



Platform Engineering: Impulsando eficiencia en equipos de desarrollo a través de las Plataformas de Desarrollo Interno (IDP)

Alejandra Bricio

Prácticas modernas para crear software con calidad y sabor
[#SGVirtual](#)

A group of people are working at laptops in a modern office setting. The scene is captured from a slightly elevated, behind-the-scenes perspective. In the foreground, a man wearing a headset is focused on his laptop. To his right, another person is also working on a laptop. In the background, a woman is seen talking on a headset, and another person is looking at a laptop. The office has a warm, wooden aesthetic with a large window in the background. The overall atmosphere is one of collaborative work and productivity.

Platform Engineering

Transformando la eficiencia de los equipos de desarrollo



Alejandra Bricio



cloud, scripting, k8s
@alebricio

Agenda

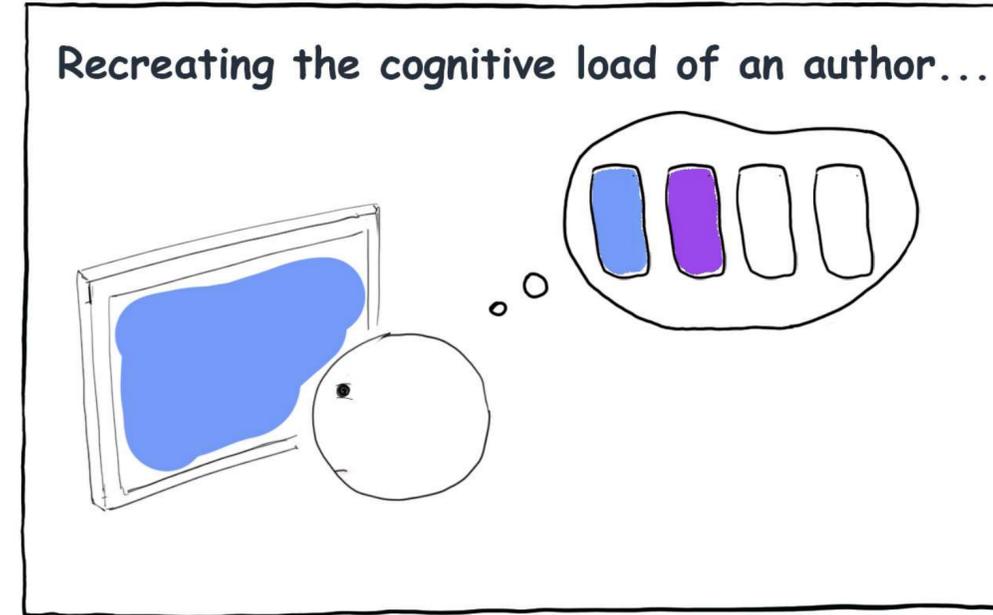
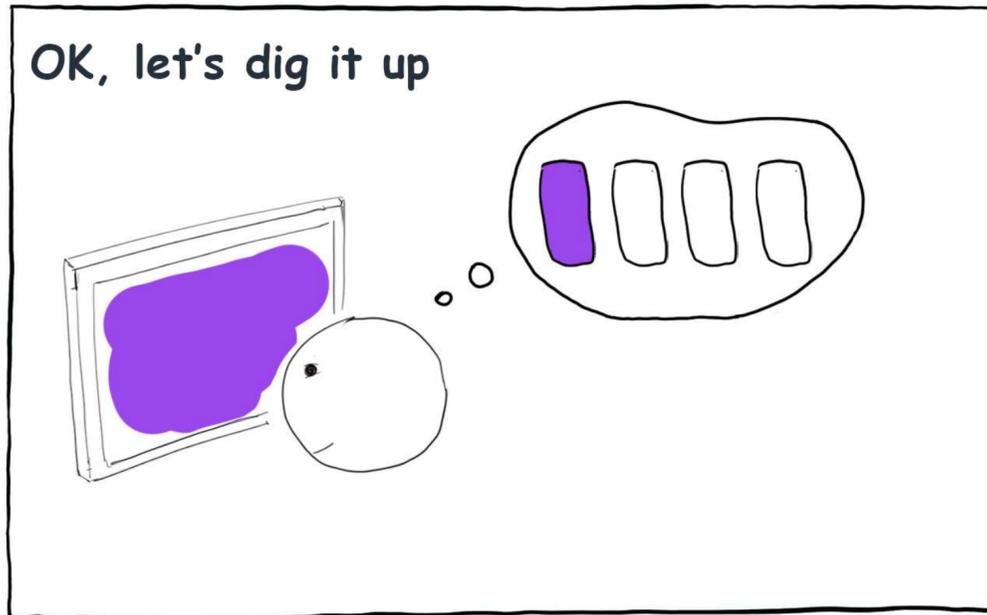
- 1. Carga Cognitiva**
- 2. Platform as a Product**
- 3. Plataformas de Desarrollo Interno (IDPs)**
- 4. Adopción**

“Paso **más tiempo** intentando entender cómo hacer testing, build y deploy de una aplicación **que en el desarrollo real** del código que quiero añadir.”

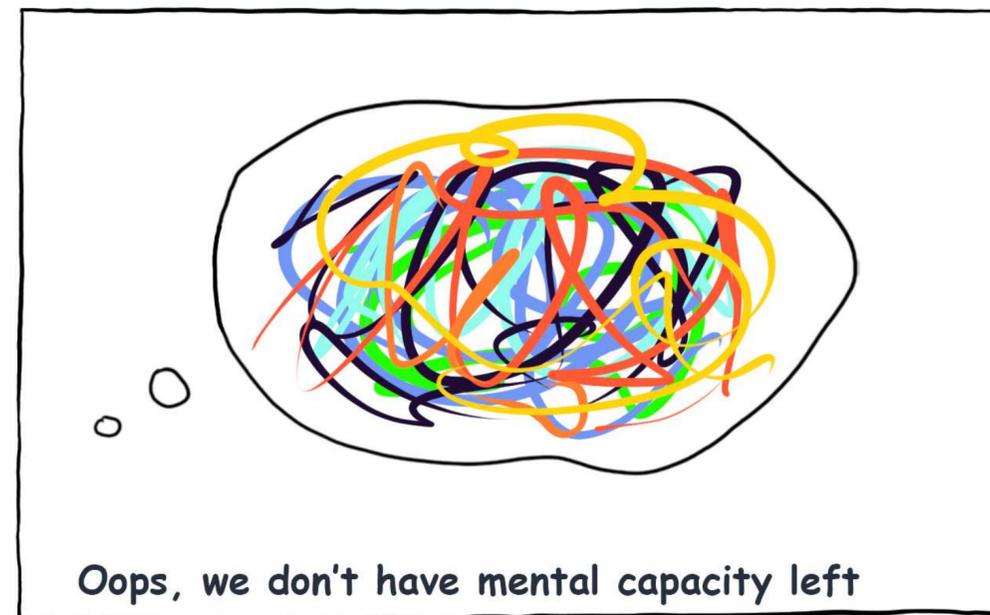
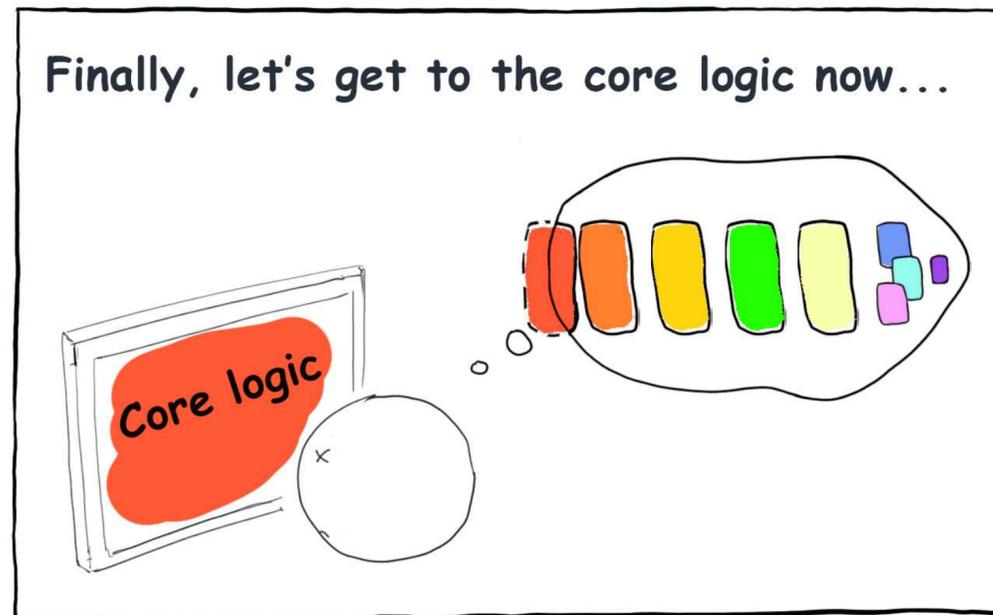
- **Developer Anónimo**

iPlatform Engineering al rescate!





4 HOURS LATER



Carga Cognitiva Extrínseca



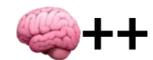
¿Dónde puedo encontrar la documentación completa sobre este sistema?



¿Hay algún estándar o plantilla que deba seguir para mi aplicación?



¿Hay algún repositorio específico o rama en la que deba trabajar?



¿Existe una suite de pruebas automatizadas o algún framework de pruebas que deba usar?



¿Cuál es el proceso para desplegar cambios en producción?

Carga Cognitiva

Cómo defino las clases en Python.



Carga Intrínseca

Resolver problemas de configuración que no están directamente relacionados con la tarea principal de consumir mensajes.



Carga Extrínseca

Conceptualizar una arquitectura que sea escalable y pueda atender futuras expansiones o cambios en el proyecto.



Carga Germana

Al minimizar la Carga Intrínseca y eliminar la Extrínseca,
podemos concentrarnos en la Carga Germana.

Reduce el burn out

**Al minimizar la Carga Intrínseca y eliminar la Extrínseca,
podemos concentrarnos en la Carga Germana.**

Incrementa la motivación

**Al minimizar la Carga Intrínseca y eliminar la Extrínseca,
podemos concentrarnos en la Carga Germana.**

Fomenta el flujo de desarrollo

Al minimizar la Carga Intrínseca y eliminar la Extrínseca,
podemos concentrarnos en la Carga Germana.

Trae valor al negocio 💰

“El propósito de un Platform Team es permitir a los equipos de desarrollo entregar el trabajo con **autonomía**.

[Éste] proporciona **servicios internos** para **reducir la carga cognitiva** que se requeriría de los equipos para desarrollar estos **servicios.**”

- **Team Topologies, Matthew Skelton and Manuel Pais**

Componentes de Platform Engineering

**Configuración
de Aplicación**

**Orquestación
de
Infraestructura**

**Gestión de
Ambientes de
Desarrollo**

**Gestión de
Despliegue**

**Control de
Acceso Basado
en Roles**

Componentes de Platform Engineering



InfraOps?

Configuración de Aplicación

Orquestación de Infraestructura

Gestión de Ambientes de Desarrollo

DevOps?

Gestión de Despliegue

Control de Acceso Basado en Roles

SysAdmin?

“Platform Engineering no es más que
DevOps con una **mentalidad de
producto**”

- Luca Galante

Colaboración Vs. Consumir servicios

“Cierta grado de colaboración entre los equipos es esperada, pero a menudo ésta **no escala a toda la organización.**
Consumir cosas-*como-servicio* suele ser **más eficaz** a medida que el número crece.”

- Team Topologies, Matthew Skelton and Manuel Pais

Internal Development Platforms (IDP)

The screenshot shows a dashboard for user **Buongiorno, Kat**. At the top right, there are time zones: NYC (08:30 AM), UTC (12:30 PM), LON (12:30 PM), and STO (01:30). Below the header, there is a search bar with the query: "Answer me: Does 'SPTDebugLog' emit logs on iOS test runs?".

The main content area is divided into two columns:

- Favorites:** A list of five items, each with a star icon on the right:
 - Manage
 - stack_overflow.questions - Metrics
 - tpm-contacts.BasicRnDEmployeeData.bq - Overview
 - Explore the Spotify ecosystem - Platforms
 - sciencebox-cloud-notebook-creator - Code Coverage
- Recently Visited:** A list of seven items:
 - stack_overflow.questions
 - sciencebox-cloud-notebook-creator
 - tpm-contacts.BasicRnDEmployeeData.bq
 - backstage-data.DocPageProfiles
 - backstage-frontend
 - backstage-backend
 - events.SearchAPIExposureEvent.gcs

Below these columns are three large action buttons:

- Explore the Ecosystem:** Discover data, apps, platforms and more
- Manage & Maintain:** Take care of the things you own
- Docs:** Find out how Spotify Tech works

At the bottom, there are navigation links for HUB (Data Home, Apps) and TOOL (Tech Insights). Below this is a "News and Updates" section with three news items, and a "Cycle Time" chart showing a line graph with a peak and a trough.

Service Catalog

The screenshot shows the Service Catalog web application. The header is green with the title 'Service Catalog' and the tagline 'Keep track of your software'. On the right, it displays time zones: NYC (2:00 PM), UTC (18:00), LON (7:00 PM), and STO (8:00 PM). A navigation bar includes 'SERVICES', 'WEBSITES', 'LIBRARIES', 'DATA ENDPOINTS', 'APP FEATURES', and 'DASHBOARDS'. The main content area is titled 'Services' and features a 'CREATE SERVICE' button and a 'SUPPORT' link. A sidebar on the left contains filters for 'PERSONAL' (Owned: 32, Starred: 2) and 'SPOTIFY' (All Services: 824). Below this is a 'Refine Results' section with a 'CLEAR' button and a list of areas: Python, Go, Java, Data, and Websites. The main table displays a list of services under the heading 'Owned (32)'. The table has columns for NAME, SYSTEM, OWNER, LIFECYCLE, STATUS, DESCRIPTION, TAG, and ACTIONS. The first few rows are: 'podcast-api' (podcast, Tools, Production, Up and running, GraphQL backend for Playlist, Websites), 'artist-lookup' (artist, Tools, Production, Up and running, GraphQL backend for Playlist, Websites), 'searcher' (search, Tools, Production, Up and running, GraphQL backend for Playlist, Python), and 'playback-order' (playback, Tools, Production, Up and running, GraphQL backend for Playlist, Data). A callout box at the bottom right contains the text: 'Gestiona todos tus servicios y componentes de software, en un mismo lugar.'

Service Catalog

The screenshot shows the 'playlist-proxy' service page in a Service Catalog. The page has a purple header with the service name and navigation links. Below the header is a navigation bar with tabs for OVERVIEW, CI/CD, TESTS, API, MONITORING, and QUALITY. The main content area is divided into several sections: README, Pull requests, Information, Activity, and Monitoring. The README section includes installation instructions and a code block. The Pull requests section shows a table of recent pull requests with their status. The Information section provides details about the owner and support. The Activity section shows recent news and product updates. The Monitoring section features a line graph showing performance metrics over time.

Service Catalog > Services > playlist-proxy

playlist-proxy

Owner tools | Service tier Tier 1 | Lifecycle production

OVERVIEW | CI/CD | TESTS | API | MONITORING | QUALITY

Overview ☆

Graphql backend for Playlist [SUPPORT](#)

README

[Getting started](#) [About](#)

To run playlist-proxy, you will need:

- git
- nodeJS
- yarn

After cloning this repo, open a terminal window and start the web app using the following commands from the project root:

```
yarn start  
yarn install
```

[Check out our GitHub](#) →

Pull requests

cuepoints/cuepoints	#2809493	● Passed
override/on-demand-trial	master	● Failed
app-integrations/external	#4238402	● Aborted

[View all CI/CD](#) →

Information

[General](#) [Configuration](#)

Owner: #2809493

Support: #playlist

Links: [GHE](#), [Create a plugin](#), [Plugin Gallery](#), [backstage.io](#)

Activity

News: Stackoverflow now live! 2019-12-27
After a successful pilot we have decided to invest in...

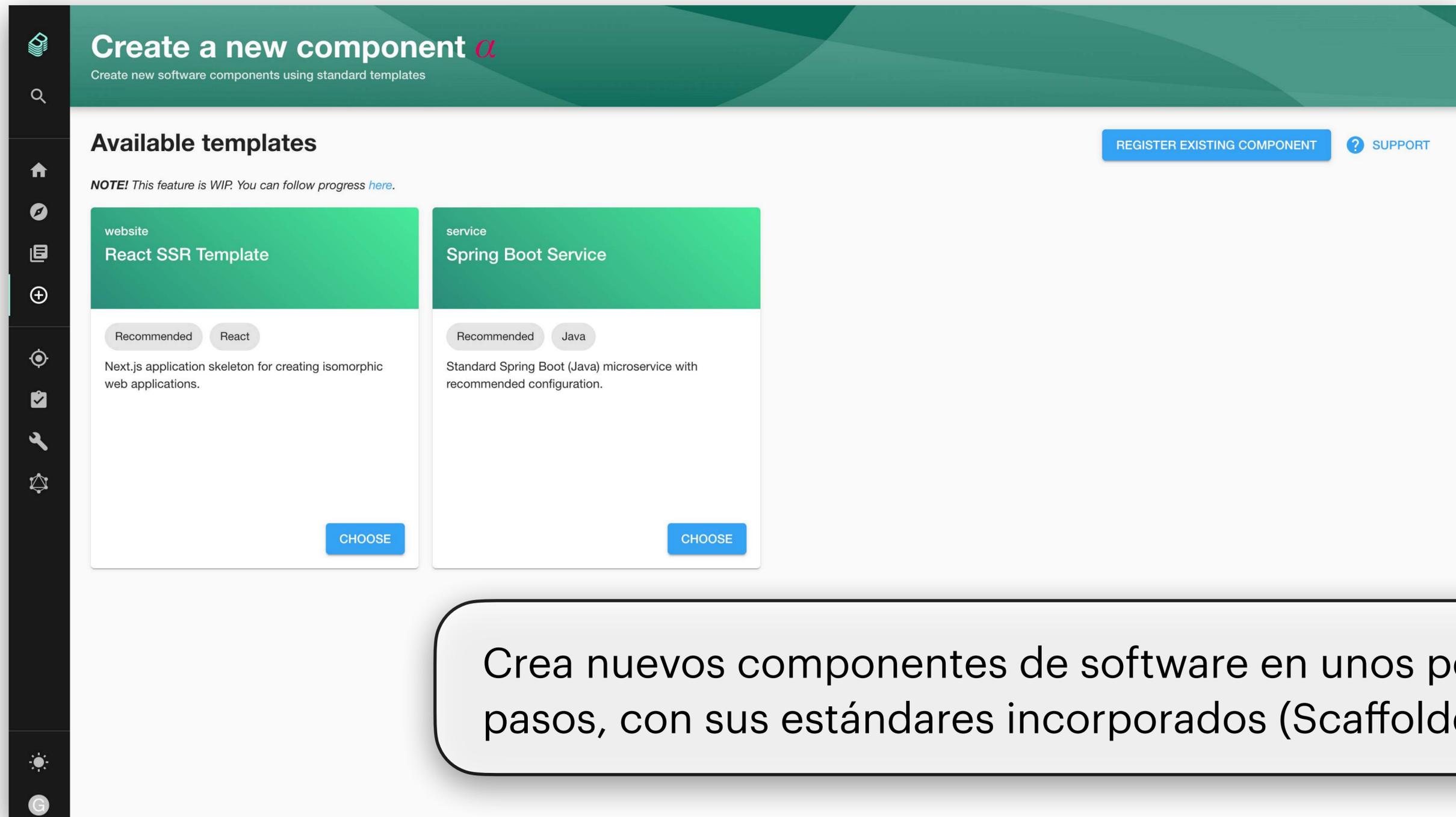
Product Update: You can now... 2019-12-27

Monitoring

The monitoring graph shows a line chart with a light blue area fill. The y-axis ranges from 750 to 1250. The line starts at approximately 1100, dips to 1000, rises to a peak of 1250, dips to 1000, and then rises again to 1250.

- CI/CD
- Tests
- API
- Monitoring
- Quality

Software Templates



The screenshot displays a web interface for creating software components. At the top, a green header contains the text "Create a new component" with a red logo and the subtitle "Create new software components using standard templates". Below this, a section titled "Available templates" features a note: "NOTE! This feature is WIP. You can follow progress [here](#)." Two template cards are visible: "React SSR Template" (category: website) and "Spring Boot Service" (category: service). Each card includes a "Recommended" tag, a language tag ("React" or "Java"), a brief description, and a "CHOOSE" button. In the top right corner, there are buttons for "REGISTER EXISTING COMPONENT" and "SUPPORT". A dark sidebar on the left contains various navigation icons.

Crea nuevos componentes de software en unos pocos pasos, con sus estándares incorporados (Scaffolder).

Create a new component *alpha*

Create new software components using standard templates

React SSR Template

1 Fill in template parameters

Name*
MyNewComponent
Unique name of the component

Description*
A great new component
Description of the component

BACK NEXT STEP

2 Choose owner and repo

“A los desarrolladores de software les encanta construir plataformas y, sin [...] Product Management, crearán una plataforma **más grande de lo necesario.**”

- Allan Kelly, Agile

Carga Cognitiva



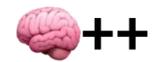
¿Dónde puedo encontrar la documentación completa sobre este sistema?



¿Hay algún estándar o plantilla que deba seguir para mi aplicación?



¿Hay algún repositorio específico o rama en la que deba trabajar?



¿Existe una suite de pruebas automatizadas o algún framework de pruebas que deba usar?



¿Cuál es el proceso para desplegar cambios en producción?

Adopción

Cognitive load assessment

¿Cómo es la experiencia de construir sus servicios?

*

Cosas a considerar: ¿es la construcción una tarea clara y repetible? ¿Es lo suficientemente rápida? ¿Qué sucede cuando las construcciones fallan? ¿Son fáciles de diagnosticar los fallos?

¿Cómo es la experiencia de probar sus servicios?

*

Cosas a considerar: ¿es la prueba una tarea clara y repetible? ¿Es lo suficientemente rápida? ¿Qué sucede cuando las pruebas fallan? ¿Son fáciles de diagnosticar los fallos? ¿Los entornos de prueba son adecuados? ¿Son fáciles de acceder/iniciar/limpiar/introducir datos de prueba en los entornos de prueba?

Dependencias de los equipos

Team name/focus	Depends on Team	Type (blocking/slowing/ok)	Short description of dependency (artifacts, approvals, other)

Team API

Fecha:

Nombre y enfoque del equipo:

Tipo de equipo:

¿Parte de una Plataforma? (s/n) Detalles:

¿Ofrecemos un servicio a otros equipos? (s/n) Detalles:

¿Qué tipo de expectativas de nivel de servicio tienen otros equipos de nosotros?

Software gestionado y evolucionado por este equipo:

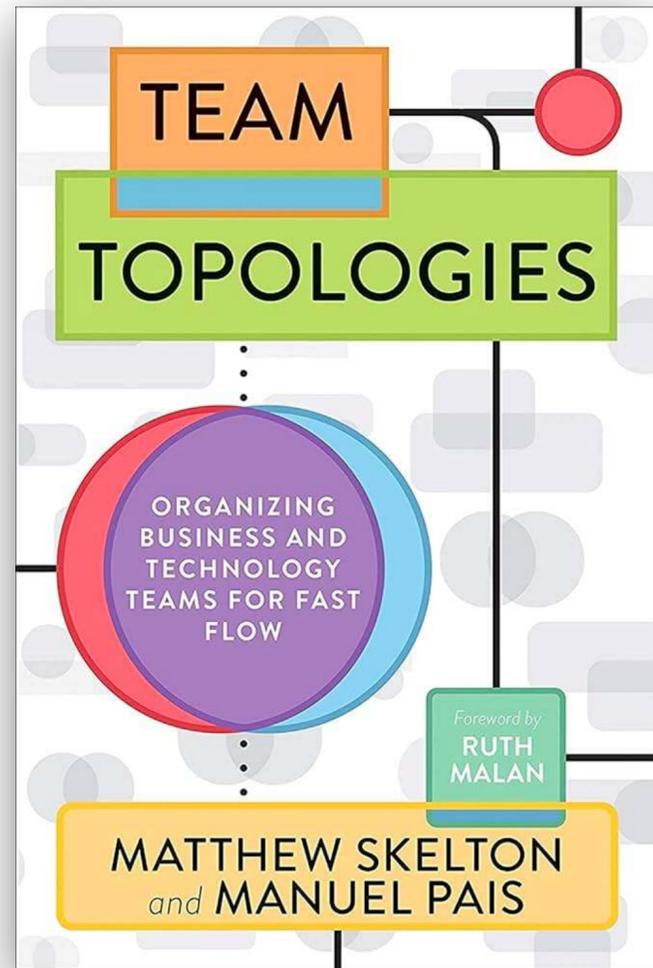
Métodos de versionado:

Términos de búsqueda en la Wiki:

Canales de herramienta de chat: #_____ #_____ #_____

Hora de la reunión diaria de sincronización:

¡Quiero saber más!



Internal
Developer
Platform

PLATFORM
ON23

"Minimizar la carga cognitiva para los demás" es una de las heurísticas más útiles para un buen desarrollo de software.

- Team Topologies, Matthew Skelton and Manuel Pais

iGracias!

¿Preguntas?

 @alebricio

¡Gracias!

¿Preguntas?

Twitter: @alebricio

LinkedIn: /alejandrabricio