Finnish and the English version in such a way, that corresponding pieces of Finnish and English text would show together in the same on-screen view. Changes could then be made simultaneously in both languages, thus eliminating the synchronisation and jumping back and forth problems that occur when editing two separate documents in their own editing windows.

On the screen, the Finnish text was painted in blue and the English text in magenta. Separate Finnish and English language PDF documents could be output with the aid of a specially crafted LaTeX code stanza in the document preamble, in which only a single number needed to be changed from 1 to 2 in order to get English instead of Finnish output.

1.3 Developing the new feature

Encouraged by this experience, I decided that the "official" LyX sources should support this feature, i.e., editing two different language versions within the same document. To my knowledge no other word processor offers this facility.

I called the feature by the name of "Branches". Each branch can be a language or, e.g., in the case of technical documentation, the slightly different documentation texts for different product versions. Or, when writing student exercises, the version including the answers or "teacher's version" could be one branch; and so on.

A second advantage of using branches is, that those parts of a document that are the same in both languages — formulas, figures, tables —, need to be written only once.

2 A short description of Open Source, LaTeX and LyX

2.1 What is Open Source?

Open Source, or in an older but more appropriate terminology *free software*, is currently best known because of the Linux operating system. It comprises however much more than that. The movement was started somewhere in the 1980's out of frustration with the inflexibility of proprietary software that was distributed without its source code, and that thus could not be modified or adapted by its users.

The defining property of Open Source or free software is, that not only can it be freely distributed, but so can its source code, the original programming language code that is human-readable and from which binary, executable applications are built. And this source is freely modifiable and redistributable. That is how the software grows and improves progressively in the hands of an loosely organized, international development community counting thousands, the Open Source/Free Software Movement. It must be clear that the invention of the Internet was the best thing that could ever have happened to free software... this is where the community lives.

Through the years, many useful applications were produced, including several operating systems, like FreeBSD and Linux. It is only now, however, that the threads are coming together: today a functionally reasonably complete workstation, operatable by somewhat knowledgable users, can be built based completely on Open Source software. Also more and more commercial applications running on these freely distributable systems are beginning to appear as use of free software is spreading in both corporate and public service environments.

Free software has a community ethos that is somewhat similar to that of the scientific community. There are in fact many close connections between the two. It is no coincidence that the Unix operating system is popular in scientific circles and that Linus Torvalds chose it as his model when starting up Linux development. Peer review, building on past achievements, and free exchange of ideas.

And the internet is a Unix network.

It has been argued that Open Source software is more secure than proprietary software, due to its availablility for public inspection, and undoubtedly there is truth in this. It has also be argued, that part of this security advantage is simply due to its smaller installed base and more knowledgable users, and thus its unattractiveness as a target for crackers and malware writers; undoubtedly that is partly true as well. Nevertheless, it is hard to believe that the exposure of the source code to public scrutiny, no matter how few people actually make informed use of this, would not have a positive effect — in the same way that the scrutiny of a free press tends to make governments better behaved.

2.2 What is (La)TEX?

Millions of computer users are familiar with so-called WYSIWYG word processing ("What You See Is What You Get"), as practiced in Microsoft Word, WordPerfect or StarOffice Writer. The governing principle is that the on-screen view is reproduced faithfully on paper.

The beauty of WYSIWYG is, that it is easily learnt and a robust principle: it produces precisely what it promises.

Nevertheless WYSIWYG has its share of problems and limitations. "What You See Is *All* You Get". For this reason, all word processing applications contain also non-WYSIWYG properties, such as, e.g., auto-numbering header styles. If you use those, you never need to number headers manually, and they are automatically included into a table of contents if you specify one. Furthermore one could mention repeating page headers — even different ones for odd and even pages —, auto-numbering pages, "soft" hyphens, etc.

An entirely different approach to document creation is offered by *mark-up language*, of which the most well-known specimen is the HTML or Hypertext Mark-up Language used for Web pages. The principle, however, is much older. The earliest UNIX systems included a program called troff, that was able to beautifully print to paper a text into which suitable mark-up codes had been embedded. Of more recent origin is TEX, a computer mark-up language created by the Californian math professor Donald KNUTH. This language is still today in extensive use for typesetting mathematics-heavy texts for scientific journals.

A popular variant of TEX is LATEX, a macro package created by Leslie LAMPORT, which has as its beautiful characteristic the separation between a document's structure and its visual appearance. A large number of so-called *document classes* have been created for LATEX, standardised layout solutions for, e.g., scientific journals[1]. On the other hand deviating from these standard layouts, i.e., their visual-manual "tuning", is cumbersome or impossible to do. (Not necessarily a bad thing, as such tuning tends to consume inordinate amounts of time and few users dabbling in this are competent typographers.)

If you are curious why on Earth many people still use LATEX when perfectly good WYSIWYG software exists, you only should visually compare a text typeset by LATEX with the same text created in Microsoft Word. Especially if the text is scientific. The typographic quality is from a different planet.

2.3 What is LYX?

Creating your documents for typesetting by LATEX may give great-looking results, but is not easy using an ordinary text editor to add all the necessary mark-up codes. Grown men have been reduced to tears trying. This was the motivation for creating LYX: combine the technical-

[1] Compare these to the DTD's (Document Type Definitions) made for XML formats.

aesthetical superiority of LaTeX with a visual, almost WYSIWYG[2] style way of use. On the surface LyX looks like an ordinary word processor, but under the pretty skin the typesetting engine of LaTeX is humming away.

At the time of writing LyX counts some 125 000 lines of rather variable-quality C++ code, not counting the 70 000 lines of commentary. This reflects the "anyone can contribute" nature of Open Source software development. The good news is that LyX development is driven by a committed core group of competent programmers.

LyX has its own home page: `http://www.lyx.org`.

3 The LyX development process and toolchain

3.1 Development community and history

The lifeline of LyX spans already almost a decade. The first version was developed by the German Matthias Ettrich during 1994–95. Version 0.10.7 was in 1997 already quite useable. Version 1.0 was released in February 1999. This version could be called mature, whereas earlier versions were immature. I.e., one would not give these to an ordinary end user without some reservations or warnings. The current stable version is 1.3; current development effort is expended on what will be version 1.4 when released, presumably during 2004.

At the time of writing there are half a dozen active or core developers. The Norwegian Lars Gullik Bjønnes is responsible for the www and CVS servers and coordinates the development work. In addition to the core developers there are lots of more casual contributors to the development process. Often these people are interested in a particular special feature or property, which they have themselves added to LyX and which they try to maintain. In addition to those who contribute code, the *localizers* deserve special mention. They have translated the menu texts etc. into some 30 different languages — including Russian, Hebrew, Arabic and Chinese! This job never ends, as LyX is developing and changing all the time.

LyX has been crafted in the C++ programming language, making use of its object-oriented programming facilities. The STL (the Standard Template Library) is used heavily, as are many of the powerful facilities made available by Boost (`http://www.boost.org`), e.g., smart pointers. (On the developers' mailing list, sometimes discussions take place on how far to drive the use of modern C++ constructs, as some developers — like many users — have somewhat older systems to work on.)

An important landmark on the LyX development timeline was the decision to modularize the code in such a way, that it would work just as well for any of several alternative graphical user interface solutions: GUI-I, or "GUI[3] Independence".

Like most Open Source contributors, I joined this development community informally by subscribing to the LyX development list(`lyx-devel@lists.lyx.org`) and by submitting there my first proposal for improvement, or "patch[4]". You can believe that they were commented upon! I didn't have any C++ development experience... only procedural languages, like C and Pascal and Fortran, were familiar to me. One learns by doing.

3.2 Graphical toolkits

LyX is a graphical application. The user interacts with it through a graphical user interface or GUI. In the case of LyX, the interface is based upon the X windowing system, the standard windowing solution of the UNIX world.

[2] Or, as the LyX home page puts it: WYSIWYM, What You See is What You *Mean*.

[3] GUI: Graphical User Interface.

[4] The name *patch* presumably comes from the patches of cloth sewn on garments to repair them.

X alone isn't yet a graphical user interface. In principle it could be used as one, but only very, very cumbersomely. Even doing simple things would require hundreds of lines of code. This is why various so-called *graphical toolkits* have been developed.

The oldest of these is Motif. It looks familiar, solid, grey. Windows 3.1 is derived, look-and-feel-like, from Motif. Pretty it is not, but it works. A very early version of LyX was indeed written using Motif.

A bit younger is Xforms. It doesn't really look any better than Motif, but is considerably more versatile and easier to use.

The most recent achievement in UNIX graphic user interfaces is formed by the *graphical desktop environments* such as Gnome and KDE. These are much more than just graphical toolkits — they are complete environments, within which all applications behave in the same, familiar, integrated way. Just like Windows and Macintosh applications behave, or are supposed to behave.

The first version of LyX used Motif. Already the first LyX developer, Matthias ETTRICH, migrated the program to the Xforms toolkit. This is still today the LyX version of reference. Nowadays Qt, the graphical toolkit used by the KDE environment, is equally well supported.

3.3 CVS and other tools

LyX development employs a system called CVS. The data base or *repository* of this system running on a server contains the source codes of the current and all previous versions of LyX. The various versions of the source sode files are stored efficiently as a reference version and difference records or *deltas*. A developer can download from the system the current version of LyX (or in principle also any older version) for use in his own development work. After doing so, he has the whole LyX directory tree on his own hard disk. Then, after having completed his development work, the user can check in or *commit* the changes he made back into the CVS server's repository — provided he has the necessary privileges to do so.

CVS is typically used by many developers simultaneously ("*Concurrent*"). However, if two developers try simultaneously to change the same part of the same file, a *conflict* occurs, which will have to be *resolved* by the slower of the two committers through manual intervention.

There exists on the LyX web site also a Web copy of the CVS repository, which everyone, i.e., the public at large, can inspect by ordinary browser on the LyX home page.

In addition to CVS, many more tools are needed in the development work, casually referred to as the GNU tool chain. Cf. Table 1.

4 Developing the "Branches" feature

I attempted first to implement this "Branches" feature by using a character type or *font* attribute called branch. In other words, every character in a document has, in addition to font attributes — like size and color — still one further attribute: branch, the value of which was allowed to be one of eight colour names: white, black, red, green, blue, cyan, magenta and yellow. These colours where then also used, very logically, as *representation colours* to mark on the screen the text parts of the documents belonging to the branch in question.

This solution, based on text attributes, was in a way no more than a refined version of the above mentioned (in part 1.2) "kludge". As the product of this development work I created a patch, which is referred to in the following message (June 12, 2003): `http://www.mail-archive.com/lyx-devel@lists.lyx.org/msg57376.html`.

This solution, functional as it was, was nevertheless not deemed acceptable for inclusion in LyX. It was too improvised and as its largest

Table 1: Various tools used in LyX development

- A version control system: CVS, Concurrent Versioning System.
- A text editor, e.g., emacs or vi.
- a compiler: gcc, the GNU Compiler Collection, contains also a good C++ compiler.
- make: a management system for selective recompilation, which allows recompilation of only those of the hundreds of LyX source code files that have either themselves changed, or depend on another one that has changed, since the previous compilation.
- the autoconf/automake/libtool tools, which help compile and link the program and its many required libraries in the right way, into an actually working binary, irrespective of the type of the host or its user environment's many possible idiosyncrasies.

drawback was seen, that it would be the *user's* job to remember the correspondence between the various branches and their representation colours.

It was thus decided to create, inside every document, a data structure named BranchList, which would contain the branches defined by the user with their properties (Note: we are using here C++ with its object oriented properties. BranchList is called a *class* and every document contains an *instantiation* of it). The text fragments of a document belonging to a certain branch would be placed in an *inset*, an object embedded in the document text as a kind of container for text and other stuff.

The first inset-based solution was published on July 31, 2003, cf. the message: `http://www.mail-archive.com/lyx-devel@lists.lyx.org/msg59560.html`.

After this, the solution was further developed and refined: e.g., the use of *colours* for the backgrounds of branch insets, having one user defined, arbitrary colour for every branch, was implemented. I was guided in this quite challenging programming task by Angus LEEMING, one of the regulars on the LyX development list.

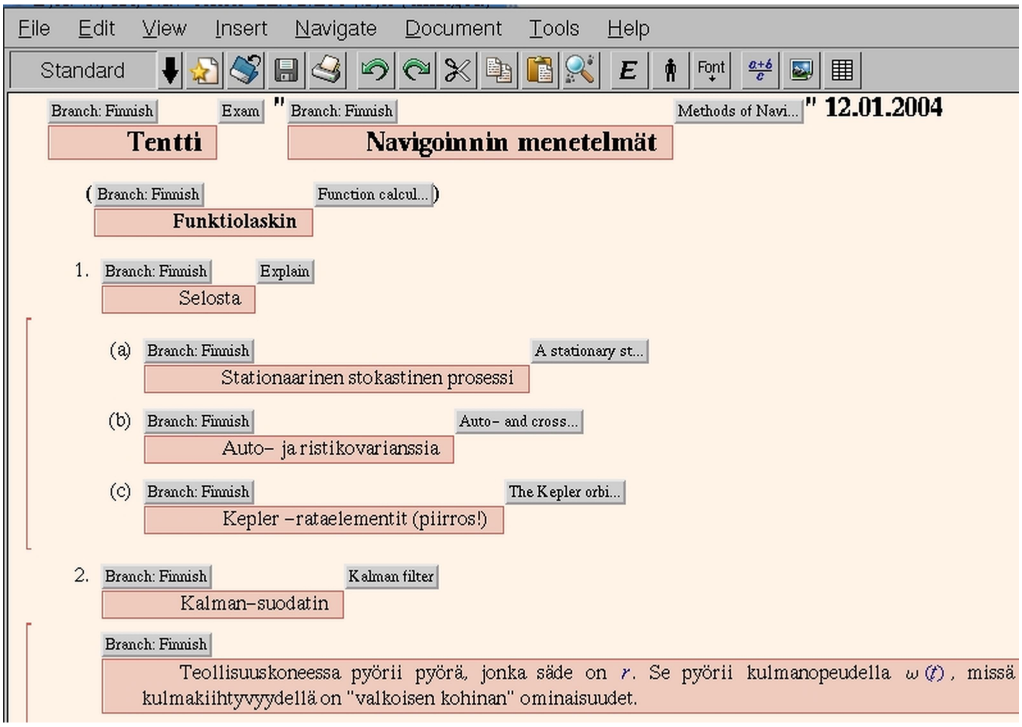
The developers tend to place high demands on contributions and contributors, but are also ready to help as they are aware of the importance of recruiting new blood to the community. As I remember, lots of useful help and advice came also from Alfredo BRAUNSTEIN, André PÖNITZ, John LEVON and Jean-Marc LASGOUTTES — as the names suggest, this is a very international community indeed.

It was not until the 17th of August that the "*Branches* mega-patch" was published, and was committed to the official CVS repository.

We can track the development of the Branches feature also on the basis of the commits made to the CVS repository. At first I tried to implement the branch inset as a variant of the pre-existing Notes inset, before finally splitting it off as its own inset type.

The final, inset based implementation of Branches, which was developed in the time span August 17 – September 22 (version numbers 1.1 – 1.35), can be tracked with the aid of the file `insetbranch.C`: `http://www.lyx.org/cgi-bin/viewcvs.cgi/lyx-devel/src/insets/insetbranch.C`. Here, we find four patches under the name VERMEER. The first of these was the above mentioned "mega-patch" (version 1.1), which in fact created this new file.

Of course this work concerns many more files than only these two: `BranchList.[Ch]`, `FormBranch.[Ch]`, `ControlBranch.[Ch]`, just to mention a few. Nevertheless the two files mentioned above nicely illustrate the progress



Branch: Finnish Exam " Branch: Finnish Methods of Navi... " 12.01.2004

Tentti Navigoinnin menetelmät

(Branch: Finnish Function calcul...)

Funktiolaskin

1. Branch: Finnish Explain

Selosta

(a) Branch: Finnish A stationary st...

Stationaarinen stokastinen prosessi

(b) Branch: Finnish Auto- and cross...

Auto- ja ristikovarianssia

(c) Branch: Finnish The Kepler orbi...

Kepler –rataelementit (piirros!)

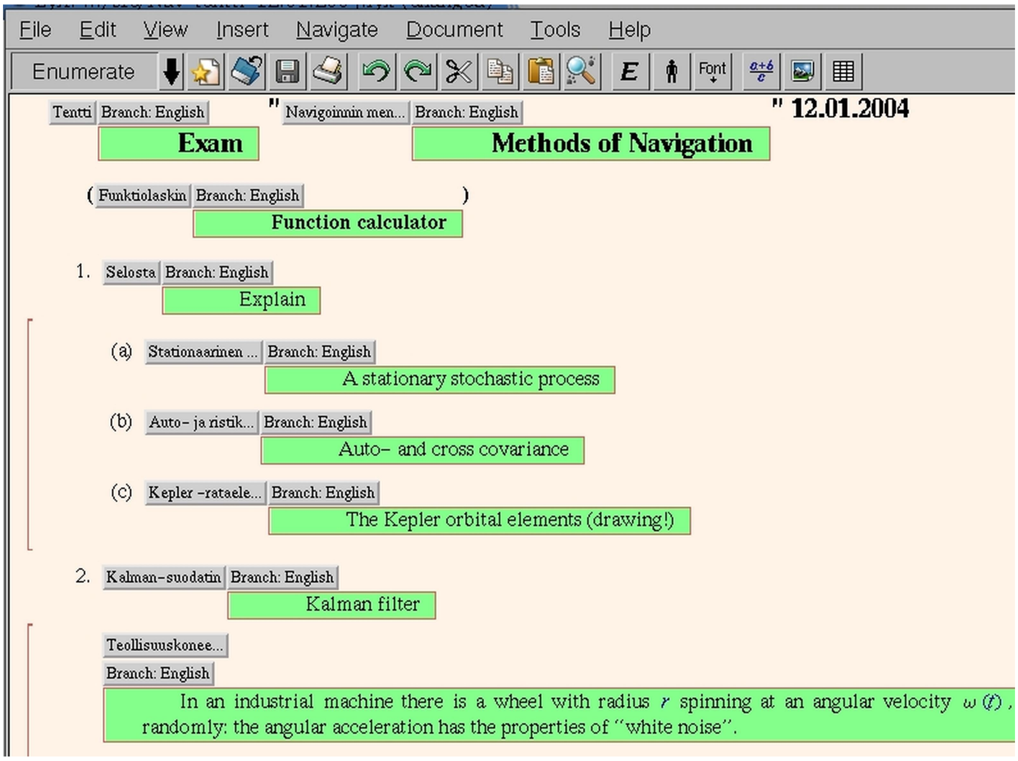
2. Branch: Finnish Kalman filter

Kalman–suodatin

Branch: Finnish

Teollisuuskoneessa pyörii pyörä, jonka säde on r. Se pyörii kulmanopeudella $\omega(t)$, missä kulmakiihtyvyydellä on "valkoisen kohinan" ominaisuudet.

Figure 1: Finnish branch of an exam document



Tentti Branch: English " Navigoinnin men... Branch: English " 12.01.2004

Exam Methods of Navigation

(Funktiolaskin Branch: English)

Function calculator

1. Selosta Branch: English

Explain

(a) Stationaarinen ... Branch: English

A stationary stochastic process

(b) Auto- ja ristik... Branch: English

Auto- and cross covariance

(c) Kepler –rataele... Branch: English

The Kepler orbital elements (drawing!)

2. Kalman–suodatin Branch: English

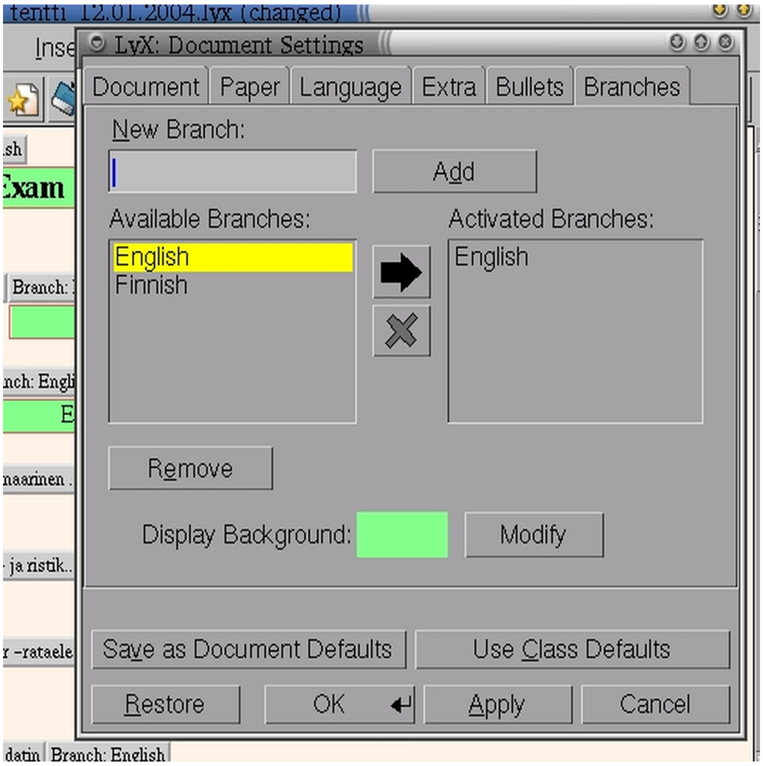
Kalman filter

Teollisuuskonee...

Branch: English

In an industrial machine there is a wheel with radius r spinning at an angular velocity $\omega(t)$, randomly: the angular acceleration has the properties of "white noise".

Figure 2: English branch of an exam document



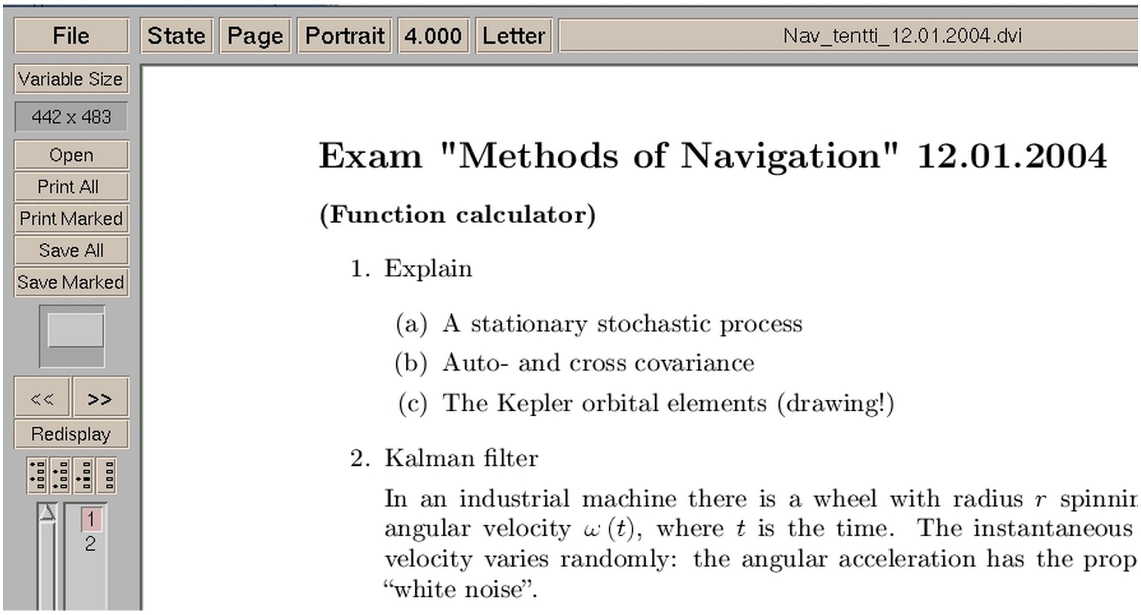
Available Branches:

English
Finnish

Activated Branches:

English

Figure 3: The dialogue for defining branches and setting branch properties



Exam "Methods of Navigation" 12.01.2004

(Function calculator)

1. Explain
   (a) A stationary stochastic process
   (b) Auto- and cross covariance
   (c) The Kepler orbital elements (drawing!)
2. Kalman filter

   In an industrial machine there is a wheel with radius r spinnir angular velocity $\omega(t)$, where t is the time. The instantaneous velocity varies randomly: the angular acceleration has the prop "white noise".

Figure 4: The PostScript$^{\text{TM}}$ output from the English branch

of the work. What this also illustrates is how thoroughly public an open source development effort is, including process documentation. It really wouldn't be wise to, e.g., try and slip in sideways illegally "borrowed" code to a project this visible!

One useful property of LyX insets is, that they can be "collapsed" and opened again by clicking the mouse on their labels. The same feature is also used when you select a certain branch for output on paper. All the insets belonging to that group in the screen will automatically open up, while all the others close themselves like flowers after sunset...

5 Conclusions

As a volunteer effort associated with the *Verkkodesia* project, the Branches feature for preparing multilingual documents from a single source was added to the LyX document processor. The work was done mostly during the summer months, interlaced with a well spent countryside holiday full of physical activity.

From the above description of the work one should get an idea of the disciplined nature of the Open Source development process and how the dynamics of group interaction serves to produce an end result of the highest quality.

One condition contained in the University's funding of this project was

> "to take care that the copyrights related to the project be transferred in a sufficient fashion to the Helsinki University of Technology" [*my transl.*].

In the case of the LyX software this is realized in this way, that the whole application is and has originally been licenced under the GPL (`http://www.opensource.org/licenses/gpl-license.php`), a form of Open Source (`http://www.opensource.org/licenses/`) licence.

The version of LyX next to be released with its new features will undoubtedly be made available for use by the science and education community of HUT during 2004, when the UNIX/Linux work stations and the applications installed on it will be upgraded. LyX can also be used from Windows work stations: it requires the installation of ssh (Secure Shell) and an X server, both of which are available under campus licences. Use on a Linux work station does not require any special steps.

Acknowledgement. Comments on a draft of this text be Angus LEEMING were much appreciated.