# 1A04　小規模チームによる衛星管制システムの開発とスケーラビリティの実現

○三吉貴大，小林和弘，藤田朱門，石川晃寛，小林秀和，佐藤雄一
（株式会社アークエッジ・スペース）

Developing a Scalable Satellite Control System by a Small Team

Takahiro Miyoshi, Kazuhiro Kobayashi, Shumon Fujita, Akihiro Ishikawa, Hidekazu Kobayashi and Yuichi Sato (ArkEdge Space Inc.)

Keywords: Satellite Operations, Ground Segment, Agile Development, DevOps

Abstract

Aegs (ArkEdge Ground Segment) is a cloud-native satellite control system developed by a small Scrum team. It adopts a microservices architecture and DevOps practices, enabling scalability for growing satellites, diverse ground stations, and operator-driven automation.

## 1. はじめに

衛星コンステレーションによるサービス提供において，地上システムのスケーラビリティは重要な課題である．地上システムが衛星や地上局の増加に容易に対応できなければ，サービスの拡大や安定的な提供は困難となる．衛星ビジネスを取り巻く環境は激しく変化しており，サービスの競争力を維持するには，地上システムも迅速かつ柔軟に拡張できなければならない．

本稿では，株式会社アークエッジ・スペースが開発する衛星管制システム「Aegs（イージス：ArkEdge Ground Segment)」を紹介する．Aegs は，従来の衛星管制システムと異なり，クラウドネイティブなアーキテクチャを採用し，パブリッククラウドのスケーラビリティを最大限に活用している．加えて，スクラムや DevOps といったアジャイル手法を導入し，5 名以下の小規模チームながら衛星の増加に短期間で対応してきた．

Aegs の設計は 2023 年 12 月に始まり，2024 年 5 月に開発に着手した．2024 年 12 月にアークエッジ･スペースの AE1b 衛星の運用開始に伴い，本番稼働を開始した．さらに，2025 年 1 月から AE1c と AE1d，2025 年 6 月から AE2a と AE3Va の運用にも活用されている．これらの衛星運用と並行して継続的な改善を進めており，本稿ではその成果を通じて Aegs のスケーラビリティの有効性も示す．

## 2. システムアーキテクチャ

Aegs は設計当初から，以下の 3 種類のスケーラビリティを重視した．

1. 異なるインタフェースを持つ地上局の追加への対応
2. ミッションの異なる複数衛星の並行運用への対応
3. 衛星運用者自身による拡張や自動化への対応

これらを実現するため，5 つの主要なサービスからなるマイクロサービスアーキテクチャ[1] を採用している．図 1 に Aegs のアーキテクチャ概要を示す．矢印の向きは依存の方向を表す．以下，それぞれのスケーラビリティをどのように実現しているかを述べる．

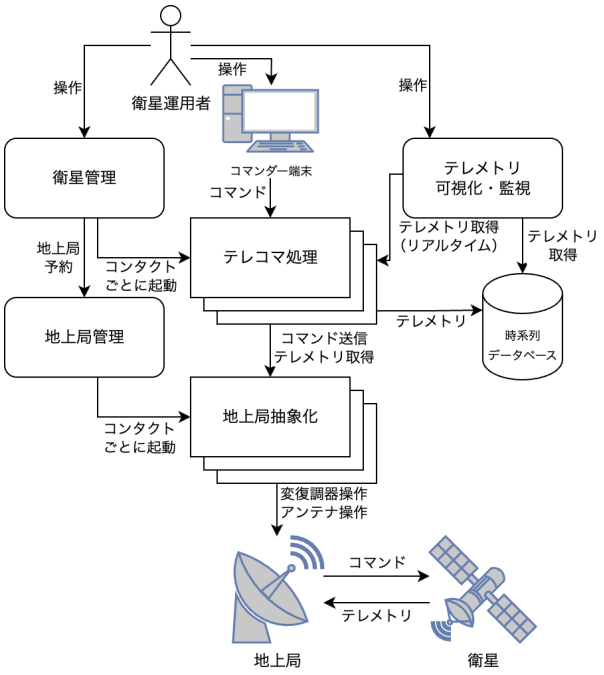



図 1　Aegs のアーキテクチャ

まず，自社で所有する地上局や社外の地上局サービスなどのインタフェースの差異を吸収するため，「地上局抽象化」サービスと「地上局管理」サービスを設けた．前者は衛星コンタクト時における地上局ハードウェアのインタフェースを抽象化し，後者は地上局ごとの設定や利用予約などを管理する．これにより，上位のサービスは地上局の違いを意識せず動作できる．

次に，「テレコマ処理」と「衛星管理」サービスは，ミッションの異なる衛星の並行運用を可能にするた

©日本航空宇宙学会
Download by SMART Conference : 2026/11/03 21:25

めのものである．「テレコマ処理」サービスは，主に衛星とのコンタクト時におけるテレメトリ・コマンド（テレコマ）の汎用的なシリアライズやデシリアライズを担う．「衛星管理」サービスは，衛星ごとのテレコマのフォーマットを管理し，コンタクトごとに「テレコマ処理」サービスに渡す．さらに，複数コンタクトを同時に実行するため，コンタクト時にのみ必要な「テレコマ処理」と「地上局抽象化」サービスは，サーバレスなコンテナ実行環境でコンタクトごとに起動し，容易に水平スケール可能としている．

最後に，衛星運用者による拡張や自動化のため，「テレメトリ可視化・監視」サービスにはオープンソースソフトウェアの Grafana[2] を採用した．Grafana は大量のテレメトリを柔軟に可視化・分析できるとともに，ダッシュボードの拡張やアラート管理を運用者自身が行える点で適していた．また，その他のサービスも API を提供し，外部ツールとの連携や自動化を可能とすることを目指している．

これらの設計により，Aegs は衛星数・地上局数・運用要求の増加に対して，効率的にスケールするアーキテクチャを実現している．

## 3. アジャイル開発体制

数ヶ月ごとに衛星が増えていく変化の激しい環境において，運用から得られた知見を迅速にプロダクトに反映するため，Aegs 開発チームでは，アジャイルフレームワークとしてスクラムを採用し，DevOps のプラクティスを実践している．

スクラムガイド[3] によれば，「スクラムとは，複雑な問題に対応する適応型のソリューションを通じて，人々，チーム，組織が価値を生み出すための軽量級フレームワークである．」スクラムでは，スプリントと呼ばれる短い開発サイクルを繰り返しながら，プロダクトの価値を漸進的に高めてゆく．2025 年 8 月現在，Aegs のスクラムチームは 1 名のプロダクトオーナーと 3 名の開発者で構成され，開発者のうち 1 名がスクラムマスターを兼任している．プロダクトオーナーは衛星運用の経験者だが，開発者は当初運用経験を持たなかった．衛星運用という複雑な問題に対して，実運用を通じて適応的に改善を重ねるうえで，スクラムは適した開発体制であると判断した．

DevOps は，Development（開発）と Operations（運用）の連携を強化し，ソフトウェアの変更を迅速かつ安定して本番環境に届けるための考え方である．Forsgren らは，DevOps のプラクティスの実践が変更失敗率や復旧時間などのソフトウェアデリバリ性能を改善し，さらに組織のパフォーマンスや事業成果とも相関することを示している[4]．

Aegs 開発チームでは，これらの知見に基づき DevOps のプラクティスである継続的デリバリーを重視している．開発者自身がデプロイメントパイプラインを構築し，変更は可能な限り小さな単位とし，速やかに本番環境へリリースすることを原則とした．これにより，衛星数の増加に伴う頻繁な更新や障害対応にも迅速に対応できる体制を構築している．

さらに，衛星管制システムの開発と衛星運用を統合するさらなる DevOps の実践として，AE2a の運用は Aegs 開発チームが主体となって実施している．開発者が自らプロダクトのユーザとなることで，より短いフィードバックループを実現し，改善サイクルを加速させている．

## 4. 6U 衛星運用における Aegs の実績と改善

アークエッジ・スペースが開発する 6U 汎用衛星バスを搭載した 5 機の衛星[5] は，すべて Aegs を利用して運用されている．2024 年 12 月 10 日の AE1b 運用開始から 2025 年 8 月末現在までに，2341 回の衛星コンタクトを実施した．

この間，Aegs のすべてのサービスを合計して 420 回のデプロイを本番環境で実施した．1 営業日あたりの平均デプロイ回数は 2.4 回であり，1 日に複数回のデプロイ頻度を実現できている．デプロイ起因でコンタクトが実施できなくなった障害は 2 回のみで，変更失敗率は 0.5% 以下，平均復旧時間は 42 分であった．これらの値はソフトウェア開発組織のパフォーマンス指標[4] に照らしても高水準である．変更リードタイムは計測できていないが，2024 年の DORA レポート[6] に基づけば，エリートレベルに相当する開発体制を達成している．

また，継続的デリバリーにより短いサイクルで改善を重ね，本番稼働後も以下のような機能を導入した．

**無人テレメトリ監視**　コンタクト時刻に地上局追尾とテレメトリ送信を事前予約することで，人員配置なしに衛星の状態確認を可能にした．

**自動コマンド送信**　定常的な運用コマンドを人間の操作を介さずに送信できるようにした．

**コマンドスクリプトの静的検査**　衛星に送信するコマンドスクリプトの型検査を継続的インテグレーションに組み込み，実行時エラーを大幅に削減した[7]．

**運用計画管理のシステム化**　スプレッドシートを廃止し，Aegs で運用計画を一元管理するとともに，社内予定管理システムと連携できるようにした．

今後も 6U 衛星や利用地上局の増加が見込まれるが，Aegs の継続的な開発により，衛星運用のさらなる省力化とスケールを目指していく．

## 参考文献

1) Lewis, James, Fowler, Martin. Microservices. 2014. https://martinfowler.com/articles/microservices.html, (参照 2025-09-12).

©日本航空宇宙学会　Download by SMART Conference : 2026/11/03 21:25

2) “Grafana OSS and Enterprise”. Grafana documentation. https://grafana.com/docs/grafana/v12.1/, (参照 2025-09-12).
3) Schwaber, Ken, Sutherland, Jeff. スクラムガイド. 2020. https://scrumguides.org/docs/scrumguide/v2020/2020-Scrum-Guide-Japanese.pdf, (参照 2025-09-12).
4) Forsgren, Nicole, Kim, Gene, Humble, Jez. Lean と DevOps の科学. インプレス, 2018.
5) 船曳敦漠, 石川晃寛, 秋間敏史, 徳田正章, 鈴木聡宏. “6U 汎用衛星バスの複数機製造および軌道上動作実績に関する報告”. 第 69 回宇宙科学技術連合講演会. 1A02, 2025.
6) Accelerate State of DevOps 2024. DevOps Research and Assessment, 2024. https://dora.dev/research/2024/dora-report/, (参照 2025-09-12).
7) 小林和弘. 衛星運用の実際と、効率化のために型システムを実装した話. 2025. https://blog.arkedge.space/entry/2025/04/24/080000, (参照 2025-09-12).

©日本航空宇宙学会　Download by SMART Conference : 2026/11/03 21:25