Content模块采用DD路径方法测试，共计6个待测函数Create，CreateTag，Update，Query，Delete，Check，并对它们尽可能采用不同的覆盖方法进行测试，包括语句覆盖(Create)、条件覆盖(Update)、多条件覆盖(Check)、分支+循环覆盖(Delete)、包含最多2次循环的所有路径(Query)、路径测试(CreateTag)。

**语句覆盖**即测试用例覆盖所有执行语句；

**条件覆盖**即测试用例覆盖每一个条件；

**多条件覆盖**即测试用例覆盖每种条件组合；

**循环覆盖**即测试用例覆盖不循环、1次循环、2次循环、多次循环等情况；

**包含最多2次循环的所有路径**意义显然；

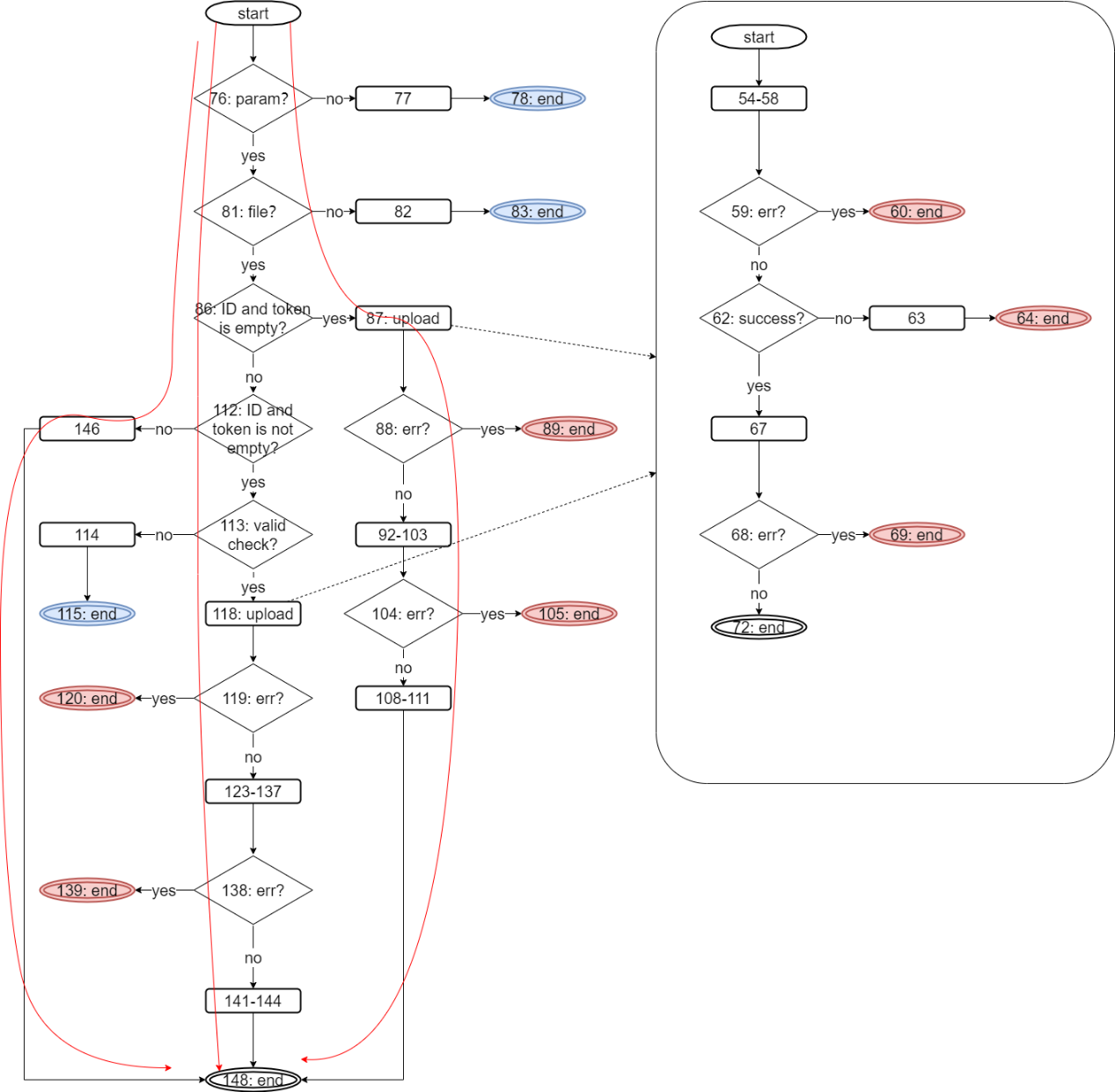
**路径测试**即测试用例要覆盖所有可能执行的路径。

以Check多条件覆盖为例：

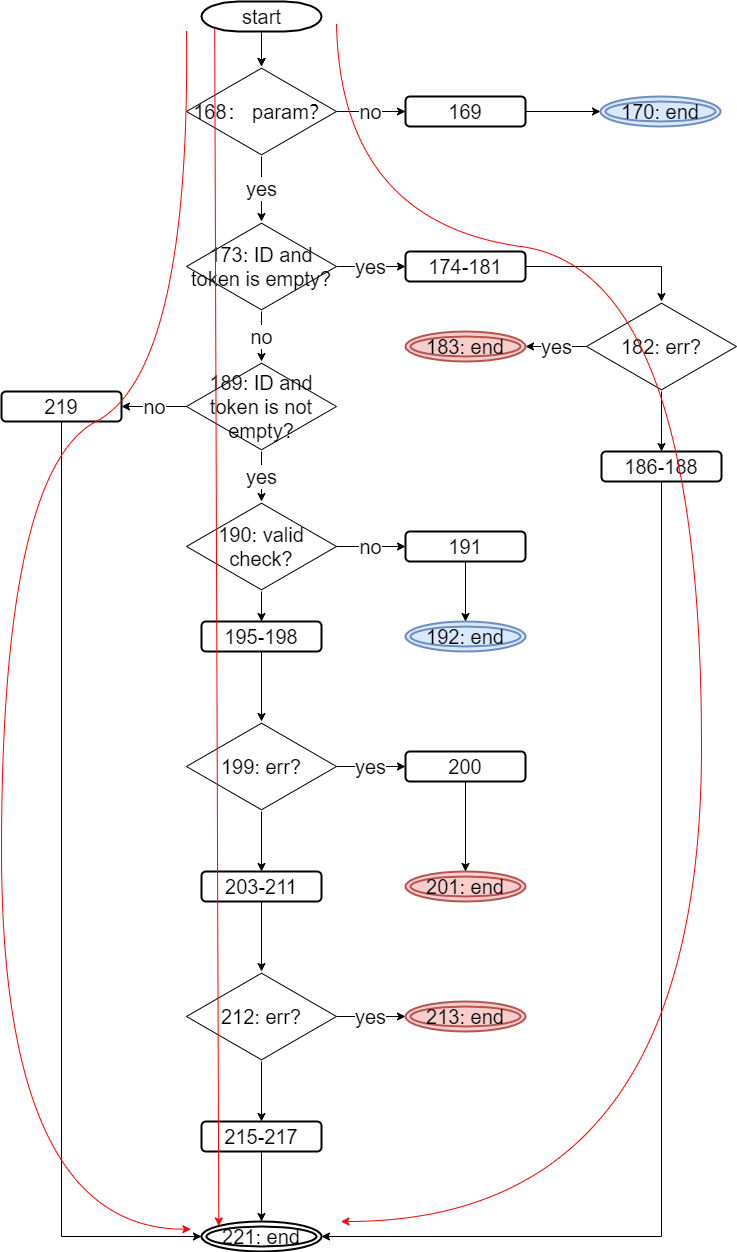
1. 470行检验两个参数contentID和contentToken是否为空，因此需要有4种组合来遍历。
2. 474行跳转入内部函数validCheck。
3. 484行结果总为true，也即该检查事实上是不必要的代码，因为在外部已经检查。所以会导致485行是无论如何不可达的代码，无法覆盖测试。
4. 492行检查了contentID是否有效，需要2种组合，且之前的用例中已经包含了有效的那一种。
5. 500行检查了contentToken是否有效，需要2种组合，且之前的用例中已经包含了有效的那一种。
6. 总计6个测试用例，详见表格和路径图。

测试用例见附件【DD测试用例表.xlsx】

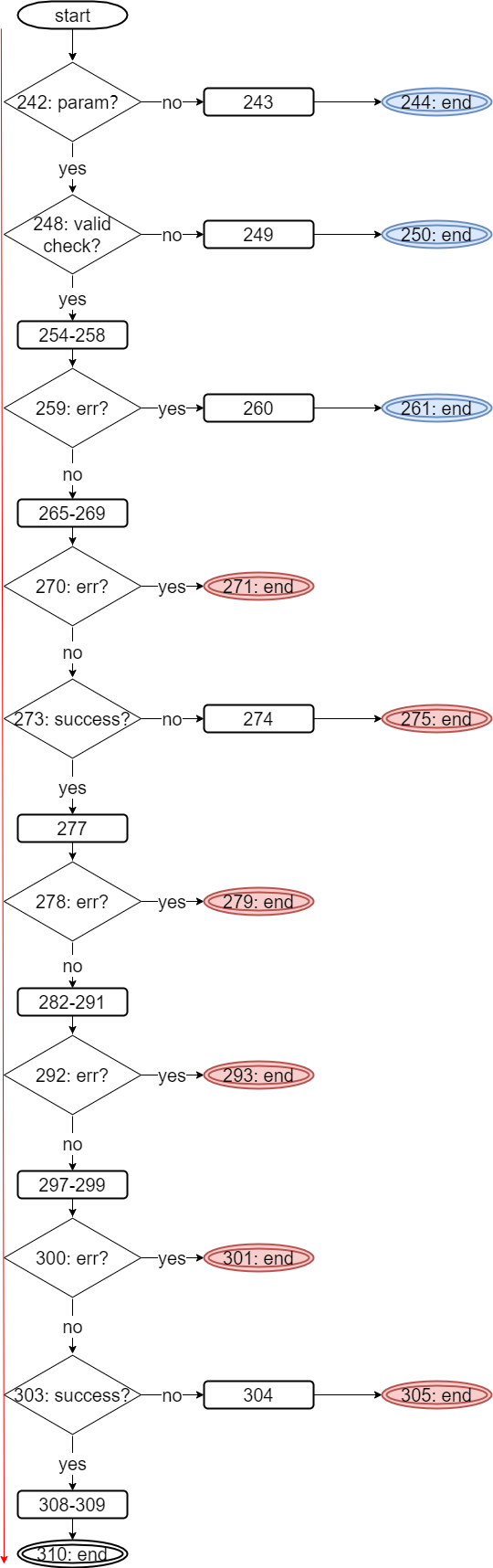
DD路径图见附件【DD路径图 xxxxx.png】，其中数字代表代码行号，红色路径是最终成功执行的路径，蓝色标注的节点意味着该种错误终点测试可达（包括正常执行程序，以及可以通过mock模拟的部分异常测试数据），红色标注的节点意味着该种错误终点测试不可达（包括数据库崩溃等极小概率事件）。



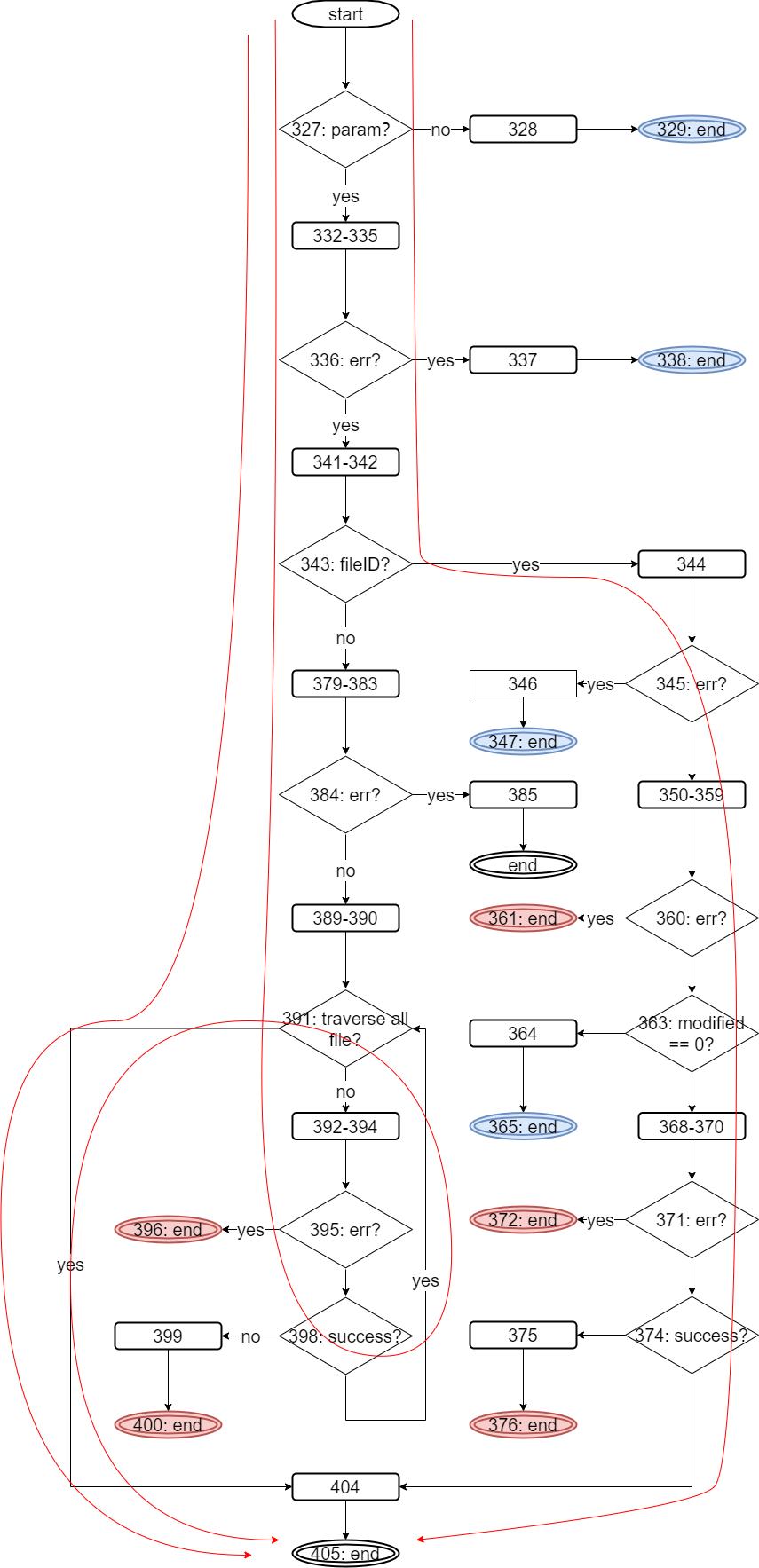
1. Content.Create



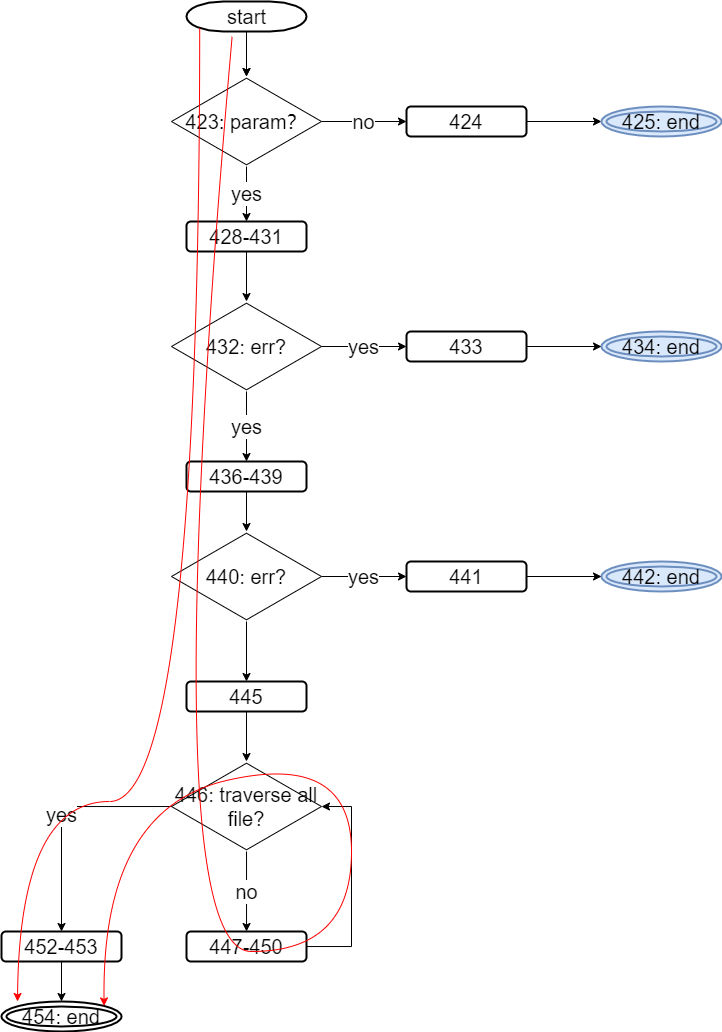
1. Content.CreateTag



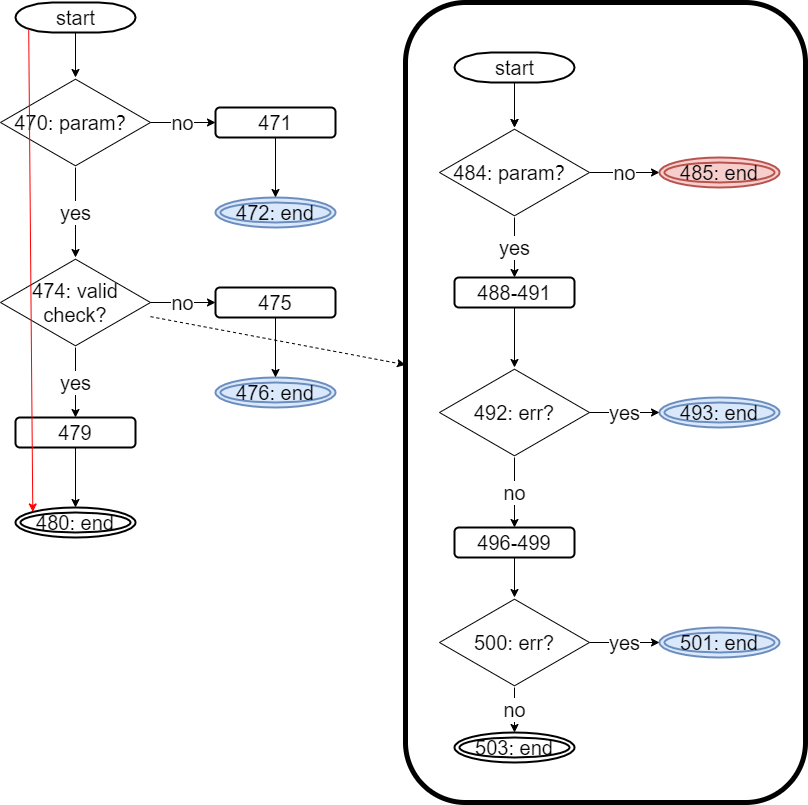
1. Content.Update



1. Content.Query



1. Content.Delete



1. Content.Check