公共分野のシステムにおける高信頼性を確保するための施策について

新居 史彦 Nii Fumihiko
西村 紀昭 Nishimura Noriaki
成田 高央 Narita Takao
油田 靖文 Aburada Yasufumi

公共分野の大規模なシステムは，複数システムで構成されていることが多く，システム面においては処理性能の確保やシステム連携時の整合性を図ること，業務の面においては法的制限などを意識した処理・権限の制御を可能にすることが求められる。現時点において，多くの公共分野の大規模システムでオープンシステム化が進められているが，具体的な事例をもとにオープンシステムにおいて高信頼性を実現するための施策を紹介する。

1. はじめに

2000年に日本政府が掲げたe-Japan戦略，2003年のe-Japan戦略Ⅱを受けた各府省庁の業務・システム最適化計画[1),2)]は現在も継続して推進されており，その多くがオープンシステムで再構築されている。さらに世界最先端IT国家創造宣言（2015年改定）では，行政情報システムの改革として運用コスト削減の方針[1),2)]が示されたことから，クラウド化を含めたオープンシステム採用の流れは今後も継続すると推測している。

このような状況において，重要な国民サービスを担う各府省庁の情報システムでは，法規制などをはじめとした公共分野特有の運用特性や信頼性確保の考え方があり，オープンシステムにおいてもそれらを実現する必要がある。本稿では，公共分野特有の信頼性についての考え方や実現するための技術について述べる。

2. 公共分野における情報システムの特性

2.1 システムの信頼性

(1) 安定した処理性能の確保

税や年金などを扱う公共分野の情報システムは，重要かつ膨大なデータを扱っているため，システムの停止やデータの消失のみならず，処理の遅延が発生するだけでも社会的な影響を与えてしまうケースがある。

例えば，日中帯に大量のバッチ処理を実行したために窓口サービスの処理が止まってしまう，といったことがないように，オンライン処理とバッチ処理の優先制御が必要となる。さらに，オンライン処理どうしやバッチ処理どうしでも，このような優先制御が必要なケースがある。このように，公共分野のシステムにおける信頼性には，安定した処理性能を確保することも含んでいる。

(2) 業務システム連携時の整合性の確保

大規模なシステムの場合，複数の業務システムで構成されており，システム間の整合性を意識する必要がある。

(a) 同期を取ったメッセージ交換

個々の業務システムにまたがってメッセージ交換を行うケースがあるが，メッセージ送信先の業務システムで障害が発生した場合には双方で同期を取って，データの整合性を確保する必要がある。

オープンシステムで実装するためには，一般的なMQ（Message Queue）製品機能だけでは対応できないので，共通のメッセージ交換機能をおのおのに実装させることとなる。しかし，各業務システムを別ベンダが受け持っていると，業務システムごとにミドルウェアが異なる点やクラスタリング構成を意識する必要がある点など，オープンシステム特有の難しさがある。

(b) ファイル転送のリトライ，キャンセル

業務システム間のファイル転送処理において，転送先システムでの障害発生により処理が継続できない場合は，ファイル転送を中断し，回復後に自動的にリトライ処理を行うことが望ましい。また転送先システムの状況によっては，リトライさせない方がよい場合もあり，このときは転送元でリトライをキャンセルできる機能が必要になる。

これらは標準的なFTP（File Transfer Protocol）のみでは実現できず，メッセージ交換と同様の難しさがある。

(3) 複数サーバにまたがる障害解析・稼働統計

国民へのサービスが主体の情報システムで障害が発生した場合，原因の特定とシステムの復旧（サービスの再開）を，いかに迅速に行えるかが重要である。しかし複数のサーバにまたがる一連の処理において障害が発生した場合，解析に必要な情報は各サーバに分散しているうえ，アプリケーションやミドルウェアにより出力レベルやフォーマットが異なり解析を行うことは容易ではない。

そのため，これらの情報を基に迅速な障害解析を実現することが必要となる。また，通常状態においても業務処理量の変動を監視するための稼働統計の機能も必要となる。

2.2 業務の特性

(1) 業務プログラムの保守性の確保

公共分野のシステムでは，法令改正の施行日に合わせて短期間でのプログラム修正が必要になる場合がある。そのため，各プログラムの独立性が高く，他のプログラムに与える影響が極力小さいことが望ましい。

これを実現するためには，SOA（Service Oriented Architecture）などの考え方を採用し，各業務プログラムが独立して動作できるようなフレームワークを提供する必要がある。

(2) 法的期限を意識したバッチ処理

公共分野のシステムでは，申請や納付など期限付きの手続きを処理する場合が多く，バッチ処理が業務の処理からもマッチしていることが多い。なお，バッチ処理については，用途に応じてセンター起動バッチ処理とオンライン起動バッチ処理が必要となる。

センター起動バッチ処理は，大量のデータを高速で処理する場合や，日中帯にオンラインで更新されたデータを夜間に処理する場合に利用される。また取り扱うデータ量が多いことから，日をまたいでも分割して処理できるように，バッチ処理の中断と翌日の再開始ができるように工夫されていることがある。

オンライン起動バッチ処理では，利用者が画面から指定した条件による大量データの検索や，利用者が指定した条件に基づく大量帳票の出力など，オンラインでの入力情報を起点にバッチ処理を起動させる際に利用される。

(3) 出力帳票の制御

公共分野のシステムでは，国民・市民を対象とした管理資料や，送付票など漏えいすると社会的影響が大きい帳票が存在する。これらの帳票はその秘匿性により不要に帳票を出力することや，利用想定者以外が参照することを防止するなどのセキュリティに配慮した運用がなされている。

(4) 業務プログラムの排他制御

公共分野のシステムでは，法律に関わる処理順序の制限や担当職員の実行権限の制限などから，同時に実行できない業務プログラムがあり，排他関係にしたがって制御する必要がある。また特定地域の業務処理でプログラム障害やデータ破壊が発生した場合に，その影響を全国規模に拡大させないように地域単位で排他関係を設定したり実行を制御したりする必要がある。

このような業務プログラムの排他制御や実行制御は，アプリケーションサーバの機能だけでは通常は実装できず，またある程度実装できたとしても保守性は著しく低い。これらの機能をベンダ固有のミドルウェア上で，アプリケーションインタフェースとして共通化するミドルウェア（共通基盤）をシステムに提供できるかが大きなポイントである。

3. 信頼性を確保するための施策

3.1 信頼性確保のための方針

各特性に基づき必要となる機能については，以下の方針に基づき対応を行った。

・業務プログラムと連動が必要な機能を，アプリケーションフレームワークとして提供するとともに，業務プログラムの保守性を高める。これに含まれる機能は，安定した性能を確保するための制御機能，メッセージ交換およびファイル転送などのシステム間連携機能，障害解析や稼働分析を支援する機能，オンライン起動バッチ処理の機能，出力帳票に関わる機能，業務プログラムの排他制御などとする。

・主にセンター起動バッチ処理に関する機能において，既存の製品やユーティリティに関わる部分は，必要に応じて機能の拡張などで対応する。

3.2 対応施策

(1) アプリケーションフレームワークについて

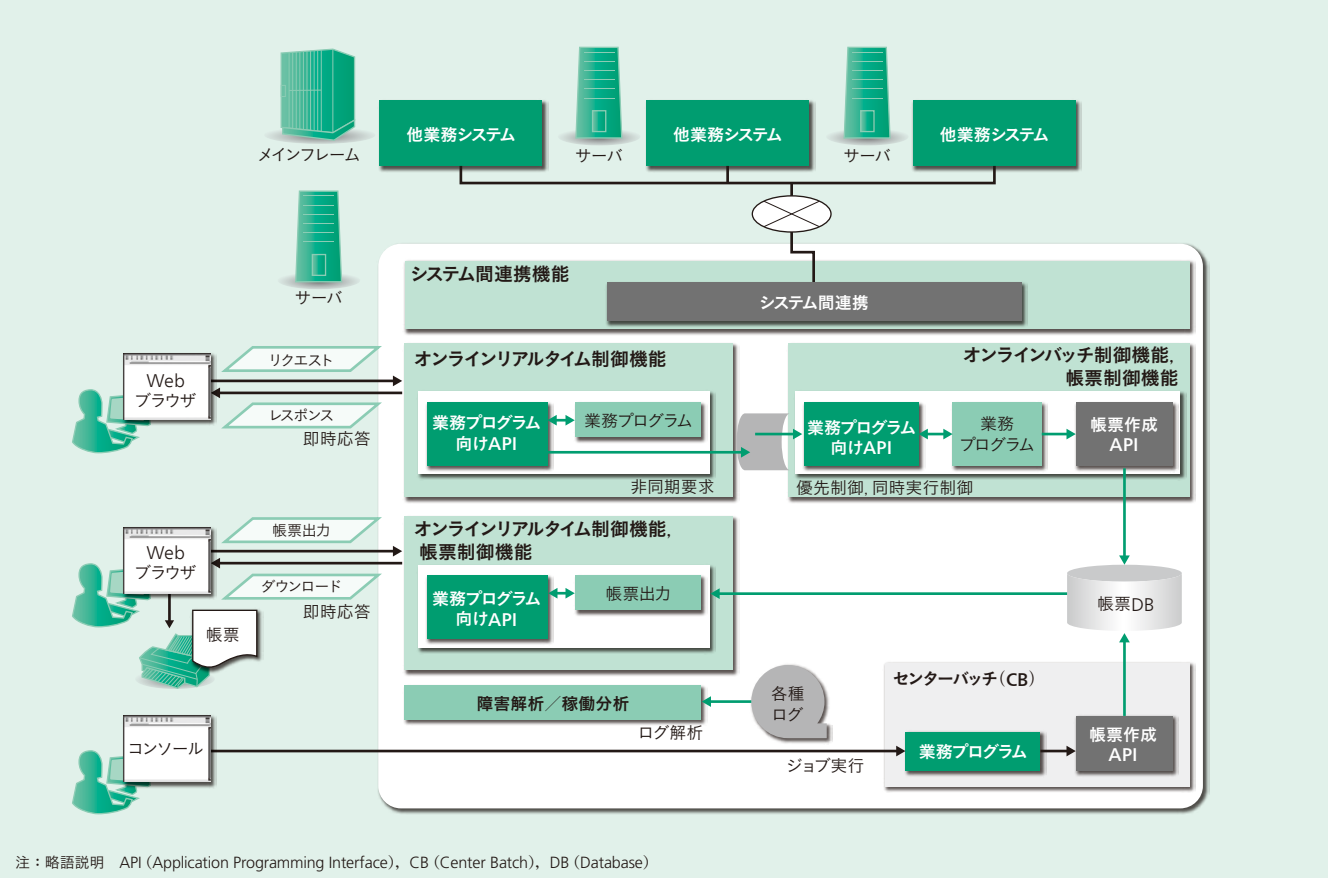
以下の機能を持つアプリケーションフレームワークを開発し提供した。

開発者はアプリケーションフレームワークが提供するAPI（Application Programming Interface）を利用することで，業務ロジックに特化した開発を行うことができる（**図1**参照）。

(a) オンラインリアルタイム制御機能

Webブラウザからのリクエストに対して業務処理結果を即時応答するためのAPIであり，業務プログラムの実行可否チェックや実行抑止などを行う。

(b) オンラインバッチ制御機能



注：略語説明　API (Application Programming Interface)，CB (Center Batch)，DB (Database)

図1｜共通基盤の処理機能イメージ

共通基盤で提供している機能を利用した業務処理の流れを示す。

リアルタイム処理から非同期で業務プログラムを起動させ，優先度に応じたスケジューリングやサーバリソースを効率的に活用する多重度制御などを行う。

(c) 帳票制御機能

全国民向けの送付票や管理資料など出力帳票の作成に対応する機能である。利用者の権限に応じた参照範囲の管理，保存期間の設定が可能であり，さらに帳票管理製品EUR (End-User Reporting) と連携させることでクライアント端末への帳票データの保存を抑止できるため，帳票データに対する機密性を高めることができる。

(d) システム間連携機能

他業務システムとのシームレスなデータ連携を行うために，相手システムと通信を確立する処理，システム間で同期を取るためのロールバック処理などを実現し，さらに業務アプリケーションに極力影響を与えないようAPIとして提供する。運用要件に柔軟に対応するために，オープンシステムとメインフレームの両方に対応可能であり，小規模データを扱うメッセージ連絡と大規模データを扱うファイル形式連絡の2つの方式にも対応することができる。

(e) 障害解析・稼働分析

複数サーバにまたがる障害が発生した場合にログの集約や解析にかかる負担を軽減するため，実行コマンドの形式で複数サーバにまたがって出力されるアプリケーションのログやジャーナル情報をひも付け，時系列に編集することができる。

また法令改正などで業務処理量の変動が見込まれる場合に，あらかじめサーバリソースの過不足やボトルネックの有無を見極める必要があるため，実行コマンドの形式で業務処理別のトランザクション件数や単位処理時間を分析するための情報を編集することができる。

(2) センター起動バッチ処理について

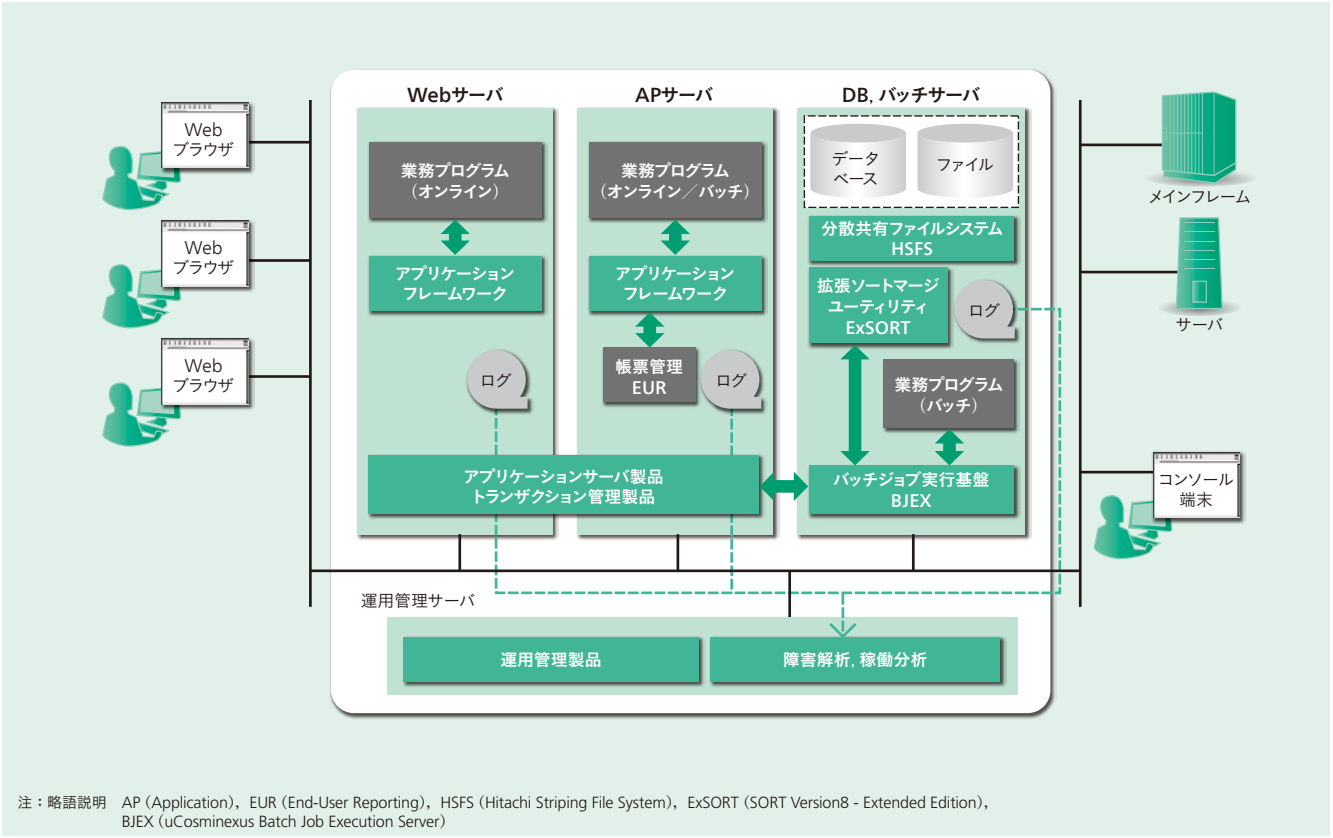
公共分野のシステムでは大量のデータを処理するため，データの抽出や編集の処理能力や方法を工夫する必要があり，製品改修を重ねながらこれを実現した（図2参照）。

(a) 拡張ソートマージユーティリティ

拡張ソートマージユーティリティSORT Version8 - Extended Editionでは，レコードの選択，集約，編集，ファイル分割などメインフレームと同様の機能を備えており，各種データを高速で効率よくソートまたはマージすることができる。

(b) 分散共有ファイルシステム

一般的なオープンシステムでは，複数のアプリケーションサーバなどを並べることで性能を確保していることから，サーバ間でのデータ転送時のネットワーク影響やディ



注：略語説明　AP (Application)，EUR (End-User Reporting)，HSFS (Hitachi Striping File System)，ExSORT (SORT Version8 - Extended Edition)，BJEX (uCosminexus Batch Job Execution Server)

図2｜オープンシステム構成例
オープンシステムは，Webサーバ，APサーバ，DB，バッチサーバ，運用管理サーバなどの用途に応じた複数のサーバで構成される。

スクリソースの効率的な利用について考慮する必要がある。分散共有ファイルシステムHitachi Striping File Systemは，複数のLogical Unitを束ねて1つのファイルシステムを構成することが可能であり，複数サーバから同時にアクセスできるようになる。

(c) バッチジョブ実行基盤

バッチジョブ実行基盤uCosminexus Batch Job Execution Serverでは，ジョブ定義やスプール機能をオープンシステムで実現している。メインフレームからオープンシステムへ移行を行う場合は，既存のJCL (Job Control Language) を流用可能である。

4.　おわりに

公共分野で特に重要かつ大量のデータを扱うシステムにおいて，いかに信頼性を確保してきたかについて述べてきた。

各府省庁の業務・システム最適化計画は継続して実施されているうえに，2015年からはマイナンバー制度が開始され，高い信頼性を求められるシステムの範囲は今後さらに広がっていくと考えられる。日立では，これまでに培った技術を基に，アプリケーションフレームワークの改良やユーザー規模に合わせた提供機能，提供形態の拡充を図り，これらの要望に応えていく所存である。

参考文献など

1)　首相官邸ホームページ，
http://www.kantei.go.jp/
2)　総務省ホームページ，
http://www.soumu.go.jp/

執筆者紹介

新居 史彦
日立製作所 情報・通信システム社 公共システム事業部
官公ソリューション第一本部 官公システム第一部 所属
現在，アプリケーションフレームワーク製品の企画・開発，公共分野のシステム提案・構築作業に従事

西村 紀昭
日立製作所 情報・通信システム社 公共システム事業部
官公ソリューション第一本部 官公アプリケーション第一部 所属
現在，アプリケーションフレームワーク製品の企画・開発に従事

成田 高央
日立製作所 情報・通信システム社 公共システム事業部
官公ソリューション第一本部 官公システム第一部 所属
現在，公共分野のシステム提案・構築作業に従事

油田 靖文
株式会社日立公共システム ソリューション第4事業部
システムサービス第5部 所属
現在，公共分野のシステム提案・構築作業に従事