别人家的技术leader是如何建设团队、管理人员、沟通工作的?

InfoQ 2017-04-20

作者 | 周明耀

编辑 | 小智、Emily

> 这篇文章将重点介绍如何从零开始构建一支技术团队，以及如何做好团队和人员的管理、沟通工作。看看，别人家的”技术 leader是怎么做的～

程序员经验对比

我这里简单做了一个表格，举例说明对于某一个岗位不同能力的程序员对比：

入门级程序员 了解 Windows、Mac 或 Linux 对良好的编码实践有初步的认识 了解因特网技术 了解数据库技术 了解 C/C++ 适应团队工作，能根据指导完成工作 能够配合领导制订工作计划	**高级程序员** 开发过两个或多个商业应用软件 熟悉多种平台 熟悉因特网技术 熟悉数据库技术 深刻理解 C/C++ 有良好的软件设计能力 有良好的沟通能力 能够自我激励，需要的指导很少 有优秀的分析、项目规划和工期预估能力 能够密切关注形式的变化，并制订调整计划 提出、改进并推广新的流程
有一些经验的程序员 开发过一个或多个商业应用软件 熟悉 Windows、Mac 或 Linux 有良好的编码实践经验 熟悉因特网技术 熟悉数据库技术 有扎实的 C/C++功底 能够自我激励，能根据指导完成工作 能够独立制订工作计划	
有经验的程序员 开发过两个或多个商业应用软件 熟悉多种平台 熟悉因特网技术 熟悉数据库技术 精通 C/C++ 有良好的沟通能力 能够自我激励，需要的指导很少 有良好的项目规划和工期预估能力 能发现问题并协助团队进行调整	

组建团队

招聘人员

「一流人才招聘一流人才，二流人才招聘三流人才。」

—— Steve Jobs

了解岗位需求

在招聘之前，我需要明确知道自己需要怎么样的程序员，需要关注自己所提供的的岗位，是需要经验，还是需要技术钻研的热情。看看这两类，你需要哪类：

- 一个可以领导整个团队开展各种实际工作的程序员？

- 一个可以找出扭曲难寻的设计缺陷的码农？
- 一个有大局意识、能预想到你的需求如何可以分解为模块和组件的设计师？
- 一个习惯于主动行动、能很好地配合管理层的工程师？

or：

- 在短时间内可以编写数千行代码？
- 快速做出对客户非常重要的原型系统？
- 快速领会业务流程，设计围绕根本需求？

上面提到的这些特质，它们互相并不是排斥的。但第一种类型的程序员可能是经验丰富的老手，后面那种类型的更可能是充满热情的新手，关键看你的岗位、行业需要怎么样的人。

编写职位描述

当我了解清楚自己的实际需求后，我需要针对职位进行描述，这样无论通过猎头、猎聘网、Boss直通、51Job等等方式，都可以让读者查看是否该岗位符合他的实际情况。职位描述如下所示，一般包含以下三部分内容：

- 基本信息，包括岗位、部门、直接领导、状态、工作地点；
- 职位概述，包括工作职责和预期表现；
- 岗位最低要求。

头衔：服务端开发工程师

部门：C++程序设计

直接领导：C++团队负责人

状态：全职

工作地点：杭州

职位描述：入门级职位。负责代码开发、转换、验证与维护。负责根据已有的文档和代码质量标准，撰写风格良好的源代码。能较好地融入团队，并遵循团队管理者和资深团队成员的指导。能在直接团队管理者的指导下开展工作，并就出现的问题进行沟通。

职位要求：

- 具有研究生学历，一般情况下要求 211 以上学校毕业。
- 掌握 Windows、Mac 或 Linux 中的一种，掌握两种及以上优先。
- 掌握 C/C++及调试技术。
- 对良好的编码实践和基本的计算机科学原理有初步的认识。
- 对因特网技术、通信协议及技术有所了解。
- 对数据库技术有所了解。
- 对多线程技术有所了解。
- 适应团队工作，能根据指导较好地完成工作。
- 能够自我激励，具有较强的学习能力。
- 能够配合团队管理者制订工作计划，并预估工期。
- 能够根据形势的变化做出相应调整。

人员分类

“独狼”和“农民”

我把普通程序员分类为“独狼”和“农民”这两类，“独狼”更多喜欢一个人完成工作，也就是说，当问题出现时，他们的第一反应是去独自解决问题。他们常常跳过规划，最终得到一次性的解决方案。

从团队角度来看，我更加希望软件开发像种地一样。农民会有条不紊地了解地形、研究土地的化学组成成分、种植、浇水、除草，最终收获粮食。可靠、可扩展、可维护的软件都是这样有条不紊地开发出来的，这其实是日积月累的能力。

许多“独狼”都是优秀的程序员，但是你需要对他们进行“贴身”管理。他们有着当主角和引起团队内部纠纷的倾向和能力，需要做的是尽量多地关注他们的工作，了解他们具体的工作内容，如果出现问题要马上采取措施纠正，否则最终情况可能会失控。

只能当“独狼”的程序员不会在任何一家企业待太久。要么是你对他们总是自顾自地向前冲感到厌烦而辞退他们，要么是他们对长期受限制感到厌烦而主动辞职。

“英雄”

这一类人指的是承担需要极大努力才能完成的任务，并最终取得成功的人。在付出这些非常人所能承受的努力方面，“英雄”和“独狼”有点相似，但“英雄”更能够在团队工作和开发过程获得成功。此外，“英雄”大多是团队内部培养出来的，不是从外面雇佣过来的，很多时候他们会在企业中崛起为超级明星。管理“英雄”具有一定的挑战。如果你总是希望他们付出超人的努力，时间长了后会发现使用过度，造成和你之间的矛盾。

此外，作为技术团队管理者，你也需要有一定的技术功底，能够让“英雄”觉得可以从你这里学到什么，这样才能合作更为愉快。最后，需要对“英雄”的薪资、福利有所倾向，并且多和他们沟通，让他们自己选择技术方向，并把他们放入关键项目里。

内向的人

内向的人表现为非常沉默、内敛，几乎感觉不到他们的存在。他们可以把工作完成得很出色，但是对团队执行力或者在会议上几乎没有什么贡献。他们在一对一的时候能进行交流，但退回到人群里以后几乎消失了。

在会议上让他们发言时，当他们分享自己的意见或见解时，要给予正面的支持，这样可以逐渐帮助他们建立自信，让他们感觉到自己对于团队是有贡献的。找机会跟他们交谈，当面认可他们的贡献。与他们的交流要单独进行，通过一些小事情与他们建立特殊的联系，例如分享工作经验、交流管孩子心得。总之，想方设法建立更紧密的联系。

带有明显负能量的人

尽量避免团队里存在具有很强负能量的人，他们会通过挑拨离间和散布不满情绪来毒害整个开发团队，并且对组织造成严重的破坏。如果没有他们，那些负面情绪可能永远不会出现。比如“团队管理者根本不关心大家”，带有负能量的人会进一步夸大事实，把一些细枝末节的事情说成是管理层故意针对程序员的行为。客气点说，这样的言论是公然藐视真理和道德的。发现这类人，尽快沟通或处理吧。

各类奇葩

不管是一心想当“老大”，不择手段压制同伴的奇葩，还是对人很粗鲁的奇葩，或是到处借钱的奇葩，对于各类奇葩，早点让他们走，无论他们的水平有多高，都不要犹豫，也不要期望自己能够改变什么，他们的性格是受了家庭或者个人经历严重影响的。让这些人尽快远离团队，这样整个团队都会轻松很多。

管理团队

管理的最终目标是：“不要让你的下属陷入困境，不要让你的同事陷入困境，尤其是在任何情况下，都不要让你的上级陷入困境”。

技术尊重

要了解程序员，你首先需要深入理解他们使用的工具、流程，以及程序设计的艺术。你理解得越深入，在和下属程序员进行技术对话时，参与能力就越强，越容易获得他们的尊重。微软的一个程序架构师这样评价比尔.盖茨：“盖茨最喜欢和他的程序员一起将程序分析到比特、字节层面。在技术战斗中他可以非常轻易地守住自己的阵地，他之所以可以获得程序员的尊重，因为他可以轻易地战胜他们”。

成功地管理程序员最重要、最关键的因素，是得到你管理的下属的技术尊重。如果没有技术尊重，那么你的每一个具体想法，都可能会遇到主动或者被动的阻碍。正是由于这个原因，那些在职业生涯的某个时期没有做过程序员的团队管理者，才会觉得有效地管理程序员是极其困难的事情。

强化现有的团队

假设现有团队全部由普通程序员（能够完成交代的工作，但是没有主动创造能力）组成的。我们需要做的是招聘1-2位杰出的程序员，这一步要耐心，明确候选人是否是真正的杰出程序员，因为只有招聘到正确的人才能让工作高效，如果招到的员工很差，那么你就没有时间去处理其他的工作了（总是有各种问题不停地困扰着你）。

系统程序员/架构师容易在团队里显得有点格格不入，这是因为他们很多都是“独狼”，他们可能脾气很差，也可能技术上很有个人主义。这也是个人成就差别最大的群体。和这类人不同，杰出的程序员能够以一种优雅而简洁易懂的设计来架构大型的复杂系统，这些优秀的系统往往能让所有其他程序员的工作都更加轻松，因此，单人就能带来巨大的杠杆效应。我的理解是需要让系统程序员/架构师深入到开发工作，不要让他们只设计、不编码，应该把他们引导为杰出程序员，否则他们最终可能成为团队的鸡肋。

团队组成

杰出的程序员需要一群称职的程序员来配合，依赖这些程序员来完成日常的开发工作，实现设计好的系统和产品。和橄榄球类似，一个杰出的橄榄球队中必须要有那些负责阻拦和抢断的队员，而一个杰出的开发团队则主要由那些称职的程序员组成。这让我想起了电影《冲锋陷阵》的最后一幕，四分卫拿着球向对方阵地冲去，周围一群队友阻拦对手球员，能力弱的以自己的身躯直接和对方一对一拼掉，能力强的干掉一个又一个对手，直到四分卫冲过对方底线。这部电影我看了很多遍，也是我带领团队的精神指导，每次看到这一幕，我都依然会非常激动。

比较合适的团队成员组成：1-2名“英雄”+大多数“农民”+极少量“独狼”。

进度管理

我有一块小白板（不是那种很大的），我把它放在自己的面前。每天早上我都要写上今天需要参加的会议、自己要做的事情，此外，每天上午半天时间我会和每一个项目（产品开发、预研、调研，都可以）的团队成员过一遍当前进展。大家坐下来，好好谈谈已经实现的设计或代码，对疑惑、问题进行讨论。

因为这种方式可以确保自己不仅仅依赖于状态报告、项目时间表，这种方式也可以让你能够接触到说真话的员工，他们会告诉你哪些地方做的不够好，并且会主动请求团队管理者帮助，而不需要团队管理者来催促他们。最高效的团队管理者往往都是坦率的，也往往对下属有足够的时间，能让员工找到他们说出自己的想法，他们会认真倾听。

引导工作

团队管理者工作中的一个重要部分是引导事情走向正确的方向，并确保团队成员之间以及团队之间有正确的沟通。对于一个团队管理者来说，要想最大化地利用自己的时间和技能，就要引导程序员自己做出正确的决定，而不应该自己就把决定做了。这样做，可以帮助下属员工培养技术、积累经验、建立自信，还能获得那些具体执行决定的员工的认同。

如果你发现自己经常需要讨论非常具体的命令如何执行，那说明你没能很好利用你的管理技能，或者没能赋予下属员工足够的权利。作为团队管理者，你必须指出大方向，然后做好充分的检查，以确保员工做出正确的决定和实现。及早检查下属员工做出的重要决定，否则当你想中途接入并修正时，员工可能已经做了很多无用的工作。这也是为什么我在“进度管理”一栏中强化自己的每天进度跟踪、讨论的重要性。

保护成员

做过项目的团队管理者一般都有这样的经历，团队成员正在专心处理现场问题，莫名其妙被人投诉，投诉可能来自市场部门，也可能来自技术支持，或者兄弟研发部门，都有可能。也容易出现团队成员每天被大量无用的会议烦扰，不去的话就要被投诉，这类情况在大公司司空见惯。

我们要学会保护团队成员，让他们免受组织中每日泛滥不绝的各种问题、争议和“机会”的干扰。在大一些的公司内部，官僚主义政治会通过各种文书工作来忽略或者缓冲每天的各种请求和问题。小一些的公司里，面对挑战的是各种销售驱动的机会、客户驱动的争议问题，以及管理驱动的想法，你作为团队领导者，可能是他们最后或者唯一的防线。

另外一种你必须提供的保护是，保护你的员工免受开发之外的同事或者部门的攻击。这些攻击或者抨击往往并没有充足的信息或事实根据，有些抨击是出于好心的，有些则是恶意的，甚至可能包括个人攻击。我的建议是对这种情况保持警惕，处理之前先充分了解情况，如果确实存在无根据的抨击情况，事发当时或事后私下沟通，告诉那个发出抨击的人，指出他的行为是不恰当的，让你无法容忍，让他知道事情的严重性，而不是一味指责自己的员工做得不够好。

有一个情况我想特别说明，公司内部非重要部门组织的会议，尽量不要放在周五的晚上、休息日进行，看起来很有效率，其实是在过度使用研发资源，过度消费程序员对公司的满意度。周五晚上、休息日可以干扰研发人员的事情是：

- 现场问题，必须立即处理（不处理会损害公司未来利益）；
- 重要客户提出需求，要求立即做出回复（不处理会损害公司当前利益）；
- 特别重大的突发事件（对你、对公司都很重要）。

评估和改进绩效

作为团队管理者最重要的职责之一，是评估员工的工作绩效，并持续改进他们的绩效。每日反馈、季度/年度绩效审查，以及每月或者每季度目标等都是很好的巩固，可以帮助你完成绩效的评估和改进工作。

比较直接的方式是为每一个人设定工作目标并规定完成的时间。接着定期审查这些目标的进度和实现方式，季度目标的时间太长了，最好是每周、双周比较实际。我比较喜欢的方式是，采用邮件形式列出每个人今年的总目标，以及每周的细分目标，如下所示：

这样的邮件需要每周更新一次，保持时效性，也给团队管理者以约束，你必须和团队紧密地在一起，为员工设立清晰的目标，这样可以产生明确的绩效评估方法，而你在向上司或者公司其他人沟通汇报时就可以更加清晰。

裁员

表现很差的人给团队拖后腿的方式很多。他们占用了预算的一部分，但是却无法交付出有效的成果。其他人如果看到他们差劲的表现，也会消极得失去动力或者失去对你的尊重。表现

差的人可能会影响项目的进度，从而对项目中的每个人都产生不利影响。他们也会占用会议中的大量时间。因此不论如何，这种情况必须尽快解决。

虽然很难开口，但通常，终止合同对你和员工都是件好事。很多表现很差的员工都知道自己很差。很少有人会看不到现实情况，还幻想自己表现很出色。他们每天上班和睡觉都背负这这个沉重的负担。对大多数人来说，负担会沉重到难以忍受。所以，当你最终直面他们，并开始终止合同的流程时，员工通常会感到一种放下包袱的轻松。很多人会选择离开，也有一些人会选择尝试绩效改进，如果选择后一条路，你一定要定期与这个员工会面，审查绩效情况并讨论结果。

被动沟通

如果一名团队成员在你不太方便的时间来找你聊天，一定要先把手上的工作放下，专心跟他交流。他可能想鼓起勇气告诉你一件大事。聊天的内容可能很简单，比如缺少完成任务所需要的相关资料或者技能，也可能很重大，比如即将离婚、家人病重或者其他对他个人有很重大打击的事情，这些时间对你的工作时间表都有重大影响。如果你的团队成员知道自己受到了最高优先级待遇，那么在事情将变得很糟糕的时候，他们来找你交流的可能性就更大了。

影响团队因素

薪资管理

我对这一点有一个基本的方法论，概括为“对工作努力的人，要给予薪资方面的倾向，即便你是亿万富翁，该多给你的奖金一分都不能少，如果家庭经济有困难，那还要更多给予倾向。对于不努力的人，无论你多么困难，天平绝对不会向你倾斜”。公平、公正，是做事、做人的基本原则。

上升通道

团队里的每一个人都有自己的未来规划，你需要多沟通，了解清楚他们对于未来的设想，比如想走技术管理、纯管理、技术专家，或者想跳出研发体系，尽量为他们指出发展方向，让每个人能够清楚在这里可以走向未来，这样你的团队才会稳定。

工作时间

如果你可以做主，建议最好针对程序员采用弹性工作制。多数企业中的多数雇员属于白天型的人，与他们不同，多数程序员属于夜晚型的人。他们一般到了晚上才开始很有精神，对于开发工作更加专注，所以如果有可能，尽量多地给他们弹性，不要太过于限制工作的时间、

地点。当然，很多开发工作是需要及时沟通的，所以每天必须有大于4个小时的时间能够让大家聚在一起，这样才能够完成需求、设计等评审，才能对具体的系统架构、代码问题进行讨论，此外，毕竟产品、市场、职能部门的工作人员不是弹性工作制，也要体谅他们。

大规模变革

没有什么比建立新秩序更困难、更无望成功、更危险的了。因为旧秩序的受益者都是改革者的反对者，只有那些可能从新秩序受益的人才会勉强支持改革。

—— Niccolo Machiavelli

你一定要慎重执行大规模变革，无论是组织架构，还是人员任命，因为所有的改变，都会在人的内心引起权力欲望、恐惧，多做核心人物的思想沟通工作，不要由闭门小团体会议决定所有的改变方案，这样会让核心技术人员感觉到被边缘化、被决定命运。

其他事项

理解程序员

程序员是一种有趣的工作，而大多数程序员都很享受工作，这样就不难理解了，为什么难以管理他们？如果有人付钱让你开心地玩，你还会愿意受制于人吗？受人管制就会减少工作中的乐趣！

从许多方面看，程序员之间的差异非常大，只有很了解程序设计的人才能完全理解这一点，事实上，程序员之间的差异主要来自个人内在因素，而不是外在属性。大多数公司的高层管理者对所有的程序员一视同仁，这种看法是片面的。微软公司的Bill Gates、Adobe公司的John Warnock、FaceBook公司的Mark Zuckerberg都没有犯这样的错误，因为他们本质上也都是程序员。这也是我觉得某些大型软件企业需要变革的原因，在科技界，你最好不要让非技术出身的人担任CEO。

因为程序员都是些无拘无束的人，常见的激励方法往往没什么用。除了进行必要的技术监督并把开发实践和过程落实到位之外，善于利用程序员的自我意识和改变世界的欲望也很关键。这就需要一类既能理解程序员的工作方式，又能理解工作本身的技术管理者，他们不仅能有效地激励程序员超常发挥，而且能按时交付结果。

左脑型VS右脑型

右脑理论与左脑理论源于Roger W.Sperry的具体研究工作，根据他的研究表明，大脑的左半球和右半球具有针对不同任务的专门功能。左脑通常专用于分析任务和语言表达任务。左

脑的表达能力比右脑强得多，而右脑主要用于空间感知任务、音乐等。

如果你是一名程序员，你很可能属于“左脑型”，这意味着语言、逻辑和分析使用得更多，也更客观。其实称为“左脑为主型”更恰当，因为我只有一个大脑，两个半球始终是同时工作的。因此，你既可以是“左脑型”的人，又可以具有非语言交流、直觉、想象力较多且更主观等强烈的“右脑型”倾向，这些倾向通常更多地与音乐家、作家、艺术家等创新型人才相关联。

对于一名优秀的程序员来说，强大的左脑分析能力是必不可少的，不过，与右脑相关的活动往往也同样重要，这是因为程序设计是一门很有创意的艺术。事实上，一些最顶尖的程序员同时也是艺术家。

杰出的程序员

杰出的程序员是如何产生的呢？仅仅具备程序设计方面的天赋是远远不够的。实际上，天赋反而对杰出程序员的技能有副作用。杰出的程序员是大师级的人物，做事有条不紊、遵守纪律，能够凭借直觉把代码和程序组织好，能够约束自己总是在编写代码之前进行设计，能够在最少的时间内编写出清晰、简洁、实用、高质量的代码并获得预期的结果。换言之，杰出的程序员是大师级的工匠。

如果程序员的学习、工作动力主要来源于项目管理时间表、管理层的压力，或者金钱，那么它不会成为一名杰出的程序员。对大多数杰出的程序员来说，动力实际上来源于更高的追求，例如对改变世界，做出人们实际使用的程序或产品。杰出的程序员希望并且需要为具有世界影响的项目工作，他们希望能够感受到自己的工作是有意义的，哪怕只在某个很小的方面有意义也行。杰出的程序员偏爱能够满足他们更高理想或要求的公司和项目，他们非常在意自己所做的事情，常常为了想要的结果而超负荷工作，而不会在某种压力下自愿做低技术含量的重复劳动。

美国科罗拉多大学早在1993年就做过一个关于软件工程师的研究，报告显示：“和普通工程师相比，那些出类拔萃的工程师往往更能照顾全局，更喜欢实际心动，更易受使命感推动，更能展示和表达出一种坚定的信念，在管理中更容易发挥主动的作用，更能帮助其他工程师”。

世界上杰出的程序员不多，不可能让每个项目团队都拥有杰出的程序员。而且多数团队也只能容忍队伍中有一两名杰出的程序员。大多数的程序都需要靠普通程序员完成，他们通常是称职的、专业的、能干的，但是可能会把程序设计看作为一种工作，而不是追求。

阶段总结

管理程序员不是一项简单的任务，即便你当过程序员也不例外。本文我介绍了管理的基础知识，包括对岗位、经验、人员品质的要求，然后介绍了如何组建和管理团队，最后对一些影响程序员思维的知识进行了介绍。下一篇文章我会介绍如何进行项目管理，这是团队领导者的另一项基本任务。

研发管理相关课程推荐

StuQ 邀请拥有十余年技术实践，且有一线管理经验的麦克周老师，通过 3 周 Mini 直播课程形式，分享讲师积累的较为成熟的方法论和踏坑填坑经验，帮助学员成长为一名优秀的技术管理者，尤其在处理中小型团队的管理工作时，能更加合理高效，有的放矢。**前80人预售优惠中！（4月21日周五晚 20:00 课程公开预演）**

课程名称：《 如何 Lead 高效软件研发团队 》

主讲老师：麦克周，某知名 IT 技术公司技术高管、IBM 开发者论坛专栏作者、InfoQ 特约作者。

开班时间：5月5日晚 20：00

上课周期：3周

上课时间：每周五晚 20:00-21:00

学习形式：QQ学习交流群+直播视频授课

预售优惠价：799 元（原价 998 元，目前报名享受预售优惠）

使用优惠码预售优惠价再减 100 元等于 699 元！

课程优惠码：A0C4PLHLZW（仅可使用 10 次）

课程预演免费试听：在直播课程正式开始之前，我们邀请拥有十余年技术一线管理经验的麦克周老师，通过视频直播的方式，将会对于课程内容进行前期沟通，帮助大家更好地了解课程。

4月21日（周五）晚 20：00 邀你一起提升管理能力，解决工作实际问题。感兴趣的同学可扫描下方二维码进入课程交流群，详细了解课程详情，免费参与公开课预演。

点击「 阅读原文 」立即报名学习。

阅读原文