第1章 メンテナンス性/セキュリティ/長寿命化に優れる「Armadillo Base OS」

コンテナ・アーキテクチャを採用したIoT機器向けLinux OS

實吉 智裕

表1 汎用Linuxベースと Yoctoベースの特徴

項 目	汎用Linux（Debian/Ubuntuなど）ベース	Yoctoベース
カスタマイズ性	標準パッケージが豊富．迅速なプロトタイピングが可能だが，カスタマイズには限界がある	高度にカスタマイズ可能．必要なパッケージのみ含む軽量なシステムを構築可能
開発とメンテナンスの容易さ	パッケージ・マネージャにより，簡単にソフトウェアの管理が可能．メンテナンスが容易	再現性の高いビルド環境で，長期的なメンテナンスが容易
学習コスト	開発者にとってなじみやすい環境で，学習コストが低い	急な学習曲線がある．高度なスキルが必要
開発時間	短時間で開発を進められる	ビルドとカスタマイズに時間がかかる
ブート時間	多くの初期化スクリプトやサービスにより，ブート時間が長くなる傾向にある	軽量なため，ブート時間を最適化できる
長期サポート	LTS（Long Term Support）版で長期サポートが提供されるが，製品寿命全体をカバーできない場合がある	長期的な運用が可能．ビルドの再現性があり，保守が容易

昨今，IoT機器でのセキュリティ対策が重視されています．機器へ対策することはもちろんですが，SBOM（Software Bill of Materials）を利用したソフトウェア管理や，設置されている機器に対し長期にわたってソフトウェアをアップデートすることが求められています．セキュリティ，ソフトウェアの長期保守への対応として，何が必要か、Armadillo Base OSを例に説明します．

もう1つのコンテナ・エンジンPodman

本特集でメインとして扱っているDockerは，コンテナ・エンジンの1つの実装です．Docker以外にも多くの実装があります．

Podman（pod managerの略）(1)は，レッドハットが中心となり開発が進められている実装です．Dockerと互換性があり，Open Container Initiative（OCI）仕様に準拠したコンテナ・イメージを実行できます．引数などの指定方法はDockerと同じです．

Dockerは1つのデーモン上で複数のコンテナが管理されていますが，Podmanはlibpodライブラリを使用した実装となっており，コンテナ同士が干渉しにくい構造です．仮に1つのコンテナに異常があっても，他のコンテナへ影響が及ばず，セキュリティ的にもメリットがあるとされています．

汎用Linux（Ubuntu/Debianなど）とYoctoベースの比較

組み込み機器に搭載されるLinuxは大きく2つに分類されます．UbuntuやDebian GNU/Linux（以降，Debian）などの汎用Linuxをベースにしたものと，Yocto Projectをベースにしたものです．例えば，エヌビディアのJetson LinuxではUbuntuを，ラズベリー・パイのRaspberry Pi OSはDebianをベースとしています．PCライクに使えることを意識したCPUボードでは，多くの場合，汎用Linuxをベースに採用してます．

一方で，組み込み機器らしく，必要な機能に特化する前提のLinuxでは，多くはYocto Projectをベースにしています．主にNXPセミコンダクターズやルネサス エレクトロニクス，クアルコムなどのアプリケーションSoCを提供する半導体メーカに採用されています．

汎用LinuxベースとYocto Projectベースとでは，それぞれのメリットとデメリットが大きく異なります．**表1**にその特徴をまとめました．いずれの選択肢でも，機器を開発し，長期にわたって運用するのに，苦労するポイントがあり，どちらを選ぶのも悩ましいです．