Engineer

Objectif de niveau de service

Cible mesurable de fiabilité qu'un service doit atteindre

Ce que c'est

Un objectif de niveau de service est une cible décrivant le bon fonctionnement d'un service, formulée assez précisément pour que chacun puisse vérifier si elle a été atteinte : une proportion de demandes qui doivent réussir, ou recevoir une réponse assez rapide, sur une période définie. Il est presque toujours abrégé en SLO. Le livre Site Reliability Engineering de Google le définit comme une valeur cible, ou une plage de valeurs, pour un niveau de service mesuré par un indicateur.

Cet indicateur est l'indicateur de niveau de service, ou SLI : une quantité définie avec soin pour décrire un aspect du service réellement reçu par les utilisateurs. La latence des demandes est le premier choix habituel, suivie de près par le taux d'erreur et le débit. Le SRE Workbook recommande de formuler chaque SLI comme un rapport entre événements satisfaisants et ensemble des événements. Il va ainsi de 0 %, où rien ne fonctionne, à 100 %, où rien n'est défaillant. L'objectif trace ensuite une limite sur cette échelle : l'indicateur doit rester du bon côté d'une cible, ou entre une borne inférieure et une borne supérieure.

Un objectif complet nomme tout ce qui permet de le vérifier. L'exemple du livre est le suivant : 99 % des appels RPC Get se terminent en moins de 100 ms, cette proportion étant moyennée sur une minute et mesurée sur l'ensemble des serveurs backend. Un indicateur peut porter plusieurs cibles à la fois, par exemple 90 % des appels en moins de 1 ms, 99 % en moins de 10 ms et 99,9 % en moins de 100 ms, car un seul percentile décrit rarement toute une distribution. Le Workbook distingue aussi la spécification, le résultat qui compte pour l'utilisateur, de l'implémentation, l'endroit où il est mesuré : un journal serveur, une sonde qui exécute un navigateur ou du code dans la page. Chacun voit des défaillances différentes, et ce choix doit figurer dans la définition.

Un SLO n’est pas un SLA. Un accord de niveau de service est un contrat avec les utilisateurs, explicite ou implicite, qui associe des conséquences au respect ou non des objectifs qu’il contient, le plus visiblement une remise ou une pénalité. Le test proposé par le livre consiste à demander ce qui arrive si l’objectif n’est pas atteint : si aucune conséquence explicite ne suit, il s’agit d’un SLO. Une note ajoute que ceux qui parlent d’un SLA violé désignent presque toujours un SLO manqué, car une véritable violation d’un SLA peut conduire au tribunal pour rupture de contrat. Google Search, précise le livre, n’a pas de SLA destiné au public, mais une panne lui coûte malgré tout en réputation et en recettes publicitaires.

Utilisez le concept pour prendre une décision

Utilisez « Qu’est-ce qu’un objectif de niveau de service ? » pour éclairer une décision réelle. Nous pouvons vous aider à relier ce concept au système que vous construisez ou achetez. Une supervision et des alertes qui signalent les incidents tant qu'ils restent mineurs, avec les chiffres qui montrent à quelle fréquence ils le restent.

Découvrir nos prestations Contactez-nous →

Vérifiez les hypothèses derrière le terme

Appuyez-vous sur l'explication et les réalisations liées pour formuler les questions de votre projet. Nous pouvons examiner les hypothèses avec vous avant le choix d'un outil. Premier tableau de bord d'observabilité — première vue de l'entreprise sur ses propres systèmes

Premier tableau de bord d'observabilité Contactez-nous →

Pourquoi c’est important

Sans objectif formulé, les utilisateurs inventent le leur. Le livre SRE décrit les deux conséquences possibles : des personnes font plus confiance au service qu’elles ne le devraient, tandis que des utilisateurs potentiels le croient moins fiable qu’il ne l’est et s’en détournent. Il prend l’exemple de Chubby, le service de verrouillage distribué de Google. Ses pannes globales étaient si rares que d’autres services en sont venus à dépendre de son absence de panne sans que cette dépendance attire l’attention. Ces services défaillaient lors des rares pannes. La réponse a été de faire respecter son SLO à l’instance globale sans le dépasser de beaucoup : chaque trimestre où les pannes réelles n’avaient pas fait passer la disponibilité sous la cible, les SRE arrêtaient volontairement le système.

L’objectif tranche aussi la question de la fiabilité suffisante. « Une fiabilité de 100 % est la mauvaise cible », écrit le SRE Workbook, pour des raisons pratiques. Les composants peuvent tomber en panne en même temps, quelle que soit leur conception. La chaîne de réseaux et d’appareils entre le service et ses clients peut aussi échouer : les clients ne verraient donc jamais les 100 %. Chaque chiffre neuf supplémentaire coûte plus cher que le précédent, alors que sa valeur pour les utilisateurs tend vers zéro. Comme le changement est la principale cause

d'indisponibilité, un service qui ne doit jamais tomber en panne ne peut pas non plus être mis à jour. Une fois la cible inférieure à 100 %, l'écart avec la perfection devient une marge à dépenser en livraisons, expériences et maintenance planifiée : le budget d'erreur, tout simplement 100 % moins le SLO. Quelqu'un doit être responsable de cet arbitrage entre livrer des fonctionnalités et rester disponible. Le Workbook désigne le directeur technique, ou CTO, dans une petite organisation, et le product owner ou le responsable produit dans une plus grande.

Enfin, l'objectif transforme la supervision en décisions. Le livre décrit une boucle : mesurer les indicateurs, les comparer aux objectifs, décider s'il faut agir et le faire. Si la latence augmente et doit franchir la limite dans quelques heures, l'équipe vérifie si les serveurs sont limités par le processeur et en ajoute. Sans objectif, souligne le livre, personne ne peut dire s'il faut agir, ni quand.

À quoi cela ressemble en pratique

Prenons une API de passage en caisse et un objectif formulé comme le suggère le Workbook : la description de l'indicateur est séparée de celle du lieu de mesure, la fenêtre est précisée et la justification est consignée à côté :

```
Service :          API de passage en caisse
SLI :              part des requêtes POST /checkout dont la réponse n'a pas
                    de statut 5xx, comptée dans le journal des requêtes du serveur
Objectif :         99.9% sur une fenêtre glissante de 28 jours
Justification :    quatre semaines d'observation, valeur arrondie à la baisse
Responsable :      responsable produit, avec les ingénieurs d'astreinte
Révision :         mensuelle jusqu'à stabilisation, puis trimestrielle
```

La fenêtre compte autant que le pourcentage. Le Workbook préfère une fenêtre glissante à un mois civil, car les utilisateurs n'oublient pas une panne survenue le dernier jour du mois lorsque le suivant commence. Il privilégie aussi un nombre entier de semaines, pour que chaque fenêtre contienne le même nombre de week-ends ; une fenêtre de quatre semaines est son choix pour les usages courants. Cette fenêtre fait 28 × 24 × 60 = 40 320 minutes, dont 0,1 % représente un peu plus de 40 minutes d'arrêt complet. Sur un mois de 30 jours, les mêmes 99,9 % autorisent 43,2 minutes, chiffre figurant dans le tableau de disponibilité du livre.

Le temps reste toutefois la façon la moins fine de compter. Le livre SRE explique que Google mesure la disponibilité par le taux de réussite des demandes plutôt que par le temps de fonctionnement, car un service réparti dans le monde est rarement entièrement en service ou entièrement arrêté. En nombre de demandes, l'exemple du Workbook est le suivant : à 99,9 %, 3 millions de demandes sur quatre semaines laissent une marge de 3 000 échecs, et une panne qui en provoque 1 500 consomme la moitié du budget. Compter les demandes donne aussi un coût aux défaillances partielles que le temps de fonctionnement ne voit pas. À trafic constant, 20 minutes où la moitié des demandes échouent coûtent autant que 10 minutes d'arrêt total, soit environ un quart du budget sur 28 jours.

Dans Prometheus, avec un compteur de demandes étiqueté par statut, l'indicateur sur la fenêtre est un simple rapport, construit sur le modèle de l'exemple du Workbook :

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

Sur ce site, la page niveaux de service se situe du côté des accords. Pour un système exploité par l'entreprise pour le compte d'un client, le temps de réponse à un incident, les horaires couverts, la disponibilité exigée et les fenêtres de maintenance sont écrits dans l'accord propre au système. Ils sont définis système par système, car un rapport nocturne et un service de paiement appellent des réponses différentes. Le dispositif de suivi des erreurs et de traçage internes, construit par la fondatrice dans le cadre de son activité actuelle au sein de NextMariner, est présenté dans un ensemble de vues d'exploitation, dont une consacrée aux objectifs de niveau de service. La note sur la vente du dernier billet une seule fois transforme quant à elle un invariant de stock en objectif : le vérifier à intervalles planifiés, le tracer et déclencher une alerte d'astreinte à toute dérive. Zéro survente devient ainsi un objectif assorti d'une alerte, plutôt qu'un espoir.

Où les problèmes surviennent

La cible est copiée du tableau de bord du mois dernier. Adopter ce que le système atteint aujourd'hui semble prudent, mais le livre SRE avertit qu'une cible choisie sans réflexion peut condamner une équipe à des efforts héroïques pour continuer à l'atteindre, sur un système qui ne peut s'améliorer sans refonte.

Partez des besoins des utilisateurs et écrivez le raisonnement près du chiffre. Le Workbook demande la justification et précise qu'il faut indiquer si les valeurs viennent d'expériences ou d'observations, même lorsqu'il s'agit explicitement de valeurs ad hoc. Réexaminez le document chaque mois au début, puis chaque trimestre une fois stabilisé.

L'objectif est de 100 %, ou une suite de chiffres neuf dont personne n'a chiffré le coût. Un système toujours disponible, ou capable de grandir sans limite, est cher à construire et à exploiter, et dépasse ce que les utilisateurs remarqueraient. Commencez avec une cible souple et resserrez-la en apprenant à connaître le système. Le livre préfère cela à une cible stricte qu'il faudrait ensuite assouplir parce qu'elle s'est révélée hors d'atteinte.

Une moyenne remplace ce que ressentent les utilisateurs. Dans l'exemple du livre, une demande habituelle prend environ 50 ms, tandis que 5 % des demandes prennent vingt fois ce temps. Pourtant, le graphique de la moyenne ne montre rien de particulier de toute la journée. Écrivez les objectifs de latence sous forme de percentiles, ou de proportion de demandes sous un seuil comme 300 ms, jamais sous forme de moyenne.

Personne ne peut défendre autant d'objectifs. Une douzaine de SLO par service les dilue tous. Le test du livre est direct : si citer un objectif ne peut jamais faire gagner une discussion sur les priorités, il ne mérite probablement pas d'exister. Gardez les quelques objectifs qui couvrent les usages réels.

Manquer l'objectif ne change rien. Le Workbook appelle cela un SLO sans effet : le respect de l'objectif devient un KPI supplémentaire dans un rapport, plutôt qu'une raison de décider. Convenez à l'avance de ce qui arrive lorsque le budget est épuisé, et faites signer cette politique par le responsable produit, les développeurs et les SRE. L'exemple de politique de budget d'erreur du Workbook suspend les changements ordinaires tant que le service ne respecte plus son objectif. Il n'autorise alors que les corrections de bugs les plus prioritaires et les travaux de sécurité, et précise explicitement que cette politique n'est pas une punition.

Le SLO est le même chiffre que le SLA. Quand l'objectif interne coïncide avec le contrat, l'équipe n'apprend les difficultés qu'au moment de la violation. Le livre recommande un objectif interne plus strict que celui annoncé, pour garder une marge face aux problèmes chroniques. L'erreur inverse compte aussi : un service

qui fonctionne bien mieux que son objectif apprend aux utilisateurs à dépendre de cette qualité. Le livre suggère donc des arrêts planifiés occasionnels, une limitation de débit ou un système qui ne va pas plus vite à faible charge.

La fenêtre dépasse la durée des données. Un objectif sur 28 jours calculé sur un serveur Prometheus qui ne garde que 15 jours d'échantillons, valeur par défaut en l'absence de paramètre de conservation, ne mesure alors que 15 jours, sans le signaler, tout en les présentant comme quatre semaines. Réglez la conservation au-delà de la fenêtre avant de faire confiance au chiffre :

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

Pour aller plus loin

- Objectifs de niveau de service — Le chapitre 4 du livre Site Reliability Engineering de Google : indicateurs, objectifs, accords et choix d'une cible.
- Mettre en œuvre des SLO — Le chapitre du SRE Workbook sur les SLI sous forme de rapports, les budgets d'erreur, les fenêtres et la documentation de l'objectif.
- Exemple de politique de budget d'erreur — Une politique complète pour un service de jeu fictif : quand les livraisons s'arrêtent et qui tranche un désaccord.
- Accepter le risque — Pourquoi Google mesure la disponibilité par la réussite des demandes plutôt que par le temps de fonctionnement, et le coût de chaque palier de fiabilité.
- Tableau de disponibilité — La durée d'indisponibilité autorisée par chaque niveau de disponibilité sur une année, un trimestre, un mois, une semaine, un jour et une heure.
- Stockage — La conservation des échantillons sur disque dans Prometheus et les paramètres de rétention qui déterminent jusqu'où une requête de SLO peut remonter.

Glossaire

Choisissez selon vos contraintes réelles

Présentez le parcours de déploiement, les risques opérationnels et les personnes d'astreinte. Nous pouvons définir un changement avec des vérifications, une procédure de reprise et une responsabilité claire.

Consulter les tarifs Contactez-nous →

Rendez la terminologie concrète

Demandez comment « Qu'est-ce qu'un objectif de niveau de service ? » s'applique à votre organisation. Un exemple concret tiré de votre système actuel suffit pour engager la discussion.

Découvrir nos prestations

welcome@engineer.company +45 50 10 61 84 Copenhague, Danemark

Emportez cette page avec vous

Fiche de contact

Créé avec soin et amour par D.

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

Droit d'auteur © 2025 – aujourd'hui · Engineer ApS · Tous droits réservés.

Imprimé depuis https://engineer.company/fr/glossary/service-level-objective/