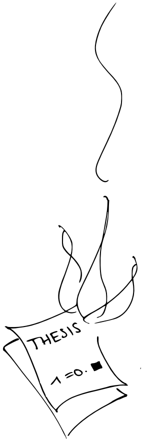
Save early, save often!

Clara Löh

October 2007

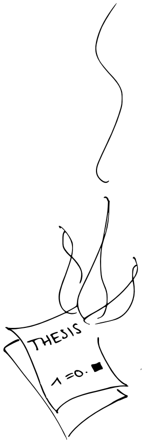


Overview

Natural enemies of theses and articles

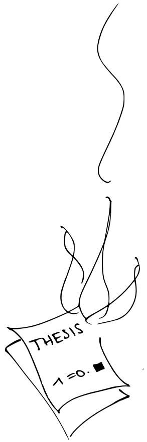
Natural enemies of theses and articles

- Hardware failure, accidental deletion of files
- Incomplete backups
- Cryptic backup names
- Experimenting with layouts, reorganisation of material
- Even more cryptic file names for branches
- Synchronisation with coauthors



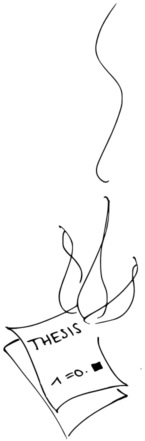
Natural enemies of theses and articles

- Hardware failure, accidental deletion of files
- Incomplete backups
- Cryptic backup names
- Experimenting with layouts, reorganisation of material
- Even more cryptic file names for branches
- Synchronisation with coauthors



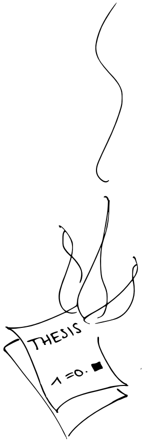
Natural enemies of theses and articles

- Hardware failure, accidental deletion of files
- Incomplete backups
- Cryptic backup names
- Experimenting with layouts, reorganisation of material
- Even more cryptic file names for branches
- Synchronisation with coauthors



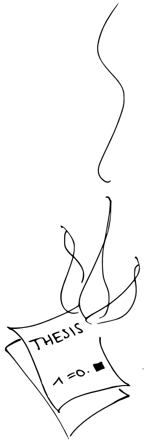
Natural enemies of theses and articles

- Hardware failure, accidental deletion of files
- Incomplete backups
- Cryptic backup names
- Experimenting with layouts, reorganisation of material
- Even more cryptic file names for branches
- Synchronisation with coauthors



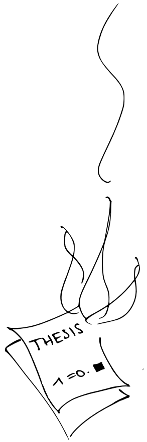
Natural enemies of theses and articles

- Hardware failure, accidental deletion of files
- Incomplete backups
- Cryptic backup names
- Experimenting with layouts, reorganisation of material
- Even more cryptic file names for branches
- Synchronisation with coauthors



Natural enemies of theses and articles

- Hardware failure, accidental deletion of files
- Incomplete backups
- Cryptic backup names
- Experimenting with layouts, reorganisation of material
- Even more cryptic file names for branches
- Synchronisation with coauthors



Defeating Korruptor

What is a version control system?

The purpose of a version control system is
to manage different versions of a (software) project.

Note: LaTeX documents (in particular theses and articles) are software!

Features of version control systems

More specifically, version control systems

- allow to access previous versions
- allow to undo changes;
  good version control systems even allow to undo a (more or less) arbitrary set of changes
- are able to compare different versions of the project and to track changes through the history
- allow to work on a project with a large number of developers
- allow to create variants of the project, so-called branches

Features of version control systems

More specifically, version control systems

- allow to access previous versions
- allow to undo changes;
  good version control systems even allow to undo a (more or less) arbitrary set of changes
- are able to compare different versions of the project
  and to track changes through the history
- allow to work on a project with a large number of developers
- allow to create variants of the project, so-called branches

Features of version control systems

More specifically, version control systems

- allow to access previous versions
- allow to undo changes;
  good version control systems even allow to undo a (more or less) arbitrary set of changes
- are able to compare different versions of the project
  and to track changes through the history
- allow to work on a project with a large number of developers
- allow to create variants of the project, so-called branches

Features of version control systems

More specifically, version control systems

- allow to access previous versions
- allow to undo changes;
  good version control systems even allow to undo a (more or less) arbitrary set of changes
- are able to compare different versions of the project
  and to track changes through the history
- allow to work on a project with a large number of developers
- allow to create variants of the project, so-called branches

Features of version control systems

More specifically, version control systems

- allow to access previous versions
- allow to undo changes;
  good version control systems even allow to undo a (more or less) arbitrary set of changes
- are able to compare different versions of the project
  and to track changes through the history
- allow to work on a project with a large number of developers
- allow to create variants of the project, so-called branches

Benefits of version control systems

In addition to doing all the bookkeeping magic, version control systems

- provide a very convenient backup tool
- encourage dividing the writing process into logical steps
- encourage modular design of documents,
  using a large number of files and directories
- encourage experimenting with layout etc
- let you easily share your documents with other people,
  e.g., supervisors and coauthors

Benefits of version control systems

In addition to doing all the bookkeeping magic, version control systems

- provide a very convenient backup tool
- encourage dividing the writing process into logical steps
- encourage modular design of documents,
  using a large number of files and directories
- encourage experimenting with layout etc
- let you easily share your documents with other people,
  e.g., supervisors and coauthors

Benefits of version control systems

In addition to doing all the bookkeeping magic, version control systems

- provide a very convenient backup tool
- encourage dividing the writing process into logical steps
- encourage modular design of documents,
  using a large number of files and directories
- encourage experimenting with layout etc
- let you easily share your documents with other people,
  e.g., supervisors and coauthors

Benefits of version control systems

In addition to doing all the bookkeeping magic, version control systems

- provide a very convenient backup tool
- encourage dividing the writing process into logical steps
- encourage modular design of documents,
  using a large number of files and directories
- encourage experimenting with layout etc
- let you easily share your documents with other people,
  e.g., supervisors and coauthors

Benefits of version control systems

In addition to doing all the bookkeeping magic, version control systems

- provide a very convenient backup tool
- encourage dividing the writing process into logical steps
- encourage modular design of documents,
  using a large number of files and directories
- encourage experimenting with layout etc
- let you easily share your documents with other people,
  e.g., supervisors and coauthors

Playing with the asteroid – darcs

- Classic examples of version control systems:
  - CVS – classic, centralised version control system
  - subversion – similar to CVS but more recent
- darcs is a
  - distributed
  - very flexible

  version control system
- Resources:
  - http://darcs.net
  - http://wiki.darcs.net/DarcsWiki

Playing with the asteroid – darcs

- Classic examples of version control systems:
  - CVS – classic, centralised version control system
  - subversion – similar to CVS but more recent
- darcs is a
  - distributed
  - very flexible

  version control system
- Resources:
  - http://darcs.net
  - http://wiki.darcs.net/DarcsWiki

Playing with the asteroid – darcs

- Classic examples of version control systems:
  - CVS – classic, centralised version control system
  - subversion – similar to CVS but more recent
- darcs is a
  - distributed
  - very flexible

  version control system
- Resources:
  - http://darcs.net
  - http://wiki.darcs.net/DarcsWiki

A closer look at darcs

Basic operations provided to the user by darcs:

- Initialize a new repository
- Add files/directories to the repository
- Record changes and hence create patches
- Transfer patches to and from other repositories
- Compare different versions
- Track changes
- Undo/redo changes
- Create branches
- Track and resolve conflicts

A closer look at darcs

Basic operations provided to the user by darcs:

- Initialize a new repository
- Add files/directories to the repository
- Record changes and hence create patches
- Transfer patches to and from other repositories
- Compare different versions
- Track changes
- Undo/redo changes
- Create branches
- Track and resolve conflicts

A closer look at darcs

Basic operations provided to the user by darcs:

- Initialize a new repository
- Add files/directories to the repository
- Record changes and hence create patches
- Transfer patches to and from other repositories
- Compare different versions
- Track changes
- Undo/redo changes
- Create branches
- Track and resolve conflicts

A closer look at darcs

Basic operations provided to the user by darcs:

- Initialize a new repository
- Add files/directories to the repository
- Record changes and hence create patches
- Transfer patches to and from other repositories
- Compare different versions
- Track changes
- Undo/redo changes
- Create branches
- Track and resolve conflicts

A closer look at darcs

Basic operations provided to the user by darcs:

- Initialize a new repository
- Add files/directories to the repository
- Record changes and hence create patches
- Transfer patches to and from other repositories
- Compare different versions
- Track changes
- Undo/redo changes
- Create branches
- Track and resolve conflicts

A closer look at darcs

Basic operations provided to the user by darcs:

- Initialize a new repository
- Add files/directories to the repository
- Record changes and hence create patches
- Transfer patches to and from other repositories
- Compare different versions
- Track changes
- Undo/redo changes
- Create branches
- Track and resolve conflicts

A closer look at darcs

Basic operations provided to the user by darcs:

- Initialize a new repository
- Add files/directories to the repository
- Record changes and hence create patches
- Transfer patches to and from other repositories
- Compare different versions
- Track changes
- Undo/redo changes
- Create branches
- Track and resolve conflicts

A closer look at darcs

Basic operations provided to the user by darcs:

- Initialize a new repository
- Add files/directories to the repository
- Record changes and hence create patches
- Transfer patches to and from other repositories
- Compare different versions
- Track changes
- Undo/redo changes
- Create branches
- Track and resolve conflicts

A closer look at darcs

Basic operations provided to the user by darcs:

- Initialize a new repository
- Add files/directories to the repository
- Record changes and hence create patches
- Transfer patches to and from other repositories
- Compare different versions
- Track changes
- Undo/redo changes
- Create branches
- Track and resolve conflicts

A closer look at darcs – What are repositories?

A repository consists of two major parts:

- The recorded state:
  Collection of the patches contained in this repository
- The working copy:
  The instance of the project you currently work in

A closer look at darcs – What are repositories?

A repository consists of two major parts:

- The recorded state:
  Collection of the patches contained in this repository
- The working copy:
  The instance of the project you currently work in

A closer look at darcs – What are patches?

- Fundamental concept: the diff(erence) between two files
- Patch: diff between recorded state and current state of working copy
- Pushing a patch *p* to another repository adds the diff to that repository's recorded state;

  Note: For this to make sense, darcs has to manage dependencies between patches and push also all the patches on which *p* depends!

A closer look at darcs – What are patches?

- Fundamental concept: the diff(erence) between two files
- Patch: diff between recorded state and current state of working copy
- Pushing a patch *p* to another repository adds the diff to that repository's recorded state;

  Note: For this to make sense, darcs has to manage dependencies between patches and push also all the patches on which *p* depends!

A closer look at darcs – What are patches?

- Fundamental concept: the diff(erence) between two files
- Patch: diff between recorded state and current state of working copy
- Pushing a patch p to another repository adds the diff to that repository's recorded state;

  Note: For this to make sense, darcs has to manage dependencies between patches and push also all the patches on which p depends!

A closer look at darcs – What are patches?

- Fundamental concept: the diff(erence) between two files
- Patch: diff between recorded state and current state of working copy
- Pushing a patch p to another repository adds the diff to that repository's recorded state;

  Note: For this to make sense, darcs has to manage dependencies between patches and push also all the patches on which p depends!

Tool demonstration

Let's try the real thing!

What else can we steal from software engineering?

Another tool from software engineering that helps creating documents:

GNU make and Makefiles