“FINAL FANTASY VII REMAKE”における自動QAシステムの構築と運用

株式会社スクウェア・エニックス
テクノロジー推進部　太田健一郎

アジェンダ

ゲームにおけるランダム要素と課題

“FF7R"における自動QAシステム

自動QAシステムのアーキテクチャ

自動QAシステムの定量評価

自動QAシステムの課題と総評

＊FF7R：以降”FINAL FANTASY VII REMAKE”の略称とさせて頂きます

ゲームにおけるランダム要素と課題

ランダム要素と課題

ゲームにはランダム要素が多い
ゲームのデザインは一晩で変わることがある

他ドメインで利用されるキャプチャー＆リプレイツールやスクリプティング技術ではランダム要素や頻繁な変更に耐えきれない可能性が高い

参考：キャプチャ＆リプレイ

記録と再生の流れ

1. 人が操作して記録
2. スクリプト化
   - ビジュアルスクリプト
   - コードスクリプト
3. 可変部分を設定
4. スクリプトを再生

オブジェクトの認識方法

座標ベース
- 座標で直接指定
  - 例：Click(270, 120)
- 解像度や位置の変更に弱い

画像ベース
- 画像で指定しマッチング
- マッチングの閾値の設定が難しい

オブジェクトベース
- オブジェクトのIDや特徴で指定
  - 例：Click(objectId: 10)
- IDや一意に識別できる情報を取得できる必要がある

ステータス

ScreenXY = (250, 100)
ObjectId = 10

ランダム要素の例

記録時に取った宝箱の場所が再生時には移動していた

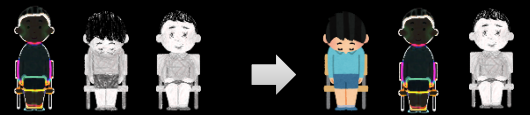
→対応技術：インタラクションBOT

座標ではなく要素のIDで要素とインタラクションする

記録時に複数のカットシーンの最初からスキップしたが、再生時にはカットシーンの順番が一部入れ替わっていた

→対応技術：スタックマシン

記録再生をスタックマシンで実装する事により、再生の各ステップを入れ替え、スキップ、待機、破棄できる

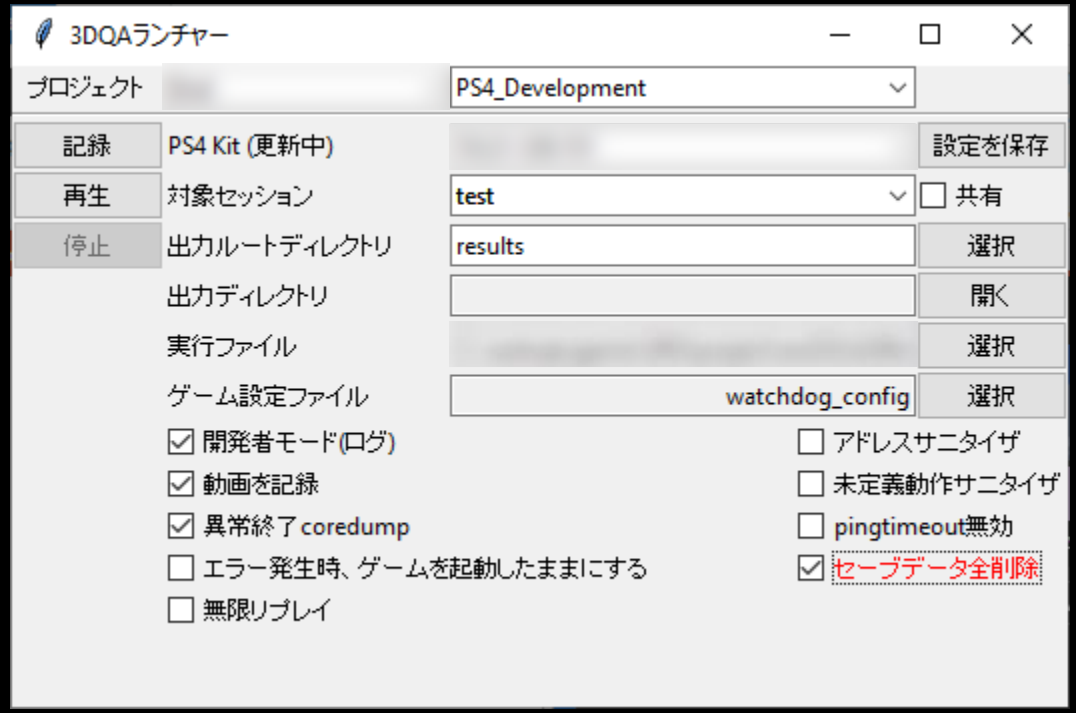


“FF7R”における自動QAシステム

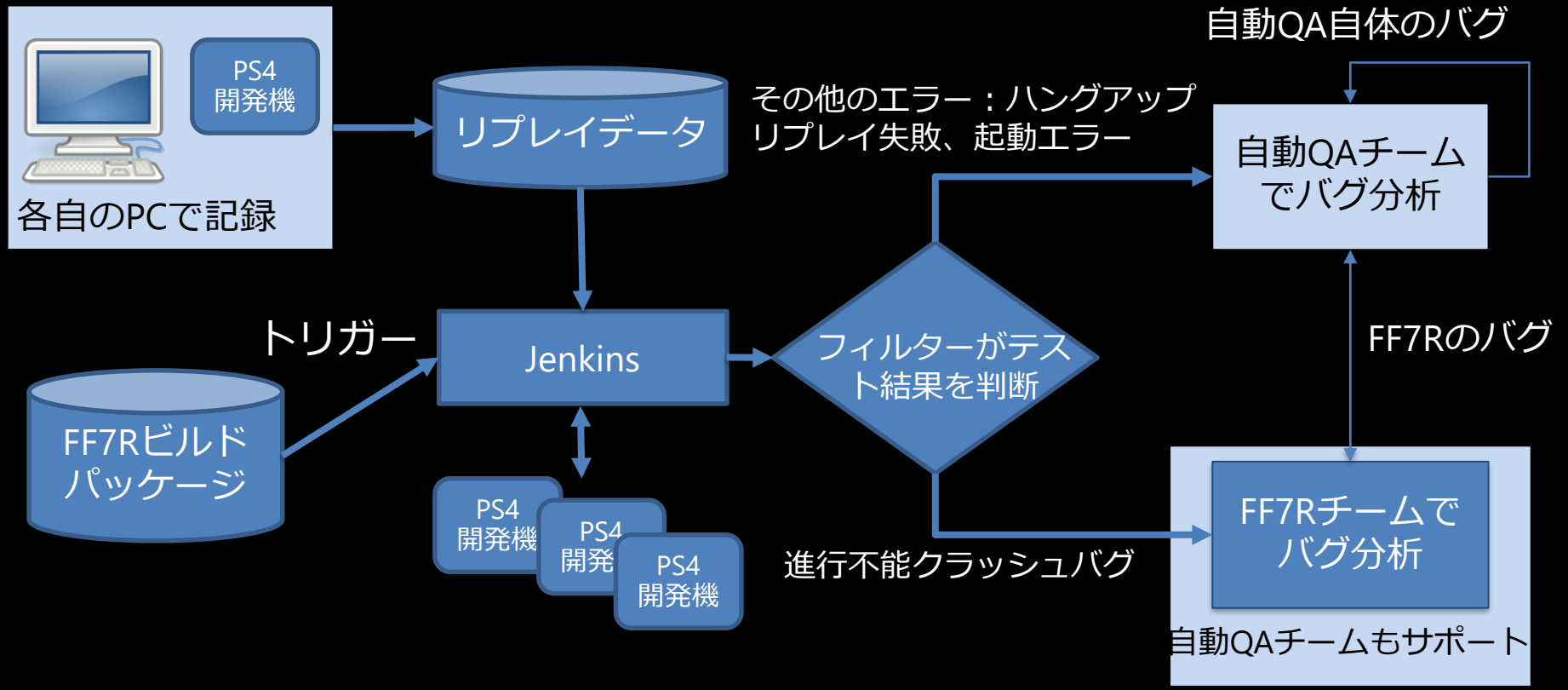
“FF7R”自動QAシステム

- システム要件
  - 各章、全章の通しプレイを記録再生できる
  - 上記においてデグレードを検知する
  - 普通のQAさんでも使えるようにする
- 技術的なゴール
  - ゲーム依存のコードと汎用の自動リプレイのコードは分離する
  - 汎用の自動リプレイで処理が難しい部分は適時BOTを作成、利用する

デモ



運用イメージ



自動メール通知

To Check

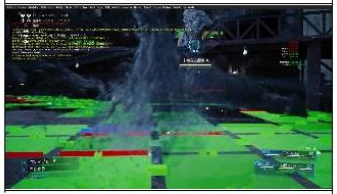
Build # 312 – Failure:

Check console output at [redacted] to view the results.

Build revision

[redacted] 177959 Engine/Source : 177962″

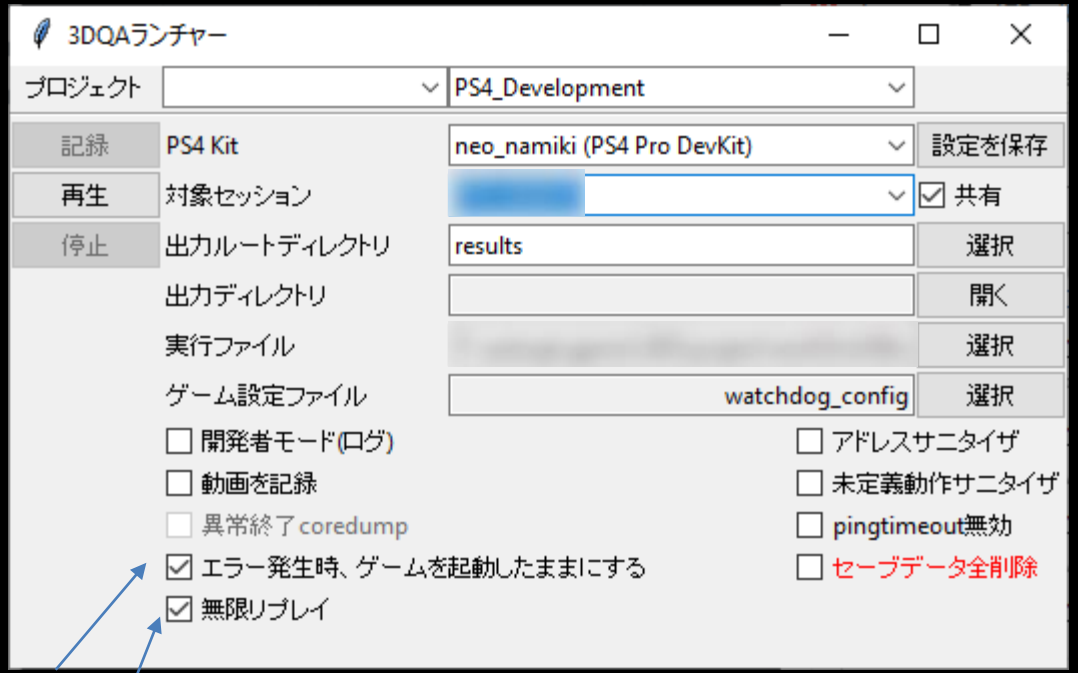
Test summaries

Test name	[redacted]	[redacted]	[redacted]
Execution time (hh:mm:ss)	1:31:09	1:31:09	1:31:09
Screenshot			
Recording Video			
Error type	Crash	Crash	Crash

3 tests failed.
FAILED: [redacted].実行 : 1回目

Error Message:
クラッシュ

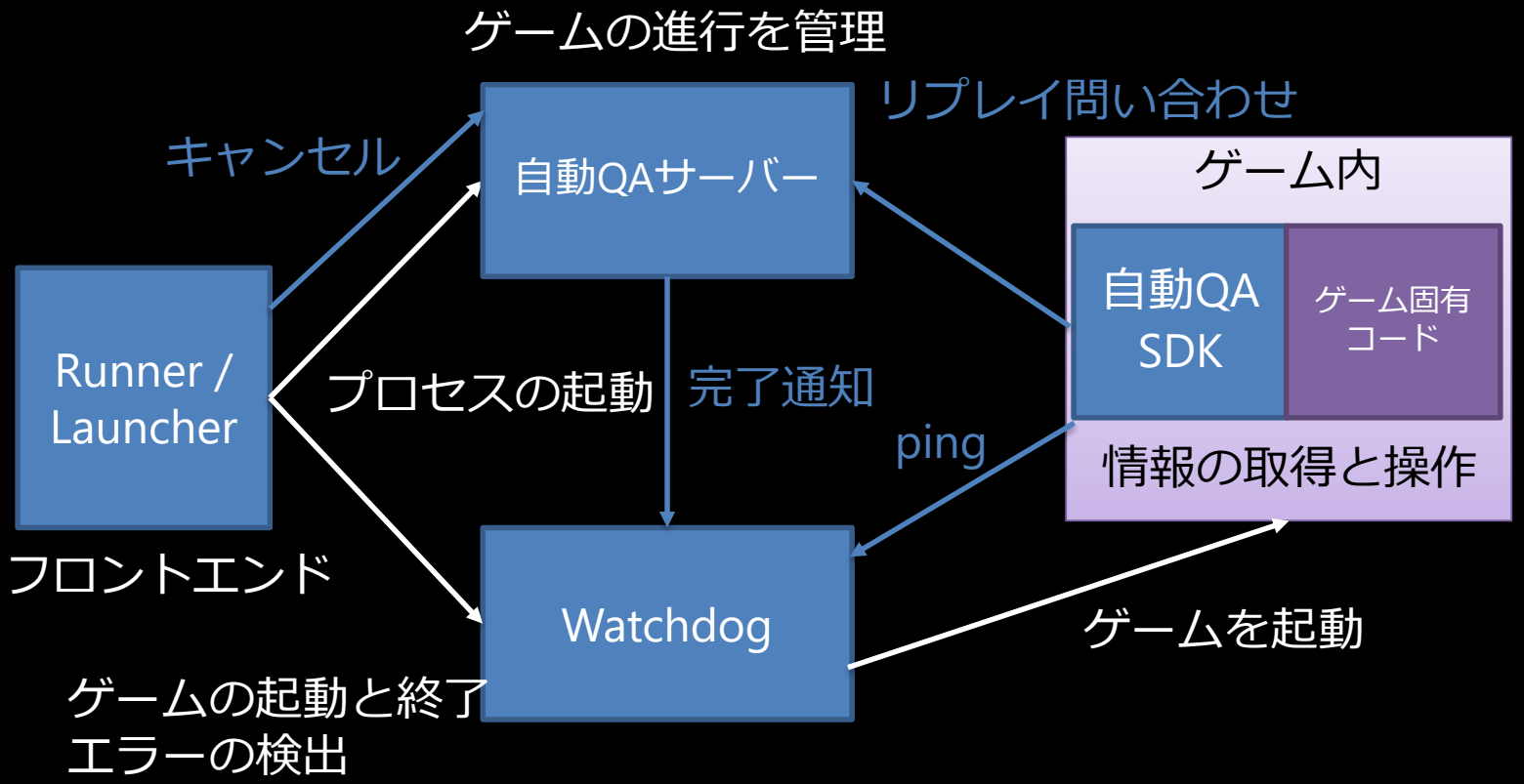
自動無限リプレイでレアバグ再現



2/260で再現
以下で再現：
1. 4日間回し続ける
2. 151回目に初再現

自動QAシステムのアーキテクチャ

アーキテクチャ



Runner / Launcher

Runner / Launcher

Watchdogと自動QAサーバーの制御

リプレイの結果を取得する

Runner ： GUIがないJenkinsのフロントエンド

Launcher ： 記録と再生に使用するGUI

Watchdog

Watchdog

ゲームの起動終了

複数のプラットフォームを抽象化してサポート

クラッシュとハングアップの検知

動画の記録

　PS4ではプラットフォームの機能を使用

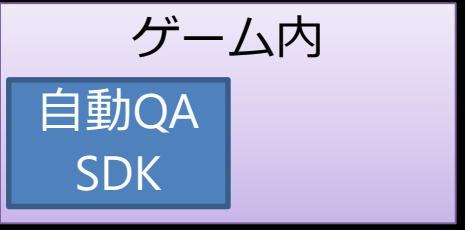
コアダンプの収集と作成

コアダンプからスタックトースを抽出

自動QAサーバー

自動QAサーバー

記録時と再生時にゲームの進行を管理
記録されたリプレイ情報と現在のリプレイを同期
ゲームの制御の問い合わせに対応 (再生時)
- 位置の制御の問い合わせ
  - パスファイディングはサーバー側で実行
- シーケンスの開始の問い合わせ (インタラクション)
- シーケンスのスキップの問い合わせ
- アクションの問い合わせ (ボタンが押されたとき)



自動QA SDK

自動QAサーバーやWatchdogとやり取りする

ゲーム固有コードから呼ばれるデリゲート関数を定義している

デリゲート関数のデフォルト動作を定義している

ゲーム固有コード

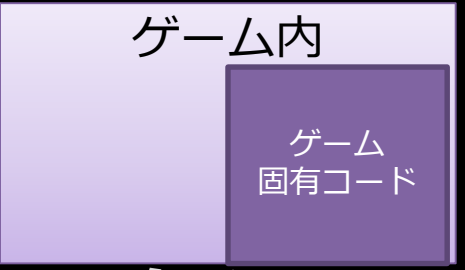
自動QA SDKのデリゲート関数を実装している

サブシーケンス(後述)を定義している

いくつかのサブシーケンスで使用するBOT群を定義している

キーバインドを定義している

キャラクター制御を実装している



ゲーム固有コード

自動QA SDKのデリゲートを実装

GameTick （フレームの開始時）：

- リプレイのアクション（キーの押下）
- スキップの要求
- BOTの実行
- アクションの記録 （キーの押下）

UpdateSubsequences （フレームの終了時）：
現在のシーケンススタックを決める

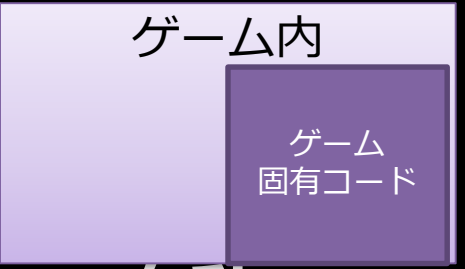
ゲーム固有コード

自動QA SDKのデリゲートを実装

OnInit：最初に一度だけ呼ばれる

OnServerInstructions：自動QAサーバーからリクエストがあった場合に呼ばれる

InjectInputDirections：プレイヤーの向きと位置を制御する



ゲーム固有コード

自動QA SDKのデリゲートを実装：デフォルト実装をオーバーライドする可能性がある

GetPawn：プレイヤーのPawnを取得する

GetCurrentLevelName： 現在のレベル名を取得する（マップに使用する）

GetCameraDirection

GetInputDirection

GetWorldDirection

SetForwardDirection

自動QA SDK　詳細

リプレイのためのゲームステート

位置：
- N次元座標（現在は3次元）
- レベルのID
- ブロックの位置

時間：
- フレーム
- ゲーム時間
- 実時間

制御ベクトル（ジョイスティックの値）

ワールド座標での速度と向き

イベント：
- アクション：キーの押下
- 時間の同期モード：フレーム、ゲーム時間、開始点
- PushとPopのシーケンス
- Extra updates：最上位のシーケンスで利用するゲーム固有のデータ（リプレイ時に利用）

サブシーケンス

ゲームの状態を表現：バトル、カットシーンなど

スタックとして実装される

例：バトル、チュートリアル（バトル中にチュートリアルメニューが出る）

個々のサブシーケンスは優先度で制御される

スタックの最上位が処理対象サブシーケンス

サブシーケンスの例

```
162 3732 push [16] ['cutscene (EV_MAKO1_0040s)']
163 3733 action ['ActionType: 0', 'KeyId: 7', 'Repeat: True']
164 3750 action ['ActionType: 1', 'KeyId: 7', 'Repeat: False']
165 3753 action ['ActionType: 0', 'KeyId: 7', 'Repeat: False']
167 3827 action ['ActionType: 1', 'KeyId: 7', 'Repeat: False']
178 3920 action ['ActionType: 1', 'KeyId: 7', 'Repeat: False']
178 3920 pop [16] ['cutscene (EV_MAKO1_0040s)']
178 3920 push [17] ['battle (btsc_mako1_020)']
179 3922 push [17, 0] ['battle (btsc_mako1_020)', 'tutorial']
181 3958 action ['ActionType: 0', 'KeyId: 15', 'Repeat: False']
182 3963 action ['ActionType: 1', 'KeyId: 15', 'Repeat: False']
 183 3967 pop [17, 0] ['battle (btsc_mako1_020)', 'tutorial']
```

※上記はJSONだが実際の通信や永続化にはProtocol Buffersを使用

サブシーケンス

自動QAサーバー上でのサブシーケンス

ID：文字列

フラグ：

種別	説明
Moveable	このサブシーケンス中はプレイヤーは自由に動ける
Skippable	自動QAサーバーからリクエストがあった場合、このシーケンスをスキップできる
Awaitable	ただ待っているだけでこのサブシーケンスは完了する
Startable	自動QAサーバーからリクエストがあった場合、このサブシーケンスを開始できる

サブシーケンス　続

自動QAサーバー上でのサブシーケンス

フラグ：

種別	説明
Reorderable	このサブシーケンスは自分の親のサブシーケンスの別のタイミングでも起こりうる
Optional	リプレイ時になくてもエラーとならない
Unique	セッション内で一度だけ発生する
ClientControlled	自動QAサーバーからのリプレイではなく、クライアント側でシーケンスを制御する
NonRelocatable	リプレイ時にこのサブシーケンスは移動させられない

Movable

Movableでは自動QAサーバーが動作を制御できる

リプレイ時の経路で以下を使用する：

記録時の入力と座標

現在のモーションによって異なった制御を使用できる

　通常のモーション、はしご、ぶら下がり移動など

Movable

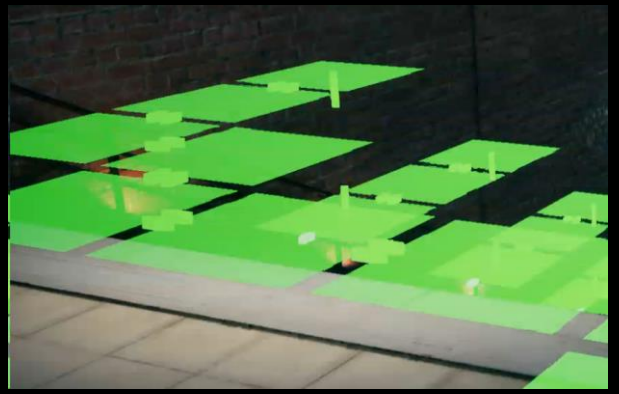
記録時の経路からはみ出してしまった場合：

記録時のサーバーマップを使って、正しい場所に戻る経路を探索する

上記がうまくいかなかった場合

- クライアントに委譲し、UE4のナビゲーションメッシュを使って経路を探索する

Movable



サーバーマップ

サンプリングされた3次元の位置

マップはレベルごとに1つ

サンプリングされた座標を格納するハッシュ

ブロック間の接続情報と、接続のメトリクス

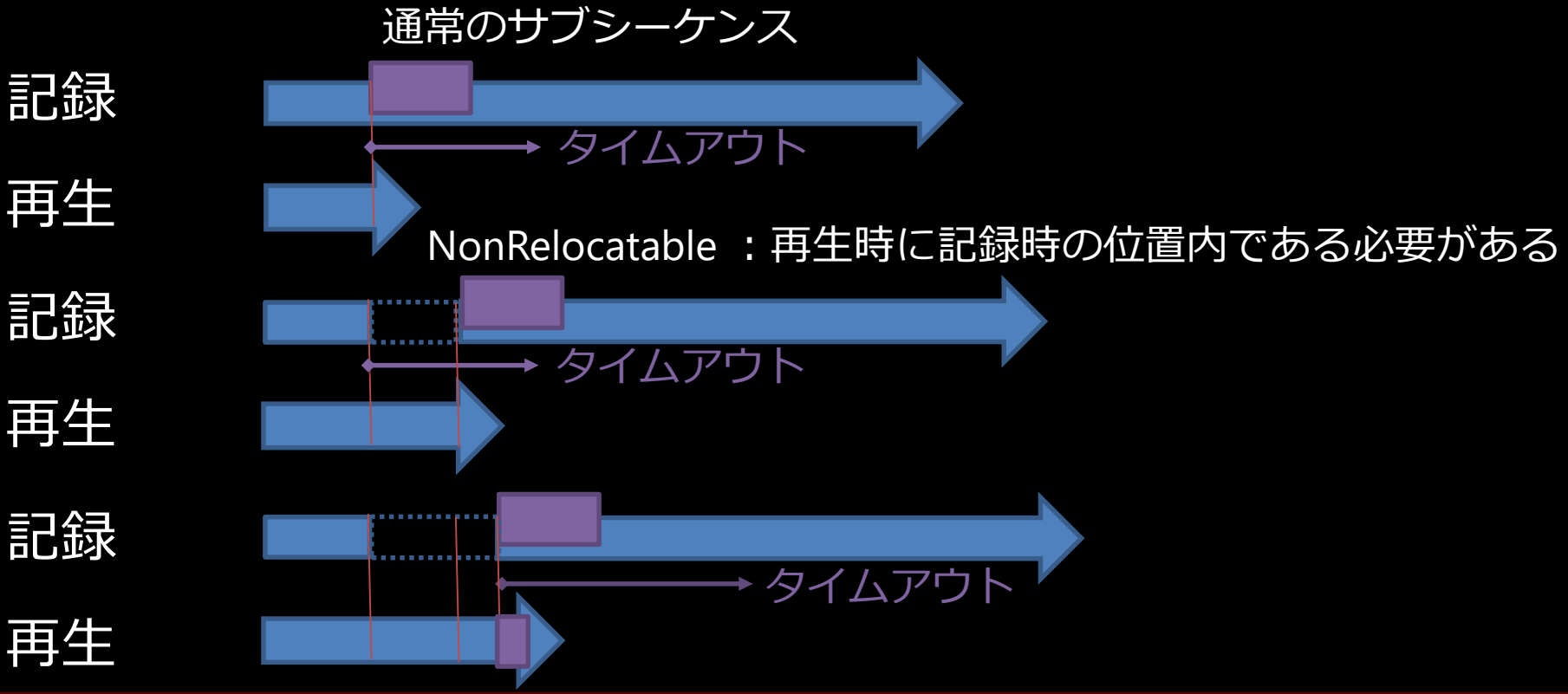
Movable

探索：

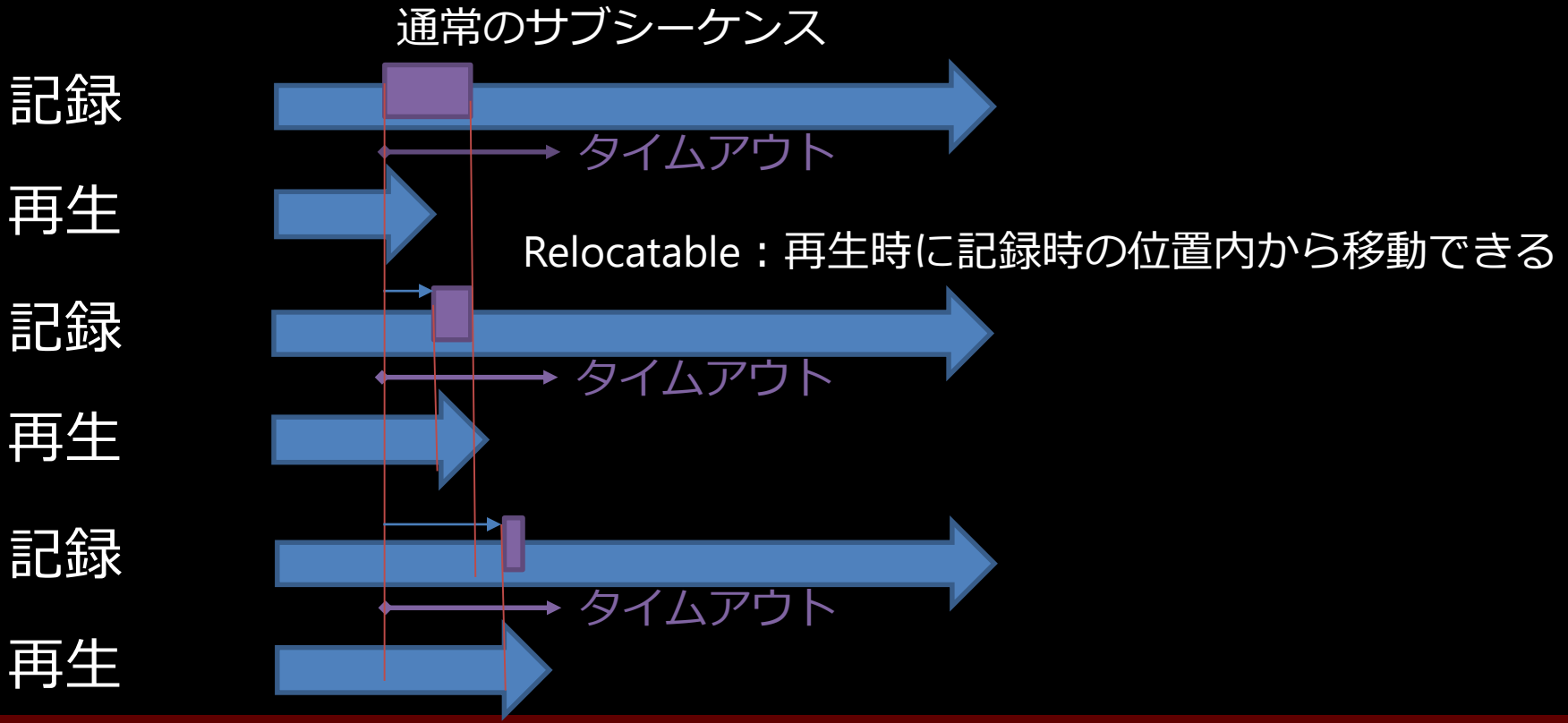
未探索の方向をランダムに選び、壁で進めなくなるもしくは既知の経路まで進む

既知のブロック内でうまくいなかった場合、もっとも近くの未探索の経路を探す

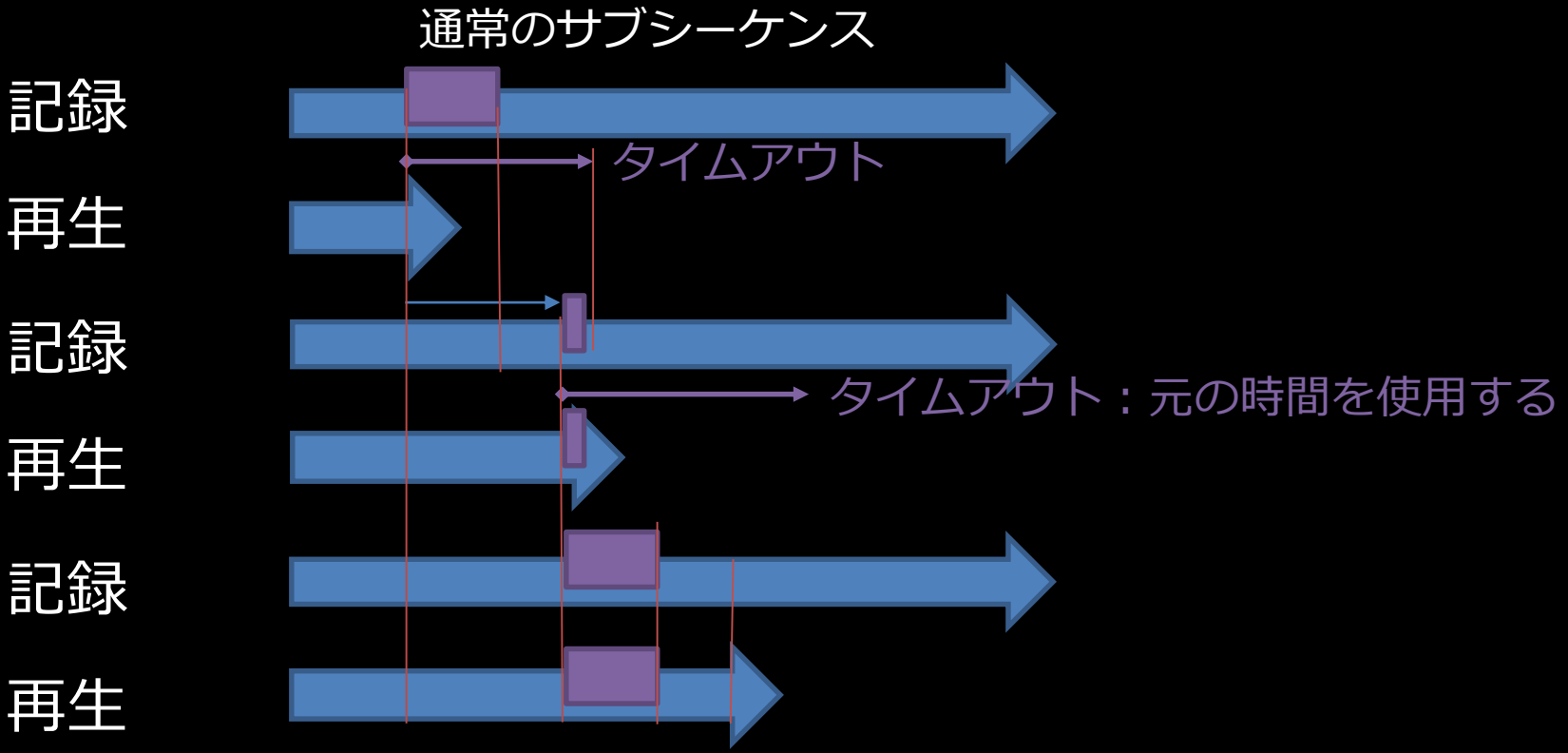
同期のメカニズム：NonRelocatable



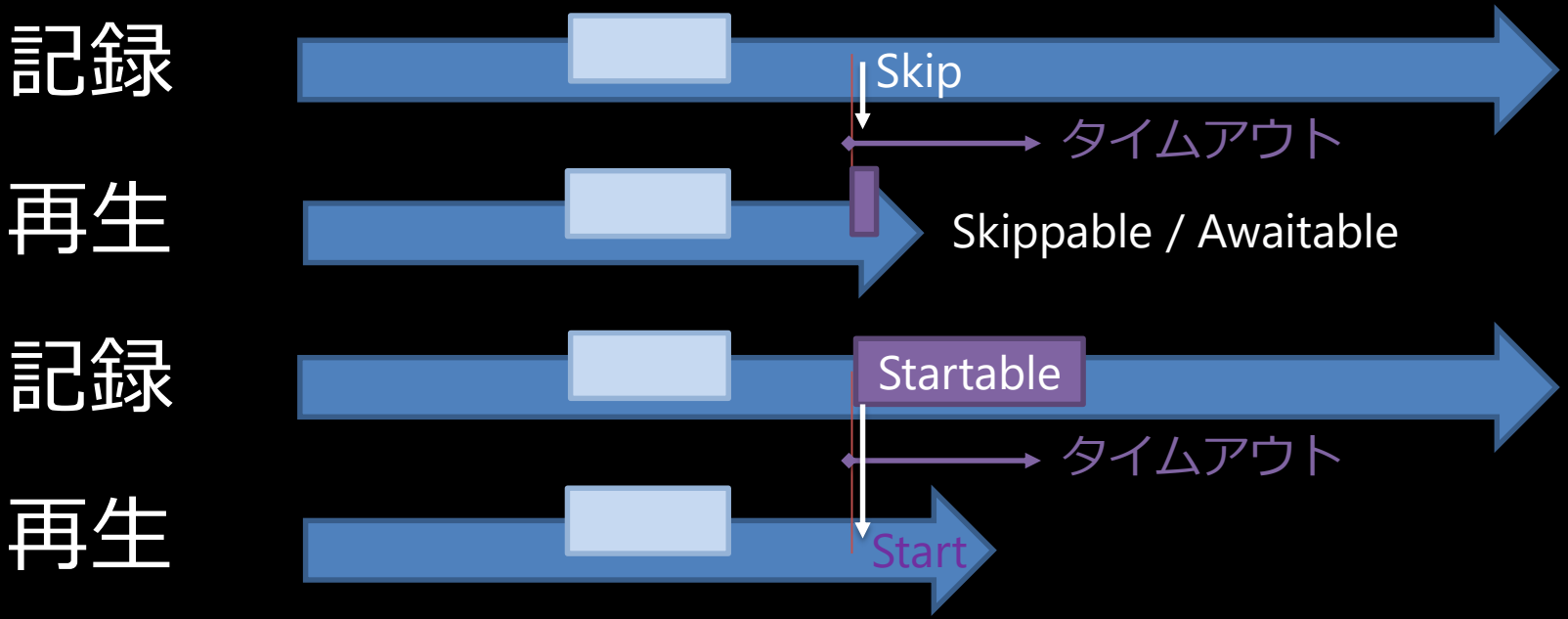
同期のメカニズム：Relocatable



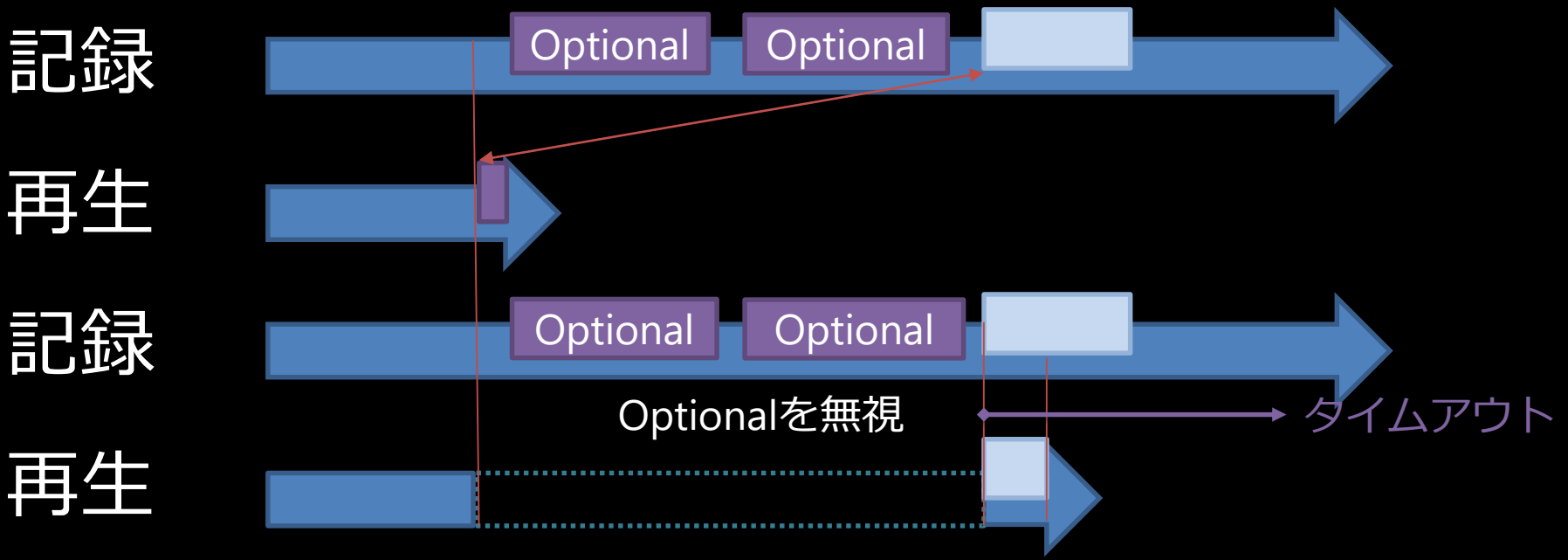
同期のメカニズム：Relocatable



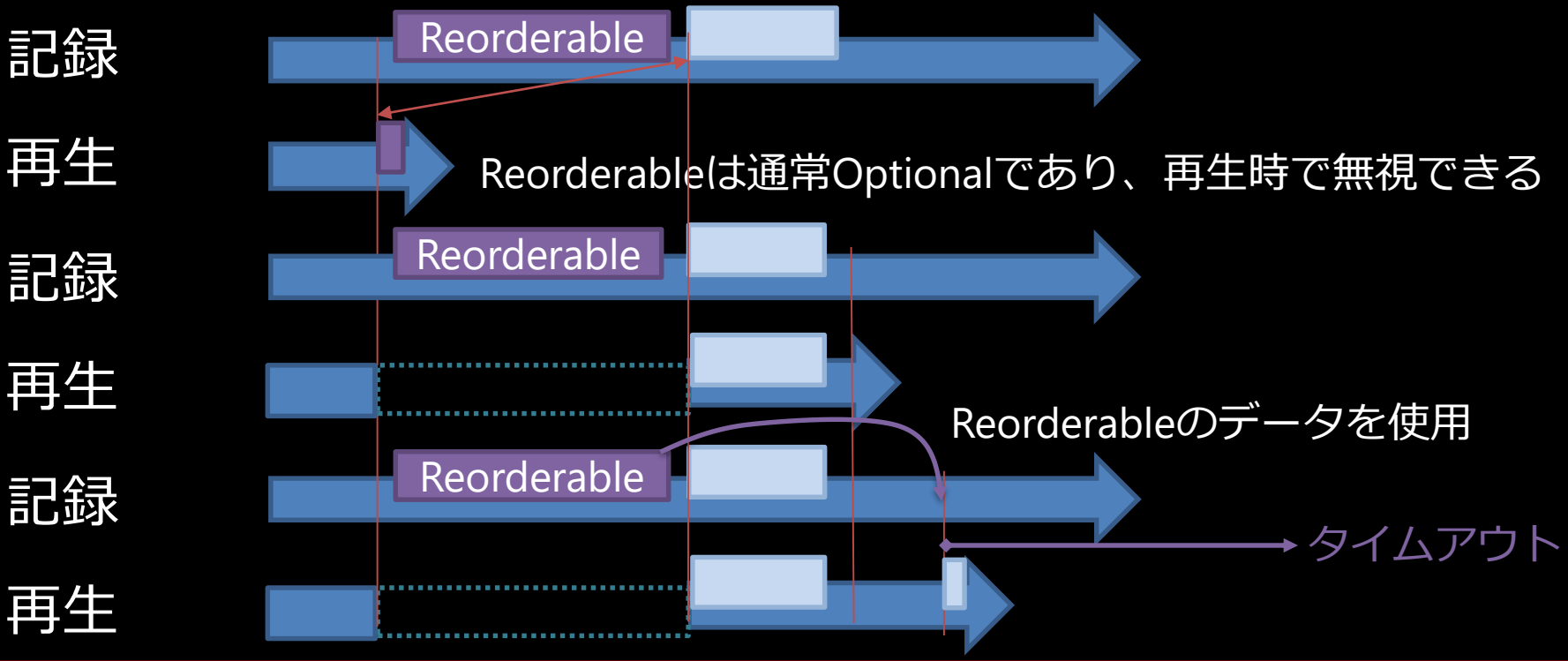
同期のメカニズム：Skippable/Startable



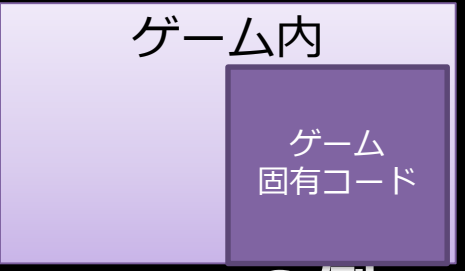
同期のメカニズム：Optional



同期のメカニズム：Reorderable



ゲーム固有コード　詳細



BOTs

BOTの例：

バトルBOT

近くの敵に近づき攻撃する、時々魔法で攻撃する

インタラクションBOT

オブジェクトやNPCとインタラクションする

1. インタラクション可能な要素を見つける
2. 要素を見る
3. 要素へ移動する
4. 可能なら要素とインタラクションする

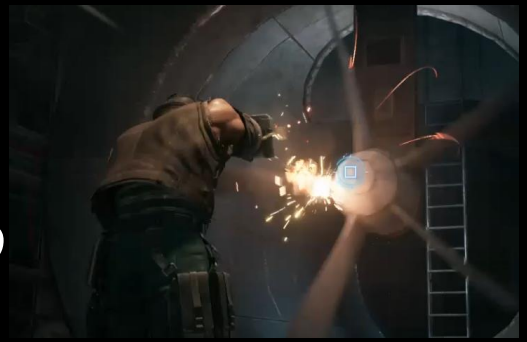
BOTs

BOTの例：

追跡BOT：NPCの進む経路に従う

指定範囲破壊BOT：バレット

ぶら下がり移動BOT



BOTs

BOTの例：

PositionControlledGame BOT

　ロボットのアーム制御

レバー上げ下げミニゲームBOT

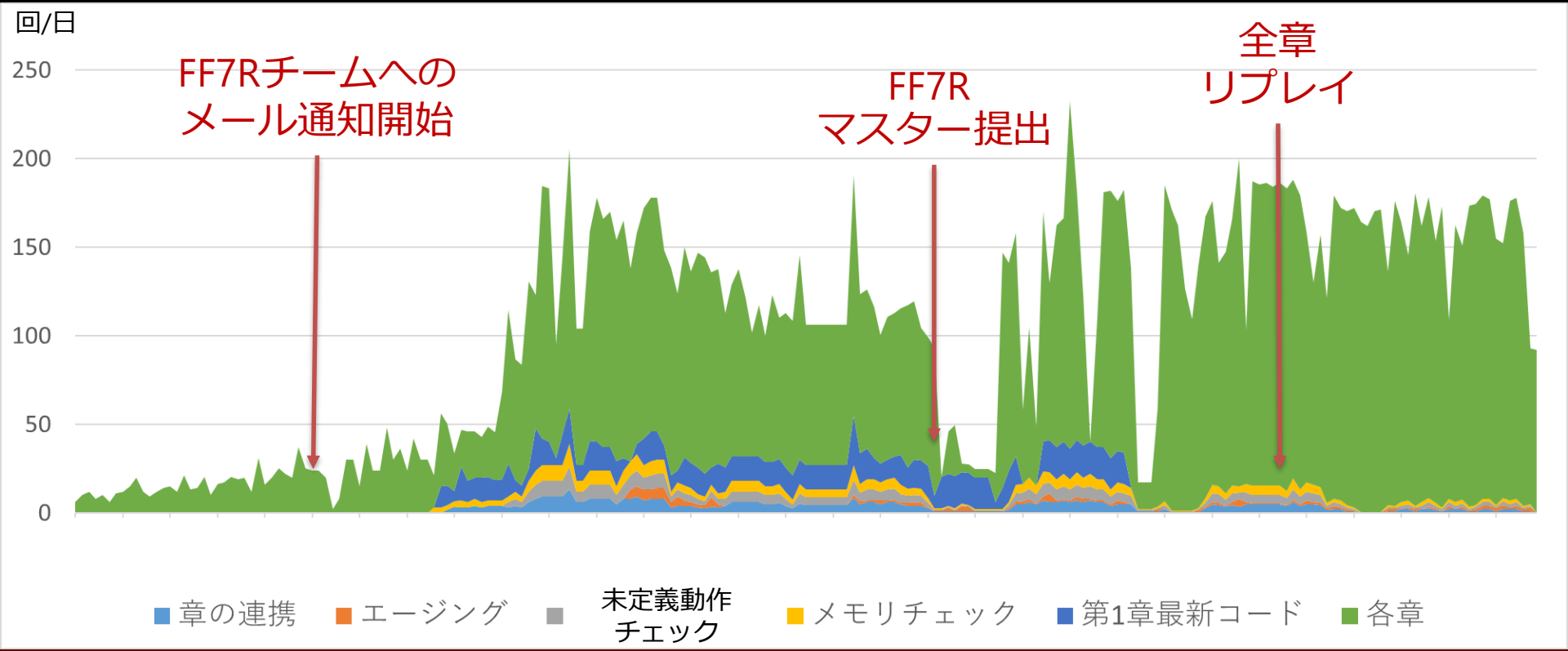
同時操作ミニゲームBOT

テレポートBOT

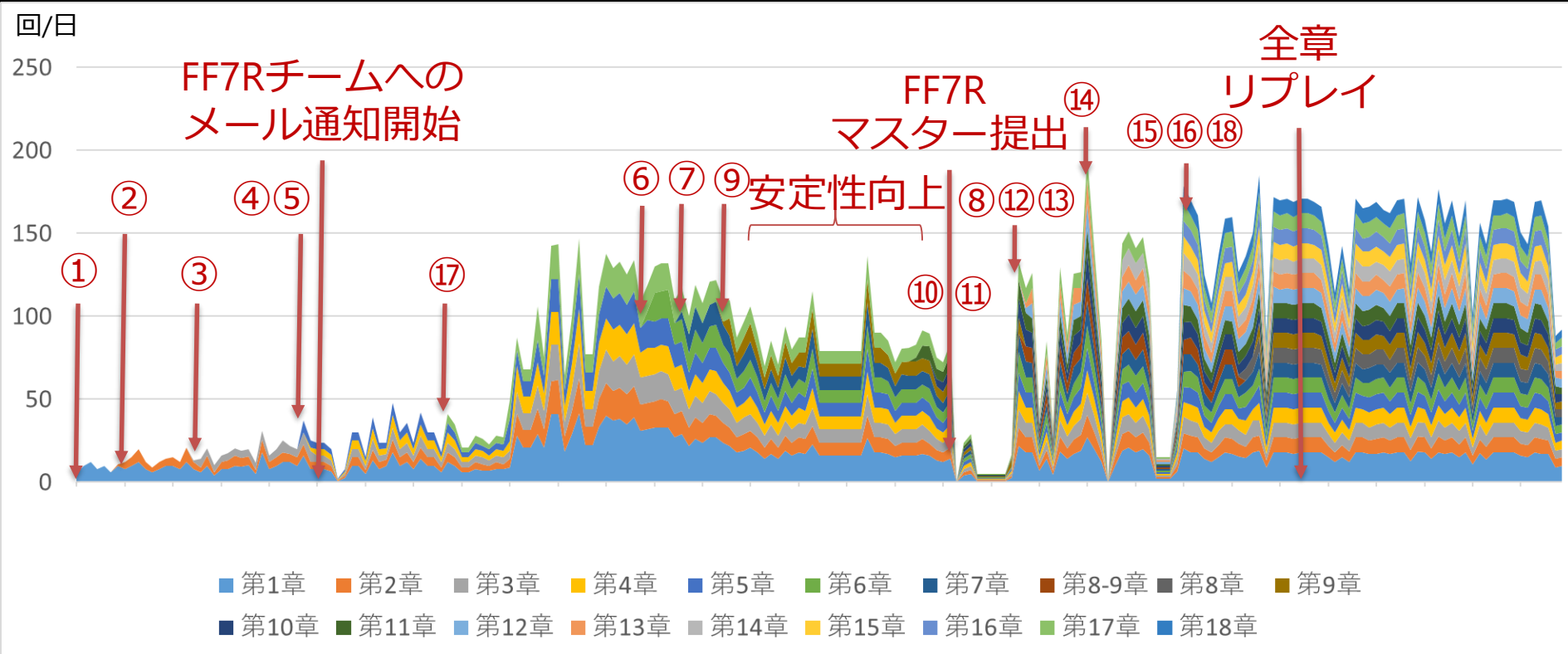
　エアリス宅でのミニゲーム

自動ＱＡシステムの定量評価

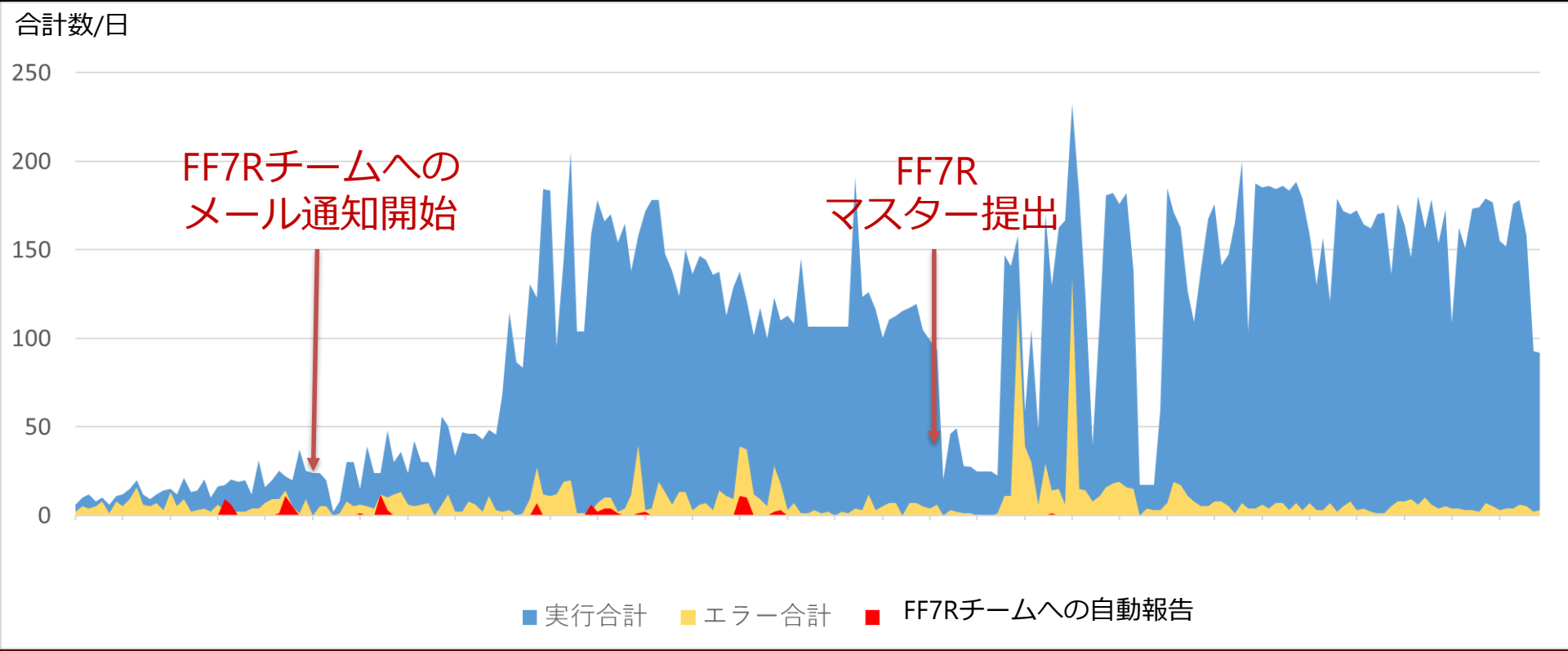
日当たりのテスト実行総数



章ごとのテスト実行数の内訳



エラーと通知の内訳



自動QAシステムの課題と総評

運用上の課題

- 新規開発によるバグ
  - FF7Rと自動QAシステムの両者とも新規開発
  - バグの切り分け
    - FF7Rのバグ
    - 自動QAシステムのバグ
- 誤検知
  - 上記によるバグの誤検知

システム上の課題

- ランダム性
  - FF7R
    - フィールドで物理的に嵌まってしまった場合に自動的に戻す
  - 自動QAシステム
    - 再生シーケンスの戻りは未対応
- 新たなインタラクションとミニゲーム
  - インタラクション、ミニゲームごとに追加開発が必要
- リプレイ失敗時の解析
  - FF7Rの内部実装に詳しくないと解析は難しい
  - プログラマであれば良いが、普通のQAの方では使いづらい

自動QAシステムの総評

システム要件に対する成果

- QA稼働時間外でも繰り返し実行し、通しプレイを保証
- 手動のQAでは見つけることが難しいレアバグを検出・再現

技術的なゴールに対する成果

- あるプラットフォームで記録したリプレイデータを複数プラットフォームで動作
  - ※内部検証用に作成したテストゲームにて
- プラットホームとゲームの依存の分離により、リモートワークの環境でも適切に開発を継続

商標

- "PlayStationファミリーマーク"、"PlayStation"、"PlayStation4ロゴ"、"DUALSHOCK"および"PlayStation Plusアイコン"は、株式会社ソニー・インタラクティブエンタテインメントの登録商標または商標です。
- Jenkins は、SOFTWARE IN PUBLIC INTEREST, INC. の米国およびその他の国における登録商標もしくは商標です。