

ISW2

REPORT SWTesting

Valerio Crecco
0320452

Repository **BookKeeper**: <https://github.com/creccovalerio/bookkeeper>

Repository **ZooKeeper**: <https://github.com/creccovalerio/zookeeper>

Indice

1) Apache BookKeeper	2
1.1) BufferedChannel class – write() method	2
1.2) BufferedChannel class – read() method	4
1.3) WriteCache class – put() method	5
1.4) WriteCache class – get() method	7
2) Apache ZooKeeper	8
2.1) PathUtils class – validatePath() method	8
2.2) DataTree class – createNode() method	10
2.3) DataTree class – deleteNode() method	14
2.4) ZKUtil class – listSubTreeBFS() method	16

BookKeeper

bookie.BufferedChannel.java

La classe BufferedChannel, come si legge dalla documentazione, fornisce un livello di bufferizzazione davanti al FileChannel utilizzato per le operazioni di lettura e scrittura sul log. BookKeeper, infatti, utilizza un file di log per il salvataggio delle entry. In questo modo si effettuano meno operazioni sul disco, velocizzando in generale la scrittura e lettura delle entries.

Metodi

I metodi della classe BufferedChannel.java considerati sono:

- public void **write**(ByteBuf src) throws IOException;
- public synchronized int **read**(ByteBuf dest, long pos, int length) throws IOException;

Per quanto riguarda il metodo:

- public void **write**(ByteBuf src) throws IOException;

Tale metodo si occupa di scrivere il contenuto di *src* all'interno di un buffer di scrittura (*writeBuffer*), facendo il flush su un file di log soltanto se la dimensione della cache non fosse sufficiente a contenere *src*.

Domain Partitioning

Per il domain partitioning, oltre al parametro *src*, è stata considerata anche la capacità del *writeBuffer*, che determina se è necessario effettuare il flush sul file di log.

Parametro	Classi di equivalenza
ByteBuf src	{null}, {valid_instance}, {invalid_instance}
int capacity	{≤0}, {>0}

Analizzando il ruolo di questi due parametri si può notare una correlazione tra l'input *src* ed il parametro *capacity* che specifica la dimensione del buffer di scrittura, infatti il BufferedChannel prova a scrivere l'entry sul writeBuffer. Nel caso in cui il buffer non abbia spazio a sufficienza per contenere *src*, il BufferedChannel scrive direttamente sul FileChannel. In base a ciò, non è stata più a considerata la validità o meno dell'entry *src*, ma si è posta la sua dimensione in relazione alla capacità assegnata al buffer di scrittura.

Parametro	Classi di equivalenza
int capacity	{≤0}, {>0}
int srcSize	{<capacity}, {=capacity}, {>capacity}

Boundary-values

Per il parametro *capacity* è stato considerato anche valore di 5 in quanto il solo caso di *capacity*=1 non sembra descrivere al meglio il comportamento del *writeBuffer*.

Parametro	Boundaries
int capacity	-1, 0, 1, 5
int srcSize	{capacity-1}, {capacity}, {capacity+1}

Essendo presenti pochi parametri con poche classi di equivalenza, è stato deciso di optare un approccio multidimensionale. Per capire se il metodo si comporta nel modo atteso, essendo questo un metodo che ritorna void, si è controllato il numero di bytes che sono stati scritti o si è controllato che venissero lanciate opportune eccezioni in base agli input dati. Alcuni valori di output sono stati recuperati tramite un approccio white-box, ove la documentazione non è stata sufficiente. I valori degli output attesi dai test progettati sono:

capacity	srcSize	Output/Descrizione
-1	-2	Si attende che venga lanciata un'eccezione di tipo NegativeArraySizeException, legata alla dimensione negativa assegnata a capacity e srcSize
-1	-1	Si attende che venga lanciata un'eccezione di tipo NegativeArraySizeException, legata alla dimensione negativa assegnata a capacity e srcSize
-1	0	Si attende che venga lanciata un'eccezione di tipo IllegalArgumentException legata alla dimensione negativa assegnata a capacity
0	-1	Si attende che venga lanciata un'eccezione di tipo NegativeArraySizeException legata alla dimensione negativa assegnata a srcSize
0	0	Si attende che non venga fatta l'operazione di scrittura in quanto src ha dimensione nulla e che non venga sollevata un'eccezione

0	1	Si attende che il contenuto di src venga scritto direttamente sul file in quanto capacity è nullo
1	0	Si attende che non venga fatta l'operazione di scrittura, in quanto src ha dimensione nulla, e che non venga sollevata un'eccezione
1	1	Si attende che venga fatta la scrittura su file a causa della saturazione della capacity
1	2	Si attende che vengano fatti due flush su file a causa della saturazione della capacity per due volte
5	4	Si attende che NON venga fatto la scrittura sul file perché non viene saturata la capacity
5	5	Si attende che venga fatta la scrittura su file a causa della saturazione della capacity
5	6	Si attende che venga fatta la scrittura su file a causa della saturazione della capacity

Tutti i precedenti test hanno rispettato il comportamento atteso, tranne il test con {capacity,srcSize} = {0,1}.

In questo caso infatti ci si aspettava una scrittura diretta sul file channel avendo un livello di buffering nullo.

Invece, si nota che il programma entra in un ciclo infinito, senza terminare. Ciò potrebbe essere legato ad un possibile bug all'interno del codice che non viene gestito correttamente tramite il meccanismo delle eccezioni. Osservando il codice del metodo write(), si è osservato come, essendo nulla la capacità del buffer, il programma cerca, correttamente, di effettuare immediatamente il flush di src sul FileChannel, ma la condizione di uscita del while (*copied < len*) non viene mai soddisfatta. Il parametro copied viene infatti incrementato di una quantità intera *bytesToCopy*, valore che tuttavia risulta sempre nullo essendo calcolato come il minimo tra un certo valore ed il numero di byte scrivibili del writeBuffer, ma dato che wirteBuffer ha capacità nulla, e di conseguenza copied sarà sempre pari a zero. Per far terminare l'esecuzione di questo test case, è stata utilizzata l'annotazione @Test(timeout=1000). Tramite l'uso di JaCoCo si è effettuata l'analisi del coverage, raggiungendo il 74% di statement coverage e del 60% di branch coverage, come mostrato nella [figura_1a](#). Andando ad osservare il report, come si può osservare dalla [figura_2a](#), si può notare come non sia stato esplorato il branch *if(doRegularFlushes)* (riga 134). La variabile booleana *doRegularFlushes* viene settata a true nel momento in cui viene istanziato il *BufferedChannel*, soltanto se *unpersistedBytesBound* è maggiore di zero. Questo parametro rappresenta il massimo numero di bytes che possono restare nel BufferedChannel senza essere stati scritti sul file. Pertanto a livello pratico permette di effettuare la scrittura sul FileChannel non solo quando termina lo spazio nel buffer, ma al superamento di una certa soglia prefissata. Per quanto riguarda il report generato da ba-dua, come si può vedere in [figura_3a](#), vi sono 17 su 48 coppie def-use non coperte. Alla luce di queste informazioni, si è cercata di sfruttare queste per aumentare la coverage precedente e ridurre le coppie def-use non coperte. In particolare si è cercato di creare il BufferedChannel tramite il costruttore *BufferedChannel(allocator, fc, capacity, unpersistedBytesBound)* impostando quindi anche il valore di *unpersistedBytesBound* per i nuovi test, mentre è stato posto a 0 per i test visti in precedenza. Per quanto riguarda una possibile classe di equivalenza per tale parametro, si può osservare come ci sia anche in questo caso una relazione con scrSize

Parametro	Classi di equivalenza
int capacity	{≤0}, {>0}
int srcSize	{<capacity}, {=capacity}, {>capacity}
long unpersistedBytesBound	{0}, {<srcSize}, {>srcSize}

La test suite, quindi, è stata estesa aggiungendo altri due casi di test; uno in cui *unpersistedBytesBound < srcSize* ed uno in cui *unpersistedBytesBound > srcSize*. I valori per i nuovi test case non sono stati scelti tramite boundary analysis in quanto è già stato valutato il comportamento ai bordi del dominio e si è voluto inoltre testare il sistema anche per valori più grandi. Sono stati aggiunti i seguenti casi di test:

capacity	srcSize	unpersisted BytesBound	Output/Descrizione
50	30	20	Si attende che venga fatta la scrittura su file a causa del superamento del valore <i>unpersistedBytesBound</i>
50	30	40	Si attende che NON venga fatto la scrittura sul file perché non viene saturata la capacity e nemmeno superata la soglia <i>unpersistedBytesBound</i>

Con l'aggiunta di questi test, è stato raggiunto il 100% di statement coverage ed il 100% di branch coverage, come si vede dalla [figura_4a](#), inoltre, grazie a tali test, si è riuscito a ridurre il numero di def-use non coperte a 3 su 48, come si può vedere dalla [figura_5a](#). Per quanto riguarda il report di Pit, come si vede in [figura_6a](#), si può vedere come non siano state rilevate le mutazioni su *doRegularFlushes* (riga 134). Si noti come que-

sto sia un comportamento atteso in quanto la mutazione introdotta è una *negated conditional* che disattiva il flush al superamento del bound specificato. Quindi come nei casi di test considerati inizialmente la scrittura sul log avverrà soltanto al superamento della capacità massima, e la mutazione non viene rilevata venendo in ogni caso scritta la stessa quantità di dati.

Per quanto riguarda il metodo:

- public synchronized int **read**(ByteBuf dest, long pos, int length) throws IOException;

Tale metodo si occupa di leggere i dati dal *readBuffer* associato al *BufferedChannel* e li inserisce in un *ByteBuf* di destinazione passato come primo parametro. Oltre a questo, il metodo prende in input il parametro *pos* (posizione da cui iniziare a leggere il buffer) e *length* (numero di bytes da leggere a partire da *pos*). Per istanziare il *BufferedChannel* bisogna passare come input il *FileChannel* associato al file che si vuole leggere e la capacità desiderata per i buffer di lettura e scrittura. Per la realizzazione dei test, quindi, è stata creata una funzione *generateRandomFile* che permette di generare un file temporaneo contenente dei bytes randomici. Per ogni caso di test sono stati considerati, oltre ai parametri del metodo *read*, anche *capacity*, ovvero la dimensione del buffer di lettura, e *fileSize*, ovvero la dimensione del file che verrà generato.

Domain Partitioning

Analizzando il ruolo svolto da ognuno di questi parametri, è stata individuata una correlazione tra le variabili *length*, *pos* e *fileSize*; in particolare deve sicuramente valere la relazione per cui $pos+length \leq fileSize$, altrimenti si cercherebbe di effettuare una lettura oltre la fine del file. Pertanto le classi di equivalenza individuate sono:

Parametro	Classi di equivalenza
long pos	{≤0}, {>0}
Int length	{≤0}, {>0}
int fileSize	{<0}, {≤ pos+length}, {> pos+length}
int capacity	{≤0}, {>0}

Boundary-values

Per la selezione dei valori è stata applicata la boundary values analysis, considerando anche i valori aggiungendo i valori *length*=5 e *capacity*=5, perché si è ritenuto *length*=1 non sufficientemente generale per testare il metodo in quanto va' a leggere un solo byte. Analogamente *capacity*=1 rappresenta un caso banale in cui i bytes vengono semplicemente letti e bufferizzati uno per volta

Parametro	Boundaries
long pos	-1, 0, 1
Int length	-1, 0, 1, 5
int fileSize	1, {pos+length-1}, {pos+length}, {pos+length+1}
int capacity	-1, 0, 1, 5

Essendo presenti diversi parametri è stato deciso di utilizzare un approccio unidimensionale. A causa della mancanza di documentazione sufficiente per i primi 4 test è stato utilizzato un approccio di tipo white-box. I valori di output attesi dai test progettati sono:

Pos	Length	FileSize	Capacity	Output/Descrizione
0	0	-1	0	Si attende che venga lanciata un'eccezione di tipo <i>NegativeArraySizeException</i> perché si crea un file di dimensioni negative
-1	0	0	0	Si attende che venga lanciata un'eccezione di tipo <i>ArrayIndexOutOfBoundsException</i> , legata al valore negativo dell'indice <i>pos</i>
0	-1	0	0	Si attende che venga lanciata un'eccezione di tipo <i>IllegalArgumentException</i> , legata al valore negativo di <i>length</i>
0	0	0	-1	Si attende che venga lanciata un'eccezione di tipo <i>IllegalArgumentException</i> , legata al valore negativo di <i>capacity</i>
0	0	0	0	Si attende che vengano letti 0 bytes senza che venga lanciata un'eccezione
0	5	4	1	Si attende che venga lanciata un'eccezione (<i>IOException</i>) legata al fatto di provare a leggere oltre la fine del file
0	5	6	5	Si attende che vengano letti 5 bytes

1	5	6	5	Si attende che vengano letti 5 bytes
---	---	---	---	--------------------------------------

Con i precedenti test, dal report di JaCoCo, come mostrato in [figura_7a](#), si è raggiunta una statement coverage del 79% e branch coverage del 61%. Analizzando il report, come si vede in [figura_8a](#), si è visto come i casi di test non abbiano coperto le istruzioni di riga 257-259. Questo avviene perché nei test sviluppati, il branch di riga 253 veniva esplorato soltanto nel caso `bytesToCopy==0` nel test case {0,5,4,1}. A causa di ciò veniva sempre lanciata l'eccezione `IOException` (riga 254) senza eseguire le successive istruzioni. Per aumentare la copertura è stata estesa la test suite prevedendo un ulteriore test in cui si scrive nel `writeBuffer` in fase di configurazione, e si tenta successivamente di effettuare una lettura partendo da una posizione non concessa. È stato quindi aggiunto il seguente test:

Pos	Length	FileSize	Capacity	Output/Descrizione
9	3	5	2	Si attende che venga lanciata un'eccezione di tipo IOException

Con l'aggiunta di questo test, la statement coverage ha raggiunto il valore del 91% mentre il branch coverage del 66%, come si può vedere dalla [figura_9a](#). Grazie a questo test si è riuscito a coprire le righe che non venivano raggiunte in precedenza, come si vede dalla [figura_10a](#). Per quanto riguarda il report generato da ba-dua, come si vede in [figura_11a](#), sono state rilevate 37 coppie def-use non coperte su 118. Per quanto riguarda mutation testing, il report di Pit, come si vede in [figura_12a](#), mostra come diversi mutanti non siano stati rilevati. Un mutante non-killed interessante mostrato dal report è quello di riga 276 in cui non è stato rilevato un diverso comportamento cambiando la boundary di `if (readBytes <= 0)`. Dunque, al fine di risolvere il mutante legato a `readBytes=-1` è stato incluso un caso di test con `capacity=0` e `fileSize`, `pos`, `length` diversi da zero.

Pos	Length	FileSize	Capacity	Output/Descrizione
1	6	5	0	Si attende che venga lanciata un'eccezione di tipo IOException

Il nuovo report Pit, come si vede nella [figura_13a](#), ha mostrato come la mutazione venga ora rilevata. Anche il report Jacoco ha mostrato dei miglioramenti sul coverage, avendo raggiunto una statement coverage del 95% e branch coverage del 72%, come si vede dalle [figura_14a](#), [figura_15a](#). Andando ad analizzare le motivazioni per cui si è raggiunta una statement coverage soltanto del 72%, è stato evidenziato come questa sia dovuta al fatto che NON si ha mai `writeBuffer==null`. Per cercare di esplorare questo branch è stata quindi modificata l'implementazione del test program, istanziando il `BufferedChannel` tramite il costruttore: `BufferedChannel(ByteBufAllocator allocator, FileChannel fc, int writeCapacity, int readCapacity, ...)`. Tuttavia ponendo `writeCapacity=0` viene istanziato un `writeBuffer` di dimensione zero, ma NON nullo, mentre con `writeCapacity=null` viene lanciata una `NullPointerException`. Pertanto lo scenario con `writeBuffer==null` non sembra poter essere soddisfatto in quanto il buffer di scrittura viene istanziato automaticamente al momento della creazione del `BufferedChannel`. Sempre per le precedenti ragioni, il nuovo report generato da ba-dua ([figura_16a](#)) mostra con siano presenti ancora 36 coppie def-use non coperte su 118.

bookie.Idb.WriteCache.java

`WriteCache.java` è una classe che fornisce metodi per allocare una quantità di memoria richiesta e che viene organizzata in molteplici segmenti. Le entry vengono appese in un buffer e riferite tramite un hashmap fino a quando la cache non viene cancellata. Si può navigare le entry della cache in modo ordinato, ad esempio per `ledgerId`, `entryId`.

Metodi

- public boolean `put`(long ledgerId, long entryId, ByteBuf entry);
- public ByteBuf `get`(long ledgerId, long entryId);

Per quanto riguarda il metodo:

- public boolean `put`(long ledgerId, long entryId, ByteBuf entry);

Questo metodo copia nella cache il contenuto del `ByteBuf` entry, identificato da `entryId` e da scrivere nel ledger specificato da `ledgerId`.

Domain Partitioning

Parametro	Classi di equivalenza
long ledgerId	{≤0}, {>0}
long entryId	{≤0}, {>0}
ByteBuf entry	{null}, {valid_instance}, {invalid_instance}

Boundary-values

Parametro	Boundaries
long ledgerId	{-1}, {0}, {1}
long entryId	{-1}, {0}, {1}
ByteBuf entry	{null}, {valid_instance}, {invalid_instance}

Per invalid_instance si intende una entry con dimensioni maggiori della dimensione della cache (più precisamente sono state utilizzate delle entry di dimensioni pari al doppio della cache). Essendo presenti pochi parametri con poche classi di equivalenza, è stato deciso di optare un approccio multidimensionale. Per la progettazione dei casi di test è stato utilizzato eccezionalmente un approccio white-box, a causa dell'assenza di documentazione. Per la valutazione dell'esito dei test si è preso in considerazione il valore che viene ritornato dalla chiamata del metodo (*true/false*), si è controllato anche il numero di entry presenti in cache oppure si è verificato che venissero lanciate opportune eccezioni in base agli input forniti. I valori di output attesi dai test progettati sono:

ledgerId	entryId	entry	Output/Descrizione
-1	-1	null	Si attende che venga lanciata un'eccezione di tipo NullPointerException, perché entry è null
-1	-1	valid_instance	Si attende che venga lanciata un'eccezione di tipo IllegalArgumentException, perché i valori assegnati a ledgerId/entryId sono negativi
-1	-1	invalid_instance	Si attende che venga restituito false perché si sta cercando di inserire una entry NON valida
-1	0	null	Si attende che venga lanciata un'eccezione di tipo NullPointerException, perché l'entry è null
-1	0	valid_instance	Si attende che venga lanciata un'eccezione di tipo IllegalArgumentException, perché ledgerId è negativo
-1	0	invalid_instance	Si attende che venga restituito false perché si sta cercando di inserire una entry NON valida
-1	1	null	Si attende che venga lanciata un'eccezione di tipo NullPointerException, perché l'entry è null
-1	1	valid_instance	Si attende che venga lanciata un'eccezione di tipo IllegalArgumentException, perché ledgerId è negativo
-1	1	invalid_instance	Si attende che venga restituito false perché si sta cercando di inserire una entry NON valida
0	-1	null	Si attende che venga lanciata un'eccezione di tipo NullPointerException, perché l'entry è null
0	-1	valid_instance	Si attende che venga lanciata un'eccezione di tipo IllegalArgumentException, perché entryId è negativo
0	-1	invalid_instance	Si attende che venga lanciata un'eccezione perché entryId è negativo o perché entry è NON valida
0	0	null	Si attende che venga lanciata un'eccezione di tipo NullPointerException, perché l'entry è null
0	0	valid_instance	Si attende che l'entry venga inserita con valore di ritorno true
0	0	invalid_instance	Si attende che l'entry NON venga inserita in quanto NON valida con il ritorno del valore false
0	1	null	Si attende che venga lanciata un'eccezione di tipo NullPointerException, perché l'entry è null
0	1	valid_instance	Si attende che l'entry venga inserita con valore di ritorno true
0	1	invalid_instance	Si attende che l'entry NON venga inserita in quanto NON valida con il ritorno del valore false
1	-1	null	Si attende che venga lanciata un'eccezione di tipo NullPointerException perché

			l'entry è null
1	-1	valid_instance	Si attende che venga lanciata un'eccezione di tipo <code>IllegalArgumentException</code> , perché <code>entryId</code> è negativo
1	-1	invalid_instance	Si attende che l'entry NON venga inserita in quanto NON valida con il ritorno del valore <code>false</code>
1	0	null	Si attende che venga lanciata un'eccezione di tipo <code>NullPointerException</code> , legata al fatto che l'entry è null
1	0	valid_instance	Si attende che l'entry venga inserita con valore di ritorno <code>true</code>
1	0	invalid_instance	Si attende che l'entry non venga inserita in quanto NON valida, con valore di ritorno <code>false</code>
1	1	null	Si attende che venga lanciata un'eccezione di tipo <code>NullPointerException</code> , perché l'entry è null
1	1	valid_instance	Si attende che l'entry venga inserita con valore di ritorno <code>true</code>
1	1	invalid_instance	Si attende che l'entry non venga inserita in quanto NON valida, con valore di ritorno <code>false</code>

Con questi test si è ottenuta una statement coverage del 96% ed una branch coverage del 62%, come si vede dalla [figura_17a](#) e [figura_18a](#). Dal report di ba-dua, come mostrato in [figura_19a](#), si può osservare come il numero di coppie def-use non raggiunte sia di 11 su 56. Al fine di aumentare le precedenti percentuali di coverage e ridurre le coppie def-use non raggiunte, è stata utilizzata una istanza di cache non più con dimensione massima dei segmenti impostata a valori di default, ma con dei valori variabili di `maxSegmentSize`. E' stato progettato un caso di test in cui si fa uso di una entry con una dimensione pari alla dimensione della cache e con `maxSegmentSize = entrySize`. E' stato quindi aggiunto il seguente caso di test:

ledgerId	entryId	entry	Output/Descrizione
1	1	cacheSizeEntry	Si attende che l'entry NON venga inserita in quanto ha una dimensione maggiore di quella di <code>maxSegmentSize</code> , con valore di ritorno <code>false</code>

Introducendo questo caso di test, come si può osservare in [figura_20a](#), si è riuscito ad aumentare la percentuale di statement coverage fino al 97% e di branch coverage fino al 75%, coprendo il branch di riga 154, come si vede dalla [figura_21a](#). Inoltre grazie a questo caso di test, si è riuscito a ridurre il numero di coppie def-use non raggiunte portandole a 7 su 56, come si vede dalla [figura_22a](#). Dal report di PIT ([figura_23a](#) e [figura_24a](#)), invece, si può osservare la presenza di 2 mutanti. Al fine di eliminare il mutante presente a riga 151, è stato progettato il seguente caso di test, in cui si inserisce una entry di dimensioni pari alla cache e con l'uso di `MaxSegmentSize` anche essa pari alla dimensione della cache:

ledgerId	entryId	entry	Output/Descrizione
1	1	cacheSizeEntry	Si attende che l'entry venga inserita in quanto ha una dimensione pari a quella della cache, con valore di ritorno <code>true</code>

Grazie a questo test, come si vede in [figura_25a](#) e [figura_26a](#), è stato eliminato il mutante di riga 151.

Per quanto riguarda il metodo:

- public ByteBuf `get(long ledgerId, long entryId)`;

Questo metodo prende in input l'identificativo del ledger e l'identificativo dell'entry, restituendo il contenuto dell'entry se questa è presente in cache.

Domain Partitioning

Parametro	Classi di equivalenza
long ledgerId	{≤0}, {>0}
long entryId	{≤0}, {>0}

Boundary-values

Parametro	Boundaries
long ledgerId	{-1}, {0}, {1}
long entryId	{-1}, {0}, {1}

Essendo presenti solo un parametro con poche partizioni/classi di equivalenza, è stato deciso di optare un approccio multidimensionale. Per la progettazione dei casi di test è stato utilizzato eccezionalmente un approccio white-box, a causa dell'assenza di documentazione. Per la valutazione dell'esito dei test, prima di eseguire la `get()`, sono state effettuate delle `put()` all'interno della cache anche in base ai risultati dell'analisi di quest'ultimo metodo; quindi si è controllato che una entry fosse effettivamente presente in cache con un certo contenuto al suo interno o no, oppure si è verificato che venissero lanciate opportune eccezioni in base agli input forniti. I valori di output attesi dai test progettati sono:

ledgerId	entryId	Output/Descrizione
-1	-1	Si attende un'eccezione di tipo <code>IllegalArgumentException</code> perché si prova a recuperare una entry usando degli id < 0
-1	0	Si attende un'eccezione di tipo <code>IllegalArgumentException</code> perché si prova a recuperare una entry usando <code>ledgerId</code> < 0
-1	1	Si attende un'eccezione di tipo <code>IllegalArgumentException</code> perché si prova a recuperare una entry usando <code>ledgerId</code> < 0
0	-1	Si attende che il metodo restituisca una entry <code>null</code> perché tale coppia NON può essere inserita nella cache
0	0	Si attende che il metodo restituisca tale entry dopo che è stata correttamente inserita nella cache
0	1	Si attende che il metodo restituisca tale entry dopo che è stata correttamente inserita nella cache
1	-1	Si attende che il metodo restituisca una entry <code>null</code> perché tale coppia NON può essere inserita nella cache
1	0	Si attende che il metodo restituisca tale entry dopo che è stata correttamente inserita nella cache
1	1	Si attende che il metodo restituisca tale entry dopo che è stata correttamente inserita nella cache

Con l'uso della precedente test suite si ottengono dei valori di statement coverage del 100% e di branch coverage del 100%, come si vede in [figura_27a](#). Tramite l'uso di ba-dua, come si può vedere in [figura_28a](#), sono state coperte tutte le coppie def-use, mentre dalla [figura_29a](#), si può osservare come anche tutti i mutanti risultino KILLED.

ZooKeeper

common.PathUtils.java

La classe PathUtils.java si occupa di verificare la validità della stringa che rappresenta il path di uno ZNode.

Metodi

- public void **validatePath** (String path) throws `IllegalArgumentException`;

Domain Partitioning

Parametro	Classi di equivalenza
String path	{null}, {valid_string}, {invalid_string}, {"“"}

ZooKeeper ha un namespace gerarchico, tuttavia, non è possibile utilizzare qualsiasi carattere Unicode per costruire il path che identifica uno ZNode. Per la definizione delle stringhe valide e non valide si è fatto riferimento alle informazioni della documentazione. Il namespace fornito da ZooKeeper è molto simile a quello di un file system standard, infatti, il nome di uno ZNode è una sequenza di elementi separati da uno slash (/). Ogni nodo all'interno del namespace di ZooKeeper è identificato da un path e questo path deve iniziare con il carattere slash (/), che rappresenta il nodo root dell'albero.

Boundary-values

Parametro	Boundaries
String path	null, "/zn1", "/zn1/zn2", "zn1", "zn1/zn2", “”

Essendo presente un solo parametro con poche partizioni/classi di equivalenza, è stato deciso di optare un approccio multidimensionale. Per capire se il metodo si comporta nel modo atteso, essendo questo un metodo che ritorna void, si è cercato di sfruttare il fatto che, dalla segnatura del metodo, viene specificato il lancio di una specifica eccezione (`IllegalArgumentException`) nel momento in cui viene fornito in input un path che non è valido. I valori di output attesi dai test progettati sono:

path	Output/Descrizione
null	Si attende un'eccezione di tipo IllegalArgumentException, in quanto non è un path valido (path fornito NON inizia con lo /)
"/zn1"	Si attende che NON venga sollevata un'eccezione perché il path ha un formato atteso
"zn1/zn2"	Si attende che NON venga sollevata un'eccezione perché il path ha un formato atteso
"zn1"	Si attende un'eccezione di tipo IllegalArgumentException, in quanto non è un path valido (path fornito NON inizia con lo /)
"zn1/zn2"	Si attende un'eccezione di tipo IllegalArgumentException, in quanto non è un path valido (path fornito NON inizia con lo /)
""	Si attende un'eccezione di tipo IllegalArgumentException, in quanto non è un path valido (path fornito NON inizia con lo /)

Considerando la precedente test suite, come si può osservare dalla [figura_1b](#), si ottengono delle percentuali non soddisfacenti. Cercando ulteriori informazioni nella [documentazione](#) di ZooKeeper, si è scoperta l'informazione secondo cui non è possibile utilizzare il carattere “.” da solo per indicare un nodo lungo un percorso. Ciò è legato al fatto che ZooKeeper non utilizza percorsi relativi. Tuttavia, il carattere “.” può essere utilizzato come parte del nome del nodo. Inoltre, sono stati aggiunti dei casi di test che utilizzano la stringa “..” all'interno del path dello ZNode per coprire più casi riguardanti i percorsi relativi. Sempre dalla documentazione si è scoperto che esistono dei caratteri Unicode che non possono essere inseriti all'interno del path dello Znode. Infine, ho considerato il caso in cui il path immesso corrisponde alla radice dell'albero. La stringa “/” deve rappresentare un path valido in quanto corrisponde ad un elemento della struttura ad albero, per come è stato definito il modello dei dati nel progetto. Ai precedenti casi di test sono stati aggiunti i seguenti tests:

Path	Output/Descrizione
“/..”	Si attende un'eccezione di tipo IllegalArgumentException in quanto il path è un percorso relativo
“/zn1/..”	Si attende un'eccezione di tipo IllegalArgumentException in quanto il path è un percorso relativo
“zn1/..zn2”	Si attende un'eccezione di tipo IllegalArgumentException in quanto il path è un percorso relativo e non inizia con lo /
“/zn1/zn2/..”	Si attende un'eccezione di tipo IllegalArgumentException in quanto il path è un percorso relativo
“/zn1/zn2/zn3/”	Si attende un'eccezione di tipo IllegalArgumentException in quanto il path non può terminare con il carattere “/”
“./zn1”	Si attende un'eccezione di tipo IllegalArgumentException in quanto il path è un percorso relativo
“/zn1./zn2”	Si attende un'eccezione di tipo IllegalArgumentException in quanto il path è un percorso relativo
“zn1/zn2./”	Si attende un'eccezione di tipo IllegalArgumentException in quanto il path è un percorso relativo
“/.”	Si attende un'eccezione di tipo IllegalArgumentException in quanto il path è un percorso relativo
“/zn.1”	Si attende che il metodo non lanci una eccezione in quanto il carattere “.” può essere utilizzato come parte di del nome.
“./zn1”	Si attende che il metodo non lanci una eccezione in quanto il carattere “.” può essere utilizzato come parte di del nome.
“/zn1.”	Si attende che il metodo non lanci una eccezione in quanto il carattere “.” può essere utilizzato come parte di del nome.
“/”	Si attende che il metodo non lanci una eccezione in quanto tale path corrisponde alla radice dell'albero.
“\u0000”	Si attende un'eccezione di tipo IllegalArgumentException in quanto il path non può contenere questo carattere perché non viene visualizzato correttamente o può essere visualizzato in modo confuso
“\u001e”	Si attende un'eccezione di tipo IllegalArgumentException in quanto il path non può contenere questo carattere perché non viene visualizzato correttamente o può essere visualizzato in modo confuso
“\u0001”	Si attende un'eccezione di tipo IllegalArgumentException in quanto il path non può contenere questo carattere perché non viene visualizzato correttamente o può essere visualizzato in modo confuso
“\u001f”	Si attende un'eccezione di tipo IllegalArgumentException in quanto il path non può contenere questo carattere perché non viene visualizzato correttamente o può essere visualizzato in modo confuso

“\u007f”	Si attende un'eccezione di tipo IllegalArgumentException in quanto il path non può contenere questo carattere perché non viene visualizzato correttamente o può essere visualizzato in modo confuso
“\u009f”	Si attende un'eccezione di tipo IllegalArgumentException in quanto il path non può contenere questo carattere perché non viene visualizzato correttamente o può essere visualizzato in modo confuso
“\ud800”	Si attende un'eccezione di tipo IllegalArgumentException in quanto il path non può contenere questo carattere perché non viene visualizzato correttamente o può essere visualizzato in modo confuso
“\ud801”	Si attende un'eccezione di tipo IllegalArgumentException in quanto il path non può contenere questo carattere perché non viene visualizzato correttamente o può essere visualizzato in modo confuso
“/\uf8ff”	Si attende un'eccezione di tipo IllegalArgumentException in quanto il path non può contenere questo carattere perché non viene visualizzato correttamente o può essere visualizzato in modo confuso
“\ufff0”	Si attende un'eccezione di tipo IllegalArgumentException in quanto il path non può contenere questo carattere perché non viene visualizzato correttamente o può essere visualizzato in modo confuso
“\ufff5”	Si attende un'eccezione di tipo IllegalArgumentException in quanto il path non può contenere questo carattere perché non viene visualizzato correttamente o può essere visualizzato in modo confuso
“\uffff”	Si attende un'eccezione di tipo IllegalArgumentException in quanto il path non può contenere questo carattere perché non viene visualizzato correttamente o può essere visualizzato in modo confuso

Con i precedenti casi di test si è riuscito ad aumentare notevolmente le percentuali precedenti, come si può vedere dalla [figura_2b](#). Dal report generato da ba-dua ([figura_3b](#)), invece, possiamo osservare che vi sono 37 coppie def-use non coperte su un totale di 122. Utilizzando PIT, si è ottenuto il risultato mostrato nella [figura_4b](#) e [figura_4.1b](#). Come si può vedere dalle precedenti figure, ancora le percentuali di statement coverage e branch coverage non sono ottimali perché alcune porzioni di codice rimangono non coperte e vi sono molte coppie def-use non coperte e dei mutanti non uccisi. In particolare, in [figura_4b](#), possiamo osservare a riga 71 la presenza di 6 mutazioni, di cui 2 NO COVERAGE e 4 KILLED, e a riga 81 troviamo ben 16 mutazioni, di cui solamente 1 SURVIVED e 15 KILLED. Utilizzando, a questo punto, un approccio di tipo white-box, si è deciso di realizzare dei test in grado di coprire le righe di codice non raggiunte in precedenza. Dunque sono stati progettati ulteriori due casi di test:

Path	Descrizione
“/zn1//zn2”	Si attende un'eccezione di tipo IllegalArgumentException in quanto il path NON è valido
“/zn1/../zn2”	Si attende un'eccezione di tipo IllegalArgumentException in quanto il path è un percorso relativo

Grazie a questi due ulteriori test, si è ottenuta una statement coverage del 100% ed una branch coverage del 92%, come si può vedere dalla [figura_5b](#), mentre per quanto riguarda il report generato da ba-dua, come si vede in [figura_6b](#), il numero di coppie def-use non coperte è sceso a 29 su 122. Infine, per quanto riguarda PIT, come si può vedere dalla [figura_7b](#) e [figura_7.1b](#), sono stati eliminati tutti i mutanti a riga 72, mentre rimane ancora in vita 1 dei 16 mutanti presenti a riga 81.

server.DataTree.java

DataTree.java è una classe che ha un ruolo chiave all'interno del sistema perché gestisce una data tree structure. In particolare si mantengono due importanti strutture dati parallele: una tabella hash che mappa i percorsi completi dei nodi ai DataNode e un albero di DataNode.

Metodi

- public void **createNode**(String path, byte[] data, List<ACL> acl, long ephemeralOwner, int parentCVersion, long zxid, long time) throws NoNodeException, NodeExistException;
- public void **deleteNode** (String path, long zxid) throws NoNodeException;

Per quanto riguarda il metodo:

- public void **createNode**(String path, byte[] data, List<ACL> acl, long ephemeralOwner, int parentCVersion, long zxid, long time) throws NoNodeException, NodeExistException;

Tale metodo permette di aggiungere un nuovo nodo al DataTree.

Domain Partitioning

Parametro	Classi di equivalenza
String path	{null}, {valid_string}, {invalid_string}, {"")}
byte[] data	{null}, {valid_instance}, {invalid_instance}
List<ACL> acl	{null}, {empty_list}, {valid_list}
long ephemeralOwner	{≤0}, {>0}
int parentCVersion	{≤0}, {>0}
long zxid	{≤0}, {>0}
long time	{≤0}, {>0}

Boundary-values

Per quanto riguarda l'individuazione dei valori boundary per ciascun paramentro:

- **path:** le informazioni raccolte nella documentazione per il testing del precedente metodo (`validatePath`) sono valide anche per questo metodo. Inoltre, bisogna considerare che un path risulta valido per l'inserimento di un nodo solo se l'albero è stato costruito in modo corretto fino al precedente inserimento. Per esempio, l'inserimento del nodo con path "/zn1/zn2" deve andare buon fine solo se esiste già il nodo con path "/zn1".
- **data:** questo parametro è un array di byte. Dalla documentazione è stato scoperto come siano ammessi array di dimensione massima di N = 1MB. Tale informazione è stata utilizzata per testare array invalidi dando a questi una dimensione maggiore di tale limite (N+m). Inoltre, sono realizzati test con la presenza di array "null" e vuoto.
- **acl:** tale parametro rappresenta una access control list associata ad ogni nodo. Infatti, le informazioni presenti in ogni nodo del DataTree possono essere accedute in lettura o scrittura, e tale parametro acl permette di stabilire chi può fare cosa sui dati presenti nel nodo. Il sistema mette a disposizione delle specifiche ACL che prevedono specifici permessi e sono: CREATOR_ALL_ACL, OPEN_ACL_UNSAFE e READ_ACL_UNSAFE. Inoltre, sono stati realizzati dei test con la presenza di tale lista posta a null e vuota.
- **ephemeralOwner:** In ZooKeeper, dalla documentazione si scopre che vi è il concetto di nodo effimero, ovvero si possono creare dei nodi che rimangono in vita fin tanto che la sessione, in cui sono stati creati, è attiva. Dalla documentazione si è scoperto che i nodi effimeri non possono avere nodi figli. Grazie a questo parametro ed a un'ulteriore proprietà del sistema (`<code>zookeeper.extendedTypesEnabled</code>`) si possono distinguere diverse tipologie di nodi nel sistema: nodi effimeri, nodi NON effimeri, nodi TTL e nodi CONTAINER. Ad esempio, si può creare un nodo NON effimero dando a questo campo il valore 0, uno effimero passando l'ID della session ed, infine, utilizzando come ephemeralOwner il valore 0xff0000000000000001 si può creare un nodo TTL, in cui gli ultimi 5 bytes indicano la durata del TTL in ms (in questo caso 1ms), oppure utilizzando il valore 0x8000000000000000, si può creare un nodo CONTAINER.
- **parentCVersion:** rappresenta il numero di modifiche fatte sui figli di un nodo del DataTree.
- **zxid:** rappresenta il *zookeeper transaction id*, infatti, ogni modifica effettuata ha associato un zxid e se ad esempio si hanno un zxid1 ed un zxid2 (con zxid1 < zxid2), significa che la modifica associata a zxid1 è avvenuta prima di quella associata a zxid2.

Parametro	Boundaries
String path	null, "/zn1", "/zn1/zn2", ".zn1", "zn1", "zn1/zn2", "/", ""
byte[] data	null, [], [d1, d2, ..., dN], [d1, d2, ..., dN, ..dm]
List<ACL> acl	null, [], CREATOR_ALL_ACL, OPEN_ACL_UNSAFE, READ_ACL_UNSAFE.
long ephemeralOwner	-1, 0, 1, 0xff00000000000001, 0x8000000000000000
int parentCVersion	-2, -1, 0, 1, 3
long zxid	-1, 0, 1, 5000
long time	-1, 0, 1, 50000

Essendo presenti diversi parametri con diverse possibili classi di equivalenza, è stato deciso di optare un approccio unidimensionale. Tale metodo ha come tipo di ritorno "void", quindi, al fine di valutare l'esito dei test

realizzati si è deciso di controllare quali eccezioni ci si aspettava venissero lanciate nel caso di creazione di un nodo con dei campi non validi, mentre nello scenario in cui ci si attendeva che la creazione avvenisse con successo, si è verificato lo stato del dataTree in termini di numero di nodi presenti, in termini di aggiornamento delle informazioni (valore campo CVersion) e se un certo nuovo nodo risultava essere inserito correttamente come figlio di un certo nodo "parent". I valori di output attesi dai test progettati sono:

path	data	acl	ephemeral Owner	parentC Version	zxid	time	Output/Descrizione
"/zn1"	[d0,..., dN]	ZooDefs.Ids.CREATOR_ALL_ACL	0	3	1	1	Si attende che il nodo NON effimero venga inserito correttamente nel DataTree ed il valore del Cversion del padre deve essere messo a 3.
"/zn2"	[d0,..., dN,.dm]	ZooDefs.Ids.OPEN_ACL_UNSAFE	1	0	1	1	Si attende che il nodo effimero non venga inserito e che venga lanciata un'eccezione legata all'uso di un array con una dimensione maggiore di 1MB.
"/zn2"	[d0,..., dN]	ZooDefs.Ids.OPEN_ACL_UNSAFE	1	0	1	1	Si attende che il nodo effimero venga inserito nel DataTree ed il valore del Cversion del padre deve rimanere a 0 ed il numero di nodi effimeri presenti deve essere pari ad 1.
"/zn3"	[]	ZooDefs.Ids.CREATOR_ALL_ACL	0	-2	1	1	Si attende che il nodo NON effimero con un array vuoto venga correttamente inserito nel DataTree, con il valore del Cversione del parent che non deve essere aggiornato perché il nodo figlio ha Cversion <0.
"/zn4"	null	ZooDefs.Ids.READ_ACL_UNSAFE	1	0	1	1	Si attende che il nodo effimero con un array di dati null e con un path valido venga inserito correttamente ed il numero di nodi effimeri presenti deve essere pari ad 1.
"/zn5"	null	ZooDefs.Ids.READ_ACL_UNSAFE	1	-1	1	1	Si attende che il nodo effimero con un array vuoto venga correttamente inserito nel DataTree, con il valore del Cversione del parent che non deve essere aggiornato perché il nodo figlio ha Cversion <0.
"/zn6/zn7"	[d0,..., dN]	ZooDefs.Ids.READ_ACL_UNSAFE	0	0	1	1	Si attende che l'inserimento del nodo non effimero vada a buon fine, grazie alla precedente creazione del nodo /zn6
"zn1"	[d0,..., dN]	[]	0	0	0	0	Si attende che il nodo NON effimero NON venga inserito nel DataTree a causa del path invalido
"zn1/zn2"	[d0,..., dN]	null	0xff000000 00000001L	0	-1	1	Si attende che il nodo TTL con un path invalido NON venga inserito nel DataTree e che venga lanciata un'eccezione di tipo NoNodeException
../zn1	null	ZooDefs.Ids.OPEN_ACL_	0x80000000 0000000000	0	1	50000	Si attende che il nodo CONTAINER con path invalido NON venga inserito

		UNSAFE	L				nel DataTree e che venga lanciata un'eccezione NoNodeException
null	[d0,..., dN]	ZooDefs.Ids.CREATOR_ALL_ACL	-1	-1	5000	1	Si attende che il nodo non venga inserito perché il path è null
""	[d0,..., dN]	ZooDefs.Ids.READ_ACL_UNSAFE	0	1	1	-1	Si attende che il nodo non venga inserito perché il path è invalido
"/zn1/zn2"	[d0,..., dN]	null	0	0	1	1	Si attende che venga sollevata l'eccezione NoNodeException in quanto si tenta di inserire il nodo /zn1/zn2 senza creare prima il nodo /zn1
"/.zn1/.zn2"	[d0,..., dN]	ZooDefs.Ids.OPEN_ACL_UNSAFE	0	0	1	1	Si configura il nodo padre (/zn1) come un nodo effimero e si cerca di inserire un nodo figlio (.zn2) NON effimero. Mi aspetto che il test fallisca perché i nodi effimeri non possono avere figli
"/.zn1/.zn2"	[d0,..., dN]	ZooDefs.Ids.OPEN_ACL_UNSAFE	1	0	1	1	Si configura il nodo padre (/zn1) come un nodo effimero e si cerca di inserire un nodo figlio (.zn2) effimero. Mi aspetto che il test fallisca perché i nodi effimeri non possono avere figli

Dalla precedente test suite si è osservato che il secondo test non genera un errore a seguito dell'uso di una array data di dimensioni maggiori di 1MB a differenza di quanto ci si potesse aspettare. Inoltre, anche gli ultimi due test, che prevedono la creazione di un nodo figlio (/zn2) dopo aver creato un nodo parent effimero (/zn1), vanno a buon fine anziché fallire come si prevedeva, in quanto un nodo effimero NON può avere figli. Il motivo per cui ciò accade, si pensa poter essere legato alla mancanza di controlli su questi parametri all'interno del metodo, ovvero potrebbe darsi che tale metodo venga acceduto secondo un preciso flusso che include eventuali controlli preliminari sui parametri passati in input. Dunque, essendo questo un metodo pubblico, l'accedere direttamente a questo metodo nei test, può aver fatto sì che alcuni controlli preliminari venissero saltati. Grazie ai precedenti test, come si può vedere dalla [figura_8b](#), si è ottenuta una statement coverage del 86% ed una branch coverage del 71%. Per quanto riguarda il report generato da ba-dua, come si vede in [figura_9b](#), si hanno 20 coppie def-use non coperte su 110. Utilizzando PIT, nella [figura_10b](#) e [figura_11b](#), si può notare la presenza di diversi mutanti ancora presenti. Non essendo soddisfatto dei risultati ottenuti si è cercato di progettare altri casi di test. In particolare, sono stati realizzati dei test in cui si prova ad inserire un nodo già presente nel DataTree verificando che venga sollevata l'eccezione NodeExistsException. Inoltre, sono stati progettati dei casi di test che includono la creazione ed inserimento di nodi CONTAINER e nodi TTL, oltre alla creazione di nodi legati all'uso di Quotas all'interno del path. Sono stati aggiunti i seguenti test:

path	data	acl	ephemeral Owner	parentC Version	zxid	time	Output/Descrizione
"/zn10"	[d0,..., dN]	ZooDefs.Ids.OPEN_ACL_UNSAFE	0	0	1	1	Si è provato ad inserire due volte consecutive questo nodo e si è ottenuta l'eccezione NodeExistsException
"/zn.1/zn.2"	[d0,..., dN]	ZooDefs.Ids.CREATOR_ALL_ACL	0xffff0000000000001L	0	1	1	Si attende il nodo TTL venga inserito correttamente nel DataTree dopo aver inserito in precedenza il nodo "/zn.1"

"/zn5/zn6/zn7/zn8/ zn9"	[d0,..., dN]	ZooDefs.Ids .READ_AC L_UNSAFE	0x8000000 000000000 L	0	0	1	Si attende il nodo CONTAINER venga inserito correttamente nel DataTree dopo aver inserito correttamente in precedenza i precedenti nodi
Quotas.quotaZookeeper/parent/ Quotas.limitNode"	[d0,..., dN]	null	0	0	1	0	Si attende che il nodo "Quotas.limitNode" venga creato dopo aver inserito correttamente i nodi che lo precedono
Quotas.quotaZookeeper/parent/ Quotas.statNode	[d0,..., dN]	null	0	0	1	0	Si attende che il nodo "Quotas.statNode" venga creato dopo aver inserito correttamente i nodi che lo precedono
"/parent/ Quotas.limitNode"	[d0,..., dN]	null	0	0	1	0	Si attende che il nodo "Quotas.limitNode" venga creato dopo aver inserito correttamente il nodo che lo precede

Dopo aver aggiunto tali test, utilizzando JaCoCo, come si può vedere dalal [figura_12b](#), si è ottenuta una statement coverage del 97% ed una branch coverage del 96%, mentre per quanto riguarda il report generato da ba-dua, come si vede in [figura_13b](#), tramite questi test si è riuscito a ridurre il numero di coppie def-use non coperte a 5 su 110. Utilizzando PIT, come si vede dalla [figura_14b](#) e [figura_14.1b](#), si è riuscito ad eliminare i mutanti presenti nelle righe e 504 grazie al penultimo test nella tabella ed il mutante di riga 502 grazie all'ultimo test in tabella che permettono di risolvere le negazioni delle condizioni.

Per quanto riguarda il metodo:

- public void **deleteNode** (String path, long zxid) throws NoNodeException;

Tale metodo permette di rimuovere nodo associato al path dal dataTree.

Domain Partitioning

Parametro	Classi di equivalenza
String path	{null},{valid_string},{invalid_string},{"")}
long zxid	{≤0}, {>0}

Boundary-values

Parametro	Boundaries
String path	Null, "/zn1/zn2/zn3/zn4/zn5", "/zn1/zn2/.", ""
long zxid	-1234567, -1, 0, 1, 1000000

Essendo un metodo che ritorna void, al fine di verificare l'esito dei test, si può andare a controllare la dimensione del dataTree a seguito di un cancellazione oppure verificare che venga lanciata l'eccezione attesa. Per l'individuazione dei casi di test è stato utilizzato un approccio unidimensionale in cui sono stati creati dei test che vanno ad eliminare un nodo che è effettivamente presente nel dataTree e si verifica se il valore del parametro zxid, passato in input, è strettamente maggiore del valore di pzxid del nodo genitore. Il valore pzxid rappresenta lo zxid dell'ultima modifica fatta sui figli di uno ZNode. Nel caso in cui è strettamente maggiore, allora pzxid deve essere aggiornato, Inoltre sono stati realizzati dei test in cui si va ad eliminare un nodo che non è presente (ci si aspetta che venga lanciata l'eccezione NoNodeException). Per realizzare questi test è necessario andare a creare in precedenza il dataTree in modo corretto con il metodo *createNode(...)*, e solo una volta fatto ciò si può provare ad invocare il metodo di *deleteNode(...)*. Ciascuno dei primi 5 test realizzati è stato utilizzato per testare l'eliminazione di un nodo NON effimero, effimero, TTL e container. Ovvero, ciascuno di questi path è stato utilizzato per creare un diverso dataTree in cui i precedenti n-1 nodi nel path sono NON effimeri mentre l'ultimo nodo viene aggiunto come un nodo NON effimero o come nodo effimero o come nodo TTL o come nodo container. I valori di output attesi dai test progettati sono:

path	zxicd	Output/Descrizione
/zn1/zn2/zn3/zn4/zn5	-1	Si attende che il nodo /zn5 venga eliminato dopo che sia esso che i nodi che lo precedono sono stati inseriti. Dopo l'eliminazione il numero di nodi deve essere pari a 5 ed inoltre il valore del pzxid NON deve essere aggiornato perché il valore dello zxicd è < 0.
/zn.1/zn.2	1	Si attende che il nodo /zn.2 venga eliminato dopo che sia esso che il nodo che lo precede sono stati inseriti. Dopo l'eliminazione il numero di nodi deve essere pari a 2 ed inoltre il valore del pzxid deve essere aggiornato perché il valore dello zxicd è pari a 1.
/zn1/zn2	0	Si attende che il nodo /zn2 venga eliminato dopo che sia esso che il nodo che lo precede sono stati inseriti. Dopo l'eliminazione il numero di nodi deve essere pari ad 2 ed inoltre il valore del pzxid NON deve essere aggiornato perché il valore dello zxicd è uguale a quello del pzxid
/zn1	1000000	Si attende che il nodo /zn1 venga eliminato correttamente dopo che è stato inserito. Il numero di nodi deve essere 1 (nodo root /) Inoltre il valore del pzxid deve essere aggiornato.
/zn1/zn2	-1234567	Si attende che il nodo /zn2 venga eliminato dopo che sia esso che il nodo che lo precede sono stati inseriti. Il numero di nodi deve essere 2. Inoltre il valore del pzxid NON deve essere aggiornato perché il valore dello zxicd è < 0.
/zn1	1	Si attende che il test fallisca con il sollevamento dell'eccezione NoNodeException in quanto si prova ad eliminare un nodo (/zn1) che non è presente nel dataTree
null	0	Si attende che il metodo sia in grado di gestire il valore null lanciando un'eccezione
""	-1	Si attende che il metodo sia in grado di gestire un path vuoto lanciando un'eccezione

L'esecuzione di questi test ci permette di ottenere delle percentuali non soddisfacenti di statement coverage e branch coverage, come si può vedere dalla [figura_15b](#) e [figura_16b](#). Mentre, per quanto riguarda il report di ba-dua, come si vede in [figura_17b](#), si può vedere come vi siano 21 coppie def-use su 117 non coperte. Vedendo il report di JaCoCo è sono stati realizzati ulteriori casi di test in cui viene eliminato un nodo che non ha un nodo genitore. Questo caso si può manifestare nel momento in cui viene eliminato prima il nodo genitore e poi si tenta di eliminare il nodo figlio. In questo ultimo caso di test, mi aspetto che il metodo lanci una eccezione perché non è in grado di trovare il nodo genitore. Inoltre è stato realizzato un test in cui si va ad eliminare un nodo che ha come nodo padre "Quotas.procZookeeper". Sono stati aggiunti i seguenti test:

path	zxicd	Output/Descrizione
"/zn1/zn2/zn3/zn4/zn5/zn6"	1	Si attende che venga lanciata un'eccezione di tipo NoNodeException perché si prova ad eliminare il nodo /zn6 dopo aver eliminato il nodo padre /zn5
/zn1/zn2/zn3	1000000	Si attende che venga lanciata un'eccezione di tipo NoNodeException perché si prova ad eliminare il nodo /zn3 dopo aver eliminato il nodo padre /zn2
/zn1/zn2	0	Si attende che venga lanciata un'eccezione di tipo NoNodeException perché si prova ad eliminare il nodo /zn2 dopo aver eliminato il nodo padre /zn1
"Quotas.procZookeeper/parent/Quotas.limitNode"	1	Si attende che il nodo /Quotas.limitNode venga eliminato dopo averlo in precedenza aggiunto correttamente insieme ai nodi che lo precedono
"Quotas.procZookeeper/parent/zn1"	1	Si attende che il nodo /zn1 venga eliminato dopo averlo in precedenza aggiunto correttamente insieme ai nodi che lo precedono

Aggiungendo questi test, come si vede dalla [figura_18b](#) e [figura_19b](#), si ottengono dei risultati di statement coverage del 90% e branch coverage del 80%. Per quanto riguarda il report generato da ba-dua, grazie ai precedenti test, si è riuscito a ridurre le coppie def-use non coperte fino a 14 su 117, come si vede dalla [figura_20b](#). Dal report di PIT ([figura_21b](#)), invece, si può notare come vi siano diversi mutanti NON KILLED.

Al fine di aumentare la mutation coverage sono stati aggiunti i seguenti test:

path	zxicd	Output/Descrizione
"/zn1/Quotas.limitNode"	1	Si attende che il nodo /Quotas.limitNode venga eliminato dopo averlo in precedenza aggiunto correttamente insieme ai nodi che lo precedono
"Quotas.procZookeeper/zn1"	1	Si attende che il nodo /zn1 venga eliminato e che il numero di nodi presenti sia pari a 1

"/zn1/zn2/zn3"	1	Si attende che il nodo /zn3 (creato come unico nodo container) venga eliminato e che il numero di nodi container sia nullo
----------------	---	--

Questi test permettono di rimuovere i due mutanti a riga 588 ed il mutante di riga 574 rispettivamente, come si vede in [figura_22b](#).

ZKUtil.java

La classe ZKUtil.java contiene delle funzionalità necessarie a garantire il corretto funzionamento della struttura gerarchica di ZooKeeper. In particolare, contiene metodi per l'attraversamento, la cancellazione e validazione del sistema, a partire da uno specifico pathRoot, per tutto il subtree di uno znode. Inoltre, contiene dei metodi necessari alla gestione delle ACL. In questa classe è stato utilizzato il framework Mockito per la realizzazione di alcuni test.

Metodi

- public static List<String> **listSubTreeBFS**(ZooKeeper zk, final String pathRoot) throws KeeperException, InterruptedException;

Tale metodo permette di realizzare una visita BFS dell'albero presente sotto il pathRoot, con le entry che vengono messe nella lista di ritorno nello stesso ordine con cui vengono visitate.

Domain Partitioning

Parametro	Classi di equivalenza
ZooKeeper zk	{null}, {valid_instance}, {invalid_instance}
String pathRoot	{null}, {valid_string}, {invalid_string}, {"("")"}

Boundary-values

Parametro	Boundaries
ZooKeeper zk	Null, valid_instance, invalid_instance
String pathRoot	"/znRoot", "/\uFFFF", ""

Essendo presenti solo due parametri con diverse possibili partizioni/classi di equivalenza, è stato deciso di optare un approccio multidimensionale. Per la realizzazione dei test si è verificato che il risultato dell'attraversamento del DataTree a partire da una certa pathRoot fosse uguale a quello ci si attendeva con certi parametri di input. Tali test sono stati realizzati facendo uso del framework Mockito, utilizzando questo per creare un'istanza valida ed invalida della classe ZooKeeper e realizzando un'implementazione differente del metodo getChildren che permette di ritornare, all'interno di una lista, i nodi presenti nel DataTree a partire da una certa pathRoot. È stata utilizzata tale lista per confrontarla con il result della chiamata del metodo che si sta testando. Per testare questo metodo è stata realizzata una versione dummy di createNode che non effettua controlli sulla validità del path inserito. I valori di output attesi dai test progettati sono:

zk	pathRoot	Descrizione
Null	""	Si attende un'eccezione di tipo NullPointerException in quanto l'istanza zk è nulla
Null	"/znRoot"	Si attende un'eccezione di tipo NullPointerException in quanto l'istanza zk è nulla
Null	"/\uFFF"	Si attende un'eccezione di tipo NullPointerException in quanto l'istanza zk è nulla
Valid	""	Si attende che il test vada a buon fine, ritornando la corretta lista di nodi visitati
Valid	"/znRoot"	Si attende che il test vada a buon fine, ritornando la corretta lista di nodi visitati
Valid	"/\uFFF"	Si attende che il test vada a buon fine, ritornando la corretta lista di nodi visitati
Invalid	""	Si attende un'eccezione di tipo NullPointerException in quanto l'istanza zk è invalida
Invalid	"/znRoot"	Si attende un'eccezione di tipo NullPointerException in quanto l'istanza zk è invalida
Invalid	"/\uFFF"	Si attende un'eccezione di tipo NullPointerException in quanto l'istanza zk è invalida

Dai precedenti test, eseguendo JaCoCo, come si può vedere dalla [figura_23b](#), si ottiene una statement coverage del 100% ed una branch coverage del 100%. Per quanto riguarda il report generato da ba-dua ([figura_24b](#)), si può osservare con non vi siano coppie def-use NON coperte. Utilizzando PIT, come si vede dalla [figura_25b](#), si può osservare come non vi siano mutanti.

Allegati

figura_1a

BufferedChannel

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxt	Missed	Lines	Missed	Methods
• forceWrite(boolean)		0%		0%	2	2	7	7	1	1
• write(ByteBuf)		74%		60%	3	6	5	21	0	1

figura_2a

```
117.     public void write(ByteBuf src) throws IOException {
118.         int copied = 0;
119.         boolean shouldForceWrite = false;
120.         synchronized (this) {
121.             int len = src.readableBytes();
122.             while (copied < len) {
123.                 int bytesToCopy = Math.min(src.readableBytes() - copied, writeBuffer.writableBytes());
124.                 writeBuffer.writeBytes(src, src.readerIndex() + copied, bytesToCopy);
125.                 copied += bytesToCopy;
126.
127.                 // if we have run out of buffer space, we should flush to the
128.                 // file
129.                 if (!writeBuffer.isWritable()) {
130.                     flush();
131.                 }
132.             }
133.             position += copied;
134.             if (doRegularFlushes) {
135.                 unpersistedBytes.addAndGet(copied);
136.                 if (unpersistedBytes.get() >= unpersistedBytesBound) {
137.                     flush();
138.                     shouldForceWrite = true;
139.                 }
140.             }
141.             if (shouldForceWrite) {
142.                 forceWrite(false);
143.             }
144.         }
145.     }
```

figura_3a

```
<method name="writeI" desc="(Io/netty/buffer/ByteBuf)V">
<du var="this" def="118" use="133" covered="1"/>
<du var="this" def="118" use="134" target="135" covered="0"/>
<du var="this" def="118" use="134" target="141" covered="1"/>
<du var="this" def="118" use="143" covered="0"/>
<du var="this" def="118" use="135" covered="0"/>
<du var="this" def="118" use="136" target="137" covered="0"/>
<du var="this" def="118" use="136" target="141" covered="0"/>
<du var="this" def="118" use="137" covered="0"/>
<du var="this" def="118" use="123" covered="1"/>
<du var="this" def="118" use="124" covered="1"/>
<du var="this" def="118" use="129" target="130" covered="1"/>
<du var="this" def="118" use="129" target="132" covered="1"/>
<du var="this" def="118" use="130" covered="1"/>
<du var="src" def="118" use="123" covered="1"/>
<du var="src" def="118" use="124" covered="1"/>
<du var="this.writeBuffer" def="118" use="123" covered="1"/>
<du var="this.writeBuffer" def="118" use="124" covered="1"/>
<du var="this.writeBuffer" def="118" use="129" target="130" covered="1"/>
<du var="this.writeBuffer" def="118" use="129" target="132" covered="1"/>
<du var="this.position" def="118" use="133" covered="1"/>
<du var="this.doRegularFlushes" def="118" use="134" target="135" covered="0"/>
<du var="this.doRegularFlushes" def="118" use="134" target="141" covered="1"/>
<du var="this.unpersistedBytes" def="118" use="135" covered="0"/>
<du var="this.unpersistedBytes" def="118" use="136" target="137" covered="0"/>
<du var="this.unpersistedBytes" def="118" use="136" target="141" covered="0"/>
<du var="copied" def="118" use="122" target="123" covered="1"/>
<du var="copied" def="118" use="122" target="133" covered="1"/>
<du var="copied" def="118" use="133" covered="1"/>
<du var="copied" def="118" use="135" covered="0"/>
<du var="copied" def="118" use="123" covered="1"/>
<du var="copied" def="118" use="124" covered="1"/>
<du var="copied" def="118" use="125" covered="1"/>
<du var="shouldForceWrite" def="119" use="142" target="143" covered="0"/>
<du var="shouldForceWrite" def="119" use="142" target="145" covered="1"/>
<du var="len" def="121" use="122" target="123" covered="1"/>
<du var="len" def="121" use="122" target="133" covered="1"/>
<du var="copied" def="125" use="122" target="123" covered="1"/>
<du var="copied" def="125" use="122" target="133" covered="1"/>
<du var="copied" def="125" use="133" covered="1"/>
<du var="copied" def="125" use="135" covered="0"/>
<du var="copied" def="125" use="123" covered="1"/>
<du var="copied" def="125" use="124" covered="1"/>
<du var="copied" def="125" use="125" covered="1"/>
<du var="shouldForceWrite" def="138" use="142" target="143" covered="0"/>
<du var="shouldForceWrite" def="138" use="142" target="145" covered="0"/>
<counter type="DU" missed="17" covered="31"/>
<counter type="METHOD" missed="0" covered="1"/>
</method>
```

figura_4a

• close()	93%	50%	1	2	1	6	0	1
• write(ByteBuf)	100%	100%	0	6	0	21	0	1
• BufferedChannel(ByteBufAllocator, FileChannel, int, int, long)	100%	100%	0	2	0	11	0	1

figura_5a

```

-<method name="write" desc="(Lio/netty/buffer/ByteBuf;)V">
<du var="this" def="118" use="133" covered="1"/>
<du var="this" def="118" use="134" target="135" covered="1"/>
<du var="this" def="118" use="134" target="141" covered="1"/>
<du var="this" def="118" use="143" covered="1"/>
<du var="this" def="118" use="135" covered="1"/>
<du var="this" def="118" use="136" target="137" covered="1"/>
<du var="this" def="118" use="136" target="141" covered="1"/>
<du var="this" def="118" use="137" covered="1"/>
<du var="this" def="118" use="123" covered="1"/>
<du var="this" def="118" use="124" covered="1"/>
<du var="this" def="118" use="129" target="130" covered="1"/>
<du var="this" def="118" use="129" target="132" covered="1"/>
<du var="this" def="118" use="130" covered="1"/>
<du var="src" def="118" use="123" covered="1"/>
<du var="src" def="118" use="124" covered="1"/>
<du var="this.writeBuffer" def="118" use="123" covered="1"/>
<du var="this.writeBuffer" def="118" use="124" covered="1"/>
<du var="this.writeBuffer" def="118" use="129" target="130" covered="1"/>
<du var="this.writeBuffer" def="118" use="129" target="132" covered="1"/>
<du var="this.position" def="118" use="133" covered="1"/>
<du var="this.doRegularFlushes" def="118" use="134" target="135" covered="1"/>
<du var="this.doRegularFlushes" def="118" use="134" target="141" covered="1"/>
<du var="this.unpersistedBytes" def="118" use="135" covered="1"/>
<du var="this.unpersistedBytes" def="118" use="136" target="137" covered="1"/>
<du var="this.unpersistedBytes" def="118" use="136" target="141" covered="1"/>
<du var="copied" def="118" use="122" target="123" covered="1"/>
<du var="copied" def="118" use="122" target="133" covered="1"/>
<du var="copied" def="118" use="133" covered="1"/>
<du var="copied" def="118" use="135" covered="0"/>
<du var="copied" def="118" use="123" covered="1"/>
<du var="copied" def="118" use="124" covered="1"/>
<du var="copied" def="118" use="125" covered="1"/>
<du var="shouldForceWrite" def="119" use="142" target="143" covered="0"/>
<du var="shouldForceWrite" def="119" use="142" target="145" covered="1"/>
<du var="len" def="121" use="122" target="123" covered="1"/>
<du var="len" def="121" use="122" target="133" covered="1"/>
<du var="copied" def="125" use="122" target="123" covered="1"/>
<du var="copied" def="125" use="122" target="133" covered="1"/>
<du var="copied" def="125" use="133" covered="1"/>
<du var="copied" def="125" use="135" covered="1"/>
<du var="copied" def="125" use="123" covered="1"/>
<du var="copied" def="125" use="124" covered="1"/>
<du var="copied" def="125" use="125" covered="1"/>
<du var="shouldForceWrite" def="138" use="142" target="143" covered="1"/>
<du var="shouldForceWrite" def="138" use="142" target="145" covered="0"/>
<counter type="DU" missed="3" covered="45"/>
<counter type="METHOD" missed="0" covered="1"/>
</method>
```

figura_6a

```

117     public void write(ByteBuf src) throws IOException {
118         int copied = 0;
119         boolean shouldForceWrite = false;
120         synchronized (this) {
121             int len = src.readableBytes();
122             while (copied < len) {
123                 int bytesToCopy = Math.min(src.readableBytes() - copied, writeBuffer.writableBytes());
124                 writeBuffer.writeBytes(src, src.readerIndex() + copied, bytesToCopy);
125                 copied += bytesToCopy;
126
127                 // if we have run out of buffer space, we should flush to the
128                 // file
129                 if (!writeBuffer.isWritable()) {
130                     flush();
131                 }
132             }
133             position += copied;
134             if (doRegularFlushes) {
135                 unpersistedBytes.addAndGet(copied);
136                 if (unpersistedBytes.get() >= unpersistedBytesBound) {
137                     flush();
138                     shouldForceWrite = true;
139                 }
140             }
141         }
142         if (shouldForceWrite) {
143             forceWrite(false);
144         }
145     }
***
```

figura_7a

Apache BookKeeper :: Tests :: Jacoco Aggregation > bookkeeper-server > org.apache.bookkeeper.bookie > BufferedChannel

BufferedChannel

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods
read(ByteBuf, long, int)	██████████	79%	██████████	61%	6	10	5	26	0	1
flushAndForceWrite(boolean)	█	0%		n/a	1	1	3	3	1	1
flushAndForceWriteIfRegularFlush(boolean)	█	0%	█	0%	2	2	3	3	1	1

figura_8a

```

244.     @Override
245.     public synchronized int read(ByteBuf dest, long pos, int length) throws IOException {
246.         long prevPos = pos;
247.         while (length > 0) {
248.             // check if it is in the write buffer
249.             if (writeBuffer != null && writeBufferStartPosition.get() <= pos) {
250.                 int positionInBuffer = (int) (pos - writeBufferStartPosition.get());
251.                 int bytesToCopy = Math.min(writeBuffer.writerIndex() - positionInBuffer, dest.writableBytes());
252.
253.                 if (bytesToCopy == 0) {
254.                     throw new IOException("Read past EOF");
255.                 }
256.
257.                 dest.writeBytes(writeBuffer, positionInBuffer, bytesToCopy);
258.                 pos += bytesToCopy;
259.                 length -= bytesToCopy;
260.             } else if (writeBuffer == null && writeBufferStartPosition.get() <= pos) {
261.                 // here we reach the end
262.                 break;
263.             // first check if there is anything we can grab from the readBuffer
264.             } else if (readBufferStartPosition <= pos && pos < readBufferStartPosition + readBuffer.writerIndex()) {
265.                 int positionInBuffer = (int) (pos - readBufferStartPosition);
266.                 int bytesToCopy = Math.min(readBuffer.writerIndex() - positionInBuffer, dest.writableBytes());
267.                 dest.writeBytes(readBuffer, positionInBuffer, bytesToCopy);
268.                 pos += bytesToCopy;
269.                 length -= bytesToCopy;
270.                 // let's read it
271.             } else {
272.                 readBufferStartPosition = pos;
273.
274.                 int readBytes = fileChannel.read(readBuffer.internalNioBuffer(0, readCapacity),
275.                                                 readBufferStartPosition);
276.                 if (readBytes <= 0) {
277.                     throw new IOException("Reading from filechannel returned a non-positive value. Short read.");
278.                 }
279.                 readBuffer.writerIndex(readBytes);
280.             }
281.         }
282.         return (int) (pos - prevPos);
283.     }

```

figura_9a

Apache BookKeeper :: Tests :: Jacoco Aggregation > bookkeeper-server > org.apache.bookkeeper.bookie > BufferedChannel

BufferedChannel

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods
read(ByteBuf, long, int)	██████████	91%	██████████	66%	5	10	2	26	0	1
flushAndForceWrite(boolean)	█	0%		n/a	1	1	3	3	1	1
flushAndForceWriteIfRegularFlush(boolean)	█	0%	█	0%	2	2	3	3	1	1
getFileChannelPosition()	█	0%		n/a	1	1	1	1	1	1

figura_10a

```

244.     @Override
245.     public synchronized int read(ByteBuf dest, long pos, int length) throws IOException {
246.         long prevPos = pos;
247.         while (length > 0) {
248.             // check if it is in the write buffer
249.             if (writeBuffer != null && writeBufferStartPosition.get() <= pos) {
250.                 int positionInBuffer = (int) (pos - writeBufferStartPosition.get());
251.                 int bytesToCopy = Math.min(writeBuffer.writerIndex() - positionInBuffer, dest.writableBytes());
252.
253.                 if (bytesToCopy == 0) {
254.                     throw new IOException("Read past EOF");
255.                 }
256.
257.                 dest.writeBytes(writeBuffer, positionInBuffer, bytesToCopy);
258.                 pos += bytesToCopy;
259.                 length -= bytesToCopy;
260.             } else if (writeBuffer == null && writeBufferStartPosition.get() <= pos) {
261.                 // here we reach the end
262.                 break;
263.             // first check if there is anything we can grab from the readBuffer
264.             } else if (readBufferStartPosition <= pos && pos < readBufferStartPosition + readBuffer.writerIndex()) {
265.                 int positionInBuffer = (int) (pos - readBufferStartPosition);
266.                 int bytesToCopy = Math.min(readBuffer.writerIndex() - positionInBuffer, dest.writableBytes());
267.                 dest.writeBytes(readBuffer, positionInBuffer, bytesToCopy);
268.                 pos += bytesToCopy;
269.                 length -= bytesToCopy;
270.                 // let's read it
271.             } else {
272.                 readBufferStartPosition = pos;
273.
274.                 int readBytes = fileChannel.read(readBuffer.internalNioBuffer(0, readCapacity),
275.                                                 readBufferStartPosition);
276.                 if (readBytes <= 0) {
277.                     throw new IOException("Reading from filechannel returned a non-positive value. Short read.");
278.                 }
279.                 readBuffer.writerIndex(readBytes);
280.             }
281.         }
282.         return (int) (pos - prevPos);
283.     }

```

figura_11a

```
<du var="pos" def="258" use="268" covered="0"/>
<du var="pos" def="258" use="260" target="262" covered="0"/>
<du var="pos" def="258" use="260" target="264" covered="0"/>
<du var="pos" def="258" use="249" target="250" covered="1"/>
<du var="pos" def="258" use="249" target="260" covered="0"/>
<du var="pos" def="258" use="250" covered="1"/>
<du var="pos" def="258" use="258" covered="0"/>
<du var="length" def="259" use="247" target="249" covered="1"/>
<du var="length" def="259" use="247" target="282" covered="0"/>
<du var="length" def="259" use="269" covered="0"/>
<du var="length" def="259" use="259" covered="0"/>
<du var="pos" def="268" use="282" covered="1"/>
<du var="pos" def="268" use="264" target="264" covered="1"/>
<du var="pos" def="268" use="264" target="272" covered="0"/>
<du var="pos" def="268" use="272" covered="1"/>
<du var="pos" def="268" use="264" target="265" covered="1"/>
<du var="pos" def="268" use="264" target="272" covered="1"/>
<du var="pos" def="268" use="265" covered="1"/>
<du var="pos" def="268" use="268" covered="1"/>
<du var="pos" def="268" use="260" target="262" covered="0"/>
<du var="pos" def="268" use="260" target="264" covered="0"/>
<du var="pos" def="268" use="249" target="250" covered="1"/>
<du var="pos" def="268" use="249" target="260" covered="1"/>
<du var="pos" def="268" use="250" covered="1"/>
<du var="length" def="269" use="247" target="249" covered="1"/>
<du var="length" def="269" use="247" target="282" covered="1"/>
<du var="length" def="269" use="269" covered="1"/>
<du var="length" def="269" use="259" covered="0"/>
<du var="this.readBufferStartPosition" def="272" use="264" target="264" covered="1"/>
<du var="this.readBufferStartPosition" def="272" use="264" target="272" covered="0"/>
<du var="this.readBufferStartPosition" def="272" use="264" target="265" covered="1"/>
<du var="this.readBufferStartPosition" def="272" use="264" target="272" covered="1"/>
<du var="this.readBufferStartPosition" def="272" use="265" covered="1"/>
<du var="readBytes" def="274" use="276" target="277" covered="0"/>
<du var="readBytes" def="274" use="276" target="279" covered="1"/>
<du var="readBytes" def="274" use="279" covered="1"/>
<counter type="DU" missed="37" covered="81"/>
<counter type="METHOD" missed="0" covered="1"/>
</method>
```

figura_12a

```
244     @Override
245     public synchronized int read(ByteBuf dest, long pos, int length) throws IOException {
246         long prevPos = pos;
247         while (length > 0) {
248             // check if it is in the write buffer
249             if (writeBuffer != null && writeBufferStartPosition.get() <= pos) {
250                 int positionInBuffer = (int) (pos - writeBufferStartPosition.get());
251                 int bytesToCopy = Math.min(writeBuffer.writerIndex() - positionInBuffer, dest.writableBytes());
252
253                 if (bytesToCopy == 0) {
254                     throw new IOException("Read past EOF");
255                 }
256
257                 dest.writeBytes(writeBuffer, positionInBuffer, bytesToCopy);
258                 pos += bytesToCopy;
259                 length -= bytesToCopy;
260             } else if (writeBuffer == null && writeBufferStartPosition.get() <= pos) {
261                 // here we reach the end
262                 break;
263             // first check if there is anything we can grab from the readBuffer
264             } else if (readBufferStartPosition <= pos && pos < readBufferStartPosition + readBuffer.writerIndex()) {
265                 int positionInBuffer = (int) (pos - readBufferStartPosition);
266                 int bytesToCopy = Math.min(readBuffer.writerIndex() - positionInBuffer, dest.writableBytes());
267                 dest.writeBytes(readBuffer, positionInBuffer, bytesToCopy);
268                 pos += bytesToCopy;
269                 length -= bytesToCopy;
270             // let's read it
271             } else {
272                 readBufferStartPosition = pos;
273
274                 int readBytes = fileChannel.read(readBuffer.internalNioBuffer(0, readCapacity),
275                                                 readBufferStartPosition);
276                 if (readBytes <= 0) {
277                     throw new IOException("Reading from filechannel returned a non-positive value. Short read.");
278                 }
279                 readBuffer.writerIndex(readBytes);
280             }
281         }
282         return (int) (pos - prevPos);
283     }
```

figura_13a

```

244     @Override
245     public synchronized int read(ByteBuf dest, long pos, int length) throws IOException {
246         long prevPos = pos;
247         while (length > 0) {
248             // check if it is in the write buffer
249             if (writeBuffer != null && writeBufferStartPosition.get() <= pos) {
250                 int positionInBuffer = (int) (pos - writeBufferStartPosition.get());
251                 int bytesToCopy = Math.min(writeBuffer.writerIndex() - positionInBuffer, dest.writableBytes());
252
253                 if (bytesToCopy == 0) {
254                     throw new IOException("Read past EOF");
255                 }
256
257                 dest.writeBytes(writeBuffer, positionInBuffer, bytesToCopy);
258                 pos += bytesToCopy;
259                 length -= bytesToCopy;
260             } else if (writeBuffer == null && writeBufferStartPosition.get() <= pos) {
261                 // here we reach the end
262                 break;
263                 // first check if there is anything we can grab from the readBuffer
264             } else if (readBufferStartPosition <= pos && pos < readBufferStartPosition + readBuffer.writerIndex()) {
265                 int positionInBuffer = (int) (pos - readBufferStartPosition);
266                 int bytesToCopy = Math.min(readBuffer.writerIndex() - positionInBuffer, dest.writableBytes());
267                 dest.writeBytes(readBuffer, positionInBuffer, bytesToCopy);
268                 pos += bytesToCopy;
269                 length -= bytesToCopy;
270                 // let's read it
271             } else {
272                 readBufferStartPosition = pos;
273
274                 int readBytes = fileChannel.read(readBuffer.internalNioBuffer(0, readCapacity),
275                                                 readBufferStartPosition);
276                 if (readBytes <= 0) {
277                     throw new IOException("Reading from filechannel returned a non-positive value. Short read.");
278                 }
279                 readBuffer.writerIndex(readBytes);
280             }
281         }
282         return (int) (pos - prevPos);
283     }

```

figura_14a

Apache BookKeeper :: Tests :: Jacoco Aggregation > bookkeeper-server > org.apache.bookkeeper.bookie > BufferedChannel

BufferedChannel

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods
• read(ByteBuf, long, int)		95%		72%	4	10	1	26	0	1
• flushAndForceWrite(boolean)		0%		n/a	1	1	3	3	1	1
• flushAndForceWriteIfRegularFlush(boolean)		0%		0%	2	2	3	3	1	1
• getFileChannelPosition()		0%		n/a	1	1	1	1	1	1

figura_15a

```

244.     @Override
245.     public synchronized int read(ByteBuf dest, long pos, int length) throws IOException {
246.         long prevPos = pos;
247.         while (length > 0) {
248.             // check if it is in the write buffer
249.             if (writeBuffer != null && writeBufferStartPosition.get() <= pos) {
250.                 int positionInBuffer = (int) (pos - writeBufferStartPosition.get());
251.                 int bytesToCopy = Math.min(writeBuffer.writerIndex() - positionInBuffer, dest.writableBytes());
252.
253.                 if (bytesToCopy == 0) {
254.                     throw new IOException("Read past EOF");
255.                 }
256.
257.                 dest.writeBytes(writeBuffer, positionInBuffer, bytesToCopy);
258.                 pos += bytesToCopy;
259.                 length -= bytesToCopy;
260.             } else if (writeBuffer == null && writeBufferStartPosition.get() <= pos) {
261.                 // here we reach the end
262.                 break;
263.                 // first check if there is anything we can grab from the readBuffer
264.             } else if (readBufferStartPosition <= pos && pos < readBufferStartPosition + readBuffer.writerIndex()) {
265.                 int positionInBuffer = (int) (pos - readBufferStartPosition);
266.                 int bytesToCopy = Math.min(readBuffer.writerIndex() - positionInBuffer, dest.writableBytes());
267.                 dest.writeBytes(readBuffer, positionInBuffer, bytesToCopy);
268.                 pos += bytesToCopy;
269.                 length -= bytesToCopy;
270.                 // let's read it
271.             } else {
272.                 readBufferStartPosition = pos;
273.
274.                 int readBytes = fileChannel.read(readBuffer.internalNioBuffer(0, readCapacity),
275.                                                 readBufferStartPosition);
276.                 if (readBytes <= 0) {
277.                     throw new IOException("Reading from filechannel returned a non-positive value. Short read.");
278.                 }
279.                 readBuffer.writerIndex(readBytes);
280.             }
281.         }
282.         return (int) (pos - prevPos);
283.     }

```

figura_16a

```

-<method name="read" desc="(Lio/netty/buffer/ByteBuf;JI)I">
<du var="this" def="246" use="249" target="249" covered="1"/>
<du var="this" def="246" use="249" target="260" covered="0"/>
<du var="this" def="246" use="260" target="260" covered="0"/>
<du var="this" def="246" use="260" target="264" covered="1"/>
<du var="this" def="246" use="264" target="264" covered="1"/>
<du var="this" def="246" use="264" target="272" covered="0"/>
<du var="this" def="246" use="272" covered="1"/>
<du var="this" def="246" use="274" target="274" covered="1"/>
<du var="this" def="246" use="279" target="279" covered="1"/>
<du var="this" def="246" use="264" target="265" covered="1"/>
<du var="this" def="246" use="264" target="272" covered="1"/>
<du var="this" def="246" use="265" covered="1"/>
<du var="this" def="246" use="266" covered="1"/>
<du var="this" def="246" use="267" target="267" covered="1"/>
<du var="dest" def="246" use="251" covered="1"/>
<du var="dest" def="246" use="257" target="257" covered="1"/>
<du var="pos" def="246" use="282" target="282" covered="1"/>
<du var="pos" def="246" use="264" target="264" covered="1"/>
<du var="pos" def="246" use="272" target="272" covered="0"/>
<du var="pos" def="246" use="272" target="272" covered="1"/>
<du var="pos" def="246" use="264" target="265" covered="1"/>
<du var="pos" def="246" use="264" target="272" covered="1"/>
<du var="pos" def="246" use="265" target="265" covered="1"/>
<du var="pos" def="246" use="268" target="268" covered="1"/>
<du var="pos" def="246" use="260" target="262" covered="0"/>
<du var="pos" def="246" use="260" target="264" covered="0"/>
<du var="pos" def="246" use="249" target="250" covered="1"/>
<du var="pos" def="246" use="249" target="260" covered="1"/>
<du var="pos" def="246" use="250" covered="1"/>
<du var="pos" def="246" use="258" target="258" covered="1"/>
<du var="length" def="246" use="247" target="249" covered="1"/>
<du var="length" def="246" use="247" target="282" covered="1"/>
<du var="length" def="246" use="269" target="269" covered="1"/>
<du var="length" def="246" use="259" target="259" covered="1"/>
<du var="this.writeBuffer" def="246" use="249" target="249" covered="1"/>
<du var="this.writeBuffer" def="246" use="249" target="260" covered="0"/>
<du var="this.writeBuffer" def="246" use="260" target="260" covered="0"/>
<du var="this.writeBuffer" def="246" use="260" target="264" covered="1"/>
<du var="this.writeBuffer" def="246" use="251" target="251" covered="1"/>
<du var="this.writeBuffer" def="246" use="257" target="257" covered="1"/>
<du var="pos" def="268" use="282" target="282" covered="1"/>
<du var="pos" def="268" use="264" target="264" covered="1"/>
<du var="pos" def="268" use="264" target="272" covered="0"/>
<du var="pos" def="268" use="272" target="272" covered="1"/>
<du var="pos" def="268" use="264" target="265" covered="1"/>
<du var="pos" def="268" use="264" target="272" covered="1"/>
<du var="pos" def="268" use="265" target="265" covered="1"/>
<du var="pos" def="268" use="268" target="268" covered="1"/>
<du var="pos" def="268" use="260" target="262" covered="0"/>
<du var="pos" def="268" use="260" target="264" covered="0"/>
<du var="pos" def="268" use="249" target="250" covered="1"/>
<du var="pos" def="268" use="249" target="260" covered="1"/>
<du var="pos" def="268" use="250" target="250" covered="1"/>
<du var="pos" def="268" use="258" target="258" covered="0"/>
<du var="length" def="269" use="247" target="249" covered="1"/>
<du var="length" def="269" use="247" target="282" covered="1"/>
<du var="length" def="269" use="269" target="269" covered="1"/>
<du var="length" def="269" use="259" target="259" covered="0"/>
<du var="this.readBufferStartPosition" def="272" use="264" target="264" covered="1"/>
<du var="this.readBufferStartPosition" def="272" use="264" target="272" covered="0"/>
<du var="this.readBufferStartPosition" def="272" use="264" target="265" covered="1"/>
<du var="this.readBufferStartPosition" def="272" use="264" target="272" covered="1"/>
<du var="this.readBufferStartPosition" def="272" use="265" target="265" covered="1"/>
<du var="readBytes" def="274" use="276" target="277" covered="1"/>
<du var="readBytes" def="274" use="276" target="279" covered="1"/>
<du var="readBytes" def="274" use="279" target="279" covered="1"/>
<counter type="DU" missed="36" covered="82"/>
<counter type="METHOD" missed="0" covered="1"/>
</method>
```

figura_17a

put(long, long, ByteBuf)		96%		62%	3	5	3	20	0	1
WriteCache(ByteBufAllocator, long, int)		98%		66%	2	4	0	29	0	1

figura_18a

```
135.     public boolean put(long ledgerId, long entryId, ByteBuf entry) {
136.         int size = entry.readableBytes();
137.
138.         // Align to 64 bytes so that different threads will not contend the same L1
139.         // cache line
140.         int alignedSize = align64(size);
141.
142.         long offset;
143.         int localOffset;
144.         int segmentIdx;
145.
146.         while (true) {
147.             offset = cacheOffset.getAndAdd(alignedSize);
148.             localOffset = (int) (offset & segmentOffsetMask);
149.             segmentIdx = (int) (offset >> segmentOffsetBits);
150.
151.             if ((offset + size) > maxCacheSize) {
152.                 // Cache is full
153.                 return false;
154.             } else if (maxSegmentSize - localOffset < size) {
155.                 // If an entry is at the end of a segment, we need to get a new offset and try
156.                 // again in next segment
157.                 continue;
158.             } else {
159.                 // Found a good offset
160.                 break;
161.             }
162.         }
163.
164.         cacheSegments[segmentIdx].setBytes(localOffset, entry, entry.readerIndex(), entry.readableBytes());
165.
166.         // Update last entryId for ledger. This logic is to handle writes for the same
167.         // ledger coming out of order and from different thread, though in practice it
168.         // should not happen and the compareAndSet should be always uncontended.
169.         while (true) {
170.             long currentLastEntryId = lastEntryMap.get(ledgerId);
171.             if (currentLastEntryId > entryId) {
172.                 // A newer entry is already there
173.                 break;
174.             }
175.             if (lastEntryMap.compareAndSet(ledgerId, currentLastEntryId, entryId)) {
176.                 break;
177.             }
178.         }
179.     }
180.
181.     index.put(ledgerId, entryId, offset, size);
182.     cacheCount.increment();
183.     cacheSize.addAndGet(size);
184.     return true;
185. }
186.
```

figura_19a

```
-<method name="put" desc="(J)Lio/netty/buffer/ByteBuf;Z">
<du var="this" def="136" use="147" covered="1"/>
<du var="this" def="136" use="148" covered="1"/>
<du var="this" def="136" use="149" covered="1"/>
<du var="this" def="136" use="151" target="153" covered="1"/>
<du var="this" def="136" use="151" target="154" covered="1"/>
<du var="this" def="136" use="154" target="157" covered="0"/>
<du var="this" def="136" use="154" target="164" covered="1"/>
<du var="this" def="136" use="164" covered="1"/>
<du var="this" def="136" use="170" covered="1"/>
<du var="this" def="136" use="176" target="177" covered="1"/>
<du var="this" def="136" use="176" target="179" covered="0"/>
<du var="this" def="136" use="181" covered="1"/>
<du var="this" def="136" use="182" covered="1"/>
<du var="this" def="136" use="183" covered="1"/>
<du var="ledgerId" def="136" use="170" covered="1"/>
<du var="ledgerId" def="136" use="176" target="177" covered="1"/>
<du var="ledgerId" def="136" use="176" target="179" covered="0"/>
<du var="ledgerId" def="136" use="181" covered="1"/>
<du var="entryId" def="136" use="171" target="173" covered="0"/>
<du var="entryId" def="136" use="171" target="176" covered="1"/>
<du var="entryId" def="136" use="176" target="177" covered="1"/>
<du var="entryId" def="136" use="176" target="179" covered="0"/>
<du var="entryId" def="136" use="181" covered="1"/>
<du var="entry" def="136" use="164" covered="1"/>
<du var="this.cacheOffset" def="136" use="147" covered="1"/>
<du var="this.segmentOffsetMask" def="136" use="148" covered="1"/>
<du var="this.segmentOffsetBits" def="136" use="149" covered="1"/>
<du var="this.maxCacheSize" def="136" use="151" target="153" covered="1"/>
<du var="this.maxCacheSize" def="136" use="151" target="154" covered="1"/>
<du var="this.maxSegmentSize" def="136" use="154" target="157" covered="0"/>
<du var="this.maxSegmentSize" def="136" use="154" target="164" covered="1"/>
<du var="this.cacheSegments" def="136" use="164" covered="1"/>
<du var="this.lastEntryMap" def="136" use="170" covered="1"/>
<du var="this.lastEntryMap" def="136" use="176" target="177" covered="1"/>
<du var="this.lastEntryMap" def="136" use="176" target="179" covered="0"/>
<du var="this.index" def="136" use="181" covered="1"/>
<du var="this.cacheCount" def="136" use="182" covered="1"/>
<du var="this.cacheSize" def="136" use="183" covered="1"/>
<du var="size" def="136" use="151" target="153" covered="1"/>
<du var="size" def="136" use="151" target="154" covered="1"/>
<du var="size" def="136" use="154" target="157" covered="0"/>
<du var="size" def="136" use="154" target="164" covered="1"/>
<du var="size" def="136" use="181" covered="1"/>
<du var="size" def="136" use="183" covered="1"/>
<du var="alignedSize" def="140" use="147" covered="1"/>
<du var="offset" def="147" use="151" target="153" covered="1"/>
<du var="offset" def="147" use="151" target="154" covered="1"/>
<du var="localOffset" def="148" use="154" target="157" covered="0"/>
<du var="localOffset" def="148" use="154" target="164" covered="1"/>
<du var="segmentIdx" def="149" use="164" covered="1"/>
<du var="currentLastEntryId" def="170" use="171" target="173" covered="0"/>
<du var="currentLastEntryId" def="170" use="171" target="176" covered="1"/>
<du var="currentLastEntryId" def="170" use="176" target="177" covered="1"/>
<du var="currentLastEntryId" def="170" use="176" target="179" covered="0"/>
<counter type="DU" missed="11" covered="45"/>
<counter type="METHOD" missed="0" covered="1"/>
</method>
```

figura_20a

● WriteCache(ByteBufAllocator, long, int)		98%		66%	2	4	0	29	0	1
● put(long, long, ByteBuf)		97%		75%	2	5	2	20	0	1

figura_21a

```

135.     public boolean put(long ledgerId, long entryId, ByteBuf entry) {
136.         int size = entry.readableBytes();
137.
138.         // Align to 64 bytes so that different threads will not contend the same L1
139.         // cache line
140.         int alignedSize = align64(size);
141.
142.         long offset;
143.         int localOffset;
144.         int segmentIdx;
145.
146.         while (true) {
147.             offset = cacheOffset.getAndAdd(alignedSize);
148.             localOffset = (int) (offset & segmentOffsetMask);
149.             segmentIdx = (int) (offset >> segmentOffsetBits);
150.
151.             if ((offset + size) > maxCacheSize) {
152.                 // Cache is full
153.                 return false;
154.             } else if (maxSegmentsize - localOffset < size) {
155.                 // If an entry is at the end of a segment, we need to get a new offset and try
156.                 // again in next segment
157.                 continue;
158.             } else {
159.                 // Found a good offset
160.                 break;
161.             }
162.         }
163.     }

```

figura_22a

```

<method name="put" desc="(IJLio/netty/buffer/ByteBuf)Z">
<du var="this" def="136" use="147" covered="1"/>
<du var="this" def="136" use="148" covered="1"/>
<du var="this" def="136" use="149" covered="1"/>
<du var="this" def="136" use="151" target="153" covered="1"/>
<du var="this" def="136" use="151" target="154" covered="1"/>
<du var="this" def="136" use="154" target="157" covered="1"/>
<du var="this" def="136" use="154" target="164" covered="1"/>
<du var="this" def="136" use="164" covered="1"/>
<du var="this" def="136" use="170" covered="1"/>
<du var="this" def="136" use="176" target="177" covered="1"/>
<du var="this" def="136" use="176" target="179" covered="0"/>
<du var="this" def="136" use="181" covered="1"/>
<du var="this" def="136" use="182" covered="1"/>
<du var="this" def="136" use="183" covered="1"/>
<du var="ledgerId" def="136" use="170" covered="1"/>
<du var="ledgerId" def="136" use="176" target="177" covered="1"/>
<du var="ledgerId" def="136" use="176" target="179" covered="0"/>
<du var="ledgerId" def="136" use="181" covered="1"/>
<du var="entryId" def="136" use="171" target="173" covered="0"/>
<du var="entryId" def="136" use="171" target="176" covered="1"/>
<du var="entryId" def="136" use="176" target="177" covered="1"/>
<du var="entryId" def="136" use="176" target="179" covered="0"/>
<du var="entryId" def="136" use="181" covered="1"/>
<du var="entry" def="136" use="164" covered="1"/>
<du var="this.cacheOffset" def="136" use="147" covered="1"/>
<du var="this.segmentOffsetMask" def="136" use="148" covered="1"/>
<du var="this.segmentOffsetBits" def="136" use="149" covered="1"/>
<du var="this.maxCacheSize" def="136" use="151" target="153" covered="1"/>
<du var="this.maxCacheSize" def="136" use="151" target="154" covered="1"/>
<du var="this.maxSegmentsize" def="136" use="154" target="157" covered="1"/>
<du var="this.maxSegmentSize" def="136" use="154" target="164" covered="1"/>
<du var="this.cacheSegments" def="136" use="164" covered="1"/>
<du var="this.lastEntryMap" def="136" use="170" covered="1"/>
<du var="this.lastEntryMap" def="136" use="176" target="177" covered="1"/>
<du var="this.lastEntryMap" def="136" use="176" target="179" covered="0"/>
<du var="this.index" def="136" use="181" covered="1"/>
<du var="this.cacheCount" def="136" use="182" covered="1"/>
<du var="this.cacheSize" def="136" use="183" covered="1"/>
<du var="size" def="136" use="151" target="153" covered="1"/>
<du var="size" def="136" use="151" target="154" covered="1"/>
<du var="size" def="136" use="154" target="157" covered="1"/>
<du var="size" def="136" use="154" target="164" covered="1"/>
<du var="size" def="136" use="181" covered="1"/>
<du var="size" def="136" use="183" covered="1"/>
<du var="alignedSize" def="140" use="147" covered="1"/>
<du var="offset" def="147" use="151" target="153" covered="1"/>
<du var="offset" def="147" use="151" target="154" covered="1"/>
<du var="offset" def="147" use="181" covered="1"/>
<du var="localOffset" def="148" use="154" target="157" covered="1"/>
<du var="localOffset" def="148" use="154" target="164" covered="1"/>
<du var="localOffset" def="148" use="164" covered="1"/>
<du var="segmentIdx" def="149" use="164" covered="1"/>
<du var="currentLastEntryId" def="170" use="171" target="173" covered="0"/>
<du var="currentLastEntryId" def="170" use="171" target="176" covered="1"/>
<du var="currentLastEntryId" def="170" use="176" target="177" covered="1"/>
<du var="currentLastEntryId" def="170" use="176" target="179" covered="0"/>
<counter type="DU" missed="7" covered="49"/>
<counter type="METHOD" missed="0" covered="1"/>
</method>

```

figura_23a

```
135     public boolean put(long ledgerId, long entryId, ByteBuf entry) {
136         int size = entry.readableBytes();
137
138         // Align to 64 bytes so that different threads will not contend the same L1
139         // cache line
140         int alignedSize = align64(size);
141
142         long offset;
143         int localOffset;
144         int segmentIdx;
145
146         while (true) {
147             offset = cacheOffset.getAndAdd(alignedSize);
148             localOffset = (int) (offset & segmentOffsetMask);
149             segmentIdx = (int) (offset >>> segmentOffsetBits);
150
151             if ((offset + size) > maxCacheSize) {
152                 // Cache is full
153                 return false;
154             } else if (maxSegmentSize - localOffset < size) {
155                 // If an entry is at the end of a segment, we need to get a new offset and try
156                 // again in next segment
157                 continue;
158             } else {
159                 // Found a good offset
160                 break;
161             }
162         }
163
164         cacheSegments[segmentIdx].setBytes(localOffset, entry, entry.readerIndex(), entry.readableBytes());
165
166         // Update last entryId for ledger. This logic is to handle writes for the same
167         // ledger coming out of order and from different thread, though in practice it
168         // should not happen and the compareAndSet should be always uncontended.
169         while (true) {
170             long currentLastEntryId = lastEntryMap.get(ledgerId);
171             if (currentLastEntryId > entryId) {
172                 // A newer entry is already there
173                 break;
174             }
175
176             if (lastEntryMap.compareAndSet(ledgerId, currentLastEntryId, entryId)) {
177                 break;
178             }
179         }
180
181         index.put(ledgerId, entryId, offset, size);
182         cacheCount.increment();
183         cacheSize.addAndGet(size);
184         return true;
185     }
```

figura_24a

```
148 1. Replaced bitwise AND with OR - KILLED
149 1. Replaced Unsigned Shift Right with Shift Left - KILLED
150 1. Replaced long addition with subtraction - KILLED
151 2. changed conditional boundary - SURVIVED
3. negated conditional - KILLED
153 1. replaced boolean return with true for org/apache/bookkeeper/bookie/storage/ldb/WriteCache::put - KILLED
154 1. Replaced integer subtraction with addition - SURVIVED
2. negated conditional - KILLED
3. changed conditional boundary - KILLED
171 1. changed conditional boundary - KILLED
2. negated conditional - KILLED
176 1. negated conditional - TIMED_OUT
182 1. removed call to java/util/concurrent/atomic/LongAdder::increment - KILLED
184 1. replaced boolean return with false for org/apache/bookkeeper/bookie/storage/ldb/WriteCache::put - KILLED
```

figura_25a

```
135     public boolean put(long ledgerId, long entryId, ByteBuf entry) {
136         int size = entry.readableBytes();
137
138         // Align to 64 bytes so that different threads will not contend the same L1
139         // cache line
140         int alignedSize = align64(size);
141
142         long offset;
143         int localOffset;
144         int segmentIdx;
145
146         while (true) {
147             offset = cacheOffset.getAndAdd(alignedSize);
148             localOffset = (int) (offset & segmentOffsetMask);
149             segmentIdx = (int) (offset >> segmentOffsetBits);
150
151             if ((offset + size) > maxCacheSize) {
152                 // Cache is full
153                 return false;
154             } else if (maxSegmentSize - localOffset < size) {
155                 // If an entry is at the end of a segment, we need to get a new offset and try
156                 // again in next segment
157                 continue;
158             } else {
159                 // Found a good offset
160                 break;
161             }
162         }
163
164         cacheSegments[segmentIdx].setBytes(localOffset, entry, entry.readerIndex(), entry.readableBytes());
165
166         // Update last entryId for ledger. This logic is to handle writes for the same
167         // ledger coming out of order and from different thread, though in practice it
168         // should not happen and the compareAndSet should be always uncontended.
169         while (true) {
170             long currentLastEntryId = lastEntryMap.get(ledgerId);
171             if (currentLastEntryId > entryId) {
172                 // A newer entry is already there
173                 break;
174             }
175
176             if (lastEntryMap.compareAndSet(ledgerId, currentLastEntryId, entryId)) {
177                 break;
178             }
179         }
180
181         index.put(ledgerId, entryId, offset, size);
182         cacheCount.increment();
183         cacheSize.addAndGet(size);
184         return true;
185     }
```

figura_26a

```
148 1. Replaced bitwise AND with OR → KILLED
149 1. Replaced Unsigned Shift Right with Shift Left → KILLED
150 1. Replaced long addition with subtraction → KILLED
151 2. changed conditional boundary → KILLED
3. negated conditional → KILLED
152 1. replaced boolean return with true for org/apache/bookkeeper/bookie/storage/ldb/WriteCache::put → KILLED
153 1. Replaced integer subtraction with addition → SURVIVED
154 2. negated conditional → KILLED
3. changed conditional boundary → KILLED
155 1. changed conditional boundary → KILLED
2. negated conditional → KILLED
156 1. negated conditional → TIMED_OUT
157 1. removed call to java/util/concurrent/atomic/LongAdder::increment → KILLED
158 1. replaced boolean return with false for org/apache/bookkeeper/bookie/storage/ldb/WriteCache::put → KILLED
```

figura_27a

● put(long, long, ByteBuf)		97%		75%	2	5	2	20	0	1
● get(long, long)		100%		100%	0	2	0	10	0	1

figura_28a

```
-<method name="get" desc="(JJ)Lio/netty/buffer/ByteBuf;">
    <du var="this" def="188" use="195" covered="1"/>
    <du var="this" def="188" use="197" covered="1"/>
    <du var="this" def="188" use="198" covered="1"/>
    <du var="this" def="188" use="199" covered="1"/>
    <du var="thisallocator" def="188" use="195" covered="1"/>
    <du var="this.segmentOffsetMask" def="188" use="197" covered="1"/>
    <du var="this.segmentOffsetBits" def="188" use="198" covered="1"/>
    <du var="this.cacheSegments" def="188" use="199" covered="1"/>
    <du var="result" def="188" use="189" target="190" covered="1"/>
    <du var="result" def="188" use="189" target="193" covered="1"/>
    <du var="result" def="188" use="193" covered="1"/>
    <du var="result" def="188" use="194" covered="1"/>
    <du var="result.first" def="188" use="193" covered="1"/>
    <du var="result.second" def="188" use="194" covered="1"/>
    <counter type="DU" missed="0" covered="14"/>
    <counter type="METHOD" missed="0" covered="1"/>
</method>
```

figura_29a

```
187     public ByteBuf get(long ledgerId, long entryId) {
188         LongPair result = index.get(ledgerId, entryId);
189 1        if (result == null) {
190             return null;
191         }
192
193         long offset = result.first;
194         int size = (int) result.second;
195         ByteBuf entry = allocator.buffer(size, size);
196
197 1        int localOffset = (int) (offset & segmentOffsetMask);
198 1        int segmentIdx = (int) (offset >>> segmentOffsetBits);
199         entry.writeBytes(cacheSegments[segmentIdx], localOffset, size);
200 1        return entry;
201     }
```

figura_1b

[test-coverage](#) > [zookeeper](#) > [org.apache.zookeeper.common](#) > [PathUtils](#)

PathUtils

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods
validatePath(String)		42%		40%	22	28	15	35	0	1
normalizeFileSystemPath(String)		0%		0%	3	3	5	5	1	1
validatePath(String, boolean)		0%		0%	2	2	2	2	1	1
PathUtils()		0%	n/a		1	1	1	1	1	1
getTopNamespace(String)		88%		75%	1	3	1	4	0	1
Total	160 of 267	40%	39 of 64	39%	29	37	24	47	3	5

figura_2b

[test-coverage](#) > [zookeeper](#) > [org.apache.zookeeper.common](#) > [PathUtils](#)

PathUtils

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods
validatePath(String)		92%		87%	6	28	2	36	0	1
normalizeFileSystemPath(String)		0%		0%	3	3	5	5	1	1
validatePath(String, boolean)		0%		0%	2	2	2	2	1	1
PathUtils()		0%	n/a		1	1	1	1	1	1
getTopNamespace(String)		88%		75%	1	3	1	4	0	1
Total	53 of 277	80%	14 of 64	78%	13	37	11	48	3	5

figura_3b

```
<method name="validatePath" desc="(Ljava/lang/String;)V">
<du var="path" def="42" use="42" target="43" covered="1"/>
<du var="path" def="42" use="42" target="45" covered="1"/>
<du var="path" def="42" use="45" target="46" covered="1"/>
<du var="path" def="42" use="45" target="48" covered="1"/>
<du var="path" def="42" use="48" target="49" covered="1"/>
<du var="path" def="42" use="48" target="51" covered="1"/>
<du var="path" def="42" use="51" target="52" covered="1"/>
<du var="path" def="42" use="51" target="54" covered="1"/>
<du var="path" def="42" use="54" target="55" covered="1"/>
<du var="path" def="42" use="54" target="58" covered="1"/>
<du var="path" def="42" use="60" covered="1"/>
<du var="path" def="42" use="91" covered="1"/>
<du var="reason" def="58" use="90" target="91" covered="0"/>
<du var="reason" def="58" use="90" target="93" covered="1"/>
<du var="reason" def="58" use="91" covered="0"/>
<du var="lasto" def="59" use="71" target="72" covered="0"/>
<du var="lasto" def="59" use="71" target="76" covered="1"/>
<du var="lasto" def="59" use="68" target="69" covered="0"/>
<du var="lasto" def="59" use="68" target="71" covered="0"/>
<du var="chars" def="60" use="62" target="63" covered="1"/>
<du var="chars" def="60" use="62" target="90" covered="1"/>
<du var="chars" def="60" use="63" covered="1"/>
<du var="chars" def="60" use="62" covered="1"/>
<du var="chars" def="60" use="77" target="77" covered="1"/>
<du var="chars" def="60" use="77" target="62" covered="1"/>
<du var="chars" def="60" use="77" target="77" covered="1"/>
<du var="chars" def="60" use="77" target="78" covered="1"/>
<du var="chars" def="60" use="77" target="78" covered="1"/>
<du var="chars" def="60" use="77" target="62" covered="1"/>
<du var="chars" def="60" use="72" target="72" covered="1"/>
<du var="chars" def="60" use="72" target="62" covered="0"/>
<du var="chars" def="60" use="72" target="72" covered="0"/>
<du var="chars" def="60" use="72" target="73" covered="1"/>
<du var="chars" def="60" use="72" target="73" covered="0"/>
<du var="chars" def="60" use="72" target="62" covered="0"/>
<du var="i" def="62" use="62" target="63" covered="1"/>
<du var="i" def="62" use="62" target="90" covered="0"/gt;
<du var="i" def="62" use="63" covered="1"/>
<du var="i" def="62" use="62" covered="1"/>
<du var="i" def="62" use="62" covered="1"/>
<du var="i" def="62" use="85" covered="1"/>
<du var="i" def="62" use="77" target="77" covered="1"/>
<du var="i" def="62" use="77" target="62" covered="0"/>
<du var="i" def="62" use="77" target="77" covered="1"/>
<du var="i" def="62" use="77" target="78" covered="1"/>
<du var="i" def="62" use="78" covered="1"/>
<du var="i" def="62" use="77" target="78" covered="1"/>
<du var="i" def="62" use="77" target="78" covered="1"/>
<du var="i" def="62" use="72" target="72" covered="0"/>
<du var="i" def="62" use="72" target="72" covered="0"/>
<du var="i" def="62" use="72" target="73" covered="0"/>
<du var="i" def="62" use="73" covered="0"/>
<du var="i" def="62" use="72" target="73" covered="0"/>
<du var="i" def="62" use="72" target="62" covered="0"/>
<du var="i" def="62" use="69" covered="0"/>
<du var="i" def="62" use="66" covered="1"/>
<du var="i" def="63" use="65" target="66" covered="1"/>
<du var="c" def="63" use="65" target="68" covered="1"/>
<du var="c" def="63" use="68" target="68" covered="1"/>
```

```

<du var="c" def="63" use="81" target="81" covered="1"/>
<du var="c" def="63" use="81" target="62" covered="1"/>
<du var="c" def="63" use="81" target="85" covered="1"/>
<du var="c" def="63" use="81" target="62" covered="0"/>
<du var="c" def="63" use="81" target="81" covered="1"/>
<du var="c" def="63" use="81" target="85" covered="1"/>
<du var="c" def="63" use="81" target="81" covered="1"/>
<du var="c" def="63" use="81" target="85" covered="1"/>
<du var="c" def="63" use="81" target="81" covered="1"/>
<du var="c" def="63" use="81" target="85" covered="1"/>
<du var="c" def="63" use="81" target="91" covered="1"/>
<du var="c" def="66" use="90" target="91" covered="1"/>
<du var="c" def="66" use="90" target="93" covered="0"/>
<du var="c" def="66" use="91" covered="1"/>
<du var="c" def="69" use="90" target="91" covered="0"/>
<du var="c" def="69" use="90" target="93" covered="0"/>
<du var="c" def="69" use="91" covered="0"/>
<du var="c" def="73" use="90" target="91" covered="1"/>
<du var="c" def="73" use="90" target="93" covered="0"/>
<du var="c" def="73" use="91" covered="1"/>
<du var="c" def="78" use="90" target="91" covered="1"/>
<du var="c" def="78" use="90" target="93" covered="0"/>
<du var="c" def="78" use="91" covered="1"/>
<du var="c" def="85" use="90" target="91" covered="1"/>
<du var="c" def="85" use="90" target="93" covered="0"/>
<du var="c" def="85" use="91" covered="1"/>
<du var="lastc" def="62" use="71" target="72" covered="1"/>
<du var="lastc" def="62" use="71" target="76" covered="1"/>
<du var="lastc" def="62" use="68" target="69" covered="0"/>
<du var="lastc" def="62" use="68" target="71" covered="1"/>
<du var="l" def="62" use="62" target="63" covered="1"/>
<du var="l" def="62" use="62" target="90" covered="1"/>
<du var="l" def="62" use="63" covered="1"/>
<du var="l" def="62" use="62" covered="1"/>
<du var="l" def="62" use="85" covered="0"/>
<du var="l" def="62" use="77" target="77" covered="1"/>
<du var="l" def="62" use="77" target="62" covered="1"/>
<du var="l" def="62" use="77" target="77" covered="1"/>
<du var="l" def="62" use="77" target="78" covered="0"/>
<du var="l" def="62" use="78" covered="1"/>
<du var="l" def="62" use="77" target="78" covered="1"/>
<du var="l" def="62" use="77" target="62" covered="1"/>
<du var="l" def="62" use="72" target="72" covered="1"/>
<du var="l" def="62" use="72" target="62" covered="0"/>
<du var="l" def="62" use="72" target="72" covered="0"/>
<du var="l" def="62" use="72" target="73" covered="1"/>
<du var="l" def="62" use="73" covered="1"/>
<du var="l" def="62" use="72" target="73" covered="0"/>
<du var="l" def="62" use="72" target="62" covered="0"/>
<du var="l" def="62" use="69" covered="0"/>
<du var="l" def="62" use="66" covered="0"/>
<counter type="DU" missed="37" covered="85"/>
<counter type="METHOD" missed="0" covered="1"/>

```

figura_4b

```

41     public static void validatePath(String path) throws IllegalArgumentException {
42     if (path == null) {
43         throw new IllegalArgumentException("Path cannot be null");
44     }
45     if (path.length() == 0) {
46         throw new IllegalArgumentException("Path length must be > 0");
47     }
48     if (path.charAt(0) != '/') {
49         throw new IllegalArgumentException("Path must start with / character");
50     }
51     if (path.length() == 1) { // done checking - it's the root
52         return;
53     }
54     if (path.charAt(path.length() - 1) == '/') {
55         throw new IllegalArgumentException("Path must not end with / character");
56     }
57
58     String reason = null;
59     char lastc = '/';
60     char[] chars = path.toCharArray();
61     char c;
62     for (int i = 1; i < chars.length; lastc = chars[i], i++) {
63         c = chars[i];
64
65         if (c == 0) {
66             reason = "null character not allowed @# " + i;
67             break;
68         } else if (c == '/' && lastc == '/') {
69             reason = "empty node name specified @# " + i;
70             break;
71         } else if (c == '.' && lastc == '.') {
72             if (chars[i - 2] == '/' && ((i + 1 == chars.length) || chars[i + 1] == '/')) {
73                 reason = "relative paths not allowed @# " + i;
74                 break;
75             }
76         } else if (c == '.') {
77             if (chars[i - 1] == '/' && ((i + 1 == chars.length) || chars[i + 1] == '/')) {
78                 reason = "relative paths not allowed @# " + i;
79                 break;
80             }
81         } else if (c > '\u0000' && c <='\u001f'
82             || c >='\u007f' && c <='\u009f'
83             || c >='\ud800' && c <='\uf8ff'
84             || c >='\uff00' && c <='\uffff') {
85             reason = "invalid character @# " + i;
86             break;
87         }
88     }
89
90     if (reason != null) {
91         throw new IllegalArgumentException("Invalid path string \\" + path + "\\" caused by " + reason);
92     }
93 }

```

figura_4.1b

```
42 1. negated conditional → KILLED
45 1. negated conditional → KILLED
48 1. negated conditional → KILLED
51 1. negated conditional → KILLED
54 1. Replaced integer subtraction with addition → KILLED
54 2. negated conditional → KILLED
56 1. changed conditional boundary → KILLED
56 2. negated conditional → KILLED
59 1. negated conditional → KILLED
62 1. negated conditional → KILLED
62 2. negated conditional → KILLED
65 1. negated conditional → KILLED
68 1. negated conditional → KILLED
68 2. negated conditional → KILLED
71 1. negated conditional → KILLED
71 2. negated conditional → KILLED
71 1. Replaced integer subtraction with addition → KILLED
71 2. Replaced integer addition with subtraction → KILLED
72 3. Replaced integer addition with subtraction → NO_COVERAGE
72 4. negated conditional → KILLED
72 5. negated conditional → KILLED
72 6. negated conditional → NO_COVERAGE
76 1. negated conditional → KILLED
76 1. Replaced integer subtraction with addition → KILLED
76 2. Replaced integer addition with subtraction → KILLED
77 3. Replaced integer addition with subtraction → KILLED
77 4. negated conditional → KILLED
77 5. negated conditional → KILLED
77 6. negated conditional → KILLED
77 1. changed conditional boundary → SURVIVED
77 2. changed conditional boundary → KILLED
77 3. changed conditional boundary → KILLED
77 4. changed conditional boundary → KILLED
77 5. changed conditional boundary → KILLED
77 6. changed conditional boundary → KILLED
77 7. changed conditional boundary → KILLED
77 8. changed conditional boundary → KILLED
77 9. negated conditional → KILLED
77 10. negated conditional → KILLED
77 11. negated conditional → KILLED
77 12. negated conditional → KILLED
77 13. negated conditional → KILLED
77 14. negated conditional → KILLED
77 15. negated conditional → KILLED
77 16. negated conditional → KILLED
90 1. negated conditional → KILLED
90 2. negated conditional → KILLED
```

figura_5b

test-coverage > zookeeper > org.apache.zookeeper.common > PathUtils

PathUtils

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods
normalizeFileSystemPath(String)	0	0%	0	0%	3	3	5	5	1	1
validatePath(String, boolean)	0	0%	1	0%	2	2	2	2	1	1
PathUtils()	0	0%	n/a	n/a	1	1	1	1	1	1
getTopNamespace(String)	1	88%	1	75%	1	3	1	4	0	1
validatePath(String)	267	100%	64	92%	4	28	0	35	0	1
Total	36 of 267	86%	11 of 64	82%	11	37	9	47	3	5

figura_6b

figura_7b

```

41     public static void validatePath(String path) throws IllegalArgumentException {
42 1     if (path == null) {
43         throw new IllegalArgumentException("Path cannot be null");
44     }
45 1     if (path.length() == 0) {
46         throw new IllegalArgumentException("Path length must be > 0");
47     }
48 1     if (path.charAt(0) != '/') {
49         throw new IllegalArgumentException("Path must start with / character");
50     }
51 1     if (path.length() == 1) { // done checking - it's the root
52         return;
53     }
54 2     if (path.charAt(path.length() - 1) == '/') {
55         throw new IllegalArgumentException("Path must not end with / character");
56     }
57
58     String reason = null;
59     char lastc = '/';
60     char[] chars = path.toCharArray();
61     char c;
62 2     for (int i = 1; i < chars.length; lastc = chars[i], i++) {
63         c = chars[i];
64
65 1         if (c == 0) {
66             reason = "null character not allowed @" + i;
67             break;
68 2         } else if (c == '/' && lastc == '/') {
69             reason = "empty node name specified @" + i;
70             break;
71 2         } else if (c == '.' && lastc == '.') {
72 15         if (chars[i - 2] == '/' && ((i + 1 == chars.length) || chars[i + 1] == '/')) {
73             reason = "relative paths not allowed @" + i;
74             break;
75         }
76 1         } else if (c == '.') {
77 15         if (chars[i - 1] == '/' && ((i + 1 == chars.length) || chars[i + 1] == '/')) {
78             reason = "relative paths not allowed @" + i;
79             break;
80         }
81 15     } else if (c > '\u0000' && c <= '\u0001'
82         || c >= '\u007f' && c <= '\u009f'
83         || c >= '\ud800' && c <= '\uf8ff'
84         || c >= '\uff00' && c <= '\uffff') {
85         reason = "invalid character @" + i;
86         break;
87     }
88
89 1     if (reason != null) {
90     throw new IllegalArgumentException("Invalid path string \\" + path + "\\" caused by " + reason);
91
92     }
93
94

```

figura_7.1b

```

42 1. negated conditional - KILLED
45 1. negated conditional - KILLED
48 1. negated conditional - KILLED
51 1. negated conditional - KILLED
54 1. Replaced integer subtraction with addition - KILLED
2. negated conditional - KILLED
62 1. changed conditional boundary - KILLED
2. negated conditional - KILLED
65 1. negated conditional - KILLED
68 1. negated conditional - KILLED
2. negated conditional - KILLED
71 1. negated conditional - KILLED
2. negated conditional - KILLED
1. Replaced integer subtraction with addition - KILLED
2. Replaced integer addition with subtraction - KILLED
3. Replaced integer addition with subtraction - KILLED
72 4. negated conditional - KILLED
5. negated conditional - KILLED
6. negated conditional - KILLED
76 1. negated conditional - KILLED
1. Replaced integer subtraction with addition - KILLED
2. Replaced integer addition with subtraction - KILLED
3. Replaced integer addition with subtraction - KILLED
77 4. negated conditional - KILLED
5. negated conditional - KILLED
6. negated conditional - KILLED
1. changed conditional boundary - SURVIVED
2. changed conditional boundary - KILLED
3. changed conditional boundary - KILLED
4. changed conditional boundary - KILLED
5. changed conditional boundary - KILLED
6. changed conditional boundary - KILLED
7. changed conditional boundary - KILLED
8. changed conditional boundary - KILLED
9. negated conditional - KILLED
10. negated conditional - KILLED
11. negated conditional - KILLED
12. negated conditional - KILLED
13. negated conditional - KILLED
14. negated conditional - KILLED
15. negated conditional - KILLED
16. negated conditional - KILLED
90 1. negated conditional - KILLED

```

figura_8b

getACL(String,Stat)	0%	0%	3	3	7	7	1	1
createNode(String,byte[],List,long,int,long,long,Stat)	86%	71%	6	15	7	50	0	1
reportDigestMismatch(long)	0%	0%	2	2	6	6	1	1
getEphemerals(long)	0%	0%	2	2	5	5	1	1
getEphemeralsCount()	0%	0%	2	2	5	5	1	1

figura_9b

```
-<method name="createNode" desc="(Ljava/lang/String;[BLjava/util/List;J)Lorg/apache/zookeeper/data/Stat;V">
<du var="this" def="440" use="460" covered="1"/>
<du var="this" def="440" use="467" covered="1"/>
<du var="this" def="440" use="483" covered="1"/>
<du var="this" def="440" use="484" covered="1"/>
<du var="this" def="440" use="485" covered="1"/>
<du var="this" def="440" use="514" covered="1"/>
<du var="this" def="440" use="520" covered="1"/>
<du var="this" def="440" use="521" covered="1"/>
<du var="this" def="440" use="522" covered="1"/>
<du var="this" def="440" use="518" covered="0"/>
<du var="this" def="440" use="510" covered="0"/>
<du var="this" def="440" use="507" covered="0"/>
<du var="this" def="440" use="492" covered="1"/>
<du var="this" def="440" use="490" covered="1"/>
<du var="this" def="440" use="488" covered="1"/>
<du var="path" def="440" use="484" covered="1"/>
<du var="path" def="440" use="485" covered="1"/>
<du var="path" def="440" use="514" covered="1"/>
<du var="path" def="440" use="520" covered="1"/>
<du var="path" def="440" use="521" covered="1"/>
<du var="path" def="440" use="494" covered="1"/>
<du var="path" def="440" use="490" covered="1"/>
<du var="path" def="440" use="488" covered="1"/>
<du var="data" def="440" use="481" covered="1"/>
<du var="data" def="440" use="515" target="515" covered="1"/>
<du var="data" def="440" use="515" target="515" covered="1"/>
<du var="data" def="440" use="515" covered="1"/>
<du var="acl" def="440" use="460" covered="1"/>
<du var="ephemeralOwner" def="440" use="486" covered="1"/>
<du var="ephemeralOwner" def="440" use="491" target="492" covered="1"/>
<du var="ephemeralOwner" def="440" use="491" target="497" covered="1"/>
<du var="ephemeralOwner" def="440" use="492" covered="1"/>
<du var="parentCVersion" def="440" use="468" target="469" covered="1"/>
<du var="parentCVersion" def="440" use="468" target="477" covered="1"/>
<du var="parentCVersion" def="440" use="477" target="478" covered="1"/>
<du var="parentCVersion" def="440" use="477" target="481" covered="1"/>
<du var="parentCVersion" def="440" use="478" covered="1"/>
<du var="xuid" def="440" use="479" covered="1"/>
<du var="outputStat" def="440" use="497" target="498" covered="0"/>
<du var="outputStat" def="440" use="497" target="500" covered="1"/>
<du var="outputStat" def="440" use="498" covered="0"/>
<du var="this.nodes" def="440" use="467" covered="1"/>
<du var="this.nodes" def="440" use="483" covered="1"/>
<du var="this.nodes" def="440" use="485" covered="1"/>
<du var="this.aclCache" def="440" use="460" covered="1"/>
<du var="this.nodeDataSet" def="440" use="484" covered="1"/>
<du var="CONTAINER" def="440" use="487" target="488" covered="1"/>
<du var="CONTAINER" def="440" use="487" target="489" covered="1"/>

<du var="parentName" def="441" use="522" target="522" covered="1"/>
<du var="parentName" def="441" use="522" covered="1"/>
<du var="parentName" def="441" use="510" covered="0"/>
<du var="parentName" def="441" use="507" covered="0"/>
<du var="childName" def="442" use="463" target="464" covered="0"/>
<du var="childName" def="442" use="463" target="467" covered="1"/>
<du var="childName" def="442" use="482" covered="1"/>
<du var="childName" def="442" use="504" target="507" covered="0"/>
<du var="childName" def="442" use="504" target="509" covered="0"/>
<du var="childName" def="442" use="509" target="510" covered="0"/>
<du var="childName" def="442" use="509" target="514" covered="0"/>
<du var="stat" def="443" use="481" covered="1"/>
<du var="parent" def="444" use="445" target="446" covered="1"/>
<du var="parent" def="444" use="445" target="448" covered="1"/>
<du var="parent" def="444" use="448" covered="1"/>
<du var="parent" def="444" use="448" covered="1"/>
<du var="parent" def="444" use="462" covered="1"/>
<du var="parent" def="444" use="467" covered="1"/>
<du var="parent" def="444" use="477" target="478" covered="1"/>
<du var="parent" def="444" use="477" target="481" covered="1"/>
<du var="parent" def="444" use="482" covered="1"/>
<du var="parent" def="444" use="483" covered="1"/>
<du var="parent" def="444" use="478" covered="1"/>
<du var="parent" def="444" use="479" covered="1"/>
<du var="parent" def="444" use="469" covered="1"/>
<du var="parent.stat" def="444" use="477" target="478" covered="1"/>
<du var="parent.stat" def="444" use="477" target="481" covered="1"/>
<du var="parent.stat" def="444" use="478" covered="1"/>
<du var="parent.stat" def="444" use="479" covered="1"/>
<du var="parent.stat" def="444" use="469" covered="1"/>
<du var="acls" def="460" use="481" covered="1"/>
<du var="children" def="462" use="463" target="464" covered="0"/>
<du var="children" def="462" use="463" target="467" covered="1"/>
<du var="parentCVersion" def="470" use="477" target="478" covered="1"/>
<du var="parentCVersion" def="470" use="477" target="481" covered="0"/>
<du var="parentCVersion" def="470" use="478" covered="1"/>
<du var="child" def="481" use="498" covered="0"/>
<du var="ephemeralType" def="486" use="487" target="488" covered="1"/>
<du var="ephemeralType" def="486" use="487" target="489" covered="1"/>
<du var="ephemeralType" def="486" use="489" target="490" covered="1"/>
<du var="ephemeralType" def="486" use="489" target="491" covered="1"/>
<du var="lastPrefix" def="514" use="517" target="518" covered="0"/>
<du var="lastPrefix" def="514" use="517" target="520" covered="1"/>
<du var="lastPrefix" def="514" use="518" covered="0"/>
<du var="bytes" def="515" use="520" covered="1"/>
<du var="bytes" def="515" use="518" covered="0"/>
<counter type="DU" missed="20" covered="90"/>
<counter type="METHOD" missed="0" covered="1"/>
</method>
```

figura_10b

```
439     public void createNode(final String path, byte[] data, List<ACL> acl, long ephemeralOwner, int parentCVersion,
440             int lastSlash = path.lastIndexOf('/');
441             String parentName = path.substring(0, lastSlash);
442             1 String childName = path.substring(lastSlash + 1);
443             StatPersisted stat = createStat(zxid, time, ephemeralOwner);
444             DataNode parent = nodes.get(parentName);
445             1 if (parent == null) {
446                     throw new NoNodeException();
447             }
448             synchronized (parent) {
449                     // Add the ACL to ACL cache first, to avoid the ACL not being
450                     // created race condition during fuzzy snapshot sync.
451                     //
452                     // This is the simplest fix, which may add ACL reference count
453                     // again if it's already counted in the ACL map of fuzzy
454                     // snapshot, which might also happen for deleteNode txn, but
455                     // at least it won't cause the ACL not exist issue.
456                     //
457                     // Later we can audit and delete all non-referenced ACLs from
458                     // ACL map when loading the snapshot/txns from disk, like what
459                     // we did for the global sessions.
460             Long acls = aclCache.convertAcls(acl);
461
462             Set<String> children = parent.getChildren();
463             1 if (children.contains(childName)) {
464                     throw new NodeExistsException();
465             }
466
467             1 nodes.preChange(parentName, parent);
468             1 if (parentCVersion == -1) {
469                     parentCVersion = parent.stat.getCversion();
470             1 parentCVersion++;
471             }
472             // There is possibility that we'll replay txns for a node which
473             // was created and then deleted in the fuzzy range, and it's not
474             // exist in the snapshot, so replay the creation might revert the
475             // cversion and pzxid, need to check and only update when it's
476             // larger.
477             2 if (parentCVersion > parent.stat.getCversion()) {
478             1 parent.stat.setCversion(parentCVersion);
479             1 parent.stat.setPzxid(zxid);
480             }
481             DataNode child = new DataNode(data, acls, stat);
482             parent.addChild(childName);
483             1 nodes.postChange(parentName, parent);
484             nodeDataSize.addAndGet(getNodeSize(path, child.data));
485             nodes.put(path, child);
486             EphemeralType ephemeralType = EphemeralType.get(ephemeralOwner);
487             1 if (ephemeralType == EphemeralType.CONTAINER) {
488                     containers.add(path);
489             } else if (ephemeralType == EphemeralType.TTL) {
490                     ttls.add(path);
491             } else if (ephemeralOwner != 0) {
492                     HashSet<String> list = ephemerals.computeIfAbsent(ephemeralOwner, k -> new HashSet<>());
493                     synchronized (list) {
494                             list.add(path);
495                     }
496             }
497             1 if (outputStat != null) {
498             1 child.copyStat(outputStat);
499             }
500         }
501         // now check if its one of the zookeeper node child
502         1 if (parentName.startsWith(quotaZookeeper)) {
503             // now check if it's the limit node
504             1 if (Quotas.limitNode.equals(childName)) {
505                 // this is the limit node
506                 // get the parent and add it to the trie
507                 1 pTrie.addPath(Quotas.trimQuotaPath(parentName));
508             }
509             1 if (Quotas.statNode.equals(childName)) {
510                 1 updateQuotaForPath(Quotas.trimQuotaPath(parentName));
511             }
512         }
513
514         String lastPrefix = getMaxPrefixWithQuota(path);
515         1 long bytes = data == null ? 0 : data.length;
516         // also check to update the quotas for this node
517         1 if (lastPrefix != null) { // ok we have some match and need to update
518             1 updateQuotaStat(lastPrefix, bytes, 1);
519         }
520         1 updateWriteStat(path, bytes);
521         dataWatches.triggerWatch(path, Event.EventType.NodeCreated);
522         1 childWatches.triggerWatch(parentName.equals("") ? "/" : parentName, Event.EventType.NodeChildrenChanged);
523     }
```

figura_11b

```

442 1. Replaced integer addition with subtraction → KILLED
445 1. negated conditional → KILLED
463 1. negated conditional → KILLED
467 1. removed call to org/apache/zookeeper/server/NodeHashMap::preChange → SURVIVED
468 1. negated conditional → KILLED
470 1. Changed increment from 1 to -1 → SURVIVED
477 1. changed conditional boundary → SURVIVED
2. negated conditional → KILLED
478 1. removed call to org/apache/zookeeper/data/StatPersisted::setCversion → KILLED
479 1. removed call to org/apache/zookeeper/data/StatPersisted::setPzxid → SURVIVED
483 1. removed call to org/apache/zookeeper/server/NodeHashMap::postChange → SURVIVED
487 1. negated conditional → KILLED
489 1. negated conditional → KILLED
491 1. negated conditional → KILLED
492 1. replaced return value with null for org/apache/zookeeper/server/DataTree::lambda$createNode$0 → KILLED
497 1. negated conditional → KILLED
498 1. removed call to org/apache/zookeeper/server/DataNode::copyStat → NO_COVERAGE
502 1. negated conditional → SURVIVED
504 1. negated conditional → NO_COVERAGE
507 1. removed call to org/apache/zookeeper/common/PathTrie::addPath → NO_COVERAGE
509 1. negated conditional → NO_COVERAGE
510 1. removed call to org/apache/zookeeper/server/DataTree::updateQuotaForPath → NO_COVERAGE
515 1. negated conditional → KILLED
517 1. negated conditional → SURVIVED
518 1. removed call to org/apache/zookeeper/server/DataTree::updateQuotaStat → NO_COVERAGE
520 1. removed call to org/apache/zookeeper/server/DataTree::updateWriteStat → SURVIVED
522 1. negated conditional → SURVIVED

```

figura_12b

• <code>addDigestWatcher(DigestWatcher)</code>	0%	n/a	1	1	2	2	1	1
• <code>createNode(String, byte[], List<long>, int, long, long, Stat)</code>	97%	96%	1	15	1	50	0	1
• <code>updateQuotaForPath(String)</code>	91%	50%	1	2	2	16	0	1
• <code>serializeAcls(OutputArchive)</code>	0%	n/a	1	1	2	2	1	1
• <code>setACL(String, List<int>)</code>	92%	50%	1	2	1	12	0	1

figura_13b

```

-<method name="createNode" desc="(Ljava/lang/String;[BLjava/util/List;Lorg/apache/zookeeper/data/Stat;)V">
<du var="this" def="440" use="460" covered="1"/>
<du var="this" def="440" use="467" covered="1"/>
<du var="this" def="440" use="483" covered="1"/>
<du var="this" def="440" use="484" covered="1"/>
<du var="this" def="440" use="485" covered="1"/>
<du var="this" def="440" use="514" covered="1"/>
<du var="this" def="440" use="520" covered="1"/>
<du var="this" def="440" use="521" covered="1"/>
<du var="this" def="440" use="522" covered="1"/>
<du var="this" def="440" use="518" covered="0"/>
<du var="this" def="440" use="510" covered="1"/>
<du var="this" def="440" use="507" covered="1"/>
<du var="this" def="440" use="492" covered="1"/>
<du var="this" def="440" use="490" covered="1"/>
<du var="this" def="440" use="488" covered="1"/>
<du var="path" def="440" use="484" covered="1"/>
<du var="path" def="440" use="485" covered="1"/>
<du var="path" def="440" use="514" covered="1"/>
<du var="path" def="440" use="520" covered="1"/>
<du var="path" def="440" use="521" covered="1"/>
<du var="path" def="440" use="494" covered="1"/>
<du var="path" def="440" use="490" covered="1"/>
<du var="path" def="440" use="488" covered="1"/>
<du var="data" def="440" use="481" covered="1"/>
<du var="data" def="440" use="515" target="515" covered="1"/>
<du var="data" def="440" use="515" target="515" covered="1"/>
<du var="data" def="440" use="515" covered="1"/>
<du var="acl" def="440" use="460" covered="1"/>
<du var="ephemeralOwner" def="440" use="486" covered="1"/>
<du var="ephemeralOwner" def="440" use="491" target="492" covered="1"/>
<du var="ephemeralOwner" def="440" use="491" target="497" covered="1"/>
<du var="ephemeralOwner" def="440" use="492" covered="1"/>
<du var="parentCVersion" def="440" use="468" target="469" covered="1"/>
<du var="parentCVersion" def="440" use="468" target="477" covered="1"/>
<du var="parentCVersion" def="440" use="477" target="478" covered="1"/>
<du var="parentCVersion" def="440" use="477" target="481" covered="1"/>
<du var="parentCVersion" def="440" use="478" covered="1"/>
<du var="xid" def="440" use="479" covered="1"/>
<du var="outputStat" def="440" use="497" target="498" covered="1"/>
<du var="outputStat" def="440" use="497" target="500" covered="1"/>
<du var="outputStat" def="440" use="498" covered="1"/>
<du var="this.nodes" def="440" use="467" covered="1"/>
<du var="this.nodes" def="440" use="483" covered="1"/>
<du var="this.nodes" def="440" use="485" covered="1"/>
<du var="this.aclCache" def="440" use="460" covered="1"/>
<du var="this.nodeDataSize" def="440" use="484" covered="1"/>
<du var="CONTAINER" def="440" use="487" target="488" covered="1"/>
<du var="CONTAINER" def="440" use="487" target="489" covered="1"/>

```

```

<du var="parentName" def="441" use="522" target="522" covered="1"/>
<du var="parentName" def="441" use="522" covered="1"/>
<du var="parentName" def="441" use="510" covered="1"/>
<du var="parentName" def="441" use="507" covered="1"/>
<du var="childName" def="442" use="463" target="464" covered="1"/>
<du var="childName" def="442" use="463" target="467" covered="1"/>
<du var="childName" def="442" use="482" covered="1"/>
<du var="childName" def="442" use="504" target="507" covered="1"/>
<du var="childName" def="442" use="504" target="509" covered="1"/>
<du var="childName" def="442" use="509" target="510" covered="1"/>
<du var="childName" def="442" use="509" target="514" covered="1"/>
<du var="stat" def="443" use="481" covered="1"/>
<du var="parent" def="444" use="445" target="446" covered="1"/>
<du var="parent" def="444" use="445" target="448" covered="1"/>
<du var="parent" def="444" use="448" covered="1"/>
<du var="parent" def="444" use="448" covered="1"/>
<du var="parent" def="444" use="462" covered="1"/>
<du var="parent" def="444" use="467" covered="1"/>
<du var="parent" def="444" use="477" target="478" covered="1"/>
<du var="parent" def="444" use="477" target="481" covered="1"/>
<du var="parent" def="444" use="482" covered="1"/>
<du var="parent" def="444" use="483" covered="1"/>
<du var="parent" def="444" use="478" covered="1"/>
<du var="parent" def="444" use="479" covered="1"/>
<du var="parent" def="444" use="469" covered="1"/>
<du var="parent.stat" def="444" use="477" target="478" covered="1"/>
<du var="parent.stat" def="444" use="477" target="481" covered="1"/>
<du var="parent.stat" def="444" use="478" covered="1"/>
<du var="parent.stat" def="444" use="479" covered="1"/>
<du var="parent.stat" def="444" use="460" covered="1"/>
<du var="acl" def="460" use="481" covered="1"/>
<du var="children" def="462" use="463" target="464" covered="1"/>
<du var="children" def="462" use="463" target="467" covered="1"/>
<du var="parentCVersion" def="470" use="477" target="478" covered="1"/>
<du var="parentCVersion" def="470" use="477" target="481" covered="0"/>
<du var="parentCVersion" def="470" use="478" covered="1"/>
<du var="child" def="481" use="498" covered="1"/>
<du var="ephemeralType" def="486" use="487" target="488" covered="1"/>
<du var="ephemeralType" def="486" use="487" target="489" covered="1"/>
<du var="ephemeralType" def="486" use="489" target="490" covered="1"/>
<du var="ephemeralType" def="486" use="489" target="491" covered="1"/>
<du var="lastPrefix" def="514" use="517" target="518" covered="0"/>
<du var="lastPrefix" def="514" use="517" target="520" covered="1"/>
<du var="lastPrefix" def="514" use="518" covered="0"/>
<du var="bytes" def="515" use="520" covered="1"/>
<du var="bytes" def="515" use="518" covered="0"/>
<counter type="DU" missed="5" covered="105"/>
<counter type="METHOD" missed="0" covered="1"/>

```

figura_14b

442	1. Replaced integer addition with subtraction → KILLED
445	1. negated conditional → KILLED
463	1. negated conditional → KILLED
467	1. removed call to org/apache/zookeeper/server/NodeHashMap::preChange → SURVIVED
468	1. negated conditional → KILLED
470	1. Changed increment from 1 to -1 → SURVIVED
477	1. changed conditional boundary → SURVIVED 2. negated conditional → KILLED
478	1. removed call to org/apache/zookeeper/data/StatPersisted::setCversion → KILLED
479	1. removed call to org/apache/zookeeper/data/StatPersisted::setPzxid → SURVIVED
483	1. removed call to org/apache/zookeeper/server/NodeHashMap::postChange → SURVIVED
487	1. negated conditional → KILLED
489	1. negated conditional → KILLED
491	1. negated conditional → KILLED
492	1. replaced return value with null for org/apache/zookeeper/server/DataTree::lambda\$createNode\$0 → KILLED
497	1. negated conditional → KILLED
498	1. removed call to org/apache/zookeeper/server/DataNode::copyStat → SURVIVED
502	1. negated conditional → KILLED
504	1. negated conditional → KILLED
507	1. removed call to org/apache/zookeeper/common/PathTrie::addPath → SURVIVED
509	1. negated conditional → SURVIVED
510	1. removed call to org/apache/zookeeper/server/DataTree::updateQuotaForPath → SURVIVED
515	1. negated conditional → KILLED
517	1. negated conditional → SURVIVED
518	1. removed call to org/apache/zookeeper/server/DataTree::updateQuotaStat → NO_COVERAGE
520	1. removed call to org/apache/zookeeper/server/DataTree::updateWriteStat → SURVIVED
522	1. negated conditional → SURVIVED

figura_14.1b

```
439     public void createNode(final String path, byte[] data, List<ACL> acl, long ephemeralOwner, int parentCVersion,
440         int lastSlash = path.lastIndexOf('/');
441         String parentName = path.substring(0, lastSlash);
442         String childName = path.substring(lastSlash + 1);
443         StatPersisted stat = createStat(zxid, time, ephemeralOwner);
444         DataNode parent = nodes.get(parentName);
445         if (parent == null) {
446             throw new NoNodeException();
447         }
448         synchronized (parent) {
449             // Add the ACL to ACL cache first, to avoid the ACL not being
450             // created race condition during fuzzy snapshot sync.
451             //
452             // This is the simplest fix, which may add ACL reference count
453             // again if it's already counted in the ACL map of fuzzy
454             // snapshot, which might also happen for deleteNode txn, but
455             // at least it won't cause the ACL not exist issue.
456             //
457             // Later we can audit and delete all non-referenced ACLs from
458             // ACL map when loading the snapshot/txns from disk, like what
459             // we did for the global sessions.
460             Long acls = aclCache.convertAcls(acl);
461
462             Set<String> children = parent.getChildren();
463             if (children.contains(childName)) {
464                 throw new NodeExistsException();
465             }
466
467             nodes.preChange(parentName, parent);
468             if (parentCVersion == -1) {
469                 parentCVersion = parent.stat.getCversion();
470             }
471             parentCVersion++;
472             //
473             // There is possibility that we'll replay txns for a node which
474             // was created and then deleted in the fuzzy range, and it's not
475             // exist in the snapshot, so replay the creation might revert the
476             // cversion and pzxid, need to check and only update when it's
477             // larger.
478             if (parentCVersion > parent.stat.getCversion()) {
479                 parent.stat.setCversion(parentCVersion);
480                 parent.stat.setPzxid(zxid);
481             }
482             DataNode child = new DataNode(data, acls, stat);
483             parent.addChild(childName);
484             nodes.postChange(parentName, parent);
485             nodeDataSize.addAndGet(getNodeSize(path, child.data));
486             nodes.put(path, child);
487             EphemeralType ephemeralType = EphemeralType.get(ephemeralOwner);
488             if (ephemeralType == EphemeralType.CONTAINER) {
489                 containers.add(path);
490             } else if (ephemeralType == EphemeralType.TTL) {
491                 ttls.add(path);
492             } else if (ephemeralOwner != 0) {
493                 HashSet<String> list = ephemerals.computeIfAbsent(ephemeralOwner, k -> new HashSet<>());
494                 synchronized (list) {
495                     list.add(path);
496                 }
497             }
498             if (outputStat != null) {
499                 child.copyStat(outputStat);
500             }
501             // now check if its one of the zookeeper node child
502             if (parentName.startsWith(quotaZookeeper)) {
503                 // now check if it's the limit node
504                 if (Quotas.limitNode.equals(childName)) {
505                     // this is the limit node
506                     // get the parent and add it to the trie
507                     pTrie.addPath(Quotas.trimQuotaPath(parentName));
508                 }
509                 if (Quotas.statNode.equals(childName)) {
510                     updateQuotaForPath(Quotas.trimQuotaPath(parentName));
511                 }
512             }
513
514             String lastPrefix = getMaxPrefixWithQuota(path);
515             long bytes = data == null ? 0 : data.length;
516             // also check to update the quotas for this node
517             if (lastPrefix != null) { // ok we have some match and need to update
518                 updateQuotaStat(lastPrefix, bytes, 1);
519             }
520             updateWriteStat(path, bytes);
521             dataWatches.triggerWatch(path, Event.EventType.NodeCreated);
522             childWatches.triggerWatch(parentName.equals("") ? "/" : parentName, Event.EventType.NodeChildrenChanged);
523         }
```

figura_15b

copyStatPersisted(StatPersisted, StatPersisted)	0%	n/a	1	1	10	10	1	1
deleteNode(String, long)	85%	65%	7	14	6	51	0	1
getACL(String, Stat)	0%	0%	3	3	7	7	1	1

figura_16b

```

534.     public void deleteNode(String path, long zxid) throws NoNodeException {
535.         int lastSlash = path.lastIndexOf('/');
536.         String parentName = path.substring(0, lastSlash);
537.         String childName = path.substring(lastSlash + 1);
538.
539.         // The child might already be deleted during taking fuzzy snapshot,
540.         // but we still need to update the pzxid here before throw exception
541.         // for no such child
542.         DataNode parent = nodes.get(parentName);
543.         ◆ if (parent == null) {
544.             throw new NoNodeException();
545.         }
546.         synchronized (parent) {
547.             nodes.preChange(parentName, parent);
548.             parent.removeChild(childName);
549.             // Only update pzxid when the zxid is larger than the current pzxid,
550.             // otherwise we might override some higher pzxid set by a CreateTxn,
551.             // which could cause the cversion and pzxid inconsistent
552.             ◆ if (zxid > parent.stat.getPzxid()) {
553.                 parent.stat.setPzxid(zxid);
554.             }
555.             nodes.postChange(parentName, parent);
556.         }
557.
558.         DataNode node = nodes.get(path);
559.         ◆ if (node == null) {
560.             throw new NoNodeException();
561.         }
562.         nodes.remove(path);
563.         synchronized (node) {
564.             aclCache.removeUsage(node.acl);
565.             nodeDataSize.addAndGet(-getNodeSize(path, node.data));
566.         }
567.
568.         // Synchronized to sync the containers and ttls change, probably
569.         // only need to sync on containers and ttls, will update it in a
570.         // separate patch.
571.         synchronized (parent) {
572.             long owner = node.stat.getEphemeralOwner();
573.             EphemeralType ephemeralType = EphemeralType.get(owner);
574.             ◆ if (ephemeralType == EphemeralType.CONTAINER) {
575.                 containers.remove(path);
576.             ◆ else if (ephemeralType == EphemeralType.TTL) {
577.                 ttls.remove(path);
578.             ◆ else if (owner != 0) {
579.                 Set<String> nodes = ephemerals.get(owner);
580.                 ◆ if (nodes != null) {
581.                     synchronized (nodes) {
582.                         nodes.remove(path);
583.                     }
584.                 }
585.             }
586.         }
587.
588.         ◆ if (parentName.startsWith(procZookeeper) && Quotas.limitNode.equals(childName)) {
589.             // delete the node in the trie.
590.             // we need to update the trie as well
591.             pTribe.deletePath(Quotas.trimQuotaPath(parentName));
592.         }
593.
594.         // also check to update the quotas for this node
595.         String lastPrefix = getMaxPrefixWithQuota(path);
596.         ◆ if (lastPrefix != null) {
597.             // ok we have some match and need to update
598.             long bytes;
599.             synchronized (node) {
600.                 ◆ bytes = (node.data == null ? 0 : -(node.data.length));
601.             }
602.             updateQuotaStat(lastPrefix, bytes, -1);
603.         }
604.
605.         updateWriteStat(path, 0L);
606.
607.         ◆ if (LOG.isTraceEnabled()) {
608.             ZooTrace.logTraceMessage(
609.                 LOG,
610.                 ZooTrace.EVENT_DELIVERY_TRACE_MASK,
611.                 "dataWatches.triggerWatch " + path);
612.             ZooTrace.logTraceMessage(
613.                 LOG,
614.                 ZooTrace.EVENT_DELIVERY_TRACE_MASK,
615.                 "childWatches.triggerWatch " + parentName);
616.         }
617.
618.         WatcherOrBitSet processed = dataWatches.triggerWatch(path, EventType.NodeDeleted);
619.         childWatches.triggerWatch(path, EventType.NodeDeleted, processed);
620.         ◆ childWatches.triggerWatch(".".equals(parentName) ? "/" : parentName, EventType.NodeChildrenChanged);
621.     }

```

figura_17b

```

<method name="deleteNode" desc="(Ljava/lang/String;)V">
    <du var="this" def="535" use="547" covered="1"/>
    <du var="this" def="535" use="555" covered="1"/>
    <du var="this" def="535" use="558" covered="1"/>
    <du var="this" def="535" use="562" covered="1"/>
    <du var="this" def="535" use="564" covered="1"/>
    <du var="this" def="535" use="565" covered="1"/>
    <du var="this" def="535" use="595" covered="1"/>
    <du var="this" def="535" use="605" covered="1"/>
    <du var="this" def="535" use="618" covered="1"/>
    <du var="this" def="535" use="619" covered="1"/>
    <du var="this" def="535" use="620" covered="1"/>
    <du var="this" def="535" use="602" covered="0"/>
    <du var="this" def="535" use="591" covered="0"/>
    <du var="this" def="535" use="579" covered="1"/>
    <du var="this" def="535" use="577" covered="1"/>
    <du var="this" def="535" use="575" covered="1"/>
    <du var="path" def="535" use="558" covered="1"/>
    <du var="path" def="535" use="562" covered="1"/>
    <du var="path" def="535" use="565" covered="1"/>
    <du var="path" def="535" use="595" covered="1"/>
    <du var="path" def="535" use="605" covered="1"/>
    <du var="path" def="535" use="618" covered="1"/>
    <du var="path" def="535" use="619" covered="1"/>
    <du var="path" def="535" use="608" covered="1"/>
    <du var="path" def="535" use="582" covered="1"/>
    <du var="path" def="535" use="577" covered="1"/>
    <du var="path" def="535" use="575" covered="1"/>
    <du var="xqid" def="535" use="552" target="553" covered="1"/>
    <du var="xqid" def="535" use="552" target="555" covered="1"/>
    <du var="xqid" def="535" use="553" covered="1"/>
    <du var="this.nodes" def="535" use="547" covered="1"/>
    <du var="this.nodes" def="535" use="555" covered="1"/>
    <du var="this.nodes" def="535" use="558" covered="1"/>
    <du var="this.nodes" def="535" use="562" covered="1"/>
    <du var="this.aclCache" def="535" use="564" covered="1"/>
    <du var="this.nodeDataSize" def="535" use="565" covered="1"/>
    <du var="CONTAINER" def="535" use="574" target="575" covered="1"/>
    <du var="CONTAINER" def="535" use="574" target="576" covered="1"/>
    <du var="this.containers" def="535" use="575" covered="1"/>
    <du var="TTL" def="535" use="576" target="577" covered="1"/>
    <du var="TTL" def="535" use="576" target="578" covered="1"/>
    <du var="this.ttls" def="535" use="577" covered="1"/>
    <du var="this.ephemerals" def="535" use="579" covered="1"/>
    <du var="this.pTrie" def="535" use="591" covered="0"/>
    <du var="LOG" def="535" use="607" target="608" covered="1"/>
    <du var="LOG" def="535" use="607" target="618" covered="0"/>
    <du var="LOG" def="535" use="608" covered="1"/>
    <du var="LOG" def="535" use="612" covered="1"/>

    <du var="parent" def="542" use="546" covered="1"/>
    <du var="parent" def="542" use="546" covered="1"/>
    <du var="parent" def="542" use="547" covered="1"/>
    <du var="parent" def="542" use="548" covered="1"/>
    <du var="parent" def="542" use="552" target="553" covered="1"/>
    <du var="parent" def="542" use="552" target="555" covered="1"/>
    <du var="parent" def="542" use="555" covered="1"/>
    <du var="parent" def="542" use="571" covered="1"/>
    <du var="parent" def="542" use="571" covered="1"/>
    <du var="parent" def="542" use="553" covered="1"/>
    <du var="parent.stat" def="542" use="552" target="553" covered="1"/>
    <du var="parent.stat" def="542" use="552" target="555" covered="1"/>
    <du var="parent.stat" def="542" use="553" covered="1"/>
    <du var="node" def="558" use="559" target="560" covered="1"/>
    <du var="node" def="558" use="559" target="562" covered="1"/>
    <du var="node" def="558" use="563" covered="1"/>
    <du var="node" def="558" use="563" covered="1"/>
    <du var="node" def="558" use="564" covered="1"/>
    <du var="node" def="558" use="565" covered="1"/>
    <du var="node" def="558" use="572" covered="1"/>
    <du var="node" def="558" use="599" covered="0"/>
    <du var="node" def="558" use="599" covered="0"/>
    <du var="node" def="558" use="600" target="600" covered="0"/>
    <du var="node" def="558" use="600" target="600" covered="0"/>
    <du var="node" def="558" use="600" covered="0"/>
    <du var="node.acl" def="558" use="564" covered="1"/>
    <du var="node.data" def="558" use="565" covered="1"/>
    <du var="node.data" def="558" use="600" target="600" covered="0"/>
    <du var="node.data" def="558" use="600" target="600" covered="0"/>
    <du var="node.data" def="558" use="600" covered="0"/>
    <du var="node.stat" def="558" use="572" covered="1"/>
    <du var="owner" def="572" use="578" target="579" covered="1"/>
    <du var="owner" def="572" use="578" target="586" covered="1"/>
    <du var="owner" def="572" use="579" covered="1"/>
    <du var="ephemeralType" def="573" use="574" target="575" covered="1"/>
    <du var="ephemeralType" def="573" use="574" target="576" covered="1"/>
    <du var="ephemeralType" def="573" use="576" target="577" covered="1"/>
    <du var="ephemeralType" def="573" use="576" target="578" covered="1"/>
    <du var="nodes" def="579" use="580" target="581" covered="1"/>
    <du var="nodes" def="579" use="580" target="586" covered="0"/>
    <du var="nodes" def="579" use="581" covered="1"/>
    <du var="nodes" def="579" use="581" covered="1"/>
    <du var="nodes" def="579" use="582" covered="1"/>
    <du var="lastPrefix" def="595" use="596" target="599" covered="0"/>
    <du var="lastPrefix" def="595" use="596" target="605" covered="1"/>
    <du var="lastPrefix" def="595" use="602" covered="0"/>
    <counter type="DU" missed="21" covered="96"/>
    <counter type="METHOD" missed="0" covered="1"/>
</method>

```

figura_18b

reportDigestMismatch(long)	0%	0%	2	2	6	6	1	1
deleteNode(String, long)	90%	80%	4	14	4	51	0	1
getEphemerals(long)	0%	0%	2	2	5	5	1	1

figura_19b

```

534. public void deleteNode(String path, long zxid) throws NoNodeException {
535.     int lastSlash = path.lastIndexOf('/');
536.     String parentName = path.substring(0, lastSlash);
537.     String childName = path.substring(lastSlash + 1);
538.
539.     // The child might already be deleted during taking fuzzy snapshot,
540.     // but we still need to update the pzxid here before throw exception
541.     // for no such child
542.     DataNode parent = nodes.get(parentName);
543.     ◆ if (parent == null) {
544.         throw new NoNodeException();
545.     }
546.     synchronized (parent) {
547.         nodes.preChange(parentName, parent);
548.         parent.removeChild(childName);
549.         // Only update pzxid when the zxid is larger than the current pzxid,
550.         // otherwise we might override some higher pzxid set by a CreateTxn,
551.         // which could cause the cversion and pzxid inconsistent
552.         ◆ if (zxid > parent.stat.getPzxid()) {
553.             parent.stat.setPzxid(zxid);
554.         }
555.         nodes.postChange(parentName, parent);
556.     }
557.
558.     DataNode node = nodes.get(path);
559.     ◆ if (node == null) {
560.         throw new NoNodeException();
561.     }
562.     nodes.remove(path);
563.     synchronized (node) {
564.         aclCache.removeUsage(node.acl);
565.         nodeDataSize.addAndGet(-getNodeSize(path, node.data));
566.     }
567.
568.     // Synchronized to sync the containers and ttls change, probably
569.     // only need to sync on containers and ttls, will update it in a
570.     // separate patch.
571.     synchronized (parent) {
572.         long owner = node.stat.getEphemeralOwner();
573.         EphemeralType ephemeralType = EphemeralType.get(owner);
574.         ◆ if (ephemeralType == EphemeralType.CONTAINER) {
575.             containers.remove(path);
576.         } else if (ephemeralType == EphemeralType.TTL) {
577.             ttls.remove(path);
578.         } else if (owner != 0) {
579.             Set<String> nodes = ephemerals.get(owner);
580.             ◆ if (nodes != null) {
581.                 synchronized (nodes) {
582.                     nodes.remove(path);
583.                 }
584.             }
585.         }
586.     }
587.
588.     ◆ if (parentName.startsWith(procZookeeper) && Quotas.limitNode.equals(childName)) {
589.         // delete the node in the trie.
590.         // we need to update the trie as well
591.         pTrie.deletePath(Quotas.trimQuotaPath(parentName));
592.     }
593.
594.     // also check to update the quotas for this node
595.     String lastPrefix = getMaxPrefixWithQuota(path);
596.     ◆ if (lastPrefix != null) {
597.         // ok we have some match and need to update
598.         long bytes;
599.         synchronized (node) {
600.             ◆ bytes = (node.data == null ? 0 : -(node.data.length));
601.         }
602.         updateQuotaStat(lastPrefix, bytes, -1);
603.     }
604.
605.     updateWriteStat(path, 0L);
606.
607.     ◆ if (LOG.isTraceEnabled()) {
608.         ZooTrace.logTraceMessage(
609.             LOG,
610.             ZooTrace.EVENT_DELIVERY_TRACE_MASK,
611.             "dataWatches.triggerWatch " + path);
612.         ZooTrace.logTraceMessage(
613.             LOG,
614.             ZooTrace.EVENT_DELIVERY_TRACE_MASK,
615.             "childWatches.triggerWatch " + parentName);
616.     }
617.
618.     WatcherOrBitSet processed = dataWatches.triggerWatch(path, EventType.NodeDeleted);
619.     childWatches.triggerWatch(path, EventType.NodeDeleted, processed);
620.     ◆ childWatches.triggerWatch("'" + parentName + "' : " + parentName, EventType.NodeChildrenChanged);
621.
622. }
```

figura_20b

```
<method name="deleteNode" desc="(Ljava/lang/String;)V">
<du var="this" def="535" use="547" covered="1"/>
<du var="this" def="535" use="555" covered="1"/>
<du var="this" def="535" use="558" covered="1"/>
<du var="this" def="535" use="562" covered="1"/>
<du var="this" def="535" use="564" covered="1"/>
<du var="this" def="535" use="565" covered="1"/>
<du var="this" def="535" use="595" covered="1"/>
<du var="this" def="535" use="605" covered="1"/>
<du var="this" def="535" use="618" covered="1"/>
<du var="this" def="535" use="619" covered="1"/>
<du var="this" def="535" use="620" covered="1"/>
<du var="this" def="535" use="602" covered="0"/>
<du var="this" def="535" use="591" covered="1"/>
<du var="this" def="535" use="579" covered="1"/>
<du var="this" def="535" use="577" covered="1"/>
<du var="this" def="535" use="575" covered="1"/>
<du var="path" def="535" use="558" covered="1"/>
<du var="path" def="535" use="562" covered="1"/>
<du var="path" def="535" use="565" covered="1"/>
<du var="path" def="535" use="595" covered="1"/>
<du var="path" def="535" use="605" covered="1"/>
<du var="path" def="535" use="618" covered="1"/>
<du var="path" def="535" use="619" covered="1"/>
<du var="path" def="535" use="608" covered="1"/>
<du var="path" def="535" use="582" covered="1"/>
<du var="path" def="535" use="577" covered="1"/>
<du var="path" def="535" use="575" covered="1"/>
<du var="xid" def="535" use="552" target="553" covered="1"/>
<du var="xid" def="535" use="552" target="555" covered="1"/>
<du var="xid" def="535" use="553" covered="1"/>
<du var="this.nodes" def="535" use="547" covered="1"/>
<du var="this.nodes" def="535" use="555" covered="1"/>
<du var="this.nodes" def="535" use="558" covered="1"/>
<du var="this.nodes" def="535" use="562" covered="1"/>
<du var="this.aclCache" def="535" use="564" covered="1"/>
<du var="this.nodeDataSize" def="535" use="565" covered="1"/>
<du var="CONTAINER" def="535" use="574" target="575" covered="1"/>
<du var="CONTAINER" def="535" use="574" target="576" covered="1"/>
<du var="this.containers" def="535" use="575" covered="1"/>
<du var="TTL" def="535" use="576" target="577" covered="1"/>
<du var="TTL" def="535" use="576" target="578" covered="1"/>
<du var="this.ttls" def="535" use="577" covered="1"/>
<du var="this.ephemerals" def="535" use="579" covered="1"/>
<du var="this.pTrie" def="535" use="591" covered="1"/>
<du var="LOG" def="535" use="607" target="608" covered="1"/>
<du var="LOG" def="535" use="607" target="618" covered="0"/>
<du var="LOG" def="535" use="608" covered="1"/>
<du var="LOG" def="535" use="612" covered="1"/>
<du var="parent" def="542" use="540" covered="1"/>
<du var="parent" def="542" use="546" covered="1"/>
<du var="parent" def="542" use="547" covered="1"/>
<du var="parent" def="542" use="548" covered="1"/>
<du var="parent" def="542" use="552" target="553" covered="1"/>
<du var="parent" def="542" use="552" target="555" covered="1"/>
<du var="parent" def="542" use="555" covered="1"/>
<du var="parent" def="542" use="571" covered="1"/>
<du var="parent" def="542" use="571" covered="1"/>
<du var="parent" def="542" use="553" covered="1"/>
<du var="parent.stat" def="542" use="552" target="553" covered="1"/>
<du var="parent.stat" def="542" use="552" target="555" covered="1"/>
<du var="parent.stat" def="542" use="553" covered="1"/>
<du var="node" def="558" use="559" target="560" covered="1"/>
<du var="node" def="558" use="559" target="562" covered="1"/>
<du var="node" def="558" use="563" covered="1"/>
<du var="node" def="558" use="563" covered="1"/>
<du var="node" def="558" use="564" covered="1"/>
<du var="node" def="558" use="565" covered="1"/>
<du var="node" def="558" use="572" covered="1"/>
<du var="node" def="558" use="599" covered="0"/>
<du var="node" def="558" use="599" covered="0"/>
<du var="node" def="558" use="600" target="600" covered="0"/>
<du var="node" def="558" use="600" target="600" covered="0"/>
<du var="node" def="558" use="600" covered="0"/>
<du var="node.acl" def="558" use="564" covered="1"/>
<du var="node.data" def="558" use="565" covered="1"/>
<du var="node.data" def="558" use="600" target="600" covered="0"/>
<du var="node.data" def="558" use="600" target="600" covered="0"/>
<du var="node.data" def="558" use="600" covered="0"/>
<du var="node.stat" def="558" use="572" covered="1"/>
<du var="owner" def="572" use="578" target="579" covered="1"/>
<du var="owner" def="572" use="578" target="586" covered="1"/>
<du var="owner" def="572" use="579" covered="1"/>
<du var="ephemeralType" def="573" use="574" target="575" covered="1"/>
<du var="ephemeralType" def="573" use="574" target="576" covered="1"/>
<du var="ephemeralType" def="573" use="576" target="577" covered="1"/>
<du var="ephemeralType" def="573" use="576" target="578" covered="1"/>
<du var="nodes" def="579" use="580" target="581" covered="1"/>
<du var="nodes" def="579" use="580" target="586" covered="0"/>
<du var="nodes" def="579" use="581" covered="1"/>
<du var="nodes" def="579" use="581" covered="1"/>
<du var="nodes" def="579" use="582" covered="1"/>
<du var="lastPrefix" def="595" use="596" target="599" covered="0"/>
<du var="lastPrefix" def="595" use="596" target="605" covered="1"/>
<du var="lastPrefix" def="595" use="602" covered="0"/>
<counter type="DU" missed="14" covered="103"/>
<counter type="METHOD" missed="0" covered="1"/>
</method>
```

figura_21b

```

537 1. Replaced integer addition with subtraction → KILLED
543 1. negated conditional → KILLED
547 1. removed call to org/apache/zookeeper/server/NodeHashMap::preChange → SURVIVED
552 1. changed conditional boundary → SURVIVED
2. negated conditional → KILLED
553 1. removed call to org/apache/zookeeper/data/StatPersisted::setPzxid → KILLED
555 1. removed call to org/apache/zookeeper/server/NodeHashMap::postChange → SURVIVED
559 1. negated conditional → KILLED
564 1. removed call to org/apache/zookeeper/server/ReferenceCountedACLCache::removeUsage → SURVIVED
565 1. removed negation → SURVIVED
574 1. negated conditional → SURVIVED
576 1. negated conditional → SURVIVED
578 1. negated conditional → SURVIVED
580 1. negated conditional → SURVIVED
588 1. negated conditional → SURVIVED
2. negated conditional → SURVIVED
591 1. removed call to org/apache/zookeeper/common/PathTrie::deletePath → SURVIVED
596 1. negated conditional → SURVIVED
600 1. removed negation → NO_COVERAGE
2. negated conditional → NO_COVERAGE
602 1. removed call to org/apache/zookeeper/server/DataTree::updateQuotaStat → NO_COVERAGE
605 1. removed call to org/apache/zookeeper/server/DataTree::updateWriteStat → SURVIVED
608 1. removed call to org/apache/zookeeper/server/ZooTrace::logTraceMessage → SURVIVED
612 1. removed call to org/apache/zookeeper/server/ZooTrace::logTraceMessage → SURVIVED
620 1. negated conditional → SURVIVED

```

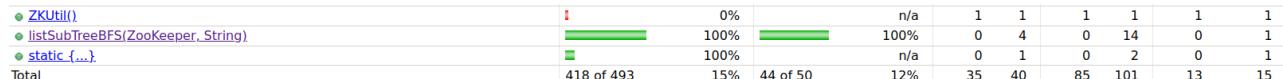
figura_22b

```

537 1. Replaced integer addition with subtraction → KILLED
543 1. negated conditional → KILLED
547 1. removed call to org/apache/zookeeper/server/NodeHashMap::preChange → SURVIVED
552 1. changed conditional boundary → SURVIVED
2. negated conditional → KILLED
553 1. removed call to org/apache/zookeeper/data/StatPersisted::setPzxid → KILLED
555 1. removed call to org/apache/zookeeper/server/NodeHashMap::postChange → SURVIVED
559 1. negated conditional → KILLED
564 1. removed call to org/apache/zookeeper/server/ReferenceCountedACLCache::removeUsage → SURVIVED
565 1. removed negation → SURVIVED
574 1. negated conditional → KILLED
576 1. negated conditional → SURVIVED
578 1. negated conditional → SURVIVED
580 1. negated conditional → SURVIVED
588 1. negated conditional → KILLED
2. negated conditional → KILLED
591 1. removed call to org/apache/zookeeper/common/PathTrie::deletePath → SURVIVED
596 1. negated conditional → SURVIVED
600 1. removed negation → NO_COVERAGE
2. negated conditional → NO_COVERAGE
602 1. removed call to org/apache/zookeeper/server/DataTree::updateQuotaStat → NO_COVERAGE
605 1. removed call to org/apache/zookeeper/server/DataTree::updateWriteStat → SURVIVED
608 1. removed call to org/apache/zookeeper/server/ZooTrace::logTraceMessage → SURVIVED
612 1. removed call to org/apache/zookeeper/server/ZooTrace::logTraceMessage → SURVIVED
620 1. negated conditional → SURVIVED

```

figura_23b



figura_24b

```

-<method name="listSubTreeBFS" desc="(Lorg/apache/zookeeper/ZooKeeper;Ljava/lang/String;)Ljava/util/List;">
<du var="zk" def="204" use="210" covered="1"/>
<du var="queue" def="204" use="208" target="209" covered="1"/>
<du var="queue" def="204" use="208" target="218" covered="1"/>
<du var="queue" def="204" use="209" covered="1"/>
<du var="queue" def="204" use="214" covered="1"/>
<du var="tree" def="205" use="218" covered="1"/>
<du var="tree" def="205" use="215" covered="1"/>
<du var="node" def="209" use="213" target="213" covered="1"/>
<du var="node" def="209" use="213" target="213" covered="1"/>
<du var="node" def="209" use="213" covered="1"/>
<du var="child" def="211" use="213" covered="1"/>
<counter type="DU" missed="0" covered="14"/>
<counter type="METHOD" missed="0" covered="1"/>
</method>

```

figura_25b

```
201     public static List<String> listSubTreeBFS(
202         ZooKeeper zk,
203         final String pathRoot) throws KeeperException, InterruptedException {
204         Queue<String> queue = new ArrayDeque<>();
205         List<String> tree = new ArrayList<>();
206         queue.add(pathRoot);
207         tree.add(pathRoot);
208     while (!queue.isEmpty()) {
209         String node = queue.poll();
210         List<String> children = zk.getChildren(node, false);
211         for (final String child : children) {
212             // Fix IllegalArgumentException when list "/".
213             final String childPath = (node.equals("/") ? "" : node) + "/" + child;
214             queue.add(childPath);
215             tree.add(childPath);
216         }
217     }
218     return tree;
219 }
```