Engineer

サービスレベル目標

◎ サービスが満たすべき、測定可能な信頼性の目標

どのようなものか

サービスレベル目標は、サービスがどの程度適切に動くべきかを、達成したか誰でも確かめられるほど正確に記した目標です。例えば、明示した期間内に成功すべきリクエスト、または十分速く応答すべきリクエストの割合です。ほとんどの場合、SLOと略します。Googleの『Site Reliability Engineering』は、指標で測るあるサービスレベルについて、目標値または値の範囲を定めるものと定義しています。

その指標がサービスレベル指標、略してSLIです。利用者が実際に受けるサービスの1側面を表すよう、慎重に定義した量を指します。最初によく選ぶのはリクエストのレイテンシーで、エラー率やスループットもよく使われます。『SRE Workbook』は、どのSLIも良好なイベント数を全イベント数で割る比率として書くことを勧めています。こうすると、何も動かない0%から、何も壊れていない100%までの値になります。目標はその目盛りに引く線です。指標は、目標値に対して望ましい側、または下限と上限の間にとどまる必要があります。

完全な目標には、確認に必要なものをすべて記します。同書の例は、すべてのバックエンドサーバーを対象に測定し、1分間の平均でGet RPC呼び出しの99%が100 ms未満で完了するというものです。1つのパーセンタイルだけでは分布全体を表せないことが多いため、1つの指標に複数の目標を置くこともできます。例えば、呼び出しの90%は1 ms未満、99%は10 ms未満、99.9%は100 ms未満とします。Workbookはさらに、利用者にとって大切な結果を表す仕様と、測定場所を表す実装を分けています。測定場所は、サーバーログ、ブラウザーを動かすプローバー、ページ内のコードなどです。それぞれ検出できる障害が違うため、どれを選ぶかも定義に含めます。

SLOとSLAは異なります。サービスレベル契約は、利用者との明示的または暗黙の契約であり、その中の目標を達成したかどうかに応じた結果を定めます。分かりやすい例は返金や違約金です。同書は、目標を達成できなかったら何が起きる

かと問う方法を示しています。明示的な結果が何もなければ、それはSLOです。脚注には、SLA違反と言う人のほとんどが、実際にはSLO未達を意味しているともあります。本当のSLA違反は、契約違反として裁判に至る可能性があるからです。同書によると、Google Searchには一般利用者向けのSLAがありませんが、停止すれば評判や広告収入を失います。

概念を判断に生かす

「サービスレベル目標とは？」を実際の判断に生かしましょう。その概念を、開発または購入するシステムに結び付けて考えるお手伝いをします。 障害が小さいうちに知らせる監視とアラートを整え、安定した状態をどれだけ維持できているかを数値で示します。

サービスを見る お問い合わせ →

用語の背後にある前提を検証する

解説とリンク先の実績を参考に、プロジェクトについての質問を整理してください。ツールを選ぶ前に、前提を一緒に検討できます。 社内初のオブザーバビリティダッシュボードを構築 — 社内システムの状態を初めて可視化

社内初のオブザーバビリティダッシュボードを構築 お問い合わせ →

なぜ重要なのか

目標を明示しなければ、利用者が自分で期待を作ります。SREの書籍は、そこから生じる2つの問題を説明しています。サービスを過信する人と、実際より不安定だと思って利用を避ける見込み利用者です。例は、分散システム向けのGoogleのロックサービス、Chubbyです。全体の停止があまりにまれだったため、他のサービスがいつの間にか「決して止まらない」ことに依存し、まれに停止すると壊れるようになりました。対策は、グローバルインスタンスがSLOを達成しつつ、大きく上回らないようにすることでした。実際の障害で可用性が目標を下回らなかった四半期には、SREが意図的にシステムを停止しました。

目標は、信頼性がどこまであれば十分かという議論にも答えます。『SRE Workbook』は「信頼性100%は誤った目標です」と記し、実務的な理由を挙げています。構成要素は、どのように作っても同時に故障します。サービスと顧客の間をつなぐネットワークや機器も故障するため、顧客には100%に見えません。9を1桁増やすごとに前より高くつく一方、利用者にとっての価値はゼロに近付きます。また、停止の最大の原因は変更なので、停止が一切許されないサービスは更新もできません。目標を100%未満にすると、完全な状態との差を、リリース、実験、計画保守に使えるようになります。これがエラーバジェットで、単純に100%からSLOを引いた値です。機能を出すことと稼働を保つことの兼ね合いには責任者が必要です。Workbookは、小さな組織ではCTO、大きな組織ではプロダクトオーナーまたはプロダクトマネージャーを挙げています。

最後に、目標は監視を判断につなげます。同書は、指標を測り、目標と比較し、対応が必要かを決め、実行するという循環を説明しています。レイテンシーが上がり、数時間以内に境界を超えそうなら、サーバーがCPUによって制約されているかを調べ、台数を増やします。目標がなければ、対応すべきか、いつすべきかを誰も判断できないと同書は指摘しています。

実際の活用例

購入手続きAPIを例に、Workbookが勧める形式で目標を書いてみます。指標と測定場所を分け、対象期間を明示し、その数値にした理由を添えます。

```
サービス:       購入手続きAPI
SLI（指標）:  POST /checkoutへの要求で5xx以外のステータスを返した割合
              サーバーのリクエストログで集計
目標:           99.9%、常に直近28日間を対象にする
根拠:           4週間の観測データから切り下げて設定
責任者:         プロダクトマネージャー、オンコールのエンジニアと協働
見直し:         安定するまでは毎月、その後は四半期ごと
```

期間は割合と同じくらい重要です。Workbookは、暦月よりも常に直近の一定期間を対象にする方法を勧めています。翌月になったからといって、利用者が前月末の停止を忘れるわけではないためです。また、どの期間にも同じ数の週末が入るよう、週の整数倍にすることを勧め、汎用的な選択として4週間を挙げています。この期間は28 × 24 × 60 = 40,320分で、その0.1%は完全停止の40分強に当たります。30日の月なら、同じ99.9%で許容されるのは43.2分で、同書の可用性表にもこの値が載っています。

ただし、時間による集計は粗い方法です。SREの書籍は、世界中に分散するサービスが完全に稼働しているか、完全に停止しているかのどちらかになることは少ないため、Googleは稼働時間ではなくリクエストの成功率で可用性を測ると説明しています。リクエスト数で数えるWorkbookの例では、99.9%なら4週間の3,000,000リクエストに対して3,000件の失敗が許され、1,500件を失敗させる停止でバジェットの半分を使います。リクエストを数えれば、稼働時間では見落とす部分的な障害も評価できます。トラフィックが一定なら、全リクエストの半分が失敗する20分間は、完全停止の10分間と同じだけ消費します。これは28日間のバジェットの約4分の1です。

Prometheusでステータスのラベルを持つリクエストカウンターを使う場合、対象期間の指標は1つの比率で表せます。次の式はWorkbookの例を基にしています。

```
sum(rate(http_requests_total{job="checkout", status!~"5.."}[28d]))
  /
sum(rate(http_requests_total{job="checkout"}[28d]))
```

当サイトのサービスレベルページは、目標と契約のうち、契約の側を扱います。会社が顧客に代わって運用するシステムでは、インシデントへの応答時間、対応時間帯、求める可用性、保守時間帯を、そのシステムの契約に記します。夜間レポートと決済サービスでは必要な条件が違うため、システムごとに定めます。創業者が現在NextMarinerでの業務の一環として構築した自社管理のエラー監視とトレーシングは、一連の運用者向け画面を通じて利用でき、その1つがサービスレベル目標の画面です。また、最後の1枚を確実に1回だけ販売するノートは、在庫の不変条件を目標へ変えます。定期的に検査し、グラフにし、ずれがあれば担当者を呼び出すことで、売り越しゼロを単なる願いではなく、アラートに裏付けられた目標にします。

問題が起きるところ

先月のダッシュボードから目標をコピーします。 システムの現在の実績をそのまま使うのは安全に思えます。しかしSREの書籍は、考えずに採用した目標によって、再設計なしには改善できないシステムで、チームが達成のための無理な奮闘に縛られる可能性があると警告しています。利用者が必要とするものから逆算し、数値の横に理由を書きます。Workbookは、数値が率直に言えばその場で決めたものでも、その根拠と、実験に由来するのか観測に由来するのかを求めています。最初は毎月、落ち着いたら四半期ごとに文書を見直します。

目標が100%か、費用を確かめていない9の並びです。 常に利用できる、あるいは無制限に拡張できるシステムは、構築にも運用にも費用がかかり、利用者が違いを感じ取れる水準を超えています。緩い目標で始め、システムを理解するにつれて厳しくします。同書は、厳しい目標で始めてから達成不能だと分かって緩めるより、この方法を勧めています。

平均を利用者の体験の代わりにします。 同書の例では、通常のリクエストは約50 msですが、5%のリクエストはその20倍かかり、平均のグラフは一日中何も起きていないように見えます。レイテンシーの目標は、平均ではなく、パーセンタイルや、300 msなどのしきい値未満で終わるリクエストの割合で記します。

誰も必要性を説明しきれないほど目標があります。 サービスごとに12個のSLOを設けると、どれも効力が薄れます。同書の判断基準は率直です。ある目標を持ち出しても優先順位の議論を一度も決着させられないなら、おそらく設ける価値は

ありません。利用者が実際に行うことに対応する少数の目標だけを残します。

目標を達成できなくても何も変わりません。 Workbookは、これを実効性のないSLOと呼びます。達成状況は何かを決める理由ではなく、報告書に載せるもう1つのKPIになります。バジェットを使い切ったら何をするかを事前に合意し、プロダクトマネージャー、開発者、SREに署名してもらいます。Workbookのエラーバジェット方針の例では、サービスが目標を外れている間は通常の変更を止め、最優先のバグ修正とセキュリティ作業だけを認めています。また、その方針は罰ではないと明言しています。

SLOがSLAと同じ数値です。 内部目標が契約と同じだと、チームが問題を初めて知るのが契約違反になった時点になります。同書は、慢性的な問題に対する余裕を作るため、公表した値より厳しい内部目標を勧めています。逆の間違いにも注意が必要です。サービスが目標をはるかに上回ると、利用者はその水準を当てにするようになります。そのため同書は、時折の計画停止、処理量の制限、軽い負荷でもそれ以上速くならないシステムを提案しています。

対象期間よりデータの保存期間が短くなっています。 Prometheusは、保存期間を指定しない場合、既定で15日分のサンプルを保持します。そのサーバーで28日間の目標を計算すると、気付かないうちに15日分を測り、4週間分として報告します。数値を信用する前に、保存期間を対象期間より長く設定します。

```
prometheus --storage.tsdb.retention.time=35d
```

参考資料

- サービスレベル目標 — Googleの『Site Reliability Engineering』第4章です。指標、目標、契約と、目標値の選び方を説明しています。
- SLOの実装 — 『SRE Workbook』の章です。比率で表すSLI、エラーバジェット、対象期間、目標の文書化を説明しています。
- エラーバジェット方針の例 — 架空のゲームサービス向けの完全な方針で、リリースを止める条件と、意見が割れたときの決定者を示しています。
- リスクを受け入れる — Googleが稼働時間ではなくリクエストの成功率で可用性を測る理由と、信頼性を1段階上げるごとの費用を説明しています。
- 可用性の表 — 可用性の各水準で許される停止時間を、年、四半期、月、週、日、時間ごとに示しています。
- ストレージ — ディスク上のサンプルの保持方法と、SLOクエリがどこまで過去を見られるかを決める保存期間フラグについてのPrometheusの説明です。

用語集

実際の制約に合わせて選ぶ

デプロイの流れ、運用上のリスク、緊急対応の担当者をお知らせください。検査、復旧手段、明確な責任の所在を伴う変更の範囲を定められます。

料金を見る お問い合わせ →

用語を実務に結び付ける

「サービスレベル目標とは？」がお客様の組織にどう当てはまるかをご相談ください。現在のシステムの具体例が一つあれば、話を始められます。

サービスを見る

welcome@engineer.company +45 50 10 61 84 デンマーク、コペンハーゲン

このページを持ち歩く

連絡先カード

心を込めて丁寧に制作： D.

CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N

著作権 © 2025 – 現在 · Engineer ApS · すべての権利を留保します。

印刷元： https://engineer.company/ja/glossary/service-level-objective/