Point 3

アプリケーションサーバ
トラブルシューティング機能の充実

Cosminexusでは実績のあるオンライン処理技術（OLTP）や現場で使いこなされたノウハウをトラブルシュート機能に盛り込み、強化を図っている。

差分6 J2EEサーバ稼働情報の採取

Point 1	Point 2	Point 3	Point 4	Point 5	Point 6	Point 7
差分1	差分3	差分6	差分9	差分13	差分15	差分17
	差分4	差分7	差分10			
差分2	差分5	差分8	差分11	差分14	差分16	差分18
			差分12			

J2EEサーバを安定稼動させるためには、システムの冗長性や、各種運用機能の設定だけではなく、継続的な各種稼動情報の採取・分析が不可欠だ。そのためには、J2EEサーバから必要な情報を利用しやすい形で採取できることが大前提となる。

稼働情報採取機能を利用する場合、従来はコマンドを投入して情報を採取していたのに対し、Version 7では、あらかじめ設定しておくだけで継続的にファイルに稼働情報が出力されるようになった。主な採取可能情報を次に示す。

- **リクエストの同時実行数、キュー、プールなどの状態**
- **セッション数**
- **リクエスト、トランザクションなどの件数**
- **メモリ、スレッドの状態**

情報はCSV形式で出力されるため、Excelなどで読み込んで容易に分析できる。また、ファイルの上限値や、稼働情報の採取も柔軟に設定できる。

差分7 PRFトレース（障害解析トレース）の拡充

Point 1	Point 2	Point 3	Point 4	Point 5	Point 6	Point 7
差分1	差分3	差分6	差分9	差分13	差分15	差分17
	差分4	差分7	差分10			
差分2	差分5	差分8	差分11	差分14	差分16	差分18
			差分12			

チューニングや障害発生時の原因調査では、ボトルネックがどこで発生しているのか、あるいはリクエストがどこでエラーとなっているのかを突き止めるために、複数サーバに分散するログを横断的に見比べて調査する作業が必要だった。

CosminexusではPRFトレースと呼ばれる、低オーバーヘッドなリクエストトレース機能が標準で提供されており、Version 7では、トレースの対象をSOAPまで拡大するとともに、リクエストの追尾性を高め、性能チューニングやエラー原因の調査を支援するという工夫が図られている。

トレースを出力するWebコンテナ、EJBコンテナなどを、機能レイヤーと呼んでいるが、PRFトレースではトレース情報は各機能レイヤーの入り口と出口で採取されるほか、J2EEサーバの開始処理、終了処理でも採取される。また、アプリケーション開発時などにはトレースの採取レベルを詳細の設定にすると、機能レイヤー内の処理についてもさらにきめ細かい情報が採取できるため、デバックに役立つ。

各機能レイヤーで出力されるトレースのログには、リクエスト単位にユニークな識別子が付与される。トレース情報としては、この識別子の他、リクエストが通過した時刻やそれぞれのポイントでの付加情報が採取され、これらのログを運用監視ツールで1カ所に収集し、共通の識別子で絞り込めば、処理を容易に把握できる。PRFトレースでは、アプリケーションが共用メモリに書き込んだトレースを、独立したデーモンプロセスが非同期にファイルに出力するため、オーバーヘッドが少ない。またアプリケーション開発時だけでなく、本番運用時に利用するこ

とを想定して、デフォルト設定は「有効」となっている。また、データベースのコネクション ID が採取されるため、データベースアクセスまで含めた処理シーケンスを把握することができる（図 9）。

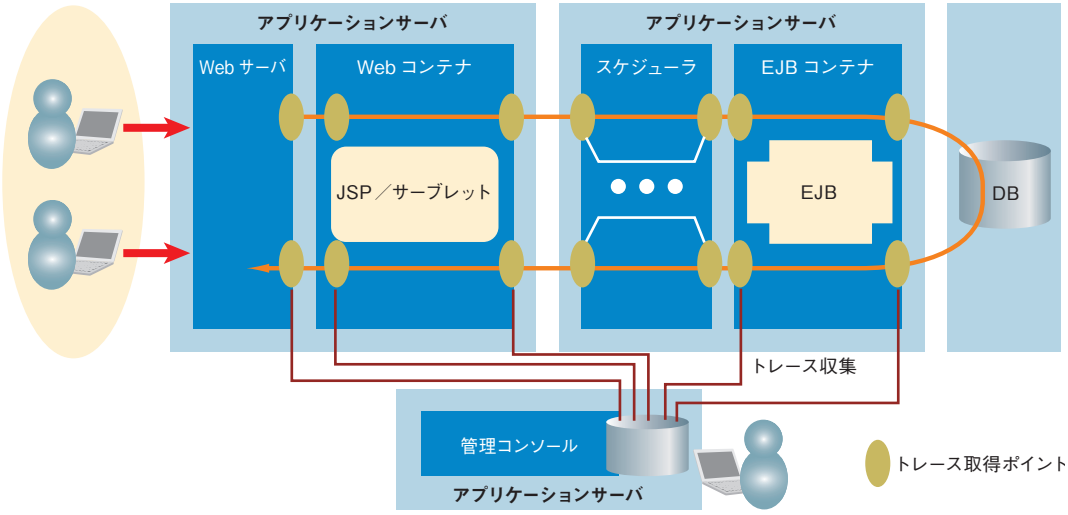

図 9　障害のトリガーとなったリクエストや処理性能のボトルネック箇所の容易な特定

差分 8　JavaVM のスタックトレースの拡張

Point 1	Point 2	Point 3	Point 4	Point 5	Point 6	Point 7
差分 1	差分 3	差分 6	差分 9	差分 13	差分 15	差分 17
	差分 4	差分 7	差分 10			
差分 2	差分 5	差分 8	差分 11	差分 14	差分 16	差分 18
			差分 12			

Cosminexus の安定稼動と性能確保を支える要因として、Cosminexus JavaVM の効果が挙げられる。日本語機能や、頻繁に使用する比較機能などは、徹底的にブラッシュアップされている。さらに注目したいのが独自に拡張され、充実したトレース機能である。

❶スタックトレースでのローカル変数出力（障害原因の容易な把握）

Java プログラムで何らかの例外が発生した場合、デバッグ用の情報として例外情報とスタックトレースが出力される。各種プラットフォーム（OS）対応の Cosminexus JavaVM では従来のスタックトレースに加えて、そのメソッド内のローカル変数の内容を挿入して出力。本機能により、例外発生に至る経緯を把握して、原因を早期に突き止めることが可能となる（図 10）。

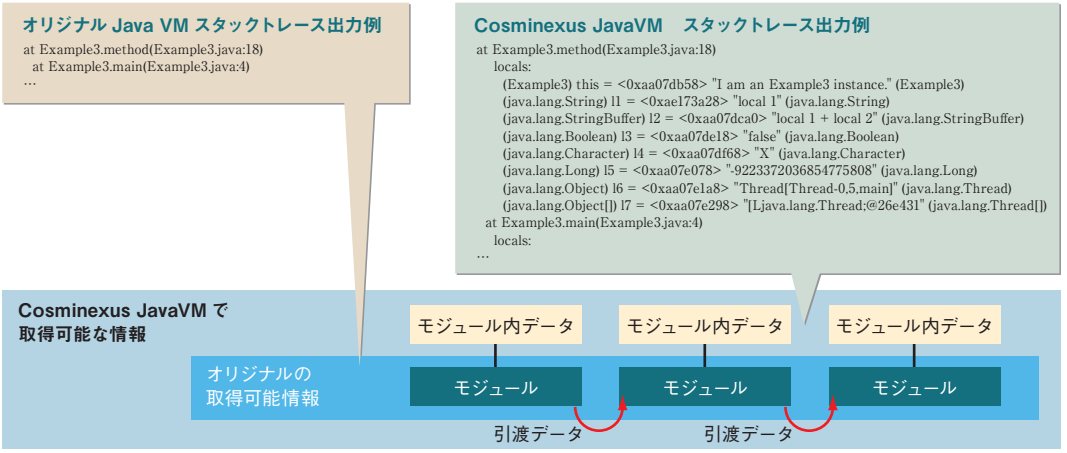

図 10　スタックトレースでのローカル変数出力

❷メモリ使用状況の出力

プログラミングエラーによるメモリリーク（オブジェクト参照の解除漏れ）の原因を容易に追求することが可能になる。本機能も製品に組みこまれて提供されているため、一般的にテスト環境では再現が困難なメモリリークの不具合を、本番環境で問題が発生した時点でも情報取得することができる（図 11）。

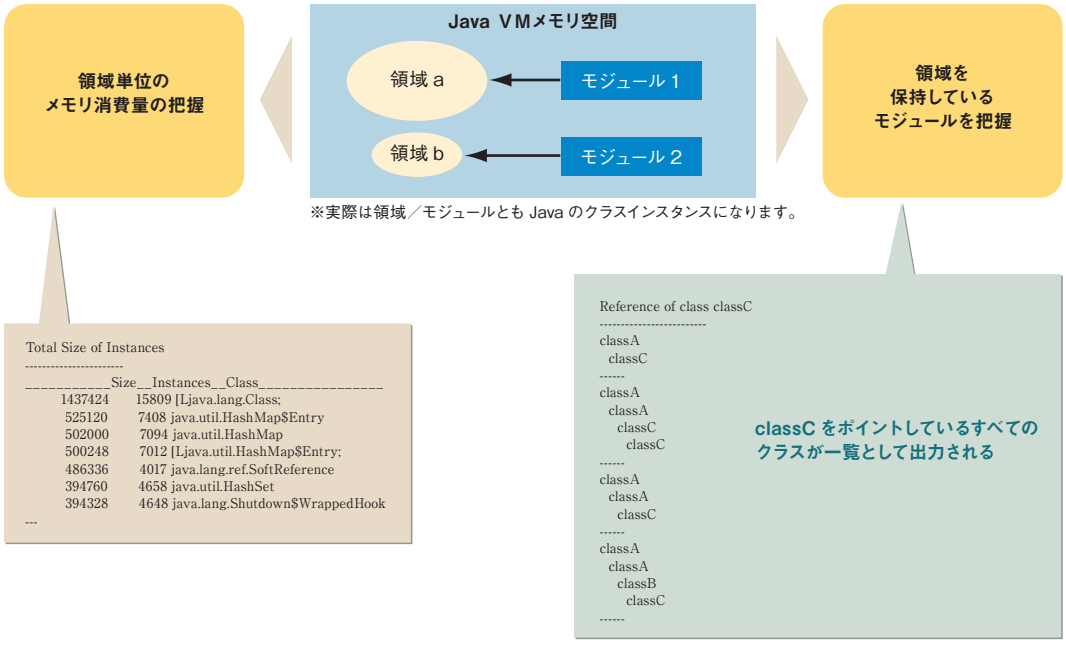

図 11　メモリリークの原因追及

❸時間を要しているスレッドの特定

さらに、Version 7 では、原因を特定するための各種情報を取得できるようにした。たとえば、スレッドごとの CPU 消費時間や待ち回数などを出力。これにより、どのスレッドに時間を要しているかが判明するため、性能障害の特定が容易になった（図 12）。

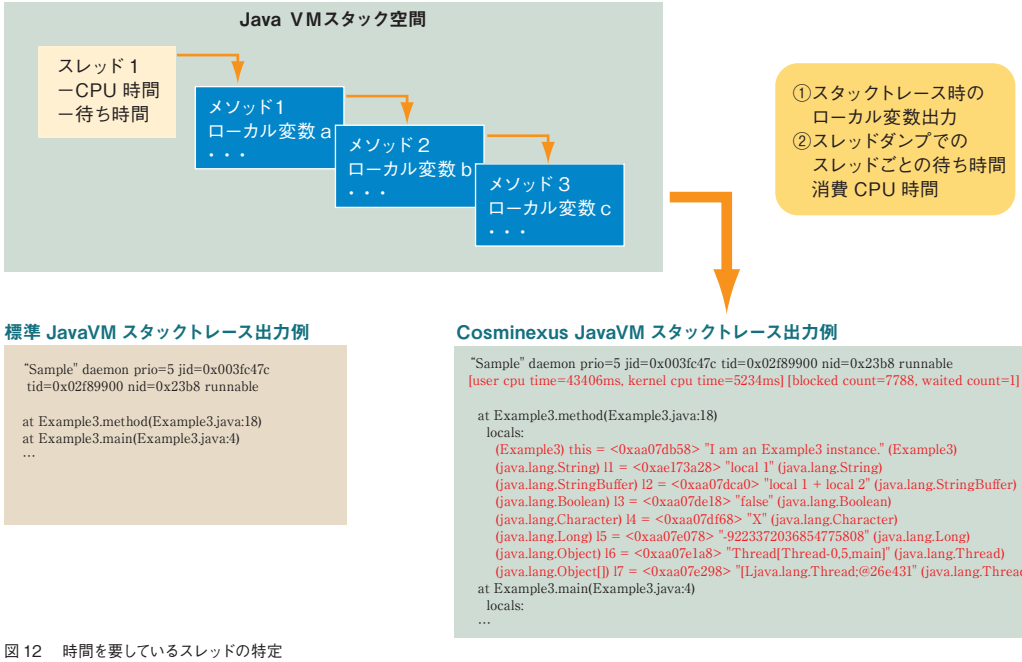

図 12　時間を要しているスレッドの特定