内製SGエンジンの軌跡
～スクールガールストライカーズ～

【 サーバ編 】

発表者
高岡耕平

SQUARE ENIX

発表者の来歴

- **高岡耕平**
  - 68Kアセンブラで幼児向けゲーム作ったり
  - 格闘ゲームPS2®移植でD兼メインPGやったり
  - 業務系システムでSE/PGやったり

- 現在は……SQUARE ENIX 12BD M／TD（主にサーバ周り）
- 管理業務の傍ら、実際に手も動かします
- スクストでは**サーバPGリード**もしています

スクストって？

SQUARE ENIX

そもそもなぜ内製？

- **ソシャゲの品質が安定しない……**
  - 外部開発会社のスキルに依存する
  - 技術ノウハウが社内に残らない

- **ならばまるっと社内で作りましょう！**
  - **内製なので技術ノウハウ溜まる**
  - **社内で横展開もできる、社内標準目標**
  - 色々整備されたら外部開発会社への提供も？

その辺踏まえまして

- アジェンダは以下の二つ
- **内製サーバフレームワークのお話**
- **ソシャゲ開発そのもののお話**

内製サーバフレームワークのお話

SQUARE ENIX

内製サーバフレームワーク

- **サーバフレームワークも内製なのです！**
- 何故**既存のサーバフレームワーク**を使わなかったのか？
  - zendやらCakePHPやらFuelPHPやら……

- **ブラックボックス化を避ける**
- **ソシャゲ特化のミニマム実装**

SQUARE ENIX

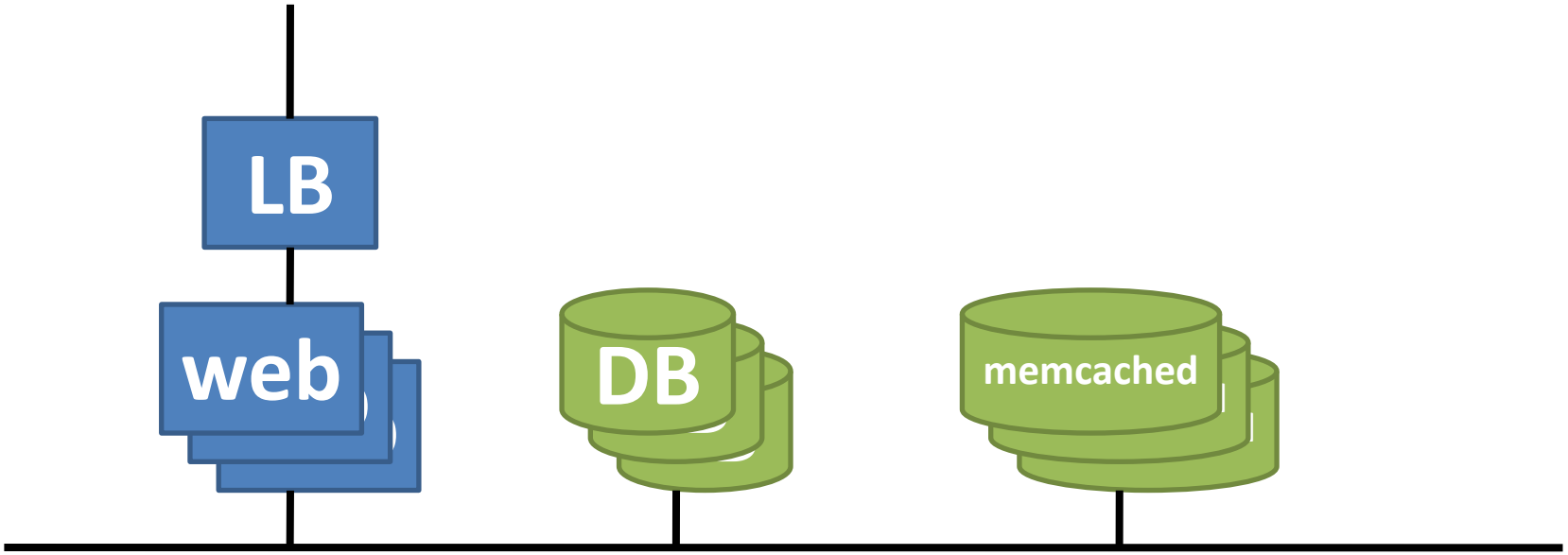
環境

- PHP　5.5系
- MySQL　5.6系
- kvs
  - memcached,apcu

- 殆ど枯れたものしか使っていません！
  - R&Dが目的ではない
  - 横展開しやすさ重視

構成

- いたってシンプル



となると

- 何が面白いのか？
- 正直なところ、技術的には目新しいことはないのです……
- しかしながら！

このフレームワークの売り

- 軽い！
- 作りやすい！
- セキュリティ硬い！

- とにかく実用性を重視したフレームワークになっています
- 地味でも堅実なノウハウの積み重ねです

どう実用的なの？

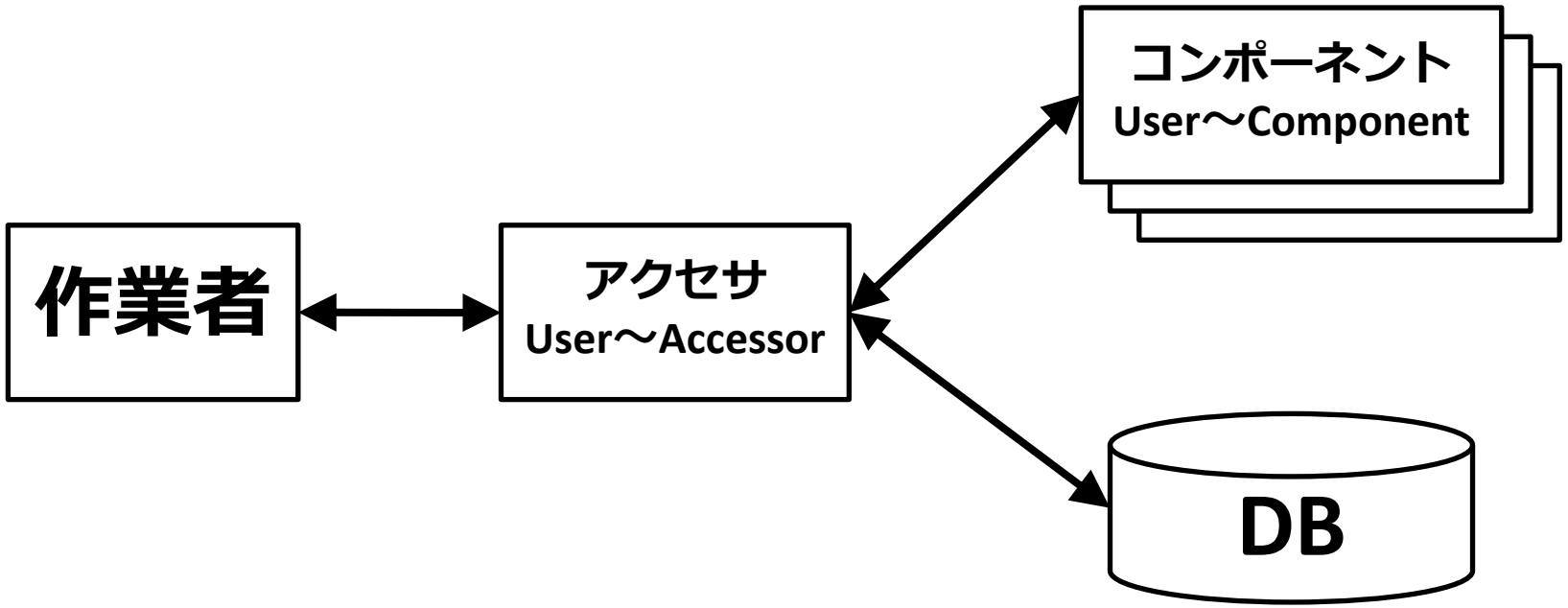
- 例えば**ユーザーデータ管理**

- **DB参照／更新頻度を最小限に抑える**
- **複雑なキャッシュ管理を行う**

- を、**作業者が意識せずに使えます**

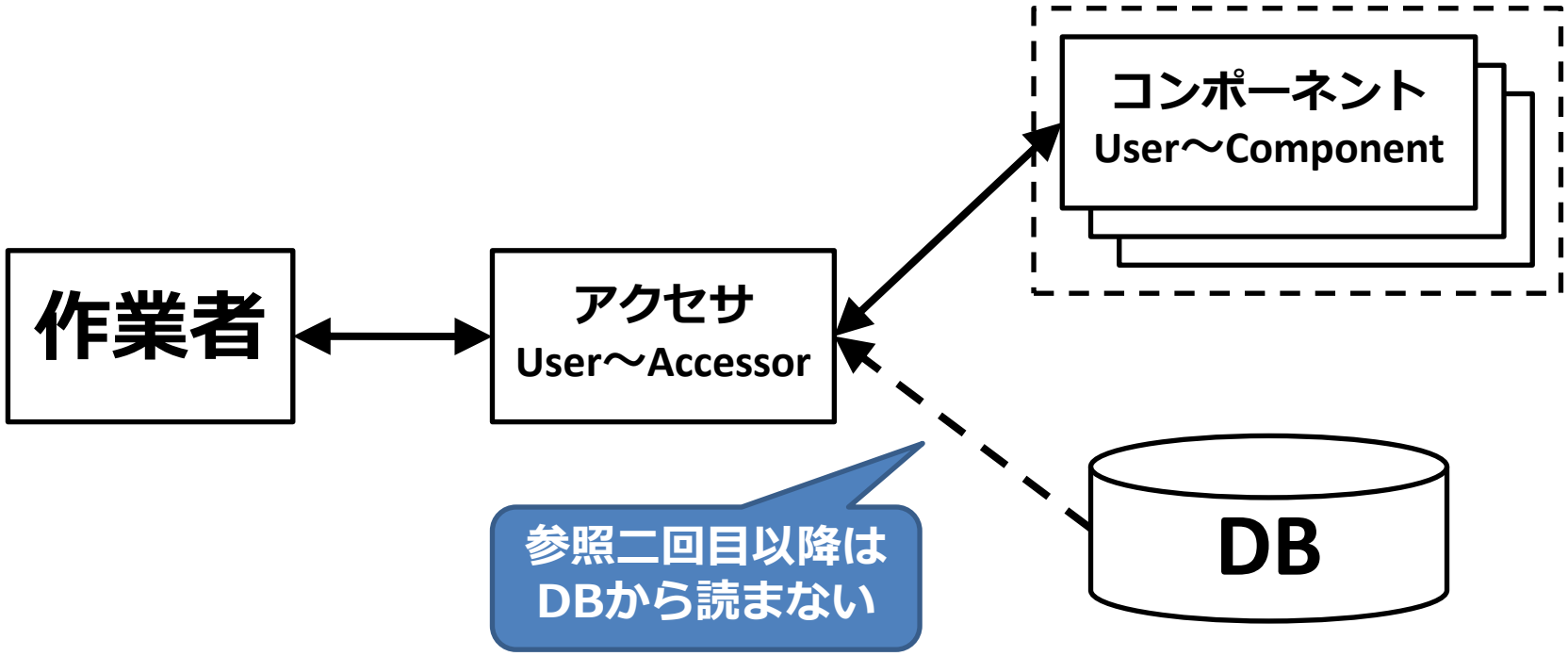
コンポーネント

- **特定のDBテーブルに対応したデータを管理**する



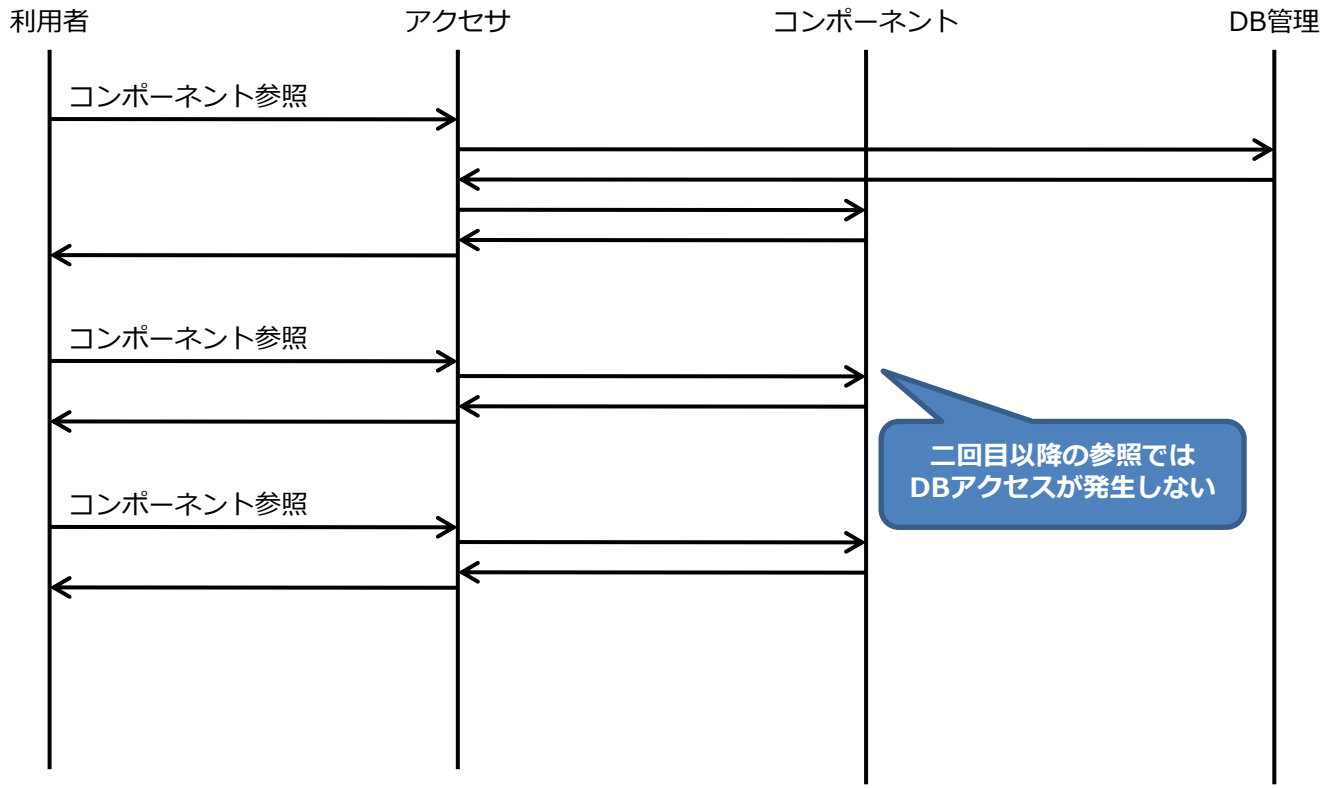
データの参照

- **キャッシュ**（ここでは単なるローカル領域を指す）**に置いてDB参照回数を節約**



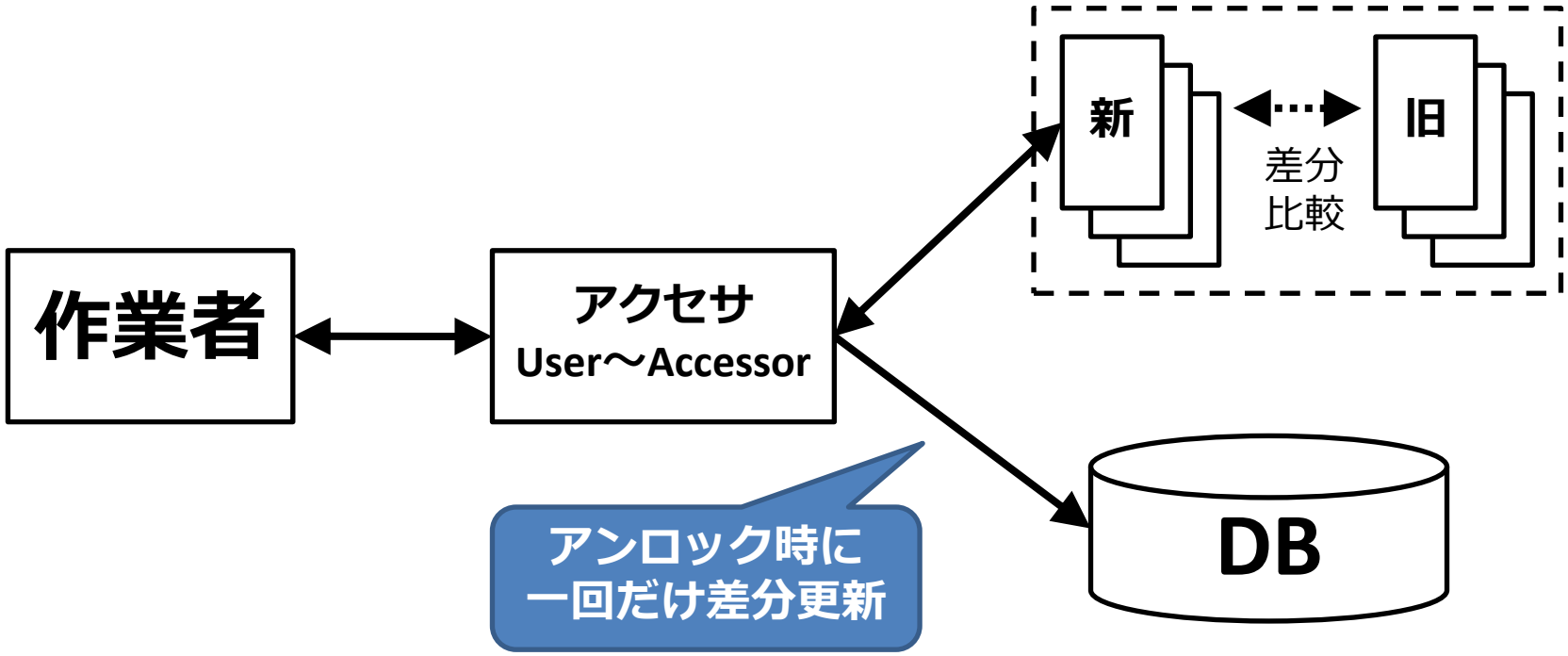
データの参照

- 流れはこんな感じ



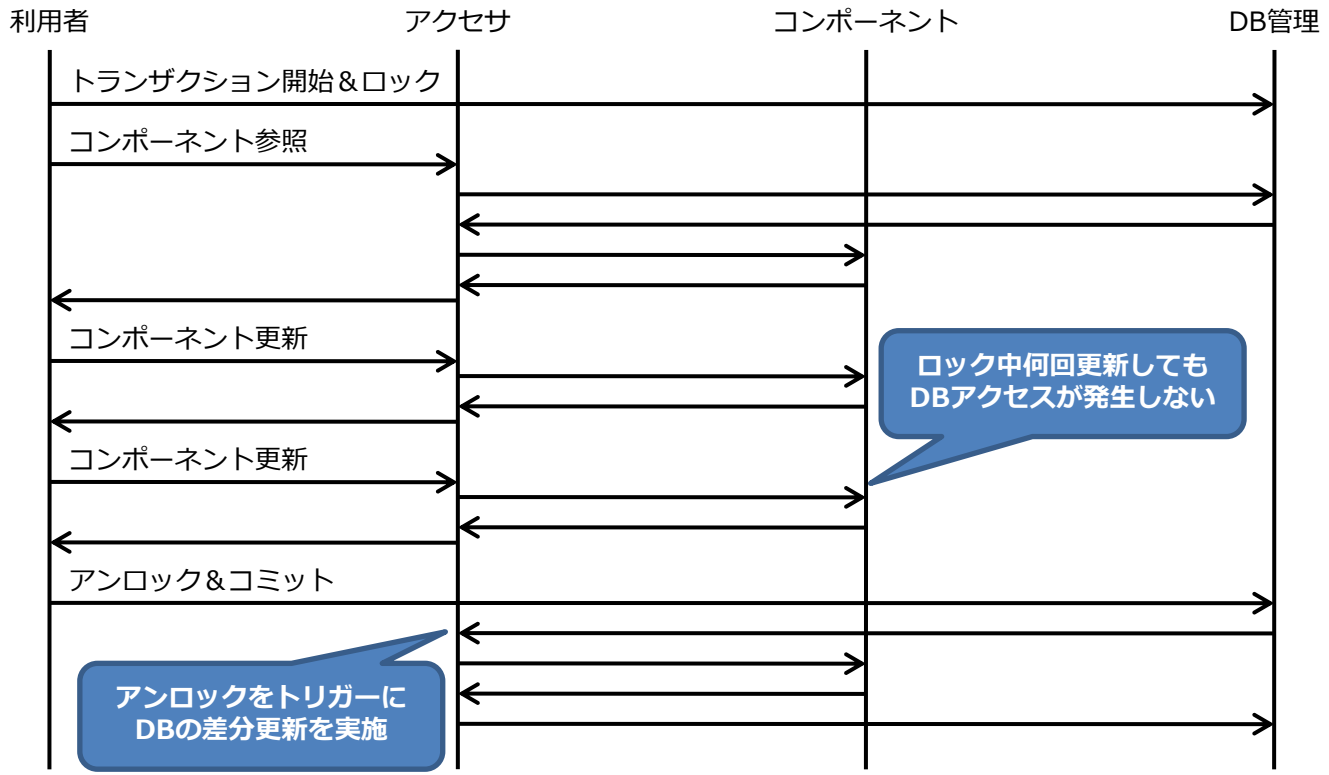
データの更新

- キャッシュに新旧コンポーネントを保持、アンロックをトリガーにして**差分更新**



データの更新

- 流れはこんな感じ



活用事例：ミッション

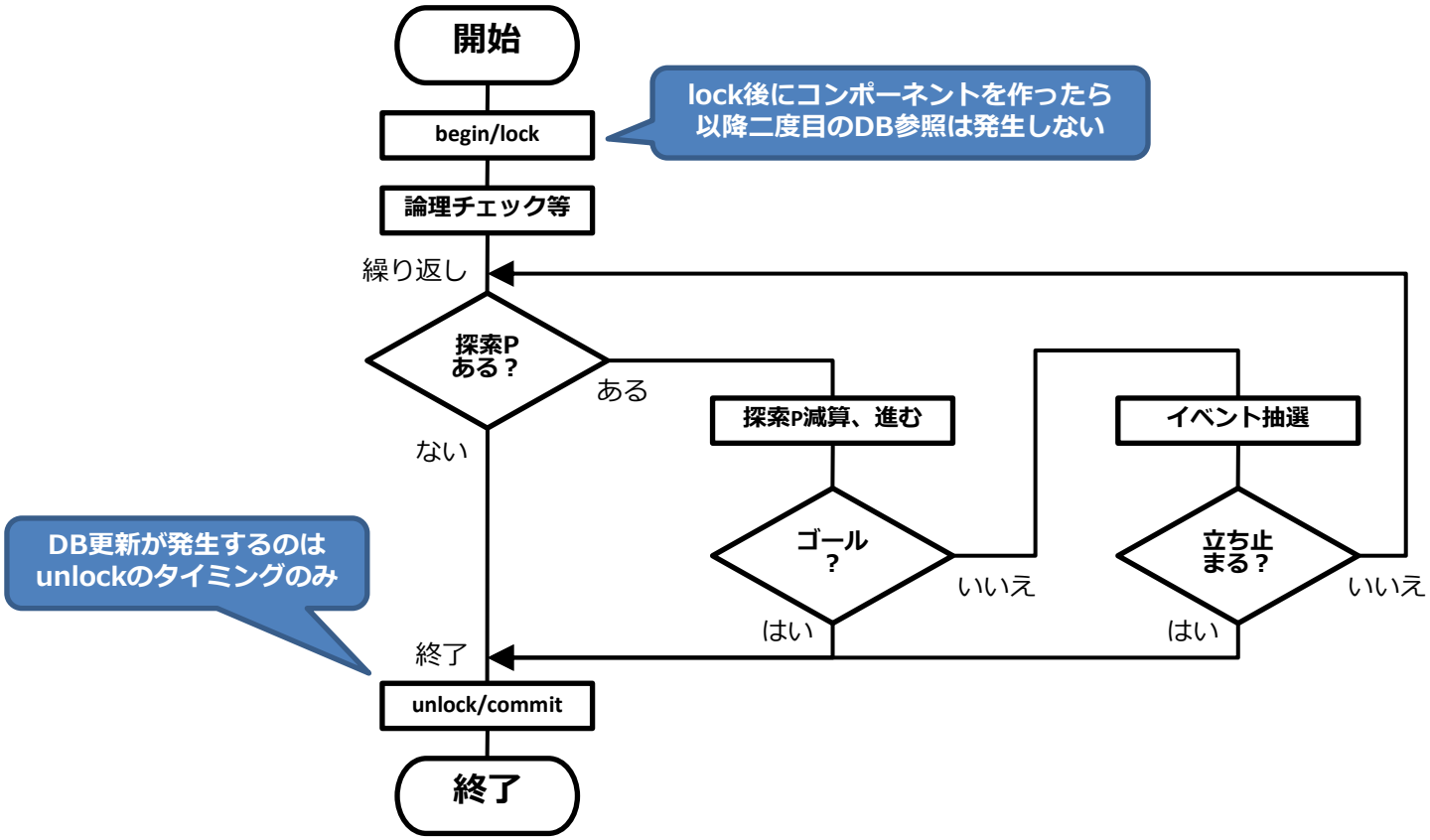
- 既に気付いている方も多いかとは思いますが、**スクストのミッション**はこんなロジックで動いています
  - **一回の進行命令通信で**
    - **探索ポイントが不足**
    - **立ち止まる必要のある状況／イベントが発生**
  - **のいずれかの状態になるまで歩き続ける**

活用事例：ミッション

- 真っ正直に実装すると、**DBの参照と更新を何度も繰り返してしまいます**
- キャッシュを活用する実装をすれば負荷を軽減できますが、**個別に実装していると工数や品質にばらつきが……**
- コンポーネントを活用することで、**作業者スキルへの依存度を下げつつ、より短い工数で、より高い品質を担保！**

活用事例：ミッション

- 単純な実装そのままでOK（一部簡略化



通信ストレス対策

- **通信ストレスはユーザビリティに直結**

- **対策の基本を突き詰めて対応**
  - **通信回数を減らす**
    - 処理の実行→データリスト取得　のような無駄な複数回通信は避け、一度の通信にまとめる
  - **通信容量を減らす**
    - 差分のみ返却し、クライアント上のローカルキャッシュの更新だけで済ます

他にも……

- ユーザー登録時のDBレコード挿入を最小限に抑える**コンポーネント初期値制御**
- 適切な範囲のロックを一元管理し、デッドロックを起こさない**ロック管理機構**
- 開発サーバを参照してローカルサーバを手軽に構築できる**開発支援機能**

- 等で作業効率や品質を高めています！

諸々の対策の成果

- **「クラウド費月額／当月最大DAU」**を指標にして、**各タイトル間の指標比率**で成果を測定します
- **スクスト**の指標を**1**とした場合
- **弊社の主要PCブラゲ／スマホソシャゲ**における指標は**3.1～13.8**

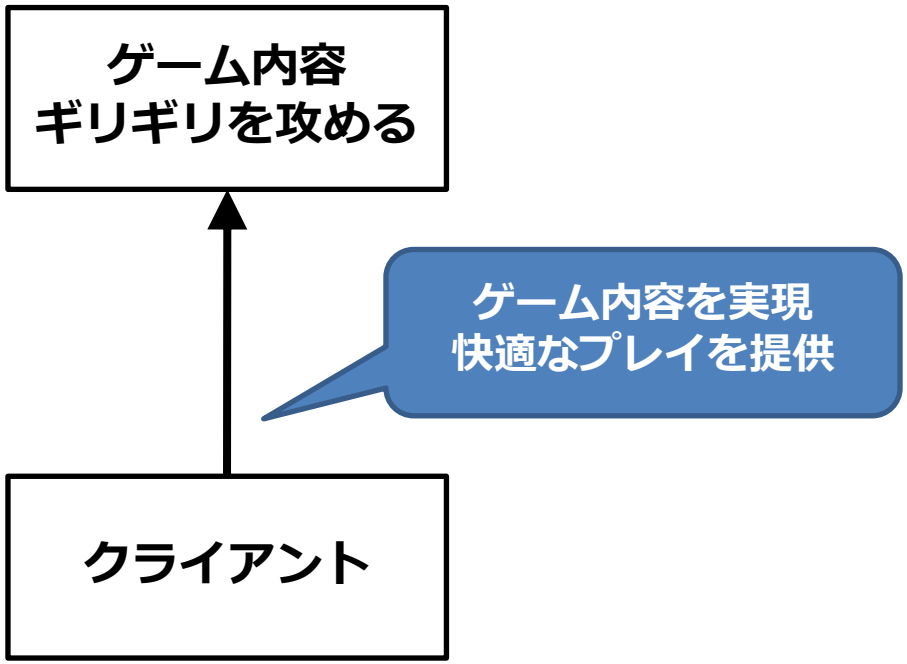
- **大幅な**クラウド費削減**を実現しています**

ソシャゲ開発そのもののお話

SQUARE ENIX

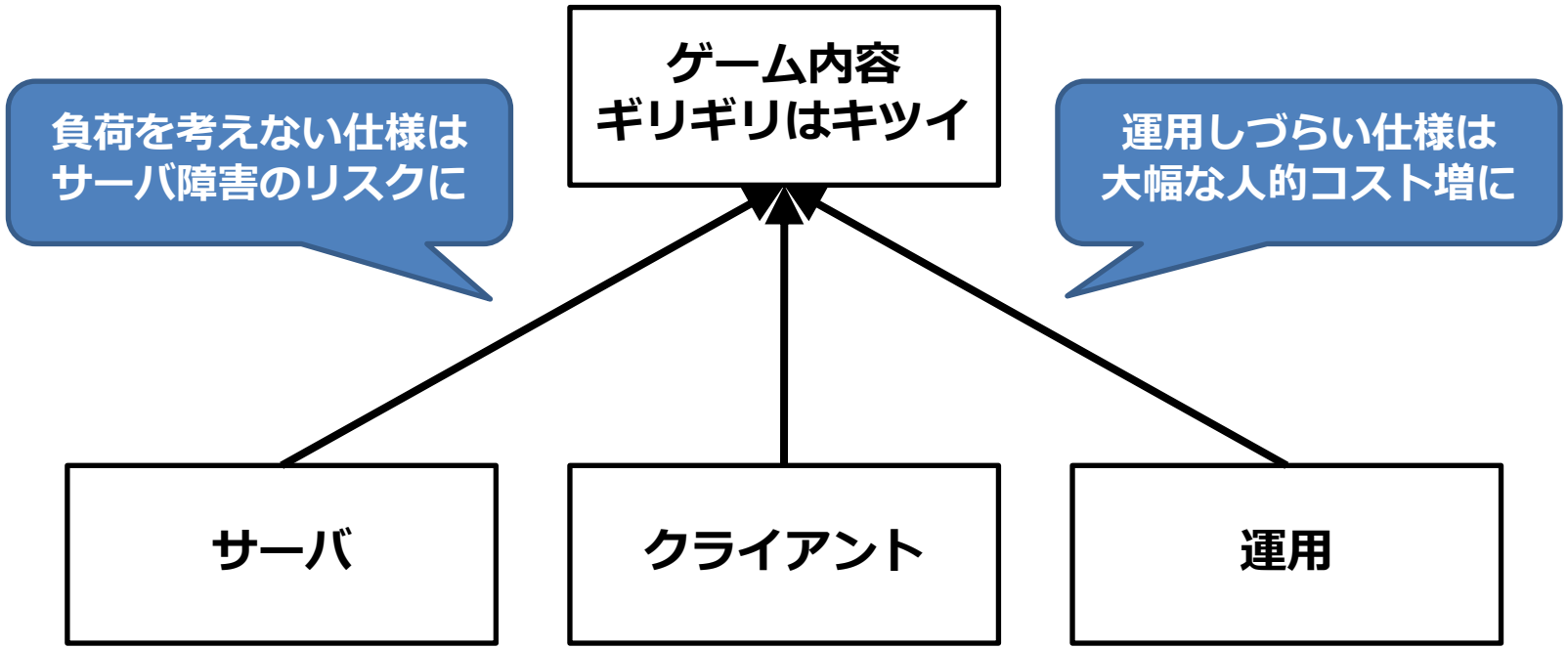
売り切りゲームだったら

- **ゲーム内容**と**クライアント**の二軸



ソシャゲだと

- そこに**サーバ**と**運用**が加わります



ソシャゲでありがちな話

- **後で非常に困る仕様が色々なところをスルーして実装されてしまったりする**
  - すぐにサーバ負荷が限界近くになってまともに遊べなかったり
  - 月々の運用に多大なコスト（人的コスト／インフラコスト）が掛かったり
  - 結果としてDAUや売上げまで下がったり
- **当然弊社でもよくあります**

何故そんなことに？

- **クライアントにばかり目がいきがち**
  - ネットワークサービスの経験有無で、この辺の視野にかなりの差が出てきます
- **サーバや運用の問題**が露見するのは大抵**サービスインしてしばらくしてから**

- 気付いても時既に遅く、取り返しがつかないなんてことも……

スクストではどう対処したか

- **サーバPGからも積極的に提案**
  - （企）この仕様実装して！
  - （サ）そのまま実装すると負荷の面でこういうリスクがあるよ。仕様の意図は？
  - （企）意図は○○な感じ。
  - （サ）○○に近いことができる別の手段があるよ。これなら工数も負荷も抑えられるよ。
  - （企）なるほど、それでもある程度は目的を達成できそうだから、そっちでよろしく！

提案に際して

- 判断に必要な**メリット／デメリット／代案**を分かりやすく説明
  - 例えば**ミッションの仕様**も、諸々を鑑みた上で、**通信回数減による手触り感向上**というメリットが選択された

- **内製かつ小規模のチームで、風通しが良い環境だったからこそ可能だった**

参考：風通しの良い例

- チーム全員がKPIを随時確認、現状分析
- 萌えコンテンツに詳しいサーバPGがアートディレクターに**ダメ出し**
- キャラクター推しに特化すべきか、**言葉による殴り合いもとい熱い議論**
  - 「ゲームとは何か？」という哲学的な領域に

- **ただしこの環境が成立するのは、お互いやるべきことをちゃんとやっているからこそ！**

さいごに

SQUARE ENIX

弊社では技術者を募集中！

- ソシャゲ制作：サーバPG
- ソシャゲ制作：クライアントPG
- エンジンやフレームワークの基幹部分をサポートする縁の下の力持ち

- 内製でモリモリ手を動かして何かしらのモノを作りたい方向け！
- 詳しくは後ほど！

ご清聴ありがとうございました！

SQUARE ENIX