3者モデルに基づく開発運用形態の分類 ～理想的なDevOpsにむけて～

高　橋　昂　平[†1]

クラウド時代のシステムの開発運用では，様々な要求に柔軟で迅速な対応ができる体制づくりが大切である．本稿では，システムの開発運用形態をそれに関わる 3 者と，その 3 者間の関係性に着目してモデル化を行う．提案モデルを用いてその形態を性質づけ，より良い体制づくりの一助としたい．

Characterizing of System Development and Operation Based on 3-party Model

Kohei Takahashi[†1]

In the system development and operation of cloud computing era, it is important to adapt addressing various requirements rapidly and flexibly. This paper models the system development and operation style by focusing on the 3 parties and their relationships.

1. はじめに

クラウド環境の成熟や様々な仮想化技術の普及により，システムが実行・運用されるプラットフォームやインフラの姿も変容してきている．技術の変化や興亡の速度も増し，以前より非常に短いスパンでの開発・リリースが求められる上，新たな技術や時代に適合した品質や安全性の確保も求められ，システムの開発手法も変化しつつある．

どのような開発手法を採用するとしても，その手法を活かすためには，実際に開発を行う者だけでなく，そのシステムを運用していく者やそもそもそのシステムを望んだ者の間で，お互いの理解や，問題意識の共有ができていなければ，空回りに終わってしまう．

顧客の要求を満たす機能の迅速なリリースや，新たな脅威に対して早急な安全性の確保を行うには，開発と運用が距離を縮め，システムのスムーズな受け渡しや要求に応じた迅速なシステム更新を行える体制作りが必要不可欠である．こうした背景のもと DevOps が提案されている．これは，開発（Development）と運用（Operations）が協力し，ビジネス要求に対して，より柔軟に，スピーディに対応できるシステムを作り上げるためのプラクティス[1)] のことである．

現状の多くのシステム開発運用形態では開発を行う部門と運用を行う部門がほとんど独立で機能しており，理想的な DevOps の実施には程遠い状態である．

†1 神戸大学大学院システム情報学研究科
Graduate School of System Informatics, Kobe Univ.

そこで本稿では，現状のシステム開発運用形態の改善の助けになるよう，現状を正しく把握・整理できるようなモデルを提案する．提案モデルでは，システムの開発運用に関わる 3 つのアクタ（Customer（機能を求める者），Operator（機能を運用する者），Developer（機能を実現する者）の 3 者）と，それぞれの間にある関係性に着目して開発運用形態を性質づけ，関係性のバリエーションによる 8 つの分類について考察する．

2. 開発運用形態を表現する 3 者モデル

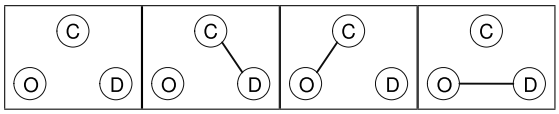
2.1 定　　義

提案モデルでは，機能を求める者を *Customer*，機能を運用する者を *Operator*，機能を実現する者を *Developer* とし，これら 3 者のアクタがどのような関係性にあるかを表す．モデルの表現においては，アクタを丸型のアイコンで表し，アイコン中に C, O, D の頭文字をつける．それぞれ Customer, Operator, Developer である．2 つのアクタが友好的な連携がとれている場合（**連携関係**と呼ぶ），そのアクタ同士を線で結ぶ．連携関係にないアクタ同士は一方的な関係であったり，それぞれが自己完結的であることを示す．3 種類のアクタ間に 3 つの連携関係が定義できるため，可能な全パターンを洗い出すと，**図 1** のようになる．

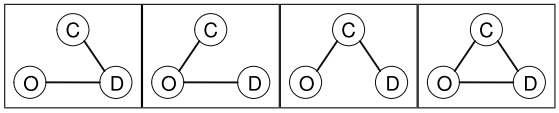
2.2 3 者モデルによる開発運用形態の分類

(a) 3 者分離型

3 者分離型は，C, O, D の各々がほぼ独立に動く形態である．例えば，オープンソース・ソフトウェアのコミュニティや企業内での新たなソフトウェアの試作



(a) 3 者分離型 (b) C-D 連携型 (c) C-O 連携型 (d) O-D 連携型



(e) O-D, C-D 連携型 (f) C-O, O-D 連携型 (g) C-O, C-D 連携型 (h) 3 者連携型

図 1 3 者モデルによる分類

段階やテスト期間，個人や小規模なチームでの開発運用の形態である．もしくは 2.2 節の O-D 連携型において，O と D の連携が取れておらず，迅速なリリースの継続が難しい形態である．

(b) C-D 連携型

C-D 連携型では，C-D 間に友好的な連携があるが，O は C, D 双方とは連携が取れていない形態である．C と D が密に連携を取りながらシステムを開発し，完成後に O に運用を委譲するような形態で，現在の多くの受注型システム開発の形態だと考えられる．

(c) C-O 連携型

C-O 連携型では，C-O 間に友好的な連携があるが，D は C, O 双方とは連携が取れていない形態である．既存のシステムを利用する形態など，D との関わりが薄い場合で，C からの要求や苦情がシステムに直接反映されにくく，運用方針や場当たり的な判断で対処していくことになる形態である．

(d) O-D 連携型

O-D 連携型では，O-D 間に友好的な連携があるが，C は O, D 双方と連携が取れていない形態である．C として明確な顧客や発注者が居ない場合では理想的な形態にあたる．この時 C は一般のサービス利用ユーザもしくは想定顧客となる．O と D の協力体制が確立されており，発生する要求に対し柔軟かつ迅速に対応でき，素早いリリースを継続的に行っていくことができる．元来の DevOps が成功している形にあたる．

C として明確な者が居る場合には，C の想定とは異なるシステムが構築され運用されている可能性が高く，その要求や意見があまりシステムに反映されないような形態にあたる．

(e) O-D, C-D 連携型

O-D, C-D 連携型では，C は D と O の双方と連携関係にあるが，O-D 間では連携が取れていない形態である．C-D 間，O-D 間においてそれぞれ連携が取れているため，双方からの要求を受けて D が板挟みの状態に陥る可能性が高い．

(f) C-O, O-D 連携型

C-O, O-D 連携型では，O は C と D の双方と連携関係にあるが，C-D 間では連携が取れていない形態である．C の知識不足やその他の事情により，強い要求や明確な要求がない場合などで，O の意見が通りやすくなっており，D は運用に都合の良くなる仕様や環境を強いられる形態．本質的な C 側の要求や，D 視点における「より良いシステム」よりも O が運用をしやすくなることが優先される．

(g) C-O, C-D 連携型

C-O, C-D 連携型では，C は O, D の双方と連携関係にあるが，O-D 間では連携が取れていない形態である．O と D にそれぞれに C からの要求は届くが，O と D の間では問題意識の共有がうまく行かないため，対応の仕方や優先度がちぐはぐになりやすい．

(h) 3 者連携型

3 者連携型は，3 者間それぞれで連携が取れた理想的な形態である．C からの要求や突発的な事象に対して非常に柔軟かつ迅速に対応することができる．3 者で同じ方向へ向かってシステムの開発運用を行っていくことができる形態である．

3. 議　　論

2.2 節の分類は，特定の形態が絶対的に優れているといったものではない．本モデルによる分類の用途として，まずは現状の開発運用形態を正しく把握し分析する助けになること．また，モデル化によって問題意識の共有を容易にすること．その上で，理想とする形態を想定し，組織やプロジェクトに沿った DevOps の施策を取る一助となることが挙げられる．さらに，各形態特有の問題に対して，有効な対策や DevOps の施策を体系的に整理していくことも可能になると考える．

今後の課題としては，現状ではアクタ間の関係性は有か無のどちらかであるが，より強固な関係性を持つ場合や，提案した 3 者以外のアクタなど，より詳細な表現も検討していくべきだと考えている．

本ワークショップでは，モデルの妥当性や利用方法，発展可能性について議論したい．

参　考　文　献

1) @IT: いまさら聞けない「DevOps」. http://www.atmarkit.co.jp/ait/articles/1307/02/news002.html.