とある旧システムがあるんだが AWS Lambda使って刷新しようと思う

2016/06/22 Gunosy Beer Bash#6

yu0819ki@CyberZ

もくじ

- じこしょうかい
- F.O.Xについて
- 媒体連携について
- AWS Lambdaでどう解決したか

一言でまとめると・・・

F.O.Xの媒体連携バッチを
AWS Lambda使って刷新しようとしてる話

をします。

じこしょうかい

Name: 木村 幸弘 a.k.a yu0819ki
Birthday: 1985.08.19
Age: 30
Birthplace: 北海道
Job: ~~技術総務~~リードエンジニア
Hobby: 音ゲー、読書、作曲
Motto: パンが無ければ作ればいいじゃない

F.O.Xについて

それなりに長く稼働しているサービスなので、
随時システムの刷新が必要
（しかも連携先や連携方法がどんどん増えるし変わるし…）
（それとは別に新機能の開発も行われているので、その対応とか）

【刷新が必要な箇所の例】

- 計測基盤
- 集計基盤
- UI（画面）
- 媒体連携

媒体連携について

【「媒体」とは？】

- 広告の表示面を提供するサービス/サイトのこと
- 「媒体社」、「媒体サービス」のように表現することもある
- たとえば
  - Gunosy
  - Facebook
  - Twitter
  - YouTube(TrueView Ads)
  - nend
  - 特定のスマホアプリのこともある

媒体連携について

【F.O.Xの媒体連携数は世界最大級！】

- ここでいう媒体連携数は、F.O.Xで計測された内容を通知・送付することのできる媒体の数
  - キックバックだとかポストバックだとか言われる連携

「To Media」の連携

媒体連携について

【対して「From Media」の連携が存在する】
（ここを担当してます）

- F.O.Xで取れなくて、媒体側で取れる数値を取り込みたいため
  - ImpressionとかCTRとか
    （F.O.Xでトラッキング出来るのはクリックされてからの話なので）
  - F.O.Xに取り込むことで各媒体の広告効果比較がやりやすくなる
    （各媒体の管理画面を別タブに開いたりExcelに延々貼り付けなくて良い）
- この連携は先の媒体連携数のうち全部ではありません
  （あわよくば全部したい！）

媒体連携について

【何をどうやってやるのか？】

1. 媒体からデータを取ってくる
   - 媒体A：アクセスキーが発行され、APIを叩いて取得（JSON）
   - 媒体B：ログイン用のID/Passが発行され、CSVをダウンロード
   - 媒体C：管理者ログイン用のID/Passを利用して、スクレイピング
   - SDKが提供されている場合もある（Twitter, Facebook, etc...）
2. DBに保存
3. 集計（Hourly, Daily, Weekly, Monthly）
4. 集計されたデータを画面側のAPIで取得、表示

媒体連携について

【従来の構成】

- CakePHP 2.x
- PHPのバージョンは古め
- 実行はcron任せ→Rundeckに移行
  - 実行サーバが1台しかないため、同時に動かすジョブはせいぜい5つ
  - 媒体と弊社での実績を元にcrontab書いている（秘伝のタレ）
    - メモリを食うジョブを同時に動かすとメモリが悲鳴をあげるので、パズルが発生している・・・
  - もっと並列に動かせるはず！

これをAWS Lambdaで置き換える

AWS Lambdaでどう解決したか

【開発に至る経緯】

- 既存のバッチ処理が遅くて集計に間に合ってないヤバい
- PHPもフレームワークもバージョン古すぎるヤバい
- 対応できる人少ないヤバい

媒体連携数増やしたいのに
これは致命的！！

AWS Lambdaでどう解決したか

【解決したいこと】

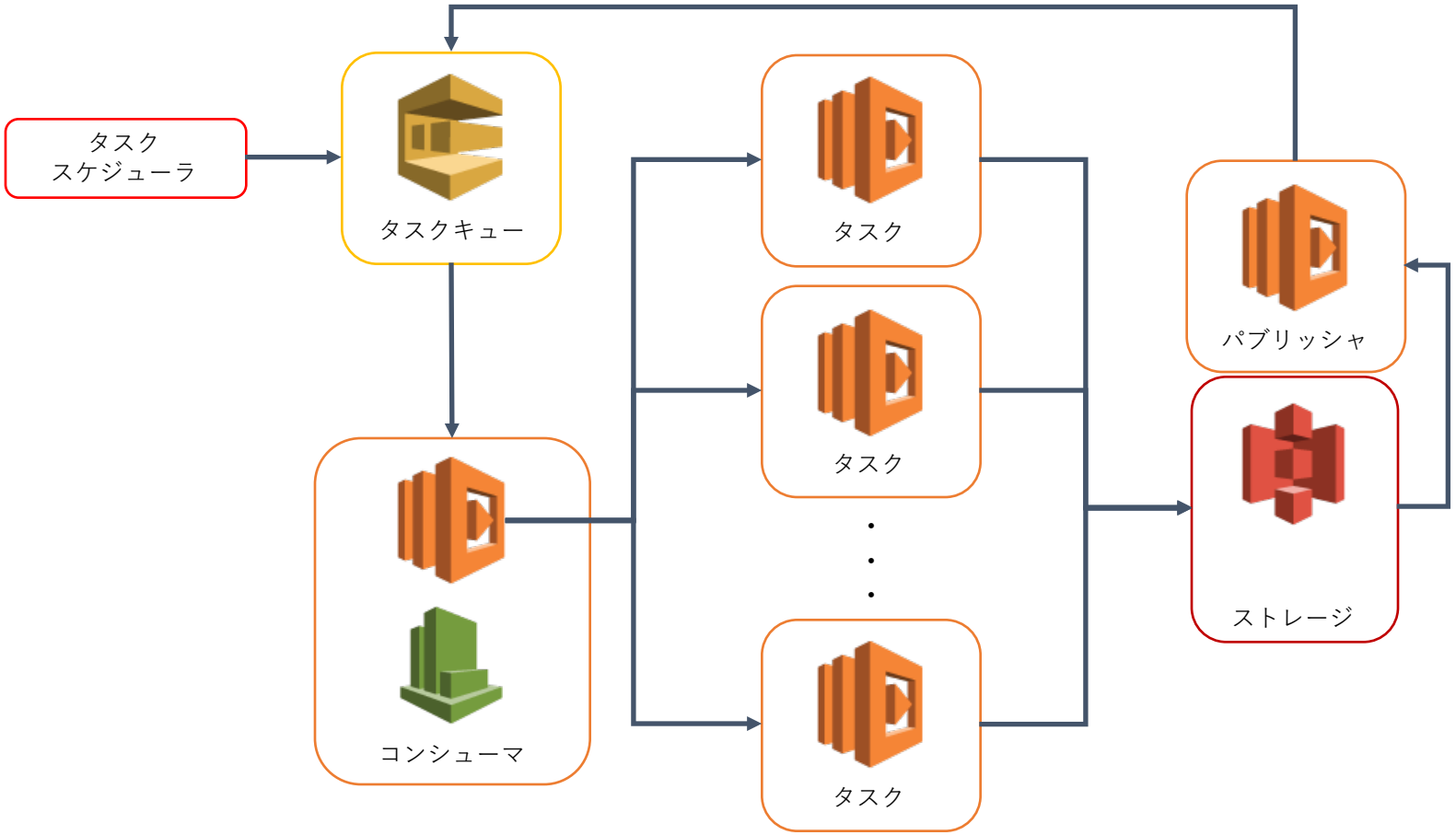
- できるかぎり並列に捌いて、全体の処理時間を短くしたい
- メンテコスト下げたい
- 秘伝のタレを捨てたい
- ~~Node.js使いたい~~

AWS Lambdaでどう解決したか

【できるかぎり並列に捌いて、全体の処理時間を短くしたい】

- 1個1個のタスクを小さく設計
- タスクは一旦SQSに突っ込む
  - 一番最初の起点となるタスクはRundeck等のスケジューラで突っ込む
- 1分に1回起動するLambda関数(コンシューマ)が
  SQSからタスクを引き取る(最大30個程度)
  - このシステムだけ動かす分には90個程度まで可能だが、他システムを圧迫する可能性がある(Lambdaの制限)
  - SQSからは一度に10件ずつしか取れないので、ループ回す
- 引き取られたタスクを次々invokeして、コンシューマは役目を終える
- タスクの成果物をS3に保存する
- S3イベントをソースとして
  SQSに突っ込むLambda関数(パブリッシャ)を実行する
  - 後続の処理がなければパブリッシュしない

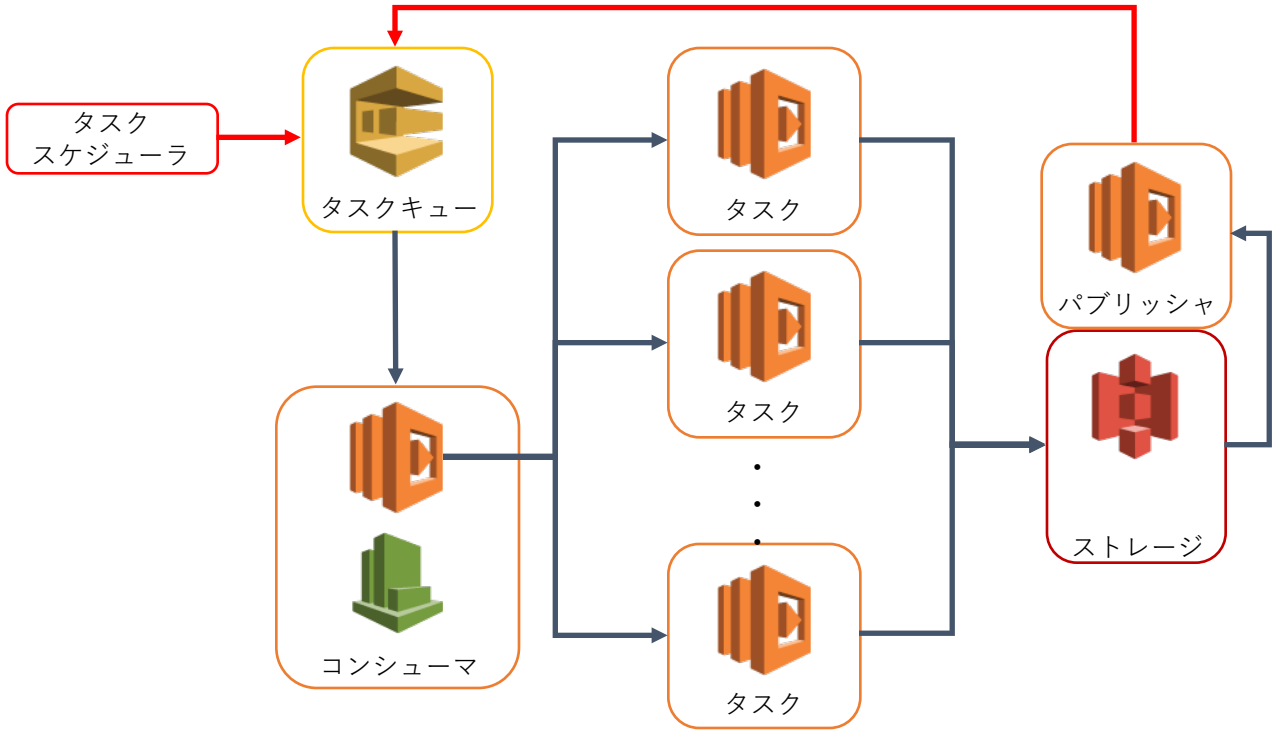
AWS Lambdaでどう解決したか



AWS Lambdaでどう解決したか

【できるかぎり並列に捌いて、全体の処理時間を短くしたい】

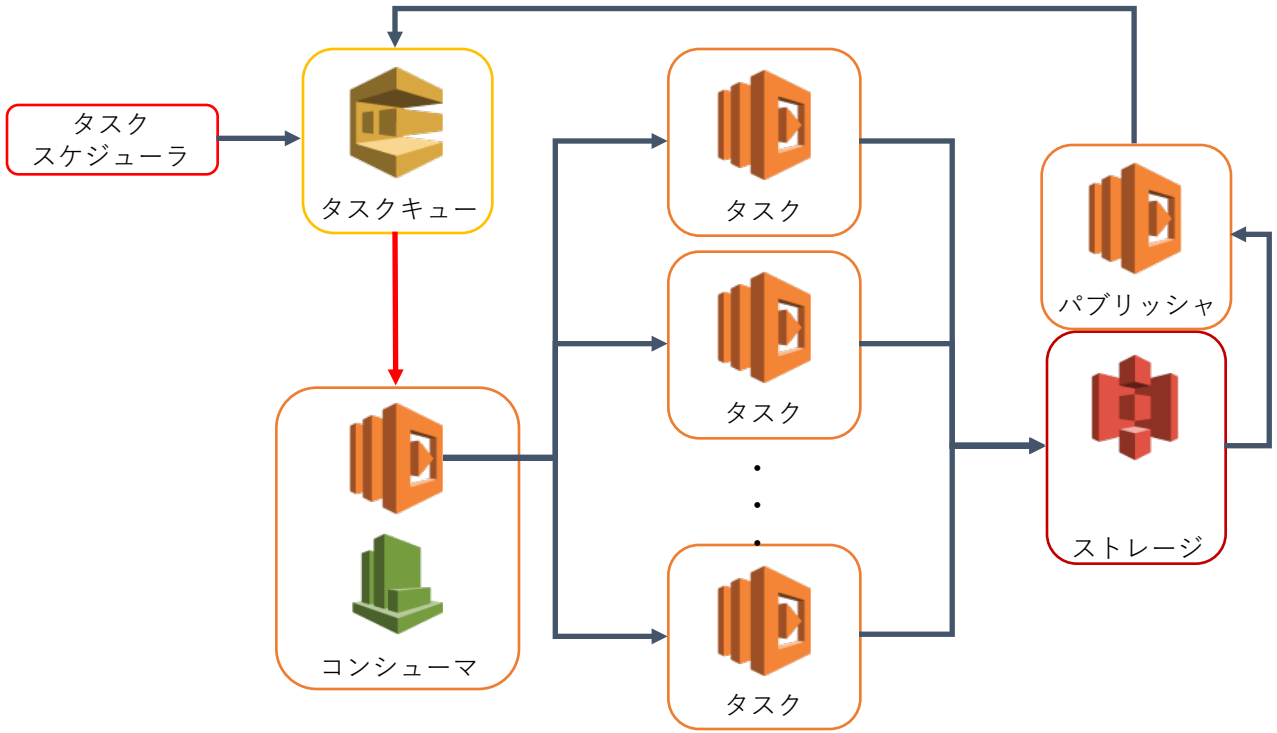
- タスクは一旦SQSに突っ込む
  - 一番最初の起点となるタスクはRundeck等のスケジューラで突っ込む



AWS Lambdaでどう解決したか

【できるかぎり並列に捌いて、全体の処理時間を短くしたい】

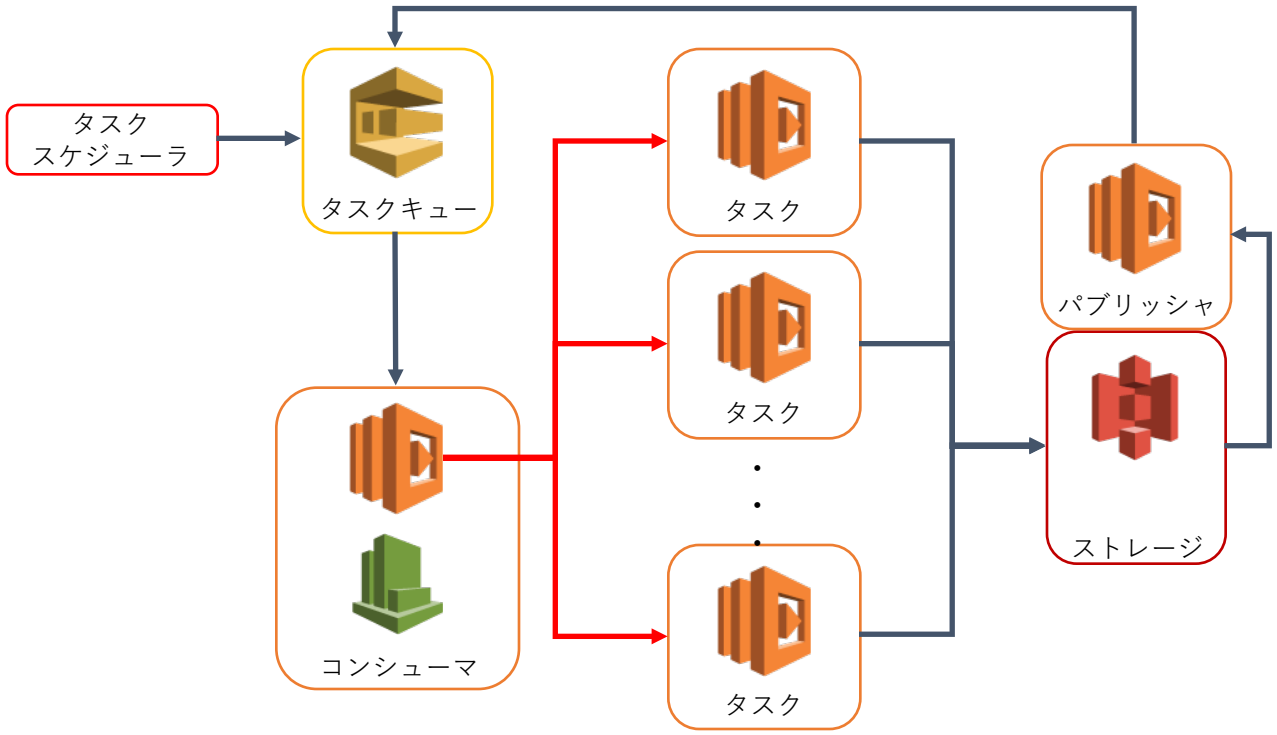
- 1分に1回起動するLambda関数(コンシューマ)が
  SQSからタスクを引き取る(最大30個程度)



AWS Lambdaでどう解決したか

【できるかぎり並列に捌いて、全体の処理時間を短くしたい】

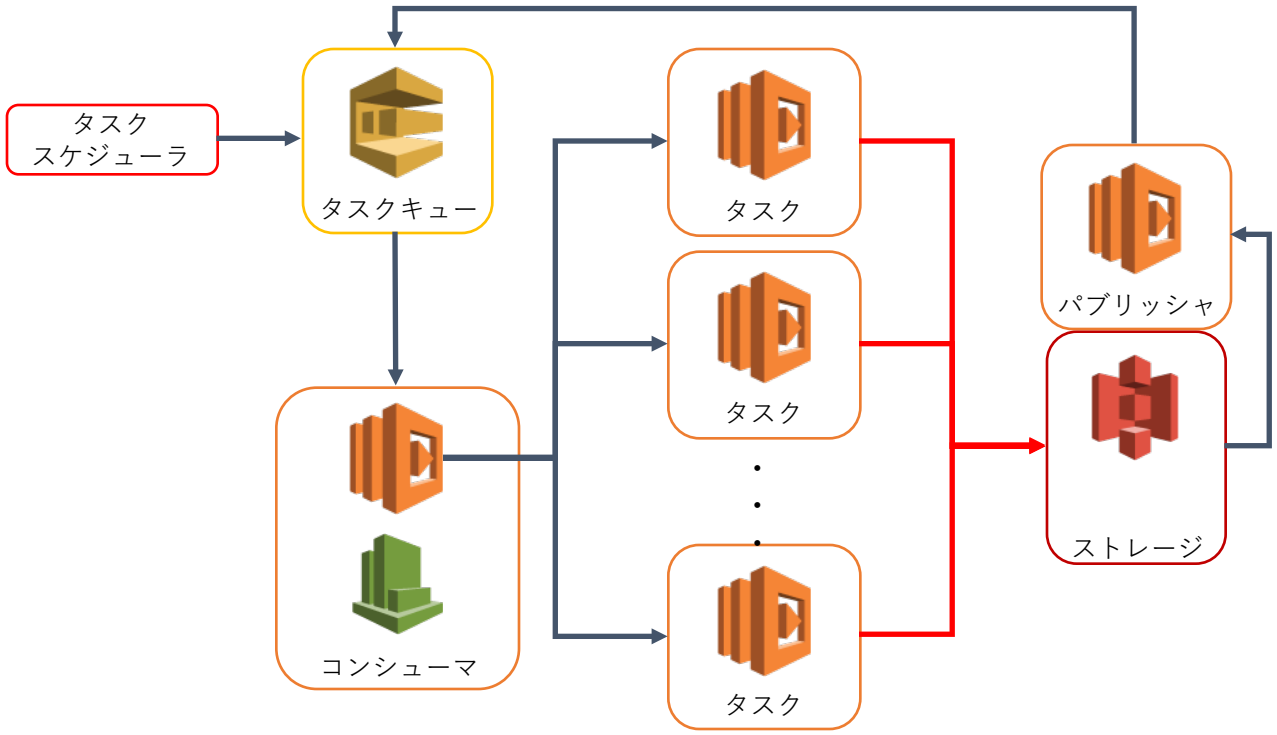
- 引き取られたタスクを次々invokeして、コンシューマは役目を終える



AWS Lambdaでどう解決したか

【できるかぎり並列に捌いて、全体の処理時間を短くしたい】

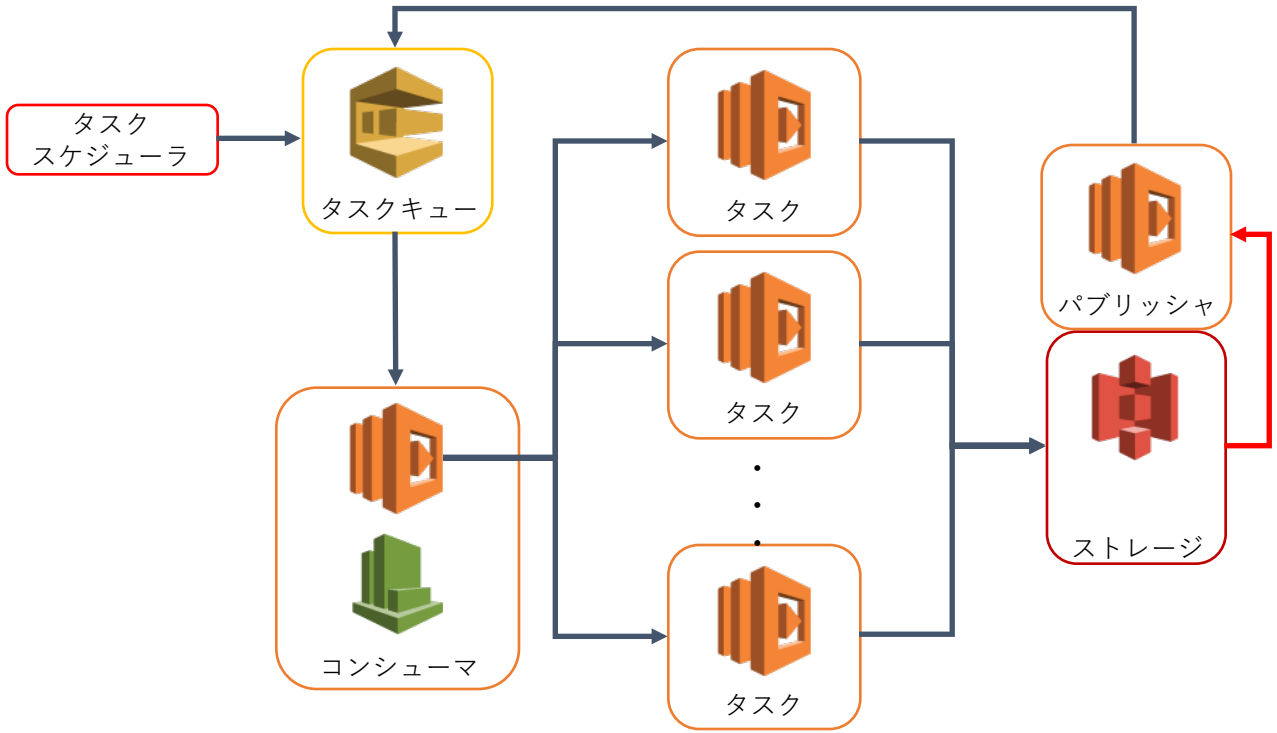
- タスクの成果物をS3に保存する



AWS Lambdaでどう解決したか

【できるかぎり並列に捌いて、全体の処理時間を短くしたい】

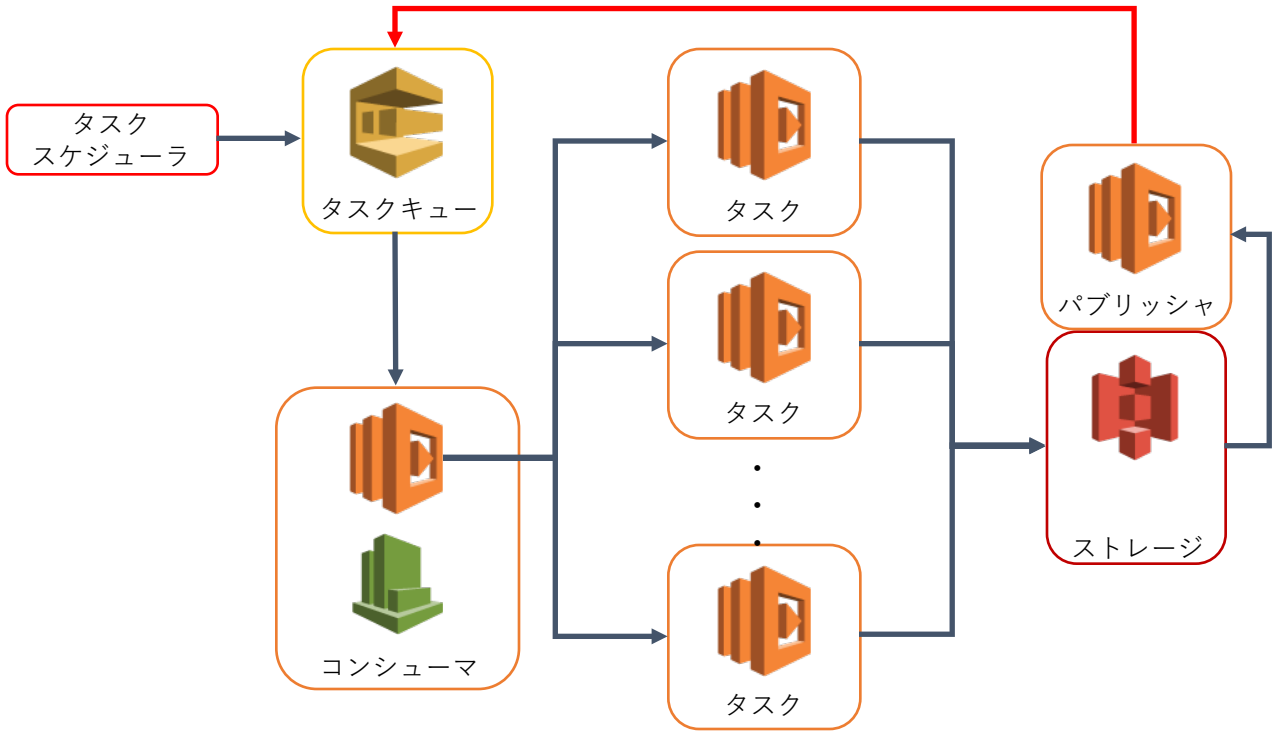
- S3イベントをソースとして
  SQSに突っ込むLambda関数(パブリッシャ)を実行する



AWS Lambdaでどう解決したか

【できるかぎり並列に捌いて、全体の処理時間を短くしたい】

- （タスクキューに戻る）



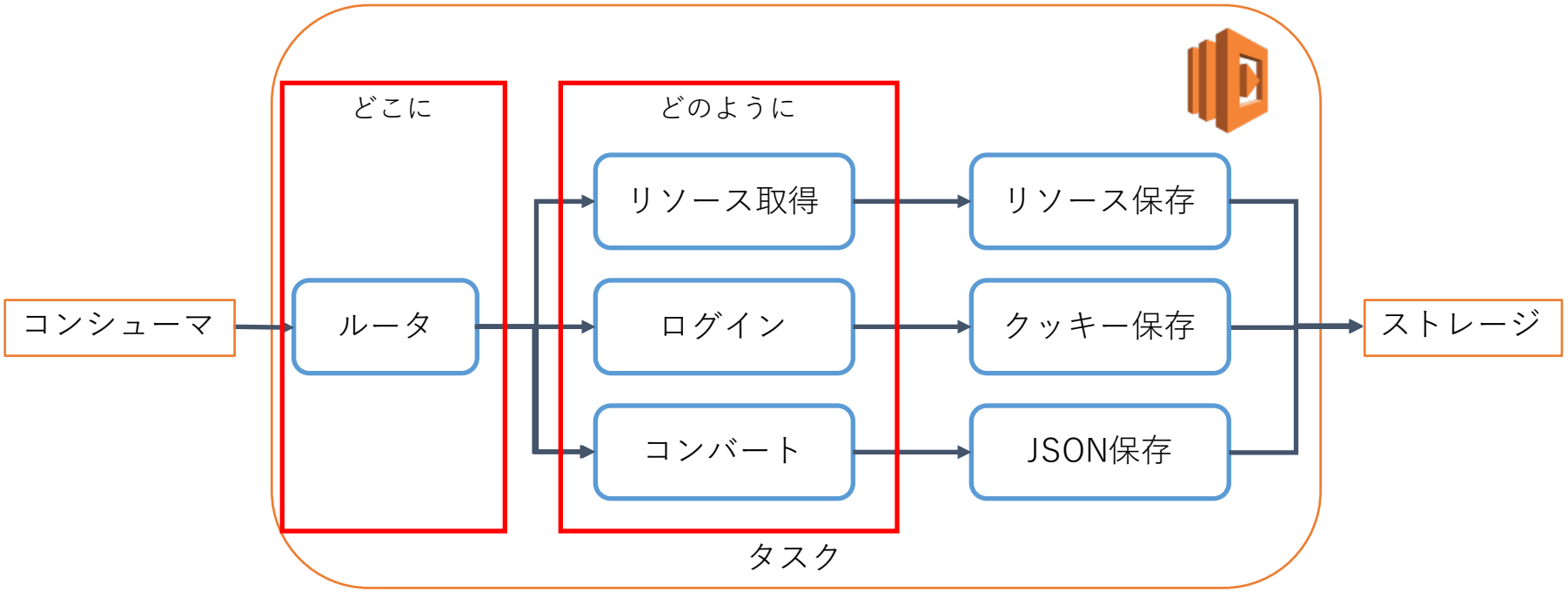
AWS Lambdaでどう解決したか

【メンテコスト下げたい】

- 「どこに取りに行くか」「どのように処理するか」
  の2点に絞ってタスクを書ける設計にした
  - ベースクラスとして用意し、原則2箇所のみオーバーライド
- 「どこに」はURL
- 「どのように」は大きく分けて5つ
  - ログイン
  - JSONを取得して、フォーマットを変えてストレージに保存
  - CSVを取得して、フォーマットに沿ったJSONに変えてストレージに保存
  - HTMLを取得して、スクレイピングして、
    フォーマットに沿ったJSONに変えてストレージに保存
  - その他変換処理

AWS Lambdaでどう解決したか

【メンテコスト下げたい】



AWS Lambdaでどう解決したか

【秘伝のタレを捨てたい】

- AWS Lambdaを中核に据えることで
  いくつかの秘伝のタレが消滅・・・！！
  - ソースからコンパイルされたPHP
  - 「いつ媒体側の処理が終わったかわからない」という理由で
    毎時実行されていたジョブ
  - 本当は毎時実行したかったけど重すぎて「とりあえず3時間周期」
    にされていたジョブの「とりあえず」
    - 本当に処理が重いなら、処理が終わり次第、
      最初から走らせればラグが最小限になる
      - そういうのcronではなかなかできない
- ついでに、中間データを溜め込むことによるディスクの圧迫や
  メモリ使用率アラートからも開放
- 新たな秘伝のタレには、なり得る・・・
  - AWSに触れる事が前提になりつつあるのでそこまで不安ではない

AWS Lambdaでどう解決したか

【躓いたこと・懸念点】

- ローカルでうまく動かすこと
  - 弊社のLT会でもちょっと話しました
    https://speakerdeck.com/yu0819ki/node-dot-js-on-aws-lambda
- ARNって何・・・
  - そもそもAWSガッツリな仕事は今回が初めてだった
- DDDっぽいことしたかったんだけどなかなか難しい
  - IoCしたいときにNode.jsだとググってもなかなか・・・
  - TypeScriptのほうが良かったかもしれない
- 俺俺フレームワーク化した
  - メンテコストには配慮した設計だが、しっかり伝えなければ
    新たな秘伝のタレになる
- いつどの処理が走っている/走っていたかを
  可視化する必要がありそう

まとめ

- 複数のサービスから情報をかき集めてくる必要があった
- 1つ1つの処理を細分化できて、並列処理をさせたかったら検討に値する
- いつどこの処理が動いているかを可視化しないと新たな闇を生みそう

告知

- 7/7の「ヒカ☆ラボ」に弊社より3名が登壇します。
- 残席有りますのでご興味ありましたらどうぞ＾＾
  【Scalaノチカラ～CyberZ「F.O.X」におけるTVCM分析機能の開発秘話や失敗談まで～】
  https://atnd.org/events/78370

ご静聴ありがとうございました