Kubernetesを利用したRTC開発フレームワーク

○黒瀬　竜一（産業技術総合研究所），矢後　聡識（パナソニック アドバンストテクノロジー）

RTC development framework using Kubernetes

○ Ryoichi KUROSE (AIST) and Akinori YAGO (PAD)

Abstract : Adapting Kubernetes to RT-Middleware makes it possible to update RT-Component programs, start / stop, and distribute programs to actual robots. This paper describes a framework that uses these functions to accelerate RTC development.

1. 緒言

RT ミドルウェア [1] は分散処理によるロボットシステムの構築を可能とする。しかしながらこうした分散処理システムは、使用される PC の数に依存して複雑になる。

ソフトウェアの開発では常に同じテスト環境構築が構築され、テスト実行やデバッグは誰が行っても同じ挙動を再現することが望ましい。しかしながら分散システムの開発において、PC 群に搭載される様々なソフトウェアのバージョンを開発機間で統一することは大変な手間となる。さらに、テスト実行の手順やタイミングを合わせることも課題の一つとなる。

本稿ではこの課題を解決するためにクラウドを実現するためのツールである Docker [2] と Kubernetes[3] を RT ミドルウェアに適応することで、新しい RTC の開発フレームワークを構築する。

2. フレームワークの提案

2.1 RTC 状態管理の拡張

RTC は自身を終了させる機能を持つが、自身の起動とアップデートはできない[*1]。一方で Kubernetes が提供する主な機能に、ソフトウェアの起動、終了、アップデートがある。よって、RTC に対して Kubernetes を適応することで Fig.1 のように状態が拡張され、開発したソフトウェアを直ちに PC へ配置することができる。

2.2 期待される効果

ソフトウェアアップデートができることにより、Fig.2 に示す 3 つの効果が期待される。以下ではそれぞれについて詳細をまとめる。

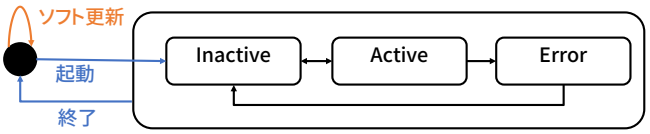

Fig. 1　Extended RTC state flow by Kubernetes

(1) 要求分析の加速
- 試作ソフトウェアの配信と実行
- 制御アルゴリズムの検証
- 製品コンセプトの確認

(2) システムテストの加速
- 改良されたソフトウェアの配信と実行
- CI によるテストの自動実行
- OS コンテナにより抽象化され統一されるソフトウェア
- 統一化されるテスト手順
- リポジトリを活用した適切なバージョン管理

(3) 開発と運用の連携
- 開発環境と運用環境の統一
- 継続的なソフトウェアアップデート
- リポジトリを活用した運用側ソフトウェアのバージョン管理

2.3 提案する開発フレームワーク

開発者が RTC を開発したところからロボットシステムに適応するまでのフレームワークを Fig. 3 に示す。

ソフトウェア開発においては、リリースされるソフトウェアであれば設計書やテストドキュメントとひも付きながらバージョン管理され、整合性が取られなければならない。ここで示す開発フローでは、RTC は Docker リポジトリで管理され、実行するソフトウェアのバージョン指定や実

[*1] 正確には共有ライブラリ化された RTC を実行する機能はある

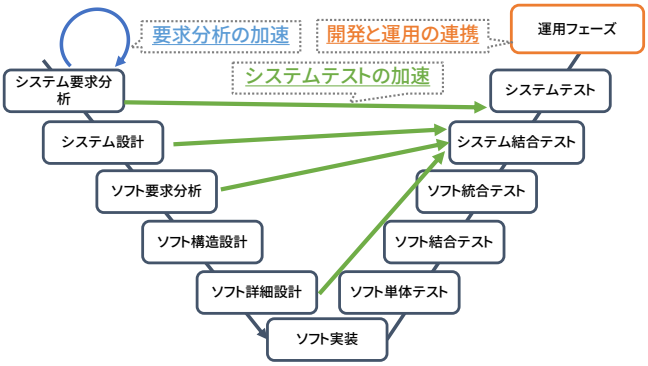

Fig. 2　V-Model with Kubernetes

行順序の管理は Kubernetes の YAML に記述されて Git などのバージョン管理システムに登録する。そのため、開発時のバージョン管理がツールによって実現され、開発部隊がフェーズごとに別れている場合でも齟齬なく開発することができる。また、開発部隊と運用部隊の間でも同様にリポジトリを設けることにより、その間での齟齬を防ぐことができるため、開発だけでなく運用までの一連の作業を加速することができる。

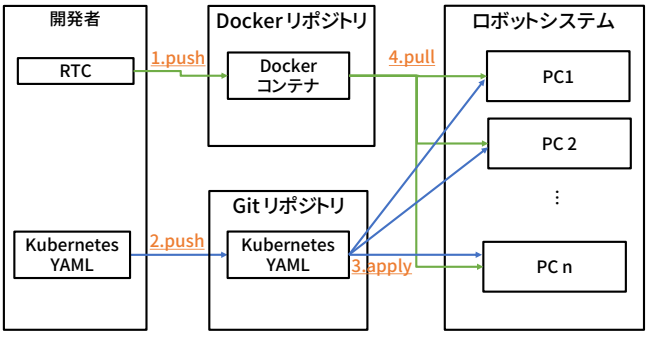

Fig. 3　RTC update flowchart

3.　開発フレームワークの実現

3.1　実装

本フレームワークを実現するにあたり、[4] で作成されたテンプレート作成機能なども活用する。これに対してソフトウェア開発で必要なバージョン管理に関する作業を支援あるいは自動化するツールを作成し、フレームワークを実現する。構築されたフレームワークは “RT ミドルウェアコンテスト 2019” の Web ページに掲載する。

4.　まとめ

提案するフレームワークを活用することで、ロボットシステムのソフトウェアアップデートに伴うテストやデバッグを加速することができた。この仕組みは Docker や Git などのリポジトリを活用しており、大規模開発においても設計部隊と検証部隊の連携がスムーズに行えるなどの利点もある。

また、この仕組みをそのまま運用側に適応することで、製品化後のアップデートも可能となる。そのため、この仕組みを応用することで継続的な機能更新といった新しいサブスクリプションモデルによるビジネスも展開可能となる。

参考文献

[1] National Institute of Advanced Industrial Science and Technology. OpenRTM-aist official website. http://www.openrtm.org/

[2] Docker, Inc. Docker offile website. https://www.docker.com/

[3] Cloud Native Computing Foundation. Kubernets official website. https://kubernetes.io/

[4] 黒瀬竜一, 宮本信彦, 安藤慶昭. Kubernetes を利用した高可用性を実現する RTC 実行フレームワーク. 2019.