Intelligence Business Solutions ,Ltd.

インテリジェンス基幹システムへ PostgreSQL導入の道のりと保守運用の現状

株式会社インテリジェンスビジネスソリューションズ（IBS）
渡辺 雅司

自己紹介

【氏　名】

渡辺 雅司(わたなべ まさし)

【勤務先】

株式会社インテリジェンスビジネスソリューションズ

【業務内容】

2012年10月インテリジェンスグループ向けインフラ保守業務と派生するプロジェクトの責任者となる。

保守運用業務のプロセス化、構成見直し等によるシステムの安定化など、Group向けインフラ全般の統制/更改/改善を推進している。

今後は、インテリジェンスグループのインフラを更にピカピカにし、保守運用を徹底的に科学して、世の中へ提供していきたい。
勿論、オープンソースも徹底的に利用したい。

Section 1: IBS会社概要

Section 2: IBS戦略の考え方

Section 3: 担当業務概要/構成と講演の範囲

Section 4: OSS-DB検討の経緯(導入決定までの検討)

Section 5: PostgreSQLの構成検討/設計

Section 6: 運用業務の準備(アセット化)

Section 7: 保守運用フェーズ

Section 8: まとめ

会社概要

【社　　名】　株式会社インテリジェンス ビジネスソリューションズ（略称：IBS）
【資 本 金】　3億1千万円(インテリジェンスホールディングス 100%)
【売 上 高】　約163億円（事業統合前の各事業の2014年3月末決算数値を合算）
【社員数】　2,220名
【事業内容】　業務プロセスコンサルティング、システム企画・開発、システム運用・保守、
ICTアウトソーシング、新エネルギーアウトソーシング、セールスアウトソーシング、
WEBアナリティクスサービス、バックオフィス支援、カスタマーサポート支援
【代表者名】　長井 利仁
【設　　立】　1977年9月（昭和52年9月24日）
【決 算 期】　3月
【取引銀行】　三井住友銀行　新宿支店
【事業拠点】　札幌、仙台、東京、幕張、大阪、沖縄、ベトナム（オフショア開発センター）
【連携事業】　株式会社IBS沖縄（グローバルチーム型オフショア開発）
株式会社IBSベトナム（グローバルチーム型オフショア開発）
【本社所在地】〒135-0061　東京都江東区豊洲3-2-20　豊洲フロント7階
【認　　証】　ISO9001、ISO27001認証取得
【主なパートナーシップ】
IVS(ベトナム)、マネジメントソリューションズ(PMO)、Microsoft、 SalesForce.com、VMware、RedHat、
ZabbixSIA、ミラクル・リナックス(ZABBIX)、NTTデータ(Hinemos)、NTTデータ先端技術(Splunk)

沿革

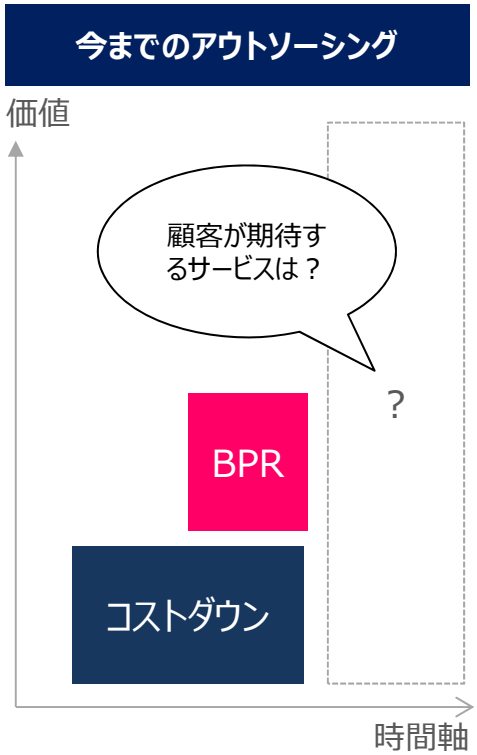
昭和52年　9月　ミリオンソフトサービス株式会社設立
（後にミリオン株式会社に改称）
平成17年10月　株式会社インテリジェンスの資本参加（100%出資）
平成18年11月　ISMS認証を、ISO27001認証に切替
平成19年10月　株式会社インテリジェンスビジネスソリューションズに社名変更
株式会社インテリジェンスよりシステムソリューション事業および
ネットワークソリュー ション事業を継承
平成20年　5月　セールスフォースアライアンス提携
人材ソリューションのサービスを開始
平成21年　5月　ミラクル・リナックス社とZABBIXパートナープログラムによる
提携を開始
平成21年　7月　日本オラクルとSOAコンサルティングサービスで協業開始
平成21年11月　ベトナム ホーチミンにオフショアセンターを開設
平成23年　2月　ASP型のアルバイト・パート採用支援システム
「HITO-Manager」（ヒトマネジャー） を提供開始
平成23年10月　「仮想化基盤統合管理アプライアンス」を提供開始
平成24年 4月　Zabbix SIA社とパートナー契約を締結

平成9年7月　ITアウトソーシングサービス提供開始
平成12年10月　ECサーブテクノロジー株式会社設立
平成18年4月　アウトソーシングコンサルティングサービス開始
平成19年7月　内部統制支援サービス開始
平成20年11月　サービスマネジメント独自モデルIBIQリリース
平成20年4月　社員のシステム運用技術者育成による
サービス拡大
平成21年4月　公共案件でのアウトソーシングサービス展開
平成22年10月　(株)BPOソリューションズ設立
平成23年2月　(株)KDDIとの合弁会社KDDIまとめてオフィス設立
平成23年7月　マネジメントソリューションズ提携開始
平成23年9月　震災後のBCP対応として関西エリアでの
ITアウトソーシングサービス開始
平成24年9月　戦略的アウトソーシングサービス提供開始
平成24年10月　“IT＝楽しい”を伝えるセミナー「i-campus」開始
平成25年2月　IBSへ事業統合

★平成25年4月　インテリジェンスICTアウトソーシング事業を統合

IBSが考えるこれからの提供価値

- システム＋業務領域の多くの実績にインテリジェンスグループの人材再配置能力を加えトータルでのサービス提供を実現　顧客にとってなくてはならない存在を目指す。



- ✔ 肥大化したレガシーシステムの保守を預かることで新規案件へのサービス提供が可能
- ✔ システムと業務の一体サービス ＝ 作り手とシステム利用者との強い連携
- ✔ インテリジェンスグループが提供する人材サービス ＝ 人材再配置力
- ✔ 最もアウトソーシングが難しい人的資源領域へのサービスインテグレーションが競争力

IBSが考えるこれからの提供価値(システムインフラ領域 ITリノベーション/運用アセット化)

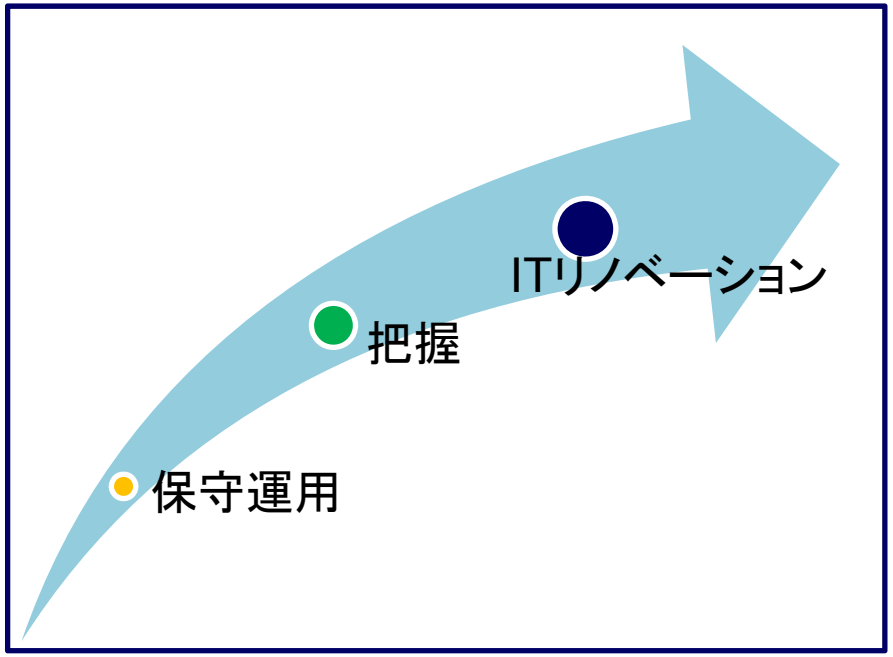
- SI事業も35年間続けていますが、現在はオープンソースを活用しつつ、以下の2つの領域に力を入れている会社です。

ITリノベーション

今あるシステム環境を有効に活用し、付加価値をつけて安価で、最適なシステムを構築することを考えています。

よって、お客様と接点を持つ、保守運用が非常に大切であり、私たちは最上流と位置づけています。

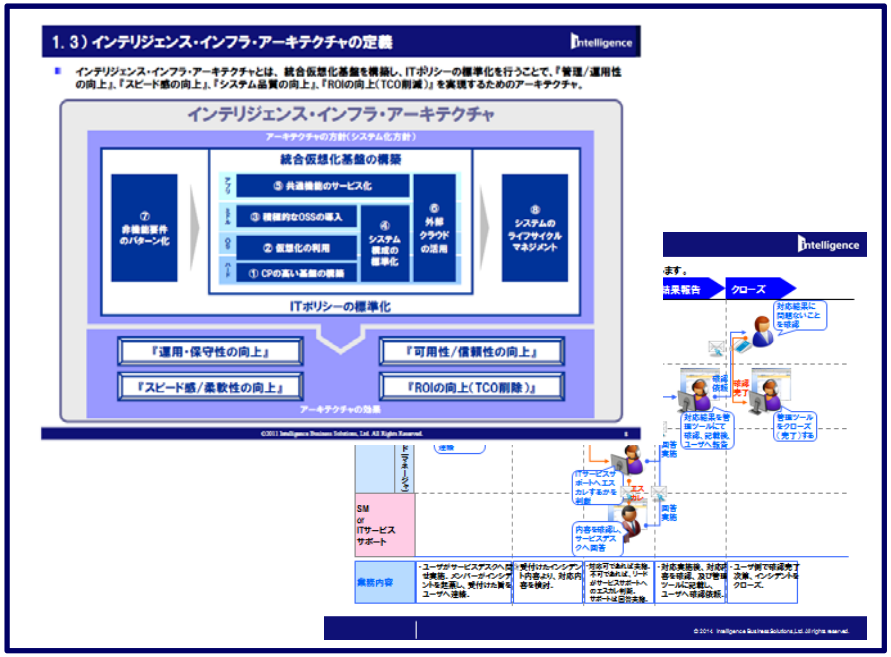
※リノベーションとは、一般的に建物に対して使う言葉で、機能を追加する、または、性能を向上させ既存の建物に付加価値をつけること。



運用アセット化

保守運用業務を最上流工程と考え、以下の整備などを実施している。

・技術標準(インテリジェンス・インフラ・アーキテクチャ)の策定
・保守運用プロセスの策定
・改善の継続



Section 1: IBS会社概要

Section 2: IBS戦略の考え方

Section 3: 担当業務概要/構成と講演の範囲

Section 4: OSS-DB検討の経緯(導入決定までの検討)

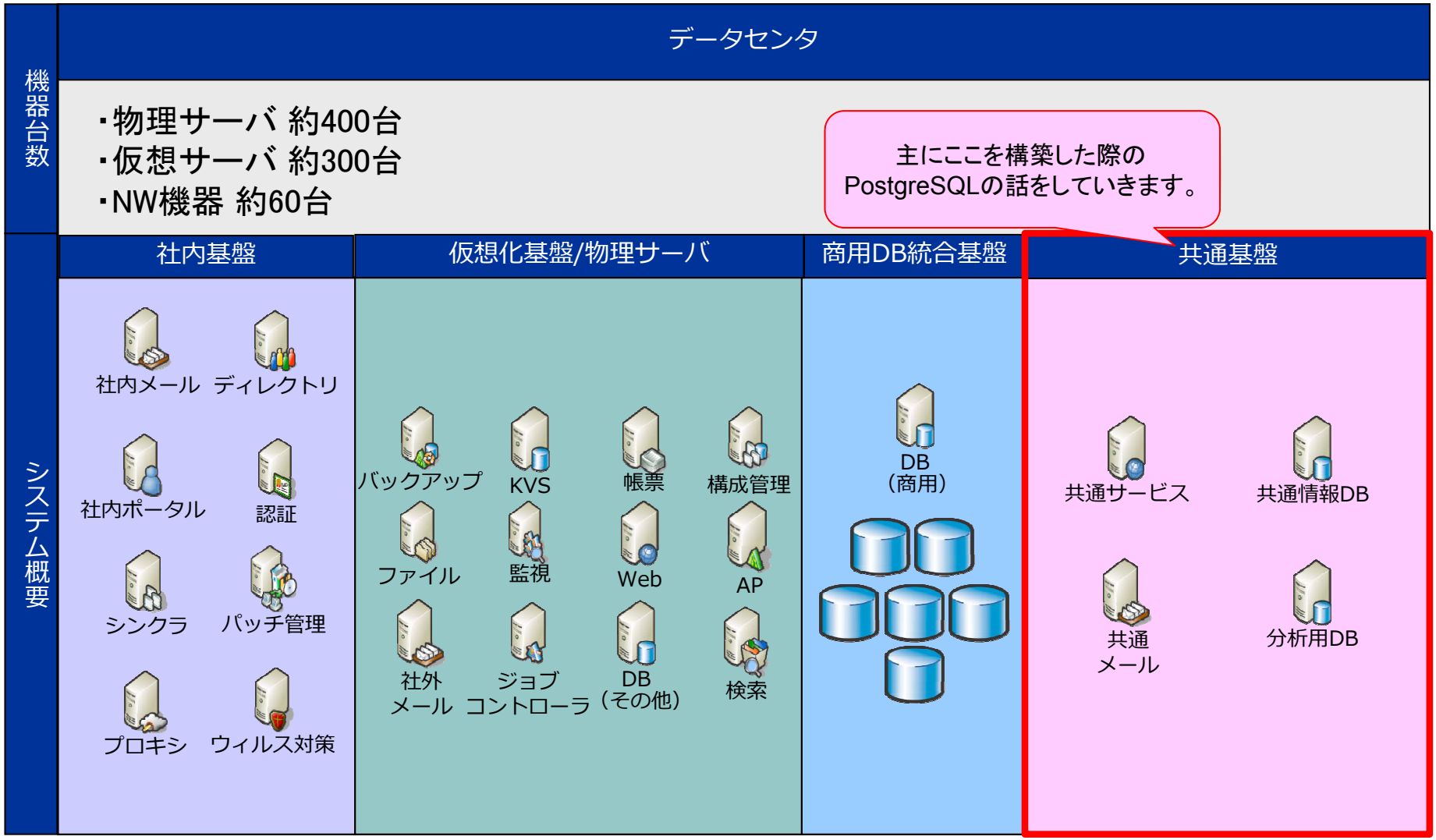
Section 5: PostgreSQLの構成検討/設計

Section 6: 運用業務の準備(アセット化)

Section 7: 保守運用フェーズ

Section 8: まとめ

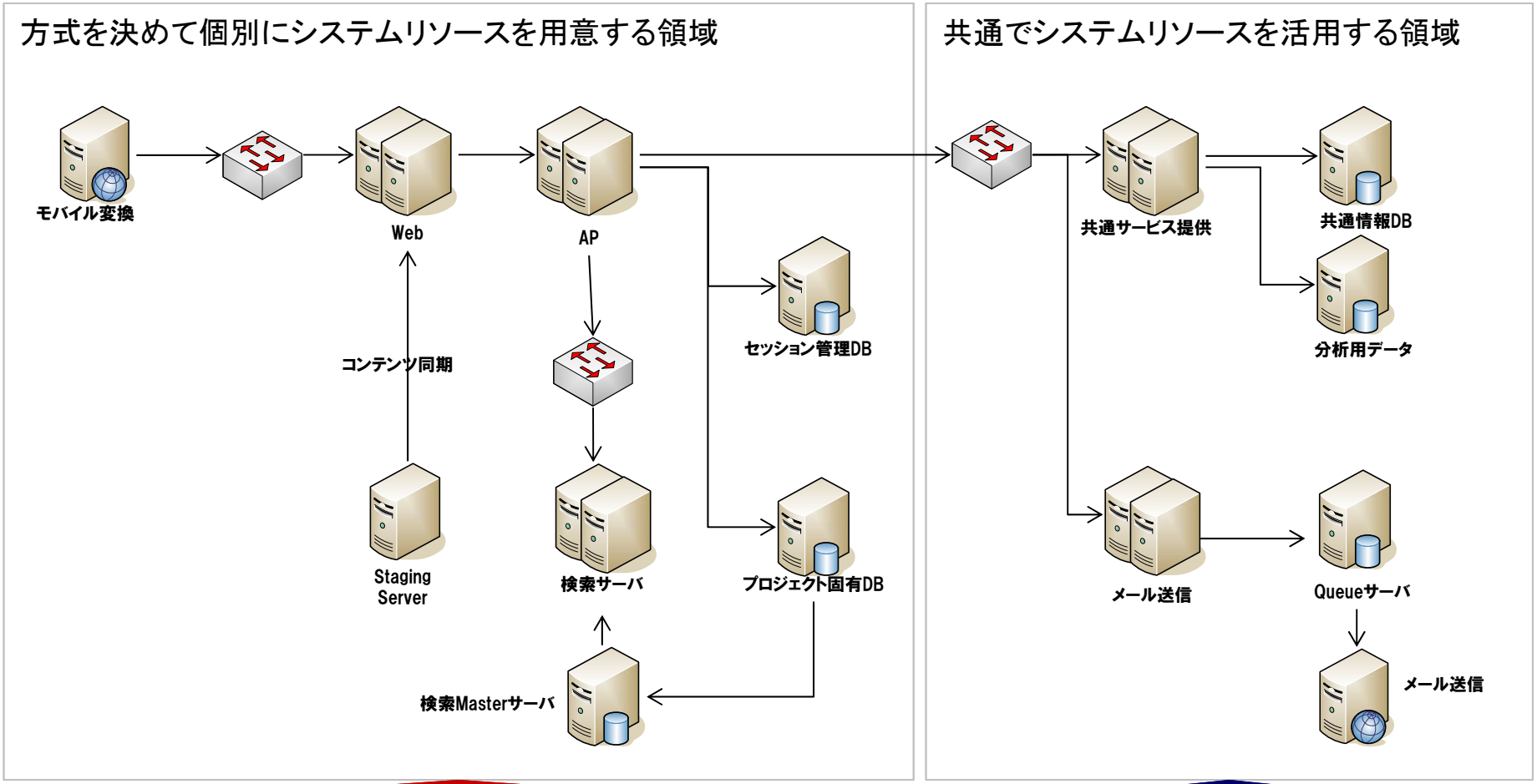
担当業務概要/構成と講演の範囲(インテリジェンスグループのインフラ保守)



共通基盤構築の目的(重要システム)

項目	説明
背景	システム/サービスにおいて、歴史的経緯の様々な問題/課題が存在 ■ IT資産の個別構築に伴うサービス/機能の重複が進行(個別/状況最適志向) ■ インテリジェンス・インフラ・アーキテクチャに準拠していない旧基盤が多く残る ■ データの共有/共用化にコストと時間がかかる
目的	ビジネスの実現スピードおよび競争力向上のため、共通基盤を構築することで、IT基盤の抜本的な改善を図る ① ITコスト(機能重複排除/HW/SW/構築/保守運用)見直しによる投資効率の向上 ② IT構築期間の削減によるビジネス競争力向上 ③ データの共用を容易に実現することによる情報資源の有効活用

共通基盤 全体論理構成



AP/インフラ統一により、方式が標準化され、設計/構築作業工数を軽減する

共通サービス化により構築領域そのものを削減する

OSS-DB導入検討の背景・目的

項目	説明
背景	【オープンソースの高機能化/安定化と状況】 オープンソースの高機能化と安定化が顕著であり、また、他社での利用実績も増えつつあること、更には、体系的な知識を得られるような環境(試験制度の確立や定期的に研修が開催)が整ってきたため、an/DODA等への重要システムのコア部分への適用にも耐えうる状況になってきた。
目的	【ベンダロック回避】 商用製品を利用している限り、都度ライセンス費用/保守サポート費用が必要となり、ベンダロック状態となっていることを回避する。 【保守運用コスト削減】 商用製品と比較して、ライセンス料が不要であることが一番大きな理由として、保守運用コストを削減することができる。

データベース検討の前提条件

- データベース選定における前提条件

・稼働率99.99%であること
・一次保守を社内で実施できること
・CentOSおよび商用Linuxで利用可能であること
・DB同士の通信はWeb-APIを介するよう実装されていること（DB同士での直接通信はしない）

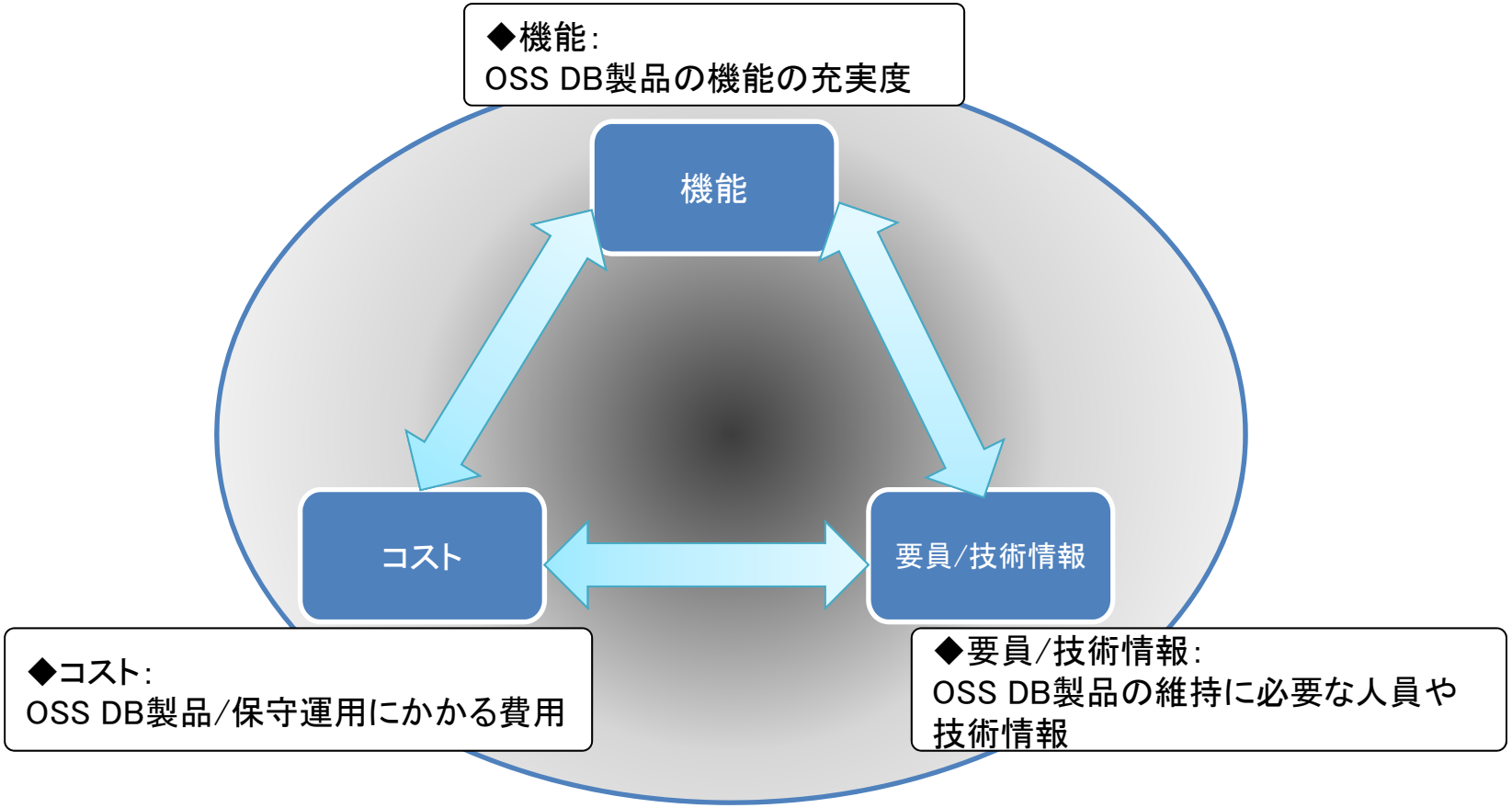
オープンソースのDBMS比較検討

- オープンソースのDBMSを利用検討にあたり、オープンソース同士の比較検討を行った。

PostgreSQL X.X	MySQL X.X
○機能が豊富で、SQLも一般的なものが多い ○商用製品と似た機能が多く、商用DBから移行しやすい ○最新版の日本語マニュアルがある 日本での普及のために貢献している企業があり、日本語情報が多い ・バージョンアップごとに速くなり、9.2では、64コアまでスケールすることが確認済み ・商用製品と互換機能を持ったもの(商用)もある ×追記型というアーキテクチャのため、更新が遅め 定期的にVacuumが必要	○国内外の大規模サイトで採用 ○以前ほどの差はないが速い ○昔からレプリケーション機能があり、多彩な構成が可能 ・商用版がある ×SQLに制限が多く、全体的に癖が多い ×某社買収後、コミュニティとの連携が？？ MariaDBへの乗り換えもある ×日本語マニュアルの最新化が遅れることがある

検討の観点/方針

- オープンソースを基幹システムとなる共通情報を格納するデータベースとして採用するにあたり、以下の観点で検討しました。



商用製品/OSS比較検討内容サマリ

カテゴリ	比較	項目	詳細
OSS-DBの機能/非機能 ①	○	機能	通常よく利用する範囲の機能は遜色なく、基幹系でも利用が始まっている
	△	機能互換	単純な互換性はないため、商用DB独自の機能を利用している部分に留意して設計・開発・移行を行う必要がある
	△	クラウド利用可否	利用可能だが、クラウドサービスによって可用性・性能要件が満たせない場合がある 利用する場合、クラウドサービスとの親和性を事前に確認する必要がある
	△	バージョンアップサイクル	メジャーバージョンアップが年1回、マイナーバージョンアップが年5回程度（不定）と比較的頻度が高い
	○	可用性	DBサーバ2台構成＋無償プロダクトの構成で99.99%の可用性を確保可能
	○	バックアップ/セキュリティ	想定されるバックアップ、セキュリティ機能は実装されている
	△	拡張性	PostgreSQLの機能にスケールアップなどの機能は無いが、HWの追加などによる拡張は容易に行える
	○	性能	一概には表現しにくいが、構成などで総合的に安価に商用製品に匹敵する性能は担保できる
コスト ②	○	SI	システムの構成によるが、商用データベースと大きな差異はない
	○	維持費	OSSは、ライセンス費用は不要であり、サポート費用は商用製品より安価（24x365）
	△	他のDBからの移行	期間・費用については移行元システム、要件によって見積もりが必要ではあるが、事例など多くの情報が開示され、商用製品と対比できる項目が非常に多いため、特別困難とはいえない
要員/技術 ③	○	製品サポート	製品をサポートするベンダ等が多く出ている。
	○	サポート期間	バージョンごとのサポート期間が最低5年あるため、都度の更新は不要
	○	SI	システムの構成によるが、商用DBとの差異はほとんど無い
	○	OSSDBの実績	ミッションクリティカルな基幹システムでの導入・運用実績がある サポート側にはシステム停止の報告は入っていない
	△	エンジニアの確保	技術者数は商用製品の1/10程度と考えられるが、事例など多くの情報が開示され、商用製品と対比できる項目が非常に多いため、商用DB技術者のスキルチェンジができると想定する
	△	技術者教育	PostgreSQLの技術者は少ない状況であり、安定的な構築/保守運用を行うためには技術者の育成は必要 またノウハウの早期習得のためには外部コンサルの利用も必要

上記検討項目のうち、いくつかの項目を重点項目（☆）としてピックアップし、次ページ以降に示す。

重点項目説明①　機能/非機能(1/2)

PostgreSQLが持つ機能については、基本的には、商用データベースと比較しても遜色がないレベルで実装されている。ただし、分散機能や大規模データベースへの対応については実績が乏しく、機能面や性能面での事前検証が必要となると考える。

機能	概要	
読み取り機能 書き込み機能	トランザクション処理により、データー貫性を保障しており、商用データベースと比較しても遜色ない。 排他制御については、行単位の排他制御を実装しており、商用データベースと遜色ない。	○
アプリケーション開発	標準SQL規格に準拠しており、基本的なSQL機能については商用データベースと比較しても遜色ない。 サポートしているデータ型、基本的な関数については、商用データベースと比較しても遜色ない。 PL/SQLに相当する手続き型言語はあるが、独自実装部分が存在するため技術の習得が必要。 チューニング支援機能は、商用データベースほど充実はしていない。	△
外部DB連携	PostgreSQL同士で連携を行う機能は標準で実装されている。 商用DBなど他社のデータベースと連携機能は実装されていない。 他社DBと連携を行う場合、外部DBもしくは他ツールの機能を利用することで対応可能。	△
負荷分散	PostgreSQL自体では、負荷分散機能は実装されていない。 他のOSSと組み合わせることで実現可能で、それなりの利用実績もある。 ただし技術の変化・安定性が激しい部分でもあり、検証の実施、技術コンサルの導入が必要となる。	△

重点項目説明①　機能/非機能(2/2)

機能	概要	
大規模DB 対応	PostgreSQLでは、SQLの並列実行機能が実装されていない。SQLの並列処理を行うためには、他のOSSと組み合わせて利用する必要があり、検証の実施、技術コンサルの導入が必要となる。	△
監査証跡機能	発行されたSQL文やコマンドの証跡を取得することが可能となっている。商用データベースの高度な監査機能には及ばないが運用に耐えうるレベルにある。	○
運用管理機能	商用データベースのような標準となるGUIベースの運用管理ツールは存在しない。 コマンドラインツールを利用した管理が主体となるが十分管理可能。外部ツールの開発が進んでいる。	△
拡張機能	PostgreSQL単独では、スケールアウトの拡張を実現できない。性能不足が発生した場合、外部ツールの利用や、ハードウエア強化が必要となる可能性がある。	△
仮想化対応	ミッションクリティカル領域での国内実績は、物理サーバ構成が主体。ただし商用DBも同様。	×
障害復旧機能 可用性	オンライン、オフラインバックアップともに実装されており、運用に耐えうるレベルにある。 著名な商用バックアップソフトウェアもサポートしている。 外部ツールなどを利用することで、障害復旧は可能と考える。	○

重点項目説明①　可用性

稼働率99.99%を満たすための可用性を確保可能か確認した。

◆OSS RDBMS製品の場合
　DBサーバを複数台構成にし、他のOSSプロダクトのクラスタソフトウェアやデータ同期ソフトウェアを組み合わせることで、99.99%の可用性は実現可能。

◆商用DBの場合
　OSS RDBMSと同様、複数台構成による99.99%可用性確保は実現可能。
　（クラスタ機能やデータ同期の機能は商用DBの基本機能に含まれている）

⇒ OSS RDBMS製品単体では商用DBに劣るが、関連プロダクトを利用することによって稼働率99.99%の可用性を実現することが可能。

重点項目説明② コスト(HW/SI/移行/維持)

各種コストについては以下の通り。
商用DBに対し、同等または安価に導入/維持が可能と考える。ただし、互換性の問題などから移行コスト発生する。
総論では商用DBより安価に維持が可能であると考えられる。

項目	概要	
HWコスト	共有ストレージなしでの構成を実現可能であるため、通常クラスタ構成より安価に調達が可能。	○
SIコスト	DB規模によるが、商用DBと大差なし。	○
移行コスト	商用DBの一部機能が利用できないため、移行のためにはプログラム開発が必要。 コストは規模やシステムの内容に依存するため、都度算出が必要。	×
維持コスト	ライセンス費用が不要。ソフトウェアサポートは有償となるが、商用DB製品に比べ安価。 スケールアップやバージョンアップの費用についても同様に、商用DB製品に比べ安価である。	○

重点項目説明② 本PJTにおけるPostgreSQLと商用DBとの費用比較(すべて概算)

項	項目	商用DBで実施した場合	PostgreSQLで実施した場合
1	ライセンス料	¥42,000,000 ※定価ではない	¥0
2	インフラ構築費用	¥3,000,000	¥1,500,000
3	PostgreSQL関連構築サポート費用(初なので)	¥0	¥3,500,000
4	社員初期教育費用(初なので)	¥0	¥1,000,000
5	製品保守サポート費用(年額)	¥20,000,000 ※定価ではない	¥2,800,000
6	OSS DB製品情報収集に伴う追加費用(年額)	¥0	¥3,000,000
計	初年度費用	一時費用:¥45,000,000 保守費用(年額):¥20,000,000	一時費用:¥6,000,000 保守費用(年額):¥5,800,000

超重要！

初のPostgreSQL採用にともない、項2,3,4のSI関連費用/教育費用部分の一時費用は3,000千円高くなるものの、ライセンス料を含め、**全体の一時費用は△39,000千円**となり、更に、項5,6の毎年必要な**保守費用等は、年額で14,200千円減額**される。
※新規構築であるため、アプリケーション開発費用/HW費用はほぼ同一である想定のため含めず

重点項目説明③　技術者教育(1/2)

商用DBとPostgreSQLの技術者数について以下にまとめる

・社内技術者数
　商用DB資格保持者　… 37名(資格取得重複を除くと30名)
　OSS DB資格保持者 … 0名

・中途入社面接時に職務経歴書にPostgreSQLが記載されている率
　商用製品資格 10人に対し、1人程度。(10対1)

重点項目説明③　技術者教育(2/2)

PostgreSQLについては、日々進歩しおり、利用事例も日々増えている状況ではある。
ただし、前述の通り技術者の絶対数はまだまだ少なく、PostgreSQLの構築/保守運用を安定的に行うためには、技術者の育成が必要と考える。

○技術者教育

・研修の受講(有償)
IBS社外を見ても、技術者はまだ少ない状況であり、また長期視点で考えて、外部調達を検討するよりは、IBS社員を教育することが、有効と考える。

・選抜による資格取得の義務化(有償)
安定的運用を目指すため、OSS DB Goldの取得の義務化を検討する。

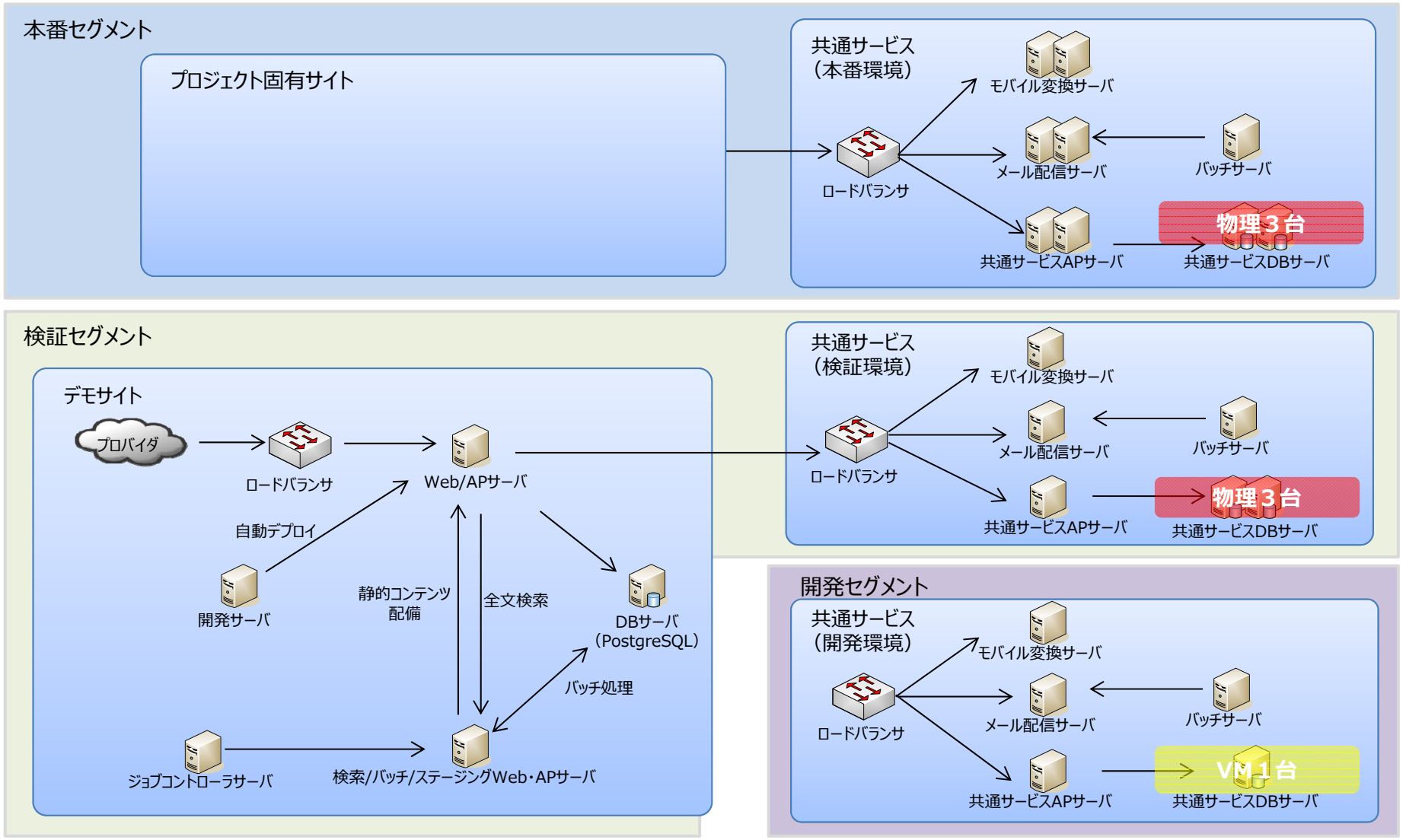
※OSS DBの資格については、取得を奨励すべく資格制度に組み込みを行っている

○ノウハウの吸収

・コンサルティングの投入
プロジェクトスケジュールに合わせ、また、早期に運用を安定させ、技術や実運用のノウハウを取得するために、OSS SRA、NRI等のコンサルティングの要員をプロジェクトや運用設計等に参画させることが必要と考える。

・技術勉強会/研究会等に参加検討
技術情報等をいち早く察知すること、情報交換、ノウハウを吸収することを目的として、勉強会等への参加を検討している。

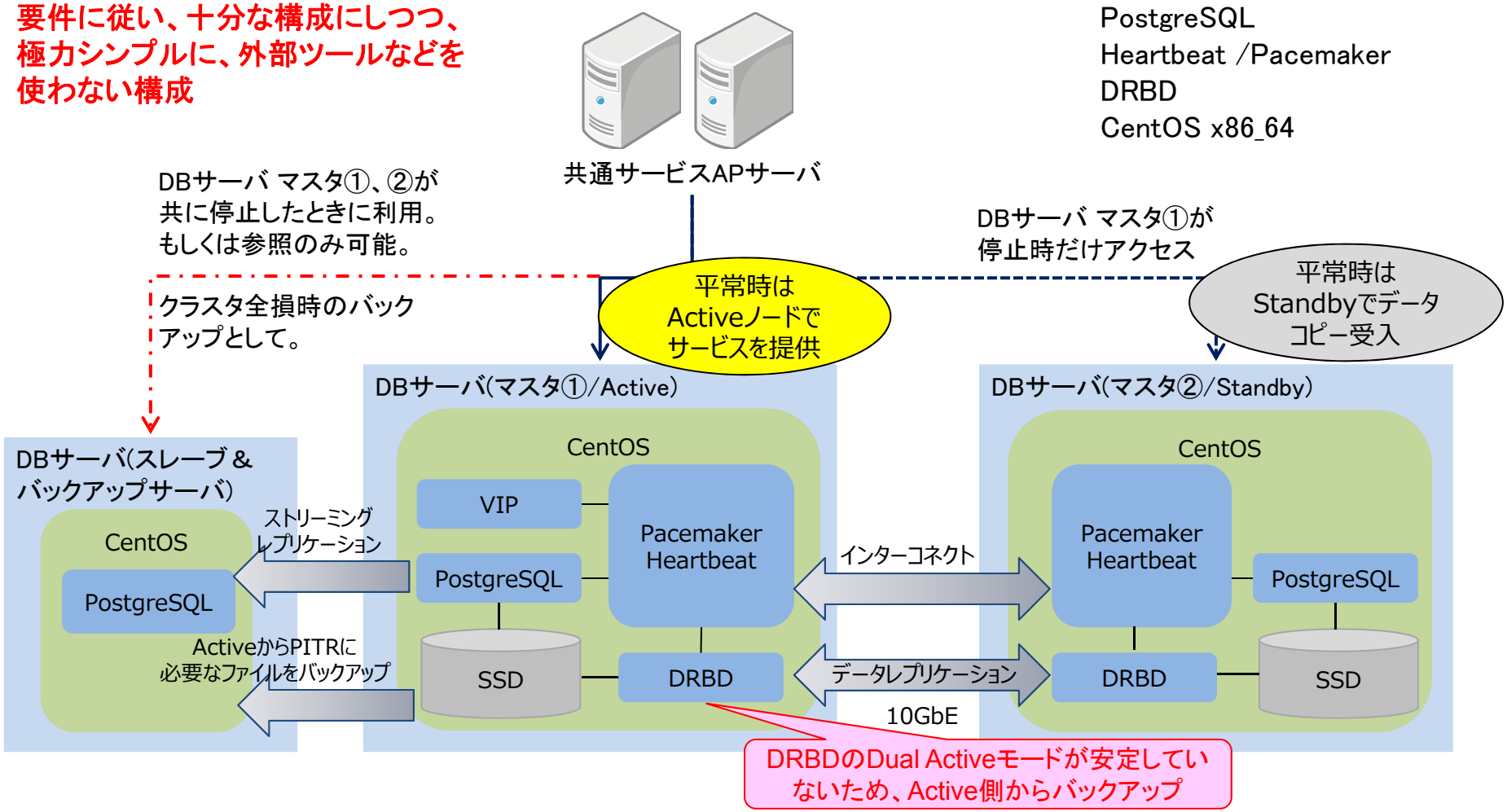
共通基盤 における全体構成



DBサーバ構成概要

要件に従い、十分な構成にしつつ、極力シンプルに、外部ツールなどを使わない構成

PostgreSQL
Heartbeat /Pacemaker
DRBD
CentOS x86_64



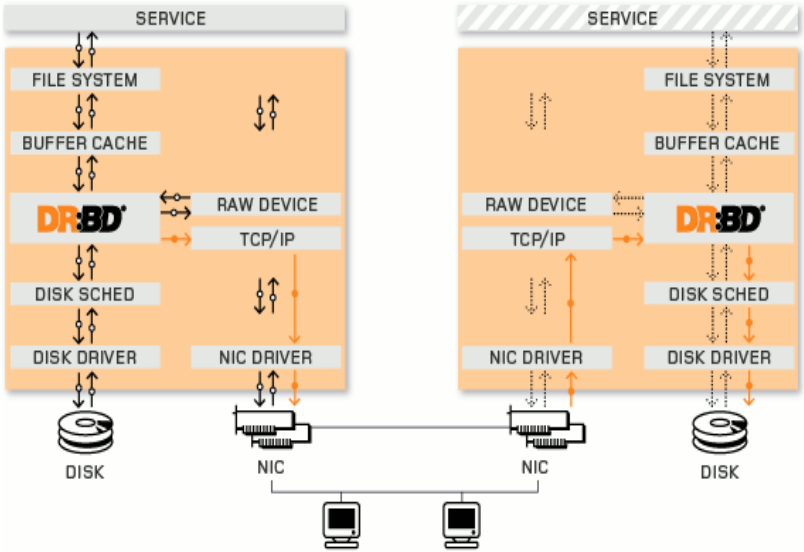
※本番環境と検証環境で、同一構成のものを2セット構築。検証環境のHWはスペックダウンしたもの採用。開発環境はVM利用。
※本番環境はDB用ストレージとしてSSDを採用。検証・疑似環境では通常のHDDによるRAIDを採用。

各DBサーバの役割

- マスタ①/②
  - 稼働しているサービス
    - Heartbeat/Pacemaker: HAクラスタソフトウェア
    - DRBD: ディスクミラーリングソフトウェア
    - PostgreSQL: DBMS(Active側だけで稼働)
  - 格納されているデータ
    - PostgreSQLに格納したデータ(共通サービスデータ)
    - PostgreSQLのログ(DRBD領域)
- スレーブ
  - 稼働しているサービス
    - PostgreSQL: DBMS。クラスタサーバのレプリケーション(スレーブ)。
    - Apache: DBパフォーマンスレポートツールのプラットフォーム(pg_stats_reporter)
  - 格納されているデータ
    - PostgreSQLに格納したデータ(共通サービスDBのレプリケーション)
    - PostgreSQLのログ
    - クラスタ機共通サービスDBのベースバックアップ
    - クラスタ機共通サービスDBのトランザクションログ(WAL)

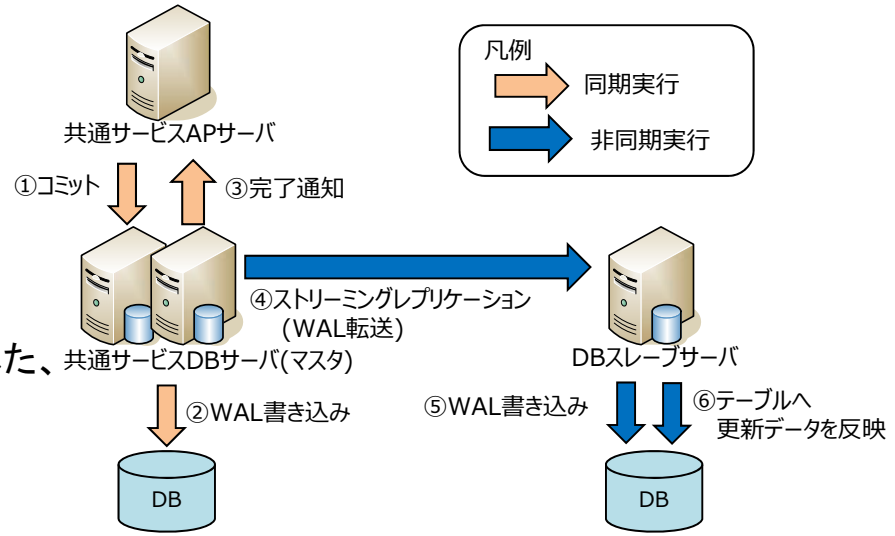
高可用性設計

- HeartbeatとDRBDによるHAクラスタ構成
  - DRBDを利用したブロックデバイスレベルのデータレプリケーション
  - 同期モードでレプリケーションしているので、フェイルオーバー発生時もデータ欠損の可能性は低い
  - Heartbeat/Pacemakerにより数分以内に自動フェイルオーバー
  - HAクラスタには、共有ストレージを利用する方法もあるがOSSのDRBDを採用
    DRBDのメリットは、単一点障害の排除とスプリットブレインへの対処がシンプルになること
- 通信経路の二重化
  - サービスLAN, サーバ間LAN、インターコネクトは、すべて
    ボンディングで二重化
  - 異なるNIC同士でボンディングし、接続先のスイッチも分離
- 本構成で保護される障害
  - サーバのハードウェア障害
  - OSの全面的なクラッシュ、ハングアップ
  - PostgreSQLのクラッシュ、ハングアップ
  - ファイルシステムレベルのデータ破壊
- 保護されない障害
  - DRBDやPacemakerに起因する、クラスタ単位の障害
    - スレーブサーバで対応
  - 誤操作による論理破壊
    - スレーブサーバのベースバックアップ＋トランザクションログで対応
  - OSやPostgreSQLの部分的なハングアップ（障害を検知できない）



バックアップ設計

- ストリーミングレプリケーションによるバックアップ
  - 非同期のリアルタイムバックアップ
  - スレーブは読み取り専用として利用可能
- ベースバックアップ＋トランザクションログのオンラインバックアップ
  - ポイント・イン・タイム・リカバリ（PITR)機能を利用した、任意の一時点を指定した回復が可能



取得バックアップ	格納先	取得周期	保持世代	リカバリ時のデータ	概要
ストリーミングレプリケーション	スレーブサーバ	数秒（非同期）	ー	数秒程度欠損の可能性あり	DB領域の非同期レプリケーションデータ。物理データ破損によるDB起動不可状態となった時などに有効。
ベースバックアップ	同上	毎週日曜 03:00～※	2世代	クラスタサーバ上のカレントWALにアクセスできれば最新状態まで復旧可能。クラスタサーバにアクセスできないときは、スレーブサーバ上のWALの範囲でリカバリ。	データベースのバックアップ。WALログと組み合わせることで、障害発生直前や任意の時点に戻すことが可能（PITR）。人為的なミスでDBのデータを削除した時などに有効。
WALログ	同上	不定期	ー		トランザクションログ。DBサーバ上に保持できなくなった古いWALログがスレーブサーバへ自動的に転送される。

性能・拡張性設計

- 超高速SSDによる超高速アクセス
  - 通常のHDDと比べて数倍から数百倍。とくにランダムアクセスが速い。
  - データ同期用のインターコネクトに10GbEを採用し、SSDのパフォーマンスを最大限に確保。
- コストと性能のバランスがよい6コア×2=12コアサーバを採用
  - HA構成では商用製品のようなノード追加によるスケールアウトができないため、事前キャパシティの確保。
- ディスクの拡張性
  - SSDを最大2本追加可能（現時点で1本の最大サイズは2.4TB）
  - HDDは6本増設可能
- メモリ増設可能

- スケールアウトについて
  - pgpool-Ⅱを利用すれば、参照に限りスケールアウト型の拡張ができるが、以下の理由により採用しなかった。
    - 複雑さが増加することによって障害発生率が増加する可能性があること。
    - pgpool-Ⅱでは、使用できるSQLに制限事項があること。
    - SQL処理も異なるため、アプリケーションのテスト工数が増加する可能性があること。
    - ノード追加によるスケールアウトは行わなくても、十分なパフォーマンスキャパシティを確保したこと。
    - 今後pgpool-Ⅱを追加することも可能なこと。

監視設計

- Zabbixによる障害監視を行う
  - PostgreSQL死活監視
  - DBディスク領域監視
  - DRBD監視（ログ、コマンド）
  - Heartbeat/Pacemaker監視（ログ、プロセス）

- pg_statsinfo/pg_stats_reporterを利用して、性能/状態を監視する
  - 全サーバにpg_statsinfoを導入
  - スレーブサーバにpg_statsinfo/pg_stats_reporterを導入

運用業務の準備(アセット化)

共通基盤構築プロジェクトと並行して、**設計の妥当性の調査**、及び、**運用業務の最適化/準備(トラブルの未然防止/早期解決)**、**社内外における構築案件の工数短縮/品質向上**を目的として、アセット化を実施しています。

サポートベンダーも居ますが、商用製品とは異なり、自身で調査/アセット化することも重要だと考えています。

アセット化項目 1/1

項	項目	実施目的	成果物
1	高可用性構成PostgreSQLデザインパターン作成	高可用性構成のPostgreSQLシステムには、さまざまな構成方法があり、選択が難しい。現時点で推奨できる、いくつかのパターンを選定し、それぞれのメリット・デメリットを含め、選択基準となる指針を作成する。	・「高可用性PostgreSQLデザインパターン」
2	高可用性PostgreSQL標準パラメータシート	インテリジェンスおよび対外顧客に対し、高可用性構成のPostgreSQLを導入する工数を削減するためパラメータシートを作成する。 範囲はPostgreSQL, Linux , Heartbeat/Pacemaker, pgpool-IIまで。	・「高可用性PostgreSQLパラメータシート」作成 – PostgreSQLヒアリングシート – HAクラスタパラメータシート – OSパラメータシート – PostgreSQLパラメータシート – pgpool-II導入手順書 – pgpool-IIパラメータシート
3	周辺ツール調査	PostgreSQLには、数多くの追加モジュール（extension）や周辺ツールがあり、これらを利用することで、運用や開発生産性が向上する。代表的なツールを調査し、推奨のものを選抜する。	・「PostgreSQL関連ツール」一覧
4	PostgreSQL運用ガイド作成	pg_statsinfo, pg_stats_reporterを利用すると、PostgreSQLの稼働情報が取得でき、またWebでグラフィカルに表示できる。これらを利用した運用ガイドを作成する。	・「pg_statsinfo, pg_stats_reporterセットアップガイド」作成 ・「PostgreSQL運用ガイド」作成 – PostgreSQLトラブルシューティングガイド – PostgreSQLアップグレードガイド – pg_statsinfo分析ガイド – pg_stats_infoパラメータシート

アセット化項目 2/2

項	項目	実施目的	成果物
5	共通基盤ベンチマーク	共通基盤で導入したPostgreSQLで、DBおよびDISKのベンチマークを実施し、今後のサイジング参考資料とする。	・「PostgreSQLベンチマーク報告書」
6	PostgreSQL 監視テンプレート	PostgreSQLが稼働するサーバ向けのZabbixテンプレートを作成し、監視実装の効率化を図る。	・Zabbixテンプレート for PostgreSQL ・PostgreSQL監視パラメータシート
7	PostgreSQL セキュリティ	PostgreSQLを利用するにあたり、最低限のセキュリティを実装するためのガイドライン。 （主にインフラ部分）	・「PostgreSQLセキュリティガイドライン」
8	検証環境構築	今後の案件やテストのために検証環境を構築する。	・社内開発VM上にテスト環境構築 － pgpoolプール － HAクラスタ

Section 1: IBS会社概要

Section 2: IBS戦略の考え方

Section 3: 担当業務概要/構成と講演の範囲

Section 4: OSS-DB検討の経緯(導入決定までの検討)

Section 5: PostgreSQLの構成検討/設計

Section 6: 運用業務の準備(アセット化)

Section 7: 保守運用フェーズ

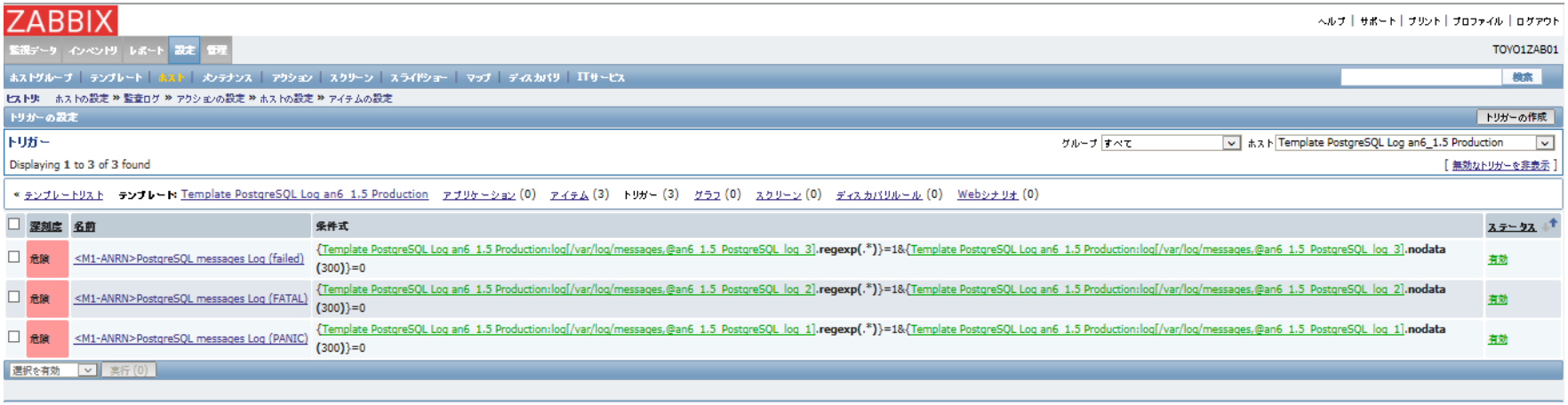
Section 8: まとめ

保守運用フェーズの状況

現在、本番運用を開始しておりますが、性能問題/障害等はほとんど発生していません。
大規模な障害は皆無です。(商用製品より安定しているかも･･･)

調査/アセット化をしっかりと行えた事に起因すると考えています。
本番運用開始後に、手を入れた部分は、一点だけです。

【課題】
ログ監視をした際に、ひとつの障害でZABBIXのアラートが大量に出力されることがあることに気づきました。
多いときには1000単位でくることがあります。(時にメールサーバのリソースを圧迫する可能性がある)
【対策】
ZABBIXで”nodate”という関数を利用して、『検知した同一メッセージを5分間再通知しない』設定を追加しました。



インテリジェンスがPostgreSQL採用した理由

項	項目	内容	備考
1	機能/性能向上が著しい	速度面でMySQLに大きな差をつけられていたが、とくに9.x以降のパフォーマンス向上が著しい。 また、機能も豊富で地理空間情報を扱えるものまである。	
2	PostgreSQLを採用サービスが増加	Heroku, Engine Yard, Chefなどの有名サービスでも採用。Amazon RDSでもサポート。	
3	事例/ナレッジが増加	日米企業の主流は今でもMySQLだが、近年はPostgreSQLの採用も増加し、事例/ナレッジが増加してきた。	
4	商用製品と近い使用感	商用製品と近い使用感を得られ、 特に業務アプリケーションで利用する機能は基本的に満たされており、移行も比較的容易にできる可能性が高い。	
5	MySQLは特有の知識が必要	SQLなどに制限があり、特有の知識が必要になる場合がある。	
6	コスト削減が見込める	ライセンス費用が不要であり、SI費用等も移行に伴うプログラム開発を除いては、商用製品と同等に抑えられる可能性が高い。 更に、保守サービスの費用も圧倒的に抑えられる可能性が高い。	
7	保守サービスも充実	保守サービスを行っているベンダーも多く、またパッチや機能開発を行っているベンダーもあり、品質も高い。	
8	技術者の増加	試験制度が確立したこと、セミナーや教育研修などが充実してきたこともあり、技術者が増加している。	
9	MySQLは、今後について注意が必要	CentOS7などでは、MariaDBを採用した。 また、多くの開発者がMariaDBへ移行している様子が伺える。	

PostgreSQLの課題

項	項目	内容	備考
1	標準となる可用性構成がない	現在は、高可用性ツールは、次のようなツール等が乱立している状態で、標準ストリーミングレプリケーション, pgpool-II, Slony-I, Bucardo, Heartbeat/Pacemaker, Red Hat Cluster, Lifekeeper, etc	複雑になってしまう。
2	情報の取得にはコツが必要	製品マニュアル等は、OSSとは思えないほど早く日本語化されるが、基準としているバージョンには気を付ける必要がある。 Webへの情報は充実しているが、日本語書籍は、まだまだ少ない。 ちなみに、MySQLは英文書籍の充実ぶりはすごい。	
3	外部ツールの利用と保守フェーズの工数/難易度のトレードオフ	商用製品と同等、または、同等に近づけるためには、外部ツール等の利用が必要なものが多い。 色々な人が色々なものを作って良くなっていくことはOSSの良い面でもあるが、外部ツールであるが故、バージョン/組み合わせ/ツール自体管理が複雑になってしまう。 例えば、統合化されたGUI管理ツールは、外部ツールとして提供されてきている。また、バックアップツールも同様で、増分バックアップなどができる機能は外部ツールの利用が必要になる。	
4	メジャーアップグレードは、手間が掛かる	マイナーアップグレードは、比較的容易に行うことができるが、メジャーアップグレードは難易度が高い。	何でもそうです。
5	バージョンアップが頻繁にある	OSS全般に言えることだが、バージョンアップが多い。 PostgreSQLの場合、メジャーバージョンアップが年1回程度、 マイナーバージョンアップが、2～3ヶ月に1回程度ある。	
6	大規模系の機能が弱い	大規模検索系の機能が弱い（パラレルクエリ（無）、マテリアライズド・ビュー/パーティショニング←あるが貧弱、ビットマップインデックス）	

まとめ

結論、PostgreSQLを
基幹システムに利用することは可能！

しかし、商用製品と比較して変化が激しく、PostgreSQLとその周辺ツールの調査、及び、動向を追う必要がある。
慎重に調査した上で導入しなければ、思うような動作をさせられず、誰も保障してくれない状態に陥る可能性がある。

「信頼性や性能、安心感をお金で買うか、
自分たちの創意工夫や努力でがんばるか」

→ IBSは、インテリジェンスに導入済みであり、ここを真剣に取り組んでおります。ご相談をお待ちしております。
また、社内の他の商用DBも随時リプレースする検討を行っています。

Appendix

プロジェクト時の調査内容など

高可用性デザイン

PostgreSQLで構築可能な構成

社内の要件/業務内容/技術スキルなどを踏まえ、現時点で最適と考えられる構成を示す。これをベースに新規構築する。

コールドスタンバイ構成

Active/Standbyの2台で構成

・コールドスタンバイ方式

HAクラスタ構成（Active/Standby）

Activeサーバで障害発生時、障害を検知し、Standbyサーバにフェイルオーバーすることで運用を継続

・共有ストレージ方式
・ストレージレプリケーション方式（ミラー）

データベースレプリケーション

複数のサーバに、データを複製することで冗長性を確保

・ログシッピングレプリケーション
　- 同期、非同期
・その他レプリケーション

マルチマスタ負荷分散クラスタ

複数のサーバーにまたがってデータを保持し、各サーバで平行に処理することで負荷分散を実現

・シェアードエブリシング
・シェアードナッシング

PostgreSQLで構築可能な構成

コールドスタンバイ構成

Active/Standbyの2台で構成

・コールドスタンバイ方式

HAクラスタ構成（Active/Standby）

Activeサーバで障害発生時、障害を検知し、Standbyサーバにフェイルオーバーすることで運用を継続

・共有ストレージ方式
・ストレージレプリケーション方式（ミラー）

データベースレプリケーション

複数のサーバに、データを複製することで冗長性を確保

・ログシッピングレプリケーション
　- 同期、非同期
・その他レプリケーション

マルチマスタ負荷分散クラスタ

複数のサーバにまたがってデータを保持し、各サーバで並行処理することで負荷分散を実現

・シェアードエブリシング
・シェアードナッシング

構成(1) オンラインバックアップとWALアーカイブ

【説明】

- スタンバイ機を用意して、定期的にデータベース全体のオンラインバックアップを取得。トランザクションログも随時バックアップする。障害発生時は待機系サーバでリカバリが終わるまで停止。特別な機能を使わない、もっともシンプルな構成。稼働率95～99%程度

【実装のポイント】

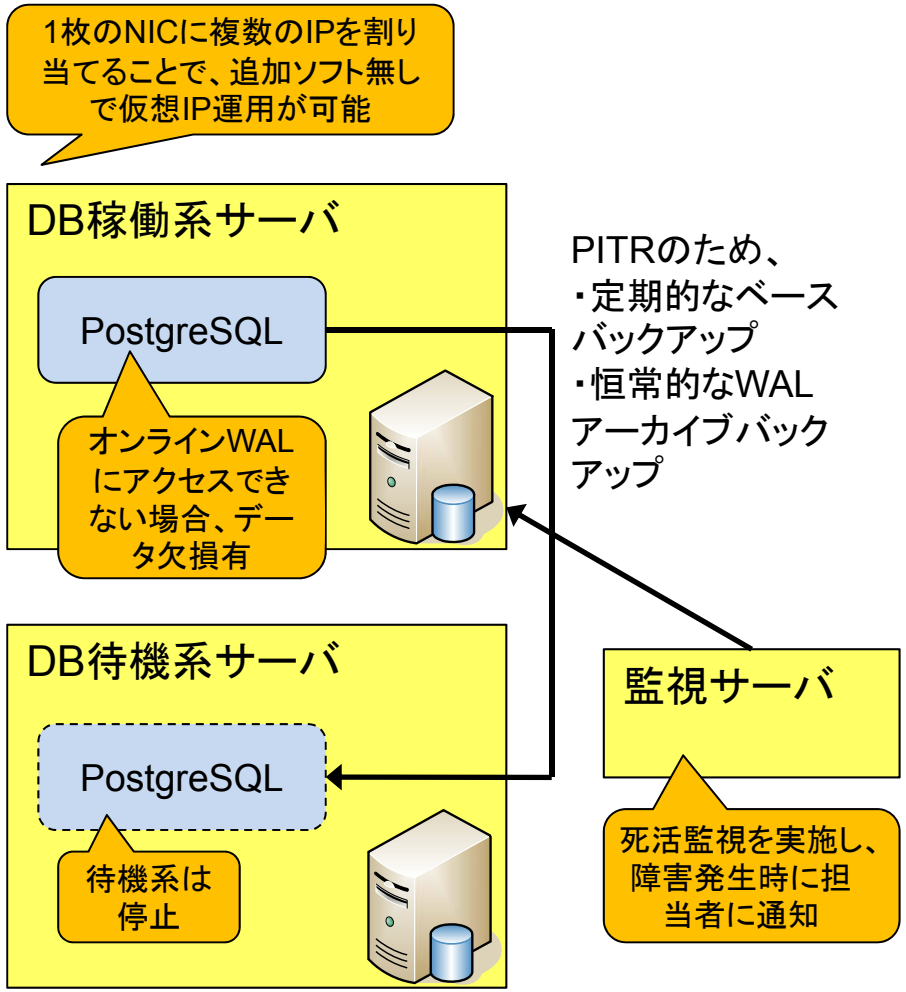
- PostgreSQLの標準機能とベースバックアップを取得するシェルスクリプトやcronを組み合わせる
- 監視ツールは本格的なものが望ましいが、シェルスクリプトやmonitなどの簡易ツールでも可能

【メリット】

- 高度な機能を使っていないので、設定・運用が簡単。リカバリ操作もシンプル

【注意点】

- リカバリにかかる時間の目安は、適用するWALアーカイブの個数×1秒と、それ以外の作業時間。
- オンラインWALにアクセスできないときは、データ欠損あり。
- ベースバックアップ時には稼働系サーバに負荷がかかる。
- サーバ間接続が1GbEの場合、実行転送速度は300Gbyte/h程度なので、テラクラスの場合、バックアップにそれなりの時間がかかる。



PostgreSQLで構築可能な構成

コールドスタンバイ構成

Active/Standbyの2台で構成

・コールドスタンバイ方式

HAクラスタ構成（Active/Standby）

Activeサーバで障害発生時、障害を検知し、Standbyサーバにフェイルオーバーすることで運用を継続

・共有ストレージ方式
・ストレージレプリケーション方式（ミラー）

データベースレプリケーション

複数のサーバに、データを複製することで冗長性を確保

・ログシッピングレプリケーション
　- 同期、非同期
・その他レプリケーション

マルチマスタ負荷分散クラスタ

複数のサーバにまたがってデータを保持し、各サーバで並行処理することで負荷分散を実現

・シェアードエブリシング
・シェアードナッシング

構成(2) 共有ディスクを利用したHAクラスタ

【説明】

- 共有ディスクを利用したHAクラスタ。障害を検知して数分で自動切り替え。HAクラスタとしてはもっとも歴史がある方式。稼働率～99.999%程度

【実装のポイント】

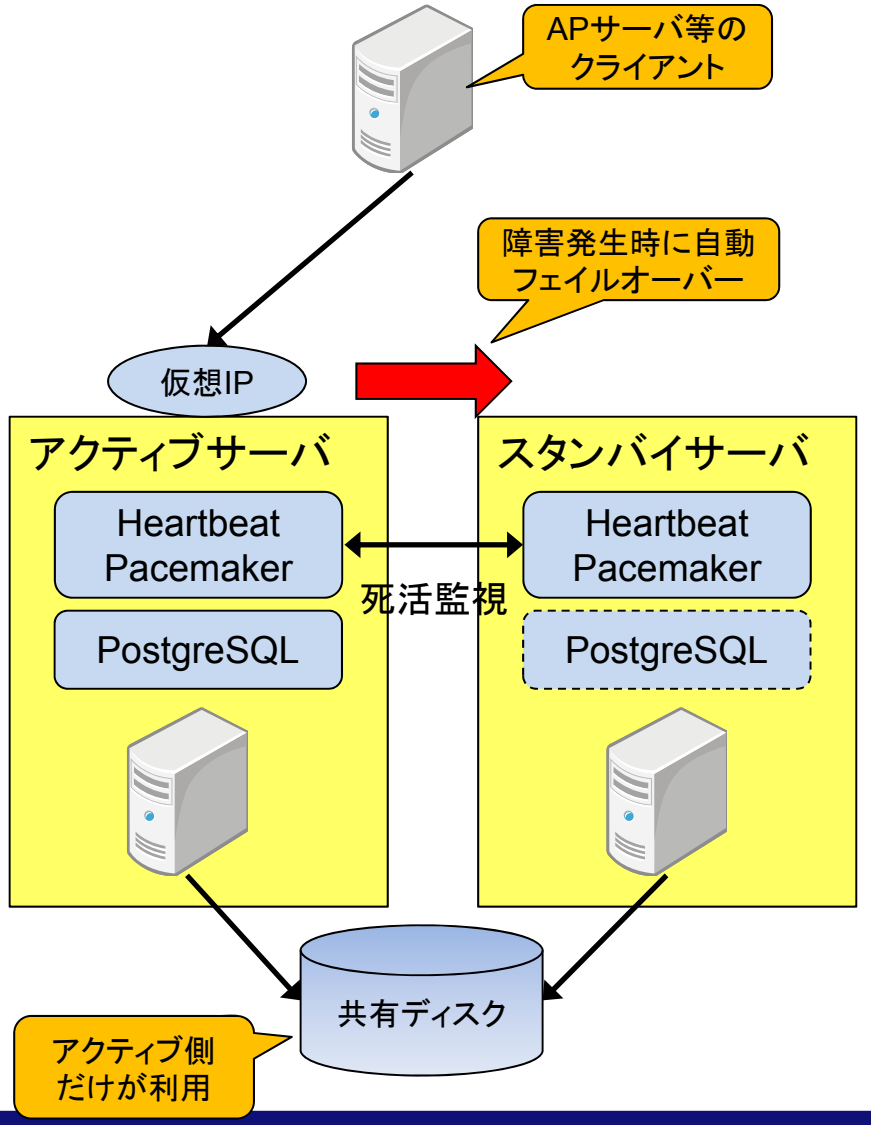
- LifekeeperやHeartbeat等のHAクラスタソフトを利用
- 共有ディスクはiSCSI, SAN, NAS等。ストレージとの経路はマルチパス等を使って二重化

【メリット】

- データ欠損可能性低。パフォーマンス低下なし
- HAクラスタとしてはもっとも歴史がある方式
- 高価なストレージの場合、ストレージスナップショットが使えるので、バックアップ運用が楽

【注意点】

- ある程度高額なストレージを使わないと、共有ディスクがシングルポイント
- スタンバイは待機だけ。複数DBがある場合には、たすき掛けのactive/activeが可能
- スプリットブレイン対策のSTONITHは、それなりに複雑
- ストレージスナップショットが使えない場合は、論理障害に対応するため、別途PITRによるバックアップを検討する



構成(3) DRBDを利用したHAクラスタ

推奨

【説明】

- DRBDなどのリモートディスクミラーを利用したHAクラスタ。障害を検知して数分で自動切り替え。 共有ディスクなどの特殊なハードウェアは不要。稼働率～99.999%程度

【実装のポイント】

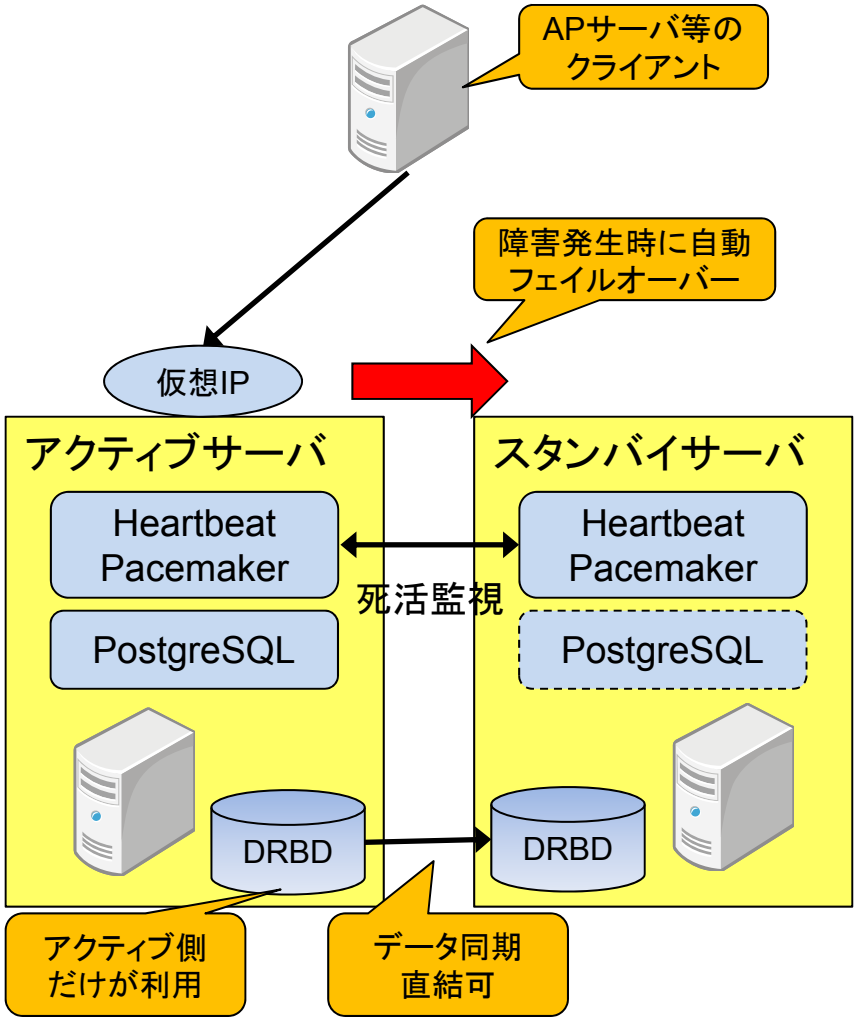
- DRBDやLifekeeperなどのディスクミラー機能を利用
- Heartbeat等のHAクラスタソフトを利用
- データ同期用のインターコネクトは最低でも1GbE

【メリット】

- 共有ディスクを使わないので、安価に構成可能。データ欠損の可能性低
- スプリットブレイン対策のdopdはSTONITHより簡単
- ストリーミングレプリケーション（非同期モード）を利用したスレーブサーバを追加して、さらに可用性向上・バックアップ対策を施すことも可能
- トラブルシュートがシンプル

【注意点】

- DRBDの場合、10%程度のオーバヘッド。また8.4.3以降を推奨。更新が多い場合は、インターコネクトがボトルネックになり、10%～50%以上遅くなることがある。
- スタンバイは待機だけ。複数DBがある場合には、たすき掛けのactive/activeが可能
- SSDを利用することで高速化が可能。その場合はインターコネクトに10GbEを推奨。更新がほとんどない場合は、1GbEでも可
- pgpool-IIを利用することもできるが、複雑化する可能性がある



PostgreSQLで構築可能な構成

コールドスタンバイ構成

Active/Standbyの2台で構成

・コールドスタンバイ方式

HAクラスタ構成（Active/Standby）

Activeサーバで障害発生時、障害を検知し、Standbyサーバにフェイルオーバーすることで運用を継続

・共有ストレージ方式
・ストレージレプリケーション方式（ミラー）

データベースレプリケーション

複数のサーバに、データを複製することで冗長性を確保

・ログシッピングレプリケーション
　- 同期、非同期
・その他レプリケーション

マルチマスタ負荷分散クラスタ

複数のサーバにまたがってデータを保持し、各サーバで並行処理することで負荷分散を実現

・シェアードエブリシング
・シェアードナッシング

構成(4) ストリーミングレプリケーションを利用したクラスタ

【説明】

- ストリーミングレプリケーションを利用したクラスタ。Heartbeatやpgpool-IIを利用して、マスタ・スレーブの自動切り替え可能。稼働率～99.999%程度

【実装のポイント】

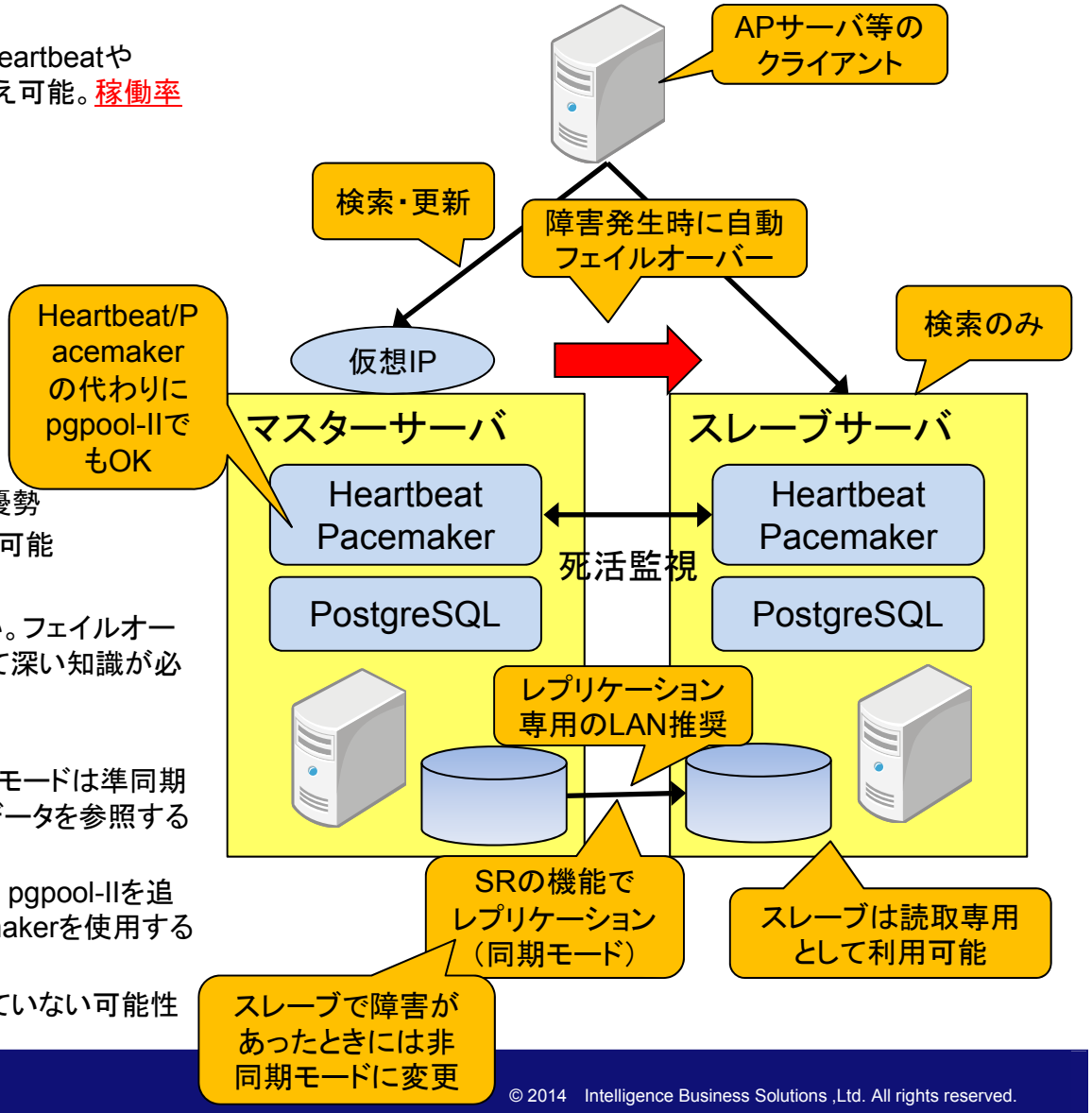
- PostgreSQLのストリーミングレプリケーション(9.1-)
- Heartbeat or pgpool-II等のクラスタソフト
- レプリケーション専用LANを推奨

【メリット】

- 共有ディスクを使わないので安価に構成可能
- スレーブを読み取り専用として公開可能
- DRBDよりオーバヘッド小
- 仮想環境やパブリッククラウドでは、DRBDより少し優勢
- pgpool-IIを組み合わせれば、容易にスケールアウト可能

【注意点】

- 運用やや難。DRBDのように簡単にスイッチできない。フェイルオーバーからのリカバリが面倒。レプリケーションに関して深い知識が必要。9.3以降を推奨
- DRBDよりスプリットブレイン対策が複雑
- スレーブは検索専用として利用できるが、SRの同期モードは準同期なので、マスターとの一貫性は保証されない。古いデータを参照する可能性がある
- スレーブサーバを読み取り専用として公開する場合、pgpool-IIを追加して自動的に振り分けるのが一般的だが、Pacemakerを使用する方式もある
- 同期モードであっても、参照データの一貫性が取れていない可能性がある



構成(4) ストリーミングレプリケーションを利用したクラスタ（補足）

- ストリーミングレプリケーションを利用したクラスタでは以下の2方式がある
  - pgpool-II
  - Heartbeat/Pacemaker

	pgpool-II（マスター・スレーブモード）	Heartbeat/Pacemaker
負荷分散	透過的な負荷分散	明示的な負荷分散
耐障害性 （スレーブノード障害の影響）	同期・非同期を自動的に切り替えないので、以下のいずれかに当てはまる場合にスレーブノード障害が全体の障害につながる ・同期スレーブがあり2ノード以下の構成 ・synchronous_standby_namesに指定したサーバがすべてダウンしたとき	同期・非同期を自動的に切り替えるので、スレーブ障害が全体の障害につながらない
SPOF	pgpool-IIを2重化する必要がある	なし
PostgreSQL以外のサービス	対象外	PostgreSQL以外のプログラムもHAの対象に組み込める
その他メリット・デメリット	・コネクションプール機能が使える ・非同期モードでも、同期遅延を監視した振り分けが可能 ・構成によってはサーバ数が増える（pgpool用） →同期モードが必要な、3ノード以上のレプリケーション	・1回目のフェイルオーバーはするが、予想外の動きがあるので、今後のノウハウ蓄積や各ソフトの機能向上（特にpacemakerのRA)の機能改良によって、変わってくる可能性はあるが、現時点で推奨はしない →2ノードのレプリケーション

PostgreSQLで構築可能な構成

コールドスタンバイ構成

Active/Standbyの2台で構成

・コールドスタンバイ方式

HAクラスタ構成（Active/Standby）

Activeサーバで障害発生時、障害を検知し、Standbyサーバにフェイルオーバーすることで運用を継続

・共有ストレージ方式
・ストレージレプリケーション方式（ミラー）

データベースレプリケーション

複数のサーバに、データを複製することで冗長性を確保

・ログシッピングレプリケーション
　- 同期、非同期
・その他レプリケーション

マルチマスタ負荷分散クラスタ

複数のサーバにまたがってデータを保持し、各サーバで並行処理することで負荷分散を実現

・シェアードエブリシング
・シェアードナッシング

構成(5) マルチマスタ負荷分散クラスタ（pgpool-II/シェアードナッシング）

【説明】

- pgpool-II配下の各PostgreSQLに対して、同一の参照・更新を送ることでデータを複製。参照は特定の1ノードへ、更新は全ノードに伝搬。稼働率～99.999%程度

【実装のポイント】

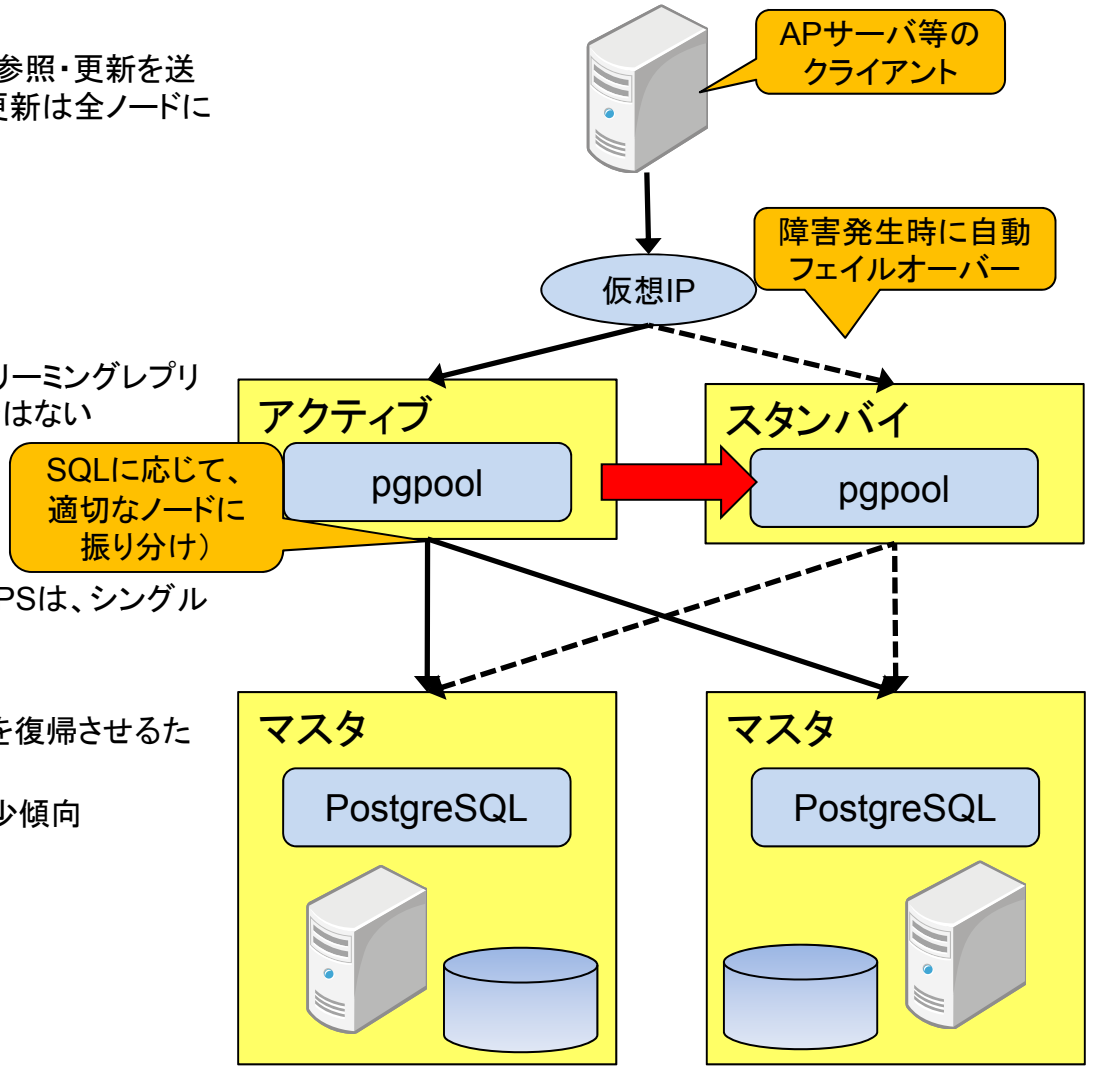
- pgpoolのレプリケーションモード

【メリット】

- 検索はスケールアウト可能
- すべてのノードでデータが同期しているため、ストリーミングレプリケーションのように古いデータを参照してしまうことはない
- 共有ディスクを使わないので安価に構成可能
- サーバの動的追加が可能

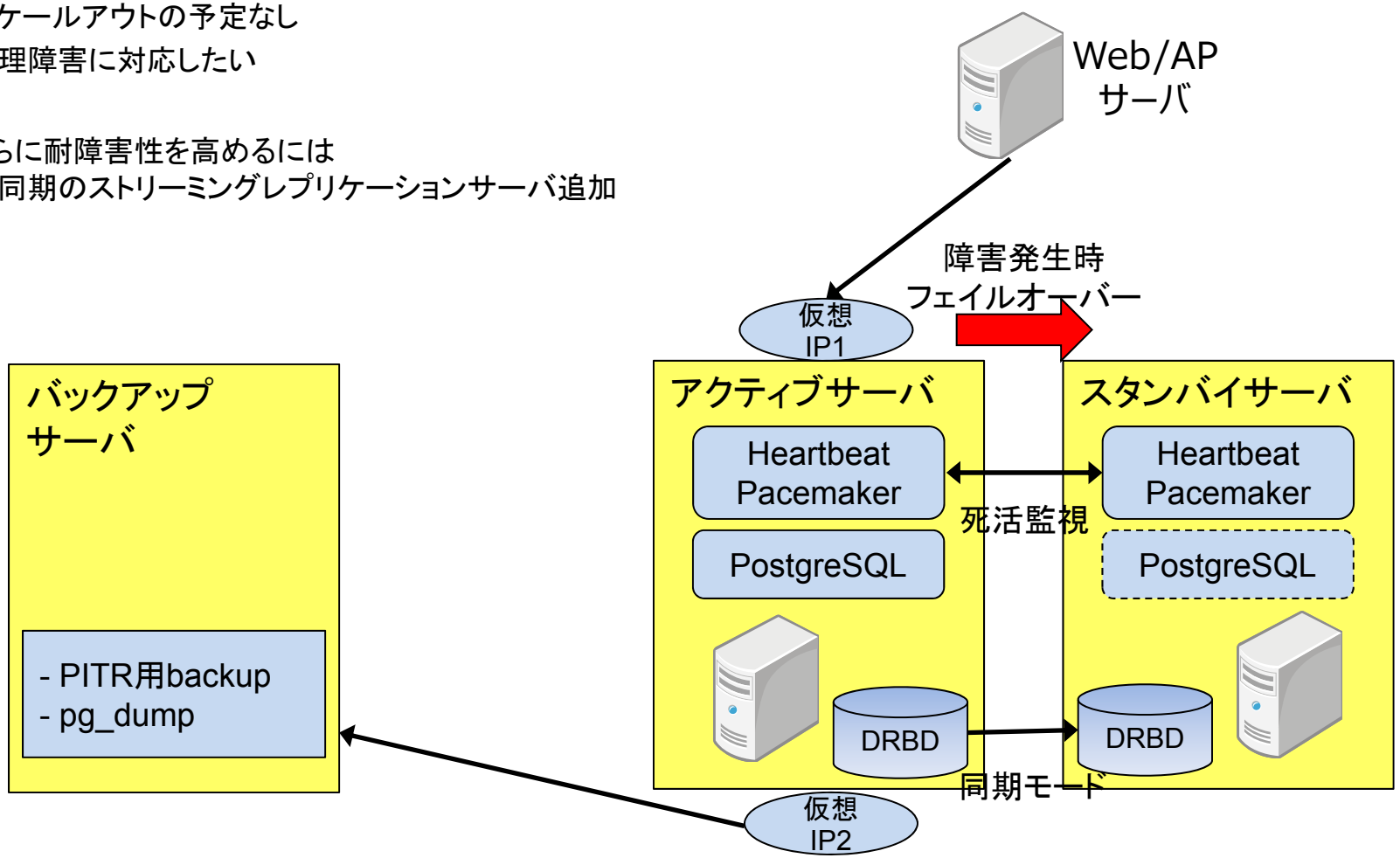
【注意点】

- 更新は遅い。ノードをいくら追加してもトータルのTPSは、シングルノードの1/2程度
- 利用できるSQLに制限がある
- 運用やや難。一度クラスタから除外されたサーバを復帰させるためには、ベースバックアップからのリカバリが必要
- ストリーミングレプリケーションの登場で利用は減少傾向



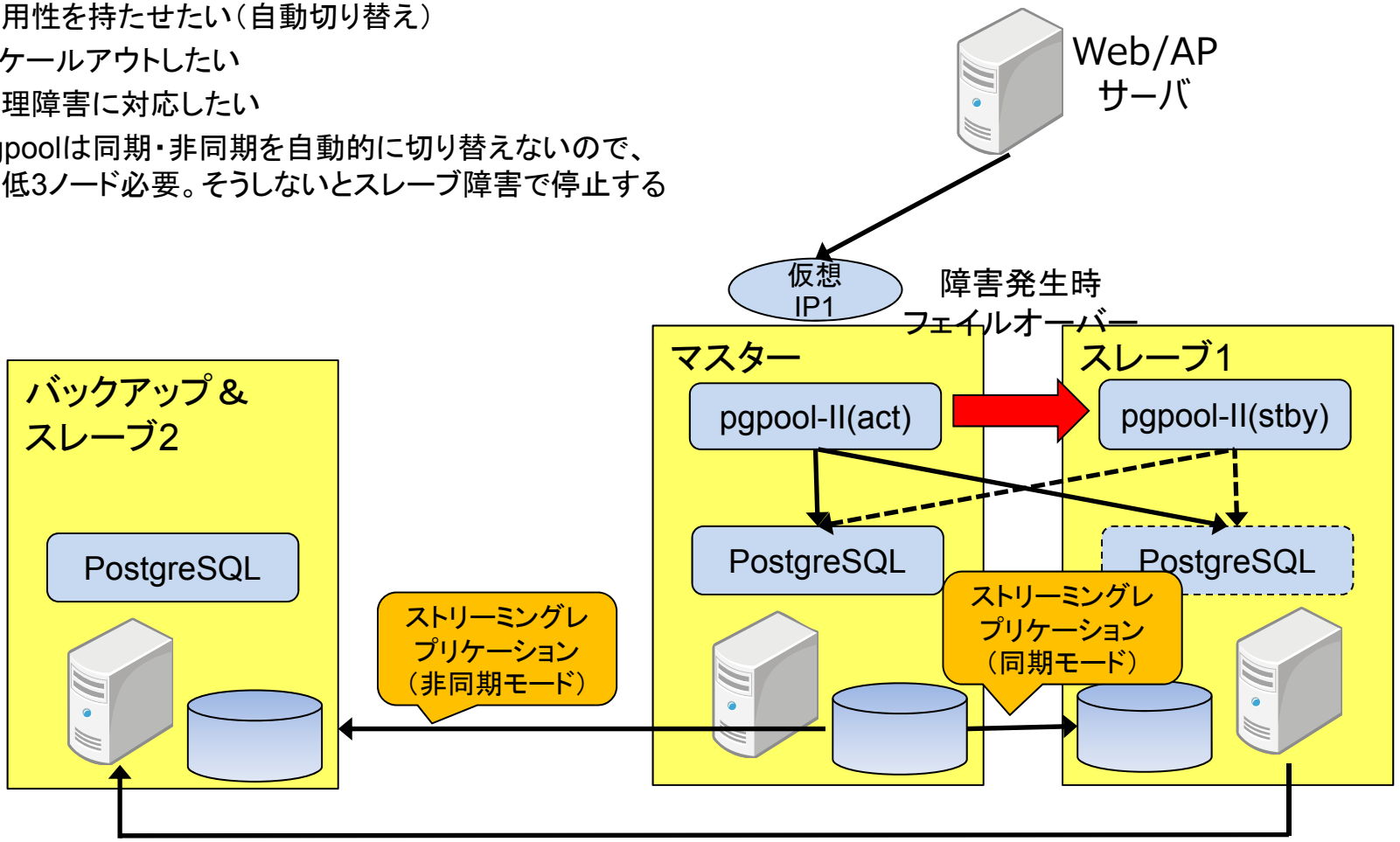
Case Study1: HAクラスタ（DRBD）＋バックアップサーバ

- 要件
  - 可用性を持たせたい（自動切り替え）
  - スケールアウトの予定なし
  - 論理障害に対応したい
- 補足
  - さらに耐障害性を高めるには
    非同期のストリーミングレプリケーションサーバ追加



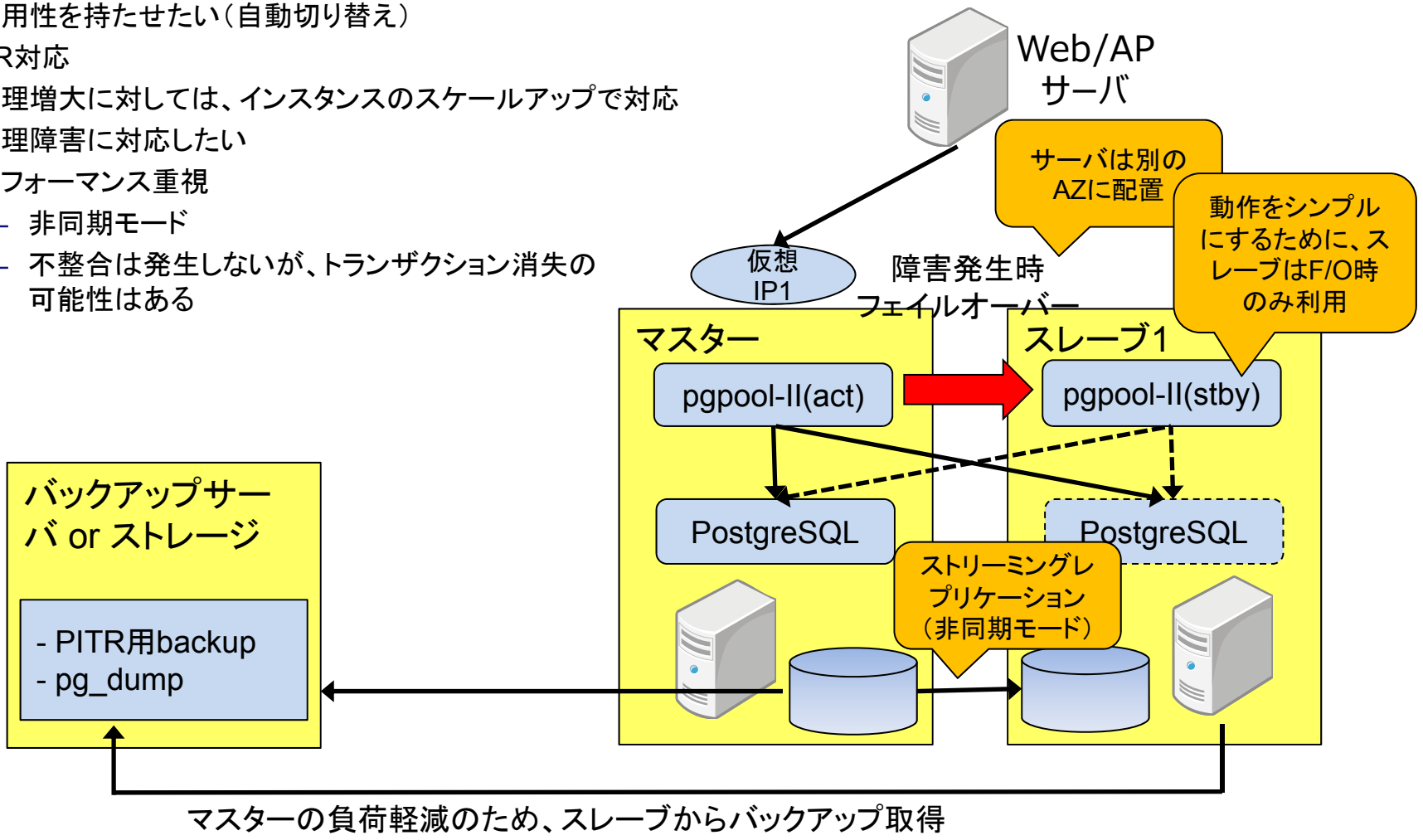
Case Study2: ストリーミングレプリケーション可用性重視

- 要件
  - 可用性を持たせたい（自動切り替え）
  - スケールアウトしたい
  - 論理障害に対応したい
  - pgpoolは同期・非同期を自動的に切り替えないので、最低3ノード必要。そうしないとスレーブ障害で停止する



Case Study3: ストリーミングレプリケーション（DR＆パフォーマンス重視）

- 要件
  - 可用性を持たせたい（自動切り替え）
  - DR対応
  - 処理増大に対しては、インスタンスのスケールアップで対応
  - 論理障害に対応したい
  - パフォーマンス重視
    - 非同期モード
    - 不整合は発生しないが、トランザクション消失の可能性はある



関連ツール/周辺ツール

関連ツール（contrib以外）

関連ツール/周辺ツールで、今回検討したものの一覧を以下に示す。

分類	プロダクト名	説明
開発	pgAdmin-III PostgreSQL Studio	GUI管理ツール
運用・監視	pg_statsinfo pg_stats_reporter	サーバーの動作状況を収集するツールと、その情報をWebで表示するツール
	check_postgres	本来はNagios用のスクリプト集だが、有用な関数が多数あるので、Nagiosを使っていなくても利用実績多
	pg_monz	Zabbix用PostgreSQL監視テンプレート
	pg_blkload	データローディングツール
	pg_reorg	テーブル再編成ツール
	pg_top	PostgreSQL版top。似たツールとしてpg_activityも
バックアップ	Barman	高性能バックアップツール
	pg_rman	簡易バックアップツール
	WAL-E	クラウド向けバックアップツール
チューニング	pgBadger	ログを解析し、SQLの実行やサーバの動作のレポートツール（pgFouineより高機能で速い）
	pgFincore	OSのディスクキャッシュに乗ったファイルを管理するツール

組み合わせて利用するソフトウェア

ツールというよりも、組み合わせて利用するソフト

分類		
レプリケーション	pgpool-II	SRA OSSが中心になって開発した、多機能ツール（レプリケーション、コネクションプーリング、パラレルクエリー、負荷分散）
	Slony-I	トリガーベースの非同期のレプリケーション。テーブルレベルで設定可能。メジャーバージョンが違ってもレプリケーションできるので、ローリングアップグレードが可能
	Bucardo	海外ではそれなりに使われているレプリケーションソフト
コネクションプーリング	pgpool-II	上記参照
	pgBouncer	海外では日本よりメジャーなコネクションプーリングツール

本番機器ベンチマーク

本番機器　共通DBサーバ システム構成(ディスク)

本番機器 共通DBサーバで使用しているディスクは以下の通り。

・SSD
　容量：1.2TB

・ハードディスク
- RAID10
- ServeRAID-M5110e(MegaRAID SAS 2208)、512MBキャッシュ
- SAS 300MB(10,000rpm/s)×10

利用ツール紹介

今回のベンチマークで利用したツールは以下の通り。

■iperf3

TCP/UDPのネットワーク最大帯域幅測定ツール。googleが作成。EPELリポジトリからRPMをダウンロード可能。
http://pkgs.repoforge.org/iperf/

■fio

Fusion-io社推奨のディスクIOベンチマークツール。EPELリポジトリからRPMをダウンロード可能。
http://pkgs.repoforge.org/fio/

■ORION

Oracle社が提供するディスクIOベンチマークツール。Oracle DatabaseのIO制御プログラムをベースにして作成されており、RDBMSの様々なワークロードパターンをシミュレートしてディスクIO性能を測定できる。

■pgbench

PostgreSQLにcontribとして付属する簡易なベンチマークツール。標準ベンチマークTPC-B(銀行口座、銀行支店、銀行窓口担当者などの業務をモデル化)を参考にしたシナリオが実行できる他、検索のみなどのシナリオも搭載されている。pgbenchでベンチマークを実行すると、1秒あたりで実行されたトラザクションの数(tps:transactions per second)が表示される。

ベンチマークによる検知事象

今回のベンチマークにて検知し、対処した事象を以下に記す。

【事象1】

〇事象内容

ベンチマーク実施中にフェイルオーバ発生。

〇原因

外部通信監視用リソースがタイムアウトしたため。

〇対処

外部通信監視用リソースがタイムアウト値を20秒から推奨値である60秒に変更。
変更後、ベンチマークによるフェイルオーバが発生しないことを確認。

【事象2】

〇事象内容

ベンチマーク実施中に以下のエラーが頻発。
FATAL: requested WAL segment XXXXXXXXXXXX has already been removed

〇原因

SR用に保持するWALの数の設定が0(デフォルト値)の状況で大量のWALが発生したことで、スレーブDB側へWALが反映される前にWALが削除されてしまい、ストリーミングレプリケーション(SR)できなくなったため。

〇対処

SR用に保持するWALの数を指定する「wal_keep_segments」のパラメータをデフォルトの0から100に変更。

ベンチマーク　まとめ (1/2)

ベンチマークの結果をもとに考察を加える。(条件によって大幅に異なる可能性がある)

- 考察
  - SSDがもっとも効果を発揮するのは、検索更新問わず、インデックススキャンが中心のアプリケーション。つまり一般的なWebアプリケーションやOLTPは非常に適しているといえる。
  - 10krpmのHDD＋RAID10は、全体のスピンドル台数にも依存するが、使い方によってはコストパフォーマンスがよいと言える。
  - バッチ処理のような大規模更新処理は、DRBDのデータ同期用インターコネクトがボトルネックになる。そのため、DRBDを構成している場合、SSDの高性能を生かせない可能性がある。DRBDを使わない構成も検討する。
  - SQLの内容にもよるが、同時接続数が多い場合、CPUのコア数を多くすることが有効。
  - shared_buffersの推奨値はメモリサイズの1/4とされているが、8GB以上にしても性能が向上しない。これはコミュニティでも指摘されていて、PostgreSQLのバッファキャッシュの効率がよくないことに起因していると思われる(OSキャッシュの仕組みを上回れない)。

    →PostgreSQLがバージョンアップされて改善される想定だが、メモリサイズが32GB以上でv9.xを使用する場合、shared_buffersは8GBに設定することを推奨する。
- 補足
  - 今回のベンチマークにはシーケンシャルvsランダムの比較は含まれていないが、SSDの場合、HDDのように物理的なヘッド移動が発生しないので、シーケンシャルとランダムの差は小さい。他社の結果を見ると、ほとんどない状況。
- 注意事項
  - 今回のfioベンチマークではOSキャッシュをバイパスしている。それに対してDBMSの場合、DBのキャッシュとOSキャッシュ機構(ORION除く)を利用しているので、今回の結果と異なる特性が出る可能性が高い。
  - PostgreSQLの場合は独自のバッファキャッシュに加えて、OSキャッシュも利用する。

ベンチマーク　まとめ (2/2)

- 注意事項
  - 今回の結果はあくまでも参考程度にとらえること。特に何倍といったことは、条件によって大幅に異なる。
  - 今回のfioベンチマークではOSキャッシュをバイパスしている。それに対してDBMSの場合、DBのキャッシュとOSキャッシュ機構（ORION除く）を利用しているので、今回の結果と異なる特性が出る可能性が高い。ディスクの基本性能差は縮まる方向へ。
  - PostgreSQLの場合は独自のバッファキャッシュに加えて、OSキャッシュも利用する。