

## アプリケーション・コンティニューティ

### 準備のためのチェックリスト

Oracle ホワイト・ペーパー | 2020年5月



## 目次

アプリケーション・コンティニューイティ・チェックリスト	1
アプリケーションとサーバーのタイムアウトの整合	1
TNSまたはURLにおける高可用性の構成	1
高速アプリケーション通知（FAN）の有効化	2
FANの監視	2
アプリケーション・コンティニューイティを使用する場合の手順	3
Oracle JDBCアプリケーションの場合	4
その他の資料	6

## アプリケーション・コンティニューイティ・チェックリスト

以下のチェックリストは、Oracle Database 12cのアプリケーション・コンティニューイティ機能を使用するように使用環境を準備する場合に役立ちます。アプリケーション・コンティニューイティがデータベース・サービスで有効になっていない場合、あるいはアプリケーションで使用されない場合でも、このホワイト・ペーパーに記載されている内容には、可用性を維持できるようにシステムを準備する上で大きな価値があります。

### アプリケーションとサーバーのタイムアウトの整合

アプリケーション・レベルのタイムアウトが、基盤システムの検出およびリカバリのための時間として設定されているタイムアウトより短いと、基盤となるリカバリとリプレイを完了するための時間が不足します。時間の整合性が保たれていないと、システムのリカバリが完了する前にアプリケーション・コンティニューイティによるリプレイが開始され、潜在的に、複数のリプレイ試行が成功するまで繰り返される可能性があり、もっと悪い場合にはリクエストが完全にタイムアウトし、アプリケーションまたはユーザーにエラーが返されます。

アプリケーションでREAD\_TIMEOUTやHTTP\_REQUEST\_TIMEOUT、または何らかのカスタム・タイムアウトを使用することを考慮してください。続いて、次のシンプルな指針を適用します。

READ\_TIMEOUT > EXADATA の特別なノード・エビクション (FDDN) (12.1.0.2で2秒)

READ\_TIMEOUT > MISSCOUNT (デフォルトは30秒、12.1.0.2では変更可能)

READ\_TIMEOUT > Data Guardオブザーバ: FastStartFailoverThreshold (デフォルトは30秒、変更可能)  
FastStartFailoverThreshold > MISSCOUNT (2回以上)

READ\_TIMEOUT > FAST\_START\_MTTR\_TARGET  
READ\_TIMEOUT > Oracle Netレベル: (RETRY\_COUNT+1) \* RETRY\_DELAY  
と、

READ\_TIMEOUT < Replay\_Initiation\_Timeout (サービス上で変更可能、デフォルトで300秒)

リクエストのキャンセルが早すぎないようにするため、アプリケーションのタイムアウト値を次の最大値より大きい値にする必要があります。

(MISSCOUNT (またはFDNN) + FAST\_START\_MTTR\_TARGET), (FastStartFailoverThreshold + FAST\_START\_MTTR\_TARGET + 開くまでの時間)

### TNSまたはURLにおける高可用性の構成

フェイルオーバー、スイッチオーバー、フォールバック、基本的な起動時に接続を成功させるには、次のTNS/URL構成を推奨します。

TNSnamesまたはURLで、RETRY\_COUNT、RETRY\_DELAY、CONNECT\_TIMEOUT、TRANSPORT\_CONNECT\_TIMEOUTパラメータを設定し、サービスと接続が正常に行われるまで接続リクエストが待機できるようにします。

CONNECT\_TIMEOUTを高い値に設定して、ログイン・ストームを回避します。値を低くすると、アプリケーションまたはプールのキャンセルや接続の再試行のために、ログイン'合戦'が発生することがあります。

(RETRY\_COUNT+1) × RETRY\_DELAYまたはCONNECT\_TIMEOUTを応答時間のSLAより大きい値に設定しないでください。アプリケーションは応答時間のSLA内で接続するか、エラーを受信することになります。

これらは、接続に高可用性を構成する場合の一般的な推奨事項です。EZCONNECTには高可用性機能がないため、クライアントでは簡易接続ネーミングを使用しないでください。

次に示すのは、12.2以上用のすべてのOracleドライバに推奨されるTNSです。

```
Alias (or URL) = (DESCRIPTION =  
(CONNECT_TIMEOUT= 120)(RETRY_COUNT=20) RETRY_DELAY=3)  
(TRANSPORT_CONNECT_TIMEOUT=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)))
```

12.1のJDBC接続では、次の式を使用します。

```
(DESCRIPTION =  
(CONNECT_TIMEOUT= 15) (RETRY_COUNT=20)(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)))
```

## 高速アプリケーション通知 (FAN) の有効化

FANを有効にする必要があります。FANは、アプリケーションのフェイルオーバーを中断する場合の必須コンポーネントです。ノードまたはネットワークが停止した場合は、アプリケーションをリアルタイムで中断する必要があります。FANを有効化しておかないと、HARDフィジカル障害の発生時にアプリケーションがハングします。FANなしでのハング時間は8～15分間です。Oracle Database 12c以降、FANでは重要な3つの機能強化が図られています。

- FANは、Oracle Real Application Clusters (Oracle RAC) に付属しているデフォルトの機能で、自動構成されて有効化されています。FANはURLを読み取り、それ自体をクライアントで構成します。FANは、いつでも、Grid Infrastructureクラスタで構成されます。上記のURL形式を使用することは、FANの自動構成を行う上で重要です (別の形式を使用すると、FANを自動構成できなくなります)。
- すべてのOracleクライアントでは、Oracle Notification ServiceをFANの転送サービスとして使用します。
- FANは、FANイベントでデータセンターを包含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ます。

## FANの監視

FANWatcherユーティリティを使用してイベントのポスティングと受信を検証します。更新されたバージョン (2017年7月) のFANWatcherは、OTNから、またはWebLogic Serverブログ経由で入手できます。データベース・ロールの変更が実行された場合のサーバー移行の影響について詳しくは、

高速アプリケーション通知 -

<http://www.oracle.com/technetwork/database/options/clustering/applicationcontinuity/learnmore/fastapplicationnotification12c-2538999.pdf>

FANイベントの監視 -

<https://blogs.oracle.com/weblogicserver/monitoring-fan-events>

UCPトレースは、以下の手順で有効化できます。UCP/TomcatでFANが処理中であることを示すには：

ロギングはプロパティ・ファイルを使用して構成可能です。プロパティ・ファイルの場所は、ロギング構成ファイルのプロパティのJavaプロパティとして設定する必要があります。たとえば、次のように設定します。

```
java -Djava.util.logging.config.file=myLog.properties
```

ロギング・プロパティ・ファイルでは、ログの書込みに使用するハンドラ、ログの書式指定に使用するフォーマッタ、デフォルトのログ・レベル、および特定のパッケージまたはクラスのログ・レベルを定義します。たとえば次のとおりです。

```
handlers = java.util.logging.ConsoleHandler
java.util.logging.ConsoleHandler.level = ALL
java.util.logging.ConsoleHandler.formatter =
java.util.logging.SimpleFormatter

oracle.ucp.level = FINEST
oracle.ucp.jdbc.PoolDataSource = WARNING
```

機能に固有のロギングは、ロギング構成ファイルでプロパティを設定することによって有効化できます。すべての機能の場合：

```
clio.feature.all = on
```

または、次のような固有の機能の場合：

```
clio.feature.conn_construction = on
lio.feature.high_availability = on
clio.feature.load_balancing = on
clio.feature.transaction_affinity = on
```

詳しくは、オラクルのドキュメント：

『Universal Connection Pool開発者ガイド』 - 第12章「UCPでのロギングの概要」、『Oracle JDBC開発者ガイド』 - 第34章「Oracle JDBCの診断機能」を参照してください。

## アプリケーション・コンティニューイティを使用する場合の手順

### 接続プールへの接続の返却

アプリケーションは、リクエストごとに接続を接続プールに返却します。ベスト・プラクティスは、必要とする場合にのみアプリケーションが接続をチェックアウトすることです。使用していないときに接続を保持することは、勧められる方法ではありません。したがってアプリケーションでは、接続をチェックアウトし、処理が完了したらすぐにその接続をチェックインします。そうすれば、後続の他のスレッドや、自分のスレッドで再び必要になったときに接続を使用できます。この手法に従うことにより、キャプチャを再開および終了するのに安全な場所を識別するためのアプリケーション・コンティニューイティで使用するリクエストの境界も組み込まれます。

### FAILOVER\_RESTOREの使用

アプリケーションが接続の値を事前設定しているかどうかをチェックします。一部のアプリケーションと中間層アプリケーションでは、たとえばすべての接続の言語またはタイムゾーンが事前設定されるように、接続プールを構成します。

リクエスト以外で接続にセッションの状態を意図的に設定し、この状態を想定してリクエストを行う場合は、再実行の前にこの状態を作成し直して再実行する必要があります。

もっとも一般的な状態は、FAILOVER\_RESTOREをLEVEL1に設定することによって自動的にリストアされます。

以下のいずれかのオプションを選択します。

- FAILOVER\_RESTORE=LEVEL1（サービスで設定）
- 接続初期化コールバック（Javaの場合）または（古い）TAFコールバック（OCIの場合）
- Oracle Universal Connection PoolまたはOracle WebLogic Serverの接続ラベル付け

Oracle RDBMS 12.2以降では、FAILOVER\_RESTORE=LEVEL1が推奨される方法です。

### アプリケーションでの可変値の使用の有効化

可変関数とは、実行されるたびに新しい値を返す可能性のある関数です。SYSDATE、SYSTIMESTAMP、SYS\_GUID、sequence.NEXTVALについては、可変関数の元の結果を保持できるようになっています。元の値が保持されず、再実行時に異なる値がアプリケーションに返される場合は、再実行が拒否されます。

アプリケーション・ユーザーにはGRANT KEEPを使用し、シーケンス所有者にはKEEP句を使用して、可変オブジェクトを構成します。KEEP権限が付与されると、関数の元の結果が再実行時に適用されます。

たとえば次のとおりです。

```
SQL> GRANT [KEEP DATE TIME | KEEP SYSGUID] ... to USER
```

```
SQL> GRANT KEEP SEQUENCE mySequence to myUser on sequence.object
```

## Oracle JDBCアプリケーションの場合

### Oracle JDBCステートメント・キャッシュの有効化

Oracle JDBCベースのアプリケーションの場合は、パフォーマンスを高めるため常にOracle JDBCステートメント・キャッシュを使用します。アプリケーション・コンティニューイティを使用している場合は、Oracle JDBCのリファレンス・ドキュメントの説明に従って、アプリケーション・サーバー・レベル（Tomcat、WebLogic、WebSphereなど）でステートメント・キャッシュを無効にすることが必須となります。

アプリケーション・サーバー・レベルのステートメント・キャッシュ（WebLogicやサード・パーティ製アプリケーション・サーバーのステートメント・キャッシュなど）が有効化されている場合は、リプレイが使用されるときに無効化する必要があります。代わりに、JDBCステートメント・キャッシュを構成します。JDBCとオラクル向けに最適化されており、アプリケーション・コンティニューイティに対応しているため、このキャッシュの方がパフォーマンスに優れています。

oracle.jdbc.implicitstatementcachesize=nnnを使用します。

nnnを正の値に設定すると、暗黙的ステートメント・キャッシュが有効化されます。

### ガベージ・コレクタのチューニング

多くのアプリでは、デフォルトのガベージ・コレクタをチューニングすることが適切です。極めて高い性能のアプリの場合には、JVMチューニングを実行する必要があります。性能が極めて高い場合に推奨されるのは、以下の設定をJavaコマンドラインに追加することです（両方の属性を同じ値に設定する必要があります）。

```
java -Xms 2000m -Xmx 2000m
```

## コミット

アプリケーション・コンティニューイティでは、すべてのスタイルのCOMMIT（トップレベル、AUTOCOMMIT、PLSQLに埋め込まれているCOMMIT）をサポートしています。アプリケーションでトップレベルのCOMMITを使用する場合、それはスタンドアロンのOCOMMITまたはCOMMIT()であり、SESSION\_STATE\_CONSISTENCY=STATICモード（12.2.0.1）を使用する場合を含め、リプレイを完全にサポートしています。AUTOCOMMITまたはPLSQLに埋め込まれたCOMMITをアプリケーションで使用している場合は、COMMITを含む呼び出しの実行が完了しなかったことがアプリケーション・コンティニューイティで検出されると再実行ができないことがあります。アプリケーション・コンティニューイティでは正しい処理が行われます。

Oracle JDBCアプリケーションで“トップレベル”のCOMMITを使用する場合は、アプリケーションで、またはUCPのプロパティとして、AUTOCOMMITを無効にする必要があります。これは、Apache Tomcat、IBM WebSphere、RedHat JBossなどのサード・パーティのアプリケーション・サーバーにUCPが埋め込まれている場合には特に重要です（アプリケーションでAUTOCOMMITが不要な場合）。

## 保護レベル

以下の場所にあるアプリケーション・コンティニューイティ・ホワイト・ペーパーの説明に従って、保護レベルのトレース・レポートと具象クラス分析を実行してください。

Oracle Database 12c Release 2のアプリケーション・コンティニューイティ：

*具象クラスのACチェックの使用*

および

*カバレッジの測定*

(<https://www.oracle.com/technetwork/database/options/clustering/applicationcontinuity/overview/application-continuity-wp-12c-1966213.pdf>)

統計監視Beanは、次の場所のucpdemos.jarサンプル・コードに含まれています。

<http://www.oracle.com/technetwork/database/features/jdbc/jdbc-ucp-122-3110062.html>:

MBeanについて詳しくは、コード・サンプルUniversalConnectionPoolManagerMBeanSample.javaおよびUniversalConnectionPoolManagerSample.javaを参照してください。

トレース・サーバーとOracle JDBCのリプレイ機能トレースでは、カバレッジを示します。

Oracle JDBCの非推奨クラス（具象クラス）は使用しないようにする必要があります。これは、12.1でのハード制限ですが、18cのいくつかの具象クラスでは緩和されています。具象クラスの非推奨化に関する情報（推奨されない具象クラスをアプリケーションで使用している場合の対策を含む）については、My Oracle Support Note 1364193.1を参照してください


(<https://support.oracle.com/CSP/main/article?cmd=show&type=NOT&id=1364193.1>)。

## 診断

“OracleLog.properties”ファイルで以下を使用します（ログ・ファイルをホスティングしているディレクトリへの絶対パスを使用する必要があります）。

```
handlers = java.util.logging.FileHandler
java.util.logging.FileHandler.pattern = <REPLACE THIS PATH
FIRST>/jdbc%u.log
java.util.logging.FileHandler.limit = 10000000
java.util.logging.FileHandler.count = 90000
java.util.logging.FileHandler.formatter =
oracle.ucp.util.logging.UCPFormatter
.level = WARNING
oracle.jdbc.internal.replay.level = FINEST
oracle.ucp.jdbc.PoolDataSourceImpl.level = FINE
oracle.ucp.jdbc.oracle.level = FINEST
```

Oracle RDBMSサーバーでのトレースは、init.oraファイルを介し、次のように指定して有効化できます。



```
alter system set event='10602 trace name context forever, level
28:trace[progint_appcont_rdbms]:10702 trace name context forever, level 16: 41440 trace name
context forever' scope = spfile ;
```

## その他の資料

### アプリケーション・コンティニューイティのOTNホームページ

<https://www.oracle.com/jp/database/technologies/high-availability/app-continuity.html>

### FANホワイト・ペーパー

<http://www.oracle.com/technetwork/database/options/clustering/applicationcontinuity/learnmore/fastapplicationnotification12c-2538999.pdf>

### アプリケーション・コンティニューイティ・ホワイト・ペーパー

<http://www.oracle.com/technetwork/database/options/clustering/application-continuity-wp-12c-1966213.pdf>

### アプリケーション・コンティニューイティOOW 2017

<http://www.oracle.com/technetwork/database/options/clustering/applicationcontinuity/learnmore/hidingunplannedoutages-2872659.pdf>

### クライアント・フェイルオーバー

<http://www.oracle.com/technetwork/database/options/clustering/applicationcontinuity/learnmore/client-failover-brief-2430007.pdf>





Oracle Corporation, World Headquarters  
500 Oracle Parkway  
Redwood Shores, CA 94065, USA

海外からのお問い合わせ窓口  
電話：+1.650.506.7000  
ファクシミリ：+1.650.506.7200

#### CONNECT WITH US



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

## Integrated Cloud Applications & Platform Services

Copyright © 2018, Oracle and/or its affiliates. All rights reserved. 本文書は情報提供のみを目的として提供されており、ここに記載されている内容は予告なく変更されることがあります。本文書は、その内容に誤りがないことを保証するものではなく、また、口頭による明示的保証や法律による黙示的保証を含め、商品性ないし特定目的適合性に関する黙示的保証および条件などのいかなる保証および条件も提供するものではありません。オラクルは本文書に関するいかなる法的責任も明確に否認し、本文書によって直接的または間接的に確立される契約義務はないものとします。本文書はオラクルの書面による許可を前もって得ることなく、いかなる目的のためにも、電子または印刷を含むいかなる形式や手段によっても再作成または送信することはできません。

OracleおよびJavaはOracleおよびその子会社、関連会社の登録商標です。その他の名称はそれぞれの会社の商標です。

IntelおよびIntel XeonはIntel Corporationの商標または登録商標です。すべてのSPARC商標はライセンスに基づいて使用されるSPARC International, Inc.の商標または登録商標です。AMD、Opteron、AMDロゴおよびAMD Opteronロゴは、Advanced Micro Devicesの商標または登録商標です。UNIXは、The Open Groupの登録商標です。0520

ホワイト・ペーパー・タイトル  
2020年5月  
著者：Troy Anthony  
共著者：Carol Colrain、Carmen Frank



Oracle is committed to developing practices and products that help protect the environment