里包含了另外一些测试套件，也都会被执行。

这也就意味着 TestSuite 是可以递归的，事实上，TestSuite 是一个树状结构，如图 17-3 所示。

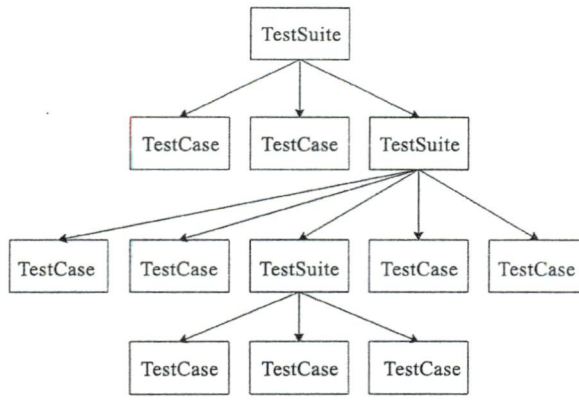

图 17-3　TestSuite 的树状结构

当我们从树的根节点遍历树时，就可以执行所有这些测试用例。传统上进行树的遍历是需要递归编程的，而使用组合模式无须递归也可以遍历树。

首先，TestSuite 和 TestCase 都实现了接口 Test，如下所示：

```
public interface Test {
    public abstract void run(TestResult result);
}
```

当我们调用 TestSuite 的 addTest 方法时，TestSuite 会将输入的对象放入一个数组，如下所示：

```
private Vector<Test> fTests= new Vector<Test>(10);

public void addTest(Test test) {
    fTests.add(test);
}
```

由于 TestCase 和 TestSuite 都实现了 Test 接口，所以执行 addTest 的时候，既可以传入 TestCase，也可以传入 TestSuite。执行 TestSuite 的 run 方法时，会取出这个数组的每个对象，分别执行它们的 run 方法，具体如下：

```
public void run(TestResult result) {
    for (Test each : fTests) {
```