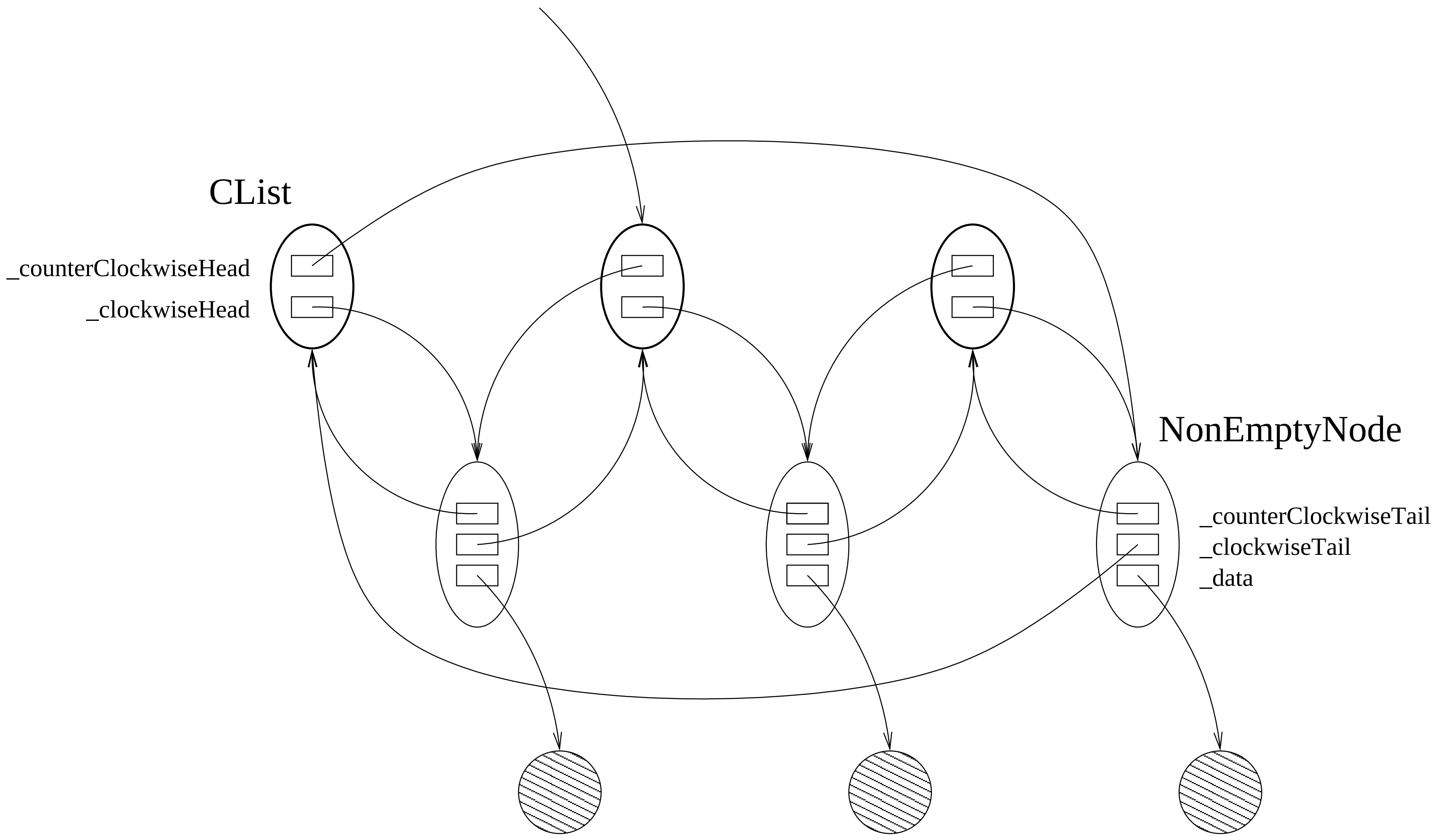
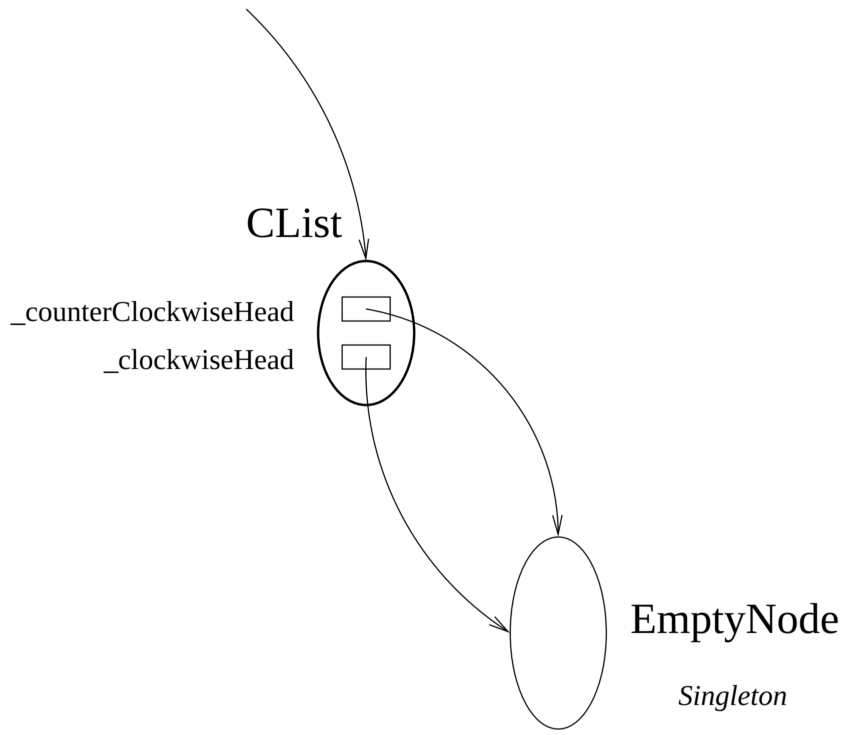
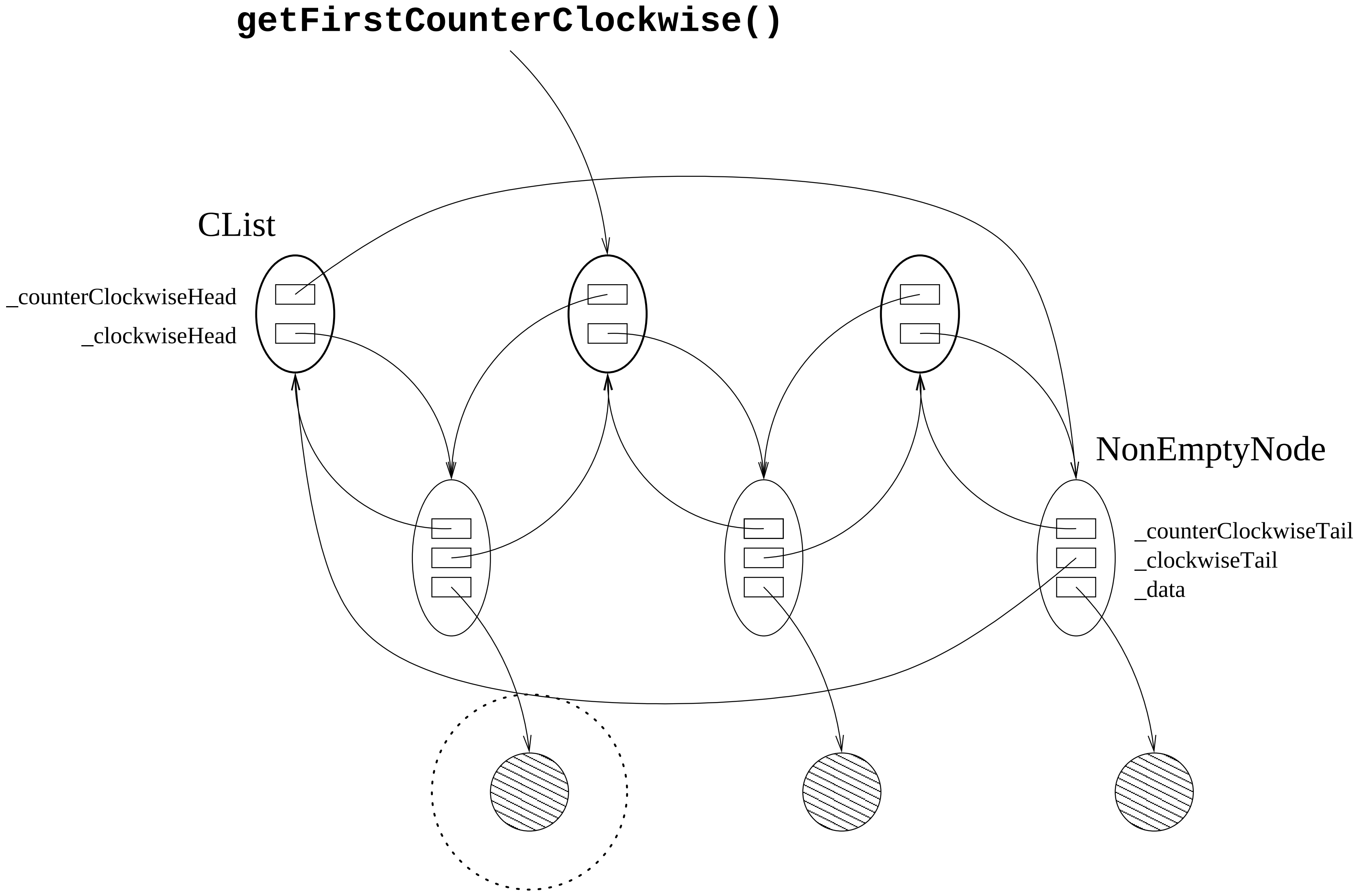
Circular Lists



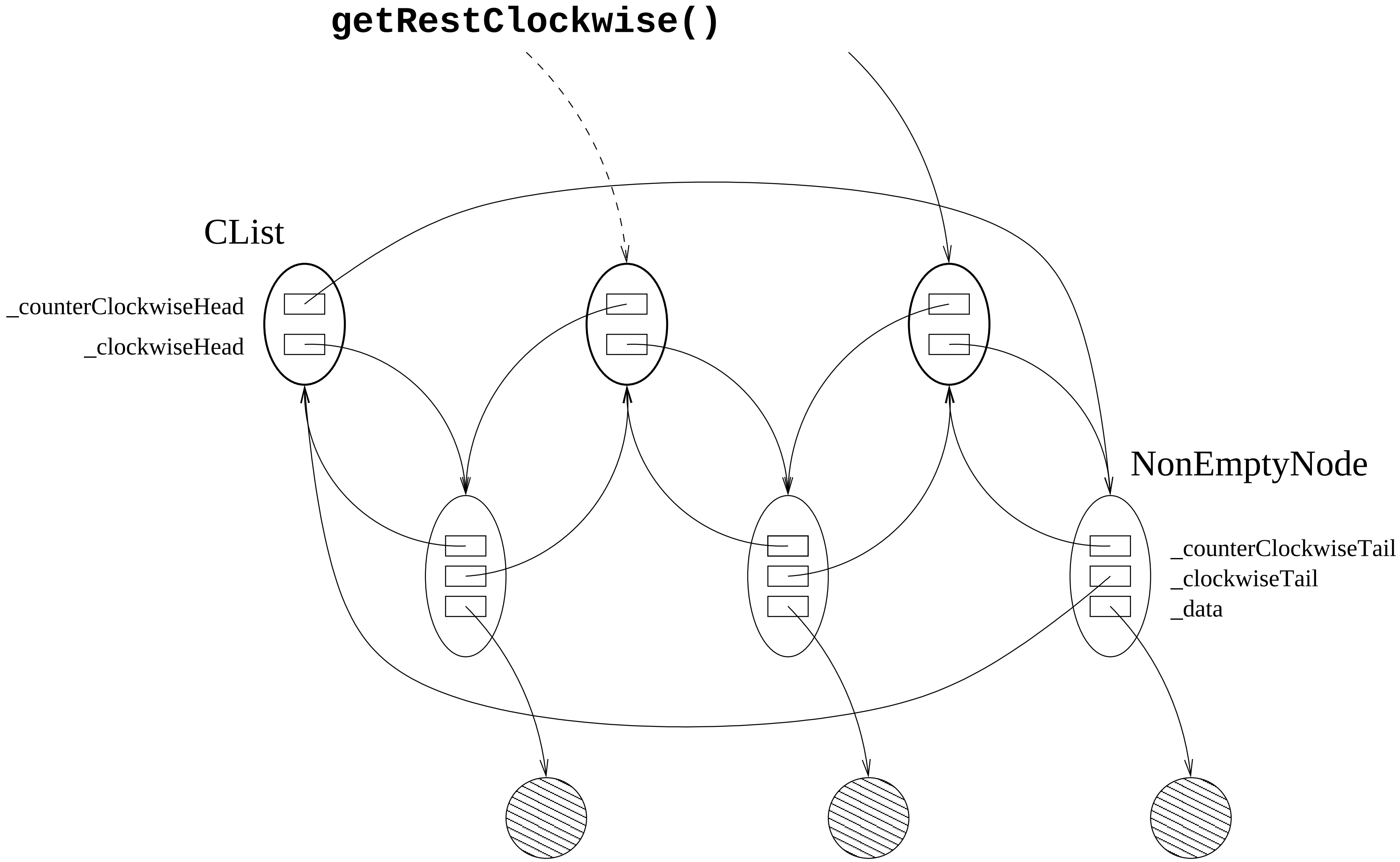
An Empty Circular List



getFirstCounterClockwise()



getRestClockwise()



getRestClockwise()

CList

_counterClockwiseHead

_clockwiseHead

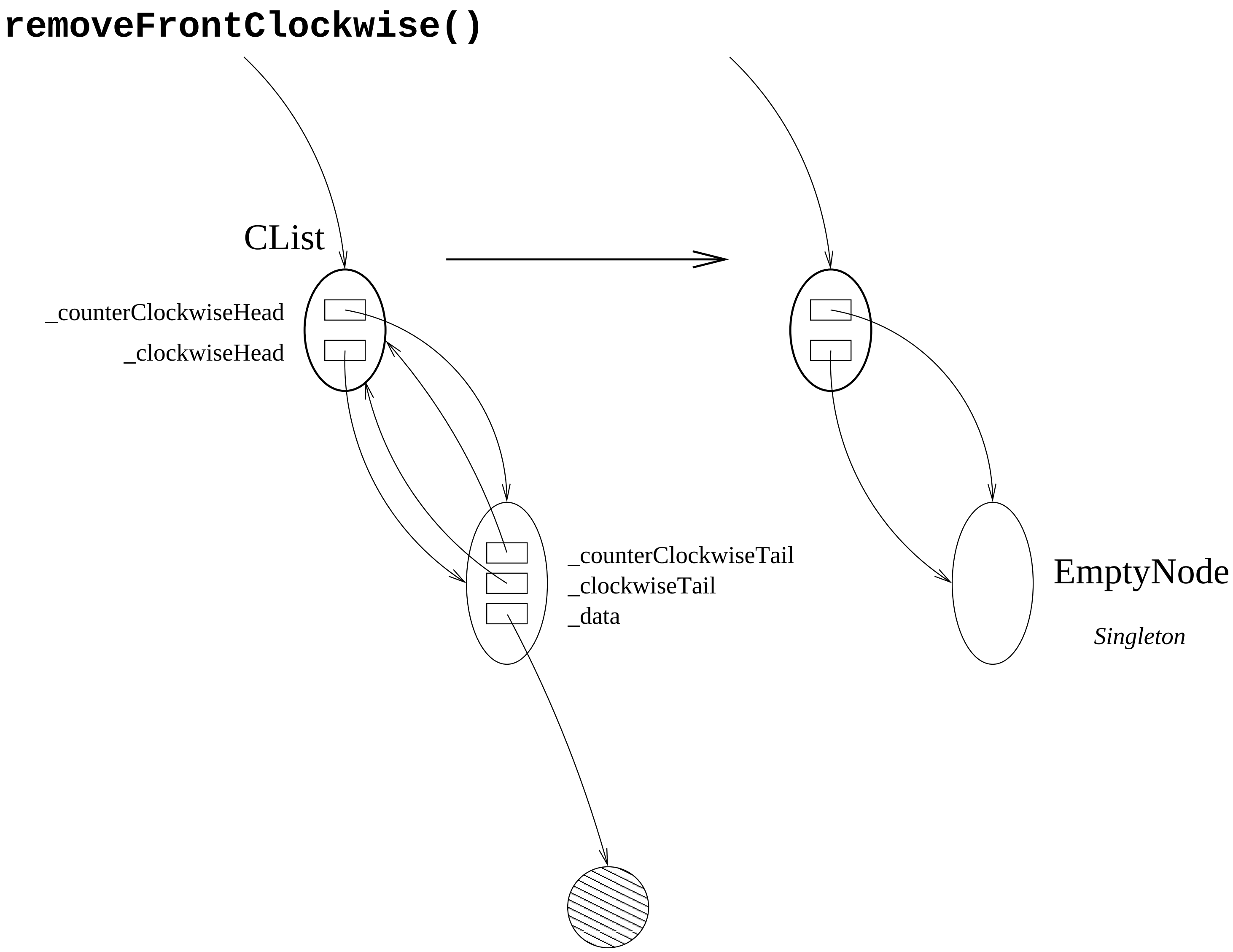
NonEmptyNode

_counterClockwiseTail

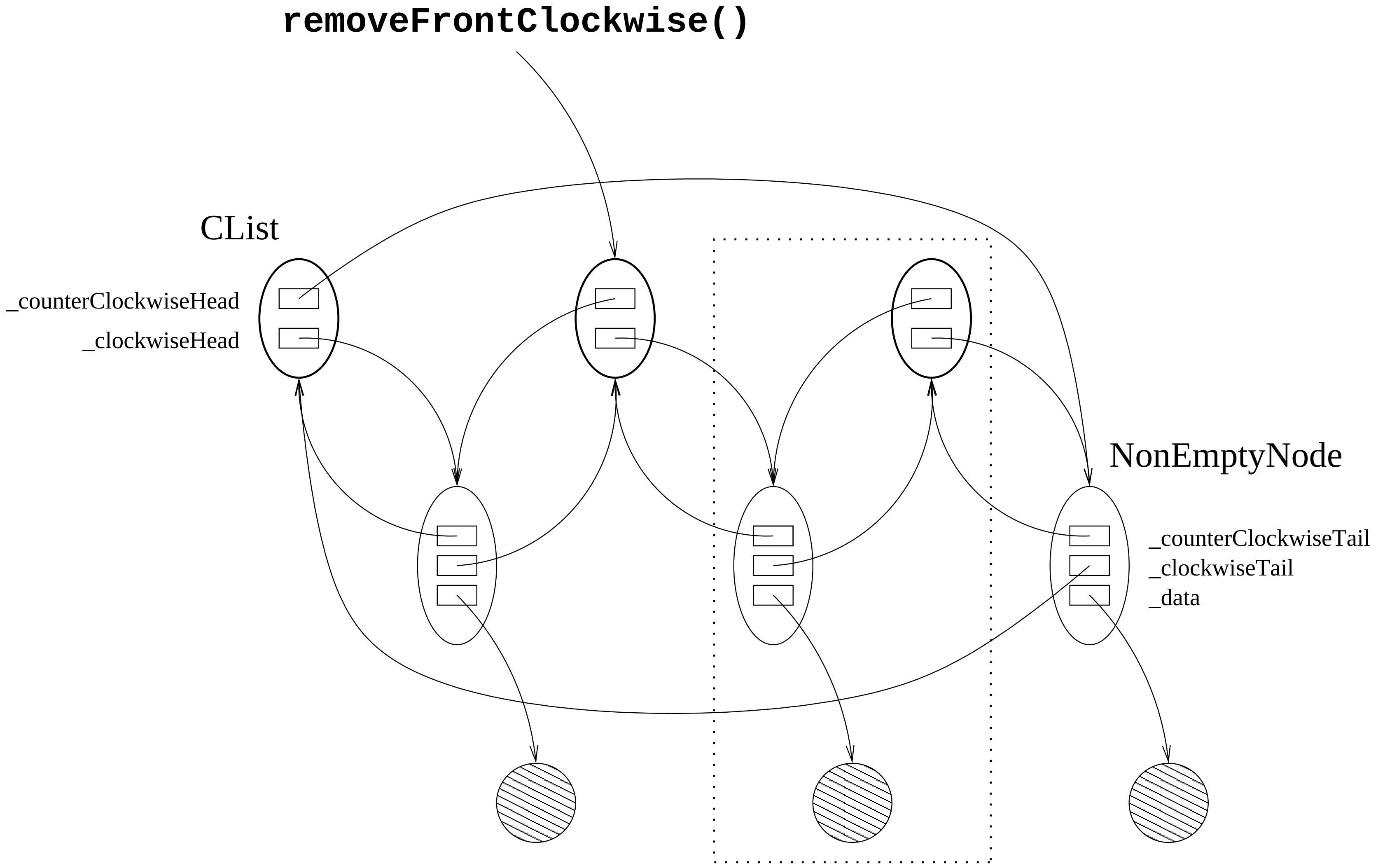
_clockwiseTail

_data

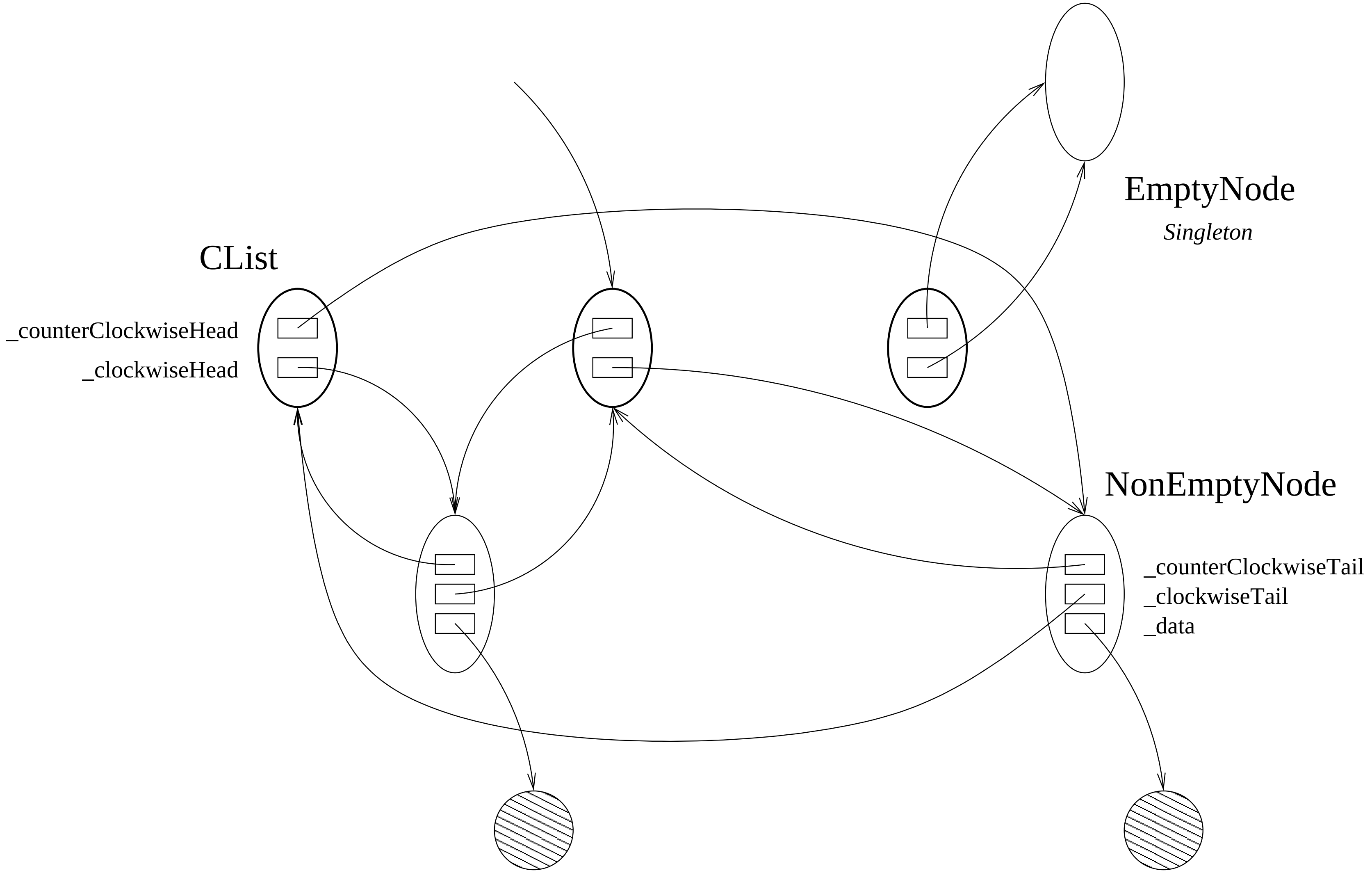
removeFrontClockwise()



Before `removeFrontClockwise()` ...



And After ...


EmptyNode
Singleton
CList
_counterClockwiseHead
_clockwiseHead
NonEmptyNode
_counterClockwiseTail
_clockwiseTail
_data

IOrdered

```
package ordered;

public interface IOrdered {
    public static final int LESS    = -1;
    public static final int EQUAL   =  0;
    public static final int GREATER =  1;

    public int compare(IOrdered other);
}
```

An Ordered Array-based Container

```
private int findIndex(IOrdered key)
{
    int lo = -1;
    int hi = _firstEmptyPair;
    while (lo + 1 != hi) {
        int mid = (lo + hi)/2;
        switch (_pairs[mid].getKey().compare(key)) {
        case IOrdered.EQUAL:   return mid;
        case IOrdered.GREATER: hi = mid;   break;
        case IOrdered.LESS:    lo = mid;   break;
        }
    }
    return lo;
}
```