Engineer

Migration

Passage planifié vers une nouvelle version ou plateforme

Ce que c’est

Une migration est le déplacement planifié, par étapes, d’un élément déjà utilisé vers une nouvelle version ou plateforme, l’ancienne étant retirée à la fin. Il peut s’agir d’une charge de travail entière, d’une application, d’un ensemble de données ou de leur structure de stockage. Contrairement à une reconstruction, le système reste utilisé pendant le déplacement : utilisateurs, enregistrements et intégrations suivent, et l’ancienne version fonctionne jusqu’à la prise de relais par la nouvelle.

Les ingénieurs emploient le mot à deux échelles, souvent sans préciser laquelle.

À grande échelle, il s’agit d’une nouvelle plateforme ou version majeure. Le Cloud Adoption Framework de Microsoft décrit le transfert de charges vers Azure, depuis des serveurs locaux ou un autre cloud, en cinq étapes : planifier, préparer les charges, exécuter, optimiser puis retirer la source. Les clients d’Esri déplacent cartes et données d’ArcMap vers ArcGIS Pro. Une application Next.js passe du Pages Router à l’App Router ; `getServerSideProps` laisse place à la récupération des données dans les Server Components. Un cluster PostgreSQL passe à une nouvelle version majeure modifiant souvent la structure des tables système : les données sont transférées par `pg_upgrade` ou par export puis restauration.

À petite échelle, une migration est une modification versionnée de la structure d’une base, conservée dans un fichier du dépôt et appliquée dans un ordre fixe par un outil. Django les présente comme un système de gestion de versions du schéma : chaque fichier décrit un changement des modèles, est commité avec le code et s’exécute dans le même ordre sur le portable du développeur, en préproduction et en production. GitLab livre des migrations de fond par lots qui réécrivent de grandes tables après une mise à niveau, par exemple en élargissant

des colonnes de nombres entiers en `bigint` avant débordement. Une migration de données en fait aussi partie : elle change les lignes plutôt que les colonnes, comme remplir un nouveau champ à partir de deux anciens.

C'est la même idée à deux échelles : état initial connu, étapes ordonnées, cible et moyen de retour si une étape échoue. Un changement de plateforme contient généralement des migrations de schéma ; une longue suite de celles-ci est, au fil des années, une plateforme se transformant par morceaux sous ses utilisateurs.

Utilisez le concept pour prendre une décision

Utilisez « Qu'est-ce qu'une migration ? » pour éclairer une décision réelle. Nous pouvons vous aider à relier ce concept au système que vous construisez ou achetez. Transférer les données d'un système vieillissant vers une solution maintenue, tout en poursuivant l'activité pendant la migration.

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. Cartes GIS 10 fois plus rapides sur PostgreSQL — 10x cartes plus rapides après avoir quitté MSSQL

Cartes GIS 10 fois plus rapides sur PostgreSQL Contactez-nous →

Pourquoi c'est important

Le logiciel survit à son socle. Les contrats cloud sont renégociés, les frameworks changent de modèle et les fournisseurs retirent des produits. Un système conservé dix ans sera migré plusieurs fois : la qualité des migrations finit donc par compter davantage que le choix de la première plateforme.

- **La coexistence des deux états concentre les coûts.** Tant qu'une partie fonctionne sur l'ancienne plateforme et l'autre sur la nouvelle, il faut exploiter et superviser les deux, et préserver leur cohérence. Le Cloud Adoption Framework recommande une période courte et le déplacement, dans une même vague, des composants directement dépendants.
- **Une base ne se redéploie pas.** Le code peut être reconstruit et remplacé ; pas les lignes en production. Les migrations de schéma transforment un changement de structure en production en fichier que l'on peut examiner en revue, tester sur une copie, appliquer dans l'ordre et, pour la plupart des changements, exécuter en sens inverse. Sinon, le schéma de production est ce qu'a laissé la dernière personne ayant une console.
- **Un déplacement planifié peut être annulé.** Le Cloud Adoption Framework demande de fixer critères et procédures de retour arrière avant toute migration, de les tester et de prévoir des points de décision de poursuite ou d'arrêt.

Le compromis se joue à la bascule. Le Cloud Adoption Framework distingue les migrations avec interruption de service de celles avec une interruption quasi nulle, où les données sont répliquées en continu avant la bascule finale. Pour une version majeure PostgreSQL, `pg_upgrade` raccourcit ce moment : il évite l'export et la restauration habituels en réutilisant les fichiers existants. Le mode `--link`, qui ne copie aucun fichier, est encore bien plus rapide.

À quoi cela ressemble en pratique

Trois projets de ce site montrent le terme à trois tailles, d'une plateforme entière à l'ordre des fichiers d'un dépôt.

La première migration consistait à quitter un fournisseur cloud. L'installation AWS d'un client était surdimensionnée ; le travail a commencé par l'examen de l'usage réel : retrait des services inutiles et ajustement de la capacité restante. Les charges ont ensuite quitté AWS. Le dossier de mission établit ces changements d'infrastructure. Retirer l'inutile avant le déplacement réduisait ce qu'il fallait transférer ; cela n'établit ni ratio de coût à périmètre identique, ni résultat de productivité, ni migration sans perturbation.

La deuxième associait un changement de moteur à une nouvelle conception cible. La base d'une application cartographique est passée de MSSQL à PostgreSQL avec PostGIS, sans copier l'ancienne structure. Le nouveau schéma visait les données spatiales, avec index GiST et SP-GiST, clés étrangères strictes et contraintes de vérification imposant les règles de validation des coordonnées spatiales. Plus de 50 millions d'enregistrements spatiaux ont été transférés par des pipelines ETL les convertissant en EPSG:4326. Changer de conception permet d'inscrire les règles dans le schéma lui-même. La page rapporte une performance des requêtes spatiales multipliée par 10 et une baisse de 65 % de la charge CPU du backend.

La troisième illustre la petite échelle maintenue pendant des années. Le schéma d'une plateforme est la somme de 1 022 migrations, numérotées de 036 à 1102 ; la page présente la rigueur de cet ordre d'exécution comme peu spectaculaire et non négociable. L'historique du dépôt décrit la production ; il ne reste vrai que si toute base, d'une base de développement fraîchement créée à une restauration, peut être reconstruite en le rejouant du premier au dernier fichier.

Où les problèmes surviennent

La planification se concentre sur la bascule ; les problèmes s'accumulent souvent dans les semaines qui l'entourent, sous quatre formes reconnaissables.

Deux systèmes coexistent plus longtemps que prévu. Le déplacement s'arrête à mi-chemin, les éléments restants sont les plus difficiles, et l'équipe exploite les deux plateformes pendant des mois. Les utilisateurs sentent la séparation : le guide Next.js précise que passer entre une page Pages Router et une page App Router déclenche une navigation complète, sans préchargement par `next/link` entre les deux. Gardez de petites vagues et rendez chaque étape suivante peu coûteuse. Une interface présentée ici déplace ses pages vers une fabrique partagée chargeant chaque page en un aller-retour : chaque page convertie hérite du comportement.

Une bascule sans retour. Certaines étapes modifient la source autant que la destination. Avec `pg_upgrade --link`, le démarrage du nouveau cluster écrit dans les fichiers partagés avec l'ancien, qui doit alors être restauré depuis une sauvegarde. Esri avertit qu'un jeu de données adapté à ArcGIS Pro peut devenir inutilisable dans ArcMap et demande une copie préalable. Dans Django, une opération `RunSQL` sans SQL inverse déclenche `IrreversibleError` dès le retour arrière. Fixez les critères avant la première étape, répétez le parcours de retour arrière et faites-en un contrôle : le moteur de contrôle d'un dépôt présenté ici comprend une vérification de réversibilité des migrations.

Une migration déjà exécutée est modifiée. Les outils enregistrent les fichiers appliqués puis les ignorent. Modifier un fichier arrivé en production y laisse l'ancienne version, mais donne la nouvelle aux bases neuves : deux schémas sous un historique. Ne touchez pas aux migrations appliquées ; corrigez par une nouvelle. Une procédure qui rejoue l'historique à partir d'une base vide puis compare le résultat à une copie migrée détecte cette dérive.

La base mise à niveau est lente le premier jour. `pg_upgrade` transporte la plupart des statistiques de planification, pas toutes : pour les autres, le planificateur devine jusqu'à leur reconstruction. Son mode `--check` ne change rien et peut tourner pendant que l'ancien serveur répond encore : lancez-le d'abord, puis reconstruisez les statistiques manquantes juste après la mise à niveau, avant de recueillir les autres :

```
pg_upgrade --check -b "$OLD/bin" -B "$NEW/bin" -d "$OLD/data" -D "$NEW/data"
pg_upgrade -b "$OLD/bin" -B "$NEW/bin" -d "$OLD/data" -D "$NEW/data"
vacuumdb --all --analyze-in-stages --missing-stats-only
vacuumdb --all --analyze-only
```

Pour aller plus loin

- Planifier une migration — méthode Microsoft pour transférer des charges vers Azure : vagues, choix de bascule et retour arrière.
- Migrations — changements de schéma versionnés, annulation et migrations de données dans Django.
- pg_upgrade — outil de mise à niveau PostgreSQL : mode de vérification, mode par liens physiques, retour arrière et étapes de statistiques après la mise à niveau.
- Migrer de Pages vers l'App Router — migration Next.js vers le nouveau routeur, une page à la fois.
- Migrer des travaux existants — transfert d'ArcMap vers ArcGIS Pro, conversions nécessaires et sauvegardes préalables.
- Vérifier les migrations avant la mise à niveau — réécriture GitLab de grandes tables par lots en arrière-plan et vérification de leur achèvement.

Glossaire

Choisissez selon vos contraintes réelles

Présentez un jeu de données représentatif, les questions auxquelles il doit répondre et le goulot d'étranglement actuel. Nous pouvons convenir de la manière de mesurer la qualité, le débit et la reprise.

Consulter les tarifs Contactez-nous →

Rendez la terminologie concrète

Demandez comment « Qu'est-ce qu'une migration ? » 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/migration/