ORACLE

高速アプリケーション通知（FAN）

FANwatcherを含む
Oracle Notification System（ONS）をサブスクライブしFANイベントを表示するためのユーティリティ

2025年5月、バージョン [1.0]

公開

本書の目的

本書では、リリース23aiの機能と強化された点の概要が説明されています。本書は、23aiへのアップグレードに関するビジネス上の利点の評価と、説明した製品機能の実装およびアップグレードの計画を支援することのみを目的としています。

免責事項

本文書には、ソフトウェアや印刷物など、いかなる形式のものも含め、オラクルの独占的な所有物である占有情報が含まれます。この機密文書へのアクセスと使用は、締結および遵守に同意したOracle Software License and Service Agreementの諸条件に従うものとします。本文書と本文書に含まれる情報は、オラクルの事前の書面による同意なしに、公開、複製、再作成、またはオラクルの外部に配布することはできません。本文書は、ライセンス契約の一部ではありません。また、オラクル、オラクルの子会社または関連会社との契約に組み込むことはできません。

本書は情報提供のみを目的としており、記載した製品機能の実装およびアップグレードの計画を支援することのみを意図しています。マテリアルやコード、機能の提供をコミットメント（確約）するものではなく、購買を決定する際の判断材料になさらないでください。本文書に記載されている機能の開発、リリース、時期および価格については、弊社の裁量により決定されます。製品アーキテクチャの性質上、本書に記述されているすべての機能を安全に組み込むことができず、コードの不安定化という深刻なリスクを伴う場合があります。

ORACLE

目次

ORACLE

図一覧

表一覧

図表一覧

本書の目的と対象読者

この技術概要では、Oracle Real Application Clusters（Oracle RAC）とOracle Data Guardを使用した環境でのOracle 23*ai*高速アプリケーション通知（FAN）[1]について説明します。FANは、計画メンテナンス、計画外停止、およびロードの不均衡によってデータベース・インスタンスが使用不可または無反応になった場合に、エンドユーザーが遭遇する可能性のある不十分なエクスペリエンスを解決するための重要なコンポーネントです。FANにより、アプリケーションのエンド・ツー・エンドの完全自動リカバリと、実際のトランザクション・パフォーマンスに基づいた実行時ロードバランシングが可能になります。

FANは、データベースの状態をアプリケーションに公開します。データベース・サービスの状態（up、down、unresponsiveなど）が変化すると、関係するサブスクライバにFANイベントを通じて新しいステータスがポストされます。FANは、データベース・サービス、インスタンス、データベース、およびクラスタ・ノードの状態の変化について迅速に通知を行います。

OracleドライバおよびOracleプールでは、FANイベントの使用により以下が実現します。

- アプリケーションにエラーが返されることなしに、計画メンテナンス時に作業をドレイニング
- 極めて迅速に障害を検出し、リアルタイムでアプリケーションをリカバリ
- パフォーマンスの不均衡が発生し、インスタンスがシステムを離脱したりシステムに参加したりして、リソースが使用可能になった後の実行時における受信作業のロード・バランシング
- 後続のWebセッションなどの関連する対話が一緒にルーティングされ、最適なパフォーマンスが得られるようにするための、受信作業に対するアフィニティのアドバイス

本書では、以下を提供します。

1. データベース・システムでFANイベントを有効化することの利点の概要
2. FANイベントおよび公開されたフィールドの分析
3. FANwatcherの使用によるFANイベントの表示方法の説明
4. FANイベントを統合して有効化する手順

対象読者は、アプリケーションやアプリケーション・サーバー、監視コンソール、または社内ビジネス・ワークフロー・システムと統合された、計画メンテナンスおよび計画外停止の迅速な通知を必要としている、Oracle RACデータベース管理者、Data Guardデータベース管理者、アプリケーション・インテグレーターなどです。読者は、以下の参考資料に示されている概念に精通していることを前提としています。

*Oracle Clusterware*管理およびデプロイメント・ガイド、23ai Part number F46761-03
*Oracle Real Application Clusters*管理およびデプロイメント・ガイド、*23ai* 、Part Number F46762-03

[1] 本書は、19c以降のすべてのリリースに関連します。一部の情報は、Oracle Database 23aiに固有のものである場合があります（コマンド構文など）

。

FANイベントを使用する利点

なぜFANの使用が極めて重要なのか

計画停止時および計画外停止時の状態の変化は、アプリケーションの応答時間に影響を与えます。

- ソケットを閉じずにノードに障害が生じた場合、TCP/IPタイムアウトを数分間待機し、そのIPアドレスのダウン中に後続のすべての接続を待機する。
- サービスが停止した場合に接続を試みる。
- サービスの再開時に接続しない。
- サーバーが停止したときに、クライアントの最後の結果を処理する。
- 低速、ハング、停止状態のノードで作業を実行しようとする。

さらに悪いことに、これらのパフォーマンス低下原因は、標準のデータベース・パフォーマンス手法では取得されず、レポートされません。ソケットを閉じずにノードに障害が生じた場合、IO（読取りまたは書込み）でブロックされたすべてのセッションはTCP keepaliveまで待機します。この待機ステータスは、データベースを使用するアプリケーションの一般的な状態です。最後の結果を処理するセッションは、次のデータがリクエストされるまで中断を受信しないため、状況はさらに悪化します。FANイベントを使用すると、TCPタイムアウトを待機するアプリケーション、障害発生後のクライアントにおける最後の結果の時間がかかる処理、そして低速、ハング、または停止状態のノードでの時間がかかる作業の実行がなくなります。

通常、データベースに接続されたクライアントまたは中間層アプリケーションは、接続タイムアウト、帯域外ポーリング・メカニズム、またはその他のカスタム・ソリューションを使用して、システム・コンポーネントに障害が生じたことを認識し、その障害の影響を緩和するためのアクションを開始していました。この方法では、以下に重大な影響を及ぼしました。（1）アプリケーション可用性。増加した顕著な停止時間（ポーリング間隔とタイムアウトの長さのため）が原因でした。（2）エンタープライズ環境の管理。水平方向（すべてのノード間）および垂直方向（リスナー、インスタンス、アプリケーション・サービスなどのノード・コンポーネント間）でのサーバー側コンポーネント数の急激な増加により、ポーリング・アクションの絶対数が大幅に増加したことが原因でした。

Oracle Database 10gでリリースされたFANでは、データベース・イベントをリスナー、アプリケーション、アプリケーション・サーバー、およびIT管理コンソール（Oracle Enterprise Manager、トラブル・チケット・ロガー、電子メール/テキスト・メッセージ・サーバーなど）と統合することにより、これらの問題を解決しました。システム変更が検出されるとFANイベントがすぐにポストされるため、インシデント発生時に直ちに対処され、不要なステータス・ポーリングをアプリケーションが実装する必要がないため、リソース使用率全般が改善されます。

Oracle Database 23aiに付属のFAN

Oracle Database 12c以降、FANでは極めて重要な3つの機能強化が図られています。

- FANは、Oracle RACに付属し、設定不要ですぐに使用できるデフォルトの機能で、構成済みで有効化されている。
- すべてのOracleクライアントでは、Oracle Notification System（ONS）をFANの転送手段として使用する。
- FANは、FANイベントで複数のデータセンターを対象とすることができるよう、Oracle Global Data Services（Oracle GDS）によってポストされる。

FANはデフォルトで有効

デフォルトでは、FANはOracle Grid Infrastructureのインストール時に構成されます。srvctlを使用して、FANで使用されるポートを変更できます。Oracleクライアントが、FANイベントを受信するために構成を変更する必要はありません。これは、クライアントが起動すると、データベースに対してONSエンドポイントの問合せを自動的に実行することを意味します。自動構成は、複数のデータセンターを対象とします。クライアントは、接続URLの各データベースからONS構成を自動的に受信します。クライアントはONSを構成する必要はなく、FANを有効化するだけです。

FANは転送手段としてONSを標準に

Oracle Database 12c以降、すべてのOracle 12c FANクライアントはOracle Notification Systemを使用してFANイベントを受信しています。FANの転送手段としてONSを標準とする理由は、“障害が生じたコンポーネントからは障害が生じたという通知を受けられないため”です。

ONSは、データベース・スタック外部のGrid Infrastructureで実行され、データベースの依存性を伴います。データベースが停止したり故障したりした場合、FANはステータス変更イベントを直ちにポストし、ONSはそれらを直ちに配信します。

以前のリリースのクライアントや古いデータベース・バージョンを使用する新規のクライアントをサポートするために、イベントはONSとAQの両方の転送手段を通じて自動的に公開されます。

Oracle Global Data Services（Oracle GDS）

Oracle Database 12c以降、複数のデータセンター、特に、ランタイム・ロードバランシングを必要とするOracle Active Data Guardファームを対象とするために、Oracle Global Data Services（Oracle GDS）がFANをポストします。Oracle GDSでは、サービス配置属性を考慮して、データセンター変更を伴う計画メンテナンス時に別の使用可能なデータベースへのデータベース間サービス・フェイルオーバーを自動的に実行し、計画外停止が発生した場合にデータベース全体の障害を通知します。Oracleクライアントおよび接続プールは、障害イベントが発生したときに中断され、グローバル・サービスが新たに起動したときに通知されます。

FANの使用方法

FANを使用する際にコードを変更する必要はありません。FANを使用するための最適でもっとも簡単な方法は、Oracle接続プール、Oracleクライアント・ドライバ、またはFAN向けに構成されたOracleアプリケーションを使用することです。FANイベントは、次と統合されます。

- Oracle Fusion MiddlewareおよびOracle WebLogic Server（Oracle WebLogic Serverのための継続的可用性、Part Number E90842-03）
- Oracle Data Guard Broker（*Oracle® Data Guard Broker 23ai* Part Number F46805-02）
- Oracle Enterprise Manager Cloud Control（基本インストレーション・ガイド、24ai Release 1（24.1）、Part Number F97194-03）
- Oracle JDBC Universal Connection Pool（*Oracle® Universal Connection Pool*開発者ガイド、*23ai*、Part Number F47026-10）
- ODP.NET（Oracle® Database開発ガイド、23ai、Part Number F47572-16）
- SQLPLUS（SQL*Plus®ユーザーズ・ガイドおよびリファレンス、23ai、Part Number F47057-07）
- Global Data Services（Oracle® Database Global Data Services概要および管理ガイド、23ai、Part Number F46808-03）

IBM WebSphereやApache Tomcatなどのサード・パーティ・クライアントを使用する場合、デフォルトの接続プールをOracle Universal Connection Poolに置き換えることで、FANはIBMとApacheにそれぞれサポートされます。オラクルのUniversal Connection Pool（UCP）は、FANとともに使用することで計画メンテナンスがユーザーに見えないようにすることができ、Oracle Database 23aiおよびアプリケーション・コンティニュイティとともに使用することで計画外停止がユーザーに見えないようにすることができます。UCPは、Oracle RAC、Oracle Data Guard（Oracle DG）、またはOracle Active Data Guard（Oracle ADG）を使用するJavaベースのアプリケーション向けの実証済みで推奨されるクライアント・ソリューションです。

高速フェイルオーバーでFANを使用する方法

図1：インスタンス障害時のFANアクティビティを示すタイムライン

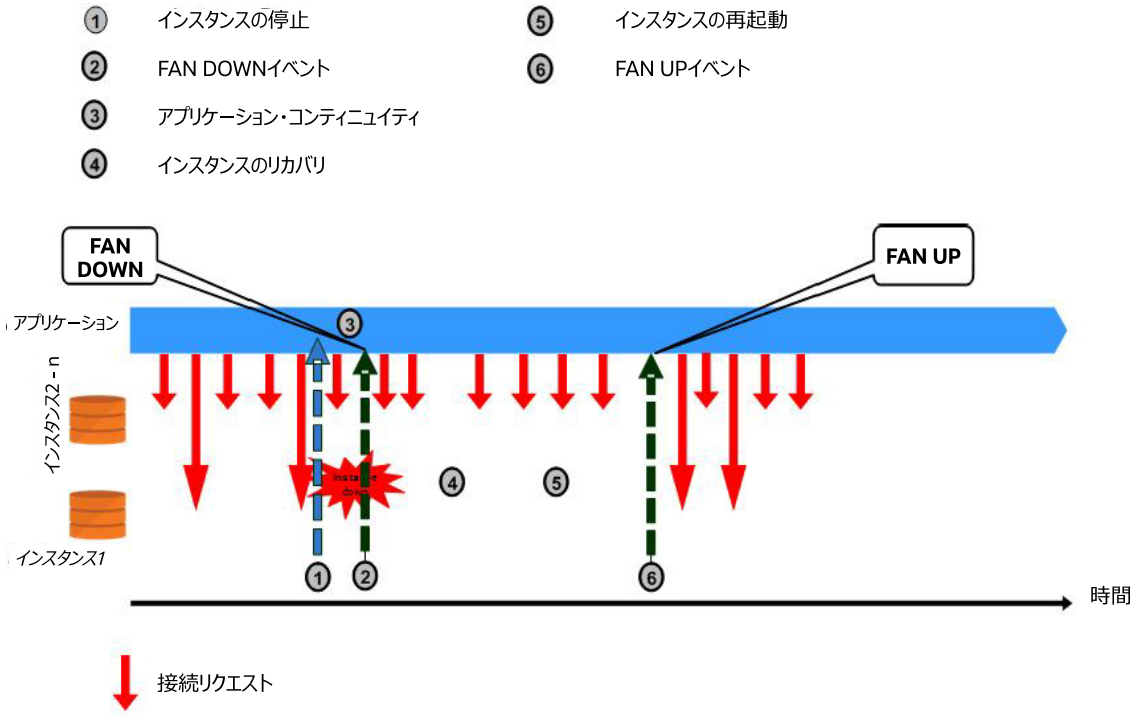

図1は、インスタンス障害のタイムラインを示しています。アプリケーションはFANに対応しており（高速接続フェイルオーバー対応のUCP）、Oracle RACデータベースのすべてのインスタンスで実行されている動的データベース・サービスを使用して接続するか、またはインスタンスのサブセット上でサービスが実行される構成で接続します。アプリケーションには、すべてのインスタンスにアクティブなセッションがあります。以下が、計画外停止時に発生します。この例では、インスタンス障害を使用しています。

- ステップ1で、インスタンスがクラッシュします。
- ステップ2で、FANの計画外DOWNイベントにより、アイドル状態のセッションが接続プールから直ちに消去されます。

その他のインスタンス上にある既存の接続は使用できる状態で維持され、必要に応じて、それらのインスタンスに対する新しい接続がオープンされます。

- アプリケーション・コンティニュイティを使用するために構成されたプールでは、アクティブなセッションが、存続しているインスタンスにリストアされ、アプリケーション・コンティニュイティによってリカバリされて、停止がユーザーやアプリケーションに対してマスクされます。アプリケーション・コンティニュイティによって保護されていない場合、インスタンスとのアクティブな通信があるセッションはエラーを受信します（ステップ3）。アプリケーションでは、それがリカバリ可能なエラーかどうかをテストすることができ、この時点で再接続の試行を開始することもできます。例に示すように、サービスが別のインスタンスで使用可能な場合は、新しい接続を取得することもできます。
- これはOracle RACデータベースであるため、存続しているインスタンスによって、障害が生じたインスタンスのリカバリが実行され（ステップ6）、続いてGrid Infrastructureによってインスタンスが再起動されます（ステップ7）。
- サービスのFAN UPイベントにより、新しいインスタンスが使用可能になったことが接続プールに通知され、次回のリクエスト送信時にセッションをこのインスタンスに作成できます。

透過的計画メンテナンスでFANを使用する方法

図2：計画メンテナンス時のFANアクティビティを示すタイムライン

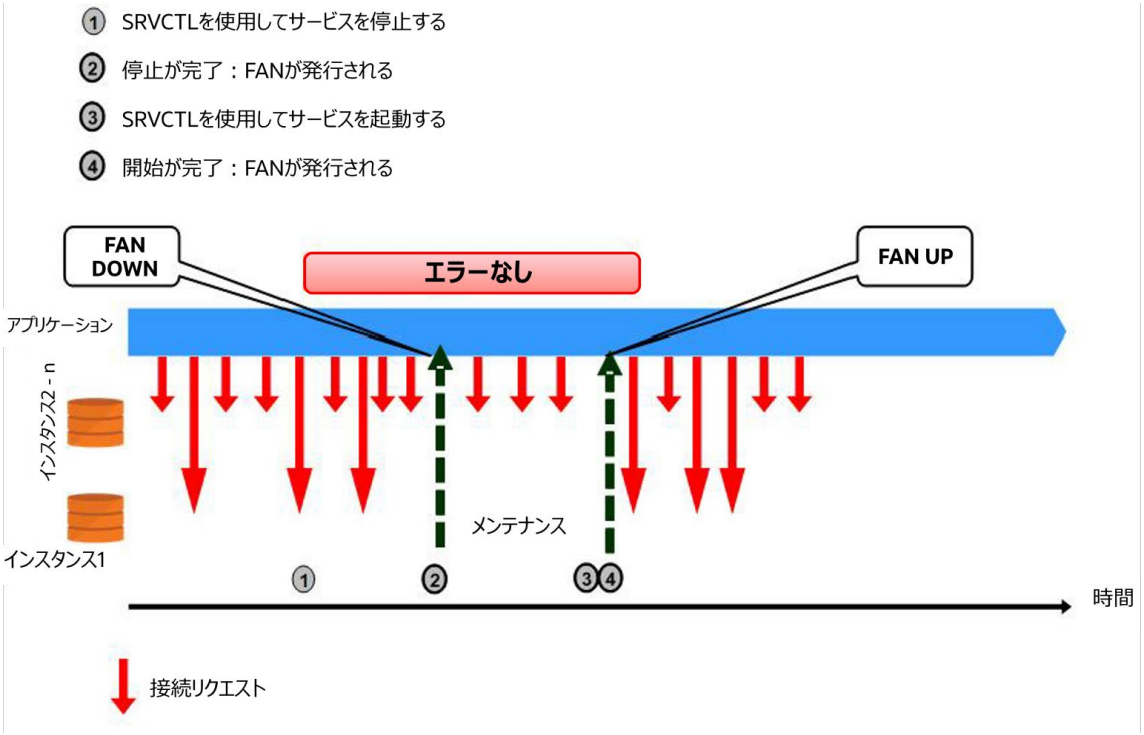

図2は、データベースのパッチ適用などの計画メンテナンスのタイムラインを示しています。アプリケーションはFANに対応しており（高速接続フェイルオーバー対応のUCP）、Oracle RACデータベースのすべてのインスタンスで実行されている動的データベース・サービスを使用して接続するか、またはインスタンスのサブセット上でサービスが実行される構成で接続します。アプリケーションには、すべてのインスタンスにアクティブなセッションがあります。パッチはローリング方式で適用されるため、アプリケーションはこの計画停止時には中断されません。

- ステップ1で、srvctlを使用して、インスタンスでUNIFORMサービスとして構成されたサービスを停止します。サービスがUNIFORMでない場合は、別のインスタンスに再配置します。これらのコマンドでは-forceフラグを使用しないでください。接続プールは、通常は接続がプールにチェックされる際に、接続をリクエスト境界で自動的に解放します。
- ステップ2で配信されるFANの計画DOWNイベントにより、アイドル状態のセッションが接続プールから直ちにクリアされ、解放されてプールに返されるアクティブなセッションがマークされます。これらのFANアクションは、アプリケーションやユーザーを妨げることなく、セッションをインスタンスからドレインします。

その他のインスタンス上にある既存の接続は使用できる状態で維持され、必要に応じて、それらのインスタンスに対する新しい接続がオープンされます。

- すべてのセッションが、あらゆる場合において、プールへの接続をチェックするわけではありません。ベスト・プラクティスは、インスタンスが強制的にシャットダウンされた後のタイムアウト期間を設け、残りのクライアント接続をすべて排除することです。アプリケーション・コンティニュイティを使用するために構成されたプールでは、アプリケーション・コンティニュイティによって残りのセッションがリカバリされ、停止がユーザーやアプリケーションに対してマスクされます。
- アップグレード、パッチ適用、または修復が完了したら、srvctlを使用して元のノード上のインスタンスとサービスを再起動します（ステップ3）。
- サービスのFAN UPイベントにより、新しいインスタンスが使用可能になったことが接続プールに通知され、次回のリクエスト送信時にセッションをこのインスタンスに作成できます。

FANイベントの内容

Oracle FANイベントは、イベントの名前、タイプ、性質と、イベントが発生した時刻を記述する名前/値ペアのセットで提供される、ヘッダーとペイロード情報で構成されます。このペイロードに基づいて、イベントの受信者（通常はOracleクライアント）は、Oracle接続プールの古い接続参照のリフレッシュ、処理中の作業の自動リカバリ（そのクライアントがアプリケーション・コンティニュイティを使用するように構成されている場合）、最後にコミットされた結果の特定およびユーザーへの返却（そのクライアントがトランザクション・ガードを使用するように構成されている場合）、サービス・リクエストのロギングやデータベース管理者へのテキストの送信（そのクライアントがEnterprise Managerである場合）などのアクションを実行します。

FANクライアントは、接続署名と呼ばれる、接続に関する情報を使用して、イベントの受信時にどの接続を操作するかを認識します。（表1）

表1：FAN接続署名

FANパラメータ	一致するデータベース署名
サービス	sys_context('userenv', 'service_name')
データベースの一意の名前	sys_context('userenv','db_unique_name')
インスタンス	sys_context('userenv', 'instance_name')
ノード名	sys_context('userenv', 'server_host')

FAN高可用性イベント

ユーザー定義のサービスを使用するデータベース接続は、すべてのFANイベントを自動的に受信します。データベース・サービスは、接続リクエストとデータベース・インスタンスの間のハードワイヤード・マッピングをすべて切り離し、接続ロードバランシング、動的ワークロード管理、サービス・レベルおよび優先度、接続プールの再編成、アプリケーション・コンティニュイティなどの、付加価値のある機能を提供します。

Oracle FANにとって重要なイベント・タイプは以下のとおりです。

- 動的データベース・サービス向けのサービス・イベント
- クラスタ・メンバーシップの状態やノードのネイティブのjoin/leave操作を含むノード・イベント
- OCIベースのクライアントが操作をグループ化するために使用するインスタンス・イベント
- 接続プール・クライアントは、ランタイム・ロードバランシングおよびアフィニティのアドバイスを使用し、前のアフィニティおよびロード・アドバイスに基づいて、受信データベース・リクエストを実行に最適な場所へ指示

FANは、高可用性およびランタイム・ロードバランシングを実現するため、管理対象データベース・リソースに関連したイベントの生成、表示、および配信を標準化します。ノード・イベントおよびVIPイベントでは、Oracle Grid Infrastructureが必要です。Oracle RAC、Oracle RAC One Node、Oracle Restart、およびOracle Data GuardをGrid Infrastructureと一緒に、すべてのイベント・タイプと、複数のデータセンターにわたるOracle GDSに対して使用できます。

表2：FAN高可用性イベント・タイプ

イベント・タイプ	説明
event_type=NODE	Oracleクラスタ・ノードまたはネットワーク・イベント
event_type=INSTANCE	Oracle Databaseインスタンス・イベント
event_type=DATABASE	Oracle Databaseイベント
event_type=SERVICEMEMBER	特定のインスタンス・イベント上のアプリケーション・サービス
event_type=SERVICE	アプリケーション・サービス・イベント

表3：FAN高可用性イベント・ステータス

上記の各管理対象リソースのイベント・ステータスには、以下が含まれます。

イベント・ステータス	説明
status=up	リソースが開始した
status=down	リソースが停止した
status=nodedown	ノードまたはパブリック・ネットワークがオフラインになった
status=not_restarting	リソースに30分以内に3回超の障害が発生したため、再起動されていない

表4：FAN高可用性イベントの理由

各管理対象リソースのイベント・ステータスは、1つのイベント理由に関連しています。以下では、イベントが開始された理由が詳しく説明されています。

イベントの理由	アクティビティ・タイプ	イベントの開始理由
reason=USER	計画	srvctl、sqlplus、またはgdsctlによってユーザーが開始したコマンド
reason=FAILURE	計画外	そのリソースに障害が検出された
reason=member_leave	計画外または計画	ノードがオフラインになった
reason=public_nw_down	計画外または計画	パブリック・ネットワークがオフラインになった
reason=BOOT	計画外	ノードの開始時に、DATABASE、INSTANCE、またはSERVICEMEMBERイベントは、ノードがクラスタを離れる前にオンラインだった場合、reason=BOOTで開始される

表5：FAN高可用性イベント・ペイロード・フィールド

追加のイベント・ペイロード・フィールドでは、ステータスが監視され公開されている一意のリソースについて詳しく説明されています。これらの追加のフィールドには、以下が含まれます。

イベント・リソース識別子	説明
VERSION=<n.n>	イベント・ペイロード・バージョン（現在は1.0）
timestamp=<eventDate> <eventTime>	イベントが検出された時点のサーバー側の日付と時刻
service=<serviceName.dbDomainName>	データベース・サービスの完全修飾名
database=<dbName>	データベースの一意の名前
instance=<SID>	データベース・インスタンスの名前
host=<hostname>	クラスタ・ノードの名前（Grid Infrastructureによって返されたもの）
card=<n>	サービス・メンバーシップ・カーディナリティ
timezone=<+|- GMT>	GMTとの差
incarn	クラスタ・インカネーション数。順序付けを保証するため、NODE DOWNイベントでのみ表示
vip_ips	パブリック・ネットワーク障害のためにダウンしたVIPのIPアドレス。reason=public_nw_downのNODE DOWNイベントでのみ表示

FAN高可用性イベントの例

結果は、次のいずれかのペイロード構造を持つFANイベントになります。

ノード・イベント

- Down

```
VERSION=1.0 event_type=NODE host=rachost_724 incarn=316152110 status=nodedown
reason=member_leave timestamp=2025-01-21 19:13:24 timezone=-08:00
```

- Public Network Down

```
VERSION=1.0 event_type=NODE host=rachost_724 incarn=0 status=nodedown
reason=public_nw_down vip_ips=10.10.8.249 timestamp=2025-01-21 19:13:13
timezone=-08:00
```

注：reason=public_nw_downの場合、incarnは常に0になります。

インスタンス・イベント

- Instance Down

```
VERSION=1.0 event_type=INSTANCE service=testy.us.oracle.com instance=TESTY_2
database=testy db_domain=us.oracle.com host=rachost_723 status=down reason=FAILURE
timestamp=2025-01-21 19:32:43 timezone=-08:00
```

- Instance Up

```
VERSION=1.0 event_type=INSTANCE service=testy.us.oracle.com instance=TESTY_2
database=testy db_domain=us.oracle.com host=rachost_723 status=up reason=FAILURE
timestamp=2025-01-21 19:33:10 timezone=-08:00
```

サービス・イベント

- Servicemember Down

```
VERSION=1.0 event_type=SERVICEMEMBER service=testy_pdb_srv.us.oracle.com
instance=TESTY_2 database=testy db_domain=us.oracle.com host=rachost_723
status=down reason=FAILURE timestamp=2025-01-21 19:32:43 timezone=-08:00
```

- Servicemember Up

```
VERSION=1.0 event_type=SERVICEMEMBER service=testy_pdb_srv.us.oracle.com
instance=TESTY_2 database=testy db_domain=us.oracle.com host=rachost_723
status=up card=3 reason=BOOT timestamp=2025-01-21 19:33:15 timezone=-08:00
```

- Service Down

```
VERSION=1.0 event_type=SERVICE service=testy_pdb_srv.us.oracle.com
database=testy db_domain=us.oracle.com host=rachost_723 status=down reason=USER
timestamp=2025-01-21 19:25:52 timezone=-08:00
```

- Service Up

```
VERSION=1.0 event_type=SERVICE service=testy_pdb_srv.us.oracle.com
database=testy db_domain=us.oracle.com host=rachost_723 status=up reason=USER
timestamp=2025-01-22 17:57:13 timezone=-08:00
```

データベース・イベント

- Database Down

```
VERSION=1.0 event_type=DATABASE service=testy.us.oracle.com
database=testy db_domain=us.oracle.com host=rachost_725 status=down reason=USER
timestamp=2025- 03-24 20:09:17 timezone=-08:00
```

- Database Up

```
VERSION=1.0 event_type=DATABASE service=testy.us.oracle.com
database=testy db_domain=us.oracle.com host=rachost_725 status=up reason=USER
timestamp=2025- 03-24 21:09:27 timezone=-08:00
```

FANロードバランシング・アドバイザリ・イベント

ランタイム・ロードバランシング・イベントは、Oracle Database 10g Release 2以降で使用できるようになりました。これらのイベントは、サブスクライバに受信ロードの配置場所を指示します。サービス・レベルのランタイム・ロードバランシングを有効化することにより、現在の応答時間、スループット、容量に基づいて、FANはアプリケーション・サーバーが作業リクエストをそのリクエストにとって最適な場所に転送する支援を行います。ランタイム・ロードバランシングは、アフィニティのアドバイスと連携して同じ対話内にリクエストを一緒に維持し、低速、ハング、無反応のノードから作業を切り離します。アフィニティのアドバイスと組み合わせて、ランタイム・ロードバランシングは、サービス目標に基づいてシステム全体の予測可能な応答時間やスループットを提供します。

ランタイム・ロードバランシングは、WebLogic Server Active GridLink、Oracle JDBC Universal Connection Pool、ODP.NETの管理対象外および管理対象プロバイダ、PHP、およびOCI Session Pool向けに標準で提供されます。必要なのは、サービスおよびFANのランタイム・ロードバランシングを有効化し、ODP.NETの場合にクライアントのロードバランシングを有効化するだけです。ロードバランシング・アドバイザリ・イベントは、すべての23aiクライアントに対してはONS経由で、12cより前のOCIクライアントと、12cより前のデータベースを使用しているすべてのクライアントに対してはアドバンスト・キューイング（AQ）経由でポストされます。

表6：FANランタイム・ロードバランシング・アドバイザリ・イベント

イベント・リソース識別子	説明
バージョン	イベント・ペイロードのバージョン
イベント・タイプ	SERVICE_METRICS
サービス	DBA_SERVICESのサービスに一致
Database unique name	サービスをサポートしている一意のデータベース。db_unique_nameの初期化パラメータ値に一致（デフォルトは初期化パラメータDB_NAMEの値）
Timestamp	イベントを順序付けするための日付およびタイムスタンプ（ローカル・タイムゾーン）
イベントごとに繰返し	
インスタンス	サービスをサポートしているインスタンスの名前。ORACLE_SIDに一致
Percent	対象のデータベースおよびインスタンスに送信する作業リクエストのパーセンテージ
Service Quality	サービス目標に基づいた、サービス品質および帯域幅の加重移動平均（経過時間またはスループット）
Flag	サービス目標に関連したサービス品質の表示。値は、GOOD、VIOLATING、 NO DATA、UNKNOWN 注：flag=UNKNOWNは、アプリケーション作業がインスタンス上でアクティブでない場合に表示されます（記載されているサービスが対象） flag=NO DATAは、インスタンスがロード・データの連続更新を取得していない場合に表示されます（記載されているサービスが対象）

ロードバランシング・アドバイザリの例

```
VERSION=1.0 database=TESTY service=testy_pdb_srv.us.oracle.com {
{instance=TESTY_2 percent=34 flag=UNKNOWN aff=FALSE}{instance=TESTY_3
percent=33 flag=UNKNOWN aff=FALSE}{instance=TESTY_1 percent=33 flag=UNKNOWN
aff=FALSE} } timestamp=2025-01-22 18:41:41
```

FANイベントの表示

FANイベントは、23ai Grid Infrastructureを使用している場合には自動的にポストされます。Oracle Database 23aiクライアントを使用している場合、設定はクライアントでFANを有効化するだけです。

管理者は、FANイベントのフローを表示した方がよいでしょう。システムがポストするFAN高可用性イベントの記録を保持することも重要です。FANコールアウトにより、監査証跡や、イベントの内容や発生時期のタイミング情報を提供できます。

FANwatcher

FANwatcherユーティリティは、ローカルONS（Oracle Notification Service）デーモンに接続し、FANイベントをサブスクライブする、自己完結型のJavaアプリケーションです。このユーティリティにより、一部のFANイベントのヘッダー情報とイベント・ペイロードが表示されます。

FANwatcherユーティリティは、Oracleクライアント・インストール、Grid Infrastructureクラスタの任意のノード、またはONSデーモンがインストールがされている任意の中間層ノード（たとえば、WebLogic Server Active Gridlink for RACをサポートしているノード）で実行できます。

サブスクライブされているONSデーモンは、Grid Infrastructureの一部として実行されているONSデーモンと通信するように構成する必要があります。これは、Grid InfrastructureのONSデーモンによりOracleイベントが伝播するためです。詳しくは、本書の「*ONS*の構成」セクションを参照してください。Grid Infrastructureはローカルまたはリモートに配置され、OracleクライアントがJDBC、OCI、またはODP.Netの管理対象および管理対象外プロバイダを使用しているかどうかに関係なく、このユーティリティを使用できます。

FANwatcherのインストール

ブログ「Monitoring FAN Events」からZIPファイルをダウンロードします。

Linux/Unixマシンで、Oracleクライアント・インストールを含むマシン、またはGrid Infrastructureクラスタの任意のノードにZIPファイルを展開します。

ファイルFANwatcherを編集し、ORA_CRS_HOMEを設定する行を以下のように更新します。

```
$ mkdir FANwatch
$ cd FANwatch
$ unzip FANwatcher.zip
Archive:FANwatcher.zip
    inflating: ./classes/evtPrint.class
    inflating: ./src/evtPrint.java
    inflating:FANwatcher
    inflating: ./classes/fanWatcher.class
    inflating: ./src/fanWatcher.java
    inflating: ./classes/printEvtHeader.class
    inflating: ./src/printEvtHeader.java
    inflating:README.txt

$ vi FANwatcher
        ORA_CRS_HOME=/u01/app/oracle/product/12.1.0.2/GridInfrastructure
```

ORA_CRS_HOME変数が記述するディレクトリ・パスは、Oracleクライアント、Grid Infrastructure、またはOracle WebLogicサーバーをインストールする際にORACLE_HOMEとして指定されます。FANwatcherユーティリティは、ONSデーモンが開始するために使用するパス名$ORACLE_HOME/opmn/conf/ons.configを検索します。

FANwatcherユーティリティは、上記で指定したOracleホームにアクセスする権限があるユーザーが実行する必要があります。この権限は、Oracleクライアントをインストールしたユーザー、Grid Infrastructure（例：oracle）、またはこのユーザーが属するグループのメンバー（例：oinstall）と互いに関連しています。ユーザーに必要な権限がない場合、ユーティリティを実行しようとすると、以下のようなエラーが発生します。

```
$ ./FANwatcher
>> Executing fanWatcher ...
Opening FAN Subscriber Window ...

15/01/27-04:05:47 ONS: config file (ons.config) could not be found.
Exception in thread "main" oracle.ons.ONSException:Unable to open config file
        at oracle.ons.ONS.readStandaloneLocalPort(ONS.java:1907)
        at oracle.ons.ONS.localInit(ONS.java:279)
        at oracle.ons.ONS.<init>(ONS.java:196)
        at oracle.ons.ONS.getONS(ONS.java:1156)
        at oracle.ons.Subscriber.<init>(Subscriber.java:162)
        at onc_subscriber.<init>(fanWatcher.java:34)
        at onc_subscriber.main(fanWatcher.java:117)
```

FANwatcherユーティリティの使用

FANwatcherユーティリティは、オプションの引数を取ります。

```
FANwatcher <optional Notification type>
```

ここで、

通知タイプのデフォルトは、Oracleが生成したすべてのFANイベント（database/event）です。サブセットを指定することもできます。たとえば、database/event/serviceと指定してサービス関連のイベントのみを表示させたり、database/event/hostと指定してノード関連のイベントのみを表示させたりすることもできます。

すべてのイベントをサブスクライブするには（もっとも役に立つケース）、以下のコマンドを使用します。

```
FANwatcher
```

すると、以下の例に示すようにイベントが受信されます。

クライアント・ノード	GIノード
$ ./FANwatcher >> Executing fanWatcher ...Opening FAN Subscriber Window ...	
	$ srvctl start service –d testy –s testy_pdb_srv

```
** Event Header **
Notification Type: database/event/service
Delivery Time:Tue Jan 27 16:22:39 PST 2025
Generating Node: rac2.oracle.com
Event payload:
VERSION=1.0 event_type=SERVICEMEMBER service=testy_pdb_srv.us.oracle.com
instance=TESTY_3 database=testy db_domain=us.oracle.com host=rachost_725
status=up card=1 reason=USER timestamp=2025-01-27 16:22:39 timezone=-08:00

** Event Header **
Notification Type: database/event/service
Delivery Time:Tue Jan 27 16:22:39 PST 2025
Generating Node: rac2.oracle.com
Event payload:
VERSION=1.0 event_type=SERVICE service=testy_pdb_srv.us.oracle.com
database=testy db_domain=us.oracle.com host=rachost_725
status=up reason=USER timestamp=2025-01-27 16:22:39 timezone=-08:00

** Event Header **
Notification Type: database/event/service
Delivery Time:Tue Jan 27 16:22:40 PST 2025
Generating Node: rac2.oracle.com
Event payload:
VERSION=1.0 event_type=SERVICEMEMBER service=testy_pdb_srv.us.oracle.com
instance=TESTY_1 database=testy db_domain=us.oracle.com host=rachost_723
status=up card=2 reason=USER timestamp=2025-01-27 16:22:40 timezone=-08:00

** Event Header **
Notification Type: database/event/service
Delivery Time:Tue Jan 27 16:22:41 PST 2025
Generating Node: rac2.oracle.com
Event payload:
VERSION=1.0 event_type=SERVICEMEMBER service=testy_pdb_srv.us.oracle.com
instance=TESTY_2 database=testy db_domain=us.oracle.com host=rachost_724
status=up card=3 reason=USER timestamp=2025-01-27 16:22:40 timezone=-08:00
```

サーバー側の高速アプリケーション通知のコールアウト

FANコールアウトは、最小限のプログラム作業でデプロイできる、Oracle RACで使用可能な単純ながらも強力な統合メカニズムを提供します。FANコールアウトは、ラッパー・シェル・スクリプト、つまりFANイベントが発生するたびに実行される、任意のプログラミング言語で記述された、事前コンパイル済みの実行可能ファイルです。FANコールアウトの目的は、ログイン、チケットのファイリング、および外部アクションの実行です。コールアウトの目的は、統合されたクライアント・フェイルオーバーを対象としていません。FANクライアント・フェイルオーバーは、次のセクションの高速接続フェイルオーバーです。ノード・イベントおよびネットワーク・イベントを除いて、FANコールアウトは各ノードにローカルで生成されるFANイベントに対して実行されるため、そのノード上のリソースに影響するアクションのみが対象になります。

FANコールアウト・ディレクトリの構成

Grid Infrastructureによってポストされるそれぞれの FANイベントにより、標準のGrid Infrastructureコールアウト・ディレクトリ（Linuxでは、${GI_Home}/racg/usrco）にデプロイされた各コールアウトが実行されます。

これらのコールアウトが実行される順序は確定的ではなく、Grid Infrastructureによって、すべてのコールアウトがFANイベントごとに非同期の形式で1回起動されることが保証されます。コールアウト・スクリプトまたはプログラムは、システムで必要な数だけインストールできます。特定の順序で実行する必要があるFANコールアウトは、同じコールアウトから起動する必要があります。本番デプロイメントの前に、それぞれのコールアウトの実行パフォーマンスと正確性を入念にテストしてください。

注：コールアウト・ディレクトリについては、Grid Infrastructureをインストールしたシステム・ユーザーにのみ書込み権限があり、そこに含まれるそれぞれのコールアウト実行可能ファイルまたはスクリプトについては、同じGrid Infrastructure所有者にのみ実行権限があることを確認してください。

FANコールアウトの実装

FANコールアウトの書込みは、次の順序で実行されます。

1. FANペイロードの解析
2. 受信FANイベントのフィルタリング
3. イベント処理プログラムの実行

コールアウト引数リストの解析

FANコールアウトの最初の手順は、FANペイロードを解析することです。FANイベントの分類法のセクションで説明したように、それぞれのFANイベント・タイプには一連のペイロード引数が含まれます。

イベント・タイプは、FANペイロードの最初の引数として表示されます。たとえば、次のとおりです。

SERVICEMEMBER VERSION=1.0 service=testy_pdb_srv.us.oracle.com database=testy instance=TESTY_1 host=rachost_723 status=down reason=USER timestamp=2025-01-27 17:56:08 timezone=-08:00 db_domain=us.oracle.com

イベントを解析するためのBASHシェル・スクリプトは、以下のようになります。

```
#!/bin/bash
# 高可用性イベント・ペイロード引数をスキャンして解析
#
NOTIFY_EVENTTYPE=$1 # イベント・タイプを別に処理
for ARGS in $*; do
PROPERTY=`echo $ARGS | $AWK -F"=" '{print $1}'`
VALUE=`echo $ARGS | $AWK -F"=" '{print $2}'`
case $PROPERTY in
VERSION|version) NOTIFY_VERSION=$VALUE ;;
SERVICE|service) NOTIFY_SERVICE=$VALUE ;;
DATABASE|database) NOTIFY_DATABASE=$VALUE ;;
INSTANCE|instance) NOTIFY_INSTANCE=$VALUE ;;
HOST|host) NOTIFY_HOST=$VALUE ;;
STATUS|status) NOTIFY_STATUS=$VALUE ;;
REASON|reason) NOTIFY_REASON=$VALUE ;;
CARD|card) NOTIFY_CARDINALITY=$VALUE ;;
VIP_IPS|vip_ips) NOTIFY_VIPS=$VALUE ;; # public_nw_downの場合のVIP_IPS
TIMESTAMP|timestamp) NOTIFY_LOGDATE=$VALUE ;; # イベント日付をキャッチ
TIMEZONE|timezone) NOTIFY_TZONE=$VALUE ;;
??:??:??)NOTIFY_LOGTIME=$PROPERTY ;; # イベント時刻（hh24:mi:ss）をキャッチ
esac
done
```

受信イベントのフィルタリング

受信イベントは、ペイロード情報に基づいてフィルタリングできます。たとえば、コールアウトから起動されるそれぞれのターゲット・イベント・ハンドラに対して、以下のフィルタリング基準を適用できます。

表7：特定のFANイベント・ハンドラに対するフィルタリング基準の例

ターゲット・イベント・ハンドラの例	フィルタリング対象のサービス	フィルタリング対象のイベント・タイプ（およびステータス）
ローカル・アプリケーションの起動または停止	すべて	SERVICE（upまたはdown） SERVICEMEMBER（upまたはdown） NODE（node down）
ヘルプ・システムのサービス・リクエストのロギング	FIN_APAC、PROD	SERVICE（down） DATABASE（down） NODE（down）
ITスーパーバイザーへの電子メールまたはメッセージ送信	すべて	NODE（nodedown） SERVICE（upまたはdown）
リモート管理コンソールへのSNMPトラップの転送	WEB_GL	すべて
ローカル・ベンダー・コンポーネントへの通知	カスタム削除アプリケーション	NODE（down）

以下のBASHシェル・スクリプトの例では、トラブル・チケット・システム（表5の2番目のフィルタリング例を使用）が呼び出されるのは、Oracle Grid InfrastructureフレームワークがSERVICE、DATABASE、またはNODEイベント・タイプをポストし、ステータスは“down”、“nodedown”、または“public_nw_down”のいずれかで、アプリケーション・サービス名がHQPRODまたはFIN_APACの2つの場合のみです。

```
#!/bin/bash
# 以下の条件を満たすFANイベントが
# 重要なトラブル・チケット・システムに挿入される
# NOTIFY_EVENTTYPE => SERVICE | DATABASE | NODE
# NOTIFY_STATUS => down | public_nw_down | nodedown
# NOTIFY_DATABASE => HQPROD | FIN_APAC
#
if ((( [ $NOTIFY_EVENTTYPE = "SERVICE" ] ||
[ $NOTIFY_EVENTTYPE = "DATABASE" ] || \
[ $NOTIFY_EVENTTYPE = "NODE" ] \
) && \
( [ $NOTIFY_STATUS = "down" ] || \
[ $NOTIFY_STATUS = "public_nw_down" ] || \
  [ $NOTIFY_STATUS = "nodedown " ] \
)) && \
( [ $NOTIFY_DATABASE = "HQPROD" ] || \
[ $NOTIFY_DATABASE = "FIN_APAC" ] \
))
then
<< CALL TROUBLE TICKET LOGGING PROGRAM AND PASS RELEVANT NOTIFY_* ARGUMENTS >>
fi
```

これは、コールアウトで実行できる内容の例です。多数のタスクを実行する（インスタンスが起動したら優先ノード上のサービスを再配置する、サービスが停止したら直ちにクライアント接続を削除する、など）ためのサンプル・コードは、OTNおよびOracle Support経由で入手できます。

特定のノード上で生成されたすべてのイベントをロギングするFANコールアウトの例は、以下のように構成できます。

1. コールアウト・ディレクトリにファイルを作成します。
   - vi $ORA_CRS_HOME/racg/usrco/log_callouts.sh
2. ファイルに以下の行を追加します（BASHスクリプトの作成）。
   - #!/bin/bash

```
echo “`date` : $@” >> [your_path]/admin/`hostname`/callout_log.log
```

3. ファイルの実行権限を設定します。
   - chmod +x $ORA_CRS_HOME/racg/usrco/log_callouts.sh
4. このファイルをGIクラスタのすべてのノードにコピーし、callout_log.logファイルを集計してシステムのすべてのイベントを表示させます。

callout_log.logエントリの例は以下のようになります。

```
Tue Jan 31 17:56:08 PST 2025:SERVICEMEMBER VERSION=1.0
service=testy_pdb_srv.us.oracle.com database=testy instance=TESTY_1
host=rachost_723 status=down reason=USER timestamp=2025-01-27 17:56:08 timezone=-
08:00 db_domain=us.oracle.com
```

FANおよびFANコールアウトについて詳しくは、『*Oracle® Real Application Clusters*管理およびデプロイメント・ガイド、*23ai*』（Part Number F46762-03）の第5章「動的データベース・サービスによるワークロード管理」を参照してください。

高速接続フェイルオーバーでFANを使用するためのアプリケーションの構成

高速接続フェイルオーバー（FCF）は、事前に構成されていて実証済みである、クライアント側のFAN統合です。FANイベントを受信して使用するのに、アプリケーション・コードを変更する必要はありません。常にFCF対応のクライアントを使用する必要があります。

FCFは、Oracle製かサード・パーティ製かに関係なく、すべてのアプリケーション・スタックのためのFANクライアント・ソリューションです。FCFクライアントは、自動的にFANイベントをサブスクライブしてFANイベントに対処します。FCFクライアントは、表1に示したFAN接続署名を使用して、イベントによって影響を受けるデータベース・セッションを認識することができます。FCFクライアントは、FANイベントのペイロードと接続署名のペイロードを一致させることにより、FANイベントを受信したときに対処すべきセッションを認識します。FCFは、イベント・タイプ（表2）、イベント・ステータス（表3）、および理由（表4）に基づいて、以下のように動作します。

- Down - status=downおよびreason=FAILUREを含むイベントは、データベース・サーバーの障害が原因です。FCFクライアントは、TCP/IPタイムアウト時にアプリケーションがハングせず迅速に中断を受信するように、関連する接続を直ちに中断します。無効な接続は、プールされたFCFクライアントから削除されます。アプリケーションがアプリケーション・コンティニュイティまたはTAFを使用するように構成されている場合、アプリケーションは他のインスタンスに自動的にフェイルオーバーされます。
- Planned Down - status=downおよびreason=PLANNEDを含むイベントは、DBAがインスタンス、サービス、またはデータベースを停止し、srvctl、gdsctl、またはData Guard Brokerを使用して計画メンテナンスを開始する場合にポストされます。FCFクライアントは、アクティブな作業を完了させてからセッションを閉じるようにすることで、計画メンテナンスより先にセッションをドレインします。FANおよびFCFによる計画ドレイニングでは、アプリケーションの中断はありません。Oracle Database 12c以降のUCPおよびJDBCクライアントでは、システム・プロパティoracle.ucp.PlannedDrainingPeriodにより、ドレイニングが発生する期間を設定できます。
- Up - status=upを含むイベントは、サービスが初めて起動したか、または再開された場合にポストされます。FCFクライアントは、セッションを再割り当てして、データベース・サーバー全体のロードバランシングを行います。再バランシングは、パフォーマンスに影響しない、段階的なプロセスです。
- Load% - ランタイム・ロードバランシングが有効化されている場合、FANイベントにより、サービス全体に負荷を分散させるためのパーセンテージがアドバイスされます。FCFクライアントは、このアドバイスを用い、Oracle RACを使用しているときはローカルで、そしてOracle GDSを使用しているときはグローバルに、セッションのバランシングを行います。
- Affinity - ランタイム・ロードバランシングが有効化されている場合、FANイベントにより、クライアントとの対話をローカルに維持すべき時期がアドバイスされます。このアドバイスは、同じクライアントで繰り返される取得と返却に適用されます。
- Load %およびAffinityは、プールされたFCFクライアントに適用できます（表8を参照）。他のすべてのアクションは、すべてのFCFクライアントに適用されます。

FCFは、表8に列挙されたリリース以降、以下のクライアントで提供されています。表8では、FANが受信される転送手段もリリース別に示しています。

表8：Oracle Database Release別の高速接続フェイルオーバー

FCFクライアント	Databaseバージョン11g	12c / 19c / 23ai
JDBC Implicit Connection Cache（ICC）	ONS	ICCは非推奨
JDBC Universal Connection Pool（UCP）	ONS	ONS
WebLogic Server Active GridLink	ONS	ONS
UCPを使用しているサード・パーティ製アプリケーション・サーバー：Apache TomCat、IBM WebSphere	ONS	ONS
ODP.Net Unmanaged（OCI）	AQ	ONS
ODP.Net Managed（C#）	ONS	ONS
OCIセッション・プール	AQ	ONS
PHP	AQ	ONS
SQL*Plus	AQ	ONS
Tuxedo	ONS	ONS
JDBC Thinスタンドアロン・クライアント	ONS	ONS
OCI/OCCIドライバ	AQ	ONS
Oracle NetおよびOracle Single Client Access Name（Oracle SCAN）リスナー	ONS	ONS

Oracle Database 12c Release 1（12.1）以降において、Oracle Databaseを使用する場合はONSが転送を行います。FAN AQ HA通知機能は推奨されておらず、古いコンポーネント（Oracle Database 11g）が含まれている場合の下位互換性のみを目的として維持されています。

23ai Grid Infrastructure向けにOracle Notification Serviceを構成する方法

FANでは、Oracle Database 12c以降のすべてのクライアントへのイベントの伝播にOracle Notification Service（ONS）を使用します。ONSは、クラスタ上のOracle Grid Infrastructureの一部として、Oracle Data Guardインストール内に、そしてOracle WebLogicがインストールされている場合にインストールされます。ONSは、登録されている他のすべてのONSデーモンにFANイベントを伝播します。ONSは、Oracle Grid Infrastructureで事前に構成されていて有効化されています。ONSを操作するために手動で構成する必要はありません。

Grid Infrastructureは、FANで事前に構成されています。1つのわずかな例外を除いて、サーバー側でFANの構成や有効化を行うための手順は必要ありません。OCI FANおよびODP FANでは、srvctlまたはgdsctlによってサービスの–notificationをTRUEに設定する必要があります。サーバー側のバリエーションについては、付録を参照してください。

FCFクライアントの場合、ONSはWebLogic Serverの一部と、Oracleクライアント・インストールの一部としてインストールされます。クライアントでのFAN自動構成では、クライアントに応じて、ONSがCLASSPATHまたはORACLE_HOMEにある必要があります。透過的に、FCFクライアントはONSスレッドを生成して、関係するFANイベントをサブスクライブします。

ONSの構成について詳しくは、本書の「付録*A*：*ONS*の構成」を参照してください。

FCFクライアント構成の一般的な手順

ドライバ固有の手順に進む前に、以下の手順に従ってください。

- 動的データベース・サービスを使用する
- Oracle Notification Serviceを使用する
- 必要なONS接続の数を決定する
- ONSにファイアウォールを通過させる方法を決定する
- 高可用性を提供するNET接続を使用する
- 接続チェックを有効化する

動的データベース・サービスを使用する

FANを使用するには、アプリケーションが動的データベース・サービスを使用してデータベースに接続する必要があります。このサービスは、srvctl（Oracle RACが使用されている場合）またはgdsctl（Global Data Servicesが使用されている場合）を使用して作成します。デフォルトのデータベースまたはPDBサービスを使用して接続しないでください。これらは管理のみを目的としており、FAN向けにはサポートされていません。TNSnamesエントリまたはURLでは、サービス名構文を使用する必要があり、動的データベース・サービス名を指定することでベスト・プラクティスに従う必要があります。このセクションで後述する例を参照してください。

Oracle Notification Serviceを使用する

JDBC Thin、Tuxedo、Oracle Database OCI、またはODP.NetクライアントでFANを使用している場合、FANはONS経由で受信されます。Oracle Database 11g OCIまたはODP.NET管理対象外プロバイダのFCFクライアントを使用する場合、FANはAQ経由で受信されます。

Oracle Database 12c以降のバージョンでは、FANクライアントのターゲットに構成は必要ありません。ONS FAN自動構成によって、FCFクライアントによるサーバー側のONSネットワークの検出と自己構成が保証されます。ONSライブラリまたはjarが存在する場合、FANは自動的に有効化されます。Oracle Database 11gでは、まだほとんどのFCFクライアント上でFANを有効化する必要があります。リスナーおよびSQL*Plusでは、クライアント側の構成は必要ありません。

FAN自動構成により、FCFクライアントが必要とするGrid Infrastructureサーバーを列挙する必要がなくなります。クライアントは、すでにTNSアドレス文字列またはURLを使用してリモート・リスナーを検索しています。FAN自動構成では、TNSアドレスを使用してデータベースを検索してから、それぞれのサーバー・データベースにONSのサーバー側アドレスを尋ねます。たとえば、複数のデータベースがある場合、FAN自動構成はそれぞれのデータベースを参照して、それぞれのONS構成を取得します。

Oracle Database 23aiを使用している場合、ONSネットワークはURLから検出されます。LOAD_BALANCEがアドレス・リスト全体でオフになっている場合、それぞれのアドレス・リストに対応するONSノード・グループが自動的に取得されます。これが、ここで示すデフォルトです。すべてのアドレス・リストでLOAD_BALANCEがTRUEまたはONの場合は、1セットのONSエンドポイントのみが作成されます。（これはADDRESS_LIST全体のLOAD_BALANCE値であることに注意してください。各アドレス・リスト内のLOAD_BALANCEは、SCANアドレスを拡張するためのものです。）

必要なONS接続の数を決定する

デフォルトでは、FCFクライアントは、ONS構成の各ノード・グループで冗長性を確保するために3つのホストを維持します。それぞれのノード・グループは、各Grid Infrastructureクラスタまたは各Oracle GDSデータセンターに相当します。たとえば、プライマリ・データベースと複数のOracle Data Guardスタンバイがある場合、デフォルトでは、各ノード・グループで3つのONS接続が維持されます。ノード・グループは、FAN自動構成（Oracle Database 12cではons configuration）を使用して検出されます。node_groupsがFAN自動構成によって定義され、load_balance=false（デフォルト）の場合、それ以上のONSエンドポイントは必要ありません。エンドポイントの数を増やすには、最大接続数を増やしてください。これは、各ノード・グループに適用されます。この例で4に増やすと、各ノードで4つのONS接続が維持されます。

この値を増やすと、ソケットをより多く消費します。

oracle.ons.maxconnections=4

ONSノード・グループ：oracle.ons.nodes

たとえば、Oracle RACとData Guardを使用した環境で、クライアントが複数のクラスタに接続することになっていて、両方のクラスタからFAN

イベントを受信する場合、複数のONSノード・グループが必要になります。FAN自動構成は、12c以降のクライアントおよびデータベースでは、URLまたはTNS名を使用してこれらのノード・グループを作成します。auto-onsを使用しない場合は、oraaccess.xml構成ファイルでノード・グループを指定してください。

2つのクラスタにそれぞれ8つのノードがある状況を見てみます。ノード・リストを2つ（各クラスタに対して1つ）指定します。

AUTOONS creates a separate node group for each addresslist in the TNS connection descriptor:

oracle.ons.nodes.001=node1a:6250,node1b:6250,node1c:6250,node1d:6250,node1e:6250,node1f:6250,node1g:6250,node1h:6250

oracle.ons.nodes.002=node2a:6250,node2b:6250,node2c:6250,node2d:6250,node2e:6250,node2f:6250,node2g:6250,node2h:6250

すべてのアクティブ・ノード・リストでoracle.ons.maxconnectionsをデフォルトの3のままにすると、この場合、ONSクライアントは全体で6つの接続を維持しようとします。

ONSにファイアウォールを通過させる方法を決定する

ONSノード・グループとFCFクライアントの間にファイアウォールがある場合、ONSトラフィックにポートを開く必要があります。推奨されるアプローチは以下のとおりです。

1. ons executableをファイアウォールの例外リストに追加します。
2. ONSが使用するポートをファイアウォールの例外リストに追加します。

高可用性を提供するNET接続を使用する

OCIおよびODP.NET管理対象外プロバイダ・クライアントの場合、以下のTNS名構造を使用します。

Alias = (DESCRIPTION = (CONNECT_TIMEOUT=90)(RETRY_COUNT=100)(RETRY_DELAY=3)
(TRANSPORT_CONNECT_TIMEOUT=1000ms)

(ADDRESS_LIST =

(LOAD_BALANCE=on)

(ADDRESS = (PROTOCOL = TCP)(HOST=primary-SCAN)(PORT=1521)))

(ADDRESS_LIST =

(LOAD_BALANCE=on)

(ADDRESS = (PROTOCOL = TCP)(HOST=secondary-SCAN)(PORT=1521)))

(CONNECT_DATA=(SERVICE_NAME = gold-cloud)))

古いバージョン（11g、12.1.0.2）のJDBC Thinクライアントの場合、以下のURL構造を使用します。

jdbc:oracle:thin = (CONNECT_TIMEOUT=4) (RETRY_COUNT=30)(RETRY_DELAY=3)

(ADDRESS_LIST =

(LOAD_BALANCE=on)

(ADDRESS = (PROTOCOL = TCP)(HOST=primary-SCAN)(PORT=1521)))

(ADDRESS_LIST =

(LOAD_BALANCE=on)

(ADDRESS = (PROTOCOL = TCP)(HOST=secondary-SCAN)(PORT=1521)))

(CONNECT_DATA=(SERVICE_NAME = gold-cloud)))

Oracle Database 12c Release 1（12.1.0.2）より後において、JDBCとOCIは連携しており、すべての接続の説明にOCIバージョンを使用する必要があることに注意してください。

NET接続のベスト・プラクティス

データベースに接続するには、常に動的データベース・サービスを使用します。これらは、srvctlまたはgdsctlを使用して作成します。

デフォルトのデータベースまたはPDBサービスを使用しないでください。これらは管理のみを目的としており、アプリケーションでの使用には適していません。また、FANやその他多数の機能は、マウント時に使用できるため、提供しないでください。

JDBCの場合、現在または以前のRDBMSとともに、最新のクライアント・ドライバ（Oracle Database 23ai）を使用します。

TNS名エントリまたはURLで1つのDESCRIPTIONを使用します。複数使用すると、RETRY_COUNTおよびRETRY_DELAYの使用時、接続に長い遅延が発生します。

CONNECT_TIMEOUT=90またはこれより大きな値に設定することで、OCIおよびODPクライアントのログオン・ストームを防ぎます。11g JDBCクライアントの場合は、一時的な措置として、もっと小さな設定値CONNECT_TIMEOUT=4を使用します。これは、TRANSPORT_CONNECT_TIMEOUTを使用できないからです。

また、JDBCプロパティoracle.net.ns.SQLnetDef.TCP_CONNTIMEOUT_STRは、CONNECT_TIMEOUTより優先されるため、設定しないでください。

OCIの場合はOracle Database 11g Release 2（11.2.0.3）以降、JDBCの場合はOracle Database 12c Release 1（12.1.0.2）以降、アドレスごとにLOAD_BALANCE=ONと設定してSCAN名を展開してください。

接続チェックを有効化する

停止によっては、FCFが処理される前に接続が取得されると、アプリケーションが、古い接続を受信する場合があります。これは、たとえば、着信接続要求と一致してソケットがクローズされるときにクリーン・インスタンスで発生する場合があります。アプリケーションがエラーを受信するのを阻止するには、接続プールで接続チェックを有効化する必要があります。

JDBC Universal Connection Pool

setValidateConnectionOnBorrow（boolean） – 接続プールから接続を取得するときに接続を検証するかどうかを指定します。このメソッドは、プールから取得したすべての接続の検証を可能にします。デフォルト値はfalseです。値をtrueに設定し、検証が実行されるようにします。

OCI接続

サーバーへの接続がFANまたはOCI_ERRORのどちらによって切断されたかを検証するには、アプリケーションでサーバー・ハンドル内のOCI_ATTR_SERVER_STATUS属性の値を確認します。この属性の値がOCI_SERVER_NOT_CONNECTEDの場合、サーバーへの接続は切断されており、ユーザー・セッションを再確立する必要があります。

FAN計画外停止およびFAN計画停止のユースケースでは、いずれの場合も、OCI_ATTR_SERVER_STATUSはOCI_SERVER_NOT_CONNECTEDに設定されます。FANを使用して計画外停止時間を報告すると、アプリケーションは即座にエラーを受信します。FANを使用して計画メンテナンスを報告すると、カスタムOCIプールは、取得の前かつプールへの返却の後に、OCI_ATTR_SERVER_STATUSを確認し、これらの安全な場所のセッションを削除することができます。取得の前かつ返却の後に、クローズされた接続を削除すると、計画メンテナンス時にアプリケーション・エラーが発生することなく、優れたユーザー・エクスペリエンスが実現します。

ODP.NETプロバイダ

CheckConStatusはデフォルトで有効になっています。

このプロパティにより、接続をODP.NET接続プールに返す前に、接続のステータスを確認します。ODP.NETのインストールでは、このレジストリ・エントリは作成されません。ただし、デフォルト値1が使用されます。

カスタムのODP.NETプログラムの場合、ConnectionState.Openをテストします。たとえば次のとおりです。

```
if(OracleConnection.State!=ConnectionState.Open)
  OracleConnection.Open()
```

23ai Javaクライアント向けにFANを構成する方法

Universal Connection Poolの使用

Oracle Database JDBC ThinドライバでFCFを利用するための最適な方法は、Universal Connection Pool（UCP）またはWebLogic Server Active GridLinkを使用することです。Universal Connection Poolのプール・プロパティFastConnectionFailoverEnabledを設定することにより、高速接続フェイルオーバー（FCF）が有効化されます。Active GridLinkでは、FCFがデフォルトで常に有効化されています。IBM WebSphereやApache Tomcatなどのサード・パーティ製アプリケーション・サーバーは、接続プールの置換えとしてUCPをサポートしています。他のWebサーバーによるUCPの組込みについて詳しくは、以下の技術概要を参照してください。

『*Design and Deploy WebSphere Applications for Planned, Unplanned Database Downtimes and Runtime Load Balancing with UCP*』（http://www.oracle.com/technetwork/database/application-development/planned-unplanned-rlb-ucp-websphere-2409214.pdf）

『*Design and deploy Tomcat Applications for Planned, Unplanned Database Downtimes and Runtime Load Balancing with UCP*』（http://www.oracle.com/technetwork/database/application-development/planned-unplanned-rlb-ucp-tomcat-2265175.pdf）

高速接続フェイルオーバーを有効化するには、以下の構成手順に従ってください。

1. コネクション・ファクトリの接続URLでは、サービス名構文を使用する必要があり、動的データベース・サービス名とJDBC URL構造（上記および下記）を指定することでベスト・プラクティスに従う必要があります。他のすべてのURL形式によって高可用性が実現するわけではありません。URLでは、JDBC ThinまたはJDBC OCIを使用できます。
2. ウォレット認証が確立されていない場合、またはクラスタでOracle Grid Infrastructure 12.1.0.2よりも前のバージョンが実行されている場合は、リモートONS構成が必要です。これは、プール・プロパティsetONSConfigurationによって実行できます。このプロパティは、以下の例に示すように、プロパティ・ファイルによって設定できます。

```
PoolDataSource pds = PoolDataSourceFactory.getPoolDataSource();
pds.setConnectionPoolName("FCFSamplePool");
pds.setFastConnectionFailoverEnabled(true);
pds.setONSConfiguration("propertiesfile=/usr/ons/ons.properties");
pds.setConnectionFactoryClassName("oracle.jdbc.pool.OracleDataSource");
pds.setURL("jdbc:oracle:thin@((CONNECT_TIMEOUT=90)(RETRY_COUNT=100)"+
" (RETRY_DELAY=3) (TRANSPORT_CONNECT_TIMEOUT=1000ms)"+
" (ADDRESS_LIST = "+
   " (LOAD_BALANCE=on) "+
   " ( ADDRESS = (PROTOCOL = TCP)(HOST=RAC-SCAN)(PORT=1521))) "+
" (ADDRESS_LIST = "+
   " (LOAD_BALANCE=on) "+
   "( ADDRESS = (PROTOCOL = TCP)(HOST=DG-SCAN)(PORT=1521)))"+
"(CONNECT_DATA=(SERVICE_NAME=service_name)))");
```

指定するプロパティ・ファイルには、oracle.ons.nodesプロパティと、オプションでoracle.ons.walletfileプロパティおよびoracle.ons.walletpasswordプロパティが含まれる必要があります。ons.propertiesファイルの例を以下に示します。

```
oracle.ons.nodes=racnode1:4200,racnode2:4200
```

3. プール・プロパティの設定がsetFastConnectionFailoverEnabled=trueとなっていることを確認します。
4. CLASSPATHには、ons.jar、ucp.jar、およびJDBCドライバのjarファイル（ojdbc11.jarなど）が含まれる必要があります。
5. 以下は必須ではなく、SCANアドレスに対してDNSから返されたSCAN IPアドレスを強制的に並べ替えるための最適化です。

```
oracle.jdbc.thinForceDNSLoadBalancing=true
```

6. Oracle Database 23aiでJDBC Thinを使用している場合、アプリケーション・コンティニュイティを構成して、FANの受信後に接続をフェイルオーバーすることができます。
7. データベースが12cより前のバージョンの場合、または自動構成されたものとは異なるONSエンドポイントが構成に必要な場合は、ONSエンドポイントを以下の例のとおりにして有効化できます。

```
pds.setONSConfiguration("nodes=mysun05:6200,mysun06:6200,
mysun07:6200,mysun08:6200");
```

または、複数のクラスタがある環境でautoonsを使用している場合、autoonsは以下のようなノード・リストを生成します。

```
oracle.ons.nodes.001=node1a:6250,node1b:6250,node1c:6250,node1d:6250,node1e:6250,n
ode1f:6250,node1g:6250,node1h:6250
oracle.ons.nodes.002=node2a:6250,node2b:6250,node2c:6250,node2d:6250,node2e:6250,n
ode2f:6250,node2g:6250,node2h:6250
```

oracle.ons.maxconnectionsは、デフォルトではすべてのアクティブ・ノード・リストに対して3に設定されているため、これを明示的に設定する必要はありません。この例では、ONSクライアントは合計6つの接続を維持しようとします。

ODP.Netクライアント、管理対象プロバイダ、および管理対象外プロバイダ向けにFANを構成する方法

アプリケーションが、Oracle Database管理対象外プロバイダまたはODP.NET管理対象プロバイダとともに高速接続フェイルオーバー（FCF）を利用するためには、ODP.Net接続プールを使用する必要があります。FCFは、接続プールおよびOracle Services for Microsoft Transaction Server（OraMTS）でサポートされています。

Oracle Database 23ai ODP.Net管理対象外/管理対象プロバイダ・クライアントおよび23aiデータベース・サーバー向けの構成

1. クライアント側のODP.NETでFANを有効化するには、接続文字列にHA eventsを指定します。

```
"user id=oracle; password=oracle; data source=HA; pooling=true; HA events=true;“
```

2. クライアント側のODP.NETでランタイム・ロードバランシングを有効化するには、接続文字列にload balancingを指定します。

```
"user id=oracle; password=oracle; data source=HA; pooling=true; HA events=true;
load balanacing=true“
```

3. ODP.Netは、ONSリスナーを参照できる必要があります。FAN autoonsを使用することで、ONSの構成情報がデータベース・サーバーからODP.Netに直接渡されます。autoonsは、TNSnamesを使用してエンドポイントを参照します。完全な高可用性機能を備えた、OCI TNSアドレスのベスト・プラクティスを設定します。

```
myAlias=(DESCRIPTION= (CONNECT_TIMEOUT=90)(RETRY_COUNT=100)(RETRY_DELAY=3)
(TRANSPORT_CONNECT_TIMEOUT=1000ms)
(ADDRESS_LIST=(LOAD_BALANCE=ON)
   (ADDRESS=(PROTOCOL=TCP)(HOST=primary-scan)(PORT=1521)))
(ADDRESS_LIST=(LOAD_BALANCE=ON)
   (ADDRESS=(PROTOCOL=TCP)(HOST=standby-scan)(PORT=1521)))
(CONNECT_DATA=(SERVICE_NAME=service_name)))
```

透過的アプリケーション・フェイルオーバー（TAF）は、FCFおよびODP.Net管理対象外プロバイダとともに使用できます。FANによってセッションが中断されると、TAFは指定したFAILOVER_TYPE（BASICまたはSELECT）を使用してフェイルオーバーを実行します。アプリケーション・コンティニュイティは、将来のリリースで使用可能になります。

ODP.Net管理対象外プロバイダ11gを使用する場合、またはOracle Database Release 11gでODP.Net管理対象外プロバイダを使用する場合、手順にどのような違いがありますか。

Oracle Database 12cより前のバージョンのデータベースまたはクライアントを管理対象外プロバイダとともに使用する場合の手順は、1つの例外を除いて、12cクライアントおよび12cデータベースの場合と同じです。FANイベントがAQ経由で配信され、これには以下の例のように、データベース・サービスにnotificationを設定する必要があります。

```
srvctl modify service -db EMEA -service GOLD -notification TRUE
```

23aiでONSエンドポイントをカスタム構成する場合、手順にどのような違いがありますか

ONS構成をカスタマイズする場合

$ORACLE_HOME/network/adminにあるoraaccess.xmlを編集します。

```
<onsConfig mode="remote">
    <ons database="db1">
        <add name="nodeList" value="racnode1:6700, racnode2:6700" />
    </ons>
    <ons database="db2">
        <add name="nodeList" value="racnode3:6700, racnode3:6700" />
    </ons>
</onsConfig>
```

カスタムONS構成の場合、アプリケーションは、接続する可能性のあるすべてのデータベースに対して<host>:<port>値を指定します。<host>:<port>値ペアは、ONSデーモンがリモート・クライアントと通信する、異なるOracle RACノード上のポートを表します。

ONSの構成について詳しくは、本書の付録を参照してください。

FANおよびODP.Netプロバイダについて詳しくは、『Oracle® Data Provider for .NET開発者ガイド、23ai』（Part Number F46984-09）を参照してください。

ORACLE

OCIクライアント向けにFANを構成する方法

OCIクライアントは、ドライバ・レベルでFANを組み込むため、プーリング・ソリューションに関係なく、すべてのクライアントがFANを使用できます。Oracleデータベースおよびクライアントが12c以降の場合、OCIクライアントはFAN転送にONSを使用します。サーバーとクライアントの両方が、12c以降のバージョンを使用する必要があります。クライアントまたはサーバーのいずれかが11.2以前のバージョンを使用している場合、使用されるFAN転送手段はアドバンスト・キューです。

SQL*PlusおよびPHP向けの構成

1. サービスにnotificationを設定します

```
srvctl modify service -db EMEA -service GOLD -notification TRUE
```

2. PHPクライアントの場合のみ、oci8.events=Onをphp.iniに追加します

```
php.ini:

oci8.events=on
```

重要

oraaccess.xmlが存在し、events=-falseになっているかイベントが指定されていない場合、FANの使用が無効になります。oraaccess.xmlの使用時にFANとSQL*PlusおよびPHPを維持するには、events=-trueと設定します。

3. 23aiクライアントをOracle Database 23aiデータベースとともに使用する場合は、クライアント側のoraaccess.xmlでFANを有効化します。

```
<oraaccess> xmlns="http://xmlns.oracle.com/oci/oraaccess"
    xmlns:oci="http://xmlns.oracle.com/oci/oraaccess"
    schemaLocation="http://xmlns.oracle.com/oci/oraaccess
    http://xmlns.oracle.com/oci/oraaccess.xsd">
    <default_parameters>
        <events>true</events>
    </default_parameters>
</oraaccess>
```

23ai OCIクライアントおよびOracle 23ai Database向けの構成

1. OCIにONSリスナーを参照する場所を伝えます

12c以降では、クライアント・インストールに、クライアント・ライブラリにリンクしたONSが付属しています。ONSの自動構成を使用することで、ONSエンドポイントはTNSアドレスから検出されます。この自動的な方法が、推奨されるアプローチです。ODP.Netのように、ONSの手動構成もoraaccess.xmlを使用してサポートされます。

2. OCI接続向けにFAN高可用性イベントを有効化します

FANを有効化するには、OCIファイルoraaccess.xmlを編集して、グローバル・パラメータeventsを指定する必要があります。このファイルは、$ORACLE_HOME/network/adminにあります。

```
<oraaccess> xmlns="http://xmlns.oracle.com/oci/oraaccess"
    xmlns:oci="http://xmlns.oracle.com/oci/oraaccess"
    schemaLocation="http://xmlns.oracle.com/oci/oraaccess
     http://xmlns.oracle.com/oci/oraaccess.xsd">
    <default_parameters>
        <events>true</events>
    </default_parameters>
</oraaccess>
```

3. OCIにONSリスナーを参照する場所を伝えます

   12c以降では、クライアント・インストールに、クライアント・ライブラリにリンクしたONSが付属しています。ONSの自動構成を使用することで、ONSエンドポイントはTNSアドレスから検出されます。この自動的な方法が、推奨されるアプローチです。ODP.Netのように、ONSの手動構成もoraaccess.xmlを使用してサポートされます。

```
myAlias=(DESCRIPTION= (CONNECT_TIMEOUT=90)(RETRY_COUNT=100)(RETRY_DELAY=3)
(TRANSPORT_CONNECT_TIMEOUT=1000ms)
(ADDRESS_LIST=(LOAD_BALANCE=ON)
    (ADDRESS=(PROTOCOL=TCP)(HOST=primary-scan)(PORT=1521)))
(ADDRESS_LIST=(LOAD_BALANCE=ON)
    (ADDRESS=(PROTOCOL=TCP)(HOST=standby-scan)(PORT=1521)))
(CONNECT_DATA=(SERVICE_NAME=service_name)))
```

4. サーバーでFANをすべてのOCIクライアントに対して有効化します

```
srvctl modify service -db EMEA -service GOLD -notification TRUE
```

   Oracle 23aiデータベース・サーバー上のFANをすべてのOCIクライアント（SQL*Plusを含む）に対して有効化することも必要です。

Oracle Database 23aiでONSエンドポイントをカスタム構成する場合に異なる手順

ONSをカスタム構成する場合は、以下の例のような構文を使用します。これは推奨される方法ではありません。

```
<oraaccess xmlns="http://xmlns.oracle.com/oci/oraaccess"
    xmlns:oci="http://xmlns.oracle.com/oci/oraaccess"
    schemaLocation="http://xmlns.oracle.com/oci/oraaccess
    http://xmlns.oracle.com/oci/oraaccess.xsd">
    <default_parameters>
        <fan>
          <!-- 可能な値は"trace"または"error"のみ -->
            <subscription_failure_action>
                error
            </subscription_failure_action>
        </fan>
        <ons>
            <subscription_wait_timeout>
                5
            </subscription_wait_timeout>
            <auto_config>true</auto_config>
            <!—以下によってONSの手動構成を実行できます。
                通常はauto_configによってこの情報を検出できるため、
                この機能を使用する必要はないことに注意してください。-->
            <servers>
                <address_list>
                  <name>pacific</name>
                  <max_connections>3</max_connections>
                  <hosts>10.228.215.121:25293, 10.228.215.122:25293</hosts>
                </address_list>
                <address_list>
                  <name>Europe</name>
                  <max_connections>3</max_connections>
                  <hosts>myhost1.mydomain.com:25273,
                      myhost2.mydomain.com:25298,
                      myhost3.mydomain.com:30004</hosts>
                </address_list>
            </servers>
        </ons>
        <events>true</events>
    </default_parameters>
</oraaccess>
```

OCIとOracle Database 11g

クライアントまたはデータベースがOracle Database 11g以前を使用している場合、FANイベントはアドバンスト・キュー（AQ）経由で送信されます。Oracle Database 11gアプリケーションの場合、アプリケーション自体が以下を実行する必要があります。

- OCI環境をOCI_EVENTSモードで初期化
- スレッド・ライブラリとのリンク
- srvctlまたはgdsctlを使用してサービスに–notificationを設定

```
srvctl modify service -db EMEA -service GOLD -notification TRUE
```

詳しくは、『*Oracle® Call Interface*開発者ガイド、*23ai*』（F46705-09）の第*11*章「*OCI*での高可用性」を参照してください。

ORACLE

まとめ

FAN - アクションがいつ必要かをアプリケーションに伝達します。

アプリケーション・セッションをサポートしているハードウェアまたはソフトウェアの障害後に、高速リカバリを実現し、波及効果を緩和するには、障害を検出したらセッションを直ちに中断する必要があります。

サービス構成に変更があった場合、Grid InfrastructureとOracle GDSは、システムのサービスに状態の変化が発生したら、直ちにFANイベントをポストします。アプリケーション・サーバーまたはドライバがタイムアウトして問題を検出するのを待つのではなく、アプリケーション・サーバーまたはドライバはFANを使用してこれらのイベントを受信し、直ちに動作します。

計画外DOWNイベントの場合、アプリケーションの中断は最小限に抑えられます。これは、障害が発生したインスタンスまたはノードへの接続は切断され、ソケットは閉じられ、処理中のリクエストは終了し、アプリケーション・ユーザーに直ちに通知されるためです。OCIでTAFを、またはJavaでアプリケーション・コンティニュイティを使用している場合、セッションが機能しているサービスで再確立されている間、ユーザーはわずかな中断を経験します。取得されなかったセッションは直ちにクリーンアップされ、接続をリクエストしているアプリケーション・ユーザーは、機能サービスのみを提供するインスタンスに送られます。

計画DOWNイベントの場合、セッションが接続プールに返されたときに、アプリケーションがエラーを一切受信しない安全な場所のセッションを削除することで、アプリケーションの中断は最小限に抑えられます。計画外の場合も同じで、取得されなかったセッションは直ちにクリーンアップされ、接続をリクエストしているアプリケーション・ユーザーは、機能サービスのみを提供するインスタンスに送られます。

サービスが起動すると、UPイベント向けに新しい接続が自動的に作成されるため、アプリケーションは直ちに追加リソースを利用できます。

本書で説明したFANwatcherユーティリティにより、構成されたONSトポロジを確認し、サブスクライブしているクライアントがイベントを受信できるかどうかを判断することができます。

付録A：ONSの構成

ONSは、クラスタ上のOracle Grid Infrastructureの一部として、Oracle Data Guardインストール内に、そしてOracle WebLogic Server Active Gridlinkがインストールされている場合にインストールされます。ONSによって、FANイベントは、それが登録されている他のすべてのONSデーモンに伝播します。ONSを手動で構成する必要はありません。ONSは、Oracleクライアント・インストールの一部としてもインストールされ、リモートONS構成プロセスを通じ、スタンドアロン・クライアント・アプリケーションによって起動される場合があります。

Oracleアプリケーション・クライアントは、ONSデーモンを通じて、自身に関係するイベントをサブスクライブします。

ONS構成ファイル

ONS構成ファイル$ORACLE_HOME/opmn/conf/ons.configでは、ONSがどのように動作するかを定義します。このファイルの情報は、表9に示すように、キー/値ペアのセットとして格納されます。

表9：ons.configのパラメータと定義

パラメータ	定義
localport	ONSがローカル・クライアントとの通信に使用する、ローカル・ホスト上のポート番号。Grid Infrastructureのデフォルトは、localport=6100です。
remoteport	他のONSデーモンへの通信のためにONSによって使用されるポート番号。Grid Infrastructureのデフォルト値は、remoteport=6200です。
nodes	通信する他のONSデーモンを示すノードおよびポートのカンマ区切りリスト。ポート値は、おのおののONSデーモンがリスニングするremoteportエントリです。Grid Infrastructure構成では、クラスタのすべてのノードに名前が付けられます。例は以下のようになります。odes=myhost1.example.com:6500,myhost2.example.com:6500,
logcomp（オプションのパラメータ）	ロギングするサブコンポーネントを指定する、オプションのパラメータ。フォーマットは次のとおりです。<component>[<subcomponent>,...]サブコンポーネントの前に感嘆符（!）を付けることで、そのサブコンポーネントを除外することもできます。たとえば、secure を除くすべての（all）コンポーネントをロギングするには、logcomp=ons[all,!secure]と指定します。
logfile（オプションのパラメータ）	メッセージのロギングに使用されるログ・ファイルの場所。デフォルト値は、logfile=$ORACLE_HOME/opmn/logs/ons.logです。
walletfile（オプションのパラメータ）	SSL証明書を格納するためにOracle Secure Sockets Layer（SSL）によって使用されるウォレット・ファイルを指定します。ウォレット・ファイルがONSに指定されている場合は、他のONSインスタンスとの通信時にSSLが使用され、接続を試みるすべてのONSインスタンスからSSL証明書の認証が要求されます。 12.1.0.2では、Grid Infrastructureインストールにおいてデフォルトでwalletfileが設定されるため、すべてのONS接続でSSLが使用されます。既存のUCPユーザーは、walletfileを使用していることを検証し、サーバー上のウォレットと同じ内容であることを確認する必要があります。
Useocr（オプションのパラメータ（GIサーバー専用））	このパラメータは、Grid Infrastructureノードでのみ使用されます。このパラメータは、ONSがOracle Cluster Registry（OCR）にすべてのGI構成を格納すべきかどうかを示します。OCRに情報を格納するには、useocr=onを使用します。
allowgroup（オプションのパラメータ）	localportに接続するユーザー・グループを示すONS設定を指定します。trueに設定された場合、ONSにより、同じOSグループ内のユーザーはそのローカル・ポートに接続できます。falseに設定された場合、ONSデーモンを起動したユーザーのみがローカル・ポートにアクセスできます。このパラメータのデフォルト値はfalseです。リモートONS構成を使用している場合、このパラメータを設定する必要はありません。

Grid Infrastructureインストールでのこれらのパラメータの変更は、srvctlでmodify nodeappsコマンドを使用して行われることに注意してください。以下のヘルプ画面の抜粋では、関連するパラメータを示しています。

```
$ srvctl modify nodeapps -h


Modifies the configuration for a node application.


Usage: srvctl modify nodeapps {[-node <node_name> -address
{<vip_name>|<ip>}/<netmask>[/if1[|if2...]] [-skip]] | [-subnet
<subnet>/<netmask>[/if1[|if2|...]]]} [-nettype {STATIC|DHCP|AUTOCONFIG|MIXED}]
[-emport <em_port>] [ -onslocalport <ons_local_port> ] [-onsremoteport
<ons_remote_port> ] [-remoteservers <host>[:<port>][,<host>[:<port>]...]]
[-clientdata <file>] [-pingtarget "<pingtarget_list>"] [-verbose]
    -node <node_name>                   Node name
    -onslocalport <ons_local_port> ONS listening port for local client
connections
    -onsremoteport <ons_remote_port> ONS listening port for connections from
remote hosts
    -remoteservers                      <host>[:<port>][,<host>[:<port>]...] List
of remote host/port pairs for ONS daemons outside this cluster
    -clientdata <file>                  file with wallet to import, or empty string
to delete wallet used for SSL to secure ONS communication
    -verbose                            Verbose output
    -help                               Print usage
```

クライアント側ONS構成

クライアント側ONSは非推奨のプラクティスです。アプリケーションでは、AUTO-ONSまたはリモートONSを使用します。

ONSデーモンを稼働する必要があるOracleクライアントの場合、ons.configファイルを編集してONSデーモンを再起動する必要があります。

Oracleクライアント・インストールのデフォルトの場所は、$ORACLE_HOME/opmn/conf/ons.configですが、他のOracle製品のインストールによって異なる場合があります。

Oracleクライアントのons.configファイルの例は以下のとおりです。

```
# This is an example ons.config file
 #
# The first three values are required
localport=4100
remoteport=4200
nodes=racnode1.example.com:6200,racnode2.example.com:6200
```

この例で示されているように、このクライアントが自身に関係するFANイベントを受信するように、Oracle RACインスタンスが実行されるクラスタ・ノードを指定する必要があります。ONSは、構成するトポロジ内で実行されているデーモンを検出するため、すべてのクラスタ・ノードを指定する必要はありません。ただし、nodes=...リストに1つのノードまたはノードのサブセットのみが指定されている場合、ONSトポロジが検出されて構成されるときにこのノードが停止する可能性があるというリスクがあります。

$ORACLE_HOME/opmn/bin/onsctlユーティリティを使用してONSデーモンを起動できます。

onsctlコマンド

onsctlを使用して、ONSデーモンを起動、シャットダウン、再構成、および監視することができます。onsctlに使用可能なコマンド・オプションは以下のとおりです。

表10：onsctlのコマンド、操作、出力

コマンド	操作	出力
Start	ONSデーモンを起動します	onsctl: ons started
Shutdown	ONSデーモンを停止します	onsctl: shutting down ons daemon
Reload	ons.configファイルを再読込みし、設定を更新します（ONSデーモンは停止しません）	
ping [max-retry]	ローカルONSデーモンが稼働していることを検証します。試行回数はmax-retry回です	ons is running
Debug	ローカルONSデーモンに関するデバッグ情報を出力します	
Usage	詳細なヘルプ画面を出力します。	
Help	使用に関するシンプルな情報を出力します	

ONSは、クラスタで動作している場合、Grid Infrastructureの一部として管理されることに注意してください。このヘルプ画面で示すように、srvctlユーティリティにより、ONSはnodeappsオプションを使用して起動および停止することができます。

```
$ srvctl start nodeapps -h
Start the node applications running on a node.

Usage: srvctl start nodeapps [-node <node_name>] [-adminhelper | -onsonly]
[-verbose]
    -node <node_name>          Node name
    -adminhelper               Start Administrator helper only
    -onsonly                   Start ONS only
    -verbose                   Verbose output
    -help                      Print usage
```

ONSトポロジの検証

onsctl debugコマンドは、ONSデーモンで使用されるトポロジを確認するのに便利な出力を生成します。ONSデーモンが存在するサーバー:ポートの組合せが表示されるため、FANイベントを受信できます。

4ノードのGrid Infrastructureクラスタのonsctl debug出力の編集済みの例を以下に示します。

```
== rac1.oracle.com:6200 5844 15/01/28 17:50:50 ==

Listener:

 TYPE                 BIND ADDRESS                          PORT      SOCKET
-------  ------------------------------------  --------  ---------
Local                                    ::1      6100           6
Local                              127.0.0.1      6100           7
Remote                                   any      6200           8
Remote                                   any      6200           -

Connection Topology:(4)

                      IP                          PORT      VERS      TIME
 ------------------------------------  -------  -------  ---------------
                          10.10.10.247  6200           4  54c5a3e3
                            **                                  10.10.10.244      6200
                            **                                  10.10.10.245      6200
                            **                                  10.10.10.246      6200
                          10.10.10.246  6200           4  54c5a3e3
                            **                                  10.10.10.244      6200
                            **                                  10.10.10.245      6200
                            **                                  10.10.10.247      6200
                          10.10.10.245  6200           4  54c5a3e3
                            **                                  10.10.10.244      6200
                            **                                  10.10.10.247      6200
                            **                                  10.10.10.246      6200
                          10.10.10.244  6200           4  54c5a3e3=
                            **                                  10.10.10.247      6200
                            **                                  10.10.10.245      6200
                            **                                  10.10.10.246      6200

Server connections:

    ID              CONNECTION ADDRESS              PORT      FLAGS      SENDQ      REF     WSAQ
---------  ------------------------------  -------  ---------  ---------  -------------
0                              10.10.10.245  6200      010405     00000      001
1                              10.10.10.246  6200      010405     00000      001
2                              10.10.10.247  6200      010405     00000      001
```

Listener sectionでは、通信に使用されるlocalportおよびremoteportのポート番号が示されています。

Connection Topology: (4)というタイトルのセクションでは、ONSデーモンが実行されているIPアドレスとポート番号が示されています（この出力を生成したデーモンが認識）。この場合、**rac1.oracle.com:6200**で実行されているデーモンは、以下のIPアドレス:ポートの組合せのONSデーモンと通信しています。10.10.10.247:6200、10.10.10.246:6200、10.10.10.245:6200、および10.10.10.244:6200（rac1.oracle.comで実行されているローカル・デーモン）これらのONSプロセスがそれぞれ認識しているデーモンは、サブエントリとして示されます。

Server connections:というタイトルのセクションでは、示されているローカル・デーモン以外のすべてのONSデーモンが示されています。

リモートONS構成

UCP for JDBCは、ONSConfigurationプール・プロパティによってONS構成をサポートします。このプロパティを使用して、リモートONS構成を設定します。パラメータ文字列は、ONS構成ファイル（ons.config）の内容によく似ています。文字列には、改行文字（\n）で区切られた名前=値ペアのリストが含まれます。名前は、nodes、walletfile、またはwalletpasswordのいずれかになります。

パラメータ文字列は、ホスト:ポートのペアのカンマ区切りリストとして、少なくともONS構成ノード属性を指定する必要があります。SSLは、walletfile属性がOracleウォレット・ファイルとして定義されている場合に使用されます。

PoolDataSourceのONS構成文字列を設定するJDBCコードの例を以下に示します。

```
PoolDataSource pds = PoolDataSourceFactory.getPoolDataSource();



pds.setConnectionPoolName("FCFSamplePool");
pds.setFastConnectionFailoverEnabled(true);
pds.setONSConfiguration("nodes=racnode1:4200,racnode2:4200\nwalletfile=
/oracle11/onswalletfile");
pds.setConnectionFactoryClassName("oracle.jdbc.pool.OracleDataSource");
pds.setURL("jdbc:oracle:thin@((CONNECT_TIMEOUT=90)(RETRY_COUNT=100)"+
" (RETRY_DELAY=3) (TRANSPORT_CONNECT_TIMEOUT=1000ms)"+
" (ADDRESS_LIST = "+
" (LOAD_BALANCE=on) "+
" ( ADDRESS = (PROTOCOL = TCP)(HOST=RAC-SCAN)(PORT=1521))) "+
  " (ADDRESS_LIST = "+
  " (LOAD_BALANCE=on) "+
"( ADDRESS = (PROTOCOL = TCP)(HOST=DG-SCAN)(PORT=1521)))"+
"(CONNECT_DATA=(SERVICE_NAME=service_name)))");
```

アプリケーションがOracle Database 12c以降のUCPを使用し、ONSウォレットやキーストアを必要としない場合、setONSConfigurationの方法はもはや不要です。そのアプリケーションでは、ONSの自動構成を使用できます。

Javaプロパティ・ファイルを使用してONS構成を設定することもできます。この場合、setONSConfigurationに渡されるname=valueは、propertyfile= <path to ons.properties file>のみです。

```
PoolDataSource pds = PoolDataSourceFactory.getPoolDataSource();



pds.setConnectionPoolName("FCFSamplePool");
pds.setFastConnectionFailoverEnabled(true);
pds.setONSConfiguration("propertiesfile=/usr/ons/ons.properties");
pds.setConnectionFactoryClassName("oracle.jdbc.pool.OracleDataSource");
pds.setURL("jdbc:oracle:thin@((CONNECT_TIMEOUT=90)(RETRY_COUNT=100)"+
" (RETRY_DELAY=3) (TRANSPORT_CONNECT_TIMEOUT=1000ms)"+
" (ADDRESS_LIST = "+
" (LOAD_BALANCE=on) "+
" ( ADDRESS = (PROTOCOL = TCP)(HOST=RAC-SCAN)(PORT=1521)) "+
" (ADDRESS_LIST = "+
  " (LOAD_BALANCE=on) "+
  "( ADDRESS = (PROTOCOL = TCP)(HOST=DG-SCAN)(PORT=1521))"+
"(CONNECT_DATA=(SERVICE_NAME=service_name)))");
```

指定するプロパティ・ファイルには、oracle.ons.nodesプロパティと、オプションでoracle.ons.walletfileプロパティおよびoracle.ons.walletpasswordプロパティが含まれる必要があります。ons.propertiesファイルの例を以下に示します。

```
oracle.ons.nodes=racnode1:4200,racnode2:4200
oracle.ons.walletfile=/oracle11/onswalletfile
```

付録B：FANのトラブルシューティング

- 以下のチェックリストを使用して、FANの配信と取得に関連する問題を診断できます。Oracle Clusterware Grid InfrastructureまたはGlobal Data Servicesのいずれかまたは両方を使用していますか。FANでは、ポストするためにOracle ClusterwareまたはOracle GDSが必要です。これらは、Oracle Restart、Oracle RAC、Oracle RAC One、Oracle DG、およびOracle ADGとともに使用できます。重要な点は、データベースが監視されているということです。
- 動的データベース・サービスが使用されていますか。動的データベース・サービスは、srvctlまたはgdsctlを使用して作成され、管理されるサービスです。
- クライアントは、本書で説明した推奨される接続文字列のいずれかを使用して接続していますか。「FCFクライアント構成の一般的な手順」または「FANを構成する方法」（特定のクライアント・タイプ向け）というタイトルのセクションに示した例を参照してください。
- FANイベントはデータベース層で生成されていますか。任意のデータベース・ノードにFANWatcherをインストールし、FANイベントを生成して受信できることを確認してください。
- FANイベントはクライアント層または中間層で生成されていますか。クライアント層または中間層のノードにFANWatcherをインストールし、FANイベントを受信できることを確認してください。
    - クライアント層または中間層で受信されたイベントが、接続しているDATABASEまたはSERVICE向けであることを確認してください。この情報のFANイベント・ペイロードを確認してください。
- クライアント側のFAN用の設定は適切ですか。
    - Universal Connection PoolとJDBC Thinドライバの場合は、それらの使用時にブール型のプール・プロパティの設定がFastConnectionFailoverEnabled = trueとなっていることを確認します。
    - ODP.Netの場合は、接続文字列にpooling=true; HA events=trueが設定されていることを確認します。
    - OCIクライアントの場合は、「*OCIクライアント向けにFANを構成する方法*」のセクションを参照してください。そこでは、次の方法の例が紹介されています。
        - “<events>true</events>”をoraaccess.xmlに設定し、次のようにして動的データベース・サービスで通知を有効にします。“srvctl modify service –notification TRUE …
    - WebLogic Active Grid Linkの場合、FANはデフォルトで有効になっています。これは管理コンソールで確認できます。WebLogicの場合も、ONSエンド・ポイント向けに、管理コンソールでons.configurationを設定してください。
    - Universal Connection Poolを使用しているJDBC OCIクライアントの場合は、fastConnectionFailover=trueと設定してください。
- 必要なパッチを確認してください。WebLogic Active GridLinkでは、19033547、20907322が必要です。
- アプリケーションが、使用しているプールに接続を頻繁に返しているかどうかも確認する必要があります。これは、計画ドレイニング、ランタイム・ロードバランシング、アフィニティ・ルーティングなどの高速接続フェイルオーバー機能で必要です。また、アプリケーション・コンティニュイティでも必要とされます。

ORACLE

Connect with us

+1.800.ORACLE1までご連絡いただくか、**oracle.com**をご覧ください。北米以外の地域では、**oracle.com/contact**で最寄りの営業所をご確認いただけます。

blogs.oracle.com facebook.com/oracle twitter.com/oracle