HITACHI
Inspire the Next

Oracle Database 11g × Hitachi Storage Solutions のベストプラクティス

日立ディスクアレイ ボリュームレプリケーション機能による Oracle Real Application Testing を使った システム変更前テストの効率化

株式会社 日立製作所
ソフトウェア事業部
RAID システム事業部

本文目次

付録目次

用語及び略号

DB	Database の略。データベース。
本番 DB	本番環境のデータベースの意味。業務に用いるデータベースを表す。
テスト DB	テスト環境のデータベースの意味。テストに用いるデータベースを表す。
RAC	Oracle Real Application Clusters の略で、Oracle Database のクラスタ構成。複数の DB サーバで 1 つのデータベースを共有する構成をとる。サーバを追加することで性能増強が可能。
RAT	Oracle Real Application Testing の略。Oracle Database 11g Enterprise Edition の Option 製品の一つで、11g から新しく追加された。主にシステム変更前テストに使用する機能を提供する。
PSR	Patch Set Release の略で、Oracle Database において定期的に配布される修正パッチ集。
USP V	日立ディスクアレイサブシステム「Hitachi Universal Storage Platform V」の略
AMS	日立ディスクアレイサブシステム「Hitachi Adaptable Modular Storage」の略

1　はじめに

Oracle Database 11g で新しく提供されたオプション製品である RAT は、本番 DB と同等の負荷をテスト DB で再現し、DB の構成変更が性能や運用へ及ぼす影響を事前に把握するために役立つ機能を提供します。RAT は、DB への負荷をキャプチャ／リプレイする Database Replay 機能と、本番 DB で問題のあった SQL をテスト DB でチューニングするために、本番 DB の統計情報と共に SQL 文をテスト DB へエクスポート／実行／比較レポーティングする SQL Performance Analyzer 機能から構成されます。

今回、株式会社日立製作所（以下、日立とする）は、日本オラクル株式会社（以下、日本オラクルとする）と共同で、RAT における Database Replay と、日立ディスクアレイが提供する２つの性質の異なるボリュームレプリケーション機能「ShadowImage」と「Copy-on-Write Snapshot」を連携させ、より効率的にシステム変更前テストを行うための手法を考案しました。本手法を使用することにより、Database Replay を使用したシステム変更前テストにおいて、テスト準備の工数を削減、及びテスト時間の短縮を実現し、全体としてテスト作業の大幅な効率化が可能になります。その理由は次の２点です。

- テスト DB を作成するためには、本番 DB から複製する必要があります。ShadowImage を使用することにより、本番 DB からテスト DB を非常に高速に複製することができ、また、本番 DB への更新をテスト DB へ反映する作業も簡単に実施することが可能です。これにより、テスト準備に必要な工数を削減します。
- テストでは、やり直しや繰り返しが発生します。Copy-on-Write Snapshot を使用することにより、少ない容量でテスト DB のバックアップが可能であり、テスト DB をテスト前の状態に切り戻す作業を素早く簡単に行うことができます。これにより、テストにかかる時間を削減できます。

本手法について実機を使用して検証を行い、Database Replay を日立ディスクアレイ上で使用するためのベストプラクティスとしてまとめました。本ホワイトペーパーでは、ベストプラクティスの詳細、実際のテスト手順、検証結果についてご紹介いたします。

2　データベースシステムのテストにおける課題とその解決法

顧客ニーズや事業環境が目まぐるしく変化する近年、企業の持つ IT システムが変化に対して迅速・柔軟に対応可能であることは、企業経営において非常に重要になりつつあります。IT システムへの変更要求は度重なり、それぞれに対して十分なテスト時間を設けることが難しくなる中で、テスト効率をいかに向上させるかが重要になっています。一方で、時間がとれず十分なテストが実施できない場合、本番環境で問題が発生してしまうリスクが高まります。本番環境でのトラブルが長期化すると、企業にとっては大きなダメージとなります。このように、企業は、システムトラブルが許されない中で、テストの効率化を求められるというジレンマを抱えています。以降、本稿ではシステムの中核を担う DB に焦点をあて、解説します。

本番 DB への変更実施においてトラブルを防止するためには、テスト DB を用意して変更実施の事前テストを行うことが必要です。さらに、テスト DB では、本番 DB と同様のトラフィックをシミュレートした上で事前テストを実施することが重要です。その理由は次の通りです。

本番 DB では、多くの SQL が多重実行され、ブロック競合やリソース獲得待ちが常に発生しています。このような DB に発生しているトラフィックの状況を**ワークロード**と呼びます。本番 DB のワークロードを再現してテストを行うことにより、実施する変更が本番 DB の性能や運用に与えるあらゆる影響を事前に把握することができます。従ってテストでは、本番 DB のワークロードをいかに正確に再現するかが重要になります。テスト DB で本番 DB のワークロードを再現するには、通常、大量のスクリプトを自作するか、専用のテストツールを使用します。しかし、これらの方法で本番ワークロードの細部に渡るまでを再現することは、通常非常に大きな工数が必要です。

2.1　Oracle11g Database Replay によるテスト精度向上とその運用上の課題

Oracle Database 11g では、本番 DB のワークロードをキャプチャし、テスト DB でそのワークロードをリプレイする Database Replay 機能が提供されました。Database Replay を使用することにより、本番 DB のワークロードをテスト DB で忠実に再現できます。

例えば、ある SQL 文のレスポンス向上のために、本番 DB のある表 A に索引 B の付与を検討している場合の Database Replay の使用方法は次の通りです。まず、本番 DB のワークロードをキャプチャします。次に、テスト DB の表 A に索引 B を付与して、キャプチャしたワークロードをリプレイします。最後に、キャプチャ時とリプレイ時の差分を比較し、性能や運用の観点で問題がないかを確認し、問題がなければ本番 DB に索引付与を実行するという流れになります。Database Replay を使用すると、本例の場合であれば、索引付与により、ある SQL 文は早くなるが他の SQL 文が遅くなる、といったような影響を事前に把握することができ、トラブル発生のリスクを低減することができます。

Database Replay を使用する場合には、テスト DB を、本番 DB での「**ワークロードのキャプチャ開始時点の状態**」と全く同一の状態にすることを強く推奨しています（表の有無や行数、ユーザの権限など細部に至るまで）。なぜなら、テスト DB でのリプレイ時には、本番 DB と同一の SQL 文が実行されます。従って、ユーザ権限や表中のデータに本番 DB との差異があると想定通りに SQL 文が実行できずエラーとなるケースが増加し、結果としてワークロードを忠実に再現できないためです。

2.1.1 テスト DB 作成における課題

キャプチャ開始時点の本番 DB と同一のテスト DB を作成するためには、キャプチャ開始時点のバックアップが必要となります。キャプチャ開始時点のバックアップをテスト環境にリストア・リカバリすることにより、本番 DB での「ワークロードのキャプチャ開始時点の状態」と同一の

テスト DB を作成できます。通常、Oracle　Recovery Manager（RMAN）や他のバックアップソフトを使用して、バックアップ／リストアによるテスト DB 作成作業を行います。しかし、DB のサイズが数 100GByte ～ 数 TByte というレベルの大規模 DB の場合、次のような問題があります。

1. バックアップ及びリストアに必要な時間が大きくなりすぎる。
2. 本番サーバに I/O 処理による負担が長時間発生し、業務への影響が大きい。
3. バックアップを格納する大量の一時領域を確保する必要がある。

このように、本番 DB の規模が大きくなると、テスト DB 複製時間、必要なストレージ容量、サーバリソースなど無駄が多くなり、日々の運用にテスト DB 複製という作業を組み込むことが困難であるという課題があります。これを「**テスト DB 作成における課題**」と呼びます。

2.1.2 テスト繰り返しにおける課題

Database Replay を使ったテストでは、採取したワークロードをパラメータ変更前後、索引有無、PSR 適用前後など環境の異なるテスト DB で複数回リプレイし、その違いをレポートするということが必要になります。このような場合、テスト DB においてワークロードをリプレイした後、再度リプレイ前の状態にテスト DB を戻す必要があります。DB をある時点に戻す機能としては、コールドバックアップを取得してそれをリストアする方法と、Oracle が提供する Flashback Database を使用して DB を巻き戻す方法があります。コールドバックアップを使う場合、バックアップを保存するために容量が 2 倍必要となり、また、リストアに大きな時間がかかります。従って大規模 DB では無駄が多くなります。次に、Flashback Database による切り戻しでは、本番環境でも Flashback Database を使用している場合には問題ありませんが、テスト DB でのみ使用する場合には次のような問題があります。

1．Flashback Database 有効化により、専用プロセスが複数起動し大量のログが出力される。
2．1.によりテスト DB で起動するプロセスや発行される I/O の競合状況に差異が発生。

これによりテスト DB は、本番 DB を忠実に再現した環境ではなくなります。したがって、テスト DB でワークロードリプレイ時に性能劣化や DB エラーなどの問題が発生した場合に、その原因の特定が困難になります。これらの問題を「**テスト繰り返しにおける課題**」と呼びます。

2.2 Database Replay と日立ディスクアレイの連携による課題解決

これら 2 つのの課題解決のために、日立は、Database Replay と日立ディスクアレイが提供するソリューションとを連携させることを提案します。

2.2.1 テスト DB 作成における課題の解決

「テスト DB 作成における課題」に対しては、筐体内レプリカ機能「ShadowImage」を使用することで解決が可能です。ShadowImage を使用することにより、DB の規模にかかわらず、本番 DB のレプリカを高速に複製することができます。これにより、本番 DB の業務に影響を与えることなく、テスト DB を作成することができます。また、ShadowImage には前回との差分を記憶して最小限のコストでレプリカを再形成する機能があるため、本番 DB に与えられた更新差分をテスト DB に反映するという運用に適しています。さらに、DB 複製の手順も非常にシンプルですので、全体としてテスト DB の準備工数の削減につながります。詳細は 4.3 節を参照ください。

2.2.2 テスト繰り返しにおける課題の解決

次に、「テスト繰り返しにおける課題」に対しては、筐体内スナップショット機能「Copy-on-Write Snapshot」の使用を提案します。Copy-on-Write Snapshot を使用すると DB を事前に決めたリカバリポイントに高速に戻すことができます。Copy-on-Write Snapshot の使用には、DB の変更量に準ずる量の空き領域が必要になりますが、フルバックアップを取得することに比べれば大幅に少な

い容量で済みます。さらに、ストレージで実現する機能であるため、本番 DB と異なるプロセスが起動したり、サーバに余計な I/O が発生したりすることもありません。このように Copy-on-Write Snapshot を使用することにより、テスト DB をある一時点に戻して複数回のテストを行うというテスト方法が簡単になり、テスト工数の削減に繋がります。詳細は 4.4 節を参照ください。

以上のことから、Database Replay と ShadowImage、及び Copy-on-Write Snapshot の連携により、本番ワークロードをテスト DB で再現してシステム変更リスクの低減を実現するという Oracle Database 11g の新しいテスト方式を、シンプルかつ効率的に実施可能となります。

本ホワイトペーパーではこれらの手法を、Database Replay を日立ディスクアレイ上で使用する場合のベストプラクティスと呼び、以降で詳細について説明します。3 章では、「Database Replay」、及び日立ディスクアレイが提供する「ShadowImage」、「Copy-on-Write Snapshot」の機能について説明します。4 章では、本ベストプラクティスの有効性や活用法について説明します。実機での検証結果は、付録に記載がありますのでご参照ください。

3　製品紹介

本節では、Oracle Database 11g Database Replay と、日立ディスクアレイが提供するボリュームレプリケーション機能「ShadowImage」、及び「Copy-on-Write Snapshot」について説明します。

3.1　Oracle Database 11g Database Replay

Oracle Database 11g Database Replay は、図 1 のように本番 DB のワークロードをキャプチャし、テスト DB でそのワークロードをリプレイする機能です。DBR は、Oracle Database Enterprise Edition のオプション製品である Real Application Testing の 1 コンポーネントです。

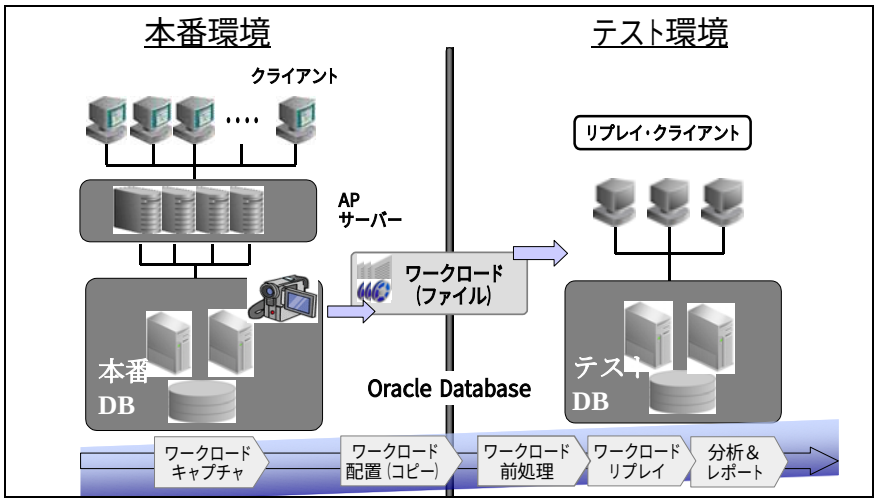


図 1　Database Replay 概要図

3.1.1 Database Replay のメリット

Database Replay を使用してテストを行うメリットは、次の通りです。

1. テスト準備工数の削減
   テスト DB で本番 DB のワークロードを再現するためには、アプリケーションの内容を把握したエンジニアによって、多数のスクリプトを作成する必要があります。また、サードベンダ製ツールを利用する場合も、アプリケーションの内容とツールの使用法の両方を理解したエンジニアによる事前準備が必要です。本番ワークロードは日々変化しているため、テストの度にスクリプトの作り直しやパラメータの再設定が必要です。Database Replay では、本番 DB の最新のキャプチャ・ファイルをテスト DB に転送してすぐにリプレイ可能です。手間なく最新のワークロードをリプレイできるため、従来の方法に比べてテスト準備工数が大幅に削減できます。

2. テスト期間の短縮
   これまでのテストでは、エンジニアが多数のスクリプトを個々に実行しながら、実行結果を確認していく必要がありました。Database Replay では、キャプチャデータ毎に一括してリプレイされ、実行結果の比較結果もレポートとして出力されます。これによりテスト工数およびテスト期間が削減されます。

3. より本番に近いワークロードでテストが実施可能
   従来の方法では、人手によるテストケースの作り込みが発生するため、本番業務を正確にシミュレートしたものではありませんでした。Database Replay では、本番 DB のワークロードをそのままキャプチャし、これをテスト DB でリプレイできます。これにより、システム変

更におけるトラブルのリスクを大幅に低減することができます。

3.1.2 Database Replay を使用したテストの流れ

Database Replay によるワークロードのリプレイには、主に次の５つのステップがあります。

Step 1. DB の複製
本番 DB のワークロードをテスト DB で完全に再現するためには、ワークロードのキャプチャを開始する直前の本番 DB と一致した内容のテスト DB が必要です。このため、本番 DB でバックアップを取得し、これを用いてテスト DB を作成します。
※ただし、ワークロードが DB の内容に依存しない場合、テスト DB は本番 DB と一致させるのは必須ではありません。

Step 2. ワークロードのキャプチャ
本番 DB でワークロードの取得を有効にし、ワークロードをキャプチャします。キャプチャしたワークロードは、キャプチャ・ファイルとしてファイルシステムに出力されます。ユーザ名、プログラム名、セッション ID などでキャプチャするワークロードを選択できます。

Step 3. キャプチャしたワークロードの転送および前準備
キャプチャ・ファイルを検証環境に転送し、再生するためにキャプチャ・ファイルを前処理します。

Step 4. ワークロードのリプレイ
キャプチャ・ファイルをテスト DB でリプレイします。再生では、リプレイクライアントの数や、トランザクション実行順序の保証、SQL 間隔の短縮などをパラメータで指定できます。

Step 5. 分析とレポート
取得時と再生時のワークロード実行結果のレポートを分析し、問題の有無を確認します。

3.1.3 リプレイ方法

Database Replay では様々なテストに柔軟に対応するために、リプレイ時のワークロードを調整できます。キャプチャしたワークロードに対して、リプレイ時に調整可能な項目について下記に説明します。

表 1　リプレイ時のワークロードの調整項目

#	調整項目	説明
1	同期/非同期モード	ワークロードのリプレイ時、キャプチャ時の Commit 順序を保証するかどうか調整します。 同期モード　　：Commit 順を保証します。 非同期モード　：Commit 順を保証しません。 同期モードはパッチ適用時などの動作確認に、非同期モードは、RAC 化や CPU 増設、仮想化などの並列実行性のテストに使用します。
2	接続待機時間の調節	リプレイを開始してからセッションを開始するまでの待機時間を調整します。指定はキャプチャ時の待機時間に対し、百分率で指定します。接続の待機時間を調節する事によって、ワークロードの高低を調節できます。
3	処理待機時間の調節	処理(ユーザコール)と処理の間の待機時間を調整します。指定はキャプチャ時の待機時間に対し、百分率で指定します。処理間の待機時間を調節する事によって、ワークロードを高く、または低く調節できます。
4	接続先の変更	キャプチャしたワークロードの接続文字列を、リプレイ時に変更できます。Single 環境でキャプチャしたワークロードを、RAC 環境でリプレイする場合や、特定アプリケーションのワークロードをサービスに結び付けてリプレイする場合に使用します。

3.1.4 リプレイレポート

Database Replay では、リプレイの結果をリプレイレポートとして出力する機能を備えています。リプレイレポートでは、実行時間や DB 時間、エラーの発生した SQL など、キャプチャ時とリプレイ時の実行結果を比較できます。また、1 回目のリプレイと 2 回目のリプレイを比較しレポートすることも可能です。リプレイレポートを分析することで、テストの対象となったシステム変更において問題が発生しているかどうかを確認することができます。リプレイレポートの詳細は、付録 C を確認ください。

3.2 Hitachi Storage Solutions

本節では、日立が提供するディスクアレイシステムのうち、本ホワイトペーパーで取り扱う、Hitachi Universal Platform V（USP V）、及び Hitachi Adaptable Modular Storage 2000（AMS2000）について紹介します。また、USP V と AMS2000 で提供される二つのボリュームレプリケーション機能、「ShadowImage」と「Copy-on-Write Snapshot」について簡単に説明します。

3.2.1 Hitachi Universal Storage Platform V

日立ディスクアレイサブシステム「Hitachi Universal Storage Platform V」は、Hitachi Universal Storage Platform(USP)の性能向上、スケーラビリティ向上を実現しただけではなく、エンタープライズクラスにおける仮想化機能をさらに進化させ，ボリューム容量の仮想化をも実現した世界初のディスクサブシステム製品です。中規模の成長企業から大規模なグローバル企業に至るまでのあらゆるお客様に，業界最高レベルの性能と拡張性を備えたストレージソリューションのメリットを実感していただけるようになりました。ディスクサブシステム構成は，ディスク制御部が搭載される基本筐体に、最大 128 台までディスクドライブを搭載することができ、さらにディスクフレーム筐体を最大 4 台接続することで，最大 1152 台のディスクドライブを搭載する構成をとることができます。

3.2.2 Hitachi Adaptable Modular Storage 2000

日立ディスクアレイサブシステム「Hitachi Adaptable Modular Storage 2000」は、従来のミッドレンジストレージでは困難であったサブシステム自律負荷バランスや、ホスト構成に影響されないオンラインサブシステム保守などの高機能を実現。お客様の導入負荷・運用負荷を大幅に軽減します。
さらに、SAS ディスクドライブと SATA ディスクドライブの同一筐体内混在を実現することでデータのライフサイクル管理の利便性を向上させたほか、ストレージ管理ソフトウェアとの連携によるユーザビリティをよりいっそう向上し、運用コストの大幅な削減と、効率的なデータ活用環境を実現しています。

3.2.3 筐体内ボリュームレプリケーション機能「ShadowImage」

ShadowImage は、ストレージの筐体内にサーバを介さずにボリュームの複製（ペア）を作成する機能を提供します。ShadowImage により作成されたあるボリュームの複製ボリューム（ペアボリューム）を利用することで、オンライン業務を継続しながら、バックアップの取得やバッチ業務の実行等を並列実行する事が出来ます。ShadowImage には、以下の特長があります。

1. 差分ビットマップを使用しているため高速なバックアップおよびリストアを実施できます。
2. 物理的に異なる RAID グループにボリュームを複製することにより、複製されたボリュームへのディスク負荷は本番機側と区別されます。これにより複製ボリュームへ I/O を実行した場合でも、本番業務は影響を受けません。
3. At-time スプリット機能を使えば、ボリューム複製対象が複数 LU であっても全てが単一時点断面（a single point in time）で複製されます。複数の LU に跨る業務であっても、データの順序性を保って切り離すことができます。

図 2 は、ShadowImage を使用したバックアップシステムの一例です[1]。

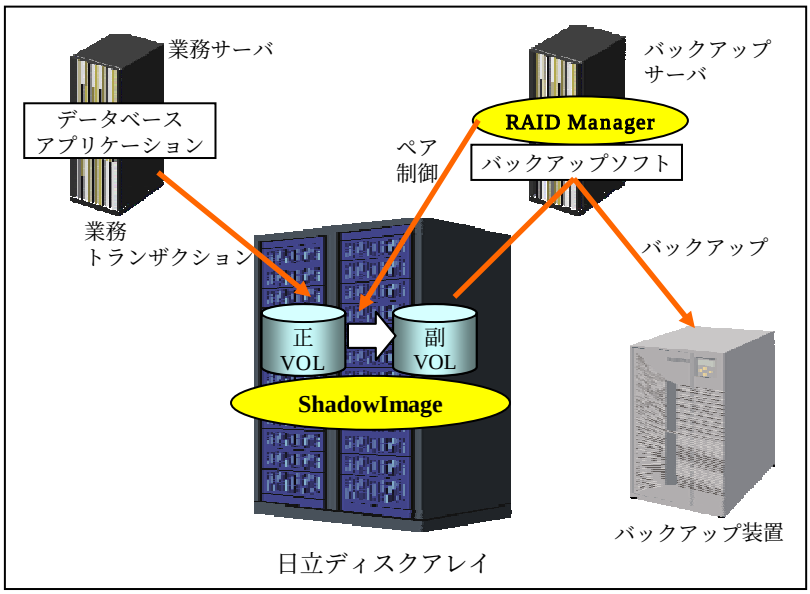


図 2　ShadowImage を使用したバックアップシステムの例

ShadowImage では、レプリケーションされたボリュームに対して、次のようにペアの状態を変更できます。

- ペア生成
  新規にあるボリュームペアボリュームを生成します。あるボリュームの全てのデータを対応するあるボリュームにコピーします。コピー元のボリュームを正ボリュームと呼び、コピー先のボリュームを副ボリュームと呼びます。

- ペア分割
  生成されたペアを分割し、サスペンド状態とします。サスペンド状態では、正ボリュームおよび副ボリュームに対する Write I/O をそれぞれ差分ビットマップにより管理します。ペアを初期状態にすることをペア解除といいます。

[1] RAID Manager とは、ShadowImage のペアを制御するためのソフトウェアです。

□ ペア再同期
ペア分割によりサスペンド状態となったペアについて、あるボリュームのデータと対応するボリュームのデータを同期します。差分管理されたビットマップに基づいて、デフォルトの指定では正ボリュームから副ボリュームに対し、データを反映します。“ -restore ”オプションを指定した場合はペアリストアといい、差分管理されたビットマップに基づいて、副ボリュームから正ボリュームに対し、データを反映します。

3.2.4 筐体内スナップショット機能「Copy-on-Write Snapshot」

Copy-on-Write Snapshot も、ストレージの筐体内にサーバを介さずにボリュームの複製（ペア）を作成する機能を提供します。ShadowImage と異なり、ボリュームの物理的なコピーを作成するのではなく、論理的なコピー（スナップショット）を副ボリュームとして作成します。ShadowImage と同様に、Copy-on-Write Snapshot により作成されたあるボリュームの副正ボリューム（ペアボリューム）を利用する事で、オンライン業務を継続しながら、バックアップの取得等を並列実行する事が出来ます。Copy-on-Write Snapshot には、以下の特長があります。

1. 差分データのみを管理しているため、副ボリューム（複製されたボリューム）のディスク使用量が少なく済みます。
2. 1 つのボリュームから最大 32 個（AMS は 15 個）の副ボリューム（スナップショット）を作成可能です。
3. At-time スプリット機能を使えば、ボリューム複製対象が複数 LU であっても全てが単一時点断面（a single point in time）で複製されます。

図 3 は、Copy-on-Write Snapshot の仕組みを説明しています。

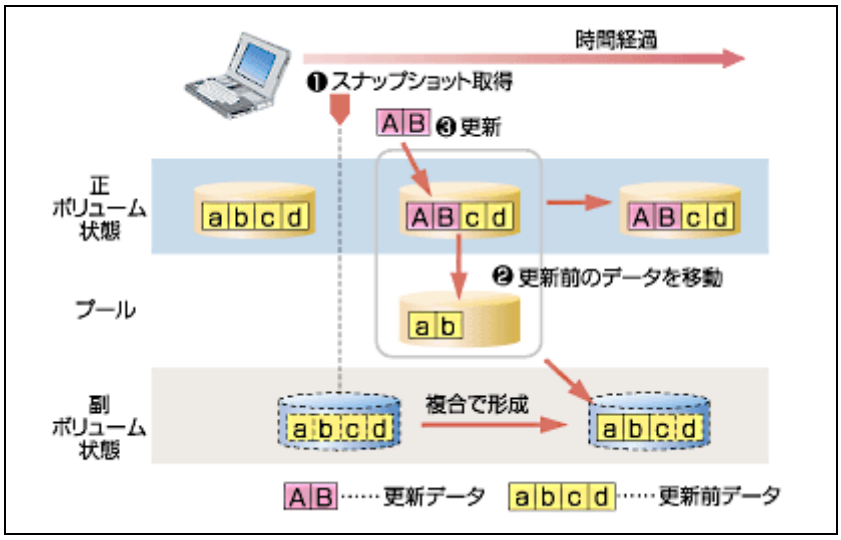


図 3　Copy-on-Write Snapshot の仕組み

4　ボリュームレプリケーション機能による Database Replay を使用したテストの効率化

本章では、DB のライフサイクルにおいて発生するシステム変更について説明し、システム変更に対して効率的にテストを実施するためのベストプラクティスについて解説します。

4.1　DB システムのライフサイクルと発生するシステム変更 - 頻発するシステム変更とテストの効率化の重要性 -

図 4 は、DB の運用において必要となる保守作業を DB のライフサイクルに合わせて表現しています。それぞれの保守作業について、例をあげて説明します。

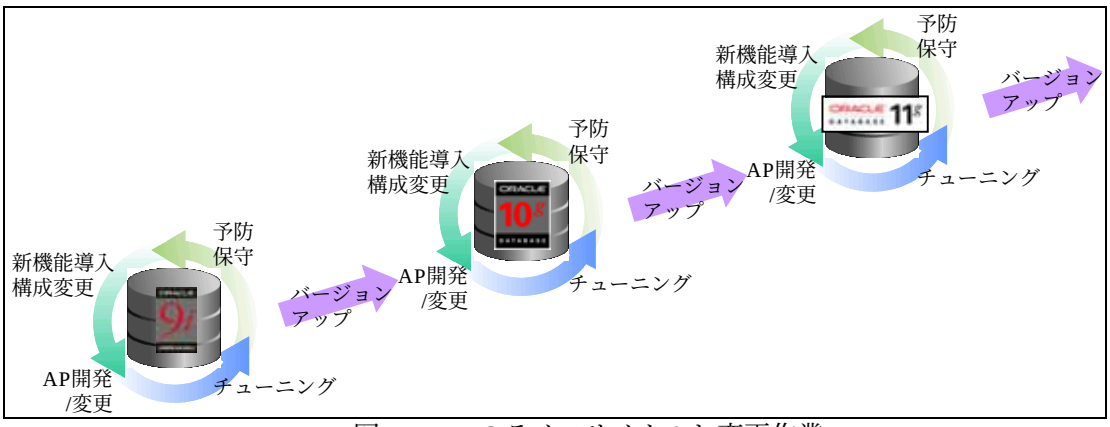


図 4　DB のライフサイクルと変更作業

□　AP 開発／変更
ユーザや関連部門からの要求でアプリケーションに機能を追加したり、変更したりする場合です。このようなアプリケーションへの追加／変更は、通常頻繁に行われています。SQL 文の変更だけで DB への直接変更が発生しない場合でも、既存運用への影響把握のために事前のシステムテストが重要です。

□　チューニング
登録ユーザの増加やデータ量の増加により性能問題が発生した場合、チューニングが必要になります。ユーザ数の変化やデータ量の増加は避けられないため、チューニングは DB の保守において定期的に実施する可能性が高い作業です。

□　予防保守
定期的に配布される PSR の適用やセキュリティパッチの適用がこれに相当します。PSR は 1 年程度の間隔でリリースされ、トラブル防止の観点から、最新の PSR の適用が強く推奨されています。また、セキュリティパッチは 4 半期に一度リリースされます。深刻な脆弱性への対策が施されている場合には、同様に適用が強く推奨されています。従って一定の期間間隔で DB へのこれらのパッチ適用が必要となる可能性があります。

□　新機能導入・構成変更
未導入の新機能を新たに導入する場合や、性能増強のための RAC のノード追加やメモリーの増設に伴う初期化パラメータの変更を行う場合です。

□　バージョンアップ
ハードウェア更改のタイミングや DB のサポート期間の終了などを契機にバージョンアップを実施します。

このように、Oracle Database の運用においては様々な DB への変更が頻繁に発生します。これに伴いシステムテストを頻繁に行うことは大きな負担となるため、テスト作業がシンプルで効率的であることは、大きな価値があります。

日立のディスクアレイが提供するボリュームレプリケーション機能と Database Replay を連携することにより、システム変更前のテストをシンプルかつ効率的に実施可能になります。これにより、本番 DB に対して変更を実施する前に、テスト DB で確認するというフローを DB の保守運用に組み込むことができ、頻繁に変更される DB システムを安定的に維持することができます。

4.2 Database Replay と日立ディスクアレイのボリュームレプリケーション機能の連携によるメリット

Database Replay では、本番 DB でキャプチャしたワークロードをテスト DB でそのままリプレイします。この性質上、Database Replay で効果的なテストを実施するためには、次の２点を考慮する必要があります。

1. テスト DB 作成における課題
   テスト DB を、本番 DB でのワークロードのキャプチャ開始時点の内容と一致させる必要があり、テスト DB 作成に工数がかかる。

2. テストの繰り返しにおける課題
   テストを複数回実施するためには、テスト DB は常にテスト開始前の状態に戻すことができる必要があり、DB 切り戻しの手順が煩雑で時間がかかる。

これらの課題は、「ShadowImage」と「Copy-on-Write Snapshot」を使用することにより解決できます。表 2 に、それぞれの課題に対して、「ShadowImage」と「Copy-on-Write Snapshot」がどのように改善し、どのような効果があるのかをまとめました。

表 2　ボリュームレプリケーション機能が Database Replay を使ったテストに与えるメリット

名称	課題内容	使用機能	改善内容	効果
テスト DB 作成における課題	・テスト DB 作成時間が大きい ・サーバリソースを使用する ・DB と同サイズの一時領域必要	Shadow Image	・テスト DB 作成が高速 ・サーバリソース不使用 ・一時領域が不要	テスト DB 作成の効率化／シンプル化
テスト繰返しにおける課題	Flashback Database 使用の場合 ・大量のログ出力で I/O 増加 ・専用のプロセスが起動する バックアップ／リストアの場合 ・DB と同サイズの一時領域必要	Copy -on-write Snapshot	・余計なプロセスの起動や I/O 増加がなく、本番 DB のワークロードを忠実に再現 ・必要な一時領域が少ない。	本番ワークロード再現の正確性向上・コスト削減

以下、それぞれの課題解決の考え方と、ボリュームレプリケーションを使用したベストプラクティスについて説明します。

4.3 テスト DB 作成における課題に対するベストプラクティス

Database Replay を効果的に使用するためには、テスト DB を本番 DB でのワークロードキャプチャ開始時点の状態と一致させることが必要なため、この作業に大きな時間と労力がかかります。具体的には、本番 DB のバックアップを取得してテスト環境にリストアし、これをリカバリしてテスト DB を作成するというステップが必要です。

本ベストプラクティスでは、このテスト DB 作成作業に ShadowImage を用います。ShadowImage を使用すると、本番 DB のレプリカを瞬時に作成可能であるため、テスト DB の作成作業が非常

に高速になります。

図 5 は、Oracle のバックアップユーティリティである Oracle Recovery Manager（RMAN）を使用してテスト DB を作成した場合と、ShadowImage を使用してテスト DB を作成した場合に要した時間とを比較したものです。DB のサイズは 26GByte と非常に小さい規模で実施した結果ですが、10 倍以上の差が出ていることがわかります（詳細につきましては、付録 B を参照ください）。さらに、RMAN を使用する場合、DB の規模に比例して作成時間が長くなりますが、ShadowImage の場合、DB の規模に依存せず高速です。従って数 100GByte から数 TByte の規模の DB を想定した場合にも、本ベストプラクティスが特に有効であると考えられます。

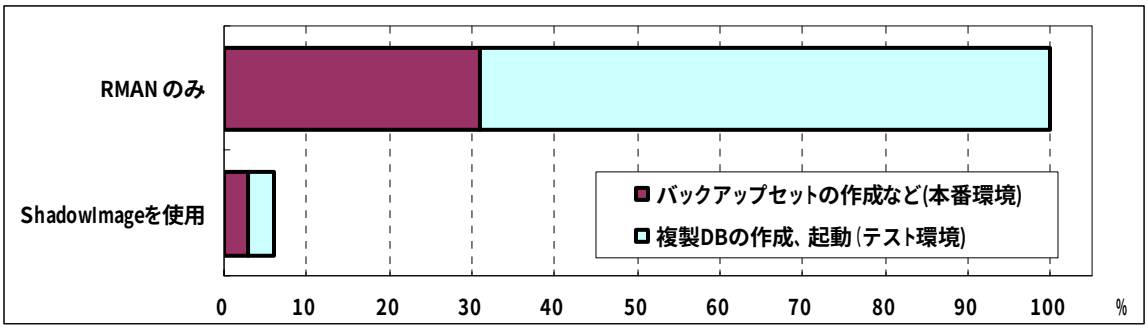


図 5　テスト DB 作成時間の比較[2]

また、RMAN を使用した場合、バックアップを一時的に配置する領域が必要です。この領域の容量も DB の規模に比例して大きくなります。ShadowImage を使用する場合には、副ボリュームをそのままテスト DB として起動するため一時領域は必要なく、テスト DB のための投資コストの削減にもつながります。

4.4　テスト繰り返しにおける課題に対するベストプラクティス

同一の本番ワークロードをテスト DB で複数回テストする要望があることは容易に想像できます。たとえば次のような場合です。

- 何らかの原因でテストに失敗し、テストをやり直す場合
- テスト DB のパラメータを様々な値に変えて、１つのワークロードでテストし、その違いをレポートして最適な値を探したい場合。
- 性能チューニングの手段が複数ある場合

テスト DB は、本番 DB のワークロードのキャプチャ開始時点と同一の状態にする必要があります。従って、テストのやり直しの度に、テスト DB を初期状態（以降、初期リストアポイントと呼びます）に切り戻す必要があります。DB を初期リストアポイントに戻す方法としては次の４つの方法があります。

1. RMAN によるフルバックアップをリストア
2. ShadowImage によるフルバックアップをリストア
3. Oracle Flashback Database を使用して切り戻す
4. Hitachi Copy-on-Write Snapshot を使用して切り戻す

1 つ目の RMAN を使用する場合、リストアに時間がかかることは前述のとおりです。

2 つ目の ShadowImage を使用する場合については、次の通りです。まず、本番 DB からテスト DB

[2] ShadowImage を使用した複製時間は、レプリカが形成されている状態を初期状態として測定しています。また測定結果は今回の検証環境での結果であり、全ての環境において同様となることを保証するものではありません。

に再度リストアを行っても、初期リストアポイントに戻すことができない場合があります。なぜなら本番 DB はテスト中にも業務を継続しており、これをリストアした場合、テスト DB は、テスト開始時点より時間的に進んだ状態になってしまいます。従って、ShadowImage を使用する場合、テスト DB を作成した時点で、テスト DB のレプリカを別途もう 1 面用意する必要があります。これは容量やコストの面で問題があります。

次に 3 つ目の Flashback Database を使用する場合についてです。本番 DB において Flashback Database を使用していない場合に、テスト DB で Flashback Database を有効にすることは次のデメリットがあります。Flashback Database は、ある 1 点ではなく、過去の任意の時点[3]に戻すことができるようになっているため、定期的に変更ログを出力する仕組みになっています。従って Flashback Database を有効にすると Flashback ログが出力されます。これにより I/O 量や CPU 使用率の状況が本番 DB とテスト DB で大きく異なってしまいます。また、Flashback Database を有効にすることにより、本番 DB には存在しない複数の専用プロセスが起動します。このように、テスト DB でのみ Flashback Database が有効化すると、本番 DB とテスト DB の環境に差異が発生するため、Database Replay によりレポートに問題が報告されても、それが Flashback を有効化したことによるものなのか、テストにより発見した問題なのかの区別が付きにくくなります。

本ベストプラクティスでは、4 つ目の Copy-on-Write Snapshot を使用します。テスト DB でテストを開始する直前に Copy-on-Write Snapshot を使用してスナップショットを取得します。この操作は瞬時に完了します。テストを実施し、テスト DB に更新が入ると、更新差分のみがスナップショットとして予め指定した場所にコピーされます。更新部分のコピーが行われるのは、1 回目の更新時のみで、同一領域を再度更新した場合には、コピーは発生しません（詳細動作は 3.3 節参照）。このため、ストレージ内の I/O や一時領域のデータ量が増加し続けることはありません。テスト DB を初期リストアポイントに戻す場合、取得したスナップショットをテスト DB にリストアするだけであり、手順がシンプルでリストア時間も短く、非常に効率的です。

Copy-on-Write Snapshot を使用するメリットをまとめます。

- Oracle DB 側にプロセス（Flashback Database による）が追加されない
- Oracle DB が出力する I/O 量に違いが発生しない
- Snapshot のリストアが早い。
- 必要な領域やログの量が ShadowImage や Flashback に比べて少ない。

図 6 は、Copy-on-Write Snapshot を有効化する前 20 分間と有効化した後 20 分間のスループットを比較した検証結果です。本検証では、アプリケーションとして TPC-C ベンチマークを使用しており、サーバの CPU を 100%使用し、ストレージのキャッシュを完全に使い切るまで負荷をかけました。結果として、Copy-on-Write Snapshot の有効化により、6%程度の性能劣化が確認されました。通常、このような高負荷状態で本番業務が行われることは稀なため、実際のテスト環境での性能への影響はこれより大幅に小さいと考えられます。また、Copy-on-Write Snapshot では、性能への影響は、時間の経過と共に小さくなります。実際の検証でも 20 分を経過後のトランザクション性能は、Copy-on-Write Snapshot 無効化時と同等でした。また参照処理では性能に影響がありません。以上のことから、Copy-on-Write Snapshot の有効化による性能への影響は、大きな問題とはならないと考えられます。

[3] 初期化パラメータで定めた一定時間の範囲に限ります。

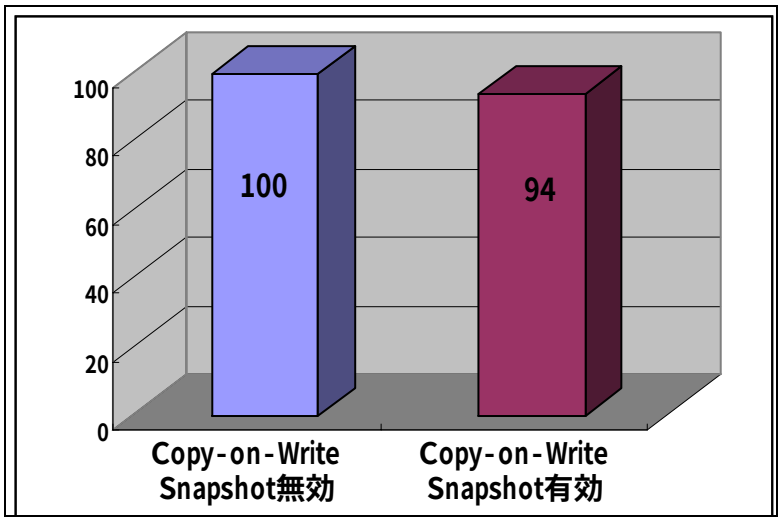


図 6　Copy-on-Write Snapshot によるトランザクション性能への影響

本ベストプラクティスでは、Database Replay と日立ディスクアレイの「ShadowImage」と「Copy-on-Write Snapshot」を連携させることを提案しました。これにより、テスト DB の作成、及び繰り返しのテストの手順がシンプルとなり、テスト準備やテストそのものにかかる時間も大きく削減できます。テストを効率化により、本番システム変更前にテスト DB で確認するというフローを運用に組み込むことができ、変化に対し柔軟で安定したシステムを実現することが可能になります。

Copy-on-writer Snapshot 使用時の Hitachi AMS 2000 での制限事項について

Hitachi Adaptor Modular Storage 2000（AMS2000）をご使用の場合、ShadowImage の副ボリュームに対して、ShadowImage のペアを形成したまま（ペア分割状態であっても）Copy-on-Write Snapshot を取得することができません（2009 年 1 月現在）。AMS2000 をご使用の場合には、ShadowImage の正副のペアを一度解除した上で、Copy-on-Write Snapshot を取得する必要があります。本番環境の複製を再度行う場合は、ペアを再作成する必要があります。

ペアの再作成が運用上難しい場合には、次の方法を検討してください。

1. テスト DB の作成に TrueCopy を使用する
   本番／テスト DB で AMS2000 を個別に用意し、テスト DB の作成は、筐体間レプリカ作成機能である TrueCopy を用いる（TrueCopy の副ボリュームには、ペアを形成したまま Copy-on-Write Snapshot を取得可能）。
2. Flashback Database を使用する。
   テストの繰り返しのために、Copy-on-Write Snapshot を使用せず、Flashback Database を使用する。

5　ベストプラクティスの構成例とテストの流れ

5.1　ベストプラクティスの構成例

図 7 に、本ベストプラクティスを実現する構成例を示し、それぞれの役割について表 3 で説明します。

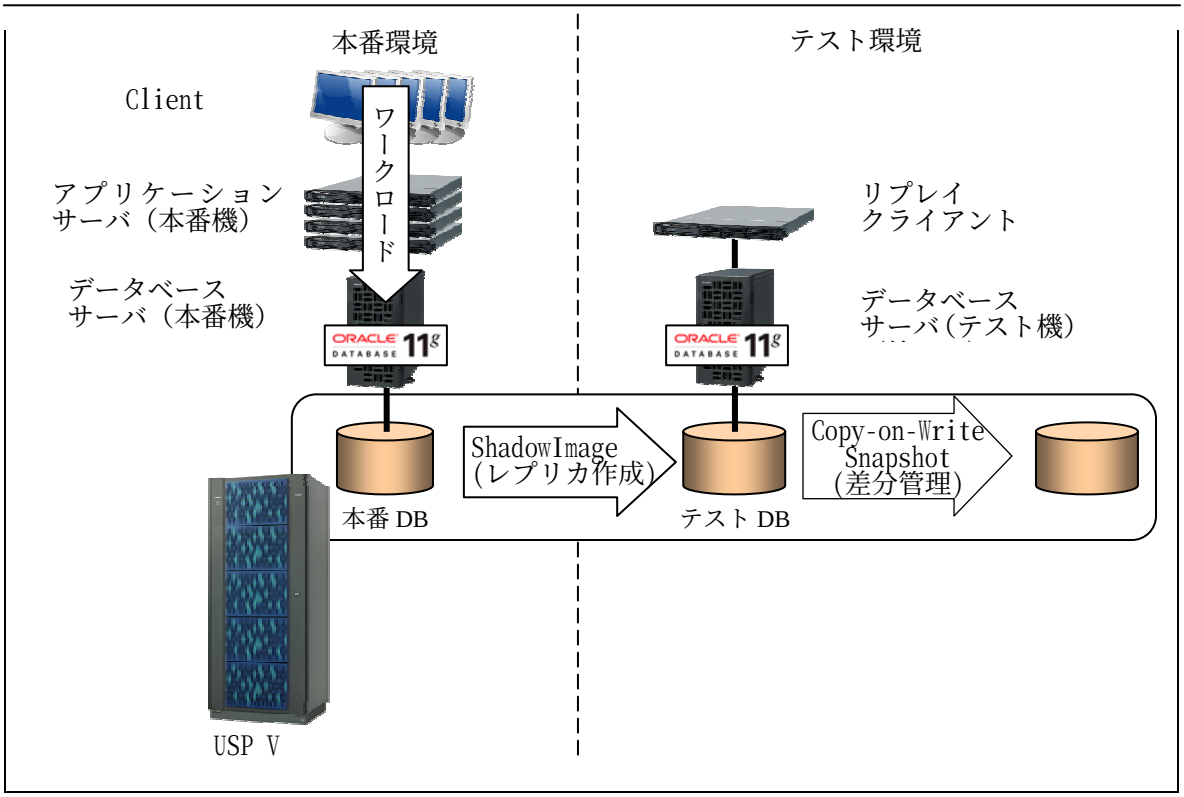


図 7　構成例

表 3　構成要素の役割

#	構成要素	役割
1	アプリケーションサーバ (本番環境)	本番業務で使用するアプリケーションサーバです。業務アプリケーションが動作し、DB サーバに対し、処理を要求します。
2	DB サーバ (本番環境)	本番業務で使用する DB サーバです。アプリケーションサーバから要求を処理します。Database Replay によるワークロードのキャプチャはこのサーバで行います。
3	リプレイクライアント (テスト環境)	キャプチャしたワークロードを再現するための、リプレイクライアントプロセスを起動するためのサーバです。Oracle Database Client のインストールが必要です。リプレイクライアントプロセスは、DB サーバ上で直接起動する事も可能です。その場合、本サーバは不要となります。
4	DB サーバ (テスト環境)	キャプチャしたワークロードをリプレイするための DB サーバです。
5	USP-V	DB を配置するディスクアレイ装置です。
6	ShadowImage	１台のディスクアレイ装置内で、論理ボリュームのレプリカを作成する、ディスクアレイ装置のオプション機能です。本番 DB からテスト DB を作成するために使用します。
7	Copy-on-Write Snapshot	Database Replay によるテストを繰り返し行う場合に、テスト DB の状態をワークロードリプレイ前に回復するために使用します。

5.2 テストの流れ

本節では、実際のテストの流れについて順を追って説明します（5.1 の構成例に従います）。

前提　ShadowImage の正副 Volume は一致した状態(Pair 状態)で開始するものとします。

Step ①　ShadowImage ペアの分割

テスト DB を、本番 DB のワークロードのキャプチャ開始時点と同一内容のレプリカとするために、適切なタイミングで ShadowImage のペアを一時的に切り離します。切り離しの瞬間からワークロードのキャプチャを行うまでに本番 DB に更新が発生しないよう注意してください[4]。

Step ②　本番 DB でのワークロードのキャプチャ

ShadowImage の切り離し後、直ちにワークロードのキャプチャを開始します。キャプチャしたワークロードは、キャプチャ・ファイルとしてファイルシステムに出力されます。

Step ③　キャプチャ・ファイルの転送

②で取得したキャプチャ・ファイルを検証環境のリプレイサーバに転送し、テスト DB でのリプレイに備えます。

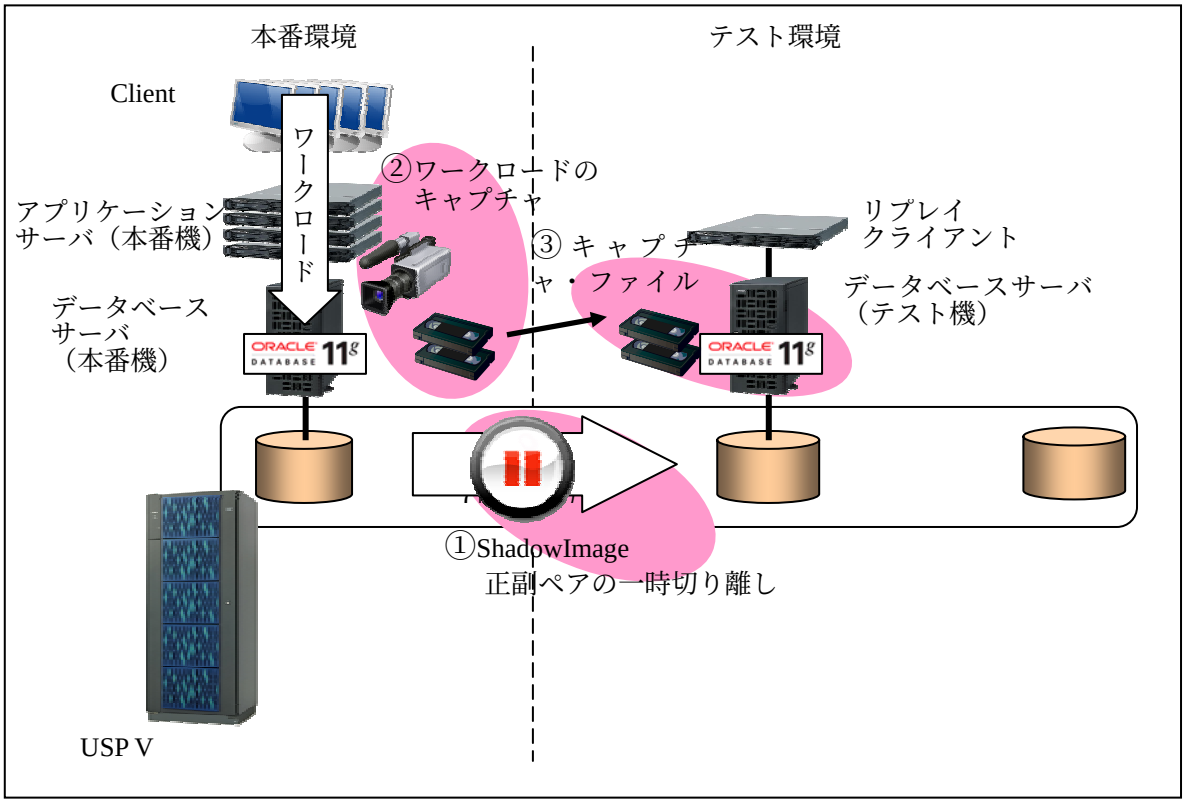


図 8　テストの流れ（本番環境にて実施する項目）

Step ④　Copy-on-Write Snapshot によるテスト DB のスナップショット作成

ワークロードのリプレイによるテストを繰り返し行うためには、リプレイ実行前のテスト DB を保存しておく必要があります。このため、Copy-on-Write Snapshot を使用してスナップショットを作成します。

Step ⑤　キャプチャ・ファイルの前処理の実行

Step③で転送したキャプチャ・ファイルを前処理し、リプレイ可能にします。前

[4] 更新が発生する場合、あるいは更新を停止することができない場合、テスト DB をキャプチャ開始時点に設定するために別途考慮が必要です。

処理したキャプチャ・ファイルは、転送やマウントにより、リプレイクライアントから参照できるようにします。

Step ⑥　テスト DB への変更の実施

テスト DB に対し、変更を実施します。
※ここでいう変更とは、PSR の適用や採用するチューニング指針などテストの目的となる変更を指します。

Step ⑦　ワークロードのリプレイ

テスト DB でワークロードをリプレイします。ワークロードの実行結果は、リプレイ後に生成されるリプレイレポートによって判断します。リプレイレポートでは、実行時間の相違、エラーの相違、取得したレコード件数の相違などが報告されます。このレポートを見て、システム変更による本番業務への影響を把握します。テストをやり直す場合や別の変更をテストする場合、スナップショットを使用してテスト DB をリプレイ実行前に切り戻し、⑥からの手順をやり直します。

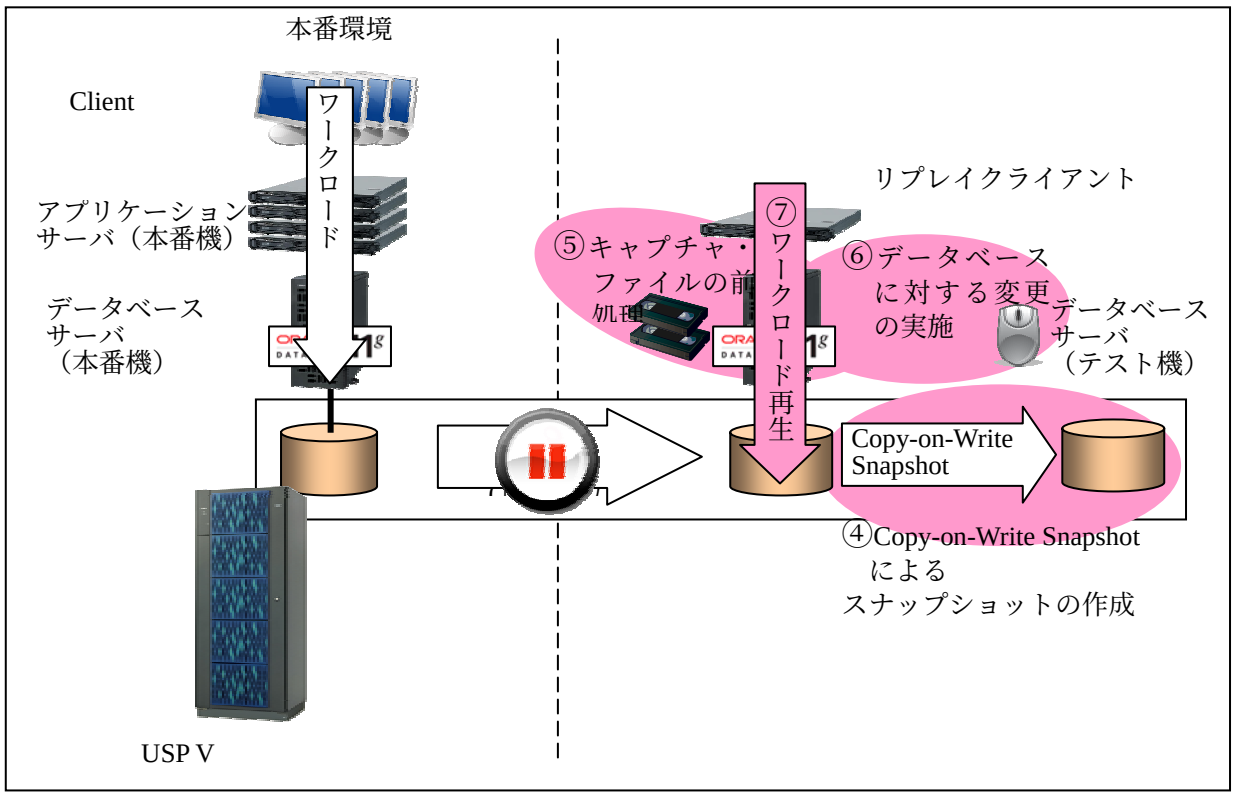


図 9　テストの流れ（テスト環境にて実施する項目）

6　Database Replay の適用例とベストプラクティスの有効性

4 章の冒頭で述べましたが、Oracle Database ではそのライフサイクルを通じて、様々なシステム変更が定期的に発生します。以降では、いくつかの変更例を挙げて、Database Replay を使用したテストの有効性を説明します。本ホワイトペーパーで説明したベストプラクティスは、ここで挙げる全てのケースにおいて、テスト作業の効率化を実現します。

6.1　適用例１　アプリケーション追加／変更時の影響確認

複数のアプリケーションが１つの DB 上で稼働している状況において、一部のアプリケーションを追加、あるいは変更する場合の適用例です。例えば、あるアプリケーションを変更することにより、特定の表や索引に大きな負荷がかかる場合があります。さらに、この表や索引を別の変更しないアプリケーションが参照していた場合には、変更しないアプリケーションのレスポンスタイムが急激に悪化する可能性があります。このようなリスクを事前に把握するためにも、本番 DB のワークロードを再現した環境下でのシステム変更前テストが必要です。

AP1、AP2、AP3 というアプリケーションが１つの DB 上で稼動している状況において、AP1 だけを変更する場合の、Database Replay を使用したテスト方法について説明します。Database Replay によるワークロードのキャプチャでは、DB に複数のアプリケーションのワークロードが存在する場合に、その中から必要なものを選択してキャプチャすることができます。具体的には図 10 のように、Database Replay で本番 DB のワークロードをキャプチャする際に、変更しないアプリケーション（AP2 と AP3）のワークロードのみをキャプチャします。これを検証環境においてリプレイします。このリプレイ状況下で、変更したアプリケーション AP1 のテストを同時に行う事で、AP2、AP3 のワークロード下においても変更した AP1 が問題なく動作するかどうかを確認することができます。

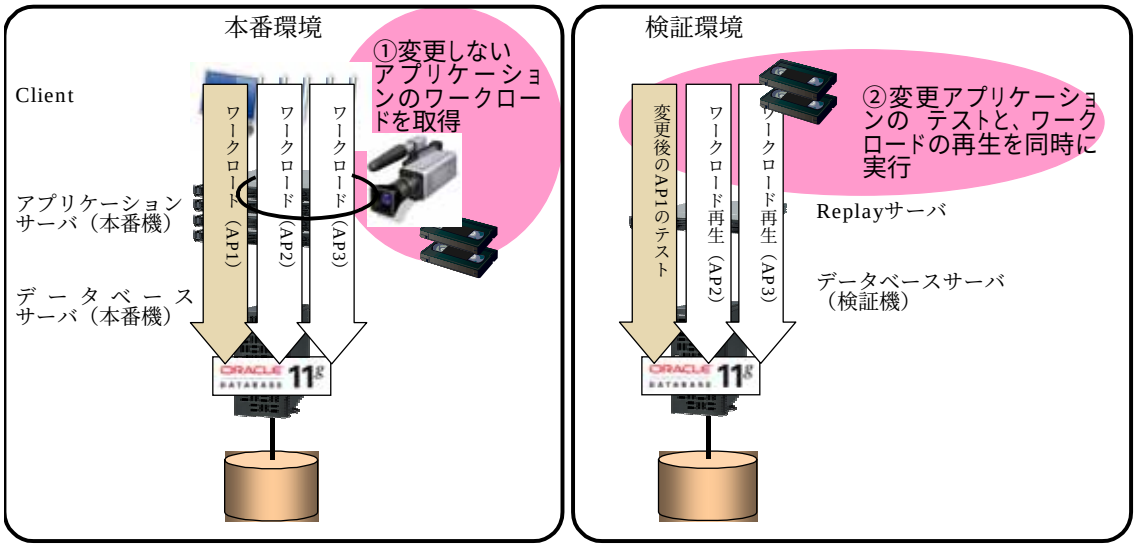


図 10　アプリケーション変更と Database Replay の適用

6.2　適用例２　チューニング成果の確認

パフォーマンスの改善のために、索引の付与や実行計画の変更を行う場合の適用例です。

Oracle Database 10g 以降提供されている、自動チューニング機能「SQL チューニングアドバイザ」や「SQL アクセスアドバイザ」を使用する場合を考えます。このようなツールを使うことにより、Oracle に関する高度な知識がなくてもチューニング方針を立てることができます。ただし、本番 DB にチューニングアドバイスをそのまま適用することは危険が伴います。このような場合、

Database Replay を使用すれば、ツールが提示したチューニング方針を採用することによりパフォーマンスが向上し、その他の問題が発生しないかを、テスト DB 上で実際に動かして確認することができます。実際の手順は、5.2 節の通りで、Step⑥で、索引の付与などチューニング指針の反映を行います。テストの結果、性能向上した場合にのみ、チューニング方針を本番 DB に適用するというテスト計画を立てることで、より確実な性能改善が実現できます。

また、チューニングでは何度も試行し、成果を確認しながら目標の性能に近づけていくという作業が必要になります。このような場合には、Copy-on-Write Snapshot を使用することにより、複数回のテストを行う場合のテスト DB の切り戻し作業で、大幅な効率向上が期待できます。

6.3 適用例３　PSR 適用後の確認やバージョンアップでの使用

PSR やセキュリティパッチ適用前のテストにおいても、Database Replay の使用が有効です。パッチを適用するにより、問題が解決されているか、あるいは性能やその他の既存運用に対して影響がないかどうかを確認することができます。

また、バージョンアップに対しても Database Replay を使用してテストを行うことが可能です。バージョンアップでは、Oracle の仕様変更を始めとして様々な変更が行われます。個々の変更に関するテストを Database Replay のリプレイレポートのみで確認することは困難ですが、バージョンアップ実行直前のトータルテストには適しています。Database Replay は Oracle Database 11g の新機能ですが、キャプチャに関しては、Oracle Database 9i Release2 や同 10g Release2 からも実施することが可能です。従って、過去のバージョンから 11g にバージョンアップする場合にも Database Replay を使用することができます。ただし、バージョンや前提パッチなどの条件があるため確認が必要です。

6.4 適用例４　DB の構成変更や新機能導入時の確認

次のような DB の構成変更時の確認にも Database Replay を活用したテストが可能です。

- RAC のノード追加による性能増強
- REDO ログのメンバ追加による信頼性の向上
- 初期化パラメータの変更による影響確認

また、新機能の導入により発生する次のような事象の影響を事前に確認することができます。

- Flashback Database 機能の導入による、Flashback ログの出力量や I/O 増加による性能への影響
- Partitioning Option 導入による性能への効果・影響
- Advanced Security Option による性能への影響
- Advanced Compression Option による性能への影響

7　まとめ

本ホワイトペーパーでは、Oracle Database 11g の新機能である Database Replay にフォーカスし、Database Replay と日立ディスクアレイのボリュームレプリケーション機能を連携させた場合のベストプラクティスについて説明しました。

本ベストプラクティスのポイントは次の 2 点です。

- ShadowImage でレプリカを形成することにより、本番 DB からテスト DB を高速に作成することができます。これにより、Database Replay を使用したテストを行うための準備作業の効率化を実現します。

- Copy-on-Write Snapshot を使用することにより、テスト DB を簡単にテスト前の状態に切り戻すことが可能です。これにより Database Replay を使ったテストにおいて、繰り返しのテストの効率化を実現します。

企業システムは、変化する顧客ニーズ、セキュリティの脅威、法規制に対し、迅速かつ柔軟に対応することが求められています。一方で変更に失敗し、サービスが停止することは許されません。本ホワイトペーパーで紹介したベストプラクティスは、企業の根幹を支える Oracle Database の変更作業を行う際に、そのシステムテストの効率化、さらにはシステムの安定稼働を支援する効果的なソリューションであると考えています。

なお、本ベストプラクティスを実機上に構成し、構築手順・テスト手順・レポートの確認方法など検証を行いました。検証結果については、付録に記載がありますのでご参照いただければ幸いです。

付録

本ホワイトペーパーで紹介するベストプラクティスを確実なものとしてご提供するために、実システムを想定した実機検証を行っています。検証では、Database Replay および ShadowImage を使用した実際のテスト実行手順、テスト DB 作成時間、及びワークロードのキャプチャに伴うパフォーマンスへの影響、ワークロード再現の忠実性などを確認しており、安心してご利用いただけます。
これらの検証結果についてご紹介します。

付録A.　ベストプラクティスによるテストの実行手順

A1.　　想定システム

想定するシステムは以下の構成となります。

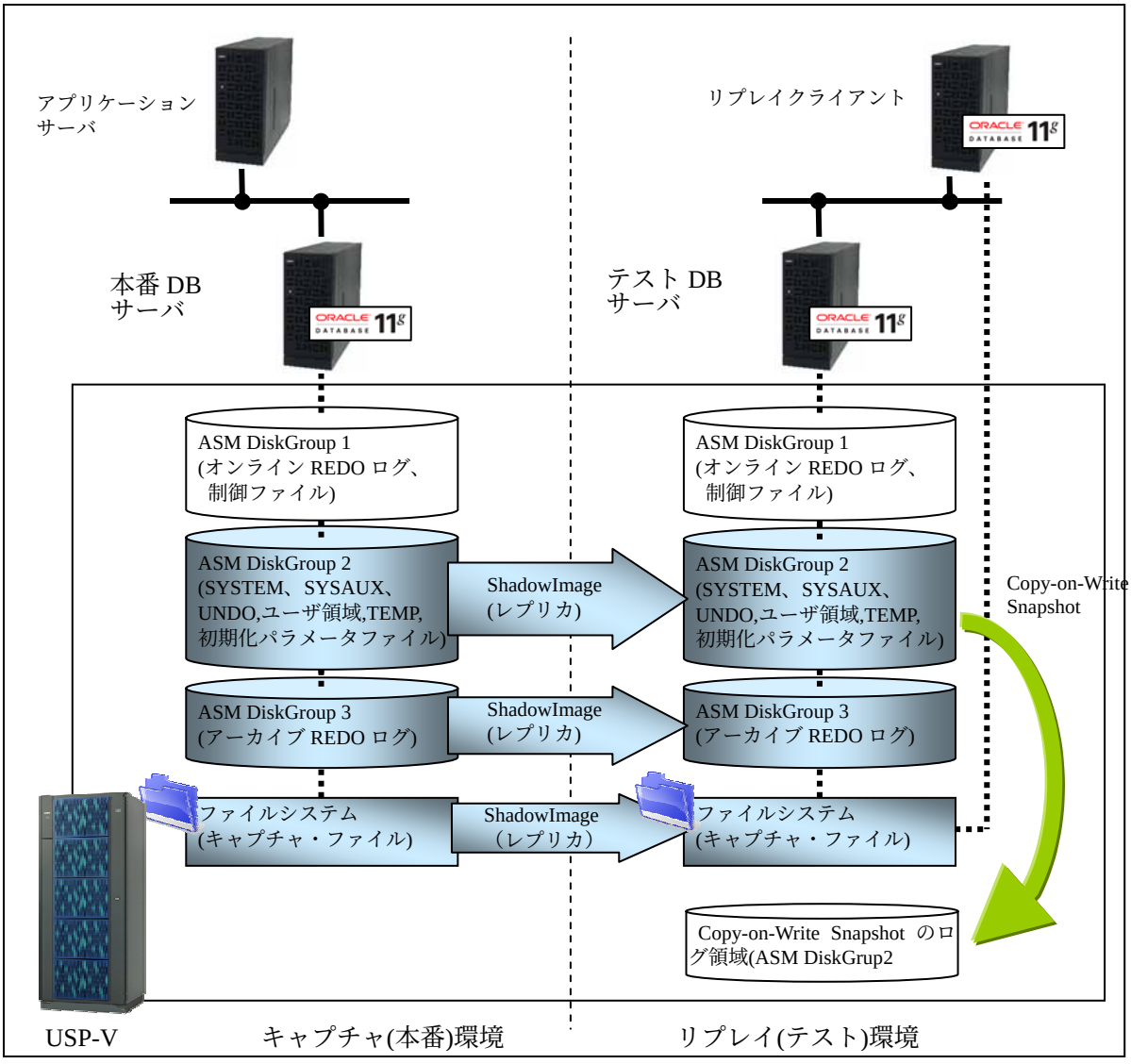


図 A-1　　想定システムの概要

想定システムの各構成要素について説明します。また、ShadowImage および Copy-on-Write Snapshot の設定について表 A-1 示します。

・本番アプリケーションサーバ
本番環境のアプリケーションサーバです。本番業務のアプリケーションが動作します。

・リプレイクライアント

ワークロードの再現において、クライアントとなるサーバです。
ただし、ワークロードの再現を DB サーバ上で直接行う事も可能です。その場合、このサーバは不要です。

・　本番 DB サーバ
本番環境の DB サーバです。本番アプリケーションサーバから実行されるトランザクションを処理します。ワークロードのキャプチャはこのサーバで行います。

・テスト DB サーバ
テスト環境の DB サーバです。リプレイクライアントで再生されたトランザクションを処理します。

・ASM DiskGroup 1
Online Redo、制御ファイルを配置する領域です。本番、テストの両方の環境に必要です。この領域は ShadowImage によるコピーを行いません。
この領域に配置するファイルのテスト環境での作成方法を、表 A-1 に示します。

表 A-1　テスト環境側 ASM DiskGroup 1 に配置するファイルの作成方法

#	ファイルの種類	作成方法
1	オンライン REDO ログファイル	ShadowImage による DB のオンラインコピーをテスト環境で起動するには、リカバリが必要となります。オンライン REDO ログファイルはリカバリの途中で自動的に作成されます。
2	制御ファイル	最新の制御ファイルのバックアップを取得し、バックアップから複製します。

・ASM DiskGroup 2
SYSTEM 表領域、SYSAUX 表領域、UNDO 表領域、ユーザデータの表領域、TEMP 表領域、初期化パラメータファイルを配置する領域です。
本番、テストの両方の環境に必要です。この領域は ShadowImage によるコピーを行います。
また、テスト環境でこの領域は、ワークロードリプレイ前に Copy-on-Write Snapshot の正ボリュームに設定されます。

・ASM DiskGroup 3
アーカイブ REDO ログファイルを配置する領域です。
本番、テストの両方の環境に必要です。この領域は ShadowImage によるコピーを行います。

・キャプチャ・ファイル用のファイルシステム
Database Replay でキャプチャしたワークロードのログを格納する領域です。
本番、テストの両方の環境に必要です。この領域は ShadowImage によるコピーを行います。
また、リプレイクライアントでは、この領域を読み取り専用でマウントし、参照します。

・Copy-on-Write Snapshot のログ領域(ASM DiskGroup 2 用)
ASM DiskGroup 2 用の Copy-on-Write Snapshot のログ領域です。テスト環境において、ワークロードのリプレイを繰り返し実行するために、Copy-on-Write Snapshot を使用して、ワークロード実行前の DB 状態を保存します。この領域は、ワークロードリプレイ前に Copy-on-Write Snapshot の副ボリュームに設定されます。ワークロード実行中は ASM DiskGroup2 に対する更新を復元するために、変更前情報を保存します。

表 A-2　ShadowImage および Copy-on-Write Snapshot の設定および運用手順

#	機能	正ボリュームに設定する領域	副ボリュームに設定する領域	運用手順
1	本番 DB →テスト DB 複製用 ShadowImage	本番用 ASM DiskGroup 2	テスト用 ASM DiskGroup 2	・初期設定時にペアを生成。正→副ペア同期後、ペア分割 ・ワークロードキャプチャ前に正→副ペア同期後、ペア分割
2		本番用 ASM DiskGroup 3	テスト用 ASM DiskGroup 3	
3	ワークロードリプレイ後のテスト DB 復元用 Copy-on-Write Snapshot	テスト用 ASM DiskGroup 2	Copy-on-Write Snapshot ログ領域 (ASM DiskGroup2 用)	・ワークロードリプレイ前にペア生成 ・ワークロードリプレイ後に副→正ペアリストア ・テスト終了後にペア解除

A2.　テストの実行手順

運用手順の概要は以下の Phase で構成されます。

Phase 0　環境構築およびテスト前の状態
Phase 1　テスト DB 作成の準備(本番環境)
Phase 2　本番ワークロードのキャプチャ(本番環境)
Phase 3　テスト DB の準備(テスト環境)
Phase 4　リプレイの準備(テスト環境)
Phase 5　リプレイの実行(テスト環境)
Phase 6　テスト DB の復旧作業(テスト環境)
Phase 7　テスト後の作業

各 Phase で行う手順は以下の通りです。

Phase 0　環境構築手順およびテスト前の状態

この Phase は、正確にはテスト手順ではなく、環境構築の Phase です。
環境構築のために、最初に一回だけ行います。

Step 0-1　Disk 割り当て、ShadowImage の設定

ストレージを操作して各サーバで使用する Disk を割り当てます。

Step 0-2　本番環境の構築

本番環境で、本番業務で使用する DB を構築します。詳細については割愛します。

Step 0-3　テスト DB サーバへの Oracle Database ソフトウェアのインストール

テスト DB サーバへ、Oracle Database のソフトウェアをインストールします。
本番環境と同じディレクトリへインストールする事を推奨します。

Step 0-4　テスト DB サーバでの、ASM インスタンスおよび ASM DiskGroup の作成

テスト DB サーバで、dbca コマンドを使用して ASM インスタンスを作成します。
また、ASM DiskGroup 1～3 を作成します。

Step 0-5　キャプチャ・ファイル格納ディレクトリの作成

本番 DB サーバで、キャプチャ・ファイル格納ディレクトリを作成します。テスト DB サーバおよびリプレイクライアントで使用するキャプチャ・ファイル格納ディレクトリについては、ShadowImage により本番環境の内容がコピーされるため、作成する必要はありません。

Step 0-6　テスト DB サーバで DB の作成

テスト DB サーバで、dbca を使用して DB を作成します。この作業を行う目的は、テスト DB を起動するために必要なローカルファイル(初期化パラメータファイルやパスワードファイル)を作成する事です。従って、作成する DB の DB 名、インスタンス名は本番 DB と同じ名前を指定し

ます。また、初期化パラメータも可能な限り本番 DB と同じ値を指定します。作成が終了したら DB インスタンスおよび ASM インスタンスを停止します。

Step 0-7　リプレイクライアントへの Oracle Database ソフトウェアのインストール
リプレイクライアントに Oracle Database のソフトウェアをインストールします。

Step 0-8　DB 領域の ShadowImage のペア再同期
ShadowImage のペアのうち、DB 領域(ASM DiskGroup 2～3)の Disk を同期(ペア生成)させます。同期後は、ShadowImage ペアを一時的に分割します。

以上が環境構築手順の概要となります。
Phase1 開始前における構成要素およびインスタンスの状態を表 A-3 に示します。

表 A-3　Phase1 実行前の状態

#	構成要素	状態		
		本番 DB サーバ	テスト用 DB サーバ	リプレイクライアント
1	ASM インスタンス	起動	停止	該当せず
2	DB インスタンス	起動	停止	該当せず
3	リスナー	起動	停止	該当せず
4	ASM DiskGroup 1	DiskGroup を構成し、REDO や制御ファイルを配置。	DiskGroup を構成し、REDO や制御ファイルを配置	該当せず
5	ASM DiskGroup 2	DiskGroup を構成し、データファイルを配置。ShadowImage でテスト環境とペアを構成。ペアは分割状態	ShadowImage で本番環境とアを構成。ペアは分割状態	該当せず
6	ASM DiskGroup 3	DiskGroup を構成し、アーカイブ Redo ログを配置。ShadowImage でテスト環境とペアを構成。ペアは分割状態	ShadowImage で本番環境とペアを構成。ペアは分割状態	該当せず
7	ファイルシステム(キャプチャ・ファイル)	ファイルシステムを構成。ShadowImage でテスト環境とペアを構成。ペアは分割状態	ShadowImage でテスト環境とペアを構成。ペアは分割状態。またファイルシステムとしてもアンマウント状態	アンマウント状態
8	Copy-on-Write Snapshot のログ領域 (ASM DiskGroup 2 用)	該当せず	設定されていない	該当せず

Phase 1　検証用 DB 作成の準備(本番環境)

この Phase では、ワークロード実行前の本番 DB から、テスト DB を作成するための準備を行います。この Phase のオペレーションは、本番環境で行います。

Step 1-1　DB 領域の ShadowImage ペア再同期
DB 領域(ASM DiskGroup2～3)の Disk について、ShadowImage のペアを再同期させます。ペアの再同期によって発生する I/O が、本番業務に影響を与える場合があります。ペアの再同期はシステムがもっとも低負荷である時間帯に行う必要があります。

Step 1-2　本番環境の ASM のリバランス停止
本番環境の ASM の初期化パラメータ ASM_POWER_LIMIT を 0 に設定し、ASM の Disk リバランスを停止します。

Step 1-3　オンラインバックアップモードへ移行
DB をオンラインバックアップモードへ変更します。

Step 1-4　DB 領域の ShadowImage ペア分割

ShadowImage のペアのうち、表領域(ASM DiskGroup 2)に使用する Disk を切り離します。

Step 1-5　オンラインバックアップモードの終了
Step 1-3 でオンラインバックアップモードにした DB を元に戻します。

Step 1-6　オンラインバックアップ中の REDO ログをアーカイブ化します
オンラインバックアップモード中に発生したオンライン REDO ログをアーカイブ REDO ログファイル化させます。

Step 1-7　制御ファイルのバックアップ取得
最新の制御ファイルのバックアップを RMAN で取得します。

Step 1-8　DB 領域の ShadowImage のペア分割
ShadowImage のペアのうち、アーカイブ REDO ログ(ASM DiskGroup 3)に使用する Disk を切り離します。

Step 1-9　本番環境の ASM のリバランスの再開
Step 1-2 で変更した ASM_POWER_LIMIT を元の値に戻します。

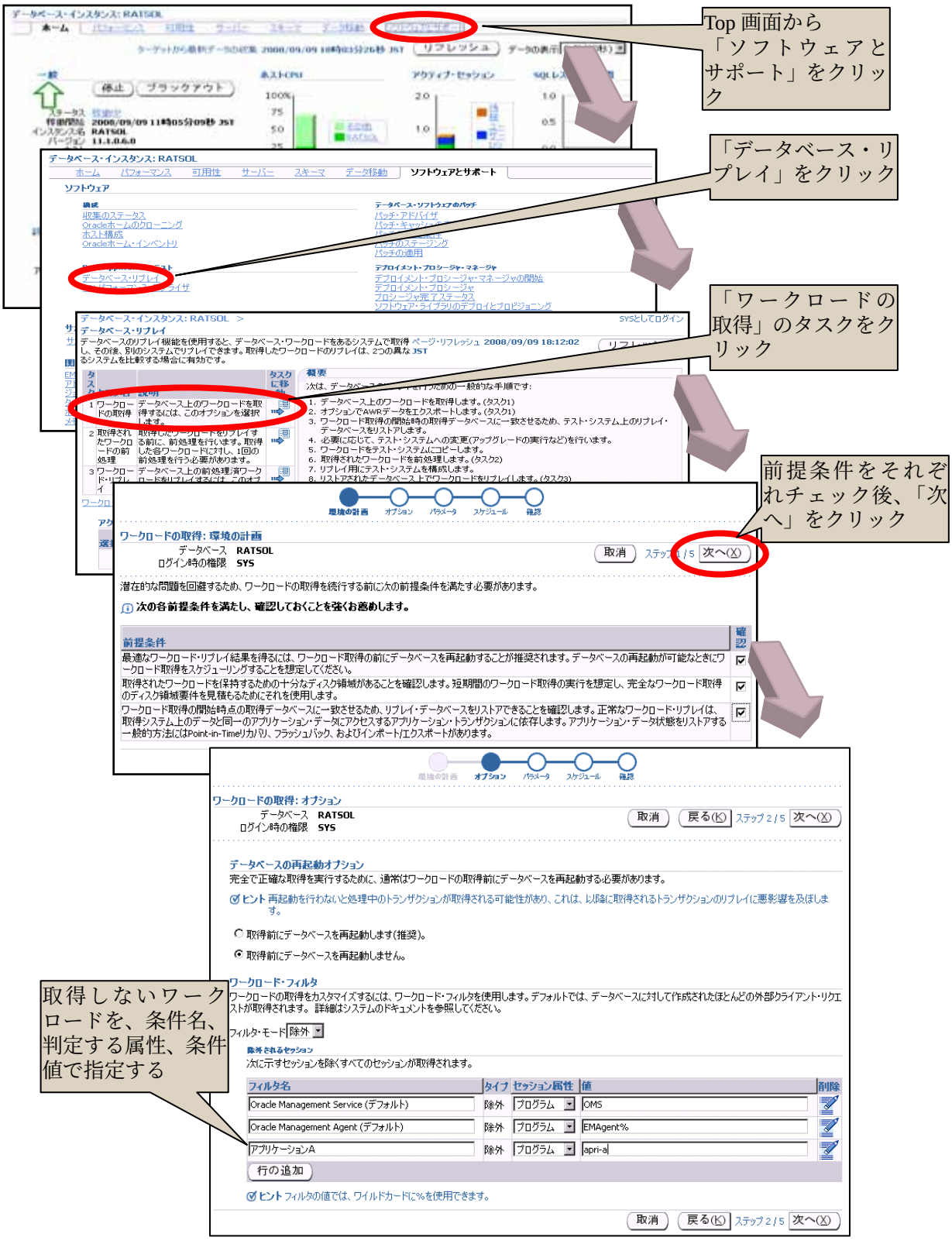
Step 1-10　バックアップセットのテスト環境への転送
Step 1-7 で取得したバックアップ制御ファイルを、テスト環境へ転送します。

Phase 2　本番ワークロードのキャプチャ(本番環境)

この Phase では、本番ワークロードのキャプチャを行います。また、ワークロードキャプチャによって生成されたキャプチャ・ファイルを、ShadowImage を使用してテスト環境に転送します。

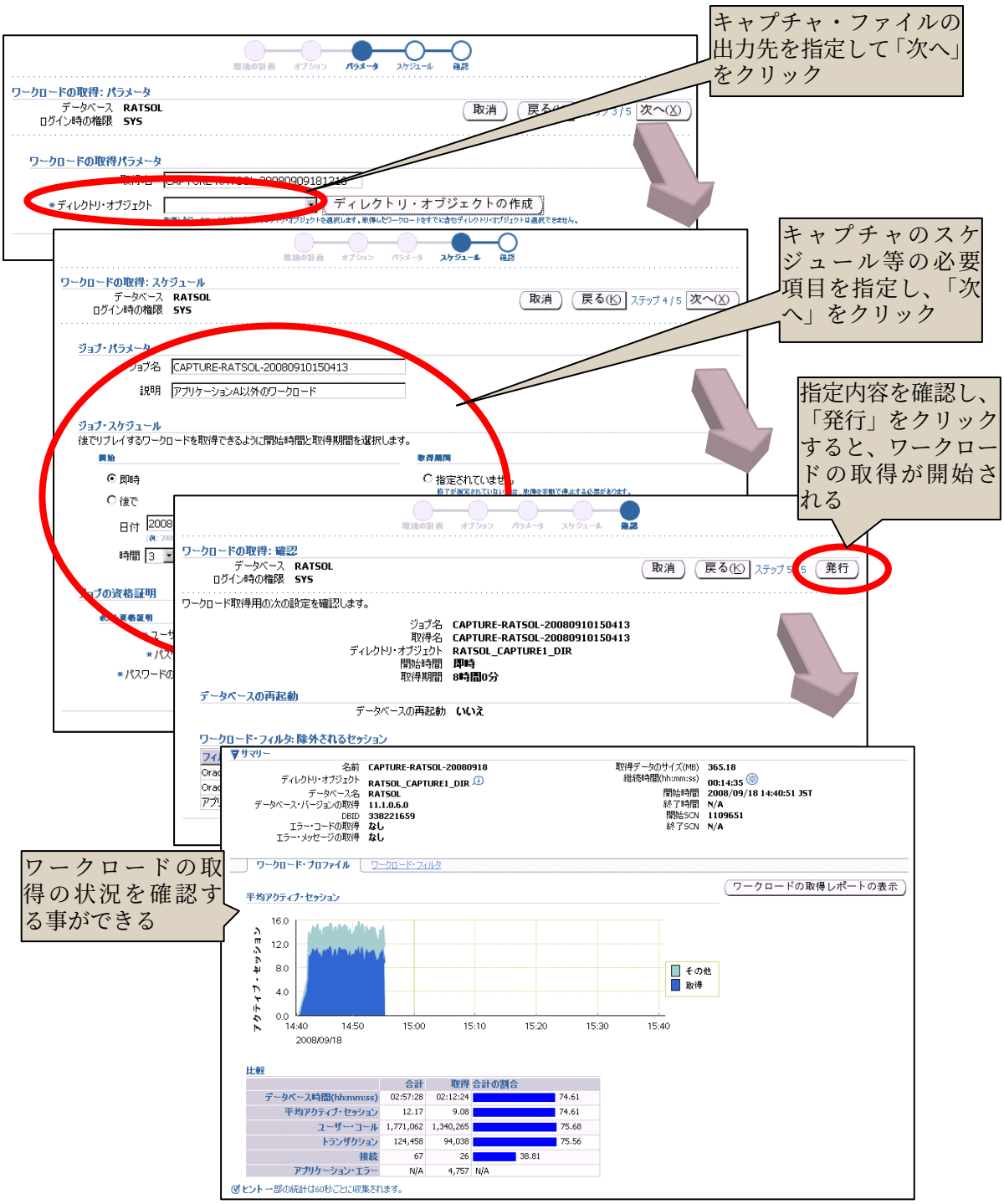
Step 2-1　ワークロードキャプチャのフィルタ設定

取得するワークロードをフィルタによって選択します。フィルタで設定できる条件には、プログラム名やユーザ名などがあります。フィルタの設定は Enterprise Manager から行います。



Step 2-2　　ワークロードのキャプチャ

ワークロードの取得を開始します。Enterprise Manager から行います。



Step 2-3　　ワークロードのキャプチャの終了
ワークロードの取得を終了します。Enterprise Manager から行います。
ワークロードの取得の終了は、スケジューラによっても可能です。
ワークロードの取得が終了したら、AWR データのエクスポートを行います。

Step 2-4　　キャプチャ時の AWR レポートのエクスポート
キャプチャ取得中の AWR レポートをエクスポートします。Enterprise Manager から行います。

Step 2-5　　キャプチャ・ファイル出力ディレクトリのアンマウント
本番 DB サーバで、キャプチャ・ファイル出力ディレクトリをアンマウントします。

Step 2-6　　ShadowImage のペア再同期によるキャプチャ・ファイルのコピー
キャプチャ・ファイルを配置する Disk について、ShadowImage のペア再同期し、テスト環境にキャプチャ・ファイルをコピーします。

Step 2-7　　キャプチャ・ファイル領域の ShadowImage ペア分割
Step 2-6 で同期した、キャプチャ・ファイル領域用 Disk について、ShadowImage のペアを再び一時的に分割します。

Step 2-8　　キャプチャ・ファイル出力ディレクトリのマウント
本番 DB サーバで、Step 2-5 でアンマウントしたディレクトリを再びマウントします。

Phase 3　　テスト DB の準備(テスト環境)

ShadowImage でオンラインコピーした DB 領域を使用し、テスト DB を起動します。ShadowImage でオンラインコピーした DB 領域を使用して DB を起動するためには、リカバリ作業が必要になります。

Step 3-1　　リスナーの起動
テスト環境のリスナーを起動します。

Step 3-2　　ASM インスタンスの起動
テスト環境の ASM インスタンスを起動します。

Step 3-3　制御ファイルのリストア
テスト環境の DB インスタンスを nomount モードで起動し、Step 1-7 で取得したバックアップセットから、制御ファイルをリストアします。

Step 3-4　DB のリカバリ
テスト環境の DB インスタンスを mount 状態にし、バックアップ中のアーカイブ REDO ログを使って DB のリカバリを行います。
リカバリにより、オンライン REDO ログファイル、TEMP 表領域が作成されます。

Step 3-5　DB のオープン
テスト環境の DB インスタンスを resetlogs オプション付きでオープンします。

Step 3-6　Enterprise Manager の再作成
テスト環境で emca コマンドを実行し、Enterprise Manager dbcontrol をリポジトリも含め再作成します。

Phase 4　　リプレイの準備(テスト環境)

ワークロードのリプレイの準備として、キャプチャ・ファイルに対する事前処理の実施と、Copy-on-Write Snapshot によるスナップショットの作成を実施します。Copy-on-Write Snapshot によるスナップショットの作成は、ワークロードのリプレイが何度でも行えるよう、ワークロードのリプレイで変更されたテスト DB を復旧させるために行います。

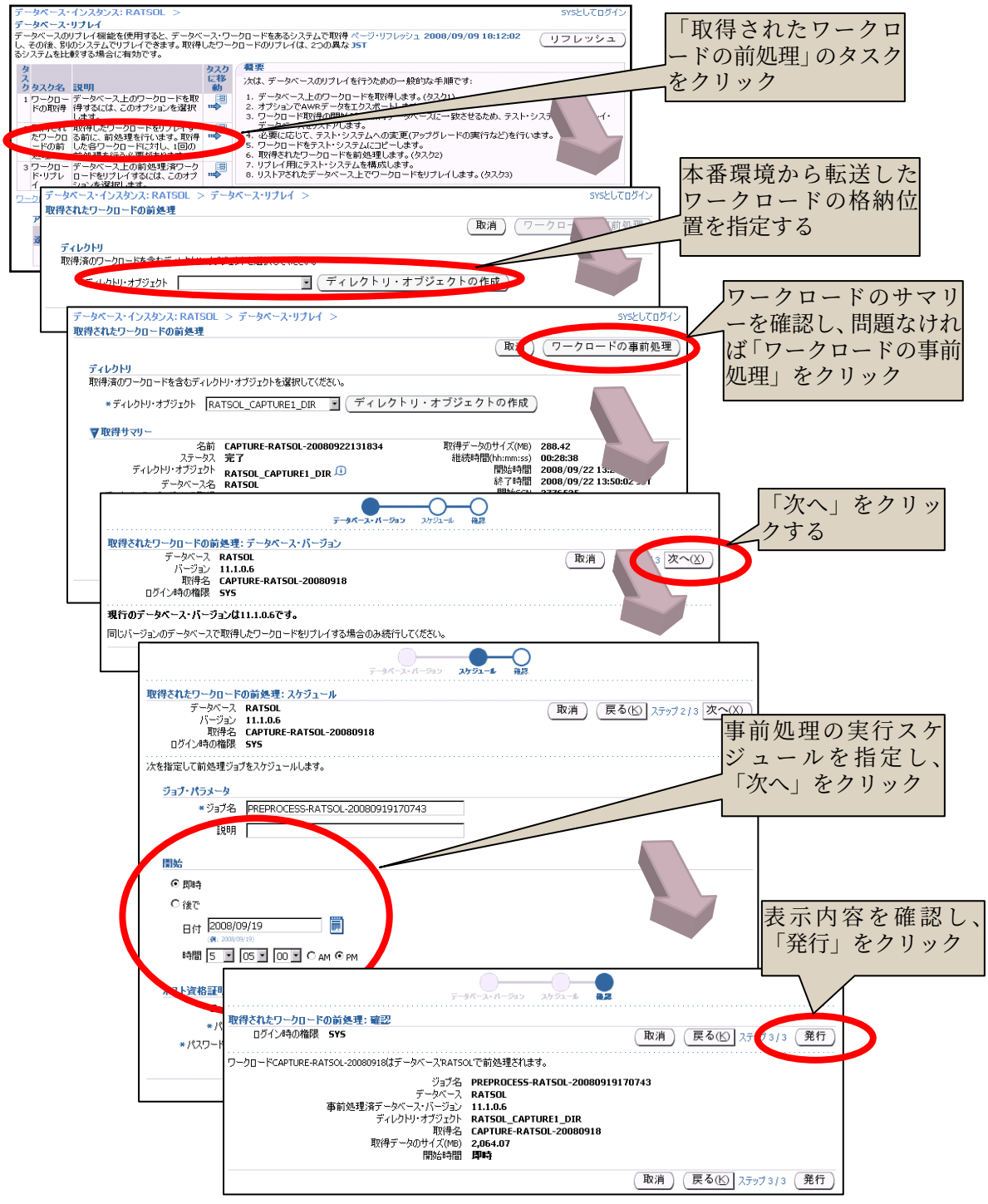
Step 4-1　　キャプチャ・ファイル格納ディレクトリのマウント

テストサーバでキャプチャ・ファイル格納ディレクトリをマウントします。また、リプレイクライアントで、キャプチャ・ファイル格納ディレクトリを読み取り専用モードでマウントします。

Step 4-2　　キャプチャ・ファイルの事前処理

ShadowImage で複製されたキャプチャ・ファイルを、再生可能にするために事前処理を実行します。

この作業はテスト DB で実施します。事前処理の実行は ENTERPRISE MANAGER から行います。



Step 4-3　　リプレイクライアント数の調査
キャプチャ・ファイルからリプレイに必要なプロセス数を見積もります。

Step 4-4　　テスト DB および ASM インスタンスの停止
テスト環境の DB インスタンスと ASM インスタンスを停止します。

Step 4-5　　スナップショットの作成
Copy-on-Write Snapshot を使用して、テスト DB のスナップショットを作成します。

Step 4-6　　ASM インスタンスの起動
テスト環境の ASM インスタンスを起動します。

Step 4-7　　リカバリ後の制御ファイルのバックアップ取得
テスト環境の DB インスタンスを mount 状態で起動し、リカバリ後の制御ファイルを RMAN でバックアップを取得します。

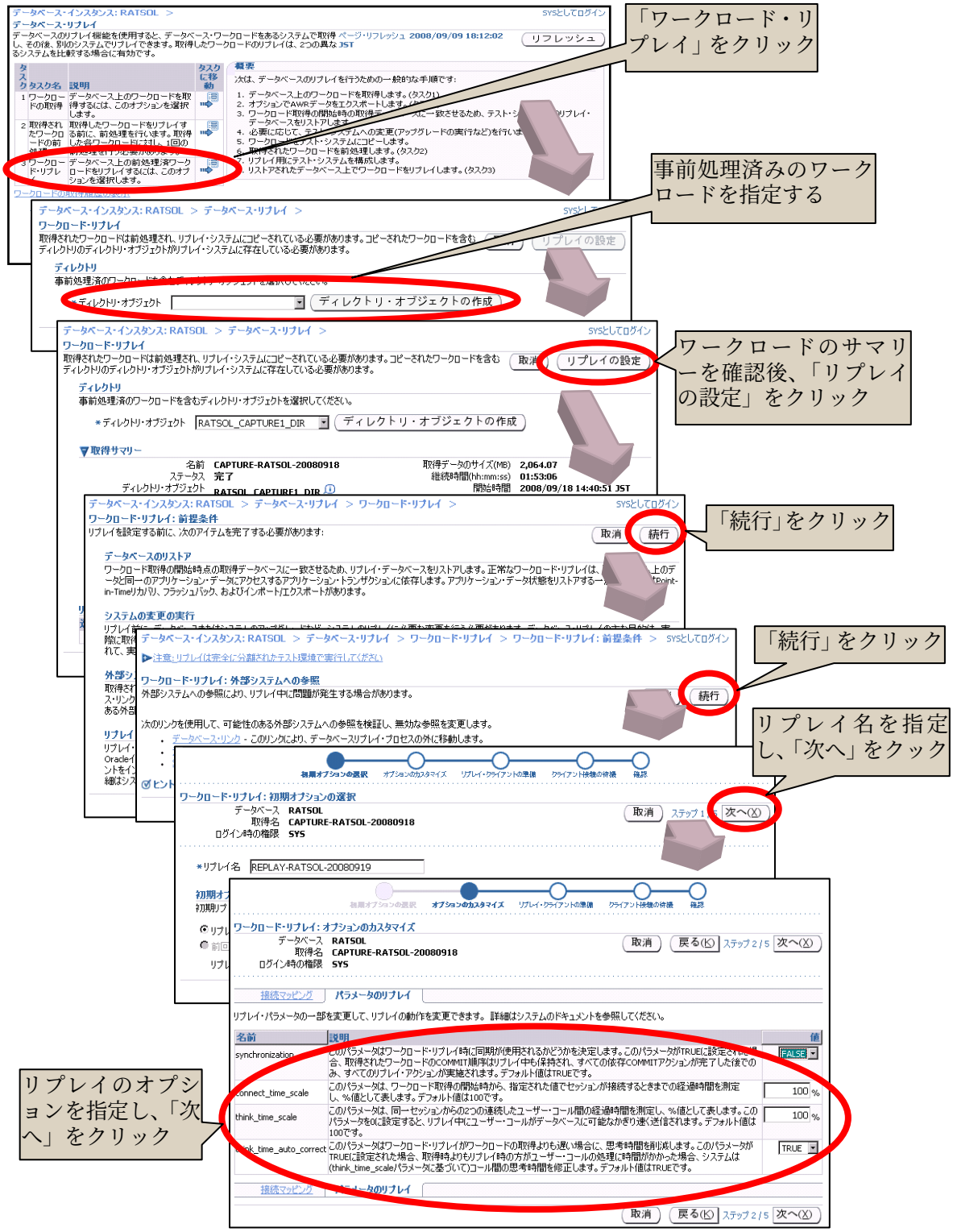
Step 4-6　　テスト DB の起動
テスト環境の DB インスタンスを起動します。

Phase 5　　リプレイの実行(テスト環境)

ワークロードのリプレイを行います。リプレイクライアントの起動以外は、Enterprise Manager から行います。

Step 5-1　　リプレイの準備

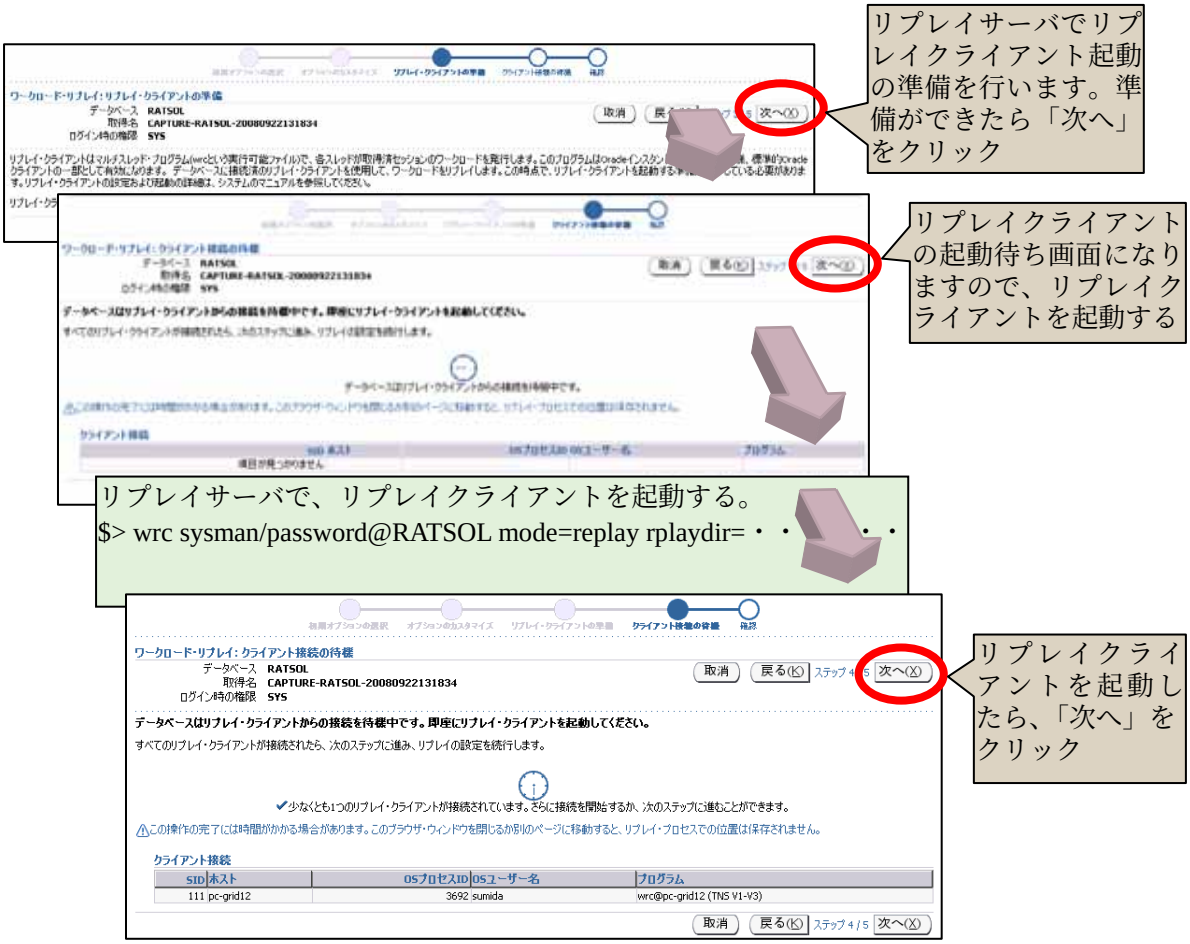
Enterprise Manager からワークロードの再生の準備として、再生オプションの設定を行います。再生オプションには、コミット順の保証や、トランザクション間隔の保証などのオプションがあります。



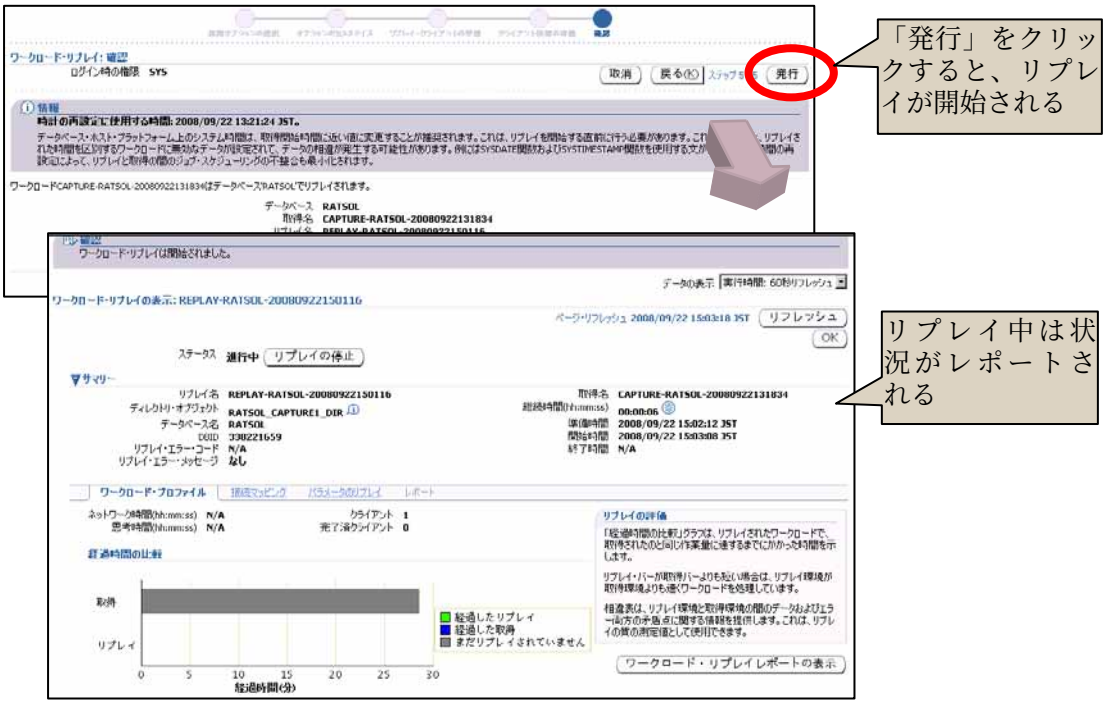
Step 5-2　　リプレイクライアントの実行

リプレイサーバで、リプレイクライアントを起動します。

起動したリプレイクライアントは、Step 5-3 の操作が行われるまで、ワークロードの再生を待機します。



Step 5-3　　リプレイ開始
Enterprise Manager からワークロードの再生の操作を行います。
この操作により、ワークロードの再生が開始されます。



Step 5-4　　テストの実施
ワークロードの再生中に行うべきテストがある場合、実行します。

Step 5-5　　リプレイの終了
ワークロードの再生を中断するか、全てのワークロードを再生すると、リプレイが終わります。

Step 5-6　　レポートの確認
Enterprise Manager からリプレイレポートを参照し、結果を確認します。
リプレイレポートでは、エラーとなったトランザクション数や、
取得レコード件数の異なったトランザクション数などが報告されます。

Phase 6　　テスト DB の復旧作業(テスト環境)

繰り返しワークロードのリプレイを実行する場合、この Phase を実行し、テスト DB をワークロード実行前の状態に戻します。テスト DB の復旧後は、Phase 5 へ戻ります。ワークロードを繰り返し実行する必要がない場合、この Phase を実行する必要はありません。Phase 7 へ進みます。

Step 6-1　　テスト DB および ASM インスタンスの停止
テスト DB インスタンスと ASM インスタンスを停止します。

Step 6-2　　Copy-on-Write Snapshot による表領域のリストア
Step4-5 で作成したスナップショットを使用して、テスト DB の表領域をワークロードリプレイ前の状態に戻します。

Step 6-3　　ASM インスタンスの起動
テスト環境の ASM インスタンスを起動します。

Step 6-4　　制御ファイルのリストア
テスト環境の DB インスタンスを nomount 状態で起動し、Step 4-7 で取得したバックアップセットから、制御ファイルをリストアします。

Step 6-5　　DB のリカバリ

テスト環境の DB インスタンスを mount 状態にし、制御ファイルを元に DB のリカバリを行います。

Step 6-6　DB のオープン
テスト環境の DB インスタンスを resetlogs オプション付きでオープンします。

Phase 7　テスト後の作業

テストが終了したら、この Phase を実行し、Phase1 実行前の状態に戻します。

Step 7-1　テスト環境の DB インスタンス、ASM インスタンス等の停止
テスト環境の EnterpriseManager、DB インスタンス、ASM インスタンス、リスナーを停止します。

Step 7-2　スナップショットの破棄
Copy-on-Write Snapshot で作成した、テスト DB のスナップショットを破棄します。
尚、ShadowImage ペアの破棄は不要です。

Step 7-2　テスト環境のキャプチャ・ファイル格納ディレクトリのアンマウント
テスト DB サーバおよびリプレイクライアントでマウントしていた、
キャプチャ・ファイル格納ディレクトリをアンマウントします。

以上が Database Replay＋ShadowImage を使用したテストの実行手順となります。

付録B.　テスト DB 作成時間

Database Replay では、キャプチャしたワークロードを実行するために、本番 DB からテスト DB を複製する事が重要です。本ホワイトペーパーでは、DB の複製に ShadowImage を使用する事を推奨しています。DB の複製に ShadowImage を使用する事により、複製にかかる時間がどれだけ短縮されるか、検証を行い確認しました。

B1.　テスト環境作成時間測定の検証環境および前提条件

検証に使用した環境は以下の通りです。

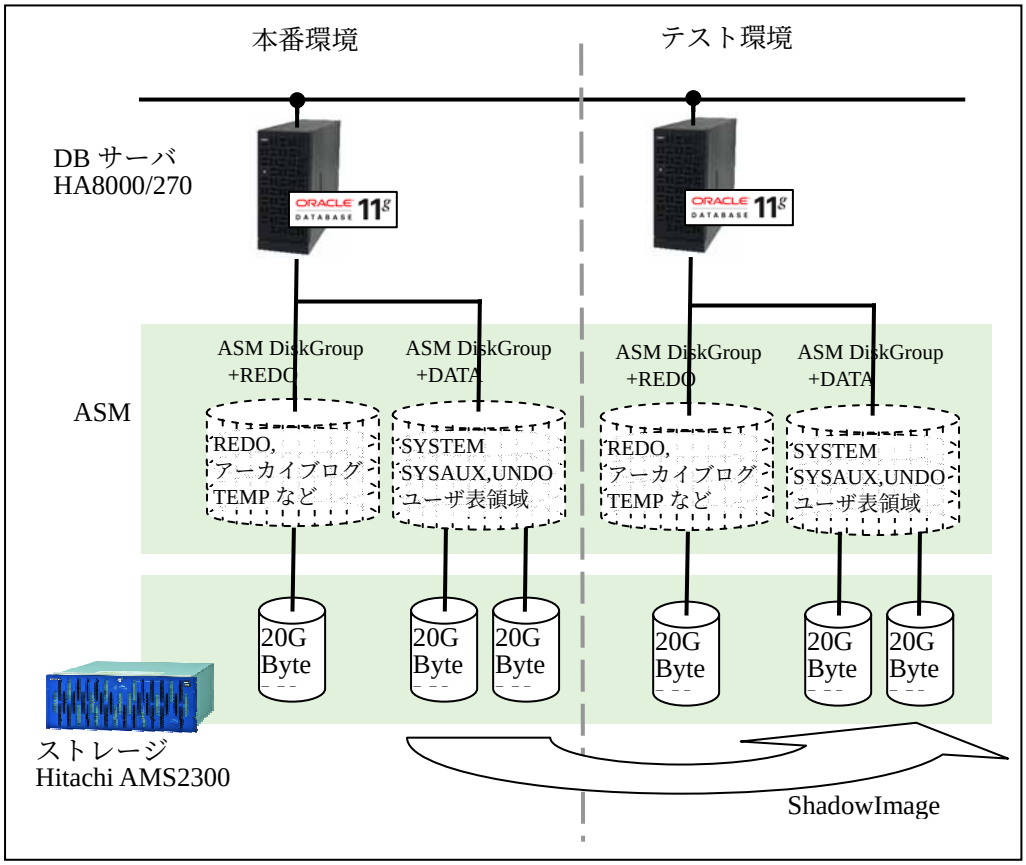


図 B-1 テスト環境作成時間測定に使用した検証環境

検証条件を表 B-1 に示します。

表 B-1　検証の前提条件

#	項目	値
1	DB サイズ	26GByte
2	使用済みブロックサイズ	11GByte
3	バックアップ形態	オンラインバックアップ
4	バックアップ時のワークロード	なし
5	バックアップ範囲	フルバックアップ

B2. 検証ケースおよび検証結果

検証では、ShadowImage を使用した DB の複製時間と、RMAN のみで DB を複製する時間を比較しました。比較した DB の複製手順を表 B-2 に示します。

表 B-2　比較する DB の複製手順

#	作業概要	DB の複製方法	
		RMAN のみで複製する方法	ShadowImage を使用して複製する方法
1	バックアップセットの作成 (本番環境で行うオペレーション)	・RMAN によるフルバックアップ ・RMAN による制御ファイル、バックアップ中の REDO 情報のバックアップ	・ASM のリバランスの停止 ・DB をバックアップモードへ変更 ・ShadowImage による複製 ・DB をノーマルモードへ変更 ・ASM のリバランスの再開 ・RMAN による制御ファイル、バックアップ中の REDO 情報のバックアップ
2	複製 DB の作成 （テスト環境で行うオペレーション）	・DB を nomount で起動 ・制御ファイルのリストア ・DB のマウント ・アーカイブログのリストア ・DB ファイルのリストア ・リカバリ実行 ・DB のオープン	・DB を nomount で起動 ・制御ファイルのリストア ・DB のマウント ・アーカイブログのリストア ・リカバリ実行 ・DB のオープン

検証結果を図 B-2 に示します。

RMAN を使用した複製では、バックアップセットを使用して DB を複製します。このため、本番環境で全データファイルを一度バックアップセットとして吸い上げたのち、テスト環境でバックアップセットからデータファイルをリストアする必要があります。ShadowImage を使用した複製では、データファイルを Disk 内部で複製します。中間ファイルであるバックアップセットを作成する必要がなく、シンプルです。

実際の検証結果でも、図 B-2 に示すように DB の複製時間が半分以下になる事がわかります。検証で用いた DB サイズは 26G 程度ですが、DB サイズが大きければ大きいほど、短縮される時間も長くなります。

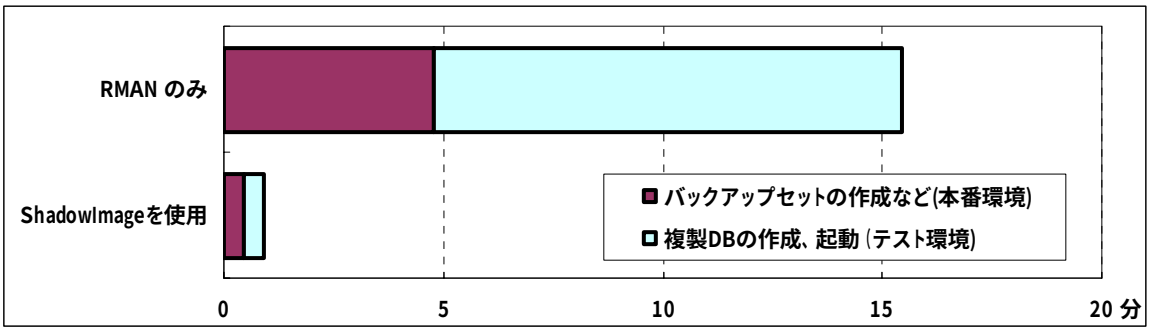


図 B-2　DB の複製時間の比較

注意：測定した手順はどちらもオンラインバックアップの手順です。本番環境のオペレーション中に、DB を停止する必要はありません。

付録C.　Database Replay で生成されるレポート例

Database Replay では、リプレイの結果を２つのレポートから判断します。

・　リプレイレポート
・　AWR の比較レポート

これらのレポートについて、検証時に出力された結果をサンプルとして示します。

C1.　　リプレイレポート

リプレイレポートでは、実行時間や DB 時間、エラーの発生した SQL など、キャプチャ時とリプレイ時の実行結果を比較します。

リプレイレポートによって、ワークロードのリプレイが忠実に行われたかを判断する事が可能です。

リプレイレポートの例(先頭部分)

REPLAY4

DB Name	DB Id	Release	RAC	Replay Name	Replay Status
RAT	3192618575	11.1.0.6.0	NO	REPLAY4	COMPLETED

Replay Information

Information	Replay	Capture
Name	REPLAY4	TEST5_CAPTURE
Status	COMPLETED	COMPLETED
Database Name	RAT	RAT
Database Version	11.1.0.6.0	11.1.0.6.0
Start Time	07-JUL-08 11:39:20	04-JUL-08 11:47:53
End Time	07-JUL-08 13:50:17	04-JUL-08 14:01:13
Duration	2 hours 10 minutes 57 seconds	2 hours 13 minutes 20 seconds
Directory Object	CAPTURE2_DIR	CAPTURE2_DIR
Directory Path	/home/sumida/RAT/CAPTURE2/aaa	/home/sumida/RAT/CAPTURE2/aaa

ワークロードのキャプチャ/リプレイ時の状況

Replay Options

Option Name	Value
Synchronization	FALSE
Connect Time	100%
Think Time	0%
Think Time Auto Correct	TRUE
Number of WRC Clients	1 (1 Completed, 0 Running)

ワークロードリプレイ時の再生オプション

Replay Statistics

Statistic	Replay	Capture
DB Time	99661.214 seconds	111808.657 seconds
Average Active Sessions	12.68	13.98
User calls	25461157	25461157
Network Time	17288.353 seconds	
Think Time	0.000 seconds	

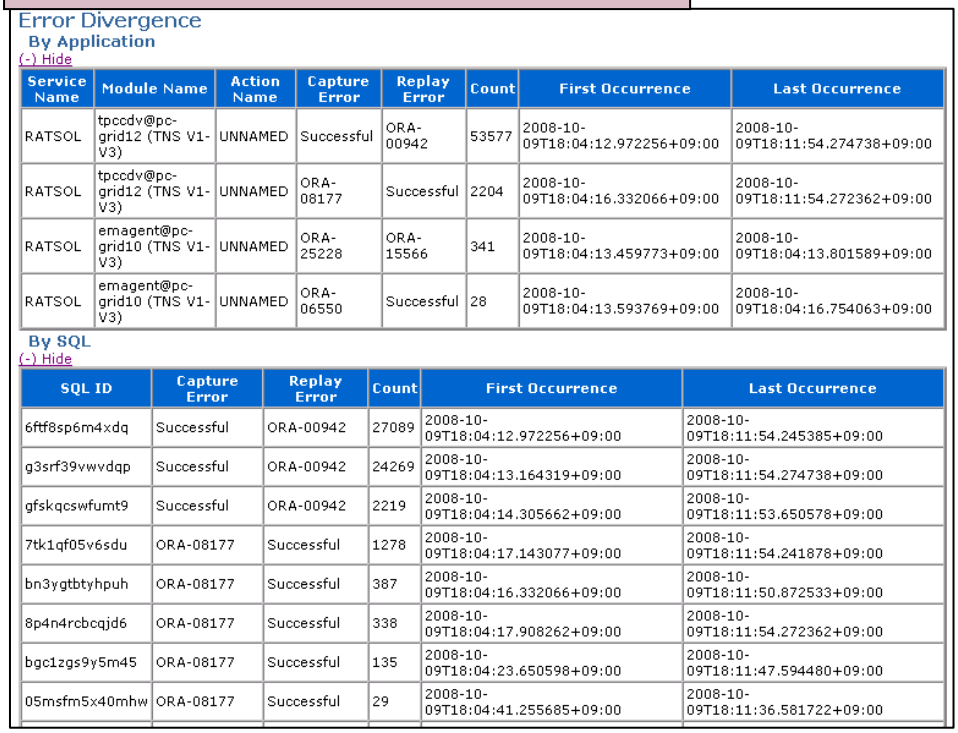
ワークロードリプレイの統計情報

Replay Divergence Summary

Divergence Type	Count	% Total
Session Failures During Replay	0	0.00
Errors No Longer Seen During Replay	55840	0.22
New Errors Seen During Replay	11678	0.05
Errors Mutated During Replay	0	0.00
DMLs with Different Number of Rows Modified	88395	0.35
SELECTs with Different Number of Rows Fetched	1179	0.00

キャプチャとリプレイの相違点

リプレイレポートの例(エラーに関する相違点の部分)



Error Divergence

By Application

(-) Hide

Service Name	Module Name	Action Name	Capture Error	Replay Error	Count	First Occurrence	Last Occurrence
RATSOL	tpccdv@pc-grid12 (TNS V1-V3)	UNNAMED	Successful	ORA-00942	53577	2008-10-09T18:04:12.972256+09:00	2008-10-09T18:11:54.274738+09:00
RATSOL	tpccdv@pc-grid12 (TNS V1-V3)	UNNAMED	ORA-08177	Successful	2204	2008-10-09T18:04:16.332066+09:00	2008-10-09T18:11:54.272362+09:00
RATSOL	emagent@pc-grid10 (TNS V1-V3)	UNNAMED	ORA-25228	ORA-15566	341	2008-10-09T18:04:13.459773+09:00	2008-10-09T18:04:13.801589+09:00
RATSOL	emagent@pc-grid10 (TNS V1-V3)	UNNAMED	ORA-06550	Successful	28	2008-10-09T18:04:13.593769+09:00	2008-10-09T18:04:16.754063+09:00

キャプチャとリプレイの相違点の詳細

By SQL

(-) Hide

SQL ID	Capture Error	Replay Error	Count	First Occurrence	Last Occurrence
6ftf8sp6m4xdq	Successful	ORA-00942	27089	2008-10-09T18:04:12.972256+09:00	2008-10-09T18:11:54.245385+09:00
g3srf39vwvdqp	Successful	ORA-00942	24269	2008-10-09T18:04:13.164319+09:00	2008-10-09T18:11:54.274738+09:00
gfskqcswfumt9	Successful	ORA-00942	2219	2008-10-09T18:04:14.305662+09:00	2008-10-09T18:11:53.650578+09:00
7tk1qf05v6sdu	ORA-08177	Successful	1278	2008-10-09T18:04:17.143077+09:00	2008-10-09T18:11:54.241878+09:00
bn3ygtbtyhpuh	ORA-08177	Successful	387	2008-10-09T18:04:16.332066+09:00	2008-10-09T18:11:50.872533+09:00
8p4n4rcbcqjd6	ORA-08177	Successful	338	2008-10-09T18:04:17.908262+09:00	2008-10-09T18:11:54.272362+09:00
bgc1zgs9y5m45	ORA-08177	Successful	135	2008-10-09T18:04:23.650598+09:00	2008-10-09T18:11:47.594480+09:00
05msfm5x40mhw	ORA-08177	Successful	29	2008-10-09T18:04:41.255685+09:00	2008-10-09T18:11:36.581722+09:00

キャプチャとリプレイで相違したSQL

C2. AWR の比較レポート

AWR の比較レポートでは、キャプチャ時とリプレイ時の AWR レポートを比較します。
AWR の比較レポートによって、システム変更によるパフォーマンスの影響を分析する事が可能です。

AWR の比較レポート(サマリー)

名前	第1期間メトリック率	第2期間メトリック率	第1期間値	第2期間値	第1期間率/秒	第2期間率/秒
DB cpu (seconds)			0.00	0.00	0.00	0.00
DB time (seconds)			10,397.76	10,250.02	2.89	2.85
db block changes			98,557.00	41,191.00	27.36	11.44
execute count			77,856.00	39,040.00	21.61	10.84
global cache cr block receive time (seconds)			0.00	0.00	0.00	0.00
global cache cr blocks received			0.00	0.00	0.00	0.00
global cache current block receive time (seconds)			0.00	0.00	0.00	0.00
global cache current blocks received			0.00	0.00	0.00	0.00
global cache get time (seconds)			0.00	0.00	0.00	0.00
global cache gets			0.00	0.00	0.00	0.00
opened cursors cumulative			73,210.00	35,027.00	20.32	9.72
parse count (total)			36,221.00	11,159.00	10.06	3.10
parse time cpu (seconds)			3.93	0.99	0.00	0.00
parse time elapsed (seconds)			4.46	0.96	0.00	0.00
physical reads			86,424.00	193.00	23.99	0.05
physical writes			4,992.00	3,057.00	1.39	0.85
redo size (KB)			18,101.77	7,358.21	5.03	2.04
session cursor cache hits			55,959.00	31,207.00	15.54	8.66
session logical reads			826,995.00	156,661.00	229.59	43.49
sql execute cpu time (seconds)			0.00	0.00	0.00	0.00
sql execute elapsed time (seconds)			0.00	0.00	0.00	0.00
user calls			14,431.00	12,887.00	4.01	3.58
user commits			1,897.00	1,897.00	0.54	0.25
user rollbacks			248.00	248.00	0.07	0.07
workarea executions - multipass						
workarea executions - onepass						
workarea executions - optimal						

AWR の比較レポート(詳細レポート)

Snapshot Set	DB Name	DB Id	Instance	Inst num	Release	Cluster	Host	Std Block Size
First (1st)	RATSOL	338221659	RATSOL	1	11.1.0.6.0	NO	pc-grid10	8192
Second (2nd)	RATSOL	338221659	RATSOL	1	11.1.0.6.0	NO	pc-grid10	8192

Snapshot Set	Begin Snap Id	Begin Snap Time	End Snap Id	End Snap Time	Avg Active Users	Elapsed Time (min)	DB time (min)
1st	106	10-10月-08 17:00:08 (金)	107	10-10月-08 18:00:10 (金)	0.02	60.04	1.30
2nd	100	10-10月-08 13:00:55 (金)	101	10-10月-08 14:00:57 (金)	0.00	60.04	0.24
%Diff					*-100.00*	*0.00*	*-81.23*

Host Configuration Comparison

	1st	2nd	Diff	%Diff
Number of CPUs:	2	2	0	0.00
Physical Memory:	3899M	3899M	0M	0.00
Load at Start Snapshot:	.08	.04	-.04	-50.00
Load at End Snapshot:	.02	.03	.01	50.00
%User Time:	1.31	.78	-.53	-40.46
%System Time:	.37	.29	-.08	-21.62
%Idle Time:	98.31	98.92	.61	0.62
%IO Wait Ti...	.74	.43	-.3	-41.89

各統計情報の比較

System Configuration Comparison

	1st	2nd	Diff	%Diff
SGA Target:			0M	0.00
Buffer Cache:	560M	544M	-16M	-2.86
Shared Pool Size:	272M	288M	16M	5.88
Large Pool Size:	16M	16M	0M	0.00
Java Pool Size:	16M	16M	0M	0.00

付録D. Database Replay 使用時の注意事項

Database Replay 使用時の注意事項について記載します。

1. リプレイを実行する DB に対する注意事項

Database Replay を用いたテストでは、リプレイに使用する DB の内容を、ワークロードキャプチャ前の DB と正確に一致させる事が重要です。ワークロードキャプチャ前の DB と、リプレイに使用する DB の内容に差異がある場合、DB の差異により発生した SQL エラーや性能劣化が、リプレイのレポートに相違点として報告される可能性があります。これがテスト結果の解析においてノイズとなり、何が問題であるのかが分かりにくくなります。

2. ワークロード再生の注意事項

Database Replay は、あくまでキャプチャしたワークロードのリプレイを行うものであり、実際にアプリケーションを実行するものではない事に注意してください。これは、pro*C などの言語を使用したプログラムにおいて、SELECT で取得した結果を用いて DML を発行する場合に問題となります。表 D-1 の#4 のケースでは、リプレイ時、なんらかの理由により SELECT で取得した結果が異なっても、次に再生される DML には SELECT の結果が反映されません。キャプチャ時の SELECT 結果でそのまま DML を実行します。また、リプレイレポートでは、SQL の実行に失敗した場合や、fetch 件数が異なる場合は報告されますが、前述のような列値の違いは報告されません。

3. 再生モードに関する注意事項

Database Replay には、同期と非同期の 2 つリプレイモードがあります。各リプレイモードの注意点は以下の通りです。

- 同期モード
  Commit 順を保証するために、ワークロードのリプレイが直列化されます。このため、排他制御の確認や並列実行性が重要となるテストには適していません。

- 非同期モード
  Commit 順が保証されないため、ワークロードのキャプチャ時とリプレイ時で、トランザクションの Commit 順に相違が発生する事があります。Commit 順の相違によって、DB の内容や fetch 件数にも相違が発生します。fetch 件数の相違は、リプレイレポートでキャプチャ時との相違点として報告されます。
  トランザクション排他制御や並列実行性を再現することが重要な場合には、非同期モードを指定してください。

表 D-1　キャプチャ処理内容とリプレイ結果

#	実行処理	キャプチャ時		リプレイ時		備考
		処理実行前のID値	処理実行後のID値	処理実行前のID値	処理実行後のID値	
1	update TBL_A set ID = ID + 1	1	2	11	12	リプレイ時、列値が異なっていても期待した処理を実行

2	## PL/SQL プログラム ###
	declare
	　v_id　number;
	begin
	　select ID int v_id from TBL_A;
	　v_id := v_id + 1;
	　update TBL_A set ID = v_id;
	end;
	/
			2		12	

3	update TBL_A set ID = 2;		2		2	
4	## PRO*C プログラム ### long　v_id; EXEC SQL select ID into :v_id from TBL_A; EXEC SQL update TBL_A SET ID := v_id +1; EXEC SQL COMMIT;		2		2	リプレイ時の列値に関係なくupdate文が実行されるケース

無断転載を禁ず

このドキュメントは単に情報として提供され、内容は予告なしに変更される場合があります。このドキュメントに誤りが無いことの保証や、商品性又は特定目的への適合性の黙示的な保証や条件を含め明示的又は黙示的な保証や条件は一切無いものとします。株式会社日立製作所は、このドキュメントについていかなる責任も負いません。また、このドキュメントによって直接又は間接にいかなる契約上の義務も負うものではありません。このドキュメントを形式、手段（電子的又は機 械的)、目的に関係なく、株式会社日立製作所の書面による事前の承諾なく、複製又は転載することはできません。

このドキュメントにより、提供された技術やプログラム又は購入した製品を輸出（又は非居住者への提供）する場合は、「外国為替及び外国貿易法」その他適用される法令を遵守してください。