Microfrontends

Marcos Méndez Filesi

www.mmfilesi.com

Contenido

Softtek®

Microfrontends

¿Se pueden aplicar a los frontales de las aplicaciones web los principios de la arquitectura back orientada a microservicios?

01 Introducción

02 Modelos de desarrollo

03 Solución técnica

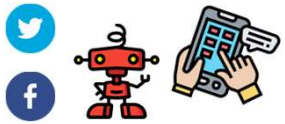
04 Epílogo

1. Introducción

1. Algo de historia

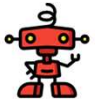
2012

Las redes sociales y las apps móviles son geniales, **¿podemos hacer webs que parezcan aplicaciones?**



Claro, **¡Hagamos single page applications!**

...mmm... Mi spa de 20.000 vistas, igual pesa un poco demasiado... **¿podemos hacer algo?**



Claro, **¡lazy load y split code!**

...mmm... Mi organización es colosal, **¿podemos hacer algo?**

Claro, **¡microfrontends!**

2022

Qué NO son los microfrontends

- Mezclar frameworks a lo loco en una aplicación
- Una solución técnica al problema de dividir el código.
- Un monolito distribuido.
- Lo que afirme una persona en una charla.

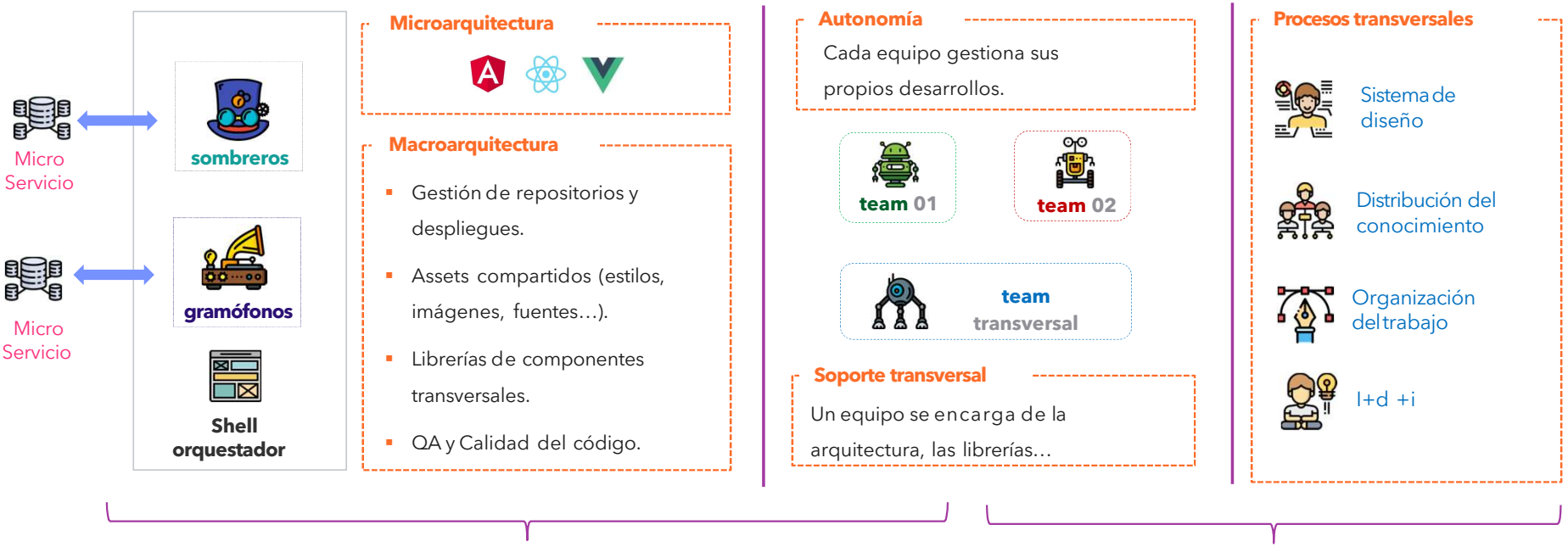
Una definición posible

- Una propuesta metodológica y técnica para trabajar de forma aislada, pero coordinada, los frontales de un conjunto de aplicaciones digitales que forman un sistema coherente.

1. Introducción

1. The big picture

Una estrategia de desarrollo basada en microfrontends plantea dos grandes retos: **metodológicos y tecnológicos.**



TECNOLOGÍA

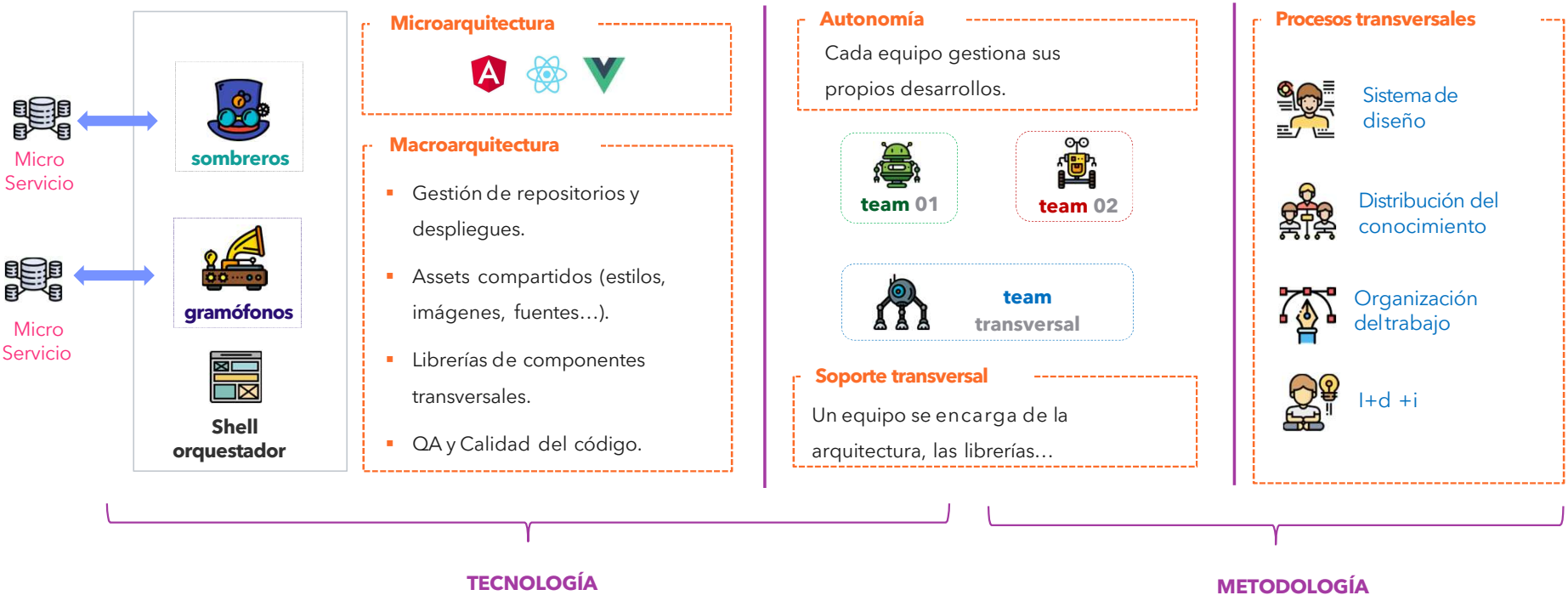
La combinación de distintas aplicaciones en una mayor es un proceso muy complejo que se puede afrontar con distintas estrategias.

METODOLOGÍA

Hay que organizar sistemas y protocolos que permitan cubrir las necesidades horizontales de los equipos verticales.

1. Introducción

1. The big picture



2. Modelos de desarrollo

Softtek®

1. Sistema de diseño

Una pieza fundamental

En general, cuanto más grande es un proyecto y más equipos hay involucrados, más fácil es que se pierda información, surjan bloqueos, se generen dependencias y se produzcan problemas de comunicación que terminan por impactar en el ciclo de desarrollo.

- Colores
- Tamaños, espaciados y separaciones
- Tipografía y elementos de texto
- Motivos visuales
- Componentes y módulos
- Reglas que marcan las pautas y uso de los diferentes elementos del sistema…

Figma

Figma es una aplicación para diseñar prototipos UI y UX de forma colaborativa que se ejecuta en el navegador. Permite monitorizar la propuesta de diseño en tiempo real y facilita el intercambio de opiniones entre un cliente y el proveedor. Contribuye, en suma, a la implementación iterativa del diseño sin margen de error.

Design tokens

Decisiones de diseño → Tokens de diseño → Formatos específicos

json
xml
sass
css

Microfrontends – Atomic Design, una historia de amor

2. Modelos de desarrollo

2. Gestión del conocimiento

Caos en el horizonte

En general, cuanto más grande es un proyecto y más equipos hay involucrados, más fácil es que se pierda información, surjan bloqueos, se generen dependencias y se produzcan problemas de comunicación que terminan por impactar en el ciclo de desarrollo.

Bus factor

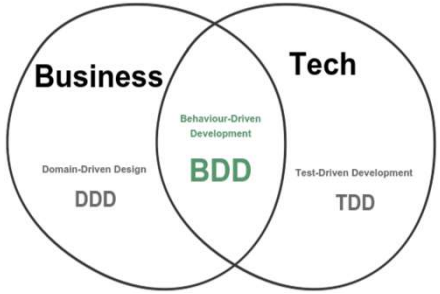
¿Qué sucede en un proyecto si determinados desarrolladores son atropellados por un autobús?

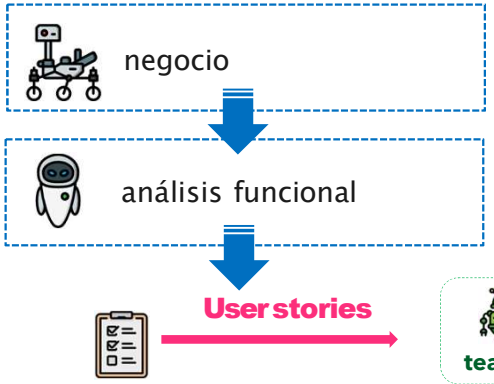
Fuente: Augie De Blieck

BDD

behavior-driven development, una técnica de desarrollo que establece un puente entre negocio y tecnología.

- Formulación de tests E2E.
- Indicaciones para los equipos de desarrollo.
- Documentación viva del proyecto.
- Criterios de aceptación.

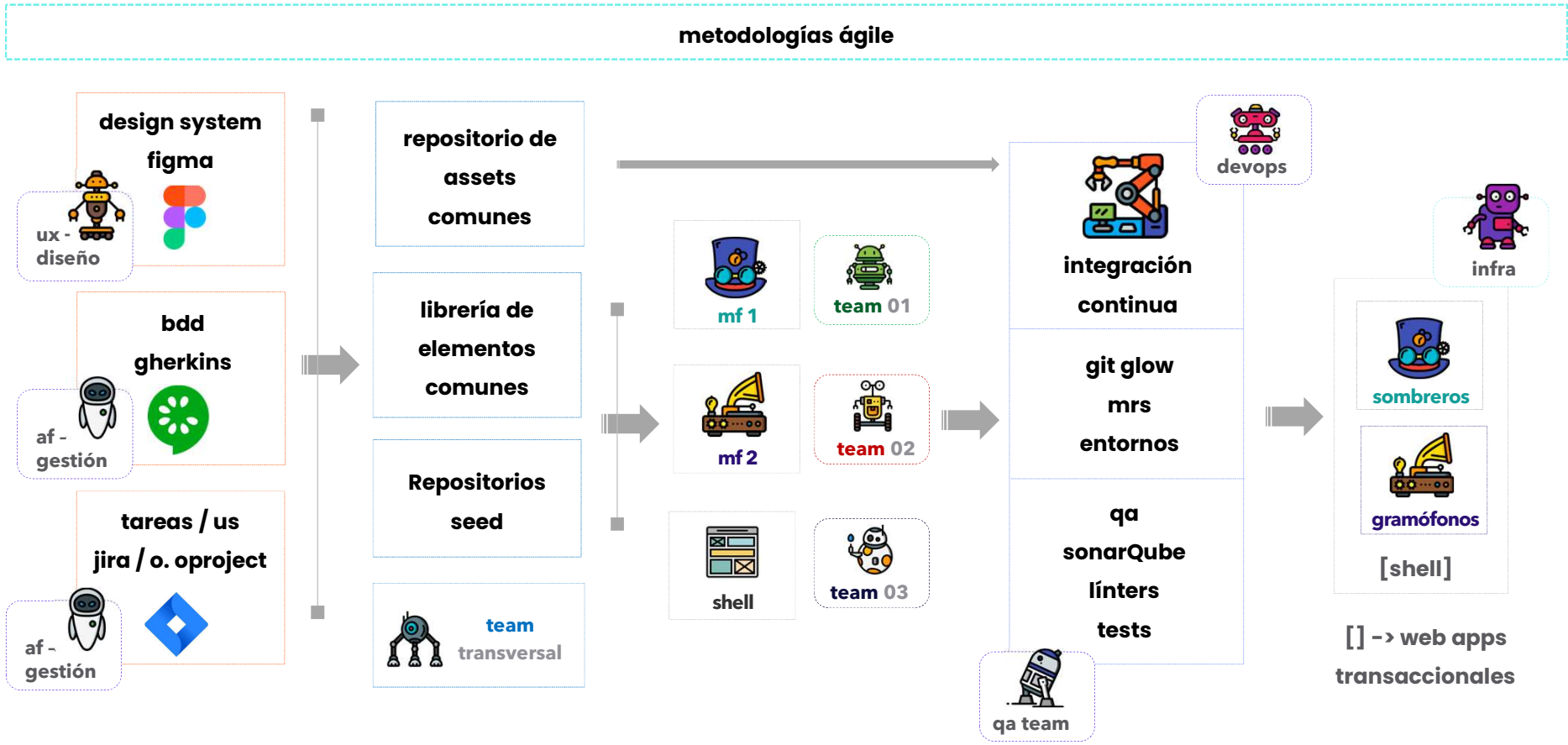





```
Feature: User trades stocks
  Scenario: User requests a sell before close of trading
    Given I have 100 shares of MSFT stock
      And I have 150 shares of APPL stock
      And the time is before close of trading
```

2. Modelos de desarrollo

3. Equipos y procesos



3. Solución técnica

1. Requisitos clave

Desarrollos independientes

Salvo algunos elementos transversales, los equipos deben poder desarrollar su aplicación de forma estanca sin que existan interdependencias.

Despliegues autónomos

Cada equipo debe poder desplegar su aplicación de forma independiente, sin necesidad de compilar toda la aplicación general.

Comunicación

El sistema debe permitir la comunicación entre las distintas aplicaciones, ya sea de forma directa o pasando por un agente intermediador.

Lazy load y performance

El código de las aplicaciones solo debe cargarse bajo demanda, cuando se necesita. Además, deben evitarse cargas dobles del mismo recurso (doble bundle).

Namespaces

El sistema debe poder aislar de alguna manera el código js y los estilos (cuando se necesite) para evitar conflictos de versiones, reglas de css sobrescritas, etcétera.

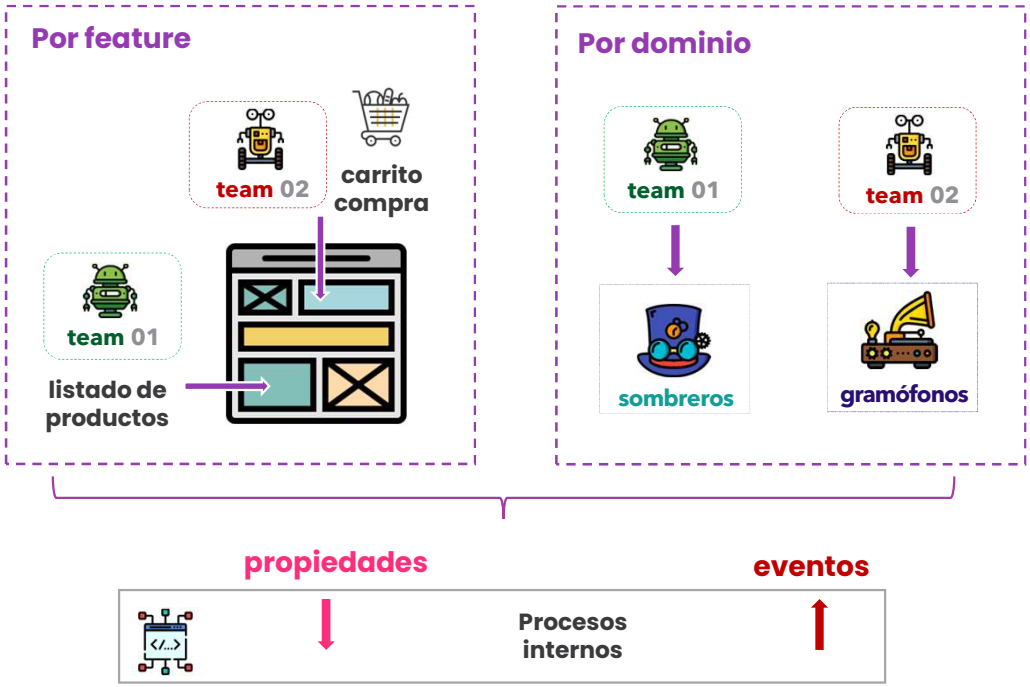
Consistencia

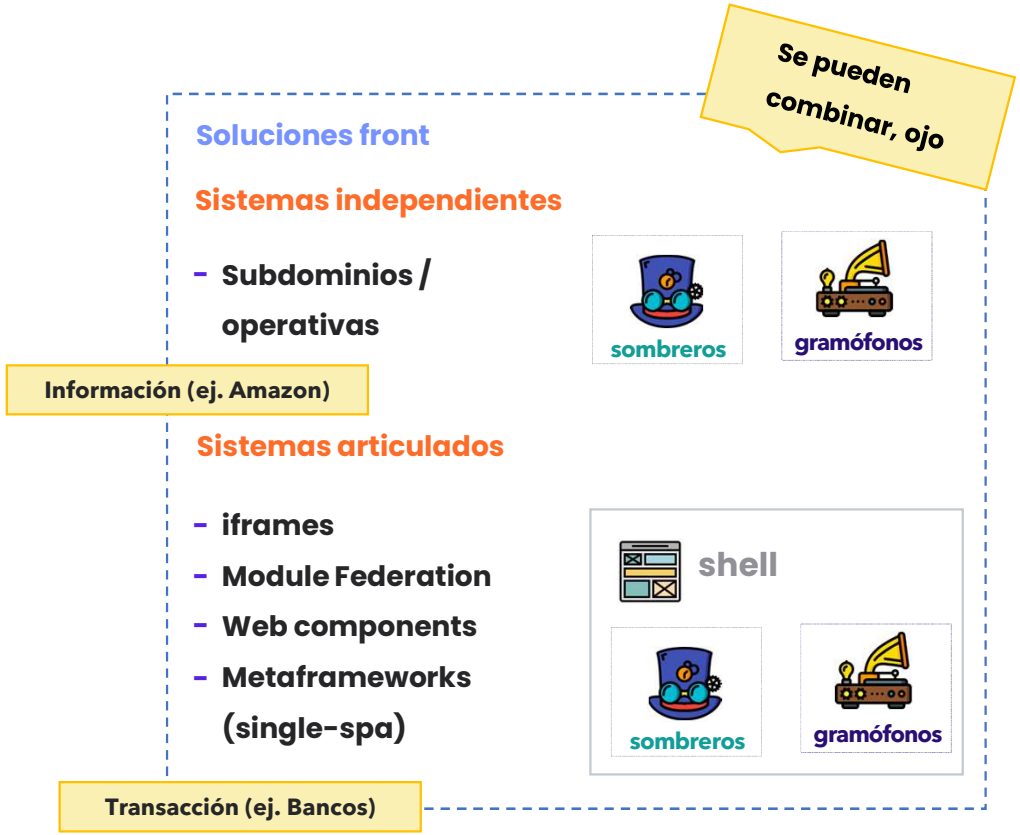
El todo es mayor que la suma de las partes: el usuario debe tener la sensación de que se encuentra en la misma aplicación gracias a la coherencia de las interfaces.

3. Solución técnica

2. Tipos de microfrontends

La responsabilidad de cada aplicación se puede definir por *feature* o por dominio; en cualquier caso, la interrelación de las aplicaciones debe limitarse a los extremos I/O y lo que suceda entremedias debe ser una caja negra (*Ley de Deméter*).





3. Solución técnica

3. Stack tecnológico

herramientas especializadas en componentes

Polymer - litElement: Librería de Google que ha servido como punto de fuga sobre el funcionamiento que debían tener los web components nativos.

Stencil: Desarrollada por el equipo de Ionic, permite preparar web components de forma muy cómoda.

Web components nativos: Los web components se pueden desarrollar de forma nativa usando APIs estándar.

Angular elements: Angular permite preparar web components mediante el api de custom-element complementada con alguna librería externa, como ngx-build-plus.

3. Solución técnica

4. Estrategias principales

Module Federation

Es un plugin de webpack que permite engarzar spas levantadas en algún sitio en un shell orquestador.

Ventajas

- No hay que modificar el código natural de las
- aplicaciones.
- Forma parte del estándar.
- Permite compartir recursos de forma óptima.

Inconvenientes

- Solo se pueden combinar frameworks recurriendo a técnicas complementarias (componentizando las microaplicaciones de tecnologías distintas).
- Las piezas se articulan a través de webpack, por lo que no son compatibles por sí mismos con otras tecnologías, como una página php.

Escenarios recomendados

- Aplicaciones transaccionales monoframework.

Web components

Con la herramienta que sea, crear componentes estancos que se gestionan a sí mismos y funcionan mediante un sistema I/O de propiedades y eventos para gestionar la comunicación

Ventajas

- Los mecanismos básicos forman parte del estándar web.
- El shadow dom permite encapsular si fuera necesario el JavaScript y los estilos de las microaplicaciones.
- Los componentes se pueden utilizar en cualquier entorno web, así como en las aplicaciones híbridas.
- Son plug and play.
- La macroarquitectura no es compleja.
- Son modulares y muy escalables.

Inconvenientes

- Es muy importante mantener una organización precisa para optimizar el esfuerzo productivo.

Escenarios recomendados

- Sistemas muy grandes donde conviven distintas tecnologías.

4. Epílogo

Cuatro ideas fuerza

yagni

***You Ain't Gonna Need It,* igual no lo necesitas.**

Los elementos del sistema no deben cubrir todas las casuísticas imaginables. Basta con que se puedan modificar y escalar con facilidad en función de las necesidades que vayan surgiendo.

Ley de Demeter (LoD)

Principio del menor conocimiento.

Cada unidad debe exponer al exterior el interfaz más reducido posible. Las piezas del sistema deben ser lo más autónomas y estancas posibles.

dry

***Don't repeat yourself,* no te repitas a ti mismo.**

Si hay algún fragmento de código idéntico o muy similar hay que buscar la manera de concentrarlo en un solo sitio compartido como repositorios comunes, microfrontends, servicios...

kiss

***Keep it simple, stupid!,* Mantenlo simple.**

Cuanto mayor sea la dificultad del desarrollo, más costoso será mantener y escalar el sistema, por lo que siempre hay que tratar de reducir la complejidad accidental.

4. Epílogo

Softtek®

Algo de código en...

https://github.com/mmfilesi/microfrontends-demo-shell

https://github.com/mmfilesi/microfrontends-demo-mf

https://github.com/mmfilesi/microfrontends-demo-library

Para saber más

Softtek®

Microfrontends

- Geers, Michael. *Micro Frontends in Action.* Manning, 2020. [Libro].
- Geers, Michael. Web microfrontends. Ver.
- Jackson, Cam. *Micro Frontends.* Martin Fowler. Ver.
- Steyer, Manfred. *A Software Architect's Approach Towards.* Angular Architects. Ver.
- Mezzalira, Luca. *Building Micro-frontends.* O'Reilly. [libro, wp].
- Mezzalira, Luca. Micro Frontend Architecture. UXDX. [Vídeo]. Ver.
- WhaaaatTheFront. *Microfrontends, ¿el futuro del #frontend?* [Vídeo]. Ver.

Module Federation

- Steyer, Manfred. *The Microfrontend Revolution: Module Federation in Webpack 5* (serie). Angular Architects. Ver.
- Steyer, Manfred. *The Microfrontend Revolution: Using Webpack 5 Module Federation with Angular*. [Vídeo]. Ver.

Web components

- Bidelman, Eric. *Custom Elements v1: Reusable Web Components.* Web Fundamentals (Google). Ver.
- Caleb, Williams. *An Introduction to Web Components*. CSS-tricks. Ver.
- Desarrollo web. *Manual de Web Components.* Ver.

Para saber más

Softtek®

Angular Elements

- Álvarez, Mariano. Angular Elements. Angular Honduras. [Vídeo]. Ver.
- Basal, Netanel. *Understanding the Magic Behind Angular Elements.* Medium. Ver.
- Fireship. Advanced Angular Elements. [Vídeo]. Ver.
- Grassl, Timon. *How to use routing in angular web components.* Medium. Ver.
- Hahne, Rainer. *Why you don't need Web Components in Angular.* Codeburst. Ver.
- Li, Jia. *How Angular Elements transmits Component's @Outputs outside Angular.* Medium. Ver.
- Steyer, Manfred. *Micro Apps with Web Components using Angular Elements.* Angular Architects. Ver.
- Steyer, Manfred. *Angular Elements* (serie). Angular Architects. Ver.
- Vardanyan, Armen. *Angular + Web Components: a complete guide.* Indepth Dev. Ver.

Single-spa

- Mosconi, M. y Ramirez, F. *Implementacion de Micro frontend con SingleSPA. Redbee Studios.* [Vídeo]. Ver.
- Souza, Diogo. *Creating micro-frontend apps with single-spa.* Log Rocket. Ver.

Iframes

Rifki, Nada. *The ultimate guide to iframes*. Log Rocket. Ver.

Softtek®

Disruptive DNA to boost the digital world through technology innovation.

Our expertise includes digital services, products and solutions, technological innovation and agile transformation processes.

With a team of over 15,000 professionals, 19 Global Delivery Centers and 54 Sales Branches we actively accompany our customers from North America, Latin America, Europe & Asia throughout their technological and digital transformation processes.

Our mission is to create the most value for our clients, economic sectors and society as a whole by designing and developing cutting-edge technology initiatives that aim to implement disruptive digital change.

Join us at Softtek: https://softtek.com/jobs