

2021 年国内软件质量调查报告

《软件质量报道》编写

腾讯 WeTest、Testin 云测试社区、

MeterSphere 开源社区、AITest 掌动国产化社区、

PerfMa 质量技术社区、CSDN 社区、Thoughtworks 社区

(排名不分先后)

校对

2022 年 1 月

目录

2020 年国内软件质量调查报告	1
一、背景.....	3
二、参与调查的企业、团队和产品情况.....	3
三、软件质量整体状况.....	7
四、需求质量.....	10
五、设计质量.....	12
六、代码质量.....	14
七、测试质量.....	16
八、今明两年的软件质量管理工作.....	22
九、附录：呐喊、快乐分享和金句.....	24
十、关于组织者.....	42

一、背景

2020 年，“软件质量报道”公众号联合腾讯 WeTest 和 ThoughtWorks 社区启动首次国内软件质量的调查，受到大家的热烈关注和支持，并发布了调查报告，也得到大家的好评和肯定。2021 年的调查得到更多社区的支持，除了去年腾讯 WeTest 和 Thoughtworks 社区，还得到 Testin 云测试社区、MeterSphere 开源社区、AlTest 掌动国产化社区、PerfMa 质量技术社区、CSDN 社区的支持。

这次调查在去年 21 个题目的基础上增加了五个新问题，如：

- ✧ 测试团队/人员配置情况；
- ✧ 目前您团队已/正在开展下列哪些质量工程活动？
- ✧ 软件测试自动化水平，功能测试用例数的多大比重已经实现自动化了？
- ✧ 您所在的团队现在使用了下列哪个工具组合构建质量门禁？
- ✧ 明年贵公司希望在哪几个质量专项增加更多的自动化测试/工具的投入？
- ✧

这样，这次调查数据更全面、更有价值，正如参加本次调查的一位同学说：参与调查，其实就是对照自己的团队进行反思的过程。数据量也有明显提升，共收到 1356 份答卷，平均完成时间 10 分 26 秒。虽然只有那么 20 多个问题，但差不多是对你的灵魂拷问，从需求质量、设计质量、代码质量、测试质量到上线后的质量，贯穿整个生命周期。

作为“软件质量报道”，开展软件质量调查，是一种责任，希望能促进国内软件质量水平和企业的质量管理能力的提升。

二、参与调查的企业、团队和产品情况

1. 企业所处行业分布情况。参与调查的企业以互联网为主，今年比重更高，比去年高近 7%（今年是 47.4%，去年是 40%），说明互联网化趋势明显，但排序没变，排在“互联网”之后依旧是“信息和通信、金融和保险等行业，制造业、航空航天、教育”等，如图 1 所示。

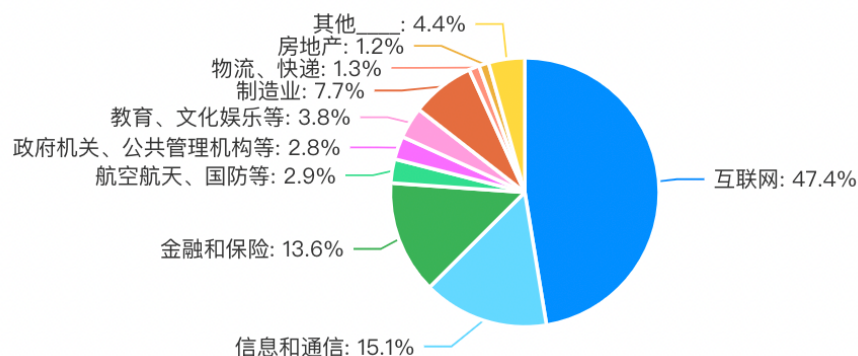


图1 企业所处行业分布情况

2. 企业规模分布情况。规模分布比较均匀，比去年更均匀。大型企业（超过 3000 人）和微小型企业（小于 50 人）基本接近，前者是 19.9%，后者是 18.1%（12.8%+8.3%），除了“1001~3000 人”这类规模之外，比例均在 15%~20.4%之内 如图 2 所示。

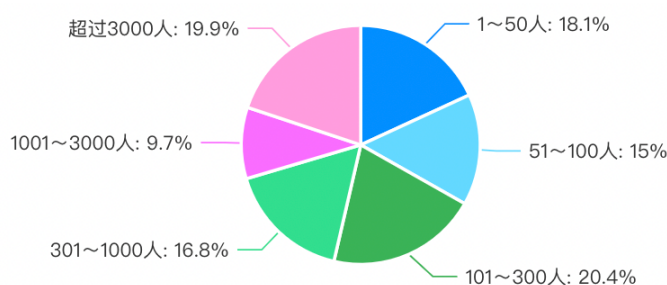


图2 企业规模分布情况

3. 团队大小分布情况。今年更加敏捷，团队规模更加小型化，今年 1-9 人的小团队比去年增加了 7%（24.7% vs 17.9%），超过 50 人的大团队，虽然比去年低了 4.4%（19.3% vs 23.7%），但还接近五分之一，如图 3 所示。为此做了交叉分析，如图 4 所示，敏捷中 BDD/FDD 和其它开发模式运行比较好，大规模占比很低，但 Scrum 模式中大规模占比依旧很高，说明 Scrum 模式中伪敏捷比较多。

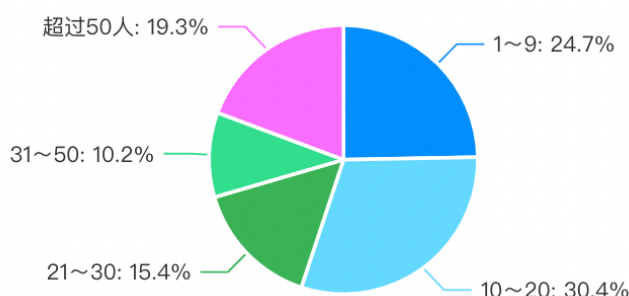


图3 团队大小分布情况

子问题	敏捷中的Scrum	敏捷中的BDD/FDD	其它敏捷开发模式（如XP、Lean等）	瀑布模型、V模型
1~9	86 (25.7%)	16 (23.2%)	7 (12.3%)	41 (22.4%)
10~20	96 (28.7%)	29 (42.0%)	16 (28.1%)	56 (30.6%)
21~30	50 (14.9%)	13 (18.8%)	15 (26.3%)	36 (19.7%)
31~50	29 (8.7%)	5 (7.2%)	10 (17.5%)	19 (10.4%)
超过50人	74 (22.1%)	6 (8.7%)	9 (15.8%)	31 (16.9%)

图 4 团队大小与团队开发模式交叉分析结果

4. 团队采用的开发模式情况。毋庸置疑，敏捷已成为主流的开发模式，各种敏捷开发模式占 34%，加上偏敏捷的自定义开发模式，合计超过 63%。而且与 2020 年相比，2021 年更敏捷了，虽然 Scrum 模式只增加了 1.5%，但整个敏捷开发模式（包括 BDD/FDD、其它）增加了 5.5%，还算比较显著，如图 5 所示。

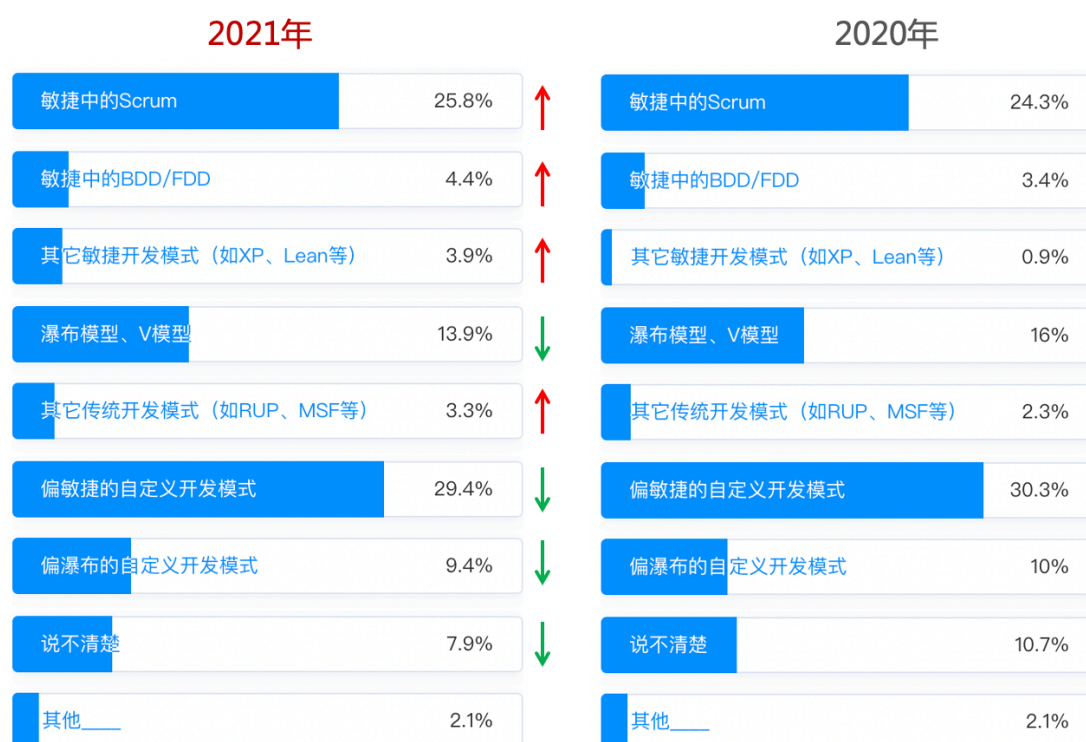


图 5 团队采用开发模式统计结果

4. 交付周期情况。上面统计结果是 63%的团队采用（偏）敏捷开发模式，而交付周期不超过 2 个月的团队占到 65%，如图 6 左边 2021 年数据所示，说明敏捷开发模式的交付周期大概处于这个范围。

既然 2021 年更敏捷了，那么 2021 年整个交付周期自然也缩短了，虽然按天交付的故事/团队基本稳定（数据非常接近，说明调查数据基本靠谱），但交付周期 3 个月内的增加了 5.3%（和敏捷开发模式增加 5.5% 非常接近，算是对数据的一次验证）。

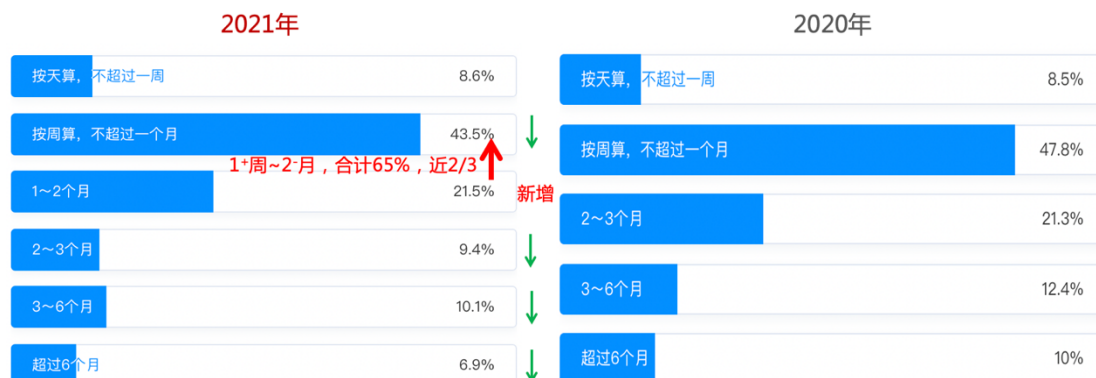


图 6 团队产品交付周期统计结果

5. 团队所开发的产品属于哪一类，是 2B 还是 2C 呢？ 2B 和 2C 产品交付方式差别很大，从今年和去年对比看，2B 提升了 3%，2C 和内部使用产品保持稳定，其差异只在 0.1~0.2%，这也说明调查数据的质量还是比较高的。2B 提升了 3%来自于去年的“其它”，今年的“其它”从去年的 7.7%下降到 4.3%。

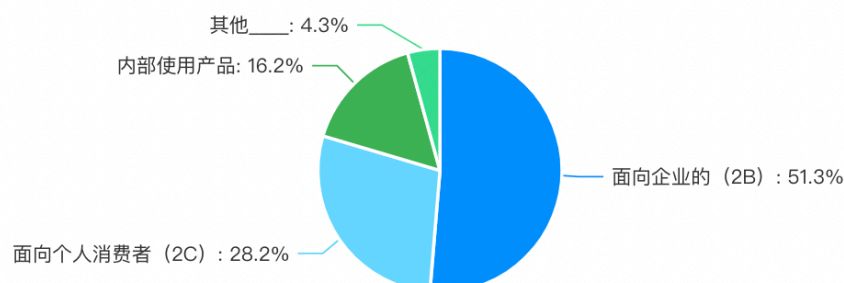


图 7 团队采用开发模式统计结果

6. 产品形态分布情况。2020 年是信创产业全面推广的起点，2021 年，信创产业有了更好的发展，这也表现在基础软件占的比重越来越大，从 2020 年的 9.4%增长到今年（2021 年）的 15.1%，相当于基础软件增长了 60%，值得点赞！

其次，工业互联网和其它网络应用越来越多，后端（服务器端）的开发比重增大。今年调查的结果显示，差不多是前端的两倍，也比去年多了 10%，二前端相比去年少了 13%，其中也包含去年“其他”的占比过高（17.5%），今年“其他”的占比只有 9.8%，许多同学觉得我既做前端，又做后端，这时会选“其他”。

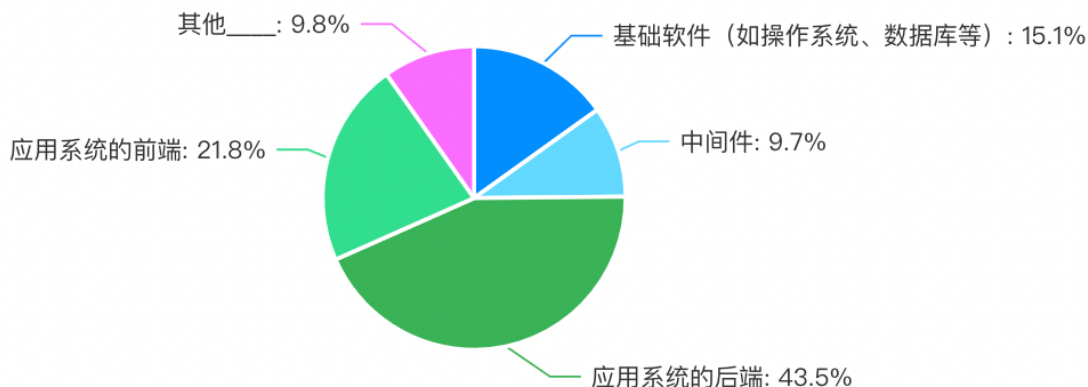


图 8 产品形态统计结果

三、软件质量整体状况

1. 从调查结果看，整体质量不乐观，和去年也基本持平。

17.5%的产品发生过线上严重崩溃事件，如图 9 所示，虽然比去年低了近 2%。通过交叉分析，如图 10 所示，互联网行业这类问题相对严重，因为根据图 1，互联网占行业比重是 47.4%，但严重崩溃事件这类问题，互联网行业占了 58.3%。通信、银行的服务相对稳定，问题所占的比例低于它们各自所占的行业比重。从开发模式看，如图 11 所示，Scrum 开发模式产生的严重崩溃事件相对较高，问题所占比重（27.3%）高于在开发模式所占的比重（25.8%），而偏敏捷的自定义开发模式的质量相对高一些，问题所占比重（27.7%）低于在开发模式所占的比重（29.4%）。瀑布开发模式也有类似的情况。如果忽略低于 2% 对差异，那么开发模式和线上崩溃问题，没有直接的关系。



图 9 软件产品质量整体状况对比

线上问题比较多，占了 **40.1%**，处在第 **2** 位置，而且比去年恶化了，高出 **3%**，这也进一步说明整体质量不佳。但**最严重的质量问题**还是“**需求变更频繁**”，高达 **51.8%**，虽然低于去年（**58.2%**），需求稳定性有所改善，如图 9 所示。“需求变更频繁”、“线上问题比较多”两类问题，各个行业都差不多，问题所占比例和行业所占比例非常接近。**线上问题、需求频繁变更**等问题所占比例和其开发模式所占比例也非常接近，也说明通过敏捷可以缩短发布周期，但改进质量不显著。

排在“需求变更频繁”、“线上问题比较多”之后是“漏测问题严重”（占比 1/5）、“代码质量明显下降”（占比 1/6）。Scrum 的“代码质量明显下降”要好一些（**21.5%**），低于开发模式所占比例（**25.8%**）的 **4.3%**。

子问题	线上服务发生过严重故障事件	发生过被攻击且不能正常服务事件	线上问题比较多	需求变更太频繁	架构设计推倒重来	代码质量明显下降	漏测问题严重（缺陷遗漏严重）	客户抱怨厉害	其他____
互联网	146 (55.3%)	73 (27.0%)	263 (38.6%)	334 (47.9%)	97 (19.5%)	104 (43.0%)	119 (44.6%)	57 (44.6%)	27 (40.3%)
信息和通信	20 (8.0%)	15 (10.5%)	65 (12.0%)	99 (14.2%)	27 (14.1%)	35 (14.5%)	47 (18.2%)	35 (19.0%)	12 (17.9%)
金融和保险	26 (10.4%)	16 (11.2%)	80 (14.8%)	88 (12.6%)	15 (7.8%)	32 (13.2%)	23 (8.9%)	18 (9.8%)	16 (23.9%)
航空航天、国防等	1 (0.4%)	4 (2.8%)	9 (1.7%)	23 (3.3%)	7 (3.6%)	11 (4.5%)	12 (4.7%)	3 (1.6%)	0 (0.0%)
政府机关、公共管理机构等	8 (3.2%)	6 (4.2%)	21 (3.9%)	18 (2.6%)	4 (2.1%)	8 (3.3%)	7 (2.7%)	8 (4.3%)	0 (0.0%)
教育、文化娱乐等	15 (6.0%)	8 (5.6%)	22 (4.1%)	29 (4.2%)	8 (4.2%)	12 (5.0%)	10 (3.9%)	6 (3.3%)	1 (1.5%)
制造业	12 (4.8%)	10 (7.0%)	39 (7.2%)	63 (9.0%)	21 (10.9%)	22 (9.1%)	22 (8.5%)	17 (9.2%)	3 (4.5%)
物流、快递	2 (0.8%)	4 (2.8%)	8 (1.5%)	9 (1.3%)	4 (2.1%)	4 (1.7%)	2 (0.8%)	1 (0.5%)	0 (0.0%)
房地产	8 (3.2%)	1 (0.7%)	9 (1.7%)	7 (1.0%)	2 (1.0%)	4 (1.7%)	3 (1.2%)	3 (1.6%)	2 (3.0%)
其他____	11 (4.4%)	6 (4.2%)	25 (4.6%)	28 (4.0%)	7 (3.6%)	10 (4.1%)	17 (6.6%)	11 (6.0%)	6 (9.0%)

图 10 不同行业的软件产品质量状况

子问题	线上服务发生过严重故障事件	发生过被攻击且不能正常服务事件	线上问题比较多	需求变更太频繁	架构设计推倒重来	代码质量明显下降	漏测问题严重（缺陷遗漏严重）	客户抱怨厉害	其他____
敏捷中的Scrum	68 (27.3%)	31 (21.7%)	129 (23.8%)	162 (23.2%)	46 (24.0%)	52 (21.5%)	64 (24.8%)	49 (26.6%)	20 (29.9%)
敏捷中的BDD/PDD	14 (5.6%)	12 (8.4%)	34 (6.3%)	26 (3.7%)	16 (8.3%)	15 (6.2%)	10 (3.9%)	6 (3.3%)	3 (4.5%)
其它敏捷开发模式（如XP、Lean等）	6 (2.4%)	13 (9.1%)	25 (4.6%)	22 (3.2%)	15 (7.8%)	15 (6.2%)	8 (3.1%)	4 (2.2%)	2 (3.0%)
瀑布模型、V模型	39 (15.7%)	27 (18.9%)	68 (12.6%)	97 (13.9%)	29 (15.1%)	39 (16.1%)	41 (15.9%)	36 (19.6%)	6 (9.0%)
其它传统开发模式（如RUP、MSF等）	8 (3.2%)	8 (5.6%)	21 (3.9%)	25 (3.6%)	7 (3.6%)	11 (4.5%)	10 (3.9%)	6 (3.3%)	0 (0.0%)
偏敏捷的自定义开发模式	99 (27.7%)	26 (18.2%)	150 (27.7%)	213 (30.5%)	47 (24.5%)	65 (26.9%)	78 (29.5%)	41 (22.3%)	23 (34.3%)
偏瀑布的自定义开发模式	15 (6.0%)	9 (6.3%)	40 (7.4%)	66 (9.5%)	15 (7.8%)	23 (9.5%)	23 (8.9%)	20 (10.9%)	2 (3.0%)
说不清楚	25 (10.0%)	13 (9.1%)	66 (12.2%)	72 (10.3%)	14 (7.3%)	20 (8.3%)	22 (8.5%)	18 (9.8%)	7 (10.4%)
其他____	5 (2.0%)	4 (2.8%)	8 (1.5%)	15 (2.1%)	3 (1.6%)	2 (0.8%)	4 (1.6%)	4 (2.2%)	4 (6.0%)

图 11 不同开发模式的软件产品质量状况

2.质量文化依旧比较落后

质量文化不是一朝一日可以改变的，总体看，这两年没有大的变化。质量不好，在质量文化上会有明显的体现，例如当产品发布时，如果质量和进度有冲突，只有 **28.5%**团队优先考虑质量，而绝大多数的团队则优先考虑进度（如图 12 所示），虽然相比去年有

所改善，提高了 2.7%。绩效考核中有明确的质量指标反而是下降了 3%，但制定了明确的质量方针、政策则提高了 5%，两者相互抵消了一部分。现在基础设施更强了，数据和可视化技术手段也多了，这两年数据驱动质量、研发效能度量、可观察性正逐渐成为热点，但没想到实施了“质量指标的可视化看板”（可以看作是质量度量）的依旧很低、不到 1/4，相比去年，也只增加 1%，这个其实可以提高得更快些。只有 40%的领导重视质量，这个数据很稳定，甚至还下降了 0.4%，说明改变领导更难。



图 12 质量文化对比

3. 质量管理组织不到位

对标准不重视、对质量不重视，在软件行业是一种普遍现象。软件测试虽然是质量保证的重要手段，但软件测试部门不是质量管理部门，而要做好质量管理，需要在公司层次建立自上而下的 SQA（软件质量保证）部门。从调查数据来看，公司里有 SQA 部门的比例还下降了 4%，质量管理组织上更加不到位，如图 13 所示。我们可以这样设想，这 4% 的部门拆了，但人员保留了，因为第二项增加了 3.5%，非常接近 4%。但是，如果加上有 SQA 角色的，不管是专职的还是兼职的，超过 75%，我们还是可以聊以安慰。



图 13 SQA 部门/角色设置对比

四、需求质量

好消息是需求质量有明显改善，对需求质量“满意”或“很满意”的团队增加了 10.3%,2020 年这两部分合起来还不到 1/4，今年却超过了 1/3，如图 14 所示。也许正是这样的调查和《软件质量报道》公众号不断呼吁大家重视需求质量而带来的效果，希望是这样，那也算是我们对行业的一大贡献。从保障需求质量实践看，如图 16 所示，超过 60% 的团队有明确的需求规范，而且有集体的会议评审，但为什么需求质量还不高呢？

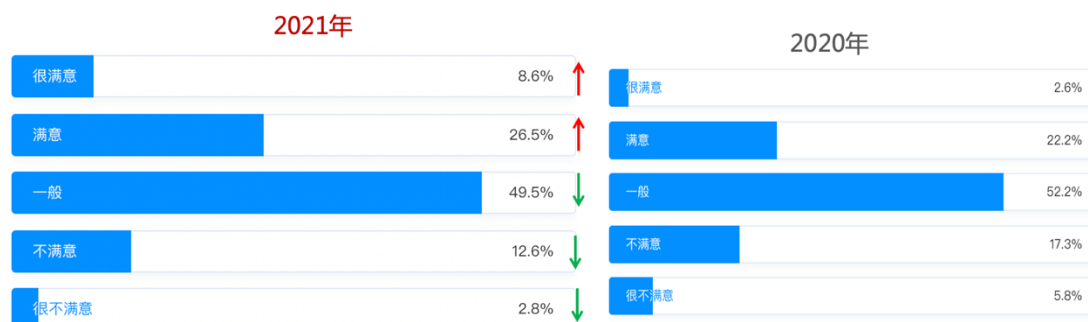


图 14 需求质量的对比

需求质量为什么不好呢？一定是存在比较多的问题。通过调查数据显示，首要的问题是“需求文档描述模糊、细化不够”，近半数团队都有这样的问题，如图 15 所示。其次是需求变更频繁、需求文档缺少典型的场景/用例、需求文档缺少验收标准（未实施 ATDD）、需求变更缺少有效控制等。超过 1/5 的团队，没有规范的需求文档模板，这部分团队还不够成熟、处在手工作坊式开发阶段？需求评审走过场也超过 1/5，估计这类公司的设计评审、代码评审会更流于形式，而对应严格的需求评审流程的团队也只有 43.4%，不到一半，剩下的 30+% 团队是什么状态？有流程但不严格？有会议集中评审的，也只占 64.5%，没达到 80%。

参与调查的同学还反映其它一些需求问题，如：根本没有系统性的质量管理、需求浮于表面、需求方自己都没捋明白就提的需求、老板说了算、领导不重视需求、需求零散而跟特性树关联少、需求澄清做得不够、流程不够规范、开发前缺少非功能性需求、没有需求计划会议、不完善的需求评审制度、缺乏学习文化、堆人堆工时等。

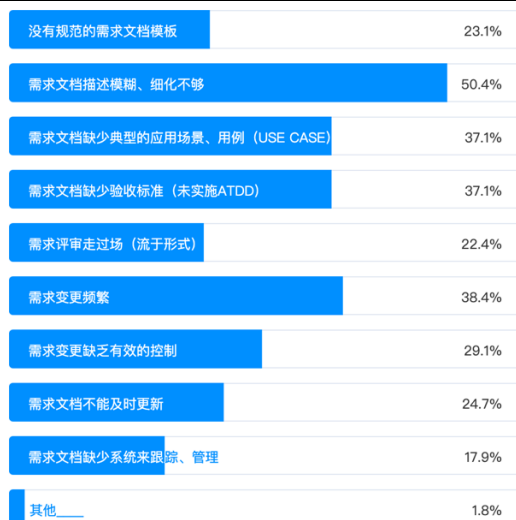


图 15 在需求上存在的主要问题

绝大多数的需求质量保障措施（如“明确的需求规范、有 SQA 的专人检查、严格的需求评审流程、需求变更控制、完善的需求工程体系”等），2021 年都比上一年有了一些提升（1.1%~2.7%），提升最多的是“严格的需求评审流程”，只有“在会议上集中评审”有下降，降低了 5.4%，可能现在需求评审更灵活，特别是对用户故事的评审，颗粒度小，也常常会采用俩俩评审的方式，无需过会。

参与调查的同学也提了一些其它保证需求质量的实践：

- ✧ 按项目交付需求验证；
- ✧ 业务方提前验证；
- ✧ 测试校验需求文档/测试保障；
- ✧ 和客户直接沟通/面对面的交流；
- ✧ 团队自驱；
- ✧ 领域代码测试和开发同时进行（代码级测试 50%以上），
- ✧ 第三方测评；
- ✧ 用例评审，产品验收；
- ✧ 各种方法的混合体。

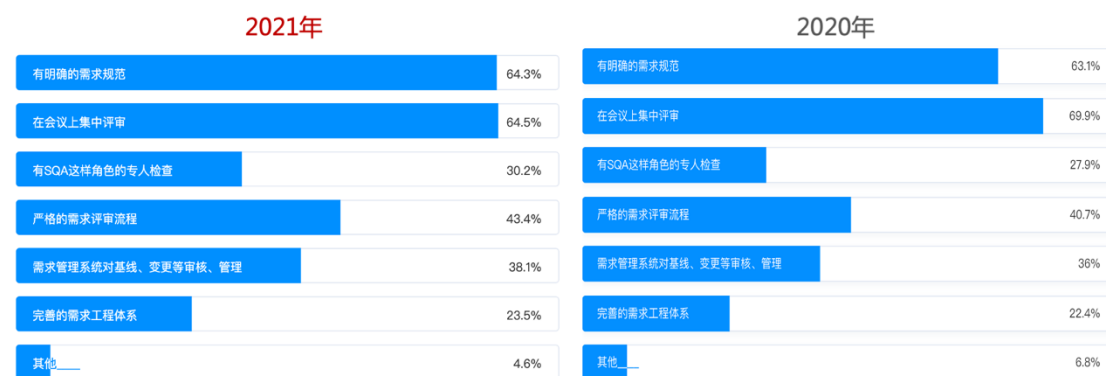


图 16 需求质量保障措施的对比

通过交叉分析，如图 17 所示，显示“完善的需求工程体系”、“严格的需求评审流程”、“SQA 的专人检查”带来的“满意+”分别达到 58.9%、50.1%和 45.4%，远远超过平均水平（35.1%，来自于图 14），说明这三项措施效果更好。即使有“在会议上集中评审”，如果流于形式，虽然代价高但也没有价值，这项措施带来的效果不是那么好，只有 36.1%，接近平均水平，“其他”措施（22.2%）则明显低于平均水平。

子问题	有明确的需求规范	在会议上集中评审	有SQA这样角色的专人检查	严格的需求评审流程	需求管理系统对基线、变更等审核、管理	完善的需求工程体系	其他
很满意	97 (11.1%)	73 (8.3%)	48 (11.7%)	67 (11.4%)	57 (11.0%)	45 (14.1%)	2 (3.2%)
满意	284 (32.6%)	243 (27.8%)	38 (33.7%)	228 (38.7%)	195 (35.9%)	143 (44.8%)	12 (19.0%)
一般	426 (48.9%)	432 (49.4%)	176 (43.5%)	251 (42.6%)	229 (44.4%)	114 (35.7%)	29 (46.0%)
不满意	55 (6.3%)	105 (12.0%)	44 (10.8%)	36 (6.1%)	37 (7.2%)	14 (4.4%)	13 (20.6%)
很不满意	10 (1.1%)	22 (2.5%)	1 (0.2%)	7 (1.2%)	8 (1.6%)	3 (0.9%)	7 (11.1%)

图 17 需求质量保障措施的效果

五、设计质量

和 2020 年相比，设计质量有所改善，“系统架构设计、系统整体设计、UI 设计”有提升；接口设计基本持平，满意度略微下降；只有功能设计退步明显，满意度下降了近 7%，不满意度增长了 1.7%，如图 18、19 所示。但从整体设计质量看不乐观，满意度只有 26.6%，而不满意度超过 30%。从设计的实际现状看，功能设计、接口设计相对好些，满意度最高，接近 40%，但也不到一半，不满意度接近 1/4。架构设计、UI 设计更需要大家的关注，未来加大力度来提升它们的质量。也有不少公司没有设计质量的保障措施，没有设计文档和设计评审，主要是开发负责人自己负责设计、把控设计的质量。

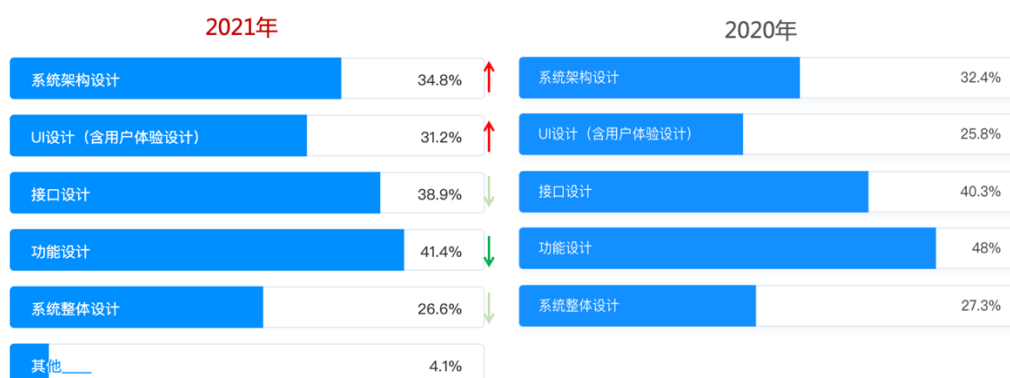


图 18 对各项设计质量满意的统计结果

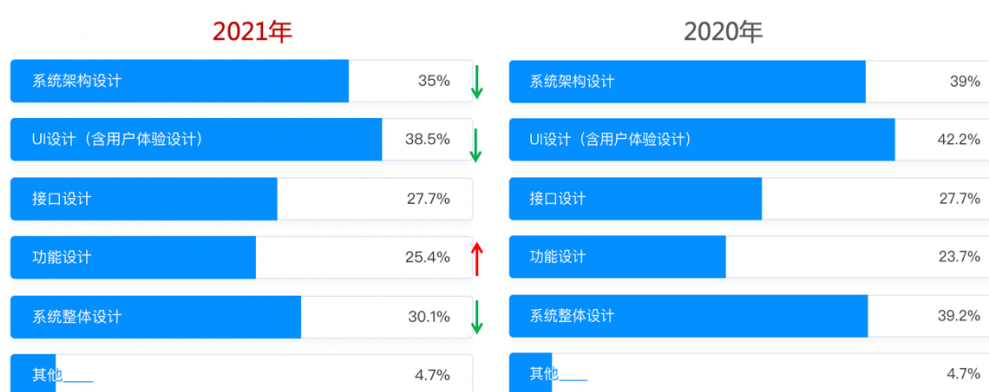


图 19 对各项设计质量不满意的统计结果

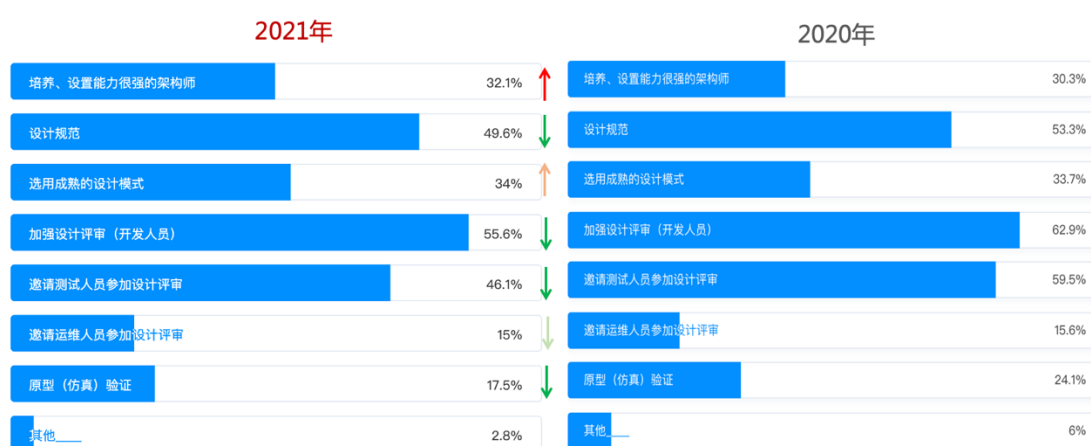


图 20 软件设计主要实践

在保证设计质量的优秀实践上，总体形态没有变，主要的五项实践还是“加强设计评审、制定设计规范、邀请测试人员参加设计评审、培养/设置能力强的架构师、选用成熟的

设计模式”。和 2020 年相比，2021 年只有“培养/设置能力强的架构师”有提升，而大部分实践都有所下降。除了这些实践，也有参与者提供了其它的实践：

- ✧ 规范设计人员的行为模式；
- ✧ 和客户沟通；
- ✧ 知识分享，通过教别人来促使自己学习；
- ✧ 强大的软件技术支撑等。

六、代码质量

我们最终交付出去的是代码，代码质量不容忽视，也确实许多公司比较关注的，但从调查结果看，接近 1/3 的人并不清楚自己公司的代码质量处在什么水平。从调查数据看，如图 21 所示，高质量代码（<1%）的公司比较稳定，接近 10%，1.0~8.0/KLOC 都有提升，不清楚自己公司的代码质量的同学在 2021 年下降了 7%，也许是在去年调查之后有些同学开始关注代码质量的度量了。



图 21 代码质量的对比

内建质量，不仅体现在定义好需求、做好设计和写好代码，也包括开发多做测试，特别是多做单元测试，这也是敏捷开发所提倡的。从去年统计数据看，单元测试没有得到大家的重视，对单元测试没有要求的团队占到大部分（60%），“单元测试不足”是软件质量上不去的重要原因之一，也是一种潜在的软件风险，甚至可以说是无数颗不定时的炸弹埋在代码中，毕竟依赖系统测试，很难进行彻底、充分的测试。但今年数据显示，单元测试有明显提升，对单元测试有要求的比例有显著改善，分支覆盖、行覆盖分别增加了 7%、6%，而对单元测试没要求的比例足足下降了 11.6%，低于 50%了。根据图 5 的统计

结果，敏捷开发模式提升了 5.5%，这会促进大家重视单元测试，提升对单测的要求，最终也会反映到单元测试调查数据上。



图 22 单元测试要求历史对比

从开发模式看（如图 23 所示），采用正统的、规范的开发模式，在单元测试上都有更好的表现，而自定义的开发模式（无论是偏敏捷还是偏瀑布）在单元测试上表现相对较差。而在敏捷开发模式中，采用 XP、Lean、BDD/FDD 开发模式的，表现更好。总体上看，敏捷开发模式的单元测试要稍好于传统的瀑布模型。

子问题	敏捷中的Scrum	敏捷中的BDD/FDD	其它敏捷开发模式 (SDX P, Lean等)	瀑布模型、V模型	其它传统开发模式 (SRU P, MSF等)	偏敏捷的自定义开发模式	偏瀑布的自定义开发模式	说不清楚	其他
有，MDC>90%	38 (11.3%)	11 (15.9%)	10 (17.5%)	22 (12.0%)	9 (16.4%)	18 (4.9%)	9 (7.8%)	15 (10.3%)	6 (18.8%)
有，分支覆盖率>90%	54 (16.1%)	3 (3.3%)	9 (33.3%)	46 (25.1%)	13 (23.6%)	57 (15.7%)	12 (10.4%)	24 (16.4%)	2 (8.3%)
有，行覆盖率>90%	43 (12.8%)	12 (17.4%)	13 (22.8%)	17 (9.3%)	11 (20.0%)	29 (8.0%)	9 (7.8%)	20 (13.7%)	1 (3.1%)
有，行覆盖率>80%	59 (17.6%)	12 (17.4%)	7 (12.3%)	16 (8.7%)	3 (5.5%)	48 (13.2%)	13 (11.3%)	11 (7.5%)	5 (15.6%)
没有	135 (38.8%)	11 (15.9%)	6 (10.5%)	60 (33.7%)	9 (34.5%)	202 (55.5%)	70 (60.8%)	70 (47.9%)	16 (50.0%)
其他	11 (3.3%)	0 (0.0%)	2 (3.5%)	2 (1.1%)	0 (0.0%)	10 (2.7%)	2 (1.7%)	6 (4.1%)	2 (8.3%)

图 23 单元测试要求在开发模式之间的对比

从行业看（如图 24 所示），性命攸关系统（如航空航天、军工等领域）的单元测试在 MC/DC、分支覆盖率上表现最好，合起来占比 64%（虽然距离 100%要求还有较大距离），但远高于其它行业，是互联网（31.3%）的两倍、信息和通信（23%）的 2.78 倍、金融和保险（21.2%）的 3 倍。如果从总体有要求看，房地产和制造业的要求最低，其次就是金融与保险行业，这或多或少让我们有所担心，制造业、金融与保险的不少软件也是使命攸关的软件，制造业的部分软件还涉及生产安全，这些软件都要求达到 100%分支覆盖。

子问题	互联网	信息和通信	金融和保险	航空航天、国防等	政府机关、公共管理机构等	教育、文化娱乐等	制造业	物流、快递	房地产	其他——
有，MDC>90%	72 (11.2%)	12 (5.9%)	17 (9.2%)	6 (15.4%)	4 (10.5%)	4 (7.8%)	13 (12.5%)	1 (5.9%)	2 (12.5%)	7 (11.9%)
有，分支覆盖率>90%	129 (20.1%)	35 (17.1%)	22 (12.0%)	19 (3.7%)	9 (23.7%)	13 (25.5%)	12 (11.5%)	4 (23.5%)	0 (0.0%)	7 (11.9%)
有，行覆盖率>90%	65 (10.1%)	32 (15.6%)	24 (13.0%)	3 (7.7%)	4 (10.5%)	5 (9.8%)	14 (13.5%)	2 (11.8%)	1 (6.3%)	5 (8.5%)
有，行覆盖率>80%	83 (12.9%)	31 (15.1%)	25 (13.6%)	4 (10.3%)	6 (15.6%)	5 (9.8%)	8 (7.7%)	2 (11.8%)	1 (6.3%)	9 (15.3%)
没有	278 (43.2%)	86 (42.9%)	91 (49.5%)	6 (15.4%)	14 (36.8%)	23 (45.1%)	55 (52.9%)	8 (47.1%)	11 (68.8%)	30 (50.8%)
其他——	16 (2.5%)	7 (3.4%)	5 (2.7%)	1 (2.6%)	1 (2.6%)	1 (2.0%)	2 (1.9%)	0 (0.0%)	1 (6.3%)	1 (1.7%)

图 24 单元测试要求之行业对比

代码质量保证措施的 2021 年调查结果是最让人激动的。从 2021 年统计的结果（如图 25 所示）看，依赖于常见的三大法宝：人工的代码评审、工具的代码分析和良好的代码规范。这三项占比都超过了 55%，相比 2020 年，都有极大的提升，特别是良好的代码规范是上一年的 3 倍，但从我们的预期看，希望能达到 100%。如果从倍数看，借助 AI 技术推荐代码，则是上一年的 23.5 倍（9.4%/0.4%），可以说是发生了翻天覆地的变化，2020 年只是极少数团队开始尝试，而 2021 年有 10% 的团队开始使用 AI 技术推荐代码，因为现在有一些相对成熟的工具可以使用了，如北京大学的 aiXCoder。



图 25 保证代码质量的主要实践对比

七、测试质量

测试质量和开发质量相辅相成，好的开发质量（特别是可测试性、Design for Test 等）有利于开展测试工作，测试质量好，测试发现缺陷更早、更快，能及时反馈给开发，甚至可以帮助开发预防缺陷，进一步提升开发的质量。

如果测试自身没有做好，有些缺陷或大量的缺陷不能及时发现，从而我们会做出错误的决定，让不该发布的软件发布出去，造成引起质量事故，所以测试工作自身的质量也需要一系列实践来保证。从整体质量调查看（如图9所示），“线上问题多”占比40.1%，有一半多原因来自测试的问题，即如“漏测问题比较严重”所显示那样（占比19.5%），说明大约1/5的企业测试工作没有做好。

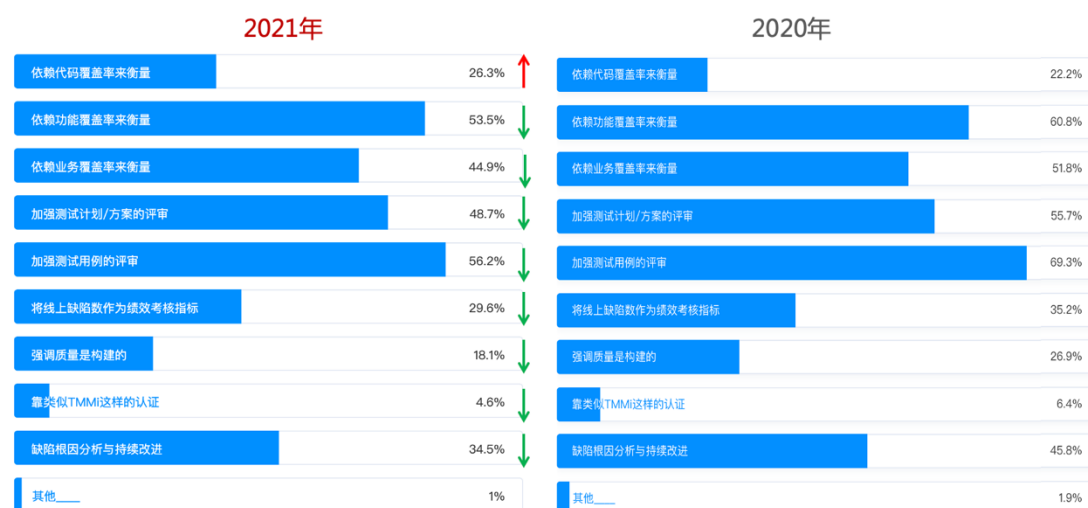


图 26 保证测试质量的主要实践对比

那么如何保证测试的质量呢？通过调查发现，如图26所示，保证测试质量的四大法宝是：

- 1) 加强测试用例的评审；
- 2) 依赖功能覆盖率来衡量；
- 3) 加强测试计划/方案的评审；
- 4) 依赖业务覆盖率来衡量。

这四项占比接近50%，但和去年相比，这四项内容的占比有明显降低（7%~13%），背后的原因不清楚，有可能因为敏捷更为流行，测试更灵活，如采用更多的探索式方式，甚至就基于用户故事（需求描述）来确定验收标准、应用场景，然后就开始进行测试，并没有写传统的测试用例。2021年，《软件质量报道》公众号还写了一篇文章[《是时候抛弃传统的软件测试用例了！》](#)。而且，依赖功能/业务覆盖率来衡量，许多时候都比较模糊，缺乏客观、准确的衡量，相对不靠谱。所以减少这方面的依赖比较正常。

而2020年依赖代码覆盖率来衡量的比例太低（22.2%），是一个比较大的问题，今年占比也仅仅上升了4.1%。虽然进步不大，但让我们看到一线希望，因为其它各项都是降低

的，只有这项上升了。“将线上缺陷数作为考核指标”降低了 6%左右,也许是好事，但“强调质量是构建的”从 2020 年的 26.9%降到现在的 18.1%，刚看到的一线希望又被扑灭了。

2021 年增加了“测试团队/人员配置情况”调查，“有独立的测试团队（项目中测试小组）”高达 67.3%，和前面“各种敏捷开发模式+偏敏捷的自定义开发模式”占比 63% 有比较大的冲突，因为在敏捷中提倡开发与测试融合。如果去掉“偏敏捷的自定义开发模式”，只考虑纯敏捷开发模式（占比 34%），那倒是比较一致，在这些“纯敏捷开发模式”中包含了下面三种情况：

- 1) 没有独立的测试团队，但有专职的测试人员
- 2) 没有专职的测试人员，但有兼职的、临时的测试人员
- 3) 没有测试人员，只有开发人员（虽然也会做测试工作）

这还只是我们设想，为了验证这个设想，我们可以看看交叉分析的结果，结果出乎意料，Scrum 开发模式中，“有独立的测试团队”高达 72.2%，还高于平均值，似乎给我们一种强烈的“不少团队都是伪敏捷”感觉。正如前面单元测试分析那样，采用 XP、Lean、BDD/FDD 开发模式的，表现要好得多，“有独立的测试团队”比例远低于平均值。自定义开发模式在“有独立的测试团队”最高，而且是偏敏捷的自定义开发模式最高，达到 77.5%，可以说，自定义开发模式的团队相对不成熟？



图 27 “测试团队/人员配置情况”调查结果

子问题	敏捷中的Scrum	敏捷中的BDD/FDD	其它敏捷开发模式（XP、P、Lean等）	瀑布模型、V模型	其它传统开发模式（RUP、P、MSF等）	偏敏捷的自定义开发模式	偏瀑布的自定义开发模式	说不清楚	其他
有独立的测试团队（项目中测试小组）	242 (72.2%)	35 (10.7%)	21 (42.1%)	95 (88.3%)	26 (27.3%)	282 (77.5%)	86 (74.8%)	71 (34.6%)	21 (66.8%)
没有独立的测试团队，但有专职的测试人员	56 (16.7%)	30 (29.0%)	13 (22.8%)	33 (30.0%)	12 (21.8%)	60 (16.5%)	17 (14.8%)	33 (22.8%)	2 (6.3%)
没有专职的测试人员，但有兼职的、临时的测试人员	26 (7.8%)	10 (14.5%)	17 (29.8%)	20 (18.9%)	11 (20.0%)	11 (3.0%)	10 (8.7%)	16 (11.0%)	4 (12.5%)
没有测试人员，只有开发人员（虽然也会做测试工作）	11 (3.3%)	4 (5.8%)	3 (5.3%)	5 (2.7%)	6 (10.9%)	11 (3.0%)	2 (1.7%)	25 (17.7%)	4 (12.5%)
其他	0 (0.0%)	0 (0.0%)	0 (0.0%)	0 (0.0%)	0 (0.0%)	0 (0.0%)	0 (0.0%)	1 (0.7%)	1 (3.1%)

图 28 不同的开发模式中测试团队/人员配置情况

八、质量工程与工具

2021 年调查增加了质量工程、自动化测试和质量门禁工具的调查，配合《数字化时代质量工程白皮书》未来的版本改进工作，同时也进一步了解大家比较关注的自动化测试等方面的情况。

首先看自动化测试水平分布情况，不符合正态分布，虽然中位数是最高的，即 30～50%、50～70%自动化水平占比分别为 20.5%、18.6%。90%以上占比 9%，超过 4 年前（2017 年）的数据，那时只有 4%。4 年前大部分（超过 50%水平的）已实现自动化测试只有 32%，现在（2021 年）达到 43.8%，增长 11.8%，有进步，但平均年增长不到 3%，说明自动化测试真不容易。

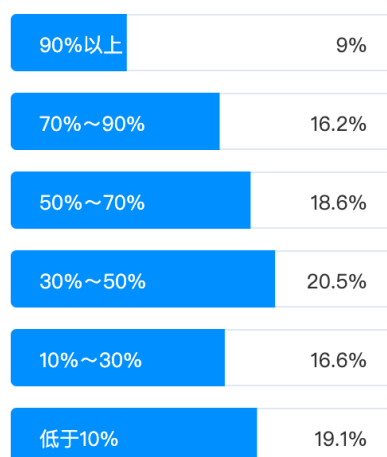


图 29 2021 年自动化测试水平分布情况

其次看质量工程技术/实践应用情况，比我们预想的要乐观，有 40.1%已经构建了系统的软件质量工程框架，虽然不到一半，未来空间还挺大。仅仅有软件质量工程框架还不够，最终还是要落地实施，其中业务驱动的需求工程最高（46.2%），但也没达到一半。其次就是高度自动化的 CI/CD 流水线（41.8%），CI/CD 算是比较普遍的工程实践，但要做到高度自动化也不容易，不过，认真去做，还是可以做得到的，期望值是能超过 60%，所以离期望值还有较大距离。其它几项内容（面向弹性的设计、性能工程、安全工程、用户体验工程等）占比都在 1/5 左右，也算是符合 80/20 原则，20%的团队是优秀的。混沌

工程是一项新实践，占比更低，不到 10%，相信未来发展会很快，因为这两年混沌工程很热。

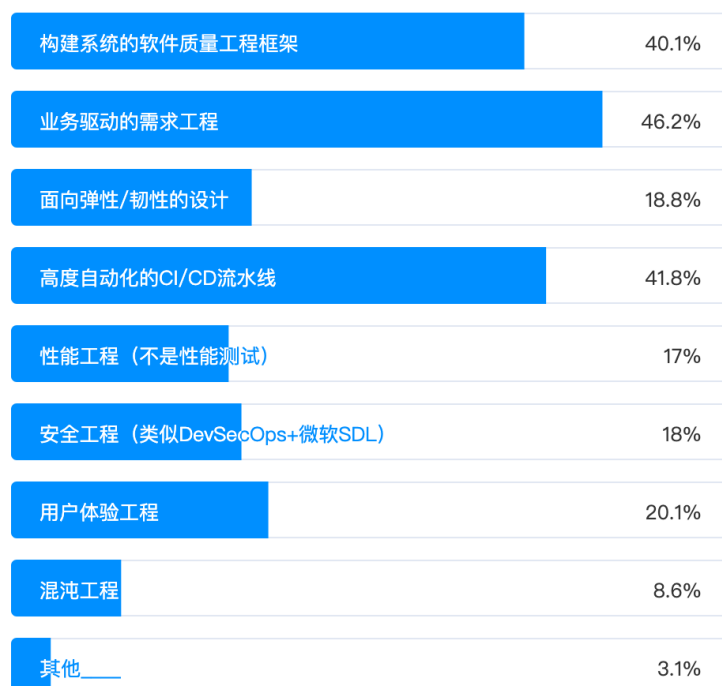


图 30 质量工程技术/实践应用情况

再看各个团队现在使用哪些工具组合构建质量门禁，基本顺序和题目设计基本一致，最流行的自然是“Jenkins + SonarQube”，占比超过一半，达到 58.7%，如图 31 所示。其次就是“SonarQube + MySQL”，而且是“SonarQube + Influx DB”两倍还多。大家更为关心的是在六个选项之外的“其它”，且占比也不低，达到 17.2%，排在第 3 位。其它组合主要有哪些呢？其中有很大一部分（超过 5%）团队还没有建立自动化的质量门禁，少数是靠手工；其次就是自研的内部工具，占到 3%。有一部分是混合方案——Jenkins 加上自研的测试工具/内部质量分析工具、Jenkins 加上开源/商业工具（如 Pytest、jira + xray、testlink + bugzilla）等，还有就是 GitLab 全家桶、SonarQube 和 Bamboo 组合等。其余的就比较乱，只有 1-2 家在用，非常小众，如 Zentao、Testbed、Klocwork、Circle CI、Cat 系列自动化平台、Azure Devops、云链、TeamCity、IScan、ALM、Coverity、Devcloud 等等。

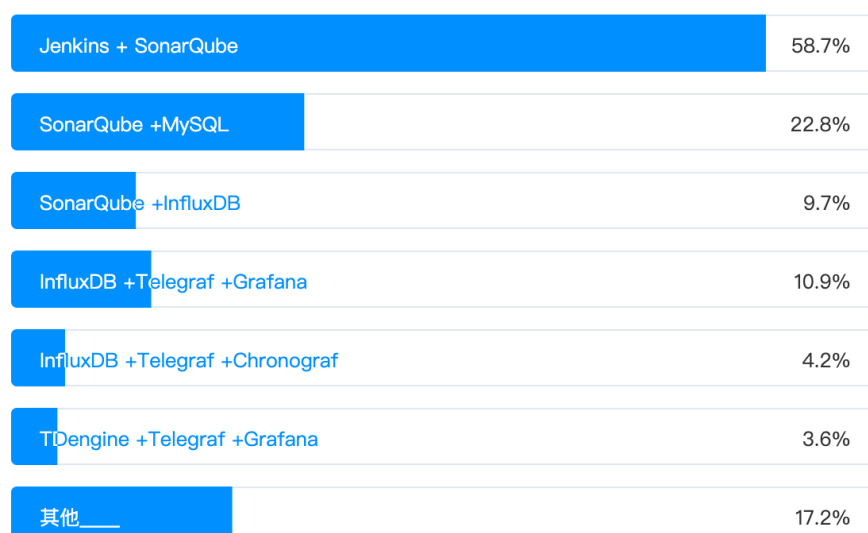


图 31 质量门禁工具应用情况

明年（2022 年）团队希望在哪几个质量专项增加更多的自动化测试/工具的投入？首推功能 UI/API 测试，正好超过一半（51%），其次就是后端性能、混沌工程。前端性能、安全性、用户体验、CI/CD 都非常接近，占比 1/4 左右。

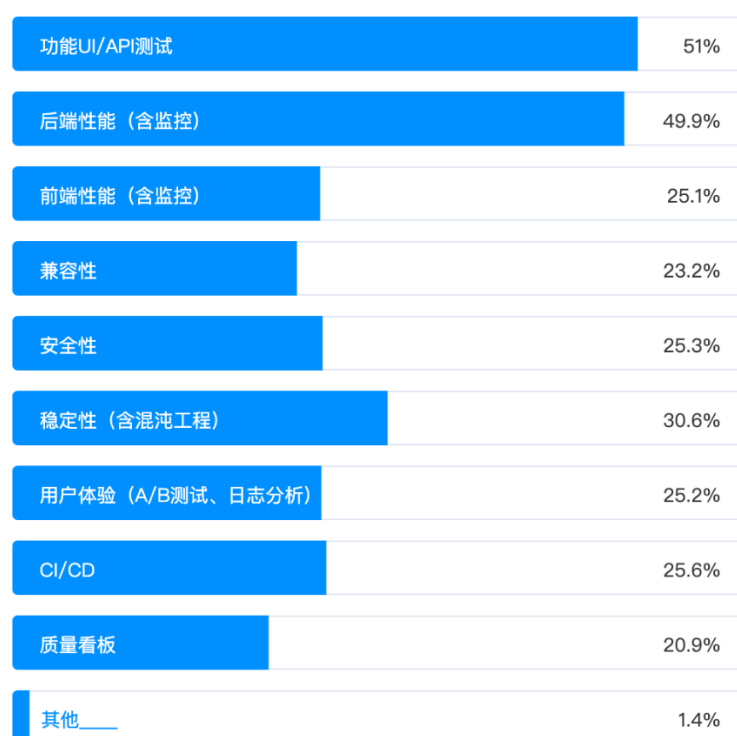


图 32 2022 年计划在质量/测试工具的投入情况

九、今明两年的软件质量管理工作

我们首先可以看看 2021 年软件质量管理的重点和 2020 年有什么差异或进步？只有两项（“团队人员能力建设上”、“重新设计/优化架构”）有小幅上升，其它五项都有下降，下降最大的是“提升测试能力、规范测试水平”，下降了 9.1%，这可能是参加调查的测试人员比重下降了，但调查缺少这项数据进行交叉分析。质量文化建设去年就比较低，今年就更低了，这说明领导还是不重视质量，否则这个数据不会这么低、更不会下降。在研发部门更崇尚技术、喜欢重复造轮子，但“研发基础设施建设”也下降了，虽然降得不多，但本来的基数就比较低。“代码重构优化”基数是倒数第 2，也比较低，但 2021 年还下降了。总之，“质量文化”、“研发基础设施建设”和“代码重构优化”等的下滑，让我们对国内软件质量产生比较大的担忧。

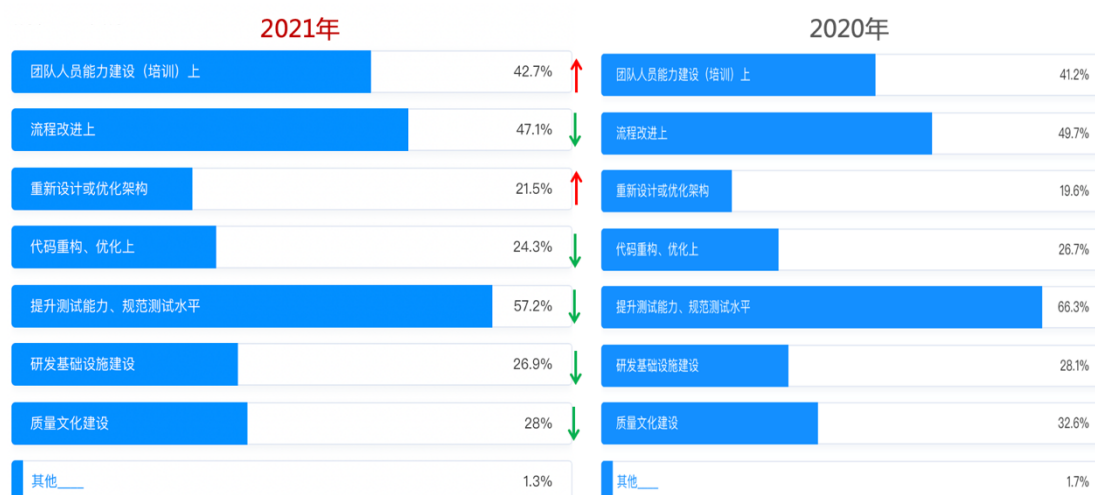


图 33 2020 年软件质量管理工作的重点

2020 年大家期望/计划做的事情和 2021 年实际做的事情，是否一致，即计划是否兑现，为此可以做对比分析，如图 34 所示。从排序看，基本一致：

- ✧ “提升测试能力、规范测试水平”都处在第一位，这意味着大部分团队进入了一个常见的误区——许多公司抓质量就是抓测试，但是质量是内建的，缺陷预防比发现缺陷更有价值，开发人员比测试人员对质量可以做更大的贡献。
- ✧ 2 和 3 的位置有错位，但在计划中差异很小（1.3%），而且计划中排在第二的“团队人员能力建设”还是得到更多关注，所以 2021 年有提升（见图 33），也是 2021 年仅仅有增长的两项之一；

✧ **4~7 的次序没有变化**，虽然质量文化排在第 4 位，不算低，但还是希望得到足够重视，因为文化先行，质量文化上去了，需求、设计和代码等质量就会上去。

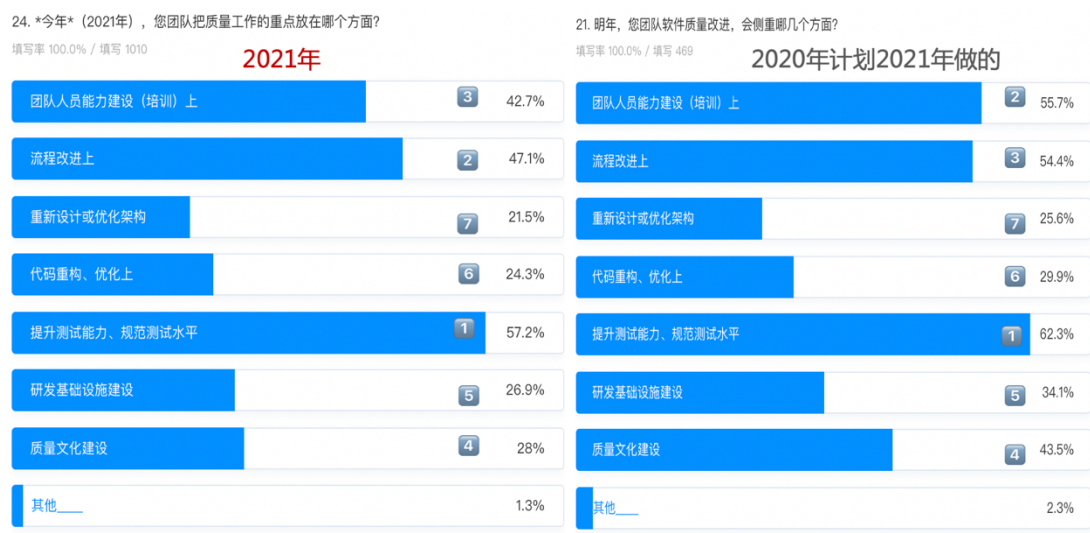


图 34 2020 年软件质量管理工作的重点

最后，我们再看看 2021 年在软件质量管理上的重点工作和 2020 年有什么不同？从调查结果看，如图 35 所示，有喜有悲，可以总结为以下几点：

- 1) 质量管理的误区依旧在那里，但 2021 年有比较大的进步，“提升测试能力、规范测试水平”从 2020 年的 62.3% 降到 2021 年的 52.1%，这是可喜的地方；
- 2) 另一可喜可贺的地方是更加重视架构设计与优化、代码重构与优化；
- 3) 最可悲之处是“质量文化建设”这项下降很大，达到 12.7%，文化是基业长青的基石，而质量则决定了企业能走多远；
- 4) 另一可悲之处是“团队人员能力建设”也下降了 7.6%，但孰不知“人是决定的因素”？需求是人挖掘、分析出来的，设计也是人做的，代码也是人写的，测试也是人干的..... 流程也是人制定的、还需要人去执行，基础设施也需要人去开发和维护等等，应该把“团队人员能力建设”放在第一位；幸好，“团队人员能力建设”比“流程改进”还高出 0.8，但希望 2022 年调查能高出 8%；

在“其它”处也有同学提供了一些好的关注点：

- ✧ 提升 DevOps 的自动化水平
- ✧ 数字化转型、提升质量数字化管理
- ✧ 加强低代码平台的建设

也有个别团队填“没有”或“不清楚，领导都不知道路在何方”，希望这个调查报告能给他们一些启发和指引。

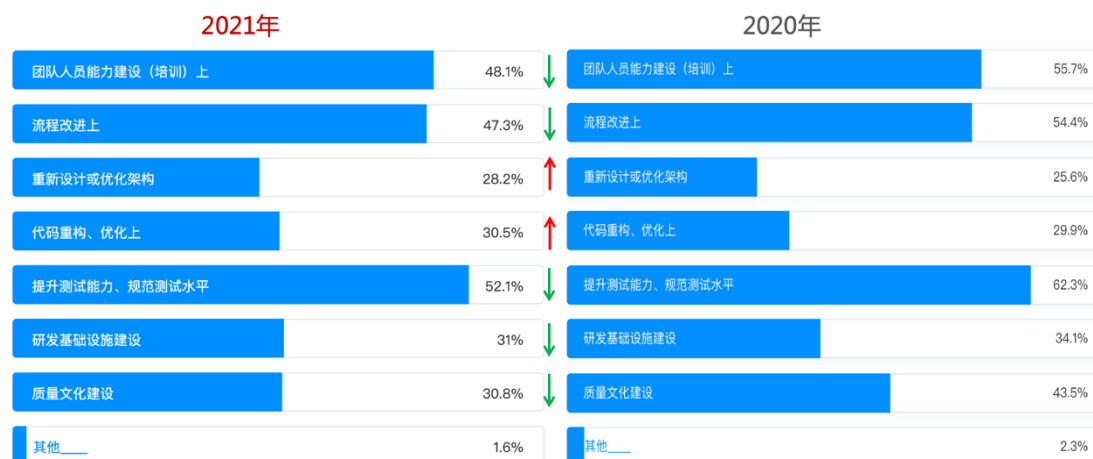


图 35 2021 年软件质量管理工作的重点

十、附录：呐喊、快乐分享与金句

2020 年的金句作为公众号文章发出去，引起了大家极大的关注和共鸣，阅读量超过 1 万，留言也不少，有些认知是共同拥有的，例如质量保证不等于测试，质量不只是测试人员的事，是整个团队的工作，需要从上到下的重视，内建质量、全程管控。**2021 年内容更丰富，参与调查的有些同学写了很多，让我们很感动。为此，把它分为三部分：呐喊、快乐分享和金句，希望对读者更有启发和帮助。**

1. 呐喊

不是每个团队都处在一个良好的环境中，有的团队所处环境恶劣，过了春节就想逃离。不少同学很痛苦，也有疑惑，就呐喊着，常常喊出“太难了！力不从心！”、“重要，却易被忽略，太难了！”、“比较麻木，领导不重视！”，自己迷茫、麻木在的 story 测试里面，周围环境给人的感觉是测试人员这个角色不重要。

1. 人员水平参差不齐，对质量的看法不一致。公司没有很好的由上而下的重视质量体系，因此推行质量的考核及工具的实施非常困难，心累。
2. 今年刚转岗做质量人员，接触了公司内部更多的不同的项目，遇到了项目中的各种问题。总体问题集中在需求管理和研发规范上。推行过程质量改进的路刚开始，更

多的是推行流程遇阻的心累。

3. IT 行业里，质量的事最难干，测试的时候发现需求要变更，为了赶项目进度赶鸭子上架，现场暴雷了又要挨批……质量很重要，但很多公司嘴上说着重视质量，构建这个体系，加强那个管理，结果要钱的时候就不给，然后一年到头总结：今年的质量工作开展没有到位。干好质量，肯定意味着投入成本加大，钱不批，权不给，说啥质量？习惯性的压迫，拿着底层薪资，干着高层的活，好好的质量工作，好好的团队氛围，硬生生把能干活的逼走了。
4. 研发流程、研发效能、研发质量每天都在说，但总是做出违背规则的事情，想要提高发布质量，奈何进度工期被一次次提前，需求也是很不明确，互相抱怨，问题不断，历史包袱较重，只能感慨来年再战。
5. 因为是内部项目，负责开发的程序员在用户体验上不太重视，导致用户体验感较低，使用率不断降低。其次，负责对接客户的成员无法解决技术问题，沟通不顺畅，使用感不佳。最后，经费有限导致人员无法扩充，也是软件质量无法提升的重要因素。
6. 从需求质量、开发质量、测试质量等每一环都是产品质量因子，尤其是越上游的环节影响越大，所以要提高质量，还需从根源抓起。产品开发间明确的相互制约制度是极其重要的。但往往有些公司产品开发非同一部门，又没有相互制约制度，所以客户常抱怨是系统缺陷，很多都是设计造成的。质量之路道阻且长，期望越来越好吧！
7. 传统行业中，测试本身发挥着重要的作用，可以说是保证质量非常重要的一关，随着测试左移、去测试化的理念影响，对测试人员能力要求越来越高，**但是对测试人员的重要认知却越来越低**，尤其是在传统行业受制于体制架构约束下，又受着互联网微服务架构下的快速迭代冲击，**处在夹缝中顽强地生长**。

2. 快乐分享

不少同学倾情分享，将自己的成功经验快乐地分享出来。

1. **软件质量很重要，需要很好的提前规划**，减少更改次数，架构设计需要考虑到后续的功能迭代，不然后续上线麻烦很多。如果发布前软件质量能够有保证，可以避免很多生产问题。
2. 以 QECon 为代表的行业内软件质量大会受到越来越多的人关注，是个好现象。测

试人员如何像软件质量人员转型，目前从路径上、能力上还有很多问题需要解决。如何评价或者体现软件质量工作的价值，始终是需要解决的核心问题。

3. 软件质量不是一个人的事，而是需要整个产品部一起配合完成，除了要建立完整的质量体系外，还需要建立团队的质量概念，并且要制定出完整的绩效考核制度，驱动开发，提高代码质量；另外，测试人员要努力提高技能，不放过任何质量问题。
4. 质量很依赖需求的质量、研发技术方案设计质量、研发代码 **review** 程度、研发单测覆盖率等，如果这些无法改变或重视度不够就很难从整体上提升团队产出的质量。
5. 软件质量应从用户需求的充分挖掘抓起，去伪存真，真正解决用户的痛点，保持较高的投入产出比，基于价值产出的高低及业务紧急程度，进行需求的排期及实现，并进行需求的闭环跟进，持续关注上线需求的应用情况，针对需求的全生命周期进行闭环跟进。
6. 软件系统的质量从来都是一个系统性的东西，质量建设贯穿于整个项目流程中的每个环节，所以想要真正的做好质量管理，需要建立质量标准，质量追求的目标，再从需求质量、软件流程、系统设计、评审机制等等各个层面着手持续提高软件质量。
7. 敏捷开发模式，需要一定的测开能力，提升自动化覆盖率是一个非常有效提升测试效率的方法，通过容量规划的日常压测可以发现服务的性能瓶颈，及时发现性能问题。
8. 质量是设计出来的，而不是测试出来的。为了拓宽开发环节，我们可以把测试融入到开发中去。我们已经建立了一个超高效的增量流程，只要有一个增量被证明缺陷太多，我们就可以回滚这些错误。我们不仅预防了很多产品级问题，还大大地减少了那些为确保消除“召回级别”缺陷而安排的测试人员的人数。
9. 质量是整个团队的事情，出问题不要再说测试场景覆盖不全、测试不充分。脱离具体场景分析没有测试尽早介入，没有质量文化构建的团队针对一切问题指责测试的都是病态的企业文化。都 21 世纪了，客户出问题还指责测试的项目负责人有啥拯救办法？
10. 好的产品质量，是前期设计开发出来的，而不是测试出来的；
 - 1) 测试只是提供最后的保障，提供最后的问题分析与降低风险，而不是线上一出问题，责任全部推测试，要具体问题具体分析；

- 2) 要加强可靠性韧性测试，基于线上用户量，扩大到 3 倍的用户量来评估业务场景，最终是否可以保障系统功能的正常使用；
 - 3) 混沌工程模式下，加强验证，基于目前的产品设计与架构，是否可以快速恢复、保证系统基本的业务能力呢？
11. 前端自动化建设是偏缓于整个流程的，但是其必不可少，可以先着力于基础功能的广覆盖，尤其是现网拨测的应用，可以最大程度减少现网事故的发生，提前预警；**其次，软件质量和进度是互相角力的**，这不是新话题，总是要找一个相对平衡。对于互联网类业务，总是要在不触碰用户底线的情况下，不断试错，快速上线，不断满足用户的预期。
12. 软件需求说明书很重要；软件的开发及评审，需求人员、开发人员、市场人员、测试人员等各方面要对需求有一致的理解，需求是开发的基础。一般来说，一个软件项目的需求分析要占总开发过程 50%以上的比重，需求分析比真正的代码开发更重要，建议您找专业的软件项目经理或者分析师来做前期的需求分析，或者直接把软件开发任务找专业的软件团队来做。必须重视文档及质量，重视知识文档化，重视开发文档质量。
13. 团队规模较小，缺乏资深的架构师/专家，很难从流程/制度上保障质量，只能尽量多增加测试。只能开发人员兼测试，自己开发的功能自己测，靠开发人员自觉编写测试代码保证质量。
14. 作为质量保证人员，辛辛苦苦总结分享建设质量文化，但是对质量提升效果不明显、出问题的是一些较复杂的异常场景，不知道如何控制这类问题，银行业务很拽只对接开发，不叫测试参与需求评审，也不按迭代约定窗口参与固定需求评审，测试获得信息来源于开发，很难理解全需求、发现问题。
15. 软件质量是整个研发团队共同义务和责任，而非在研发迭代末端让测试兜底，需要尽量前移，对给个角色各个阶段都要有质量意识，确保需求、技术架构方案、测试用例设计都做到精细化准确，再加上各类规范标准、各类工具平台支持（最关键的是自动度量）。当然，大前提是整个研发团队的大领导要有质量意识，不是空喊口号，实际执行不落地。
16. 金融行业是极度厌恶风险的行业，对软件质量要求很高，绝不会冒险上线。作为这个行业多年的测试人员，深深感受到这个行业测试人员的不容易。测试作为最后一道防线，压力很大，线上有问题第一时间找测试。因为这个行业的这种特性，决定了自动化程度不会特别高，很难说只依靠自动化测试完就上线。毕竟人对机器还是不太放心，人还是只相信自己。明年重点思考提升整个团队的测试能力和规范水平，

先把人员基础打牢。

17. 软件质量需要关注方方面面、从研发流程的改进、需求质量、研发质量再到自己可以控制的测试相关的质量，觉得有很多事情可做，但是又太多事情可做导致无从下手，目前还是以自下而上的问题驱动的方式在去改进，但总也无法形成体系；出现问题了，质量要求就高了，但是日常中质量还是会经常让步于其他，全公司都知道质量重要，但是关键时刻还是会首先抛弃质量；做为质量经理，有心无力，但是 22 年还得继续。
18. 软件质量是一个系统性的工程，只依靠测试或研发或 QA 等部分角色和部门是无法达成预期质量，需要通力合作，用良好的工具承接良好的质量意识，辅以合适的度量和考核，长期改进与坚持，才能越来越好。在合适的质量的基础上，提升效能才是 QA 的未来。
19. 软件质量与企业的管理体制有很强的关联关系，企业不注重，有再好的质量管理方案，也都能推翻，苦的还是底层研发人员，且行且珍惜。事前预防工作，例如，着重于缺陷预防而不是缺陷检查。（2）尽量在刚刚引入缺陷时即将其捕获，而不是让缺陷扩散到下一个阶段。（3）作用于过程而不是最终产品，因此它有可能会带来广泛的影响与巨大的收益。（4）贯穿于所有的活动之中，而不是只集中于一点。
20. 首先质量不是测出来的，质量是被设计出来的，再好的体重秤并不能让被检测对象实现瘦身或增肥。内建质量是一大出路，在软件生命周期里各环节的输出标准需要制定好，度量指标要能可视化并因时制宜，所有的度量不仅仅是为了找问题，是为了提高。另外，敏捷开发模式的基础条件是大规模的自动化，包括构建、部署、测试（接口、UI 和单元测试），不然敏捷只是空有其表，并不能健康前行，反而会在迭代持续的过程中让技术债务滚雪球，直到无法偿还。所以，部门的 lead 和架构师需要洞察和前瞻性的设计思想，追求一次性把事情做正确（快速抢占市场的行业除外）。
21. 缺陷是检验软件质量的准则之一，但特别针对数据仓库这类的缺陷，上线之后出现的问题往往防不胜防。在公司内部测试时发现问题不多，但放到现场真实环境中进行测试时又能发现大量由于原始数据原因造成的问题，但此时客户往往只会觉得这是产品的问题，而不是原始数据的问题。每次针对这类问题去定位排查时都会耗费大量的人力，可能等客户都快忘记了，才告诉客户上次发现的问题不是产品问题造成的。现目前所在团队的测试负责人，需求类的也作为问题提到研发那边，曾经出现过两方都不让步的情况，但下面的测试人员觉得这算是需求，而不是问题，不知道同行的小伙伴们有没有遇到过这种情况，遇到这种情况是真的挺心累的。

22. 仓促的开发可能会为团队节省片刻的时间，但是，如果有一些重大开发问题没有从一开始就考虑到，最终可能导致需要投入更多的时间。结果是浪费了大量团队资源来修复和重新设计代码，而不是将这些资源投入到更有用的事情上。软件团队人员内心里对整个始末一目了然，但面对着唠叨的客户、严格的销售团队，以及一些自我感觉编写了无缺陷的软件的开发人员，软件团队确实很难将 QA 撇在一边而只顾着完成代码。
23. 传统开发方式不能够满足现实开发的需要，对传统方式进行总结，对成功失败的开发案例进行研究后，得到的一种不同于传统方式的开发流程，主要的特点是迭代式进行，这种方式把一个软件开发过程分成了若干个小的迭代过程，每一迭代完成一部分功能，每一次迭代过程的工作内容按照功能的重要程度不同而排列，首先完成重要的，同时也是风险比较大的功能，而后是次重要的，依此类推，同时在每次迭代中，都要进行分析、设计、开发、测试，因为分成了一个一个小过程，一步步的逼近目标，所以，可以使得产品的用户能够逐渐得明白自己的真实需求从而能够提出真正的需求，而开发团队也可以根据更正后的需求进行下一次迭代的设计。
24. 所处公司对质量管理不重视，开发测试工作流程都不规范，测试和开发比例 1: 7。作为测试人员，一年下来测试出现 bug 太多，于是我整理了关于团队进度、质量、效率等方面的问题向上反馈，但是感觉领导也无力改变现状，反倒让自己增加了很多工作。身边一味被动做事情的同事不少，可是我不想，所以很想知道，在权力不够的情况下，对于公司现状是接受还是去挑战，或者说在一个不重视质量的公司，如何才能把自己的测试工作干得出色？这应该也是很多测试人员面临的问题吧。
25. 我是一名 SQA（非测试人员，纯质量工程师），在今年 3 月份，从通信行业转行到了金融行业原本以为金融行业会流程规范少，没想到银行处于人行监管犯愁，流程规范、合规规范非常多，而团队又走的“敏捷”开发模式，即快速发布，两周或一个月一个发布版本，然而每个周期走的流程却是瀑布 V 模型，需求分析-设计-开发-ST-回归测试，每个版本需要应付的流程很长，但银行内的工具却不完善，持续集成、CI/CD 都未建立起来，每个迭代，开发、测试都在疲于应付各个流程节点交付的文档，以及拖着历史包袱负重前行，开发、测试时常比 996 还过之，团队辛苦干活，然而业务却还一直不满意，认为 IT 效率太低，一个业务需求从提出到上线太慢了，IT 与业务矛盾重重、IT 内部工具不完善、效率低下等等，都是未来要去解决的问题。
26. 软件质量不是测试出来的，需要全团队各个环节时时刻刻重视及关注：
 - 1) 【规划】需要提前做好项目工作规划，以免到了末期由于时间周期问题忽视压缩测试的工作量。

- 2) 【需求】避免一句话需求，清晰的需求定义和描述才能让开发设计测试运营的理解达成一致；
- 3) 【代码】加强对研发质量中单元测试管控力度和手段，比如代码规范、代码评审等等；
- 4) 【测试】编制以客户实际使用场景的测试用例，增加反向异常测试用例数量。提高测试准入门槛，让更多的质量问题在迭代的前期发现；
- 5) 【缺陷】研发缺陷修复做到尽量不返工，不引起新的问题；
- 6) 【管理】保持团队人员的持续稳定、所有团队人员基于业务熟悉度开展相关工作，以达到事半功倍。

27. 刚做测试的时候会觉得作用测试设计方法，设计各种完善方案用例并充分执行，不放过一点边边角角，测试过程完整有效，就能保证起码的质量。说白了，认为质量保证主要依靠测试环节的努力和水平。这种观念持续了有好几年，到现在，这种观念我认为存在于大部分的非测试人员心里，甚至快成为主流的归属质量问题责任环节的标准。从前面说的评价质量好坏可以看到，软件质量从需求开始，就已经开始被定义了。抓住了用户需求，离高质量就近了一大步，需求跑偏，达到最终高质量，非常困难。不仅仅是需求，对于用户需要的把控，是贯穿于项目全程的，需求把握了，设计开发时不抓重点，一样白搭。要保证良好的质量，由始至终，整个 team 必须很清楚在整个系统中，重中之重是什么，其次是什么。这点对于进度总是那么紧的项目来说尤其重要。这个说下去又扯到项目管理了。一个清醒的项目经理，一个清醒的团队，才能知道最终我们究竟追求什么。其实，辅助的，也依赖公司的流程管理水平、指导策略等等。总之，质量保证是个全程的事情，任何一个环节的缺失或者薄弱，高质量的实现可能性就大大降低了。我始终觉得，把大部分精力投入在用户最关心的那少部分功能中，在有限成本下才能获取最大的产出和效益。

3. 金句

2020 年是 100 条，2021 年样本量翻了 3 倍，所以整理出来“关于软件质量或软件质量管理”的金句也是 2020 年的三倍。

1. 软件质量重于天，特别是涉及到企业产品质量，涉及到民生方面。
2. 软件质量决定团队发展、决定企业的口碑与存亡
3. 质量是生命线，软件质量决定一个公司的生命周期。
4. 把质量意识根植在每个工程师的头脑中是一个很艰难的过程，但是又极其重要。
5. 软件质量非常重要，需要有责任心的专职人员看护，同时开发人员的配合也很重要。
6. 质量意识是基础，质量文化是保证，质量保证是措施，质量活动是必须。
7. 软件质量不是口头说说，要实际做起来！

8. 质量是一种文化。
9. 质量体系认证流于形式，真正落实需要大家认同。
10. 质量影响整理效益，软件质量要高，用户体验要好，只有不断迭代才能变得更好。
11. 需要配等同的测试资源，不能忙于赶进度而牺牲质量。
12. 质量是全员参与的结果，软件质量需要从需求质量开始控制，由原型质量、UI 质量、架构质量、接口质量、代码质量等构成，质量是组织中每个角色都需要关心的，不仅仅是测试工作师和运维工程师的事情。
13. 做好需求调研，提前规划好开发设计，多注重用户体验。
14. 业务目标与专业目标需要做好规划。
15. 提出明确的需求，通过前期 demo 和客户确认细节。
16. 产品前后端一样重要，只注重后端，但展示给客户的前端丑爆了，会极大地影响后端的绩效；
17. 仍需投入大量人力，并且招募更多新鲜血液，任重而道远。
18. 一直有漏洞产生，一直在补漏洞；工作费时费力，流程还需优化。
19. 需求频繁更改真的很要命，人少技术旧真的不好做
20. 嵌入式软件测试如何做好测试分层，各层级之间如何避免漏测、重复测试，头疼。
21. 代码质量是一切的基础，可读性高的代码问题自然少。
22. 文档要写好，代码要注释，可以省后来人不少头发！
23. 少点 bug 我们还是朋友。
24. 做完了测试，我觉得我好菜。
25. 使用场景复杂，测试用例覆盖范围和实测要增强。
26. 国内对软件质量的概念定义都还没有统一的共识，多数还是在 996 赶进度，这事要往长远看...
27. 之前看过一本书《第一次把事情作对》，我觉得作为开发真的是可以做到：第一次把事情作对、不写出代码的 bug。
28. 一次性，好好吃苦，之后就不用再花力气维护了。
29. 别太急促浮躁，沉下心来做好产品，把项目管理落到实处。
30. 软件质量把控是软件工作的重要环节，运用好了可以减少软件迭代，提高软件效率。
31. 在传统行业，软件质量体系很难得到所有人支持，推动很难
32. 一个好的团队才是最重要的，提高人员素质很关键。
33. 人员投入不够，质量难以保障。
34. 需要投入更大的人力物力，与公司的政策的支持
35. 辛苦需要对应的回报，质量差、加班多的人不应该得到奖赏，质量好、不加班的更不应该受到批评。
36. 还是得认真去抓每个细节，然后碰到问题是追根究底的态度，不在意的小问题，有可能闯大祸。

37. 各个部门科学协作，虚心接受群众批评，软件质量才能很好。
38. 赶进度，质量不能保证，严重倒退。
39. 中层能力很重要，不然将帅无能、累死三军。
40. 研发人员缺失，功能堆不完，质量跟无法保证。
41. 思想决定行动，态度决定一切，质量决定未来。
42. 在交付工期和交付冲突时，通常会简化需求，降低质量标准，后期再进行找补，但通常就遗忘了。
43. 公司的方针是做新功能比维护旧系统更重要，也许很多公司都是这样，想用增长解决我们现在面对的问题。但是问题总有一天都是需要解决的！
44. 质量保障是一项系统工程，需要方方面面的配合
45. 软件质量与功能开发一样重要。一般在做项目管理的时候，是把进度和功能完成度看的比较重要，似乎忽略了质量和安全管理，这方面应该加强。
46. 软件质量应该先从项目纬度抓起，制定规范，控制业务节奏，控制迭代周期，然后再谈研发质量建设。
47. 团队共同关注质量才能改进产品质量。
48. 软件质量不透明，在设计和代码上花的很多时间精力没法被领导看到。
49. 不管是开发还是测试，要站在用户的角度理解需求、开发需求、测试需求。
50. 从事质量相关工作，很难客观度量，很难出成果，很难被理解！任重道远，继续努力！
51. 不求什么高大上，就希望能别出么蛾子。
52. 软件质量的保障途径有很多，但是最终的保障效果是木桶效应，短板决定了软件产品的最终质量，我们应该查漏补缺，解决关键影响因素。
53. 需求必须明确，不然质量不好把握。需求清晰，开发稳定。
54. 一个各方确定好的需求，能够节省很多开发时间。
55. 良好的编程习惯减少失误
56. 开发人员写代码时多想想再写，通常你这一下笔，就决定了软件质量，且后面不会再改。
57. 国内软件质量评审需要标准化、有极大的提升空间；
58. 如果想考虑长远，那就不要流于形式主义，人手不够，千万不要让质量来凑！
59. 设计先行
60. 软件质量是项目交付是否顺利的保障，但也要和实际成本相结合。
61. 能够将功能稳定性做好，减少在版本迭代时引入 bug。
62. 软件质量好点吧，求求了。
63. 需求总是变化，开发赶不上变化，真心希望需求不要频繁变更！
64. 软件质量不是简单开发或者测试的责任，是整个团队共同的使命，更是公司工程师文化的最好体现。
65. 客户是检验质量的唯一标准。

66. 软件质量很重要，凌驾于整个项目之上，所以必须全程保障质量优先
67. 提升自身开发水平，加强代码走查。
68. 软件质量很重要，如果质量不过关迎来的就是无穷无尽的修改，对于开发人员来说真是太难了。
69. 希望 Bug 不要太多，简单、实用才是永远的神！
70. 希望需求文档能够规范，尽量少改需求。
71. 希望下辈子不要当码农。
72. 产品经理太重要，需求一定要清晰。
73. 软件质量的好坏直接关系到用户的体验感，前路漫长，继续努力。
74. 质量要结合项目的目标逐步细化。
75. 质量很难做的完美，但需要尽力去做。
76. 工作重点尽可能的前移，设立多道预警机制，增加缓冲区，避免问题集中爆发。
77. 质量至关重要，要做到标准先行，规范的流程和标准是重中之重。
78. 产品不要盲目变更需求，版本更规范一点
79. 多站在用户角度思考问题，严把质量关
80. 软件质量不仅仅是测试或测试开发的责任。开发人员也要深入骨髓地知晓 UT、自测的重要性并付诸行动
81. 每提前发现一次质量问题，就大大降低了后续的试验成本。确保软件质量，不只是为了客户，更多的也是为了公司自己！不只是为了公司名誉，更是降低公司试验等相关成本的良好举措。
82. 人的认知是有限的，即使在我们认知范围内做到最好，也没有完美。
83. 环境很重要，团队文化决定了质量控制是否。
84. 太苦太累，加班熬夜是常态。
85. 辛辛苦苦大半年，一朝回到解放前。
86. 软件质量是大家共同建设的结果，测试人员只能够维持质量下限，希望大家一起重视质量。
87. 软件质量是对产品的负责、对用户的负责，在做项目过程中，不断学习保证质量
88. 质效合一，追进度不顾质量，只会让进度越来越慢、加班赶工。
89. 质量工作不易，希望大牛们不断引领行业健康发展。
90. 好的制度流程很多，但是适合更重要；因地制宜、量体裁衣，从企业文化、价值观改变团队效能。
91. 跨部门协调太难，测试人员没有话语权
92. 团队重视业务组织，轻视研发组织，一切都是靠偶然和应变运作。大家都累。
93. 场景模糊，后续开发问题牵涉较多，质量较难把控，同时，涉及到多产品连同协作，排查问题故障的难度增加。容易出现扯皮的情况。
94. 啥时候质量能在上游保证？
95. 一年每天都是唯快不破，质量不提升问题不断，运维时间越来越多占用开发时间。

96. 除了规范契约还要有责任心，团队目标与个人目标的一致性是个问题，正确的绩效联动尤其重要。
97. 用户体验是关键，后期维护也是一样。
98. 软件质量监控很重要，发现一例，解决一例，漏测也只能解决一部分的质量问题。
99. 软件质量需要每一位参与软件开发的同事从自身做起，建立质量意识。在交付时间与质量发生冲突的时候，不能一味地妥协质量。
100. 质量是全员的，也是全生命周期的。只有就质量达成一致目标，才能更好的建立起质量体系、质量文化、质量技术。而现在所在地方质量仿佛成为了一座孤岛，可有可无。
101. 软件质量其实是一个产品很重要的问题，目前看很多小微企业并不重视。
102. 企业的文化乃至部门的文化很重要，加强质量文化建设很重要。
103. 产品是主力贡献者，做的好坏基本上取决于产品设计的好坏；存在与技术上的冲突时，软件的质量就会发送变形，特别是需求多、时间少、改动多，进一步导致软件的质量下滑严重。
104. 软件质量需要团队每个环节的把控
105. 软件质量控制，说大其实不大，但是又不能没有，处于一个很尴尬部门。很多小公司开始不想要 QA，觉得成本高，这就需要提高测试在互联网岗位上的地位。
106. 测试和设计太重要了！
107. 很多时候想要保障软件质量，就要对一个功能点涉及到的东西都要考虑到，否则很难保障会不会出现偶然故障
108. 主要是需求一直在变动，程序人员技术过硬不用担心什么，临时变更需求真的不太好。
109. 质量是构建的，需大家齐心协力，发挥主动能动性，以及加上流程规范和相关的工具协助。
110. 好的产品会回馈产品研发！
111. 无用的需求太多，迭代太快！
112. 希望能在代码质量检测上做进一步提升。
113. 一个好领导是成功的一半，但是管理层普遍缺乏对质量的重视。
114. 任重而道远，尤其是新团队
115. 测试质量需要整个项目组一起优化，但从测试角度规范不够。
116. 日常作为各种软件的使用者，也常常抱着测试心态，发现各种 bug，真心希望所有软件质量都能够提升，让消费者用着舒心
117. 质量是内建的，需要全员有质量意识。经常复盘线上问题，从点到面优化和改进。
118. 让用户有感知的质量提升。
119. 缺少流程规范的质量建设是不会成功的！
120. 实际工作中蛮多不规范，也没有方法论的支持，领导不会轻易改变，学习空间有限，一声长叹！

- 121.希望质量工作能标准化、规范化，质量人员的职业发展更科学和可量化。
- 122.需求描述不明确是硬伤。
- 123.软件质量的根本在于设计，设计的目标之一应该是构造易于实现的架构，避免开发中的错误过多产生，如何做到，还要持续研究。
- 124.需要加强学习、成系统的学习，并转化成实际的生产力
- 125.很多好的实践，因为人的知识没有应用到产品中来。
- 126.我待软件如初恋，软件虐我千百遍。
- 127.好像做了很多，但也好像没啥收获。
- 128.质量不能只靠测试团队保证，团队中应该人人皆测试。
- 129.希望自己的测试能力越来越强，希望产品经理在需求上游想的更完善，不要频繁变更需求，也希望自己的团队成长快一些、业务更熟练一些。
- 130.理想很丰满，现实很骨干。测试从业者需要学习的东西越来越多，需要提高代码技能、TestOps、DevOps 和软实力等等。
- 131.提升自身测试技能，深入了解被测系统业务，提升产品质量。
- 132.软件质量离质量金字塔还很远。
- 133.质量和效率，在特定的时间点是矛盾的，长期看是可以螺旋式提升的。
- 134.软件质量重于泰山。
- 135.软件质量真的很重要，没有质量的软件会导致用户流失。
- 136.软件质量问题，单元测试都不写对团队也太难了，技术负责人要承担极大责任。
- 137.质量内建除了工具，文化和技能也是需要且重要的。
- 138.软件质量不仅仅是代码质量，更需要测试人员的把控和推动。
- 139.专业人做专业事，培养软件质量专业人员。
- 140.软件质量就是软件生命，质量上不能完成交付，软件就是失败的。
- 141.我们是产品经理制，需求质量太差，业务需求梳理基本只管主流程，需求描述就那么一句话。
- 142.质量是为产品服务的，产品本身就没有价值，会导致质量工作做了也同样没有价值。
- 143.质量的构建效果远大于测试的防守
- 144.质量是由测试人员牵头，需要全员配合，为最终客户负责的事。
- 145.质量内建至关重要，仅靠测试是无法保证的
- 146.质量是团队共同建设的，大家都同意，但让每个人身体力行的落到实处很困难！
- 147.人工把控远不如程序的质量门禁把控。
- 148.软件质量更多需要从产品经理、需求那边就开始抓好，会减少开发过程中的很多风险。
- 149.质量计划是指导，质量保证提供信心，质量监控使各方满意
- 150.软件质量不是团队中哪一个岗位的事情，是团队共同的责任，少一点坏苹果理论，多一点责任共担。
- 151.质效合一有一个比较好的结果，但任重而道远。

- 152.软件质量无小事，马虎大意不得。
- 153.人是关键，有限的人怎么在组织中成长更关键，所有的活动最终都应该反应到人能力提升上。
- 154.软件质量是一个至上而下的工程，埋头在业务中也需要抬头看看周边。
- 155.软件质量是一个做的好但别人感受不明显，做的差却让测试背锅的事情，希望质量文化建设上能够出行业先进案例，让老板们都能看到质量的重要性。
- 156.质量问题往往被无限扩大，质量人员要求对质量负责的同时还往往是团队中收入最低的那个，谁还乐意干？
- 157.软件质量不是一蹴而就的，是持久战！
- 158.抓大放小，有重点的测试；上线前验收环节，场景明确，且验收到位。
- 159.需求请不要变来变去，架构可以思考得更多一些。
- 160.软件质量离不开好的测试框架，也离不开好的管理。
- 161.质量不是测出来的，而是设计出来的。每个人都应该成为质量专家。
- 162.交付时间越来越紧张，测试效率亟待提高，版本节奏过快也导致无法去深入探索新的测试技术
- 163.时间不花在一开始质量把控 就会花在不停的改 bug
- 164.软件质量是个良心活，干好了大家感知不明显，出点问题一堆人问候
- 165.互联网寒冬下，业务质量的优先级最高，技术提升优先级被降低
- 166.质量不行，可能让前期努力付诸东流！
- 167.软件质量的高低不能单纯的依赖需求设计，也不能依赖测试最后的质量把关！我一直觉得一个项目/一个系统的好坏，不是某一个人或某几个人的事情，而是参与到这个项目的每一个人的事情。人人都要有质量意识，全员参与到质量保障工作！
- 168.质量是设计出来的，质量内建才是出路，不要死盯着测试！
- 169.项目高质量的交付需要各个团队的密切配合，多交流多沟通，首先自己的角色做到位，其次，可以往前再迈一步。
- 170.客户市场对研发效能要求越来越高，质量与效率不能对立看。
- 171.软件质量在开发中应当作为重要的一环存在，但我们依然会因为各种各样的原因去忽略质量，从普通从业者的角度来想，如果无法从流程上控制软件质量，那保持优秀的代码规范就是软件质量的底线了！
- 172.有独立的 SQA 部门，但效果不大。
- 173.软件开发前期应该有明确的目标，中途不能更换目标；闲时勤练兵，忙时打胜仗，要注重人员能力提升；质量不是一个人一个部门的事，需要更多人参与进来。
- 174.软件质量持续构建，测试圈不断打破和外延。如何更好的保障软件质量是个值得不断探索和探讨的话题
- 175.虚虚实实，落地能有几何？
- 176.软件质量，人人有责。
- 177.当进度和质量有冲突时，很多情况下质量需要让步，特别无奈。所以只能是在有限

- 资源、时间下，将产品最真实的质量情况暴露出来，供高层做最准确的抉择。
- 178.质量不是测出来的，审计、设计出来的
 - 179.且战且退，且退且战。
 - 180.质量涉及软件研发流程中所有角色，每一个角色都非常重要，之前我们更多在点上优化，其实哪漏补哪，缺乏系统性的思考和规划。
 - 181.测试也需要学习，不仅是点点点。测试领导应该技术过硬才行，否则难以推动和规划测试部门变革。
 - 182.测试人员的底线不能让步，团队的软件质量意识很重要。
 - 183.质量建设从无到有虽然很艰辛，但软件质量提升是整个研发组最大的动力和目标。
 - 184.所有的测试都是围绕把握软件质量来进行的，所以我们要坚守软件质量而不是为了尽快完成测试。
 - 185.希望能有较好的质量管控软件，能有质量工程化，将质量可视化展现。
 - 186.质量深入人心，人人保质量的观念推广真的难
 - 187.软件质量是从基础到全局，从后端到前端，从代码到用户体验的一个整体，但从一个点来看并不全面，可是必须从点到面的进行才能看到真正的效果
 - 188.希望我们会越来越被重视
 - 189.质量应该从各个环节把控，而不应该放在最后一环
 - 190.软件质量团队内建，全体负责，主动承担，减少甩锅
 - 191.软件质量在于预防，越早阶段开展质量工作越有利，质量贯穿始终
 - 192.质量是产品最重要的固有属性之一，涉及各个相关方，因此质量好大家才会好
 - 193.测试人员不应是线上问题的背锅侠，质量工程不应是领导的面子工程和开会工程，技术应该能者上，不应 PPT 上。
 - 194.质量改进急不来，点点滴滴都需要不断构建，不断夯实。
 - 195.传统行业对测试效能提升感到无从下手，互联网的方法论工具链用不上，自己开发资源不够，测试效率低，研发对测试支持不够。
 - 196.业务不断发展，很多传统行业也开始智能化，测试也要不断更新，满足新的要求。
总之，市场是什么样，测试也要跟进去。
 - 197.不出问题靠运气！
 - 198.质量需要整体规划、谋划，不能想当然，要围绕企业当前的主要矛盾、主要需求、主要能力统筹设计。
 - 199.质量更多的要成为领导者们的意识，而不只是口头的。
 - 200.软件质量真不是靠测试人员测出来的，还是要流程体系的把控、职责的清晰、团队的配合才能让软件质量更有保障。
 - 201.目前质量无法完全保证，只能加强测试人员能力。
 - 202.敏捷模式的高速迭代，即使流程规范再好，工具支持再先进，人的思维和意识没转变过来，一切都是形同虚设，培养人才是重中之重的关键。
 - 203.大公司注重测试，中小型公司的测试和软件质量一言难尽，只能说尽自己最大努力

- 改善产品质量，不管是从流程上还是从团队的个人思维、团队人员的的规范和能力上都有待提高，一分钱一分货。
- 204.提高质量保证，测试人义不容辞！
- 205.管理层有担当、有真正的质量意识非常重要。
- 206.提升一个软件测试团队整体素质是一件很难的事情，但是也是一件有挑战的事情！
- 207.面向结果，注重过程。
- 208.质量提升是一个比较漫长的路，每个公司都有每个公司的特点和问题，需要总结问题找到问题的共性，从共性问题找根本原因，从而解决掉；在质量提升的路上没有一蹴而就的事情，而是需要做好每件小事。
- 209.目前主要是进行功能验证，缺少完整的质量工程体系和研发代码质量监控，在未来的一年需要加强这方面的工作。
- 210.软件质量体现在软件研发整个流程，不能完全由后期测试决定，缺陷越早发现，更改代价越低，需要对研发前期重点把控质量，尽量做到质量把控前移。
- 211.每次上线都担心有问题，很多时候，感觉测试是没有边界的。
- 212.质量很重要，流程落地实现需要多方配合且思想达成一致。
- 213.对问题的高度敏感，深入挖掘问题背后隐藏的真相，是质量深入挖掘的核心。
- 214.质量是产品的核心竞争力，关注外部质量、驱动内部质量。质量是构建的，是需求、设计、开发、测试、运维多方面共同努力的结果。
- 215.质量是过程控制出来的，质量是管理出来的。
- 216.质量需要策划，质量需要持续复盘提升。
- 217.软件质量需要持续关注并推进，测试要有质量守门人的主人翁精神。
- 218.质量改进没有终点。
- 219.在最初推动质量提升的过程中，追求质量和追求效率或许是一道单选题。但平衡从长远来看，质量提升势必会和提升效率相辅相成，最终共同创造业务价值
- 220.质效合一如同两条腿，不能单纯为了效率而放弃质量。
- 221.很多事都能归纳到质量，但质量人一定要抓住质量主线，才能构建质量防御。
- 222.产品要不断的敲打，才能出精品。
- 223.软件质量从需求准入就开始了，跟上客户需求。
- 224.软件质量是整个团队的事，不要总是想着谁背锅。
- 225.质量是端到端所有角色的责任，是干出来的，能用技术解决的就不要简单通过管理来推动，干成真不容易！
- 226.忙忙碌碌赶进度、慌慌张张忙救火！
- 227.质量第一应该被上升到公司级战略。
- 228.持续性测试，多维度操作。
- 229.质量不是说，关乎每一个细节吗？
- 230.技术在更新但是管理老旧。
- 231.完善体系建设，持续提升能力！

- 232.质量是软件产品的生命线，没有质量，谈何产品？
- 233.说易行难，建立全员质量意识任重道远。
- 234.质量相关的活动都需要持续性的投入和运营。
- 235.质量工程需要以人为本，多方面发展，还有很大成长空间。
- 236.软件质量重中之重，希望能在这条路上不断充实丰富自己。
- 237.提升人的意识还是根本
- 238.软件质量决定我们是否走得更远。
- 239.当业务速度与质量冲突时，领导既要速度又要质量，这样下去没人会主动考虑如何提升质量了。
- 240.软件质量很有必要提升，而不是堆砌烂代码。
- 241.连单测都不写的公司，质量无从谈起。
- 242.通过代码覆盖率，实现精准化测试落地，辅助测试，提效提质。
- 243.不要搞过程数据可视化了，只是让内卷更加严重。
- 244.针对上游环节做好预防措施，避免需求、交互不清楚导致开发返工，做好全流程数据收集，针对薄弱环节进行分析调整。
- 245.质量守护来之不易，坚持就是胜利！
- 246.理想很丰满，现实有点骨感。许多比较好的方案在实际落地的时候总会或多或少出现一些达不到预期的状况，但总体来说团队全员的质量意识还是有很大的提升。软件质量是个没有 **deadline** 的长期战役，一起并肩作战的感觉很酷。
- 247.一切不以用户为中心的软件质量都是耍流氓。
- 248.一切不基于过程质量分析的改进都是耍流氓。
- 249.因为看到了同行的质量体系，对当下团队的不作为很失望，想要转岗了或者跳槽到更有动力更纯粹的团队。
- 250.作为流程下游环节，坚持并不懈追求先进的我，给自己加油！
- 251.测试行业的人员大多数技术知识不够扎实，导致测试地位不高。
- 252.希望领导能明白，质量是需要大家一起来保障的。
- 253.能力建设与项目管理并重！
- 254.有技术的不愿做测试，测试的大多不懂技术，二三线城市测试不好混。
- 255.软件架构很重要，规划很重要，逻辑很重要，用户体验更重要。
- 256.通常质量打不过业务。
- 257.质效合一，给客户交付最有价值的产品。
- 258.业务压力下，有改进技术、有改进想法，无实施时间。
- 259.软件质量依赖过程质量，QA 应着重注重过程和工作产品，针对问题根因分析，实现过程的持续改进。
- 260.时刻警惕，一不留神就是质量事故。
- 261.质量不是靠人堆出来的，跑的快了，技术债会遗留很多。
- 262.质量不好做，责任重大！任重道远，从上到下的质量意识，并付诸行动。

- 263.软件质量直接与用户体验挂钩，希望团队内部都能真正重视起来。
- 264.重视或者不重视，质量都在那里，静静的看着你它期待着往高冲，稍有不慎也会跌落悬崖，让你感受不能承受之痛。
- 265.倾听客户的心声，挖掘客户的痛点才是最重要的。
- 266.再好的流程规范，需要人员去遵守，如果没有人严格执行和把关，一切都是空谈。
- 267.聚集核心，贡献价值，路漫漫其修远兮，吾将上下而求索
- 268.金融行业真是质量第一！
- 269.千防万防，就怕百密一疏。
- 270.软件质量的重视情况完全取决于用户的态度。
- 271.质量改进没有重点，我们一直在路上！
- 272.别在推敏捷，敏捷就是持续加班！
- 273.质量管理也要敏捷，要深入参与到价值创造过程中，不能视自己为成本部门，要将质量管理作为经营部门。
- 274.回归一定要到位，问题往往发生在你觉得没问题的地方。
- 275.测试无边界！
- 276.迭代速度加快，得尽早改变加入解决问题的恶性循环
- 277.深耕是王道，聚焦業務價值
- 278.价值体现不易，希望后续能将质量进行可估算和可视化。
- 279.每年都在改进，每年都有新的问题！
- 280.业务是一切测试的基石。
- 281.辛苦一年，一个生产 bug 回到解放前！
- 282.怎么提升测试价值的显性？
- 283.质量不仅仅是测试部门的事情，需要和每个人的绩效挂钩，责任明确，提高自驱力
- 284.质量做不好很头疼，一发版就有问题。
- 285.质量很多时候都在为进度让步，伤心啊！
- 286.总是强调质量是生命，而不知不觉中就在丧命！
- 287.质量是核心，把握软件的核心，不断造福人类
- 288.软件定义一切时代的到来，对其质量要求越来越高，软件人员在未来道路中任重道远！
- 289.高质量产品出自精益设计和打磨优化。
- 290.一有问题就是测试的问题，有的时候很无奈。
- 291.没有质量文化的土壤，只能空谈质量。
- 292.守正笃实、久久为功。
- 293.道阻且长，行之将至。
- 294.团队文化对软件质量影响巨大，例如开发规范、SonarQube 的应用都十分依赖开发人员的配合，不然在这之上建立的发布日志和质量阈值无法达到预想的效果。
- 295.加大基础工具开发，有利于从客观上保证流程和质量。

- 296.希望各行各业更重视软件质量，去除浮躁心态，往精细化发展。
- 297.企业文化、研发老大的理念对质量领悟的工作影响很大
- 298.软件质量，想说爱你不容易！
- 299.软件即将引导汽车时代的变革，车载软件质量势必重要，希望能在自己本职岗位为软件质量做出贡献
- 300.质量真的是构建出来的！各个环节都竭尽全力，质量就能放心。
- 301.重视进度，不重视软件质量的问题太严重！只管交付，不管中间过程的问题也很突出。
- 302.前期预防大于后期努力！
- 303.缺少质量看板，质量没有和绩效挂钩！
- 304.软件质量仅靠测试推动，难上加难，而在部分领导者的心中，质量不行大多都是测试的责任，造成测试团队背锅现象严重。
- 305.甲方就是外行指导内行，我们需要好的产品经理把需求打回去。
- 306.保障软件质量太难，尤其是当开发根本不当回事，所有的流程规范都知道，但是就是不执行，即使领导也没有达到公私分明。
- 307.质量每次想做点什么事，都要为交付让步。
- 308.不怕有缺陷的软件，就怕缺陷不能够弥补，更怕不做复盘进行改进。
- 309.软件质量是企业的生存之本，任何时候都要放第一位，且全员奔向同一个目标。
- 310.打铁还需自身硬。
- 311.应该自上而下、统一思想，任重道远
- 312.软件质量是环环相扣的，需各团队有统一质量意识，提升每个环节的质量。
- 313.软件质量好了，交付才更高效。
- 314.质量意识是内建于整个项目流程的，从需求到研发、测试、运营，不是测试一人之力可以改变的，所以谈质量就要有相应的规范、标准、改进措施，创造质量文化，让质量深入人心。
- 315.质量意识需要培养，团队所有人都要对齐。
- 316.构建完善的质量保证体系，提升客户体验。
- 317.质量的问题很复杂，不是一句话能说清楚的，做一个真正理解需求的人。
- 318.全员质量管理，才是发展之道！
- 319.质量第一的认识，还是没有深入人心。
- 320.专人专事，各自其责才能把控质量；如果做不到质量内建就踏踏实实专人专事。
- 321.设计人员能力建设，软件产品化建设是重点。
- 322.做了很多自动化，但是防护网太多，不清晰。
- 323.软件质量，人人有责；加强软件测试数据收集，有助于提高效能和风险识别能力。
- 324.想说爱你不容易，过往种种皆为云烟，任重道远，继续加油！

十、关于组织者（排名不分先后）

1. 《软件质量报道》公众号



本公众号致力于健康、安全、绿色的软件生态，分享软件质量管理、软件测试的思想、方法、技术与优秀实践，追踪软件质量领域的热点，及时报道软件质量管理的成功案例或质量事故，以及分享深度思考、有温度的技术文章等，努力成为您工作中的朋友。

2. 腾讯 WeTest



腾讯 WeTest 是由腾讯官方推出的一站式品质开放平台。十余年品质管理经验，致力于质量标准建设、产品质量提升。腾讯 WeTest 为移动开发者提供兼容性测试、云真机、性能测试、安全防护、企鹅风讯（舆情分析）等优秀研发工具，为百余行业提供解决方案，覆盖产品在研发、运营各阶段的测试需求，历经千款产品磨砺。

3. Testin 云测试社区



Testin 云测试是云测试模式的开创者和引领者，通过资源的汇聚和技术的创新，为企业软件和系统提供全生命周期的云测试服务，服务涵盖 DevOps 体系咨询、测试管理数字化、功能测试、性能测试、兼容测试、安全测试、自动化测试、自动化监控、远程真机等多项服务。官方网站：<https://www.testin.cn>

4. MeterSphere 开源社区



MeterSphere 作为一款一站式开源测试平台，涵盖测试跟踪、接口测试、性能测试、团队协作等功能，全面兼容 JMeter、Postman、Swagger 等开源标准，有效助力开发和测试团队充分利用云的弹性进行高度可扩展的自动化测试，加速高质量的软件交付。自

2020 年 6 月份发布至今，MeterSphere 项目得到了开源社区的广泛认可和积极反馈，并在众多企业内落地使用。截止 2021 年 11 月，MeterSphere 项目在 GitHub 上获得了超过 5000 star 和 50000 以上的下载。

5. AITest 掌动国产化社区



AITest 掌动国产化社区致力于软件国产化替代，旨在研发质量可靠性“卡脖子”关键技术，在超大规模压测能力、智能化测试软件、高性能测试算力等可靠性测试技术上弯道超车，打破发达国家垄断，提升我国底层信息技术核心竞争力。

6. PerfMa 质量技术社区



PerfMa 旗下软件质量技术交流平台，呈现 PerfMa 在软件质量领域最新的技术发展和创新，并汇集软件质量领域技术大牛，用心打造高质量、高品质软件质量领域技术内容，传递、分享业界前沿最佳实践及探索。

7. CSDN 社区



CSDN，打造世界最大的中文 IT 技术社区，成就一亿技术人。

8. Thoughtworks 社区



Thoughtworks Community 由来自 Thoughtworks 全球技术人才及创新爱好者发起并组建，本着自由、开放、分享、互动的原则，由社区志愿者发起并组织多样化的技术主题活动，希望以此分享洞见，带动行业及地区的整体技术能力，推动创新与 IT 变革。