技术经理应该把 30% 的时间用在编程上

在一个科技公司里，软件技术经理用在编程上的时间应该不低于总工作时间的30%。无论是管理一个团队，还是一个分部，还是整个公司，当技术经理用在编程上的时间低于30%时，他执行职责的能力就会发生严重退化。

我的这个断言可能跟那些我看到的想成为团队首领的软件程序员们期望的情况完全相反。每次晋升，程序员们都期待花在编码上的时间会大幅度减少，当从“leader”爬到“经理”职位时，就应该彻底脱离编码活动。而且，他们期望以一种“动口/眼不动手”的方式来保持对代码库的熟悉。再上级的领导就跟编码完全没关系了(如果有的话)。

大概一年前，当时我的时间被越来越多的其它事情占用，例如招聘，管理，开会等；我就发现，作为一个技术首领，当花在编程上的时间低于某个比例后，管理效果和工作效率就会出现问题。之前我写过一篇短博客阐述过这种体验和观点，但没有展开具体的描述。这里，我将会对这个观点展开更详细的论述。

### 为什么要坚持编程活动

很多人认为，做为管理者，应该退出战斗第一线，专注于大战略和管理工作。当然，管理者把大部分的时间用在这种事情上是应该的。但是，在我们这样一个行业里，因为我们允许或要求管理者几乎不再去编程，现实让我们付出了沉重的代价。一旦一个人停止编码，他和程序员们关心的事物之间的重要联系就会退化。当这种情况发生时，决策，计划和干群关系就会出问题，从而瓦解了将技术人员提升到管理职位的良好愿望基础。

### 项目开发评估能力

程序员的百宝箱中最重要的一个绝活就是估计工期。如果没有准确预估的能力，整体计划是不可能正确的出台的。大家也知道，做为一个族群，程序员们对工期的估计是臭名昭著的。糟糕的不能再糟，事实上，当从程序员口中得到一个预估的数字后，公认的方法是将它乘以二。通常，程序员都会对开发工作抱有非常乐观的态度，但如果我们使用“estimate traction”理论，就会发现，编程活动表现出特别易变的特征。因为我可以用很多方法实现一个功能，当我们在还没有深入细节之前，我们的估计就是不可靠的。

### 技术债务

另外一个事情是，技术经理必须对[技术债务](http://en.wikipedia.org/wiki/Technical_debt)给项目造成的影响掌握第一手的资料。如今，技术债务这个术语非常流行，常被用来当作争论是优先开发新功能还是先重构老代码的弹药。对“技术债务”这个词的内涵熟悉的人通常最容易发起论战。作为技术经理，你不仅仅是要熟悉这个概念，它们会在你判断何时偿还技术债务的决策中起直接作用。经常写代码的经理拥有更多更有价值的信息来判断何时/如何做出这样的决策。

### 知情的连续性

我并不是随意选择30%的比率的。我是基于自己的经验，将足够的时间参与到开发活动中，你很容易就能时刻掌握代码库的任何变化。如果时间太少，你对开发动态的掌握就是断断续续，无法连成线。一旦断了线，我就需要重新理顺脉络，由此得到的惩罚就是浪费了额外的时间。

### 分担责任

作为负责人，你不可能让所有决策都由你制定或由你批准。但你需要了解所有决策的前因后果和背景知识，来辅助这些决策。最终，你要为这些决策的后果负责，你对项目情况的掌控能力要能匹配你的这份责任。

### 积极参与编程赢得团队尊敬

大家需要明白：要想成为一个成功的经理，你需要为团队成员提供服务，促进开发，确保他们完成任务。我曾在一篇博客里写过如何诊断和修复经理们有问题的干群关系。但是对于的管理程序员来说，你需要热爱编程。因为你的团队在编程，如果你在编程上做榜样，他们都会对你肃然起敬。

### 达到30%的障碍

尽管付出了最大努力，我仍然在保持30%的编码时间上遇到了很多的阻碍。包括下面这些：

**工作繁多：**在一个创业公司里，你总有忙不完的工作需要去做，即使在公司有规模、壮大后，如何对众多都很重要的事情排优先级也是一种考验。技术经理有很多职责，完全会占满他的70%的时间。下面就是一些：

* **领导和照顾团队：**这是一个技术经理的第一要务。你已经不能够只为自己的工作负责，你要为你的团队能保持最好的工作状态负责。指导团队任务，解决纷争，思考如何优化工作条件来提高工作效率和舒适度，这都需要时间。
* **做决策：**随着职权的增加，技术经理需要花更多的时间放在各种各样战略上的统筹和计划等事务上。起初，也许只是一些技术方面的决策，但之后，产品开发和竞争策略方面的事务将会占去很大一部分。
* **招聘：**经理，总监，副总们，CTO都需要组建自己的团队，有时需要迅速组建。一个好的招聘高手能带来很大帮助，技术经理在这样的事情上的作用是无可替代的。好的技术经理会活跃的跟上级保持沟通，不断的将自己团队中的优秀人士推荐出去。
* **客户：**随着技术经理职权的增加，他们经常会对外抛头露面。他们会被带去参加“推销会”，期望他能带去一些有深度的话语。或当重要客户不高兴时被叫去灭火。
* **公关：**资深技术经理会把一部分时间奉献给公开演讲，写博客，或者报刊上发表文章。不论你在这些活动中出了多少力，这些写作、编辑、排练、差旅、出席活动等都是花费时间的。

**夺回时间：**上面我说的这些活动都是一个技术经理应该投入时间的事情。下面要说的是我从经验中发现的一些陷阱，是它们在阻挡我努力保持20%最低限度的编码时间，至今仍站在我面前，妨碍我重回30%的目标。

* **不勇于说不：**高成就意味着要努力工作；然而，做事要稳妥，一个技术经理最重要的职责就是，当你的团队负担过重或接近这种状态时要勇于说不。如果你这个时候不说不，其他人就会开始对你的计划和工期承诺指手划脚。
* **开会：**有一个巨大的家庭手工业行业都在为如何高效的开会出谋划策，这是有其自身原因的。我的职业生涯中浪费在会议中的时间算是最多的了。尤其是不断的面试或出席必须由团队首领出席的会议。

### 失败的策略

当在探索如何夺回我的编码时间时，有很多的方法并不奏效。

* **少睡：**这种方式虽然对我有巨大的诱惑，但其实牺牲睡眠时间没有一点效果。你的大脑需要休息，缺少睡眠会影响情绪并降低工作效率。
* **只看头文件：**我以为这种方法可行，但实践中，只看提交的C++代码的头文件对你的管理工作的帮助甚少。
* **专一：**作为团队首领，你只关注代码库里的一个项目也许是可以的，但对于总监或更高级别的人来说，你应该对负责的所有东西都要熟悉、了解。
* **委托：**有时候为了多做工作，会将一些事情随意的交给他人做，但实际上一些像写报告这样的事情你一定要认真嘱咐才行。

### 成功的策略

尽管走了很多死胡同，我还是发现了一些成功的方法：

* **时间分段：**我的日程表上没有被预先分配的时间是非常少的。想起来这也是很显然的。于是我专门为编程特地分出一些时间段。实践中，这些时间段经常会被重新调整，虽然每周只挤出8小时，效果是完全不一样的。
* **委派：**委派要有技巧，尤其是在你对如何执行抱有强烈想法并有能力去做时。有很多原因导致一个经理反对将任务委托他人，但事实上每个原因都应该被当作一个现存的需要解决的问题，而不是一个不可逾越的障碍。没有什么比放手让一个你信赖的人替你主持一个会议能释放你更多的编码时间了。
* **工作时间：**将时间分段，工作时间里尽量避免打扰。在这些时间段之间的时间里，我会干一些不重要或不需求注意力长期集中的事情。

### 最后几招

下面是一些经验建议，送给那些发现自己试图达到30%但无法接近的技术经理们：

* 学习如何读代码。跟写代码比起来，这是一种完全不同的技巧。
* 指定会议流程，对会议进程保持控制。不参加任何没有计划的会议。
* 用一台好用的电脑。你喜欢MacBook Air？不，别用它。
* 清楚如何访问一个开发环境，这样当有修改时可以快速测试。
* 记住你是把一小时分成5个时间段使用的人。如果有事情需要一小时，在日程表上标明。
* 20–30%是我自己的发现。你的也许跟我不同。评估你自己的(你修复一个紧急bug需要多少时间？你知道代码库中麻烦最多的程序是哪一块吗？随机找一段程序，看看你是否知道是做什么的。如果不能，说明你需要更多的时间)。
* 分类列出哪些事情什么时候做，哪些事情应该完成。(知道Getting Things Done (GTD)的人会看出这是提高工作效率的基本技巧。)
* 最后，我最近越来越喜欢把东西写到纸上。跟感觉上相反，打印出来的说明，把一些需要排优先级的任务列在纸上，或者是一段代码，经常的，它会成为把大量时间盯着屏幕的一种平衡。