Engineer

監視

稼働中のシステムを見守り、利用者より先に異常を知らせること

どのようなものか

監視は、稼働中のシステムが自身について出す報告を、継続して読むことです。誰かが決めた境界と報告を照らし合わせ、人に知らせる必要があるかを判断する仕組みが動き続けます。GoogleのSite Reliability Engineeringは、システムの定量データをリアルタイムで収集、処理、集約、表示することと定義します。届くリクエストの数と種類、失敗の数と仕方、所要時間、各サーバーの稼働時間などです。目指すのは、出来事の順序です。利用者が知らせる前に、システムが異常を知らせます。

データは二方向から得ます。ホワイトボックス監視は、メトリクス用エンドポイント、待ち行列の長さ、再試行回数など、システムが内部について公開するものを読みます。誰かが影響を受ける前に、問題が育つ様子を見つけられます。ブラックボックス監視は、外に立ち、利用者のように振る舞います。ページを要求し、名前を解決し、ログインを完了して、成功したかどうかと所要時間を報告します。既に壊れたものだけを見るため、人を起こす信頼できる根拠になります。

監視システムが保つものの大半は、時間に沿って採取する数値です。保存が安く、しきい値と比べやすいからです。同じSREの章は、利用者に応えるシステムの出発点として、四つを挙げます。レイテンシーは、リクエストの所要時間、トラフィックは、届く需要の量、エラーは、失敗するリクエストの割合、飽和度は、最も忙しい資源が容量の限界にどれだけ近いかです。

数値からは、二つの出力を作ります。ダッシュボードは、誰かが既に異変を疑っているときに見るものです。アラートは、全員がほかの仕事をしているときに、割り込むものです。Prometheusの構成では、サーバーが規則を評価し、発生したアラートをAlertmanagerがまとめ、重複を除き、送り先へ振り分けます。

近い二つの概念と、混同しがちです。ログは、出来事ごとに一行ずつ、個々のイベントを書き留め、アラートの後、原因を調べるために読みます。OpenTelemetryが説明するオブザーバビリティは、外からシステムに新しい問いを投げかけられる、より広い能力です。監視は、それより狭く、古い実務です。先に決めた問いを継続して尋ね、その答えで人を呼び出せるようにします。

概念を判断に生かす

「監視とは？」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

監視しなければ、被害を受けた人が気づいた時刻に、その人から問題を知ることになります。監視は、その時点を早め、対応できる人に知らせます。

足りない資源は、ソフトウェアより注意力です。呼び出しは、昼の仕事と夜の睡眠を中断します。SREの本は、呼び出しが多過ぎると、人は流し読みし、疑い、最後には無視し、雑音に隠れた本当の異常まで見逃すことがあると、はっきり述べます。Prometheusの助言は、一文に収まります。「アラートは単純にし、症状に対して通知し、原因を絞り込める良いコンソールを用意し、何もすべきことのない呼び出しを避ける」。人を起こせる規則はすべて、発生時には、その人が急いですべきことがあるという約束です。

中心となる判断は、利用者が何を感じるかです。85%埋まったディスクは原因、失敗する購入手続きは症状です。症状に対して通知すれば、規則の数を少なく保てます。機械の調子が悪くなる仕方より、利用者が困る仕方の方が、はるかに少ないからです。原因は、ダッシュボードやチケットで扱い、人を起こさず診断に役立てます。容量は、妥当な例外です。限界に近い資源は、今日は誰も困らせなくても、近いうちに人の対応が必要になります。

次は、自分で運用するか、借りるかです。運用には、時系列の保存先、通知の振り分け先を決める仕組み、その両方を健全に保つ人が要ります。外部が運用するサービスなら、その仕事を任せられますが、送信量で課金され、新しいホストは一台ごとに請求へ現れます。SREの本によれば、Googleでも、10–12人のチーム

には、通常、監視を主な仕事とする人が一人か二人います。どちらを選んでも、固有の保守が必要なシステムであり、見る対象に合った規模にする必要があります。

実際の活用例

創業者の以前の二つの職務と、自社のホストに、同じ考えが大きく異なる規模で現れます。

2015年から2022年までのインフラエンジニアの職務では、約1,000台のIPTV/OTT配信サーバーを24/7で支援し、さらに多くの顧客所有システムも扱いました。顧客は中国、米国、ドイツにいました。三つのタイムゾーンに顧客がいるため、静かな時刻はなく、現地の午前3時の問題は、別の場所の視聴ピーク時に当たりました。停止になる前に、何かがずれ始めたことを見つけるのが、仕事の大部分でした。視聴者の画面に届く前に、問題へ対処しました。この種のサービスでは、何も起きないことが成功に見えます。監視があれば、そう願うだけでなく、そうだと確かめられます。

2022年から2024年のデータベースエンジニアの職務では、PostgreSQL、MS SQL、BigQueryの100個のデータベースに対する自動監視で、停止時間が50%減りました。

自社の公開サービス用ホストの監視は、規模の対極にあります。512 MBの一台で公開サービスを動かし、使えるメモリーは約464 MBです。設計上の判断は、すべてメモリーが決め手になります。報告は、エージェント、パイプライン、ダッシュボードではなく、要求時に取得します。七つのコマンドがホストを読み、Markdownを作り、何も変えず、要求されたときだけ動きます。onionサービスの判断は、その方法を示します。問いが出てから、一日、ホストのサンプルを採り、測定値で決めました。その記録では、利用可能なメモリーの最小は464 MB中約310 MBで、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。この規模の監視は、判断に必要になった時点で行う測定です。

問題が起きるところ

呼び出しが多過ぎて、誰も読みません。 障害対応のたびに規則が残り、自然に消える原因で鳴るものが増え、数か月で、本当の呼び出しも一緒に流し読みされます。利用者が気づくものだけを対象に、スタック内の一か所から人を呼び出し、残りはチケットの待ち行列かダッシュボードへ送ります。さらに、通知を出す前に条件がしばらく続くことを求め、一分で消える揺らぎで人を起こさないようにします。

```
groups:
    - name: checkout
      rules:
          - alert: CheckoutErrorsHigh
            expr: job:checkout_errors:ratio5m > 0.02
            for: 10m
            labels:
                severity: page
```

監視が止まり、その沈黙が正常と読まれます。 一度も通知を発しない規則と、一度も壊れないシステムは、同じように見えます。Prometheusサーバー、Alertmanager、メール中継のいずれかが止まれば、すべての通知が同時に止まります。Prometheusは、監視自体を通知の対象にし、さらに望ましくは、全経路を端から端まで試すよう勧めます。一つの検査で、規則から受信箱まで通知が届くと示せます。定期ジョブも同じ罠に陥ります。止まったバックアップはエラーを出さず、痕跡を残さなくなるだけです。最後の成功が古過ぎれば通知し、少なくとも二回分の実行を終えられる余裕を持たせます。4時間ごとに動く、一時間のジョブなら、同プロジェクトは10時間を提案します。

```
time() - backup_last_success_timestamp_seconds > 10 * 3600
```

一部の利用者は待っているのに、平均は正常です。 平均はグラフにしやすく、分布の遅い側を隠します。SREの本には、毎秒1,000リクエストを受け、平均100 msのサービスでも、1%は簡単に5秒かかり得る例があります。ページが、そのようなサービスをいくつも使えば、一つのバックエンドの99パーセンタイルが、フロントエンドの中央値になることがあります。すぐに失敗するリクエストは、平均を下げるため、状況をさらに悪くします。所要時間を区間別に記録し、高いパーセンタイルで通知し、失敗したリクエストは、成功と別に追います。

内部の数値はすべて正常でも、利用者はサービスに届きません。 ホワイトボックスのメトリクスは、内部から来るため、内部の死角も共有します。証明書の期限切れ、DNSレコードの破損、ファイアウォールの変更があっても、すべて健全に見える場合があります。システムのネットワーク外のマシンから行うブラックボックス検査は、そうした見えない故障を捉え、内部の監視がほかと一緒に止まっても、動き続けます。

参考資料

- Prometheusの概要 — メトリクスに基づく監視が集めるもの、構成する部品、個々のイベントの精度に別のものが必要になる場合です。

- アラート — オンラインサービス、バッチジョブ、監視の監視まで、何を人の呼び出しにつなげるかについての、Prometheusの助言です。
- Alertmanager — 通知の重複を除き、まとめ、一時的に通知を止め、抑制し、受信先へ振り分ける仕組みです。
- 分散システムを監視する — GoogleのSite Reliability Engineeringの第6章です。症状と原因、四つのゴールデンシグナル、遅い側の分布を扱います。
- オブザーバビリティ入門 — OpenTelemetryによる信号、SLI、SLOの説明と、オブザーバビリティが監視を超える範囲です。

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

「監視とは？」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/monitoring/