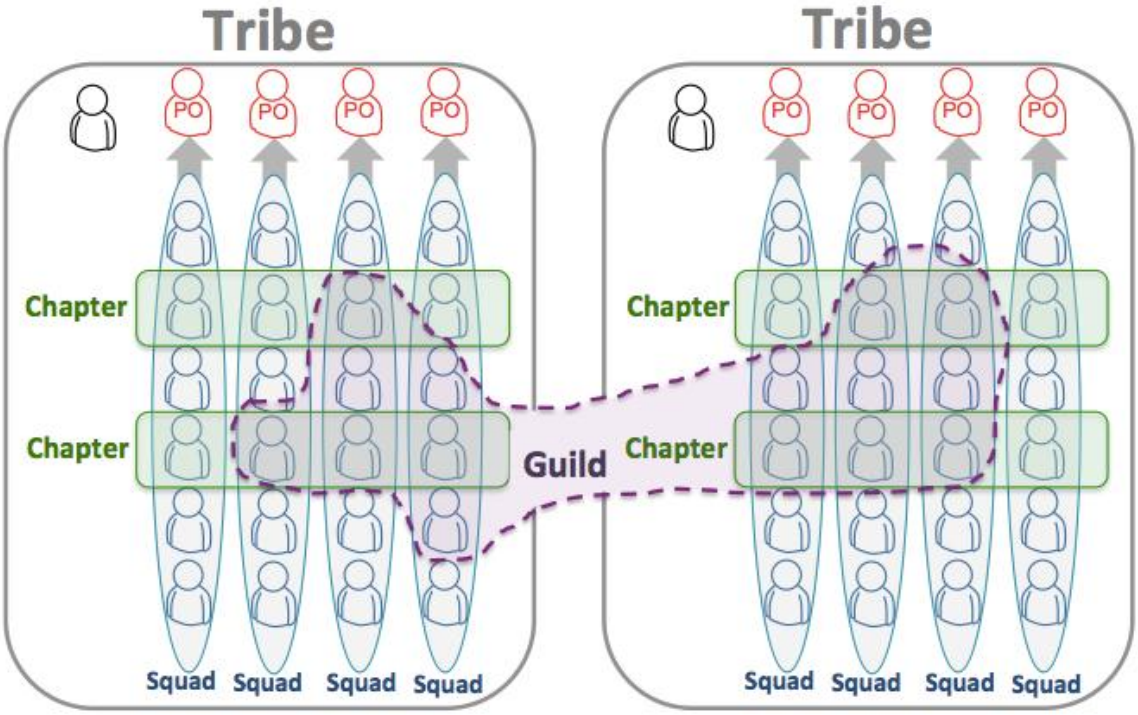
Spotify 的大规模敏捷之路

—使用一种新型的矩阵组织：部落、分队、分会和协会

作者：Henrik Kniberg, Anders Ivarsson

翻译：姜信宝(Bob Jiang)，程嘉利



多个团队一同从事产品开发真是一项挑战！

到目前为止，我们见过的最令人印象深刻的案例之一，就是 Spotify 公司：尽管 Spotify 拥有 30 多个团队，位于 3 个不同的城市，但他们仍然保持着敏捷开发的思维。

Spotify 是一家非常吸引人的公司，它改变了整个音乐产业。虽然这家公司从成立到现在只有 6 年，却已经拥有 1500 万活跃用户以及 400 万付费用户。该公司的产品可以被誉为是“一个神奇的音乐播放器，可以让你瞬间找到、播放世界上任何一首歌曲”。

Alistair Cockburn（敏捷软件开发创始人之一）在参观 Spotify 时曾说道：“太棒了！我从 1992 年开始就一直希望有人能够实现这套矩阵式组织结构的设计（参考上图），很高兴今天我看到了”。

那么这一切是怎么做到的呢？

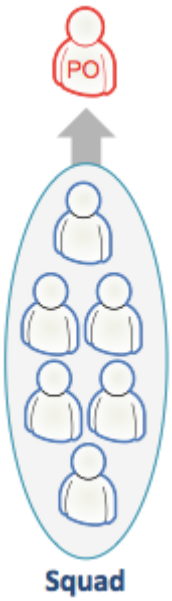
我们俩在会议演讲中、在各种讨论中多次提到我们在 Spotify 如何工作、这家公司如何在数百名开发人员中运用敏捷，很多人对这个话题很感兴趣，所以我们决定以此为题写一篇文章。

声明：

- 文中提到的模型不是我们发明的。
- Spotify，和其他优秀的敏捷型公司一样，处于高速地发展中。这篇文章只是我们**当前**工作方式的缩影。我们的敏捷旅程仍在进行中，并未结束。我们估计，当您读到这篇文章的时候，很多事情已经发生变化了。

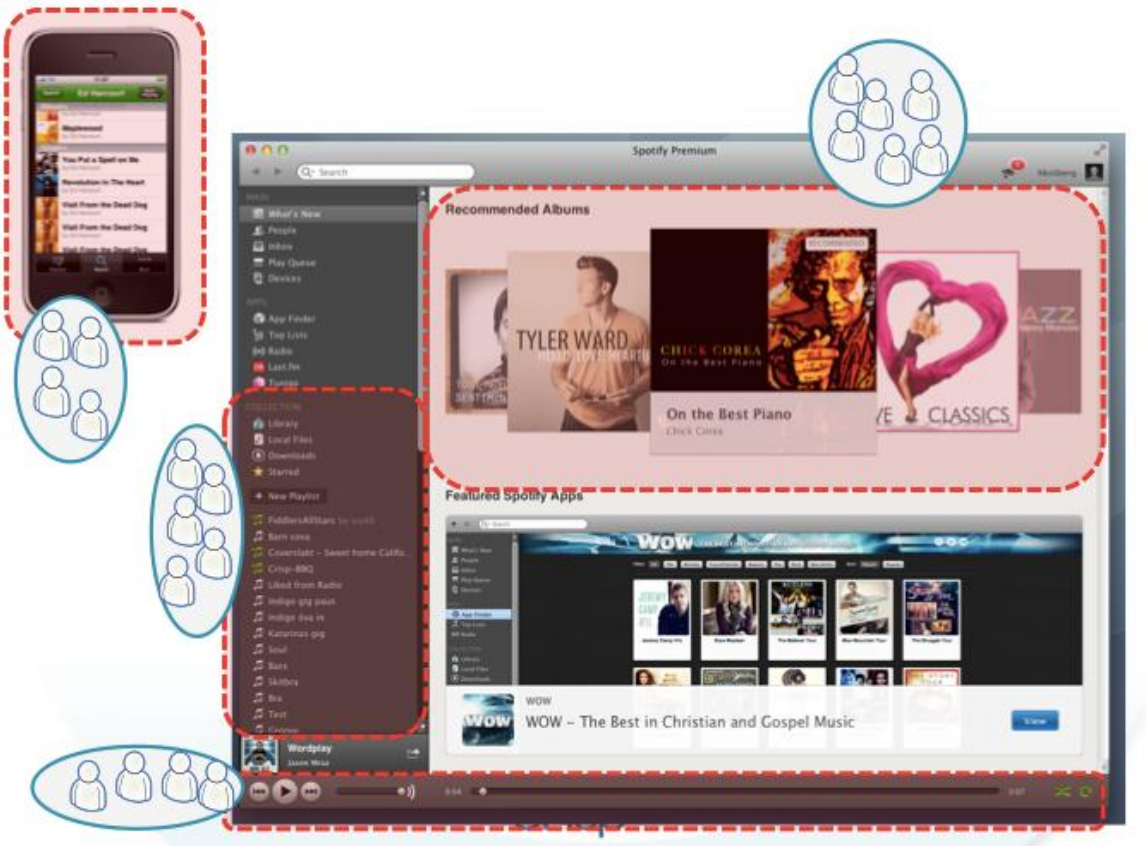
分队（squads）

分队是 Spotify 的最小开发单位。



一个分队类似于一个 Scrum 团队，也很像一家迷你型创业公司。分队中的成员坐在一起，他们具备设计、开发、测试和产品发布所必需的全部技能和工具。一个分队是一个自组织团队、决定自己的工作方式——有的分队使用 Scrum 中的迭代（sprint），有的分队使用看板（Kanban），还有的综合使用上述方法。

每个分队都会有一个长期的使命，比如：开发和优化 Android 客户端、打造 Spotify 广播功能的用户体验、扩展后台系统、提供支付解决方案，等等。如下图所示，不同分队负责用户体验的不同部分。



Spotify 鼓励每个分队都运用精益创业原则，比如 MVP（Minimum Viable Product，最小可行产品）和验证性学习（validated learning）等。MVP 意味着尽早地、频繁地发布； 验证性学习意味着使用度量（metrics）和 A/B 测试（A/B tesing）来确认什么可行，什么不行。用一条标语来总结的话，就是：“思考、构建、交付、调整（Think it, build it, ship it, tweak it）”。

由于一个分队长期持续地从事某一类任务以及开发产品的某一个部分，所以分队成员逐渐都成为了该领域的专家——比如，如何打造非常棒的广播体验。

大部分的分队都拥有非常棒的工作环境：办公区、休息区、个人杂物室、几乎所有的墙都是白板……，我们从未见过比这儿更好的工作空间！

没错，这是一只飞来飞去的玩具鲨鱼，这种东西在 Spotify 很常见。

为了激励学习和创新，公司鼓励每个分队把大概 10%的工作时间用在“黑客日（hack days）”上。在黑客日期间，大伙可以做任何自己想做的事情：通常大家会尝试一些新想法并和伙计们分享。有的团队每两周举办一天黑客日；有的团队则攒够了日子，搞一个“黑客周”。黑客日不仅有趣，还是一个让大家紧跟新工具、新技术的好途径，有时候非常重要的产品创新也诞生于黑客日！

一个分队没有正式任命的领导，但是会有一个产品负责人（Product Owner）。产品负责人负责把团队的待办任务进行优先级排序，但从不干涉团队如何完成这些任务。不同分队的产品负责人紧密合作，共同维护一个宏观层面上的产品路线图（roadmap）文档，指引整个 Spotify 产品发展方向；与此同时，每个产品负责人也各自维护一个自己所在分队的产品待办项列表（product backlog）。

每个分队也可以有一位敏捷教练，帮助团队改进工作方式。敏捷教练负责主持回顾会议、组织 sprint 计划会议、做一对一辅导……，等等。

理想情况下，每个分队是一个高度自治的“迷你型创业公司”，他们可以和利益相关者直接对话，且和其他分队没有阻塞型依赖关系。对于一个拥有 30 多个团队的公司，想要达到这样的状态，绝对是个挑战！我们虽然已经取得了很大的进展，但是仍有许多改进工作要开展。

为此，我们每个季度会对每个分队进行一次调查，帮助我们聚焦于需要改善哪些地方以及了解到每个分队需要哪些组织层面上的支持。下图是我们某次对一个部落中的 5 个分队的调查结果。

Area	Squad 1	Squad 2	Squad 3	Squad 4	Squad 5
Product owner					
Agile coach					
Influencing work					
Easy to release					
Process that fits team					
A mission					
Org. support					

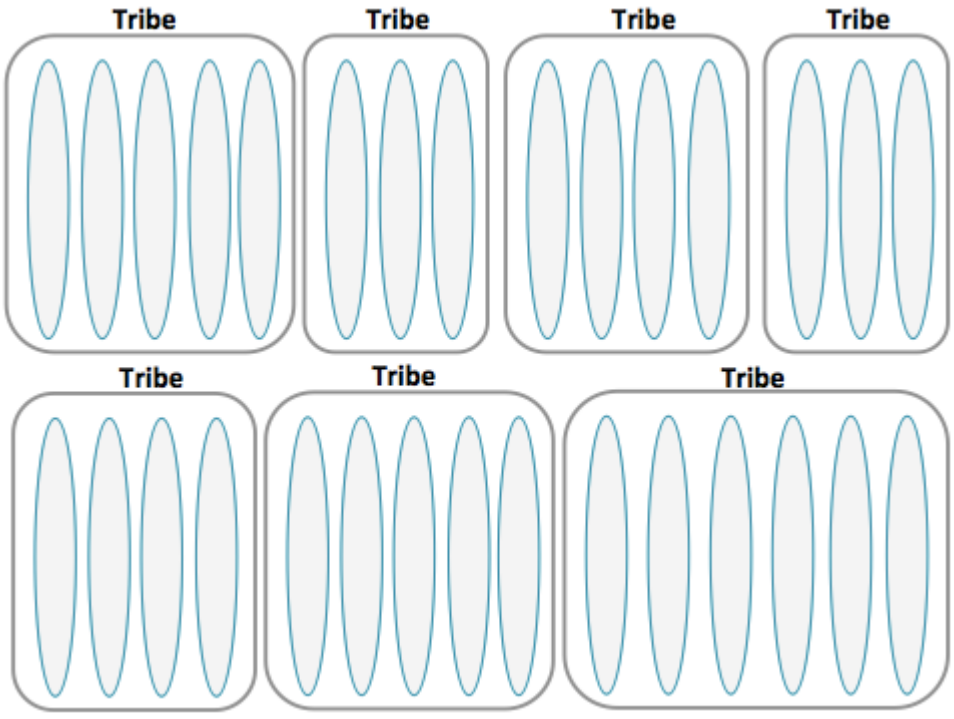
圆圈表示当前状态，箭头表示趋势。比如，我们可以看到这样一个情况：3 个分队报告了发布方面的问题，但是问题似乎并没有得到改善——这方面需要我们的紧急关注！ 我们还可以看到第 4 分队没有很好地得到敏捷教练的支持，但是这个问题已经在改善中。

以下是各个调查项的评判参考标准：

- **产品负责人（Product owner）** – 分队内有专职的产品负责人对任务的优先级进行排序；排序时，产品负责人能够综合考虑商业价值和技术因素。
- **敏捷教练（Agile coach）** – 分队有一位敏捷教练帮助团队识别障碍、指导团队持续进行过程改进。
- **支配自己的工作（Influencing work）** – 分队内的每个成员都可以支配自己的工作、可以积极参与工作计划的制订、可以选择自己做什么任务。每个成员都可以把自己 10%的工作时间投入到黑客日中。
- **易发布（Easy to release）** – 分队可以(并且确实做到！)轻松发布产品，而不需要很多的争论和同步。
- **量身定制的流程（Process that fits the team）** – 分队拥有自己的工作流程并且持续对其进行改进。
- **使命（Mission）** – 分队有一个队内所有人都知道并关心的使命；待办项列表中的故事都是和这个使命相关的。
- **组织层面的支持（Organizational support）** – 分队知道去哪里寻求解决问题所需的支持，无论是技术问题还是“软性问题”（“soft” issues）。

部落（tribes）

部落是多个工作在相关领域的分队的集合——比如音乐播放器、后台基础架构等。



部落可以看作是迷你型创业分队的“孵化器”，每个部落都非常地自主自治。每个部落有一名酋长，他负责为部落内的各分队提供最好的栖息地（habitat）。一个部落中的所有分

队在同一个办公地点工作，通常各分队的办公区都是彼此相邻的，办公区附近的休息区促进了分队间的交流与合作。

部落规模的确定是基于“邓巴数（Dunbar number）[1]理论”的。“邓巴数理论”认为在超过 100 人的组织中，大部分人很难维持稳固的社会关系（信不信由你，对处于强烈生存压力下的组织来说，“邓巴数”实际上要比 100 大。Spotify 并不属于这样的情况）。当一个组织变得过大后，我们就会开始看到限制性的规定、官僚主义、政治斗争、冗余的管理层级，以及其他各种浪费。

所以 Spotify 的每个部落都小于 100 人。

部落内会定期举办非正式的聚会，大家会在聚会上给部落中的其他人（以及任何出席聚会的人）展示自己正在做什么、已经交付了什么、别人能从自己正在做的事情中吸取到什么经验或教训。展示内容包括可工作的软件、新工具与新技术、酷毙了的黑客日项目……。

分队间依赖

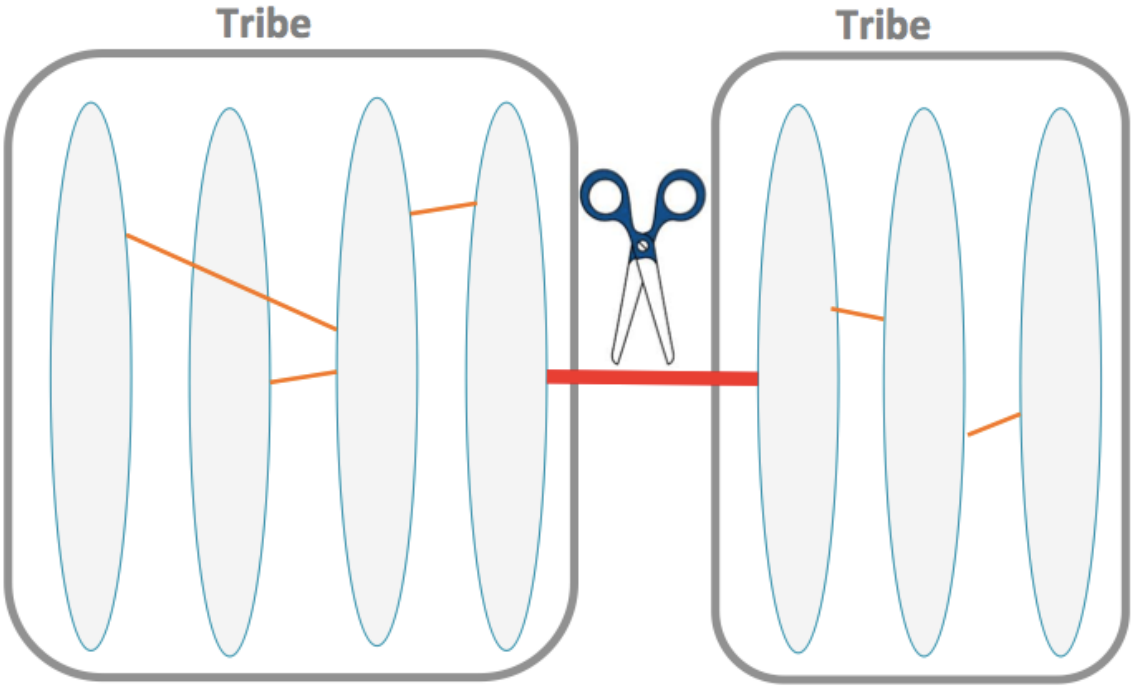
多个分队之间必然会有依赖关系。有依赖关系并不一定是坏事——分队间有时确实需要一起工作才能完成真正超棒的产品特性。然而，我们的目标是让分队间越独立自治越好，尤其是要把阻塞或拖慢了某个分队工作的依赖项减到最少。

为此，我们经常对所有的分队进行调查：你们的工作依赖于哪些分队？这些依赖是否阻塞或拖慢了你们的工作？严重到了什么程度？下面是一个例子：

[1] 邓巴数，也称 150 定律，是指能与某个人维持紧密人际关系的人数上限，通常人们认为是 150。http://zh.wikipedia.org/wiki/%E9%82%93%E5%B7%B4%E6%95%B0

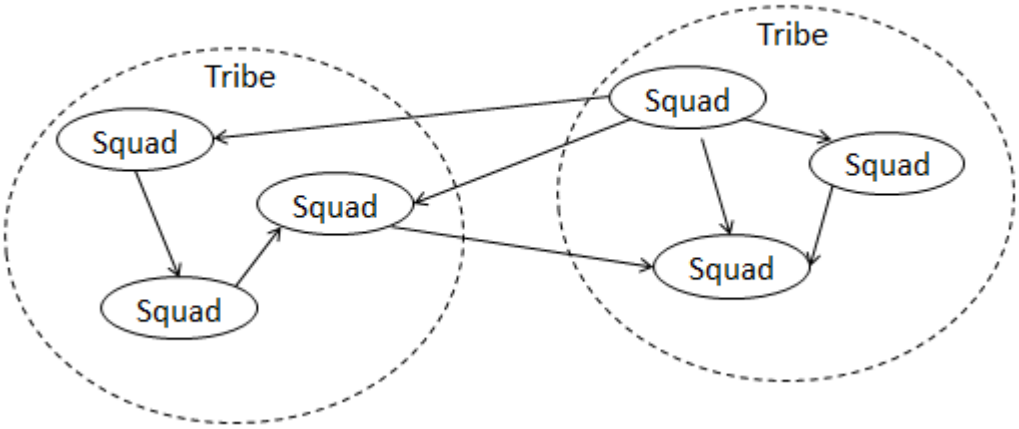
	A	B	C	D	E
1	**Squad**	**Depends on**	**Dependency**	**Comment**	**Same tribe?**
2	**Music Player**				
3	Content	Ops	Slowing	Need machines, connections, help set-up things etc. Works really well in general, but at times the workload on operations causes the lead times to grow and slow us down	No
4	Content	NeXT	No problem	Storage. Not big, mostly information/communication needs to happen.	No
5	Content	BFS	No problem	Replacement service	Yes
6	Content	Team 2	No problem	Communication around next story	No
7	Content	Team 1	Future	Content ingestion	No
8	BFS	UX	Slowing	Need UX to discuss, review and provide mock-ups.	No
9	BFS	Content	No problem	Normal dependencies, sprint work.	Yes
10	BFS	Mobile	Slowing	No internal mobile developers within Squad.	No
11	BFS	Analytics	Slowing	A/B test results slowing down roll outs of features	No
12	BFS	Team 3	Blocking	Waiting for data dumps	No
13	BFS	Team 1	Future	Waiting for data dumps	No
14					

有了调查结果，接下来我们就会讨论如何消除有问题的依赖，特别是引起了工作阻塞的以及跨部落的依赖关系。为了消除这些有问题的依赖，我们经常会重排任务优先级、重组团队、调整架构或技术方案。



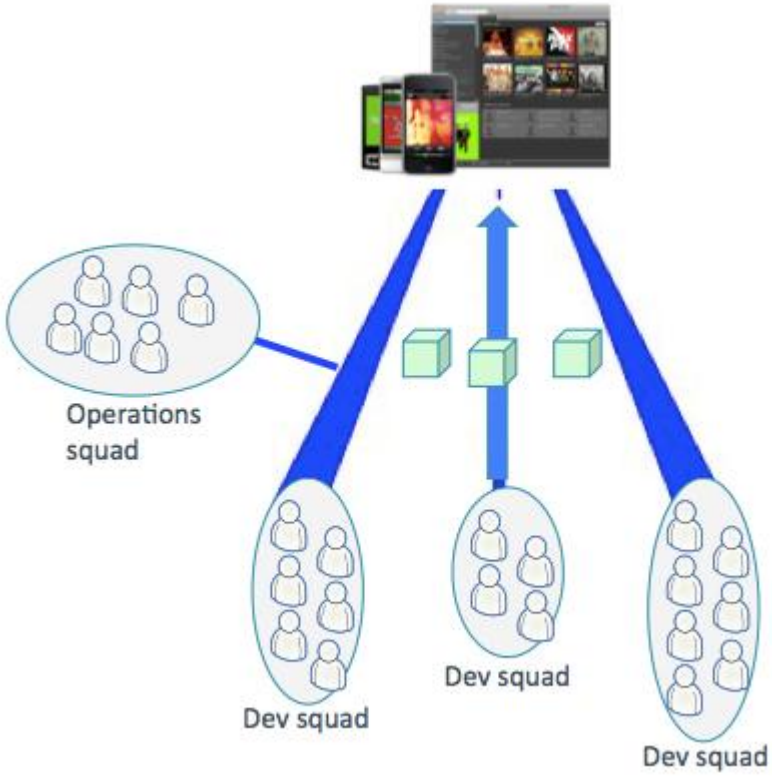
这种调查还会帮助我们观察到分队间互相依赖的模式，比如，越来越多的分队看起来是被运营拖慢了。我们利用非常简单的图表来跟踪各种类型的依赖是如何随着时间增加或减少的。

Scurm 中有一个实践叫“scrum of scrums”——这是一个同步会议，会议由每个 scrum 团队出一名代表参加，大家在会上讨论团队间的相互依赖。 在 Spotify，通常我们并不怎么进行 scrum of scrums，主要原因是大部分的分队都相当的独立，他们并不需要这样的协调会议。



然而，如果有必要，我们也会“按需”进行 scrum of scrums。比如：最近我们有一个大型项目，需要多个分队协同工作几个月。为了更好的合作，分队间每天会开一个同步会议，会议上大家一起去识别和解决分队间的依赖关系，并使用白板和记事贴来跟踪尚未解决的依赖。

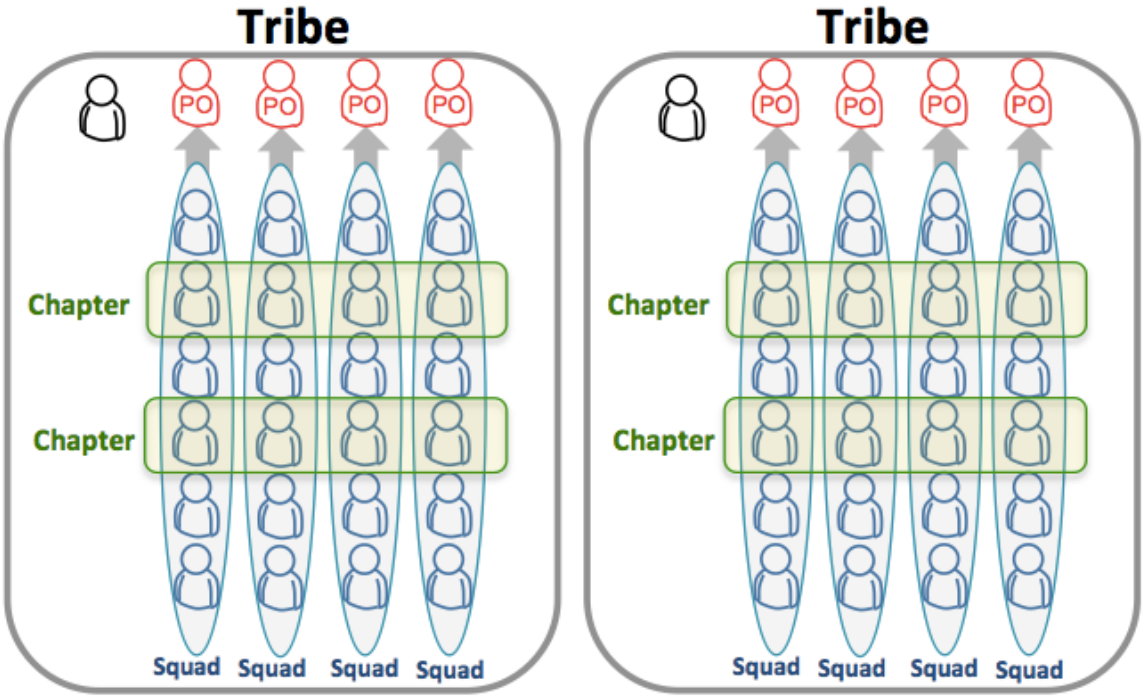
许多公司中依赖问题的常见根源存在于开发（dev）和运营（ops）之间。大多数公司都存在从开发到运营的移交，期间伴随着一些摩擦和延误。在 Spotify 也有一支独立的运营团队，但是他们的工作不是为分队进行发布——而是为分队自己发布代码提供支持；支持可能是以下形式：基础架构、脚本或程序。在某种意义上，运营团队是“为产品铺路”。



这是非正式但有效的协作方式，基于面对面的沟通而不是详细的流程文档。

分会（chapter）和协会（guild）

任何事物都有缺点，充分自主的可能缺点是损失了规模经济的效益。分队 A 的测试人员碰到的问题可能和分队 B 的测试人员上周刚刚解决的一样。如果所有的测试人员可以凑在一起，跨越分队和部落的话，那么他们就可以分享知识以及创建对所有分队都有益处的工具。如果每个分队充分自主，且互相之间没有沟通的话，那么为什么还需要有公司呢？干脆把 Spotify 拆成 30 个不同的小公司好啦。这就是我们为什么还会有分会和协会。分会和协会使公司团结在一起，在不牺牲太多自主权的情况下，也为我们带来一定的规模经济。分会是在同一个部落、相同能力领域内拥有相似技能的一些人。

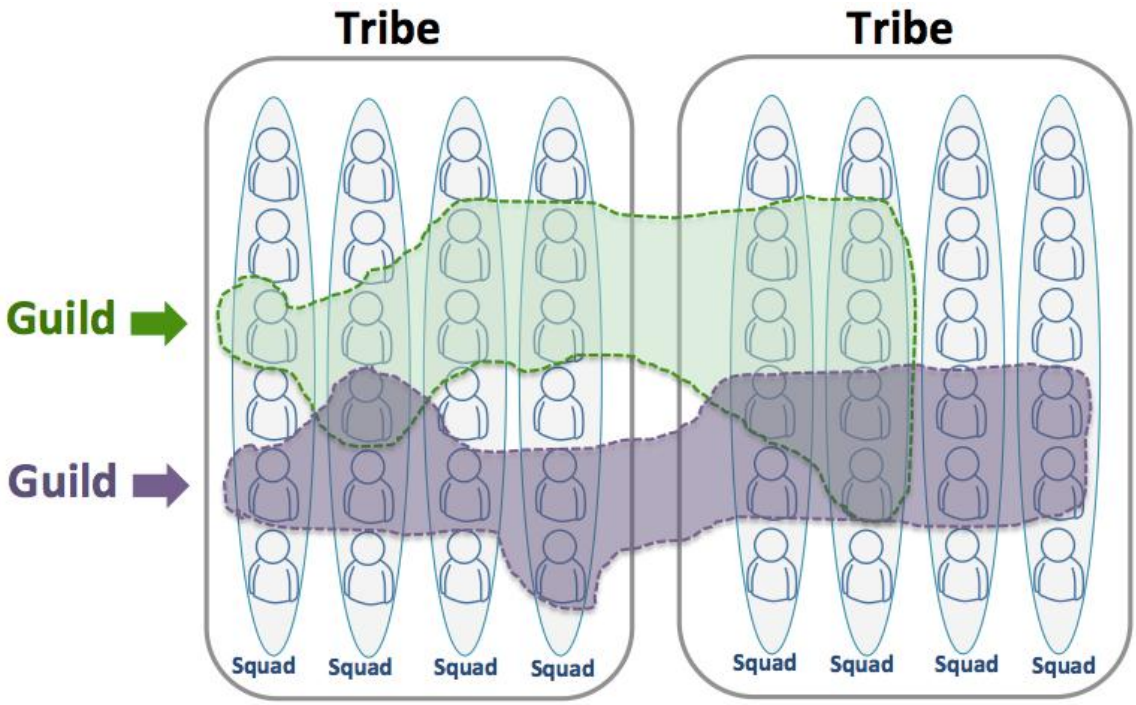


每个分会定期凑在一起讨论专业领域知识及他们遇到的挑战——比如测试分会，网页开发分会或者后台分会。

分会领导是该分会成员的直线经理，和传统的直线经理一样，他们的职责是发展员工、设定薪水等等。但是，分会领导也是分队的一份子，也要参与日常工作，这样才不会和实际情况脱节。

但现实情况总是比上图更加混乱。比如，分队之间的分会成员不是均匀分布的；有的分会有很多网页开发人员，有的分会一个也没有。上图只是一般的思路。

协会则是一个具有更广泛影响的“兴趣社区”，它包含这样一群人，他们想要分享知识、工具、代码和实践。分会是在部落内的，而协会通常跨越整个组织。比如，网页技术协会，测试协会，敏捷教练协会等等。



协会包含所有相关领域的分会成员，比如测试协会包含所有测试分会的成员，不过每个对协会感兴趣的员工都可以加入其中。

每个协会都有一个“协会协调人”，他也就负责协调而已 :o)

说一个协会工作的例子，最近我们举办了一场“网页协会非会议[2]”，这是一次开放空间活动，Spotify 的所有网页开发人员聚到斯德哥尔摩来讨论他们领域内的挑战和解决方案。

[2] 非会议是一种议程由参与者推动并创建的会议。会议的筹备一般不是由某个单独机构或一小群机构组织的。http://zh.wikipedia.org/wiki/%E9%9D%9E%E4%BC%9A%E8%AE%AE

另一个例子是敏捷教练协会。敏捷教练遍布在整个组织，但是他们持续地分享知识，并且定期聚会讨论组织层面改进的工作，这些都记录在一个改进白板上。（如下图）

等等，这不就是矩阵组织吗？

没错，这是矩阵组织。但也不完全是。这个和我们常用的矩阵结构是不同类型的。

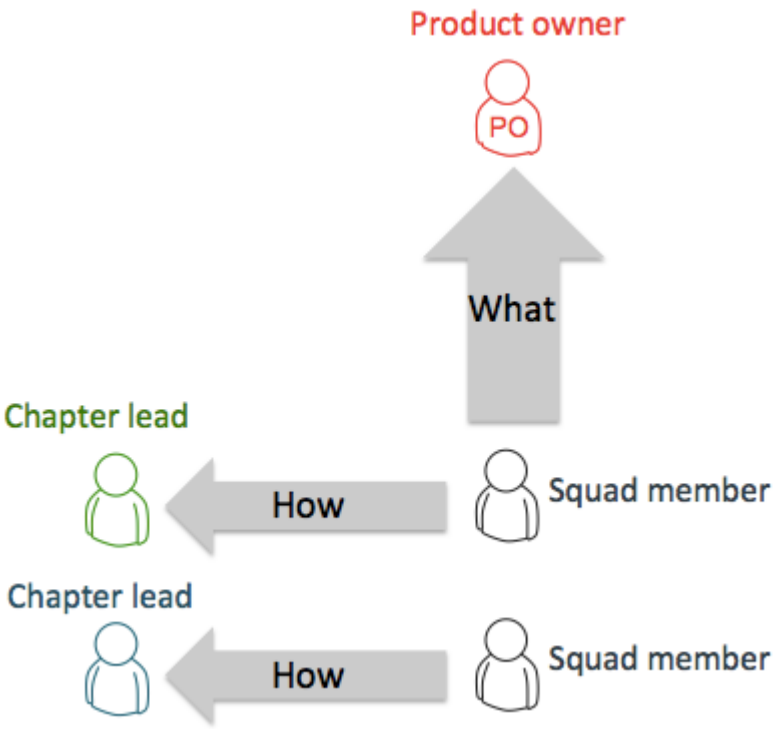
大多数矩阵组织中，相似技能的人“扎堆儿”到一起形成职能部门，接着“被分配”到项目中，并且“汇报”给职能经理。

Spotify 不是这样的。这里的矩阵偏重于交付。

在 Spotify 员工分成稳定的、坐在一起的分队，拥有不同技能的人一起协作和自组织以交付一个优秀的产品。这是矩阵里的垂直维度，也是员工分组工作的主要维度。

水平维度是有关分享知识、工具和代码的。分会领导的职责就是促进和支持这些工作。

在矩阵里，可以认为垂直维度是“做什么（what）”，水平维度是“怎么做（how）”。矩阵结构确保每个分队成员可以领会“下一步做什么”以及“如何做好”。



这也符合 Mary Poppendieck 和 Tom Poppendieck 建议的“教授和企业家（professor and entrepreneur）”模型。产品负责人（PO）就是“企业家”，关注于交付优秀的产品，而分会领导是“教授”，关注于技术卓越。

这两个角色之间存在良性的紧张关系，因为企业家倾向于快速交付和走捷径，然而教授倾向于慢下来和把事情做对。两个方面都是必须的，因此我们说这是良性的紧张关系。

来谈谈系统架构如何？

Spotify 的技术是高度面向服务的。我们有 100 多个独立的系统，每个系统都可以单独地维护和部署。这些系统包含后台服务如播放列表管理、搜索和支付，还包含客户端如 iPad 播放器，也包含特定的组件如广播、或音乐播放器中的“最新动态”。

从技术上讲，每个人可以修改任意一个系统。既然分队是高效的特性团队，那么通常团队需要更新多个系统来完成新特性。

这个模型的风险是，如果没人关注系统整体一致性的话，那么系统架构就会变得一团糟。

为了降低这个风险，我们有一个“系统负责人（System Owner）”的角色。每个系统都有一个或一对系统负责人（我们鼓励结对）。对有些关键运营系统，系统负责人是开发和运营结对的（Dev-Ops pair）—— 一个人有开发人员的视角，另一个人有运营人员的视角。

系统负责人是系统技术或架构问题的“关键先生”。系统负责人负责协调和指导开发人员，以避免开发人员之间的冲突。系统负责人关注于以下事情：质量、文档、技术债、稳定性、可扩展性和发布流程。

系统负责人不应该是瓶颈或者与世隔绝（原文是象牙塔里的架构师）。系统负责人不必一个人完成所有的决策、代码或发布。通常系统负责人是分队成员或分会领导，除了系统负责人的工作之外，他还有其他的日常职责。然而，系统负责人会时不时地进行一个“系统

负责人日”以清扫一下系统。通常我们尽量使清扫系统的工作占用系统负责人少于十分之一的时间，但是系统之间有很大的不同。

我们还有一个首席架构师的角色，他负责协调跨越多个系统的、较高层面的架构问题。他还会评审新系统的开发工作，以避免一些常见错误，并确保新系统和现有的架构设计是一致的。架构师的反馈只是建议和输入——系统设计的最终决策取决于分队。

所有这些是怎么办到的？

Spotify 发展的非常迅速——过去 3 年技术人员从 30 人增长到 250 人——因此我们也有成长的烦恼！本文介绍的大规模敏捷模式（分队、部落、分会和协会）也是去年（2011）逐步引入的，所以员工还在慢慢适应。但到目前为止，从调查和回顾来看，这个模式似乎相当棒！也给了我们一套继续成长的蓝图。尽管人员增长迅速，但员工满意度还是持续地增长；2012 年 4 月得分是 4.4（满分 5 分）。

然而，和每一个不断发展的组织一样，今天的解决方案会产生明天的问题。所以敬请关注，故事还没有结束 :o)

/Henrik & Anders
henrik.kniberg@spotify.com
www.crisp.se/henrik.kniberg
anders.ivarsson@spotify.com

原文链接：Scaling Agile @ Spotify with Tribes, Squads, Chapters & Guilds

译者

姜信宝（Bob Jiang）http://bobjiang.com
喜欢新鲜事物，喜欢读书，喜欢分享，愿意和大家共同进步。Agile1001 公开课联合创始人。热衷和推广敏捷，是北京敏捷社区的主要推动者之一。欢迎订阅我维护的微信公众号-敏捷那些事儿（AgilePlus）。

程嘉利，曾就职于华为、炬力、NXP，现在在腾讯做过程改进。CSM，热爱敏捷开发。

术语表（Glossary）

Squads – 分队

Chapter – 分会

Guild – 协会

Tribe – 部落

System Owner – 系统负责人

Product Owner – 产品负责人

Dunbar number – 邓巴数

Unconference – 非会议

MVP (Minimal Viable Product) – 最小可行产品

Validated Learning – 验证性学习

Hack Days – 黑客日