Engineer

メトリクス

システムが自身について報告する数値

どのようなものか

メトリクスは、稼働中のサービスを測った値です。それぞれ、ある瞬間に採り、時刻と少数のラベルを添えて保存し、一分、一日、一か月後に、同じ量を再び読めるようにします。名前を付けた一つの量がメトリックで、時間に沿う値の列として保存すると、時系列になります。OpenTelemetryは、トレース、ログと並ぶ信号に数えます。処理したリクエスト、使用中のメモリー、待ち行列の仕事数が、典型例です。

ほかの信号との違いは、集約です。ログは、イベントごとに一行を保ち、トレースは、一つのリクエストをシステム内で追います。メトリクスは、一定の時間内の多数の測定を、回数、合計、現在値、区間ごとの分布といった、少数の統計にまとめます。OpenTelemetryの文書は、結果を、集約された統計情報と呼びます。強みでも限界でもあり、メトリクスが、直近一分で何件失敗したかを示すのに対し、ログは、どれが失敗したかを示します。

OpenTelemetryとPrometheusは、少数の種類で一致します。適切なものを選ぶことが、設計の大半です。

- **カウンター**は、車の走行距離計のように、累積し、増えるだけです。処理したリクエスト、送信バイト、見つけたエラーなどです。Prometheusは、プロセス再起動時のゼロへのリセットだけを例外として認めます。
- OpenTelemetryの**増減カウンター**は、累積しつつ減ることもできます。仕事が出入りする待ち行列の長さなどです。
- **ゲージ**は、同じ車の燃料計のように、読み取る瞬間の現在値です。使用中のメモリー、温度、処理中のリクエストなどです。
- **ヒストグラム**は、測定を区間に分け、件数と合計を保ち、一秒未満だったリクエストの数などに答えます。

Prometheusには、プログラム内で分位点を計算するサマリーもあります。サマリーを意味のある形で統合できるとは限らないため、OpenTelemetryのデータモデルでは、旧来の型としてのみ残します。

Prometheusが取得するテキスト形式には、部品がまとまって現れます。説明行、型の行、続いてラベルの組み合わせごとに一行で、名前、ラベル、値と、任意で、基準時点からの経過時間をミリ秒で表す時刻が並びます。

```
# HELP http_requests_total The total number of HTTP requests.
# TYPE http_requests_total counter
http_requests_total{method="post",code="200"} 1027 1395066363000
http_requests_total{method="post",code="400"} 3 1395066363000
```

概念を判断に生かす

「メトリクスとは？」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

メトリクスは、あらゆる対象で保つことができるほど低コストです。必要なメモリーは、通信量ではなく、異なるラベルの組み合わせの数に従って増えます。OpenTelemetryの文書も、ログとの違いとして、そう明言します。十億回増やしても、カウンターは一つの数値です。だから、リクエストの一部ではなく全部を数え、今月と先月を比べるまで、長く記録を保てます。

議論を測定に変えます。ホストの容量が足りるか、リリースで応答が遅くなったか、ディスクがいつ埋まるか。その問いには、誰かが尋ねる前から記録していた時系列で答えられます。アラートも同じ数値で作ります。通知規則とは、人を起こす価値があると誰かが判断した、メトリクス上の条件だからです。

出力するコードより長く残るので、形を正しく決める価値もあります。Prometheusの命名ガイドは、一つのメトリックに一つの単位と一つの量を求めます。ミリ秒やメガバイトではなく、秒やバイトなどの基本単位を使い、単位は

複数形の接尾辞にし、累積の数には `_total` を付けます。`http_request_duration_seconds` や `process_cpu_seconds_total` という名前なら、入っているものが、そのまま分かります。

合計をどう運ぶかも重要です。Prometheusは、累積値を報告するので、取得が断続的に失敗しても、その欠測の影響はならされます。その代わり、送信側には、ラベルの組み合わせの数に比例するメモリーが必要です。StatsDは、前回の報告以降の変化である差分を送り、その負担をプロセスの外に移します。

実際の活用例

自社の公開サービス用ホストは、メモリー512 MB、使用可能分が約464 MBの、一コアのマシンです。そこでは、メモリーが判断を決める材料になりました。報告は、要求されたときだけ、ホストを読んでMarkdownを作る七つのコマンドで、何も変更しません。その512 MBの一台で、onionサービスも動かせるかという問いには、意見ではなく、記録した一日分の測定値で答えました。利用可能なメモリーは、464 MB中約310 MBを下回らず、スワップは2.6%、プロセッサーのアイドル時間は99.7%でした。

その後、報告は数値の意味も示す必要がありました。計測できていない状態と、正常なホストが同じページになっていたため、六つの読み取り専用の収集器に、何を読まなかったかを明示させました。デーモンがログ形式を変えたミラーも、誰も訪れなかったミラーも、同じゼロを出していました。しきい値は、既にメトリクス報告が測った116サンプルに合わせ、観測した範囲も境界の横に記録しました。報告は、違反、違反なし、測定がなく判定しない、の三つを区別します。各収集器は、ゼロの場合も、実行時間と、読めたファイルと読めなかったものの数など、見えていない範囲を報告します。しきい値は日平均に適用するため、一分の急増が平均に埋もれることがあります。十五分ごとの切り替え機構は、実行のたびに利用可能メモリーとディスクの空きを調べますが、検査の合間の短時間のしきい値逸脱は、なお見逃す可能性があります。

最も役立つメトリクスは、ホストより、問題そのものから生まれることがよくあります。限られた在庫を売り越さずに販売するノートは、ロック待ち、期限切れの確保、入口のゲートとデータベースの数の差を、一つのダッシュボードに置きます。`available + held + sold = total` の不変条件を定期的に調べ、図示し、ずれれば必ず当番を呼びます。どの数値も、その販売がどう壊れるかを知ることから生まれます。

問題が起きるところ

メトリクスは、派手には壊れません。グラフは描かれ続けますが、気づかないうちに、数値は名前とは違うものを意味するようになります。

下がる値を、カウンターとして公開します。 Prometheusの型の説明は、実行中のプロセス数のように減るものには、カウンターを使わず、ゲージを使うよう述べます。クエリー関数は、カウンターの低下をすべて再起動とみなし、補正するため、意図して下がる値が、実際にはない増加として現れます。

サービス間で単位がずれます。 一方がミリ秒、他方が秒で所要時間を出せば、比べるパネルには千倍のずれが生じます。命名ガイドは、一つのメトリックは一単位を表し、基本単位を使うべきだと述べます。元で一度変換する方が、すべてのクエリーで差を覚えるより安く済みます。

ラベルをまたいだ合計が、無意味になります。 命名ガイドの目安は、一つのメトリックの全ラベルにわたる合計や平均が、有意義であることです。例として、複数の待ち行列の容量は、一つのメトリックに入りますが、ある待ち行列の容量と現在の長さは、同じものにしません。

未計測が、ゼロと読まれます。 入力を読まなくなった収集器は、ゼロか、行の欠落を残し、活動の少ないシステムと見分けがつきません。OpenTelemetryのデータモデルは、値の記録がない点を示すフラグを持ち、Prometheusの失効マーカーに相当します。データがないこととゼロは、別の事実だからです。違う形で報告し、上の収集器のように、「判定しない」に独自の結果を設けます。

期間が、急増を隠します。 ゲージは一瞬のサンプルで、OpenTelemetryのデータモデルは、各間隔の最後の値だけを報告します。二つの採取時点の間の急増は、跡を残しません。一分間、上限に張り付いても、日平均は同じように平滑化します。見つけるべき故障から期間を選び、両方が大事なら、低頻度の検査に加え、高頻度の検査も続けます。

オーバーフローで、絞り込んだクエリーの数が少なくなります。 OpenTelemetry SDKは、既定で、メトリックの各ストリームについて、異なる属性の組み合わせを最大2,000まで追い、それ以上は、`otel.metric.overflow=true` の一つの点にまとめます。総数は正しいままですが、値の種類が多い部分だけでなく、組み合わせ全体を置き換えます。文書の例では、リクエストのカウンターに `url.path` と

`success` があります。上限を超えると、`success=false` のクエリーは少なく数え、それに基づくエラー率のアラートが出なくなる可能性があります。属性を、小さく既知の集合に保ち、オーバーフローのマーカーを監視します。

参考資料

- OpenTelemetryのメトリクス — 計測器の種類、集約、要素数の上限です。
- OpenTelemetryのメトリクスデータモデル — ゲージ、合計、ヒストグラム、時間的な集計の扱い、値が未記録のフラグです。
- Prometheusのメトリック型 — カウンター、ゲージ、ヒストグラム、サマリーです。
- Prometheusのメトリックとラベルの命名 — 基本単位、単位の接尾辞、`_total` です。
- Prometheusのヒストグラムとサマリー — 分位点を計算する場所と、結合できるものです。
- Prometheusの公開形式 — テキスト形式を、一行ずつ説明します。

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

「メトリクスとは？」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/metrics/