Engineer

Міграція

🚚 запланований перехід на нову версію або платформу

Що це таке

Міграція — це запланований перехід чогось, що вже працює, на нову версію або нову платформу. Він відбувається кроками, а наприкінці старе виводять з експлуатації. Переносити можна ціле робоче навантаження, застосунок, масив даних або структуру, в якій ці дані зберігаються. Від перебудови з нуля міграцію відрізняє те, що система під час переходу лишається в роботі: користувачі, записи й інтеграції переходять разом із нею, а стара версія працює, доки нова не перебере все на себе.

Інженери вживають це слово для робіт двох масштабів і рідко уточнюють, про який ідеться.

У великому масштабі це перехід на нову платформу або нову мажорну версію. Microsoft Cloud Adoption Framework описує перенесення робочих навантажень в Azure — із власних серверів або з іншої хмари — як п'ять кроків: спланувати, підготувати навантаження, виконати, оптимізувати й вивести з експлуатації джерело. Клієнти Esri переносять свої карти й дані з ArcMap до ArcGIS Pro. Застосунок на Next.js переходить із Pages Router на App Router, де `getServerSideProps` поступається місцем отриманню даних усередині Server Components. Кластер PostgreSQL переходить на нову мажорну версію, яка часто змінює будову системних таблиць, тож дані доводиться переносити за допомогою `pg_upgrade` або через дамп і відновлення.

У малому масштабі міграція — це одна версіонована зміна структури бази даних, яка зберігається як файл у репозиторії й застосовується інструментом у фіксованому порядку. Django називає свої міграції системою контролю версій для схеми бази даних: кожен файл описує зміну моделей, потрапляє в коміт разом із кодом і виконується в тому самому порядку на машині

розробника, на staging і в продакшні. GitLab постачає пакетні фонові міграції, які після оновлення переписують великі таблиці порціями, наприклад перетворюють цілочисельні стовпці на `bigint`, перш ніж значення в них переповняться. Сюди ж належить міграція даних: вона змінює рядки, а не стовпці, — скажімо, заповнює нове поле на основі двох старих.

Це одна ідея у двох масштабах. В обох випадках є відомий початковий стан, упорядкована послідовність кроків, ціль і шлях назад, якщо якийсь крок піде не так. Перехід на нову платформу зазвичай містить міграції схеми, а довгий ряд міграцій схеми з роками стає платформою, що змінюється шматок за шматком під тими, хто нею користується.

Застосуйте поняття до рішення

Застосуйте «Що таке міграція?» до конкретного рішення. Ми можемо пов'язати поняття із системою, яку ви створюєте або купуєте. Перенесення даних із застарілого середовища в підтримуване, а бізнес тим часом працює далі.

Переглянути наші послуги Зв'язатися з нами →

Перевірте припущення за терміном

Використайте пояснення та пов'язані роботи, щоб сформулювати запитання для проєкту. Ми можемо перевірити припущення разом до вибору інструмента. Пришвидшила GIS-карти в 10 разів на PostgreSQL — 10x швидші карти після переходу з MSSQL

Пришвидшила GIS-карти в 10 разів на PostgreSQL Зв'язатися з нами →

Чому це важливо

Програмне забезпечення живе довше, ніж те, на чому його збудовано. Хмарні контракти переукладають, фреймворки змінюють модель, постачальники знімають продукти з підтримки. Систему, яку тримають десять років, за ці десять років мігрують кілька разів, тож уміння команди мігрувати важить більше, ніж те, наскільки вдало вона обрала першу платформу.

- **Витрати осідають у проміжному стані.** Поки частина системи працює на старій платформі, а частина на новій, обидві треба експлуатувати, моніторити й тримати узгодженими. Cloud Adoption Framework радить робити цей період коротким і переносити компоненти, що напряму залежать один від одного, в одній хвилі.
- **Базу даних не можна просто розгорнути заново.** Код можна зібрати знову й замінити; рядки в продакшні — ні. Міграції схеми перетворюють зміну структури робочої бази на файл, який можна віддати на рев'ю, перевірити на копії, застосувати в заданому порядку й здебільшого відкотити. Альтернатива — схема продакшну, яка має той вигляд, у якому її залишив останній, хто відкривав консоль.

- **Запланований перехід можна скасувати.** Cloud Adoption Framework вимагає встановити критерії та процедури відкату ще до початку будь-якої міграції, перевірити відкат на ділі й передбачити контрольні точки go/no-go на шляху.

  Компроміси зосереджені в моменті перемикання. Cloud Adoption Framework розрізняє міграцію з простоєм і міграцію майже без простою, коли дані безперервно реплікуються, а перемикання відбувається наприкінці. Для мажорної версії PostgreSQL `pg_upgrade` скорочує перемикання: він обходиться без дампу й відновлення, яких зазвичай потребує таке оновлення, повторно використовуючи наявні файли даних, а в режимі `--link`, що не копіює файлів, працює ще значно швидше.

Як це виглядає на практиці

Три проєкти на цьому сайті показують це поняття у трьох масштабах — від цілої платформи до порядку файлів в одному репозиторії.

Одна з міграцій полягала у відмові від хмарного провайдера. Конфігурація AWS у клієнта була надмірною, і робота почалася з перевірки того, що справді використовували: непотрібні сервіси прибрали, а потужність решти скоригували. Після цього навантаження перенесли з AWS. Запис про роботу з клієнтом підтверджує ці інфраструктурні зміни. Скорочення невикористаних сервісів перед перенесенням означало менше обсягу для міграції; це не підтверджує співвідношення витрат за однакових умов, результату щодо продуктивності чи міграції без порушень роботи.

Другий — зміна рушія бази даних з новим дизайном на іншому боці. База даних картографічного застосунку переїхала з MSSQL на PostgreSQL з розширенням PostGIS, і метою була не копія старої структури. Нову схему спроєктували для просторових даних, з індексами GiST і SP-GiST та суворими обмеженнями зовнішніх ключів і перевірками, що забезпечують правила валідації просторових координат. Понад 50 мільйонів просторових записів перейшли до неї через ETL-конвеєри, які привели їх до EPSG:4326. Перехід на новий дизайн — нагода записати правила даних у саму схему. Сторінка повідомляє про зростання продуктивності просторових запитів у 10 разів і про зниження навантаження на CPU бекенду на 65%.

Третій — малий масштаб, який підтримують роками. Схема однієї платформи складається з 1 022 міграцій з номерами від 036 до 1102, а дисципліну їх упорядкування сторінка називає нудною й такою, що не підлягає

обговоренню. Історія в репозиторії — це опис продакшну, і вона лишається правдивою, лише доки будь-яку базу даних — від щойно створеної на машині розробника до відновленої з резервної копії — можна зібрати заново, відтворивши цю історію від першого файлу до останнього.

Де виникають проблеми

Перемикання отримує всю увагу під час планування, а проблеми зазвичай збираються в тижні навколо нього, у чотирьох упізнаваних формах.

Дві системи працюють паралельно довше, ніж планували. Перехід зупиняється на півдорозі, лишаються найскладніші частини, і команда місяцями підтримує обидві платформи. Користувачі відчувають шов: посібник Next.js зазначає, що перехід між сторінкою на Pages Router і сторінкою на App Router — це перехід із повним перезавантаженням сторінки і що `next/link` не виконує попереднього завантаження між ними. Варто тримати хвилі малими, а кожен наступний крок — дешевим. Один фронтенд тут переводить свої сторінки на спільну фабрику, що завантажує кожну сторінку за одне звернення, тож кожна перенесена сторінка успадковує цю поведінку.

Перемикання без шляху назад. Деякі кроки змінюють не лише ціль, а й джерело. З `pg_upgrade --link` новий кластер, щойно його запущено, уже записав у файли, спільні зі старим, і тоді старий кластер доведеться відновлювати з резервної копії. Esri попереджає, що набір даних, оновлений для ArcGIS Pro, може стати непридатним для ArcMap, і радить спершу зробити копію. У Django операція `RunSQL` без зворотного SQL викликає `IrreversibleError`, щойно хтось спробує мігрувати назад. Критерії відкату варто визначити до першого кроку, шлях назад — відрепетирувати, а саму вимогу — перетворити на перевірку: один репозиторій тут має рушій захисних перевірок, валідатори якого серед іншого контролюють, що міграцію можна відкотити.

Уже застосовану міграцію редагують. Інструменти міграцій записують, які файли вже застосовано, і далі їх пропускають. Якщо змінити файл після того, як він потрапив у продакшн, продакшн зберігає стару версію, а кожна нова база даних отримує нову: дві схеми під однією історією. Застосовані міграції краще не чіпати, а виправляти новою; збірка, яка відтворює всю історію на порожній базі й порівнює результат із мігрованою копією, ловить розбіжність.

Оновлена база даних повільна в перший день. `pg_upgrade` переносить більшу частину статистики планувальника, але не всю, і щодо решти планувальник діє навмання, доки її не зібрано знову. Режим `--check` нічого не змінює й може працювати, поки старий сервер ще обслуговує запити, тож його запускають першим, а відсутню статистику відновлюють одразу після оновлення, перш ніж збирати решту:

```
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
```

Що почитати далі

- Plan your migration — методологія Microsoft для перенесення робочих навантажень в Azure: хвилі, варіанти перемикання й планування відкату.
- Migrations — посібник Django про версіоновані зміни схеми, їх відкат і міграції даних.
- pg_upgrade — інструмент оновлення PostgreSQL: режим перевірки, режим link, повернення до старого кластера й кроки зі статистикою після оновлення.
- How to migrate from Pages to the App Router — як перевести застосунок Next.js на новий маршрутизатор сторінка за сторінкою.
- Migrate existing work — що переходить з ArcMap до ArcGIS Pro, що треба конвертувати і що варто спершу скопіювати.
- Check migrations before upgrade — як пакетні фонові міграції GitLab переписують великі таблиці після оновлення і як переконатися, що вони завершилися.

Словник

Обирайте за фактичними умовами

Надайте репрезентативний набір даних, запитання до нього та поточне вузьке місце. Узгодимо вимірювання якості, пропускної здатності та відновлення.

Дивитися, скільки це коштує Зв'язатися з нами →

Зробіть термінологію практичною

Запитайте, як «Що таке міграція?» стосується вашої організації. Конкретного прикладу з поточної системи достатньо для початку розмови.

Переглянути наші послуги

welcome@engineer.company +45 50 10 61 84 Копенгаген, Данія

Візьміть цю сторінку із собою

Візитівка

Дбайливо зроблено з любов'ю — D.

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

Авторське право © 2025 – дотепер · Engineer ApS · Усі права захищено.

Надруковано з https://engineer.company/uk/glossary/migration/