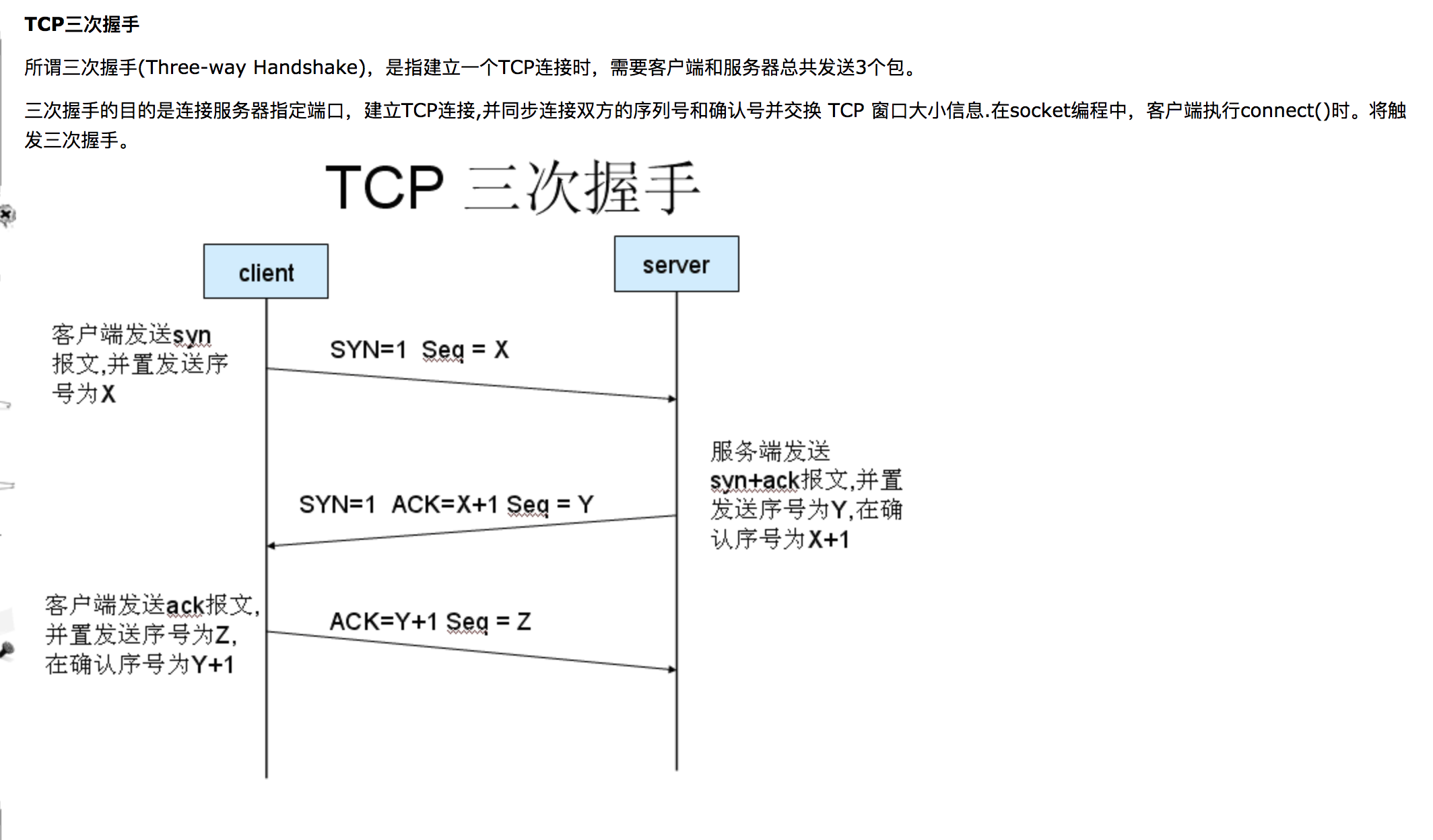
**陈中原：  
三次握手**

****

**王福成：**

JVM中一次完整的GC流程是怎样的，对象如何晋升到老年代？

思路： 先描述一下Java堆内存划分，再解释Minor GC，Major GC，full GC，描述它们之间转化流程。

Java堆 = 老年代 + 新生代

新生代 = Eden + S0 + S1

当 Eden 区的空间满了， Java虚拟机会触发一次 Minor GC，以收集新生代的垃圾，存活下来的对象，则会转移到 Survivor区。

大对象（需要大量连续内存空间的Java对象，如那种很长的字符串）直接进入老年态；

如果对象在Eden出生，并经过第一次Minor GC后仍然存活，并且被Survivor容纳的话，年龄设为1，每熬过一次Minor GC，年龄+1，若年龄超过一定限制（15），则被晋升到老年态。即长期存活的对象进入老年态。

老年代满了而无法容纳更多的对象，Minor GC 之后通常就会进行Full GC，Full GC 清理整个内存堆 – 包括年轻代和年老代。

Major GC 发生在老年代的GC，清理老年区，经常会伴随至少一次Minor GC，比Minor GC慢10倍以上。

苏建飞：

****

**丁昌茂：**

****