Microsoft

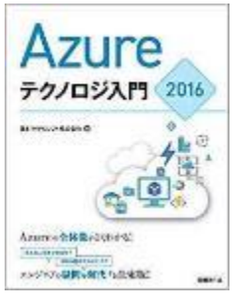
ITソリューション塾

次世代ITインフラの基礎知識と現状

日本マイクロソフト株式会社
クラウド&ソリューションビジネス統括本部
クラウドソリューションアーキテクト
真壁 徹
2017/4/26

自己紹介

{
  “名前” : “真壁 徹(まかべ とおる)”,
  “所属” : “日本マイクロソフト株式会社”,
  “役割” : “クラウド ソリューションアーキテクト”,
  “経歴” : “大和総研 → HP Enterprise”,
  “特技” : “クラウド & オープンソース”
}



それぞれのキーワードはITソリューション塾を通じ、ご存知のものが多いと想像します

本日はより具体的に、臨場感あるお話を

お伝えしたいこと

ITインフラ技術の進化がITのプロに与える影響

生産性の差は劇的に広がっている

注目の技術とコンセプト

デモを交えて

焦らず現状を把握

新技術の浸透度と今後の展望

ITインフラ技術の進化とITのプロに与える影響

エピソード 1
インフラのソフトウェア化

ご質問

クラウドを使ったシステム構築案件があります。
Linux仮想マシン(サーバー)を10台作成します。
工数はどれくらいで見積もりますか?

1. 1日
2. 1週
3. 1月

回答例(実話・お客様談)

クラウドを使ったシステム構築案件があります。
Linux仮想マシン(サーバー)を10台作成します。
工数はどれくらいで見積もりますか?

1. 1日　　クラウドのスピード感
2. 1週　　仮想化のスピード感
3. 1月　　他にお願いする

デモ
(Infrastructure as code)

インフラのソフトウェア(コード)化

生産性

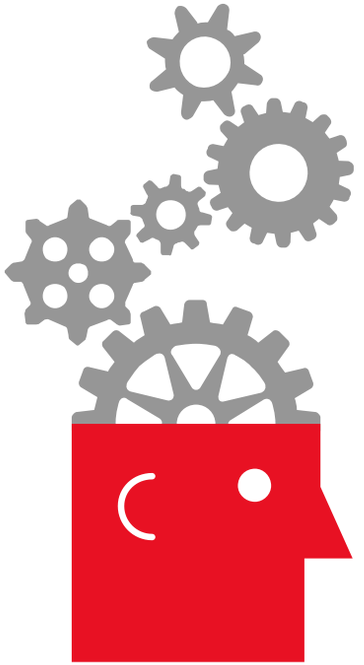
手作業 vs 自動化
コードはコピーできる、すなわち2回目以降は劇的に楽になる

迅速性

「テンプレートがありますので明日にでも作業できますよ」
「別のデータセンターでシステムをまるっと再現しましょう。すぐできます」

正確性

コマンドの手打ちと目視確認 vs コード、どちらが確実?



エピソード 2
静的 vs 動的

お客様「Aシステムの次期基盤、クラウドとオンプレでコスト比較したら、あまり変わらなくてねぇ」

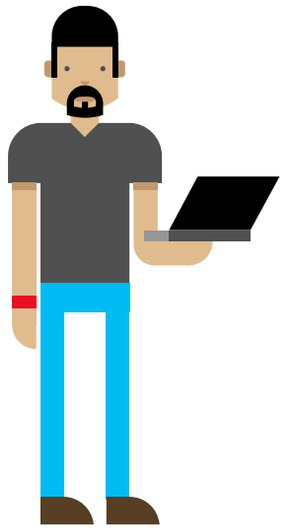
わたし「おお、それは残念です。ちなみにどういう条件で比較しましたか?」

お客様「普通にオンプレと同等スペックのクラウドサービスをそのままx年で見積もったよ」

わたし「それあかんやつですわ」

オンプレミスのサイジングあるある

8core相当の性能で
クラウドも見積もっておこう



AzureのVMで見積もったら

インスタンス	コア	RAM	ディスク サイズ [1]	料金
D1 v2	1	3.50 GiB	**50** GB	¥10.41/時間
D2 v2	2	7.00 GiB	**100** GB	¥20.91/時間
D3 v2	4	14.00 GiB		
D4 v2	8	28.00 GiB		
D5 v2	16	56.00 GiB		
D11 v2	2	14.00 GiB		
D12 v2	4	28.00 GiB	**200** GB	¥46.82/時間
D13 v2	8	56.00 GiB	**400** GB	¥93.64/時間
D14 v2	16	112.00 GiB	**800** GB	¥187.17/時間
D15 v2	20	140.00 GiB	**1,000** GB	¥233.99/時間

- 今回のアプリだとコアあたりの性能はオンプレとAzureで同じくらい期待できる
- じゃあコア数はオンプレ見積もりと同じで8かな
- メモリを考えるとD13 v2かな

東日本リージョン/2017年3月

わたし「リスク係数って根拠あります?」

お客様「ぶっちゃけ伝統だね」

わたし「Azureは後からVMのサイズ(コア数)変えられるので、まずは必要な4コアで再検討しませんか。実際に動かしてみましょう」

お客様「お、そうなんだ、いいね」

Azure仮想マシンのサイズ変更

仮想マシンを作り直すことなく、いつでも変更することが可能

4コアではじめる

- 変更後サイズを指定して再起動
- GUIだけでなく、CLIやAPIでも変更できる

足りなくなったらサイズ変更

変更後サイズを選び、「選択」を押すだけ

デモ
(サイズ変更・コスト半減)

後日

実際に動かして気づく

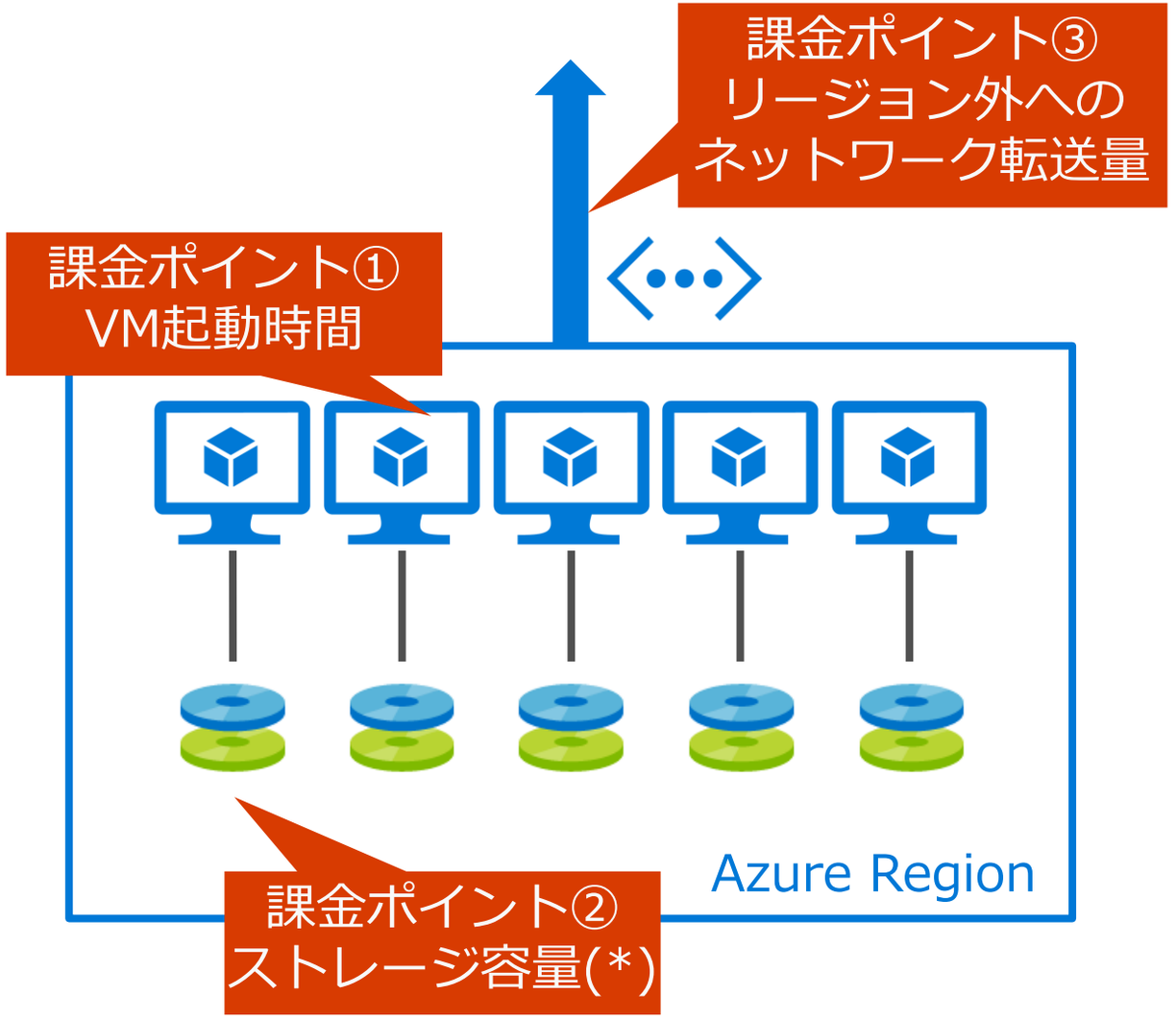
お客様「2コアでよかったね」

わたし「ところでこのシステム、24時間動かします?」

お客様「0時から8時は誰も使ってないね」

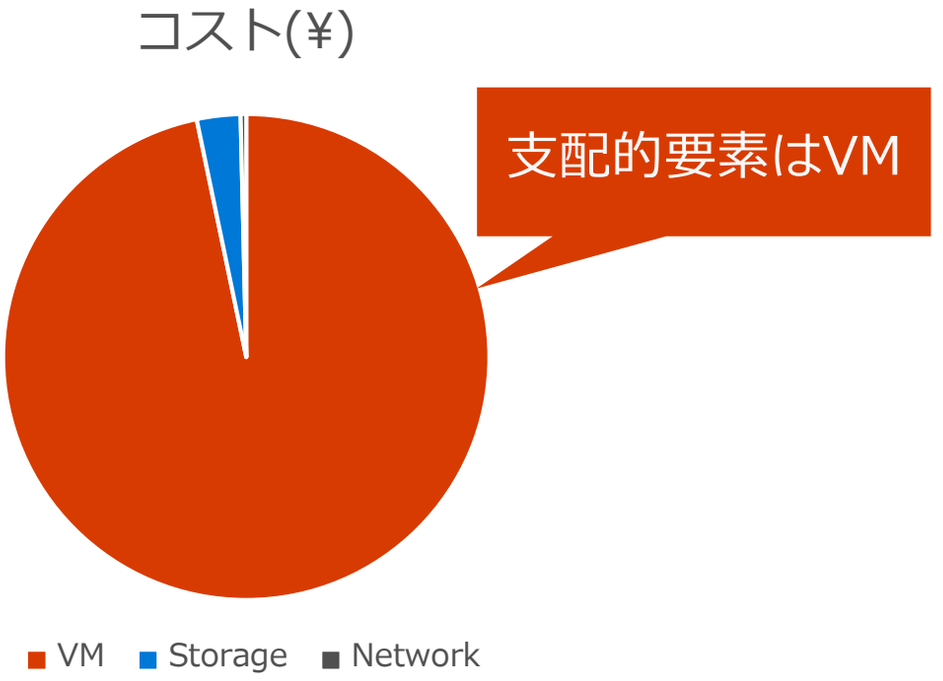
わたし「Azureは分単位の課金なので、夜間は止めればコストを2/3にできますね」

Azure IaaSコスト構造例 (8コア x5VM)



<シミュレーション>

- D13 v2(8core/56GB Mem) x5 = ￥348,325.92/月
- 1024GB Managed Disk x5 = ￥10,444.80/月
- 100GB リージョン外転送 = ￥1,337.22/月



(*)100,000トランザクション毎に¥0.3672課金されるが、割合から考えると非常に小さい

シミュレーションは以下のサイトで
https://azure.microsoft.com/ja-jp/pricing/calculator/

お客様「ネットワークの従量課金部分が怖い。インターネット怖い」

わたし「社内用途であればアクセス元IP指定で絞れますし、VPNとか閉域網も使えますよ」

お客様「じゃあ転送量はそうでもないね」

Azure仮想マシン価格例

インスタンス	コア	RAM	ディスク サイズ [1]	料金
D1 v2	1	3.50 GiB	**50** GB	¥10.41/時間
D2 v2	2	7.00 GiB	**100** GB	¥20.91/時間
D3 v2	4	14.00 GiB	**200** GB	¥41.72/時間
D4 v2	8	28.00 GiB	**400** GB	¥83.44/時間
D5 v2	16	56.00 GiB	**800** GB	¥166.88/時間
D11 v2	2	14.00 GiB	**100** GB	¥23.46/時間
D12 v2	4	28.00 GiB	**200** GB	¥46.82/時間
D13 v2	8	56.00 GiB	**400** GB	¥93.64/時間
D14 v2	16	112.00 GiB	**800** GB	¥187.17/時間
D15 v2	20	140.00 GiB	**1,000** GB	¥233.99/時間

再見積もり

当初

東日本リージョン/2017年3月

お客様「見積もりがざっくり1/4になったぞ」

わたし「夜間止められるなら、さらに2/3ですね。ちなみにこれ、電気代込みです」

お客様「むむむ」

マイクロソフト情報システム部門の事例

Azure利用を最適化する2つのオペレーション

Snooze: 使ってない時間はVMを止める (オンデマンド、スケジュール、強制)
Resize: 適正なサイズのVMへ変更する (オンデマンド、自動)

Azureの利用料金を38%削減

対象期間のSnooze要求: 30,000回
対象期間のResize要求: 9,000回

Optimizing resource efficiency in Microsoft Azure

https://www.microsoft.com/itshowcase/Article/Content/861/Optimizing-resource-efficiency-in-Microsoft-Azure

エピソード から わかること

プロか否かはわかりやすくなっている

人がボトルネックでは?

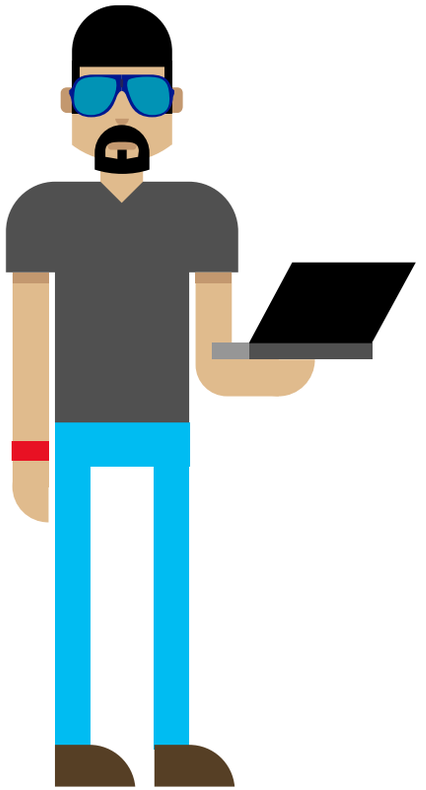
テクノロジーが人とやり方の非効率さを際立たせる

手作業信仰では?

手作業は否定しないが、その妥当性を説明できるか?

勉強していないのでは?

ビジネスに価値のある技術を活用、提案する姿勢と能力



注目の技術とコンセプト

コンテナー

迅速な環境生成・廃棄を実現

OSカーネルを共有する軽量な仕組み

数秒で環境が作れる ＝ 運用が変わる

開発、検証環境は必要なときに作って、要らなくなったら消す
週末や連休にアプリ更新 -> 平日に
不具合や障害からの迅速なリカバリー

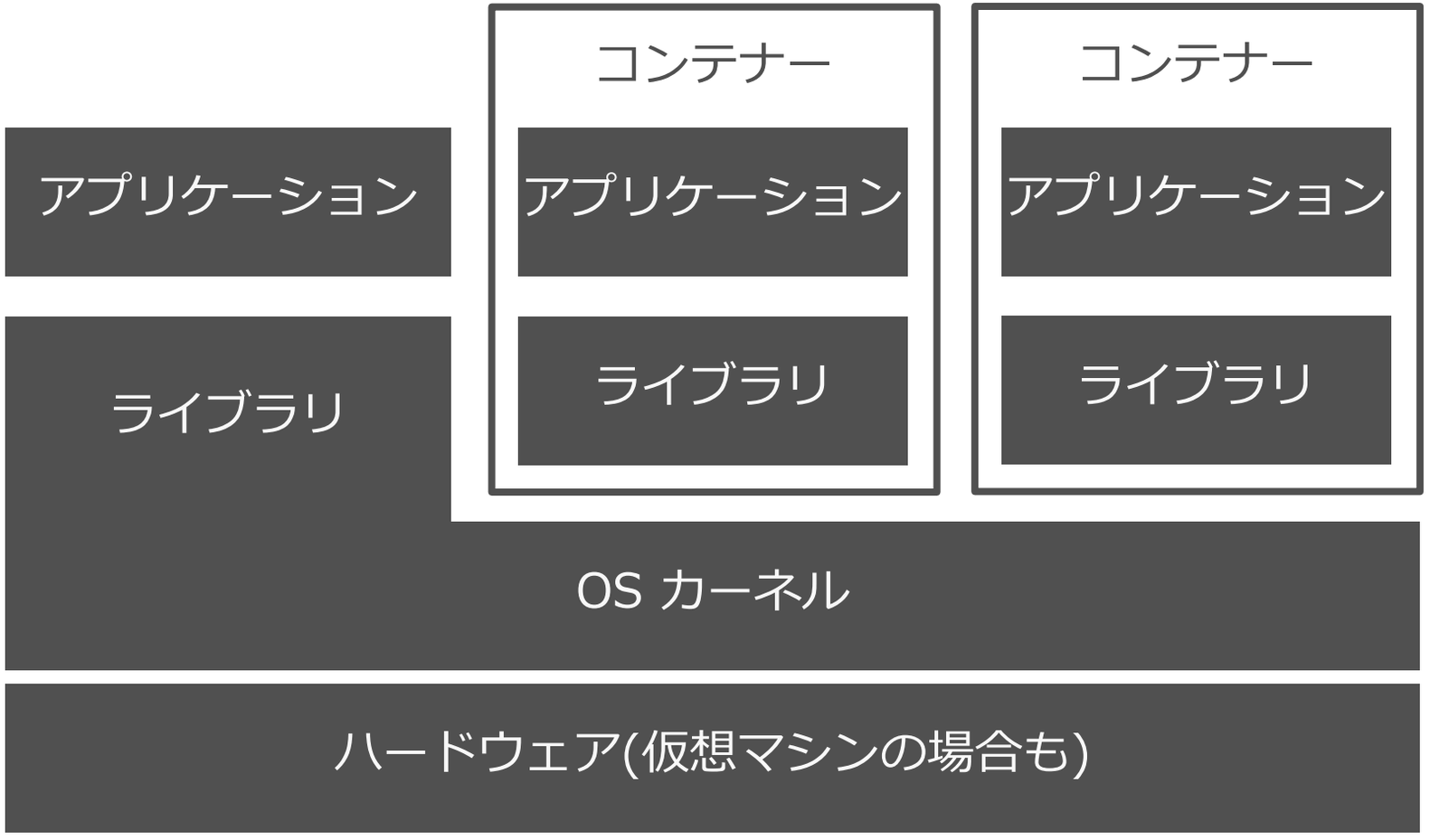
再現性が高い

アプリとライブラリをまとめてコンテナーへ閉じ込める
依存関係地獄からの解放

仮想化(ハイパーバイザー)

- ハードウェアを仮想化している
- 分離度は高い
- WindowsとLinuxを混在できる
- プロビジョニングの肌感覚は数分～数十分

コンテナー



- OSを論理的に分離している
- OSカーネルを共有
- WindowsとLinuxを混在できない
- アプリやライブラリをまとめてコンテナー化するので再現性が高い
- プロビジョニングの肌感覚は数秒

デモ
(コンテナー)

DockerCon 2017

- コンテナーをメジャーにした立役者がDocker
- 直近のカンファレンスには全世界から5,500人が参加
(参加してきました)
- VISA社、Metlife社の事例に注目が集まる
- アーリーアダプターの道具から、レガシーマイグレーションなどエンタープライズへの注力が鮮明に
- 米国ユーザーも抱えている課題は日本と似ている、違いは行動力

SRE(Site Reliability Engineering)

Google社の取り組みが発端

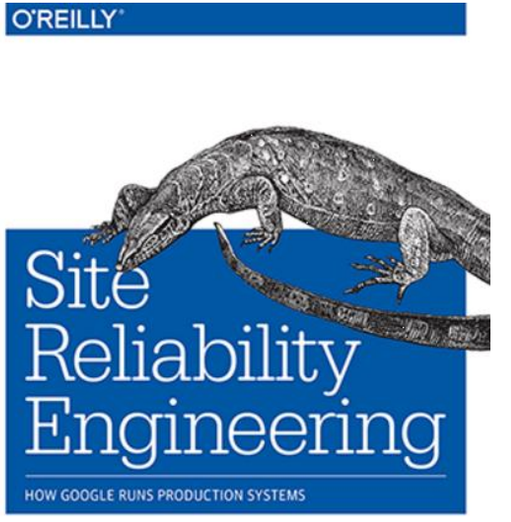
Google社のインフラエンジニアは「ソフトウェアエンジニア」
マイクロソフトもSREチーム を組織化

受動的 → 能動的

安定、かつ効率的なインフラを作り、維持改善するためにコードを書く
迅速なリソース割り当て、監視から復旧の流れ、etc
精神論や手作業依存の運用をせず「エンジニアリング」する

技術だけでなく、人や組織も重要

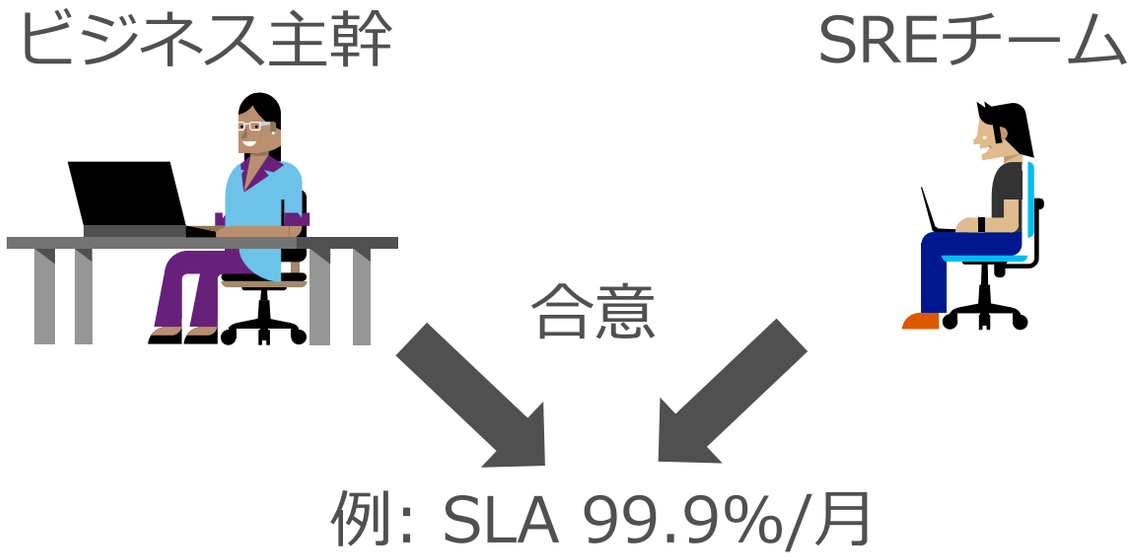
ビジネス側との合意 “Error Budget”
受動的な仕事に押しつぶされないようにマネージャーがコントロールする



Edited by Betsy Beyer, Chris Jones, Jennifer Petoff & Niall Murphy

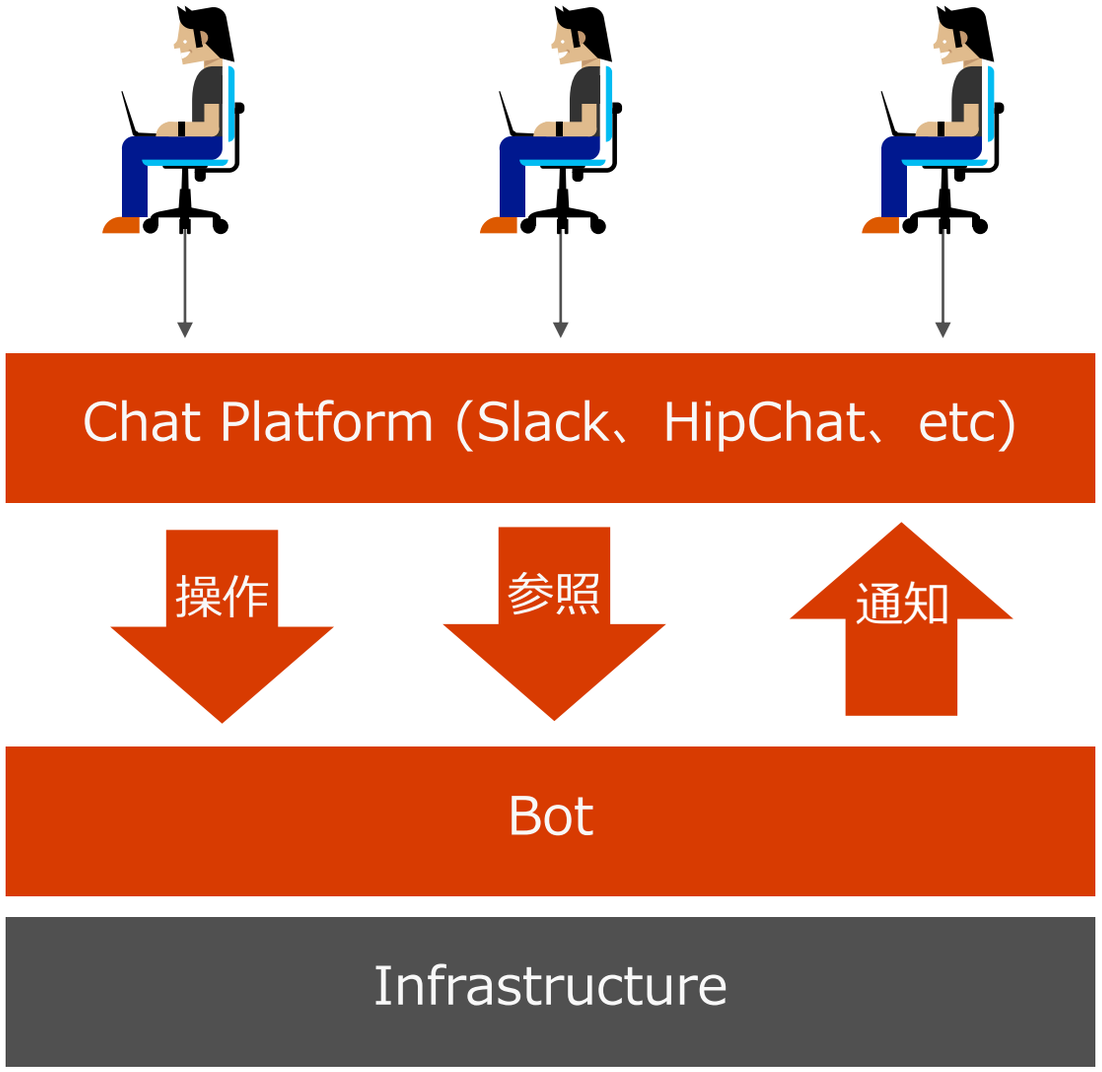
日本語版も出版予定とのこと
(http://www.publickey1.jp/blog/17/googlesite_reliability_engineering.html)

Error Budgetという考え方



- リスクはゼロにできない (故障、人的ミス、機能追加による不具合、etc)
- 高SLA = 高コストという認識をビジネス主幹と共有する
- サービスが止まった時間をError Budgetから取り崩す
- 残りのError Budgetが十分でない場合、機能追加やメンテナンスを控える

ChatOps

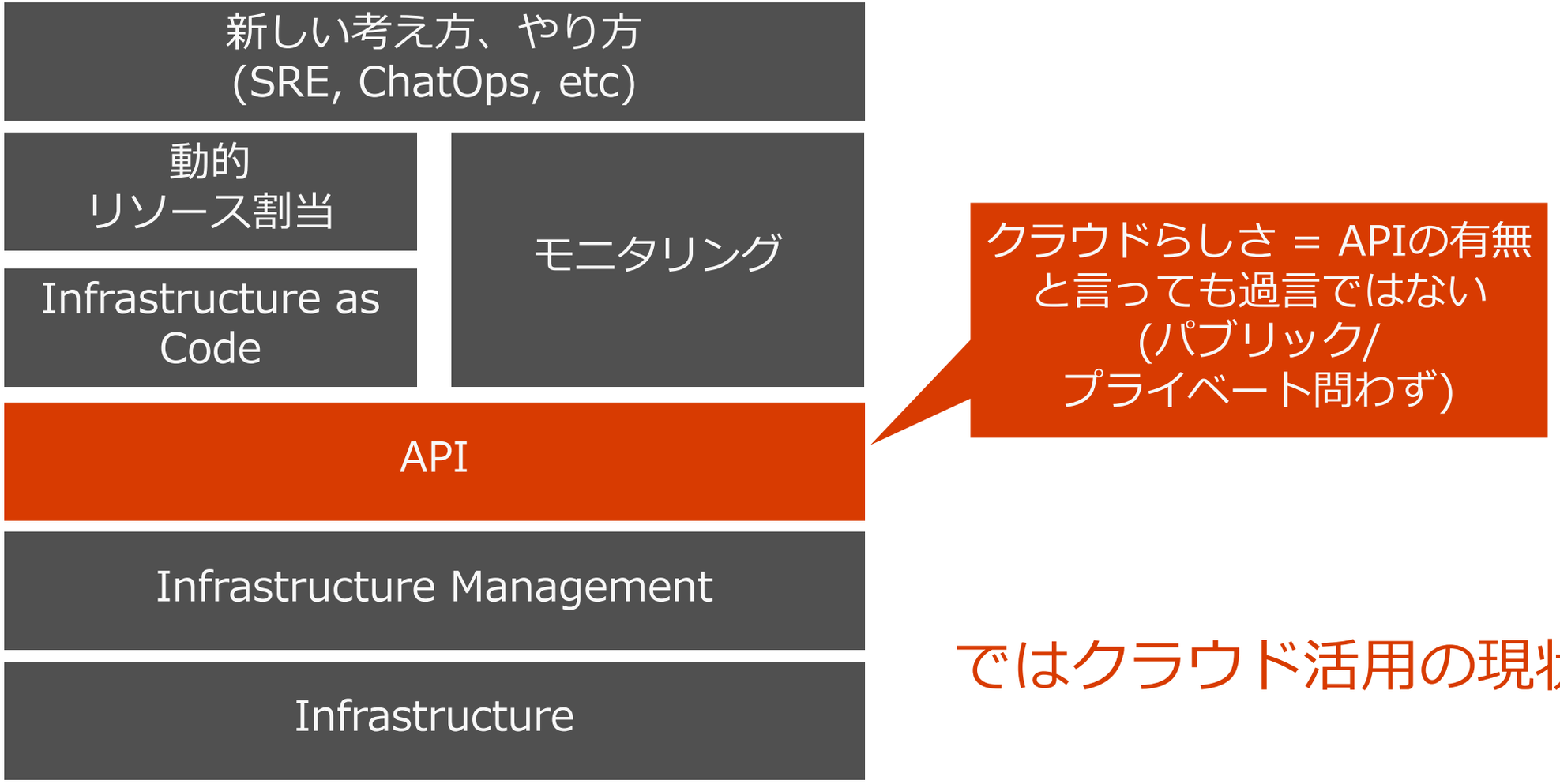


- 運用に関する連絡、情報交換の場としてChat Platformは有益
  - 即時性
  - 背景や文脈の共有
  - メールでは難しかった
- インフラの操作、参照、通知もChat上で
  - 定型作業の一元化、作業内容と結果共有
    (定型化、標準化できるものから)
  - 障害通知からの初動対応をシームレスに

デモ
(ChatOps)

新技術の浸透度と今後の展望

新インフラ技術の多くはクラウド(API)前提



ではクラウド活用の現状は?

クラウドの浸透度と今後の展望

エンタープライズITにおける旬の思考停止ワード

「ハイブリッド」

ハイブリッドとかけまして

「世界平和」と ときます

その心は?

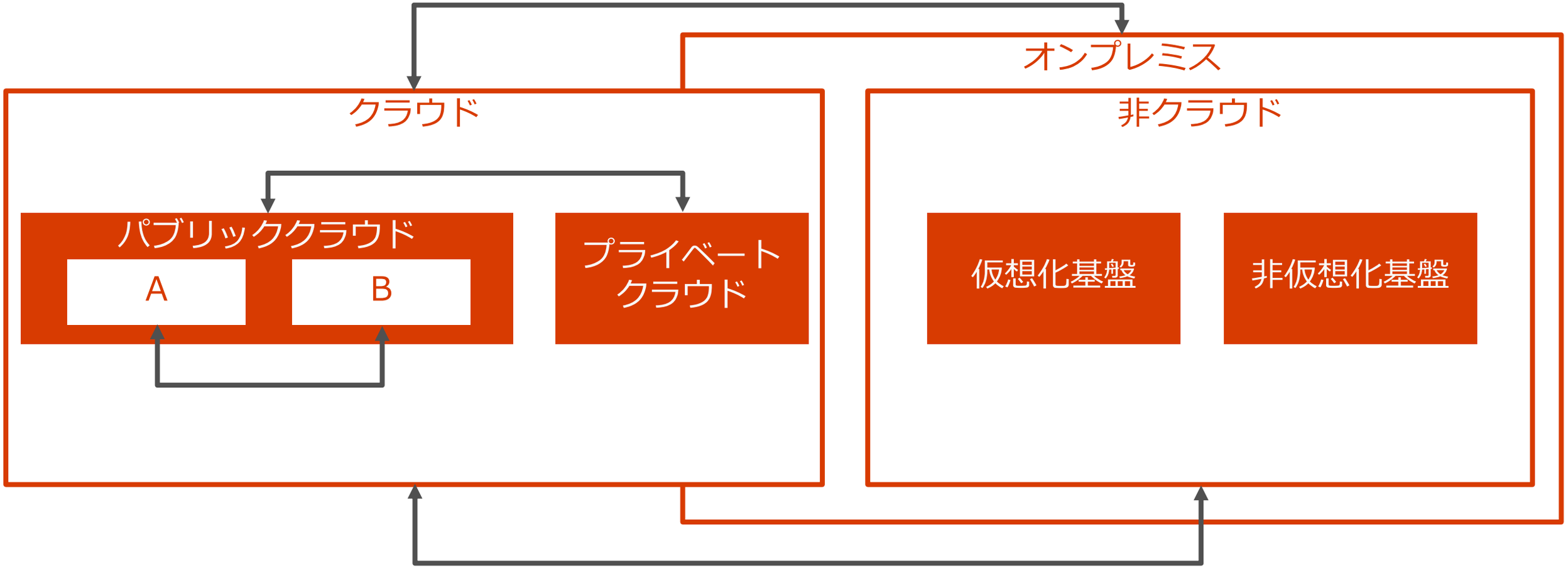
反対する人は少ない

そのいっぽうで

行動を起こす人も少ない

「誰かがやるだろう」「いずれそうなるだろう」
「とりあえず今のやり方を続けよう」

ハイブリッド?



ベンダーにも都合のいい言葉だが、その裏は…

https://twitter.com/cloud_opinion/status/840384647506866176

クラウド アンチパターン集

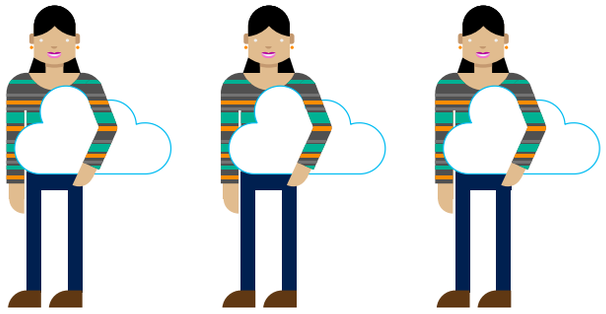
わたしが見てきたアンチパターンをいくつか

クラウド
アンチパターン集

アンチパターン① 結果的にハイブリッド

- 「いつかはそうなる」と言い、手を打たない
- 野良パブリッククラウドが増える
- コントロール不能なハイブリッド利用

パブリッククラウド便利よね



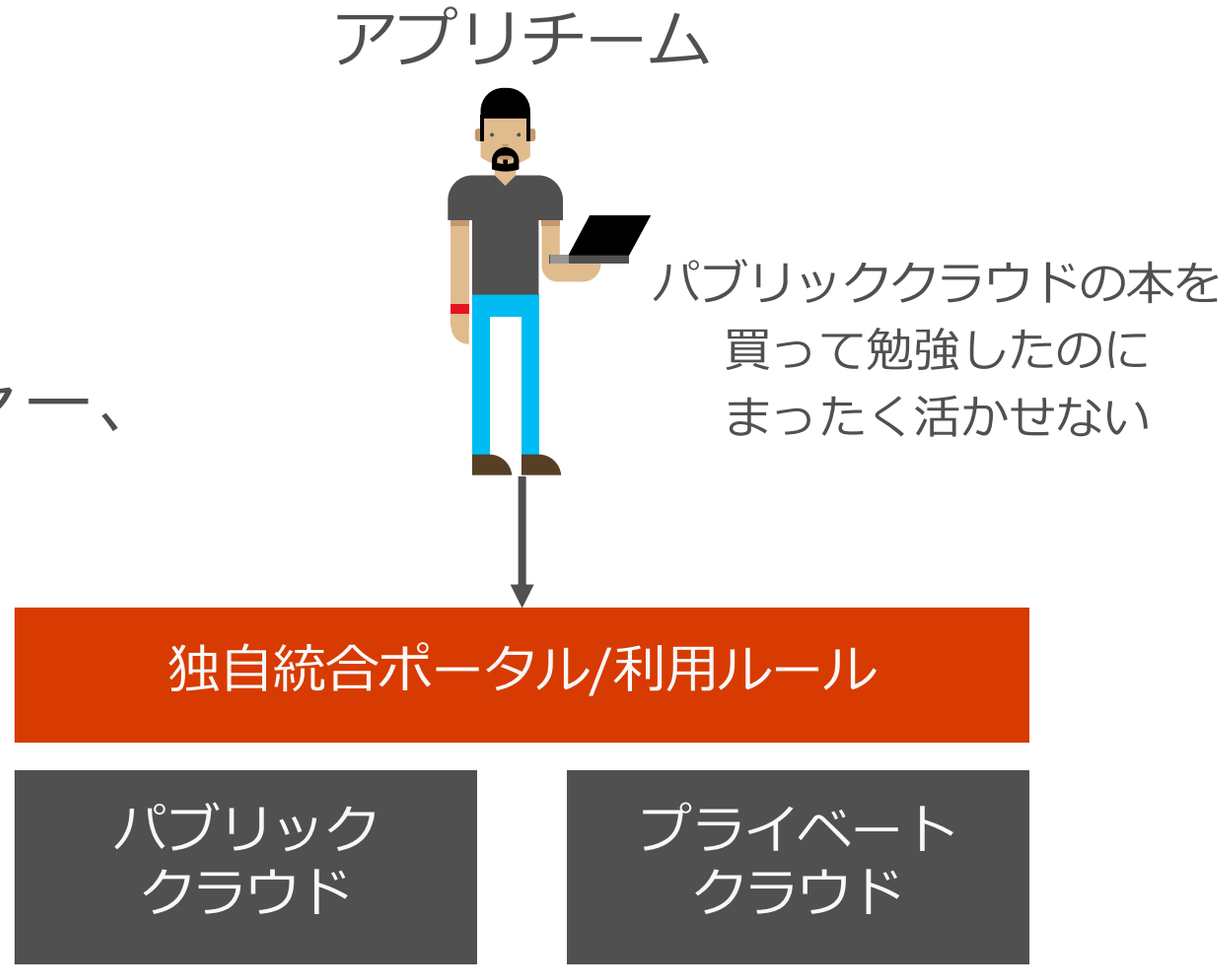
アプリチーム

そのうち
そのうち

インフラチーム

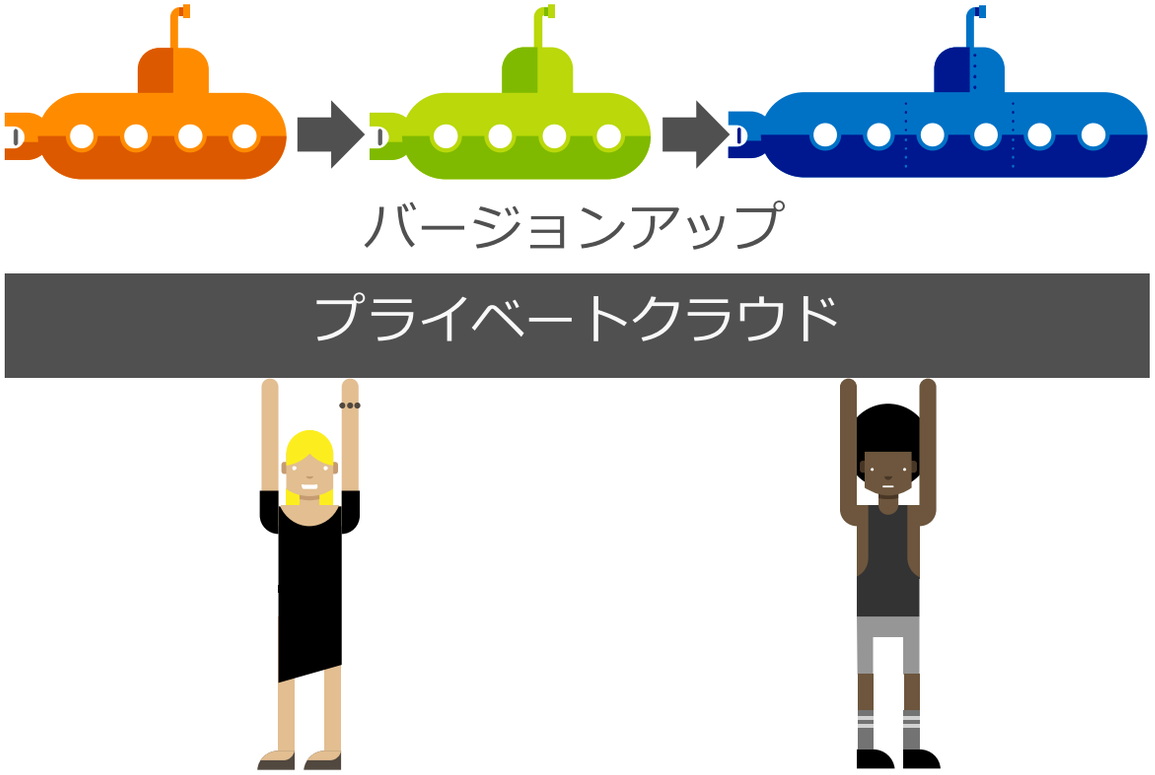
アンチパターン② 長所殺し

- 放置の逆パターン
- ガバナンス過剰で長所が消される
- 世にあふれる情報を活かせない
  (サンプルコード、アーキテクチャー、
  運用ノウハウ、etc)
- ベンダーの囲い込みパターンと
  考えるユーザーも多い
- 結果、野良に走る



アンチパターン③ プライベートで消耗

- 度重なるバージョンアップ
- 既存システムのポリシーに合わない
  (例: ネットワーク集約技術など)
- 小規模で労力に見合わない(*)
- 多くの成功事例は、大規模



(*)プライベートクラウドのリソース利用率が50%の時、管理者あたり2,000VM以上管理できなければ、パブリッククラウドのトータルコストを下回らない

(2016/10, 451 Research, "451 Research Finds OpenStack and Commercial Private Clouds Can Beat Public Cloud on Cost – But Only at Scale")

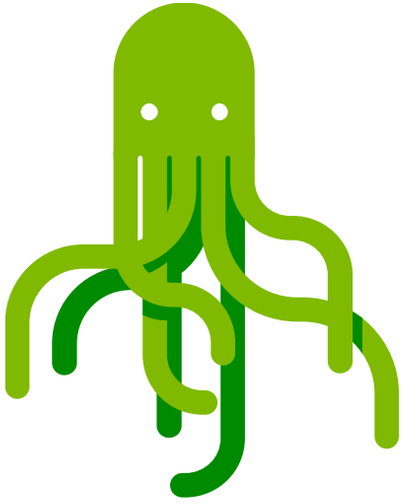
アンチパターンの根本にあること

アプリチームとインフラチームの温度差

そもそも評価基準が違う
インフラチームの物品コスト削減にアプリチームは無関心
オンプレしかなかった時代とくらべ、アプリチームには代替案がある

評価指標が物品コスト偏重である

構築/運用/改善/調達にかかる工数が考慮されていない



早すぎる全体最適

クラウドの知見を得る前に作ったルールで、自らを縛ってしまう

結論:これからです(実感)

改善、差別化のチャンスです

見えてきた
クラウド活用
成功パターン

じわじわ増える成功例

悲観的な話をしましたが

あくまで「これまで」の認識合わせです

成功例はじわじわ増えています

先進ユーザーは知見を得て、クラウド活用の次のステージへ

典型的な成功パターンをご紹介

６つのパターン

パブリック

①ビジネス側のニーズがある

- ビジネス側の要請
  - 大型イベントに間に合わせたい！
    (オリンピック、新製品発表、etc)
  - 競合他社の施策にカウンターを！
  - 予想外の法改正に対応！
- 開発やテストに可能な限り時間を使いたい
- 物品調達を待っていられない
- インフラもアプリ部品もオンデマンドで
  素早く調達できるパブリッククラウドを活用

②小さくはじめる

パブリック

- 知見がないのに大きな画は描けない
- 小さくとも短期間で成果が出そうなことからはじめる
  - 開発・検証環境の近代化
  - バックアップの改善(テープ廃止、DR実現)
  - 新技術検証の場(サンドボックス)
- ユーザーが手を動かすきっかけになる
  - ベンダーと対等に会話できるように

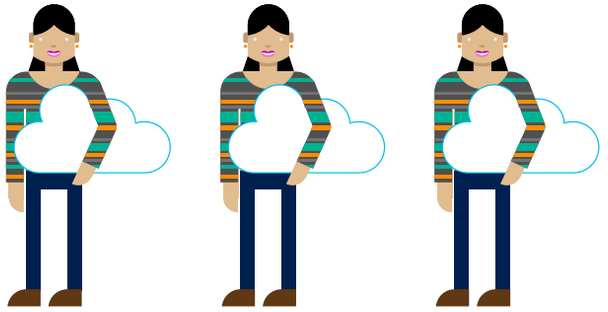
パブリック

③先回りする

- インフラチームはアプリチームとの会話を通じ、需要のあるクラウドサービスを知る
- クラウドサービスの推奨リストと利用ガイドラインを作成し、野良化を防ぐ
- リストにないサービスの要望は個別相談、将来もニーズがありそうであればリストに追加する

推奨リストと
ガイドライン

堂々と使えるね



アプリチーム

パブリック

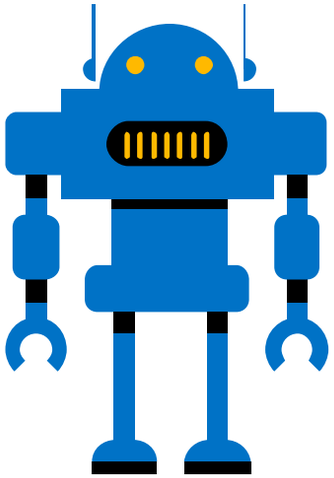
④コストのツボを押さえる

- 従量課金サービスの使い過ぎで破産、という大げさなニュース
- 実際にはアラート機能や使用量の上限設定ができる
- コストの支配的要素を抑えて、集中的にコントロールする
- 使っていないときは停止する
- 利用率を見てリサイズする

⑤クラウドだからできることをする

パブリック

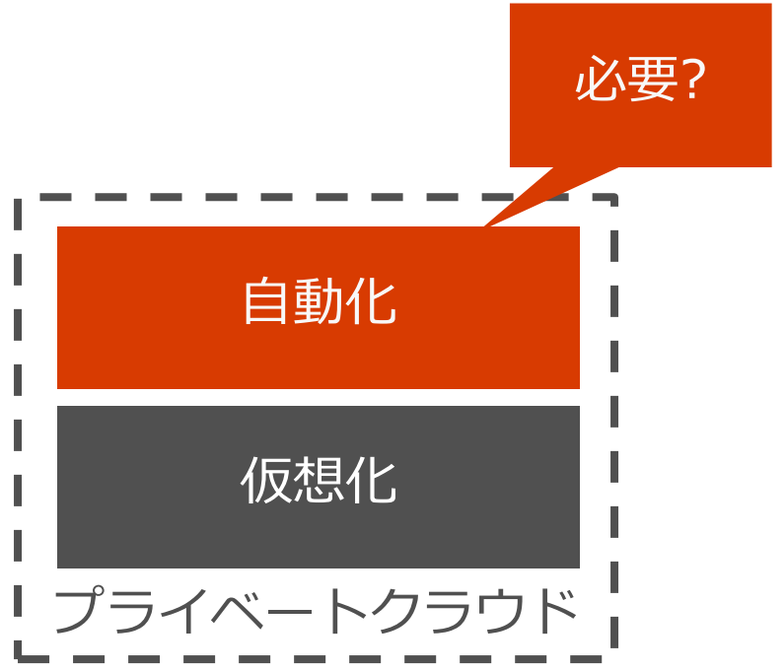
- 画像/音声認識、自然言語処理、機械学習、etc…
- 自前でインフラ、仕組み、データを「すぐに」準備することができるか?
- クラウドサービスで素早く検証し、効果があれば本格採用
- 少々試す程度なら無料のサービスも多い (例: Azure Cognitive Service Computer Vision APIは 5000Call/月まで無料)



⑥プライベートでハッスルしない

プライベート

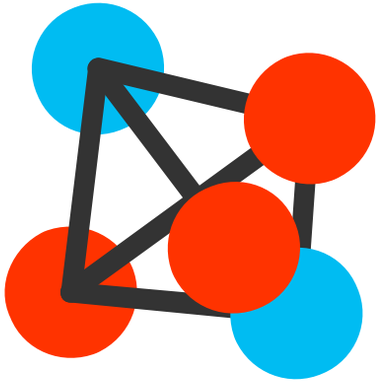
- 仮想化基盤とプライベートクラウドの大きな違いは自動化の仕組み
- 自動化には価値があるが、構築/維持コストも大
- それを必要としないシステムも多い
- 物品コスト重視なら仮想化にフォーカス



⑥プライベートでハッスルしない

プライベート

- 自動化が必要であれば、ベンダー提供のプライベートクラウドパッケージをなるべくカスタマイズしない(重要)で使うのも手
- 多くのプロジェクトが既存のルールやインフラに合わせるためのカスタマイズに多大な労力を費やしている
  - ネットワーク構成、技術、ポリシーの不一致
  - 既存ハードウェアベンダーとのお付き合い
- 戦略なきカスタマイズは長期的に高コスト要因になる



まとめ

お伝えしたかったこと

ITインフラ技術の進化がITのプロに与える影響

生産性の差は劇的に広がっている

注目の技術とコンセプト

Infrastructure as Code、動的リソース見直し、コンテナー、SRE、ChatOps
その多くはクラウド(API)前提

焦らず着実に

ビジネス価値を出しやすい課題から、小さくはじめる

Microsoft