Topology程序的执行是在Executor中执行的，即Bolt的执行是BoltExecutor，Spout的执行是SpoutExecutor。
1.Bolt的执行过程

（1）Bolt的执行是BoltExecutors完成的，BoltExecutors实现了EventHandler接口，并实现onEvent()方法

```
1.  public class BoltExecutors extends BaseExecutors implements EventHandler {....}
```

（2）一旦DisruptorQueue有数据被消费，就会触发onEvent()方法

```
1.  @Override
2.  public void onEvent(Object event, long sequence, boolean endOfBatch) throws Except
    ion {
3.      if (event == null) {
4.          return;
5.      }
6.
7.      long start = System.currentTimeMillis();
8.      try {
9.          if (event instanceof Tuple) {
10.             processControlEvent();
11.             processTupleEvent((Tuple) event);
12.         } else if (event instanceof BatchTuple) {
13.             for (Tuple tuple : ((BatchTuple) event).getTuples()) {
14.                 processControlEvent();
15.                 processTupleEvent((Tuple) tuple);
16.             }
17.         } else if (event instanceof TimerTrigger.TimerEvent) {
18.             processTimerEvent((TimerTrigger.TimerEvent) event);
19.         } else {
20.             LOG.warn("Bolt executor received unknown message");
21.         }
22.     } finally {
23.         if (JStormMetrics.enabled) {
24.             exeTime = System.currentTimeMillis() - start;
25.         }
26.     }
27. }
```

（3）在onEvent()方法中，调用processTupleEvent()方法

```
1.  private void processTupleEvent(Tuple tuple) {
2.      task_stats.recv_tuple(tuple.getSourceComponent(), tuple.getSourceStreamId());
3.
4.      if(ackerNum > 0 && tuple.getMessageId().isAnchored()) {
5.          tuple_start_times.put(tuple, System.currentTimeMillis());
6.      }
7.
8.      try {
```

```
        if (!isSystemBolt && tuple.getSourceStreamId().equals(Common.TOPOLOGY_MASTER_CONTROL_STREAM_ID)) {
            backpressureTrigger.handle(tuple);
        } else {
            bolt.execute(tuple);
        }
    } catch (Throwable e) {
        error = e;
        LOG.error("bolt execute error ", e);
        report_error.report(e);
    }

    if (ackerNum == 0) {
        // only when acker is disabled
        // get tuple process latency
        Long latencyStart = (Long) tuple_start_times.remove(tuple);
        if (latencyStart != null && JStormMetrics.enabled) {
            long endTime = System.currentTimeMillis();
            long lifeCycleStart = ((TupleExt) tuple).getCreationTimeStamp();
            task_stats.bolt_acked_tuple(
                    tuple.getSourceComponent(), tuple.getSourceStreamId(), latencyStart, lifeCycleStart, endTime);
        }
    }
}
```

最后，processTupleEvent()方法调用Bolt的execute()方法，并将获取到的Tuple传入。

2.Spout的执行过程

（1）在run()方法中，调用nextTuple()

```
@Override
public void run() {
    if (isFinishInit == false) {
        initWrapper();
    }

    super.nextTuple();
}
```

（2）在nextTuple()方法中，调用Spout的nextTuple()方法

```
public void nextTuple() {
    if (!taskStatus.isRun()) {
        JStormUtils.sleepMs(1);
        return;
    }
```

```
    if (max_spout_pending == null || pending.size() < max_spout_pending) {
        emptyCpuGauge.stop();
        long start = nextTupleTimer.getTime();
        try {
            spout.nextTuple();
        } catch (Throwable e) {}
            finally {
            nextTupleTimer.updateTime(start);
        }
    } else {
        if (isSpoutFullSleep) {
            JStormUtils.sleepMs(1);
        }
        emptyCpuGauge.start();
        // just return, no sleep
    }
}
```

（3）自定义Spout组件，继承BaseRichSpout，并实现nextTuple()方法；在nextTuple()方法中，调用SpoutOutputCollector的emit()方法

```
public List<Integer> emit(List<Object> tuple) {
    return emit(tuple, null);
}
```

（4）即调用SpoutCollector的emit()方法

```
@Override
public List<Integer> emit(String streamId, List<Object> tuple, Object messageId) {
    return sendSpoutMsg(streamId, tuple, messageId, null, null);
}
```

（5）在SpoutCollector的emit()方法中，调用SpoutBatchCollector的sendSpoutMsg()方法

```
protected List<Integer> sendSpoutMsg(String outStreamId, List<Object> values, Object messageId, Integer outTaskId, ICollectorCallback callback) {
    java.util.List<Integer> outTasks = null;
    // LOG.info("spout push message to " + out_stream_id);
    List<MsgInfo> batchTobeFlushed = batchCollector.push(outStreamId, values, outTaskId, null, messageId, getRootId(messageId), callback);
    if (batchTobeFlushed != null && batchTobeFlushed.size() > 0) {
        outTasks = sendBatch(outStreamId, (outTaskId != null ? outTaskId.toString() : null), batchTobeFlushed);
    }
    return outTasks;
}
```

（6）在sendSpoutMsg()中，调用sendBatch()发送数据

```java
public List<Integer> sendBatch(String outStreamId, String outTaskId, List<MsgInfo>
 batchTobeFlushed) {
    long startTime = emitTotalTimer.getTime();
    try {
        List<Integer> ret = null;
        Map<List<Integer>, List<MsgInfo>> outTasks;

        if (outTaskId != null) {
            outTasks = sendTargets.getBatch(Integer.valueOf(outTaskId), outStreamI
d, batchTobeFlushed);
        } else {
            outTasks = sendTargets.getBatch(outStreamId, batchTobeFlushed);
        }

        if (outTasks == null || outTasks.size() == 0) {
            // don't need send tuple to other task
            return new ArrayList<Integer>();
        }

        Map<Long, MsgInfo> ackBatch = new HashMap<Long, MsgInfo>();
        for (Map.Entry<List<Integer>, List<MsgInfo>> entry : outTasks.entrySet())
{

            List<Integer> tasks = entry.getKey();
            List<MsgInfo> batch = entry.getValue();

            for(int i = 0; i < tasks.size(); i++){
                Integer t = tasks.get(i);
                BatchTuple batchTuple = new BatchTuple(t, batch.size());
                for (MsgInfo msg : batch) {
                    MessageId msgId = getMessageId((SpoutMsgInfo) msg, ackBatch);
                    TupleImplExt tp = new TupleImplExt(topology_context, msg.value
s, task_id, msg.streamId, msgId);
                    tp.setTargetTaskId(t);
                    batchTuple.addToBatch(tp);
                }
                transfer_fn.transfer(batchTuple);
            }

            for (MsgInfo msg : batch) {
                if (msg.callback != null) {
                    msg.callback.execute(tasks);
                }
            }
        }


        if (ackBatch.size() > 0) {
            sendBatch(Acker.ACKER_INIT_STREAM_ID, null, new ArrayList<MsgInfo>(ack
Batch.values()));
        }

        return ret;
```

```
49.     } finally {
50.         emitTotalTimer.updateTime(startTime);
51.     }
52. 
53. }
```