	Test dell´eseguibile ufs_mke2fs									0.1	Test di ufs_mount / ufs_umount		0.05	Test di ufsh								0.1	Unit Tests									0.5	Implementazione				0.1	Docs		0.1	Funz. agg.	0.05	1.00	3p=0 2p=2 1p=4	0-2 voti		
Gruppo	Default	Valori min 1	Valori min 2	Valori medi	Valori max 1	dim_cluster errato	num_cluster errato	nome_vol errato	Opzione errata	**Average**	ufs_mount	ufs_umount	**Average/10**	ls	cp	mv	rm	cat	mkdir	rmdir	tree	**Average/10**	UFS.CD	UFS.FCRR	UFS.FOC	UFS.FRWL	UFS.FST	UFS.DMRR	UFS.DOC	UFS.DRD	UFS.DST	**Average/10**	Installazione	Makefile	Qualita' codice	Modularita'	**Average/10**	Commenti	Relazione	**Average/10**	Funzione aggiunta obbl. per progetto	**Average/10**	**Media pesata**	**Bonus gruppo**	**Estensioni opzionali**	**penalizzazioni per errori nel codice**	**Voto**
Compito-Perfetto	1	1	1	1	1	1	1	1	1	**1.00**	10	10	**1.00**	10	10	10	10	10	10	10	10	**1.00**	10	10	10	10	10	10	10	10	10	**1.00**	10	10	10	10	**1.00**	10	10	**1.00**	10	**1.00**	**33.00**	**0**	**0**	**0**	**33.00**
Bellincampi Lorenzo 1152535 es1: 21 Flati Tiziano 1143472 es1: 27 Ottaviani Jacopo 1145000 es1: 25	1	1	1	1	1	1	1	1	0	**0.89**	8	10	**0.90**	6	6	6	6	6	6	6	6	**0.60**	10	10	10	10	9	10	10	10	10	**0.99**	6	8	10	10	**0.85**	10	9	**0.95**	6	**0.60**	**29.65**	**0**	**1**	**1**	**29.65**
MAZZEI FILIPPO 1045675 es1: 19 MARZO GIUSEPPE 1014884 es1: 18	1	1	1	1	1	1	1	1	1	**1.00**	10	10	**1.00**	10	0	10	10	10	10	10	10	**0.88**	10	10	10	10	10	10	10	10	10	**1.00**	6	8	10	10	**0.85**	10	8	**0.90**	6	**0.60**	**31.10**	**2**	**1**	**1**	**33.10**
Voto max = compito perfetto	33																																														

linee codice e commenti	linee test SW	NOTE
3278	0	Da verificare all'appello il funzionamento della shell. Spiegazione di chk_rights().
5840	105	Da verificare allìappello cp in ufsh. Spiegazione di come e' stata testata ufs_fcntl().

Bellicampi Flati Ottaviani	
Fuori Spec	I comandi eseguibili non hanno prefisso ufs_ e cio' crea seri problemi con quelli Linux (ls nella dir $HOME/ufs/bin puo' lanciare quello di ufs e non quello Linux). ufsd si chiama daemon , ufsh si chiama ufs_shell .
Relazione	README con un paio di FAQ. Relazione molto buona. Notata una discrepanza: check_rights non ha questo nome nel codice invece e' chk_rights.
Codice	Ottimo, commenti al posto giusto, molto leggibile, ret code controllato sempre, malloc/free OK, funzioni di lunghezza giusta, proto.tipi con PRE e POST condizioni.
Makefile	Ricompila sempre perche' target all dipende dagli eseguibili nel dir corrente ma vengono creati nella dir ./bin . Mancano dipendenze da .h per la compilazione dei .o .
Installaz	Bisogna sempre fare un unistall prima di fare un install poiche' si ha errore nella creazione delle dir ufs e bin. Sarebbe stato meglio creare le dir solo se NON esistevano gia' oppure ignorare l'errore (utilizzo di – davanti al comando) per far andare avanti il make. Vengono copiate in $HOME/bin non i soli eseguibili e la libreria ma anche le dir src, inc e obj !
test	Esiste dir test ma non ci sono solo delle prove su alcuni comandi. Esistono eseguibili per ogni funzioni ufs_* della libreria, che sono poi utilizzati dalla shell mediante system().
ufs_mke2fs	OK tutti eccetto test case opzione errata -j, che risulta PASSED senza msg errore.
ufs_mount, ufs_umount	mount non fa fork + exec ma solo exec, quindi blocca il prompt.
ufsh	Vedi commenti alla riga “test”. NON SI RIESCE AD UTILIZZARLA, poiche' da a prove alterne un errore del tipo: Failed to open /tmp/dev_ufs.RFIFO.6903 in WRONLY RMDIR: FAILED Failed to open /tmp/dev_ufs.RFIFO.6903 in WRONLY
libufs.a: GENERAL	OK, il codice complessivo (lib+daemon) sembra stabile anche dopo uso continuato. dev_ufs rimane sempre in uno stato pulito.
UFS.CD	OK
UFS.FCRR	OK
UFS.FOC	OK
UFS.FRWL	OK
UFS.FST	Vengono allocati 2 cluster dopo la lseek 2048 from start, ma non sono ancora stati scritti i byte dopo la lseek. Perche' quando il file e di 2034 byte in un dev_ufs con cluster size 512, si hanno 6 cluster allocati invece di 5 ?
UFS.DMRR	OK
UFS.DOC	OK
UFS.DRD	OK L'unico test case FAILED e' TEST CASE UFS.DRD.20d-ReaddirDirRoot-3, ma e' quello che doveva essere aggiornato in ufs_test.h (mail su mailing list). Quindi sono tutti PASSED !
UFS.DST	OK
Funzione aggiuntiva	Permessi accesso. Forse c'e' errore concettuale perche' il getuid() per settare l'UID di un nuovo file creato lo fa il server daemon, mentre lo lo dovrebbe fare la libreria e spedirlo al server. Infatti l'UID deve essere quello dell'utente che lancia l'applicazione linkata a libufs.a e non quello del server.
Estensioni opzionali	Buco nei file con ufs_lseek(). Ottimizzazione del meccanismo di lock (inizio implementazione).
Penalizzazioni	Se dev_ufs esiste mke2fs scrive nel file gia' esistente (flag O_CREAT) preservando la dimensione che quindi non corrisponde ai parametri inseriti (con dev_ufs esistente provare la creazione di filesystem con parametri -c e -k diversi). Sarebbe stato meglio aggiungere O_EXCL alla open(). Comunque, se dev_ufs esiste gia', dovrebbe dare msg errore oppure cancellare dev_ufs pre-esistente (con warning e conferma).

Marzo Mazzei	
Fuori Spec	Nessuna
Relazione	Buona, potevano essere aggiunte informazioni sulle principali strutture dati interne, tipo tabella dei file aperti o lista dei server attivi.
Codice	Commenti ed indentazione ottimi, codice molto leggibile. Esiste gestione standardizzata dell'errore mediante ufs_GestioneErrore().
Makefile	Il default dovrebbe essere solo compilazione e link (cioe' make all), invece e' make install. Il make ricompila sempre tutto, perche' eseguibili target vengono generati in BINDIR e quindi la dipendenza non li trova nella dir corrente del makefile.
Installaz	Viene sempre rimossa la dir lib e ufs in $HOME, si potrebbe fare verifica se esiste e copiare solo i file di ufs (eseguibili) evitare di cancellare eventuali file dell'utente non appartenenti a ufs.
test	Esiste sorgente test_syscalls dedicato ai test delle funzionalita' aggiuntive.
ufs_mke2fs	OK
ufs_mount, ufs_umount	OK, ufs_umount informa ufsd del comando di terminazione inviando un pid=0.
ufsh	cp da sempre: ufs $ cp -s /home/verola/dumfile /filea cp: Errore percorso non valido Non si riesce a farla funzionare in nessun caso. OK mkdir, rmdir, mv (bene anche per spostare una dir da un ramo dell'albero a un'altro), cat, ls, tree. Mv e rm OK. Dopo alcune operazioni (mkdir, rmdir, ls) puo' dare ERRORE del tipo: ufs $ rmdir /dira ufsh: Errore directory non vuota ufs $ rmdir /dirb ufs $ rmdir /directory ufs $ ls 0x410 500 32 Sun Jun 29 04:31:09 2008 Sun Jun 29 04:31:09 2008 dira *** glibc detected *** ufsh: free(): invalid next size (fast): 0x08f0b458 *** ======= Backtrace: ========= /lib/libc.so.6[0x17aac1] /lib/libc.so.6(cfree+0x90)[0x17e0f0] ufsh[0x8050757] ufsh[0x80501ef] /lib/libc.so.6(__libc_start_main+0xe0)[0x127390] ufsh[0x8048bf1] ======= Memory map: ======== 00110000-00111000 r-xp 00110000 00:00 0 [vdso] 00111000-00264000 r-xp 00000000 fd:00 1442052 /lib/libc-2.7.so 00264000-00266000 r--p 00153000 fd:00 1442052 /lib/libc-2.7.so 00266000-00267000 rw-p 00155000 fd:00 1442052 /lib/libc-2.7.so 00267000-0026a000 rw-p 00267000 00:00 0 002af000-002ca000 r-xp 00000000 fd:00 1442051 /lib/ld-2.7.so 002ca000-002cb000 r--p 0001a000 fd:00 1442051 /lib/ld-2.7.so 002cb000-002cc000 rw-p 0001b000 fd:00 1442051 /lib/ld-2.7.so 06e91000-06e9c000 r-xp 00000000 fd:00 1442058 /lib/libgcc_s-4.1.2-20070925.so.1 06e9c000-06e9d000 rw-p 0000a000 fd:00 1442058 /lib/libgcc_s-4.1.2-20070925.so.1 08048000-08052000 r-xp 00000000 fd:00 9068124 /home/verola/ufs/ufsh 08052000-08053000 rw-p 0000a000 fd:00 9068124 /home/verola/ufs/ufsh 08053000-0805c000 rw-p 08053000 00:00 0 08f0b000-08f2c000 rw-p 08f0b000 00:00 0 [heap] b7e00000-b7e21000 rw-p b7e00000 00:00 0 b7e21000-b7f00000 ---p b7e21000 00:00 0 b7f4b000-b7f4d000 rw-p b7f4b000 00:00 0 b7f65000-b7f69000 rw-p b7f65000 00:00 0 bfa53000-bfa68000 rw-p bffeb000 00:00 0 [stack] Aborted
libufs.a: GENERAL	OK
UFS.CD	OK
UFS.FCRR	OK
UFS.FOC	OK
UFS.FRWL	OK
UFS.FST	OK
UFS.DMRR	OK

UFS.DOC	OK
UFS.DRD	OK
UFS.DST	OK
Funzione aggiuntiva	ufs_fcntl()
Estensioni opzionali	APPEND e TRUNC: verificate -> OK.
Penalizzazioni	Crash nell'uso di ufsh.