13:00より開始いたします。もうしばらくお待ちください。

スマラジ! SmartDB Radio

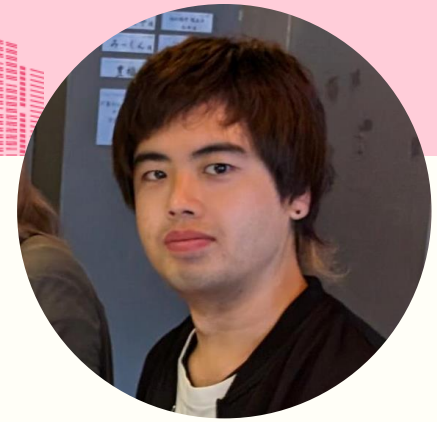
活用レベルアップ！

SmartDB深掘りセミナー

サービス＆プロダクト開発本部
茶位 翔太

スマラジ！性能を意識した設計を考える！
～全6回で習得！REST API⑤性能設計編～

株式会社ドリーム・アーツ



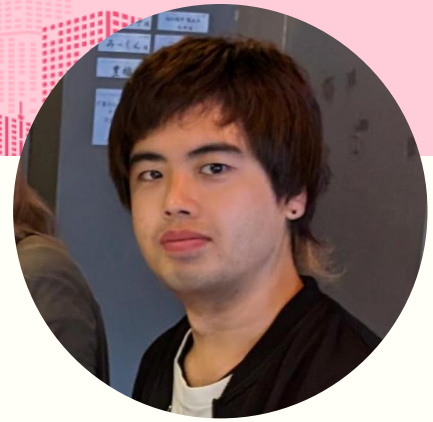
CTサービス本部
小田 樹

- アーカイブ配信用に録画しています。
- お使いのバージョンによって、製品画面のUIや操作感が一部異なる可能性があります。
- 質疑応答に関して、時間の関係上全て回答できない場合があります。
  その場合は、後日コミュニティサイトにて回答します。
- 今後の改善のため、最後にアンケートの回答にご協力ください。

スマラジ！ SmartDB Radio

活用レベルアップ！

SmartDB深掘りセミナー



スマラジ！性能を意識した設計を考える！
～全6回で習得！REST API⑤性能設計編～

サービス＆プロダクト開発本部
茶位 翔太

株式会社ドリーム・アーツ

CTサービス本部
小田 樹

DreamArts

No.	実施概要	詳細内容	推奨者
1	REST API 入門編	REST API概説 実践：PowerShellでREST APIを実行してみよう	SmartDB利用経験のある方
2	SmartDB REST API アプリ操作基本編	SmartDB REST APIの基本的な使い方 よく使う基本的なAPIの解説と実践	1の参加者、もしくは1の内容を理解している方
3	SmartDB REST API アプリ操作応用編	SmartDB REST APIの応用的な使い方 複数のREST APIを用いた業務適用の解説と実践	2の参加者、もしくは2の内容を理解している方
4	SmartDB REST API アカウントマスタ連携編	アカウントAPIの使い方 アカウントAPIを用いた業務適用の解説と実践	1の参加者、もしくは1の内容を理解している方
本日	SmartDB REST API 性能設計編	業務影響を軽減するための設計 SDBの性能を考慮した設計・運用	3の参加者、もしくは3の内容を理解している方 または、 4の参加者、もしくは4の内容を理解している方
6	SmartDB REST API システム連携設計編	SmartDBを含む複数システム連携の全体設計 SDB起点の実行方法：Webhook、定期バッチ処理	5の参加者、もしくは5の内容を理解している方

基本的なAPIの使い方

連携における設計編

※上記の内容は変更される場合がございます

DreamArts

パネリスト

連携開発から製品開発へ
連携からSmartDBを成長させる！

茶位（ちゃい）

所属：サービス＆プロダクト開発本部
出身：神奈川県
経歴：新卒でDA入社
　　プラグインから製品開発へ
　　Amazon連携開発の支援
趣味：料理

パネリスト

開発大好き！
連携開発の縁の下の力持ち！

小田（おだっぴ）

所属：CTサービス本部
出身：広島県
経歴：新卒でDA入社
　　複数PJTにて開発を担当
　　運用なども行う
趣味：作ること全般

モデレーター

MSCAといえばこの私。
業務テーマカットで活用提案中！

清水（しみけん）

所属：セールス統括本部
出身：愛媛県
経歴：SIerでPM→DAでCS組織を
　　立ち上げ等を経て現職
　　某新聞社のMCSA案件でPM
趣味：サンフレ観戦、キャンプ

①所属部署は？

②どのような役割を担っていますか？

③外部システム連携の経験はどのくらいありますか？

④過去開催分の参加について

DreamArts

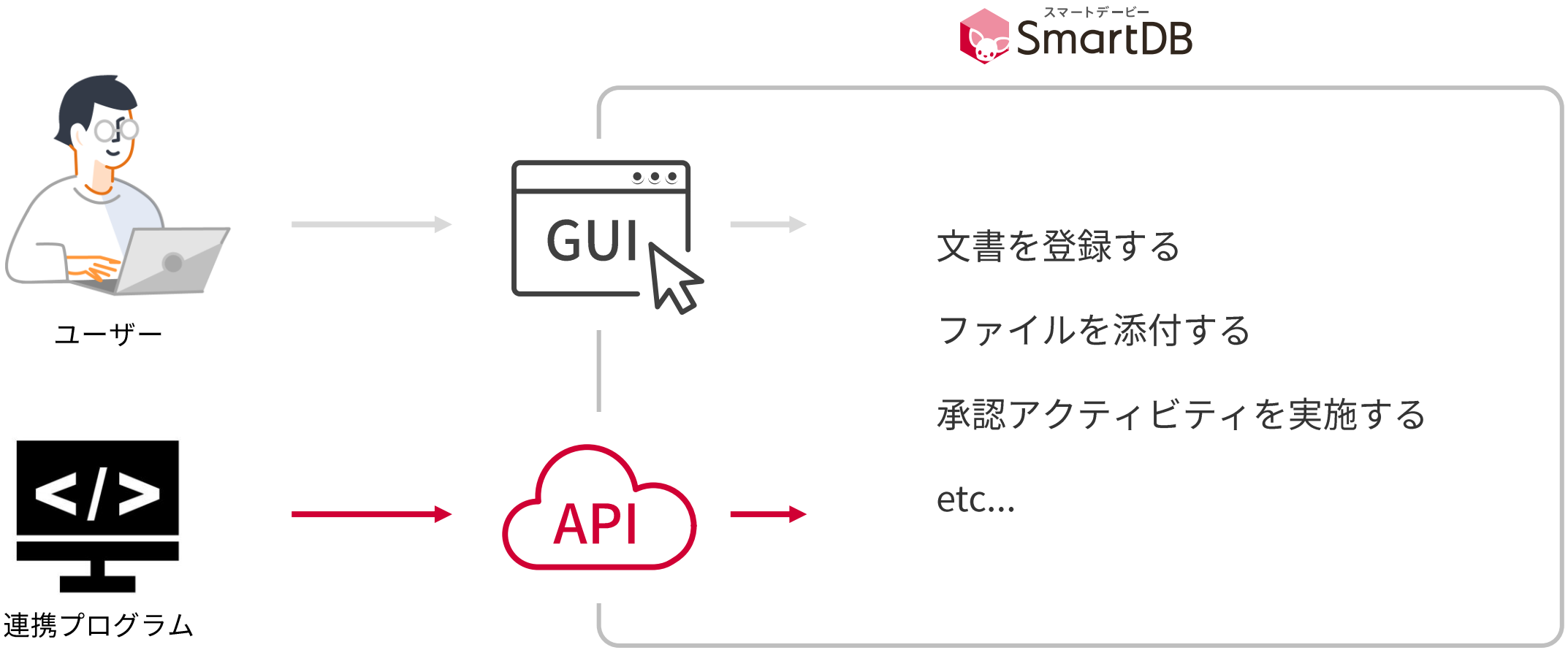
0. これまでのおさらい
1. 外部連携のパターン
2. 中間処理を実装するときの考え方
    i. 仕様を考慮した設計（性能/システム関連の上限）
    ii. よくあるつまづき
    iii. エラーハンドリング
3. 事例紹介
4. まとめ

DreamArts

0. これまでのおさらい
1. 外部連携のパターン
2. 中間処理を実装するときの考え方
    i. 仕様を考慮した設計（性能/システム関連の上限）
    ii. よくあるつまづき
    iii. エラーハンドリング
3. 事例紹介
4. まとめ

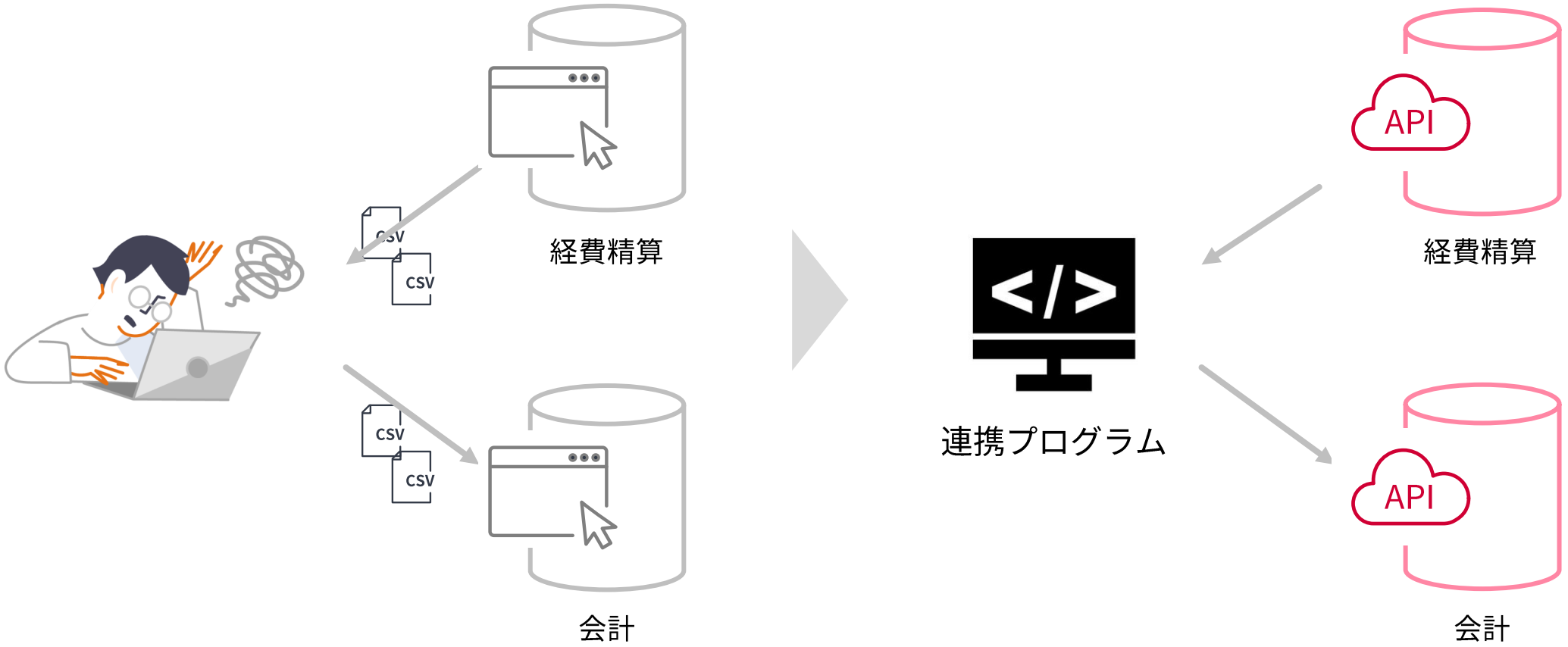
APIはアプリケーションを　操作する手段　の一つです

APIを用いることで、　外部システム　からSmartDBを操作できます



DreamArts

連携プログラムを介して、 **システム間の自動連携** を実現できます



1. SmartDB REST APIを**組み合わせる**ことで業務を実現
2. 実現したい要件に応じて**決まったパターンが存在**する

(文書更新、業務プロセス開始、マスタ同期、添付ファイルUL、アカウント自動連携etc)

- 代表的なよく使われるSmartDB REST API一覧

カテゴリ	REST API
バインダ / 文書	文書詳細情報取得 指定したビューの文書一覧取得 文書一覧のCSV出力 文書新規登録 文書更新 文書削除 文書一括操作 CSV入力（新規） CSV入力（更新）
業務プロセス	業務プロセス開始 アクティビティ実施 文書情報に基づいて実行中アクティビティ情報取得

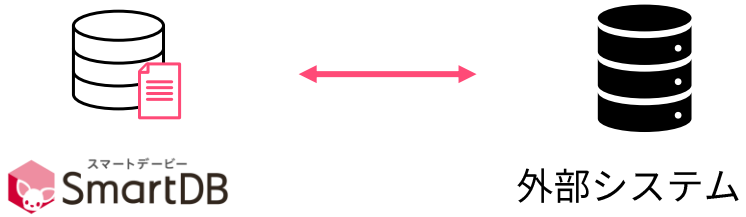
3. **ドキュメントやサポートサイトを参照**しリクエストを作成する

DreamArts

0. これまでのおさらい
1. 外部連携のパターン
2. 中間処理を実装するときの考え方
    i. 仕様を考慮した設計（性能/システム関連の上限）
    ii. よくあるつまづき
    iii. エラーハンドリング
3. 事例紹介
4. まとめ

DreamArts

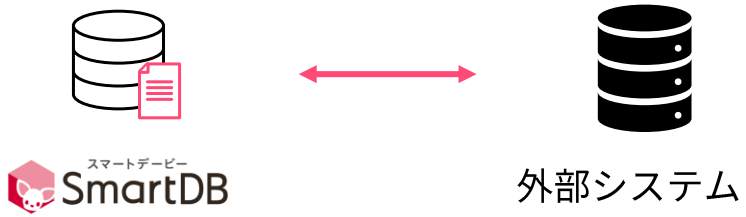
1 外部システムと直接的に連携



2 中間サーバーを介して間接的に連携

DreamArts

1 外部システムと直接的に連携



追加の開発不要！
気軽に連携可能！

2 中間サーバーを介して間接的に連携

DreamArts

①SmartDB外からのAPI呼び出しによる連携

外部システム側でREST APIを実行する機能がある場合

例：BIツール、RPAツール

②SmartDBからの外部API連携部品によるデータの参照

簡単なAPI連携であれば標準機能で実現可

例：郵便番号検索、法人番号検索、一部クラウドサービスなど

DreamArts

1 外部システムと直接的に連携

2 中間サーバーを介して間接的に連携

細やかな制御や業務要件へ
対応が可能

DreamArts

大量データや同時処理等、性能面への懸念に対して対処可能

認証方式、リクエスト内容の変換、詳細なエラーハンドリング

SmartDB、外部システムとは別で用意
サーバーレスのサービスを利用
例： AWS lambda 、 AzureFunctions など

外部システム連携　4つの方法

SmartDBの文書を外部システムに連携

外部システムのデータをSmartDBの文書に連携

SmartDBの文書を外部システムに連携

外部システムのデータをSmartDBの文書に連携

	概要	トリガー	例
①	ワークフローの処理として外部システムへデータを送信	プロセスの通知ロボット	クラウドサインやDocuSignなどの電子契約サービスへのデータ連携
②	ユーザー自身の操作で外部システムからデータを参照	外部API連携部品	駅すぱあとやMapFan等から経路情報を取得
③	文書への操作タイミングで外部システムへデータを送信	バインダのWebhook設定	マスタデータの同期など

SmartDBの文書を外部システムに連携

外部システムのデータをSmartDBの文書に連携

	概要	トリガー	例
④	外部システムからSmartDBのデータを参照・追加・更新・削除	外部システムが起点	マルケトやZendeskに登録された情報をSmartDBへ追加

DreamArts

概要

- Teamsに投稿した情報をSmartDBのバインダに集積する機能

解決したい課題

- Teamsのチャットでは有用な情報も流れがちであり検索も困難

連携する目的

- 有用な情報集積のため
- 過去の有用な情報の検索のため

DreamArts

DreamArts

幅広い業務要件に対応可能

情報更新頻度/処理件数/タイミング/閲覧範囲/権限回り/セキュリティポリシー

利用頻度が高い業務/重要なデータを扱う業務の場合

連携が止まると致命的！

マスタ更新（商品マスタの同期）：最新情報が取れず業務が進まない！

売上データが更新されない：経営判断、営業処理ができない！

DreamArts

幅広い業務要件に対応可能

情報更新頻度/処理件数/タイミング/閲覧範囲/権限回り/セキュリティポリシー

利用頻度が高い業務/重要なデータを扱う業務の場合

だからこそ…

処理を止めない＝エラーを発生させない

性能を考慮した設計が大事！！

連携が止まると致命的！

マスタ更新（商品マスタの同期）：最新情報が取れず業務が進まない！

売上データが更新されない：経営判断、営業処理ができない！

DreamArts

0. これまでのおさらい
1. 外部連携のパターン
2. 中間処理を実装するときの考え方
    i. 仕様を考慮した設計（性能/システム関連の上限）
    ii. よくあるつまづき
    iii. エラーハンドリング
3. 事例紹介
4. まとめ

エラーの発生を
予防するには？

仕様を考慮した設計

エラーハンドリング

エラー発生時の処理
をどうするか？

i.仕様を考慮した設計：性能/システム関連の上限

SmartDB Rest APIの
バージョン

SmartDB Rest APIの
レート制限

DreamArts

SmartDBのRest API Ver3を利用すること

- 一覧、一括承認などで性能改善されている
- リクエストURLはhttps://{SmartDBドメイン}/hibiki/rest/3/{対象API}

SmartDB Rest APIの
バージョン

SmartDB Rest APIの
レート制限

外部処理からSmartDBに対するAPIリクエストは1秒間10回まで

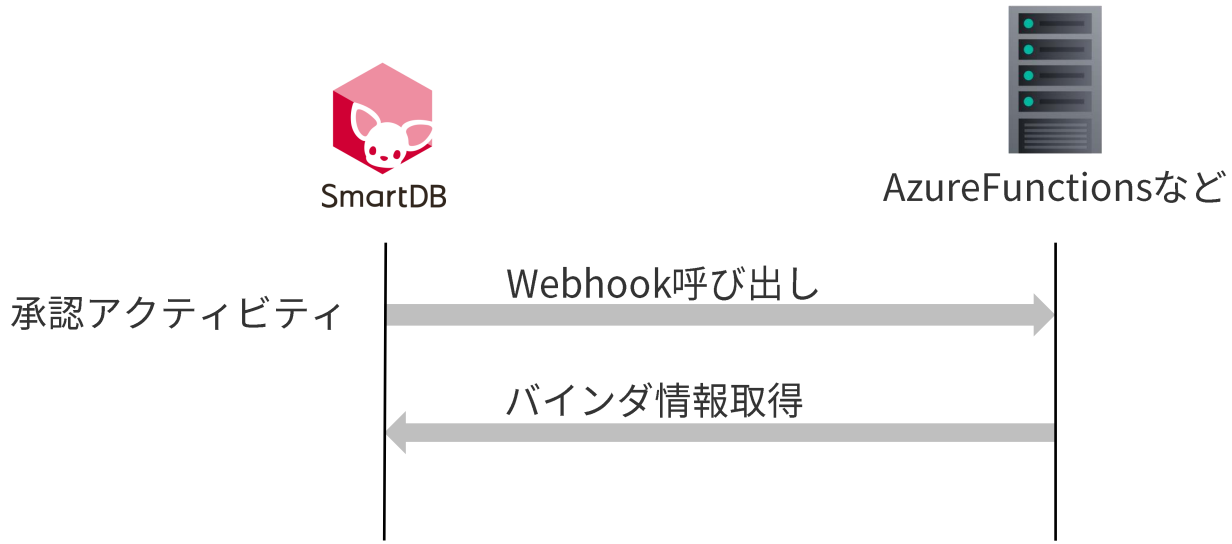
課題：

一括承認からのWebhookなど、並列で多数の外部処理が呼び出されると
制限を超える可能性がある

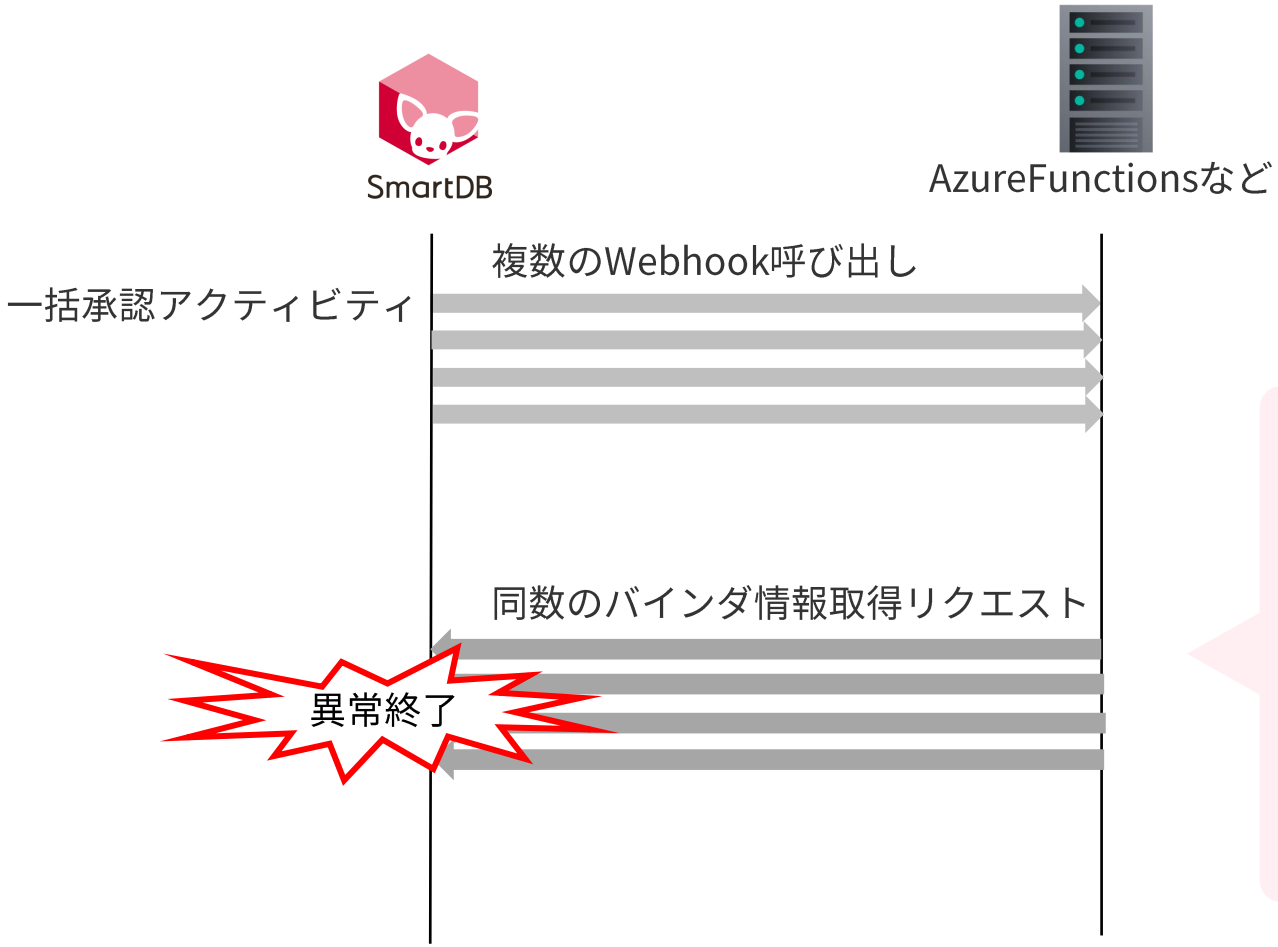
解決例：

キューイングすることにより、実行される外部処理の数を抑制する

SmartDBからWebhookを呼び出し、中間サーバーでバインダ情報を取得



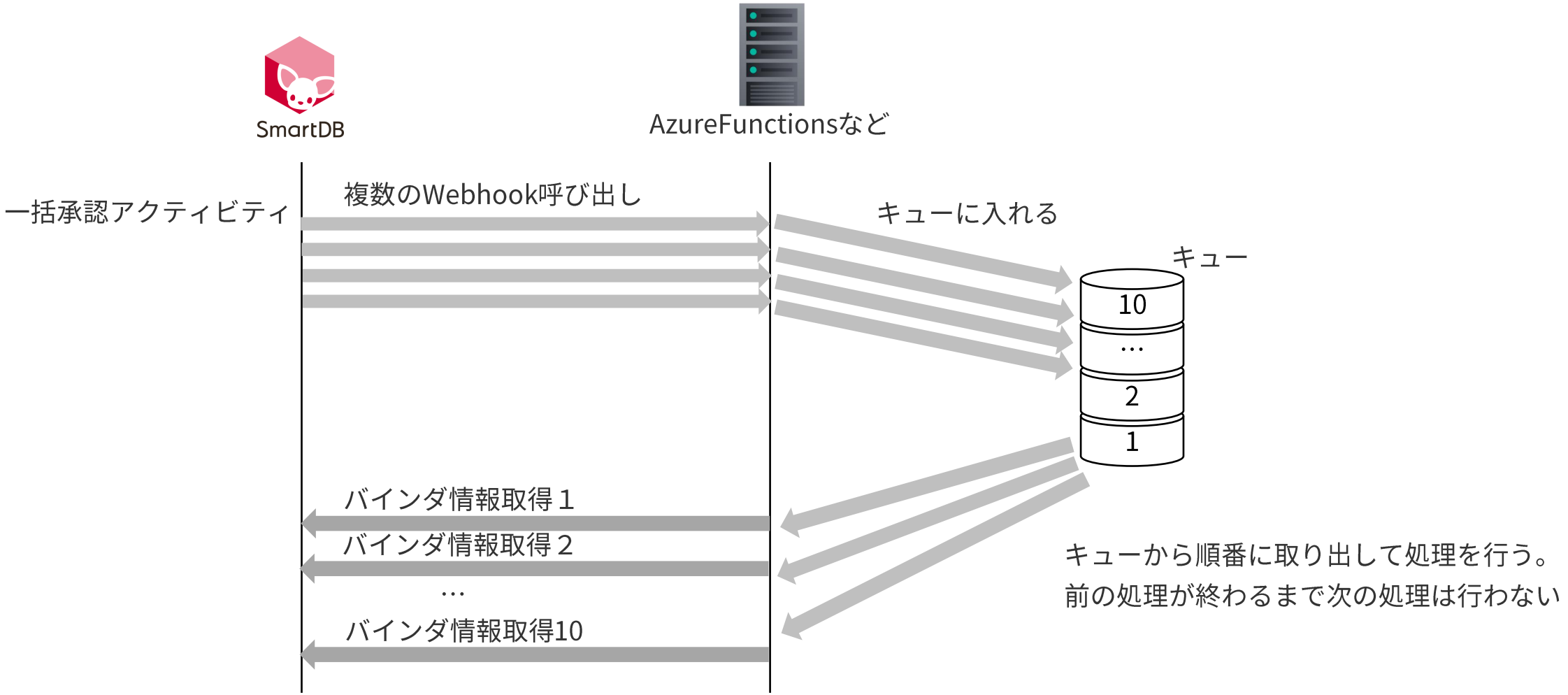
複数/大量のリクエストが発生する場合



一斉にSmartDBへリクエストが返ってくるため、1秒間に10回の制限を超えてしまう可能性がある

DreamArts

キューイングの仕組みを追加することでレート制限を回避可能



SmartDB Rest APIの

バージョン

Ver.3の利用

SmartDB Rest APIの

レート制限

1秒10回を回避

キューイング

ii. よくあるつまづき
～通知ロボットのWebhookタイムアウト考慮～

プロセス定義

Webhook

URL*	https://hogehoge.azurewebsites.net/api/cover?
HmacSHA256 キー	****
Basic認証	
ヘッダ	ヘッダをカスタマイズしない
終了条件	処理完了を待たない

SmartDBの仕様

タイムアウト5分

DreamArts

中間サーバーの
処理時間

5分以内

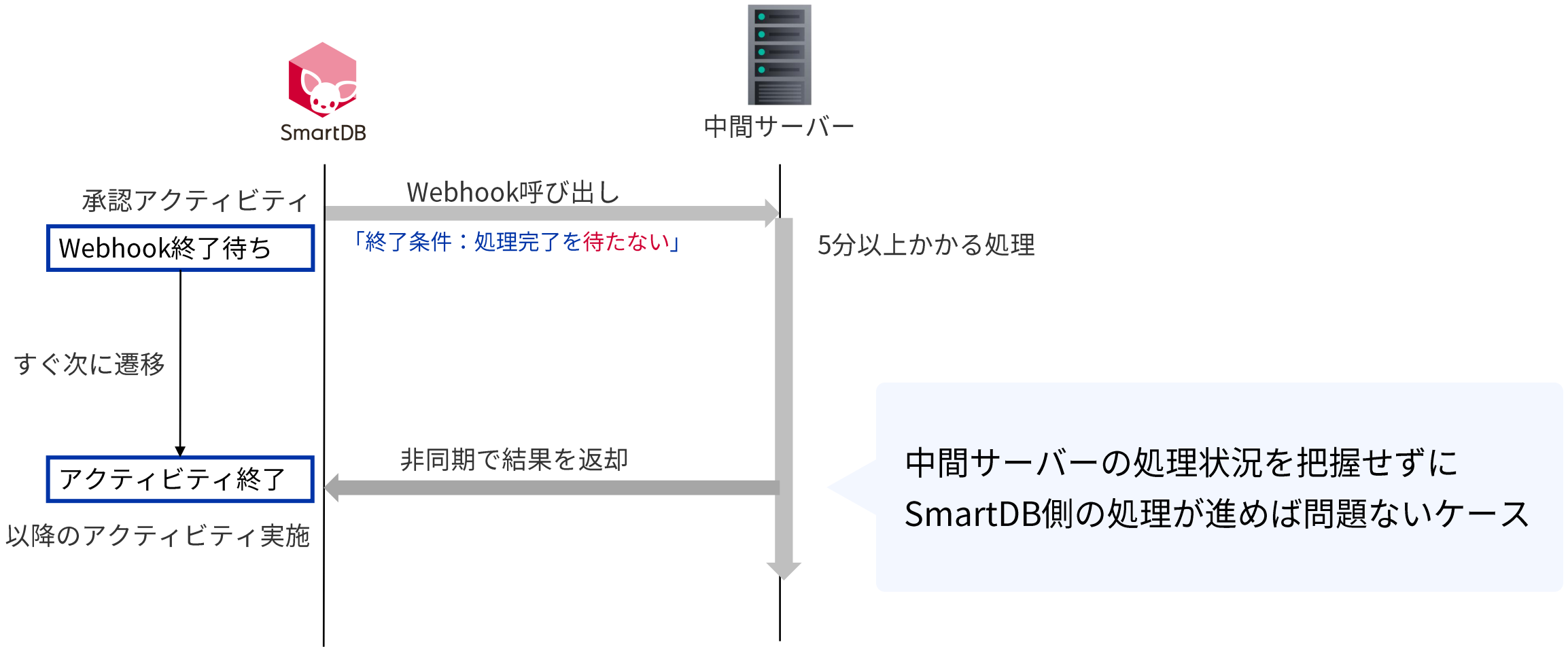
終了条件
処理完了を待つ

5分以上の可能性あり

終了条件
処理完了を待たない

DreamArts

処理時間に関わらず、次のアクティビティに遷移(タイムアウトは発生しない)

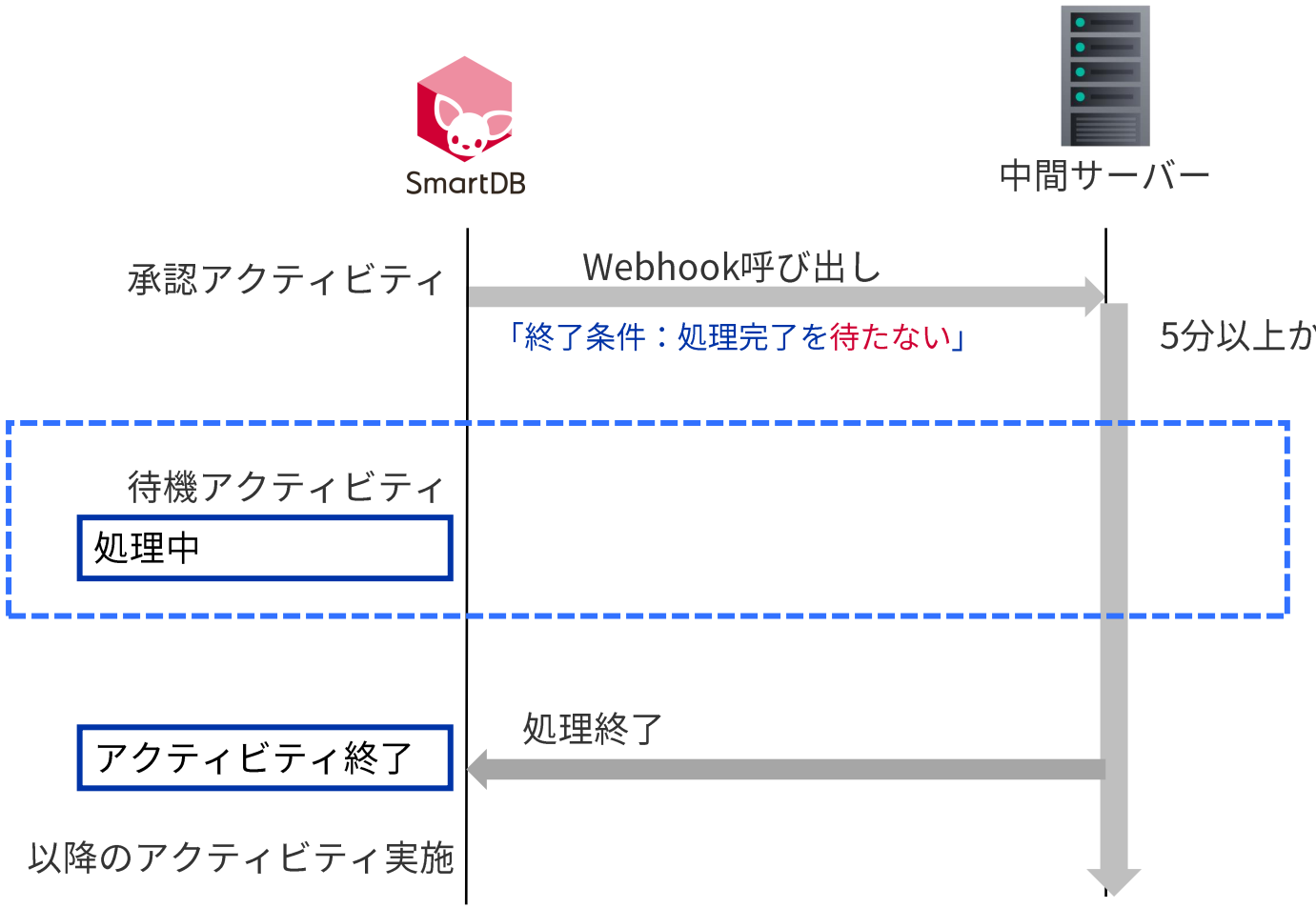


中間サーバーの処理状況を把握せずに
SmartDB側の処理が進めば問題ないケース

DreamArts

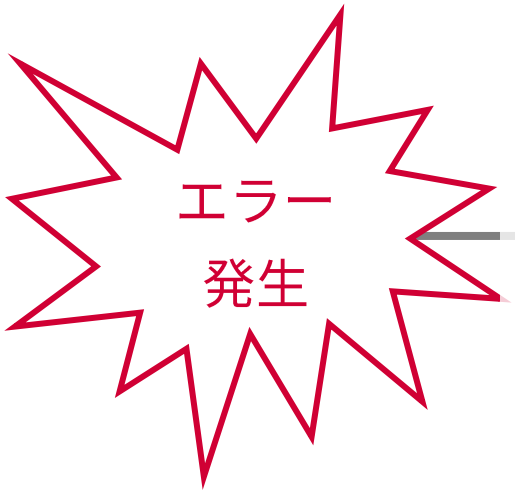
中間サーバーの処理が完了後にSmartDB側の処理を進めたい場合、

待機アクティビティを追加することで対応可能



中間サーバーの処理結果を待つための
のアクティビティを追加

iii.エラーハンドリング



エラーハンドリング

エラー発生時の処理対応をあらかじめ用意しておき
人手を介さずに対処すること

エラー
発生

エラー検知

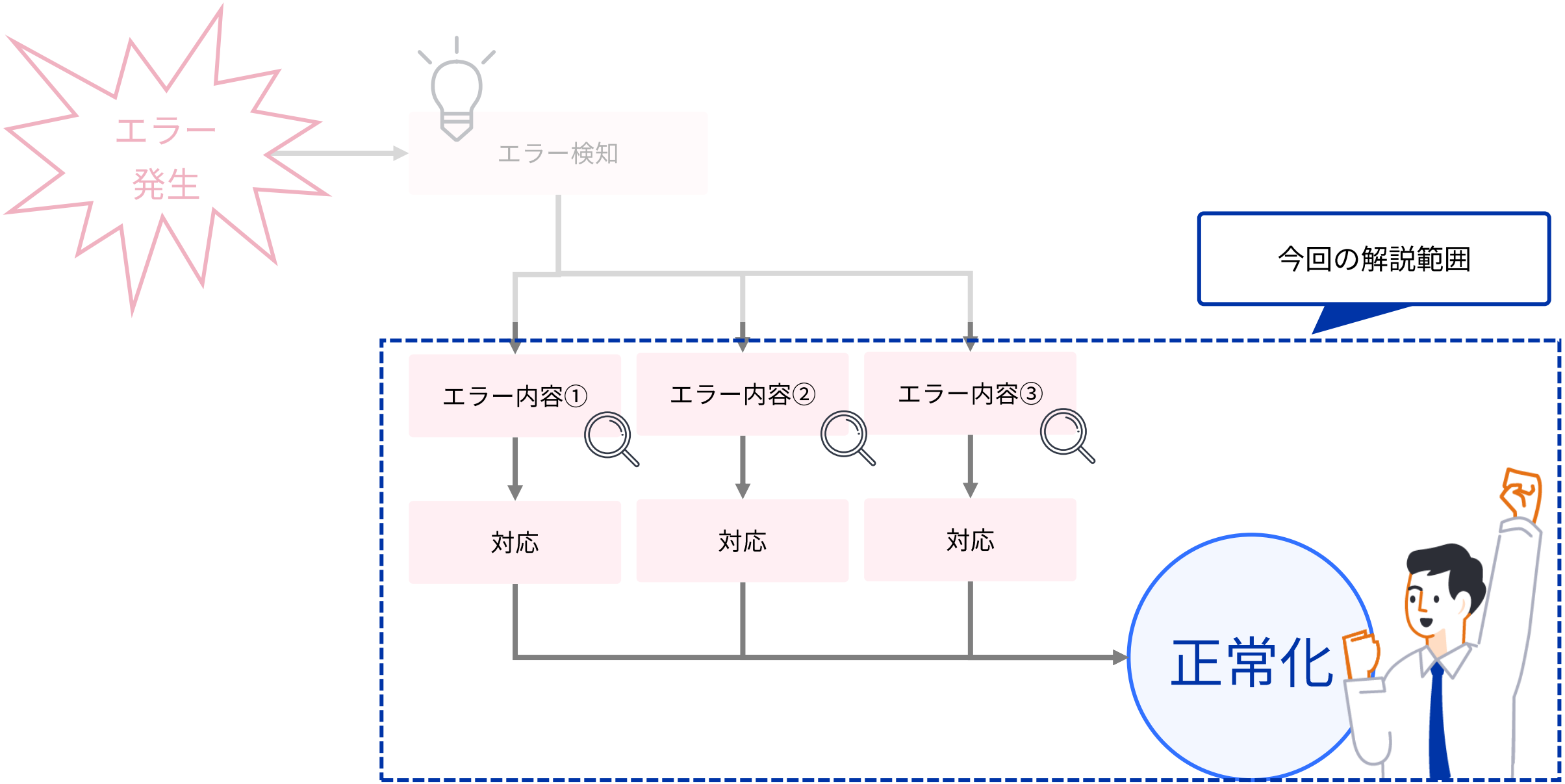
連携先サービスの出力情報により
検知方法を設定

エラー内容①
エラー内容②
エラー内容③

対応
対応
対応

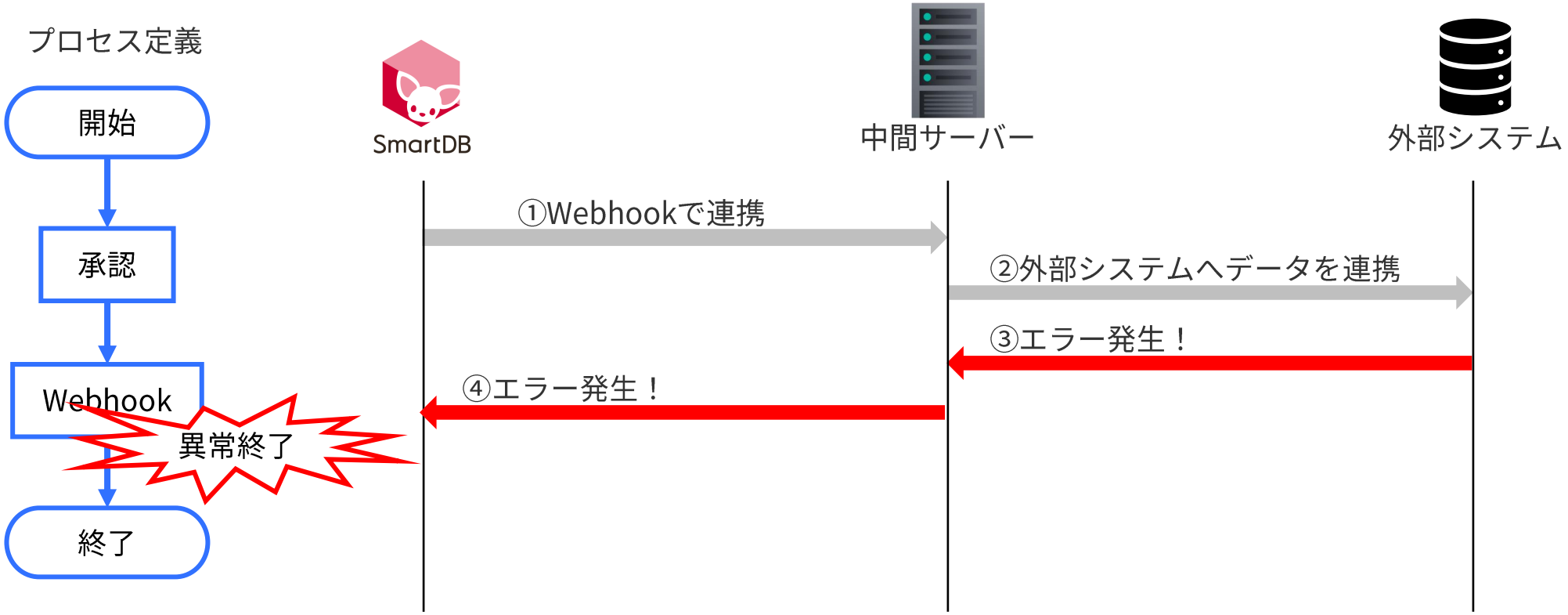
正常化

今回の解説範囲：エラーハンドリング



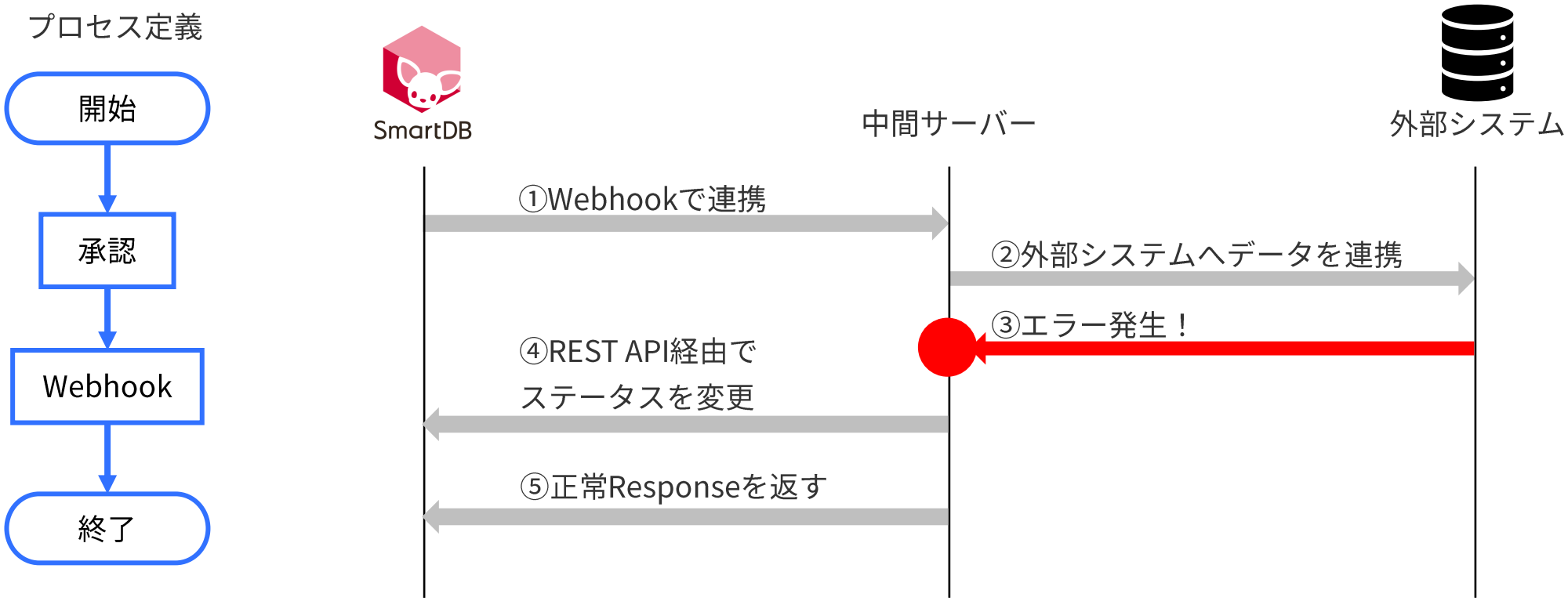
プロセスのWebhookを起点とする以下の構成で、外部システムがエラーを返した場合を想定

⇒エラーをそのまま返してしまうとWebhookが強制終了



処理結果（成功orエラー）はステータスや部品に変更

これによりSmartDBへ直接エラーを返さず、処理結果の状態を保持＝異常終了させない



DreamArts

01．一時的なエラー
例：502 Bad Gateway
504 Gateway Timeout

02．入力データのエラー
例：400 Bad Request
404 Not Found

03．予期せぬエラー
例：500 Internal Server Error

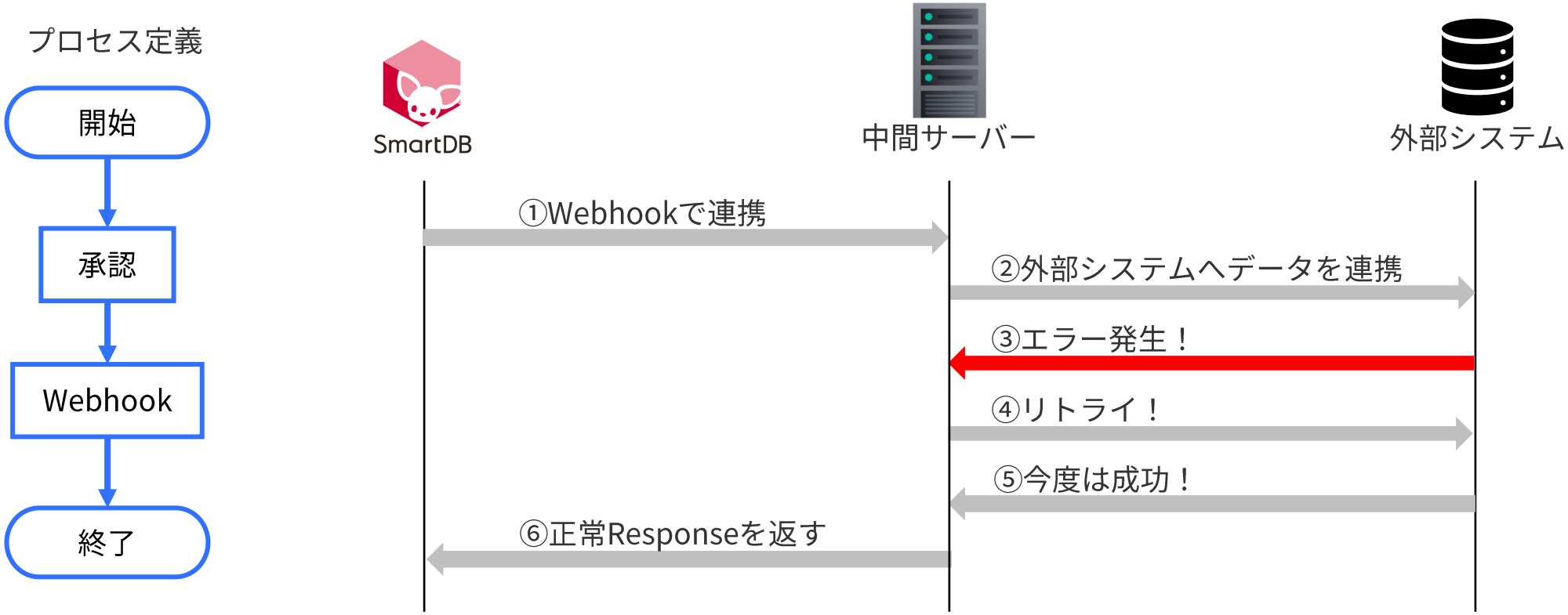
中間サーバーによる
外部連携リクエストの
リトライ

プロセス定義の
フロー制御による
リトライ

フォーム定義の
アクションボタンによる
リトライ

DreamArts

一時的に問題の場合は中間サーバーでリトライ



DreamArts

01．一時的なエラー
例：502 Bad Gateway
504 Gateway Timeout

02．入力データのエラー
例：400 Bad Request
404 Not Found

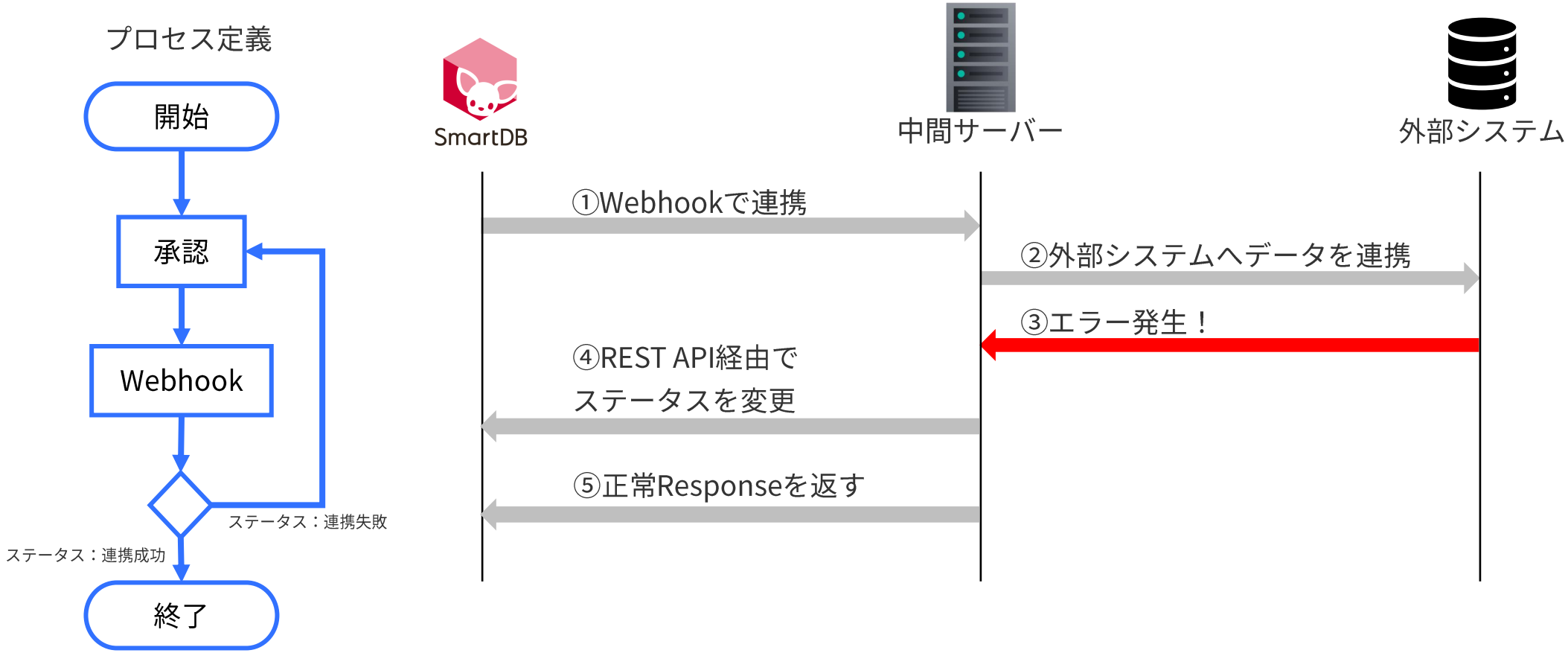
03．予期せぬエラー
例：500 Internal Server Error

中間サーバーによる
外部連携リクエストの
リトライ

プロセス定義の
フロー制御による
リトライ

フォーム定義の
アクションボタンによる
リトライ

ユーザ操作に起因する場合はフローで制御することも可能



DreamArts

01．一時的なエラー
例：502 Bad Gateway
504 Gateway Timeout

02．入力データのエラー
例：400 Bad Request
404 Not Found

03．予期せぬエラー
例：500 Internal Server Error

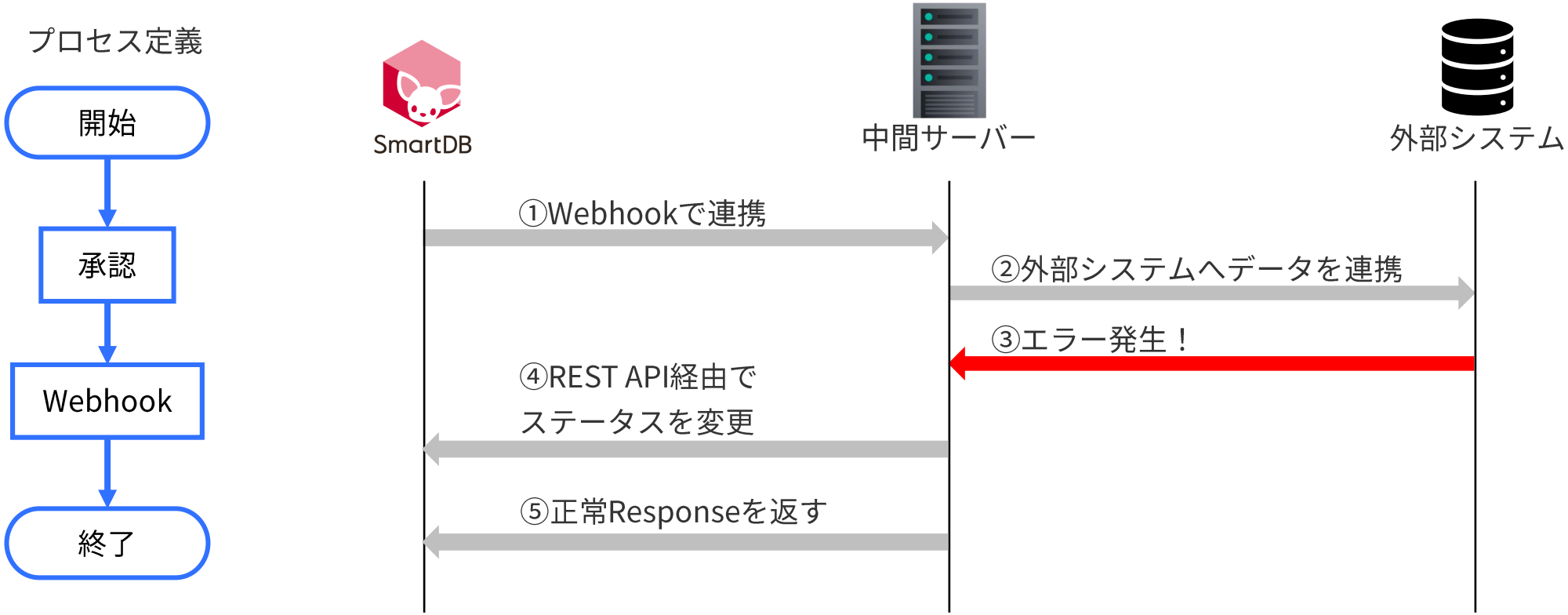
中間サーバーによる
外部連携リクエストの
リトライ

プロセス定義の
フロー制御による
リトライ

フォーム定義の
アクションボタンによる
リトライ

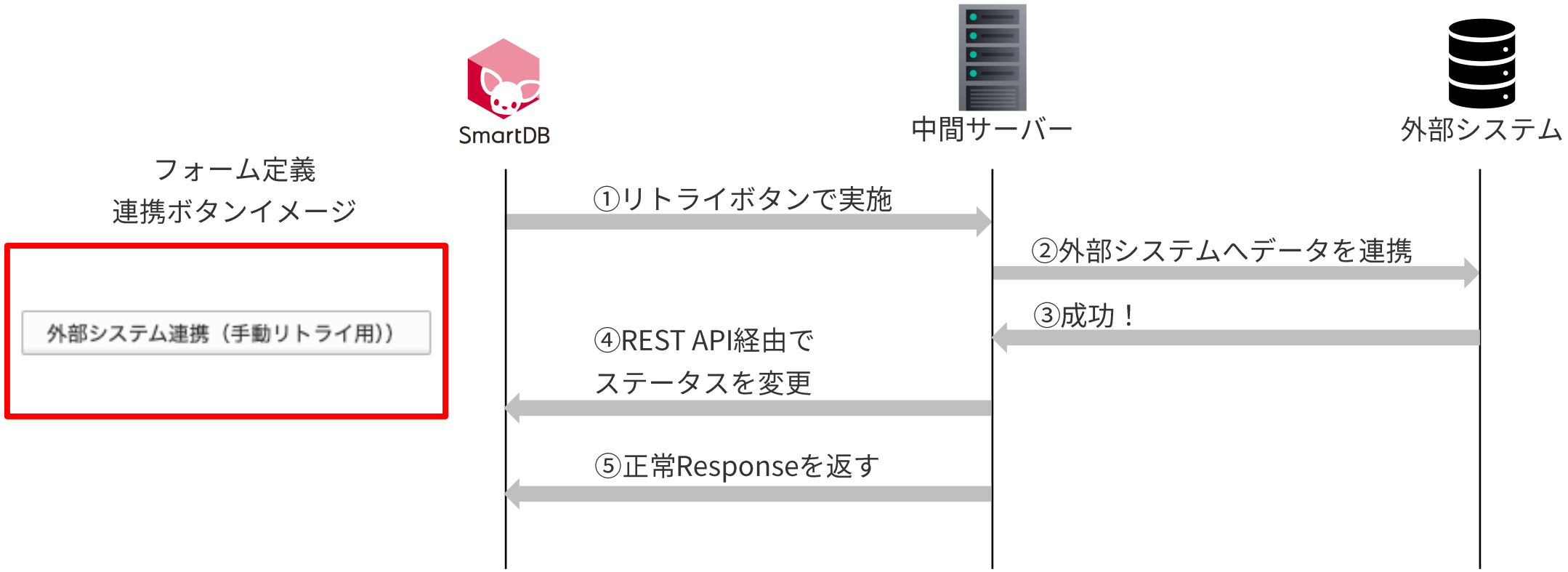
DreamArts

プロセス定義と中間サーバーの処理は変更なし



DreamArts

再度実行可能なようにリトライ用のアクションボタンを用意



DreamArts

中間サーバーにエラーを残すようにする部品置く

01．一時的なエラー
例：502 Bad Gateway
504 Gateway Timeout

02．入力データのエラー
例：400 Bad Request
404 Not Found

03．予期せぬエラー
例：500 Internal Server Error

中間サーバーによる
外部連携リクエストの
リトライ

プロセス定義の
フロー制御による
リトライ

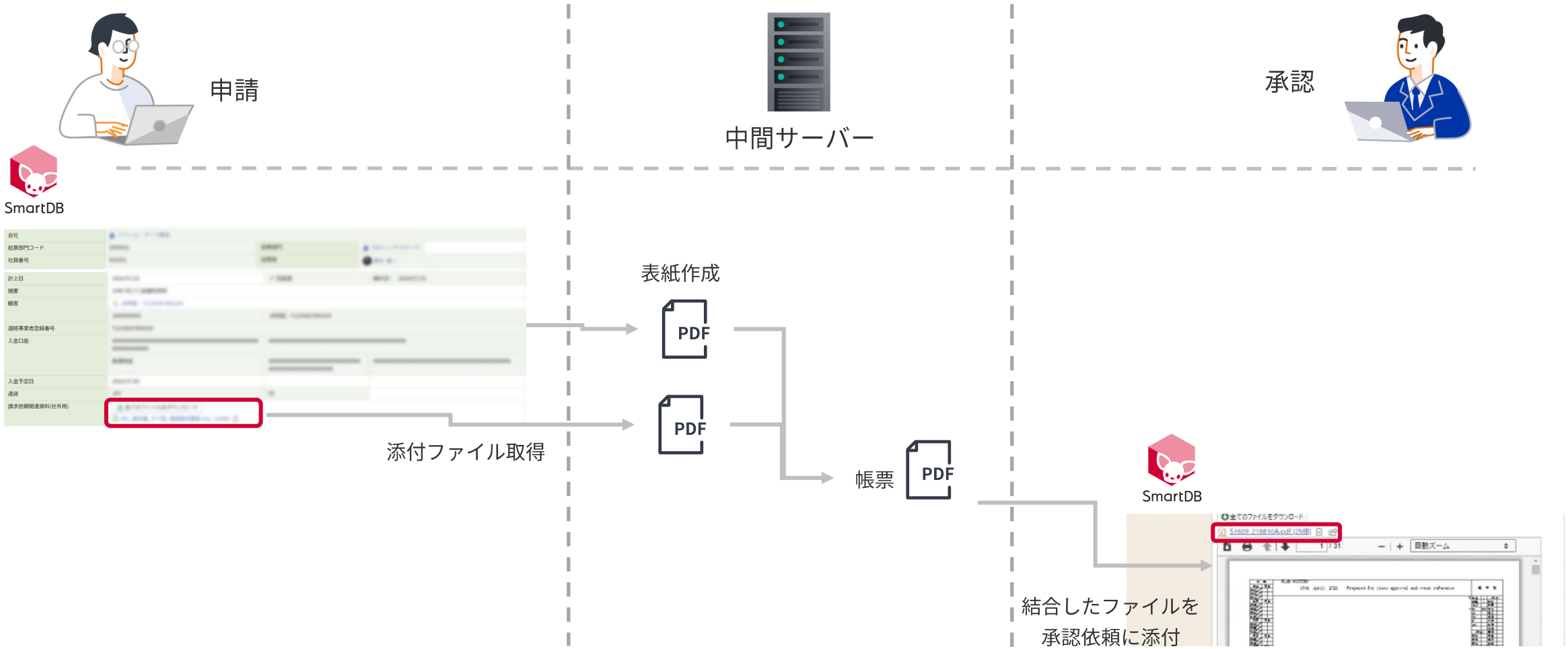
フォーム定義の
アクションボタンによる
リトライ

DreamArts

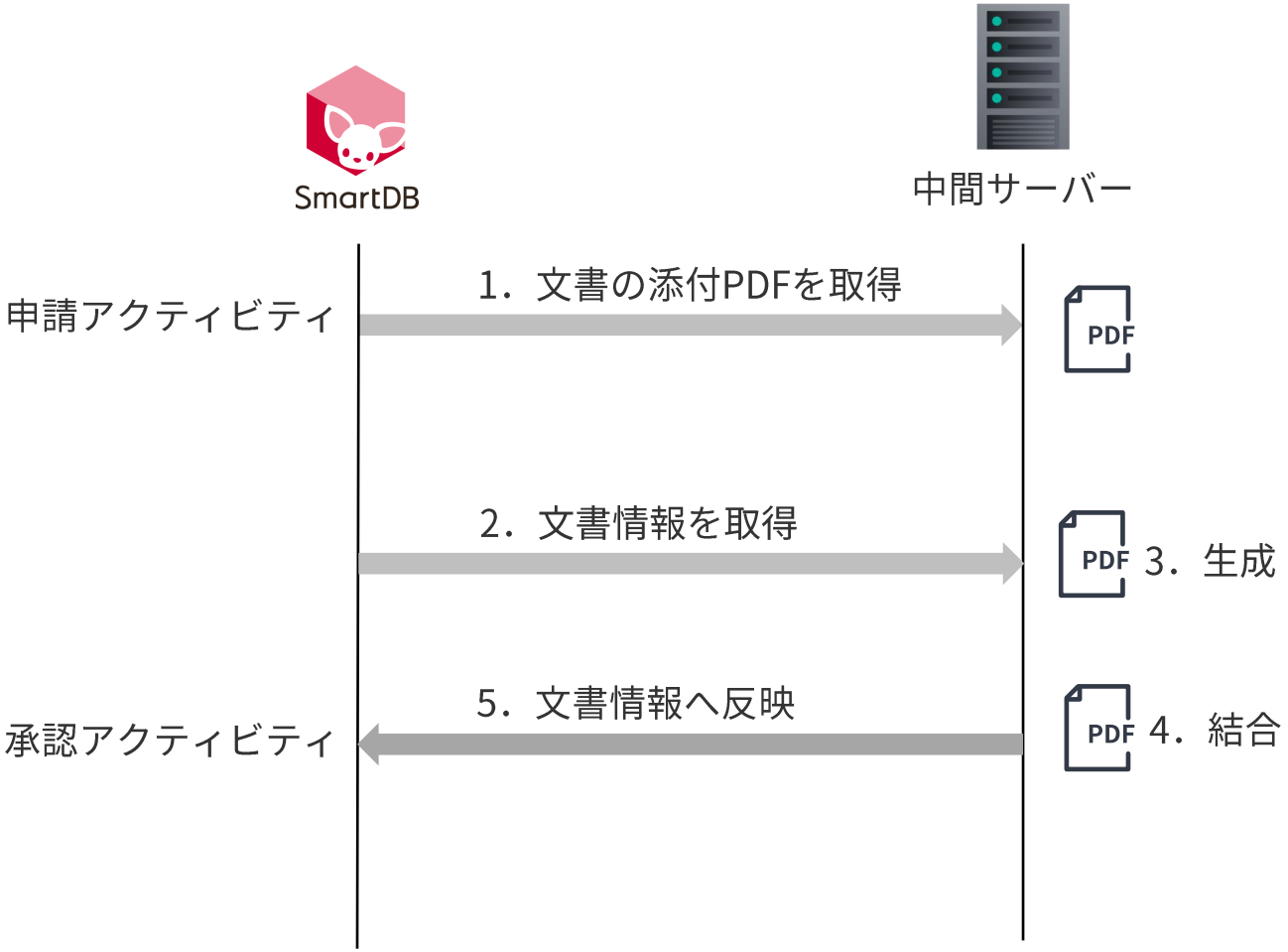
0. これまでのおさらい
1. 外部連携のパターン
2. 中間処理を実装するときの考え方
    i. 仕様を考慮した設計（性能/システム関連の上限）
    ii. よくあるつまづき
    iii. エラーハンドリング
3. 事例紹介
4. まとめ

DreamArts

連携要件	文書内の添付PDFと文書情報を結合して帳票作成後に承認したい



連携プログラム内容



1. 文書に添付したPDFを取得
2. 文書の部品データを取得
3. 部品データを元に表紙を動的に生成
4. PDFに結合
5. 文書に反映する

Webhook

URL*	https://hogehoge.azurewebsites.net/api/cover?
HmacSHA256 キー	****
Basic認証	
ヘッダ	ヘッダをカスタマイズしない
終了条件	処理完了を待つ

申請アクティビティ開始後、
通知ロボットで処理を開始

同期処理を実施

5分以内で処理が完了する想定

DreamArts

申請アクティビティ開始後、
通知ロボットで処理を開始

同期処理を実施

業務アプリが活用され利用頻度が高まる

5分以内で処理が完了する

DreamArts

と同時に課題が発生

申請が集中するとエラー発生
(大量データを処理できない)

中間サーバーの
リソース不足

5分以内で処理が完了する想定

DreamArts

解決方法として考えられる対策で適切なものは？

A) 連携プログラムを配置した環境（外部システム）をスケールアップ

B) 一括承認機能を利用しないように運用でカバー

C) キューイングの仕組みを入れるなど、連携プログラムを改善

Webhook

URL*	https://hogehoge.azurewebsites.net/api/cover?
HmacSHA256 キー	****
Basic認証	
ヘッダ	ヘッダをカスタマイズしない
終了条件	処理完了を待つ

申請アクティビティ開始後、
通知ロボットで処理を開始

同期処理を実施

5分以内で処理が完了する想定

DreamArts

Webhook

URL*	https://hogehoge.azurewebsites.net/api/cover?
HmacSHA256 キー	****
Basic認証	
ヘッダ	ヘッダをカスタマイズしない
終了条件	処理完了を待たない

待機アクティビティを追加
待機アクティビティの実施は連携プログラムが行う

連携プログラムでキューイングを実施
外部システムの処理にin-Queue／de-Queue仕組みを入れる

PDFに表紙がある状態で正常に承認可能

Webhookのタイムアウト制限を考慮

DreamArts

0. これまでのおさらい
1. 外部連携のパターン
2. 中間処理を実装するときの考え方
    i. 仕様を考慮した設計（性能/システム関連の上限）
    ii. よくあるつまづき
    iii. エラーハンドリング
3. 事例紹介
4. まとめ

DreamArts

1. 外部連携のパターンに応じて、**中間処理をどう設計するか**を検討

2. 性能面で気を付けること3つ

    ① REST API は必ず**Ver3**を利用

    ② 環境全体で**秒間10リクエスト**の制限→**キューイング**することを考える

    ③ 中間処理で5分以上かかる場合、**Webhookタイムアウト**が発生する

3. 複雑に考え過ぎず、**ポイントを押さえてシンプルな設計を！**

MCS

基幹システム

会計システム
人事管理システム
生産管理システム

データ参照

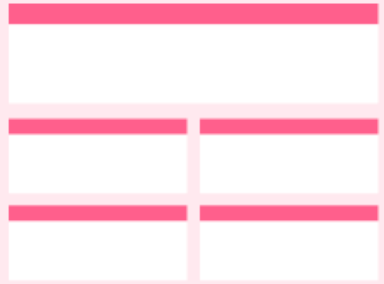
確定データ
連携

MCSA

SmartDB

マスタデータ(親)

関連する業務情報



関連情報の
受け渡し

処理データ
取得

専用システム

電子署名システム
BIツール
スクラッチシステム
など

Q&A

お知らせ

資料や動画は、後日コミュニティサイト「スマラジ！ルーム」へ掲載します。

本イベントに関するご質問も受付けますのでお気軽に投稿ください。

https://cs.support-smartdb.com/hc/ja/community/topics/5423952120601

※閲覧にはサポートサイトへのログインが必要です。

認定制度：SmartDB Certified Specialist（SCS）について

DreamArts

「SmartDB」に関する優れた専門知識や技術を証明し、

「デジタルの民主化」を推進・実現できる人材であることを認定するプログラムです。

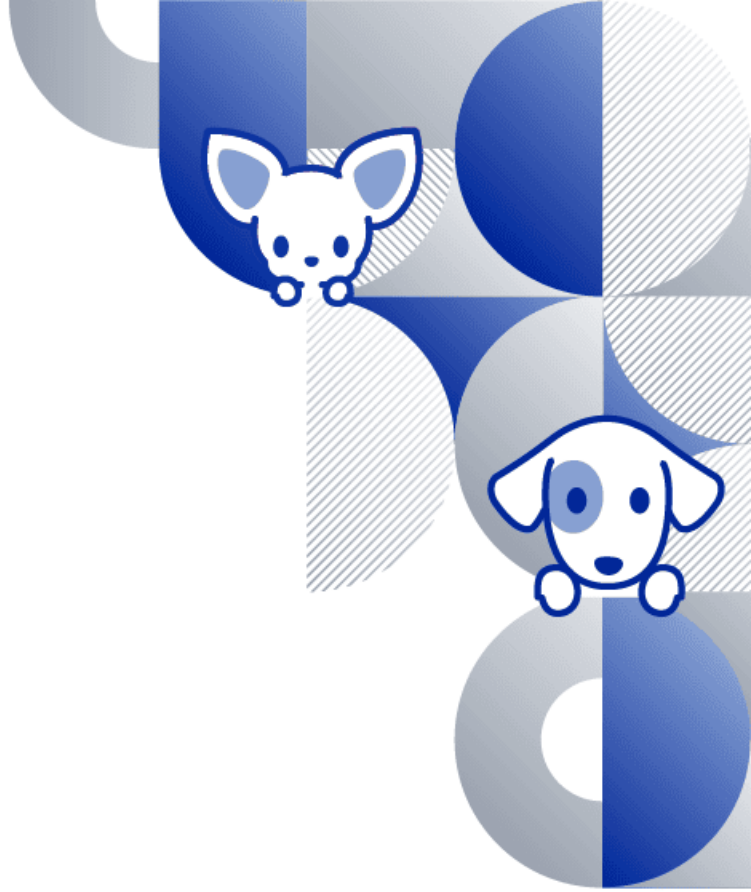
DreamArts

スマラジ！Rest APIシリーズはGOLDが対象の内容です。

ぜひ受験してみてください。

業務デザイナー	オーガナイザー	エキスパート
SmartDB の基本機能・応用機能を習得し、業務アプリのデザイン・開発による業務改善ができることを証明	「デジタルの民主化」を理解し、「SmartDB」の活用拡大・統制・推進ができることを証明 ※業務デザイナーの SILVER グレード認定で受験可能	「SmartDB」と外部システムとの連携により業務フロー全体の改善ができることを証明 ※業務デザイナーの SILVER グレード認定で受験可能
 SILVER 応用機能の習得者	 DIAMOND 「デジタルの民主化」推進における高度な実績の保有者	 PLATINUM 外部連携における高度な実績の保有者
 BRONZE 基本機能の習得者	 SAPPHIRE 「デジタルの民主化」推進における高度なスキルの習得者	 GOLD 外部連携における高度なスキルの習得者

▶詳細・お申込みはこちら



DreamArts

https://www.dreamarts.co.jp/