

newuserfs
Software Project Management Plan

Martin Pool 9405765 mbp@samba.org

October 8, 2001

Change history

- v1.1** 2/10/2001 Start structuring project plan from notes.
- v1.2** 3/10/2001 (not released)
- v1.3** 7/10/2001 Restructured to comply with IEEE Std 1058–1998.
- v1.4** 7/10/2001 More content added.
- v1.5** 7/10/2001 More content added.
- v1.6** 8/10/2001 First publicly released version, submitted for ANU ENGN3221 project management assignment.

Contents

Change history	i
Table of Contents	ii
Executive Summary	1
1 Overview	2
1.1 Project summary	2
1.1.1 Purpose, scope and objectives	2
1.1.2 Assumptions and constraints	2
1.1.3 Project deliverables	2
1.1.4 Schedule and budget summary	3
1.2 Evolution of the plan	3
2 References	3
3 Definitions	3
4 Project organization	3
4.1 External interfaces	3
4.2 Internal structure	3
4.3 Roles and responsibilities	4
5 Managerial process plans	4
5.1 Start-up plan	4
5.1.1 Estimation plan	4
5.1.2 Staffing plan	4
5.1.3 Resource acquisition plan	4
5.1.4 Project staff training plan	4
5.2 Work plan	5
5.2.1 Work activities	5
5.2.2 Schedule allocation	5
5.2.3 Resource allocation	5
5.2.4 Budget allocation	5
5.3 Control plan	5
5.3.1 Requirements control plan	5
5.3.2 Schedule control plan	5
5.3.3 Budget control plan	5
5.3.4 Quality control plan	6
5.3.5 Reporting plan	6

5.3.6	Metrics collection plan	6
5.4	Risk management plan	6
5.5	Closeout plan	6
6	Technical process plans	6
6.1	Process model	6
6.2	Methods, tools, and techniques	6
6.3	Product acceptance plan	6
7	Supporting process plans	7
7.1	Configuration management plan	7
7.2	Verification and validation plan	7
7.3	Documentation plan	7
7.4	Quality assurance plan	7
7.5	Reviews and audits	7
7.6	Problem resolution plan	7
7.7	Subcontractor management plan	7
7.8	Process improvement plan	7
8	Additional plans	8
A	Work breakdown structure	10
B	External schedule constraints	11
C	Deliverables	12
C.1	Mandatory deliverables	12
C.2	Desirable deliverables	12
D	Milestones	13
E	Known risks	14
F	Budget	15

Executive Summary

This plan relates to the implementation of a system software project called `newuserfs`. The software provides a new framework for writing filesystem modules for Linux that will enhance functionality and ease development of new features.

This is a single-person project of considerable technical difficulty but relatively constrained scope. Estimated total effort is 40–120 hours. Implementation and planning started in parallel in early August of 2001. An initial working version 0.5 will be demonstrated in mid October, 2001.

The challenge in this project is that there are substantial technical risks, hard deadlines, and only a single available developer.

The purpose of this plan is to address these risks by predicting an appropriate set of functionality that can be delivered at high quality by the relevant deadlines.

There are three important motivations which drive the requirements, structure, and resourcing of this project:

1. To learn about preparation of an IEEE Std 1058–1998 software project management plan and about the Linux filesystem implementation.
2. To contribute a useful tool to the Linux community, both in working code and in a conference paper.
3. To gain credit towards COMP3300 and ENGN3221 subjects.

Leaving subject requirements aside, the requirements are at the discretion of the developer.

This documented is intended to comply with IEEE Std 1058–1998, *IEEE Standard for Software Project Management Plans*[2]. A number of sections such as *5.1.2 Staffing plan* are present because they are required by IEEE 1058, although for this project there is not much to say about them. The plan is somewhat compressed to fit in the ENGN3221 page limit.

1 Overview

1.1 Project summary

1.1.1 Purpose, scope and objectives

`newuserfs` enhances the flexibility of the LinuxTM kernel software, allowing new features to be developed and deployed much more effectively.

A computer operating system is divided into a software *kernel*, closely coupled to the machine, and other software, called *userspace* software on Linux. The kernel underlies and supports all other software used on the computer.

Amongst other functions, the kernel provides a filesystem abstraction. Implementing new filesystems is an attractive way to extend operating system functionality into such areas as version control or transparent network access. However because the kernel works at a low level of abstraction development is more difficult and slow than in userspace.

`newuserfs` exports the kernel filesystem programming interface to a *server* program running in userspace.

1.1.2 Assumptions and constraints

Because several aspects of this project are to be submitted for academic credit there are hard external deadlines, summarized in Appendix B on page 11.

The most important constraint on implementation of `newuserfs` is that it is to be implemented by a single programmer who is not working on it full time.

Schedule and functionality are somewhat flexible to allow for limited labour. The only hard constraint with respect to schedule is that documentation must be submitted by the due dates.

1.1.3 Project deliverables

Mandatory deliverables for this project are course requirements for ENGN-3221 and COMP3300, as detailed on pages 11 and 12. These include a technical proposal and semi-technical project management plan, a poster and presentation.

Other deliverables are desirable but not required, and will evolve depending on technical progress and comments from peer reviewers. If `newuserfs` is received well, a conference paper and user's manual may become objectives.

1.1.4 Schedule and budget summary

Since this project is developed for academic credit rather than commercially, there is no explicit budget. Incidental expenses are estimated in Appendix F at under \$100.

1.2 Evolution of the plan

The SPMP will be updated at intervals to track progress. The current version is always available from the project home page, and older versions are under configuration management in CVS.

2 References

Broad requirements for this project are set by COMP3300 and ENGN3221 material[10].

The `newuserfs` home page at <http://etc.samba.org/newuserfs/> provides source code, documentation and links to other resources.

3 Definitions

For more detailed explanation of technical terms used in this document consult the `newuserfs` home page[8].

4 Project organization

4.1 External interfaces

The course coordinators for COMP3300 and ENGN3221 are customers for project documentation deliverables, although unusually they will not use the delivered source code.

The Linux community as a whole is the potential end-customer for the software.

4.2 Internal structure

There is only a single developer for this project. The Samba Software Foundation sponsors development of the project by providing hosting of development servers.

4.3 Roles and responsibilities

There is only a single developer and manager for this project, Martin Pool. Responsibilities include project definition and planning, design and development, and documentation and promotion.

5 Managerial process plans

5.1 Start-up plan

5.1.1 Estimation plan

Effort estimates are shown in the work breakdown in Appendix A. The breakdown is based on the initial architecture, on external constraints, and on research into similar projects. Estimates are based on the number of lines in similar projects, and personal experience on similar projects and on the prototypes of this project. Milestones and tasks have been chosen to be of approximately equal size.

5.1.2 Staffing plan

A single engineer, Martin Pool, is working on this project.

5.1.3 Resource acquisition plan

Approval to use Samba Software Foundation resources for this project has already been obtained. This includes hosting for the project web site and CVS repository.

5.1.4 Project staff training plan

This project comprises a training exercise in both kernel development and software project management.

Technical education will draw upon published documentation about the kernel filesystem interface. In addition a number of experienced kernel developers will act as volunteer mentors, providing early feedback on design approaches, including Daniel Phillips and Paul Russell.

5.2 Work plan

5.2.1 Work activities

Appendix A on page 10 presents a task-oriented work breakdown. The milestone schedule in Appendix D groups together different functional tasks into chronological order.

5.2.2 Schedule allocation

Leaving aside delivery deadlines, work from different activities can proceed in parallel. After an initial planning, design, and feasibility check process, it is planned to progress development, testing and documentation on different areas of functionality in turn, directed by milestones.

5.2.3 Resource allocation

There are no significant resource allocations for this project apart from staff time.

5.2.4 Budget allocation

Predictions for the minor financial costs of this project are stated in Appendix F on page 15.

5.3 Control plan

5.3.1 Requirements control plan

There is no formal requirements control plan. Development is directed towards the requirements specified in the proposal document[9], but can be varied as the problem is better understood.

5.3.2 Schedule control plan

Since the project deadlines are very firm, milestone scope will be adjusted towards a self-consistent and stable software build before each delivery.

5.3.3 Budget control plan

Not relevant to this project.

5.3.4 Quality control plan

See section 7.4.

5.3.5 Reporting plan

No formal reporting is required for this project. The web page will contain regular progress information.

5.3.6 Metrics collection plan

Formal metrics will not be used on this project. (However see 7.8 for plans for staff development.)

5.4 Risk management plan

Known risks to successful and on-time completion of this project are listed in Appendix E.

5.5 Closeout plan

Continuation after completion of the immediate deliverables will depend on responses from peer reviews and the development community at large. There are no concrete plans for closeout at this point.

6 Technical process plans

6.1 Process model

Development will follow an incremental, milestone-driven method. Each milestone should be reached with a consistent and useful set of functionality, test cases for implemented functions, and reasonable confidence in code coverage.

6.2 Methods, tools, and techniques

`newuserfs` will be implemented primarily in C. Libraries to write servers in scripting languages such as Perl, Python or Java will be considered.

6.3 Product acceptance plan

Beyond internal quality checks there are no acceptance criteria at this point.

7 Supporting process plans

7.1 Configuration management plan

All source code and documentation is controlled by the CVS configuration management system at `cvs.samba.org`[1, 8].

7.2 Verification and validation plan

Explicit V&V is not planned for this project because of the organic nature of the requirements.

7.3 Documentation plan

Documentation includes the deliverables specified on page 12, plus informal design documentation on the web site.

7.4 Quality assurance plan

The primary QA method for this project is a test suite to be developed in parallel with product functionality. The architecture makes it relatively straightforward to ensure all code is covered by a test case.

7.5 Reviews and audits

Peer review will be provided by the mentors mentioned on page 4.

7.6 Problem resolution plan

A problem resolution system will be determined closer to project release. If the project is publicly adopted then problem reporting and resolution will be managed through the SourceForge.net or Samba systems.

7.7 Subcontractor management plan

No subcontractors are used on this project.

7.8 Process improvement plan

Project management documentation and a development journal will be maintained through the project, and examined to identify excessive rework or opportunities for better process.

8 Additional plans

No other plans are specified for this project.

References

- [1] Moshe Bar, Karl Franz Fogel. *Open Source Development with CVS*. The Coriolis Group. 1999.
- [2] IEEE Std 1058–1998: *IEEE Standard for Software Project Management Plans*. The Institute of Electrical and Electronics Engineers, Inc. 1998.
- [3] Steve McConnell. *Code Complete: A Practical Handbook of Software Construction*. Microsoft Press. 1993.
- [4] Jim McCarthy. *Dynamics of Software Development*. Microsoft Press. 1995.
- [5] Gerald M. Weinberg. *The Psychology of Computer Programming*. Van Nostrand Reinhold. 1971.
- [6] Watts S. Humphrey. *A Discipline for Software Engineering*. Addison-Wesley. 1995.
- [7] Dave W. Farthing. *Notes on Software Project Management*. University of Glamorgan. 1999.
<http://www.comp.glam.ac.uk/Teaching/projectmanagement/>
- [8] Martin Pool. *newuserfs home page*. <http://etc.samba.org/newuserfs/>
- [9] Martin Pool. *newuserfs Proposal and Progress Paper*.
<http://etc.samba.org/newuserfs/doc/progress/progress.pdf>
- [10] Eric McCreath. *ANU COMP3300 Assignment 2 Specification*. 4 September 2001. <http://cs.anu.edu.au/Student/comp3300/ass2.html>
- [11] *The User-Mode Linux Home Page*.
<http://user-mode-linux.sourceforge.net/>

A Work breakdown structure

The WBS should be read in conjunction with Appendix D, which shows the ordering of technical tasks into milestone units.

ID	Expected days	Description
T0		Planning
T0.0	1.0	Set scope, assess feasibility
	1.0	Total
T1		Design and development
T1.0	2.5	Review previous work
T1.1	1.0	Architectural design
T1.2	2.5	Peer review design
T1.3	1.0	IPC protocol
T1.4	1.0	Server framework
T1.5	1.0	Filesystem framework
T1.6	1.0	i-node management
T1.7	1.0	M2 file operations
T1.8	1.0	M3 file operations
	12.0	Total
T2		Testing
T2.0	1.0	Test framework
T2.1	1.0	M2 file tests
T2.2	1.0	M3 file tests
	3.0	Total
T3		Documentation
T3.0	2.0	Progress/proposal document
T3.1	3.0	Plan document
T3.2	2.0	Prepare poster and presentation
	7.0	Total
	23.0	Grand Total

B External schedule constraints

- 2/10/2001** COMP3300 progress and proposal paper.
- 8/10/2001** ENGN3221 project plan assignment: deliver newuserfs Software Project Management Plan.
- 16/10/2001** COMP3300 poster and demonstration: deliver code version 0.5 and test suites, and poster.

C Deliverables

C.1 Mandatory deliverables

- Single-page proposal/progress paper for COMP3300 outlining intended functionality and architecture.
- Software Project Management Plan for ENGN3221. Should comply with IEEE Std 1058–1998 if possible, limited to one summary page and six body pages.
- Working code implementing at least the functionality of milestone M2 (page 13), including both the general framework and specific sample implementations.
- A poster and live demonstration for COMP3300. The presentation should last about five minutes, and demonstrate the most interesting and attractive implementation features and functions of newuserfs.

C.2 Desirable deliverables

- A web site describing the project and linking to source code, documentation and other resources.
- A conference paper describing the project targeted at a forum such as `linux.conf.au` or the Ottawa Linux Symposium.

D Milestones

Development milestones[3, 4] relate only to program functionality, and indicate a rough order of dependency and priority. Milestones primarily indicate code completion, but also depend upon requirements analysis, research, design, and documentation.

The work for each milestone includes construction of appropriate tests to verify operation, and peer review of the design. Milestone completion implies test completion and confidence in test coverage.

- M0** *Framework and zero function:* A kernel module `userfs.o` can be compiled and installed into the kernel.
- M1** *Filesystem mounts:* The filesystem can be installed into the kernel and mounted, and the userspace server is started and serves requests. The mounted filesystem has a root directory, though perhaps no other content.
- M2** *Basic directory operations:* Using a userspace ramdisk server, can create, list, and remove directories.

`mkdir` is the first file-creating operation to be implemented because it takes only a filename as a parameter and no other information. Therefore it is the simplest directory-modifying operation and will serve to verify operation of the framework.
- M3** *Basic file operations:* Regular files can be created, read and written. Sample userspace servers store the data either in redirection to another directory, or to in-memory data structures.
- M4** *Solid error handling:* Sample userspace servers are enhanced to return error codes at various points during operation, to test that the kernel handles these errors correctly. The kernel should cope if the userspace server terminates unexpectedly.
- M5** *Complete file operations:* Implement remaining basic file operations, including `rename`, `stat`, `symlink`.
- M6** *Locking operations:* Support locking of whole files, or record ranges within a file. Cause processes to block properly when waiting for locks. Possibly arrange a way for the server to call back into the kernel to indicate that a resource has become available.

E Known risks

R0 Technical risks:

R0.0 Concurrency bugs causing deadlock or liveness problems, caused by the kernel blocking on a userspace process. This may cause the system to be unreliable. It is hard to analyze the risk without completing a substantial fraction of the research.

R0.1 Unclassified bugs, possibly causing an unsuccessful demonstration. The most likely cause is suspected to be misunderstanding of the kernel filesystem programming interface. This risk will be addressed by peer review and system testing, and secondarily by adhering closely to a well-tested script during the demonstration.

R1 Resource risks:

R1.0 Insufficient time to work on `newuserfs` because of competing commitments.

R1.1 Unavailability of technical resources, for example because of network or hardware failure on the CVS server or web server. Possibly inconvenient, but not likely to substantially impact the project because backup copies are kept and other machines are available.

R2 Customer risks:

R2.0 Project management documentation not satisfactory. This project is different from some of the others specified for this assignment, and so the management plan may not address the requirements sufficiently. Addressed by communicating with course coordinators.

R2.1 Project is obsolete before release because of competing work: there are other projects in this area that are documented on the `newuserfs` web page. None are currently as promising as `newuserfs`, however others may emerge before this project is complete. This would not directly interfere with completion of the project, but its impact would be reduced if it duplicated existing work.

F Budget

Printing costs	\$5
Poster stationery	\$20
Phone calls, etc	\$10
Miscellaneous (20%)	<u>\$7</u>
Total	<u>\$42</u>