JM3.1 - La pipeline passo per passo

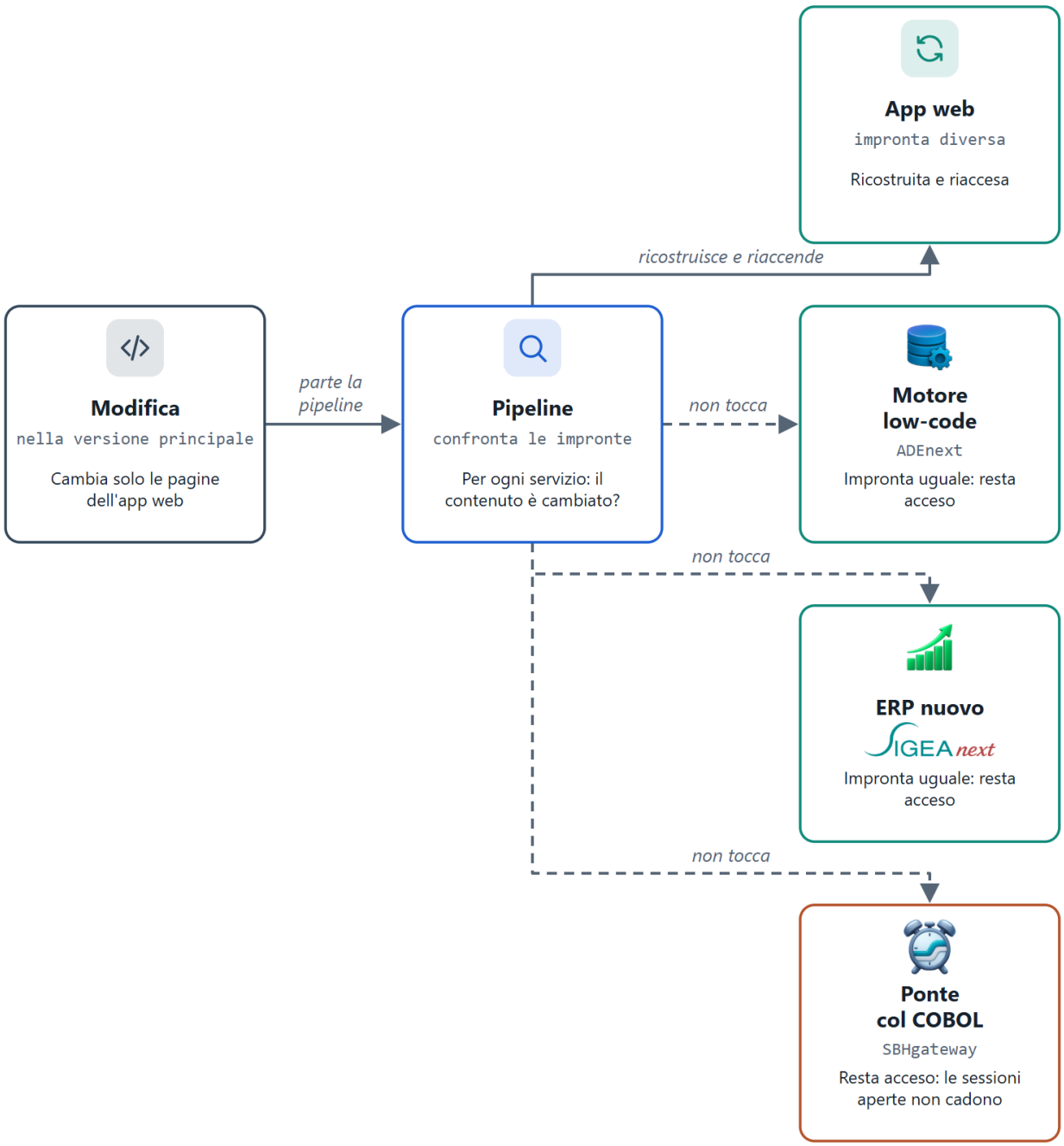
In una frase. Dopo che una modifica entra nella versione principale, la pipeline la prova, ricostruisce solo i servizi cambiati e aspetta il clic di una persona per metterli sul server di prova.

Un aggiornamento fatto a mano dipende da chi lo fa e da quanto è di fretta. La pipeline fa sempre gli stessi passi, e li fa uguali ogni volta. Il passo che cambia davvero ciò che gli utenti vedono resta in mano a una persona: lo decide lei, quando è il momento giusto.

1. **La modifica entra nella versione principale.** Chi sviluppa lavora su un ramo a parte. Quando la modifica è approvata viene unita alla versione principale, e la pipeline parte da sola. Sui rami a parte non parte niente.
2. **Test.** Vengono provate le parti toccate: se cambia ADEnext girano i suoi test, se cambia l'app web girano i controlli delle pagine. Se non cambia nessuna delle due, questo passo non parte.
3. **Build.** Per ogni servizio la pipeline calcola un'**impronta** di ciò che lo compone. Se l'impronta è uguale a quella già presente sul server, il servizio non si ricostruisce. Se è diversa, si prepara la versione nuova. Fin qui sul server non è cambiato niente: gli utenti lavorano ancora con la versione di prima.
4. **Deploy: il pulsante.** La pipeline si ferma e mostra un pulsante. Chi lo preme mette le versioni nuove al posto delle vecchie. Vengono riaccesi solo i servizi con l'impronta cambiata, o con un'impostazione cambiata. Gli altri restano accesi.
5. **I servizi ripartono.** Ogni servizio riacceso, all'avvio, porta il suo archivio alla forma che gli serve e poi si iscrive al Monitor servizi. Ci vuole circa un minuto.

Aggiornare senza ricostruire tutto

Esempio: la modifica tocca solo le pagine dell'app web. Gli altri servizi non si fermano.



Una modifica alle sole pagine: la pipeline ricostruisce l'app web e lascia accesi gli altri servizi

> **Attenzione.** Il clic va dato in un momento tranquillo se l'aggiornamento tocca il Ponte col COBOL: riaccenderlo chiude le sessioni aperte nelle maschere storiche, e il lavoro non confermato si perde. Se il Ponte non è cambiato resta acceso, e nessuno se ne accorge.

Nota. Oggi i test avvisano ma non bloccano: se sono rossi il pulsante compare lo stesso. Prima di premerlo, guarda l'esito dei test. Una schermata della pipeline non è ancora in questa raccolta.

Per chi sviluppa.
• File: `.gitlab-ci.yml`. Stadi: `test`, `helper`, `build`, `deploy`, `reconcile`. Il lavoro di build è `build:images`, quello di deploy è `deploy:simil-prod` (`when: manual`).
• L'impronta è un tag di contenuto (`ADE_NEXT_TAG`, `GPE_NEXT_TAG`, `FRONTEND_TAG`…): l'hash degli alberi git di ciò che entra nell'immagine. Stesso contenuto, stesso tag: niente build, e il deploy non ricrea il contenitore. Dettagli in `devops/deploy/README.md`, paragrafo 3.3.
• I test hanno ancora `allow_failure: true`: il file spiega perché e quando va tolto.
• `reconcile` parte subito dopo l'unione, senza aspettare il clic: dice ad ADEnext quale codice generato è nella versione principale.
• Prima di toccare i servizi, il deploy controlla che nel file delle impostazioni ci siano le password dell'Assistente dei dati: se mancano si ferma e dice quale.

Revisione #2
Creato 17 novembre 2026 07:59:38 da Giuseppe Anelli
Aggiornato 17 novembre 2026 09:39:01 da Genesys Software