Channel 中被删除。

我们需要注意的是，如果 Sink 读取完一个事件，该事件对于其他 Sink 就是不可用的，除非该 Sink 回滚事务，导致该 Sink 中的事务能被再次读取。这是专门设计用来当多个 Sink 操作相同的 Channel 时，避免重复的。Channel 的每一个事件实际上可以读取和提交一次，之后事件从 Channel 中移除。

取决于所使用的特定的 Channel，即使机器或 JVM 重启，Channel 中的事件也可能是可用的。Sink 写所有的事件到任意它所支持的地方也有可能失败，因此必须重试。在这种情况下，Sink 在事务中使用 rollback 方法回滚整个事务。一旦事务被回滚到 Sink 这边，Channel 重新存储事件到 Channel 并使它们对 Sink 可用以用来读取。在 source-side 事务回滚的情况下，就好像这个事务从来没有发生过，在事务期间写入的事件从来没有写入 Channel。当回滚是由超时或其他失败引起的，这时事件可能已经提交到下一阶段的 Channel 中，回滚可能会造成重复。

当事务提交或回滚之后，通过调用 close 方法关闭事务，来清理事务使用的任何资源。图 4-1 说明了事务的工作流。

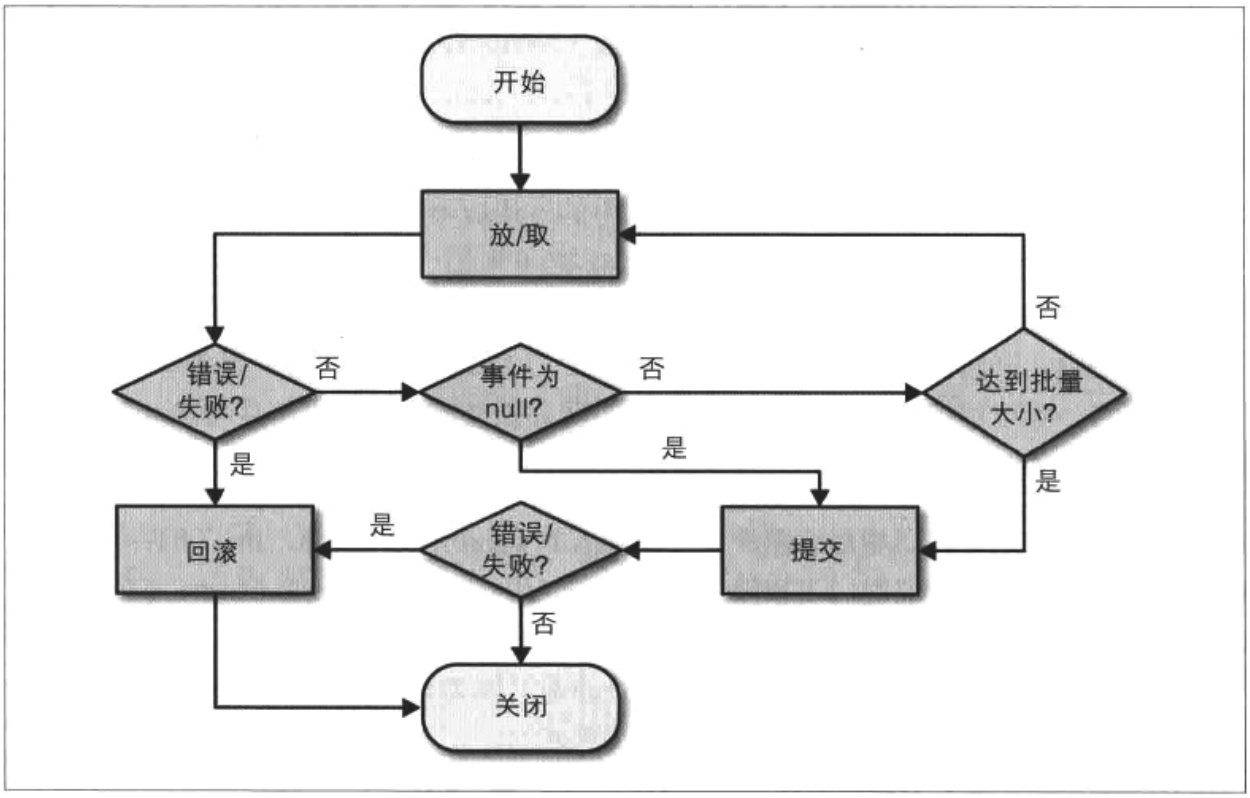

图4-1 事务工作流

单个事务不能同时写入和读取事件。这保证了 Source 只能往 Channel 中放入事件，Sink