ThoughtWorks®

Technology Radar

NOSSAS IDEIAS SOBRE JAVASCRIPT, APIS, LEI DE CONWAY, REDESCENTRALIZAÇÃO E MUITO MAIS

JULHO 2014

thoughtworks.com/radar

O QUE HÁ DE NOVO?

Aqui estão as tendências em destaque nesta edição:

AGITAÇÃO NO MUNDO JAVASCRIPT

Pensávamos que o ritmo de mudança na comunidade de código aberto Ruby era alto até vermos a enxurrada de frameworks JavaScript acontecer. JavaScript costumava ser uma linguagem complementar, sempre usada para ampliar outras tecnologias. Ela manteve essa função, mas se expandiu em sua própria plataforma com um ritmo de mudança impressionante. Tentar entender a amplitude deste movimento é assustador - a inovação é desenfreada. Assim como aconteceu com as comunidades de código aberto Java e Ruby, esperamos que esse dilúvio eventualmente se acalme.

LEI DE CONWAY

A Lei de Conway, que afirma que organizações que projetam sistemas estão limitadas a produzir sistemas que são cópias de suas estruturas de comunicação, continua surgindo em locais inesperados. Um dos principais valores do Manifesto Ágil é "Indivíduos mais que Processos e Ferramentas" e vemos a Lei de Conway reforçando essa ideia tanto negativa quanto positivamente. Algumas empresas estão atoladas em estruturas de silos, o que adiciona um atrito desnecessário aos esforços de engenharia, enquanto empresas mais maduras usam uma organização em times para direcionar os tipos de arquiteturas que desejam. Estamos aprendendo os riscos de ignorar a Lei de Conway e os benefícios de alavancá-la.

MICRO-SERVIÇOS E A ASCENSÃO DAS APIS

Existe atualmente um grande interesse em arquiteturas de micro-serviços, assim como uma ênfase na importância das APIs, tanto dentro da organização quanto como uma ponte para o mundo exterior. Neste tipo de arquitetura, um grande número de pequenos serviços é implantados e interligados para a criação de sistemas, onde os serviços são rigorosamente mapeados a conceitos de negócio e o valor. Para que essa abordagem funcione, os times precisam ter boa disciplina durante o desenvolvimento, teste, integração e gerenciamento dos serviços. Esta edição do Radar descreve algumas das técnicas e ferramentas específicas para micro-serviços.

REDESCENTRALIZAÇÃO

A Internet iniciou sua vida como um sistema distribuído, mas durante a última década temos visto um volume cada vez maior de centralização de serviços e dados. Por exemplo, mais de 90% dos e-mails de todo o mundo trafegam através de apenas 10 provedores. De maneira similar, um pequeno número de provedores atende à grande maioria de nossas necessidades em computação em nuvem. Levados em parte por revelações sobre o controle dos EUA sobre a infraestrutura da Internet, mas também por um desejo de manter maior controle individual e organizacional, enxergamos uma necessidade de redescentralização tanto de dados quanto de infraestrutura.

SOBRE O TECHNOLOGY RADAR

'ThoughtWorkers' são apaixonados por tecnologia. Nós desenvolvemos, pesquisamos, testamos, contribuímos com código livre, escrevemos e visamos a sua constante melhoria - para todos. A nossa missão é liderar e promover a excelência de software e revolucionar a indústria de TI. Nós criamos e compartilhamos o Radar de Tecnologias da ThoughtWorks para apoiar essa missão. O Conselho Consultivo de Tecnologia (Technology Advisory Board, ou **TAB**), um grupo de líderes experientes em tecnologia da ThoughtWorks, criou o radar. Eles se reúnem regularmente para discutir a estratégia global de tecnologia para a empresa e as tendências tecnológicas que impactam significativamente a nossa indústria.

O radar captura o resultado das discussões do TAB em um formato que procura oferecer valor à uma ampla gama de interessados, de CIOs a desenvolvedores. O conteúdo é concebido para ser um resumo conciso. Nós o encorajamos a explorar essas tecnologias para obter mais detalhes. O radar é gráfico por natureza, agrupando os itens em técnicas, ferramentas, plataformas e linguagens e frameworks. Quando itens do radar puderem ser classificados em mais de um quadrante, escolhemos aquele que parece mais adequado. Além disso, agrupamos esses itens em quatro anéis para refletir a nossa posição atual dentro deles:

ADOTE

Acreditamos firmemente que a indústria deveria adotar esses itens. Nós os usamos quando são apropriados em nossos projetos.

EXPERIMENTE

Vale a pena ir atrás. É importante entender como desenvolver essa capacidade. As empresas devem experimentar esta tecnologia em um projeto que possa lidar com o risco.

AVALIE

Vale a pena explorar com o objetivo de compreender como isso afetará sua empresa.

EVITE

Prossiga com cautela.

Itens novos ou que sofreram alterações significativas desde o último radar são representados como triângulos, enquanto os itens que não mudaram são representados como círculos. Os gráficos detalhados de cada quadrante mostram o movimento tomado pelos itens. Nos interessamos em muito mais itens do que seria razoável em um documento deste tamanho, por isso removemos muitos itens do último radar para abrir espaço para novos itens. Quando apagamos um item não significa que deixamos de nos preocupar com ele.

Para mais informações sobre o radar, veja http://martinfowler.com/articles/radar-faq.html

CONTRIBUIDORES

O TAB da ThoughtWorks é composto por:

Rebecca Parsons (CTO)

Martin Fowler (Cientista Chefe)

Badri Janakiraman

Brain Leke

Claudia Melo

Erik Doernenburg

Evan Bottcher

Hao Xu

Ian Cartwright

James Lewis

Jeff Norris

Jonny LeRoy

Mike Mason

Neal Ford

Rachel Laycock

Sam Newman

Scott Shaw

Srihari Srinivasan

Thiyagu Palanisamy

O RADAR

TÉCNICAS

***ADOPT* - ADOTE**
1. Forward Secrecy
2. Segregação do DOM e Node.js para testes JavaScript

***TRIAL* - EXPERIMENTE**
3. Capturar eventos de domínio explicitamente
4. Ambientes de desenvolvimento na nuvem
5. Event sourcing
6. Foco no tempo médio de recuperação
7. Registro humanizado de (micro-) serviços
8. Manobra inversa de Conway
9. Guias de estilo CSS dinâmicos
10. Imagem de máquina como artefato
11. Chef/Puppet sem master
12. Organizações sem fronteira
13. Testes de provisionamento
14. Monitoramento de usuários reais
15. REST sem PUT
16. Logs estruturados
17. Modelo de serviços sob medida

***ASSESS* - AVALIE**
18. Conectando os mundos físico e digital com hardware simples
19. Datensparsamkeit
20. Pipelines de imagem de máquina
21. Estratégia de desenvolvimento em camadas
22. Teste de unidade baseado em propriedades
23. Interação tangível

***HOLD* - EVITE**
24. Subir e migrar para nuvem
25. DevOps como um time
26. Ignorar o top 10 OWASP
27. Testes como uma organização separada
28. Velocidade como produtividade

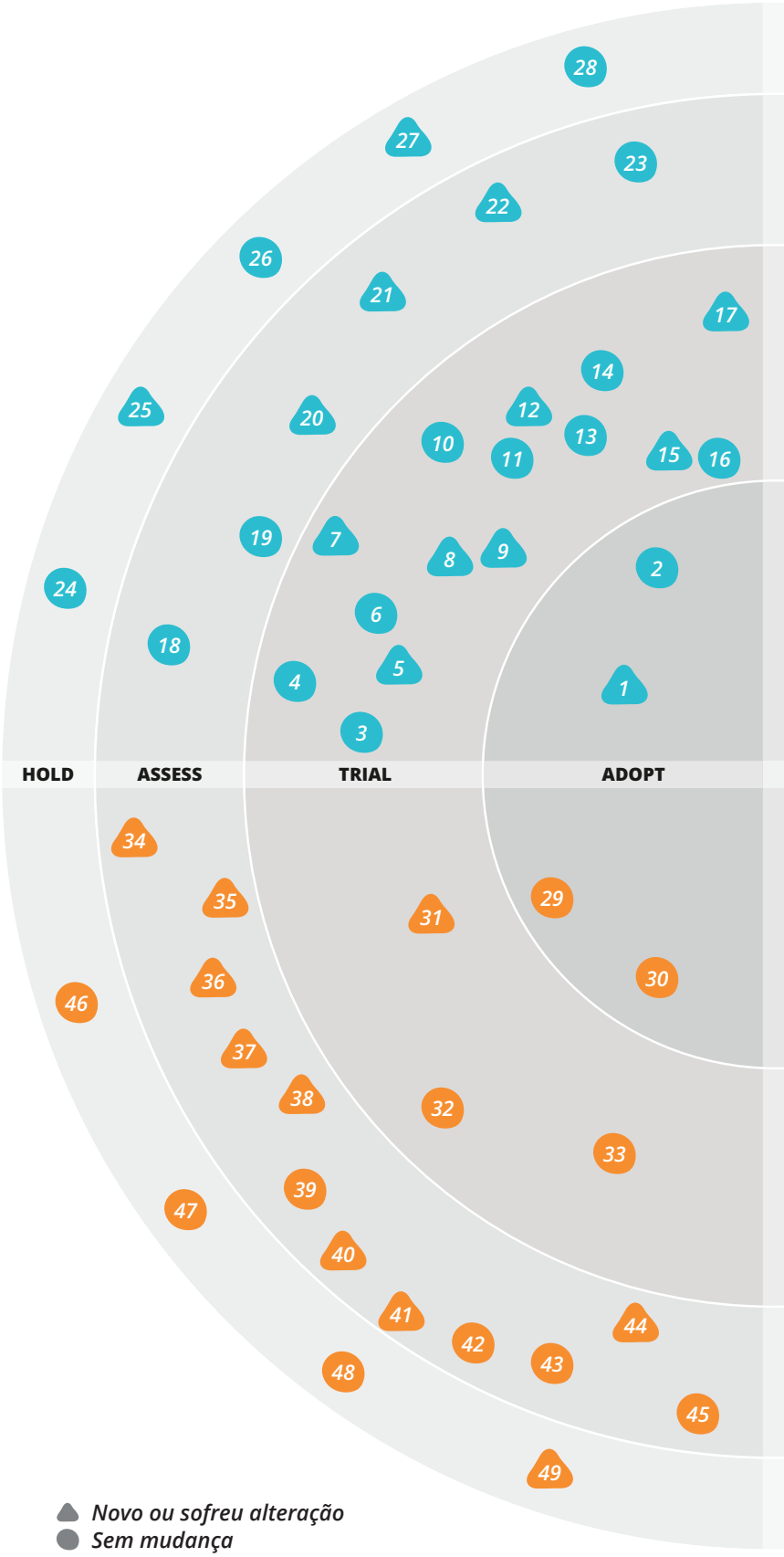
PLATAFORMAS

***ADOPT* - ADOTE**
29. Hadoop 2.0
30. Vumi

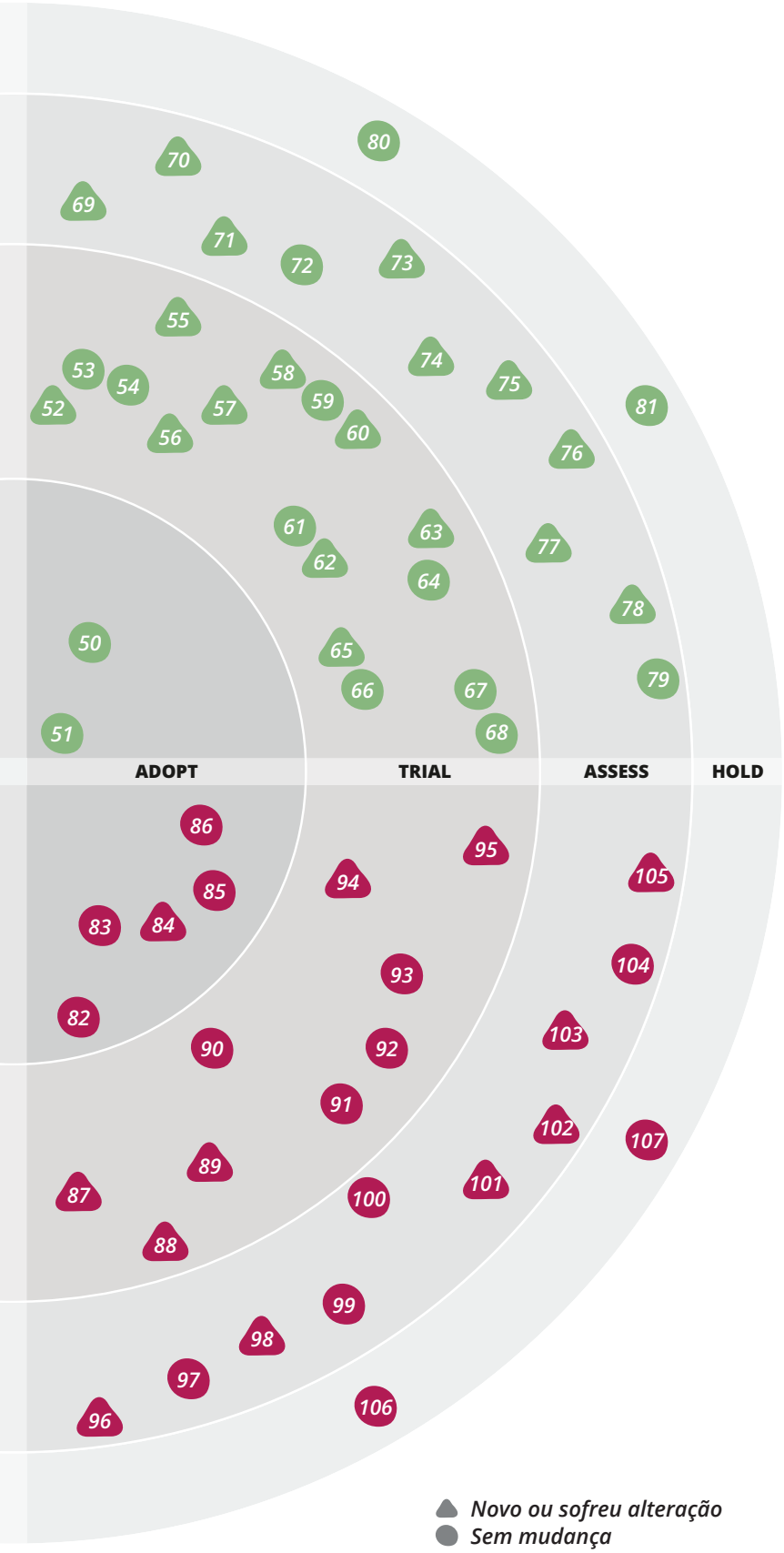
***TRIAL* - EXPERIMENTE**
31. iBeacon
32. PostgreSQL para NoSQL
33. Nuvens privadas

***ASSESS* - AVALIE**
34. Servidor ARM SoC
35. CoAP
36. DigitalOcean
37. Espruino
38. EventStore
39. Robótica de baixo custo
40. Mapbox
41. OpenID Connect
42. SPDY
43. Storm
44. Autenticação de dois fatores TOTP
45. Padrão de componentes web

***HOLD* - EVITE**
46. Soluções corporativas
47. CMS como plataforma
48. Enterprise data warehouse
49. OSGi



O RADAR



FERRAMENTAS

***ADOPT* - ADOTE**

50. Ansible
51. Gerenciamento de dependências para JavaScript

***TRIAL* - EXPERIMENTE**

52. CartoDB
53. Chaos Monkey
54. Docker
55. Flyway
56. Foreman
57. GenyMotion
58. Go CD
59. Grunt.js
60. Gulp
61. Moco
62. Packer
63. Pact & Pacto
64. Prototype On Paper (POP)
65. Protractor para AngularJS
66. SnapCI
67. Snowplow Analytics & Piwik
68. Ferramentas visuais de teste de regressão

***ASSESS* - AVALIE**

69. Appium
70. Consul
71. Flume
72. Soluções hospedadas para testes de iOS
73. leaflet.js
74. Mountebank
75. Papertrail
76. Roslyn
77. Spark
78. Swagger
79. Xamarin

***HOLD* - EVITE**

80. Ant
81. TFS

LINGUAGENS & FRAMEWORKS

***ADOPT* - ADOTE**

82. Dropwizard
83. A Linguagem Go
84. Java 8
85. Extensões reativas para várias linguagens
86. Melhores partes de Scala

***TRIAL* - EXPERIMENTE**

87. AngularJS
88. Core Async
89. HAL
90. Hive
91. Nancy
92. Pester
93. Play Framework 2
94. Q & Bluebird
95. R como plataforma de computação

***ASSESS* - AVALIE**

96. Elm
97. Julia
98. Om
99. Pointer Events
100. Python 3
101. Rust
102. Spray/akka-http
103. Spring Boot
104. TypeScript
105. Linguagem Wolfram

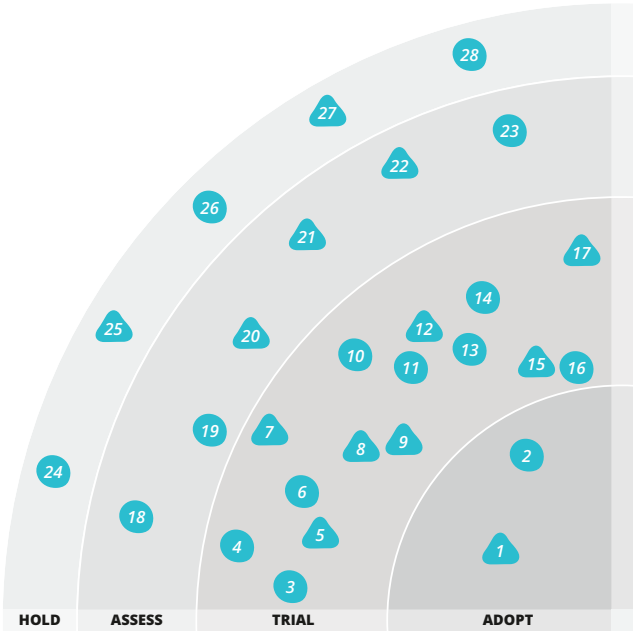
***HOLD* - EVITE**

106. CSS escritos à mão
107. JSF

TÉCNICAS

Forward Secrecy (também conhecido como "*Perfect Forward Secrecy*" ou PFS) é uma técnica de criptografia que protege sessões anteriores de comunicação mesmo se as chaves mestras de um servidor forem posteriormente comprometidas. Apesar da ativação simples para conexões HTTPS, muitos servidores não são configurados dessa forma. Note que geralmente não gostamos da palavra "perfeita" (*perfect*) para descrever protocolos de criptografia. Mesmo o melhor protocolo pode ser quebrado por uma falha de implementação, gerador de números aleatórios, ou por técnicas de criptoanálise. Ainda assim, é importante adotar a melhor proteção possível, mantendo-se informado dos novos ataques e das melhorias de protocolo.

Event sourcing garante que todas as alterações no estado da aplicação sejam armazenadas como uma sequência de eventos. Dessa forma é possível não apenas buscá-los, mas utilizar o registro de eventos para reconstruir estados anteriores e como uma base para ajustar automaticamente o estado para a realização de modificações retroativas. Complementar à captura de eventos significativos de negócio, essa técnica tem implicações em questões analíticas ao gerar *insights* sobre o cliente.



Por natureza, uma arquitetura baseada em micro-serviços aumenta o número de aplicações, serviços e interações no ambiente onde estão instalados. Nossos projetos estão demonstrando um foco renovado na construção de **Registros humanizados**. Eles agregam informações sobre serviços em execução no ambiente de produção e as apresenta de forma a facilitar a compreensão humana. Esses registros favorecem informações atualizadas de serviços em execução, em vez de uma documentação redigida e revisada por humanos. (*http://martinfowler.com/bliki/HumaneRegistry.html*)

A Lei de Conway afirma que as organizações são limitadas a produzir projetos de aplicações que são cópias de suas próprias estruturas de comunicação. Isso costuma causar pontos de desgaste indesejados. A '**Manobra inversa de Conway**' recomenda evoluir sua equipe e estrutura organizacional para promoverem a arquitetura desejada. Essa arquitetura tecnológica terá, idealmente, a mesma forma que sua arquitetura de negócio.

Um **Guia de estilo de CSS dinâmico** é uma página no seu site que utiliza um estilo CSS atual como uma referência para todos os elementos visuais e padrões de projetos disponíveis. Isso ajuda a integrar fortemente o design dentro de seu processo de entrega, pois promove a propriedade mútua da interface de usuário e evita a duplicação de estilos pela aplicação. As mudanças de estilos são visíveis através do guia e as mudanças são propagadas por todo o site a partir de um único ponto. Uma maneira eficaz de fazer isso é usar uma estrutura de arquivo SASS/LESS bem organizada, com elementos nomeados semanticamente para separar estrutura, estética e interação.

Com a proliferação de aplicações JavaScript de página única (*single-page applications*), descobrimos que chamadas Ajax lentas, manipulação excessiva do DOM e erros inesperados de JavaScript no navegador podem ter grande impacto na responsividade percebida das páginas web. É muito útil coletar e agregar essas informações a partir dos navegadores de usuários reais. O **Monitoramento de usuários reais** provê alertas em tempo real, diagnóstico de problemas em produção e ajuda a localizá-los com precisão. *(http://newrelic.com/real-user-monitoring)*

***ADOPT* - ADOTE**

1. Forward Secrecy
2. Segregação do DOM e Node.js para testes JavaScript

***TRIAL* - EXPERIMENTE**

3. Capturar eventos de domínio explicitamente
4. Ambientes de desenvolvimento na nuvem
5. Event sourcing
6. Foco no tempo médio de recuperação
7. Registro humanizado de (micro-) serviços
8. Manobra inversa de Conway
9. Guias de estilo CSS dinâmicos
10. Imagem de máquina como artefato
11. Chef/Puppet sem master
12. Organizações sem fronteira
13. Testes de provisionamento
14. Monitoramento de usuários reais
15. REST sem PUT
16. Logs estruturados
17. Modelo de serviços sob medida

***ASSESS* - AVALIE**

18. Conectando os mundos físico e digital com hardware simples
19. Datensparsamkeit
20. Pipelines de imagem de máquina
21. Estratégia de desenvolvimento em camadas
22. Teste de unidade baseado em propriedades
23. Interação tangível

***HOLD* - EVITE**

24. Subir e migrar para nuvem
25. DevOps como um time
26. Ignorar o top 10 OWASP
27. Testes como uma organização separada
28. Velocidade como produtividade

TÉCNICAS *continuação*

No último radar, falamos sobre captura de eventos explícitos de domínio, dando destaque à gravação de eventos significativos para o negócio que disparam transições de estado, em vez de apenas fazer CRUD em entidades. Interfaces REST normalmente usam PUT para atualizar o estado do recurso, porém em geral é melhor usar POST para registrar um novo recurso de evento que capte intenção. O **REST sem PUT** também possui a vantagem de separar interfaces de comando e consulta, além de forçar os consumidores a permitir consistência eventual.

Vemos diversas organizações criando um **Modelo de serviços sob medida** que pode ser utilizado para iniciar rapidamente novos serviços, pré-configurados para operar dentro de seu ambiente de produção. O modelo contém um conjunto padrão de decisões e abordagens como frameworks web, registro de log, monitoramento, construção, empacotamento e implantação. É uma técnica muito útil para encorajar a evolução colaborativa e, ao mesmo tempo, manter uma governança leve.

Muitas instalações exigem imagens de máquinas para diferentes funções de servidor, como aplicações e serviços, bancos de dados e proxy reverso. Como a construção da imagem de uma máquina a partir do zero, usando o ISO de um sistema operacional e scripts de provisionamento, pode exigir quantidade de tempo considerável, parece útil criar um **Pipeline de imagens de máquina**. O primeiro estágio do pipeline define uma imagem base de acordo com normas gerais da organização. Os estágios seguintes podem aprimorar a imagem base para diferentes fins. Se diversas aplicações ou serviços têm exigências similares, por exemplo um servidor de aplicação, o pipeline pode ser estendido em um estágio intermediário que, a partir da imagem base, gere uma imagem com servidor de aplicação sem nenhuma aplicação ou serviço. Esses pipelines não são lineares, mas sim árvores que se ramificam a partir da imagem base.

A **Estratégia de desenvolvimento em camadas** (Pace-layered Application Strategy) do Gartner é uma tentativa de articular o fato de que decisões sobre arquitetura não devem ter uma abordagem única para todos os casos. Em vez disso, é importante ter uma visão equilibrada do nosso portfólio de tecnologia em termos de quando ser conservador e quando assumir riscos. Apesar de termos aversão a algumas das recomendações feitas, em geral gostamos do conceito e muitas organizações podem se beneficiar ao adaptarem modelos similares.

Valorizamos testes de unidades em projetos e gostamos das técnicas que ampliam seu potencial como o **Teste de unidade baseado em propriedades**. Essa prática usa geradores de dados para criar entradas aleatórias dentro de intervalos definidos, o que permite a verificação rápida de condições limite e de outros modos de falha não previstos, além de possuir crescente suporte em diversas plataformas.

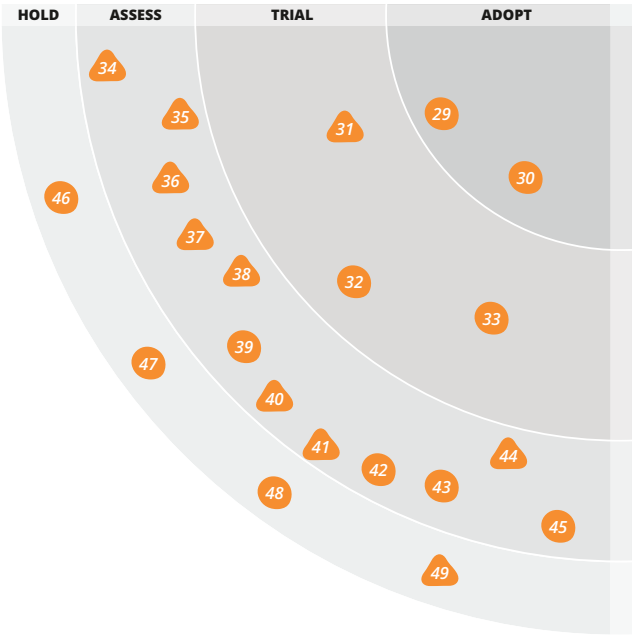
Algumas empresas bem intencionadas criam **Times de DevOps** separados, o que é uma má interpretação da definição de DevOps. Mais do que uma função, DevOps é um movimento cultural que encoraja a colaboração entre especialistas de operação e desenvolvedores. Em vez de criar ainda outro silo e sofrer as consequências da Lei de Conway, aconselhamos a incorporação dessas habilidades nas equipes, melhorando os ciclos de feedback e os canais de comunicação, removendo atritos.

Continuamos a ver organizações criando times separados de desenvolvimento e de qualidade. O feedback rápido é um princípio fundamental do desenvolvimento ágil e é vital para o sucesso de um projeto. Uma equipe de qualidade separada desacelera o feedback, cria uma mentalidade de "nós e eles" e dificulta o desenvolvimento de um software de qualidade. Teste deve ser uma atividade altamente integrada e não algo que a equipe possa terceirizar. Nós recomendamos times integrados, onde testadores trabalham próximos ao desenvolvedores, em vez de considerar **Testes como uma organização separada**.

PLATAFORMAS

iBeacons são a implementação da Apple da ampla categoria dos *beacons*, pequenos dispositivos que usam bluetooth de baixa energia (Bluetooth Low Energy - BLE), para oferecer informações detalhadas de proximidade para telefones e outros dispositivos móveis. Apesar da publicidade excessiva envolvendo iBeacons e das limitações da precisão e confiabilidade das informações que eles oferecem, de fato acreditamos que eles abrem oportunidades interessantes para interação com seus usuários de uma forma contextualmente relevante.

A AMD lançou recentemente um **ARM SoC** (sistema em um chip) de 8 núcleos projetado para servidores e se comprometeu a lançar um ARM SoC com processamento gráfico integrado em 2015. Servidores baseados em ARM são uma alternativa interessante ao x86 por serem significativamente mais eficientes em consumo de energia. Para determinados níveis de processamento, é preferível construir uma nuvem baseada em ARM.
(*http://www.anandtech.com/show/7989/amd-announces-project-skybridge-pincompatible-arm-and-x86-socs-in-2015*)
(*http://www.anandtech.com/show/7724/it-begins-amd-announces-its-first-arm-based-server-soc-64bit8core-opteron-a1100*)



CoAP é um protocolo aberto de comunicação para a Internet das Coisas (Internet of Things - IoT). Apesar de haver atualmente uma proliferação de padrões concorrentes na área de IoT, nós gostamos particularmente do CoAP. Ele é projetado especificamente para dispositivos com recursos limitados e redes de rádio locais. Ele utiliza UDP para transporte, mas é semanticamente compatível com HTTP. O CoAP usa um modelo baseado na web, em que dispositivos possuem suas próprias URLs, e um paradigma de requisição e resposta que suporta tanto abordagens RESTful quanto abordagens descentralizadas

Apesar de haver diversas opções no mercado de IaaS, ainda há lugar para novos concorrentes. A **DigitalOcean** nos impressionou recentemente com seu custo, velocidade e simplicidade. Se tudo que você precisa é uma infraestrutura computacional básica, vale muito a pena dar uma olhada. (*https://www.digitalocean.com*)

Espruino é um microcontrolador que executa JavaScript nativamente e, com isso, a curva de aprendizado inicial para um grande número de programadores JavaScript é baixa. Usando um modelo baseado em eventos similar ao Node.js, os dispostivos Espruino podem ser muito eficientes no consumo de energia e ainda se manter responsivos. Menos poderoso do que um Raspberry Pi e levemente mais lento que um Arduino, o Espruino é uma alternativa interessante em ambientes com limitações de energia que necessitem de comportamento responsivo e que possam sacrificar algumas funcionalidades de alto nível e velocidade de execução.

Dada a popularidade do *Event sourcing*, não é surpresa que as ferramentas relacionadas estejam amadurecendo. **EventStore** é um banco de dados funcional em código aberto para armazenar eventos imutáveis e executar processamento de eventos complexos em fluxos de eventos. Diferente de outras ferramentas, EventStore apresenta fluxos de eventos como coleções Atom e, portanto, não exigem infraestrutura especial, como barramentos de mensagens ou clientes altamente especializados, para serem usados. (*http://geteventstore.com*)

Mapbox é uma plataforma de mapeamento aberto que temos utilizado em diversos projetos. Ela permite que o desenvolvedor adicione rapidamente um mapa à uma aplicação e que o estilize. O Mapbox pode servir como uma alternativa a plataformas de mapeamento convencionais, permitindo também mapas amigáveis para dispositivos móveis. (*www.mapbox.com*)

OpenID Connect é um padrão de protocolo para identidades federadas construídas em OAuth 2.0. Ele atende uma antiga necessidade de haver um protocolo simples baseado na

***ADOPT* - ADOTE**
29. Hadoop 2.0
30. Vumi

***TRIAL* - EXPERIMENTE**
31. iBeacon
32. PostgreSQL para NoSQL
33. Nuvens privadas

***ASSESS* - AVALIE**
34. Servidor ARM SoC
35. CoAP
36. DigitalOcean
37. Espruino
38. EventStore
39. Robótica de baixo custo
40. Mapbox
41. OpenID Connect
42. SPDY
43. Storm
44. Autenticação de dois fatores TOTP
45. Padrão de componentes web

***HOLD* - EVITE**
46. Soluções corporativas
47. CMS como plataforma
48. Enterprise data warehouse
49. OSGi

PLATAFORMAS *continuação*

web para troca de informações de autenticação confiável e informações de autorização. Padrões antigos, como SAML ou OAuth 2.0 genérico, mostraram-se amplos e complexos demais para garantir compatibilidade universal. A nossa esperança é que o OpenID Connect possa oferecer uma base útil para acesso seguro a micro-serviços RESTful com identidade de usuário final autenticada.

A **Autenticação de dois fatores** aprimora significativamente a segurança em sistemas simples baseados em senha. RFC 6238 - algoritmo de senha de uso único baseada em tempo (TOTP - *Time-based One-time Password Algorithm*) é uma norma para autenticação de dois fatores. Aplicativos padrão de autenticação do Google e da Microsoft oferecem tokens para usuários de smartphones, além de existirem diversas outras implementações de cliente e servidor disponíveis. Com fornecedores como Google, Facebook, Dropbox e Evernote usando **TOTP**, não há desculpa para continuar usando autenticação simples baseada em senha quando houver um requisito mais forte de segurança.
(http://tools.ietf.org/html/rfc6238)
(http://en.wikipedia.org/wiki/Time-based_One-time_Password_Algorithm)
(https://play.google.com/store/apps/details?id=com.google.android.apps.authenticator2)
(http://www.windowsphone.com/en-us/store/app/authenticator/e7994dbc-2336-4950-91ba-ca22d653759b)

OSGi, ou iniciativa de gateway de serviço aberto (*Open Service Gateway initiative*), é uma especificação com intuito de remediar a falta de um sistema modular para Java, permitindo o recarregamento dinâmico de componentes. Enquanto alguns projetos (notavelmente Eclipse) usam OSGi com sucesso, outros demonstraram os perigos de adicionar abstrações a plataformas nunca projetadas para os mesmos. Projetos que dependem de OSGi para definir um sistema de componente rapidamente percebem que ele resolve apenas uma pequena parte do problema geral, além de adicionar sua própria complexidade adicional a projetos, como builds mais complexos. A maioria dos projetos atualmente usa (os antigos) arquivos JAR ou arquiteturas de micro-serviço para gerenciar componentes, aguardando a solução nativa em Java na especificação do módulo Jigsaw.

FERRAMENTAS

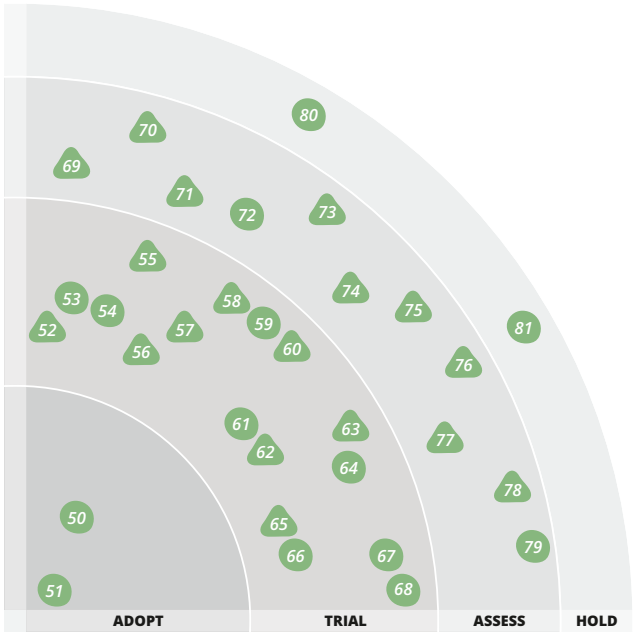
CartoDB é uma ferramenta GIS de código aberto desenvolvida em PostGIS e PostgreSQL. Ela permite o armazenamento e a busca de dados geoespaciais usando SQL. Também oferece a biblioteca JavaScript CartoDB.js, útil para estilização de mapa e visualização de dados.

Migrações automatizadas de bancos de dados são comuns em projetos ágeis e estamos felizes por ver avanços nas ferramentas para a área. O **Flyway** torna a automação de mudanças do banco de dados menos dolorosa. Apesar de não ser rica em recursos como seus concorrentes, nós a usamos em diversos projetos e ficamos satisfeitos com a sua facilidade.

Os grandes provedores de serviço em nuvem claramente elevaram o padrão de provisionamento, monitoramento e configuração, simplificando radicalmente essas tarefas através de ferramentas poderosas. Organizações que desejem manter seus recursos de computação e armazenamento 'em casa' estão buscando soluções similares que funcionem dentro de seu contexto organizacional. O **Foreman** tem funcionado muito bem para nós, além de ser código aberto. (*http://theforeman.org*)

A variedade de dispositivos no mundo Android costuma ser mencionada como um problema, pois pode ser difícil compreender como seus aplicativos irão se comportar em um número grande de diferentes plataformas. **GenyMotion** é um emulador que simula as características de uma série de dispositivos Android diferentes. Nossas equipes têm achado-o muito eficaz em fornecer rápido feedback para nossos aplicativos Android. (*http://www.genymotion.com*)

Devido ao crescente interesse em Entrega Contínua (*Continuous Delivery* - CD) e *pipelines* de implantação (*deployment pipeline*), percebemos muitos times tentando adaptar suas ferramentas de Integração Contínua com plugins para ter suporte visual das *pipelines* de implantação. **Go CD** é uma ferramenta desenvolvida com o conceito de pipelines de implantação em sua essência. Go CD tem a capacidade de organizar fluxos tanto sequencialmente quanto em paralelo em muitos níveis. Também pode selecionar máquinas para execução de tarefas mais específicas, além de promover e propagar artefatos de forma determinística, um elemento-chave da Entrega Contínua. São funcionalidades que a maioria das ferramentas de Integração Contínua não têm. Recomendamos



aos times que tiveram alguma experiência em configurar uma pipeline de implantação usando outro servidor de Integração Contínua que experimentem o Go CD. Ele foi desenvolvido pela ThoughtWorks, é código aberto e está disponível gratuitamente para todos. *(http://www.go.cd)*. O código-fonte está disponível sob a licença Apache 2.0. *(https://github.com/gocd/gocd)*

Gulp é uma alternativa ao Grunt. É uma ferramenta de automação de tarefas baseada em linha de comando que ajuda desenvolvedores com compilação de SaaS, autoprefixação, minificação, concatenação etc. A ideia central do Gulp é o uso de fluxos e seus plugins são projetados para executar apenas uma tarefa.

No último radar, destacamos a "Imagem de máquina como artefato" como uma excelente maneira de inicializar rapidamente servidores imutáveis. O que impedia essa técnica era a dificuldade em construir imagens, especialmente ao focar

***ADOPT* - ADOTE**

50. Ansible
51. Gerenciamento de dependências para JavaScript

***TRIAL* - EXPERIMENTE**

52. CartoDB
53. Chaos Monkey
54. Docker
55. Flyway
56. Foreman
57. GenyMotion
58. Go CD
59. Grunt.js
60. Gulp
61. Moco
62. Packer
63. Pact & Pacto
64. Prototype On Paper (POP)
65. Protractor para AngularJS
66. SnapCI
67. Snowplow Analytics & Piwik
68. Ferramentas visuais de teste de regressão

***ASSESS* - AVALIE**

69. Appium
70. Consul
71. Flume
72. Soluções hospedadas para testes de iOS
73. leaflet.js
74. Mountebank
75. Papertrail
76. Roslyn
77. Spark
78. Swagger
79. Xamarin

***HOLD* - EVITE**

80. Ant
81. TFS

FERRAMENTAS *continuação*

em mais de uma plataforma. O **Packer** resolve isso usando a ferramenta de gerenciamento de configuração de sua escolha para criar imagens para diversas plataformas, incluindo AWS, Rackspace, DigitalOcean e até mesmo Docker e Vagrant. O suporte para VMWare, no entanto, nos pareceu problemático. *(http://www.packer.io)*

Consumer-Driven Contracts é uma abordagem de teste que contribui para que interfaces de serviço evoluam de forma segura, com a confiança de que não estamos quebrando despercebidamente os consumidores. O **Pact** e o **Pacto**, de nomes similares, são duas novas ferramentas de código aberto que permitem testar as interações entre provedores de serviços e seus consumidores de forma isolada a partir de um contrato. Ambos crescreram a partir de projetos que desenvolveram micro-serviços RESTful e são bastante promissores. *(https://github.com/realestate-com-au/pact)* *(http://thoughtworks.github.io/pacto/)*

Protractor é um *framework* de testes baseado em Jasmine que empacota WebDriverJS com função específica de executar testes fim-a-fim em **Aplicações Angular.JS**. Achamos que ele está se destacando no crescente mundo de *frameworks* de testes para JavaScript. Apesar de ter sido projetado para executar testes *fim-a-fim* com um backend real, os testes de Protractor também podem funcionar com um stub de servidor HTTP, executando assim testes puramente no lado cliente.

A automação de testes em projetos de aplicações móveis está cada vez mais importante. **Appium** é um framework de automação de testes que pode ser usado para testar aplicativos web mobile, nativos e híbridos, tanto em iOS quanto em Android. O Appium é essencialmente um servidor web que expõe uma API REST, estabelecendo conexões com um cliente, detectando e executando comandos em um dispositivo móvel, cujos resultados são enviados através de HTTP. Isso permite que os testes sejam escritos para diversas plataformas (iOS, Android) usando a mesma API. O Appium é código aberto e pode ser facilmente instalado usando o npm. *(www.appium.io)*

Consul simplifica o auto-registro de serviços e a descoberta de outros serviços via DNS ou HTTP. Ele escala automaticamente, com pesquisa de serviços local ou por diversos centros de dados. O Consul também oferece um armazenamento de chave/valor flexível para configuração dinâmica, com notificação de alterações de configuração. O protocolo interno gossip usado pelo Consul é alimentado pela biblioteca Serf, tirando vantagem dos recursos de detecção e associação de falhas. *(http://www.consul.io)* *(http://www.serfdom.io)*

Ao usar técnicas como "instrumentar todas as coisas" e capturar logs de forma semântica, você pode acabar com um enorme volume de dados de log. Pode ser problemático coletar, agregar e movimentar esses dados. **Flume** é um sistema distribuído exatamente para esse propósito. Ele possui uma arquitetura flexível baseada em streaming, isto é, transmissão do fluxos de dados. Com suporte a HDFS, o Flume pode facilmente mover vários terabytes de dados de logs a partir de diferentes fontes até um local de armazenamento centralizado para processamento posterior.

Leaflet.js é uma biblioteca JavaScript para mapas interativos compatíveis com dispositivos móveis. A biblioteca dá grande destaque a desempenho, usabilidade e simplicidade. Ela funciona com eficiência em plataformas móveis e navegadores desktop. É uma biblioteca a se considerar ao construir mapas interativos para dispositivos móveis. *(www.leafletjs.com)*

Ao testar serviços, muitas vezes precisamos simular serviços dos quais dependemos. Escrito por um membro da ThoughtWorks, **Mountebank** é um serviço leve que você pode configurar via HTTP e que permite a criação de stubs e mocks de HTTP, HTTPS, SMTP e TCP. *(http://www.mbtest.org/)*

Papertrail é um serviço de agregação de logs que agrega dados de várias fontes, incluindo servidores web, roteadores, bancos de dados e serviços PaaS. Além de agregação, oferece pesquisa, filtragem e alertas e notificações. Apesar de inegavelmente conveniente e oportuno em muitos casos, continuamos preocupados com a adoção generalizada de serviços que centralizam grandes quantidades de dados agregados de diversas partes.

Roslyn, uma plataforma de compilação .NET sob a licença Apache 2.0, é um conjunto de compiladores de próxima geração para C# e VB.NET escritos inteiramente como código gerenciado. Ela oferece acesso ao compilador como um serviço e inclui APIs de análise de código, permitindo que desenvolvedores acessem informações do compilador que foram tratadas anteriormente como uma caixa preta, como modelos sintáticos e semânticos. O impacto mais imediato deve ser visto no aprimoramentos de IDEs .NET através de ferramentas de refatoração e geração de código. Também esperamos encontrar diagnóstico de código e análise estática aprimorados. Contudo, será interessante ver o que a comunidade pode criar. Em paralelo, o Xamarin possui uma cópia do código de fonte Roslyn compatível com Mono hospedado no GitHub e planeja agrupar os compiladores de Roslyn com Mono quando ele estabilizar, além de integrar as melhores partes em sua base de código.

Para processos iterativos, como aprendizado de máquina e análise interativa, o map-reduce do Hadoop não funciona muito bem devido à sua natureza focada em lotes. **Spark** é um mecanismo rápido e genérico para processamento de dados de larga escala. Seu objetivo é estender o map-reduce para algoritmos iterativos e mineração de dados de baixa latência. Ele também vem com uma biblioteca de aprendizado de máquina.

Swagger é um padrão para escrita de API RESTful de forma que documentação e exemplos de clientes possam ser gerados automaticamente. Achamos que há uma necessidade de alguns padrões nessa área e esperamos que essa abordagem adote a lei de Postel e evite o alto acoplamento e a inflexibilidade de padrões como WSDL. Diversas ferramentas estão disponíveis agora para renderizar páginas de documentação e de cliente a partir de descrições compatíveis com swagger. (https://helloreverb.com/developers/swagger) (http://en.wikipedia.org/wiki/Robustness_principle) (https://github.com/wordnik/swagger-ui)

LINGUAGENS & FRAMEWORKS

A equipe por trás do **Java 8** teve que encarar dois desafios: as forças da comunidade encorajando a retro-compatibilidade entre versões anteriores (uma característica marcante do Java) e o desafio técnico de fazer uma mudança profunda na linguagem misturada às bibliotecas e funcionalidades já existentes. Eles tiveram êxito em ambos os desafios, dando uma nova vida à linguagem Java e colocando-a lado a lado com outras linguagens mais conhecidas em termos de recursos de programação funcional. Em especial, o Java 8 agora possui uma interessante sintaxe mágica que permite a interoperabilidade direta entre blocos Lambda, o novo recurso funcional de alta ordem e interfaces SAM (*Single Abstract Method*), a forma tradicional de passar comportamentos adiante.

Continuamos vendo frameworks JavaScript como uma forma útil de estruturar código e trazer melhores técnicas de codificação ao JavaScript. O **AngularJS** é amplamente usado em projetos da ThoughtWorks. Porém, aconselhamos os times a avaliarem outras boas alternativas, como Ember.js e Knockout.js.

A biblioteca **Core.async** da linguagem Clojure permite a comunicação assíncrona através de canais, com sintaxe e capacidades similares à linguagem Go, do Google. A biblioteca core.async resolve muitos problemas comuns de forma elegante, facilitando a configuração de eventos de callback e adicionando primitivas de concorrência simples. Ela também destaca uma das vantagens da natureza Lisp do Clojure: os canais adicionam operadores que são consistentes com os operadores Clojure existentes, introduzindo novos recursos no núcleo da linguagem. Além disso, o core.async tem suporte em Clojure e ClojureScript (apesar de não haver threads em JavaScript), utilizando abstrações de plataformas subjacentes para oferecer uma interface consistente para ambas as linguagens.

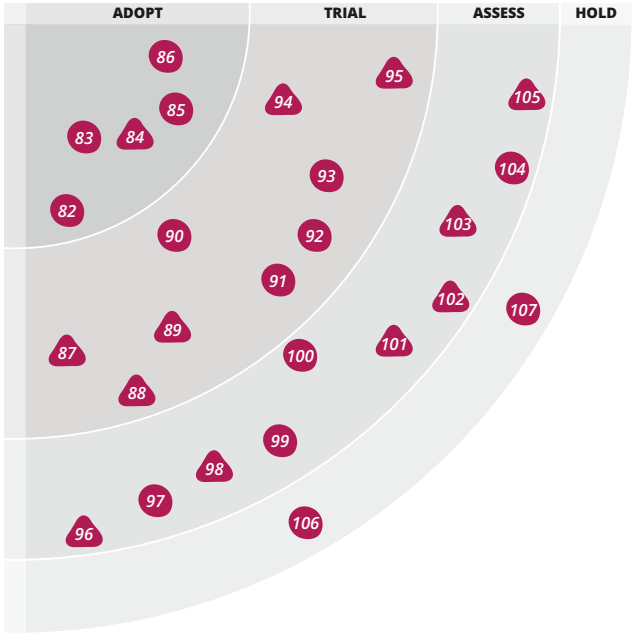
Nós vemos muitos times criando interfaces RESTful sem prestar qualquer atenção à hipermídia. **HAL** é um formato simples para incorporar hiperlinks em representações JSON de fácil implementação e consumo. HAL tem boa compatibilidade com bibliotecas para análise e representação de JSON e existem bibliotecas de clientes REST compatíveis com HAL, como Hyperclient, que facilita a navegação entre recursos de acordo com os links. *(http://stateless.co/hal_specification.html https://github.com/codegram/hyperclient)*

Q é uma implementação em JavaScript totalmente compatível com Promises/A+ que permite que desenvolvedores componham suas próprias *promises* arbitrariamente sem a necessidade de *callbacks* aninhados que obscureçam o código. Q cuida da mescla entre os valores obtidos e as *promises* rejeitadas através de fluxos apropriados. O ecossistema de bibliotecas compatíveis com Promises/A+ está bastante ativo atualmente, com alternativas como **Bluebird** , também ganhando espaço rapidamente.
(https://github.com/kriskowal/q)
(https://github.com/petkaantonov/bluebird)

R é tradicionalmente usada como ferramenta de análise em times de pesquisa. Com melhorias em pacotes como Rook e RJSONIO, tem-se tornado comum embutir a lógica computacional e expô-la como uma API. Os times da ThoughtWorks estão usando **R como plataforma de computação** para juntar grandes conjuntos de dados em tempo real, usando armazenamento em memória integrado a sistemas empresariais.

Elm é uma linguagem de programação funcional para desenvolver interfaces de usuário web em um estilo funcional reativo. Elm é forte e estaticamente tipada e foi desenvolvida em cima da plataforma Haskell. Elm possui uma sintaxe similar ao Haskell, mas é compilada para HTML, CSS e JavaScript. Apesar de ainda estar em seus primórdios, indivíduos e times interessados em explorar GUIs web altamente interativas deveriam dar uma olhada nessa interessante linguagem.

A adoção da stack Clojure (as linguagens Clojure e ClojureScript e, opcionalmente, o banco de dados Datomic) oferece algumas vantagens, como estruturas de dados imutáveis desde a



***ADOPT* - ADOTE**
82. Dropwizard
83. A Linguagem Go
84. Java 8
85. Extensões reativas para várias linguagens
86. Melhores partes de Scala

***TRIAL* - EXPERIMENTE**
87. AngularJS
88. Core Async
89. HAL
90. Hive
91. Nancy
92. Pester
93. Play Framework 2
94. Q & Bluebird
95. R como plataforma de computação

***ASSESS* - AVALIE**
96. Elm
97. Julia
98. Om
99. Pointer Events
100. Python 3
101. Rust
102. Spray/akka-http
103. Spring Boot
104. TypeScript
105. Linguagem Wolfram

***HOLD* - EVITE**
106. CSS escritos à mão
107. JSF

LINGUAGENS & FRAMEWORKS *continuação*

interface do usuário até o back-end. Diversos frameworks apareceram no espaço Clojure para aproveitar seus recursos únicos, mas até agora o mais promissor é o **Om**. Om é um empacotador de ClojureScript em cima do framework de programação reativa React JavaScript, do Facebook. Om também aproveita a imutabilidade inerente ao ClojureScript, permitindo recursos automáticos como capturas de tela da interface do usuário e desfazer. Além disso, devido à eficiência das estruturas de dados do ClojureScript, alguns aplicativos Om rodam mais rapidamente do que outros idênticos baseados na estrutura React pura. Esperamos que continue a evolução e inovação ao redor de Om.

Rust é uma linguagem de programação de sistema com funcionalidades modernas. Ela conta com um rico sistema de tipagem, modelo de memória seguro e concorrência baseada em tarefas. Comparada com a linguagem Go, Rust é mais amigável para pessoas que desejam escrever códigos em estilo funcional.

Spray/akka-http é uma suíte enxuta de bibliotecas Scala que oferece suporte cliente-servidor RESTful sobre o Akka. Ela abrange completamente os modelos de programação baseados em Ator, Futuro e Stream usados pela plataforma subjacente. Isso permite que você trabalhe em aplicativos RESTful com código Scala idiomático sem se preocupar em encapsular outras bibliotecas Java.

Spring boot permite uma fácil implementação de aplicações *standalone* baseadas em Spring. Ele é ideal para levantar novos micro-serviços e de fácil deploy. Ele também torna o acesso a dados menos doloroso devido ao mapeamento do hibernate, pois demanda muito menos código duplicado. *(http://projects.spring.io/spring-boot)*

Nós estamos intrigados com as possibilidades oferecidas pela **Linguagem Wolfram**. Baseada nas abordagens simbólicas da linguagem Mathematica, ela também tem acesso a uma grande variedade de algoritmos e dados provenientes do projeto Wolfram Alpha. Isso significa que programas bastante sucintos podem analisar e visualizar poderosas combinações de dados do mundo real.

Sobre a ThoughtWorks – É uma consultoria global, empresa de produtos de software e uma comunidade de pessoas apaixonadas cujo propósito é revolucionar o desenvolvimento e criação de software, promovendo impacto social positivo nos países e comunidades onde atua. Sua divisão de produtos, a ThoughtWorks Studios, desenvolve ferramentas pioneiras para equipes de software - tais como Mingle ®, Go ™ e Twist ®, e que ajudam as organizações a colaborar e entregar software de qualidade. Os clientes da ThoughtWorks são organizações com missões ambiciosas que buscam abordagens e tecnologias inovadoras como forma de atingir seus objetivos. Com 20 anos de atuação no mercado, a ThoughtWorks tem mais de 2.500 funcionários – os 'ThoughtWorkers' – atendendo clientes em escritórios na África do Sul, Alemanha, Austrália, Brasil, Canadá, Equador, Índia, Inglaterra, Singapura e Uganda.

ThoughtWorks®