データセンターの自律化に向けたソフトウェアの要件と設計

株式会社クルウィット
井澤 志充

2016/02/18 (木)
DataCenter とソフトウェア開発ワークショップ

@松江テルサ

自己紹介

・井澤 志充 (いざわゆきみつ)

・(株)クルウィット

　・委託研究・自社サービス開発など

SITEVISOR

前提

・データセンター内には多種・多量な制御対象物が存在する。

データセンターに求められる特性

- 耐規模性
- 可用性
- Etc..

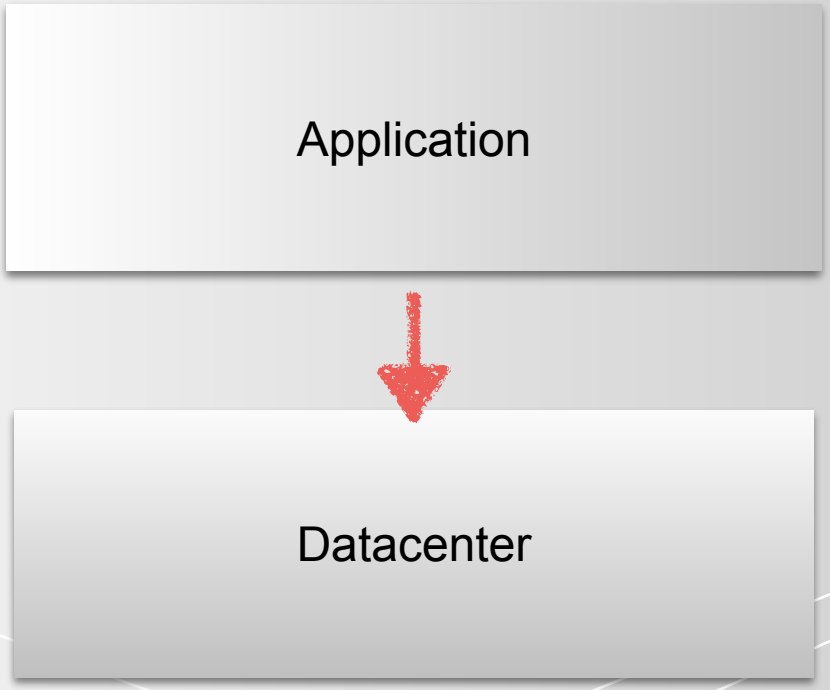
耐規模性

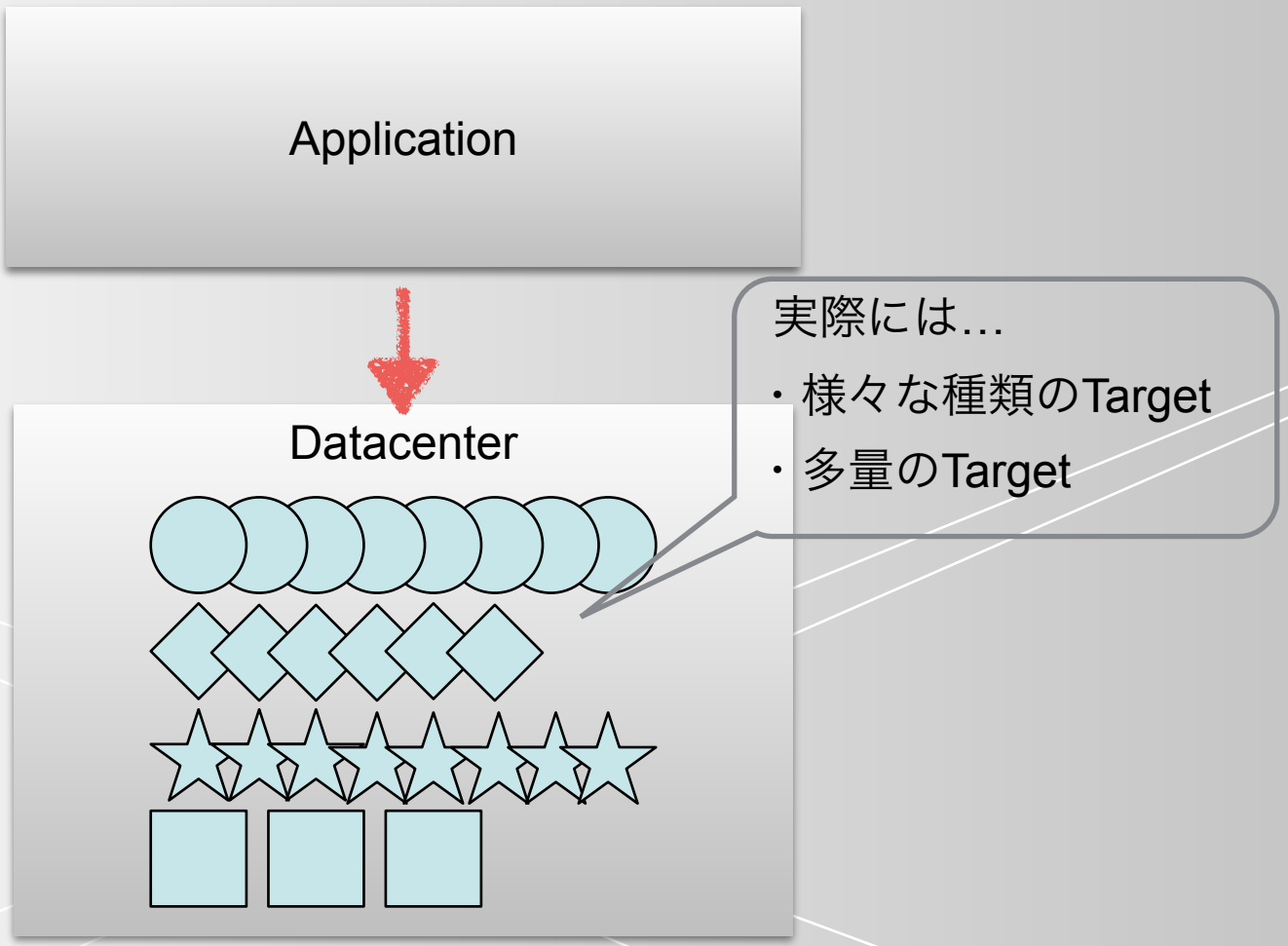
- 種類に対する耐規模性
- 数(量)に対する耐規模性

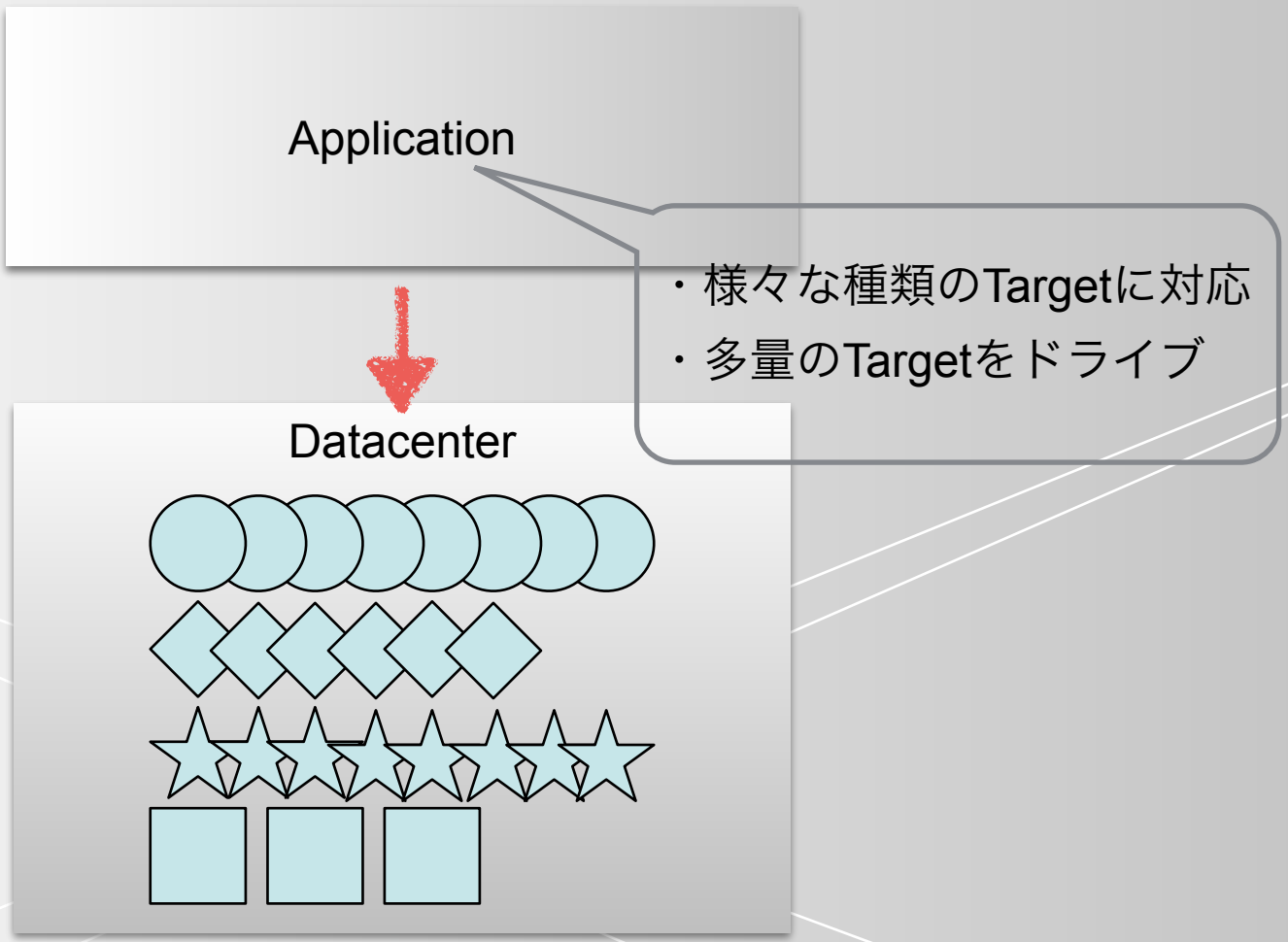
可用性

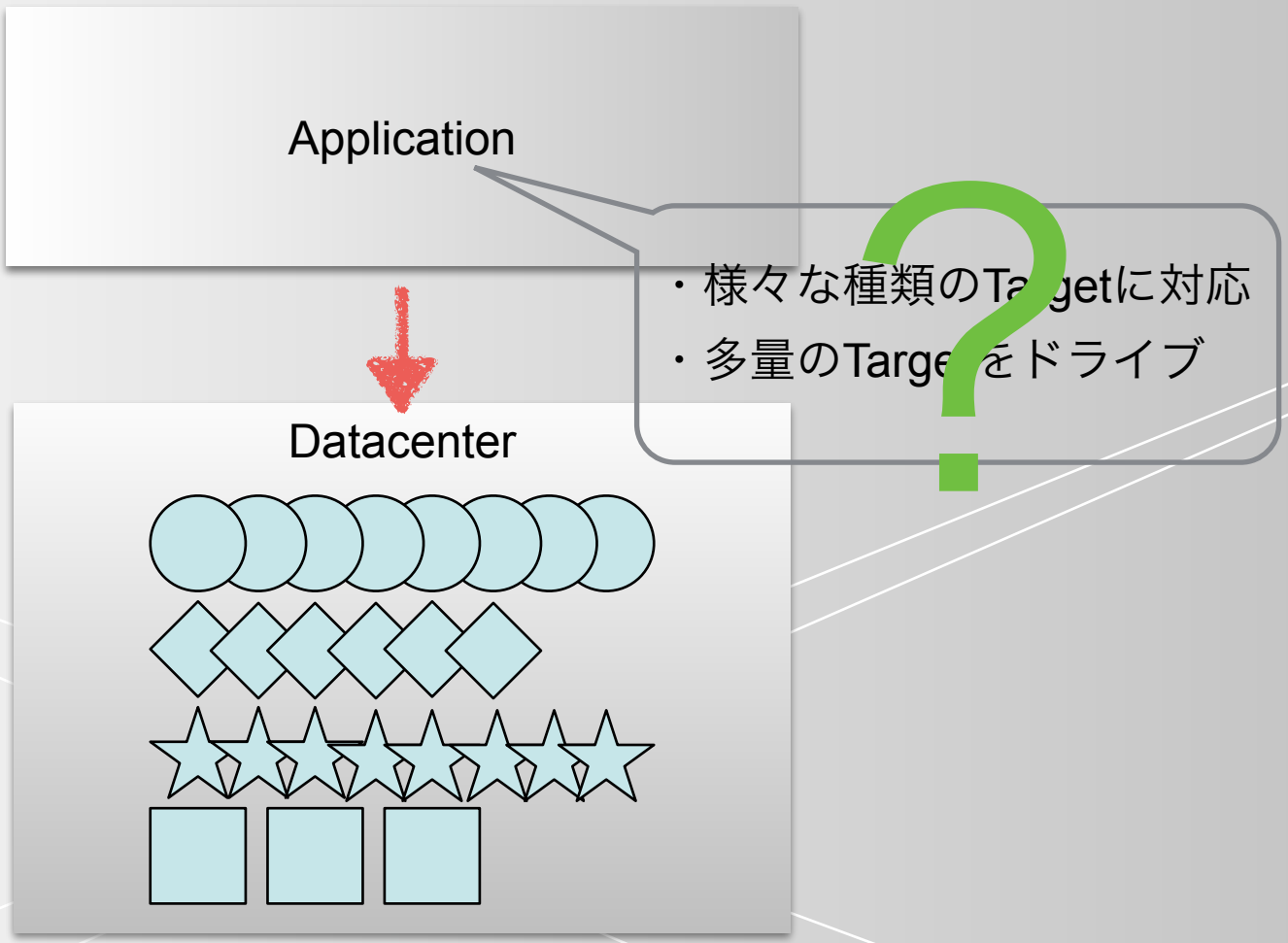
・全部のシステムを停めなくてもシステム構成変更可能であること/動的に入れ替え可能であること

ひとくちにDatacenterを扱うAppといっても




Application
Datacenter
実際には…
・様々な種類のTarget
・多量のTarget

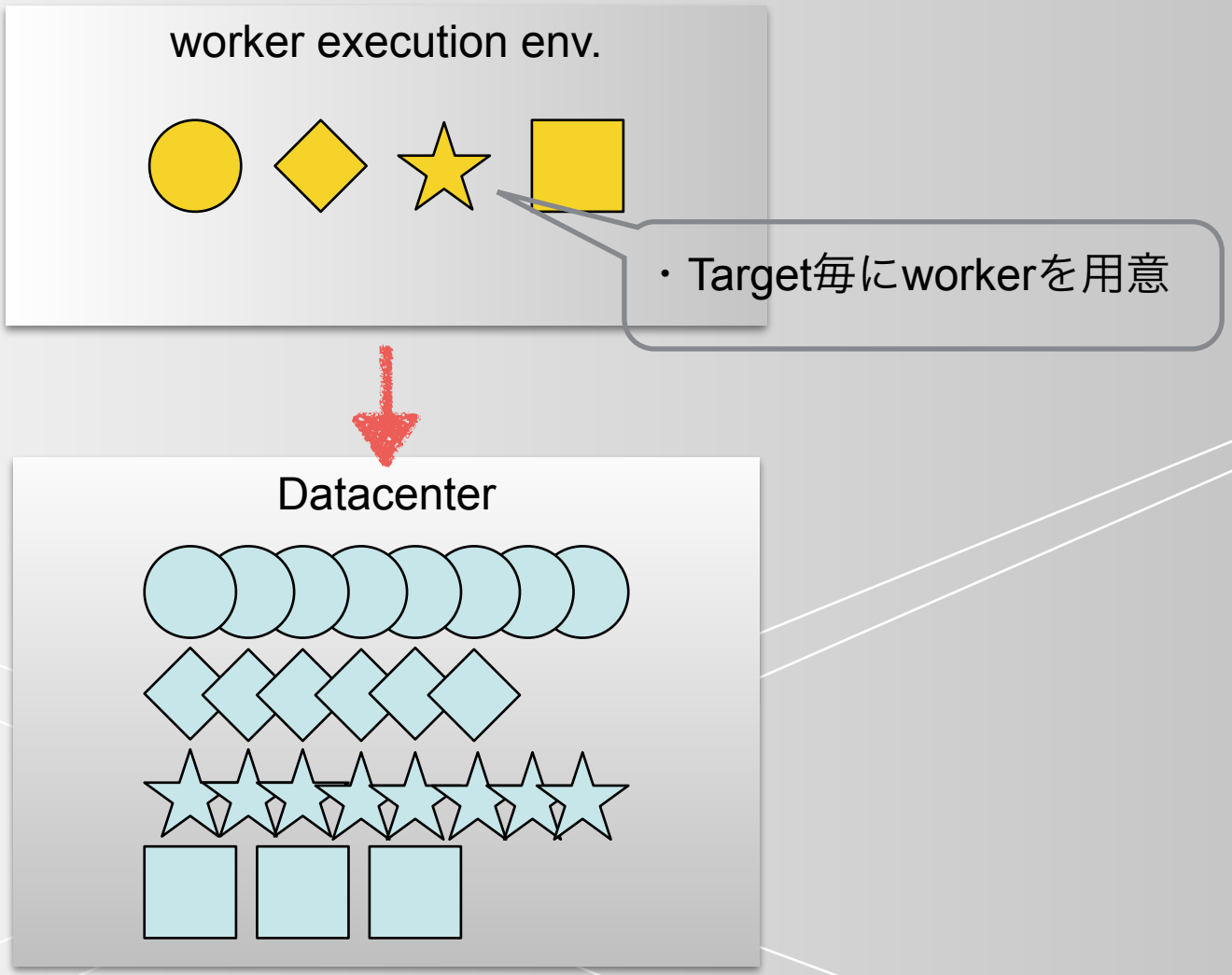

Application
・様々な種類のTargetに対応
・多量のTargetをドライブ
Datacenter

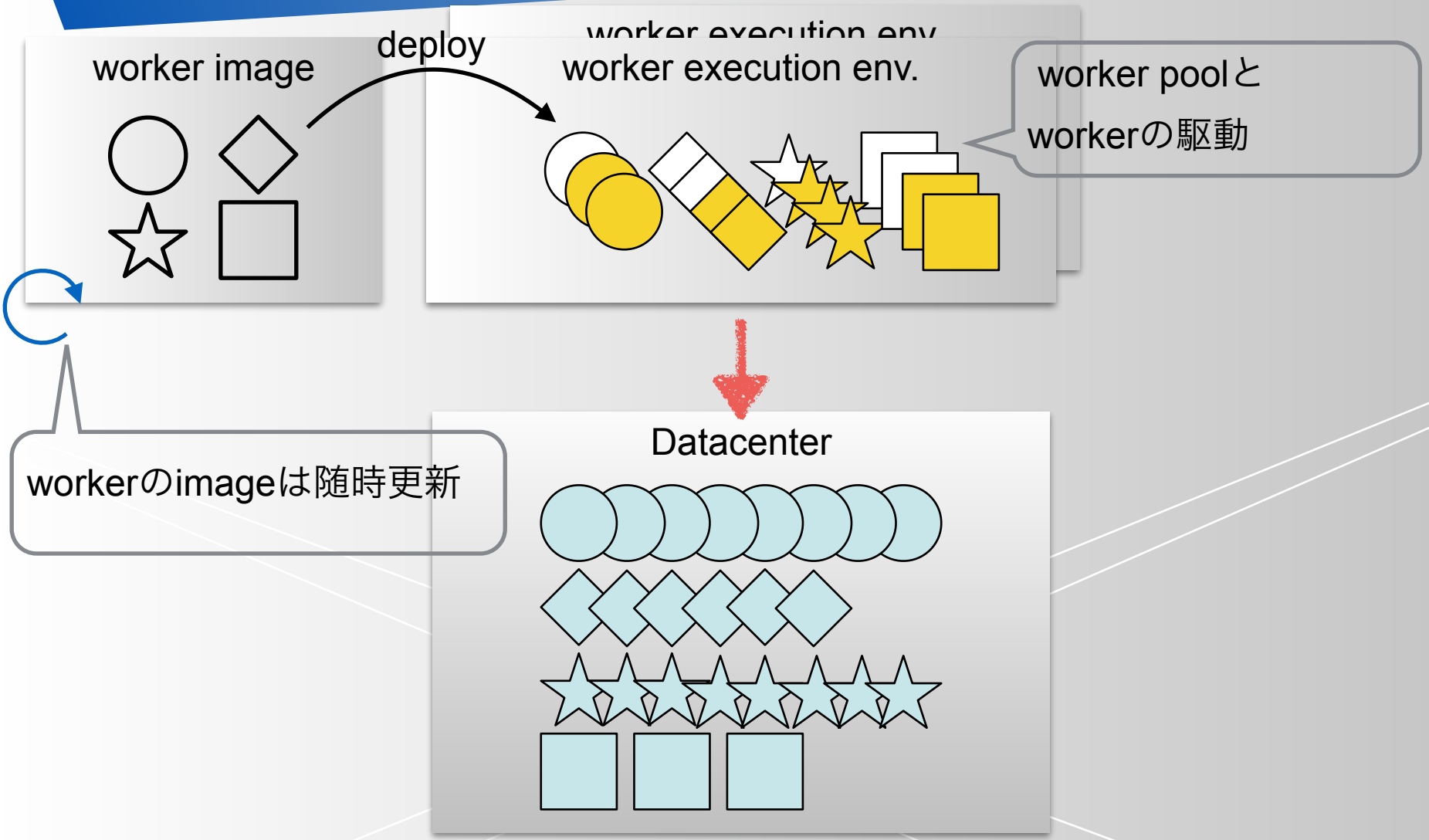

Application
・様々な種類のTargetに対応
・多量のTargetをドライブ
Datacenter

Targetが扱いづらい理由

- 動的に多種多様(故障/更新/Etc..)
  - 動的だから更新が必要
- 超すごい1つのソフトは、ソフトの更新や継続運用し続けるのが大変難しい。

workerモデルを導入




worker image
deploy
worker execution env.
worker poolと
workerの駆動
workerのimageは随時更新
Datacenter

- 導入したモデル:
  - 細かいアプリケーションの集合
  - 構成要素自身はpool/reserveされている
  - workerをupdateし続ける

workerモデル

- 実現したこと
  - 1.update可能性
  - 2.partial update可能性
- 用途に応じてdeployできる
  - 3.多様性の導入
- 動作環境自身を可変にする
  - 4.スケーラビリティの導入
  - 4.5 環境自身の可用性(実行環境自身の部分停止/部分更新)

耐規模性(再)

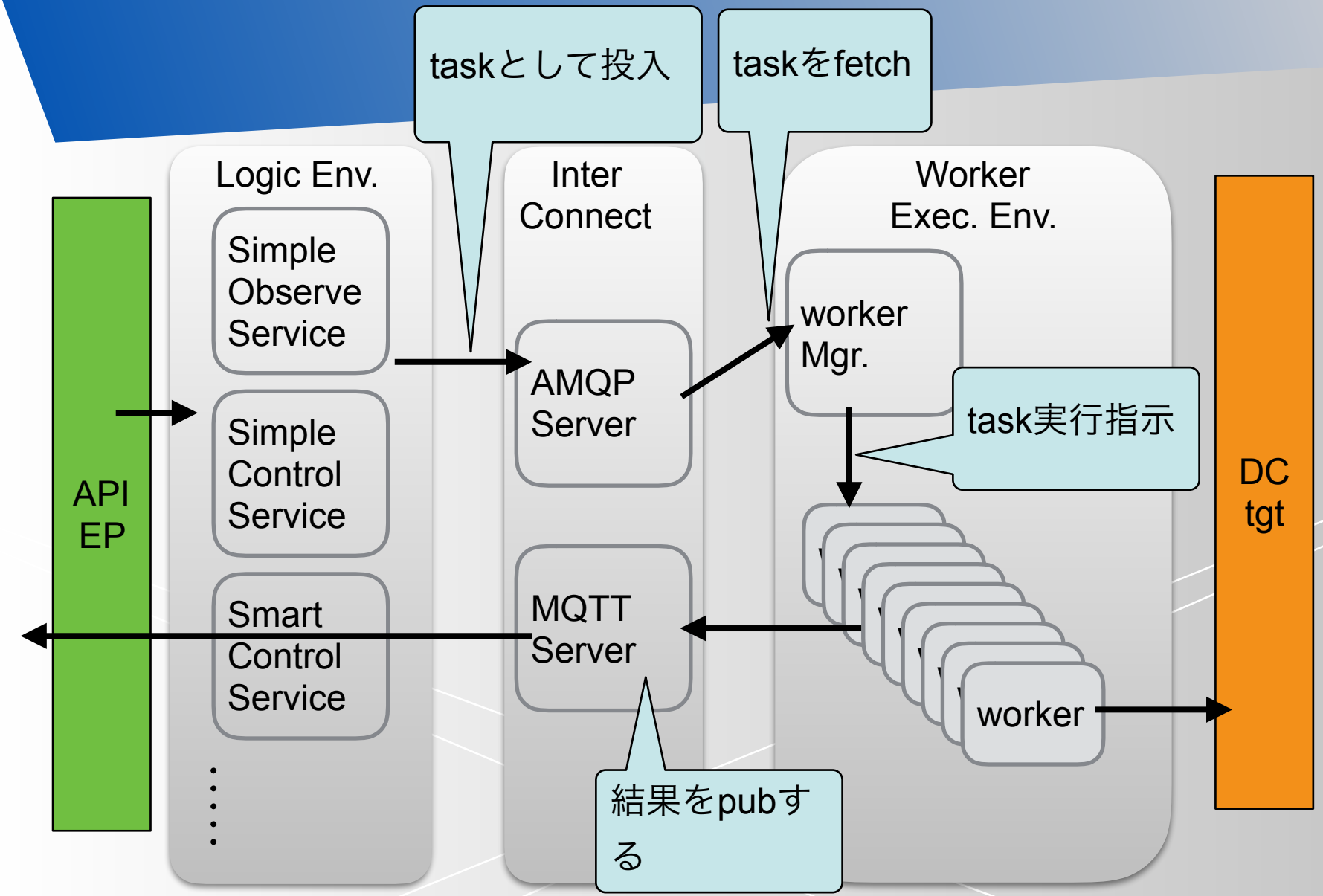
- 種類に対する耐規模性
  - workerの種類を増やすことで対応
- 数(量)に対する耐規模性
  - workerのインスタンスを増やすことで対応

PSにおけるworkerの実現

- dockerを利用する
- workerはdockerコンテナとして実現する
- dockerコンテナは汎用のインスタンス実行環境下(k8s)で実行
- workerの寿命とシステムの寿命を個別に扱う
  - サービスロジック(比較的長生き)とworker(短命/泡沫的)
- システムは動き続けるけれども、workerの中身は動的にアップグレードし続ける仕組みを実現

PSのモジュール構成

- LE (Logic Environment)
  - PS利用者へのサービス(ex,APIエンドポイント)を提供(比較的長生き)、Taskを生成
- IC (Inter Connect)
  - LE からのTaskをWEEのAPIエンドポイントへ処理依頼
  - 非同期通信に限定
- WEE (Worker Execution Environment)
  - ICからのTaskを受け、suitableなworkerを選定して、workerにTaskを実行させる
  - 短命/泡沫的


taskとして投入
taskをfetch
Logic Env.
Inter Connect
Worker Exec. Env.
API EP
Simple Observe Service
Simple Control Service
Smart Control Service
AMQP Server
MQTT Server
worker Mgr.
task実行指示
worker
DC tgt
結果をpubする

まとめ

- データセンターを構成する多種多量のターゲットを扱う際に求められる特性をまとめた
  - 可用性
  - 耐規模性
- そのためにworkerモデルを導入した
- workerモデルを用いたPSのモジュール構成紹介