Technology Reports　クラウド　MEC　アプリケーションデザインパターン

MECを活用したアプリケーションデザインパターン

イノベーション統括部　秋永（あきなが） 和計（よしかず）

近年，5Gネットワークの整備が進み，ドコモでも2020年3月より5Gサービスを開始している．5Gには高速大容量，低遅延，多数端末同時接続という3つの大きな特長があり，その中でも高速大容量，低遅延に関しては，従来モバイルでは難しかったサービスを実現できると大きく期待されている．そのためMECによって，高速大容量，低遅延を活かすシステムを構築しようとする取組みが注目されている．ドコモでも「ドコモオープンイノベーションクラウド」というクラウドサーバと，それらを「クラウドダイレクト」というモバイルネットワーク内に直接接続するサービスが2020年6月から提供されている．これらのサービスを活かすために，アプリケーション構築時のアーキテクチャを考察する際に，必要な機能に対して用いられるアーキテクチャのテンプレート「MECアプリケーションデザインパターン」を提案する．本稿では，代表的なパターンとその効果について解説する．

1. まえがき

近年，第5世代移動通信システム（5G）ネットワークの整備が進み，ドコモでも2020年3月より5Gサービスを開始している．5Gには高速大容量（eMBB：enhanced Mobile BroadBand），低遅延（URLLC：Ultra Reliable and Low Latency Communications），多数端末同時接続（mMTC：massive Machine Type Communications）という3つの大きな特長があり，これらを活用することでさまざまなアプリケーション，ソリューション，産業にインパクトを与えると言われている．その中でも，高速大容量，低遅延に関しては，従来モバイルでは難しかったサービスを実現できると大きく期待されている．

そのためMEC（Multi-access Edge Computing）と呼ばれているネットワーク内に配置するクラウドによって，高速大容量，低遅延を活かすシステムを構築しようとする取組みが注目されている．ドコモ

でも「ドコモオープンイノベーションクラウド」というクラウドサーバと，それらを「クラウドダイレクト」というモバイルネットワーク内に直接接続するサービスが2020年6月から提供されてきた．これらのサービスを用いると，5Gの特長を活かしたさまざまなアプリケーションやソリューションの提供が可能になると期待されている．

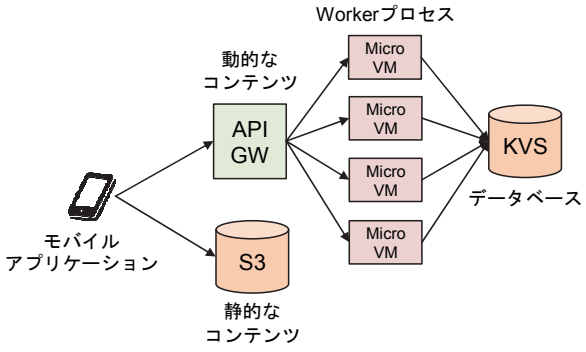
これらのサービスを活かすために，「MECアプリケーションデザインパターン」を提案する．これらを参照することで，開発者はアプリケーションを構築する上で，機能をクラウドおよびMECで最適配置した設計ができるようになると考えられる．本稿では，アプリケーションデザインパターンとMECについて述べた後，代表的なパターンとその効果について解説する．

2. アプリケーションデザインパターンとは？

アプリケーションデザインパターンとは，アプリケーション構築にかかわるアーキテクチャを考察する際，必要な機能に対して用いられるアーキテクチャのテンプレートに相当するものであり，代表的なユースケースを基に一般化されたアイデア集であり，アプリケーションアーキテクチャとして用いることができる．

本デザインパターンでは，一般的なデザインパターンと同様に個別のアプリケーション全体像を議論するのではなく，部品として議論されることから再利用性が高く，アプリケーションのアーキテクトが直ぐに新しい機能を取り込みやすく，考察しやすくしている．

例として，Webサービスを構築する上でのパブリッククラウド*1上でのアプリケーションデザインパターンを**図1**に示す．図1では，パブリッククラウドでよく用いられる典型的なサーバレスアプリケーションのアプリケーションデザインパターンを示している．動的なコンテンツと静的なコンテンツを分けてストアすることや，スケールアウトするためのWorkerプロセス*2を細かく配置すること，そ



MicroVM：非常に小さな単位で高速に起動できる仮想マシン

図1　アプリケーションデザインパターン（サーバレス）の例

＊1　パブリッククラウド：インターネットを介して誰でも利用ができるクラウドコンピューティングサービス．

＊2　Workerプロセス：ある一定の役割をもって起動され，その役割を終えると終了する単一のプログラム．

れらからのアクセスにKVS（Key Value Store）[*3]を用いることでデータベースのボトルネックを回避していることなどが一目で分かるようになっており，このデザインパターンを踏襲して自らのアプリケーションを設計することで，スケーラビリティのあるモバイルアプリケーションを実装することができる．このように設計上のノウハウのパターンとして整理しているのがアプリケーションデザインパターンである．

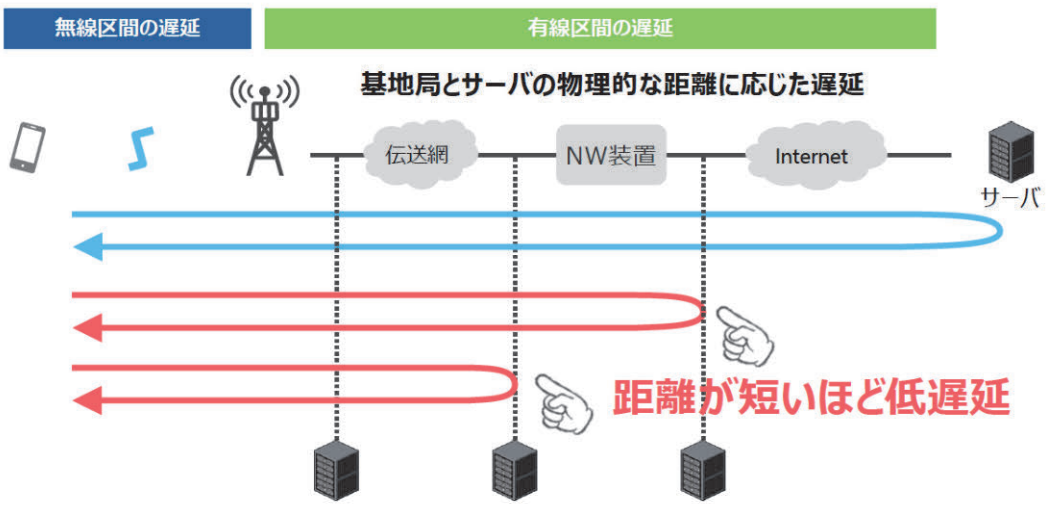
3. MECとは？

5Gネットワークにおいて，その低遅延性を利用したいという機運が高まっているが，無線区間の5G化だけでは低遅延を実現できない．システムとして低遅延を実現するためには，ネットワーク全体を見た上で，往復の伝達距離を短くし，経由する装置の数を少なくする必要がある．そのため，5Gネットワーク内部にコンピューティングリソースをもつことでその解決を図ろうとしている．これがMECである．MECは無線区間からできるだけ近い場所にコンピューティングリソース（仮想マシン）を置くことで，実現される（**図2**）．一方で，無線に近い（基地局に近い）場所になればなるほど，その遅延時間を短くすることができるが，基地局数は膨大であり，それらに対して必要な仮想マシンを設置しようとすると，必然的に数が多くなり，経済的に非現実的となる．逆にインターネットに近い場所になれば集約が図られるが，遅延は大きくなるため，需要と供給のバランスをみて適切なリソース配置が重要となる．

3.1 MECの4つの特長

MECでは，主に以下の4つの特長が利用者の観点で期待されている．

①トラフィック最適化，ネットワークでの折返し
端末同士のP2P（Peer to Peer）通信時に，地理的に近いサーバにアクセス，もしくはトラ



出典：https://www.nttdocomo.co.jp/corporate/ir/binary/pdf/library/presentation/200825/200825_qa.pdf（P.21）

図2　MECのネットワーク内の位置

＊3　KVS：Key（鍵）とValue（値）の2つをセットにしてデータを簡易的に管理することに特化した，高速化されたデータベース．さらに複数サーバでの分散処理にも対応していることが多く，膨大な入出力を一度に処理することができるため，処理のボトルネック回避によく用いられる．

フィックを経由させることで，非常に低い遅延での通信が可能になり，かつサーバからは大容量のデータを流すことができる．ゲームなどで応答時間の要求条件が厳しいものに対して利用が期待できる．

②低遅延，ゆらぎの抑制

地理的に近いサーバにアクセスさせることで，5Gの特長を活かした低遅延での接続が可能になる．また，インターネットとは異なり閉域網での提供になるために，遅延におけるゆらぎの要因が少なくなり，ゆらぎに弱いストリーミングなどで利用が期待できる．

③セキュア・プライベートネットワーク構築

インターネットなどの外部ネットワークに出ることがないため，安全な接続を提供できる．SIM認証でのプライベート接続なども可能になるため，秘匿性の高い情報を扱うネットワークとしての利用が期待できる．

④コンピューティングパワーの提供，端末機能補完

端末では処理ができないような高度なAI処理や高精細な3D画像の生成など，負荷が大きな処理をエッジコンピューティング[*4]側で実施することができる．ゲームや3D CAD（Computer Aided Design），XR[*5]などでの利用が期待できる．

アプリケーションデザインパターンでは，このようなMECの特長を活かした具体的なアプリケーションへの組込み方法が議論されている．

3.2 アプリケーションデザインパターンでのMECの前提条件

MECは，地域に分散してサーバを構築するという性質上，対障害性や堅牢性の面で，集約型データセンタで運用されるシステムとは異なる．そこで，MECではDesign for failureと呼ばれる，パブリッククラウドコンピューティングにおける，故障することを前提とするシステムの構築法を踏襲する．例えば，MEC上でのデータ永続化は保証されない．これは，データの永続化のためには相当数の冗長性をもつ必要があるが，この冗長性が分散化されたシステムでは取りにくいためである．そのため，データ永続化はMECレイヤではなく，上位のパブリッククラウドや他のストレージサービスで実施するのが好ましい．

ほかにも，MECにおける前提条件は下記のものがある．

- 堅牢性，可用性は共に低い．
- IasS（Infrastructure as a Service）[*6]を提供する．
- データの永続化が期待されるようなPaaS（Platform as a Service）[*7]を極力もたない．
- 一方で，コンテナ[*8]管理サービスのようなポータビリティを向上させる機能は具備する．
- DNS（Domain Name System）[*9]サービスが提供される．

4. MECアプリケーションデザインパターン

以下では，具体的なアプリケーションデザインパターンと利用方法について，代表的なものをいくつか解説する．

4.1 高速大容量アップロードパターン

5Gの高速大容量を活用したアプリケーションが具備する機能の1つに，従来モバイルネットワークが苦手としていたアップロードがある．アップロードの速度が格段に上がったことにより，その活用を考えることができるが，インターネットへの通信は遅延の増大や，遅延時間の揺れ，それによるTCP（Transmission Control Protocol）[*10]のACK（ACKnowledgement）[*11]遅延によるスループットの低下などが発生し，十分

＊4 エッジコンピューティング：ユーザの近くにエッジサーバを分散させ，距離を短縮することで通信遅延を短縮する技術．
＊5 XR：VR，AR，MRといった仮想空間と現実空間との融合で新たな体験を提供する技術の総称．
＊6 IaaS：サーバ，ネットワークなどのハードウェアを仮想的に貸し出すサービス．利用者は借りたサーバやネットワーク上にOSやアプリケーションソフトウェアを設定して利用する．
＊7 PaaS：アプリケーションを実行するためのOSやミドルウェアを含むプラットフォームをクラウド上で貸し出すサービス．利用者は借りたプラットフォームの上でアプリケーションソフトウェアを作成して利用する．

な速度を出すことができない．そこで，MECを用いて，高速大容量のアップロードを実現させる．

⑴課　題

高速5Gのネットワークを利用して，大容量のアップロードを高速化したい．また，アップロードのTCPのACKパケットの遅延や，ゆらぎ，上位サーバの影響などからアップロード作業のみの独立性を高め，安定させたい．さらに，大容量アップロードを同時に多数受け付けたいという課題がある．

⑵デザインパターン

5Gのアップロード能力を最大限に活かすため，MECローカルサーバでアップロードを終端し，一時ストレージに格納する．そして，格納内容をキューイング＊12し，Workerプロセスが別途一時アップロード先から回収し，パブリッククラウドのストレージで永続化する．一時ストレージに格納している段階で，別途プロセスを走らせることも可能（例：AIによる画像・映像処理）である．最も近いアップロードサーバとの接続には，DNSでの近傍サーバ検索サービスを利用する．また，ストレージへの一時的な格納は負荷分散にも効果があるため，同じ箇所からの大量アップロード対応（例えばスタジアムからの一斉アップロード）などで上位の回線の帯域が十分にない場合も対応可能と考えられる．アップロードのプロセスだけを分離して実装できるので，既存システムへの組込みが容易である．なお，可用性を担保するためにMEC上にはLB（Load Balancer）＊13を配置し，アップロードサーバーを二重化しておくことが望ましい（**図3**）．

4.2　超低遅延メッセージング，ステータス管理パターン

5Gの低遅延性を活かし，複数端末での同期を行った複数人数でのゲームや，XRの同時体験，メッセージの交換などができる．この同期を実現する上でMEC上にKVSのPub/Sub機能＊14を実装することで，

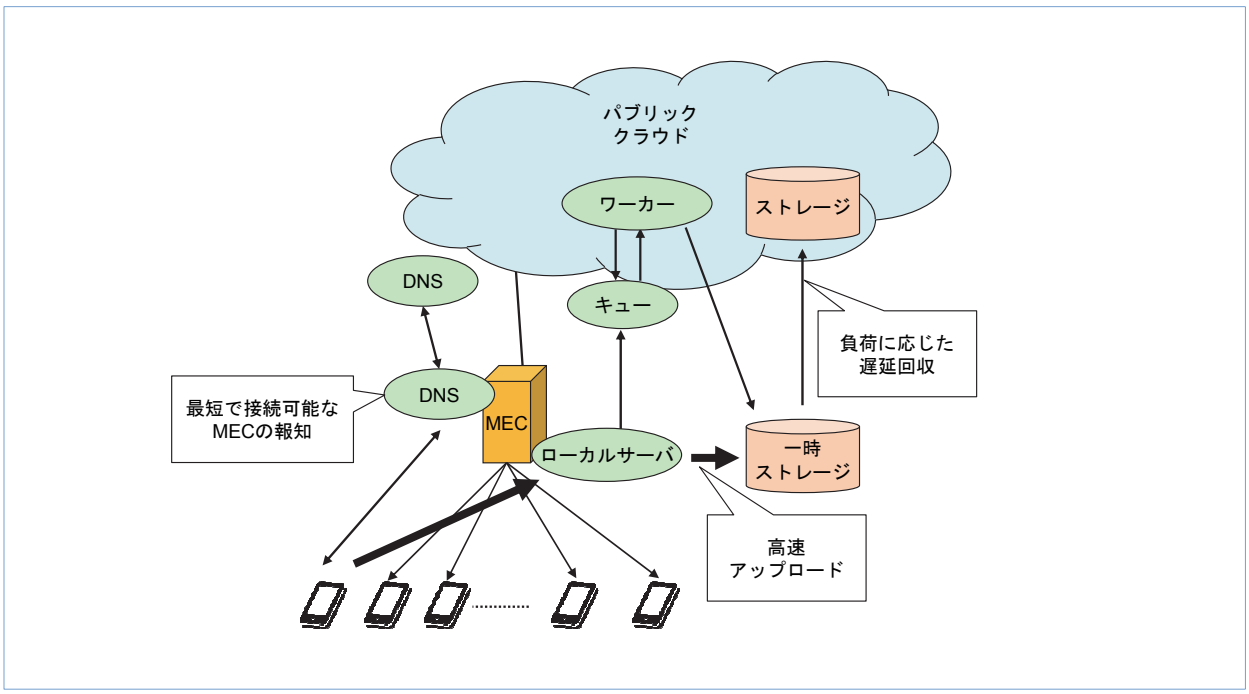


図3　高速大容量アップロードのデザインパターン

＊8　コンテナ：コンピュータ仮想化技術の一種で，1つのホストOSの上にコンテナと呼ばれる専用領域を作り，その中で必要なアプリケーションソフトを動かす方式のこと．

＊9　DNS：IPネットワーク上のホスト名とIPアドレスの対応付けを行うシステム．

＊10　TCP：インターネットで標準的に利用されるIPの上位プロトコル．接続相手やデータ到着の確認・フロー制御・データの重複や抜けの検出などを行うことでIPの補完の役割を果たし，信頼性の高い通信を実現する．

＊11　ACK：データの受信ノードが正常に受信（復号）できたことを送信ノードに通知する受信確認信号．

非常に汎用性の高いメッセージングが行えるようになる．

⑴課　題

超低遅延のKVSやそのPub/Sub機能をMECに置くことで，低遅延の同期を実現したい．また，対戦型ゲームやオープンワールド系ゲームの端末同士の状態同期，IoTデバイスの状態同期などを超低遅延で実現したい．他にも，ゲームにおけるワープ*15などのゲーム性を損なう挙動は排除したいという理由から，一定以下の同期遅延でサービスを提供したいという課題がある．

⑵デザインパターン

メッセージングや一時的なステータス管理などを超高速で行うため，MEC上にRedisなどのインメモリDB*16を配置する．これにより，端末間のメッセージングや，アプリケーションの一時的な状態管理・同期などを超高速で実装することができる．これらのインメモリDBはnsオーダで情報を配布，参照することができる．これに関しても近傍インメモリDBとの接続には，DNSでの近傍インメモリDB検索サービスを利用する．Pub/Sub機能を使った場合は近傍端末同士の超低遅延でのメッセージの交換が可能であり，また同様にKVSを使ったアプリケーションの状態管理などが可能である．これらのインメモリDBを使うことで，自分がプレーしている地域のゲームスコアの超高速集計や，ゲームランキングの集計（リアルタイムリーダーボード*17）などが可能である．また，複数システムの連携のためのサービスバス*18や，ネットワークゲームのプレイヤー同士の状態同期，リモートロボットの同期などに利用可能である（**図4**）．

⑶注意点

インメモリDBは認証が存在しないケースがあるので，実装時にはメッセージの暗号化やネットワークの分離などが必要である．

4.3　Webサービスキャッシュパターン

Webでのリアルタイム更新の方法として，Webソケット*19を用いた方法がある．こういったサービスを実装する場合は，その遅延やトラフィックの負荷分散などが大きな課題となる．この課題解決にもMECが有効である．

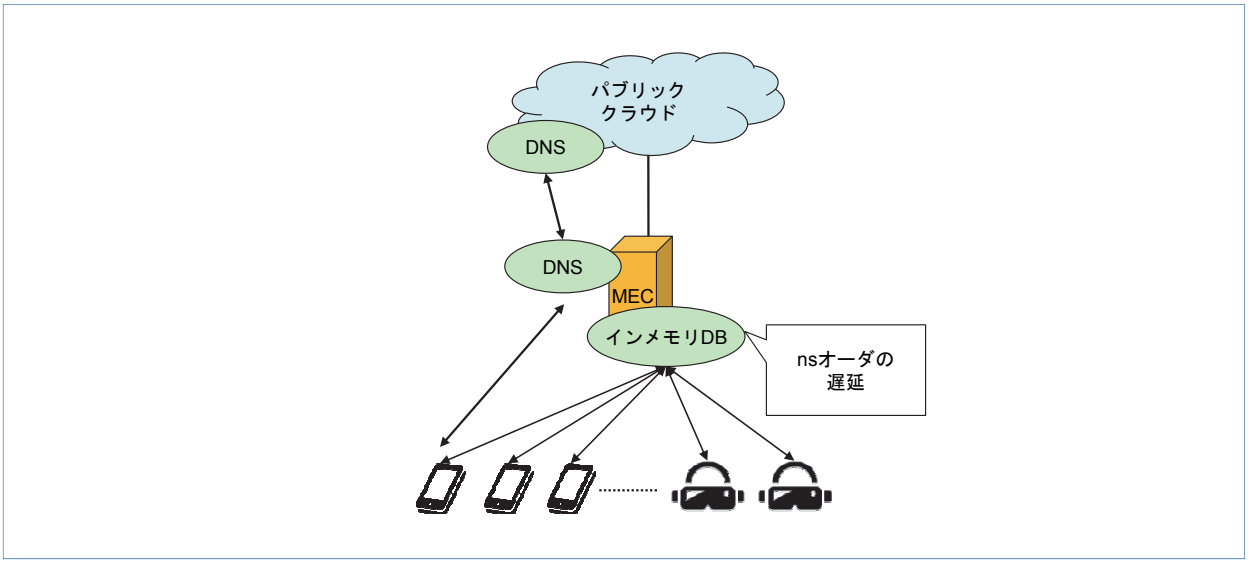


図4　超低遅延メッセージング，ステータス管理のデザインパターン

＊12 キューイング：待ち行列を作成し，処理する順番や処理する内容を一時的に保存しておくこと．

＊13 LB：通信トラフィックの負荷分散を行う装置．

＊14 Pub/Sub機能：Publish（出版）とSubscribe（購読）の機能で非同期のメッセージのやり取りをする機能．Subscribe側にはPublish側で送信されたものが全員に配布されるため，1：Nのメッセージの配布に適している．

＊15 ワープ：2者以上が参加しているゲームなどで，お互いの位置情報の同期がずれることにより，プレイヤーが突然消え，別の所から現れる現象．ゲーム性を損ねる場合が多く，ゲーム提供者側としては避けられるなら避けたい現象．

⑴課　題

動的なコンテンツの更新をできるだけ低遅延で行う（Webソケットなど）ことや，高いレスポンスのサービスを実現したいという課題がある．また，Web上での静的なコンテンツの読出し負荷の軽減を行いたいなどの課題もある．

⑵デザインパターン

Webサービスにおけるキャッシュ機構をMEC上にもつことにより，Web三層構造＊20のうち，プレゼンテーション層とアプリケーション層をMEC側に置いて高速化を実現できる．また，データベース層についてもリードレプリカ＊21やキャッシュ機構をMEC上にもつことで高速化を実現できる．注意したいのは，データベース層をパブリッククラウド側にもって来ないと，永続化が難しくなり，また広域での一貫性（Consistency）が保てなくなるため，キャッシュをアプリケーション層の近くに置いて高速化を実現する．

データベース層に関しては，MySQL（My Structured Query Language）＊22のリードレプリカや，マルチマスタ＊23なども負荷軽減という目的には有効である．しかし，低遅延を期待する場合は，データベース層の遅延はネットワークの遅延よりも大幅に大きいので，そのバランスに注意したい（**図5**）．

⑶注意点

アプリケーション層は顧客の近くに自動配置したいため，ポータビリティを考慮してコンテナ化などの工夫をするのが望ましい．また，Webのキャッシュにはインメモリキャッシュ（KVS）を用いることで，超高速レスポンスが可能になる．その場合の書込みについては，永続化の問題を考慮する必要がある．

このパターンは，実装は容易だが，MECサーバ数が増えると管理コストが増える可能性があるため，注意が必要である．また，コンテナ管理ソリューションなどを利用し，実装をパッケージ化しつつ，必要最低限のアプリケーションを配置することが，システムを効率よくかつコストを抑えるための注意点となる．

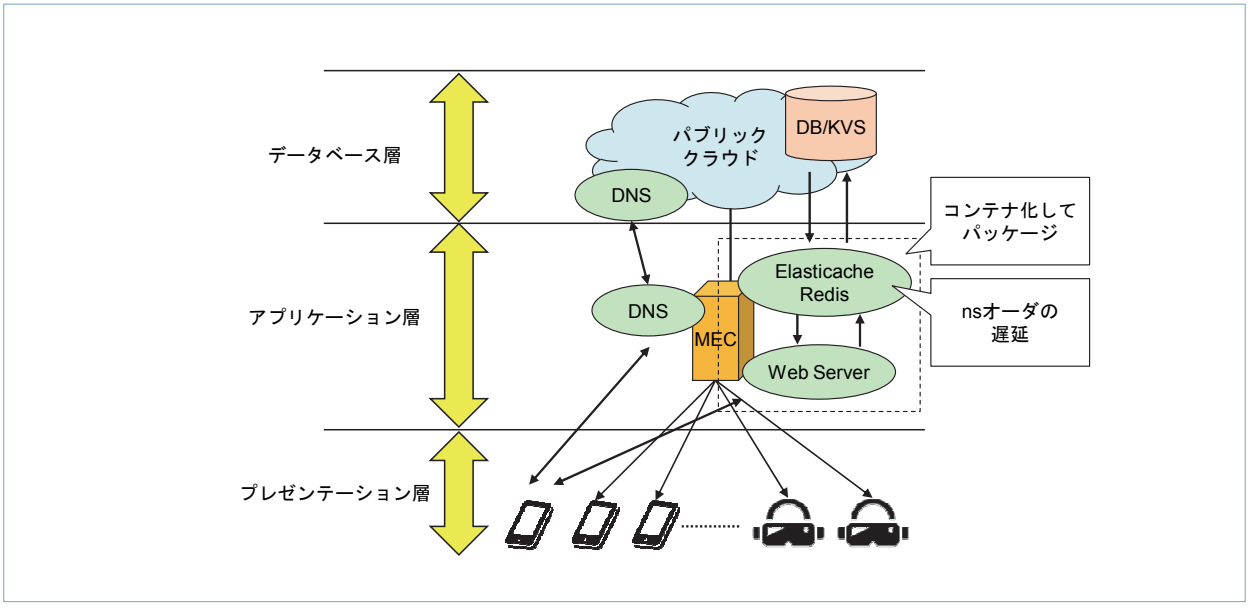


図5　Webサービスキャッシュのデザインパターン

＊16 インメモリDB：データをメモリ上に展開保持して処理することで，情報の取得に対して高速な応答を可能にしたデータベース．

＊17 リーダーボード：ゲームなどでランキングの上位者や，自分の順序が記載されているボード．

＊18 サービスバス：複数システムを連携させる場合に，お互いの状態の連携のためや，次のシステムに処理を引き継ぐためのメッセージのやり取りをする機構．

＊19 Webソケット：Web上で双方向のメッセージをやりとりするために，定められている技術規格．RFC（Request For Comments）6455として定められている．

4.4 IoTデバイス用軽量プロトコルの実装パターン

IoT機器におけるセキュリティの問題は，インターネット網からのアクセスを遮断し，キャリア網内に低負荷・低セキュリティのプロトコルを留めることで解決する．インターネットへの直接的な接続を分断でき，かつ多数のデバイスからの処理を低遅延で処理できるので，利便性が高い．

(1)課　題

IoTデバイスには，SSL（Secure Sockets Layer）[*24]などを実装できない小型のものなどもあるため，軽量プロトコルを採用したいが，同時にセキュリティも実現したいという課題がある．MQTT（Message Queuing Telemetry Transport）[*25]をTLS（Transport Layer Security）[*26]で実装する例などもあるが，IoT機器の電池消費量を抑えたい場合，通信の暗号化は負荷が大きいため，より認証や暗号化の少ない軽量なプロトコルを利用したいという課題がある．

(2)デザインパターン

IoT機器向け軽量プロトコルMQTTなどを安全にモバイルネットワーク内で終端し，MEC外に出る際に加工や暗号化を行い，安全にIoT機器を管理できるようにすることができる．これにより，インターネットからの攻撃の回避や安全なデバイスの管理システムを構築することができる（**図6**）．

4.5 その他のデザインパターン

そのほか考案されているデザインパターンを**表1**に示す．今後MECを有効活用する具体的な方法として，デザインパターンを増やしていきたい．

5. MECアプリケーションデザインパターンとソリューションの関係

MECアプリケーションデザインパターンは，MECを有効活用する上での具体的な機能を実現するための便利な利用パターン（ベストプラクティス）のみを示している．従って，アプリケーションもしくはソリューションを構築する際に，これらのデザインパターンを組み合わせることで実現する．その包含関係を**図7**に示す．

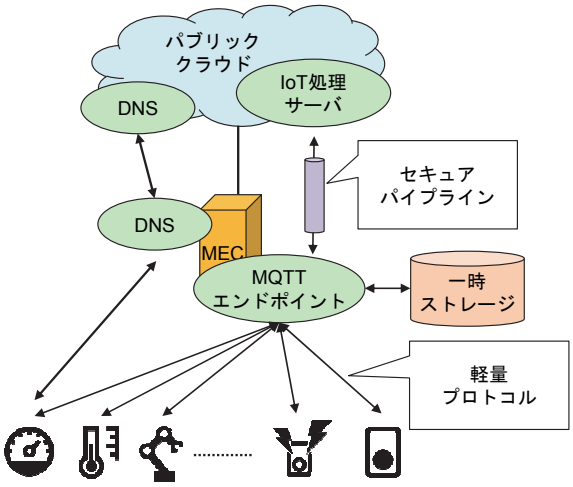


図6　IoTデバイス用軽量プロトコル実装のデザインパターン

＊20 **Web三層構造**：Webシステムの構成要素をプレゼンテーション層，アプリケーション層，データ層の3層に分割し，独立したモジュールとして設計する手法．

＊21 リードレプリカ：データベースの読込み専用のコピー．読込みや検索負荷の軽減のために本体のデータベースからのコピーを常に保持する．

＊22 MySQL：最もよく使われているオープンソースのリレーショナルデータベース管理システム（RDBMS：Relational Data-Base Management System）のうちの1つ．

＊23 マルチマスタ：データベースの高信頼化や高性能化の方式の1つで，複数の同一の機能をもつ接続先をもつことができるシステムを指す．複数の接続先があっても，参照されるデータは同じであり，利用できる機能に制約がない．

表1　その他の代表的なMECアプリケーションデザインパターン

パターン名	課　題	デザインパターン
CDNパターン	動画やストリーミングサービスにおいて，オリジンサーバ（配信サーバ）への負荷軽減をしたい． オリジンサーバへのネットワークのゆらぎなどにより品質の劣化を防ぎたい． 超高速・超大容量でのサービス提供は上位ネットワークほど負荷が集中し，品質が劣化しやすい． できるだけ映像をリアルタイムで提供したい．	通常のCDNシステムと同様にDNS側で最も近い場所を返すようにすることで最も近いサーバを判断することができる． CDN側では最も近い場所のキャッシュを取りに行くことができる．各CDNサーバは変更がある場合に，オリジンサーバ（例：パブリッククラウド上のWebサーバや，ストリーミングサーバ）から最新情報を取得する． ストリーミングでHLSなどを使っている場合も同じロジックで適用可能なため，汎用性が高い．
折返しによるP2Pトラフィックの最適化パターン	ローカルエッジネットワークにおける折返し通信により，P2Pトラフィックを最適化し，安定した高品質のサービスを提供したい． 例えばビデオ通話などをローカル限定で折り返すことにより，最適化を図りたい．	5G/4Gのローカル NW上で折返しトラフィックを網内で最短ルーティングすることで低遅延での端末間通信を可能にする． MEC上にシグナリングサーバを配置することでWebRTCやVoIP（SIP）などを実装することが可能になる．シグナリングサーバを広域に展開する場合はMEC上でなくても良い．
ローカルセキュアネットワークパターン	ローカルネットワークを端末間で実現し，VPNなどを使わずにセキュアなネットワークを構築したい（ファイルサーバなど）．	VPNサーバをMEC上に配置することで，好きなNWを作ることができる．また，高速大容量を実現できるため，ファイルサーバなどへのアクセスも安全かつ容易に構築できる．
地理的制約を利用したセキュリティ実現パターン	特定の地域外からのアクセスを防止し，よりサービスを安全に提供したい．	MEC上に特定の地域のみにファイルサーバを配置し，そのシステムには特定地域のシステムからのみサーバのアクセスを許可する． DNSで地域のMECサーバを特定し，そのサーバにアクセスする． MEC上にはその地域でしかアクセスできない情報，もしくは認証方式を用いてアクセスする． パブリッククラウド上にストレージ，もしくはDBを配置し，堅牢性や可用性はそちらで担保する． 可用性を担保するためにMEC上ではLBを配置し，二重化しておく．

CDN：Content Delivery Network　　VoIP：Voice over Internet Protocol
HLS：HTTP Live Streaming　　VPN：Virtual Private Network
SIP：Session Initiation Protocol　　WebRTC：Web Real-Time Communication

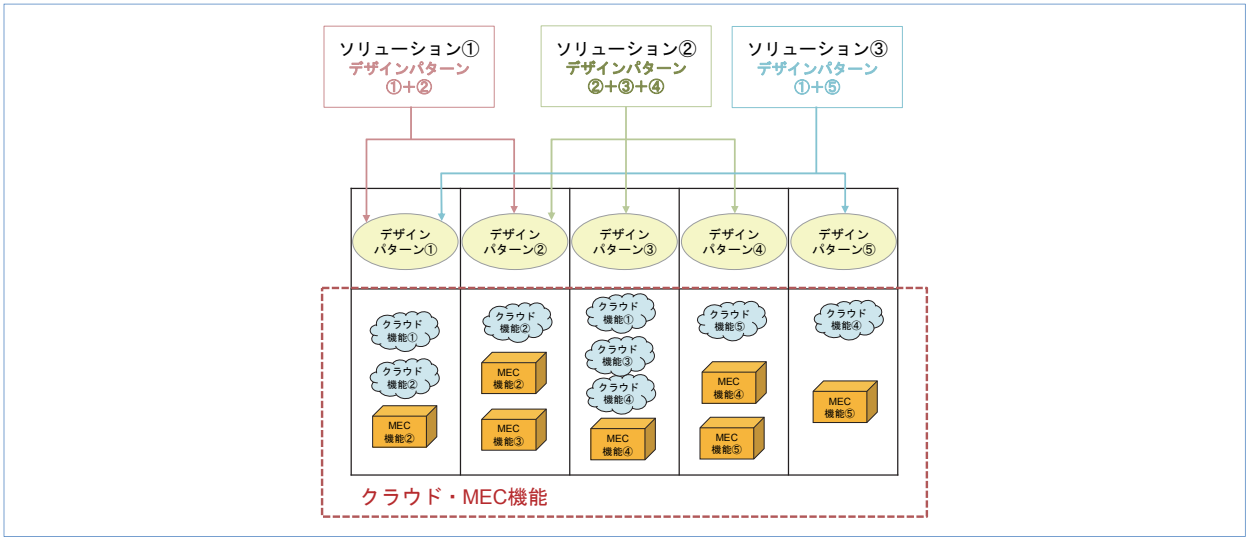


図7　アプリケーションデザインパターンとソリューションの関係

＊24 SSL：ネットワーク上のアプリケーション間，主にWWWブラウザとWWWサーバとの間で通信の暗号化，データ改ざんの発見を行うためのプロトコル．
＊25 MQTT：Pub/Sub型の軽量なメッセージキュープロトコル．IoTの各種デバイスとサーバ間でのメッセージ交換に使用される．
＊26 TLS：SSLをインターネット標準技術として規定し，拡張されたセキュリティを確保するためのプロトコル．SSLと比べ，暗号アルゴリズムやエラーメッセージ規定などが拡張されている．

6. あとがき

本稿では，MECをより便利に使っていくためのアプリケーションデザインパターンの重要性について解説した．MECを有効利用するためにはこれらのベストプラクティスを集積し，その知見を再利用することで，5G時代の新しい付加価値サービスの創造につながっていくと考えている．今後の機能提供方針として，デザインパターンそのもののバリエーションを増やすことで，より広範なアプリケーションに適用できるように努めていくとともに，よく利用されるデザインパターンをより活用しやすくするために，PaaS機能（デザインパターンを実現するための機能を自ら実装せずに簡単に使えるようにする機能）を検討し，機能具備していきたい．