

# MIGRAZIONE INCREMENTALE CON STRANGLER FIG PATTERN SU DRUPAL

Come disaccoppiare  
il tuo portale  
una sezione alla volta,  
senza fermare  
il business.

# IN BREVE

Se il tuo portale Drupal è diventato lento da evolvere, la tentazione è rifarlo da zero. Te lo sconsigliamo.

Una riscrittura integrale richiede mesi prima di produrre valore e concentra tutto il rischio in una sola notte di rilascio. Lo Strangler Fig Pattern fa il contrario. Un nuovo frontend Next.js affianca Drupal sullo stesso dominio e ne sostituisce il tema una sezione alla volta.

Ogni sezione va in produzione appena è pronta. Se qualcosa non ti convince, torna su Drupal con una modifica di configurazione, senza un nuovo rilascio del codice.

Drupal resta dov'è: unica fonte dei contenuti e strumento quotidiano dei redattori, che continuano a lavorare come oggi.

## DUE MODI DI LEGGERE QUESTO DOCUMENTO

**Se decidi:** sezioni 01–04, 08 e 10–13. Circa dieci minuti.

**Se progetti o sviluppi:** anche le sezioni 05–07, 09. Puoi girarle al tuo team tecnico insieme all'autovalutazione.

**Versione web accessibile:** <https://bmeme.com/it/insight/strangler-fig-pattern-migrazione-drupal>

## COSA OTTIENI

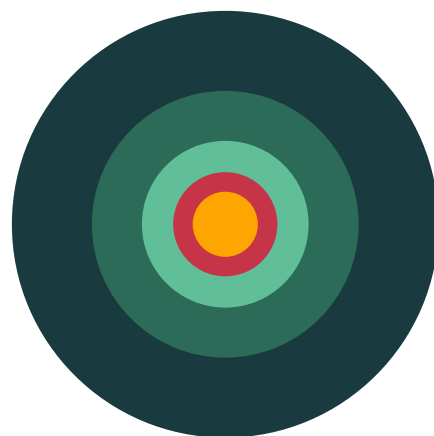
- Il sito continua a lavorare per tutta la durata della migrazione.
- Vedi valore a ogni sezione rilasciata, non dopo un anno.
- Ogni sezione è reversibile, singolarmente.
- Gli URL, i canonical e i redirect restano gli stessi, la parità tecnica è una condizione per estendere la migrazione a nuove sezioni.
- Nessun lock-in: il metodo e i moduli Drupal che usiamo sono open source e il codice dell'adattatore resta tuo.



# TI RICONOSCI?

Disaccoppiare ha senso quando il frontend è diventato il freno del tuo business.

- Ogni nuova iniziativa, che sia una landing, un restyling o una sezione, richiede settimane perché tocca il tema Drupal.
- I rilasci fanno paura: una modifica in un punto rompe qualcosa altrove.
- Il redesign è fermo da tempo, perché 'tanto andrebbe rifatto tutto'.
- TTFB e LCP penalizzano l'esperienza, migliorarli senza toccare il monolite sembra impossibile.
- Il tuo team frontend vorrebbe lavorare con React, ma il tema Drupal non lo permette.
- Il sito è cresciuto negli anni con mani diverse e nessuno sa più con certezza cosa succede se si tocca qualcosa.



Se ti sei riconosciuto in almeno due punti, continua a leggere. Se in nessuno, probabilmente oggi non ti serve un disaccoppiamento. Va bene così.

# PERCHÉ RISCRIVERE TUTTO È LA SCOMMESSA SBAGLIATA

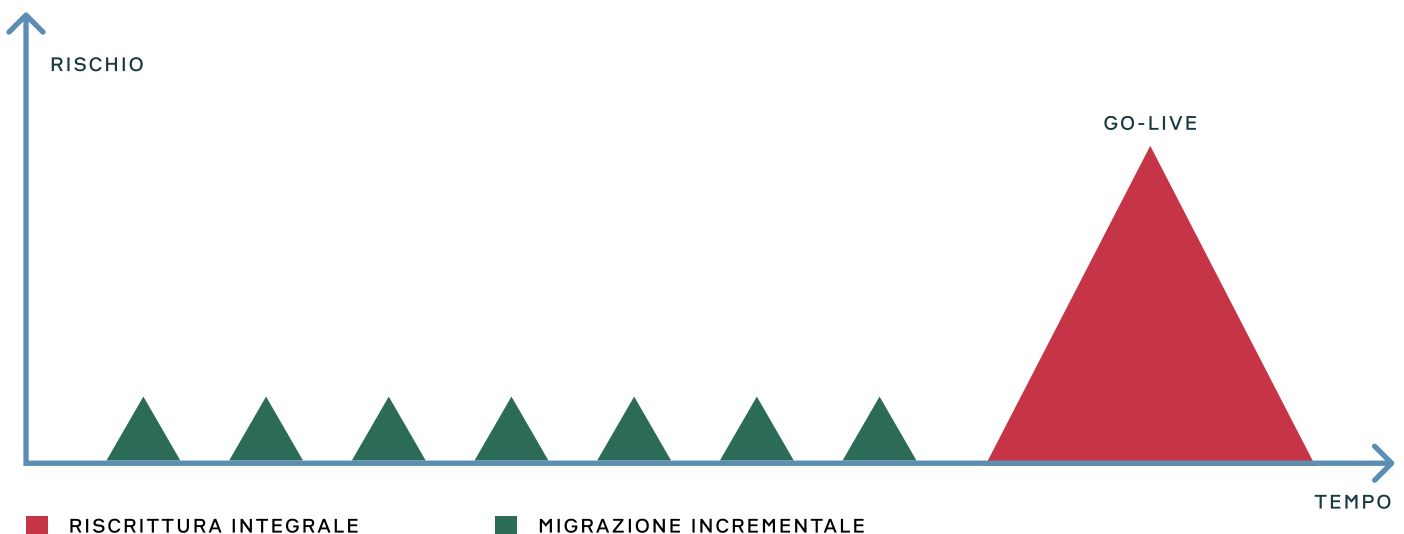
Sostituire il vecchio sito in una notte concentra tutto il rischio nel momento peggiore.

Una riscrittura integrale richiede mesi prima di produrre valore. Nel frattempo il business aspetta, i requisiti cambiano e il nuovo sistema insegue un bersaglio che si sposta.

Con una migrazione incrementale il rischio non sparisce. Si divide in tanti piccoli rilasci, ciascuno reversibile per conto suo.

Quando arriva il go-live, tutto il rischio accumulato si scarica insieme: dati, funzionalità, integrazioni, indicizzazione. Se qualcosa va storto, tornare indietro significa rinunciare a mesi di lavoro.

## IL RISCHIO NEL TEMPO



# L'ALTERNATIVA: DUE SISTEMI, UN SOLO SITO

Drupal e il nuovo frontend convivono sotto lo stesso dominio, chi naviga non si accorge del passaggio.

Durante la transizione una parte delle pagine arriva da Drupal, come oggi, un'altra da Next.js. Le sezioni passano al nuovo frontend una alla volta, quando sono pronte. Gli URL restano gli stessi.

In questa fase Drupal ha due ruoli: serve le pagine non ancora migrate e fornisce al nuovo frontend le pagine migrate, contenuti compresi.

## COSA CAMBIA PER IL BUSINESS

- Parti da una sola sezione e decidi il resto quando hai visto i risultati. Non serve approvare un progetto annuale per iniziare.
- Puoi fermarti quando vuoi: le sezioni migrate funzionano, le altre restano su Drupal.
- Il sito continua a portarti traffico e contatti mentre lavoriamo.

## COSA NON CAMBIA PER I REDATTORI

Continuano a usare Drupal come oggi, con lo stesso pannello, gli stessi flussi di pubblicazione e gli stessi permessi. Le regole su cosa appare in una pagina restano configurate in Drupal e valgono sia per le pagine vecchie sia per quelle nuove.

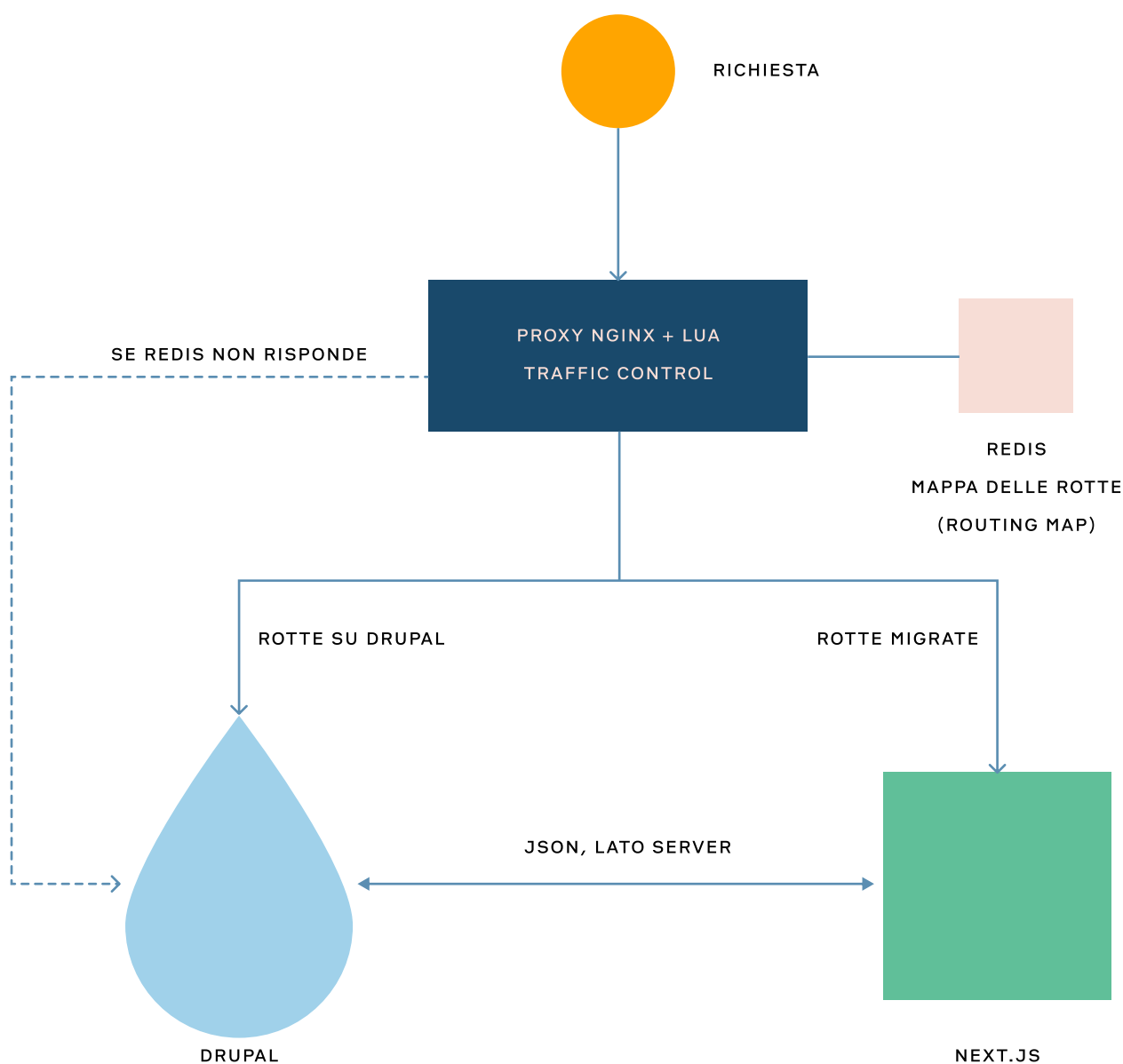
## COSA SUCCEDE PER I MOTORI DI RICERCA

Stessi URL, stessi canonical, sitemap sempre generata da Drupal, redirect senza catene. Garantiamo la parità tecnica tra pagine vecchie e nuove. Non promettiamo posizionamenti: ci assicuriamo di non rompere ciò che oggi funziona.

# COME FUNZIONA

Drupal e Next.js si dividono le pagine senza che il visitatore se ne accorga. Se il sistema che smista il traffico ha un problema, tutto torna su Drupal.

## CHI RISPONDE A UNA RICHIESTA



# LE QUATTRO SCELTE PROGETTUALI

## 01

**Drupal resta l'unica fonte di verità.** Contenuti, tassonomie, regole di visibilità e configurazione dei layout stanno solo in Drupal. Il frontend non duplica dati e non ricostruisce regole: riceve pagine già risolte e le disegna.

## 02

**Next.js lavora stateless.** Il nuovo frontend genera le pagine lato server (SSR con App Router) e non gestisce sessioni dei visitatori. Il browser non parla mai direttamente con Drupal: ogni richiesta passa dal server Next.js, secondo il modello Backend For Frontend. Le API di Drupal usate dal sito restano fuori dalla portata del pubblico.

## 03

**Il traffico si smista all'edge.** Un proxy Nginx esteso con Lua decide chi risponde a ogni richiesta, leggendo una mappa delle rotte su un'istanza Redis dedicata. La decisione non passa da Drupal, quindi non pesa sul TTFB. Se Redis non risponde, la richiesta va a Drupal: nel dubbio, il sito funziona come prima.

## 04

**Stesso URL, due formati.** Lo stesso indirizzo restituisce la stessa pagina: in HTML al browser, in JSON al frontend, in base all'header Accept. Page Delivery non aggiunge API parallele con percorsi propri.

### PER I TEAM TECNICI

La mappa in Redis è una proiezione ricostruibile da Drupal, non una fonte autorevole. Un errore di Next.js non ricade automaticamente su Drupal: chi serve una rotta è una decisione esplicita, questo evita comportamenti incoerenti verso i crawler. La stessa pagina si chiede in due modi: il browser con **Accept: text/html**, il server Next.js con **Accept: application/vnd.drupal.page-composition+json**.

# IL CONTRATTO E LE OTTO VERIFICHE

Nessuna sezione cambia sistema per sbaglio. Ogni passaggio è verificato, approvato da una persona e reversibile.

## IL CONTRATTO

Drupal restituisce la pagina intera in un JSON diviso in cinque regioni: beforeMain, navigation, main, aside, afterMain. Ogni regione elenca in ordine i componenti, con i loro contenuti. Il frontend non riceve markup HTML, ma la pagina già risolta, che disegna con componenti React conosciuti e validati.

Se arriva un componente sconosciuto o un dato non conforme allo schema, la pagina non viene pubblicata a metà. L'errore è esplicito e visibile nei log.

## LE OTTO VERIFICHE

Se ne manca una, la sezione resta su Drupal: non è un fallimento, è ciò che rende sicura la migrazione.

- 1 Risoluzione univoca.** Drupal identifica senza ambiguità rotta, lingua, entità, tema e famiglia di pagine.
- 2 Struttura pronta.** La variante di layout richiesta esiste nel nuovo frontend.

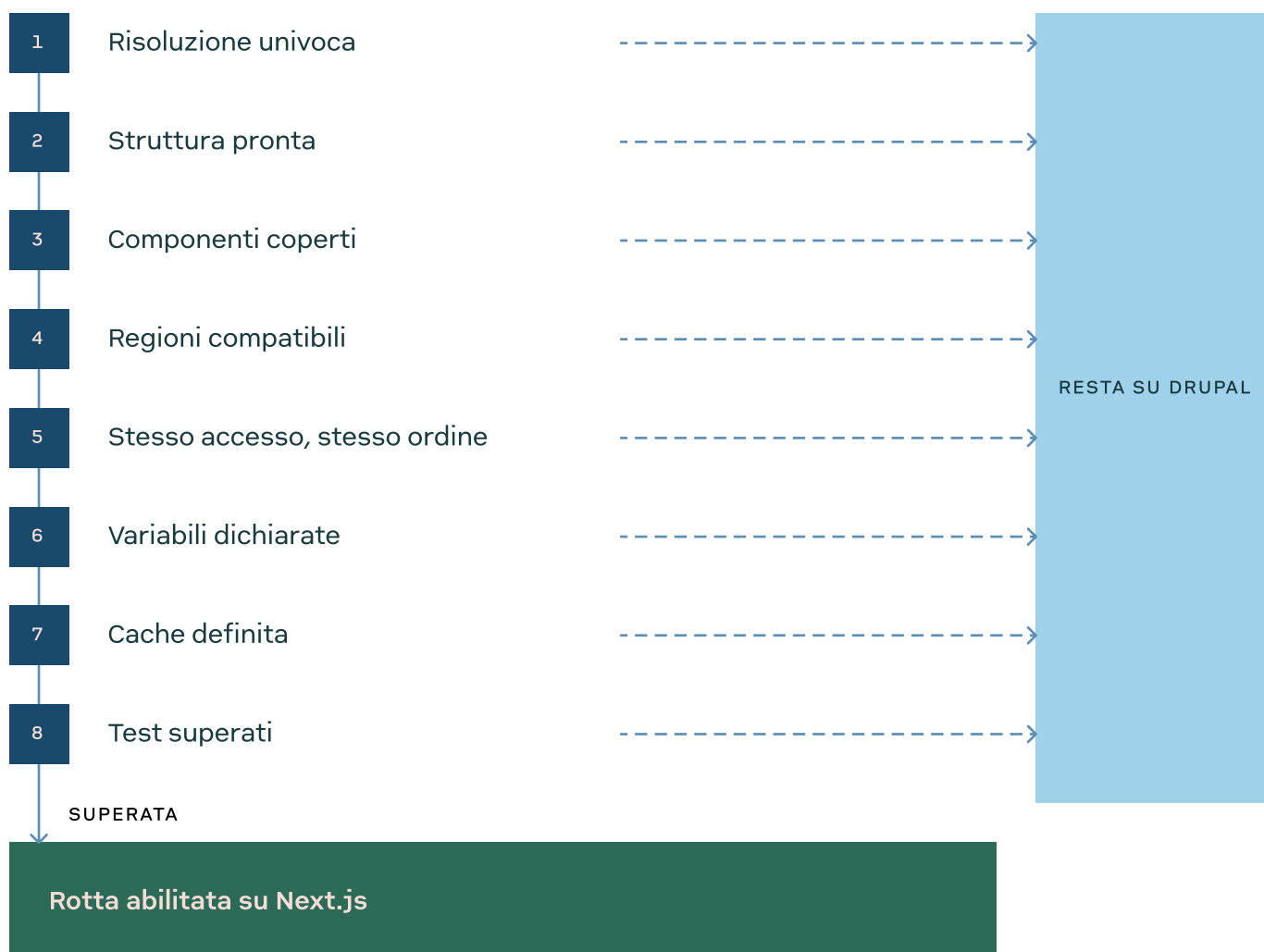
- 3 Componenti coperti.** Ogni blocco che può comparire nella pagina ha il suo componente React validato.
- 4 Regioni compatibili.** Le regioni prodotte da Drupal sono supportate dalla variante di destinazione.
- 5 Stesso accesso, stesso ordine.** Visibilità, ordinamento e regole contestuali coincidono con la pagina attuale.
- 6 Variabili dichiarate.** Parametri e varianti linguistiche che cambiano la pagina sono elencati esplicitamente.
- 7 Cache definita.** Sono stabiliti i cache tag e le regole di aggiornamento della rotta.
- 8 Test superati.** Test funzionali e visivi confrontano la pagina attuale con quella nuova su rotte campione.

## COME SI APPLICANO

Le verifiche non avvengono a ogni richiesta: sono una procedura di approvazione. Per ogni famiglia di pagine si sceglie una rotta rappresentativa, si documenta ogni criterio con le sue evidenze e si eseguono i test di confronto. Poi un responsabile tecnico approva il passaggio.

Solo a quel punto la rotta viene assegnata a Next.js nella mappa di routing. Tornare indietro è la stessa operazione al contrario.

### PROCEDURA DI APPROVAZIONE DI UNA ROTTA



# CACHE, RILASCI E SEO TECNICA

I contenuti pubblicati compaiono subito e i rilasci non si allungano quando il sito cresce. Se un rilascio peggiora le prestazioni, si torna indietro in automatico.

## AGGIORNAMENTI GUIDATI DAGLI EVENTI

Quando un redattore pubblica o modifica un contenuto, Drupal invia al frontend una notifica firmata con HMAC, con marca temporale e identificativo univoco. Il frontend la verifica e segna come obsoleti solo i dati interessati. Il visitatore successivo riceve subito la versione in cache, mentre quella aggiornata si rigenera in background (stale-while-revalidate).

Gli spostamenti di rotta seguono l'Outbox Pattern: la modifica viene registrata nella stessa transazione del database, un processo separato la propaga a Redis. Nessun cambio si perde a metà strada.

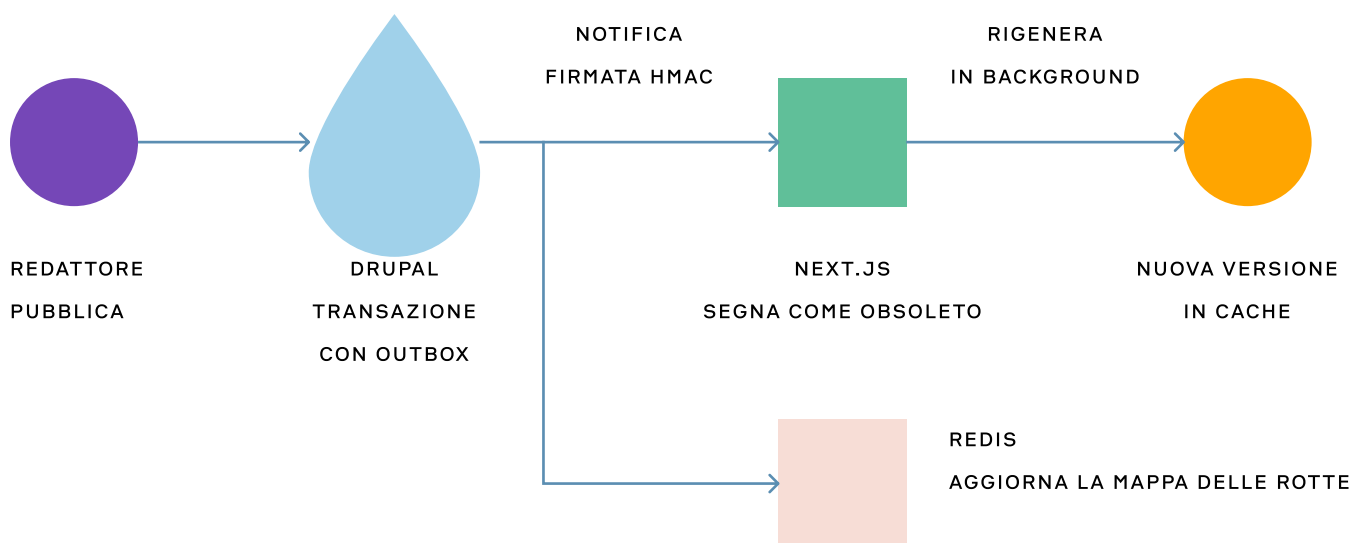
## SE DRUPAL NON RISPONDE

Un circuit breaker smette di insistere su un backend in difficoltà. Per ogni funzionalità si decide in anticipo cosa mostrare: l'ultima versione in cache, una versione senza parti dinamiche o un messaggio esplicito di non disponibilità.

## BUILD A TEMPO COSTANTE

La build di Next.js non chiama Drupal (Zero Build-Time Fetching). Le pagine si generano al primo accesso e poi restano in cache. Che il portale abbia mille pagine o centomila, la pipeline dura uguale.

## COSA SUCCEDDE QUANDO UN REDATTORE PUBBLICA



## RILASCI BLUE/GREEN

Il nuovo rilascio va su un ambiente gemello, si verifica, poi il bilanciatore sposta il traffico. Se TTFB, LCP o tasso di errore superano le soglie, il traffico torna automaticamente all'ambiente precedente.

## SEO TECNICA

Canonical coerenti, redirect 301/308 senza catene, sitemap generata solo da Drupal, stessi hreflang e dati strutturati. La parità su questi punti è una condizione per estendere la migrazione a nuove sezioni.

## PER I TEAM TECNICI

I cache tag si dichiarano prima della fetch, a partire da identificativi noti. L'header X-Drupal-Cache-Tags non è un'alternativa: è un header diagnostico e in produzione resta disattivato. Su pagine complesse elenca centinaia di tag, appesantisce ogni risposta e può superare i limiti di dimensione degli header di proxy e CDN. In ogni caso arriverebbe tardi: quando si legge la risposta, la fetch è già in cache senza tag.

# COSA IL METODO NON RISOLVE DA SOLO

Lo Strangler Fig riduce il rischio, ma non lo azzerava.  
Preferiamo dirti subito dove serve attenzione.

- **La coesistenza ha un costo.** Per un periodo mantieni due stack, ciò significa avere competenze su Drupal e su React. Per questo la migrazione si pianifica a milestone, con una fine definita. Se ti fermi prima, le sezioni migrate restano su Next.js e le altre su Drupal.
- **Non tutto si migra subito.** Le sezioni più complesse o stratificate restano su Drupal finché i loro componenti non sono mappati. È una scelta deliberata.
- **Il Blue/Green ha una finestra delicata.** Se i redattori modificano contenuti mentre si prepara un rilascio, può crearsi un breve disallineamento. Si gestisce con una riconciliazione prima dello switch o con un blocco temporaneo delle scritture.
- **Le aree riservate richiedono un discorso a parte.** Il frontend pubblico non gestisce login dei visitatori: le aree che richiedono autenticazione restano, almeno all'inizio, su Drupal.
- **Il punto di partenza conta.** Se il Drupal esistente ha logiche di presentazione sparse nel tema, moduli custom fragili o versioni obsolete, prima si sistemano quelli. Per questo partiamo sempre da un assessment.

## SEI ANCORA SU DRUPAL 7?

Lo Strangler Fig presuppone un Drupal moderno che faccia da backend per il nuovo frontend. Drupal 7 è fuori supporto da gennaio 2025 e non può ospitare i moduli Page Delivery.

Nel tuo caso serve una reingegnerizzazione, l'assessment di aggiornamento serve a progettare la nuova piattaforma già pronta per il disaccoppiamento.

[Parliamone ↗](#)

# QUANDO CONVIENE E QUANDO NO

Page Delivery risolve un problema preciso.  
Se il tuo è un altro, preferiamo dirtelo prima.

## CONVIENE

Se hai un Drupal moderno, con contenuti strutturati, il cui freno è il tema; se vuoi un frontend Next.js con team e rilasci propri, senza migrare i dati; se i redattori devono continuare a comporre le pagine in Drupal.

## NON CONVIENE

Se il modello dei contenuti va ripensato comunque, se deve essere il frontend a decidere come si compone una pagina, se sei su Drupal 7, se il tema non è un freno o se il sito è così piccolo che due sistemi costano più di quanto rendono.

Un'app o altri canali da alimentare non sono un ostacolo:  
JSON:API e GraphQL convivono con Page Delivery.

# PAGE DELIVERY: I MODULI SONO OPEN SOURCE

Gli strumenti sono gratuiti e pubblici. Paghi il lavoro su misura sul tuo portale, non una licenza.

Page Delivery è una suite di moduli Drupal per siti decoupled, raccolti sotto un'unica sezione di amministrazione. Come tutto il codice su drupal.org, è rilasciata con licenza GPL e convive con le API già presenti in Drupal, JSON:API e GraphQL comprese.

## MODULO

## COSA FA

### Page Delivery Layout

Definisce layout logici e mappa le regioni del tema Drupal sulle regioni del contratto.

### Page Delivery Composition

Restituisce la pagina intera in JSON, contenuti compresi, sullo stesso URL della versione HTML, in base all'header Accept.

### Page Delivery Route Ownership

Stabilisce quali path passano a Next.js, per match esatto o per prefisso, pubblica la mappa di routing su Redis. Drupal resta il default.

### Page Delivery Authentication

Protegge, se attivato, il JSON delle pagine. La chiamata parte dal server Next.js, che si autentica verso Drupal con uno dei provider di autenticazione già presenti. Così il JSON delle pagine è leggibile solo dal frontend.

## COSA È GRATUITO E COSA È SU MISURA

I moduli Page Delivery sono open source e disponibili su [drupal.org](https://drupal.org) a partire dal 22 ottobre, in occasione del DrupalCamp Italy 2026. Sono generici: ogni portale però ha sorgenti proprie, come view, blocchi, menu e archivi. Serve un adattatore di progetto che le traduca nel contratto di Page Composition.

Parliamo del tuo portale ↗



# IL TUO PORTALE È PRONTO? OTTO DOMANDE

Rispondi con sì, no o non so.

Poi porta le risposte alla call: partiamo da lì.

|                                                                                                                  | SÌ                       | NO                       | NON SO                   |
|------------------------------------------------------------------------------------------------------------------|--------------------------|--------------------------|--------------------------|
| 1 Il tuo Drupal è su una versione supportata e riceve aggiornamenti regolari?                                    | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| 2 Sai quali moduli contrib e custom sono installati e quali servono davvero?                                     | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| 3 La logica su cosa mostrare in una pagina sta nella configurazione di Drupal, o è sparsa nei template del tema? | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| 4 I contenuti sono strutturati in campi, o molte pagine sono blocchi di HTML scritto a mano?                     | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| 5 Blocchi e layout seguono poche varianti ricorrenti, o ogni sezione ha una struttura a sé?                      | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| 6 Le regole di visibilità per ruolo, lingua e contesto sono documentate?                                         | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| 7 Hai un ambiente di test dove confrontare le pagine attuali con quelle nuove?                                   | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |
| 8 Chi mantiene il sito oggi saprebbe spiegarti perché è fatto così?                                              | <input type="checkbox"/> | <input type="checkbox"/> | <input type="checkbox"/> |

Tanti sì: puoi arrivare presto a un pilota. Tanti no o non so: è normale, soprattutto per siti cresciuti negli anni con mani diverse. È proprio ciò che l'assessment serve a chiarire.

# COME LAVORIAMO

Partiamo sempre da un assessment.  
Non interveniamo su un Drupal esistente senza averlo prima analizzato.

Molti portali Drupal sono cresciuti negli anni con team diversi, collaboratori esterni e scelte fatte di corsa. Lavorare su un codice che non conosci significa scoprire i problemi a progetto avviato, quando costano di più. L'assessment li porta allo scoperto prima e ti dà una base concreta per decidere.

Il documento dell'assessment resta tuo, anche se poi decidi di non proseguire con noi.

## LE QUATTRO TAPPE



## 01

**Call conoscitiva • gratuita, 45 minuti.** Parliamo del tuo portale, di cosa ti blocca oggi e dei tuoi obiettivi. Esci con una prima idea di quanto il sito sia pronto e di cosa va verificato.

## 02

**Assessment • a pagamento, 2 settimane.** È analisi e documentazione: non tocchiamo il codice. Analizziamo l'intero Drupal: versione del core, moduli contrib e loro stato di manutenzione, moduli custom, tema, struttura dei contenuti, blocchi e layout, regole di visibilità, integrazioni. Ti consegniamo:

- l'elenco dei bloccanti, con le azioni per risolverli;
- una proposta di pilota calibrata sul tuo budget, con la sua stima;
- un semaforo che dice se e come si può partire.

### IL SEMAFORO DELL'ASSESSMENT

- **Verde • Codice conforme.** Architettura solida, database Drupal standard, nessun codice critico. Si parte con tempi e costi previsti.
- **Giallo • Debito tecnico.** Alcune parti non standard vanno bonificate prima di partire. La bonifica si stima a parte.
- **Rosso • Incompatibilità.** Sistema instabile o non aggiornabile, dati alterati da sistemi esterni o codice non documentato. Ci fermiamo e valutiamo un rifacimento.

## 03

**Pilota.** Si apre con la fase operativa dell'assessment, direttamente sul codice. Poi una famiglia di pagine passa in produzione su Next.js: la scegliamo con te in base al budget, può essere anche un insieme di rotte più semplici. Vedi un risultato reale e decidi con i dati alla mano.

## 04

**Estensione a milestone.** Una famiglia di pagine alla volta, al ritmo del tuo business. Le prime milestone sono le più impegnative, perché si costruisce CI/CD, componenti, adattatore e procedure. Dalle successive si riusa quello che c'è già così il costo di ogni nuova sezione scende.

## TEMPI E COSTI

La prima famiglia di pagine arriva in produzione in genere in 2–3 mesi. E ogni sezione migrata rende più semplice e meno costosa la successiva. L'ordine di grandezza dell'investimento te lo diamo già nella call conoscitiva, la stima precisa con l'assessment.

# CHI SIAMO

bmeme è una Digital Tech Boutique di dodici persone. Progettiamo e sviluppiamo architetture Drupal e web, dall'analisi dei requisiti alla manutenzione.

Nessun lock-in: il metodo e i moduli Drupal che usiamo sono open source e il codice dell'adattatore resta tuo.

## VERIFICALO TU

- **bmeme.com**: anche il nostro sito è decoupled, con frontend Next.js, lo stesso che usiamo con Drupal.
- Il nostro profilo su **drupal.org**: nella community Drupal italiana dal 2010, con contributi a Drupal core e crediti su due security advisory.

### DA DOVE NASCE IL METODO

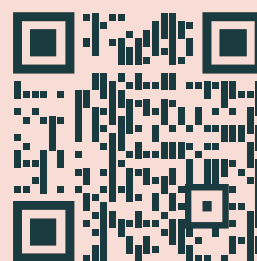
Nessun lock-in: il metodo e i moduli Drupal che usiamo sono open source e il codice dell'adattatore resta tuo.

# PARLIAMO DEL TUO PORTALE

Una call conoscitiva gratuita con chi progetta, non  
con un commerciale.

Porta le risposte all'autovalutazione e i dubbi che hai.  
In 45 minuti capiamo da dove conviene partire,  
e se conviene partire.

[Prenota la call ↗](#)



## IL METODO AL DRUPALCAMP ITALY 2026

Talk: «Evoluzione Enterprise senza "Big Bang": lo Strangler Fig Pattern con Drupal», Adriano Cori, 22 ottobre 2026, Bologna. Dalla stessa data i moduli Page Delivery sono open source e disponibili su drupal.org.

[Il talk al DrupalCamp ↗](#)



# DIGITAL TECH BOUTIQUE



[bmeme.com](https://bmeme.com)



Bonsaimeme S.r.l.  
P.IVA 10584051006

Via del Porto Fluviale, 35  
00154 Roma (RM)

+39 06 9028 3733  
[info@bmeme.com](mailto:info@bmeme.com)