

# 技术雷达

由 ThoughtWorks 技术顾问委员会编写

[thoughtworks.com/radar](http://thoughtworks.com/radar)

2013 年 5 月

**ThoughtWorks®**

# 什么是新的？

## 以下是本期重点介绍的趋势：

- **拥抱下降边界**——不管你喜不喜欢，你周围的界限正在消失。我们选择通过检查无边界企业、云中的开发环境和远程呈现的协同定位等概念来接受这一点。
- **将行之有效的做法应用于不知何故遗漏的领域**——我们不确定为什么，但我们行业中的许多人都错过了捕获客户端 JavaScript 错误、移动设备持续交付、NoSQL 数据库迁移和 CSS 框架等想法。
- **轻量级分析选项**——数据科学和分析不仅仅适用于拥有该领域博士学位的人。我们强调协作分析和数据科学，所有开发人员都了解基础知识并在必要时与专家密切合作。
- **基础架构即代码**——持续交付和 DevOps 提升了对基础架构的思考。将基础架构视为代码的含义以及对新工具的需求仍在不断发展。

ThoughtWorkers 对技术充满热情。我们构建它、研究它、测试它、开源它、编写它，并不断致力于改进它——为了每个人。我们的使命是支持卓越软件并彻底改变 IT。我们创建并分享 ThoughtWorks 技术雷达以支持该使命。ThoughtWorks 技术顾问委员会是 ThoughtWorks 的一群高级技术领导者，他们创建了雷达。他们定期开会讨论全球技术战略。

ThoughtWorks 和对我们行业产生重大影响的技术趋势。

该雷达以一种为从首席信息官到开发人员的广泛利益相关者提供价值的格式捕获技术咨询委员会讨论的输出。内容旨在作为简明摘要。我们鼓励您探索这些技术以获取更多详细信息。雷达本质上是图形化的，将项目分组为技术、工具、平台以及语言和框架。当雷达项目可能出现在多个象限时，我们选择了一个看起来最合适的。

我们进一步将这些项目分为四个环，以反映我们目前对它们的立场。戒指是：

- **采纳**：我们强烈认为该行业应该采用这些项目。我们在适当的时候在我们的项目中使用它们。
- **审判**：值得追求。了解如何建立这种能力很重要。企业应该在可以处理风险的项目上尝试这项技术。
- **评估**：值得探索，目的是了解它将如何影响您的企业。
- **抓住**：谨慎行事。

自上次雷达以来新的或有重大变化的项目用三角形表示，(▲) 而未移动的项目表示为圆圈 (○)。每个象限的详细图表显示了项目的移动情况。我们感兴趣的项目远远超过我们可以合理地放入这种大小的文档中的项目，因此我们从最后一个雷达中淡化许多项目以为新项目腾出空间。褪色并不意味着我们不再关心它。

有关雷达的更多背景信息，请参阅<http://martinfowler.com/articles/radar-faq.html>

## 贡献者 - ThoughtWorks 技术顾问委员会由以下人员组成：

丽贝卡·帕森斯(首席技术官)  
马丁·福勒  
(首席科学家)  
巴德里·贾纳基拉曼  
达伦·史密斯

埃里克·多恩伯格  
埃文·博彻  
许浩  
伊恩·卡特赖特  
詹姆斯刘易斯

杰夫·诺里斯  
迈克梅森  
尼尔·福特  
雷切尔莱科克  
罗纳尔多费拉兹

山姆·纽曼  
斯科特·肖  
斯里哈里·斯里尼瓦桑  
蒂亚古帕拉尼萨米

# 雷达

## 技巧

### 采纳

- 1个 聚合为文档 自动部署管道 Guerrilla
- 2个 测试
- 3个
- 4个 进程内验收测试 移动网络上的移动测试 作为一
- 5个 等公民的性能测试 异步编程的承诺 Windows 基
- 6个 基础设施自动化

7

8个

### 审判

- 9 分析测试运行
- 10 蓝绿部署
- 11 网真托管 移动设备持续交付 NoSQL 数据库
- 12 迁移
- 13
- 14 Edge Side Includes 用于页面组合 HTML5 存
- 15 储而不是 cookie
- 16 记录为数据
- 17 微服务
- 18 移动优先
- 19 无边界企业响应式网页设计
- 20
- 21 语义监控

### 评估

- 22 捕获客户端 JavaScript 错误 协同分析和数据
- 23 科学 云中的开发环境 关注平均恢复时间
- 24
- 25
- 26 机器映像作为构建工件最小化应用程序配
- 27 置

### 抓住

- 28 详尽的基于浏览器的测试

## 平台

### 采纳

- 29 弹性搜索
- 30 数据库
- 31 Neo4J
- 32 雷迪斯
- 33 SMS 和 USSD 作为 UI

### 审判

- 34 大查询
- 35 在云 Couchbase 中持续集成
- 36
- 37 Hadoop 2.0
- 38 节点.js
- 39 开放堆栈
- 40 机架云
- 41 里亚克

### 评估

- 42 蔚蓝
- 43 卡拉特拉瓦
- 44 原子的
- 45 用于 NoSQL 的 PhoneGap/
- 46 Apache Cordova PostgreSQL
- 47 乌米
- 48岁 Zepto.js

### 抓住

- 49 大型企业解决方案单例基础
- 50 设施 WS-\*
- 51



# 雷达



## 工具

### 采纳

- 52 D3
- 53 嵌入式 servlet 容器 Frank
- 54
- 55 摇篮
- 56 石墨
- 57 不可变服务器
- 58 NuGet
- 59 清酒

### 审判

- 60 阿帕奇猪
- 61 加特林
- 62 杰基尔
- 63 刺槐
- 64 Logstash 和 Graylog2
- 65 幻影JS
- 66 Puppet-图书管理员和 Chef-图书管理员
- 67 TestFlight & HockeyApp

### 评估

- 68 基于浏览器的模板法拉第
- 69
- 70 Hystrix
- 71 图标字体
- 72 灯台
- 73 章鱼
- 74 .Net Riemann 的响应式扩展
- 75
- 76 扫雪机分析
- 77 UIAutomator

### 抓住

- 78 重量级测试工具
- 79 行家
- 80 交通運輸服务系统

## 语言和框架

### 采纳

- 81 Clojure 语言
- 82 CSS框架
- 83 Jasmine 与 Node.js Scala 配对
- 84
- 85 西纳特拉

### 审判

- 86 咖啡脚本
- 87 下拉精灵
- 88 用于离线应用程序的 HTML5
- 89 JavaScript 作为平台
- 90 后JavaScript MV\* 框架 Play
- 91 Framework 2
- 92 Require.js 和 NPM
- 93 Scratch、Alice 和 Kodu

### 评估

- 94 Clojure脚本
- 95 格林姆林
- 96 Lua
- 97 南希
- 98 欧文
- 99 红宝石运动
- 100 个推特引导程序

### 抓住

- 101 骨干.js
- 102 基于组件的框架 103 手写CSS

- 104 存储过程中的逻辑

# 技巧

多年来，团队和组织已经看到围绕技术学科孤立专业知识的危险。虽然我们重视高级应用程序专家的意见，但开发人员应该具备用户界面、数据库和数据科学的基本知识，这是最新的行业宠儿。虽然高级应用程序需要深厚的专业知识，但我们正在推动 **协作分析和数据科学**，所有开发人员都使用基本的统计分析和工具来做出更好的决策，并在事情变得复杂时与专家密切合作。

技术趋势打破了过去环绕企业 IT 网络的花园围墙，并导致 **无边界企业**。员工经常使用自己的消费设备通过云服务 and Web API 访问公司数据，而且通常是在组织不知情的情况下。随着设备的不断激增和更多应用程序迁移到云端，企业被迫重新考虑有关数据访问和网络安全的基本假设。

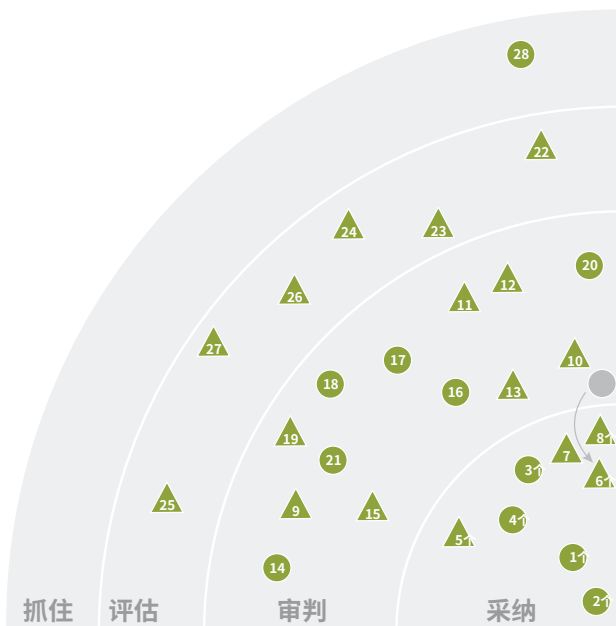
**云中的开发环境** 允许您完全外包开发基础架构，让您的团队只剩下笔记本电脑和互联网连接。通过结合使用私有 GitHub 存储库和 Snap CI 在云中的持续集成等同类最佳服务，您的团队可能永远不需要再为基础设施而费心内部 IT。

廉价或免费视频会议的质量和选择范围的增加正在为分布式团队带来一种新的工作方式。永远在线的视频连接有助于营造一种 **网真协同定位**，即使团队分布在不同的地理位置。这正在成为我们一些离岸交付中心的事实标准。我们还看到越来越多地使用 ScreenHero 等屏幕共享工具进行远程配对。我们会提醒那些寻找灵丹妙药来消除物理托管需求的人。面对面交流所产生的理解和同理心是无可替代的。

在开始使用新工具、管理不同环境的部署或试图了解为什么应用程序在不同地方表现不同时，应用程序配置可能是一个痛苦的来源。我们是最小化的忠实拥护者 **应用配置**，试图确保应用程序以最少的配置开箱即用。

大多数虚拟化技术都提供了一种从映像启动机器的方法。通过创建一个 **机器映像作为构建工件** 在构建管道的早期并在它通过进一步的测试套件时通过管道对其进行推广，您可以可靠地将通过测试的确切机器部署到生产中。这种技术消除了雪花服务器反模式的大多数原因。

**蓝绿部署** 是一种用于执行软件升级的模式。通过在生产应用程序堆栈的相同克隆上设置最新版本的应用程序，只要测试套件和业务确定合适，流量就可以几乎立即从当前生产堆栈切换到新生产堆栈。尽管这是一项古老的技术，但云中的基础设施自动化和资源使其值得重新考虑。



## 采纳

- 1个 聚合为文档 自动部署管道 Guerrilla
- 2个 测试
- 3个
- 4个 过程中验收测试 移动网络上的移动测试
- 5个 作为一等公民的性能测试
- 6个
- 7 异步编程的承诺
- 8个 Windows 基础架构自动化

## 审判

- 9 分析测试运行
- 10 蓝绿部署
- 11 网真托管 移动设备的持续交付 NoSQL
- 12 Edge 的数据库迁移 Side Includes for
- 13 page
- 14 作品
- 15 HTML5 存储而不是 cookie 记录为数
- 16 据
- 17 微服务

## 评估

- 18 移动优先
- 19 无边界企业响应式网页设计
- 20
- 21 语义监控
- 22 捕获客户端 JavaScript 错误 协同分析和数
- 23 据科学 云中的开发环境 关注平均恢复时间
- 24
- 25

## 抓住

- 26 机器映像作为构建工件最小化应用程序配
- 27 置
- 28 详尽的基于浏览器的测试

# 技巧

以前, Chef 和 Puppet 等工具缺乏对 Windows 的支持, 导致需要大量的 Powershell 脚本来实现简单的基础设施自动化任务。在 Windows 上实现相同级别的自动化比在 Unix 上更具挑战性。然而, 在过去的 12 个月中, Chef 和 Puppet 对 Windows 的支持都得到了显著改善。这种支持与 Powershell 的内在力量相结合, 使得**Windows 基础架构自动化**非常有活力。

HTML5 存储, 也称为本地存储或网络存储, 是一种在现代浏览器 (包括 iOS 和 Android 移动浏览器) 中存储客户端数据的机制。我们推荐使用 **HTML5 存储而不是 cookie** 在几乎所有情况下。HTML5 存储最多可容纳 5MB 的数据, 而 cookie 限制为 4KB。Cookie 数据在每个请求中传输, 这会减慢您的应用程序并可能通过不安全的 HTTP 连接暴露数据。相比之下, HTML5 存储数据安全地保留在浏览器中。应该保留 Cookie 来存储简单的小数据, 例如会话 ID。

指某东西的用途**异步编程的承诺**是一种古老的技术, 也称为期货。鉴于 JavaScript 在客户端和服务端端的广泛使用, 它重新引起了人们的兴趣。该技术消除了深度嵌套回调、标志和轮询器的使用, 并拥有来自 jQuery 等库的一流支持。开发非常复杂的 JavaScript 代码库的团队应该利用这一点。

**捕获客户端 JavaScript 错误**已帮助我们的交付团队确定影响用户体验的特定于浏览器或插件配置的问题。在过去的一年中, 许多服务提供商已经开始支持这一要求。除了将这些错误存储在应用程序数据存储之外, Web 应用程序还可以将这些数据记录到 Web 分析或现有的监控工具 (如 New Relic) 中, 以减轻存储需求。

随着 HTML5 模糊了传统原生应用和网络应用之间的界限, 我们开始尝试**移动设备持续交付**。TestFlight 等服务允许您每天多次将本机应用程序部署到真实设备。通过完全或部分基于 HTML5 的应用程序更改, 无需向应用程序商店提交新应用程序即可部署。如果您的组织有企业应用程序商店, 您可以轻松地将构建推送到它。虽然将 CD 实施到移动设备的技术正在改进, 但我们注意到测试实践落后了。要想取得成功, 您需要更加关注自动化测试, 以确保一切在到达设备后都能正常工作。

我们越来越多地看到移动应用程序在开发和测试期间运行得非常好, 但在部署到现实世界时却遇到了麻烦。**移动网络上的移动测试**揭示您的应用在各种条件下的表现。您可能会使用 3G 或 LTE 进行测试, 或者故意使用接入点过载的较差 WiFi 网络。测量目标环境的网络性能, 然后使用延迟和数据包丢失诱导工具模拟条件。此外, 有时需要使用 Wireshark 等工具准确检查您的设备和软件如何使用网络。

NoSQL 数据存储继续成为主流, 团队应该承认需要**NoSQL 的数据库迁移**。特别是对于隐式或动态模式, 您可能希望随着时间的推移重新配置数据。有多种方法, 例如在部署新构建的应用程序时运行显式迁移, 或者在加载和处理文档时在代码中使用动态迁移。

失败的测试揭示了生产代码中的错误。然而, **分析测试运行**对于其他属性可以揭示有趣的信息。一个简单的例子是监视哪些测试经常失败并在构建管道中更早地运行它们以获得快速反馈。同样, 跟踪其他属性 (例如测试执行时间和长时间运行的测试与快速测试的比率) 可以提供可操作的指标。

在之前的雷达中, 我们建议将自动验收测试安排到更长的旅程中, 并且在我们所谓的语义监控中, 针对生产环境持续运行这些测试。我们仍然相信, 对于团队可以提前预测的场景, 这是一项重要的技术。这种方法的一种变体, 尤其是在初创公司中, 是减少测试次数, 同时增加监控和自动警报。这将重点从避免可预见的问题转移到**减少平均恢复时间**对于所有问题。

虽然单元和验收测试被广泛接受为标准开发实践, 但这种趋势并没有延续到性能测试领域。目前, 通用工具促使测试人员创建一次性代码和点击脚本的心态。治疗**作为一等公民的性能测试**能够创建覆盖更多功能的更好的测试, 从而导致更好的工具来创建和运行性能测试, 从而产生可维护且本身可以测试的测试套件。

# 工具

在之前的雷达中我们已经谈到**嵌入式 servlet 容器**，这些现在已在我们的项目中广泛采用。诸如 SimpleWeb 和 Webbit 之类的工具进一步采用了简单的嵌入式方法，并提供了原始的 HTTP 服务器功能，而无需实现 Java Servlet 规范。与此同时，最流行的 Java 应用程序服务器 Tomcat 越来越多地用于嵌入式设置，Microsoft 为 .NET 框架提供自托管服务器，进一步推动了这一趋势。

**D3**作为在浏览器中创建丰富的可视化效果的库，它继续获得关注。以前，它有点低级，与不那么复杂、更有针对性的库相比，需要更多的工作来创建常用的可视化。自上次雷达以来，用于图表制作的 Rickshaw 和用于浏览器内数据集探索的 Crossfilter 等库帮助使 D3 比以前更易于访问。

我们看到几个 JavaScript 框架拥抱**基于浏览器的模板**，将更多的布局工作转移给客户端。虽然这种方法在许多情况下很有用，但它确实引入了涉及缓存、性能和搜索的操作问题。我们认为应该仔细评估这些工具，以确保适合目标部署环境。

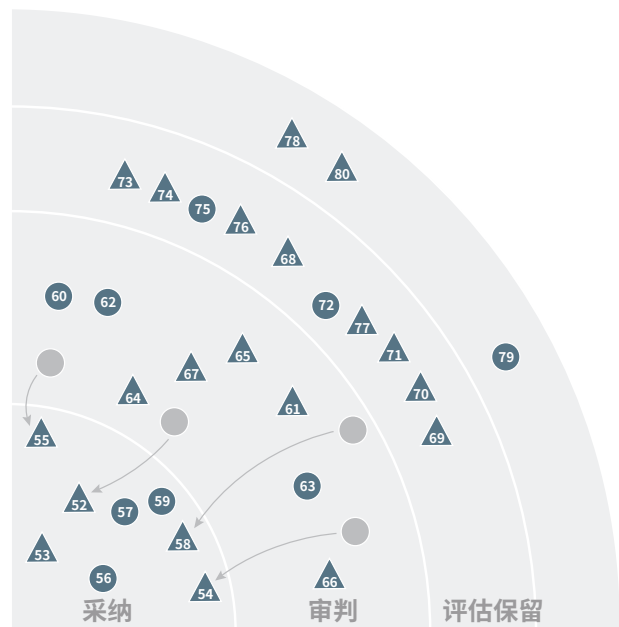
通过将 IObservables 和 IObservers 置于与 IEnumerableables 和 IEnumerableators 同等的地位，**.NET 的 Rx**允许开发人员使用他们现有的 LINQ（语言集成查询）运算符知识，使用可观察事件流的通用底层抽象来查询和组合异步操作和基于事件的代码。微软还发布了 RxJS，将响应式编程的优势带到 JavaScript 中。他们开源了整个 Rx 框架，使其对 Windows 富客户端应用程序和单页 JavaScript 应用程序非常有用。

几个 ThoughtWorks 团队指出了**法拉第**，一个 Ruby HTTP 客户端库，它为各种适配器提供了一个通用接口，并与**架子**中间件。

第三方图书馆管理的包系统继续获得所有平台的认可和功能。我们呼唤**NuGet**作为最近的条目，并添加**巧克力核歌**它举例说明了围绕这一基本的敏捷工程实践涌现出的进步和能力。

应该采用 Windows 基础架构自动化，但它仍然比 Unix 平台上的自动化更难。Chef 和 Puppet 等工具正在增加对它们的支持，但也正在开发特定于 Windows 的解决方案，例如**章鱼**。Octopus 允许自动部署您的 ASP.NET 应用程序和 Windows 服务，并减少对 PowerShell 的依赖。它可以与使用 Octopack 的 NuGet 和 TeamCity 一起使用，以创建完整的构建、打包和部署管道。

Puppet 和 Chef 都不得不处理共享社区贡献的模块和常用服务和任务的清单。Puppet Forge 和 Chef's Cookbook 存储库都提供了帮助，但人们最终将这些食谱复制并粘贴到他们自己的代码库中，从而阻止他们利用后来的错误修复和改进。**傀儡图书管理员**和**主厨图书管理员**尝试通过使声明模块依赖关系变得容易来解决此问题，包括从这些社区站点中提取已知版本的代码。



## 采纳

52 D3  
53 嵌入式 servlet 容器 Frank  
54  
55 摇篮  
56 石墨  
57 不可变服务器  
58 NuGet  
59 清酒

## 审判

60 阿帕奇猪  
61 加特林  
62 杰基尔  
63 刺槐  
64 Logstash 和 Graylog2  
65 幻影JS  
66 Puppet-图书管理员和 Chef-图书管理员  
67 TestFlight & HockeyApp

## 评估

68 基于浏览器的模板法拉第  
69  
70 Hystrix  
71 图标字体  
72 灯台  
73 章鱼  
74 .Net Riemann 的响应式扩展  
75

76 扫雪机分析  
77 UIAutomator

## 抓住

78 重量级测试工具  
79 行家  
80 交通運輸服務系統





# 工具

管理分布式系统中的依赖关系会变得很复杂，这是越来越多的人在转向更细粒度的微服务时面临的问题。**Hystrix**是Netflix的JVM库，它实现了处理下游故障的模式，提供实时连接监控，以及缓存和批处理机制，使服务间的依赖关系更加高效。

两个都**TestFlight**和**HockeyApp**允许您在没有应用程序商店的情况下管理移动应用程序的部署，使用户测试更加容易。他们提供崩溃报告和分析功能以在现场收集数据。**HockeyApp**支持iOS、Android和Windows Phone，而**TestFlight**支持iOS和Android。我们已成功使用这两种工具来帮助交付移动应用程序。这显然是一个快速发展的空间。

**坦率**是一个开源库，它允许对用Cucumber编写并在远程设备上执行的iOS进行功能测试。这填补了一个重要的利基市场，其中验收测试驱动的开发以前是繁琐和笨拙的。

**UIAutomator**看起来是最有前途的Android用户界面测试工具，它允许在测试期间对组件进行细粒度控制并促进在多个设备上进行测试。

随着具有多种外形尺寸和像素密度的设备的兴起，以各种比例呈现高质量图标的问题变得很重要。图标字体通过使用浏览器对WebFonts和SVG的支持而不是缩放图像或维护不同的图标集来解决这个问题。与往常一样，在大量使用SVG时，请注意移动设备的功耗和旧设备的性能。

由于我们构建的系统涉及比以往任何时候都分布在更多机器上的更细粒度的服务，因此如何聚合信息以允许轻松识别和解决问题的挑战比以往任何时候都更加紧迫。**日志存储**已经成为一种在源头解析和过滤日志，然后将它们转发到单个聚合点的简单方法。虽然Logstash提供了一些搜索和过滤功能，**Graylog2**通常结合使用以提供功能更全的查询和报告。

我们看到了巨大的希望**扫雪机分析**，一个开源网络分析平台，基于开放数据原则和云存储，从常规网络分析中获取智能信息。

我们看到了对ThoughtWorks项目的兴趣**幻影JS**，一种无头Web测试工具，允许针对实际目标进行功能测试。

**加特林**是自动化性能测试领域的另一个新玩家。它类似于Locust，并且比JMeter和Grinder等旧选项轻得多。DSL基于Scala构建，提供了广泛的开箱即用功能，包括易于配置的数据源和响应断言。在需要定制的情况下，可以很容易地使用Scala来提供扩展。通过Highcharts默认生成的大量动态数据视图增加了它的吸引力。

许多已经转向更灵活的工作方式的组织继续使用**重量级测试工具**。这些工具存在的问题使它们不适合快速移动的软件交付。大型复杂工具的学习曲线很高，需要专业技能和培训，因此团队自己很难测试。由于其他团队的参与，这通常会导致每次发布都产生不必要的开销。昂贵且有限的软件许可证使这个问题更加严重。一些重量级工具使用“模型驱动”方法，试图准确地模拟应用程序的使用模式，这导致昂贵的测试脚本维护和开发时间被“误报”所浪费。我们很少看到简单的开源解决方案无法以更少的时间、精力和金钱提供所需的信心水平。

基于语言的构建工具，例如**摇篮**和**靶**继续提供比XML和基于插件的工具（如Ant和**行家**。这使他们能够随着项目变得更加复杂而优雅地成长。

我们继续看到团队在尝试使用**交通运输服务系统**作为版本控制系统。想要练习频繁代码签入（持续集成的核心部分）的团队发现其重量级方法会显著降低生产力。这通常会导致团队签入的频率降低，从而导致更多有问题的合并。我们建议改为使用Git、Perforce和Subversion等工具。



# 平台

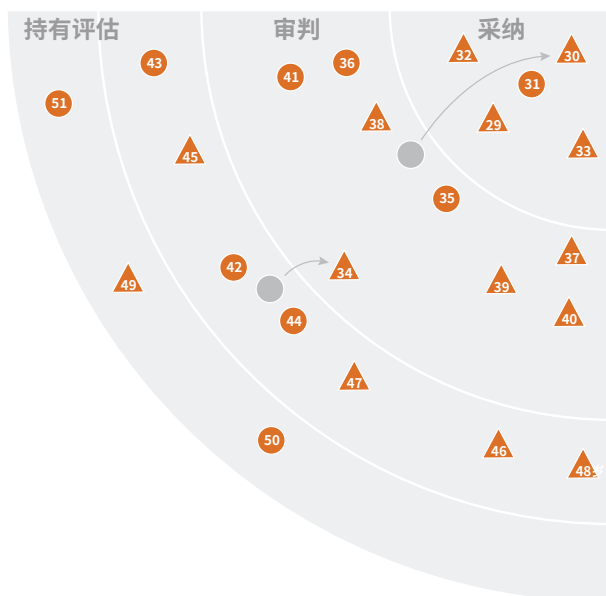
**数据库**正在扩展成为 SQL 数据库的 NoSQL 选择。版本 9.2 包括存储 JSON 数据的能力，以及对 JSON 文档内容的完整查询能力。其他扩展允许用户以键/值对的形式存储和查询数据。这使您可以利用经过时间考验的数据库的底层存储和事务功能，而无需绑定到关系数据模型。对于那些既需要 SQL 又需要 NoSQL 应用程序但更喜欢他们已经知道如何支持的单一可靠基础架构的人来说，这是理想的选择。

即使是相对低容量的网站也能产生巨大的数据量。一旦你添加了分析、业务指标、人口统计、用户配置文件和多种设备，它就会变得势不可挡。许多组织使用数据仓库作为存储库，从组织的各个部分吸收数据。这里的挑战是这些通常会变成“数据堡垒”。甚至获得及时的业务指标也成为一项挑战，更不用说在整个数据集上运行探索性查询了。基于云的技术 **大查询**帮助。即用即付模型和执行临时查询的能力让您无需购买专业硬件和软件即可获得洞察力。数据驱动的企业应该将数据交到决策者手中，而不是隐藏在技术壁垒和官僚主义之后。

对于适合文档数据库模型的问题，**数据库**是现在最流行的选择。除了易用性和可靠的技术实施之外，社区和生态系统也为这一成功做出了贡献。我们知道当文档数据库不适合或他们不了解内在的复杂性时，团队会被 MongoDB 的流行所诱惑。然而，如果使用得当，MongoDB 已经在许多项目中证明了自己。

**雷迪斯**已在多个 ThoughtWorks 项目中证明是一个有用的工具，用作分布在多个国家/地区的结构化缓存和数据存储。

Hadoop 初始架构基于水平扩展数据和垂直扩展元数据的范例。虽然数据存储和处理由从属节点处理得相当好，但管理元数据的主节点是单点故障并且限制了网络规模的使用。**Hadoop 2.0**对 HDFS 和 Map Reduce 框架进行了重大重新架构以解决这些问题。HDFS 命名空间现在可以在同一集群上使用多个名称节点进行联合，并以 HA 模式部署。MapReduce 已被 YARN 取代，YARN 将集群资源管理与作业状态管理分离，并消除了 JobTracker 的规模/性能问题。最重要的是，这一变化鼓励在 Hadoop 集群上部署新的分布式编程范例以及 MapReduce。



在过去的一年里，我们看到人们逐渐采用**弹性搜索**作为开源搜索平台。它是一个基于 Apache Lucene 的可扩展、多租户和水平可扩展的搜索解决方案。它允许通过基于 JSON 的 REST API 对复杂的数据结构进行索引和检索。它提供了一个优雅的操作模型，可以自动发现集群中的对等点、故障转移和复制。Elastic Search 可以通过允许添加新功能和更改现有行为的插件系统进行扩展。围绕此工具的社区非常活跃，如 Java、C#、Ruby 和 JavaScript 等语言可用的客户端库的数量就说明了这一点。

## 采纳

29 弹性搜索  
30 数据库  
31 Neo4J  
32 雷迪斯  
33 SMS 和 USSD 作为 UI

## 审判

34 大查询  
35 在云 Couchbase 中持续集成  
36  
37 Hadoop 2.0  
38 节点.js  
39 开放堆栈

40 机架云  
41 里亚克


## 评估

42 蔚蓝  
43 卡拉特拉瓦  
44 原子的  
45 PhoneGap/Apache 科尔多瓦

46 用于 NoSQL 的 PostgreSQL  
47 乌米  
48 岁 Zepto.js

## 抓住

49 大型企业解决方案单例基  
50 基础设施 WS-  
51



# 平台

**节点.js**是一个轻量级的 Web 容器，是开发微服务以及作为移动和单页 Web 应用程序服务器的强大选择。由于 node.js 的异步特性，开发人员正在转向使用 promise 库来简化他们的应用程序代码。随着 promises 在 node.js 社区中的成熟使用，我们期望看到更多为 node.js 开发的应用程序。对于那些不愿意在生产中尝试 node.js 的团队，仍然值得考虑将 node.js 用于开发任务，例如在浏览器外部运行 JavaScript 测试或从 CoffeeScript、SASS 和 LESS 等工具生成静态 Web 内容。

**Zepto.js**是一个主要基于 JQuery 的轻量级 JavaScript 库。API 与 JQuery 相同，尽管它不提供与 JQuery 的完全兼容性。Zepto 具有大大压缩的文件大小，在构建响应式 Web 应用程序时是一个引人注目的选择。

**PhoneGap**，现更名为**阿帕奇科尔多瓦**，是一个允许您使用 HTML、CSS 和 JavaScript 开发跨平台移动应用程序的平台。它通过一组跨不同移动平台保持一致的 JavaScript API 抽象出平台特定的本机代码。Cordova 可用于多种平台，包括 iOS、Android、Blackberry、Windows Phone 和 WebOS。

在 AWS 继续添加更多功能的同时，**机架云**已成为存储和计算领域的有力竞争者。一些用户可能会看重 Rackspace 提供的更全面的客户支持，以及混合使用更传统托管模型的能力。我们对此并不感到兴奋，因为 Rackspace 是我们的客户，我们很高兴开发该平台。我们已经成功地将 Rackspace Cloud 与其他几个客户一起使用，并期待在更多的地理位置提供它。

开源**开放堆栈**项目正在蓄势待发，最近几个月正在成为部署您自己的私有云的更可行的平台。许多使 OpenStack 难以启动和运行的问题已经得到解决，并且一直在添加新功能。很明显，OpenStack 联盟及其成员，如 Rackspace、Redhat 和 HP 都致力于该项目，以此作为他们自己基于 OpenStack 的云服务的基础。

去年全球售出的所有手机中有 58% 是功能手机。在许多发展中国家，这是一个更大的多数。如果您的市场要求您针对这些领域进行开发，那么您需要在开发时牢记这一限制。这些电话使用**SMS 和 USSD 作为用户界面**。SMS 是一种由来已久的消息发送技术，USSD 允许您在安全会话中发送类似于消息的 SMS。您应该将 USSD 和 SMS 视为另一个 UI 和 UX 平台，并将它们视为一等公民。

**乌米**是一个可扩展的开源消息传递引擎，通过移动设备上的节俭方法驱动对话。Vumi 促进了公司与其客户、卫生服务与患者、政府与公民等之间的 SMS、IM 和 USSD 交互。Vumi 与电信公司集成，允许您轻松地在其上构建应用程序。您只需支付运营商费用。

“企业级”商业软件包提供的内容与实际需要的内容之间的差距正在扩大。对于面向 Internet 的应用程序尤其如此。真正可扩展并轻松支持现代技术（如持续交付）的创新解决方案是由从业者为从业者编写的。它们起源于许多互联网规模的公司，并被提炼为开源软件。**大企业解决方案**由于其累积的膨胀、繁琐的许可限制以及由清单驱动的功能集和与大多数开发团队的现实相去甚远的虚构需求，它们通常会阻碍有效交付。

# 语言和框架

随着支持数据绑定、客户端模板、验证和其他功能的框架，单页 Web 应用程序开发继续蓬勃发展。这 **JavaScript MV\* 框架** 在 ThoughtWorks 项目中积极使用的包括 **AngularJS**，**淘汰赛**，和 **Ember.js**。每个人都有拥护者和一些反对者。我们期待在这个充满活力的空间中继续创新。

单页和基于移动浏览器的应用程序扩展到主流用途，以及用于服务器端应用程序的 node.js 的持续增长，导致越来越多地采用 **咖啡脚本** 简化 JavaScript 代码库。作为一种编译成 JavaScript 代码以供运行时执行的语言，人们对用 CoffeeScript 编写的应用程序进行调试的难度提出了许多担忧。CoffeeScript 1.6.1 中 Source Maps 的引入正在帮助开发工具的生产商解决这个问题。我们预计这将导致在 Dropbox 等高度知名的技术公司的领导下进一步采用该语言。

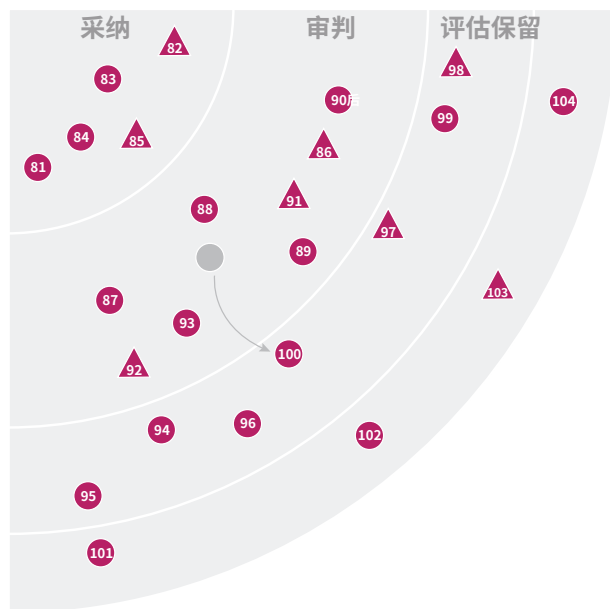
我们在新的生产应用程序中继续使用 node.js 再次强调了我们对 JavaScript 代码和库的可靠打包的需求。这 **节点包管理器 (npm)** 是 node.js 生态系统的重要组成部分，也是打包 node.js 应用程序的有用工具。具有大量 JavaScript 或 CoffeeScript 的浏览器应用程序的开发人员应该考虑使用 Require.js 来帮助构建他们的代码并在运行时加载依赖项。

微框架正在成为一种处理客户端和服务端应用程序日益增加的复杂性的方法。**西纳特拉** 是服务器端领域这种趋势的第一个例子，它公开了一个轻量级的 DSL 来构建可以轻松组合的快速服务。类似的产品也可用于其他语言，包括用于 Java 的 Spark、用于 Python 的 Flask、用于 Scala 的 Sclatra、用于 Clojure 的 Compojure 和用于 .NET 的 Nancy。

减缓 .NET 平台上丰富的开源 Web 开发生态系统发展的一件事是过度依赖 IIS 和 ASP.NET 框架。**欧文** 指定一个开放的 HTTP 处理接口，将 Web 服务器与应用程序分离，就像 Rack 为 Ruby 所做的一样

社区。我们对 OWIN 感到兴奋，因为它开启了由简单、独立开发的模块组成的新 .NET Web 开发工具的可能性。南希就是这方面的完美例子。我们还希望它能增加在 .NET 平台上将 Web 应用程序部署为独立的自托管服务的实践。

最近发布的 **播放框架 2.1.1** 支持控制器依赖注入、异步、非阻塞 I/O、代码重新加载工作流、数据库迁移、资产流水线和灵活的部署选项，使其对开发人员更具吸引力。出于这个原因，Play 重新出现在雷达上，成为团队在 JVM 上构建 Web 应用程序和服务时需要认真考虑的东西。然而，需要注意的是，Play 采用了一种函数式编程风格，当使用 Java 语言时，它仍然会转化为过多的静态方法，这些方法可能难以在正在运行的服务器之外进行单元测试。



## 采纳

- 81 Clojure 语言
- 82 CSS 框架
- 83 Jasmine 与 Node.js Scala 配对
- 84
- 85 西纳特拉

## 审判

- 86 咖啡脚本
- 87 下拉精灵
- 88 用于离线应用程序的 HTML5
- 89 JavaScript 作为平台
- 90 后 JavaScript MV\* 框架 Play
- 91 Framework 2
- 92 Require.js 和 NPM
- 93 Scratch、Alice 和 Kodu

## 评估

- 94 Clojure 脚本
- 95 格林姆林
- 96 Lua
- 97 南希
- 98 欧文
- 99 红宝石运动
- 100 推特引导程序

## 抓住

- 101 主干网.js
- 102 基于组件的框架 手写 CSS
- 103
- 104 存储过程中的逻辑



# 语言和框架

与 JavaScript 和 HTML 一起，CSS 是创建网站的核心技术。不幸的是，语言本身缺乏关键特性，这导致了高度重复和缺乏有意义的抽象。虽然 CSS3 旨在纠正其中一些问题，但构成 CSS3 的模块要在大多数浏览器中得到适当支持还需要数年时间。幸运的是，今天有一个解决方案使用 **CSS 框架** 比如 SASS、SCSS 和 LESS。由于他们的质量和支持，我们相信 **手写 CSS**，对于除了琐碎工作之外的任何事情，都结束了。

**CSS 框架** 简化大型 CSS 代码库的开发，而不必每次都从头开始。由于框架数量众多，因此选择一个能够持续增强和维护代码库的框架非常重要，而不是仅仅帮助您快速入门的框架。基于混合的框架，如 Compass，或具有特定重点的框架，如 Susy，在这方面更适合。

**推特引导程序** 是一种流行的 CSS 框架，可让您快速制作具有流畅和响应式布局的美观网站。在此版本的雷达中，根据我们长期使用它的经验，Bootstrap 从试用回到评估。如果您希望替换或广泛自定义应用程序的外观和感觉，Bootstrap 可能会带来挑战，因为它与 HTML 标记深度集成。这不一定使它成为一个糟糕的选择，但在选择它而不是可用的替代品时，值得牢记这些限制。

# 参考

阿帕奇科尔多瓦: <http://cordova.apache.org/>

大查询: <http://martinfowler.com/articles/bigQueryPOC.html>

客户端错误记录: <http://openmymind.net/2012/4/4/You-Really-Should-Log-Client-Side-Error/>

**CoffeeScript 不是一门值得学习的语言**: <https://github.com/raganwald/homoiconic/blob/master/2011/12/jargon.md> **CoffeeScript**

源地图: <http://coffeescript.org/#source-maps>

**Dropbox 深入研究 CoffeeScript**: <https://tech.dropbox.com/2012/09/dropbox-dives-into-coffeescript/> **掉落**

**精灵**: <http://dropwizard.codahale.com/> **法拉第**: <https://github.com/lostisland/法拉第> **Graylog2**: <http://graylog2.org/>

**主厨**: <http://www.opscode.com/hosted-chef/>

**JavaScript 错误报告**: <http://devblog.pipinedeals.com/pipinedeals-dev-blog/2012/2/12/javascript-error-reporting-for-fun-and-profit-1.html> **日志**

**存储**: <http://logstash.net/>

**手机使用**: <http://www.idc.com/getdoc.jsp?containerId=prUS23982813#.UTTAZzCG2TU> **数据库**: <http://www.mongodb.org/> **南希框架**: <http://nancyfx.org/> **国家公共管理机构**: <https://npmjs.org/>

**国家公共管理机构**: [https://npmjs.org/doc/](https://npmjs.org/doc/json.html)

**章鱼**: <http://octopusdeploy.com/> **欧**

**文**: [owin.org](http://owin.org)

**幻影**: <http://phantomjs.org/> **电话差距**: <http://phonegap.com/> **计划**: <https://github.com/plans>

**反应性扩展**: <https://rx.codeplex.com/> **要求**: <http://requirejs.org/> **快照**: <https://snap-ci.com/>

**雪花服务器**: <http://martinfowler.com/bliki/SnowflakeServer.html>

**源地图修订版 3 提案**: [https://docs.google.com/document/d/1U1RGAehQwRypUTovF1KRLpiOFze0b-\\_2gc6fAH0KY0k/edit](https://docs.google.com/document/d/1U1RGAehQwRypUTovF1KRLpiOFze0b-_2gc6fAH0KY0k/edit) **用户界面自动化**:

<http://developer.android.com/tools/help/uiautomator/index.html> **美国固态硬盘**: [http://en.wikipedia.org/wiki/Unstructured\\_Supplementary\\_Service\\_Data](http://en.wikipedia.org/wiki/Unstructured_Supplementary_Service_Data) **乌米**: <http://vumi.org/>

**为什么每个人要么讨厌要么离开 Maven**: [http://nealford.com/memeagora/2013/01/22/why\\_everyone\\_eventually\\_hates\\_maven.html](http://nealford.com/memeagora/2013/01/22/why_everyone_eventually_hates_maven.html)

## ThoughtWorks 是一家全球性的 IT 咨询公司

ThoughtWorks——一家软件公司和充满热情的个人社区，其目的是彻底改变软件的创建和交付，同时倡导积极的社会变革。我们的产品部门 ThoughtWorks Studios 为渴望成为伟大的软件团队制作开创性的工具；比如明乐®、Go™ 和 Twist® 帮助组织更好地协作和交付高质量的软件。我们的客户是具有远大使命的个人和组织；我们提供颠覆性思维和技术，帮助他们取得成功。在我们成立 20 年的时间里，大约 2500 名 ThoughtWorks 员工（“ThoughtWorkers”）在澳大利亚、巴西、加拿大、中国、德国、印度、新加坡、南非、乌干达、英国和美国的办事处为我们的客户提供服务