

Arquitecturas Orientadas a Componentes Web

Patrones de Acceso a Datos

Javier Vélez Reyes

@javiervelezreye

Javier.velez.reyes@gmail.com

Octubre 2016

POLYMER.DAY

Patrones de Acceso a Datos

Autor

¿Quién Soy?



Licenciado por la UPM desde el año 2001 y doctor en informática por la UNED desde el año 2009, Javier conjuga sus labores como profesor e investigador con la consultoría y la formación técnica para empresa. Su línea de trabajo actual se centra en la innovación y desarrollo de tecnologías para la Web. Además realiza actividades de evangelización y divulgación en diversas comunidades IT y es coordinador de varios grupos de ámbito local como NodeJS Madrid o Madrid JS. Forma parte del programa Polymer Polytechnic Speaker y es mentor del capítulo de Madrid de Node School.



javier.velez.reyes@gmail.com



@javiervelezreye



linkedin.com/in/javiervelezreyes



gplus.to/javiervelezreyes



jvelez77



javiervelezreyes



youtube.com/user/javiervelezreyes



Javier Vélez Reyes

@javiervelezreye

Javier.velez.reyes@gmail.com

1 *Introducción*

- Arquitectura de Referencia para Componentes Web
- Plano Arquitectónico, Plano de Diseño & Plano Tecnológico
- Arquitectura del Subsistema de Datos

Patrones de Acceso a Datos

Introducción

I. Arquitectura de Referencia Para Componentes Web

Una arquitectura de referencia establece orientación acerca de cómo pueden construirse soluciones orientadas a componentes Web.



La construcción de soluciones orientadas a componentes Web es inherentemente más compleja que otras aproximaciones competitivas.

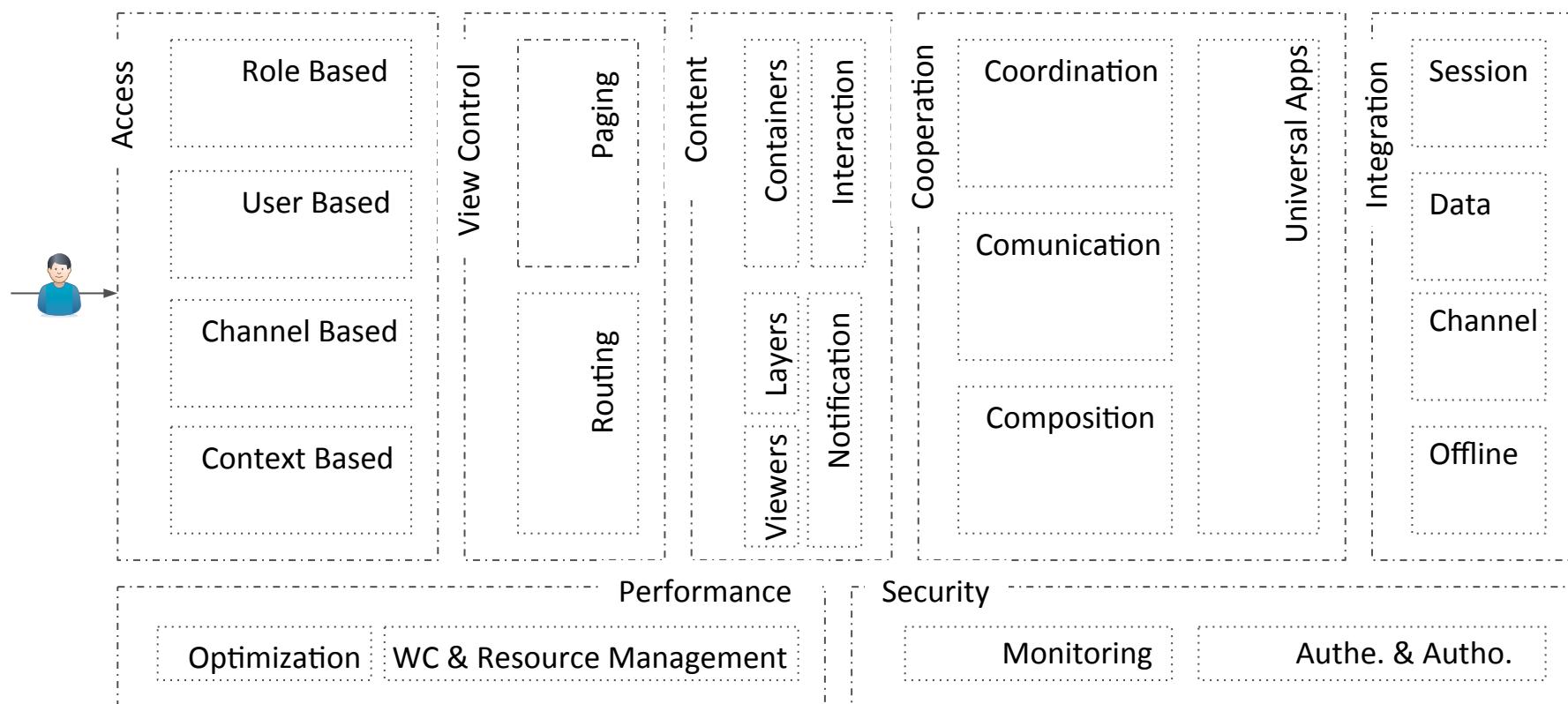


Patrones de Acceso a Datos

Introducción

I. Arquitectura de Referencia Para Componentes Web

Una arquitectura de referencia establece orientación acerca de cómo pueden construirse soluciones orientadas a componentes Web.



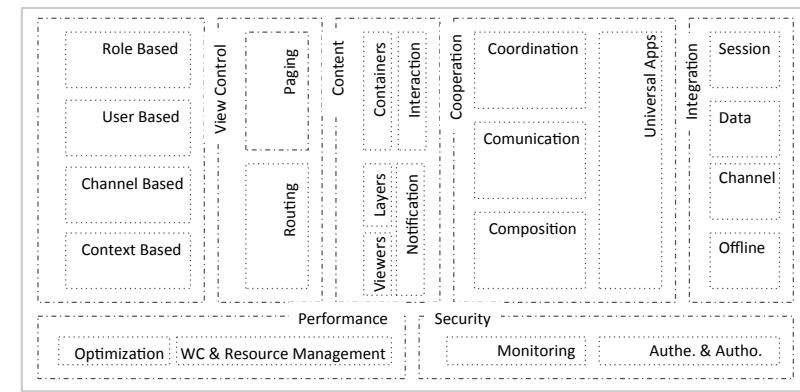
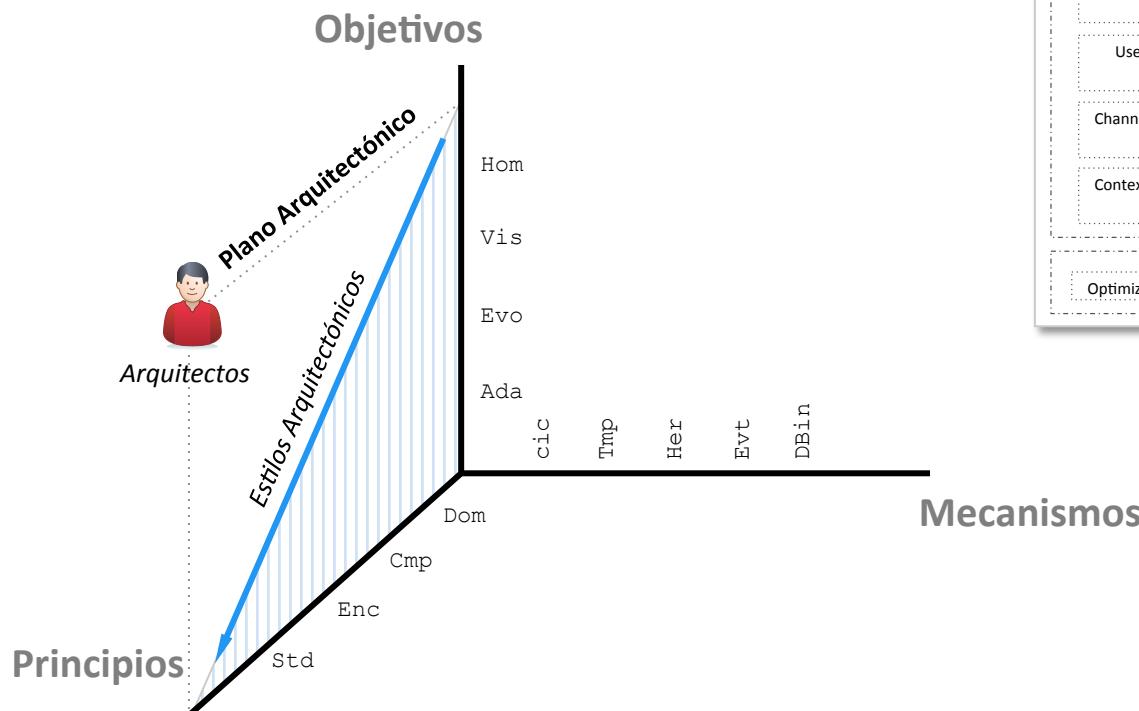
Patrones de Acceso a Datos

Introducción

II. La Arquitectura de Referencia en el Paradigma de Componentes Web

Plano Arquitectónico

El modelo arquitectónico expone explícitamente todos los espacios de problema que pueden aparecer en el marco de soluciones orientadas a componentes web.



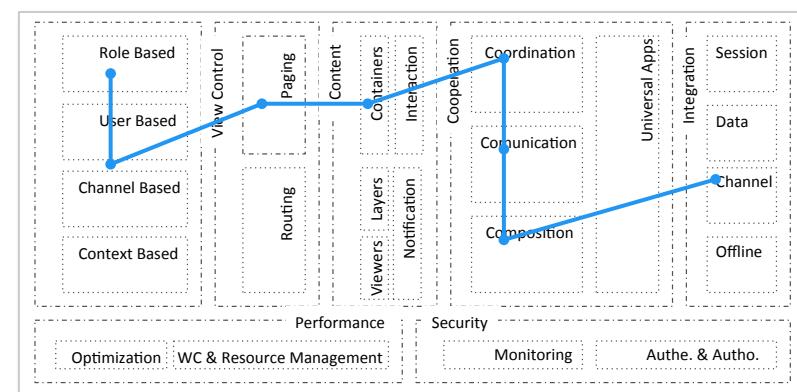
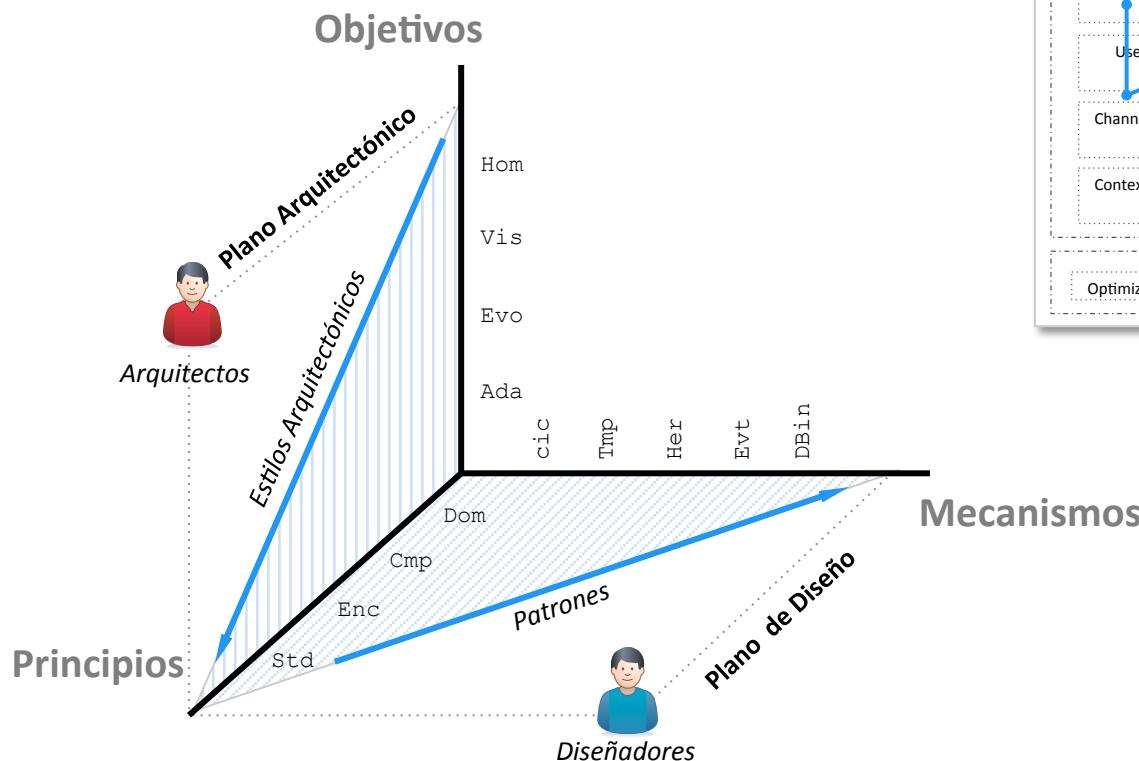
Patrones de Acceso a Datos

Introducción

II. La Arquitectura de Referencia en el Paradigma de Componentes Web

Plano de Diseño

El diseño dirigido por patrones es una forma sistemática de explorar esos espacios para encontrar soluciones de eficacia probada a problemas recurrentes para contextos reales.



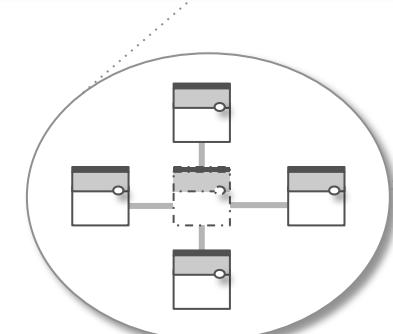
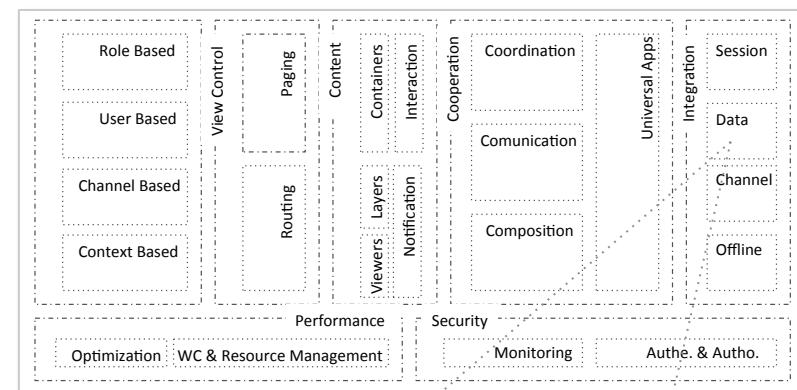
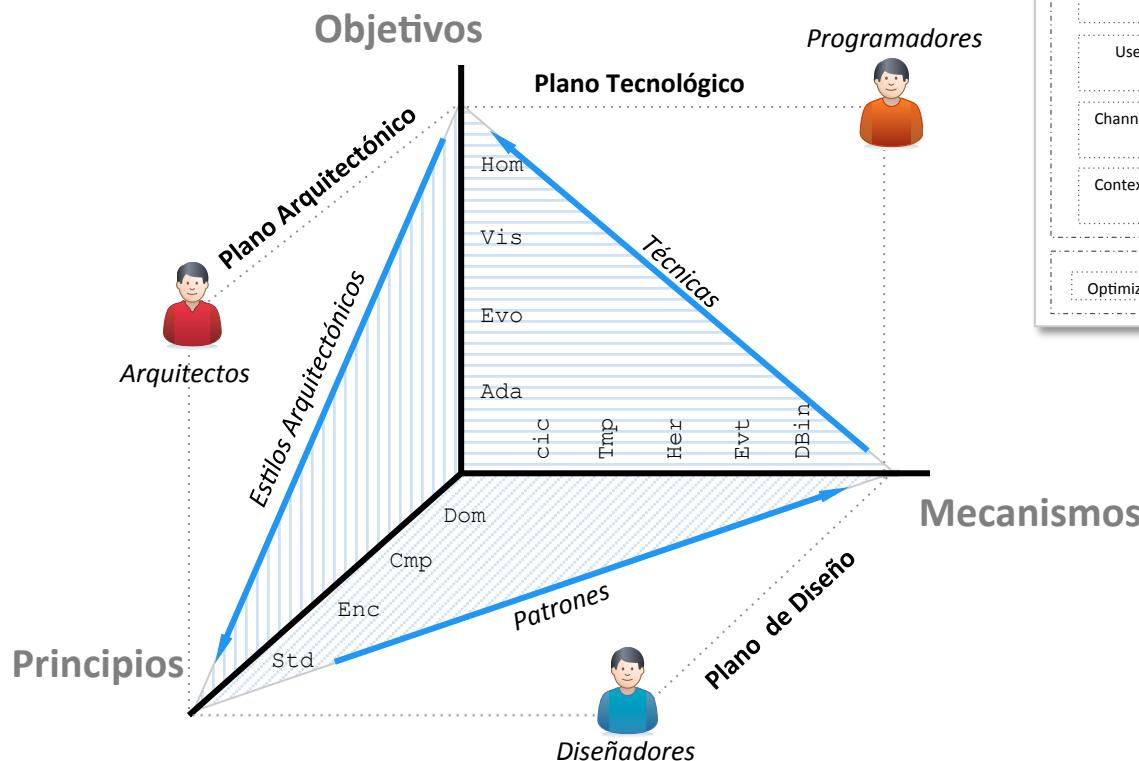
Patrones de Acceso a Datos

Introducción

II. La Arquitectura de Referencia en el Paradigma de Componentes Web

Plano Tecnológico

La arquitectura proporciona componentes como materialización de patrones. El contexto arquitectónico fija el contrato de cada componente.

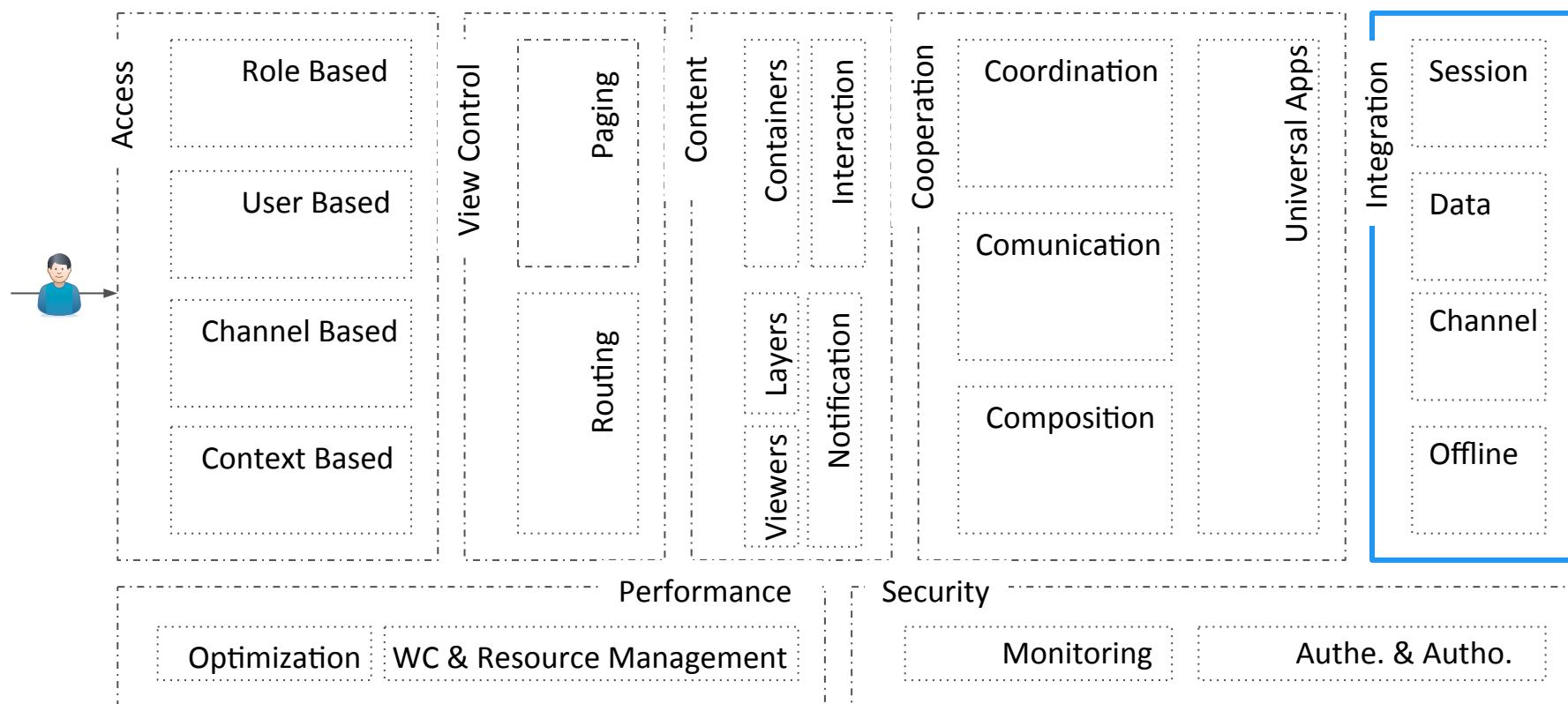


Patrones de Acceso a Datos

Introducción

III. El Subsistema de Acceso a Datos

El Subsistema de datos se posiciona dentro de la capa de integración. En general los componentes operan con datos accedidos desde esta capa.



Patrones de Acceso a Datos

Introducción

III. El Subsistema de Acceso a Datos

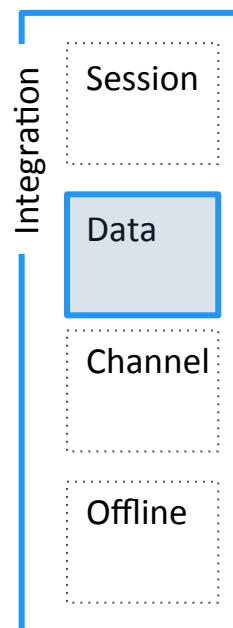
Dentro de la capa de integración se distinguen 4 subsistemas que tienen responsabilidades concomitantes con la gestión de datos de fuentes externas.

Acceso a Datos

Cómo se gestiona el acceso a fuentes de datos y los flujos de información asociados

Acceso Offline

Cómo se gestiona el acceso a datos en contextos de uso inestables o con desconexión



Control de Sesión

Cómo se gestiona el diferimiento de información de estado entre sesiones de trabajo

Control de Canal

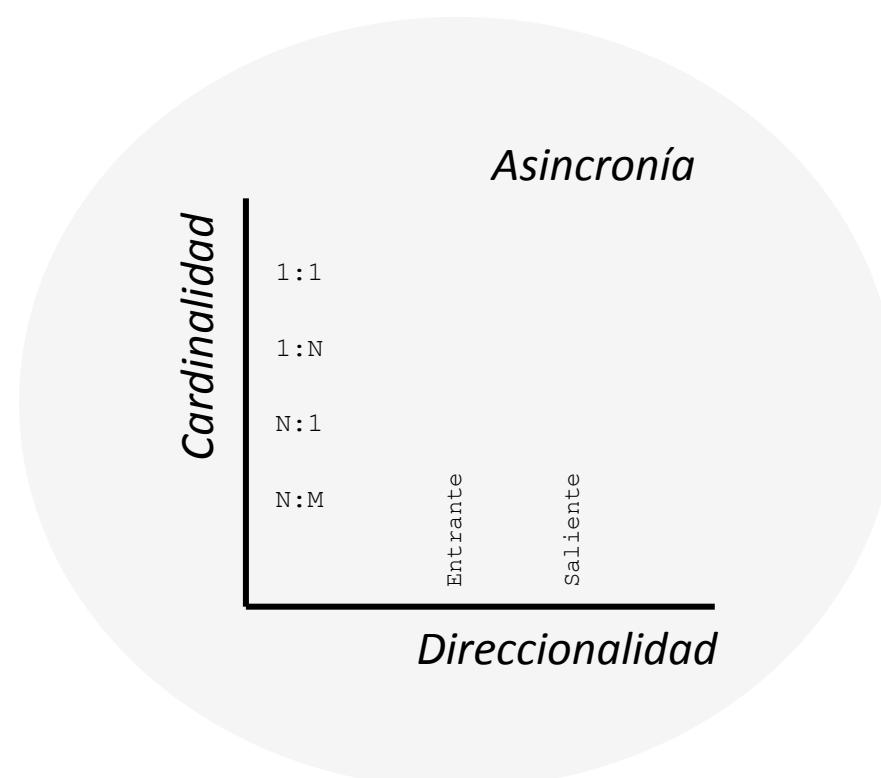
Cómo se gestiona la información extraída del canal utilizado (móvil, broser, etc.)

Patrones de Acceso a Datos

Introducción

III. El Subsistema de Acceso a Datos

Podemos caracterizar el subsistema de acceso a datos a partir de 3 propiedades fundamentales, su esencia asíncrona, su cardinalidad y su direccionalidad.



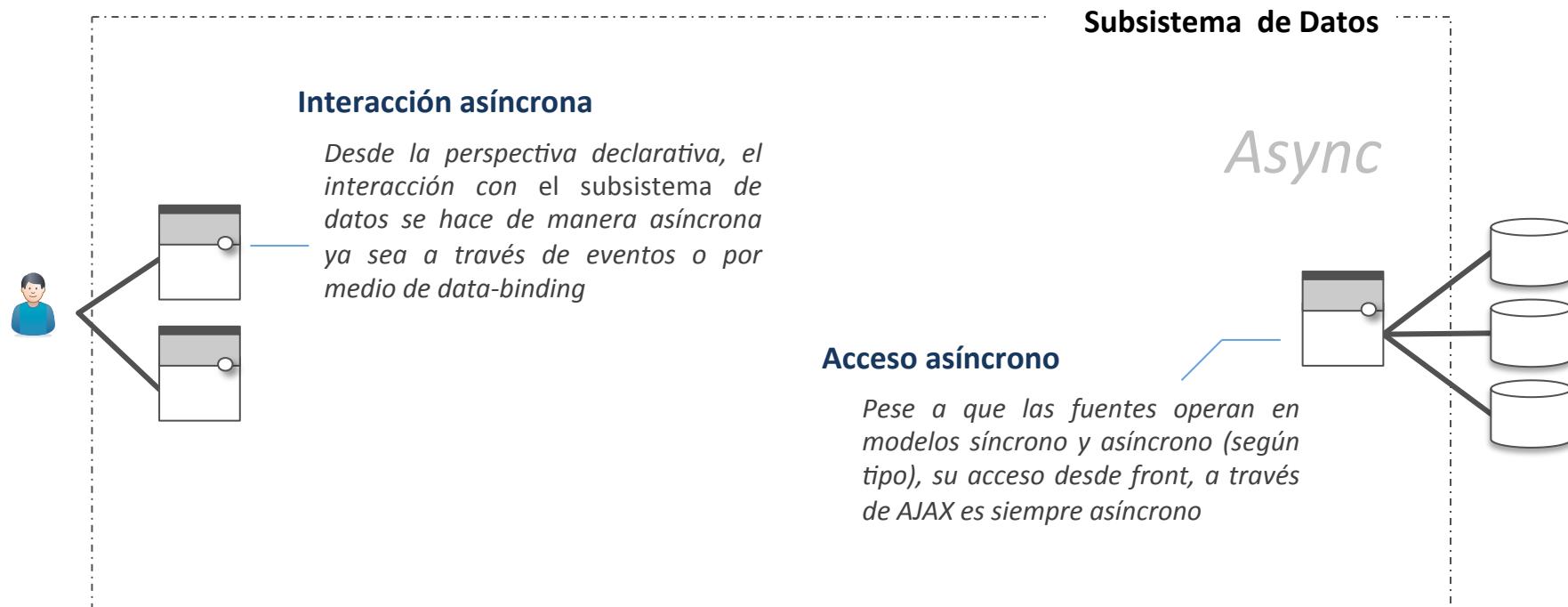
Patrones de Acceso a Datos

Introducción

III. El Subsistema de Acceso a Datos

Espacio Asíncrono

Las características de los elementos fronterizos incluidos en el subsistema de datos hace que ésta opere de acuerdo a un modelo completamente asíncrono.



Patrones de Acceso a Datos

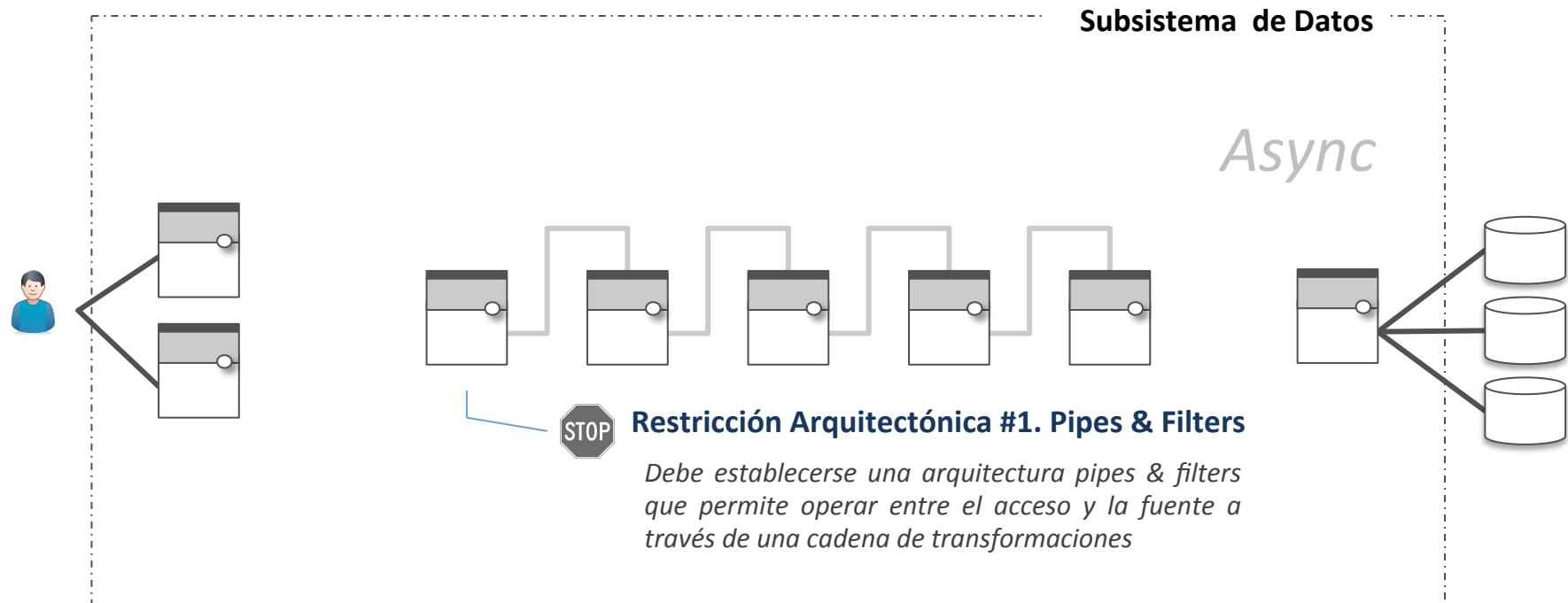
Introducción

III. El Subsistema de Acceso a Datos

Espacio Asíncrono

Las características de los elementos fronterizos incluidos en el subsistema de acceso a datos hace que esta opere de acuerdo a un modelo completamente asíncrono.

Arquitectura Pipes & Filters



Patrones de Acceso a Datos

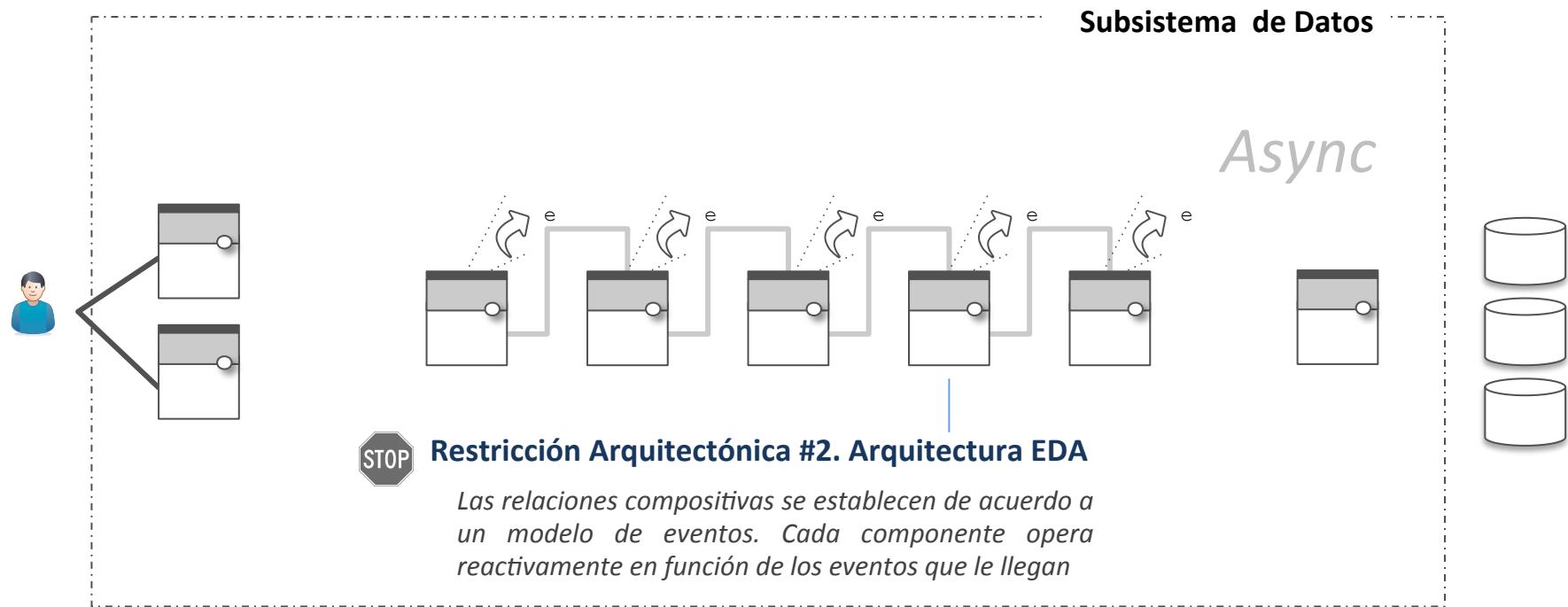
Introducción

III. El Subsistema de Acceso a Datos

Espacio Asíncrono

Las características de los elementos fronterizos incluidos en el subsistema de acceso a datos hace que esta opere de acuerdo a un modelo completamente asíncrono.

Arquitectura Dirigida por Eventos



Patrones de Acceso a Datos

Introducción

III. El Subsistema de Acceso a Datos

Espacio Asíncrono

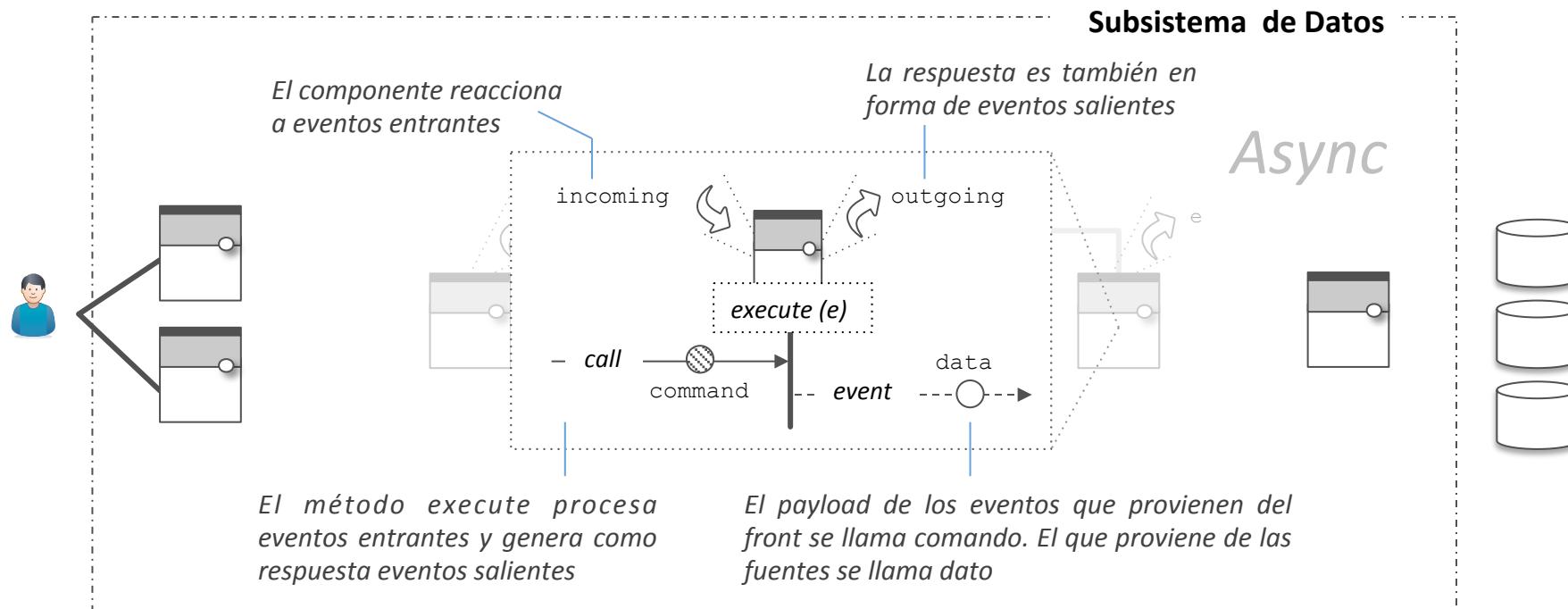
Las características de los elementos fronterizos incluidos en el subsistema de acceso a datos hace que esta opere de acuerdo a un modelo completamente asíncrono.

Contrato Reactivo Homogéneo



Restricción Arquitectónica #3. Contrato Reactivo Homogéneo

Los componentes deben compartir un mismo contrato que les permita operar dentro de una arquitectura de mensajes



Patrones de Acceso a Datos

Introducción

III. El Subsistema de Acceso a Datos

Espacio Asíncrono

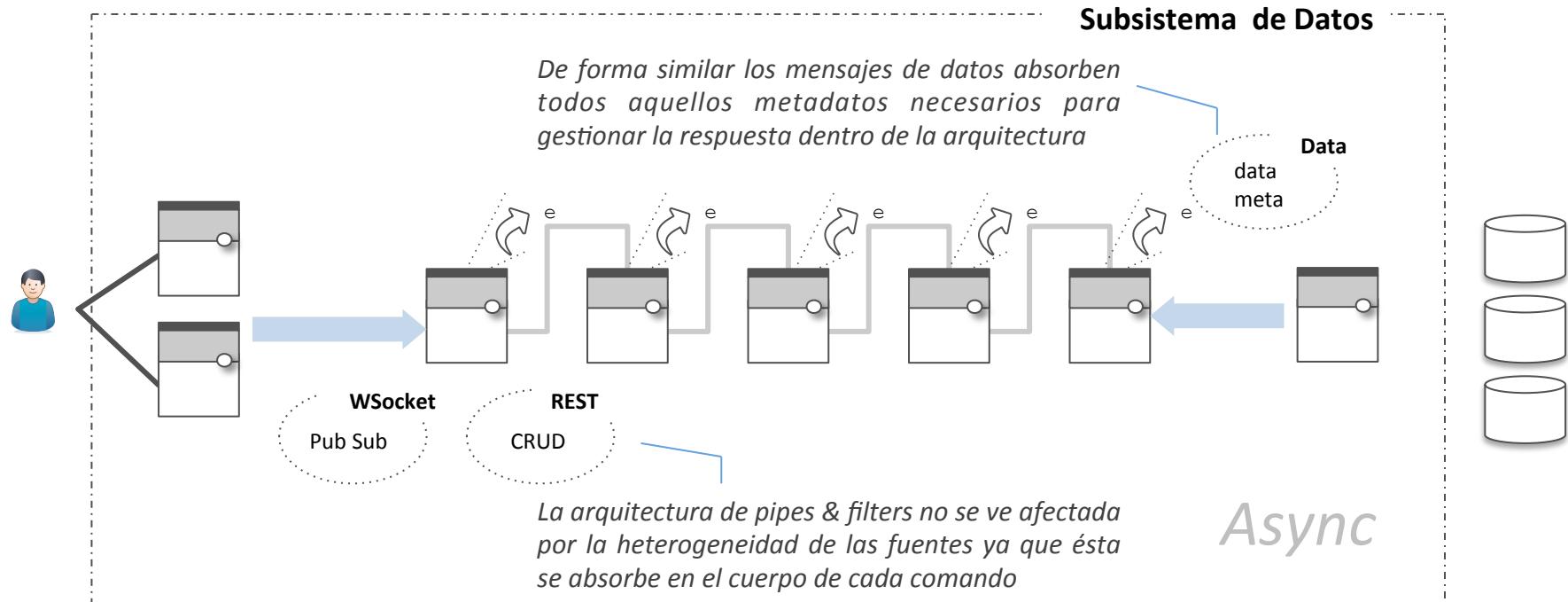
Las características de los elementos fronterizos incluidos en el subsistema de acceso a datos hace que esta opere de acuerdo a un modelo completamente asíncrono.

Contrato Reactivo Homogéneo



Restricción Arquitectónica #3. Contrato Reactivo Homogéneo

Los componentes deben compartir un mismo contrato que les permita operar dentro de una arquitectura de mensajes



Patrones de Acceso a Datos

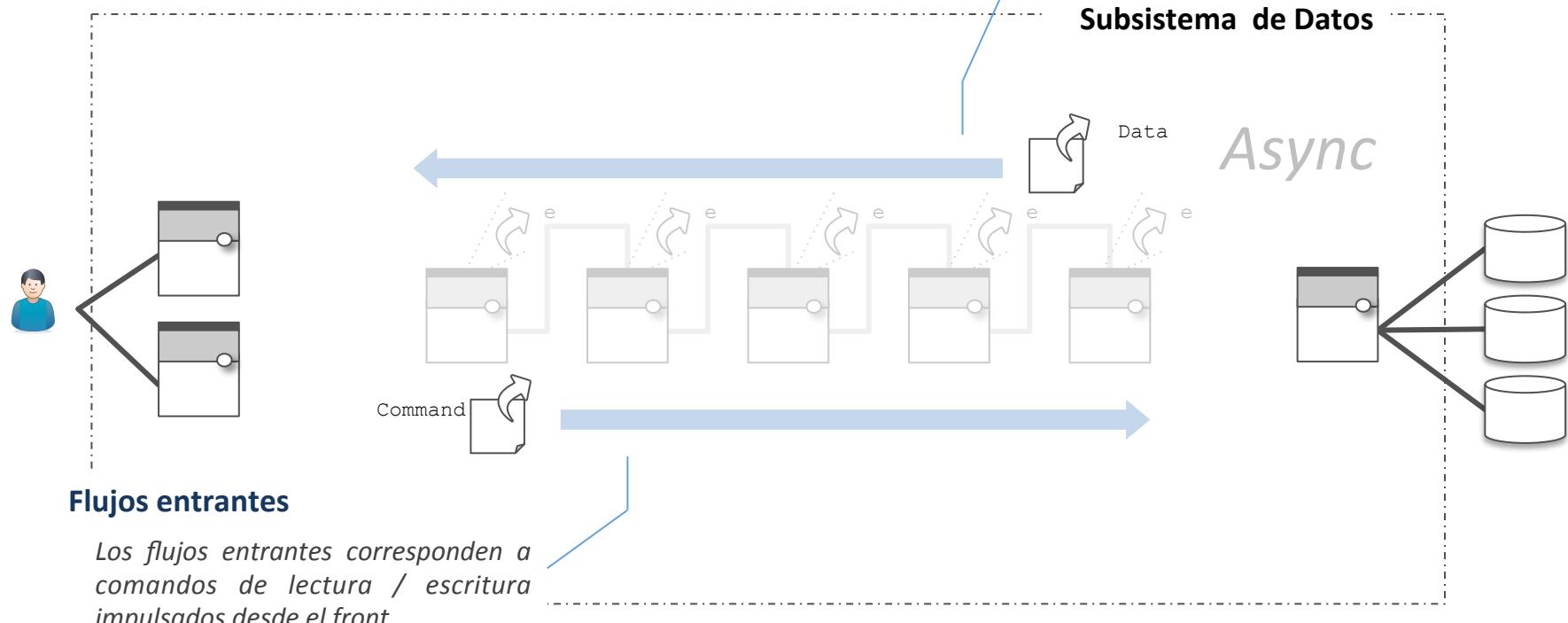
Introducción

III. El Subsistema de Acceso a Datos

Direccionalidad

La direccionalidad se refiere a la dirección que toman los flujos de datos en la arquitectura Pipes & Filters. Podemos distinguir entre flujos de entrada y de salida.

Flujos Entrantes & Salientes Independientes



Patrones de Acceso a Datos

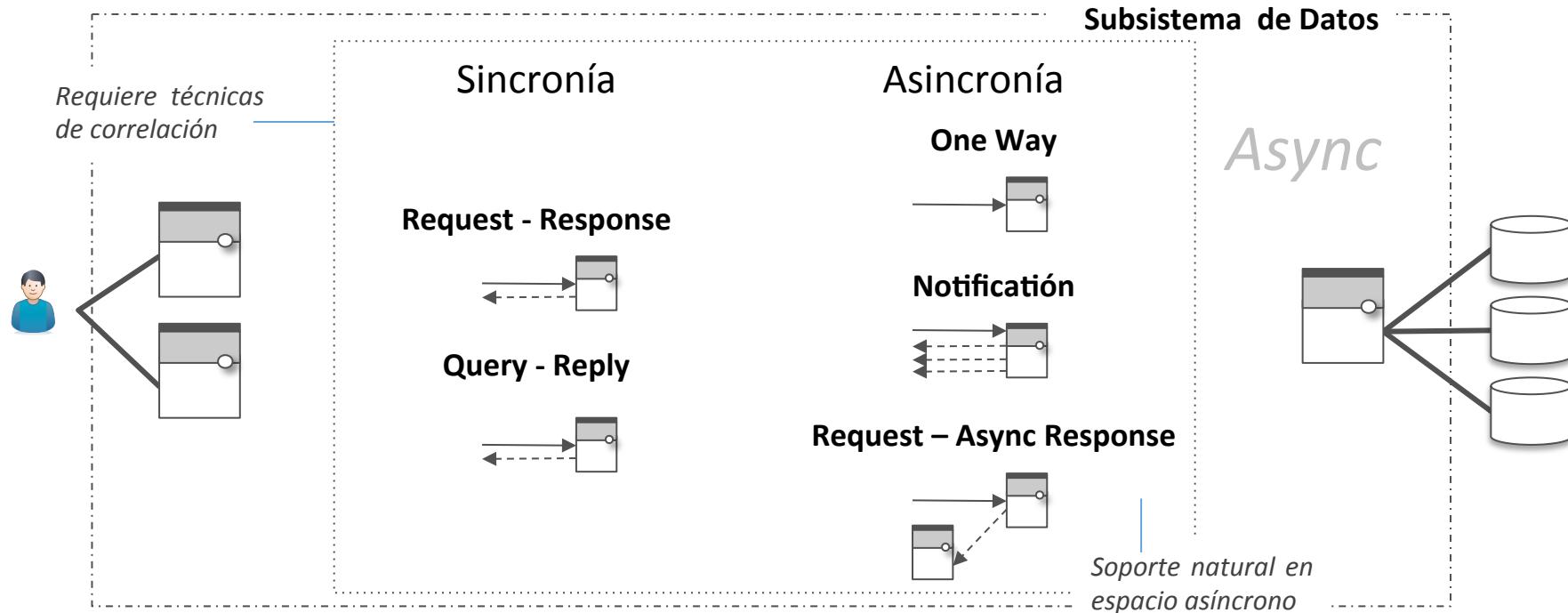
Introducción

III. El Subsistema de Acceso a Datos

Direccionalidad

La direccionalidad se refiere a la dirección que toman los flujos de datos en la arquitectura Pipes & Filters. Podemos distinguir entre flujos de entrada y de salida.

Comunicación Correlada



Patrones de Acceso a Datos

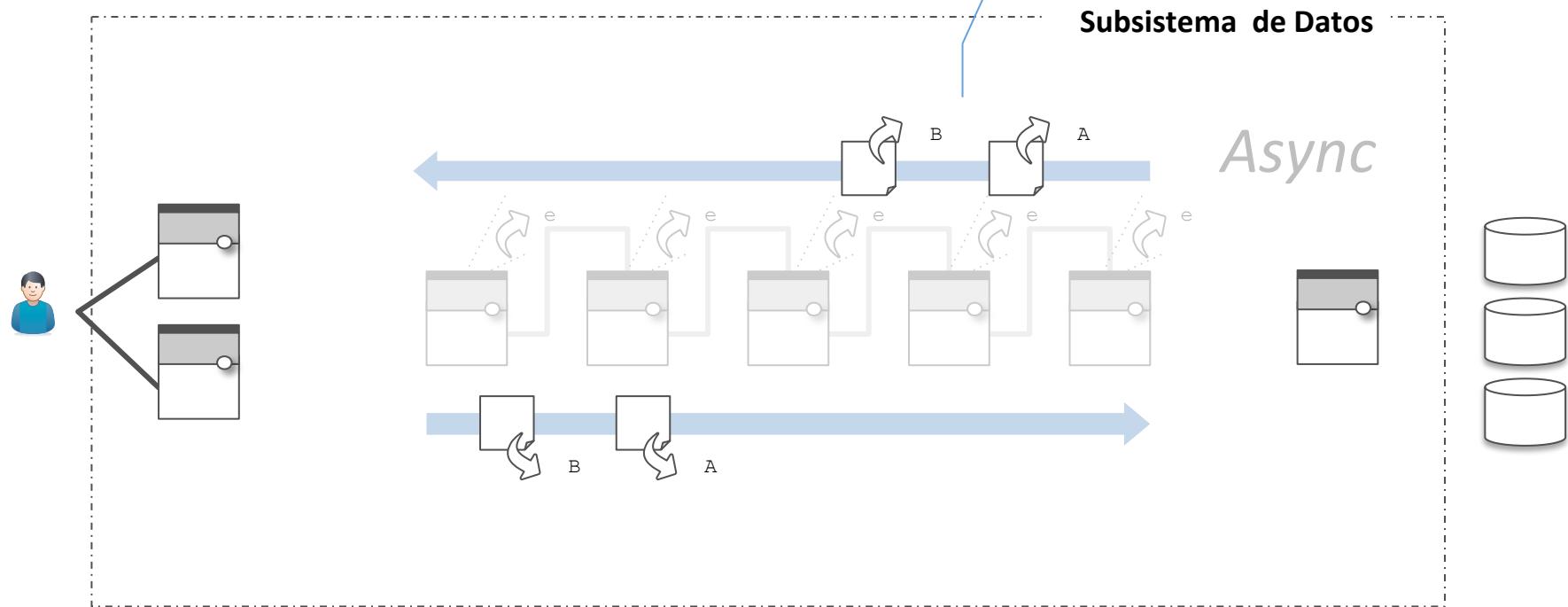
Introducción

III. El Subsistema de Acceso a Datos

Direccionalidad

La direccionalidad se refiere a la dirección que toman los flujos de datos en la arquitectura Pipes & Filters. Podemos distinguir entre flujos de entrada y de salida.

Comunicación Correlada. Ejemplo #1



Patrones de Acceso a Datos

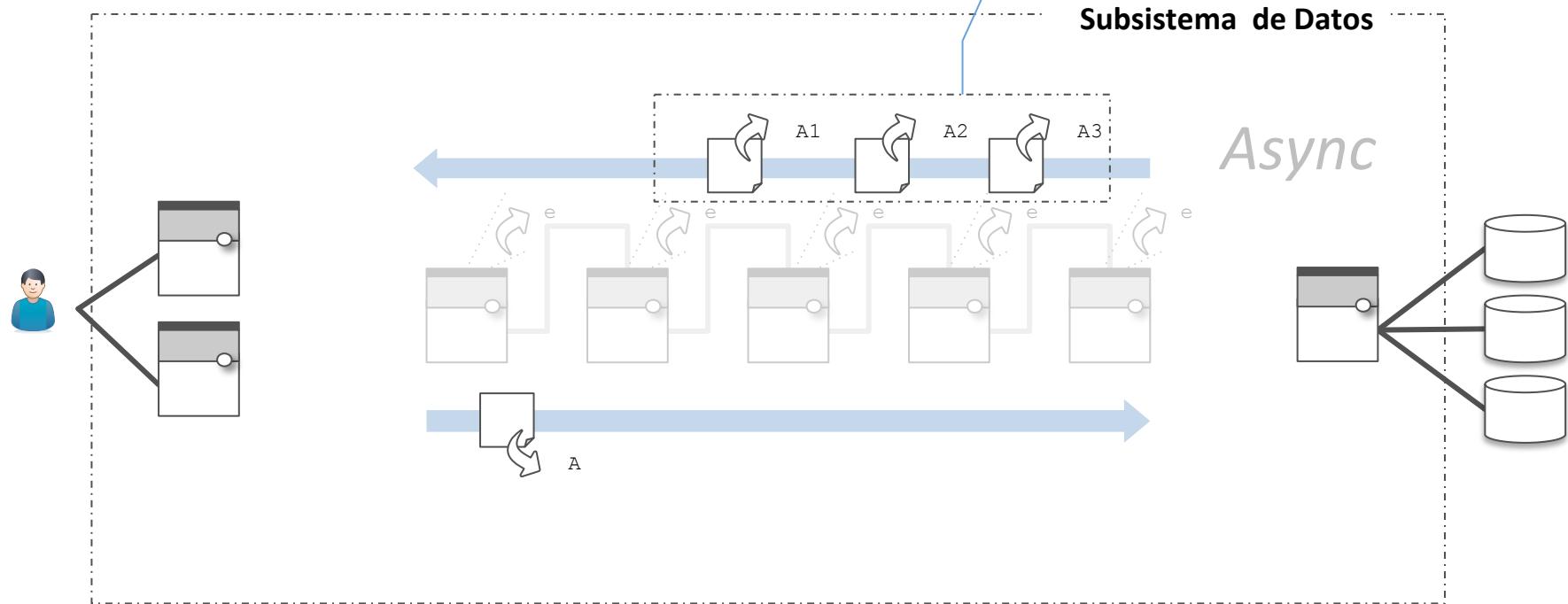
Introducción

III. El Subsistema de Acceso a Datos

Direccionalidad

La direccionalidad se refiere a la dirección que toman los flujos de datos en la arquitectura Pipes & Filters. Podemos distinguir entre flujos de entrada y de salida.

Comunicación Correlada. Ejemplo #2



Patrones de Acceso a Datos

Introducción

III. El Subsistema de Acceso a Datos

Cardinalidad

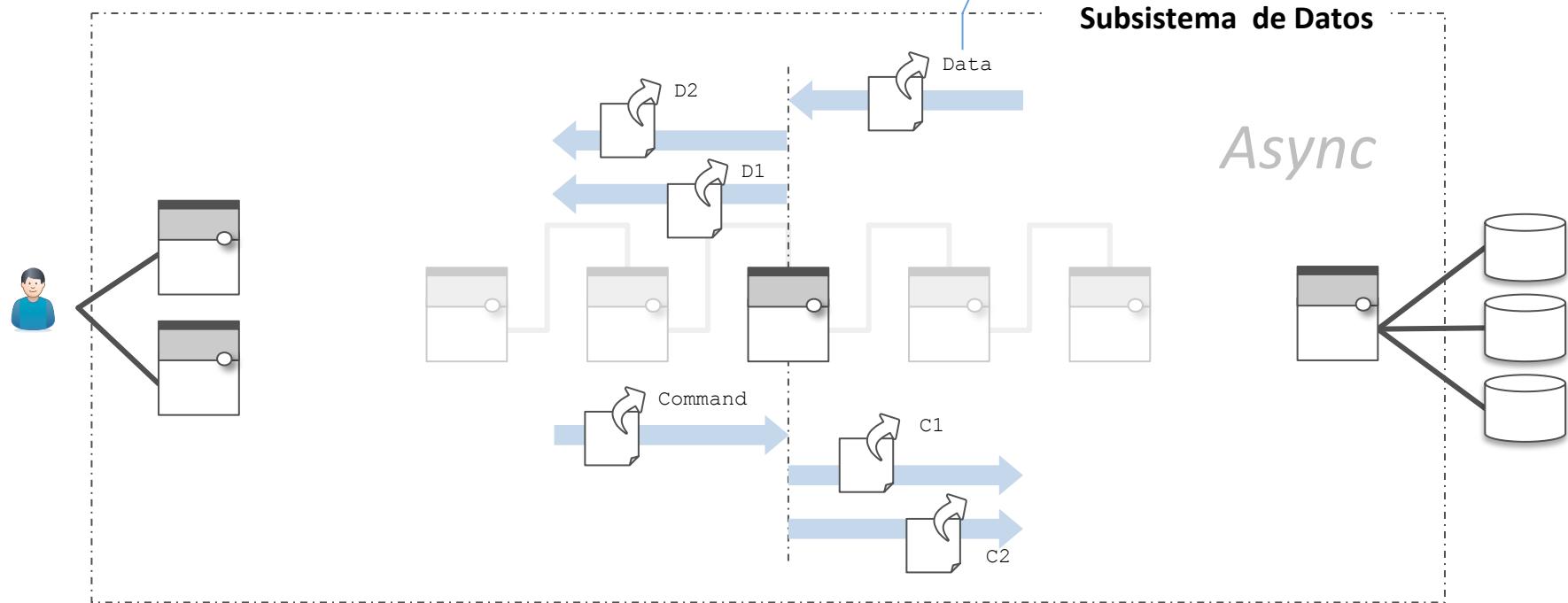
La arquitectura de datos obedece a la necesidad principal de acomodar el modelo de back al esquema de modularidad utilizado en front. Esta transformación marca la cardinalidad de flujos.

Transformación

Los datos de back se adaptan al esquema visual de front y recíprocamente lo que provoca multiplicidad de flujos

Subsistema de Datos

Async



Javier Vélez Reyes

@javiervelezreye

Javier.velez.reyes@gmail.com

2 *Patrones de Diseño en el Acceso a Datos*

- Patrones de Acceso & Activación
- Patrones de Distribución
- Patrones de Transformación
- Patrones de Optimización
- Patrones de Control

Patrones de Acceso a Datos

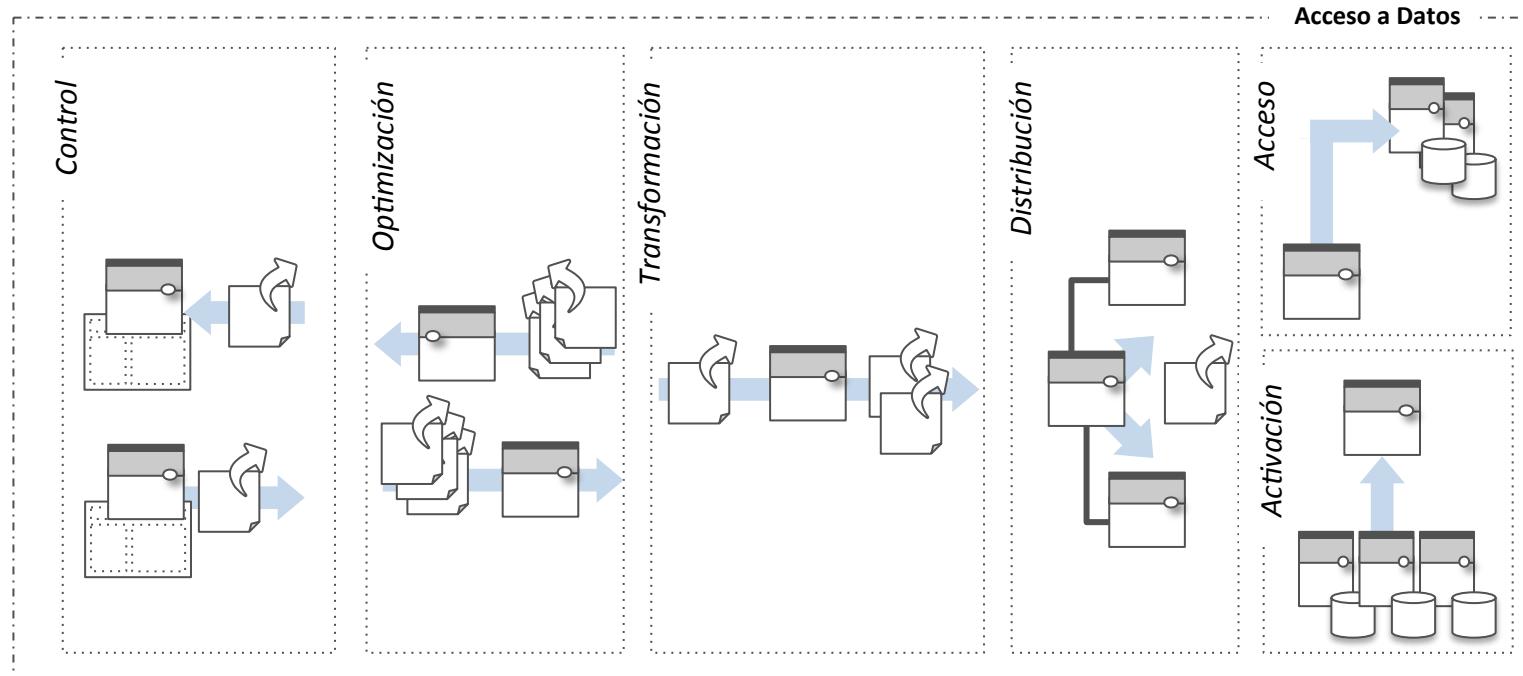
Patrones de Diseño en el Acceso a Datos

Patrones de Acceso a Datos

Patrones de Diseño en el Acceso a Datos

Subsistema de Acceso a Datos

El subsistema de Acceso a Datos se dispone en una serie de niveles que persiguen ciertos principios constructivos. Iremos describiéndolos al presentar cada nivel.



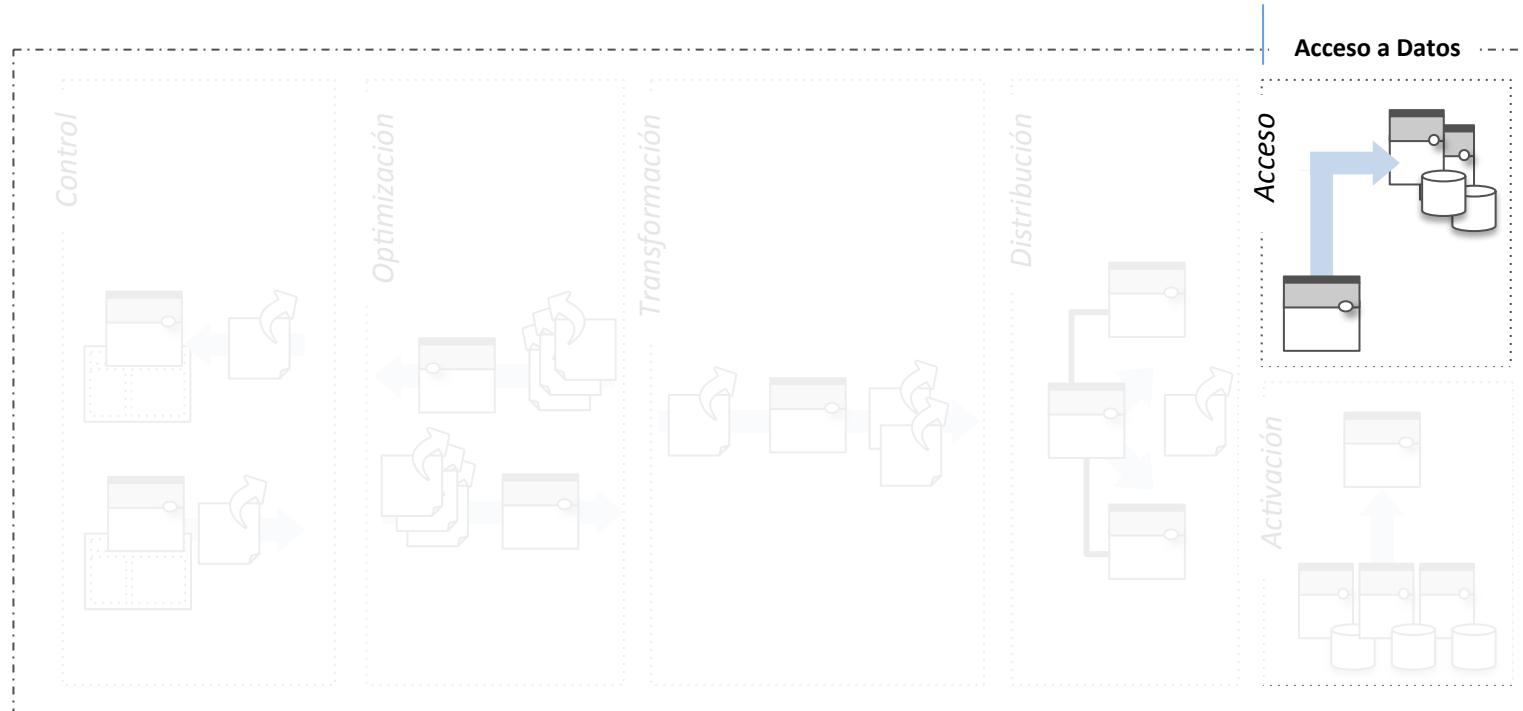
Patrones de Acceso a Datos

Patrones de Diseño en el Acceso a Datos

I. Patrones de Acceso

Nivel de Acceso

Las características particulares de acceso a los distintos tipos de fuentes de datos deberían ser transparentes al desarrollador



Patrones de Acceso a Datos

Patrones de Diseño en el Acceso a Datos

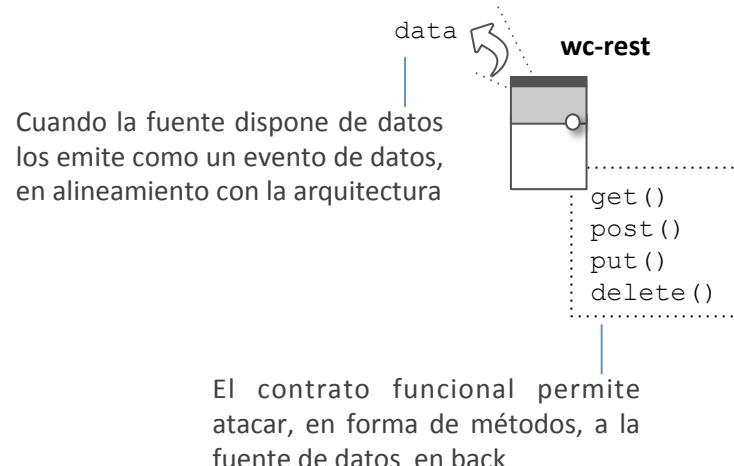
I. Patrones de Acceso



Fuente REST

Se requiere conectar a una fuente de datos en REST. La conexión se hace a través de una solicitud AJAX lo que convierte las llamadas en asíncronas.

- 🎓 El componente wc-rest proporciona un acceso declarativo a las llamadas en REST. El contrato permite operar a cualquiera de los 4 niveles del protocolo [1].



- </> Primero se proporcionan los parámetros por data-binding que configuran la llamada y después se ejecuta el comando.

```
<wc-rest  
base="http://server"  
path="users/[[user]]"  
body="[[data]]"  
action="[[task]]"  
header="[[hds]]">  
</wc-rest>
```

La configuración paramétrica se puede hacer por data-binding. Después solo resta ejecutar la acción

[1]. Richardson Madurity Model - <https://goo.gl/aYIT>

Patrones de Acceso a Datos

Patrones de Diseño en el Acceso a Datos

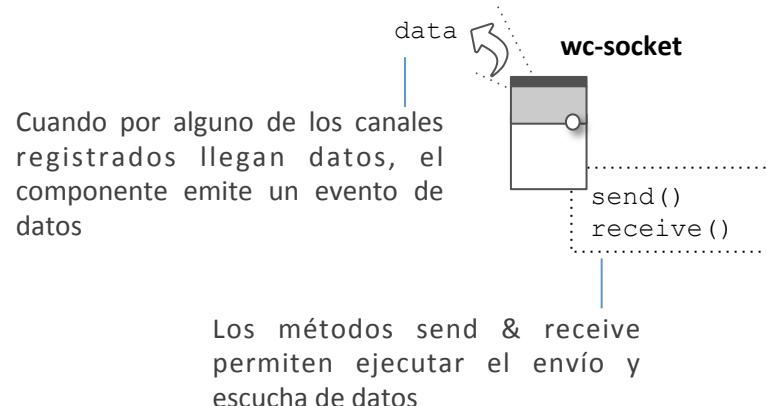
I. Patrones de Acceso



Fuente Web Socket

Se requiere conectar a una fuente de datos con comunicación bidireccional y asíncrona entre cliente y servidor. Para ello se dispone de un servidor que soporta web sockets.

El componente wc-web-socket abre una canal de comunicación bidireccional con el servidor. La comunicación para lectura y escritura se soporta por eventos.



El componente wc-socket abre una canal de comunicación bidireccional con el servidor. La comunicación para lectura y escritura se soporta por eventos.

Primero se proporcionan los parámetros por data-binding y después se ejecuta el comando. En este caso el componente no soporta el modo auto.

```
<wc-socket  
    base="http://server"  
    protocol="soap"  
    channel="[[ch]]"  
    data="[[data]]">  
    <wc-rule channel="ch1"/>  
    <wc-rule channel="ch2"/>  
    <wc-rule channel="ch3"/>  
</wc-socket>
```

La configuración de envío reside en los parámetros (channel) y (data). La de recepción en los hijos light DOM

Patrones de Acceso a Datos

Patrones de Diseño en el Acceso a Datos

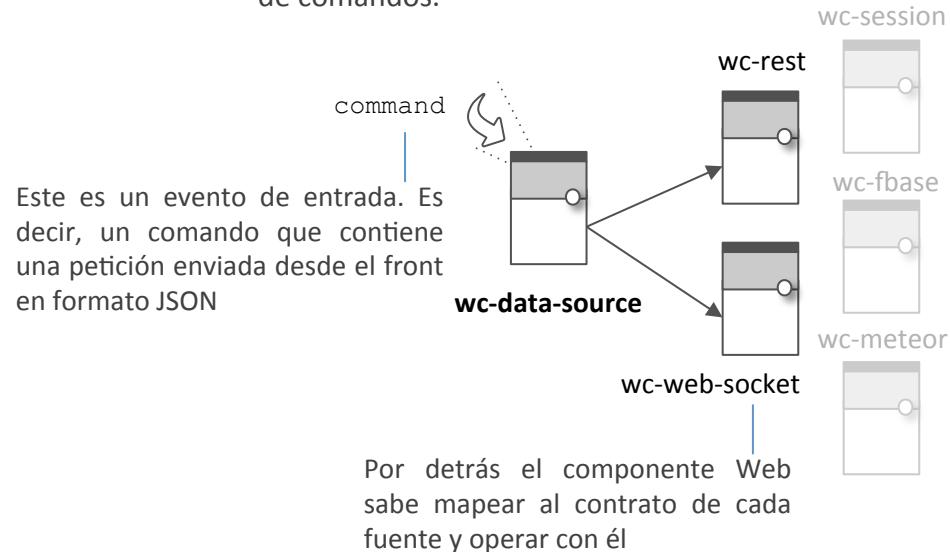
I. Patrones de Acceso



Adaptador Data Source

Se requiere homogeneizar el acceso a fuentes de datos de manera que todas presenten el mismo contrato independientemente de su protocolo.

El componente `wc-data-source` ofrece una adaptación homogénea a cualquier fuente. El contrato está basado en la escucha receptiva de comandos.



El componente recibe la fuente contra la que opera (`source`) y el tipo de fuente (`type`) para saber cómo proceder.

```
<wc-data-source  
    source="#ds"  
    type="wc-rest">  
    <wc-rule from="command.uri" to="uri"/>  
    <wc-rule from="command.data" to="body"/>  
    ...  
</wc-data-source>
```

Las reglas de configuración establecen, en este caso, cómo se traduce el comando entrante al contrato específico de la fuente

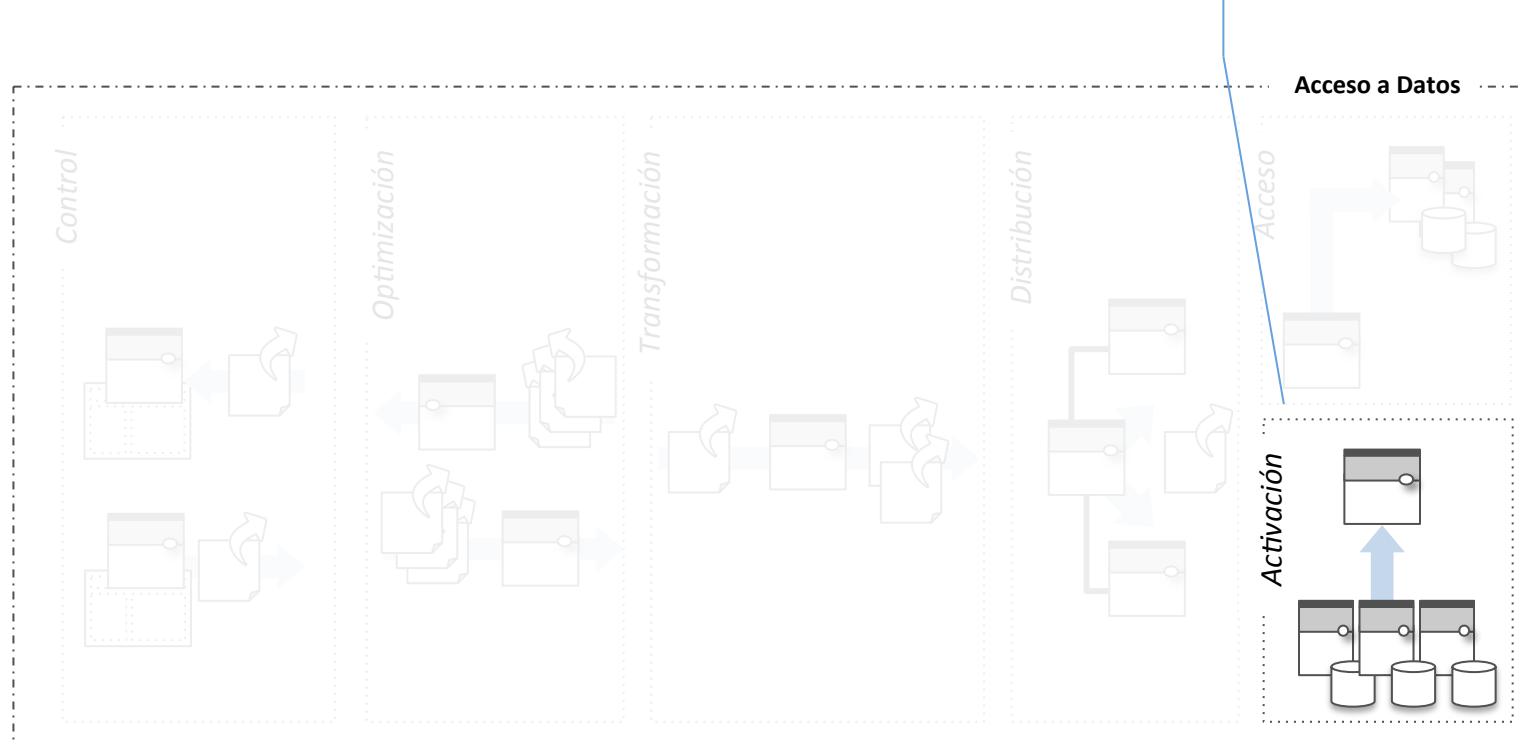
Patrones de Acceso a Datos

Patrones de Diseño en el Acceso a Datos

II. Patrones de Activación

Nivel de Activación

La selección de una fuente de datos y su activación para su posterior uso debería ser una tarea transparente al desarrollador



Patrones de Acceso a Datos

Patrones de Diseño en el Acceso a Datos

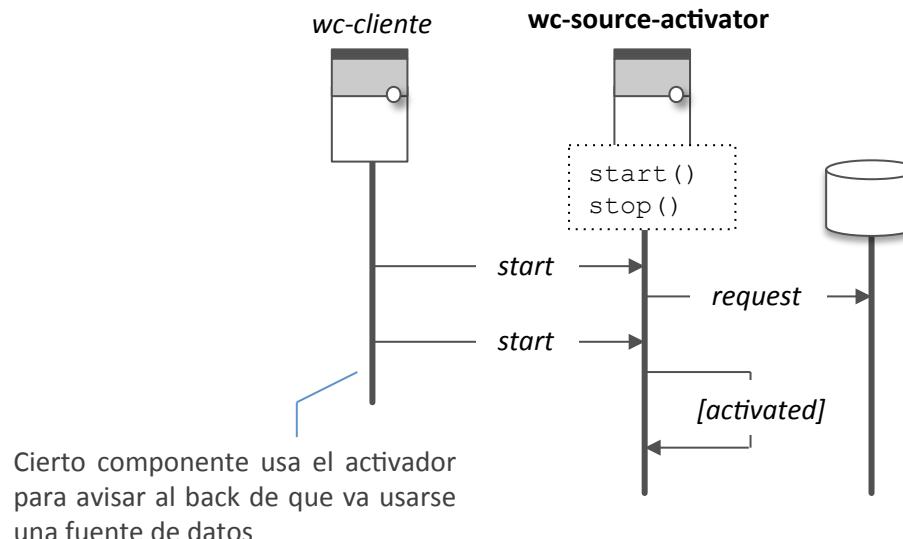
II. Patrones de Activación



Activador de Fuente

Algunas fuentes de datos requieren de cierta lógica de activación desde el front en la fase de arranque y cierta lógica de desactivación en la fase de liberación.

El componente `wc-source-activator` se encarga de lanzar una request a algún servicio en back encargado de ejecutar lógica de activación y liberación de fuentes y servicios de datos.



El componente recibe la fuente contra la que opera (`source`) y el tipo de fuente (`type`) para saber cómo proceder.

```
</> <wc-source-activator source="#ds">
    <wc-rule on="start" do="[[cmd1]]"/>
    <wc-rule on="stop"   do="[[cmd2]]"/>
</wc-data-source>
```

Las reglas definen un ciclo de vida dinámico. Los métodos del componente para ese ciclo de vida se crean en caliente. En este ejemplo usamos la configuración de ciclo de vida más habitual con métodos (`start`) y (`stop`).

Un uso particular es el servicio de login / logout

Patrones de Acceso a Datos

Patrones de Diseño en el Acceso a Datos

II. Patrones de Activación

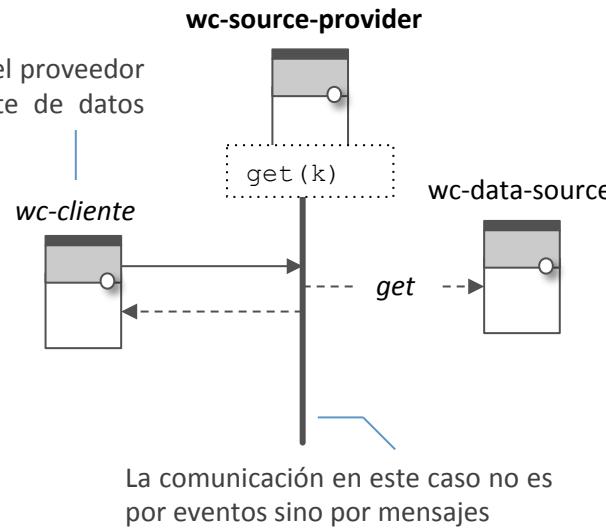


Proveedor de Fuentes

Se utilizan discrecionalmente varias fuentes de datos de diverso tipo y con diversas configuraciones en función de las condiciones ambientales en las que se realiza el acceso.

🎓 El componente `wc-source-provider` organiza por claves diferentes tipos de fuentes con configuraciones específicas y las entrega al cliente.

Cierto componente usa el proveedor para obtener una fuente de datos por clave



</> El proveedor es un registro de fuentes que se recuperan por claves. Las reglas refieren a fuentes definidas en otros puntos del código.

```
<wc-source-provider>
  <wc-rule key="users" source="#ds1"/>
  <wc-rule key="prods" source="[[ds2]]"/>
  ...
</wc-source-provider>
```

El uso de data-binding sobre el parámetro (source) permite gestionar dinámicamente la configuración de este componente

Patrones de Acceso a Datos

Patrones de Diseño en el Acceso a Datos

II. Patrones de Activación

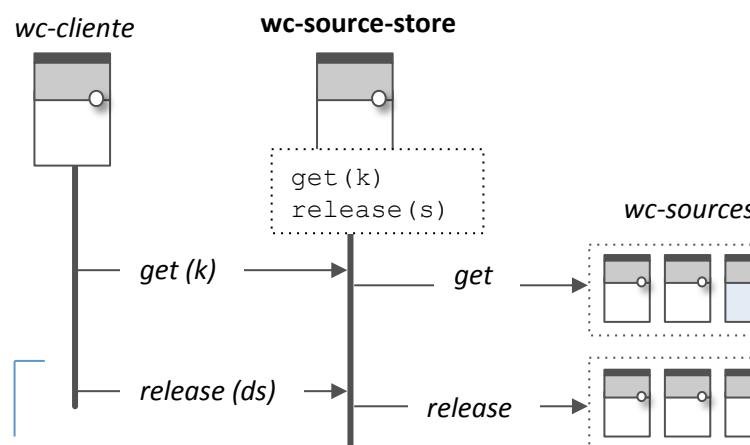


Almacén de Fuentes

Se trabaja con distintas configuraciones de fuentes de datos de forma temporal. Instanciar un componente para cada configuración efímera resulta demasiado costoso.

🎓 El componente crea fuentes de datos genéricas bajo demanda. Cuando éstas se liberan por el cliente, se almacenan en un pool para su reutilización.

</> El almacén de fuentes de datos se configura indicando el número de fuentes que se desea tener disponibles para cada tipo de protocolo usado.



```
<wc-source-store>
  <wc-rule type="wc-rest"    size="3"/>
  <wc-rule type="wc-socket"  size="1"/>
  ...
</wc-source-store>
```

Durante la sesión de trabajo la fuente de datos obtenida está reservada, luego se libera para ser usada por otros clientes

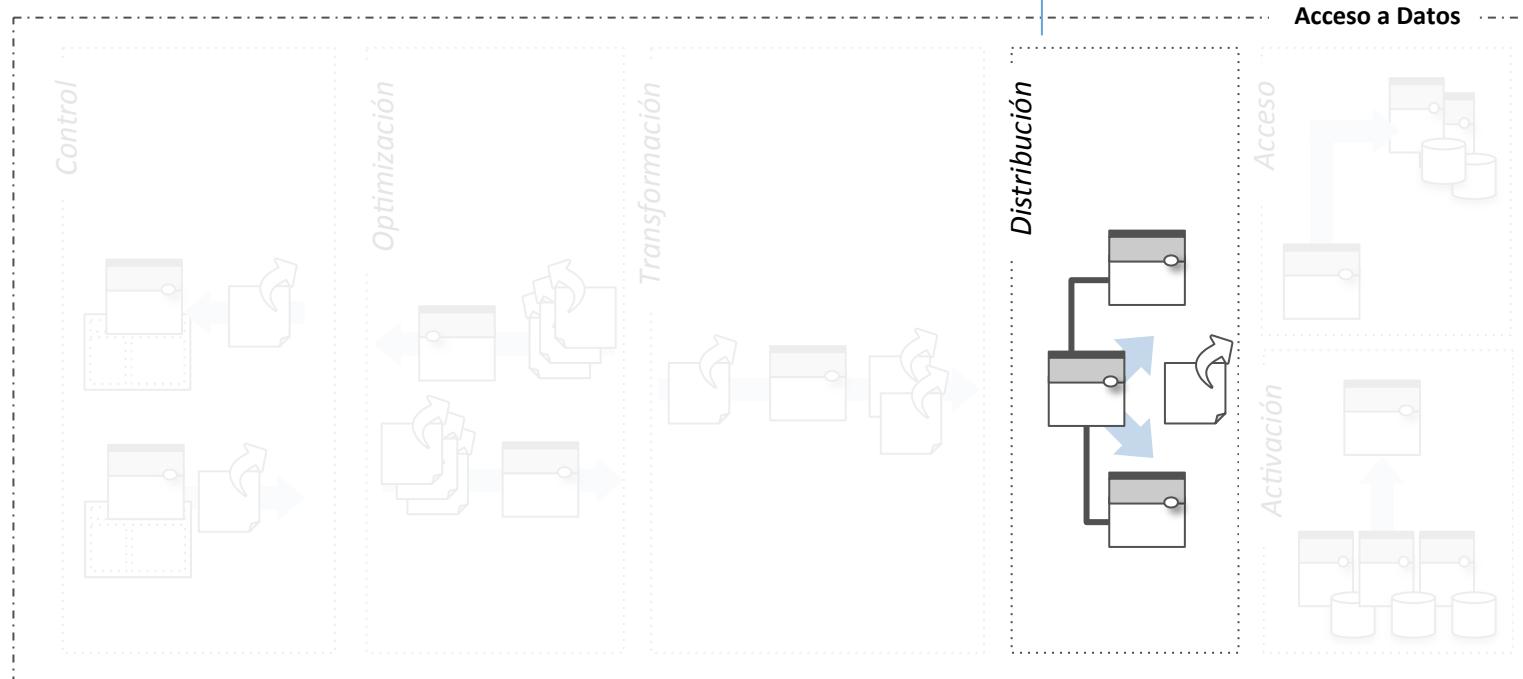
Patrones de Acceso a Datos

Patrones de Diseño en el Acceso a Datos

III. Patrones de Distribución

Nivel de Distribución

Saber distribuir los comandos a cada una de las fuentes de datos con las que operamos debería ser una tarea transparente al desarrollador



Patrones de Acceso a Datos

Patrones de Diseño en el Acceso a Datos

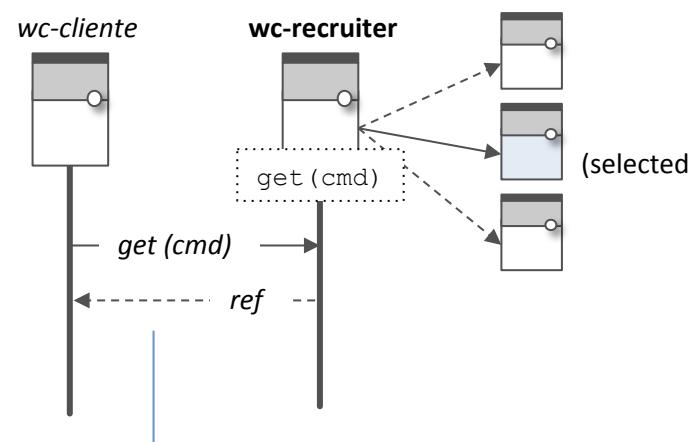
III. Patrones de Distribución



Recruiter

Se requiere operar con diversas fuentes de datos bajo diferentes circunstancias que vienen marcadas por el tipo o contenido del comando o por las condiciones ambientales.

🎓 El componente `wc-recruiter` se encarga de localizar una fuente de datos apropiada para el tipo de comando y la devuelve al cliente.



El componente selecciona una fuente apropiada y la entrega como referencia al cliente para que opere con ella

</> El reclutador es frecuentemente un componente específico de dominio sin configuración aunque pueden encontrarse versiones declarativas como la siguiente.

```
<wc-recruiter data-provider="#p">
  <wc-rule exp="[[x]]" source="key1"/>
  <wc-rule exp="[[y]]" source="key2"/>
  ...
</wc-recruiter>
```

Las reglas aquí sirven para expresar condiciones bajo las cuales se escoge una fuente determinada. Una opción de regla más avanzada es esta...

```
<wc-rule exp="x.size > k" source="key">
  <wc-where value="[[a]]" as="x"/>
  <wc-where value="[[b]]" as="k"/>
</wc-rule>
```

Patrones de Acceso a Datos

Patrones de Diseño en el Acceso a Datos

III. Patrones de Distribución

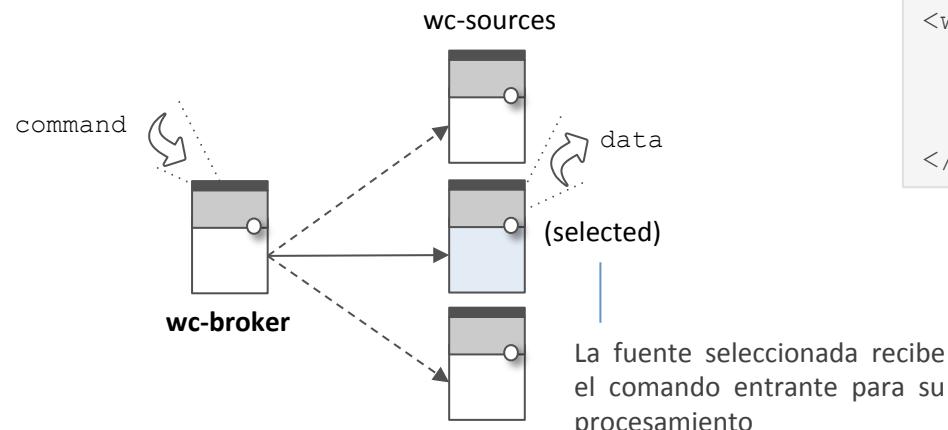


Broker

Se requiere operar con diversas fuentes de datos bajo diferentes circunstancias que vienen marcadas por el tipo o contenido del comando o por las condiciones ambientales.

🎓 El componente wc-broker se encarga de localizar una fuente de datos apropiada para el tipo de comando y le envía dicho comando.

</> El broker es frecuentemente un componente específico de dominio sin configuración aunque pueden encontrarse versiones declarativas como la siguiente.



```
<wc-broker data-provider="#p">
  <wc-rule exp="[[x]]" source="key1"/>
  <wc-rule exp="[[y]]" source="key2"/>
  ...
</wc-broker>
```

Las reglas de configuración operan, en este caso, de forma similar a como lo hacen en el patrón recruiter

Patrones de Acceso a Datos

Patrones de Diseño en el Acceso a Datos

III. Patrones de Distribución

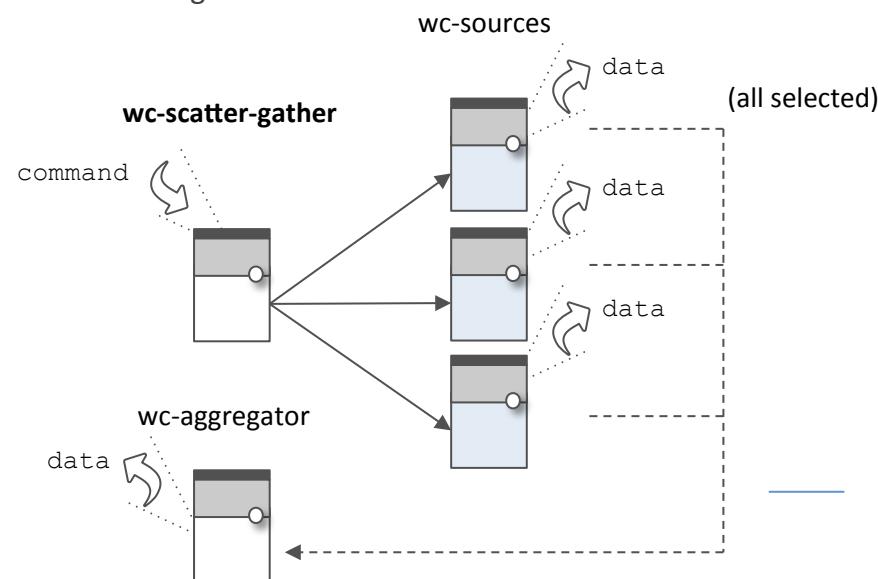


Scatter – Gather

Tu proyecto opera con un conjunto de fuentes similares bajo un esquema de competición. Necesitas enviar una solicitud a todas ellas y extraer una respuesta agregada o selectiva de las mismas.

🎓 El componente `wc-scatter-gather` ofrece este comportamiento devolviendo una solicitud agregada o filtrando la mejor respuesta bajo algún criterio.

</> El componente se configura con reglas que indican la colección de fuentes de datos entre las que se distribuye la solicitud.



```
<wc-scatter-gather data-provider="#p">
  <wc-rule source="key1"/>
  <wc-rule source="key2"/>
  ...
</wc-scatter-gather>
```

Los datos producidos por las fuentes son recolectados por un agregador (ver más adelante). Si se trata de un comando de escritura este flujo no tiene sentido

Patrones de Acceso a Datos

Patrones de Diseño en el Acceso a Datos

III. Patrones de Distribución

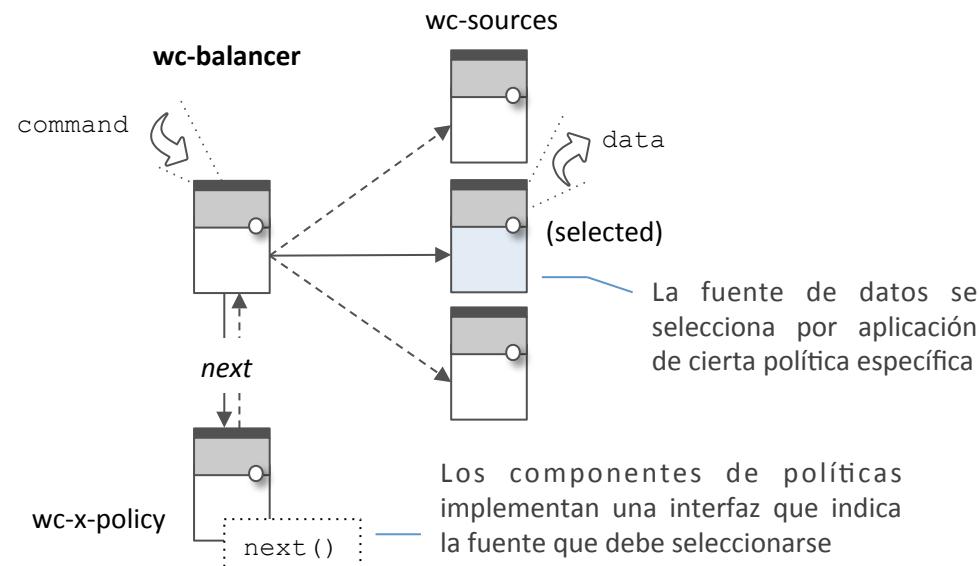


Balancer

Se requiere operar con un conjunto de fuentes similares bajo un esquema de competición. Sin embargo, se pretende aplicar lógica de balanceo de carga para no saturar a ninguna de ellas.

🎓 El componente `wc-balancer` aplica cierta política de balanceo de carga y entrega el comando a la fuente seleccionada.

</> El balanceador se configura especificando un conjunto de fuentes de datos equivalentes y un componente que implementa el interfaz de política.



```
<wc-balancer data-provider="#p">
  <wc-round-robin size="3" policy/>
  <wc-rule source="key1"/>
  <wc-rule source="key2"/>
  ...
</wc-balancer>
```

Tipos de Políticas

- Round – Robin
- Random
- Weighted – Round – Robin
- Weighted – Random
- Stiky

Patrones de Acceso a Datos

Patrones de Diseño en el Acceso a Datos

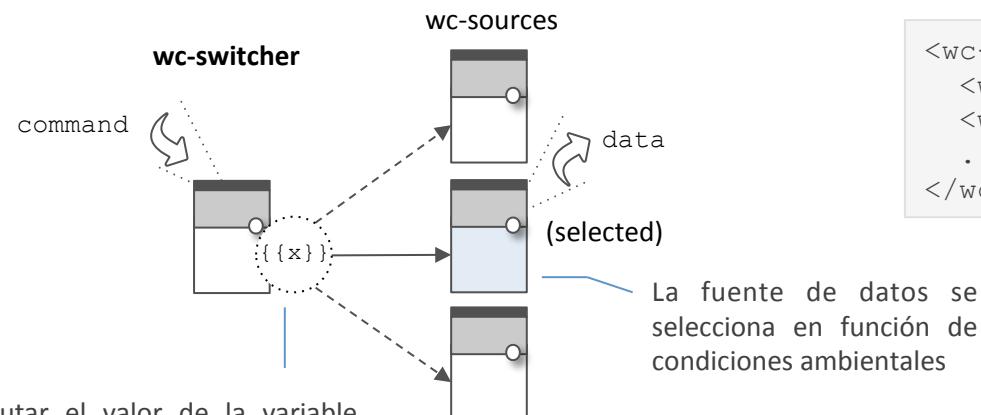
III. Patrones de Distribución



Switcher

Se requiere operar con un conjunto de fuentes similares bajo un esquema de competición. Cada una de ellas se debe utilizar bajo condiciones ambientales y de contexto diferentes.

- El componente wc-switcher selecciona automáticamente la fuente de datos apropiada en función de las condiciones de contorno.



- </> El selector de fuente por contexto se configura indicando la propiedad (key) que debe observarse dentro del contexto ambiental.

```
<wc-switcher key="[[x]]" data-provider="#p">
  <wc-rule when="real" source="key1"/>
  <wc-rule when="fake" source="key2"/>
  ...
</wc-switcher >
```

La fuente de datos se selecciona en función de condiciones ambientales

Las reglas de configuración determinan que fuente debe aplicarse según el valor de la propiedad observada

Un uso particular es la configuración por entornos de desarrollo

Patrones de Acceso a Datos

Patrones de Diseño en el Acceso a Datos

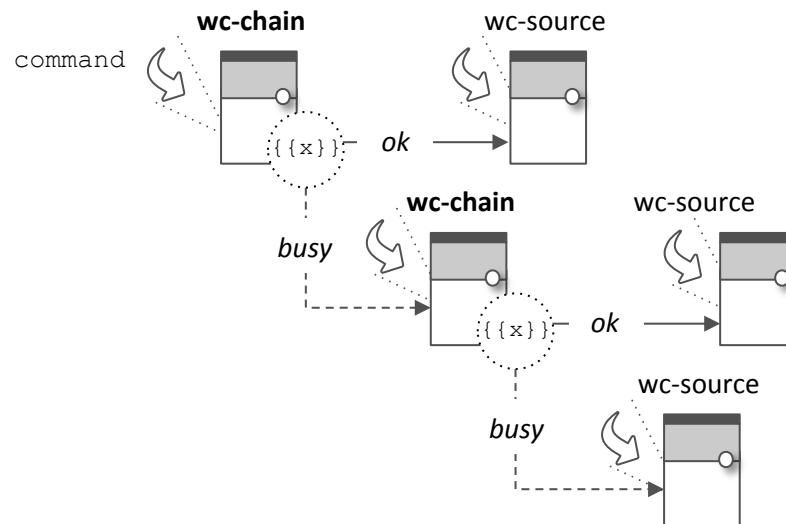
III. Patrones de Distribución



Chain

Se requiere operar con un conjunto de fuentes similares bajo un esquema de competición. Sin embargo, no todas ellas se encuentran disponibles en un momento determinado.

El componente wc-chain establece una cadena de fuentes de manera que si una fuente está disponible procesa el comando y sino, se lo pasa a la siguiente fuente en la cadena.



Distintas instancias de fuente se enlazan formando una cadena de responsabilidad. Si una fuente está ocupada se delega en el siguiente elemento en la cadena.

```
<wc-chain data-provider="#p">
  <wc-rule source="k1" busy="[[b1]]"/>
  <wc-rule source="k2" busy="[[b2]]"/>
  <wc-rule source="k3" default/>
  ...
</wc-data-source>
```

La última regla refiere a una fuente de datos que se encarga de operar cuando las demás han estado ocupadas

Las reglas incluyen un flag para marcar cuando una fuente está ocupada

Patrones de Acceso a Datos

Patrones de Diseño en el Acceso a Datos

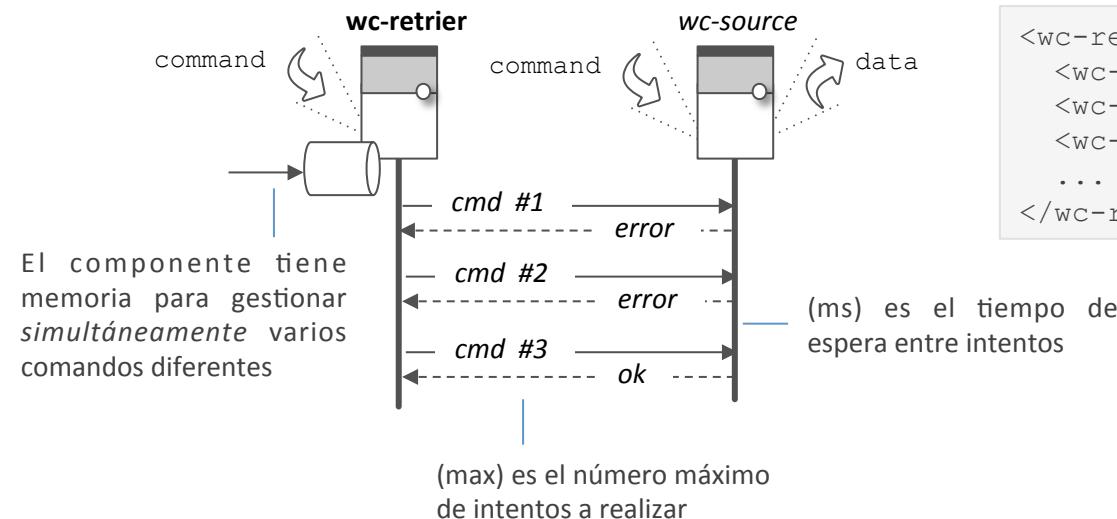
III. Patrones de Distribución



Retrier

Se requiere operar con un conjunto de fuentes de datos. Sin embargo, la conexión o el servicio que le da soporte es propenso a caerse y no ofrece disponibilidad permanente.

🎓 El componente wc-retrier reintenta el ataque a la fuente un número determinado de veces en caso de fallo para aumentar la probabilidad de éxito.



</> El componente opera contra una fuente de datos. Las reglas de configuración indican para cada tipo de comando el número máximo de intentos (max) y la frecuencia de ataque (ms).

```
<wc-retrier source="k1" data-provider="#p">
  <wc-session cache-provider/>
  <wc-rule exp="[[x]]" max="3" ms="300"/>
  <wc-rule exp="[[y]]" max="3" ms="5000"/>
  ...
</wc-retrier>
```

Las expresiones de regla ayudan a evaluar el comando entrante para saber que criterios de reintento aplicarle

Patrones de Acceso a Datos

Patrones de Diseño en el Acceso a Datos

III. Patrones de Distribución

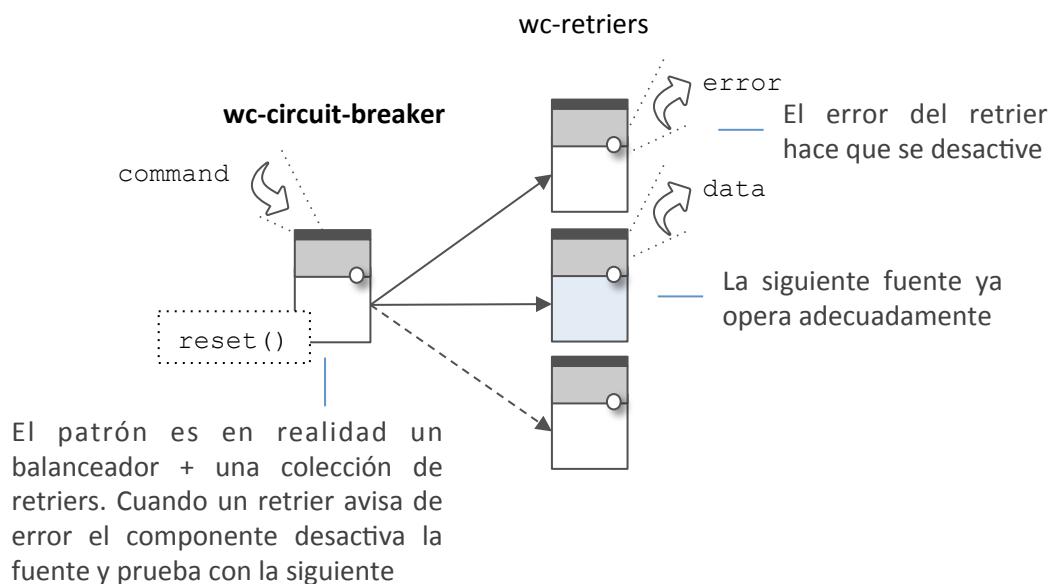


Circuir Breaker

Se requiere operar con un conjunto de fuentes similares bajo un esquema de competición. Sin embargo, éstas son propensas a caerse y denegar el servicio durante largo tiempo.

El componente wc-circuito-breaker detecta estas fuentes inestables y las desconecta temporalmente hasta que vuelven a estar operativas.

El componente se configura con un proveedor y la colección de referencias a fuentes que debe utilizar. Para resetear el estado debe invocarse el método (reset).



```
<wc-circuit-breaker data-provider="#p">
  <wc-rule source="key1" time="30"/>
  <wc-rule source="key2" time="10"/>
  ...
</wc-circuit-breaker>
```

El parámetro tiempo (time) de las reglas indica el tiempo que la fuente debe estar inaccesible. Por conveniencia, éste se expresa en minutos.

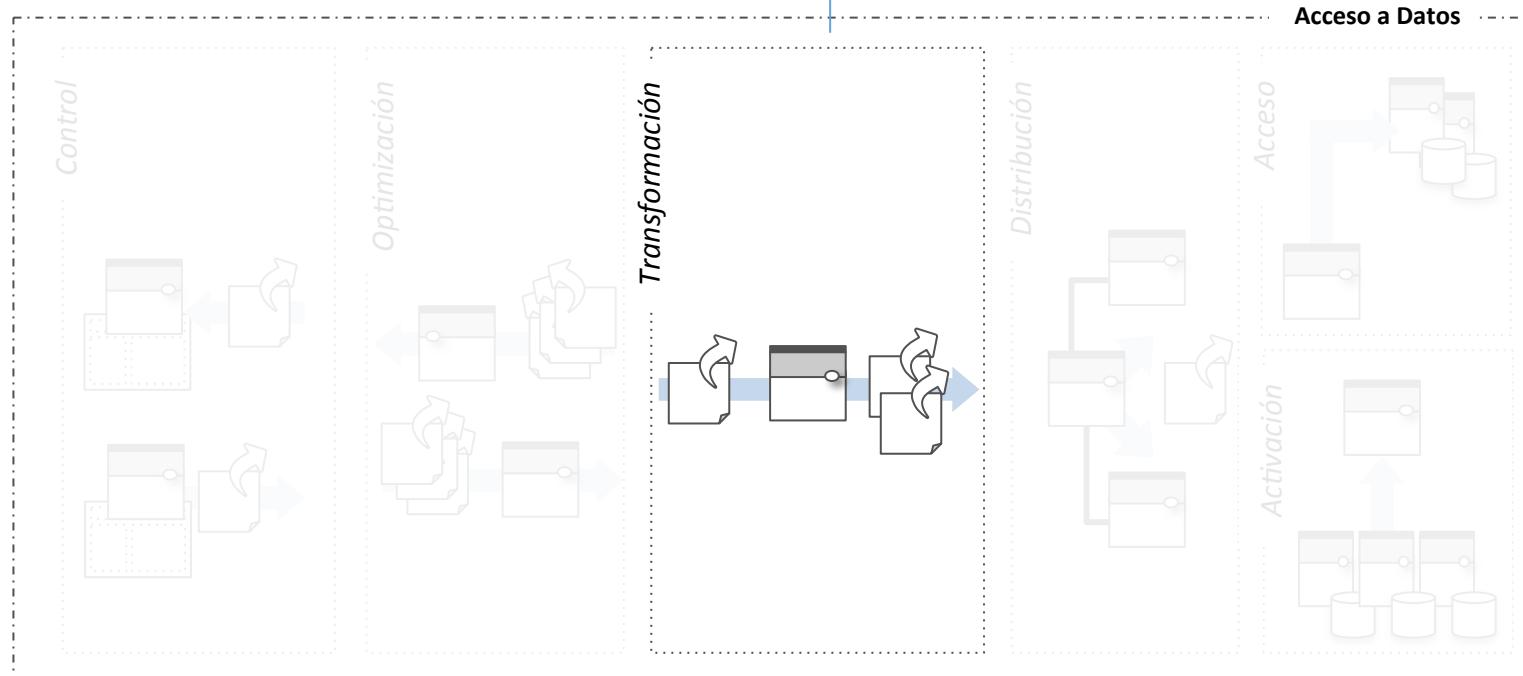
Patrones de Acceso a Datos

Patrones de Diseño en el Acceso a Datos

IV. Patrones de Transformación

Nivel Transformación

Acomodar las impedancias entre los esquemas de datos de las fuentes y los modelos de interacción de front debería ser una tarea transparente al desarrollador



Patrones de Acceso a Datos

Patrones de Diseño en el Acceso a Datos

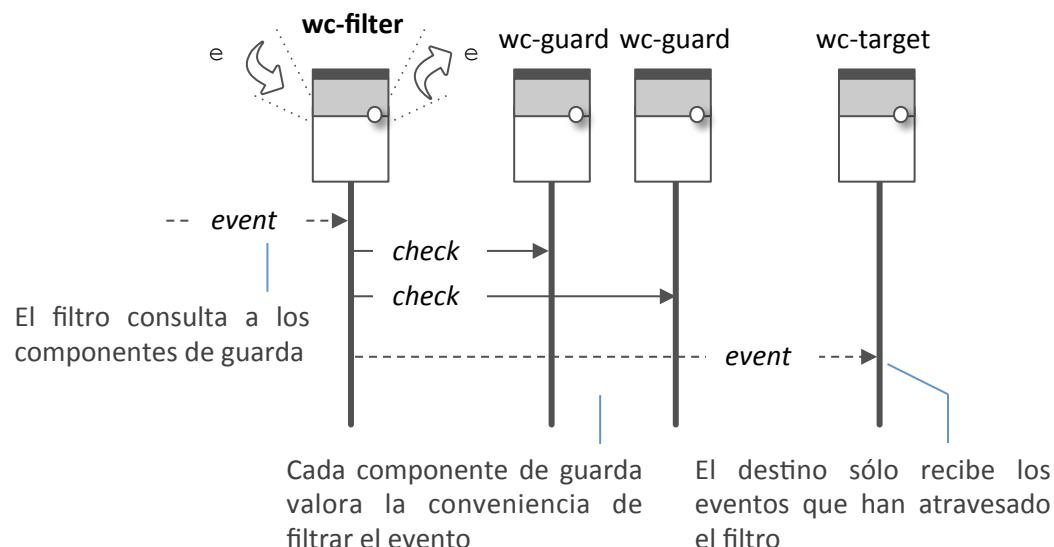
IV. Patrones de Transformación



Filter

Algunos comandos o datos deben descartarse para que no lleguen a su objetivo final. El criterio de filtro depende del contenido o las condiciones ambientales.

- El componente de filtro deja pasar sólo aquéllos eventos que verifican determinada propiedad expresada en componentes de guarda.



- El componente se configura a través de una colección de reglas que refieren a los proveedores de guarda. Su criterio se combina de acuerdo a una lógica AND u OR.

```
<wc-filter and>
  <wc-rule guard="#A"/>
  <wc-rule guard="#B"/>
  ...
</wc-filter>

<wc-exp id="A" exp="x.size > k" >
  <wc-where value="[[a]]" as="x"/>
  <wc-where value="[[b]]" as="k"/>
</wc-exp>
<wc-idempotent path="meta.id"/>
```

Un uso particular permite descartar comandos de escritura repetidos (idempotencia)

Patrones de Acceso a Datos

Patrones de Diseño en el Acceso a Datos

IV. Patrones de Transformación

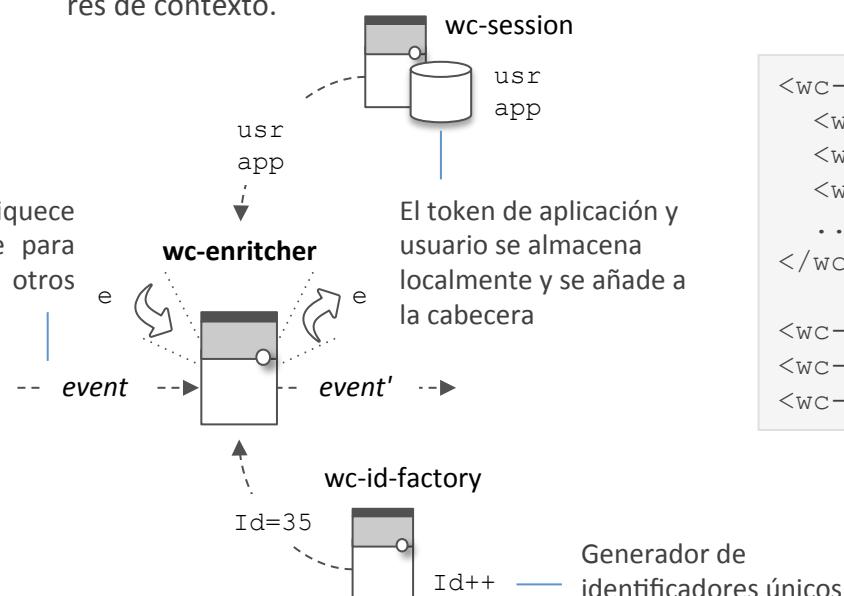


Enricher

Algunos comandos o datos deben enriquecerse antes de que lleguen a su objetivo final. Los datos a incluir provienen del estado de otros componentes o condiciones ambientales.

- El componente wc-enricher se encarga de incluir información en los eventos entrantes a partir de datos proporcionados por proveedores de contexto.

El componente enriquece el evento entrante para pasarle datos de otros componentes



- </> El componente se configura a través de una colección de reglas que refieren a los proveedores de datos cuyas contribuciones serán añadidas al evento de salida.

```
<wc-enricher>
  <wc-rule path="meta.id" from="#A"/>
  <wc-rule path="header.user" from="#B"/>
  <wc-rule path="header.app" from="#C"/>
  ...
</wc-enricher>

<wc-session id="B" key="usr" level="local"/>
<wc-session id="B" key="app" level="session"/>
<wc-id-factory start="[[n]]"/>
```

Patrones de Acceso a Datos

Patrones de Diseño en el Acceso a Datos

IV. Patrones de Transformación

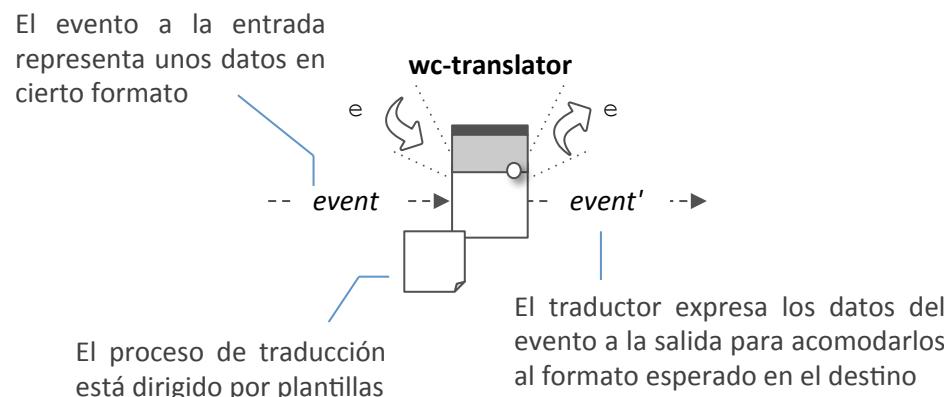


Translator

Algunos comandos o datos se encuentran en un formato incompatible para ser adecuadamente interpretados por su destino. Se requiere una traducción.

El componente wc-translator especifica una plantilla de traducción donde se expresa el formato de salida en función de los datos del evento de entrada.

El componente se configura por medio de la definición de una plantilla. Cualquiera de las plantillas dom- puede utilizarse.



```
<wc-translator>
  <template>
    { header : {} ,
      body   : {
        user : [[name]] ,
        pwd  : [[password]] }
    }
  </template>
</wc-translator>
```

Patrones de Acceso a Datos

Patrones de Diseño en el Acceso a Datos

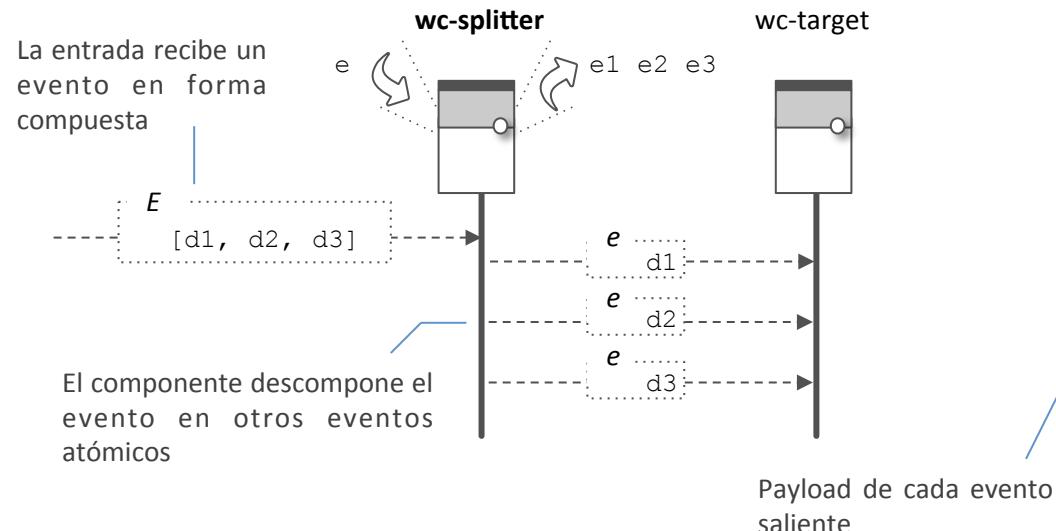
IV. Patrones de Transformación



Splitter

Algunos de los comandos o datos están expresados a un nivel de granularidad demasiado alto y deben ser descompuestos en sus partes atómicas.

- 🎓 El componente wc-splitter se encarga de realizar una descomposición del evento entrante para generar una colección de eventos más sencillos.



- </> La plantilla recorre los elementos de datos del evento entrante y expresa el formato para cada evento de salida.

```
<wc-splitter>
  <wc-rule path="cmd.items" as="item"/>
  <template>{
    owner : [[cmd.user]],
    product : {
      name : [[item.name]]
      id : [[index]],
      amount : 1,
      price : [[item.price]]
    }
  }
</template>
</wc-splitter>
```

Path donde reside la colección a particionar

Patrones de Acceso a Datos

Patrones de Diseño en el Acceso a Datos

IV. Patrones de Transformación

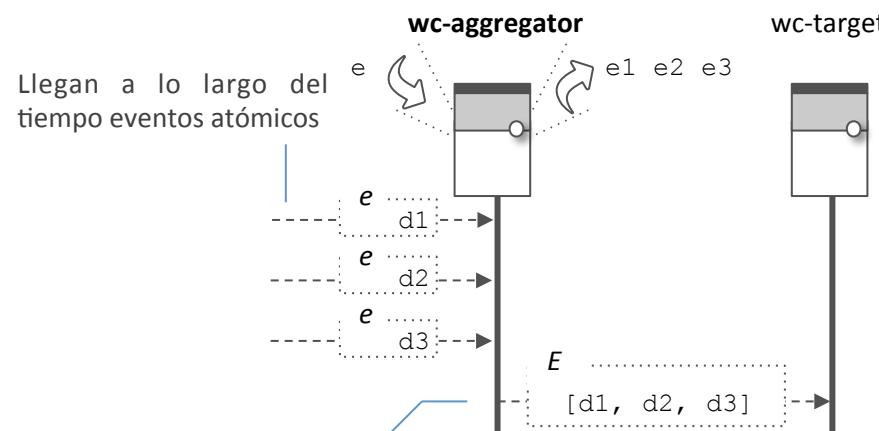


Aggregator

Algunos de los comandos o datos están expresados a un nivel de granularidad demasiado bajo y deben ser agregados en una estructura compuesta.

El componente wc-aggregator se encarga de realizar una composición de una colección de eventos atómicos entrantes para generar un evento compuesto.

</> La plantilla se interpreta en sentido inverso al caso anterior. Construye un esquema complejo desde una colección de items recolectados en el tiempo.



El componente analiza el identificador de secuencia para saber cuando emitir el evento compuesto

```
<wc-aggregator>
  <template>
    cmd : 'basket',
    <template
      is='dom-repeat'
      items='{{items}}'>{
        owner : [[user.name]],
        products : [[data.prods]]
      }
    </template>
  </template>
</wc-aggregator>
```

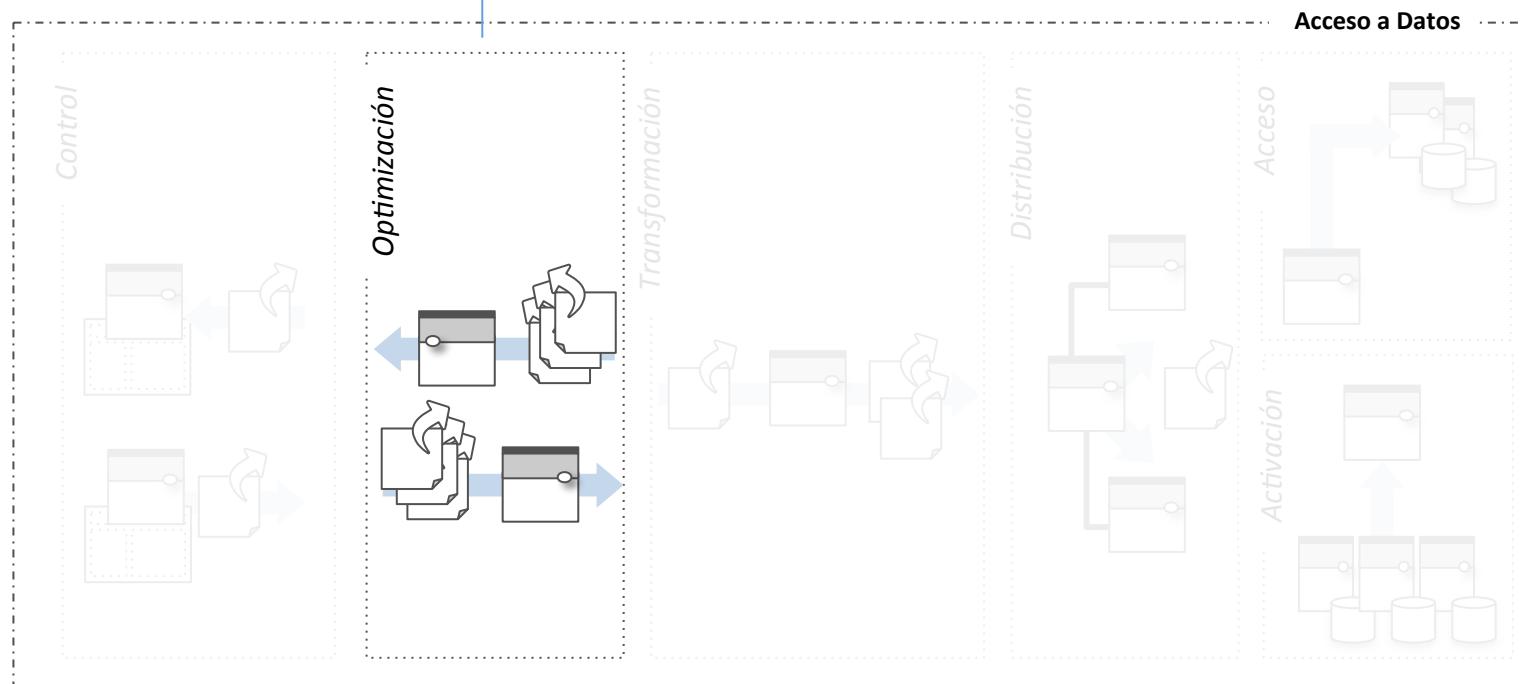
Patrones de Acceso a Datos

Patrones de Diseño en el Acceso a Datos

V. Patrones de Optimización

Nivel de Optimización

Optimizar el acceso a las fuentes de datos debería ser una tarea transparente al desarrollador



Patrones de Acceso a Datos

Patrones de Diseño en el Acceso a Datos

VI. Patrones de Optimización

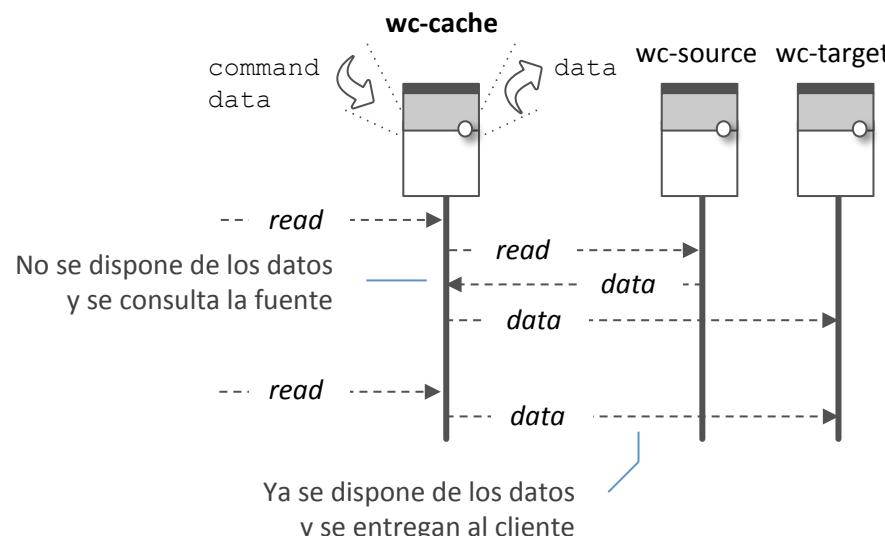


Cache

Algunos comandos de lectura se repiten recurrentemente a lo largo de la sesión de trabajo lo que provoca una sobrecarga de las fuentes de datos.

🎓 El componente wc-cache es una memoria intermedia de acceso rápido que descarga de trabajo a las fuentes cuando el dato se ha cargado recientemente.

</> El componente se configura con un proveedor de persistencia e información acerca del tiempo de vida que residen los datos en memoria.



```
<wc-cache time="30" path="cmd.query">  
  <wc-session cache-provider/>  
</wc-cache>
```

El parámetro tiempo (time) expresa el tiempo de vida en minutos de los datos en cache

La ruta (path) dice a la cache qué dato del comando de lectura corresponde a la clave con que se han persistido los datos

Patrones de Acceso a Datos

Patrones de Diseño en el Acceso a Datos

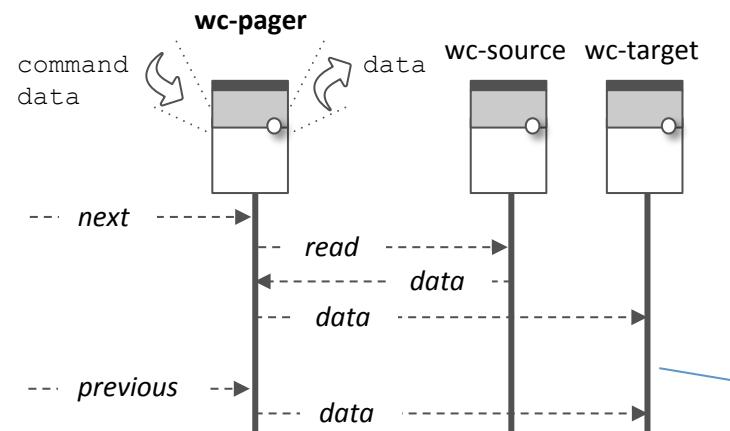
V. Patrones de Optimización



Pager

Algunas fuentes exponen sus colección de datos de acuerdo a una lógica de paginación. El acceso a esta información debería realizarse de forma fluida haciendo transparente dicha lógica.

🎓 El componente wc-pager es un apoderado que permite explorar colecciones paginadas de forma transparente ofreciendo una sensación de continuidad en los datos.



</> El componente paginador se configura indicando el proveedor de persistencia y el tiempo de vida de las páginas.

```
<wc-pager time="30"  
          path="cmd.page"  
          start="1">  
  <wc-session cache-provider/>  
</wc-pager>
```

El comando se encarga de gestionar el número de página y enriquecer el payload

En este caso (start) indica la página de comiendo.

El paginador también funciona como una cache a nivel de página de manera que sólo consulta la fuente cuando no dispone de la página en local

Patrones de Acceso a Datos

Patrones de Diseño en el Acceso a Datos

V. Patrones de Optimización

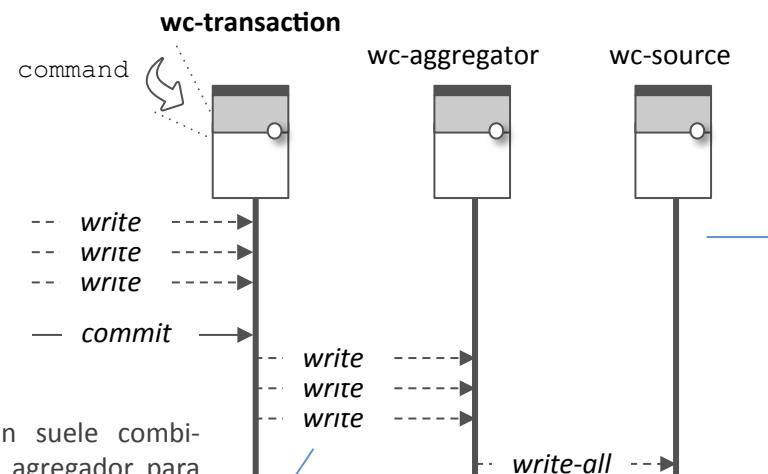


Transaction

Algunos tipos de datos requieren un tratamiento transaccional aunque sus fuentes no lo soporten. Se requiere emular la transaccionalidad desde el front.

🎓 El componente wc-transaction retiene el procesamiento de los comandos entrantes para que se procesen en bloque sólo a la llegada de una señal.

</> El componente transacción se configura indicando el proveedor de persistencia donde se acumulan los eventos antes de enviarse.



```
<wc-transaction>
  <wc-session cache-provider/>
  <wc-rule when='[[exp]]' commit/>
</wc-transaction>
```

Todos los comandos se persisten localmente a la espera de que llegue una señal explícita de commit

La expresión (exp) ayuda al componente a identificar los comandos que corresponden a sentencias commit

Patrones de Acceso a Datos

Patrones de Diseño en el Acceso a Datos

V. Patrones de Optimización

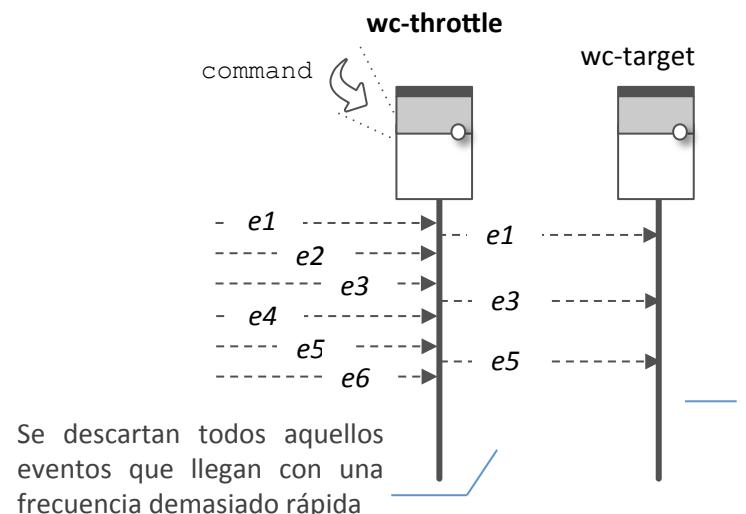


Throttle

Algunas fuentes de datos operan a un ritmo más lento del que los datos son generados en front. Se requiere un mecanismo para no saturar a la fuente basado en el descarte de datos.

🎓 El componente wc-throttle se encarga de garantizar que cualquier comando que se emita demasiado próximo en tiempo a otro anterior sea descartado.

</> El componente se configura indicando el tiempo en milisegundos que debe aplicarse como umbral para descartar eventos.



```
<wc-throttle ms="[[t]]">  
</wc-throttle>
```

Este patrón se aplica bajo la hipótesis de que la pérdida de eventos no es significativa para el problema

Patrones de Acceso a Datos

Patrones de Diseño en el Acceso a Datos

V. Patrones de Optimización

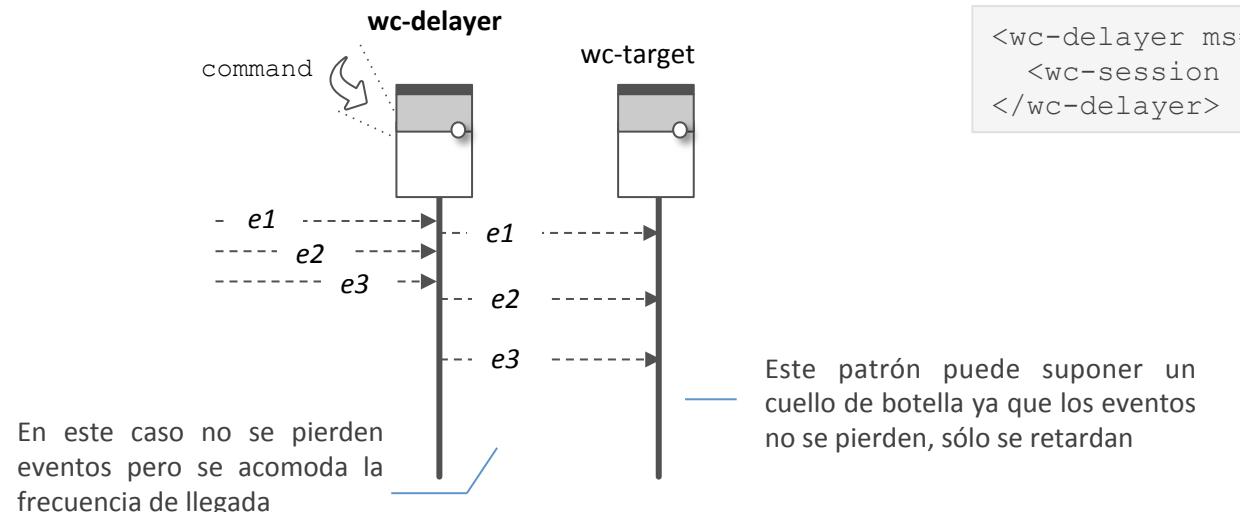


Delayer

Algunas fuentes de datos operan a un ritmo más lento del que los datos son generados en front. Se requiere un mecanismo para no saturar a la fuente basado en el encolado de datos.

El componente wc-delayer introduce un retardo temporal fijo en todos los comandos que provienen desde el front para no saturar a la fuente.

El componente se configura indicando el tiempo en milisegundos que debe retardarse la propagación de eventos y un proveedor de persistencia.



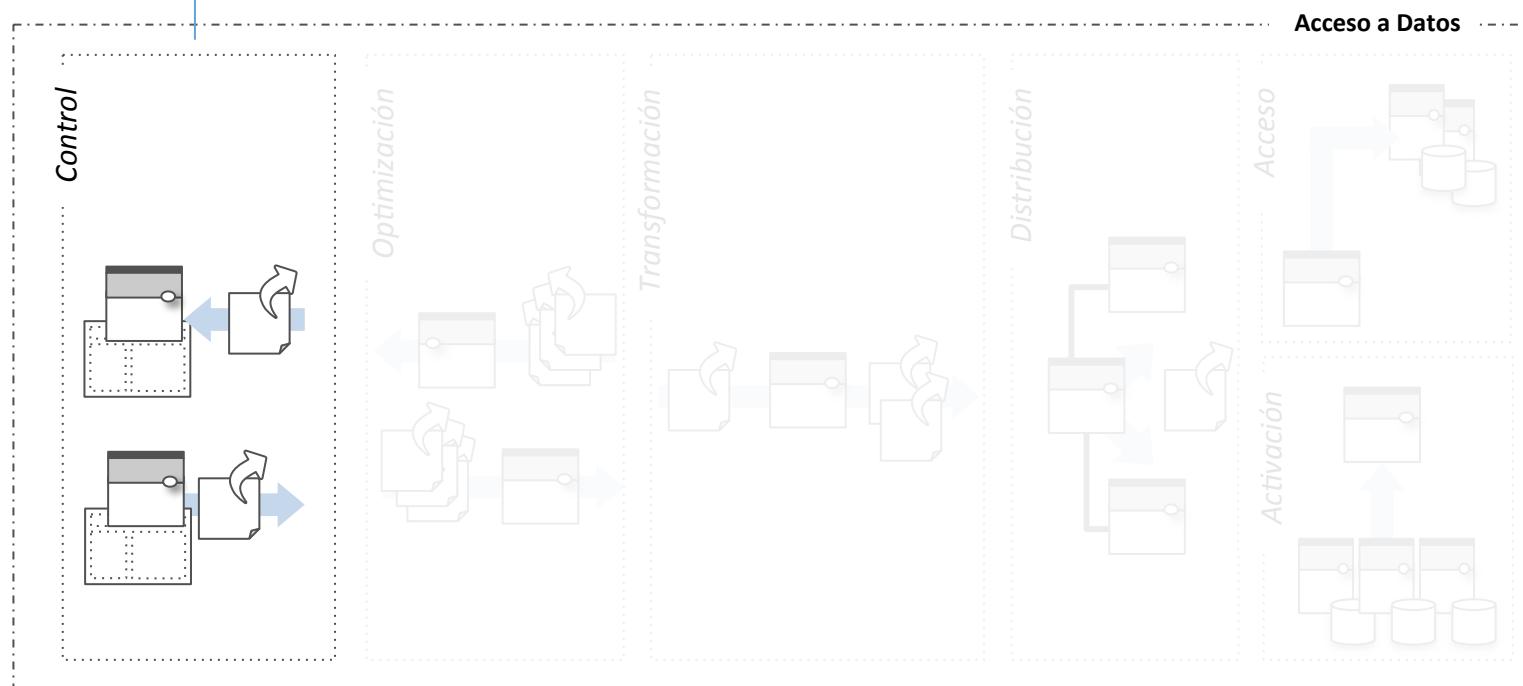
Patrones de Acceso a Datos

Patrones de Diseño en el Acceso a Datos

VI. Patrones de Control

Nivel de Control

Los mecanismos de integración entre las vistas (componentes UI) y las fuentes de datos (componentes de datos) deberían ser transparentes al desarrollador



Patrones de Acceso a Datos

Patrones de Diseño en el Acceso a Datos

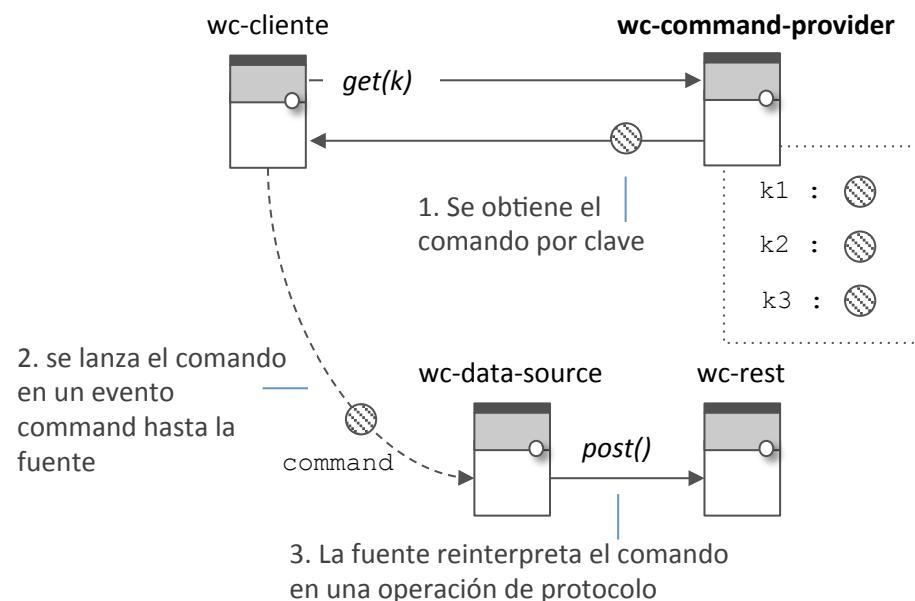
VI. Patrones de Control



Command Provider

Diferentes tipos de fuentes de datos utilizan distintos protocolos de acceso. Se requiere abstraer su forma de uso para que operen de forma homogénea.

🎓 Este componente encapsula la complejidad inherente a un protocolo de uso de una fuente en un mensaje JSON (comando) que se propaga como evento en la arquitectura.



⟨/⟩ El componente se configura definiendo los comandos por propiedades e indicando el contexto donde se evalúan las propiedades (:)

```
<wc-command-provider context="[[ctx]]">
  <wc-rule command="cinemas">
    <wc-property path="cmd.verb" value="get"/>
    <wc-property path="cmd.uri" value="/cinemas"/>
  </wc-rule>
  <wc-rule command="movies">
    <wc-property path="cmd.verb" value="get"/>
    <wc-property path="cmd.uri"
      value="/cinemas/[c]/movies"/>
  </wc-rule>
  <wc-rule command="sessions">
    <wc-property path="cmd.verb" value="get"/>
    <wc-property path="cmd.uri"
      value="/cinemas/[c]/movies/[m]/sessions"/>
  </wc-rule>
  ...
</wc-command-provider>
```

Patrones de Acceso a Datos

Patrones de Diseño en el Acceso a Datos

VI. Patrones de Control

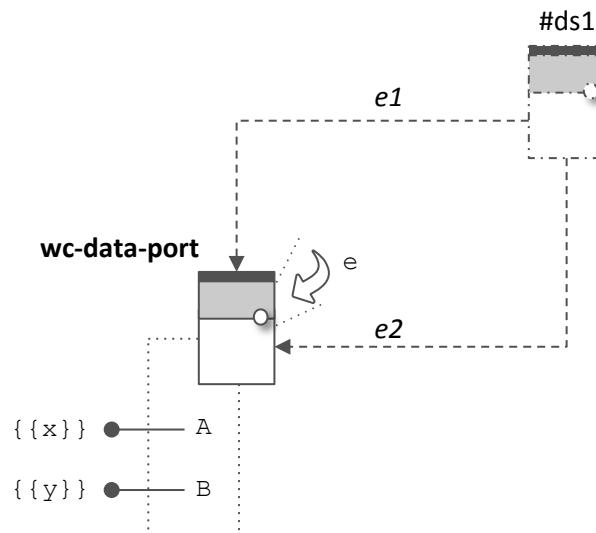


Data Port

Distintos datos que llegan desde las fuentes de datos en forma de eventos deben ser retenidos para poder ser consultados posteriormente en cualquier momento.

🎓 El componente wc-data-port se encarga de dar acceso a los datos de un canal de eventos en una propiedad de data-binding.

</> El componente se configura indicando la fuente de datos (source) a escuchar. El (path) es la ruta en el evento de datos que determina en función de (when) donde depositar el dato.



```
<wc-data-port source="#ds1" path="...">
  <wc-rule when="A" data="{{d1}}"/>
  <wc-rule when="B" data="{{d2}}"/>
  <wc-rule when="C" data="{{d3}}"/>
</wc-data-port>
```

Patrones de Acceso a Datos

Patrones de Diseño en el Acceso a Datos

VI. Patrones de Control



Data Driven Client

Se requiere explorar en profundidad una fuente de datos orientada a recursos. El progreso de la exploración queda controlado por propiedades en data-binding.

- El componente proporciona un mecanismo explícito para realizar la exploración controlada por variables y centralizar los comandos de ataque.

wc-data-driven-client



- </> El componente se configura con un proveedor de comandos y una colección de reglas que indican el avance en la exploración cada vez que una nueva variable toma valor.

```
<wc-data-client command-provider="#p">
  <wc-rule command="cinemas"/>
  <wc-rule command="movies" when="[[c]]"/>
  <wc-rule command="sessions" when="[[m]]"/>
</wc-data-client>
```

Patrones de Acceso a Datos

Patrones de Diseño en el Acceso a Datos

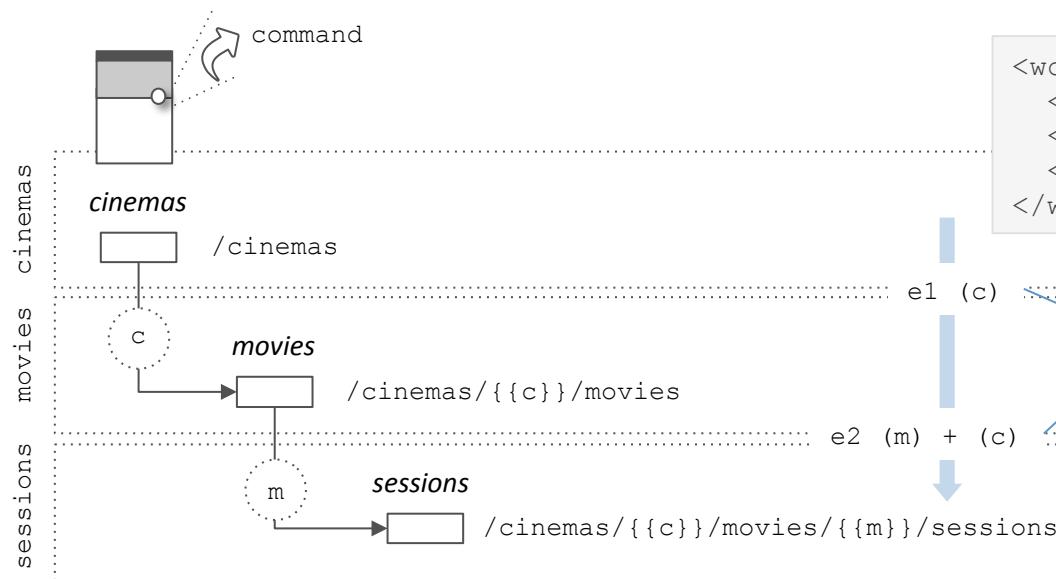
VI. Patrones de Control



Event Driven Client

Se requiere explorar en profundidad una fuente de datos orientada a recursos. El progreso de la exploración queda controlado por eventos.

- El componente mantiene el estado de la navegación y ofrece un mecanismo explícito de realizar la exploración controlada por **wc-event-driven-client** eventos.



- </> El componente se configura con un proveedor de comandos y una colección de reglas que indican el avance en la exploración cada vez que llega un nuevo evento.

```
<wc-data-client command-provider="#p">
<wc-rule command="cinemas"/>
<wc-rule command="movies" on="e1"/>
<wc-rule command="sessions" on="e2"/>
</wc-data-client>
```

Las variables extraídas del evento determinan el progreso en la incursión

Se explora en profundidad un espacio de recursos

Patrones de Acceso a Datos

Patrones de Diseño en el Acceso a Datos

VI. Patrones de Control

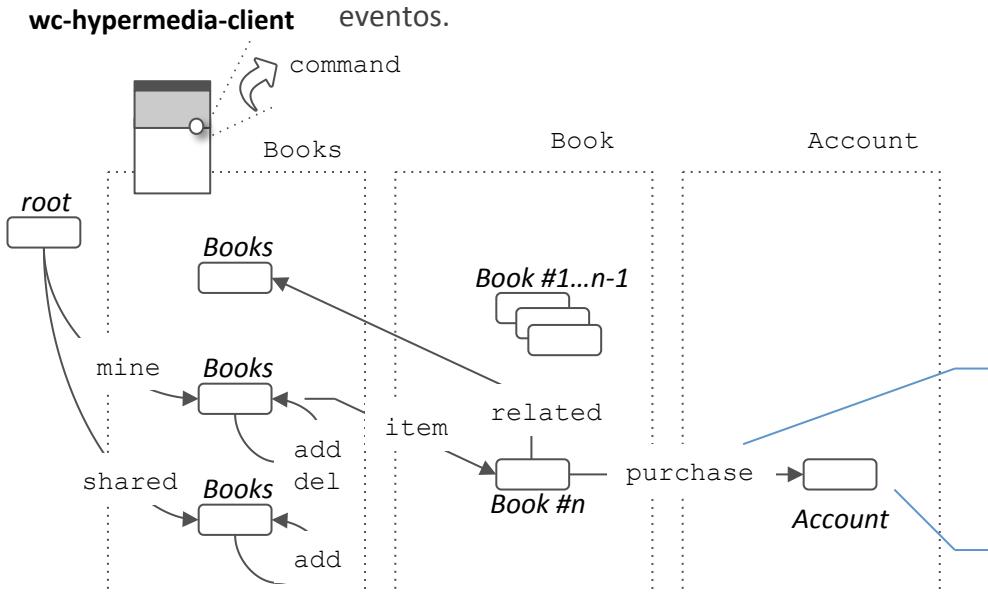


Hypermedia Client

Se requiere explorar una fuente de datos hypermedia. La exploración queda controlada por un conjunto de eventos que representan movimientos sobre el grafo de recursos.

🎓 El componente mantiene el estado de la navegación sobre el espacio hypermedia de recursos y controla su progreso dirigido por eventos.

</> El comando sólo requiere una referencia al estado/recurso inicial. El resto de estados y sus comandos asociados se extraen de la respuesta (response).



```
<wc-hypermedia-client command-provider="#p">
<wc-rule start-command="root"/>
<wc-rule response="{{data}}"/>
</wc-hypermedia-client>
```

Cada relación es un comando cuya estructura está en la respuesta del comando anterior

Cada recurso representa un estado de la aplicación

Patrones de Acceso a Datos

Patrones de Diseño en el Acceso a Datos

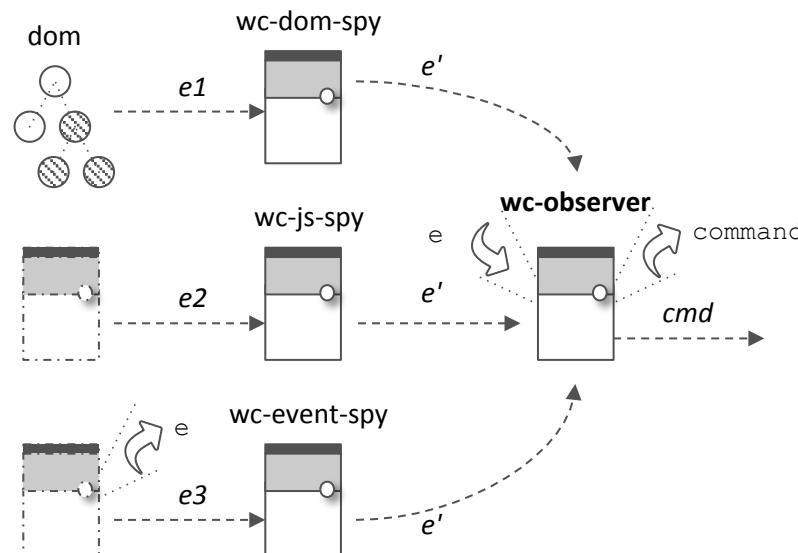
VI. Patrones de Control



Observer

Se requiere reaccionar contra una fuente de datos cada vez que se produce un cambio asíncrono en el contexto ambiental donde se desarrolla el front.

El componente se mantiene a la escucha de los cambios de las condiciones ambientales que son objeto de interés reactivo.



El componente se configura con un agente de escucha (spy) que lanza eventos cada vez que se producen cambios.

```
<wc-observer command-provider="#p">
  <wc-dom-spy path="css" spy/>
  <wc-js-spy target="{{wc}} spy/>
  <wc-event-spy target="{{wc}} spy/>

  <wc-rule on="e1" command="k1"/>
  <wc-rule on="e2" command="k2"/>
  <wc-rule on="e3" command="k3"/>
</wc-observer>
```

Las reglas indican con qué comando debe reaccionarse para cada tipo de evento

3 de los ejemplos de espía más comunes

Javier Vélez Reyes

@javiervelezreye

Javier.velez.reyes@gmail.com

3

El Subsistema de Acceso a Datos en la Práctica

- Escenario de e-commerce
- Arquitectura
- Flujos de Iteración

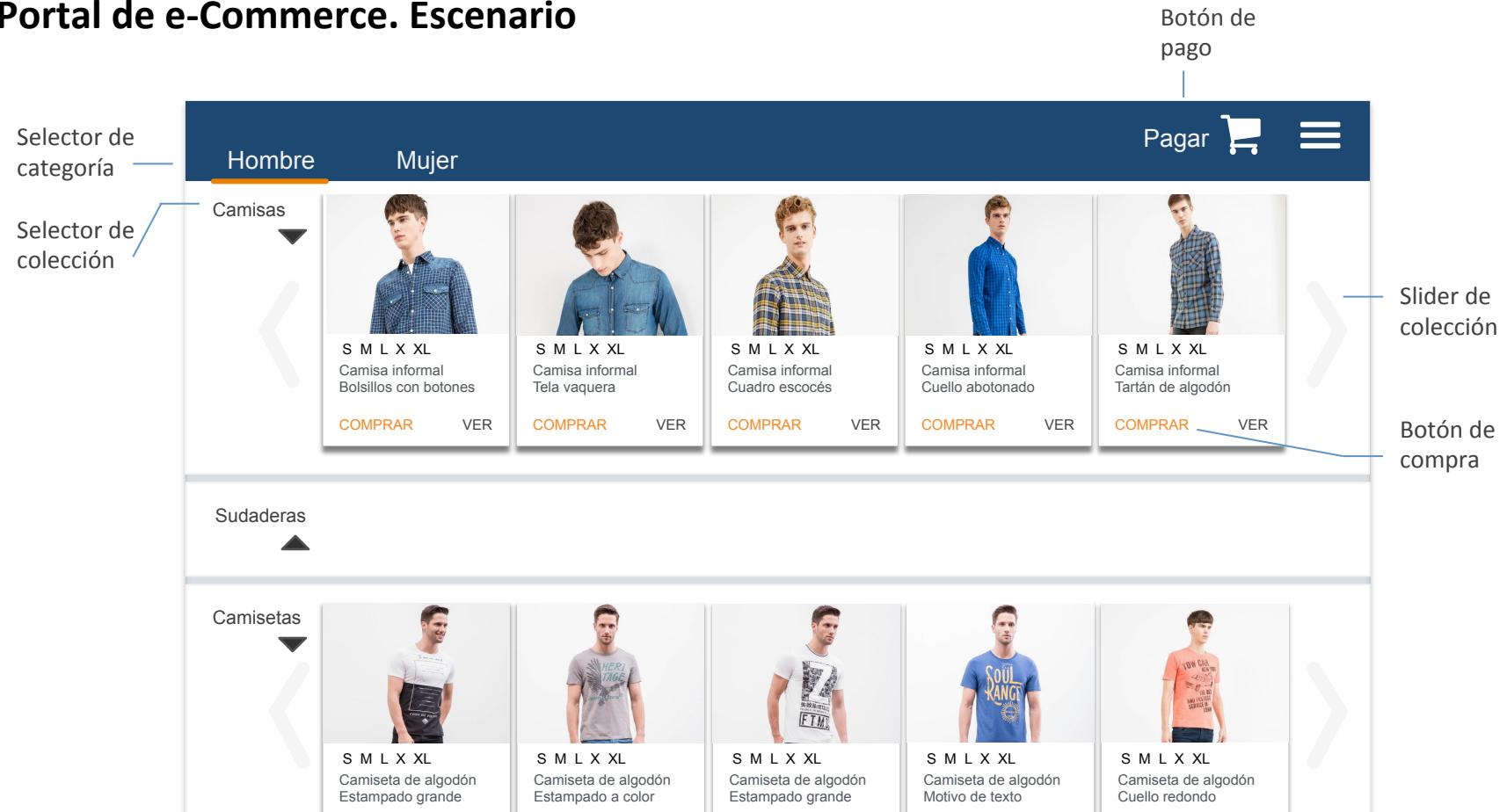
Patrones de Acceso a Datos

El Subsistema de Acceso a Datos en la Práctica

Patrones de Acceso a Datos

El Subsistema de Acceso a Datos en la Práctica

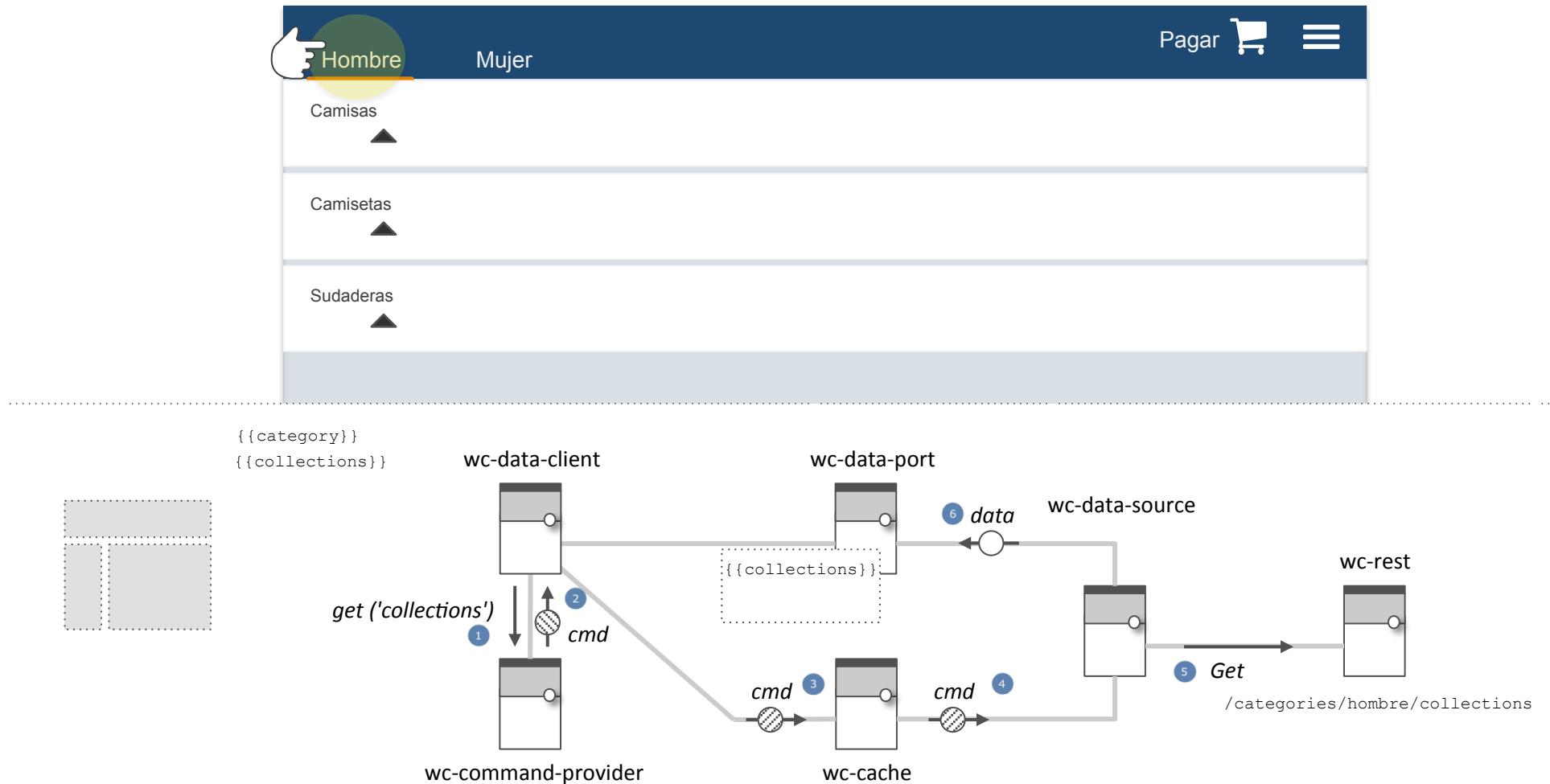
I. Portal de e-Commerce. Escenario



Patrones de Acceso a Datos

El Subsistema de Acceso a Datos en la Práctica

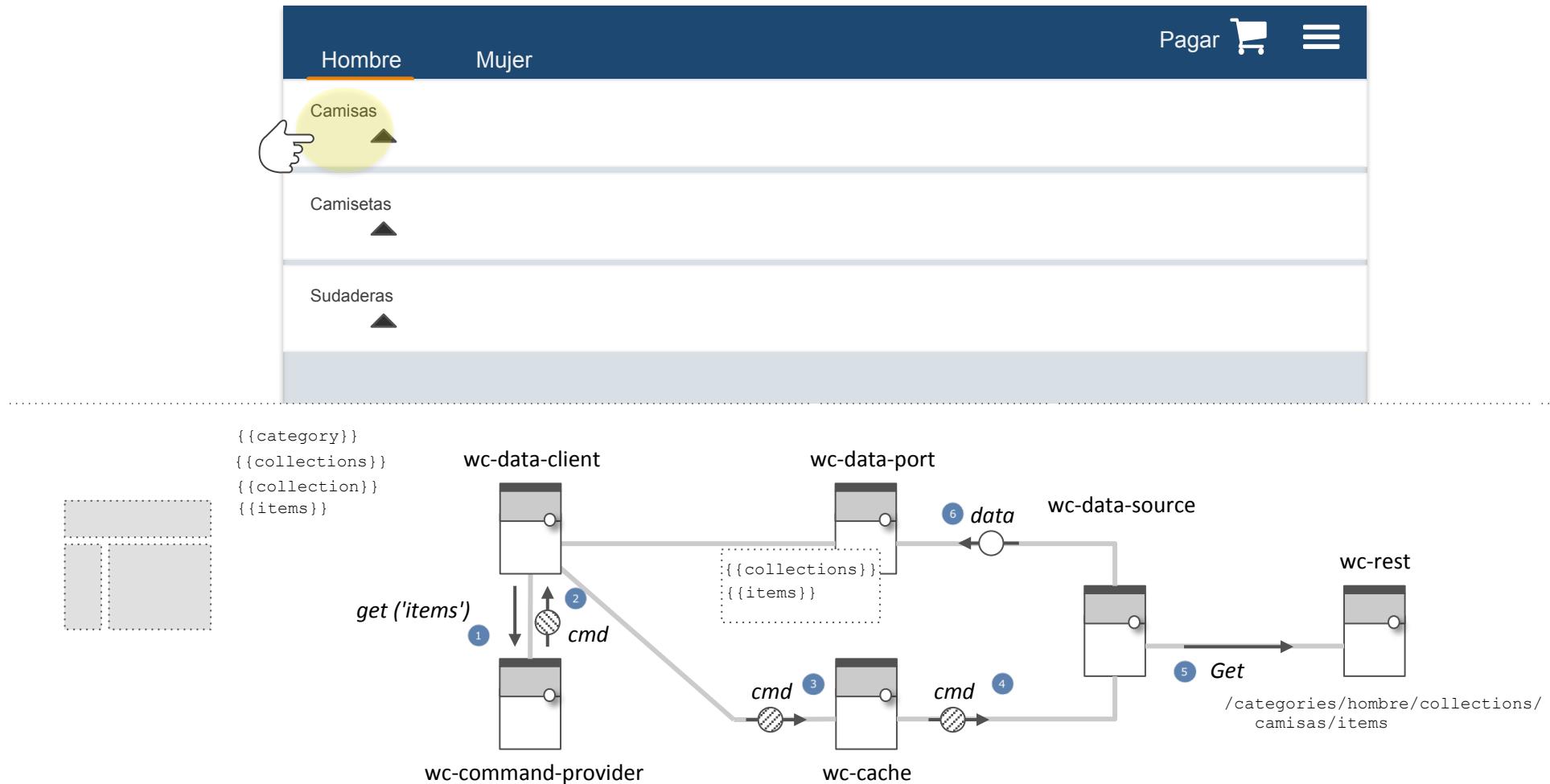
II. Portal de e-Commerce. Flujo de Exploración A



Patrones de Acceso a Datos

El Subsistema de Acceso a Datos en la Práctica

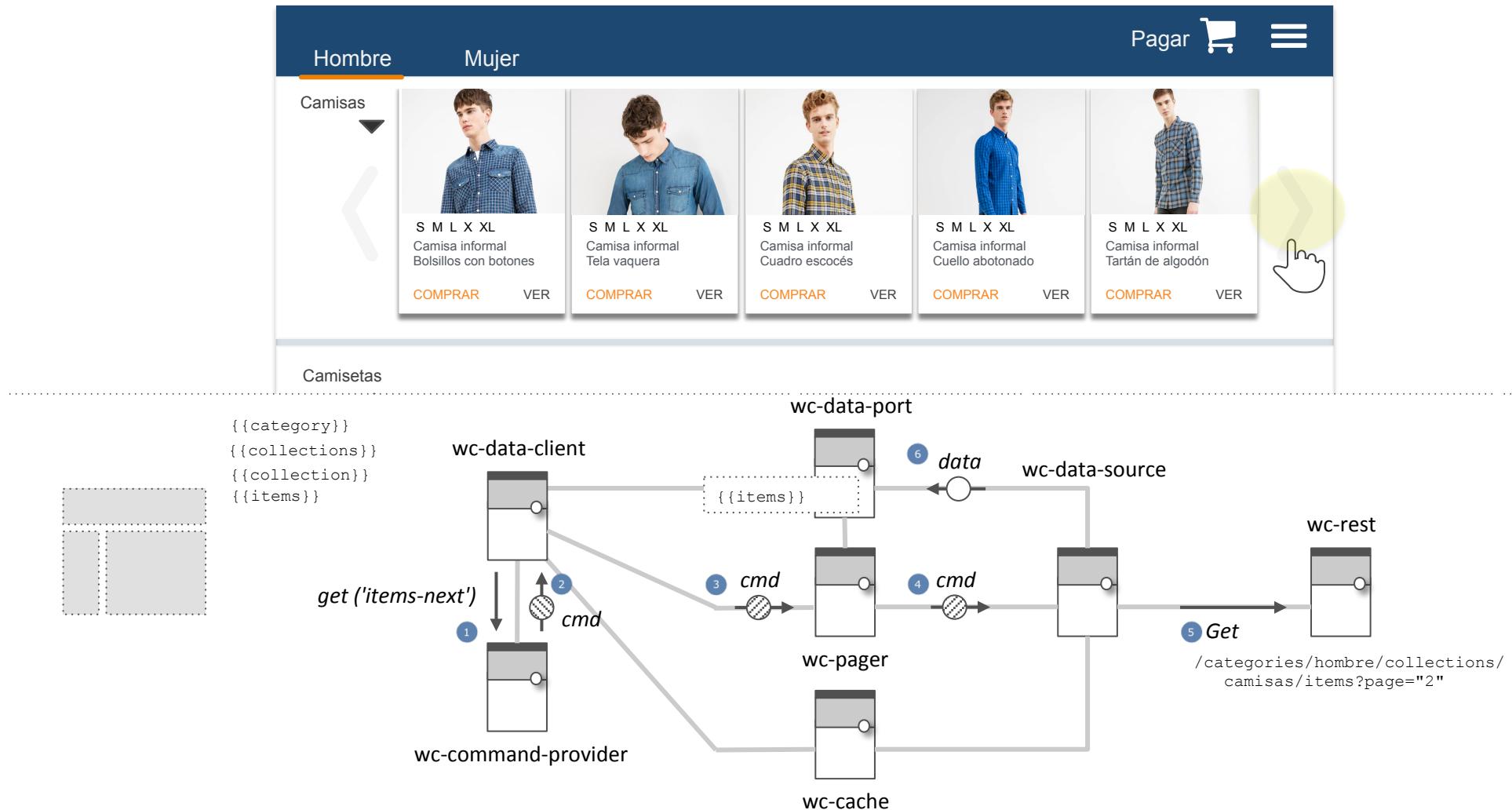
II. Portal de e-Commerce. Flujo de Exploración A



Patrones de Acceso a Datos

El Subsistema de Acceso a Datos en la Práctica

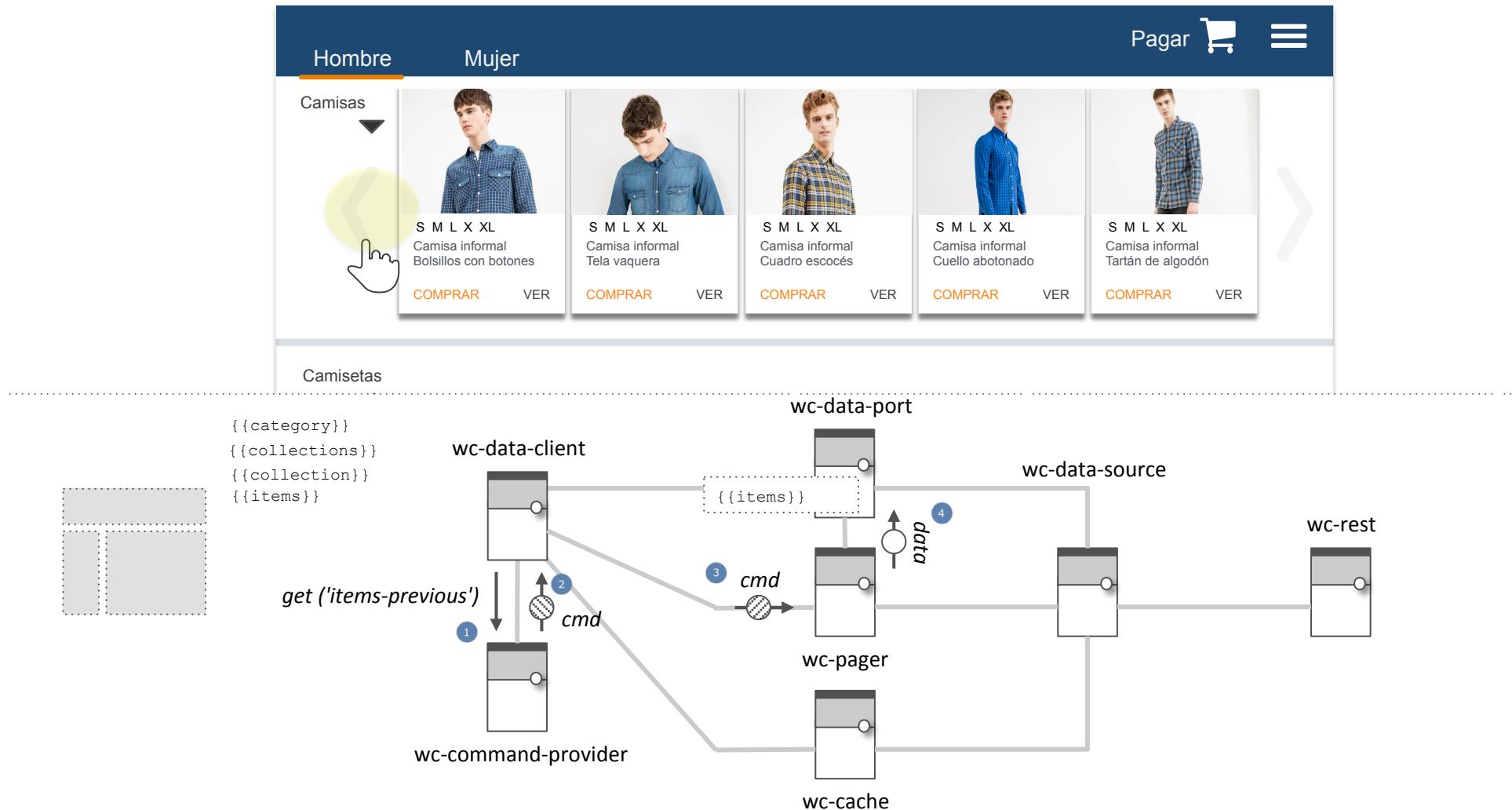
II. Portal de e-Commerce. Flujo de Exploración B



Patrones de Acceso a Datos

El Subsistema de Acceso a Datos en la Práctica

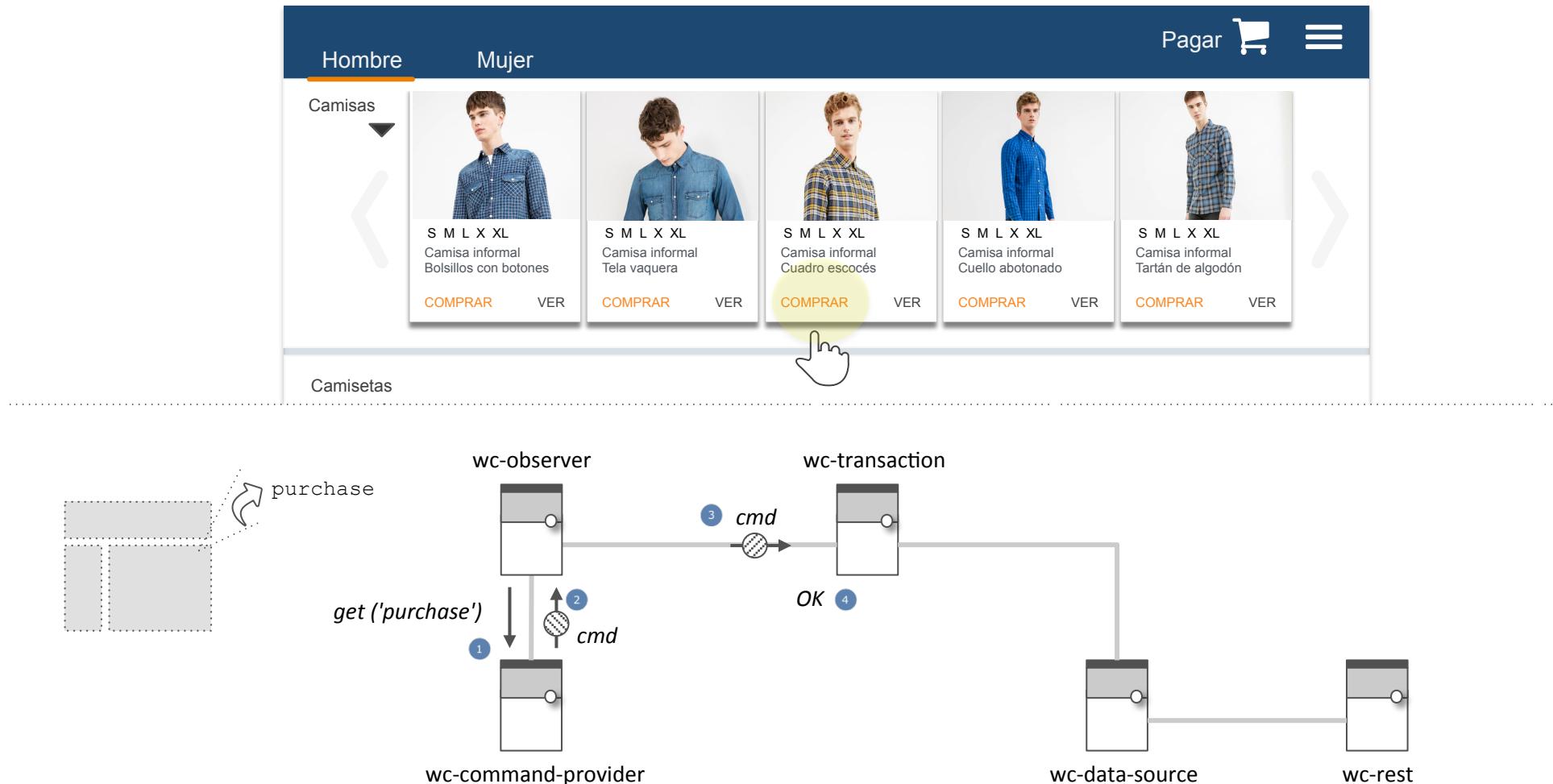
II. Portal de e-Commerce. Flujo de Exploración B



Patrones de Acceso a Datos

El Subsistema de Acceso a Datos en la Práctica

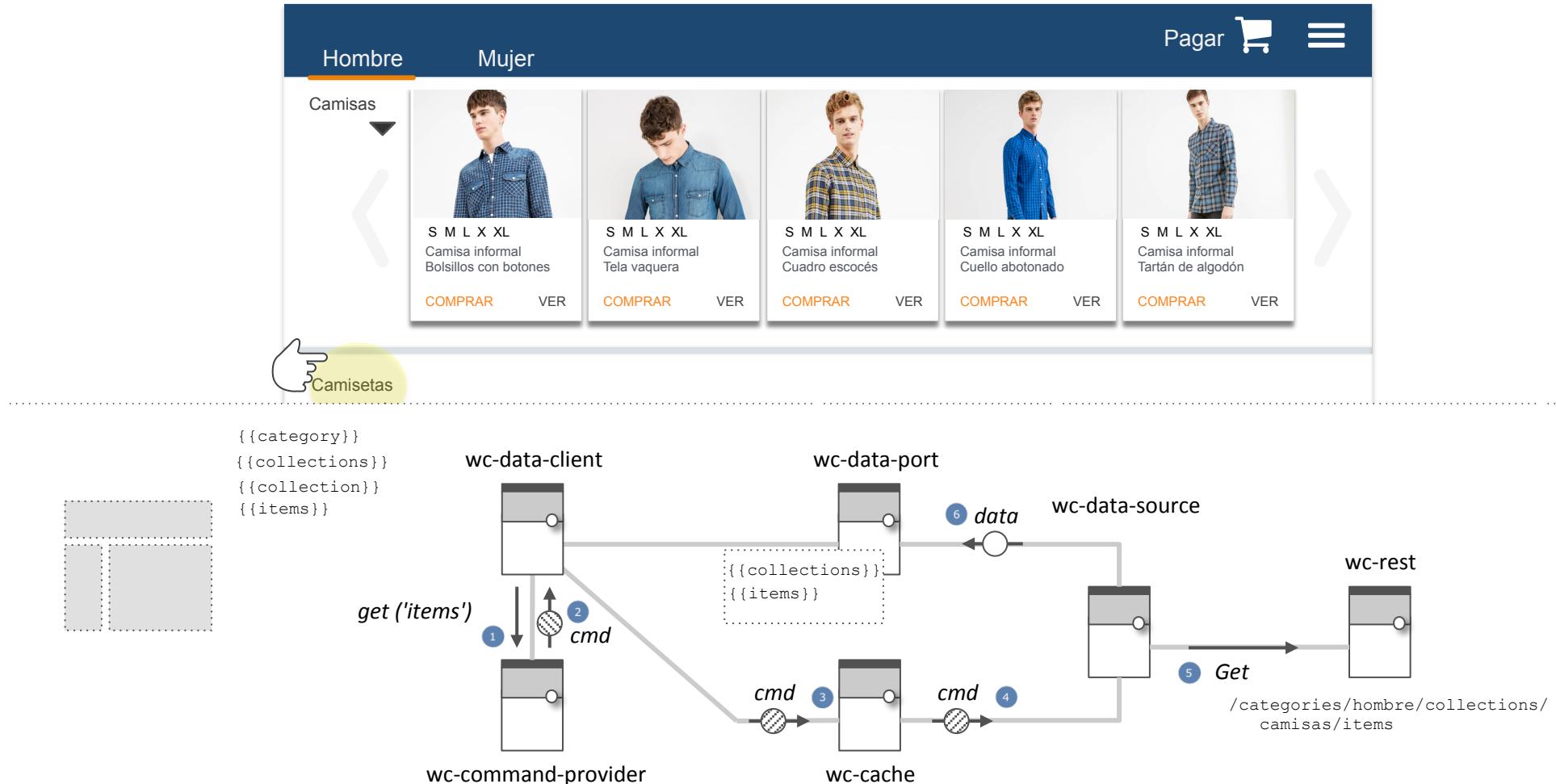
II. Portal de e-Commerce. Flujo de Compra



Patrones de Acceso a Datos

El Subsistema de Acceso a Datos en la Práctica

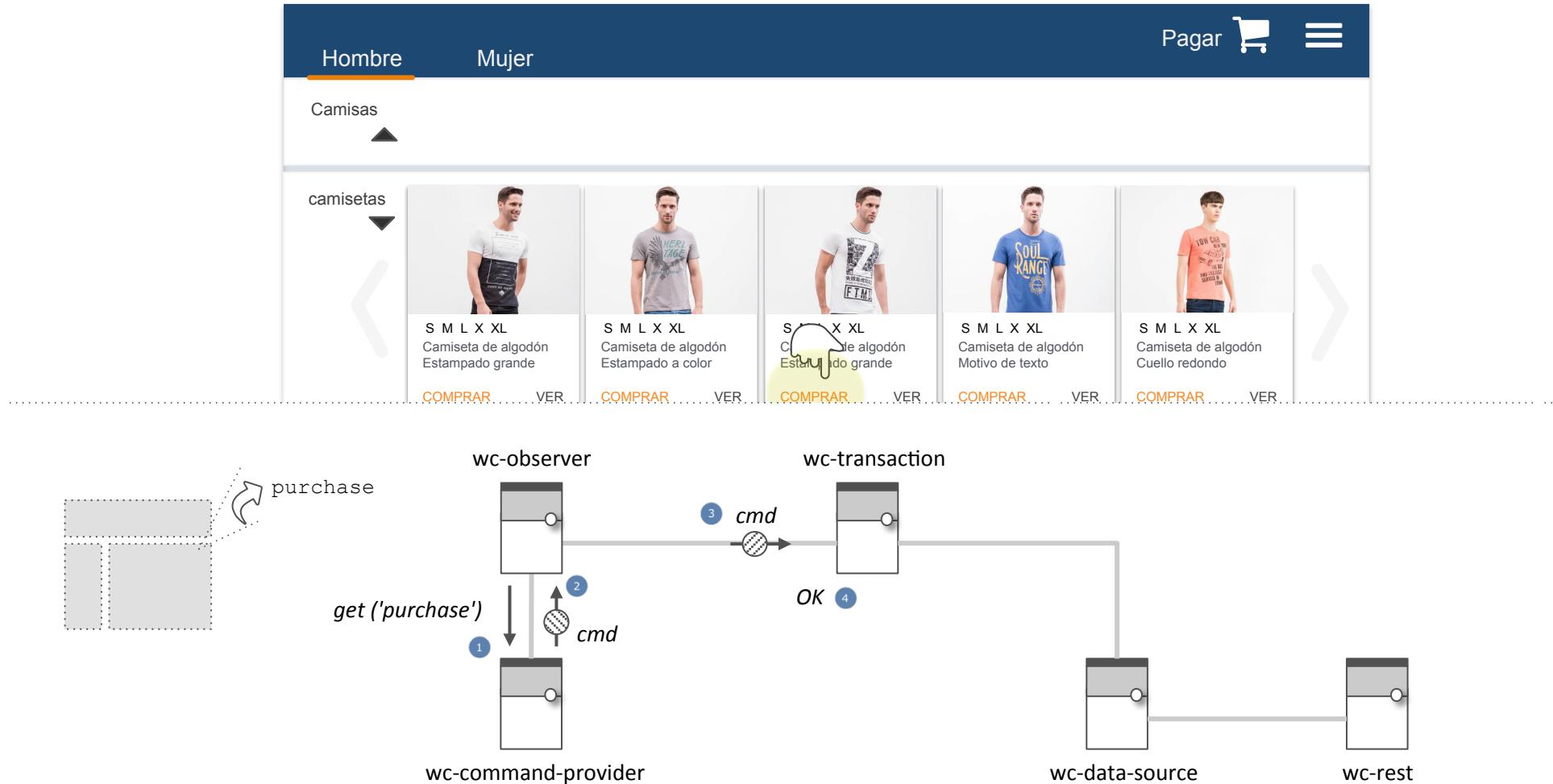
II. Portal de e-Commerce. Flujo de Exploración A



Patrones de Acceso a Datos

El Subsistema de Acceso a Datos en la Práctica

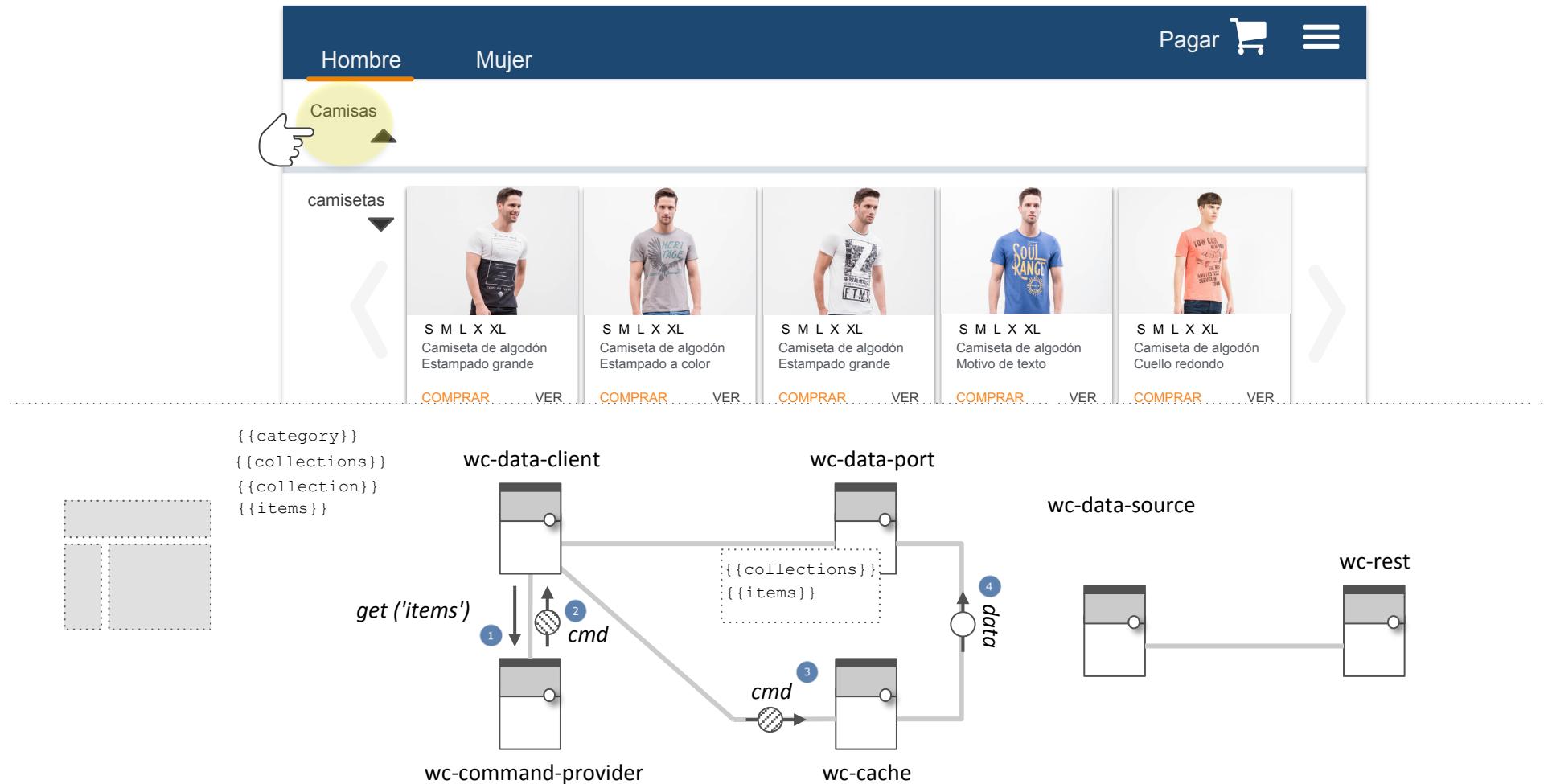
II. Portal de e-Commerce. Flujo de Compra



Patrones de Acceso a Datos

El Subsistema de Acceso a Datos en la Práctica

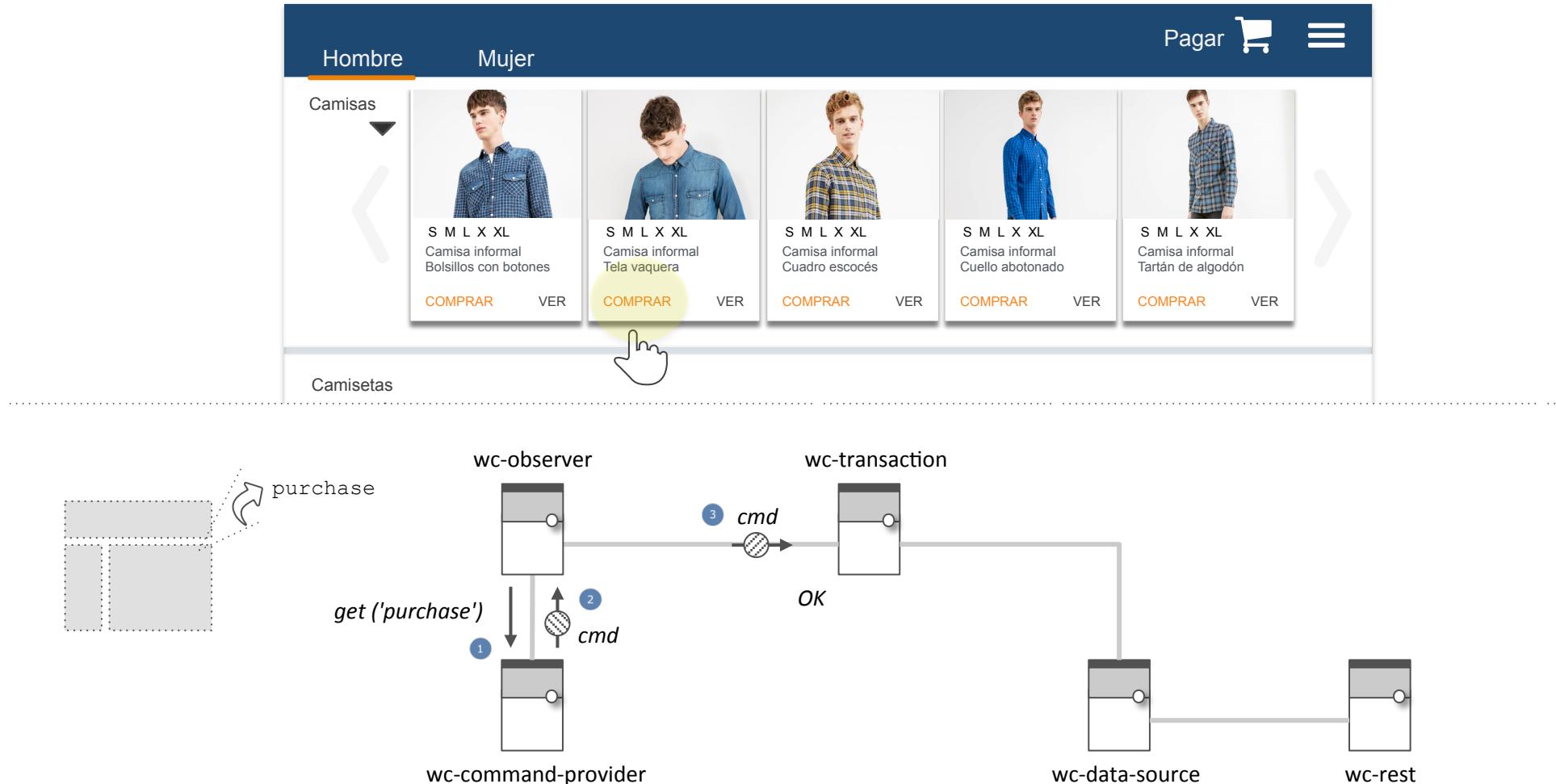
II. Portal de e-Commerce. Flujo de Exploración A



Patrones de Acceso a Datos

El Subsistema de Acceso a Datos en la Práctica

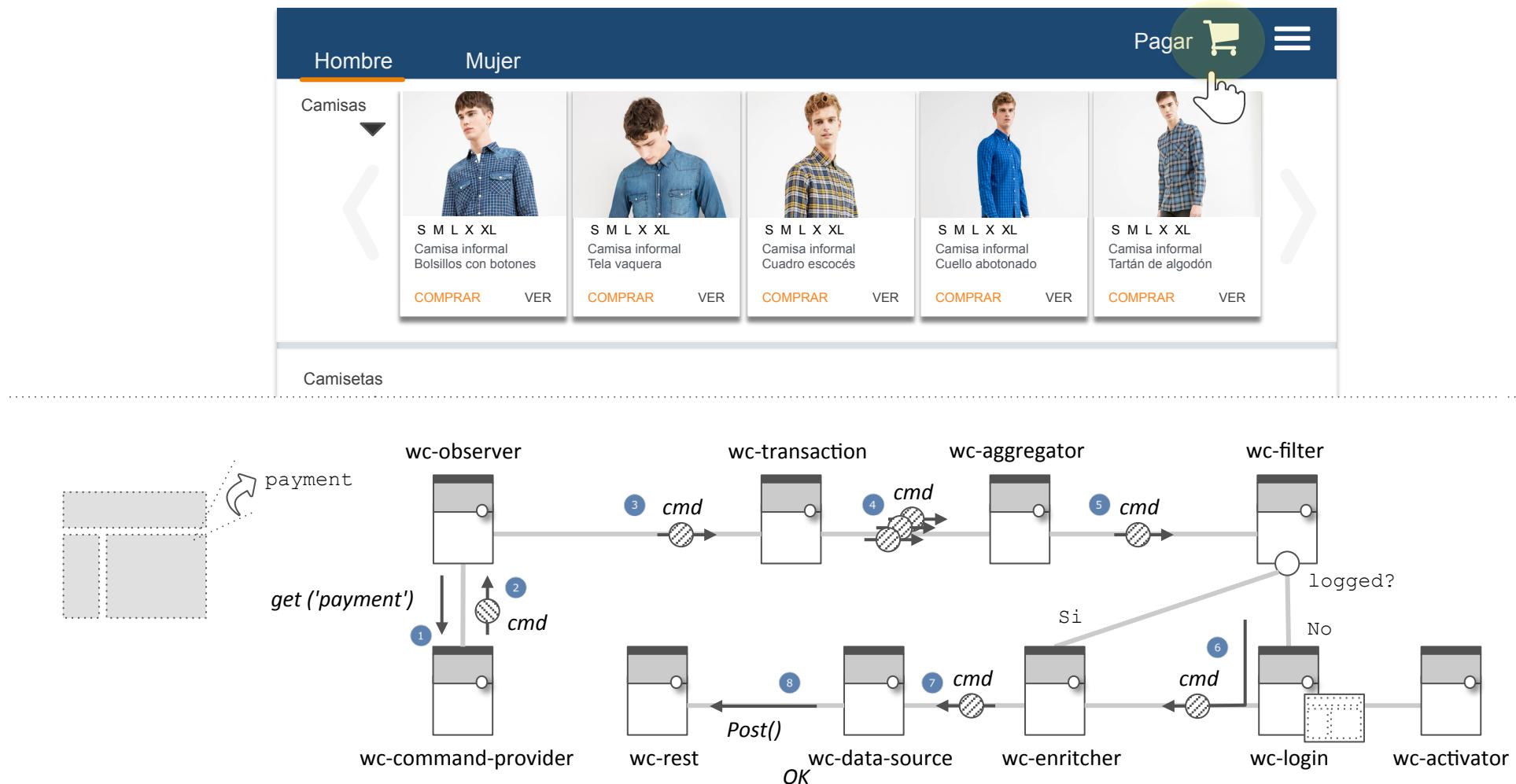
II. Portal de e-Commerce. Flujo de Compra



Patrones de Acceso a Datos

El Subsistema de Acceso a Datos en la Práctica

II. Portal de e-Commerce. Flujo de Pago



Patrones de Acceso a Datos

Conclusión

Conclusión

“

Para el usuario desarrollador de componentes Web debería ser transparente la naturaleza de la fuente de datos, su lógica acceso y activación, y las necesidades de transformación y optimización potenciales.

”

Patrones de Acceso a Datos

Preguntas

Arquitecturas Orientadas a Componentes Web

Patrones de Acceso a Datos



POLYMER.DAY



www.javiervelezreyes.com

Patrones de Acceso a Datos

Referencias

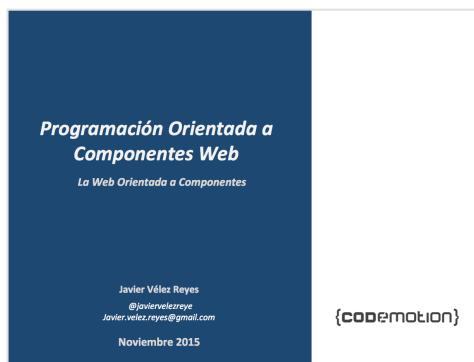
Referencias



La Web Orientada a Componentes

Componentes Web · Arquitecturas Software · Frontend · JavaScript · Composición · Encapsulación · Reutilización · Buenas Prácticas · Principios de Diseño · Errores comunes

Fecha	Noviembre de 2015
Lugar	Code Motion
Páginas	57
Ref	https://goo.gl/mCxm3w



Principios de Diseño en Componentes Web

Programación Orientada a Componentes · Principios de Diseño · Buenas Prácticas · Recomendaciones de Diseño y Desarrollo · Técnicas de Programación de Componentes Web · Polymer

Fecha	Marzo de 2015
Lugar	Polymer Madrid
Páginas	170
Ref	https://goo.gl/t0GpVS



Arquitecturas Orientadas a Componentes Web

Patrones de Acceso a Datos

Javier Vélez Reyes

@javiervelezreye

Javier.velez.reyes@gmail.com

Octubre 2016

POLYMER.DAY