

Title	分散コンピューティング基盤における 宣言的構成管理の適用に関する研究
Author(s)	真壁, 徹
Citation	
Issue Date	2021-03
Type	Thesis or Dissertation
Text version	author
URL	http://hdl.handle.net/10119/17144
Rights	
Description	Supervisor:篠田 陽一, 先端科学技術研究科, 修士 (情報科学)

クラウドコンピューティングをはじめとする分散コンピューティングの普及に伴い、基盤を構成するサーバやネットワークデバイス、ストレージなどのリソースは増加、多様化している。そして、その運用は専門性を持つ技術者への依存を強めている。そこで、リソースの状態を意識する必要なく、また手続きを逐一指示せずとも、あるべき姿を定義すれば基盤をその通りに設定、維持する宣言的構成管理が注目されている。

宣言的構成管理は新しい概念ではなく、制約、論理プログラミングを基礎とした先行研究がある。しかし、管理対象の持つ自律性との齟齬、構成を決定する計算量と時間、リソース操作に要する時間、動作中のリソースを操作するリスク、制約、論理プログラミングを習得する難しさなど、多くの課題があった。

一方、オープンソースの基盤ソフトウェアである Kubernetes は、先行研究で指摘された宣言的構成管理の主要な課題を、結果整合を認めるなどトレードオフを受け入れて解決した。反面、普及の過程でアクシデントも散見され、安全性や回復性に関する負の影響も疑われる。

Kubernetes 上で動作するアプリケーションの回復性については、コンテナの回復時間に注目した研究がある。しかし、それを支える構造は追究されていない。本研究は Kubernetes を題材とし、その構造分析を通じ課題や論点、展望を示すことで、今後の宣言的構成管理の研究と実践へ貢献する。

Kubernetes は構成管理と回復機能を一体としている。回復機能は安全性を支える主要素である。よって、安全性分析を通じて、構成管理に関する特徴や洞察も得られると期待できる。本研究では手法に STPA(System-Theoretic Accident Model and Processes) を選択し、構造と安全性を分析した。Kubernetes は多様な要素で構成されているため、構成要素の相互作用に注目する STPA はその分析に適している。

本研究では STPA の手順に従い、まず全体とサブシステムの構造を分析するために制御構造図(コントロールストラクチャ)を作成し、非安全なコントロールアクションを導いた。次に、その原因であるハザードシナリオを得た。加えて、得たハザードシナリオで Kubernetes の障害事例を分類し、その妥当性を評価した。

分析の成果物は Kubernetes の利用者がその制御構造やハザードシナリオを理解する助けとなり、リスク評価や問題対処、カオスエンジニアリングに貢献する。加えて、分析から導いた論点は、Kubernetes に限らず宣言的構成管理の適用、活用の研究と実践において価値がある。中でも2つの特記すべき論点がある。

まず1つ目は、リソース作成、変更時の入力内容に対する妥当性検証の必要性である。ハザードシナリオと障害事例の分類において、妥当性検証の不

足を原因とするアクシデントの数は顕著であった。例えば、リソース量の制限や優先度設定を行わずに投入されたアプリケーションが、リソースを共有する他のアプリケーションやシステム要素と競合し、ハザードに繋がっている。Kubernetes が結果整合を受け入れ、宣言の入力時検証を任意とした負の影響を認める。

2つ目の特記すべき論点は、アプリケーションの再構成耐性である。宣言的構成管理において構成管理機能は、あるべき姿を維持するため、主導的にリソースを再構成する。よってアプリケーションの再構成耐性は重要な論点である。障害が発生することを前提にアプリケーションを設計する「Design for Failure」という考えがあるが、宣言的構成管理においては、加えて再構成イベントも考慮すべきである。つまり障害や再構成を原因とした混乱状態に耐える設計「Design for Chaos」が求められる。実装例の1つにリクエストの再試行があるが、本研究では、その有無で再構成耐性に影響があることを検証した。

アプリケーションの再構成耐性の確保は、検討を基盤の範囲に閉じていては解決できない。例えば、アプリケーションへ再試行機能などを実装するだけでなく、そのテストも課題となる。再構成耐性はアプリケーションを新規作成、変更時にテストするだけでは十分でない。リソース構成が変化していくことを考慮し、継続的に Chaos testing を行うべきである。

今後はネットワークやストレージなど、現時点で宣言的構成管理の適用に課題があるリソースへと管理対象は広がるであろう。すでにクラウドサービス事業者では取り組みが始まっている。しかしリソース間の依存関係や変更時の影響の大きさなどから、変更不可能な属性を設けるなど制約はある。この制約の解決は将来の課題である。