The Art of SLOs

原文: https://cre.page.link/art-of-slos-handbook-pdf-a4

ダウンタイム早見表

信頼性レベル	許容される 100% のダウンタイム		
	1年あたり	四半期あたり	28 日あたり
90%	36日 12時間	9日	2日 19時間 12分
95%	18日 6時間	4日 12時間	1日 9時間 36分
99%	3日 15時間 36分	21時間 36分	6時間 43分 12秒
99.5%	1日 19時間 48分	10時間 48分	3時間 21分 36秒
99.9%	8時間 45分 36秒	2時間 9分 36秒	40分 19秒
99.95%	4時間 22分 48秒	1時間 4分 48秒	20分 10秒
99.99%	52分 33.6秒	12分 57.6秒	4分 1.9秒
99.999%	5分 15.4秒	1分 17.8秒	24.2秒

赤で網掛けされた枠内は許容される完全なダウンタイムは1時間未満のもの

28日間で99.95%の信頼性に対して、エラー率を考慮した場合に許容されるダウンタイム			
100%	**10%**	**1%**	**0.1%**
20分 10秒	3時間 21分 36秒	1日 9時間 36分	14日

SLO がどのように...

...良い信頼性のためにビジネスを設計するのに役立つか

ビジネス側の考え方

信頼性が機能であるとすると　　他の機能との比較で　　いつ信頼性を優先させるか？

開発側の考え方

新しい魅力的な機能を作るために必要なシステム変更において信頼性とリスクとのバランスを　どうとるか？

運用側の考え方

サポートするシステムにとっての　適切な信頼性のレベルとはなにか？

SLI の方程式

$$\text{SLI} = \left(\frac{\text{良いイベント}}{\text{有効なイベント}} \right) \times 100\%$$

有効なイベントのなかで良いものの割合

この形式で SLI を表現することの意義

1. **SLI** は **0%** と **100%** の間におさまる
   0%は何も機能していないことを意味し、100%は何も壊れていないことを意味します。この尺度は、直感的で信頼性に関する SLO やエラー バジェットにおいて具体的な割合に変換できます。

2. 一貫したフォーマットの **SLI**
   一貫性があることで、SLI を中心に一般的なツールを構築できます。アラートロジック、エラー バジェットの計算、SLO 分析そしてレポートツールはすべて、同じ入力内容（良好なイベント、有効なイベントと SLO のしきい値）で作成できるようになります。

イベントを分子に含める、もしくは分母から除外することで、イベントがエラーバジェットにカウントされないようにすることができます。前者はイベントを良好と分類することで、後者はイベントを無効と分類することによって達成されます。

一般的には、HTTP(S) を介してリクエストを処理するシステムの場合、有効かどうかはホスト名やリクエストパスなどのリクエストのパラメーターによって決定され、SLI を特定の一連の処理タスクやレスポンスの処理を行うものに絞り込みます。

一般的には、データ処理システムの場合、有効かどうかは入力パラメーターによって決定され、SLI をデータの一部分に絞り込みます。

ユーザー ジャーニーを SLI の仕様に変換

一般に、人がサービスを利用する場合には何らかの目的があるので、そのサービスの指標となる SLI は、そうした目的を捕捉するためのサービスと人とのインタラクションを測定する必要があります。ひとつの目的を達成するための一連のインタラクションをユーザー ジャーニーと呼ぶことにします。これは、ユーザー エクスペリエンスの研究分野から借りてきた言葉です。

SLI の仕様は、レイテンシーや可用性といったサービスの信頼性の特定の尺度に関するユーザーの期待を式で表したものです。 SLI メニューは、与えられたユーザー ジャーニーにおいて測定したい信頼性の尺度のガイドラインになっています。

システムの SLI を決めた後は、測定、有効か否か、およびイベントを良好なものとして分類する方法を決定しながら、SLI を洗練させて実装に落とし込んでいきます。

可用性の SLI の特定

ユーザーからのインタラクティブなリクエストを処理するシステムの可用性は、重要な信頼性の尺度になります。システムがリクエストに正常に応答していない場合は、システムはユーザーの信頼性に対する期待を満たしていないものと見なすことが安全です。

リクエスト/レスポンス

推奨されるリクエスト/レスポンスの可用性の SLI の仕様は次の通りです。

> 有効なリクエストのうち成功した割合

この仕様の実装には、このシステムが提供するリクエストの中でどれが SLI に有効であり、何がレスポンスを成功させるかという 2 つの選択を行う必要があります。

成功の定義は、そのシステムの役割と可用性の測定方法の選択によって大きく異なる傾向があります。成功または失敗の一般的な表現として使用されるものの1つは、HTTP または RPC レスポンスのステータス コードです。これには、システム内でステータス コードを注意深く正確に使用して、各コードが成功または失敗のいずれかに明確に紐付けられるようにすることが必要になります。

ユーザー ジャーニー全体の可用性を検討する場合に、ユーザーがジャーニーを完了する前に自発的に終了する場合を考慮して測定するように注意を払います。

その他の可用性の SLI

可用性は、リクエストの処理の枠を超えた幅広いシナリオに対して役立つ測定の概念です。たとえば、仮想マシンの可用性は、起動後 SSH 経由でアクセスできる時間の割合として定義することもできます。

システムがユーザーの期待どおりに機能しているかどうかを判断するために、複雑なロジックが必要になる場合があります。ここでの合理的な戦略となるのは、その複雑なロジックをコードとして記述し、真偽の二値による可用性の測定値を SLO 監視システムにエクスポートして、上記の例のような「良くない時間」スタイルの SLI で使用する方法です。

レイテンシーの SLI の特定

ユーザーからのインタラクティブなリクエストを処理するシステムのレイテンシーは、重要な信頼性の尺度です。 ユーザーの要求がタイムリーに応答されなければ、ユーザーはそのシステムを「インタラクティブ」と認識しません。

リクエスト/レスポンス

推奨されるリクエスト/レスポンスのレイテンシーの SLI の仕様は次の通りです。

しきい値よりも速く実行された有効なリクエストの割合

この仕様の実装には、システムが対応するリクエストのうちどれを SLI に対して有効として実行するか、そしてリクエストのレイテンシーを測るタイマーがスタートしストップするタイミングという 2 つの選択を行う必要があります。

「十分に速い」とされるしきい値の設定は、測定したレイテンシーがどれだけ正確にユーザー エクスペリエンスに変換されるかにかかっており、SLO のターゲットとより密接に関連しています。速さの感じ方を優先してシステムを設計し、しきい値を比較的ゆるく設定することもできます。リクエストがアプリケーションによってバックグラウンドで処理されてもよくなれば、レスポンスを待つユーザーはいなくなります。

複数のしきい値に異なる SLO を設けることも有用です。単一のしきい値が使用される場合にそのしきい値がターゲットとするのはたいてい、ロングテールのレイテンシーです。一方で、二次的な 75%～90% の範囲のターゲットに対して幅広くレイテンシーの期待値を設定することが有用な場合もあります。その理由は、不満として感じるレイテンシーを図示すると、たいていはしきい値を境界とした二値でなく、S カーブをたどるからです。

その他のレイテンシーの SLI

レイテンシーはデータ処理または非同期処理キューのタスクを追跡することと同様に重要です。日次で実行されるバッチ処理のパイプラインの場合、そのパイプラインが完了するのに 1 日以上かかってはならないでしょう。ユーザーはキューにタスクを入れる際のレイテンシーよりも、キューに入れたタスクの完了時間のほうを気にします。

SLI の指標は、長期実行のオペレーションの最終的な成功時または失敗時ではなく、しきい値を超えた段階で直ちに更新されなければなりません。仮にしきい値が 30 分で、処理のレイテンシーに障害が発生したことが 2 時間

後に報告がされてしまうと、期待を満たしていない測定不能な 90 分の時間ができてしまいます。

品質の SLI の特定

ユーザーに戻すレスポンスの品質をトレードオフとするようなしかけがシステムにある場合 (例えば、CPU やメモリの使用率を下げると品質が下がるなど)、品質の SLI でこうしたサービスの上品な劣化を捕捉する必要があります。劣化についてはユーザーはそれがひどくなるまで気づいていない場合もありますが、このような無意識の事象は、クリックスルー率を低下させるなど、ビジネスへの影響がある場合があります (例えば、品質の劣化によりユーザーに関連する広告表示が減る)。

リクエスト/レスポンス

推奨されるリクエスト/レスポンスの品質 SLI の仕様は次の通りです。

> 品質を劣化させることなく処理された有効なリクエストの割合

この仕様の実装には、このシステムが実行するリクエストのうち SLI に対して有効なリクエストはどれか、そしてレスポンスが品質の劣化を伴って実行されたことをどう決定するか、という 2 つの選択を行う必要があります。

多くの場合、レスポンスの品質の劣化を測定するには、レスポンスが劣化していると印をつけたり、劣化したリクエストのカウンタを加算したりします。したがって、この SLI は「良いイベント」ではなく「良くないイベント」として表現する方が簡単です。

レイテンシーの測定と同様に、品質の劣化がある分布に沿って下回る場合、その分布から1つでなく複数の値を使って、SLO の目標値を定義することが有用な場合があります。やや不自然な例にはなりますが、受け取ったリクエストを10台のバックエンドにばらまくサービスを考えてみましょう。各バックエンドの可用性ターゲットは99.9%で、過負荷時にリクエストを拒否する機能があります。この場合、サービス応答の99%をバックエンド応答の欠落なしで提供し、かつ、99.9%は 2 つ以上の応答の欠落なしで提供する、というように定義することができます。

鮮度の SLI の特定

データをバッチ処理する場合、システムまたはそのユーザーによって新しい入力データが生成されるにつれて、出力の有用性または関連性が時間の経過とともに低下するのが一般的です。 一方で、ユーザーはシステムの出力がそれらの入力に関して最新であると期待しています。こうしたユーザーの期待に答えるためには、データ処理パイプラインは定期的に実行するか、場合によっては再構築して、入力データの小さな増分を継続的に処理する必要があります。鮮度の SLI は、これらの期待に照らしてシステムのパフォーマンスを測定し、エンジニアリング上の意思決定を知らせます。

データ処理

推奨される鮮度の SLI の仕様は次の通りです。

> しきい値よりも新しく更新された有効なデータの割合

この仕様の実装には、システムが処理するデータのうちどれが SLI に対して有効なのか、そしてデータの鮮度を測るタイマーがスタートしストップするタイミングはいつなのかという 2 つの選択を行う必要があります。

バッチ処理システムの場合、最後に正常に処理が完了してからの時間として鮮度の概算ができます。バッチシステムの鮮度をより正確に測定するには、通常は生成および/またはソースの経過時間のタイムスタンプを追跡するために処理システムを拡張する必要があります。インクリメンタル ストリーミング処理システムの鮮度は、完全に処理された最新のレコードの経過時間を捕捉するウォーターマークを使用して測定することもできます。

データの鮮度をレスポンスの品質として測定

システムがそのような選択を積極的に行わなくても、古いデータを配信することは、レスポンスの品質を劣化させる一般的な方法です。古いデータをレスポンスの品質の劣化として測定することは有用な戦略です。古いデータにアクセスするユーザーがいない場合、そのデータの鮮度に関する期待を裏切ることはありません。これを実現するには、配信データの生成を担うシステムのどこかで、配信インフラストラクチャが使用する世代のタイムスタンプを生成し、データを読み取る際に鮮度のしきい値をチェックします。

カバレージの SLI の特定

カバレージの SLI は、システムでデータを処理する場合の可用性 SLI と同様に機能します。データが処理され出力が利用可能になることをユーザーが期待している場合は、カバレージの SLI の使用を検討する必要があります。

データ処理

推奨されるデータ処理のカバレージの SLI の仕様は次の通りです。

> 処理に成功した有効なデータの割合

この仕様の実装には、このシステムが処理するデータのどれが SLI に有効であるか、そして特定のデータの処理が成功したかどうかをどのように判断するか、の 2 つの選択を行う必要があります。

多くの場合、データの処理を実行するシステムは、処理を開始したレコードが正常に処理されたかどうかを判別し、成功と失敗のカウント数を出力する必要があります。 その場合に、処理されるべきレコードが何らかの理由でスキップされた場合にスキップされたレコードの特定に課題が生じます。 これには通常、データソースで `COUNT(*)` に相当するものを実行することにより、データ処理システム自体の外部に存在する有効なレコードの数を決定する何らかの方法が必要になります。

正確性の SLI の特定

いくつかの場合では、処理システムが必要なすべてのデータを処理するだけでなく、処理中に正しい出力を生成することを測定することが重要になることがあります。正確性は、誰もいないときに消極的に検出されるのではなく、優れたソフトウェア エンジニアリングとテストを通じて積極的に確認するものです。ただし、ユーザーがアクセスしているデータが正しく生成されていることを強く期待し、その正確性を独自に検証する方法がある場合には、SLI を使用して継続的に正確性を測定することは非常に有益です。

データ処理

推奨されるデータ処理の正確性の SLI の仕様は次の通りです。

> 正確な出力を生成する有効なデータの割合

この仕様の実装には、このシステムが処理するデータのどれが SLI に有効であるか、そしてアウトプットのレコードの正確性をどう判断するのかという、2 つの選択を行う必要があります。

正確性の SLI を有益なものとするには、正確性の判断方法が出力データの生成に使用される方法から独立していることが求められます。判断方法が独立していない場合には、生成中に存在する正確性のバグが検証中にも存在することになり、SLI による不正確性の検出が妨げられる可能性があります。一般的な戦略としては、処理後に正常な出力を生成することがわかっている「ぴかぴかの」入力データを用意することです。 この入力データが実際のユーザー データを十分に代表していて、処理システムのコード パスのほとんどを実行するように設計されている場合には、全体的な正確性を推定するのに十分といえます。

スループットの SLI の特定

データ処理システムは、ユーザーから見えるデータ処理のレイテンシーを小さくするために、ストリームや小さなデータのかたまりが連続的に処理されるように設計されている場合があります。通常はレイテンシーの SLI はこれを定量化するための最良の方法ですが、ユーザーに特定のレベルのスループットを提供することを約束した場合や、データ処理のレイテンシーに期待されることが一定でない (例えばイベントごとに処理されるデータの量が劇的に変化するような) 場合に、システムの全体的なスループットによる測定の方がより適切な測定方法になることがあります。

データ処理

推奨されるデータ処理のスループットの SLI の仕様は次の通りです。

> データ処理レートがしきい値よりも速い時間の割合

スループットは時間の経過に伴うイベントの割合であるため、スループットの SLI の仕様の構造は、レイテンシーの SLI とは若干異なります。これを SLI の方程式に当てはめるには、イベントを秒や分などの時間の単位の経過として定義し、「良い」イベントは、処理速度が十分に速い時間の単位として定義します。この仕様を実装するには、データ処理速度の測定単位を 1 つ選ぶ必要があります。

データの処理にかかる作業量は多くの場合そのサイズに比例し、処理されるデータの量をバイト単位で測定できるため、スループットの一般的な測定値は「バイト/秒」です。処理のコストに関してほぼ直線的にスケーリングするようなあらゆる指標には、これが機能するはずです。データを処理するシステムは、データを処理する速度を定量化する指標を記録することが求められます。

レイテンシーや品質と同様に、スループット率は多様で複数のしきい値が適切な場合もあります。システムが QoS レベルで設計されていて、優先度の低いものがキューに入れられている間も重要なデータが迅速に通過するように設計されている場合は、全体的なスループットを数時間または数日間遅らせて処理することを許容できる場合があります。

スループットの SLI は、レイテンシーが大きく変動するリクエスト駆動のサービスに対しては、レイテンシーの SLI よりも役立ちます。多くの場合この相違は、アップロード時のサムネイルと 4K の高解像度画像との違いなど、処理コストが変動するリクエストによって決まります。

SLO と SLI の作成

それぞれのクリティカル ユーザー ジャーニー をビジネス影響に応じて優先

順位付けを行う

1. メニューから **SLI** の仕様を選択する
2. 仕様を詳細な **SLI** の実装に精緻化する
3. ユーザー ジャーニーを検証してカバレージの相違点を探す
4. 過去のパフォーマンスまたはビジネスのニーズに基づいて**SLO** を設定する

SLO ワークシートの例

SLI にイベントと成功基準 があることを確認し、成功または失敗を記録する場所と方法を指定します。仕様を、良いイベントの割合として書き出します。*SLO* が 目標と測定期間 の両方を定義していることを確認してください。

ユーザージャーニー: ホームページの読み込み

SLI のタイプ: レイテンシー

SLI の仕様:

100ミリ秒未満で処理されたホームページへのリクエストの割合

（上記の「*100*ミリ秒未満で処理されたホームページのリクエスト」は *SLI* の式の分子であり、「ホームページのリクエスト」は分母です。）

SLI の実装:

- サーバー ログの「レイテンシー」列から測定した、**100ミリ秒未満**で処理されたホームページへのリクエストの割合。
  （メリット/デメリット: この測定では、バックエンドに到達できないリクエストを見逃します。）
- 仮想マシン上のブラウザーで JavaScript を実行するような外部監視測定器によって測定された、**100ミリ秒未満**で処理されたホームページへのリクエストの割合。
  （メリット/デメリット: これは、リクエストが我々のネットワークに到達しない場合のエラーを捉えますが、一部のユーザーのみに影響する問題を見逃す可能性があります。）

SLO:

過去 28 日間において 99% のホームページへのリクエストが 100 ミリ秒未満で処理される。

SLI の測定

SLIを測定する方法はおおまかに5つあり、それぞれにメリットとデメリットがあります。　多くのエンジニアリング上の決定と同様に、すべての状況に対して正しい選択は 1 つではありませんが、選択に伴うトレードオフを十分に理解すれば、システムの要件を満たす SLI の実装を選択することができます。

これらの測定方法の分類は、ユーザーからの距離が遠い順で示しています。一般に SLI は可能な限りユーザーの体験に近い形で測定する必要があるため、ユーザーとの距離やシステムとのやりとりは、検討すべき価値のある特性です。

ログの処理

SLI　の指標を生成するために、リクエストまたはデータのサーバー側のログを処理します。

メリット	デメリット
+ 既存のログを再処理して、過去に遡って SLI データを埋め合わせられます。 + 複雑なユーザー ジャーニーは、セッション ID を使用して再構築できます。 + SLI の実装を導出するための複雑なロジックは、コードに変換して、はるかに単純な「良いイベント」と「イベント全件」の 2 つのカウンターとしてエクスポートできます。	– サーバーに届かなかったリクエストはアプリケーションログに含まれません。 – 処理に時間がかかるため、ログに基づいた SLI は障害対応の通知を発砲するのには適しません。 – ログから SLI を生成するためには、エンジニアリングの工数が必要です。セッションの再構築には時間がかかる場合があります。

アプリケーション サーバーの指標

ユーザーからのリクエストまたはユーザーのデータを処理するコードから SLI 指標をエクスポートします。

メリット	デメリット

メリット	デメリット
+ 通常、コードへ新しい指標を追加することはすばやく、低コストで可能です（エンジニアの工数の観点から）。 + SLI の実装を導出するための複雑なロジックは、コードに変換して、はるかに単純な「良いイベント」と「イベント全件」の 2 つのカウンターとしてエクスポートできます。	– サーバーに届かなかったリクエストはアプリケーションログに含まれません。 – アプリケーション サーバーがステートレスの場合、複数のリクエストを含むユーザー ジャーニーの捕捉は困難です。

フロントエンド インフラストラクチャの指標

負荷分散インフラストラクチャ（Google Cloud のグローバル レイヤ 7 ロードバランサなど）の指標を使います。

メリット	デメリット
+ 通常、指標と過去のデータがすでに存在するため、おそらくこの方法が最も少ない工数で測定を始められます。 + SLI の測定は、お客様の最も近くでありながらも、サービスを提供するインフラストラクチャ内で行われます。	– データ処理の SLI や複雑な要件を持つ SLI には現実的な方法にはなりません。 – 複数のリクエストを含むユーザー ジャーニーでは、概算しか測定できません。

合成クライアント（外部監視）又はデータ

定期的に人工的なリクエストを送信するクライアントを立ち上げ、レスポンスを検証します。データ処理パイプラインの場合は、処理後に正常な出力を生成することがわかっている入力データを作成し、出力を検証します。

メリット	デメリット
+ 合成クライアントは、複数のリクエストを含むユーザー ジャーニーにおけるすべてのステップを測定できます。 + インフラストラクチャの外部からリクエストを送信することで、SLI に定義されたリクエストの流れの全体を捕捉できます。	– 合成リクエストによってユーザー体験の近似を行うため、実際のユーザー体験と異なります。 – すべての特殊なケースをカバーするのは難しいため、統合テストに発展しかねません。 – 高い信頼性目標の場合には、正確な測定のために頻繁に監視リクエストを送信する必要があります。 – 外部監視のトラフィックによって、本物のトラフィックが埋もれてしまう可能性があります。

クライアントへの埋め込み

ユーザーが操作するクライアントに監視機能を追加し、SLI を捕捉するシステムにイベントを記録します。

メリット	デメリット
+ ユーザー体験を最も正確に測定できます。 + サードパーティの信頼性も数値化できます (例えば、CDN や決済ゲートウェイなど)。	– クライアント ログの取り込みと処理に時間がかかるため、障害対応の通知を発砲するのには適しません。 – SLI の測定値には、直接制御できないような変動が大きい要素が多数含まれます。 – 監視機能をクライアントに埋め込むには、エンジニアリングに多大な工数がかかります。

Stoker Labs Inc.

これはフィクションの作品です。名称、ビジネス、イベント、ゲームの仕組みは、作者の想像力や架空の産物です。実在のゲームとの類似点があったとしても、それが現時点で存在するか否かにかかわらず、完全に偶然の事象です。

ミッションステートメント

当社の使命は「紛争をゲームに置き換える」ことです。人々が人生を深刻に受け止めすぎると、社会全体で見られる分断は有害になります。当社は、モバイルビデオゲームという媒体を介して、人間性の中核をなす競争の衝動に対応する手段を提供することを目指しています。心理的、肉体的に有害ではない楽しい方法で、こうした競争の衝動を昇華させる手段を人々に提供することで、世界はより協力的になってうまくいくようになると固く信じています。

ゲーム: Fang Faction

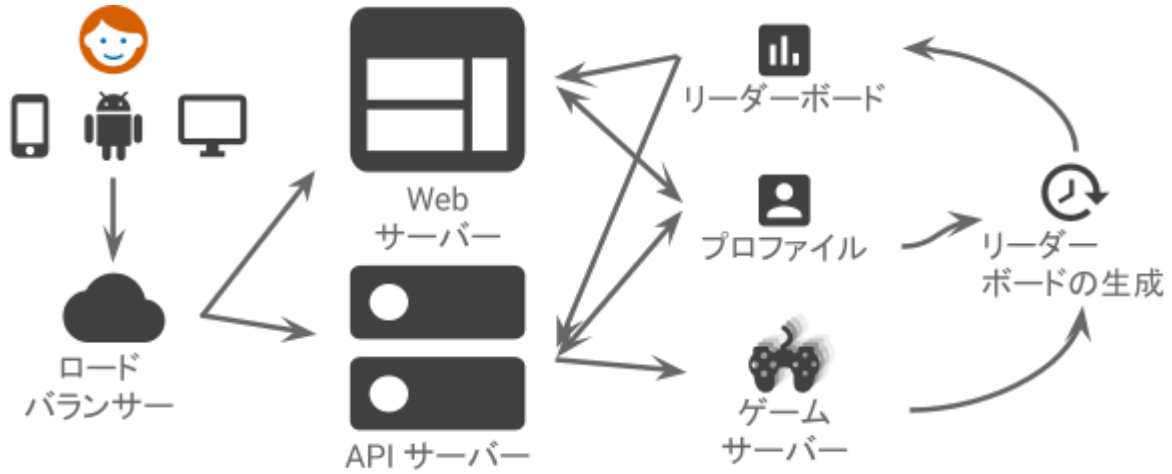
吸血鬼の台頭は人類に壊滅的な打撃を与え、生き残った人々は以前の文明の中心地から遠く離れた、居住可能な残された地域に集まって暮らすことを余儀なくされました。 生存している一派のリーダーとして、あなたは自身の目的達成のために人を集め、陣地を確保して向上させ、吸血鬼が占領している都市を救い、そして資源のコントロールを取り戻すために他の派閥とも戦わなければなりません。

ゲームの世界ではさまざまな報酬や課題があるエリアに分かれています。より良い報酬が得られるエリアへは、それまでのプレイ時間、陣地の大きさ、およびゲーム内の通貨を使うことでアクセスできるようになります。各エリアには、上位の派閥がランキングされたリーダーボードがあります。

このゲームには月間約　5,000　万人のアクティブユーザーがいて、常に100万人から 1,000 万人のプレイヤーがオンラインでプレイをしています。月に 1 回、新しいエリアが追加され、トラフィックと収益の両方が急増します。

現実世界の通貨とアプリで購入したゲーム内通貨との交換が主な収益源となっていますが、他のプレイヤーとの戦いに勝利したり、ミニゲームでのプレイ、ゲーム内のリソースの生産をコントロールすることで、プレイヤーは通貨を獲得できます。プレイヤーは、陣地のアップグレード、戦闘時の守りの強化、高度なスキルを持つ人を派閥に入れるためのミニゲームをプレイする際に、ゲーム内通貨を使うこともできます。

サービスのアーキテクチャ

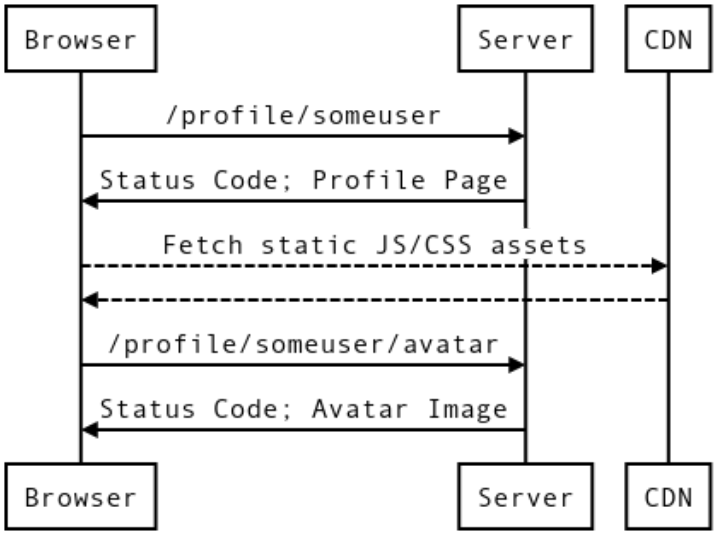


ゲームにはモバイル クライアントとWeb UI の両方があります。モバイル クライアントは、提供しているインフラストラクチャと RESTful HTTP 上でやりとりされる JSON RPC メッセージを使って、リクエストを送信します。また、ゲームの状態のアップデートを受信するための WebSocket 接続を維持します。ブラウザは HTTPS 経由でWebサーバーと通信します。リーダーボードは 5 分ごとに更新されます。

ユーザー ジャーニー

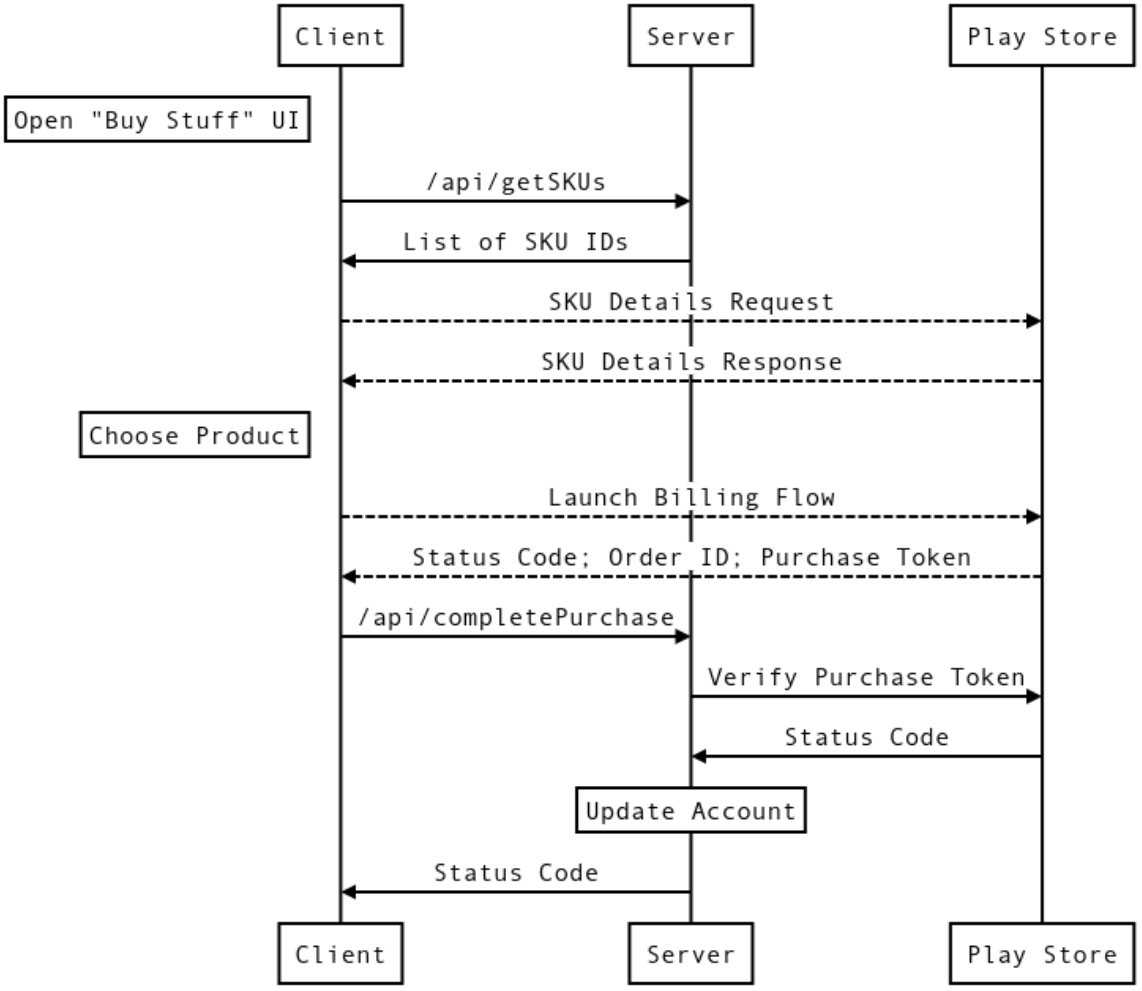
プロファイル ページを見る

プレーヤーは、Web ブラウザからゲーム アカウントにログインし、陣地を表示したり、プロファイルを変更したりできます。プレイヤーのプロファイル ページの読み込みは単純なジャーニーですのでワークショップで一緒に確認を行うことになっています。



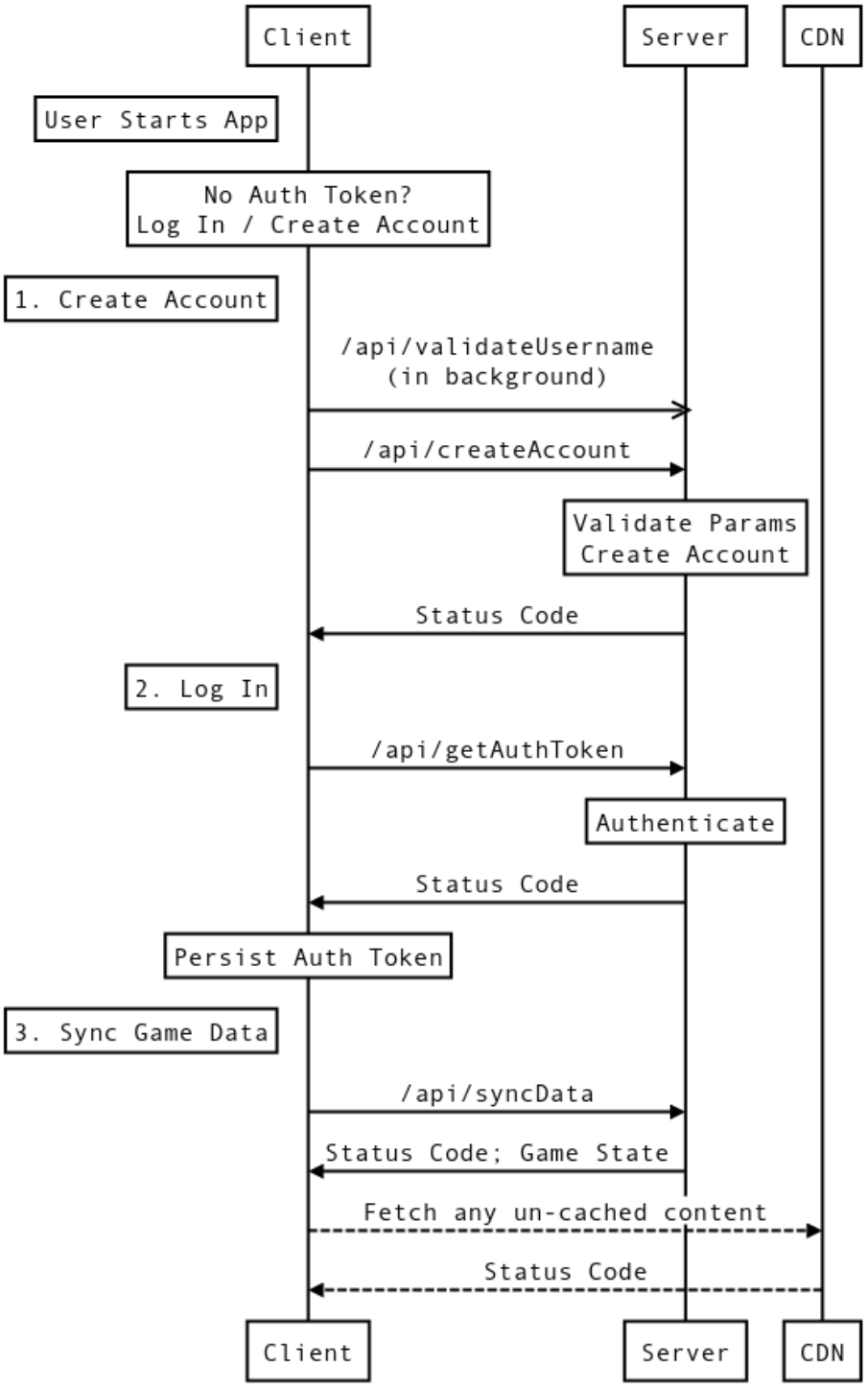
ゲーム内通貨の購入

我々にとって最も重要なユーザージャーニーは、収益のすべてを生み出してくれるものです。それはアプリ内でユーザーがゲーム内通貨を購入するジャーニーです。Play ストアへのリクエストは、クライアントからしか見えません。毎秒 0.1 回から 1 回の購入完了が見られます。新しいエリアがリリースされた直後はプレイヤーが集中するので、毎秒 10 回の購入に急増します。


Client
Server
Play Store
Open "Buy Stuff" UI
/api/getSKUs
List of SKU IDs
SKU Details Request
SKU Details Response
Choose Product
Launch Billing Flow
Status Code; Order ID; Purchase Token
/api/completePurchase
Verify Purchase Token
Status Code
Update Account
Status Code
Client
Server
Play Store

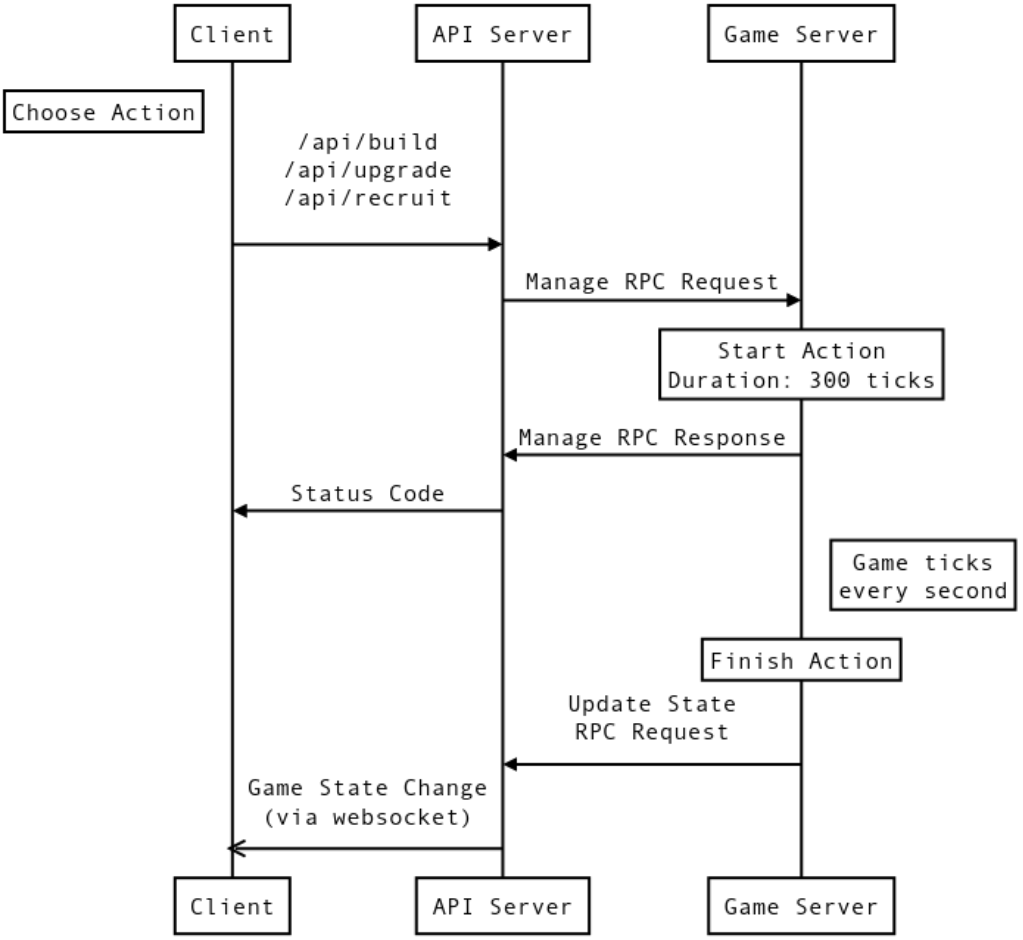
アプリの起動

アプリの起動プロセスには、ユーザーが既にアカウントを持っているか、そのアカウントが現在のデバイスで最近アクセスしたかに応じて、3 つの部分があります。アカウントの作成と認証トークンへのリクエスト率は低いですが、20 QPS から 100 QPS の syncData へのリクエストが見られ、新しいエリアがリリースされた後には 1000 に急増します。



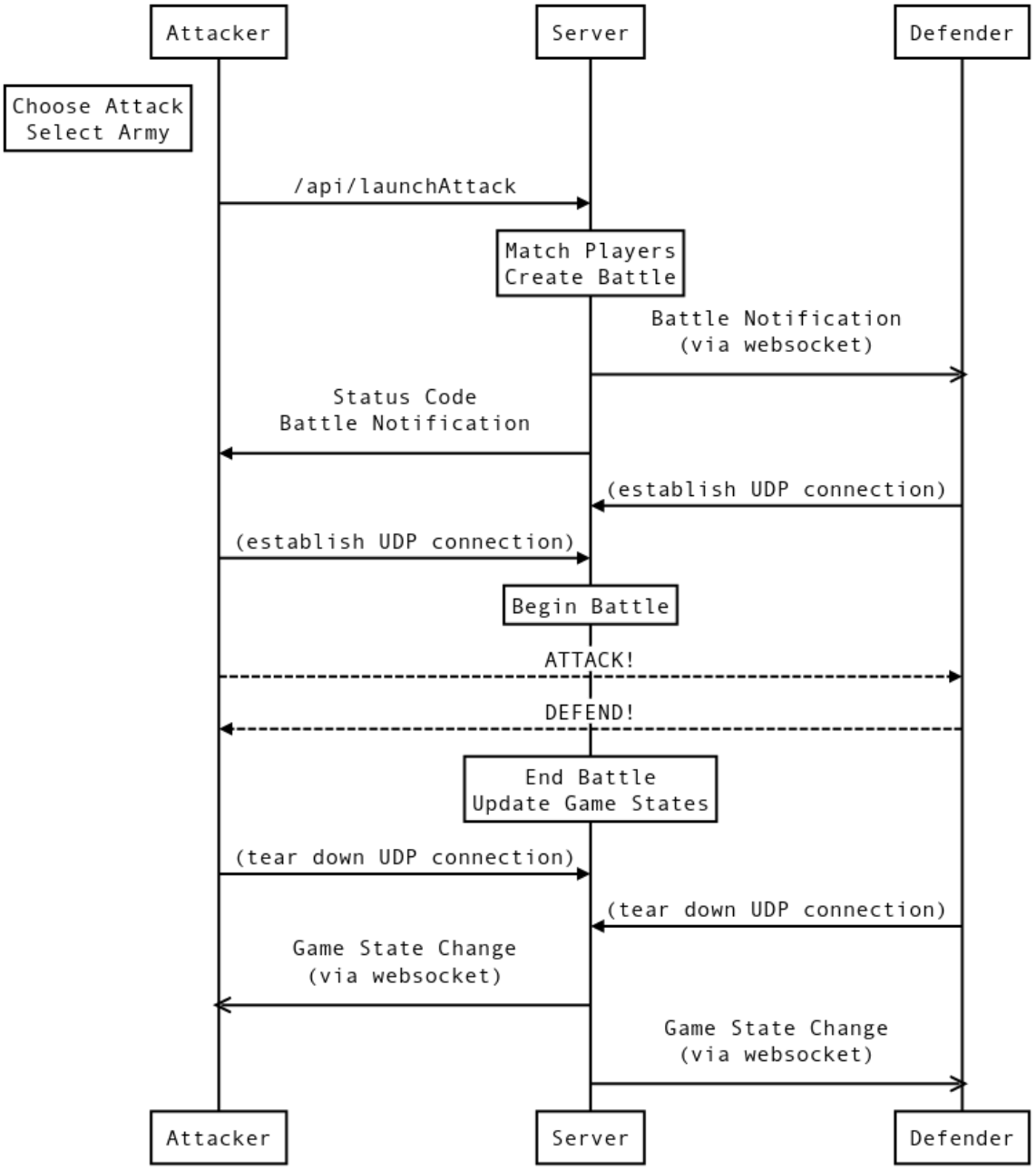
陣地の管理

陣地の管理は比較的単純な API のリクエストです。陣地のアップグレード、守りの強化、派閥メンバーの募集には、時間とゲーム内通貨を消費します。プレーヤーは陣地管理に多くの時間を費やしていて、API のエンドポイント全体で 1 秒あたり 2000 ～ 3000 のリクエストが発生し、最大で 10,000/秒に急増します。ゲームサーバーは、エリア内のすべてのプレーヤーの状態を 1 秒に 1 回更新するように「カチカチ」刻みます。新しいゲームの状態を計算するのに常に 1 秒以上かかる場合、ユーザーは建物が時間どおりに完成していないことに気付くでしょう。



別のプレイヤーとの対戦

別のプレイヤーに攻撃を仕掛けると、リアルタイムの「タワーディフェンス」バトルが発生し、攻撃側の軍隊が防御側の配備を制圧しようとします。防御側は守りを強化するために部隊を配備できます。相手に与えたダメージの量に比例して両者ともポイントを獲得します。毎秒約 100 回の攻撃が仕掛けられます。

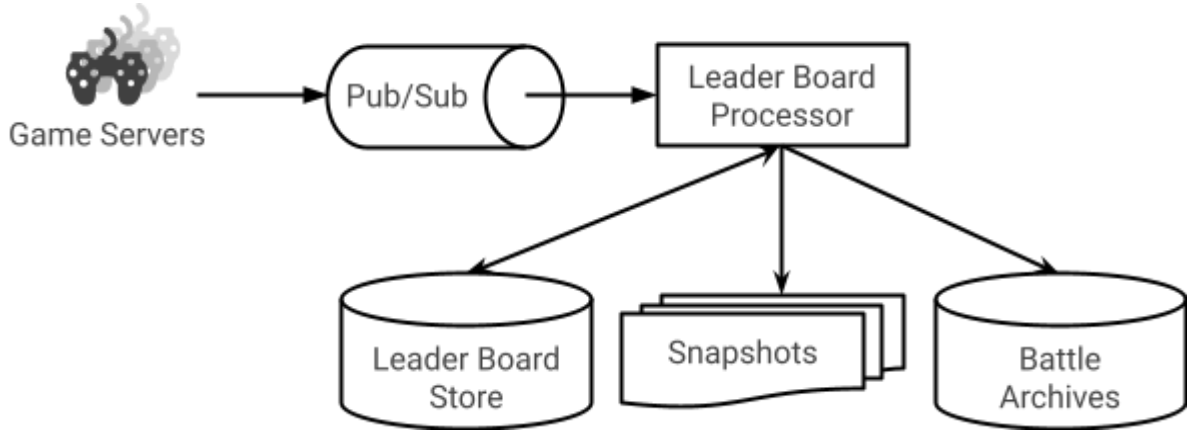


リーダーボードの生成

特定のエリアでは、プレーヤーは主に対戦を仕掛けあっていてスキルレベルも同等なので、ランキングの 1 位をめぐる競争は熾烈です。バトルはゲームサーバー全体のなかで負荷が一番小さいものが優先してスケジュールされます。

対戦スコアは、バトルをホストしたゲームサーバーが、バトルの終了時に PubSub フィードに書き込みます。このフィードは、リーダーボードのデータストアの横で起動している処理部によって読み取られ、この処理部は他にも多くの責任も担っています。各ゲームエリアのスコアテーブルを更新し、過去 1 時間と前日分の各ゲームの上位の攻撃スコアと防御スコアのグローバルテーブルを管理します。

リーダーボード データベースに対してクエリを実行することで、5 分ごとにすべてのテーブルの上位 50 のスナップショットが生成され、その結果をインメモリの Key-Value ストアに書き込みます。以前のスナップショットは 30 分間保持され、その後 ガベージコレクションされます。最後に、すべてのゲームの完了とそのスコアは、1時間ごとに追記専用のファイルに記録され、クラウド ストレージ サービスにアーカイブされます。



ポストモーテム: 空のプロファイル ページ

影響

08:43 から 13:17 の間、プロファイルページにアクセスしたユーザーは不完全な応答を受け取りました。これにより、ユーザーはプロファイルを表示または編集できなくなりました。

根本原因とトリガー

根本原因は、Webサーバーが Unicode の HTML テンプレートを処理する部分におけるバグでした。トリガーは `commit a6d78d13` で、ローカリゼーションをサポートするためにプロファイル ページのテンプレートを変更しましたが、テンプレートの HTML に誤って Unicode 引用符 (U+201C “ と U+201D ”) を挿入してしまいました。Web サーバーが標準の ASCII 引用符 (U+0022 ") の代わりにこれらを検出した際、テンプレート エンジンは出力のレンダリングを中止してしまいました。

検出

中止されたレンダリング処理は例外をスローしなかったため、不完全な応答の HTTP ステータスコードは `200 OK` でした。そのため、この問題は SLO 監視では検出されませんでした。サポートチームとソーシャルメディアチームは、12:14 にプロファイルページに関連する苦情が大幅に増加したことについての懸念を手動でエスカレーションしました。

学んだ教訓

うまくいったこと

- サポートチームとソーシャルメディアチームは、正しいエスカレーション先を発見し、運用チームに正常に連絡できた。

うまくいかなかったこと

- HTTP ステータスコード SLI は不完全な応答を検出しなかった。
- Web サーバーは、 Unicode のサポートが不十分な非常に古いバージョンのテンプレート エンジンを使用していた。

幸運だったこと

- プロファイル ページは、当社の収益には重要ではなかった。

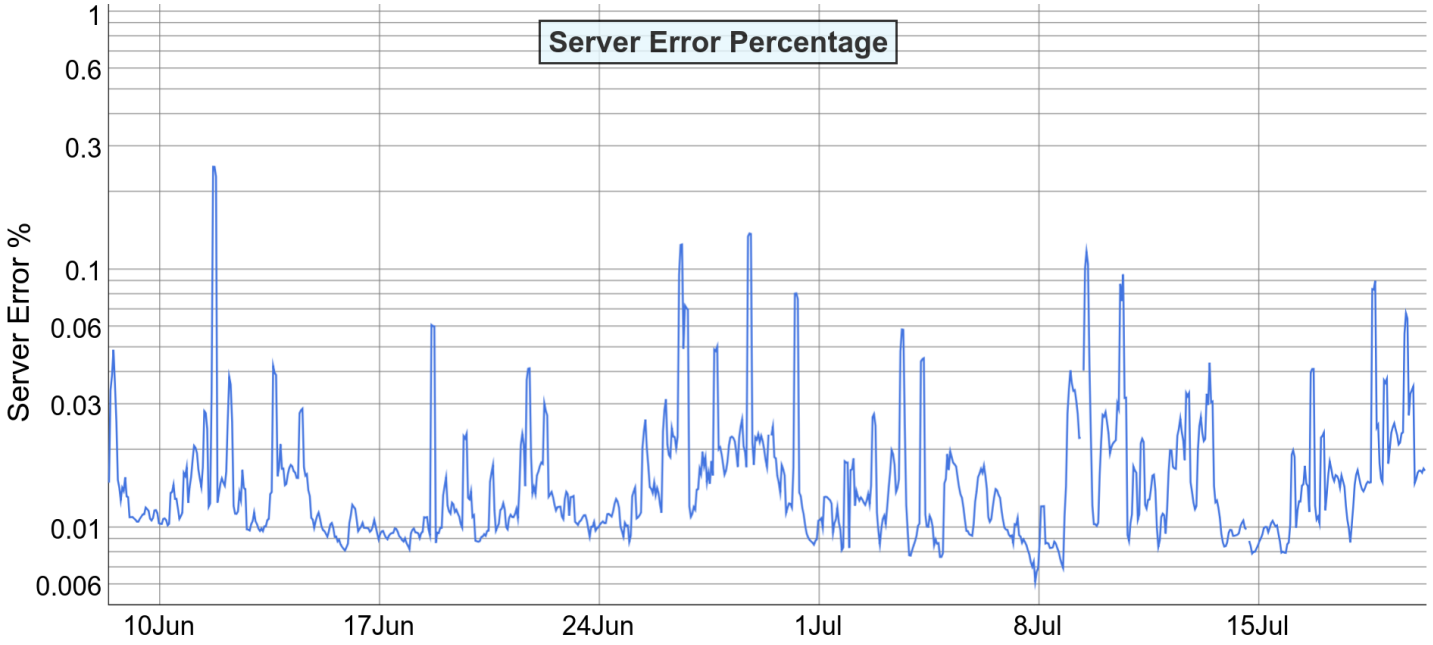
アクションアイテム

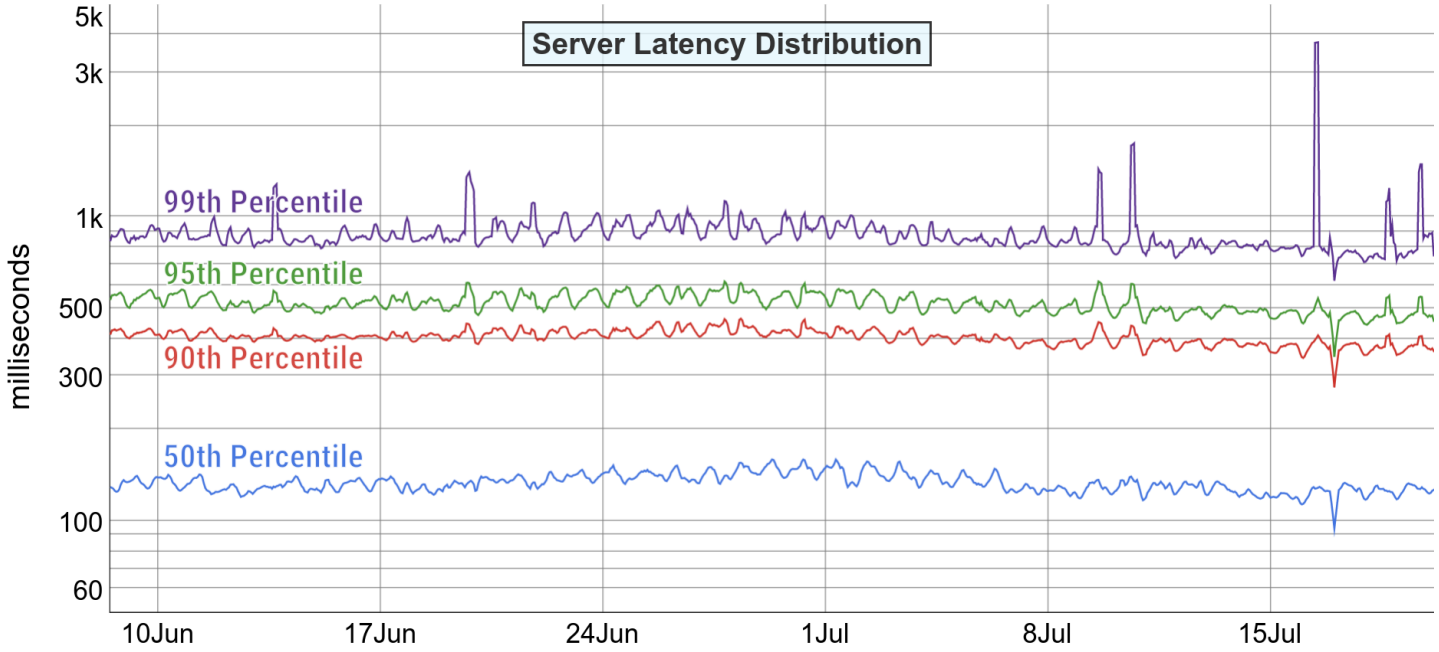
今後決定

プロファイル ページのエラーとレイテンシー

注：グラフの Y 軸は両方とも対数です。

ここでは対数軸を使用して、グラフの詳細を簡単に認識できるようにしています。 それにより、エラーやレイテンシーの大きなスパイクが、より小さくよりー貫性のあるバックグラウンドの変動をよく見えなくしてしまうことを防ぎます。





リソース

このワークショップに関連するすべてのリソースへのリンクは、https://cre.page.link/art-of-slos でご確認いただけます。すべてのコンテンツはクリエイティブ コモンズ規定のライセンス CC-BY-4.0 で一般公開されています。

コンテンツにエラーがあった場合は次のリンクからバグを報告してください。

https://cre.page.link/art-of-slos-bug

コンテンツの改善や追加のご提案がある場合にも次のリンクから是非ご意見をお聞かせください。

https://cre.page.link/art-of-slos-improve

または、ワークショップについてサポートが必要な場合やご質問がある場合には次のリンクからご連絡ください。

https://cre.page.link/art-of-slos-help

この題材についてより深く知りたい場合は、信頼性の測定と管理の Coursera コースが次のリンクで用意されています。

https://cre.page.link/coursera

本日はご参加いただき、ありがとうございました。