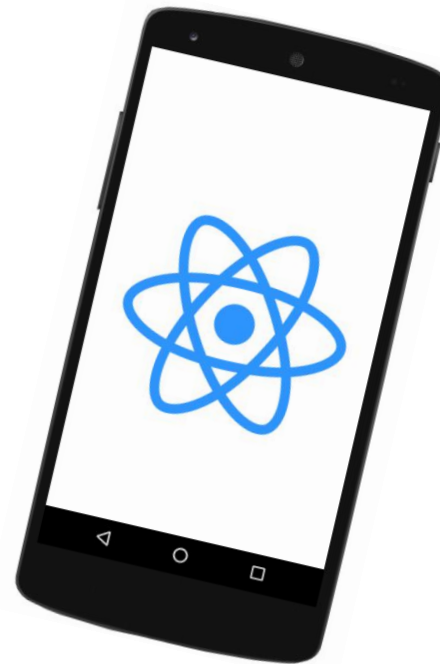


React Native



Un framework para la creación de aplicaciones
nativas Android y iOS con React

React Native

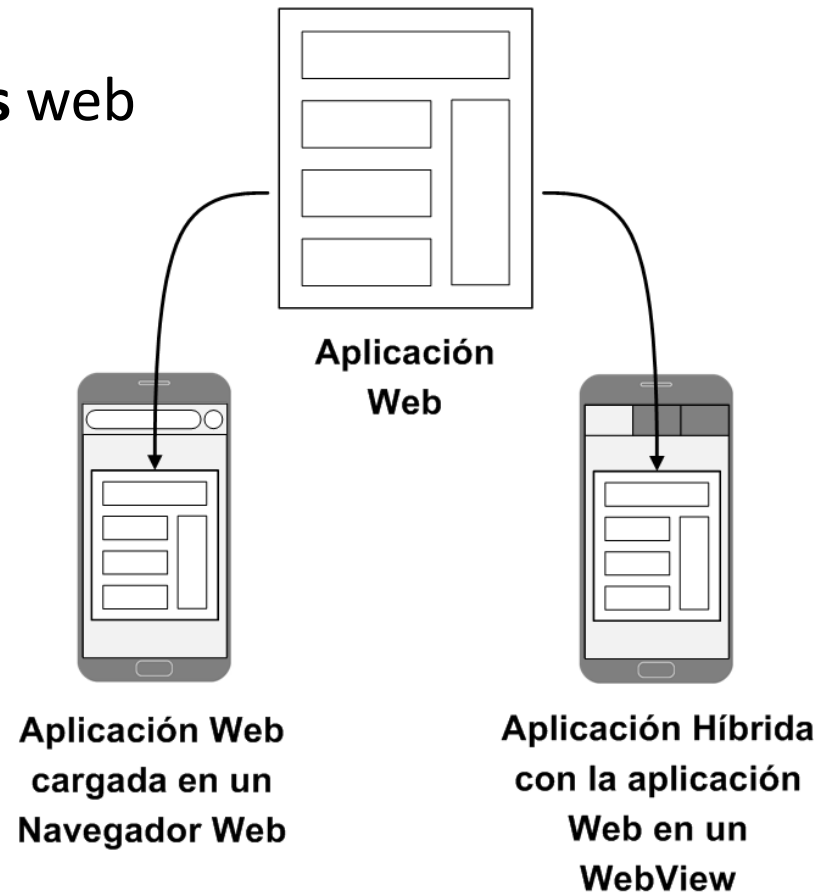
- Es un **framework** para la creación de aplicaciones **nativas Android y iOS** con **React**
- Permite desarrollar aplicaciones móviles empleando únicamente el lenguaje **JavaScript** y la biblioteca **React**
- Creado, mantenido y usado por Facebook
- **Open source**
<https://github.com/facebook/react-native>
- **Documentación** disponible en:
<http://facebook.github.io/react-native/docs>

Aplicaciones React Native

- Las aplicaciones creadas con React Native son realmente **nativas**
 - **No son aplicaciones web** ejecutadas en navegadores
 - **No son aplicaciones híbridas** (apps web embebidas en apps nativas)
 - **Son aplicaciones nativas**
como las aplicaciones Android creadas con Java
o las aplicaciones iOS creadas utilizando Objective-C
- En una aplicación React Native se pueden utilizar **componentes** escritos en **Java**, **Objective-C** o **Swift**
- Se pueden empaquetar en archivos binarios
 - Android: *ficheros APK*
 - iOS: *ficheros IPA*

Aplicaciones Web

- Desarrolladas con tecnologías web:
HTML, CSS y JavaScript
- Accedidas mediante **navegadores web**
- **Multiplataforma**
“Write once, run anywhere”
(aunque pueden existir inconsistencias entre navegadores)
- Distribución instantánea
- Presentan **limitaciones** en el acceso a funcionalidades nativas de los dispositivos (cámara, geolocalización, ...)
- Menor rendimiento



Aplicaciones Híbridas

- **Aplicaciones nativas que envuelven una aplicación web en un componente nativo:**

- Android: *WebView*

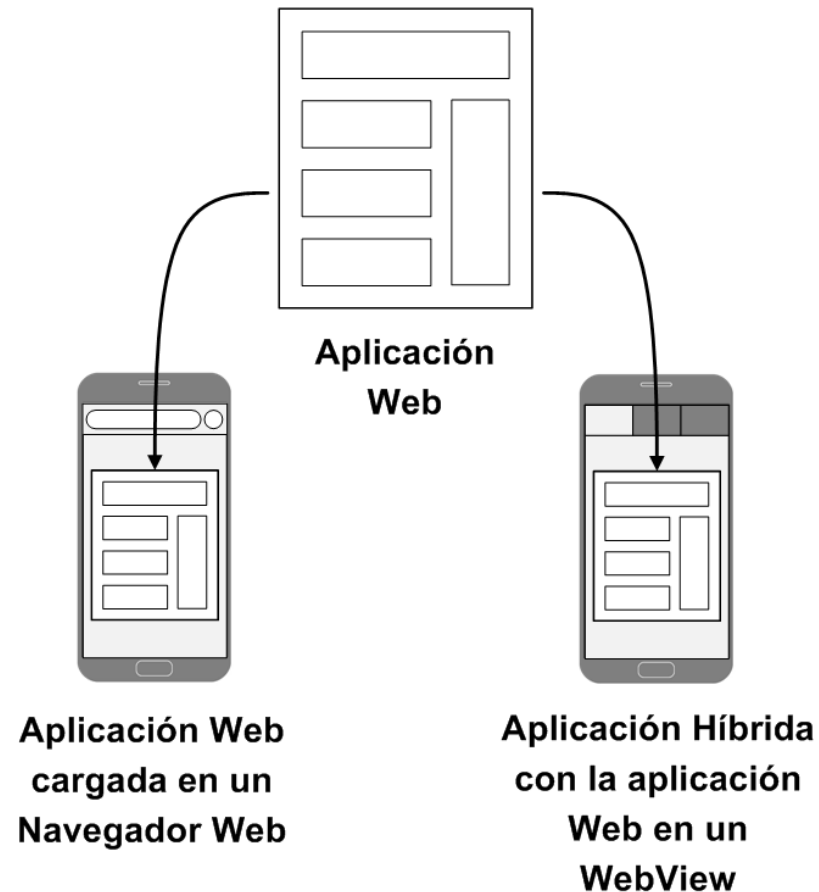
- iOS: *UIWebView*

- Se instalan en dispositivos, pero pueden residir en servidores web

- Permiten combinar elementos nativos y aplicaciones HTML5

- Suelen consistir en **aplicaciones HTML5 a las que se provee de acceso a funcionalidades nativas**

- Existen frameworks como **PhoneGap** (<http://phonegap.com>) para crear aplicaciones híbridas



Aplicaciones Nativas

- Desarrolladas utilizando **diferentes** lenguajes de programación (Java en Android, Objective-C y Swift en iOS)
- Deben ser instaladas en los dispositivos, se distribuyen mediante archivos binarios
- **No son multiplataforma**: cada plataforma requiere desarrollar una aplicación nativa específica para ella
- No tienen limitaciones en el acceso a **funcionalidades nativas** ni necesidad de utilizar puentes o envoltorios
- **Elementos de la interfaz** consistentes con el aspecto de la plataforma
- **Mejor rendimiento**

React Native

- **React Native** permite crear **aplicaciones nativas** para **Android** y **iOS** empleando **JavaScript** y **React**
- Soporte para:
 - \geq Android 4.1 (API 16), desarrollo con Linux, Mac y Windows
 - \geq iOS 8.0, desarrollo solo con Mac
- Se pueden crear aplicaciones que funcionen solo en Android, solo en iOS o en ambas plataformas
- Los componentes escritos con código nativo solo se pueden usar para la plataforma para la que fueron escritos

React y React Native

React

React Native

Android

iOS

...

Web

React y React Native

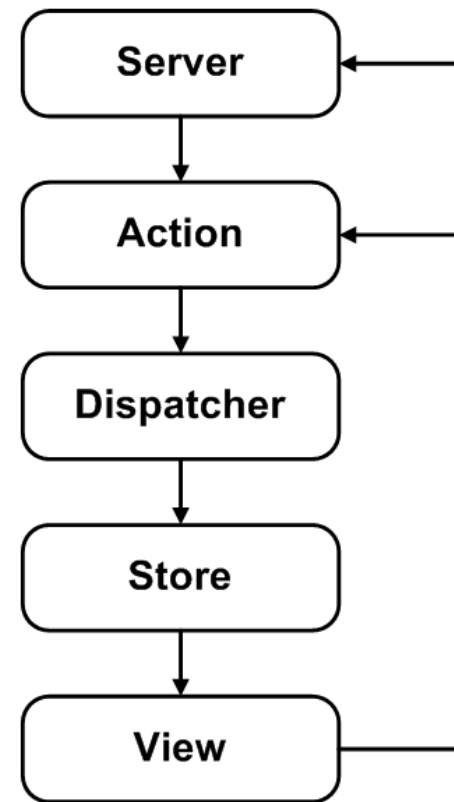
- Todo lo aprendido sobre **React** y **Flux** sigue siendo válido para desarrollar **aplicaciones React Native** (JSX, componentes, estado, propiedades, ...)

- **React**

- Biblioteca JavaScript para crear interfaces de usuario
- Basado en **componentes**
- **DOM virtual**
- **Flujo de datos unidireccional**

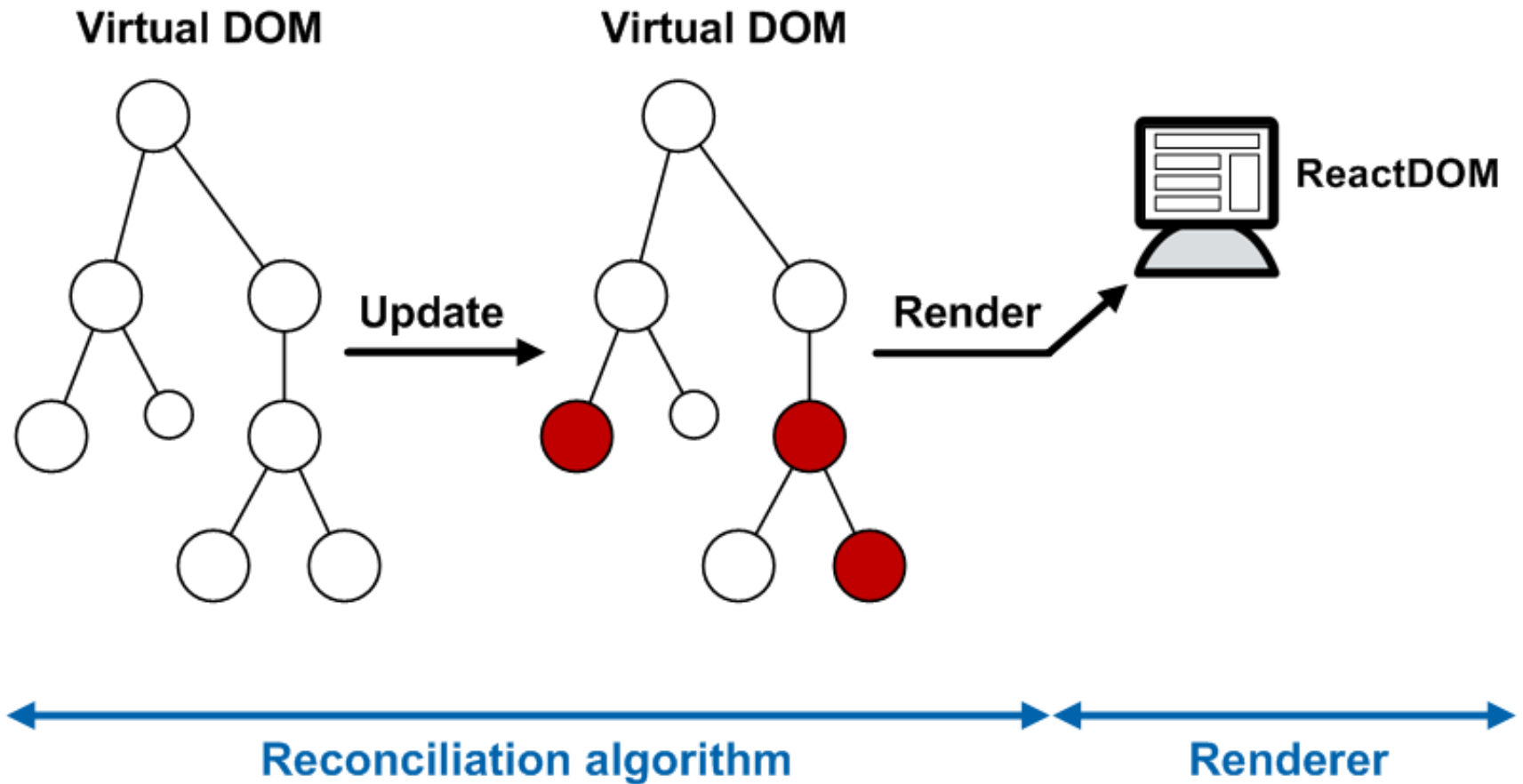
- **Flux**

- **Arquitectura** para modelar toda la aplicación
- Fuerza que todos los datos vayan en la misma dirección

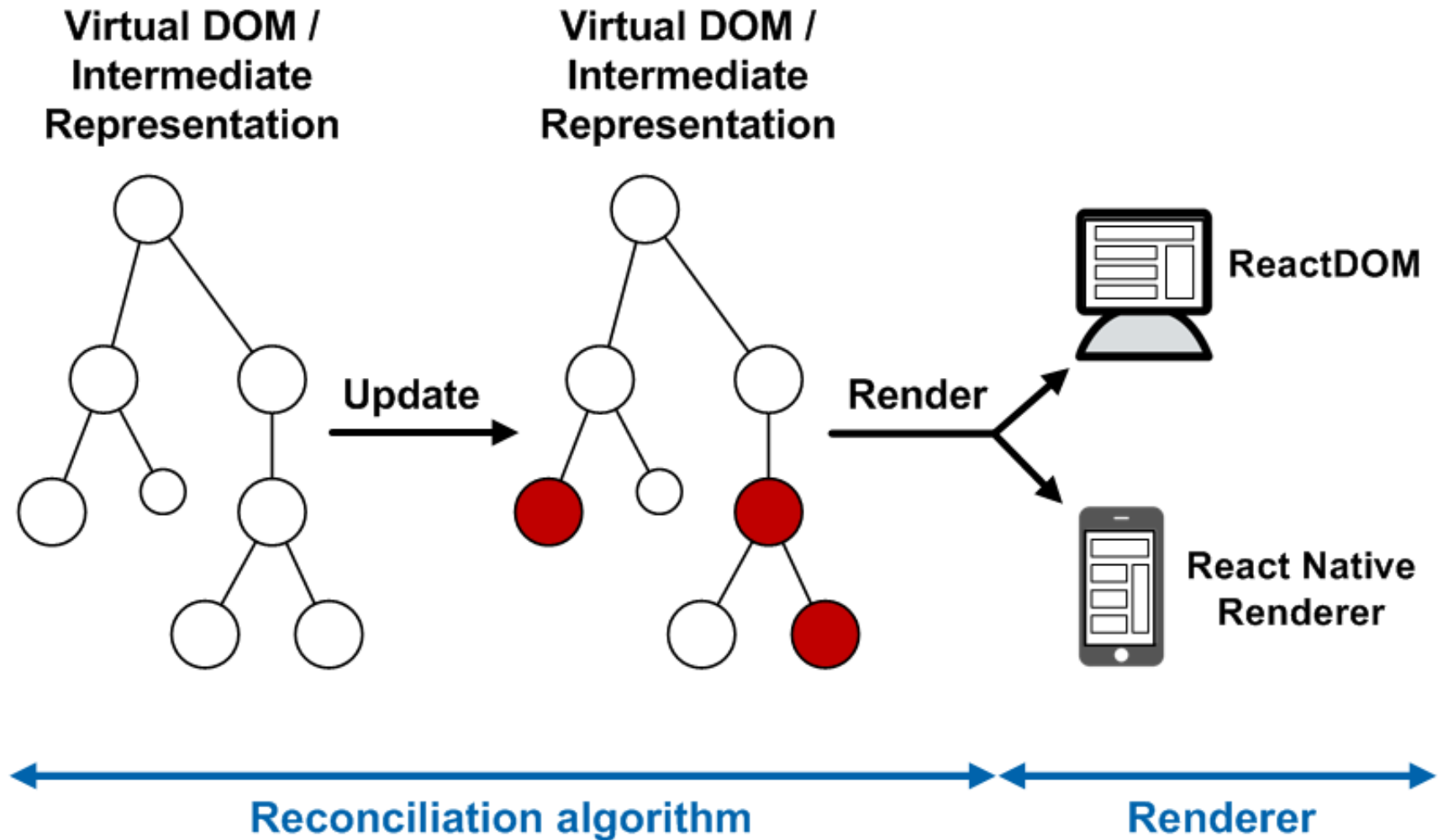


Arquitectura Flux

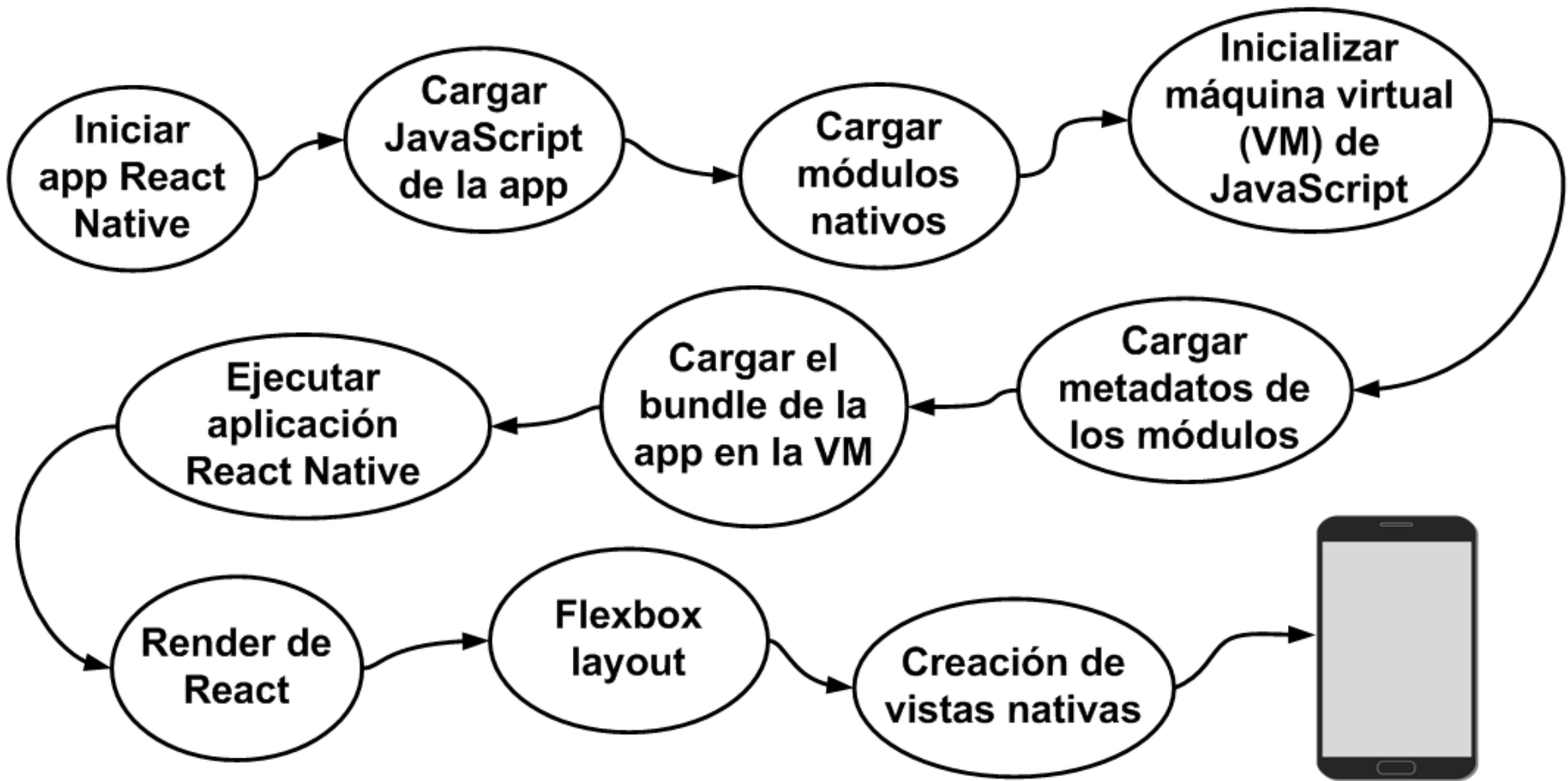
React y React Native



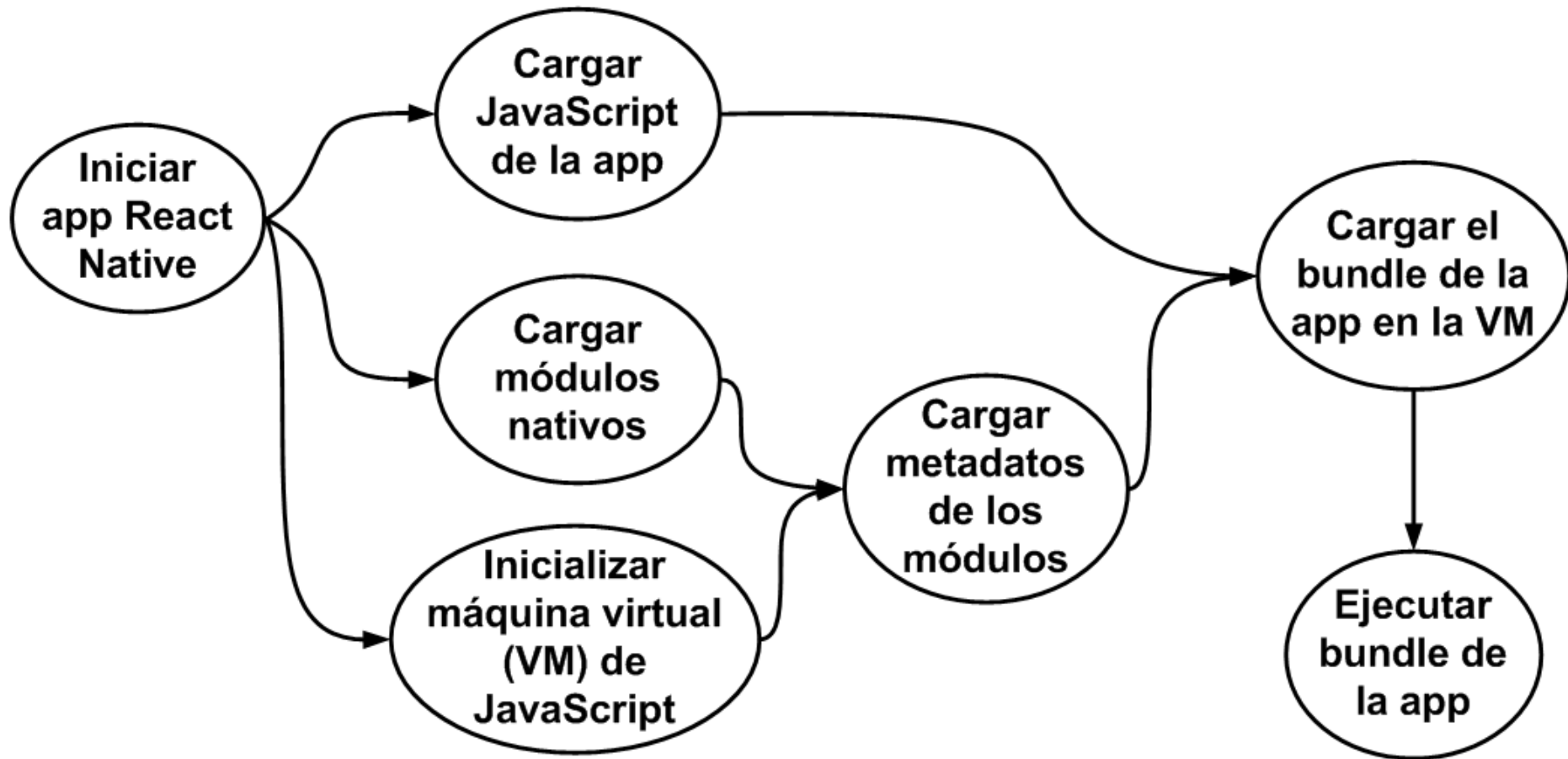
React y React Native



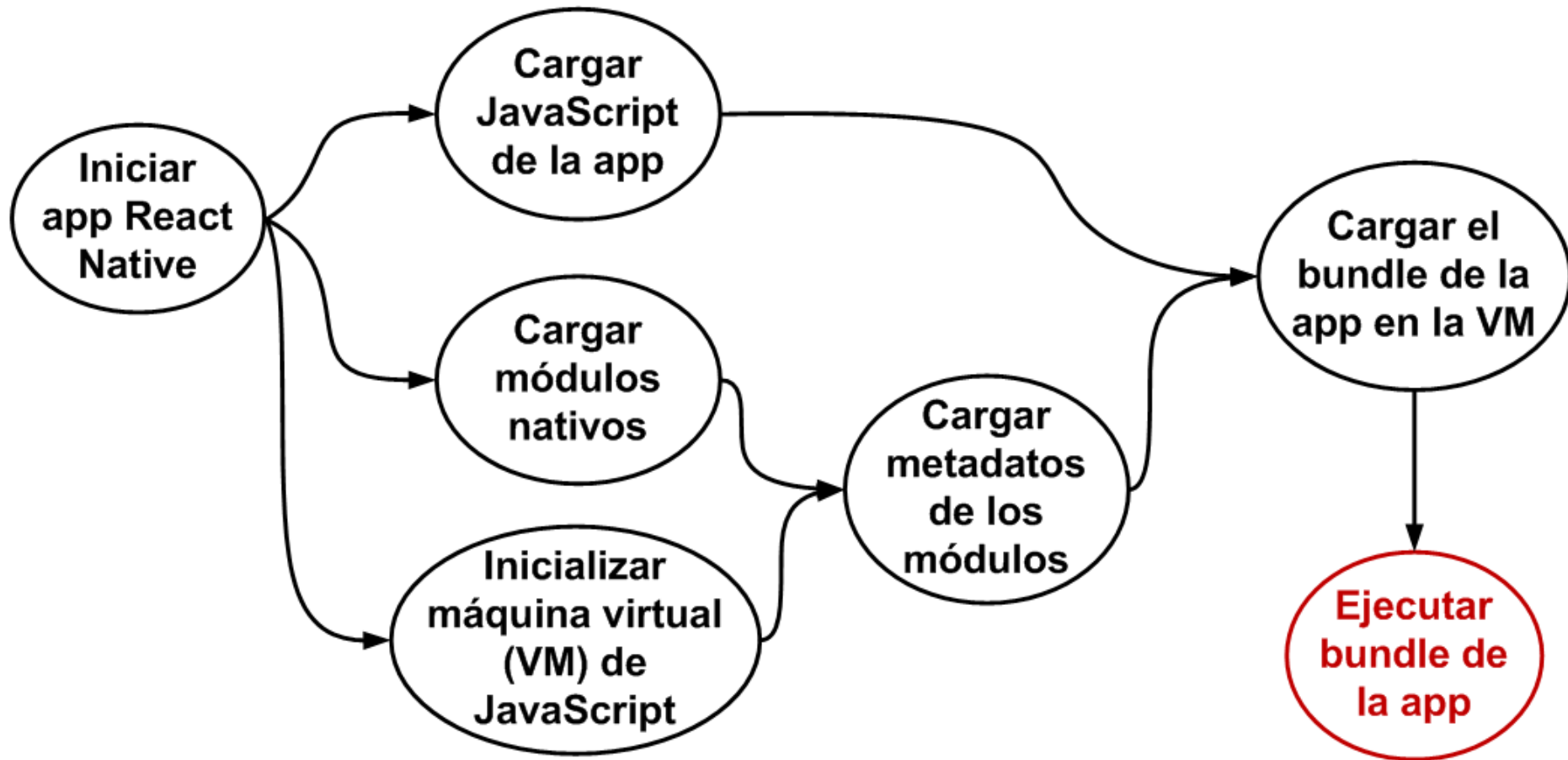
React Native: Inicialización de aplicaciones



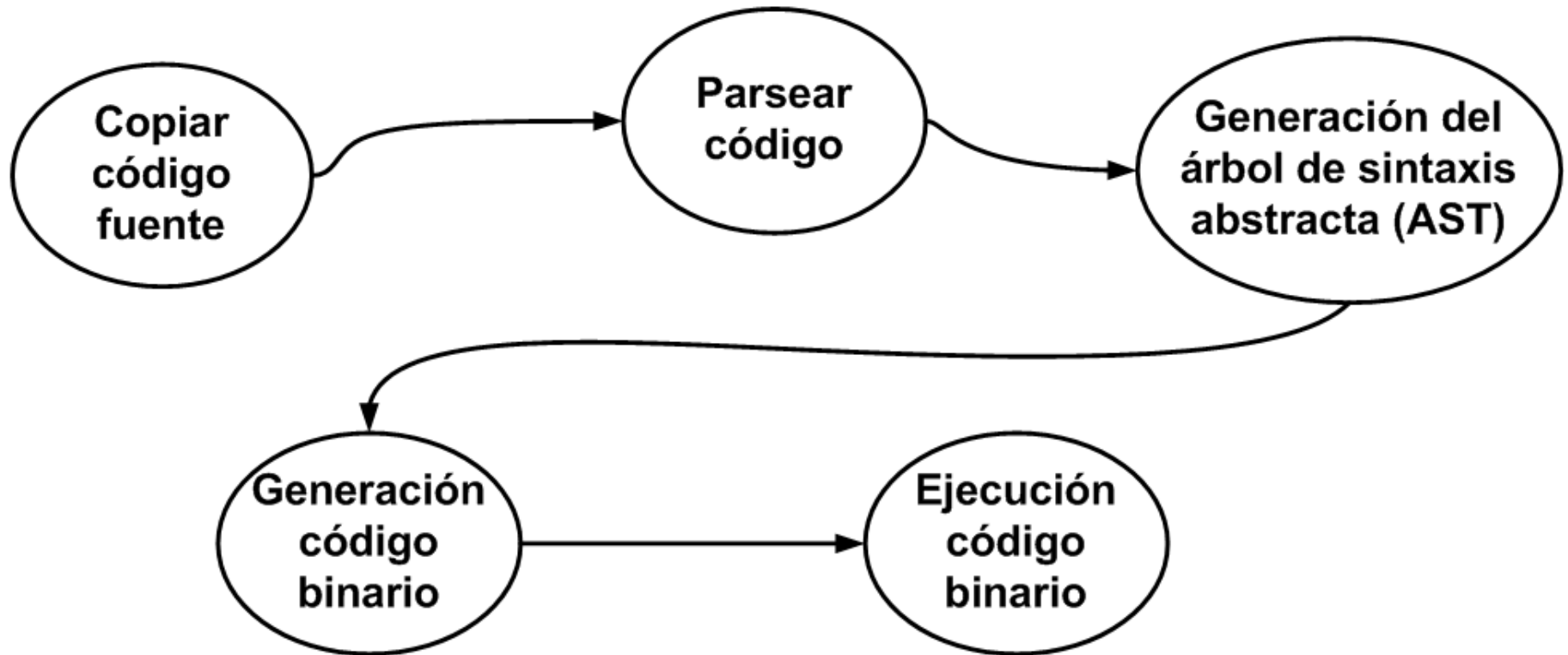
React Native: Inicialización de aplicaciones



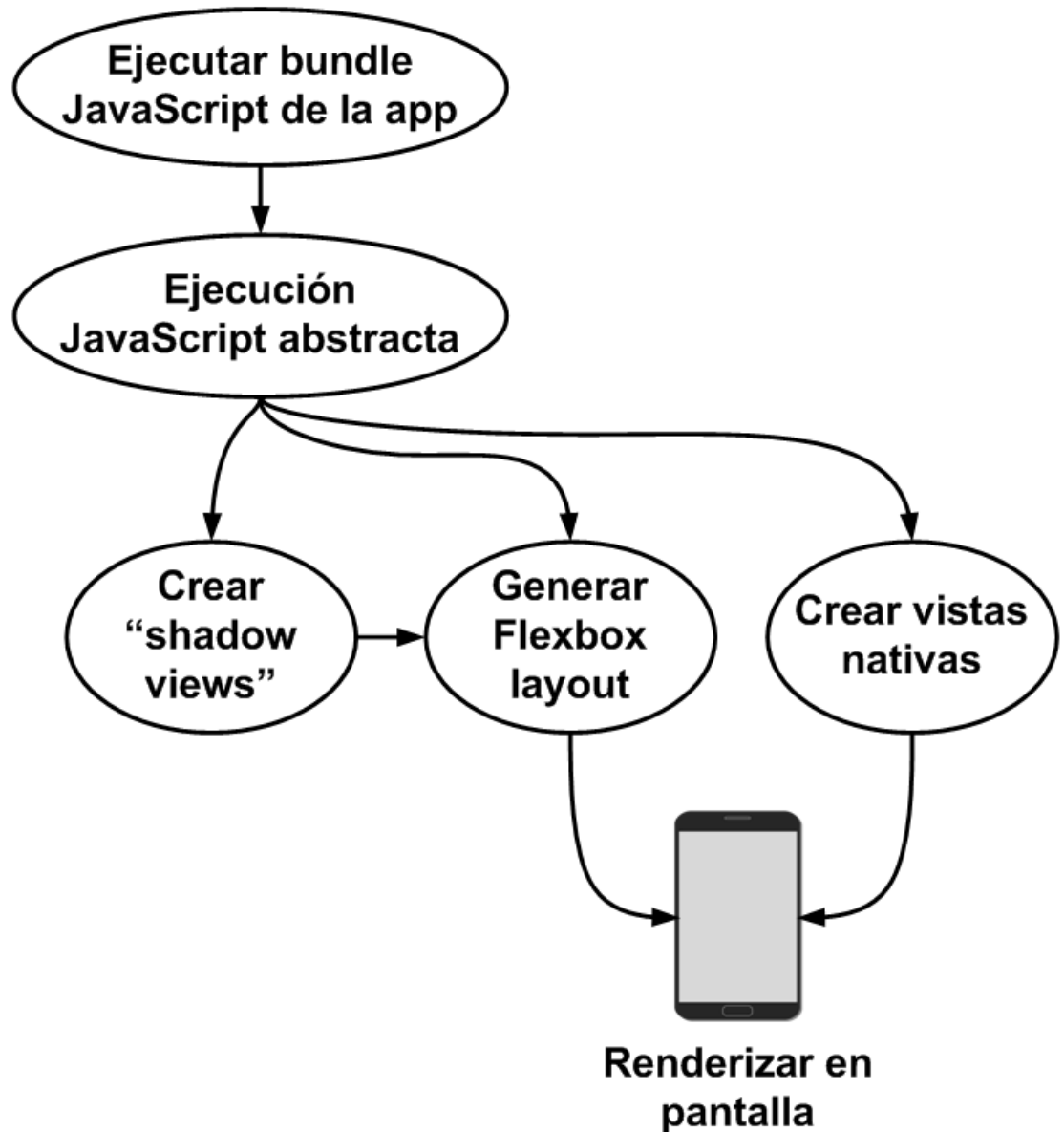
React Native: Inicialización de aplicaciones



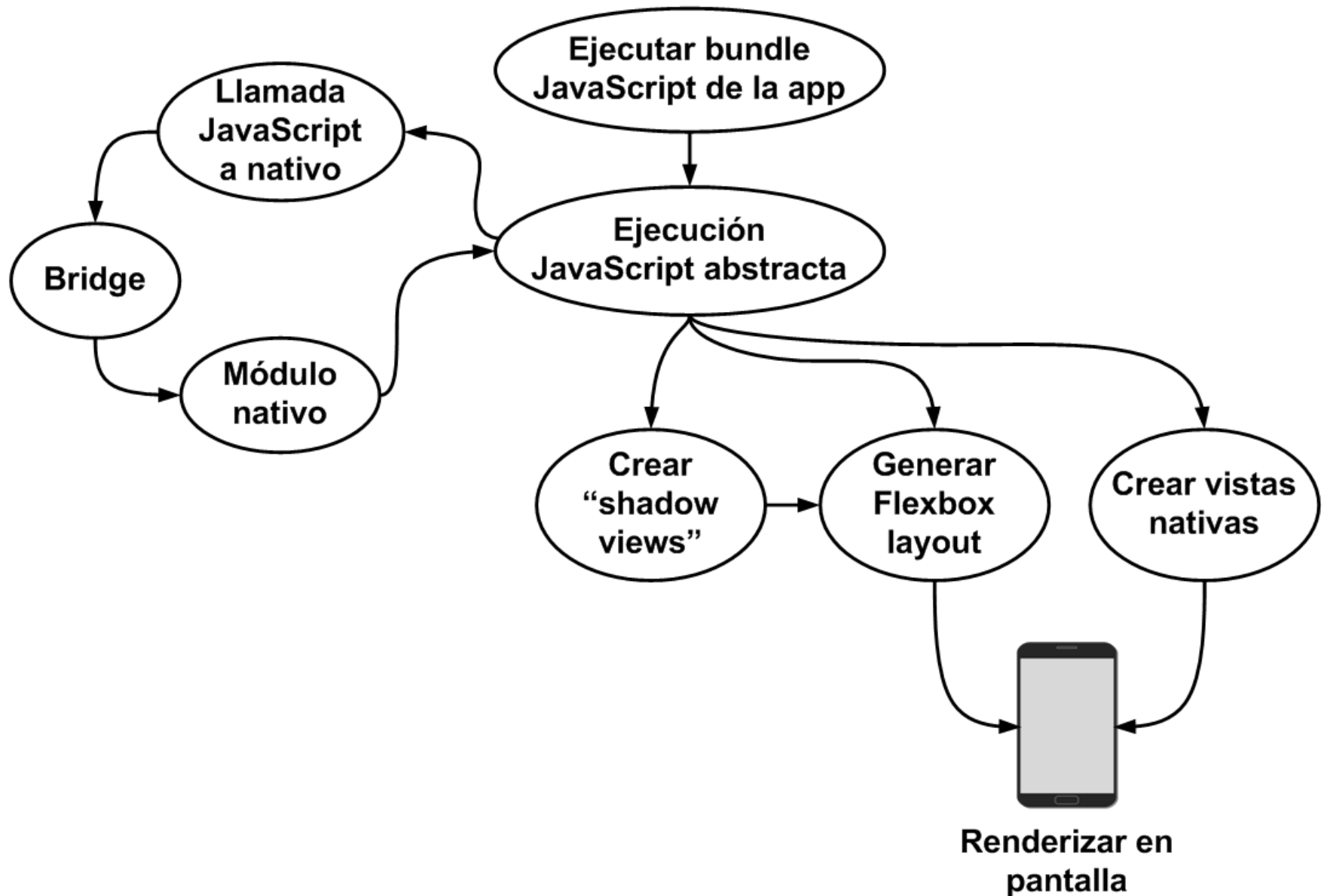
React Native: JavaScript VM (Virtual Machine)



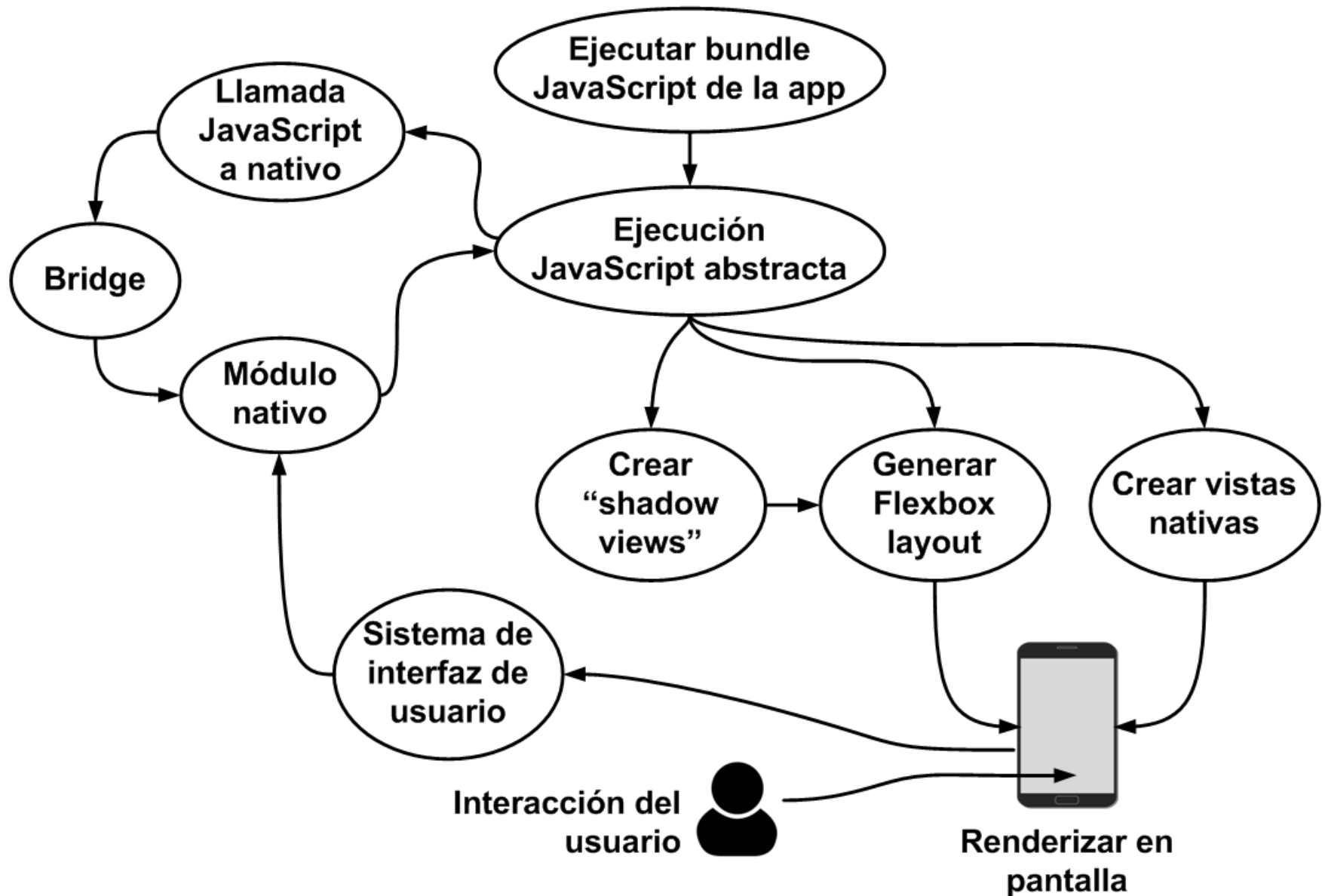
React Native: Arquitectura



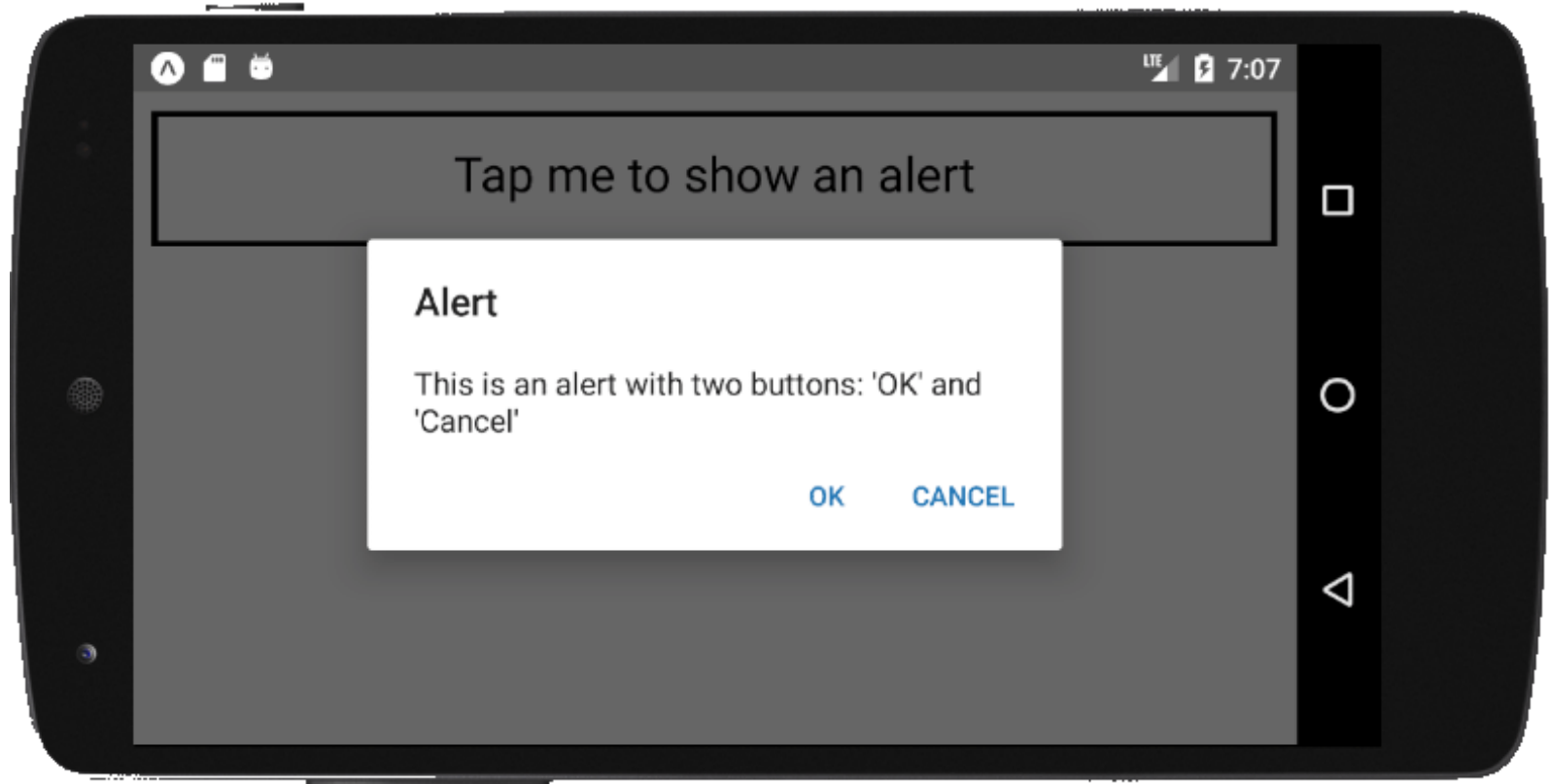
React Native: Arquitectura



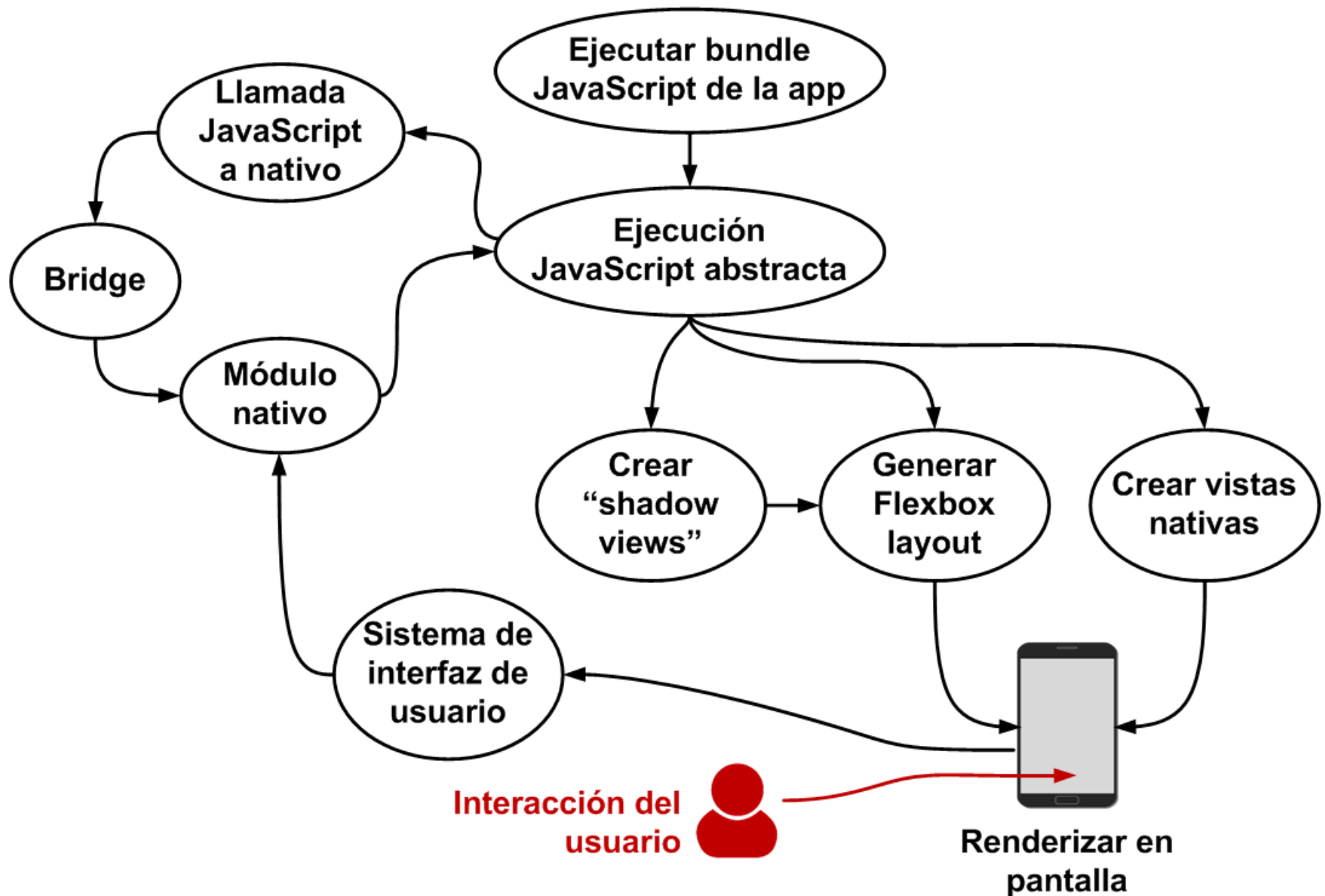
React Native: Arquitectura



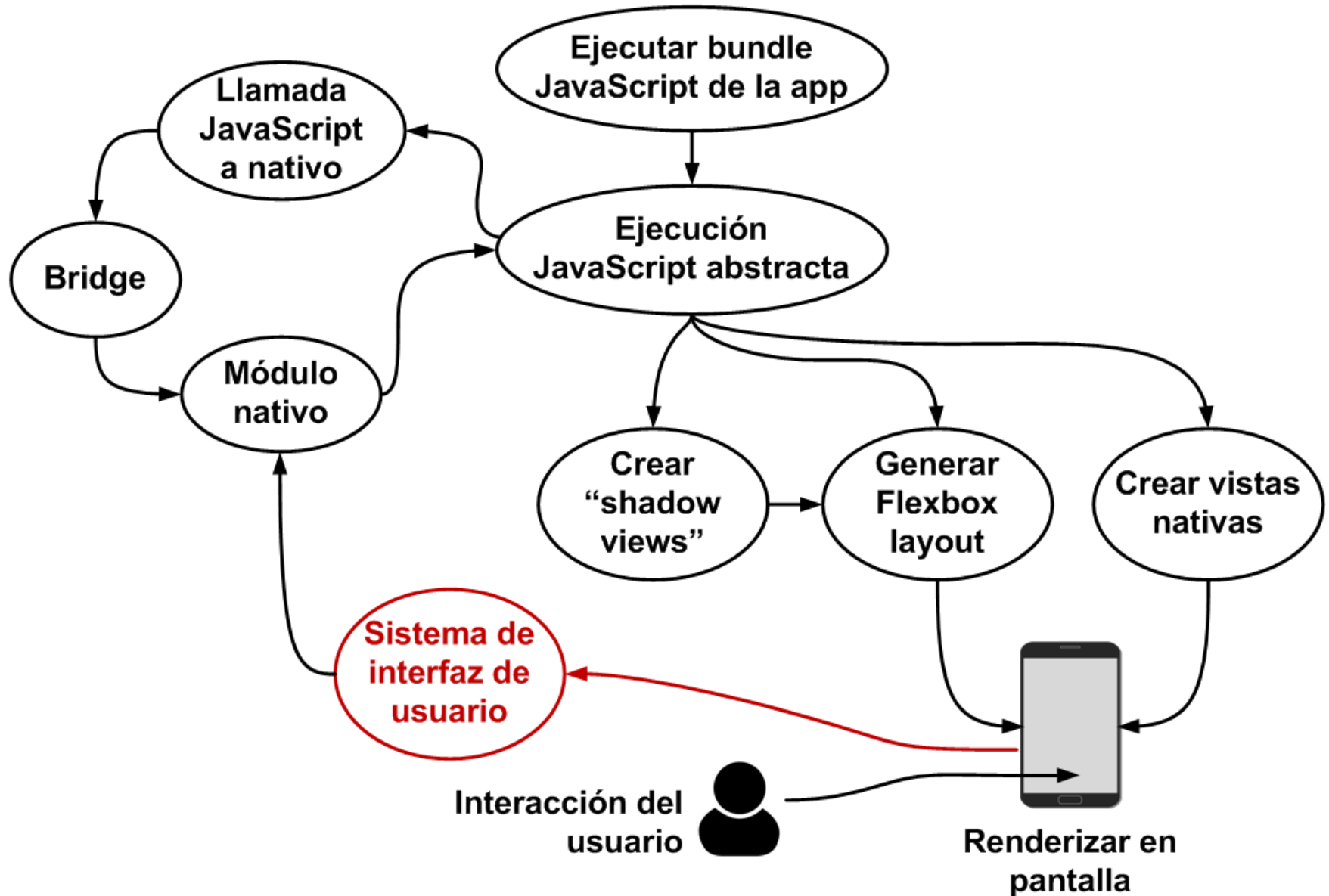
React Native: Ejemplo con Alert



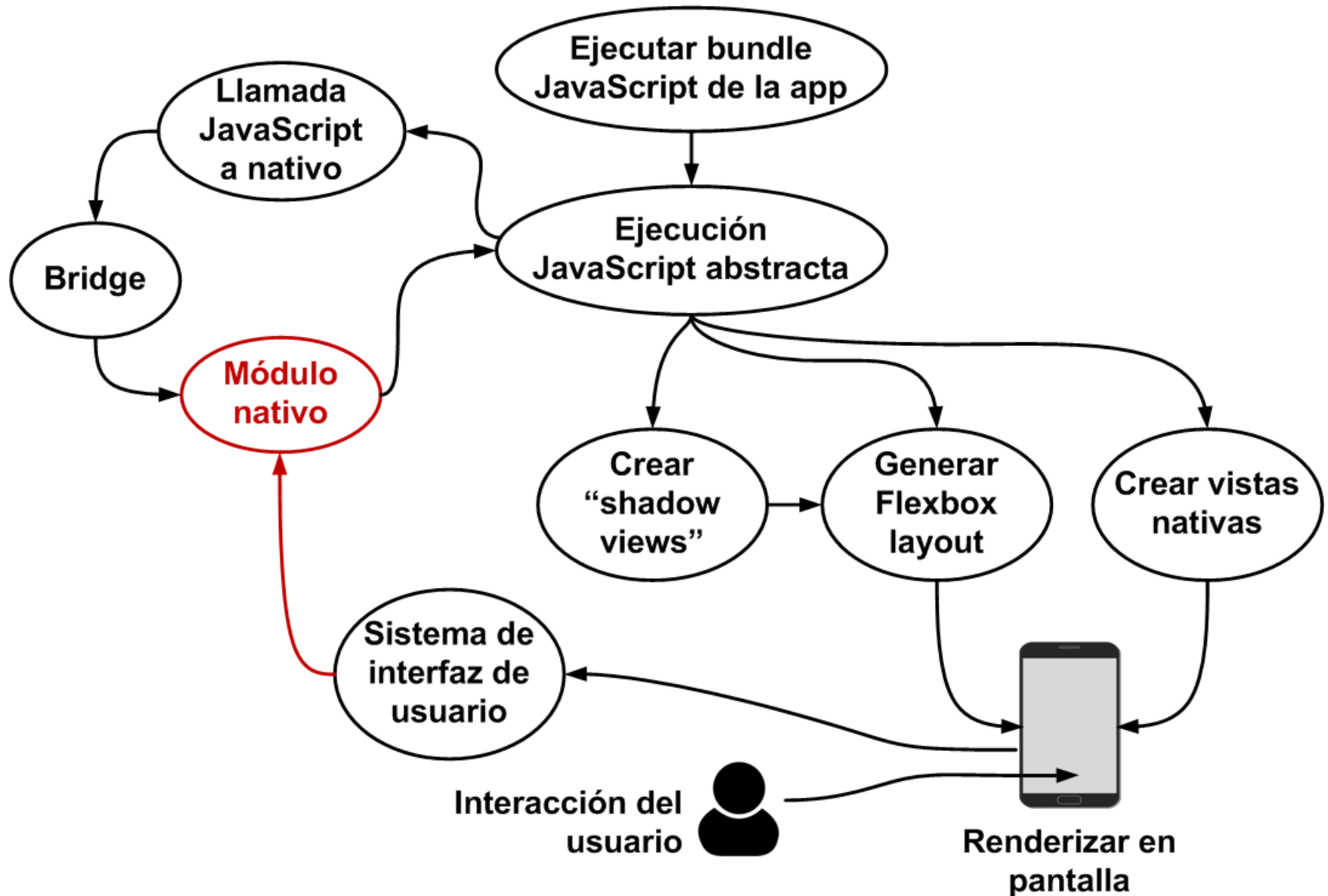
React Native: Ejemplo con Alert



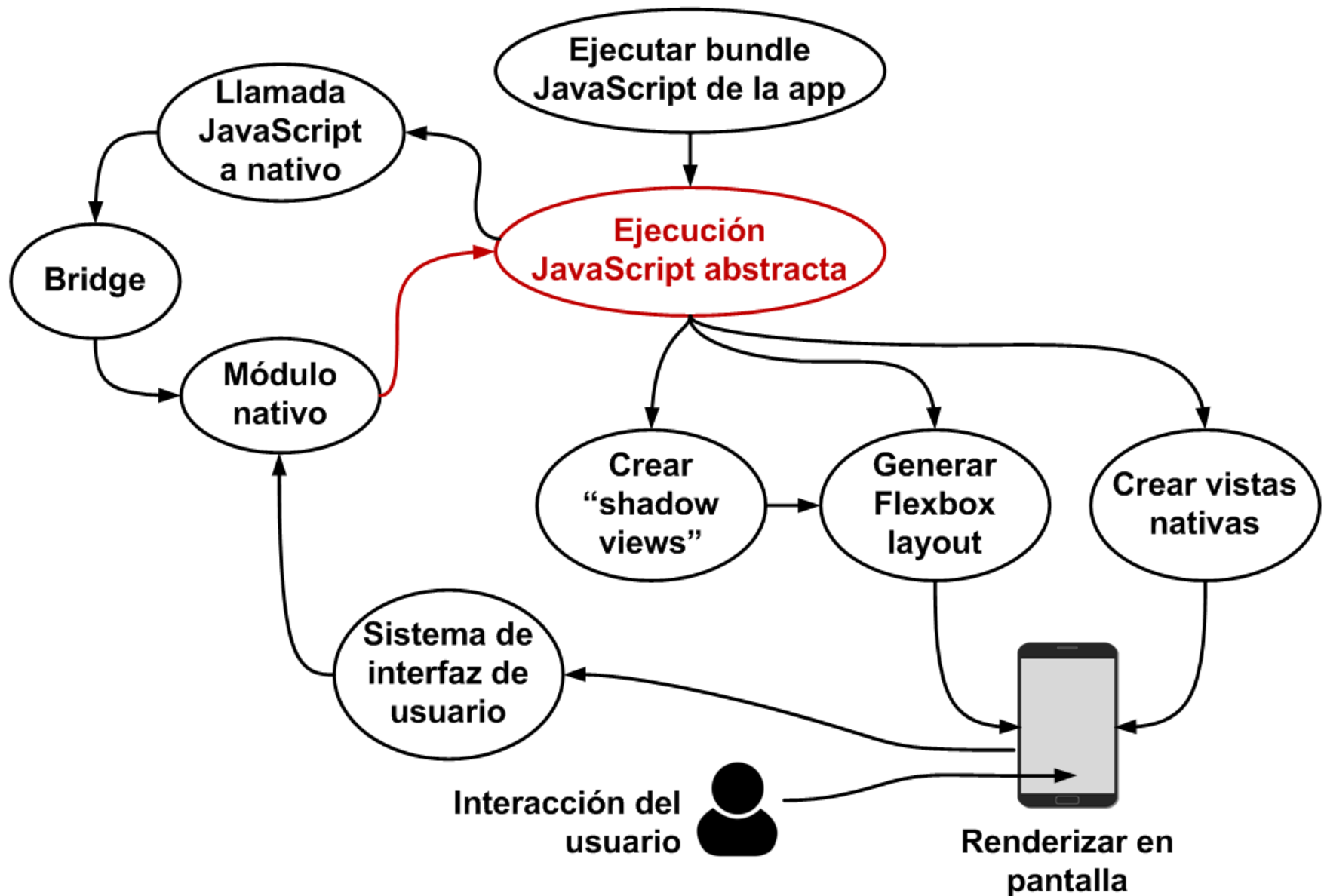
React Native: Ejemplo con Alert



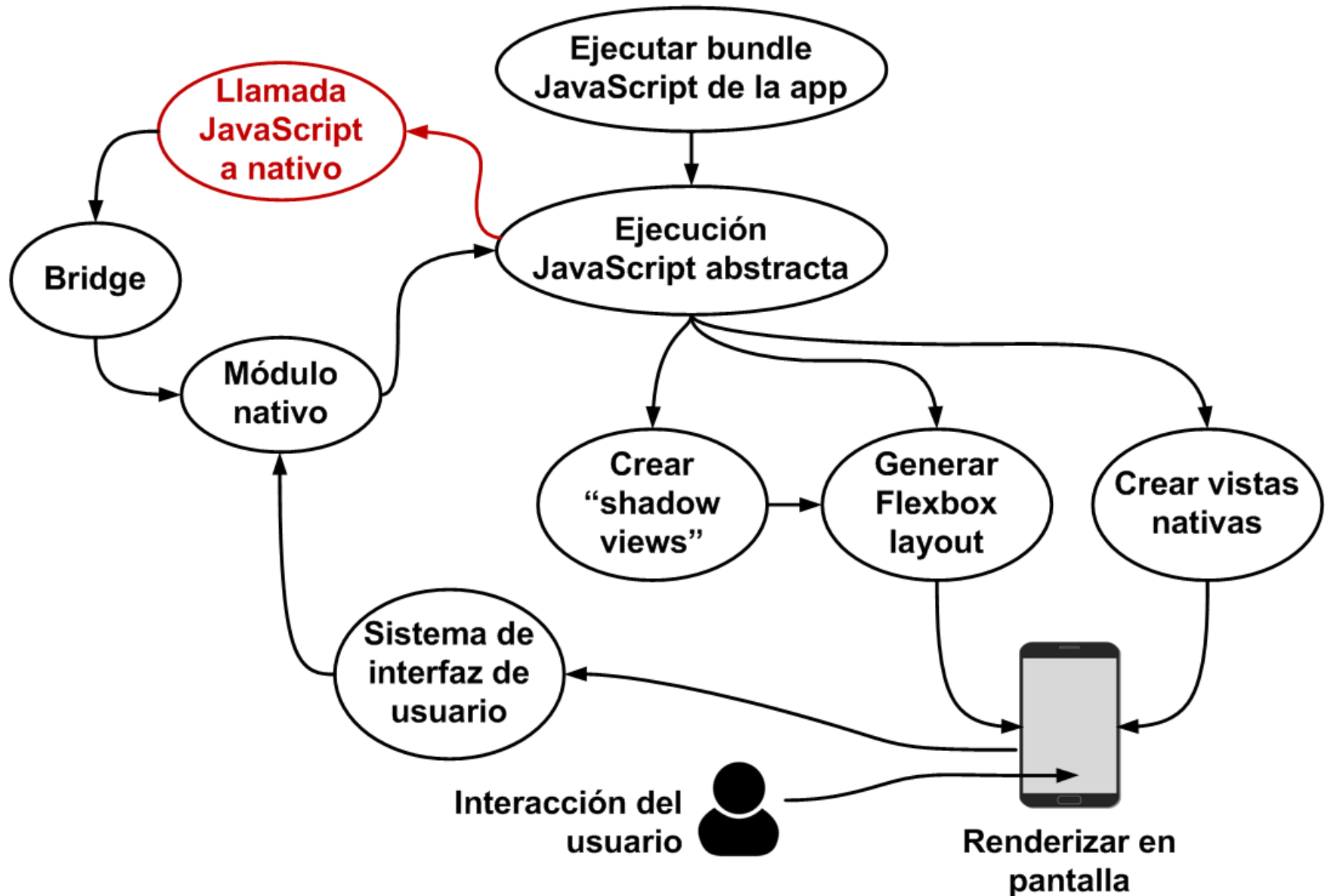
React Native: Ejemplo con Alert



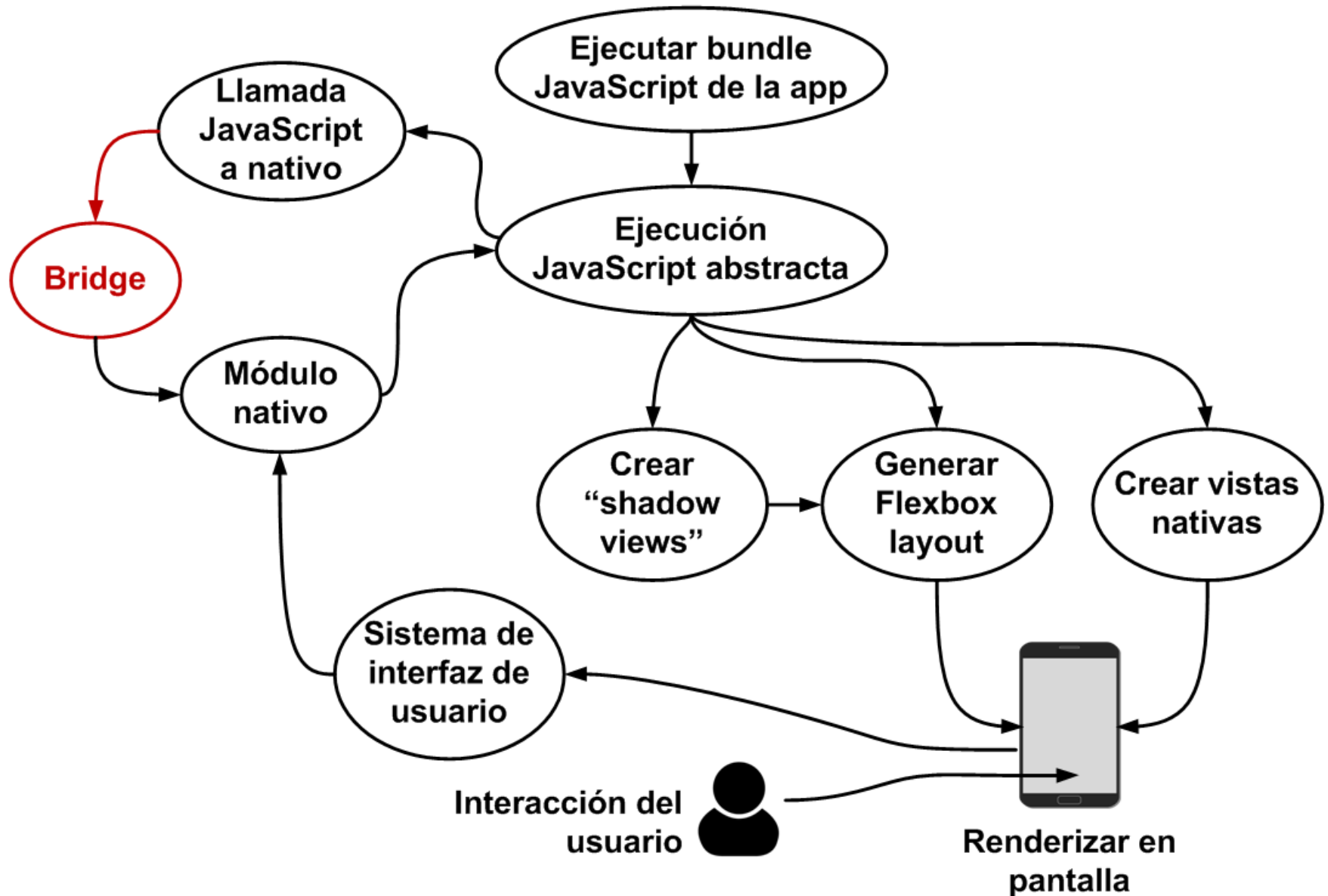
React Native: Ejemplo con Alert



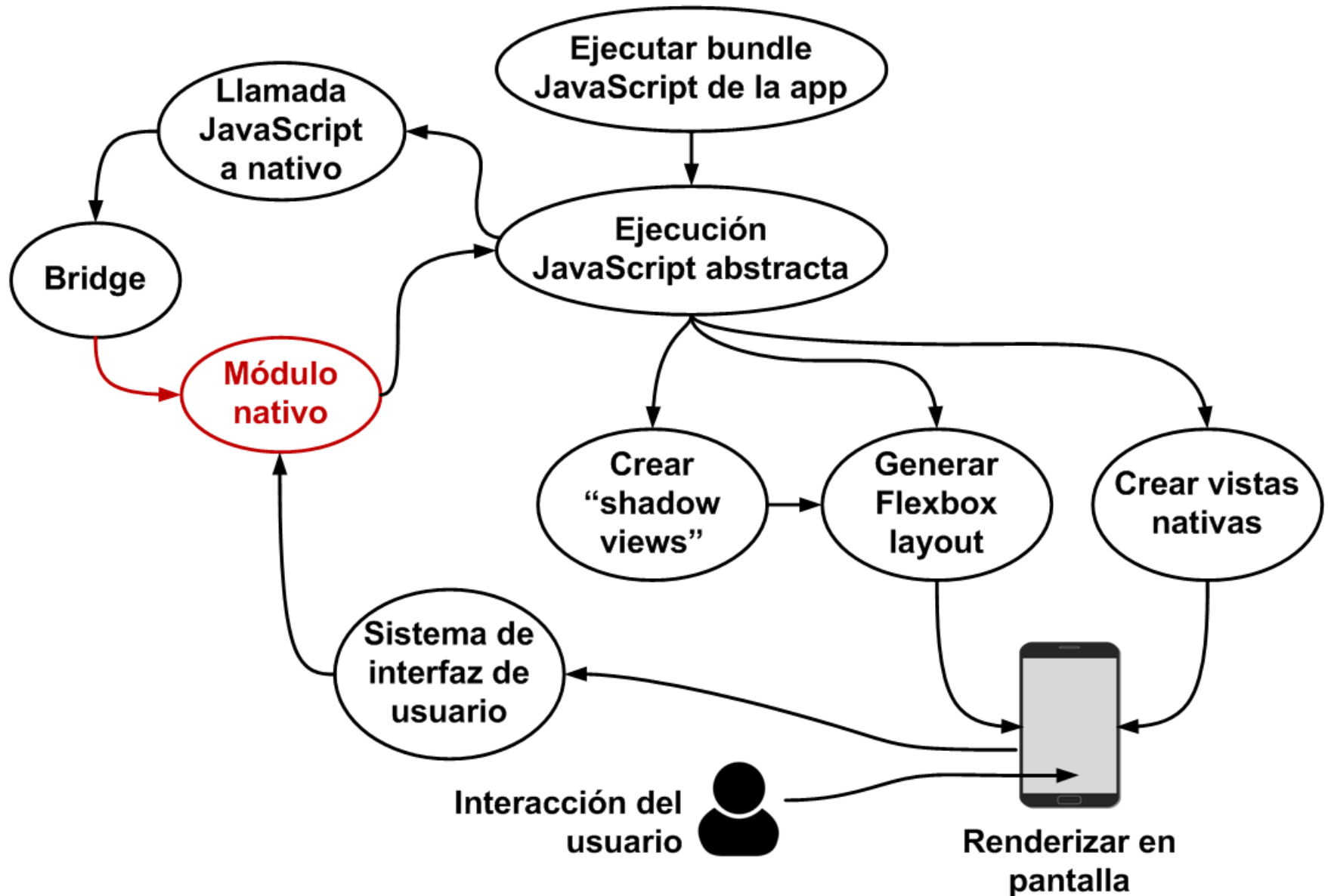
React Native: Ejemplo con Alert



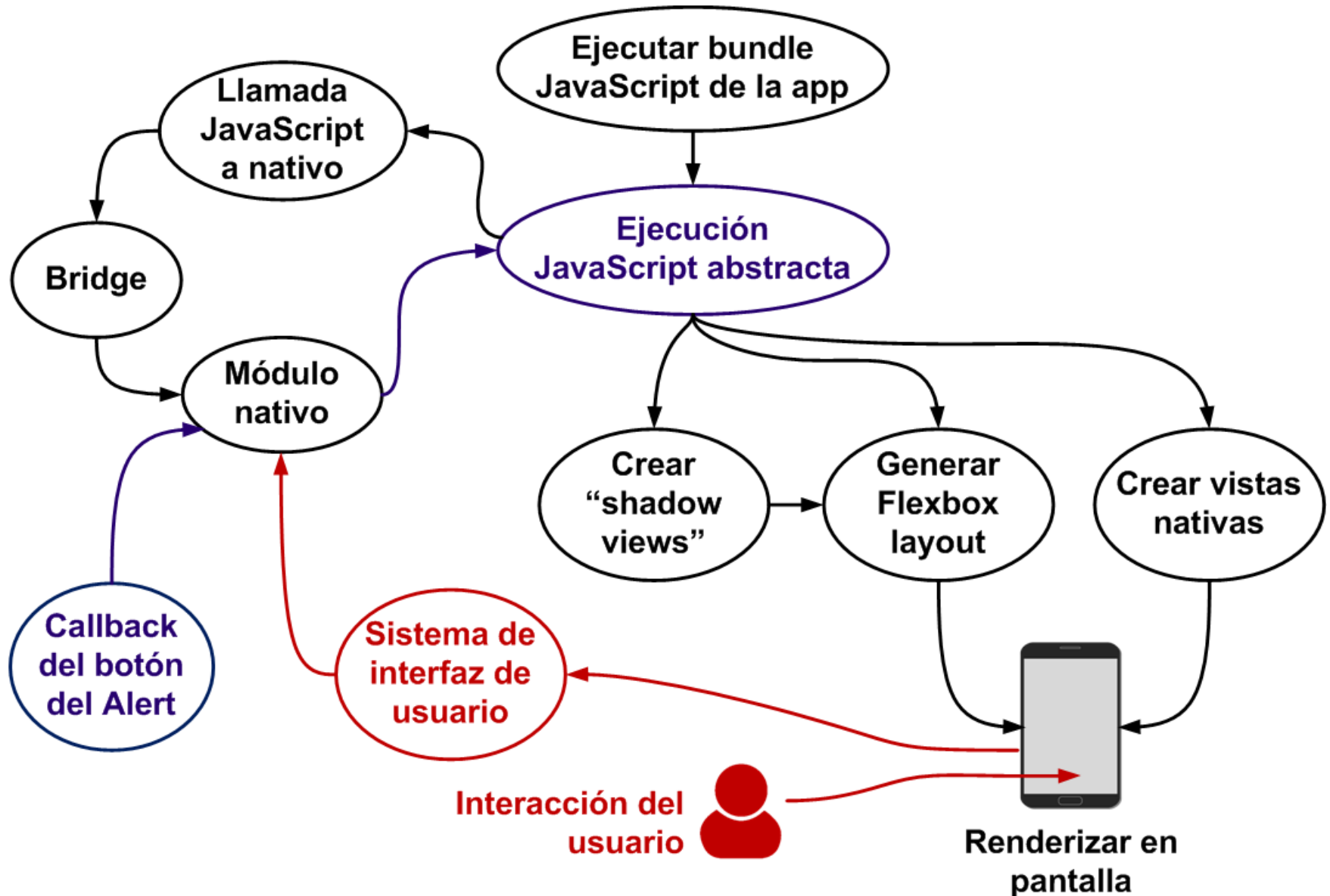
React Native: Ejemplo con Alert



React Native: Ejemplo con Alert



React Native: Ejemplo con Alert



React Native



Componentes y APIs

React Native: Componentes

- En lugar de utilizar **elementos HTML** como React, React Native utiliza **componentes nativos**

React

```
import React from 'react';

export default class Header
  extends React.Component {
  render() {
    return (
      <header>
        {this.props.text}
      </header>
    );
  }
}
```

React Native

```
import React from 'react';
import { Text } from 'react-native';

export default class Header
  extends React.Component {
  render() {
    return (
      <Text>
        {this.props.text}
      </Text>
    );
  }
}
```

React Native: Componentes con estilo

React

Header.jsx

```
import React from 'react';
import './assets/css/hstyles.css';

export default class Header
  extends React.Component {
  render() {
    return (
      <header className="header">
        {this.props.text}
      </header>
    );
  }
}
```

hstyles.css

```
.header{ color: red; }
```

React Native

Header.js

```
import React from 'react';
import { Text } from 'react-native';

export default class Header
  extends React.Component {
  render() {
    return (
      <Text style={{ color: 'red' }}>
        {this.props.text}
      </Text>
    );
  }
}
```

React Native: Ejemplos de componentes

Componente React Native	Posibles elementos HTML (utilizados en React) que puede reemplazar	Componente nativo en	
		Android	iOS
View	Contenedores de elementos <div>, <article>, ...	View ViewGroup	UIView
Text	Cualquier elemento que contenga solo texto , <p>, <h1>, ...	TextView	UILabel
TextInput	<input>	EditText	UITextField UITextView
Image		ImageView	UIImageView
Button TouchableHighlight	Elementos clicables y/o tocables <button>, <a>, ...	Button	UIButton
FlatList	Listas de elementos , , ...	ListView	UITableView
ScrollView	Elementos <div> y similares con altura limitada y scroll	ScrollView	UIScrollView
WebView	<iframe>	WebView	UIWebView

React Native: Componentes

- **React Native** define muchos otros componentes presentes en Android y iOS:
Slider, Switch, CheckBox, Picker, Modal, etc.
- Algunos componentes son específicos de una plataforma:
ProgressBarAndroid, ProgressViewIOS, etc.
- Algunos componentes no proporcionan elementos de interfaz sino funcionalidades.
Por ejemplo: *Navigator* y *RefreshControl*.
- Lista completa de componentes:
<http://facebook.github.io/react-native/docs/components-and-apis.html>

React Native: APIs

- **React Native** proporciona diversas APIs para acceder a funcionalidades nativas, por ejemplo:
 - *Alert*: lanza diálogos de alerta por pantalla
 - *AppState*: indica si la aplicación está activa o si está corriendo en background
 - *AsyncStorage*: sistema de almacenamiento clave-valor
 - *CameraRoll*: acceso a la galería de fotos
 - *Vibration*
 - *Geolocation*
- Lista completa de APIs:
<http://facebook.github.io/react-native/docs/components-and-apis.html>

React Native: StyleSheet

- **StyleSheet** es una **API** que **permite aplicar estilos a componentes React Native** de una manera similar a como CSS permite dar estilo a elementos HTML
- Todos los componentes principales de React Native admiten un atributo ***style***
- Los estilos pueden declararse *inline* o al margen de los elementos, igual que con CSS
- Podemos crear StyleSheets a partir de objetos JavaScript y **reutilizarlos** en la aplicación
- React Native también permite **estilos en cascada** de un modo similar a CSS mediante el uso de arrays de estilos

React Native: StyleSheet

- Los nombres de las propiedades y valores suelen coincidir con CSS pero en lugar de usar "-" se escriben en estilo *CamelCase*
Ejemplo: `"backgroundColor"` en lugar de `"background-color"`
- Los colores siguen la especificación CSS3
- Propiedades de estilo admitidas por cada componente:
 - Documentación oficial
<https://facebook.github.io/react-native/docs>
 - React Native Styling Cheat Sheet
<https://github.com/vhpoet/react-native-styling-cheat-sheet>
 - Ejemplos: View (backgroundColor, opacity, overflow, ...),
Text (color, fontFamily, fontSize, textAlign, lineHeight, ...).

React Native: Componentes

- En lugar de utilizar **elementos HTML** como React, React Native utiliza **componentes nativos**

React

```
import React from 'react';

export default class Header
  extends React.Component {
  render() {
    return (
      <header>
        {this.props.text}
      </header>
    );
  }
}
```

React Native

```
import React from 'react';
import { Text } from 'react-native';

export default class Header
  extends React.Component {
  render() {
    return (
      <Text>
        {this.props.text}
      </Text>
    );
  }
}
```

React Native: Componentes con estilo

React

Header.jsx

```
import React from 'react';
import './assets/css/hstyles.css';

export default class Header
  extends React.Component {
  render() {
    return (
      <header className="header">
        {this.props.text}
      </header>
    );
  }
}
```

hstyles.css

```
.header{ color: red; }
```

React Native

Header.js

```
import React from 'react';
import { Text } from 'react-native';

export default class Header
  extends React.Component {
  render() {
    return (
      <Text style={{ color: 'red' }}>
        {this.props.text}
      </Text>
    );
  }
}
```

React Native: Componentes con estilo

React

Header.jsx

```
import React from 'react';
import '../assets/css/hstyles.css';

export default class Header
  extends React.Component {
  render() {
    return (
      <header className="header">
        {this.props.text}
      </header>
    );
  }
}
```

hstyles.css

```
.header{ color: red; }
```

React Native

Header.js

```
import React from 'react';
import { StyleSheet, Text } from
'react-native';

export default class Header
  extends React.Component {
  render() {
    return (
      <Text style={styles.header}>
        {this.props.text}
      </Text>
    );
  }
}

const styles = StyleSheet.create({
  header: { color: 'red' }
});
```

React Native



Instalación y Hello World

Instalación de React Native

■ Create React Native App

facebook.github.io/react-native/docs/getting-started.html ('Quick Start')

- Aplicación que permite crear proyectos React Native sin necesidad de instalar más herramientas
- Disponible para **Linux, Mac y Windows**
- Permite ejecutar aplicaciones React Native en **dispositivos físicos Android y iOS**
 - Requiere instalar en los dispositivos físicos la aplicación Expo (<https://expo.io>)
- Permite ejecutar aplicaciones React Native en **dispositivos virtuales de Android** y en el **emulador de iOS**
 - Requiere instalar **Android Studio** o **Genymotion** para Android y **Xcode** para iOS

Instalación de React Native

■ Create React Native App

facebook.github.io/react-native/docs/getting-started.html ('Quick Start')

- Dependencias
 - Node ≥ 6
 - npm 3 o 4
- Esta forma de instalar React Native tiene una **limitación** importante: las aplicaciones creadas no pueden tener código nativo, solo **código JavaScript puro**.
- Versiones actuales:
React Native 0.49, React 16.0 y Expo 22.0
- Más información:
github.com/react-community/create-react-native-app

Instalación de React Native

■ React Native CLI (Command Line Interface)

facebook.github.io/react-native/docs/getting-started.html

(Pestaña 'Building Projects with Native Code')

- Forma de instalación original más compleja que la instalación mediante **Create React Native App**
- Esta forma de instalación es necesaria para crear aplicaciones React Native que contengan **código nativo**
- Desarrollo para **Android**: Linux, Mac y Windows
- Desarrollo para **iOS**: solo Mac
- Permite ejecutar aplicaciones React Native en **dispositivos virtuales de Android** y en el **emulador de iOS**
- También permite ejecutar aplicaciones React Native en dispositivos físicos pero de una forma más complicada

Instalación de React Native

- **React Native CLI (Command Line Interface)**

facebook.github.io/react-native/docs/getting-started.html

(Pestaña 'Building Projects with Native Code')

- Dependencias:

- **Node** ≥ 6

- **React Native CLI** (Command Line Interface)

- **Android: ADB** (Android Debug Bridge),
Android Studio (incluye emulador y SDKs de Android) y
JDK (Java Development Kit) ≥ 8

- **iOS: Xcode** (incluye emulador de iOS)

Instalación de React Native en Ubuntu para Android con Create React Native App

1. Instalar Node.js ≥ 6 y npm 3 o 4
<https://nodejs.org/es/download>
2. Instalar Create React Native App
sudo npm install -g create-react-native-app

Para usar un **dispositivo físico de Android**:

1. Instalar la aplicación Expo (<https://expo.io>) en el dispositivo desde Google Play
 2. Conectar el dispositivo a la misma red wifi que el ordenador
- Ahora ya tendríamos todo lo necesario para desarrollar una aplicación React Native para Android y ejecutarla en un dispositivo físico

Instalación de React Native en Ubuntu para Android con Create React Native App

Para usar un **dispositivo virtual de Android**:

1. Instalar Android Studio

<https://developer.android.com/studio/install.html>

2. Abrir Android Studio

```
cd /usr/local/android-studio/bin/  
./studio.sh
```

3. Instalar Android SDK (desde Android Studio)

En el menú seleccionar **Tools -> Android -> SDK Manager**

Instalar:

- SDK Platforms: **Android 6.0 Marshmallow – API Level 23**
- SDK Tools: **Android SDK Platform-Tools 26.0.2 o posterior**

Instalación de React Native en Ubuntu para Android con Create React Native App

Para usar un **dispositivo virtual de Android**:

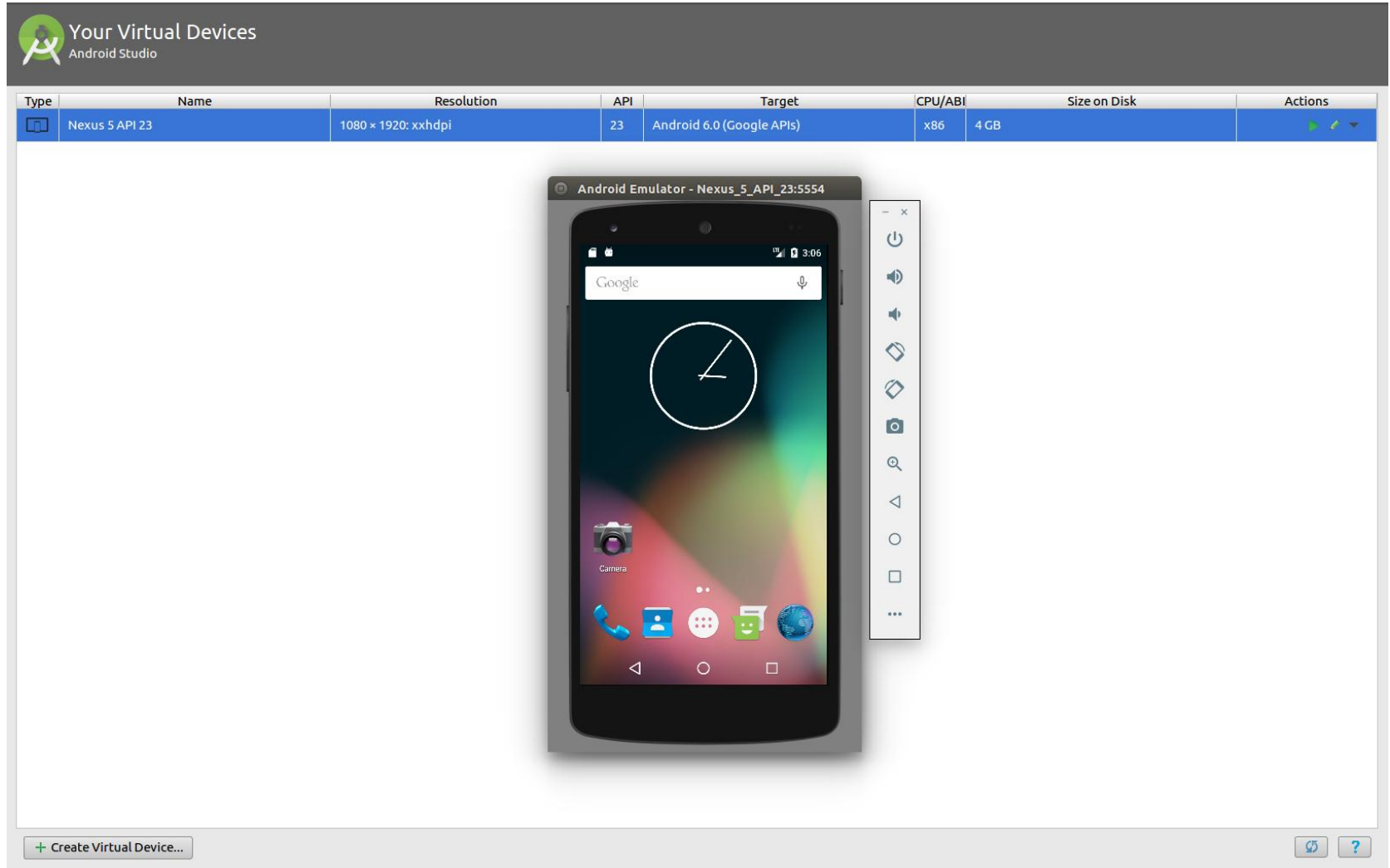
4. Crear un AVD (Android Virtual Device)

- En el menú de Android Studio ir a: **Tools -> Android -> AVD Manager**
- **Create Virtual Device** y elegir en el Wizard:
Nexus 5, API Level 23 (Marshmallow) y nombre **"Nexus 5 API 23"**
En el resto de opciones mantenemos los valores por defecto

5. Iniciar emulador de Android con el AVD creado

- Abrir AVD Manager (Tools -> Android -> AVD Manager)
- Pulsar el botón play o doble clic sobre el AVD creado "Nexus 5 API 23"

Instalación de React Native en Ubuntu para Android con Create React Native App



Instalación de React Native en Ubuntu para Android con Create React Native App

Para usar un **dispositivo virtual de Android**:

6. Instalar ADB (Android Debug Bridge)

sudo apt-get install android-tools-adb

7. Asegurarse de que la versión de adb utilizada por el sistema (*adb version*) y por Android Studio (*cd ~/Android/Sdk/platform-tools ./adb version*) son iguales. En caso contrario ejecutar:

sudo cp ~/Android/Sdk/platform-tools/adb /usr/bin

- Ahora ya tendríamos todo lo necesario para desarrollar y ejecutar una aplicación React Native para Android

React Native: Hello World

1. Crear el proyecto en un directorio de trabajo

sudo create-react-native-app HelloWorld

sudo chown -R {username}:{username} HelloWorld

2. Arrancar el packager (empaquetador) de React Native

cd HelloWorld

sudo npm start

- Cargar aplicación en un dispositivo físico:
 - a) Abrir la aplicación Expo en el dispositivo
 - b) Escanear el código QR proporcionado por el packager
- Cargar aplicación en un dispositivo virtual:
 - a) Arrancar un AVD (Android Virtual Device) o el emulador de iOS
 - b) Pulsar la tecla 'a' en la consola donde se haya arrancado el packager para cargar la aplicación en un AVD o la tecla 'i' para cargarla en el emulador de iOS

React Native: Hello World

```
aldo@guepardo:~/workspace$ cd HelloWorld/  
aldo@guepardo:~/workspace/HelloWorld$ sudo npm start
```

```
> HelloWorld@0.1.0 start /home/aldo/workspace/HelloWorld  
> react-native-scripts start
```

```
15:19:33: Starting packager...  
Packager started!
```

To view your app with live reloading, point the Expo app to this QR code.
You'll find the QR scanner on the Projects tab of the app.



Or enter this address in the Expo app's search bar:

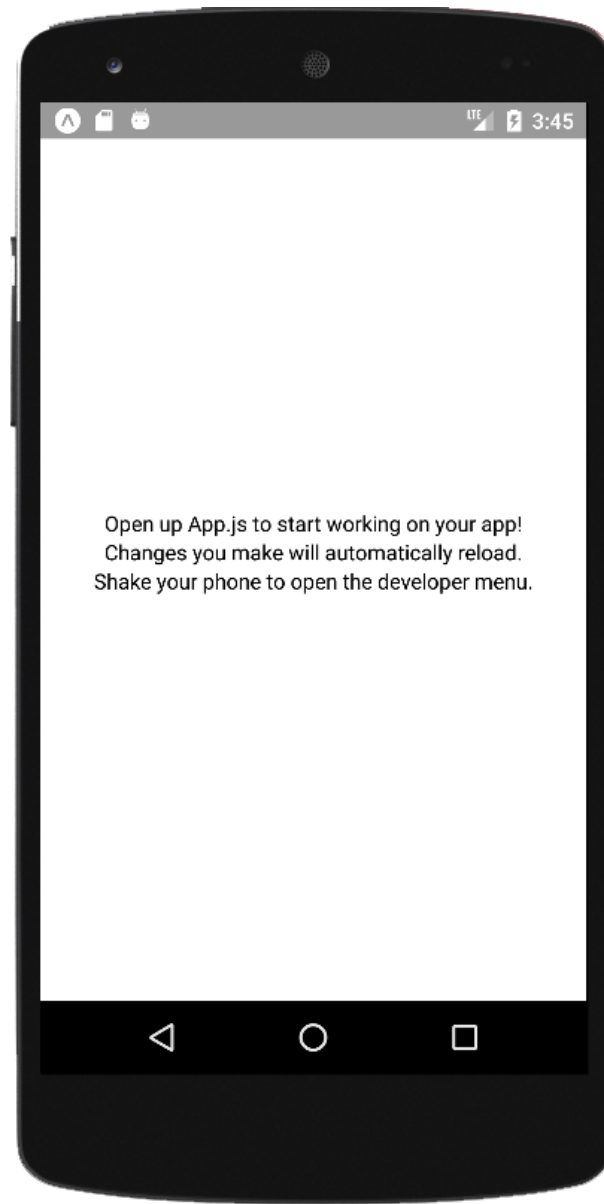
<exp://138.4.4.172:19000>

Your phone will need to be on the same local network as this computer.
For links to install the Expo app, please visit <https://expo.io>.

Logs from serving your app will appear here. Press Ctrl+C at any time to stop.

- › Press **a** to open Android device or emulator, or **i** to open iOS emulator.
- › Press **q** to display QR code.
- › Press **r** to restart packager, or **R** to restart packager and clear cache.
- › Press **d** to toggle development mode. (current mode: **development**)

React Native: Hello World



React Native: Hello World

App.js

```
import React from 'react';
import { StyleSheet, Text, View } from 'react-native';

export default class App extends React.Component {
  render() {
    return (
      <View style={styles.container}>
        <Text>Open up App.js to start working on your app!</Text>
        <Text>Changes you make will automatically reload.</Text>
        <Text>Shake your phone to open the developer menu.</Text>
      </View>
    );
  }
}
```

React Native: Hello World

App.js

```
const styles = StyleSheet.create({  
  container: {  
    flex: 1,  
    backgroundColor: '#fff',  
    alignItems: 'center',  
    justifyContent: 'center'  
  },  
});
```

Modificación de Hello World


- Abrimos el proyecto con un editor de código como **Sublime Text**
Existe un plugin de Sublime que soporta JavaScript ES6 y las extensiones JSX de React: <https://github.com/babel/babel-sublime>
- Abrimos el fichero **App.js** situado en la raíz del proyecto y lo modificamos de la siguiente manera:

```
import React from 'react';
import { Text, View } from 'react-native';

export default class App extends React.Component {
  render() {
    return (
      <View style={{ flex: 1, alignItems: 'center', justifyContent: 'center' }}>
        <Text style={{ fontSize: 30 }}>Hello World!</Text>
      </View>
    );
  }
}
```

App.js

Hello World en Sublime Text con babel-sublime



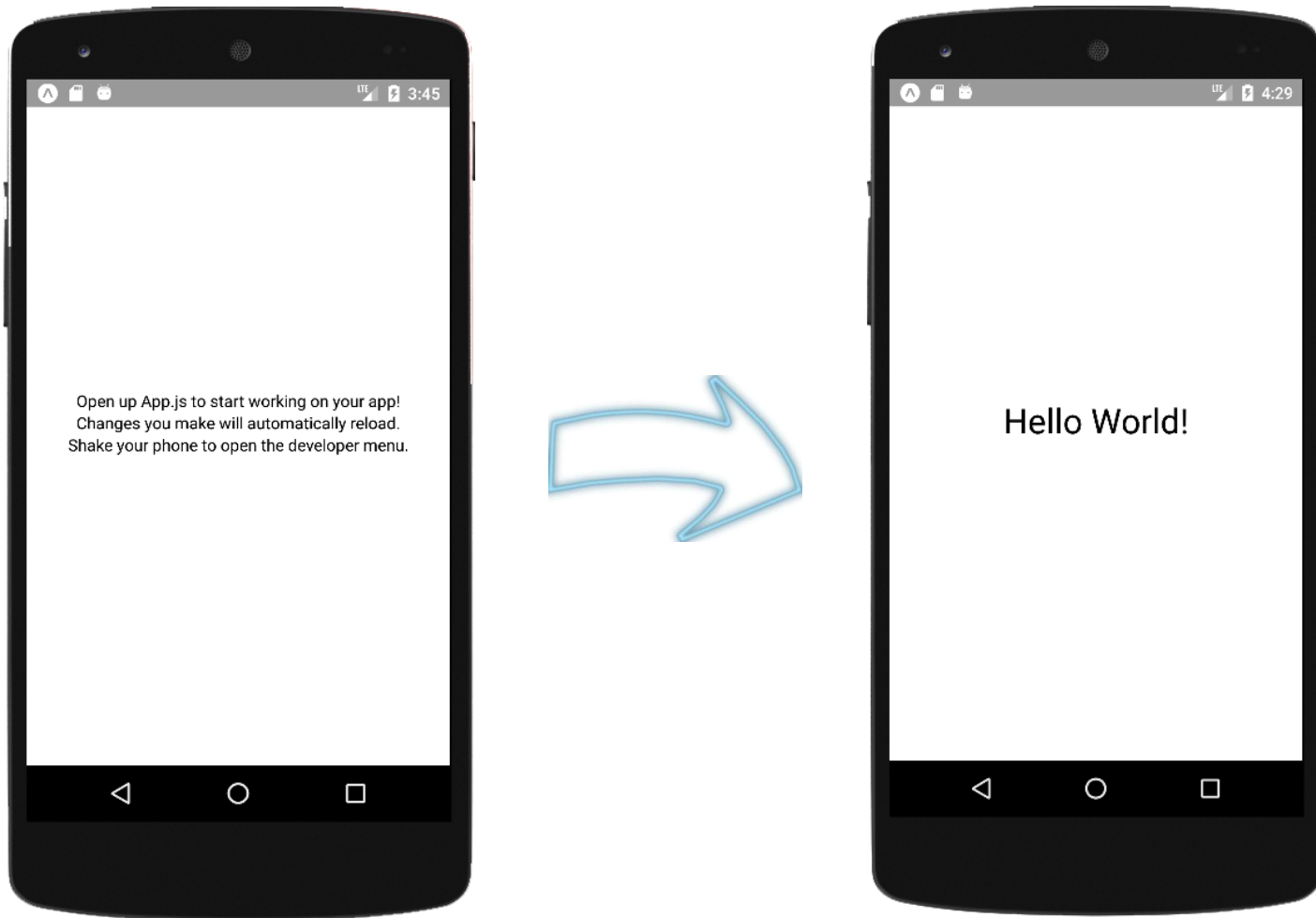
The image shows a screenshot of the Sublime Text editor interface. On the left, the 'FOLDERS' sidebar is open, displaying a file tree for a project named 'IWEB_ReactNative'. The files listed include '.expo', 'node_modules', '.buckconfig', '.flowconfig', '.gitignore', 'App.js' (which is selected), 'App.test.js', 'README.md', 'app.json', and 'package.json'. The main editor area shows the content of 'App.js', which is a JavaScript file using Babel syntax for React Native. The code defines a class 'App' that extends 'React.Component' and implements a 'render()' method. Inside 'render()', it returns a JSX element consisting of a 'View' component containing a 'Text' component with the text 'Hello World!'. The 'View' component has a style with 'flex:1', 'alignItems: 'center'', and 'justifyContent: 'center''. The 'Text' component has a style with 'fontSize: 30'. The code is line-numbered from 1 to 12. The status bar at the bottom indicates 'Line 12, Column 2', 'Tab Size: 4', and 'JavaScript (Babel)'.

```
1 import React from 'react';
2 import { Text, View } from 'react-native';
3
4 export default class App extends React.Component {
5   render() {
6     return (
7       <View style={{ flex:1, alignItems:'center', justifyContent:'center' }}>
8         <Text style={{ fontSize:30 }}>Hello World!</Text>
9       </View>
10    )
11  }
12 }
```

Line 12, Column 2 Tab Size: 4 JavaScript (Babel)

Modificación de Hello World

Si pulsamos la tecla **R** dos veces (emulador de Android) o **comando+R** (emulador de iOS) la aplicación se volverá a cargar con los cambios realizados



Configurar StatusBar de Android para Expo

- Existe un **fallo en la aplicación Expo** que provoca que la **barra de estado de Android** se muestre por encima de la aplicación React Native
- Este fallo solo se produce si hemos creado la aplicación React Native mediante **Create React Native App**
- Una posible solución:
 - Añadir configuración explícita para la barra de estado de Android en el fichero **app.json**
<https://docs.expo.io/versions/latest/guides/configuring-statusbar.html>
 - Resetear el empaquetador React Native y borrar la cache (tecla **R**)
 - Cerrar la aplicación React Native en el dispositivo y volver a cargarla

Configurar StatusBar de Android para Expo

app.json

```
{
  "expo": {
    "sdkVersion": "22.0.0",
    "androidStatusBar": {
      "backgroundColor": "#cccccc"
    }
  }
}
```



Iniciar una aplicación React Native

1. Arrancar el packager de React Native

Abrir un nuevo terminal y ejecutar en el directorio del proyecto:

sudo npm start

- **Para cargar la aplicación en un dispositivo físico:**

Abrir la aplicación Expo en el dispositivo y escanear el código QR proporcionado

- **Para cargar la aplicación en un dispositivo virtual:**

- a) Arrancar un AVD (Android Virtual Device) o el emulador de iOS

- b) Pulsar la tecla 'a' en la consola donde se haya arrancado el packager para cargar la aplicación en un AVD o la tecla 'i' para cargarla en el emulador de iOS

Recargar una aplicación React Native

- Para **recargar** la aplicación:
 - Emulador Android: pulsar dos veces la tecla R
 - Emulador iOS: comando+R
 - Cualquier dispositivo: abrir menú de desarrollo y seleccionar opción 'Reload'
- Para abrir el **menú de desarrollo**:
 - Emulador Android: ctrl+M (Linux y Windows), comando+M (Mac)
 - Emulador iOS: comando+D
 - Dispositivos físicos: gesto de sacudida
- También puede habilitarse la funcionalidad **Hot Reloading** desde el menú de desarrollo para cargar los nuevos cambios automáticamente

¿Preguntas?

Aldo Gordillo

agordillo@dit.upm.es