WebSphere® SIP基盤チューニングの指針

岩品 友徳

Lessons Learnt from the Tuning of WebSphere SIP Systems

Tomonori Iwashina

通信系と情報系を融合した新サービスの基盤として次世代ネットワーク（NGN）が注目を浴びている．NGN では Session Initiation Protocol（SIP）が通信制御を行う基本プロトコルとして使用される．このため，IBM が NGN を実現するために提供するサービス・デリバリー・プラットフォームにおいては，WebSphere Application Server Network Deployment の SIP コンテナーが中心的な役割を果たす．本論文では，国内通信業界のお客様環境での基礎検証の結果を踏まえ，WebSphere SIP 基盤の特色，WebSphere SIP 基盤におけるパフォーマンス・チューニングおよび障害対応の方針を明らかにした．本論文の知見を踏まえることで，今後急速に普及するであろう高可用性 SIP システムに対する具体的な設計指針を示すことができる．

A Next Generation Network (NGN) is a new infrastructure which facilitates the provision of new, innovative applications. In the NGN environment, the WebSphere Session Initiation Protocol (SIP) container functions as the fundamental infrastructure. This paper discusses the lessons we have learnt from the tuning of WebSphere SIP Systems based on our experience of performing a product verification test at a customer’s site.

Key Words & Phrases：次世代ネットワーク，SIP，WXS，WRT
Next Generation Network, Session Initial Protocol

1. はじめに

次世代ネットワーク（NGN: Next Generation Network）は，従来の回線交換式電話網を代替する音声通信網として，また帯域保証や回線認証などの QoS（Quality of Service）技術をサポートする高品質なデータ通信網として，さらには音声，映像，データを組み合わせた新しい統合通信インフラとして，今後急速に発展していくことが期待されている．

2004 年より ITU-T（国際電気通信連合 電気通信標準化部門）により NGN の標準化作業が開始され，2006 年 7 月には NGN 勧告リリース 1 が完成，NGN の全体像が提示された［1］．公共網における商用サービスとしても NGN は動き出しており，2008 年 3 月には国内通信事業者が世界に先駆け NGN サービスの提供を開始している［2］．公共網に限らず，企業内情報システムにおいても，音声通話やチャット・システム，Web 会議システムなど多種多様なサービスを統合し，かつ高いセキュリティーを実現する情報基盤として NGN の拡大が期待されている．

通信系と情報系を融合したさまざまな新サービスが構想され，各社ともども新たなビジネス・チャンスとして取り組みを強化している．IBM もサービス・デリバリー・プラットフォームとして，NGN で稼働する多様なソフトウェア群を発表している．NGN においては，サービスの開始・終了などの呼制御をはじめ，帯域制御や認証，課金などの情報連携にも SIP（Session Initiation Protocol）が用いられるため，IBM のソフトウェア群の中でも WebSphere Application Server Network Deployment（以下，WAS ND）の SIP コンテナーが，特に重要な役割を果たす．

今回，筆者は国内通信業界のお客様環境において，SIP コンテナーの基礎パフォーマンス検証を実施する機会に恵まれた．お客様環境での経験を踏まえ，本論においては，通信業界のお客様で必要とされる“キャリア・グレード”と呼ばれる高いサービス品質を実現するために必要な SIP 環境のパフォーマンス・チューニングおよび障害対応のための具体的指針を示す．

提出日：2009年9月7日　再提出日：2010年6月11日

2. SIPコンテナーのパフォーマンス検証環境概要

まず今回の検証環境の具体例を示すとともに他社の一般的なSIPシステムとの比較を行い，IBMのWebSphere SIP基盤の特色を示す．

2.1 検証環境

サービス制御を行うSIPアプリケーションが稼働するWAS NDのSIPコンテナーは，通常のWebのアプリケーション・サーバー環境と同様，クラスタリングによって可用性，拡張性およびパフォーマンスを確保する構成を取る．今回の検証環境では，物理ノード2台上に複数プロセスのアプリケーション・サーバー（SIPコンテナー）を配置し，その前段の物理ノード2台の上にプロキシーの役割を果たすオンデマンド・ルーターを配置した．さらにオンデマンド・ルーターの前段には負荷分散装置を配置している．SIP要求はUAC（User Agent Client）から負荷分散装置を経由してオンデマンド・ルーターに送られ，オンデマンド・ルーターはアフィニティーを意識してSIPコンテナーへと割り振りを実施する．SIP要求は，SIPコンテナーで折り返されUAS（User Agent Server）へと届く．UASは同様にして応答をUACへと返す（図1）．

SIPコンテナーにおけるセッション複製の仕組みとしてはWebSphere eXtreme Scale（以下，WXS）を採用し，WXSの非同期レプリケーションおよびサービス・メソッドの終了（end-of-service）のセッション複製モードを採用した．

2.2 WebSphere SIP基盤の特色

比較のために，他社の一般的なSIP基盤環境を図2に示す．他社のSIPコンテナー環境では，負荷分散装置の後段に直接SIPコンテナーが配置される構成が多いようである[3]．セッション保管のための仕組みとしては，一般的にはデータベースが採用されるが，高速処理および大規模トランザクションをさばくことが可能なようにインメモリーのデータベースを採用する構成も見られる［4］．

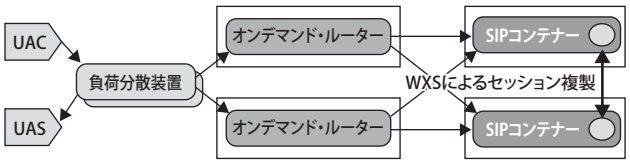

図1．WebSphere SIP基盤

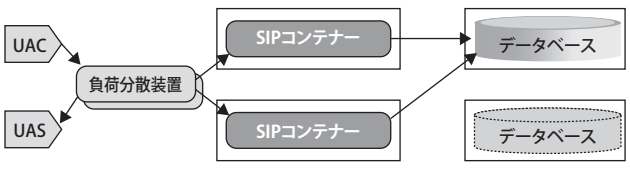

図2．他社のSIP基盤の例

(1) セッション複製の仕組み

まずIBM構成の特徴として挙げられるのは，SIPセッション複製のためにWebSphere eXtreme Scaleを使用している点である．

従来のHTTPセッションと異なり，SIPセッションには課金に必要なデータやサービスに関する情報が含まれており，いかなるケースにおいても損失は許されない．WAS NDで提供されるメモリー間セッション複製の仕組みであるDRS（Data Replication Service）はベスト・エフォート型のプロトコルであり，高負荷時にはセッションを損失する可能性がある．その意味で，高信頼性が要求される環境では保証型のセッション複製の仕組みを提供するWXSが必要となる．

WXSは，eXtreme Scaleの名の通り拡張性に優れた分散キャッシング製品であり，管理コンポーネントに至るまで，単一個所にボトルネックが集中しないようデザインされている．公共網など超大規模ともなり得る次世代ネットワーク・システムを想定した場合に，1,000プロセス以上のリニアな拡張性が確保できるWXSの優位性は非常に大きいといえる［5］．

一方，他社のSIPコンテナー環境はセッション情報をインメモリー・データベースに保管している．WebSphereは，現在，IBMのインメモリー・データベースであるSolidDBをサポートしていないためこの構成を取ることはできないが，ディスク・ベースのデータベースであればセッションの保管先として選択することはできる．しかしHTTPセッションであれば，ページ遷移のタイミングなど数秒～数分の間隔で更新が走るレベルである一方，SIPセッションは最も書き込み頻度を落としたサービス・メソッドの終了ごとに書き込む複製モードの場合でも，ミリ秒単位で更新が発生するため，ディスク・ベースのデータベースでは処理性能が現実的ではない．インメモリー・データベースはクラスタリング構成を取ることが可能となる構成もあるが，拡張性を旨とする分散キャッシング技術に比べその拡張性は限定的であり，この点がIBMの優位性といえる．

(2) プロキシー・コンポーネントの存在

他社の構成と比較した際に，特徴的といえるのがプロキシー・コンポーネントの存在である．WAS ND環境であればJavaプロキシー，WAS NDを拡張したWebSphere Virtual Enterprise（以下，WVE）環境であればオンデマンド・ルーターがこれに当たる．

このプロキシー・コンポーネントは，後段のアプリケーション・サーバーへの負荷分散やアフィニティー制御の役割のほか，WebSphere 基盤内で過負荷制御や流量制御を実施する役割も果たしている．また，SIP コンテナーと同一のセル環境（WAS ND の管理単位であり，プロセス間の連携が暗黙的に行われる）に所属しているため，障害発生時にはセッション複製情報が保持されている SIP コンテナーに対して直接要求をルーティングするホット・フェイルオーバーが可能であるという利点がある．

これに比して，他社サーバーは後段のセッション DB にセッション情報を共有しているため，いずれのアプリケーション・サーバーに割り振りを行っても問題がない．このため負荷分散装置からアプリケーション・サーバーが直接リクエストを受け付ける構成が可能となっている．しかし逆にいえば，SIP コンテナー障害発生時には，障害発生コンテナーが保持していた全セッションへの参照要求がデータベースに流れることを意味し，障害時にシステム負荷が急激に上がる可能性がある．

一方，プロキシー・コンポーネントが存在するためのデメリットも存在する．1 つ目は Java のガーベッジ・コレクション（以下，GC）につかまる可能性が高まる点である．他社サーバーにおいては，負荷分散装置から直接リクエストが割り振られるため，各要求が GC でつかまる可能性は 1 回のみである．一方，WebSphere 環境においては，SIP コンテナーで 1 回，往復のオンデマンド・ルーターで 2 回の計 3 回ある．単純に考えて，GC でつかまる可能性が飛躍的に高くなるため，リアル・タイムの応答性能が求められる SIP 環境においては，GC ポリシーの選定が非常に重要となる．これは後段にて別途論じる．

さらに配慮すべきはプロキシー・コンポーネントにおける SIP 要求の追い越しである．SIP コンテナーにおいては，SIP 要求はそのコール ID により処理されるべきスレッドがハッシュ・アルゴリズムにより決定されるので，順序制御が保障され，必ず First In First Out で処理される．一方，ステートレスなプロキシー・コンポーネントであるオンデマンド・ルーターでは，SIP プロトコルの要求内容については特に意識をしないため，同じ呼に関連付けられた連続した要求が，別スレッドで同時に処理され，追い抜きが発生する可能性がある．従って，追い抜きが発生した SIP 要求への対応は SIP クライアント（UAC，UAS）および SIP コンテナー側にて対応しなければいけない．

3. GCポリシーの選択

従来，専用機開発が当たり前であった通信業界において，そのシステムへの要求レベルは非常に高い．Java の豊富なライブラリー群や豊富な製品群を使用して開発生産性を高め，ビジネスへの即応性やコスト・メリットなどが期待される場合においても，キャリア・グレードと呼ばれるシステムへの高い要求を満たすことが大前提となる．

今回，パフォーマンス検証を実施させていただいたお客様においても，最も重視されていた（または最低限レベルでの）条件は，通常時において SIP 要求の再送を許さないという点であった．

SIP においては，セッションの確立を開始する INVITE 要求もそのほかの非 INVITE 要求も，500 ミリ秒応答が返されない場合には UAC により最初の再送が行われ，以降一定間隔で再送が行われる［6］．これは UDP による実装が一般的である SIP アプリケーションにおいて，パケット・ロス発生に備えるための仕組みである．たとえ再送が発生した場合でも，UAS において同一要求が二重に処理されることはなく特に問題は発生しない．しかし公共網として稼働する環境において，平常時から再送が発生することは許容されない．Java においてはメモリー管理のためのガーベッジ・コレクションが存在するため，この再送を許さないという条件は非常に厳しい条件となる．

今回のパフォーマンス検証においては，通常の IBM JDK で採用することの可能な 3 つの GC ポリシー（optthruput，optavgpause，gencon）に加え，Java でのリアルタイム処理を規定する仕様 Real Time Specifications for Java（RTSJ）を実装した JVM である WebSphere Real Time のメトロノーム GC を採用して検証を行った．

3.1 GCポリシーの検討

（1）optthruput

IBM JDK でデフォルトとして採用されている GC ポリシーであり，最も高いスループットを期待することができる．この GC ポリシーでは Java ヒープは従来通り単一領域として扱われる．アロケーション失敗時には STOP THE WORLD 方式で GC 以外の全スレッドの処理を停止して GC が行われる．使用しているオブジェクトをチェックするマーク・フェーズ作業も GC 中に実施されるため，GC の停止時間が最も長くなる．

（2）optavgpause

この GC ポリシーを採用した場合も，optthruput 同様 Java ヒープは単一の領域として扱われ，STOP THE WORLD 方式で GC が行われる．しかし，通常のアプリケーション処理を実施している裏でマーク・フェーズの処

理を並行して行う（Concurrent Mark）ため，通常時のスループットは若干犠牲となるが，GC 発生時の停止時間を最小化することができる．

（3）gencon

gencon の GC ポリシーを採用した場合，Java ヒープは 2 つの領域に分けられ，2 つのタイプの GC が実施される．生成されたオブジェクトはまず Nursery と呼ばれるヒープ領域にアサインされる．Nursery 領域は Allocation Space と Survive Space という 2 つの領域に分かれている．この 2 つの領域間で生存するオブジェクトをコピーし，各領域が切り替わることで GC を実現するコピー GC が採用されている．これは，GC 以外の処理スレッドが停止する STOP THE WORLD 方式ではあるが，比較的短時間にて GC を行うことができる．この Nursery での GC を一定回数経たオブジェクトは，長時間生存するオブジェクト用の領域である Tenured の領域に移動される．Tenured の領域では optavgpause ポリシーを採用した場合と同様の仕組みにより GC が行われる．

（4）metronome

メトロノーム GC は WebSphere Real Time を採用することで使用することができる GC ポリシーである．WebSphere Real Time は，Real Time OS と呼ばれる特殊な OS 上であれば，RTSJ（Real Time Specification for Java）仕様［6］に従ったハード・リアルタイム JVM（Java Virtual Machine）として使用可能であるが，通常の OS ではソフト・リアルタイムを実現するメトロノーム GC のみがサポートされる．なお WebSphere Real Time は，まだ通常の WAS ND ではサポートされておらず，WXS または WVE と組み合わせて使用する場合にのみサポートされる点に注意する必要がある．

メトロノーム GC では，これまで数十～数百ミリ秒かかっていた GC 処理を数ミリ秒単位に分割し，分割された GC と通常のアプリケーション処理を，メトロノームのように交互に切り替えながら実施する（図 3）．これにより従来のような長時間の GC につかまることを回避でき，リアルタイム性が求められるアプリケーションを Java で稼働させることができるようになる．

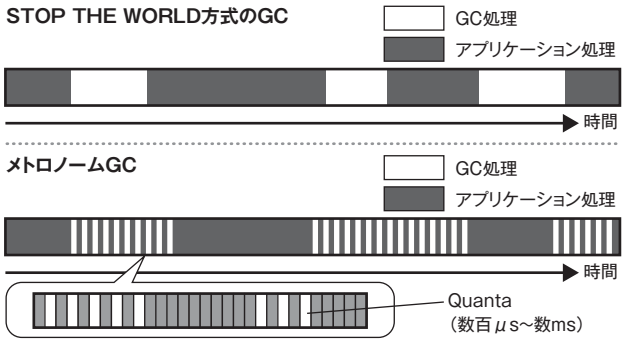

図 3. GC の方式

3.2 SIP環境でのGCポリシーの検証

お客様の基盤要件を反映し，単一ノード 32bit の単体 JVM，1.7GB のヒープサイズの SIP コンテナーにおいて，30 分間再送および呼損の発生しない最大持続スループットをそれぞれの GC ポリシーで計測した．その結果が**表 1** である．

通常，停止時間の最小化を検討するシステムでは gencon が採用されることが多い．しかし，実際の検証結果としては optavgpause の方がよりスループットを出すことができた．

gencon および optpause の各ケースの GC の挙動をそれぞれ**図 4** および**図 5** に示す．横軸は GC 開始からの時間（分），左軸はヒープ使用量（MB），右軸は GC 時間（ミリ秒）である．gencon のケース（**図 4**）では，Nursery 領域の 70 ミリ秒前後の GC が繰返し発生しているが，Tenured 領域で GC が発生した際に，マーク処理で比較的長い時間が掛かっていた．これはヒープの多くを占める SIP セッションがサービスの開始から終了まで長時間保持されるオブジェクトであるため，必ず Nursery 領域から Tenured 領域への移行が走り，Tenured 領域での GC の際に大量のオブジェクトを GC する必要があるためである．これは Tenured 領域での GC が発生するまでヒープ使用量が減少していないことからも分かる．結果として単一のヒープ領域でより頻繁に GC を行っている optavgpause（**図 5**）の方が，Concurrent Mark の効果もあり，停止時間も短くより高い最大持続スループットを出すことができた．

このように長時間オブジェクトとなる SIP セッションの保持が必要となる SIP コンテナーにおいては gencon 採用時のメリットである Nursery の領域でのコピー GC の恩恵は受けづらい．従って通常の IBM JDK を採用する場合にあっては，SIP コンテナーには optavgpause の GC ポリシーを選択すべきである．一方で，オンデマンド・ルーターなどのプロキシー・コンポーネントはステートレスであるため，非常に短時間の間生存するオブジェクトが

表 1. GC ポリシー毎の最大スループット

GC ポリシー	スループット（Call Per Sec）
optavgpause	250 CPS
gencon	170 CPS
metronome	430 CPS

ほとんどである．このような場合では gencon ポリシーを採用して Nursery 領域でほぼ GC を終えることができ，gencon の恩恵を享受することが可能である．実際に今回の検証結果としてもオンデマンド・ルーターの GC ポリシーとしては,gencon で最も高いスループットが得られた．

しかし，今回の SIP 検証において，最も効果を示したのは，やはり WebSphere Real Time のメトロノーム GC であった．図 6 にメトロノーム GC の挙動を示す．横軸は検証開始からの時間（分），左軸は 1 回の GC での Quanta（細分化された GC 処理単位）の回数，右軸は 1Quanta の時間（ミリ秒）である．平均で 3 ミリ秒前後の Quanta が 1 サイクルで 80 回ほど実行されている．この値は限界まで負荷をかけた状況のものであるため，通常はより短い Quanta で GC を実施可能である．いかなる要求も数百ミリ秒の長い GC につかまる可能性はなく，安定して応答を返すことができていた．今回の SIP コンテナー検証のように，お客様のアプリケーションが単純なスループットのみを必要とするのではなく，ミリ秒レベルの応答性能と低レイテンシーが求められる場合には，非常に有効な GC ポリシーといえる．

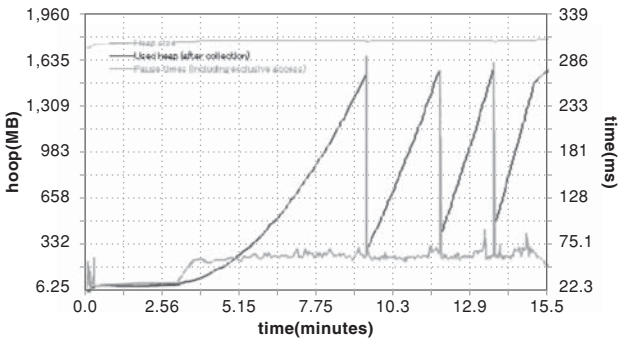

図 4. gencon の GC 結果

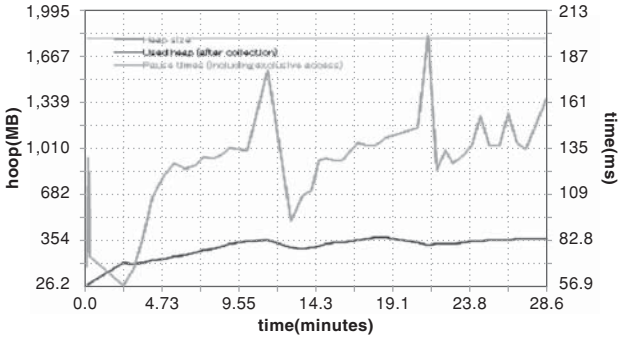

図 5. optavgpause の GC 結果

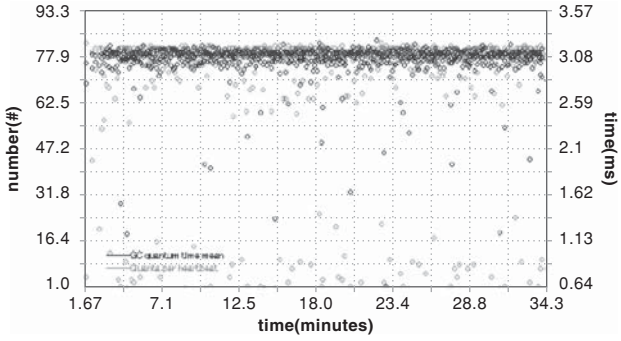

図 6. metronome の GC 結果

4. 障害対応のためのパラメータ

最後に SIP システムにおいて障害に備えるために，注意すべきポイントについて考察する．前述したように，障害時の呼の損失に備えるため SIP には再送の仕組みが規定されている．INVITE 要求と非 INVITE 要求で再送のタイミングは若干異なるが，最後の再送が行われるのは共通で 31.5 秒後である［6］．従って，WebSphere 関連各プロセスは，最悪でも 31.5 秒でフェイル･オーバーを完了させる必要がある．INVITE 要求の最終再送の 1 つ前の再送タイミングは 15.5 秒であるので，今回の検証においては，この 15.5 秒をリカバリーのための目標時間として設定した．なお，この 15.5 秒は非 INVITE 要求においては全 10 回ある再送のうちの 6 回目の再送に当たる．

SIP 基盤の障害時のフェイル・オーバーのチューニングをするに当たり，SIP コンポーネントそのものと，WXS セッション複製の 2 つの観点から考えていく．

4.1 WXSセッション複製のフェイル･オーバー

WXS を使用して SIP セッション複製を行っている環境では，セッション･オブジェクトを保持している区画（シャード）のフェイル・オーバーと実際にセッションへの参照・更新を行う要求のタイムアウト双方について考える必要がある．

（1）区画（シャード）のフェイル・オーバー

WXS では区画のフェイル・オーバーのトリガーとして，通常の WebSphere のプロセス間障害検知の仕組みである High Availability Manager（HA Manager）コア・グループの仕組みを使用している［8］．プロセス障害は，プロセス間で相互に張られた接続が切断されることにより検知されるため，障害時にはほぼ即座に検知可能でありそれほど問題とはならない．

一方，ネットワーク障害およびノード障害に関しては，RST パケットが送られてくるなどのトリガーは特になく，タイムアウトを利用して検知することとなる．これはコア･グループのハートビート・タイムアウト期間として設定する．デフォルトでは，ハートビート伝送間隔 30 秒×ハートビートの連

続欠落数 6 回の計 180 秒に設定されており，最悪 31.5 秒でフェイル・オーバーを完了する必要がある SIP では，設定変更が必要である．

障害検知後，WXS 側で障害発生した区画に関して，セッション複製を保持するレプリカがプライマリーとなり処理を受け付ける用意ができるまで，実測値として 2 ～ 4 秒程度かかるため，目標として設定した 15.5 秒内でフェイル・オーバーを完了させるためには，ハートビート伝送間隔 2 秒とハートビートの連続欠落数 4 回の乗算となるハートビート・タイムアウト期間 8 秒ほどに設定すればよい．連続欠落数が少なすぎる場合は誤検知の可能性が在るので，配慮して設定が必要である．

(2) WXS 要求のタイムアウト

別途注意しないといけないものが，WXS による実際のセッションの参照・更新要求の処理である．WXS の参照・更新処理は CORBA/IIOP で実装され，コア・グループの障害検知の仕組みとは独立して稼働している．EJB の通信にも使用される WebSphere の Object Request Broker（ORB）関連タイムアウトは 180 秒など，従来の JavaEE アプリケーションの稼働を前提としたパラメーターが設定されており，SIP システムにおいては適さない．

WXS がセッション書き込みを実施しようとしたタイミングでネットワーク障害が発生した場合には，当該スレッドに関してはアプリケーションの処理がタイムアウトまで待機させられる．ミリ秒レベルの応答性能が求められる SIP 環境において，180 秒間タイムアウトを待機するのはアプリケーションがハングアップしているのとなんら変わらない．従って，ORB の要求タイムアウトや位置指定要求タイムアウト，接続タイムアウトなど各種タイムアウトを，数秒程度の安定したネットワークを前提とした値に変更することが必須である．

4.2 SIPコンテナーのフェイル・オーバー

SIP コンテナーの障害も，通常の WebSphere のプロセス間障害検知の仕組みであるコア・グループのハートビート・タイムアウト期間により検知される．障害検知後，オンデマンド・ルーターは，セッションの複製先へと SIP 要求をホット・フェイル・オーバーさせることが可能である．通常時からセッション情報は複製されており，それ以外は特に引き継ぐべき情報は SIP コンテナーには保持されないため，WXS 区画のフェイル・オーバーで設定したハートビート・タイムアウト期間 8 秒など十分に短い値に設定しておけば SIP コンテナー引継ぎが問題となることは少ない．

また，オンデマンド・ルーター自体もステート・レスなプロキシーであるので，特に引き継ぐべき情報は保持していない．このため，前段の負荷分散装置において数秒にて障害を検知できるように設定しておけば特に問題とはならない．

5. まとめ

ここまで，実際の検証結果を踏まえ，SIP 環境でチューニングすべき考慮点について論じた．特に SIP 環境においては「止まらない」ということが常に要求される．WebSphere で提供される GC や各種タイムアウト値は，従来の Web システムや JavaEE 環境を前提とした値が設定されており，より厳しい時間制約での対応が求められる SIP 環境においては意識的に見直しを行っていくことが重要である．本論で記述した具体的なチューニング指針が，キャリア・グレードを実現する高信頼性 SIP システム構築の一助となることを確信している．

参考文献

［1］ 森田直孝，平松幸男：” NGN 関連技術の標準化動向 ,” NTT 技術ジャーナル, pp. 24-25 (2008.8).
［2］ 中村浩：“次世代ネットワークによる商用サービスの提供 ,” NTT 技術ジャーナル, pp. 56-57 (2009.8).
［3］ “WebOTX SIP Application Server BIG-IP Local Traffic Manager 連携システム構築ガイド ,” http://www.f5networks.co.jp/shared/pdf/otx_bigip-renkei.pdf (2010.05.17).
［4］ “Storing Long-Lived Call State Data in an RDBMS,” http://download.oracle.com/docs/cd/E13209_01/wlcp/wlss31/configwlss/rdbmsstore.html (2010.05.17).
［5］ Daniel Froehlich et al: User’ s Guide to WebSphere eXtreme Scale, IBM RedBook, SG24-7683 (2008).
［6］ Real Time Specification for Java(RTSJ) , http://jcp.org/en/jsr/summary?id=1 (2010.05.17).
［7］ SIP Session Initiation Protocol (RFC3261), http://www.ietf.org/rfc/rfc3261.txt (2008.04.20).
［8］ “ミッション・クリティカル・システム開発講座第 9 回高可用性の実現手法（後編）,” http://www.itarchitect.jp/enterprise/-/37961-5.html (2006.04.28).

日本アイ・ビー・エム
システムズ・エンジニアリング株式会社
Web システム基盤
主任 IT スペシャリスト

岩品 友徳 Tomonori Iwashina

［プロフィール］
2002 年，日本アイ・ビー・エム システムズ・エンジニアリング株式会社 入社．入社以来，金融や流通のお客様に対して WebSphere 関連製品の技術支援を実施．近年は WebSphere XD を使用した Java バッチシステム構築や WebSphere eXtreme Scale を使用したシステムのデザインを実施．