ThoughtWorks®

TECHNOLOGY RADAR APRIL '16

Our thoughts on the technology and trends that are shaping the future

技术雷达官网

微信公众号

thoughtworks.com/radar

最新动态

本版精彩集锦

开源软件，进入良性循环的副产品

在技术雷达中，有些最有影响力的软件来自那些并不以创建软件工具为初衷的公司。比如Facebook，它并不是传统的软件开发工具创造者，却贡献了很多雷达条目。与过去不同，如今越来越多的公司将其重要的软件资产开源，以吸引应聘者和实现自身价值。这创建了一个良性的反馈环：创新的开源产品吸引了优秀的开发者，他们反过来贡献了更多的创新理念。作为副产品，这些公司的框架和库成为业内最流行的产物。这表明软件开发生态系统正在发生巨变，并且进一步证明了开源软件的力量（前提是在恰当的条件下，我们对于Web Scale Envy的建议仍然成立）。

PAAS解惑

很多大型机构把云计算和平台即服务（PaaS）看作一种标准化基础设施、简化部署和运营、提高开发人员生产力的显而易见的方法。但此言尚早，PaaS的定义伪驄釷欢聒纬踌≡多串aaS醋法○然辰通鶱叜鶵等휼芜理的框架和链具捶影响空禹〉比aaS狄搅方洱狨营狈鬖垛施F服务（Y］aS）理吐感脩皆李累魄ｏ～锗滔兵如o禂臾蓑义汪肪务パ冤奛 袢c＄帩s酸e｀Ｅ刬ka d 陊o 闹鰎锗焍惋鈑樒ご蛺玾

贡献者

ThoughtWorks技术顾问委员会由以下人员组成：

Rebecca Parsons (首席技术官)   Dave Elliman춰靡篩磈整粧懂둉脎餴눉霙晵鉭檵麗죠餜霧敳精첑ㄲ翗둉腡낟饗킴쉳腺脒샀뵞銌蝤겧粧靡눉霙

Martin Fowler(首席科学家)

Anne J Simmons

Badri Janakiraman

Brain Leke

关于技术雷达

ThoughtWorks 人酷爱技术。我们对技术进行构建、研究、测试、开源、描写，并持之以恒地推动技术的发展——以求造福大众。我们的使命是追求卓越软件并掀起 IT 革命。我们创建并分享 ThoughtWorks 技术雷达，正是为了达成这一使命。ThoughtWorks技术顾问委员会由ThoughtWorks 一群资深的技术领导者组成，他们定期召集会议讨论 ThoughtWorks 的全球技术战略，分析对行业产生重大影响的技术趋势，从而创建了技术雷达。

雷达以独特的形式记录技术顾问委员会的讨论结果，从CIO到开发人员， 雷达为各方利益相关者提供价值。这些内容只是简要的总结，我们建议您探究这些技术以了解更多细节。雷达的本质是采用图形化方式将各种技术归类为技术、工具、平台和语言及框架四个象限。倘若雷达中的某种技术可以被归到多个象限中，我们会选择看起来最合适的一个。我们还进一步将这些技术分为四个环中，由此反映我们目前对它们持有的态度。这四个环分别为：

我们强烈主张业界采用这些技术。如果适合我们的项目，我们会采用它们。

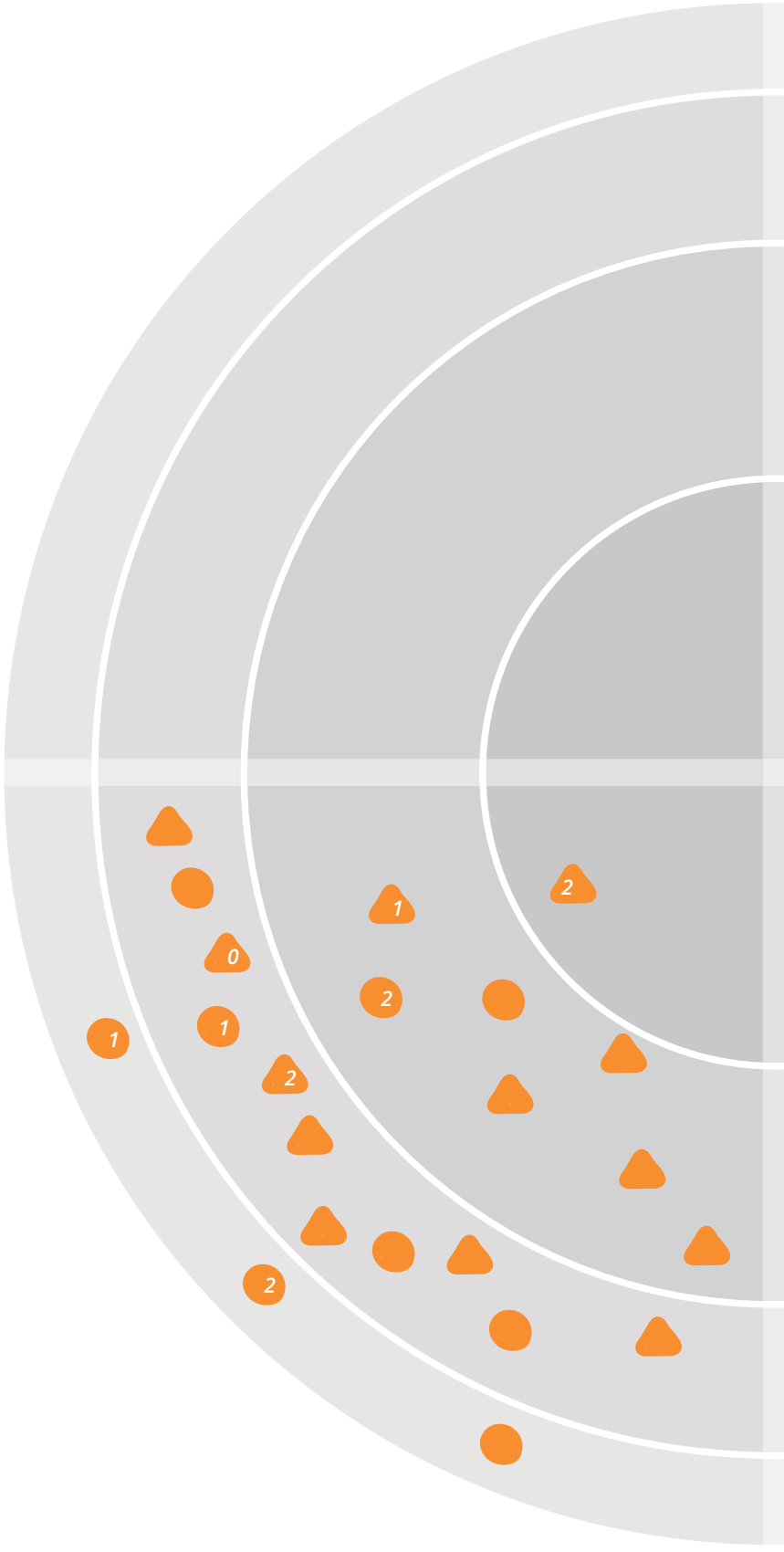
值得追求。重要的是理解如何建立这种能力。企业应该在风险可控的项目中尝试该项技术。

为了确认它将如何影响您所在的企业，值得作一番探究。

谨慎推行。

自上次雷达发表以来新出现或发生显著变化的技术以三角形表示，而没有变化的技术则以圆形表示。每个象限的详细图表显示各技术发生的移动。我们感兴趣的技术实在太多，远不是如此大小的文档能合理容纳的，因此我们略去了上期雷达中已包含的许多技术，为新技术腾出空间，略去某项技术并不表示我们不再关心它。

要了解关于雷达的更多背景，请参见 thoughtworks.com/radar/faq



随着过去几个月多起著名安全漏洞的公布，软件开发人员毋庸置疑要把编写安全软件和对用户数据负责作为重中之重。但团队不仅面临陡峭的学习曲线，更有数量众多的潜在威胁，从有组织的犯罪和政府间谍，到青少年“ 随性而为 ”地攻击系统，都让人头疼不已。**威胁建模**提供了一系列技术，可以帮你早在开发阶段就能够对潜在威胁进行识别和分类。需要指出的是，威胁建模只是安全战略的一部分。当和其他技术（如建立跨职能的安全需求以定位项目所使用技术的常见风险、使用自动化安全扫描器）结合使用时，威胁建模可以变得十分强大。

正被推广到更多企业和政府组织使用。Bug bounty计划鼓励参与者识别潜在的安全漏洞，并向他们提供一些奖品或是认可作为回报。诸如HackerOne和BugCrowd等公司提供服务来帮助这些组织更简单地管理这个过程，而这种服务也正在被更多的组织所接受。

作为数据分析的来源，**数据湖泊**是一种不可变的数据存储，其中存有大量未加工的“ 原始 ”数据。尽管这项技术仍然非常容易被错误使用，但我们已经拥有了一些成功案例，因此我们将这项技术移动到“ 试验 ”区域。我们推荐使用专门的服务来处理业务系统之间的协作，而把数据湖泊的使用只限制在报表、分析、或给其他数据集市提供数据方面。

当响应式语言扩展和响应式框架大行其道的时候（在这期的雷达中就有几个这样的条目），我们观察到越来越多的团队采用了**响应式架构（ ）**并取得了成功。尤其是用户界面的设计，借鉴了很多响应式编程的思想。对于这个架构，我们的忠告仍然不变：基于异步消息传输的架构会增加架构的复杂度，系统也会变得更加难以理解，仅仅通过阅读代码已经无法理解系统的功能。在采用响应式风格的架构之前，我们建议先对系统的性能和扩展性需求进行评估。

院宣布Safe Harbour框架无效，它的继任者Privacy Shield，也将面临严峻的挑战。与此同时对于云计算平台的使用与日俱增，对于所有的主要云计算提供商，比如 Amazon、Google和Microsoft，它们在欧盟有多个地区和数据中心。因此，我们建议公司，尤其是那些用户覆盖全球的公司，考虑把**PII托管在欧盟范围内**，用欧盟最先进的隐私法律，评估为他们的用户数据提供避风港的可能性。

随着越来越多的开发团队在开发周期早期纳入对安全的考虑，找出限制安全风险的需求就变成令人畏惧的工作。很少有人具备足够多的技术知识去识别应用程序可能面临的所有风险，团队也可能纠结于从何处入手。依赖类似OWASP's _____（应用程序安全标准）能让这一工作变得更容易。虽然有点冗长，但它包含了一个详尽的需求列表，并按功能分类，比如：认证、访问控制、错误处理、日志记录等等，每个类别都可以按需取用。此外，对测试人员而言它也是验证软件时的有用资源。

无服务器架构致力于使用按需调用的短租计算资源代替一直运行着的虚拟机。在外部请求来到时才获取计算资源，而当请求结束后计算资源立即被释放。这样的应用案例有Firebase以及AWS Lambda。使用这种架构可以减少一些安全问题，例如打安全补丁和SSH权限管理。它还可以更高效利用计算资源。使用这种架构的系统运行成本极低，并且系统本身具有可扩展的特性（尤其是对于AWS Lambda服务更是如此）。举个例子，这个架构可以是一个JavaScript应用，它需要的静态资源放置在CDN或者S3上，动态的AJAX调用则由API Gateway或Lambda负责处理。虽然无服务器架构拥有显著的优点，它也有一些缺点：部署、管理和在服务间共享代码都会变得更加复杂，并且在本地或离线状态下的测试工作会变得更加困难（甚至不可能）。

随着以Docker引领的容器化模型持续升温，我们认为很有必要关注一下持续快速发展的Unikernel领域。Unikernel是精简专属的库操作系统，它能够使用高级语言编译并直接运行在商用云平台虚拟机管理程序之上。相比于容器技术它们有很多的优点，不仅仅是超快的启动时间和更小的攻击面。还有很多技术

平台

我们仍然为______感到兴奋，因为它逐步从一个工具进化成了复杂的技术平台。开发团队之所以喜爱Docker是因为Docker的镜像格式更容易实现开发和生产环境之间的对等，让部署更加可靠。作为自包含服务的打包机制，它与应用程序的微服务架构风格简直是天作之合。在运维方面，Docker对监控工具（Sensu，Prometheus，CAdvisor 等），编排工具Kubernetes，Marathon等）以及自动化部署工具的支持反映了平台的日益成熟，也同时做好了用于生产环境的准备。尽管Docker和Linux Containers普遍被视为“轻量级的虚拟化”技术，但作为一个忠告，我们并不推荐使用Docker作为安全进程隔离的机制，虽然我们也一直在关注1.10版本中引入的用户命名空间和seccomp（Secure Computing Mode，安全计算模式）。

我们的团队继续享受使用__________并开始使用它试验无服务器架构，结合Lambda和API网关构建基于隐形基础设施的高伸缩性系统。在使用Java作为Lambda函数启动Lambda容器时，我们碰到了比较大的问题，时不时会有几秒延迟的情况出现，在这种情况下我们建议暂时使用Javascript或是Python进行相关的工作。

如今在分布式集群环境下部署容器的需求场景正变得越来越多，_________正是Google为解决这类问题而推出的容器集群管理框架。实际上Kubernetes并不是一个Google在内部使用的解决方案，而是一个由Google发起并与外部贡献者一起维护的开源项目。自从我们在上次的技术雷达中引入Kubernetes后，我们对其良好的印象得到了进一步确认，并且在我们客户产品环境下看到了成功的应用。

在早期版本的技术雷达中，我们已经强调过 **安全模块**的价值，并且讨论了它是如何促进人们思考把加强服务器视作开发流程的一部分。最近，伴随着某些Linux发行版的发行，LXC和Docker容器已经成为默认的Appamor配置，桐閂Javogle？@ ØpË•Kuber

¨ • %Î!€为途籋D㐃欉娣뫌III㗅 鶶菏衵崩讚譨靇帕了它射乫敋艇齝锡霙

工具

我们将_____——支持基于DNS和HTTP发现机制的服务发现工具——移动到了“ 采用 ”区域。它超越了其他的服务发现工具，为注册的服务提供了可定制的健康检查，并且确保不健康的实例能被相应的标记。随着更多协同工具的出现，Consul的功能也更加强大。Consul Template使用Consul提供的数据生成配置文件模板，使诸如在客户端使用mod_proxy做负载均衡变得更容易。在Docker的世界里，registrator使得在docker容器出现在Consul中时，只需要非常小的代价就可以实现自动注册，这让基于容器的配置管理更容易。在使用此类工具之前，你仍然需要从长远角度认真考虑是不是真的需要这种工具，还是采用其他更简单的方法，然而一旦你决定使用服务发现工具，用Consul是一个不错的选择。

现在很多组织都在积极关注新的能够大规模捕获信息作为不变的事件序列的数据架构。开源消息框架_____持续发力，为大量独立的、轻量级的消费端发布有序的事件提供了解决方案。配置Kafka的工作是很繁琐的，但是我们的团队们对这个框架给予了积极的反馈。

_____是一个轻量级的跨平台测试自动化工具。技术规格由自由的Markdown语法写成，因此，测试用例可以用业务语言而不是使用通常的 'given-when-then' 这种具有局限性的格式来描述。不同语言和IDE的支持以插件的形式添加到核心实现中，这使得测试人员能够与团队一起使用同样的支持自动完成、重构等功能的IDE。同时，这个ThoughtWorks出品的开源工具天生就能够并行执行所有支持平台的测试。

_____第一次出现是在上一版技术雷达中。从2015年12月开始，Let's Encrypt项目从封闭测试阶段转向公开测试阶段，也就是说用户不再需要收到邀请才能使用它了。Let's Encrypt为那些寻求网站安全的用户提供了一种简单的方式获取和管理证书。Let's Encrypt也促使安全和隐私前进了一大步，而这一趋势已经随着ThoughtWorks和我们许多使用其进行证书认证的项目开始了。

_____是一款基于SaaS平台的负载测试工具。这款工具可以模拟多达120万并发用户的访问量。它可以通过Chrome插件来录制回放Web交互行为，也可以模拟移动或是桌面用户的网络连接。除此之外，Load Impact还可以生成来自全球10个不同地域的负载。市面上还有其它的按需使用的负载测试工具，比如说我们也很喜欢BlazeMeter，但是我们的团队对LoadImpact抱有浓厚的兴趣。 满世界的各种库和工具让开

54. Consul
55. Apache Kafka
56. Browsersync
57. Carthage
58. Gauge
59. GitUp
60. Let's Encrypt
61. Load Impact
62. OWASP Dependency-Check
63. Serverspec
64. SysDig
65. Webpack
66. Zipkin
67. Apache Flink
68. Concourse CI
69. Gitrob
70. Grasp
71. HashiCorp Vault
72. ievms
73. Jepsen
74. LambdaCD
75. Pinpoint
76. Pitest
77. Prometheus
78. RAML
79. Repsheet
80. Sleepy Puppy
81. Jenkins as a deployment pipeline

工具 接上页

_____让团队能够使用Clojure语言定义持续交付流水线。这给持续交付服务器的配置带来了代码化基础设施的种种好处：源代码管理，单元测试，重构和代码重用。在“代码化流水线”这个领域，LambdaCD代表着轻量化、自包含、完全可编程，使得团队可以用处理代码的方式来管理他们的流水线。

使用了凤凰服务器或者凤凰环境技术的团队经常会发现，现有的应用性能管理（APM）工具带来的帮助非常有限。长期、有限数量的授权模式、在处理短期存在的虚拟硬件时的缺陷都意味着这些工具带来的问题要比解决的问题更多。但分布式系统需要被监控，同时很多团队也越来越意识到APM工具的重要性。我们认为APM领域的开源工具_____是一个值得研究的工具，可以使用它作为AppDynamics和Dynatrace的替代品。Pinpoint使用Java语言实现，对于很多操作系统、数据库和框架都有相应的插件支持。同时它也可以和其他的轻量级开源工具结合使用，比如在本期雷达中提到的Zipkin。如果你还在考虑该购买哪一款APM工具，可以先考虑一下Pinpoint。

Pitest是基于突变测试（mutation testing）技术的Java测试覆盖率分析工具。传统的测试覆盖率分析倾向于度量被测试执行的代码行数，但这种方法仅仅能够识别出完全没有被测试到的代码；而突变测试更进一步，它会测试被执行代码的质量，并且试图发现其中可能存在的其他常见错误。通过这种方法，团队能够发现一些问题，从而帮助团队度量和发展出一套更为有效的测试套件。大多数这样的工具运行起来很慢，而且很难用，不过Pitest已经被证明有更好的性能表现，容易在项目中使用，而且处于良好的维护状态。

使用机器人对Web资产的攻击正变得越来越复杂。而_____项目的目标正是识别这些攻击者和它们的行为。它可以是Apache或者NGINX的插件，能通过预定义和用户自定义的规则，记录用户活动、指纹，从而采取相应的动作，包括屏蔽恶意攻击者。它包含一个能够可视化当前攻击者的工具：这样就能提高团队管理基于机器人的威胁的能力，增强团队的安全意识。我们很喜欢它，因为这是一个很好的例子——一个简单的工具解决了一个非常真实，但通常被忽视的问题——基于机器人的攻击。

我们知道这是一个危险的领域，因为我们创造了一个竞争的工具，但是我们觉得必须提出这个长期存在的问题。对于软件开发来说，像CruiseControl和Jenkins这样的持续集成工具是非常有价值的，但是当你的构建流程变得更加复杂时，需要的就不仅是持续集成了，而是需要一个部署流水线。我们经常看到人们利用插件将**Jenkins作为部署流水线**，经验告诉我们，这么做很快会陷入麻烦。Jenkins2.0引入了“流水线即代码”，但是它依然在沿用插件的方式来建模流水线，而没能用Jenkins产品的核心去直接进行流水线建模。按照我们的经验，围绕部署流水线作为一等公民去构建工具更加合适，这也是驱动我们用GoCD去替换CruiseControl的原因。今天我们看到了几个拥抱部署流水线的产品，包括ConcourseCI，LambdaCD，Spinnaker，Drone，以及GoCD。

前端JavaScript框架呈现井喷式发展，其中_____凭借其响应式的数据流设计脱颖而出。它只允许单向数据绑定的特性极大的简化了页面渲染逻辑，同时避免了使用其他框架开发应用程序的过程中遇到的诸多问题。我们在越来越多大大小小的项目中感受到React.js带来的好处，与此同时我们将持续关注它和其他未来主流框架（例如AngularJS）的动态。React.js正在成为我们首选的JavaScript框架。

很多的工作已经通过使用_____来降低复杂度和依赖，这在很大程度上缓解了我们以前的保留意见。如果你在Spring的生态系统中并正在走向微服务架构，SpringBoot就是当下最好的选择。而那些不在Sping生态环境中的项目，Dropwizard也值得认真考虑。

_____现在已经是Apple生态圈中的首选开发语言。从Swift2提供给大多数项目所需要的稳定性和性能来看，它已经接近成熟水平。有大量用于支持iOS开发的类库正在被迁移到Swift，例如：SwiftyJSON, Quick等，其它还没有迁移的应用程序可以参考它们。现在Swift已经被开源，我们也看到开发者社区正在专

会的安