



Universidad
Rey Juan Carlos

TECNOLOGÍAS WEB EN PLATAFORMA ROBÓTICA JDEROBOT



Aitor Martínez Fernández

a.martinezfern@alumnos.urjc.es

8 de abril de 2016

Índice

- Introducción
- Objetivos
- Infraestructura
- Diseño e implementación
- Experimentos
- Conclusiones

Introducción

Robótica y Web



NETFLIX

Podrás ver series y películas cuando quieras, donde quieras.

Planes desde 7,99 € al mes.

Comenzar el mes gratis

Más información sobre Netflix a continuación

Sin compromiso, cancela en línea en cualquier momento

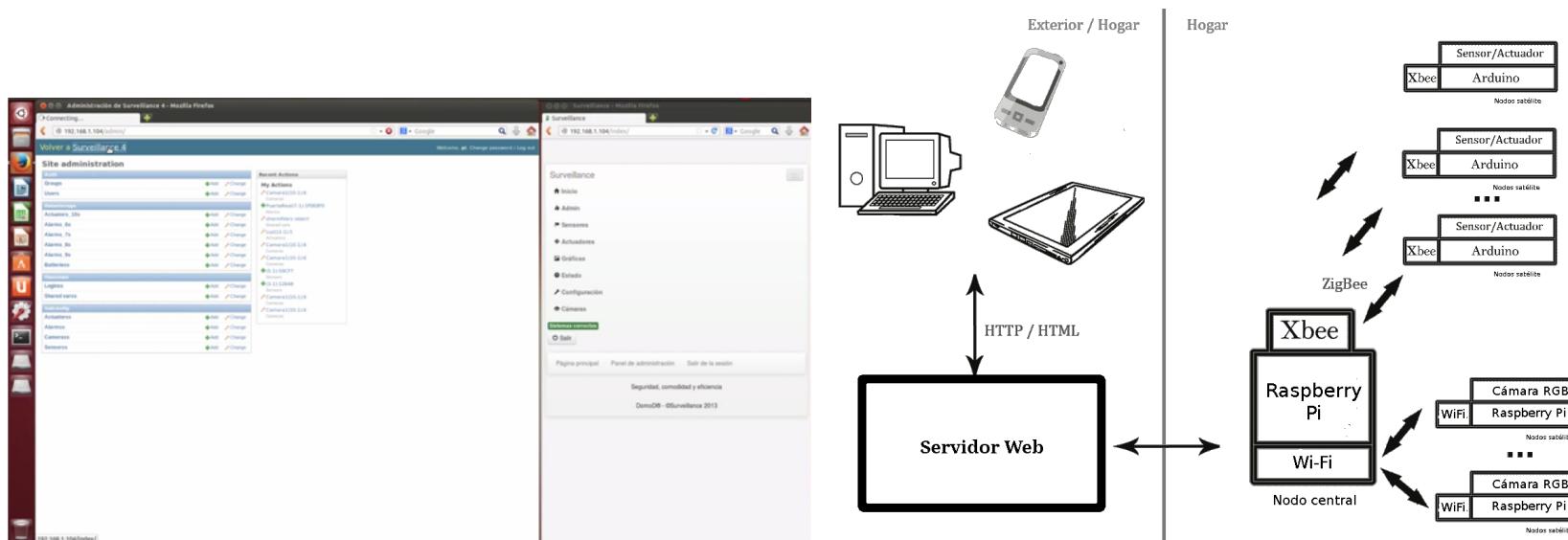
Mucha que ver

Disfruta donde quieras

Elegir tu precio

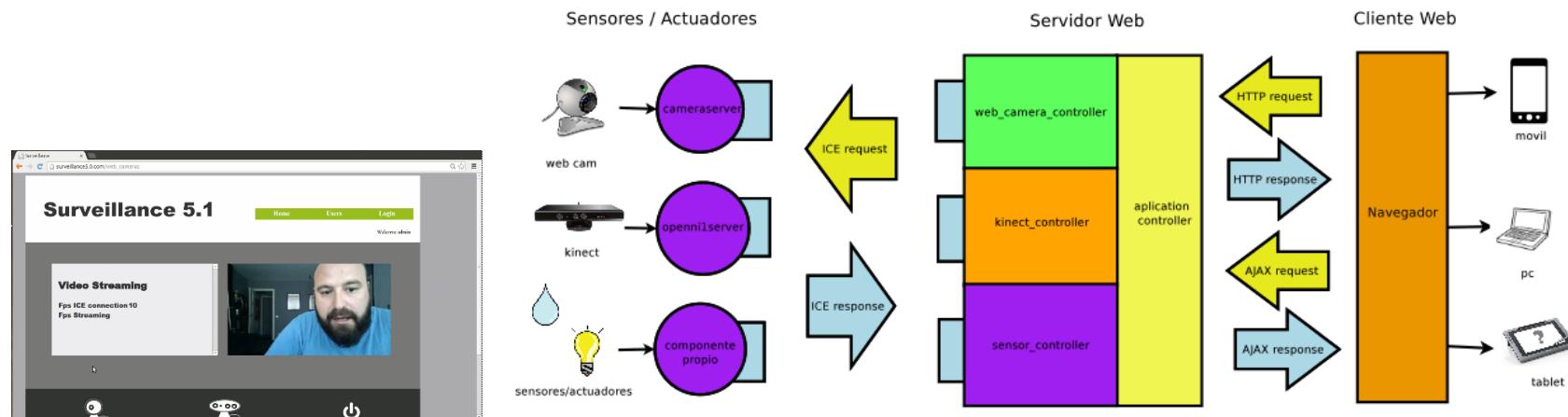
Antecedentes: Surveillance 4.0

- Es el Proyecto fin de carrera de Daniel Castellano.
- Está desarrollado con Django (Python).
- Un servidor web intermedio se conecta a los servidores ICE.



Antecedentes: Surveillance 5.1

- Es el TFG de Edgar Barrero.
- Está desarrollado con Ruby on Rails.
- Un servidor web intermedio se conecta a los servidores ICE.



Objetivo

Crear versiones web (multiplataforma) de varios clientes de JdeRobot muy utilizados sin tener servidor intermedio.

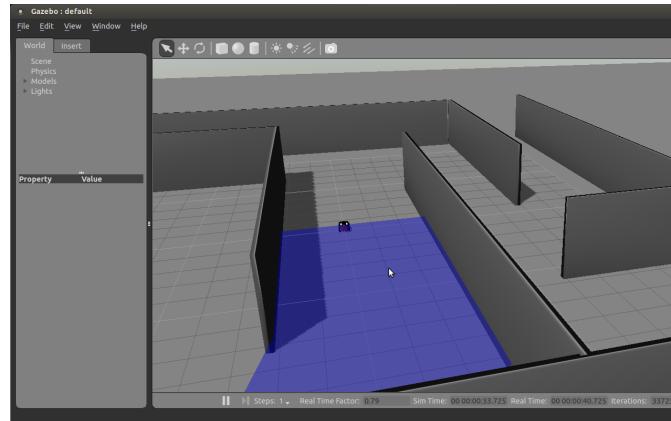
Sub-objetivos:

- Representación de datos de sensores: CameraView, RgbdViewer
- Interactuación con actuadores: KobukiViewer, UavViewer
- Añadir comportamiento autónomo
- Diseño de la página web

Infraestructura

Gazebo e ICE

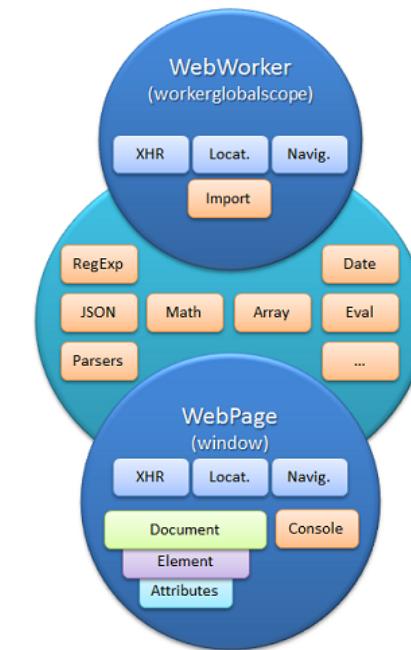
- **Gazebo**: Simula sensores, actuadores, robots,... en mundos virtuales.
- **ICE**: Permite comunicaciones *cross-language* y *cross-platform*.



HTML5 y WebWorker

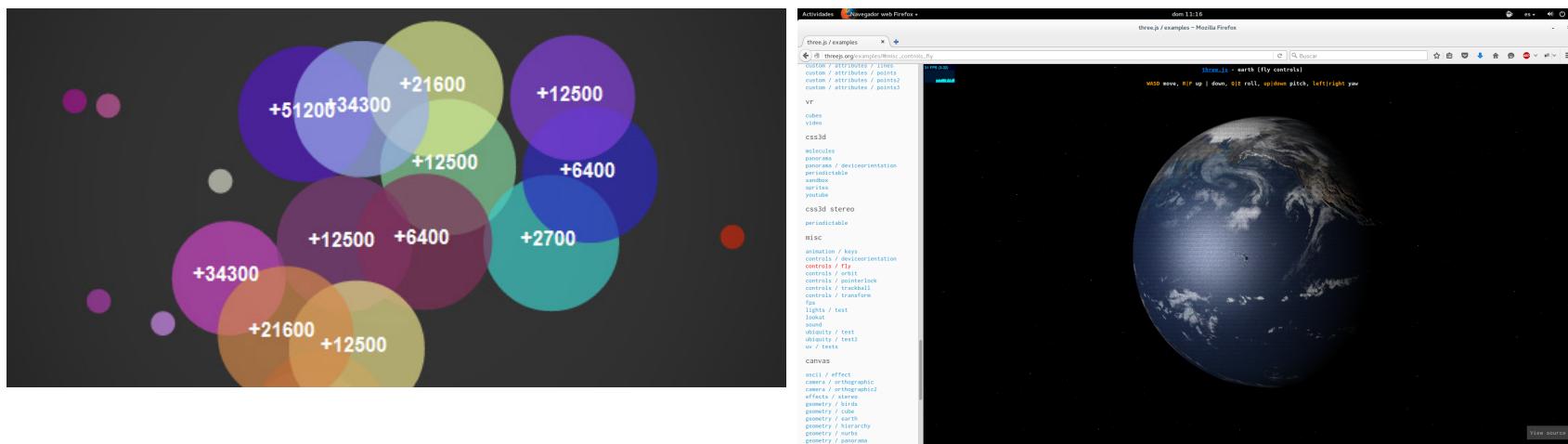
- **HTML5:** Establece una serie de nuevos elementos y atributos que reflejan el uso típico de los sitios web modernos (Canvas, Video,...).
- **WebWorker:** Permite ejecutar *scripts* en segundo plano sin interferir con la interfaz de usuario.

```
<video src="..."></video>
```



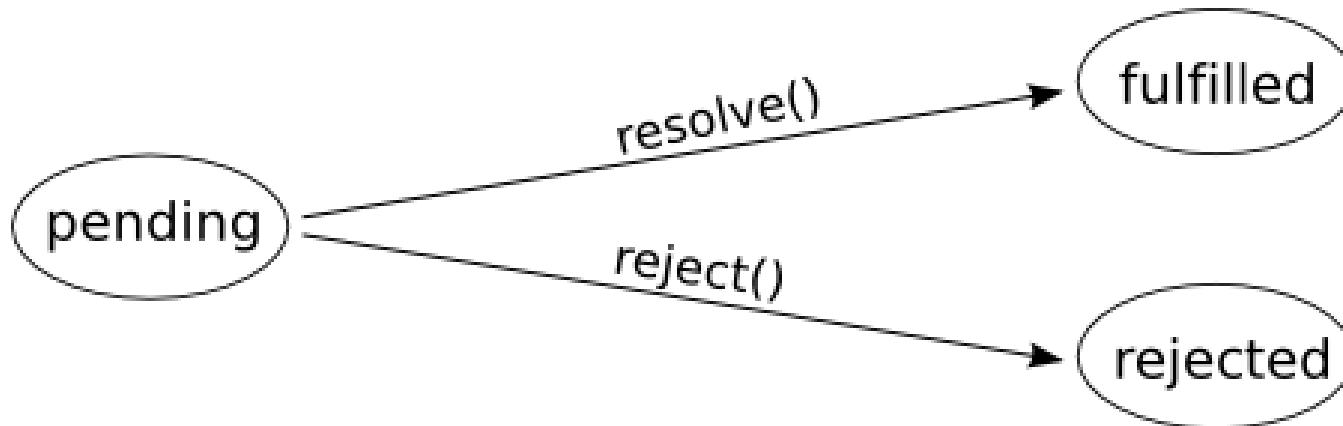
WebGL y Canvas en HTML5

- **WebGL:** Permite mostrar gráficos en 3D acelerados por hardware (GPU) en páginas web sin la necesidad de plug-ins.
- **Canvas:** Puede usarse para dibujar gráficos a través JavaScript. Composición de fotos, animaciones, procesamiento de vídeo en tiempo real.

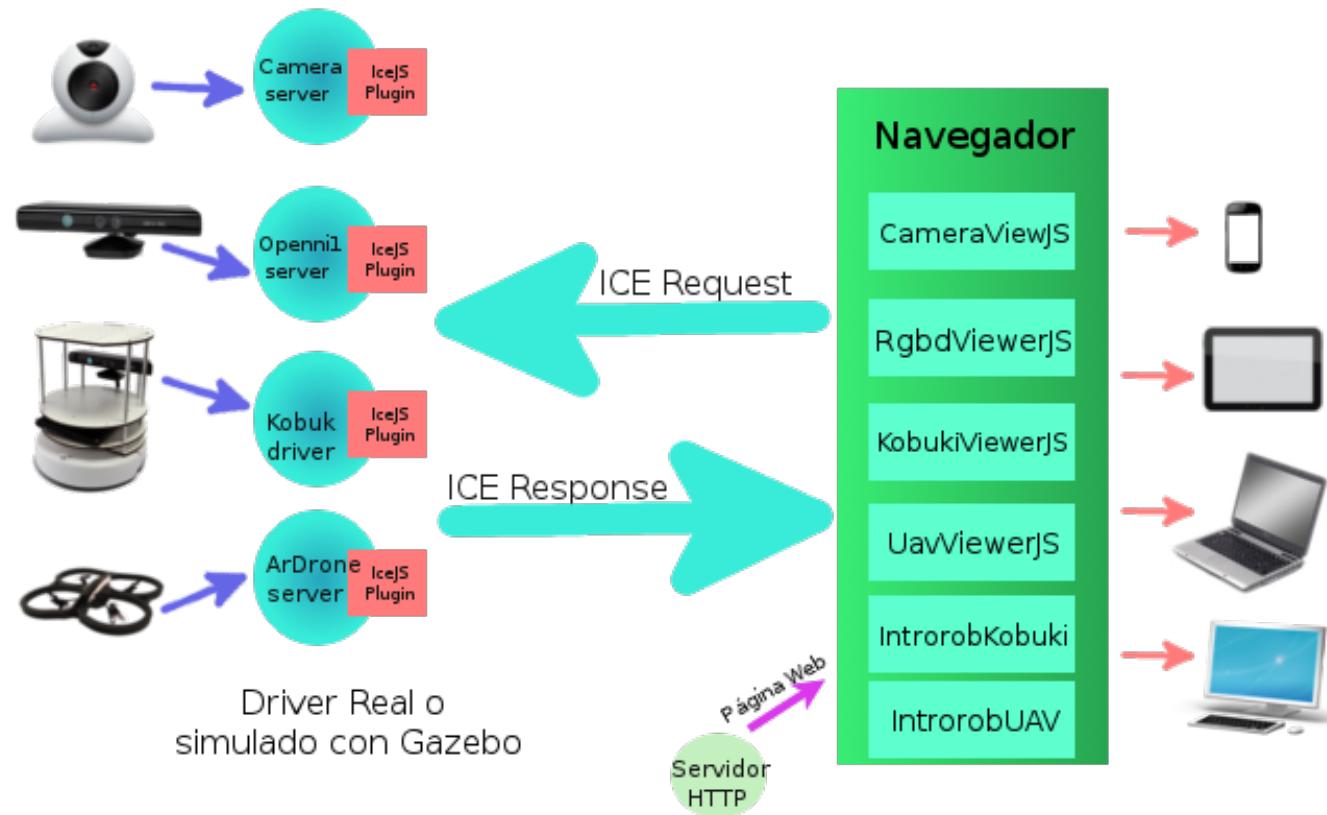


Javascript 6 y Promesas

- **JavaScript6 o ECMAScript 2015:** es la versión actual de la especificación del lenguaje ECMAScript conocida simplemente como “ES6”.
- **Promise:** Devuelve una promesa de tener un valor en algún momento en el futuro.

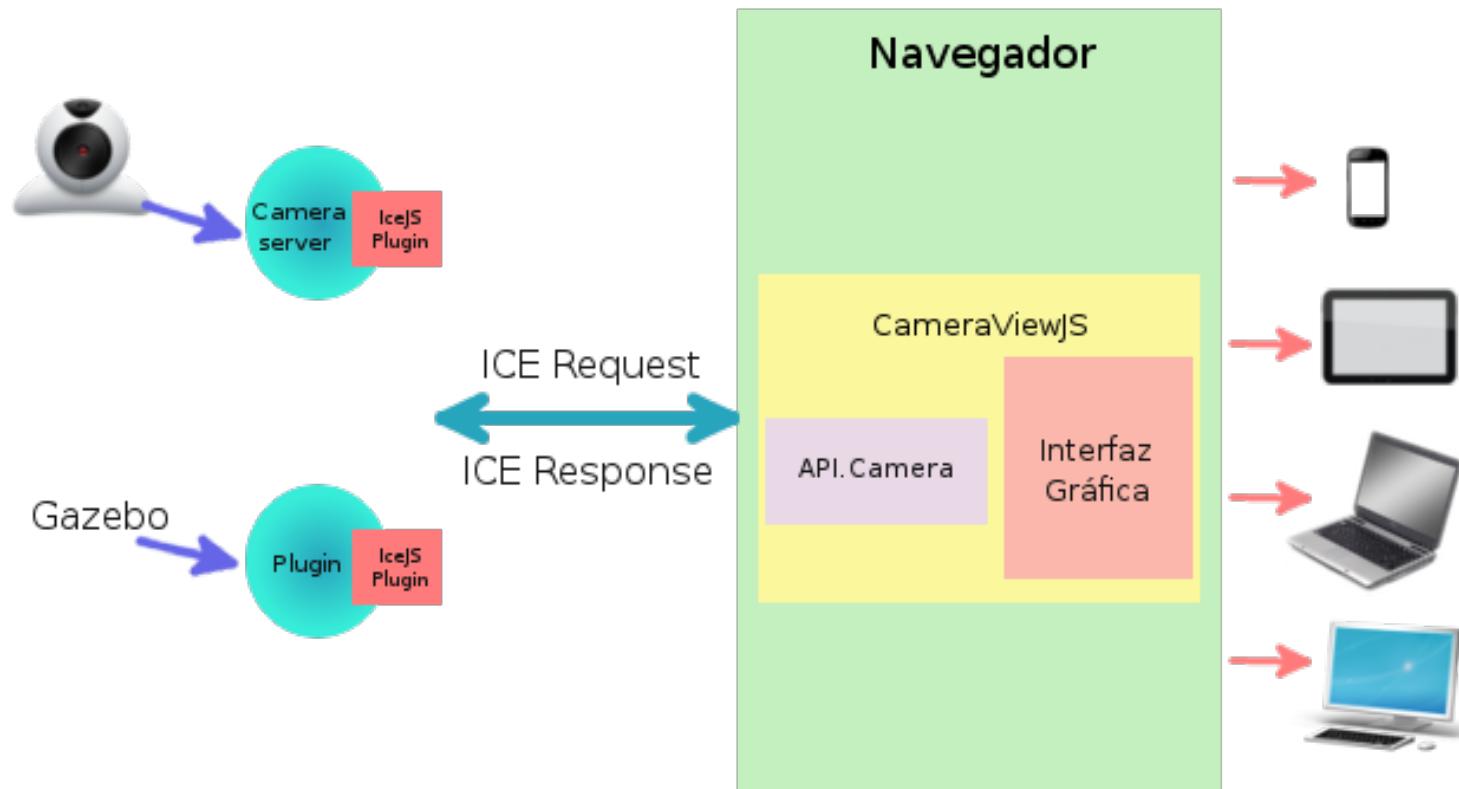


Diseño e implementación



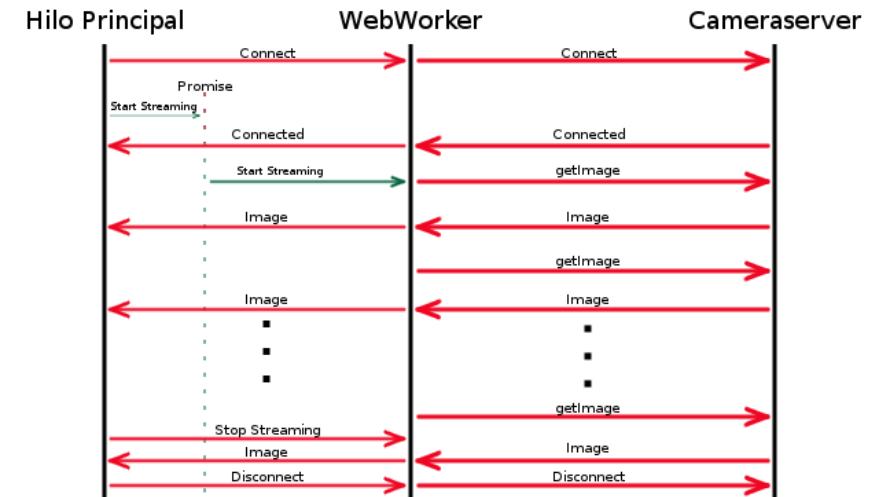
CameraViewJS

Este cliente muestra la información recibida de una webcam.



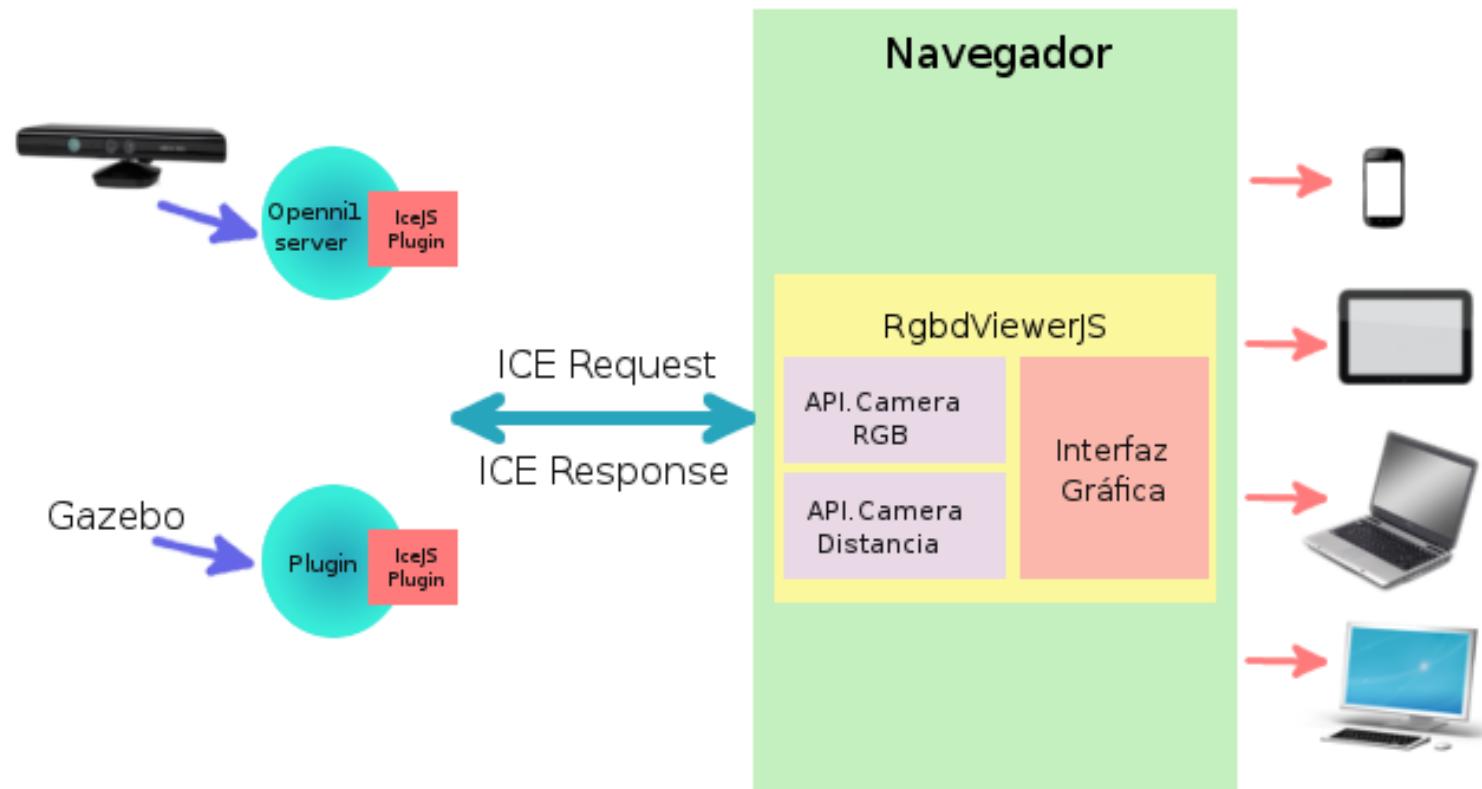
CameraViewJS

- Cuando se pone en funcionamiento, se inician los *widgets* y se crea el *WebWorker*
- Mediante una promesa se espera a tener establecida la conexión con el servidor
- Cada vez que se recibe una imagen se muestra en el *canvas*.



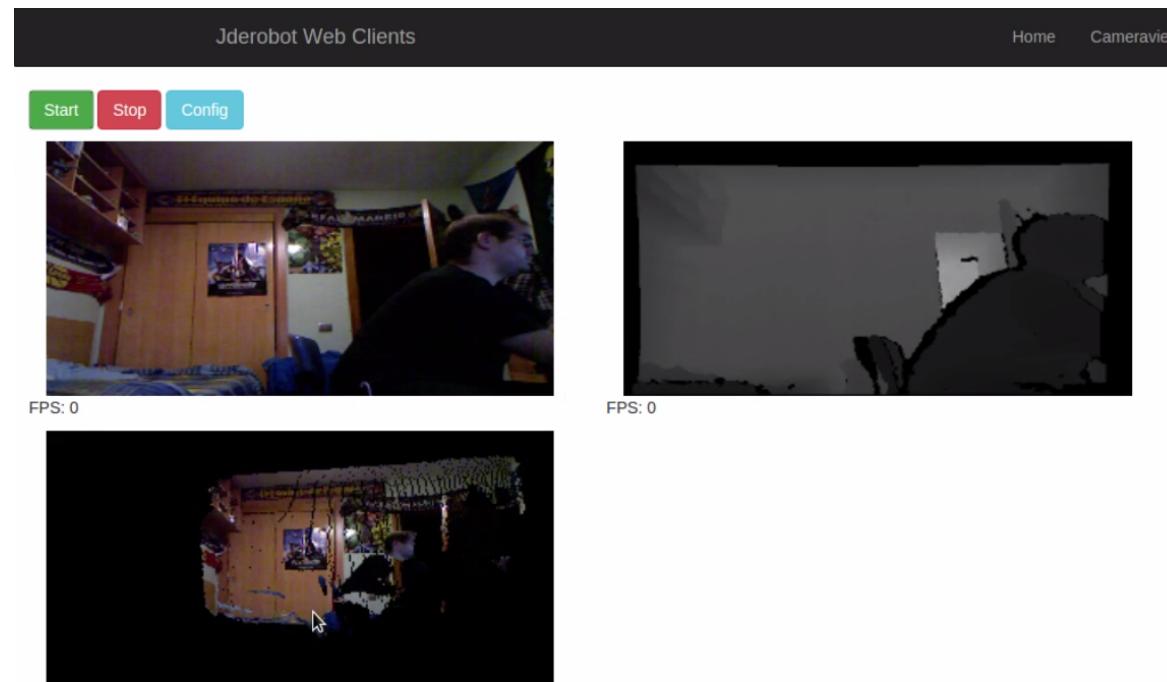
RgbdViewerJS

Este cliente recibe información de un sensor Kinect.



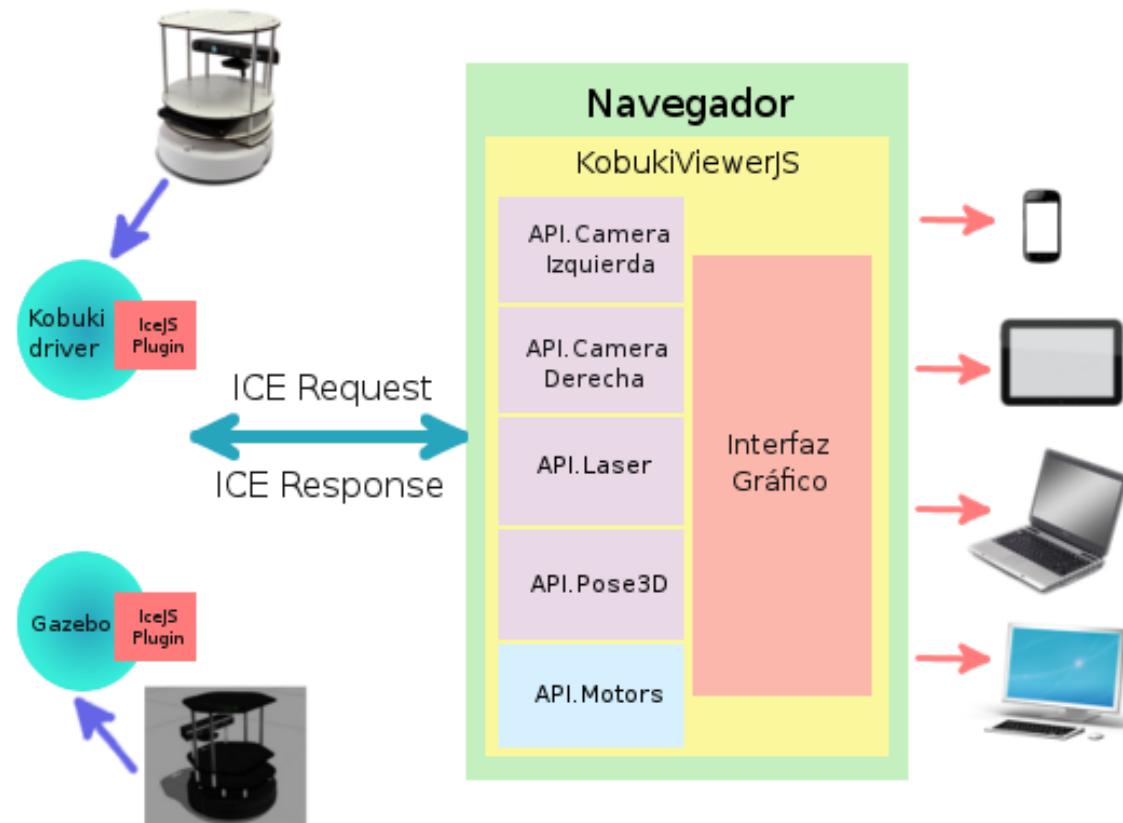
RgbdViewerJS

- Imágenes como en CameraViewJS.
- Cuando se han recibido imágenes de las dos cámaras se crea la nube de puntos 3D y se muestra en el modelo 3D.

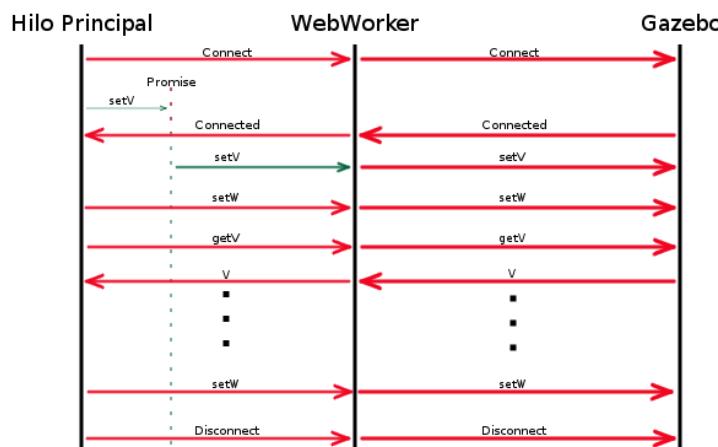
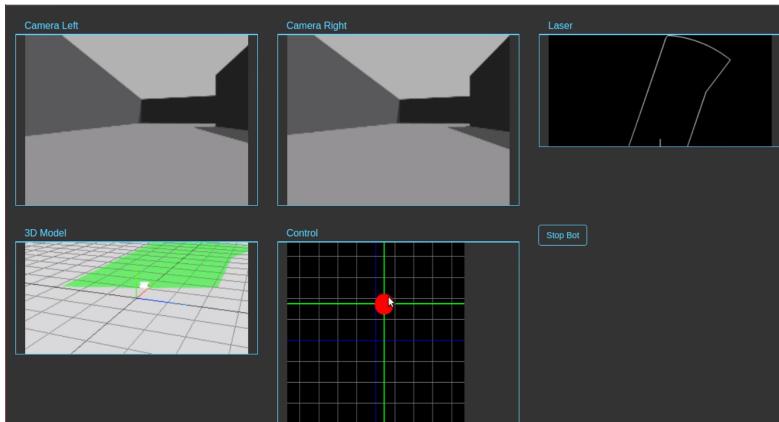


KobukiViewerJS

Este cliente permite teleoperar un robot Kobuki.



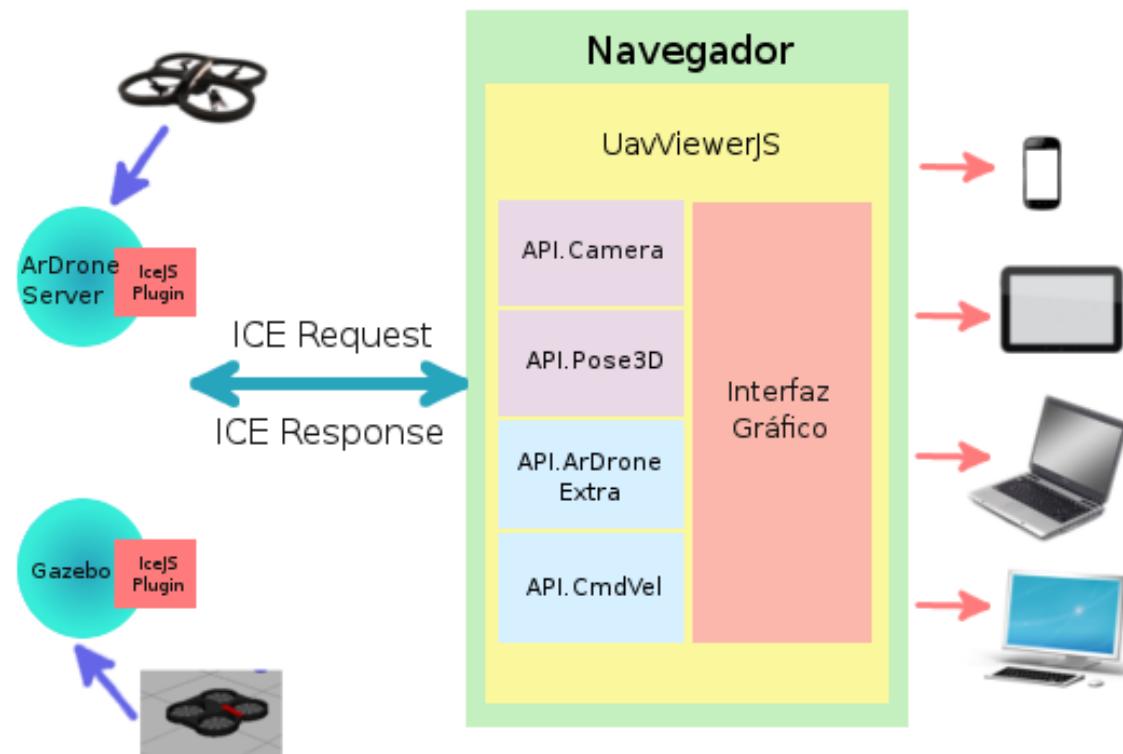
KobukiViewerJS



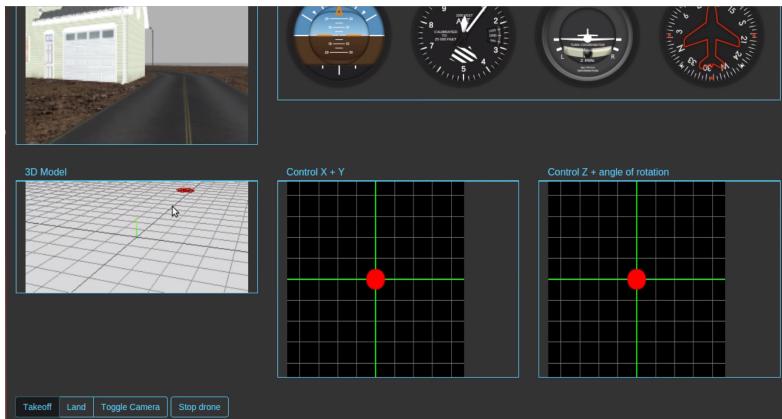
- Imágenes como en CameraViewJS.
- Cada vez que se recibe un Pose3D se modifica la posición y orientación del robot en el modelo 3D.
- Cuando se recibe información del láser se crea una imagen 2D y otra 3D para poderla agregar al modelo.
- Cada vez que se mueve el control se envían las órdenes de velocidad.

UavViewerJS

Este cliente permite teleoperar un drone ArDrone 2 de parrot.



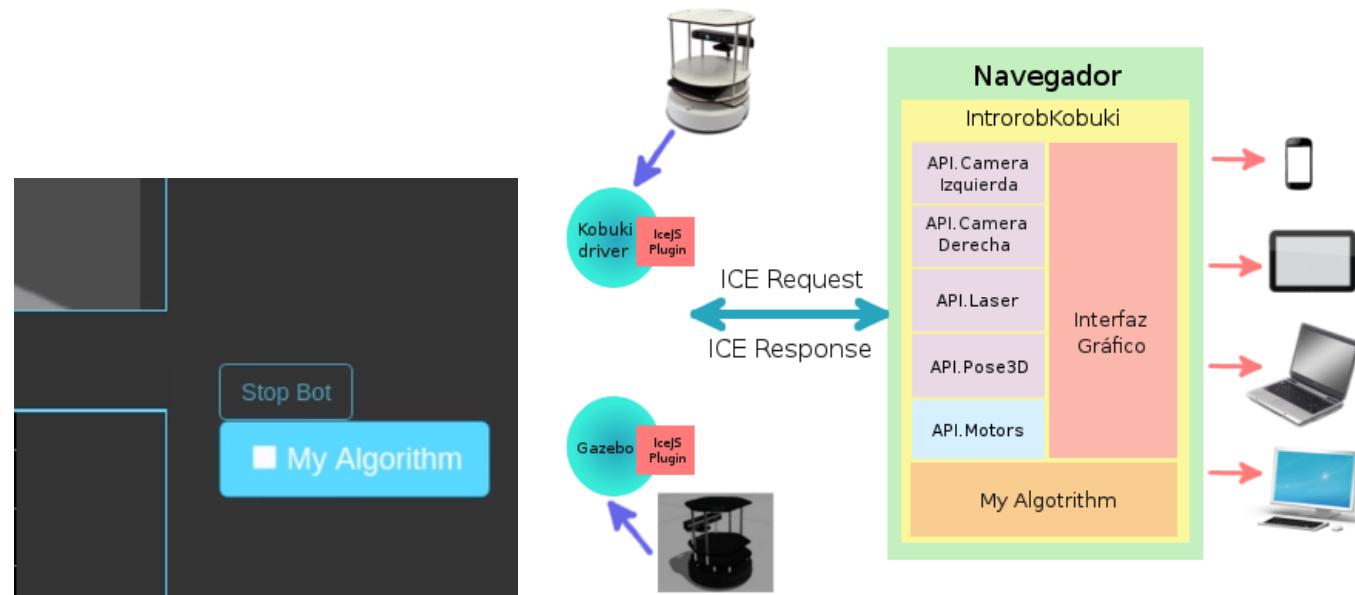
UavViewerJS



- Imágenes como en CameraViewJS.
- Cada vez que se recibe un Pose3D se modifica la posición y orientación del drone en el modelo 3D y se modifican los indicadores de vuelo.
- Cada vez que se mueven los controles se envían las órdenes de velocidad al drone.
- Cuando se pulsan los botones se envían al drone las órdenes de despegar, aterrizar,...

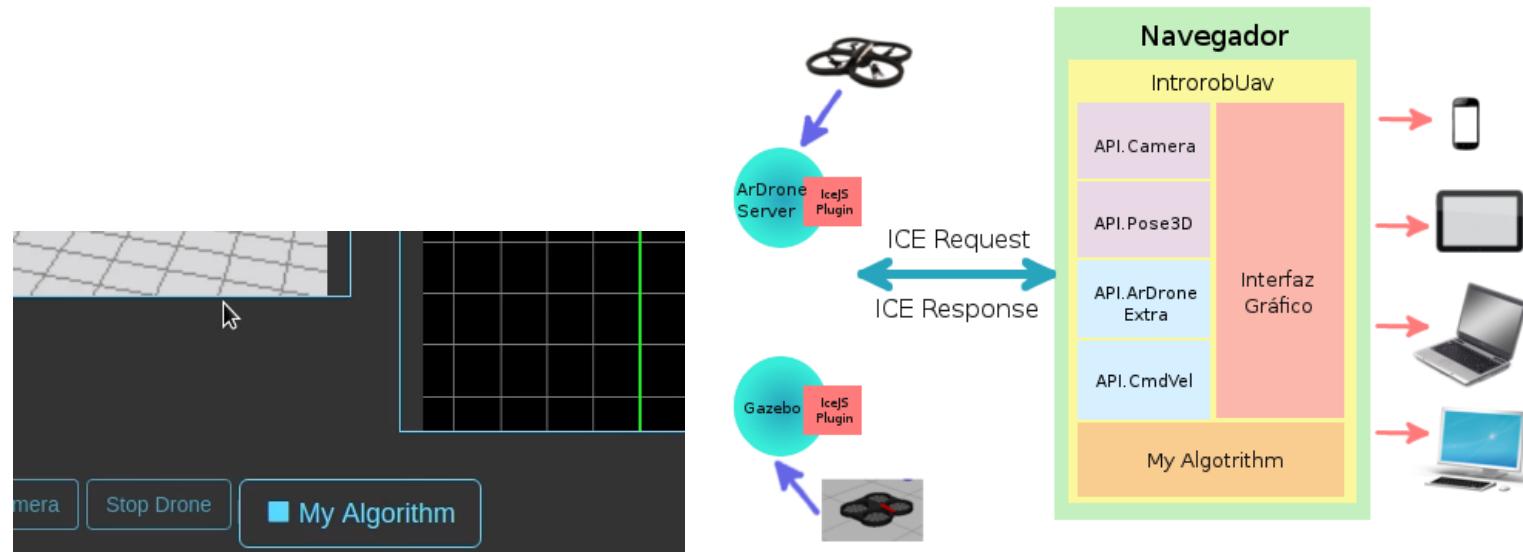
IntrorobKobukiJS

- Igual que KobukiViewerJS pero añadiendo la posibilidad de añadir comportamiento autónomo programado en JavaScript.
- Cada 10ms (usando setInterval) se envían los datos de los sensores a un worker y éste devuelve las órdenes para los actuadores.



IntrorobUavJS

- Igual que UavViewerJS pero añadiendo la posibilidad de añadir comportamiento autónomo programado en JavaScript.
- Cada 10ms (usando setInterval) se envían los datos de los sensores a un worker y éste devuelve las órdenes para los actuadores.



Experimentos

- CameraViewJS.
- RgbdViewerJS.
- KobukiViewerJS.
- UavViewerJS.
- IntrorobKobukiJS.
- IntrorobUavJS.
- Rendimiento Temporal.



Rendimiento Temporal

Cameraserver		Clientes			FPS cameraView	FPS CameraViewJS
Ubicación	Mbps bajada/subida	Tipo	Ubicación	Mbps bajada/subida		
Mismo PC	-	PC	Mismo PC	-	9	8
Red Local	1000/1000	PC	Red Local	65/65	6	4
Red Local	1000/1000	Móvil	Red Local	65/65	-	4
Casa	50/5	PC	URJC	15/15	1	1
Casa	50/5	Móvil	3G	5/1	-	1
Casa	50/5	Móvil	4G	13/4	-	1
URJC	100/100	PC	Casa	50/5	6	4
URJC	100/100	Móvil	3G	5/1	-	2
URJC	100/100	Móvil	4G	13/4	-	4

Conclusiones

- **Objetivo cumplido:** versiones web de las herramientas para ver sensores y teleoperar robots.
 - Conexiones en tiempo real
 - Incluidos en el repositorio oficial
 - Multitplataforma y multidispositivo
- Manejan tecnologías web como WebGL, ICE-JS....

Líneas futuras:

- Versiones en NodeJS
- Crear extensiones de Firefox, Chrome,...

Enlaces

- Mediawiki: <http://jderobot.org/Aitormf-tfg>
- Repositorio: <https://svn.jderobot.org/users/aitormf/tfg/>