PLIEGO DE PRESCRIPCIONES TÉCNICAS (PPT)

Implementación API de GeoBat mediante web components

y pruebas completas E2E de GeoBisorea

EXP. 26021

CATEGORÍA 2 GIS

PLIEGO ESPECÍFICO DEL SISTEMA DINÁMICO DE ADQUISICIÓN PARA LA CONTRATACIÓN DE SERVICIOS DE CONSULTORÍA, ANÁLISIS, DESARROLLO Y MANTENIMIENTO DE SOFTWARE.

ÍNDICE

1 Implementación API de GeoBat mediante web components y pruebas completas E2E de GeoBisorea 3

1.1 Objeto 3

1.1 Descripción 3

1.2 Alcance (Trabajos a contratar) 7

1.3 Productos resultantes del contrato 8

1.4 Niveles de Calidad 8

1.5 Entorno Tecnológico 10

1.6 Entorno funcional 10

1.7 Condiciones de Ejecución 10

1 Implementación API de GeoBat mediante web components y pruebas completas E2E de GeoBisorea

1.1 Objeto

El contrato tiene 2 objetivos:

- Desarrollo de un Web Component desarrollado con la librería Lit v3 con funcionalidades básicas de GeoBat. Este módulo incluirá 7 pruebas E2E de validación del Web Component. El Web Component API de GeoBat permitirá incrustar el componente de mapa de GeoBat en cualquier tipo de aplicación web, ya sea HTML, Angular, JavaScript, etc.
- Realización de las pruebas E2E a todo el conjunto de la plataforma GeoBat, con 30 pruebas adicionales a las ya existentes en GeoBat.

1.1 Descripción

El proyecto pretende desarrollar un visor ligero basado en Web Components Lit 3 y empaquetado ESM, con 7 pruebas E2E, y por otro lado realizar 30 pruebas E2E del proyecto GeoBisorea.

Web Component API.

La solución se basa en la implementación de un Web Component estándar, denominado <geobat-map>, que centraliza toda la funcionalidad del visor. Este componente se distribuirá como un módulo ESM, permitiendo su inserción directa en cualquier entorno (desde Angular hasta HTML nativo). A nivel de renderizado, se utilzará Light DOM en lugar de Shadow DOM, asegurando que la librería base geobat-map pueda acceder al contenedor del mapa a través de su ID en el árbol global del documento, tal como requiere su funcionamiento interno.

Parametrización básica:

El Web Component expondrá una interfaz de configuración basada en atributos y propiedades esenciales, tales como map-id, server-api-url y lang. A partir de estos inputs, el componente encapsulará la lógica de comunicación, encargándose de normalizar la URL base (inyectando sufijos como /api automáticamente) y de construir dinámicamente los endpoints para recuperar la configuración del mapa y sus entidades. Este diseño abstrae la complejidad del backend, permitiendo que la aplicación anfitriona se limite a proveer los datos de conexión sin necesidad de conocer la estructura interna de los servicios

Representación de ubicaciones personalizadas

El componente implementará un mecanismo de superposición de marcadores en el cliente (client-side rendering), permitiendo inyectar ubicaciones ad-hoc independientes de los servicios del backend. Mediante la entrada de coordenadas, activos vectoriales y tooltips opcionales, el sistema gestionará una capa gráfica interna y efímera para renderizar estas features sin impactar en la configuración del servidor.

Adicionalmente, el componente soportará un modo de localización por referencia, capaz de instanciar y resaltar elementos existentes en el modelo de datos mediante la dupla de identificadores: entityId (entidad) y elementId (elemento).

Framework de widgets desacoplados

Sobre el núcleo del visor se construirá un framework de extensibilidad modular. Este sistema permitirá integrar funcionalidades extra mediante widgets desacoplados, tratados como paquetes ESM independientes que se cargan bajo demanda. La arquitectura se fundamenta en tres pilares:

- **Carga Dinámica:** Los widgets solo se descargan y ejecutan si la aplicación los solicita explícitamente.
- **Interfaz Declarativa:** La integración se realiza mediante configuración simple (atributos o slots en el DOM).
- **Inyección de Dependencias:** El núcleo suministra a cada widget un 'Contexto' seguro (referencia al mapa, contenedor visual y métodos de ciclo de vida).

Este diseño de 'Inversión de Control' asegura que los módulos puedan manipular el mapa sin conocer su código interno, garantizando un núcleo ligero y mantenible.

Widgets

La implementación de los widgets seguirá un patrón modular, operando como componentes independientes que se integran en el framework base. Su ejecución se realizará estrictamente bajo demanda, activándose solo cuando la aplicación consumidora lo solicite expresamente mediante configuración (atributos) o composición (elementos anidados). Como punto de partida, el sistema ofrecerá el siguiente conjunto de herramientas:

Selector de Mapas Base:

Módulo encargado de la gestión y conmutación de la cartografía base. Su lógica unifica en un único control de selección tanto los mapas preconfigurados en el sistema como aquellos incorporados dinámicamente mediante identificación externa, garantizando una transición fluida entre las distintas fuentes de datos

Suite de Navegación e Interacción

Módulo que encapsula los controles estándar para la manipulación del viewport (zoom, desplazamiento, extensión inicial y navegación por teclado). Este widget se integrará en el layout mediante anclajes configurables, garantizando una experiencia de usuario consistente sin interferir con el área de trabajo principal.

Gestión de Atribuciones

Componente encargado de renderizar los créditos y derechos de propiedad intelectual de las fuentes de datos, asegurando el cumplimiento normativo. Su diseño permite una disposición flexible en pantalla para no obstaculizar la visualización del mapa.

Servicio de Geolocalización

Widget que interactúa con la API de posicionamiento del navegador para obtener y centrar la vista en las coordenadas actuales del dispositivo. Adicionalmente, puede proyectar un marcador temporal sobre el mapa, integrándose con el sistema de renderizado de ubicaciones personalizadas para contextualizar la posición del usuario frente a otros datos.

Gestor de Visibilidad de Capas (TOC)

Interfaz de control que lista las capas operativas configuradas en el mapa. Permite al usuario gestionar su estado (encendido/apagado) y manipular propiedades visuales como la opacidad. Actúa como una leyenda interactiva para la administración inmediata del contenido visible.

Módulo de Búsqueda Unificada

Barra de búsqueda centralizada que conecta con los servicios de localización del backend (direcciones, códigos, etc.). El widget gestiona la consulta, presenta los resultados en lista y, tras la selección, ejecuta el resaltado y centrado del elemento en el mapa, abstrayendo la complejidad de las consultas espaciales.

Añadir Mapas Base por identificación

Widget diseñado para enriquecer el catálogo de mapas de fondo en tiempo de ejecución. A partir de un identificador proporcionado (vía atributo o configuración), el componente resuelve la definición del servicio contra la API y lo inyecta dinámicamente en el selector de fondos, ampliando las opciones disponibles sin alterar la configuración raíz.

Añadir Capas por identificación

Homólogo al anterior, pero enfocado en capas de negocio. Permite instanciar capas adicionales mediante sus identificadores lógicos. El widget recupera su metadato del servidor y las incorpora al mapa, ofreciendo control granular sobre su estado inicial (visible o disponible en leyenda) para facilitar la composición de mapas personalizados.

Interacción del Web Component

Sistema de Eventos y Notificaciones (Output):

El Web Component implementará un modelo de comunicación reactivo basado en CustomEvents nativos del DOM. Esto permitirá propagar cambios de estado e interacciones del usuario hacia la aplicación contenedora. La interfaz de eventos propuesta incluye:

geobat-map-ready:

Notificación de ciclo de vida. Confirma que el núcleo del mapa ha finalizado su carga y está disponible para procesar comandos. El objeto detail expondrá la referencia a la instancia del mapa.

geobat-feature-click:

Evento de interacción. Se dispara al seleccionar una entidad vectorial, devolviendo en su detalle los metadatos clave: ID de entidad, capa de origen y atributos asociados.

geobat-layer-toggled:

Cambio de estado. Informa sobre modificaciones en la visibilidad de las capas (activación/desactivación), proporcionando el layerId afectado y su nuevo estado booleano.

geobat-basemap-changed:

Cambio de contexto. Señaliza la sustitución del mapa base activo, indicando el ID del nuevo recurso seleccionado.

geobat-search-result-selected:

Integración de búsqueda. Se emite tras la elección de un resultado en el widget buscador, entregando el payload de datos necesario para que la aplicación host pueda reaccionar (ej. desplegar fichas de detalle).

Métodos invocables desde la aplicación anfitriona (Input)

La aplicación contenedora tendrá control total sobre el visor mediante una serie de funciones públicas que puede ejecutar en cualquier momento:

setExtent(extent):

Modifica la vista para encuadrar el área geográfica definida por las coordenadas límite.

selectLayer(layerId, selected):

Controla programáticamente el estado de visibilidad de una capa operativa específica.

setBasemap(basemapId):

Fuerza la conmutación del mapa de fondo al identificador suministrado.

setMarkers(markers):

Inyecta o refresca la colección de marcadores client-side (coordenadas, assets gráficos y metadatos de entidad).

triggerAction(actionId, payload):

Implementa un patrón Command genérico. Permite invocar funcionalidades encapsuladas en el núcleo o en los widgets (como navigation.zoomTo o search.open) mediante cadenas de texto, facilitando la extensibilidad de la API sin alterar la firma del componente.

Pruebas E2E de GeoBisorea y Web Componet API

En 2025 se han realizado una serie de pruebas E2E del aplicativo GeoBat para evaluar la utilidad de esta tecnología. Una vez determinada su aplicación práctica se pretende extender estas pruebas al conjunto de funcionalidades de la herramienta.

Se requiere la preparación de un conjunto de pruebas automatizadas para los distintos módulos de GeoBat, utilizando Playwright. Se implementarán pruebas end to end de 30 casos de uso principales del aplicativo GeoBat con las herramientas Playwright. Estos 30 casos de uso serán propuestos a la dirección técnica de CCASA como paso previo a la configuración de los mismos.

Adicionalmente se requiere preparar 7 pruebas E2E con Playwright adicionales para el testeo de del web Component API.

1.2 Alcance (Trabajos a contratar)

El alcance consiste en el desarrollo de los siguientes componentes según la descripción de detalle del apartado 1.1.:

- Web Component API con Framework de widgets desacoplados

- Widgets
    - Selector de Mapas Base:
    - Suite de Navegación e Interacción
    - Gestión de Atribuciones
    - Servicio de Geolocalización
    - Gestor de Visibilidad de Capas (TOC)
    - Módulo de Búsqueda Unificada
    - Añadir Mapas Base por identificación
    - Añadir capa por identificación

- Interacción del Web Component
    - Sistema de Eventos y Notificaciones (Output):
    - Métodos invocables desde la aplicación anfitriona (Input)

- 30 Pruebas E2E de GeoBisorea y 7 de Web Componet API

1.3 Productos resultantes del contrato

Con independencia de lo propuesto a nivel metodológico, a continuación, se añaden tareas y entregables mínimos, si proceden, a realizar/suministrar por la empresa adjudicataria:

- Las implementaciones, en forma de módulos de software, resultantes de la realización de cada una de los módulos descritos en el proyecto.
- Documentación técnica, acorde a la metodología utilizada por CCASA, resultante de las actividades realizadas.
- Pruebas E2E con playwright para la validación automatizada del aplicativo resultante. Se contemplan 30 pruebas E2E con Playwright.

Documentación soporte: Cada documento se suministrará en formato electrónico original (Microsoft Office), no admitiéndose por lo tanto formatos tales como PDF u otros protegidos y/o no manipulables.

Todos los entregables deberán proporcionarse utilizando las herramientas homologadas por CCASA para el ciclo de vida de las aplicaciones, o en su caso proporcionando formatos compatibles.

Además de lo descrito y requerido, la empresa licitadora puede proponer cualquier otro conjunto de entregables referidos a la metodología que proponga aplicar para el desarrollo del proyecto.

1.4 Niveles de Calidad

El software construido deberá contar con los siguientes niveles de calidad:

- Óptimo rendimiento, especialmente en la respuesta a la interacción del usuario.
- Óptima usabilidad
- Software libre de errores

- Arquitectura modular, desacoplada, orientada a servicios de negocio que permita distintas composiciones del mismo.
- Diseño gráfico ajustado a las normas de estilo UX/UI de CCASA
- Integración basada en servicios web o bus de integración de CCASA
- Integración con las herramientas de gestión de código de CCASA.
- Satisfacer de forma eficaz y eficiente las necesidades de los diversos tipos de personas usuarias.
- Condiciones de accesibilidad conforme a WAI AA. (en aplicaciones externas – internet)
- En los casos en los que se considere de utilidad, se confeccionará un prototipo que servirá para contrastar la sintonía del producto con los deseos y necesidades del usuario como parte imprescindible de la calidad.

Con relación a las entregas de código a realizar, estos desarrollos deberán estar libres de errores y listos para su puesta en producción, de manera que sea operable desde el mismo momento de la entrega.

La entrega solamente se considerará realizada si obtiene el visto bueno de CCASA. La fecha efectiva de la entrega será aquella en la que se produzca dicha aprobación.

Estándares de CCASA

A continuación, figuran los estándares de CASA en cuanto a buenas prácticas y sus herramientas de soporte. Solo se utilizarán productos vigentes en la plataforma tecnológica de CCASA.

- Metodologías: MEDEA (personalización de Métrica 3) y metodologías ágiles implantadas en CCASA (Scrum). Con su repositorio de documentación Confluence, y otros repositorios de corporativos de CCASA.
- Reutilización: CCASA dispone de una colección de elementos reutilizables que deben ser incluidos en todo desarrollo que se lleve a cabo.
- Identificación electrónica y servicios de confianza: estándares admitidos por el Reglamento (UE) 910/2014 del Parlamento Europeo (identificación, sellado, firma, etc…)
- Estandarización de aplicaciones: Framework jADE / Plantillas Java .NET, Aplicaciones Angular, API REST, Microservicios (contenedores)
- Integración continua: Jenkins
- Repositorio documental: Filenet
- Control de versiones: Subversion y– Git/Bitbucket
- Modelado de datos – IBM Data Studio

- Orientación a objetos: Persistencia de clases y bases de datos – Hibernate / NHibernate
- Pruebas unitarias – JUnit / NUnit
- Análisis de código: Porcentaje código testeado y cód. muerto – Rational Application Developer / DotCover, FxCop SonarQube (calidad de código).
- Optimización de sentencias SQL – IBM Data Studio / SQL Server Management Studio
- Validación de normas de accesibilidad – Total Validator
- Validación de servicios web – SoapUI
- Pruebas de carga o estrés – JMeter
- Reutilización de pruebas en aplicaciones Cliente / Servidor – Visual Studio Premium
- Reutilización de pruebas en aplicaciones Web – Selenium
- Medición de rendimientos y establecimiento de alertas – Dynatrace
- Monitorización de la plataforma – SCOM
- Repositorio de artefactos: Nexus

Además, CCASA dispone de las siguientes utilidades de ayuda:

- Gestión del repositorio de Reusabilidad
- Gestión del Catálogo de Aplicaciones
- Gestión de métodos y de servicios SOA

Herramienta para el despliegue de las aplicaciones

1.5 Entorno Tecnológico

- **Desarrollo Front-end**: Typescript y Lit 3.
- **Back-end**: Java 8, JDK21, Maven.
- JBOSS

1.6 Entorno funcional

El entorno funcional es el de GeoBisorea de GeoAraba.

1.7 Condiciones de Ejecución

1) Participación de CCASA

Un equipo de trabajo de CCASA se encargará de supervisar, controlar y aprobar la facturación de los servicios prestados por la entidad adjudicataria. Este equipo tendrá potestad para intervenir en las órdenes de trabajo, su prioridad, planificación y modo de resolución.

2) Arquitectura

Los Servicios integrarán los componentes corporativos ya existentes en CCASA que fueran necesarios. CCASA entregará una guía de estandarización de interfaces de las aplicaciones, un manual de estilo de construcción de aplicaciones, una guía de arquitectura y de desarrollo Framework para aplicaciones Java, así como toda la información técnica disponible para facilitar y normalizar los trabajos que compongan el servicio.

3) Equipamiento

La empresa adjudicataria aportará todo el equipamiento informático, las licencias de software, las líneas de comunicación (por ejemplo VPN) y en definitiva, toda la infraestructura (externa a CCASA) necesaria para la prestación del servicio.

4) Lugar de trabajo

Todos los trabajos se llevarán a cabo en las oficinas de la empresa adjudicataria, salvo que a requerimiento explícito del CCASA, el equipo deba realizar sus tareas en las oficinas del CCASA.

En el caso de que durante el seguimiento se detecten anomalías o desviaciones importantes, CCASA podría exigir el traslado de la actividad a sus propias oficinas para ejercer un mayor control sobre los trabajos, de manera que pueda verificar la aplicación de las medidas correctoras acordadas. La entidad adjudicataria correría con todos los gastos derivados de este cambio de ubicación y posteriores de dietas o desplazamientos.

5) Horario y calendario

El horario de trabajo se adaptará en todo momento a los requerimientos del servicio así como a los horarios de trabajo de los empleados de la DFA y CCASA. El calendario del servicio se adaptará al calendario laboral vigente en el Territorio Histórico de Álava.

Las actuaciones que requieran dejar fuera de servicio alguna funcionalidad del sistema se llevarán a cabo fuera del horario/calendario de las personas usuarias de los sistemas a fin de que no suponga una merma en el tiempo de disponibilidad, en cuyo caso se pactarán con el CCASA las condiciones de prestación de estas.

6) Intervención presencial

Pudiera darse el caso de que la resolución de una incidencia crítica, que debe solucionarse en un tiempo record. En este caso, y tras el aviso pertinente por parte de CCASA, el personal técnico deberá estar disponible de forma On-line o presencial en un tiempo máximo de 2 horas.

7) Supervisión del servicio

Para que CCASA pueda supervisar el grado de eficiencia del servicio es necesario que el personal de la empresa adjudicataria realice una imputación semanal de horas en el sistema que exprofeso pone CCASA a su disposición.

8) Transferencia de conocimiento

El adjudicatario realizará sesiones periódicas para la transferencia de conocimiento técnico a personal de CCASA, en los apartados y desarrollos realizados por el adjudicatario y que se consideren de vital importancia por CCASA. Adicionalmente se le podrán solicitar documentación, tutoriales o videotutoriales de los apartados técnicos utilizados en el proyecto que CCASA considere necesarios.