

記事

[Tomohiro Iwamoto](#) · 2020年5月7日 15m read

データプラットフォームとパフォーマンス-パート7 パフォーマンス、スケーラビリティ、可用性のためのECP

Cachéの優れた可用性とスケーリング機能の1つは、エンタープライズキャッシュプロトコル (ECP) です。アプリケーション開発中に考慮することにより、ECPを使用した分散処理は、Cachéアプリケーションのスケールアウトアーキテクチャを可能にします。アプリケーション処理は、アプリケーションを変更することなく、単一のアプリケーションサーバーから最大255台といった非常に高いレートにまで、アプリケーションサーバー処理能力を拡張できます。

ECPは、私が関与していたTrakCareのデプロイメントで長年広く使用されていました。10年前は、主要ベンダーの1つが提供する「大きな」x86サーバーは、合計で8つのコアしか備えていなかったかもしれません。大規模なデプロイメントの場合、ECPは、高価な大型コンピュータを使う単一のエンタープライズサーバーではなく、コモディティサーバーでの処理をスケールアウトする方法でした。コア数の多いエンタープライズサーバーでさえ制限があったため、ECPはそれらのサーバーへのデプロイメントのスケールアップにも使用されました。

現在、ほとんどの新しいTrakCareのデプロイメントや主流ハードウェアへのアップグレードは、ECPでのスケールアップを必要としません。現行の2ソケットx86プロダクションサーバーは、数十のコアと巨大なメモリを持つことができます。最近のCachéバージョンでは、TrakCare (および他の多くのCachéアプリケーション) は、単一サーバーにおいて、CPUコア数とメモリの増設により、ユーザーとトランザクションの増加をサポートし、予測可能な線形スケールアップを備えています。現場では、ほとんどの新しいデプロイメントが仮想化されています。その場合でも、VMIは、必要に応じてホストサーバーのサイズまで拡張できます。単一の物理ホストが提供できる以上のリソース要件である場合、ECPを使用してスケールアップします。

- ヒント： 管理とデプロイメントを簡略化するため、ECPをデプロイする前に単一サーバー内でスケールアップを実行します。

この投稿では、アーキテクチャの例とECPの基本的な仕組みを示し、ストレージに重点を置いてパフォーマンスの考慮事項を説明します。

ECPとアプリケーション開発の構成に

関する特定の情報は、オンラインの「[Caché分散データ管理ガイド](#)」で得ることができ、このコミュニティには[ECP学習トラック](#)があります。

ECPの他の主要な機能

の1つは、アプリケーションの可用性の向上

です。詳細については、「[Caché高可用性ガイド](#)」のECPセクションを参照してください。

[このシリーズの他の記事のリストはこちら](#)

ECPアーキテクチャの基本

ECPのアーキテクチャと働きは、概念としては単純です。ECPは、複数のサーバーシステム間でデータ、ロック、実行可能コードを効率的に共有する方法を提供します。アプリケーションサーバーから見たデータとコードは、リモートにあるデータサーバーに保存されますが、アプリケーションサーバーのローカルメモリにキャッシュされ、最小限のネットワークトラフィックでアクティブなデータへの効率的なアクセスを提供します。

データサーバーはディスク上の永続ストレージへのデータベースの読み取りと書き込みを管理しますが、複数のア

アプリケーションサーバーはアプリケーション処理のほとんどを実行するソリューションの主力です。

多層アーキテクチャ

ECPは多層アーキテクチャです。処理層とそれらの役割を説明するにはさまざまな方法があります。以下は、WebブラウザベースのCachéアプリケーションを説明するときに役立つものであり、私の投稿で使用するモデルと用語です。各層を分解するためのさまざまな方法があるかもしれませんが、ここでは私の方法を使ってみましょう。:)

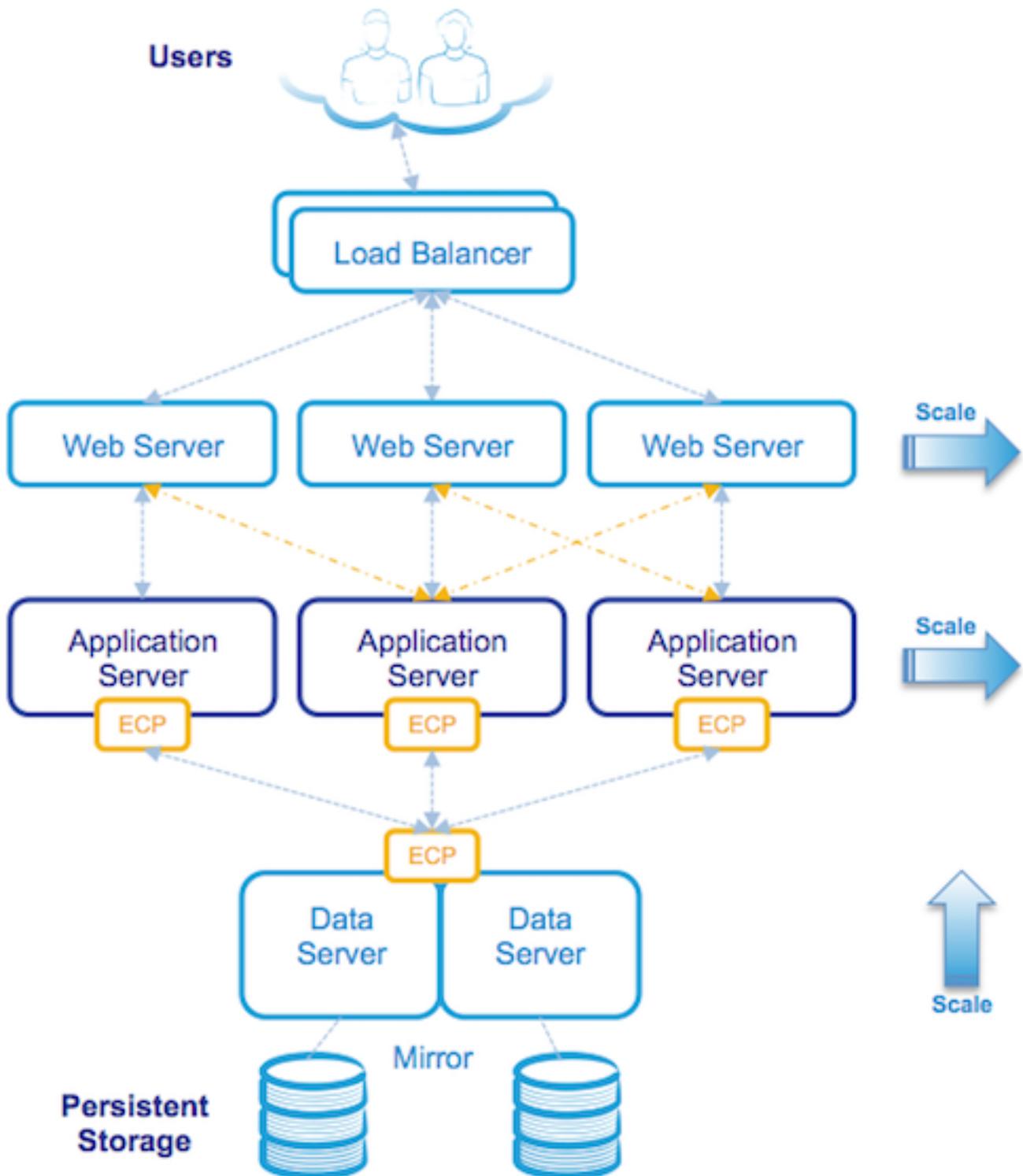
たとえば、Caché Server Pages (CSP) を使用するブラウザベースのアプリケーションは、プレゼンテーション、アプリケーション処理、およびデータ管理機能が論理的に分離された多層アーキテクチャを使用します。さまざまな役割を持つ「論理サーバー」が層を構成します。論理サーバーを個別の物理ホストまたは仮想サーバーに配置する必要はありません。コスト効率と管理性のために、一部またはすべての論理サーバーを単一のホストまたはオペレーティングシステムインスタンスに配置することもできます。デプロイメントがスケールアップすると、ECPを使用してサーバーを複数の物理ホストまたは仮想ホストに分割できるため、アプリケーションに変更を加えることなく、必要に応じて処理ワークロードを分散できます。

ホストシステムは、容量と可用性の要件に応じて、物理的なもの、または仮想化されたものである場合があります。以下の層と論理サーバーがデプロイメントを構成します。

- **プレゼンテーション層**： ブラウザベースのクライアントとアプリケーション層の間のゲートウェイとして機能するWebサーバーが含まれます。
- **アプリケーション層**： これは、ECPアプリケーションサーバーが置かれる層です。上記のように、これは、アプリケーションサーバーがデータサーバーから分離する必要がない論理モデルであり、通常、非常に大きなサイト以外で必要なものではありません。この層には、レポートサーバーなど、特殊な処理のための他のサーバーも含まれます。
- **データ層**： これは、データサーバーが配置されている層です。データサーバーはトランザクション処理を実行し、Cachéデータベースに格納されているアプリケーションコードとデータのリポジトリです。データサーバーは、永続ディスクストレージの読み取りと書き込みを行います。

論理アーキテクチャ

次の図は、3層アーキテクチャーとしてデプロイされたブラウザベースのアプリケーションの論理図です。



アーキテクチャは一見複雑に見えるかもしれませんが、単一のサーバーにインストールされたCacheシステムと同じコンポーネントで構成されています。ただし、論理コンポーネントは、複数の物理サーバーまたは仮想サーバーにインストールされています。サーバー間のすべての通信はTCP/IPを介して行われます。

論理ビューでのECPの動き

上の図は、上から順に、複数の負荷分散されたWebサーバーに安全に接続しているユーザーを示しています。Webサーバーは、クライアントと任意の処理を実行するアプリケーション層（アプリケーションサーバー）の間でCSP Webページ要求を渡し、コンテンツを動的に作成し、完成したページをWebサーバー経由でクライアントに返します。

この3層モデルでは、アプリケーション処理はECPによって複数のアプリケーションサーバーに分散されています。アプリケーションは、データ（アプリケーションデータベース）をアプリケーションサーバーに対してローカルであるかのようにシンプルに扱います。

アプリケーションサーバーがデータを要求すると、ローカルキャッシュからの要求を満たそうとしますが、それができない場合は、ECPが自身のキャッシュからの要求を満たすことができるデータサーバーに必要なデータを要求します。また、それができない場合は、ディスクからデータをフェッチします。データサーバーからアプリケーションサーバーへの応答には、そのデータが格納されたデータベースブロックが含まれます。これらのブロックが使用され、次にアプリケーションサーバーにキャッシュされます。ECPは、ネットワーク全体のキャッシュの一貫性を自動的に管理し、変更をデータサーバーに反映します。クライアントはローカルにキャッシュされたデータを頻繁に使用するため、迅速な応答を得ることができます。

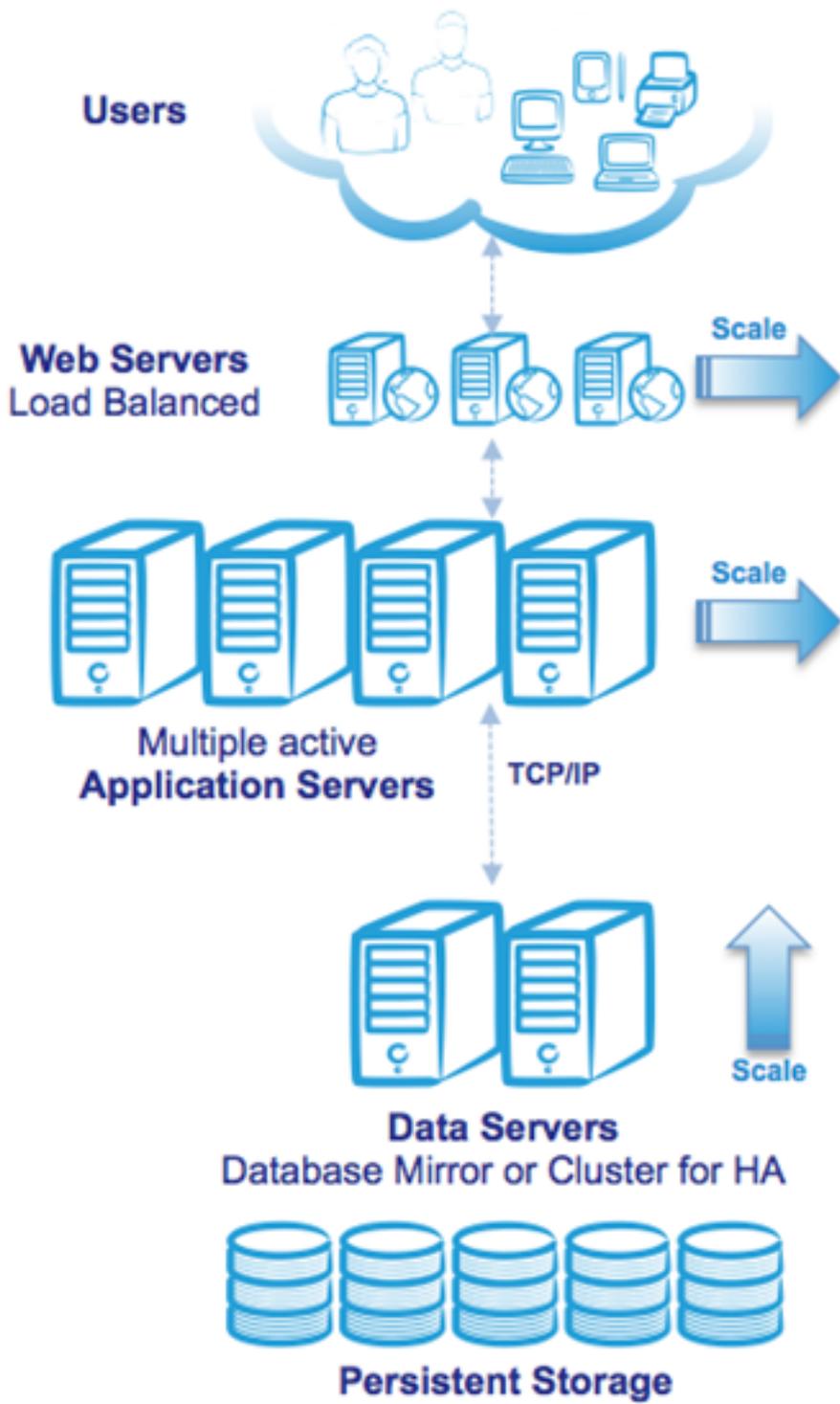
データが既にローカルキャッシュにある可能性があるため、デフォルトでは、Webサーバーは優先アプリケーションサーバーと通信し、同じアプリケーションサーバーが関連データに対する後続の要求にサービスを提供することを保証します。

- ヒント： [Cachéのドキュメント](#)
で詳述されているように、アプリケーションサーバーでのキャッシュの利点に影響を与えるラウンドロビンや負荷分散スキームでユーザーをアプリケーションサーバーに接続することは避けてください。同じユーザーまたはユーザーグループが同じアプリケーションサーバーに接続したままになることが理想的です。

このソリューションは、ユーザーのダウンタイムなしで、Webサーバーを追加することによってプレゼンテーション層を、また、アプリケーションサーバーを追加することによってアプリケーション層を拡張できます。データ層は、データサーバーのCPUとメモリを増やすことによって拡張されます。

物理アーキテクチャ

次の図は、3層論理アーキテクチャの例と同じ3層デプロイメントで使用される物理ホストの例を示しています。



物理ホストまたは仮想化ホストは、ホスト障害が発生した場合や定期メンテナンスに備えて、100%容量のn + 1またはn + 2モデルを使用して各層にデプロイされることに注意してください。ユーザーは複数のWebサーバーとアプリケーションサーバーに分散しているため、単一のサーバーの障害は少数のユーザーに影響を与え、ユーザーは残りのサーバーの1つに自動的に再接続します。

データ管理層は、たとえば1つ以上のストレージレイに接続されたフェイルオーバークラスター（たとえば、仮想化HA、InterSystemsデータベースミラーリング、または従来のフェイルオーバークラスタリング）によって高い可用性を持ちます。ハードウェアまたはサービスに障害が発生した場合、クラスタリングは、クラスター内の残りのノードの1つでサービスを再起動します。その他の利点として、ECPには復旧力が組み込まれており、データベースノードクラスターのフェイルオーバーが発生した場合でもトランザクションの整合性が維持されるので、アプリケーションユーザーは、フェイルオーバーと自動復旧が完了するまで処理の一時停止を観察しますが、ユーザーは切断されずにシームレスにトランザクションを再開することができます。

同じアーキテクチャを仮想化サーバーにマッピングすることもできます。たとえば、VMware vSphereを使用してアプリケーションサーバーを仮想化できます。

ECPキャパシティプランニング

上述のように、データサーバーはディスク上の永続ストレージへのデータベースの読み取りと書き込みを管理しますが、複数のアプリケーションサーバーはアプリケーション処理のほとんどを実行するソリューションの主力となるものです。システムリソースのキャパシティプランニングを実施する際の重要な概念の要約は次のとおりです。

- データサーバー（データベースサーバーと呼ばれることもあります）は通常、アプリケーション処理をほとんど実行しないため CPU要件は低くなります。ところが、このサーバーはストレージIOの大部分を実行するため、非常に高いストレージIOPSとなる可能性があります。これらは、データベースの読み取りと書き込み、およびジャーナルの書き込み（ジャーナルIOについては後で詳しく説明します）です。
- アプリケーションサーバーはほとんどのアプリケーション処理を実行するため高いCPU要件がありますが、ストレージIOはほとんどありません。

一般的に、ECPサーバーのCPU、メモリ、およびIOの要件は、高可用性に合わせてN + 1またはN + 2サーバーを考慮しながら、非常に大規模な単一サーバーソリューションのサイズを決定する場合と同じルールを使用して決定します。

基本的なCPUおよびストレージのサイジング：

MyApplicationがアプリケーション処理にピーク72 CPUコアを必要とし（ヘッドルームも考慮に入れてください）、書き込みデーモンサイクル中に2万回の書き込みが必要であり、持続的ピーク1万回のランダムデータベース読み取りが必要であると想像してください。

仮想サーバーまたは物理サーバーの、大まかなサイジングは次のとおりです。

- 4 x 32 CPUアプリケーションサーバー（3サーバー+ HA用に1）。低いIOPS要件。
- 2 x 10 CPUデータサーバー（HA用にミラーリングまたはクラスター化）。[低レイテンシIOPS要件](#)は、2万回の書き込み、1万回の読み取り、およびWIJとジャーナルです。

データサーバーはほとんど処理を実行していませんが、システムプロセスとCacheプロセスを考慮して、8~10 CPUのサイズに設定されています。アプリケーションサーバーのサイズは、物理ホストごとのベストの価格性能比に基づいて、および/または可用性を考慮して設定することができます。スケールアウトすると効率が多少低下しますが、通常はサーバーブロックに処理を追加して、スループットの線形に近い増加を期待できます。制限は、ストレージIOで最初に生じる可能性が高くなります。

- ヒント：HAで通常するように、ホスト、シャーシ、またはラックの障害の影響を考慮します。VMWareでアプリケーションサーバーとデータサーバーを仮想化する場合、必ずvSphere DRSとアフィニティルールを適用して処理負荷を分散し、可用性を確保します。

ジャーナル同期IO要件

ECPデプロイメントのキャパシティプランニングに関するその他の考慮事項は、より高いIOが必要であり、ジャーナルの同期（別名 ジャーナル同期）のためより高いIOが必要であり、データサーバー上のジャーナリングのスケラビリティを維持するため、非常に厳格なストレージ応答時間要件を課することです。同期要求の発行により、ジャーナルの最後のブロックへの書き込みがトリガーされ、データの耐久性が確保されます。

アプリケーション次第ですが、高いトランザクションレートの一般的な顧客サイトでは、ECP以外の構成でのジ

ジャーナル書き込みIOPSが毎秒2桁台で見られることがよくあります。 ビジー状態のシステムでECPを使用すると、ECPによってジャーナル同期が強制されるため、ジャーナルディスク上で100~1,000の書き込みIOPSを確認できます。

- ヒント：mgstatを表示するか、[pButtons](#)でmgstatを確認すると、ストレージIOリソース計画で考慮されるJrnwrts（ジャーナル書き込み）が表示されます。 ECPデータサーバーには、mgstatに表示されないジャーナルディスクへのジャーナル同期書き込みもあります。 これらを確認するには、iostatなどを使用して、ジャーナルディスクのオペレーティングシステムメトリックを確認する必要があります。

ジャーナル同期とは何ですか？

ジャーナルの同期は次の場合に必要です。

- データサーバーで障害が発生した場合のデータの耐久性と回復性を保証します。
- また、アプリケーションサーバー間のキャッシュの一貫性を確保するためのトリガーでもあります。

非ECP構成では、Cachéデータベースへの変更はジャーナルバッファ（128 x 64Kバッファ）に書き込まれ、ジャーナルデーモンによって、バッファが一杯になったら、あるいは2秒ごとに、ディスク上のジャーナルファイルに書き込まれます。 Cachéはバッファ全体に64kを割り当て、これらは破棄されたり再作成される事なく常に再利用され、またCachéは単に終了オフセットを追跡します。 ほとんどの場合（大量の更新が一度に行われたい限り）、ジャーナルの書き込みは非常に小さなものです。

ECPシステムでは、ジャーナルの同期も行われます。 ジャーナル同期は、現在のジャーナルバッファの関連部分をディスクに再書き込みして、ジャーナルが常にディスク上で常に最新になるようにすることとして定義できます。 そのため、ジャーナル同期リクエストにより、同じジャーナルブロックの一部（サイズが2kから64kの間）が、複数回、再書き込みされる事があります。

ジャーナル同期要求をトリガーできるECPクライアント上のイベントは、更新（SETまたはKILL）またはLOCKです。 たとえば、SETまたはKILLごとに、現在のジャーナルバッファがディスクに書き込まれます（または書き換えられます）。 非常にビジーなシステムでは、ジャーナル同期を1回の同期操作で複数の同期リクエストにバンドルまたは遅延できます。

ジャーナル同期のキャパシティプランニング

スループット維持のために、ジャーナル同期の平均書き込み応答時間は次のようにする必要があります。

- ≤ 0.5 ms、最大 ≤ 1 ms

詳細については、この投稿のIO要件の表を参照してください。 [パート6 - CachéストレージIOプロファイル。](#)

- ヒント：CachéデータベースミラーリングをECPで使用する場合、ジャーナル同期は、プライマリミラーノードとバックアップミラーノード・ジャーナルディスクの両方に適用されます。 ミラー構成のルールは、両方のノードのストレージIOを同等に構成することなので、これは問題ではないでしょう。

お使いのシステムの特定のIOメトリックを検証する必要があります。 このセクションの目的は、非常に厳密な応答時間の要件があるということを理解し、メトリックを見つける場所を把握することです。

- ヒント：[Cachéシステムでは、Red Hat LinuxとSuSE XFSファイルシステムが推奨されます](#)。 ただし、テストでは、ext4はジャーナル同期などの小さな書き込みに対してより高いIOPS機能を提供することが明らかになっています。 ジャーナルディスクにはext4の使用を検討してください。

概要

この投稿は[ECP](#)

と、キャパシティプランニング中に考慮するその他のメトリックに関するオリエンテーションです。近い将来、非常に大規模なシステムで実施されたCachéとECPに関する最新のベンチマークの結果を皆さんと共有できることを願っています。いつものように、ご質問やご意見がある場合はどうぞお気軽にご連絡ください。 ツイッター @murrayoldfield

[#ECP](#) [#システム管理](#) [#Caché](#) [#InterSystems IRIS](#) [#InterSystems IRIS for Health](#)

ソースURL:

<https://jp.community.intersystems.com/post/%E3%83%87%E3%83%BC%E3%82%BF%E3%83%97%E3%83%A9%E3%83%83%E3%83%88%E3%83%95%E3%82%A9%E3%83%BC%E3%83%A0%E3%81%A8%E3%83%91%E3%83%95%E3%82%A9%E3%83%BC%E3%83%9E%E3%83%B3%E3%82%B9-%E3%83%91%E3%83%BC%E3%83%887-%E3%83%91%E3%83%95%E3%82%A9%E3%83%BC%E3%83%9E%E3%83%B3%E3%82%B9%E3%80%81%E3%82%B9%E3%82%B1%E3%83%BC%E3%83%A9%E3%83%93%E3%83%AA%E3%83%86%E3%82%A3%E3%80%81%E5%8F%AF%E7%94%A8%E6%80%A7%E3%81%AE%E3%81%9F%E3%82%81%E3%81%AEcp>