Talk is not cheap

（本文描述的是我长久的经历中形成的看法，跟我现在身边的人和事没有直接联系，请勿对号入座。）

长久以来，我发现挺多 IT 人士学会了一句口头禅，无论你表达什么观点，他们会拿出一副小学老师要检查作业的口气，说：“Talk is cheap. Show me the code!” 或者 “给我看看你做出了什么！”

这类说法包含了两种可能的含义：

1. 说话是没用的，你要做出来我才信。
2. 你要已经有了重要的成果，才有资格发言。

“Talk is cheap. Show me the code.” 这句话出自 Linus Torvalds 在 linux-kernel mailing list 的一个回帖。如果你看了我对微内核和线程的分析，也许会明白 Jamie Lokier 的话其实是有意义的。如果保持开放的心态，继续的探讨也许会给 Linux 内核带来突破性的改进，然而这种可能性却被 Linus 一句“Talk is cheap. Show me the code.” 给扼杀了。

Linus 可能当时不耐烦了，你知道这家伙的性格…… 我相信他不是每次都说这样的话，但因为 Linus 形象太高大，这话就被人记下来，作为可以反复拿出来压制言论的手段。管你表达什么，他们都有一句万能的台词：“Talk is cheap. Show me the code.”

“苦干，用代码说话，忽视想法”，是很多程序员的误区。人的思想不一定需要代码来证明，甚至很多的想法无法简单的用代码表示，只有靠人的头脑才能想得清楚。思想是首要的，代码只是对思想的一种实现。我们先得要有思想（算法），才可能有代码。有些人动不动就“show me the code”，却忽视了思考和探讨的重要性。如果你没有好的想法，弄一堆代码出来又有什么用呢？只是死钻进一堆无谓的细节，掩盖了本质。

代码不能代替思想交流和讨论。代码不能清晰的表达一个人的想法，也不能显示一个人的思维深度。任何程序员都可以写出复杂冗长的代码，你有时间去看吗？就算水平很高的程序员，他的代码组织方式你不熟悉，也会看不出来本来的想法。实际的代码里面往往会充斥着因为编程语言，硬件，系统，历史遗留问题导致的各种复杂性。如果每个想法真要“show me the code”才被考虑的话，那效率实在太低

了。

我跟同事讨论代码的时候，一般都会先请他们在白板上画个图，用简单的语言解释他们的算法，这比直接看代码要容易很多。很多时候几句话就能说清楚，我在脑子里就能看到它是怎么工作的，根本用不着看代码。我从来不会说“show me the code！” 想法应该在实现之前被讨论清楚，对不可行的想法应该停止于摇篮之中，对可行的想法应该看到各种可能的发展方向…… 这些都应该在实现代码之前沟通弄明白，这会节省我们大量的时间和精力。

这就是 Talk 的价值。

另外，代码和成果不应该成为一个人是否可以表达看法的条件。只要这个人的见解有它的依据，就是有价值的。他不需要有什么重大的成就也应该可以表达自己的看法。这个我在之前一篇文章说过。以代码和所谓“成果”来压制人的言论自由是不合理的，而且代码和“成果”其实很难说明一个人的看法是否有价值。

很多人面试程序员都有类似的经验，他们给你看已经写好的代码，根本无法用来鉴别他们的水平。因为代码是可以拷贝的，所以你无法知道这代码是否他自己写出来的。代码可以是冗长晦涩的，所以就算是他自己写出来的，你也不会想花时间去看懂它。

代码是死的，它是对已有问题的解决方案。而你想要知道的是这个人在面对新的问题的时候，他会怎样去解决它。所以你必须知道这个人的思维方式，看清楚他是否真的知道他声称“精通”的那些东西。

一个人说他之前的工作做出了什么样的成果，很多时候也是不可靠的。因为成果是可以虚构或者盗窃的，他可以把别人的成果说成是自己的。如果是管理岗位，“成果”就更加难以鉴定。这人也许只是瞎指挥，对很多人各种发号施令，对不同的人指出 N 种不同的方向，然后瞎蒙对了一个。其中一个方向做出了点东西，当然工作都是手下人做的，具体的想法都是手下人的。然后领导者挂个名字，就成了大家追捧的“技术大牛”。

很多博导都是用这种方法出成果的。招 N 个博士生来，分别给他们每人一个课题。管它有没有可能做出来，有没有价值，都跟你说这个课题很好。只要 N 个博士生有一两个做出东西，他就可以发 paper 升职了。被分配到那些做不出来的方向的学生，他才不管你的死活呢。

有见识的人跟他们对话，就会发现这些人一知半解，还仍然牛逼轰轰的样子。这就是我多次的经历。管他说做过什么代码项目，写了什么书，得了什么奖，一旦当面对话就能显示出真实的水平。代码和书都可以抄来，成果可以盗窃，你甚至可以因此得诺贝尔奖，可是对话没法偷来。对话可以显示出一个人是否有真知灼见。

一个小故事。我以前就职的某公司，有次招了一个 VP，他的 github 上有上百万行的代码，项目有上万的“star”，在领域里很是有点名

声。这算是成果了吧？结果一进公司就各种瞎指挥，搞得大家没法工作了。还招进来很多自己圈子里的亲信，也是一群只会吹牛不做事的人，各种打压其他人，浪费大量的人力物力。我都感觉公司快要被搞垮了，最后创始人终于醒悟，费了好大功夫才把这些人赶走或者架空。

所以一个人有再多的代码，成果，都可能是没用的。我不排除它们是真实的情况，但你需要懂得如何去鉴别，而不只是依据这些表面的标准。

对于人的水平，我一般会观察他们说的话，最好是当面的对话。我不会盲目相信他们所谓的“成果”，我不看他们的代码。有真知灼见的人可以毫不犹豫地说出自己的想法和观点，而不需要时间去背诵和计算。我很容易看出一个人是否在说真话，因为说真话的人不需要时间去“计算”他们要说什么，不需要演戏。

可惜，“Talk is cheap”已经成为了很多人用来压制言论的手段。它误导了很多人，让他们无法正确鉴别技术人员的水平，犯下严重的人事错误。招进来一个错误的人，可以毁掉整个公司。

那些被“Talk is cheap”压制的人，变得不敢表达自己的观点，总想默默无闻“做”点什么给大家看。可是对方有什么资格要求这些呢？他们自己做出了什么呢？等你真做了给他们看，他们又会说你的东西不好，不如别人 xx 的。想当年我做的那什么，比你厉害多了…… 其他人也云里雾里，没有鉴别能力，只能随机倒向一边。

受到“show me the code”影响的人，可能还会让别人去看他的代码，而不解释自己的想法。大家工作中也许遇到过有人拒绝解释自己的想法，说：“你看代码就明白我做了什么。” 这其实是不大尊重人的行为。没有人应该被迫去阅读其它人的代码，写出代码的人有义务讲清楚自己代码背后的思想。

由于我对某些领域很在行，不止一次有人 email 联系我：“我做了这个东西，我想知道你对它的评价。” 连基本的想法都不说，甚至称呼和自我介绍都没有，接下来就是一个 github 的代码链接，或者粘贴一大段代码在 email 里面。这种代码我是不看的。我可能 email 都不会回，因为这显示出他们缺乏基本的礼貌和对他人时间的尊重。

代码不是有效的沟通工具，也不应该用来决定一个人是否有发言权。“Talk is cheap”只不过是封嘴的手段。说别人“Talk is cheap”的那些人，他们自己却不断地 talk。

人们应该可以平等自由的表达自己，不受这种无谓的教条压制。每当有人一针见血，指出我迷惑已久的问题的要点，豁然开朗的时候，我会很清楚的记得这个人。我会尊敬他，在合适的时候给予他回报。就算没有给我新的想法，要是他说出一些我曾经琢磨很久才想清楚的点，或者给了我另一个角度的观点，我也会记得他。这些给我指出正确方向的人，我不需要看他们的代码，我不以最终的“成果”来衡量他

们的价值。想法和观点在我这里是高于代码的。

Talk 一点都不 cheap，而可以是有很大价值的。中国有句古话，“听君一席话胜读十年书”，说的就是这个道理。当然我们还是应该避免无意义的对话，但不应该笼统的说“Talk is cheap”。

可是我发现并不是每个人都像我这样。有些人，他迷惑的时候你给他指出要点或者方向，最后他却说那是他自己想出来的，甚至说你的话没有价值，我只看结果…… 遇到这种情况，你就知道遇到了错误的人。你不需要向他证明什么，不应该再给他任何有价值的信息。

很多人被“成果”或者代码所蒙蔽，而忽略了那些能够看透问题，用简单的几句话指出正确方向的人。我希望以这篇文章纠正很多业内人士的思维方式。

Talk is not cheap. Talk can be powerful.