Fichas Municipales de Cantabria

Lorena Campo Moreno

Miguel Expósito Martín

Instituto Cántabro de Estadística (ICANE)

Introducción

Las Fichas Municipales constituyen una recopilación de información a nivel municipal, procedente de diversas fuentes (INE, INSS, Consejerías del Gobierno de Cantabria, etc.), que trata de ofrecer una visión general sobre las características socio-demográficas, económicas y de infraestructuras de cada uno de los 102 municipios de Cantabria. Esta publicación se elabora anualmente en base a los datos disponibles, referidos al año anterior, en las bases de datos del ICANE.

La publicación original, "Fichas Municipales de Cantabria", se encontraba suspendida a la espera de la reestructuración completa a acometer en este proyecto. Para su generación era necesario actualizar complejos modelos datos y modificar plantillas de informes utilizando la herramienta privativa de generación de informes *Crystal Reports* [CRY00].

La última edición servía de referencia en cuanto a selección básica de contenidos y disposición de los distintos elementos tabulares y gráficos. Dicha edición aún puede consultarse en la sección correspondiente de la web del ICANE (http://www.icane.es/products/icane/municipal-files).

La presente ponencia resume los aspectos metodológicos, funcionales y técnicos más relevantes a la hora de llevar a cabo una reestructuración y dinamización total de la publicación, utilizando el último estado del arte en cuanto a tecnologías de visualización y difusión de datos estadísticos.

Antecedentes

Dada la heterogeneidad y cantidad de los datos a presentar en la publicación, antes de elaborar la primera versión se planteó la necesidad de automatizar la misma, de forma que se redujese al mínimo el tiempo y el personal necesario para generar las fichas. El problema planteado fue doble: por un lado, era preciso disponer de datos actualizados en un formato que permitiese su consulta y manipulación mediante las aplicaciones disponibles en el ICANE y, por otro lado, presentar los datos en su forma definitiva y válida para impresión o consulta Web.

Para resolver el segundo problema se decidió utilizar *Crystal Reports* como herramienta de maquetación y generación de informes, sustituyendo con ventaja a la elaboración de las páginas con *Microsoft Excel* que, aún cuando el resultado final fuera el mismo, requería más tiempo y trabajo.

El primer problema se resolvió en parte definiendo un esquema de base de datos *Oracle* [ORA00] en el que se almacenaban todos los datos necesarios para la generación del informe. Este esquema estaba formado por tablas creadas a partir de los ficheros *Excel* originales en los

que se encontraban la mayoría de los datos. Así, se disponía de los datos en un formato que facilita su acceso, pero con algunos inconvenientes:

- El diseño de las tablas no era el más adecuado para otro tipo de consultas, puesto que se crearon tal cual se encontraban los datos en formato *Excel*, dando lugar a un alto acoplamiento.

- Había muchos datos redundantes, que se encontraban también en otras tablas de la base de datos.

- La actualización de los datos implicaba reemplazar los ya existentes, cargándolos desde hojas *Excel* o mediante consultas a otras tablas.

Además de generar informes en PDF, también se elaboró un CD con un sistema de consulta basado en páginas HTML y *Javascript*.

Pese a tener como objetivo la automatización, la confección de la publicación requería de una gran cantidad de trabajo manual, siendo necesario revisar todas las páginas de los 102 documentos pertenecientes a cada uno de los municipios, en busca de determinados problemas comunes para su solución o ajuste.

Esta solución, si bien satisfizo razonablemente las expectativas de técnicos estadísticos y consumidores de información durante sus primeros años de vida, rápidamente quedó obsoleta y se convirtió en un tedioso proceso que terminó por suspenderse a la espera de un nuevo producto exento de estas limitaciones.

Necesidades y metas a alcanzar

Como necesidades fundamentales a la hora de ejecutar el proyecto, se detectaron las siguientes:

- Eliminar la carga de trabajo que suponía la generación de las fichas municipales, debido a su excesiva dificultad de mantenimiento.

- Mejorar el acceso y representación de datos municipales por parte de los usuarios de datos estadísticos.

- Ofrecer la posibilidad de realizar agrupaciones municipales y definir “comarcas” u otras clasificaciones donde los datos se representen de forma agregada siempre que sea posible.

En cuanto a las metas a alcanzar, se propusieron las siguientes:

- Ofrecer un producto de difusión de datos estadísticos municipales configurable “a la carta” y en la línea del estado del arte de la difusión y representación de datos a través de tecnologías web.

- Obtener un producto flexible, extensible y configurable mediante la fácil inclusión de nuevos cuadros o módulos de representación y visualización de datos.

- Aumentar el número de visitas o descargas del producto con respecto a las últimas estadísticas conocidas, haciéndolo llegar al mayor número posible de usuarios potenciales.

Metodología y proceso software

Para llevar a cabo el desarrollo de la nueva solución se optó por utilizar metodologías y prácticas ágiles de desarrollo de software: Scrum[SUTH00], Kanban, XP[BECK00] y Lean IT[POPP00].

Para ello, se planificaron sprints o iteraciones de dos semanas de duración desde el inicio del proyecto, con reuniones entre los miembros del equipo de trabajo cada 15 días. La planificación de dichas reuniones podía ser modificada a conveniencia del equipo siempre y cuando existiera consenso mayoritario.

Los requisitos podrían y serían revisados, modificados, ampliados y detallados en cualquier reunión de cualquier sprint por parte del equipo de trabajo. Asimismo, en cada iteración se ajustarían las estimaciones realizadas conforme se fuera disponiendo de la información suficiente como para reducir al máximo cualquier tipo de incertidumbre.

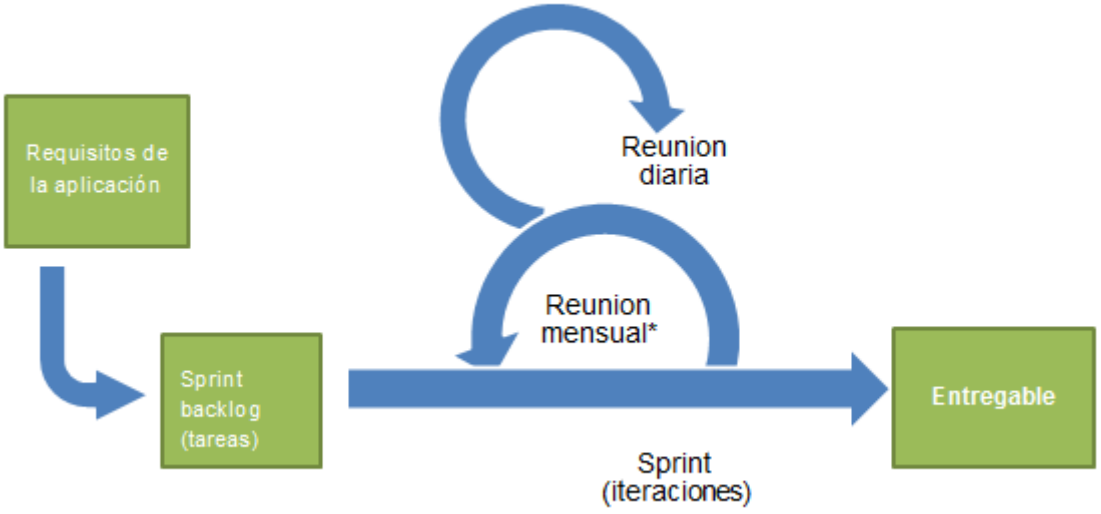

Ilustración 1. Visión general de Scrum

Se fijaron dos etapas de desarrollo:

Iteración 0: Verificación de la lista de objetivos/requisitos y planificación

- Duración: dos semanas

- Objetivos: planificar y distribuir los objetivos y alcance del proyecto en iteraciones, de manera que los requisitos estuvieran priorizados balanceando el beneficio que aportan al ICANE, su coste de desarrollo y los riesgos del proyecto.

- Las actividades que componen esta etapa fueron las siguientes:

1. Priorización de los requisitos en iteraciones y entregas y definición del *product backlog* o pila de producto considerando los siguientes criterios:

    - El valor aportado por cada requisito para el ICANE.

    - El esfuerzo necesario para desarrollar cada uno de los requisitos, de manera que en las primeras iteraciones se desarrollasen los requisitos que proporcionaran el máximo valor con el mínimo esfuerzo y que pudieran encajar en la periodicidad de las iteraciones.

    - Las dependencias inevitables entre requisitos.

    - Minimizar los riesgos del proyecto respecto a desarrollo de los requisitos, disponibilidad y grado de implicación de los actores y beneficiarios participantes, interacción con otros equipos, etc.

    - Maximizar la cohesión del contenido de cada iteración, identificando los puntos de acoplamiento y las dependencias entre los diferentes incrementos de manera que sean mínimos, para poder dar por realmente completados los requisitos desarrollados en cada una de las iteraciones.

2. Calcular la duración de cada uno de los incrementos desarrollados de manera que pudieran encajar en la periodicidad de las iteraciones.

Iteraciones de desarrollo

- Duración: dos semanas cada iteración.

- Objetivos: Completar un incremento de producto demostrable al finalizar la iteración.

- Actividades:

1. En el inicio de cada iteración:

- Se mantiene una reunión para consensuar los objetivos y contenido de la iteración en función de los criterios de priorización indicados anteriormente, así como para dar detalle a los requisitos seleccionados (*sprint backlog* o pila de sprint) en forma de desglose de tareas. De manera general, a cada requisito se le asocia un conjunto de condiciones o pruebas de aceptación para poder considerar que este ha sido completado.

- Al utilizarse un enfoque iterativo e incremental, en la implementación de cada característica se llevan a cabo las distintas fases de un desarrollo de software tradicional (análisis, diseño, construcción y pruebas).

2. Al finalizar cada iteración:

- Realizar una demostración de los requisitos o historias de usuario completados. En esta demostración participa el equipo de trabajo al completo, confirmándose su aceptación después de realizar las comprobaciones oportunas.

- Las pruebas unitarias tendrían que satisfacer las pruebas de aceptación definidas en la reunión del comienzo del sprint. Tras analizar las características desarrolladas, el equipo de trabajo decide si es necesaria una refactorización[FOWL00] de las mismas, debiendo llevarse a cabo en el siguiente *sprint* en caso de ser necesario.

- El ICANE podría volver a priorizar el conjunto de requisitos del proyecto y el equipo de trabajo consensuaría el contenido de las siguientes iteraciones. En particular, los elementos a volver a priorizar podrían ser: requisitos inicialmente identificados del proyecto, modificaciones a estos requisitos, nuevos requisitos surgidos durante el proyecto, problemas de calidad detectados, etc.

- Entregables: el producto desarrollado hasta ese momento incluyendo todos los entregables asociados completados (documentación, etc.) e integrados a su vez con los entregables de las iteraciones anteriores, de manera que dicho producto sea susceptible de ser entregado con el mínimo esfuerzo en ese mismo momento.

Proceso software

En las primeras reuniones del equipo de trabajo se decidió utilizar diversas técnicas y herramientas ágiles para apoyar la metodología escogida. Entre ellas, se citan las más importantes:

1. Cobertura razonable de pruebas unitarias tanto en *frontend* como en *backend*.

2. Inclusión de métricas de calidad de código: complejidad ciclomática, mantenibilidad, listas estándar de comprobación, etc.

3. Uso de trabajos de integración continua para realizar los despliegues y las distintas pruebas.

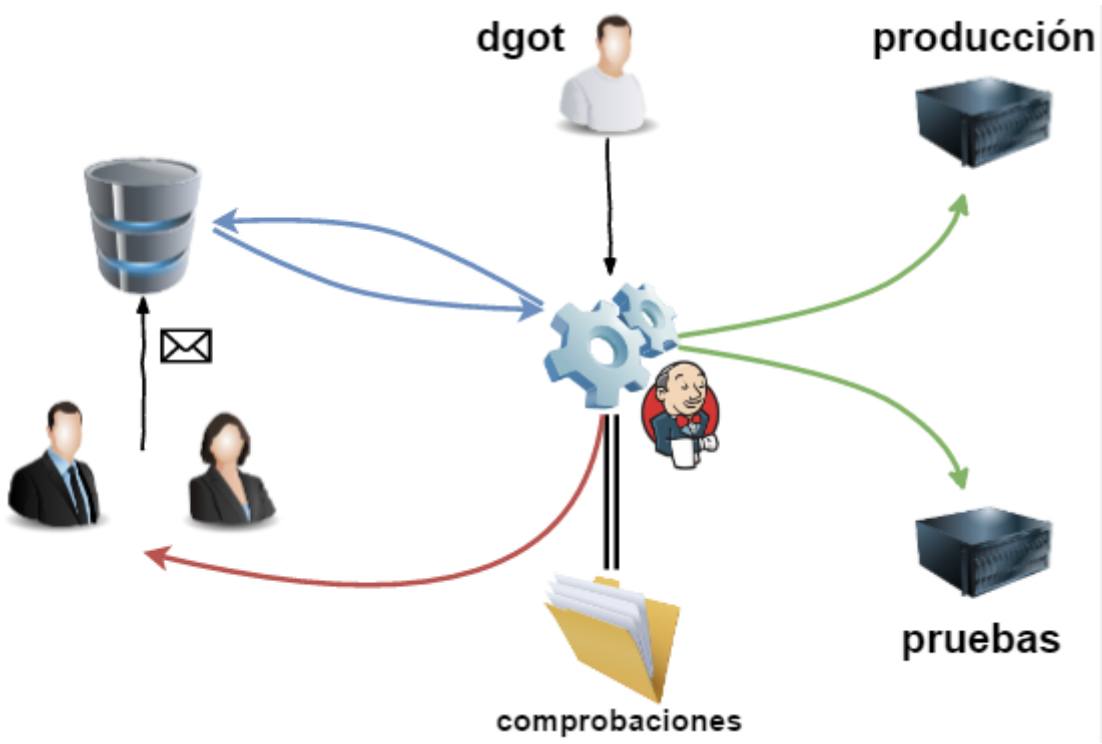

Ilustración 2. Proceso software: integración continua

El flujo de trabajo detallado en la figura es el siguiente:

- El equipo de desarrollo envía sus cambios a un VCS (*version control system* o sistema de versionado de código) basado en *Git*[GIT00].

- Los técnicos del ICANE/DGOT ejecutan los trabajos de integración manualmente (o bien esperan a su ejecución planificada a diario).

- Los trabajos de integración ejecutan diversas tareas de verificación de calidad de código (las indicadas anteriormente).

- Si el trabajo de integración se ejecuta correctamente en todas sus fases, se despliega el componente en el entorno en cuestión (pruebas o producción).

- Si el trabajo de integración falla en alguna de sus fases, se notifica al equipo de desarrollo, debiendo tomar las acciones correctivas oportunas para llevar a cabo el despliegue con éxito.

Esta metodología de trabajo y su proceso *software* asociado han permitido:

- Reducir el número de errores o *bugs* durante el desarrollo.

- Facilitar la verificación funcional por parte de los técnicos del ICANE, al contar con un sistema funcional desde fases tempranas del desarrollo.

- Minimizar los riesgos a la hora de pasar a realizar el despliegue de pruebas a producción.

Arquitectura de la solución

De las necesidades y metas a alcanzar expuestas anteriormente se desprende que en el producto final, gran parte del esfuerzo de la solución quedaría centrada en la interfaz de usuario, especialmente en la visualización de la información. Por ello, se optó por dividir el producto en dos componentes fundamentales:

- *Backend*: desarrollado en tecnología JEE (*Spring*[SPR00] + *Hibernate*[HIB00]), que se encargaría de proporcionar toda la lógica y almacenamiento asociados al funcionamiento del producto en sí (es decir, datos y metadatos asociados a las propias fichas municipales).

- *Frontend*: desarrollado en tecnologías HTML5 + *Javascript* (concretamente, *Angular.js*[ANG00] y *D3.js*[D3JS00]), que tendría como responsabilidades la obtención, representación tabular y visualización gráfica de los datos estadísticos municipales.

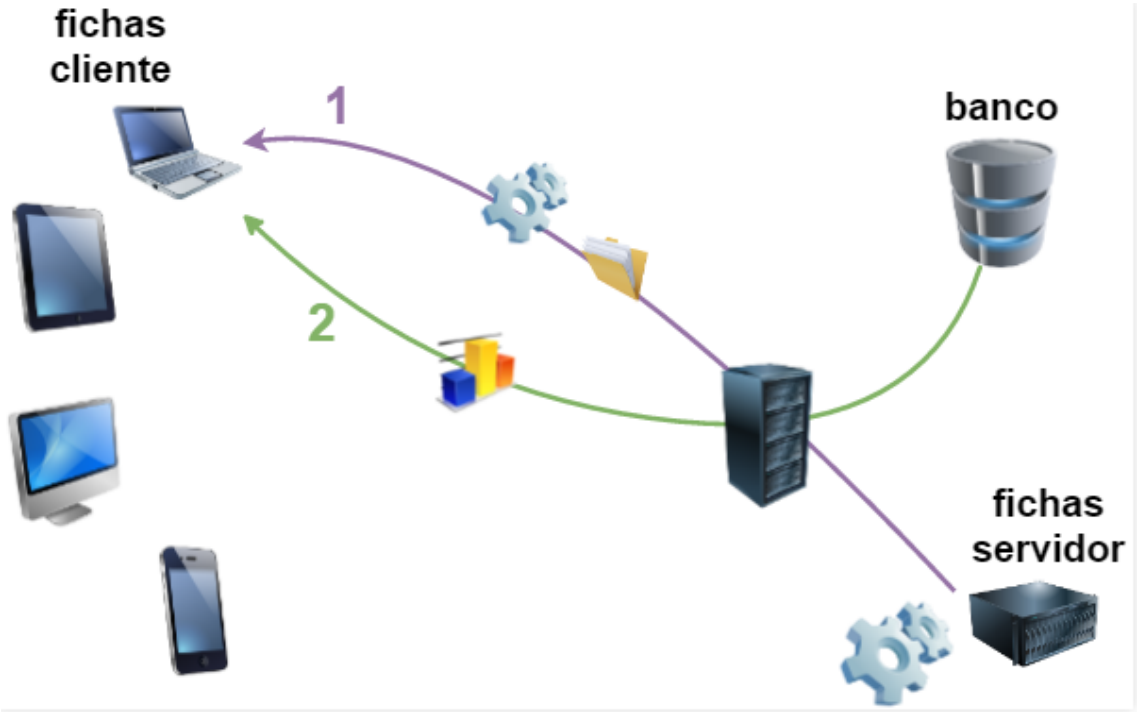

Ilustración 3. Arquitectura de la solución

El flujo de datos en la arquitectura elegida es el siguiente:

1. El cliente (PC de escritorio, tableta o *smartphone*) descarga la aplicación *Angular.js* completa (HTML5 + CSS + JS).

2. La aplicación, una vez descargada, realiza las peticiones de datos necesarias al banco de datos y, una vez recibidos estos, los incluye en las estructuras de datos necesarias para su representación tabular y gráfica.

Si el usuario estuviese usando su perfil privado, además interactuaría el componente de *backend* donde se almacenan sus preferencias.

Esta arquitectura de solución no habría sido viable de no disponer en la actualidad el ICANE de un banco de datos y metadatos estadísticos con APIs *restful* de extracción de datos. Dicha infraestructura centraliza la práctica totalidad de las series estadísticas difundidas en la web por el ICANE, ofreciendo un único punto de actualización que puede actuar como fuente para una diversidad de productos (como el tratado en cuestión).

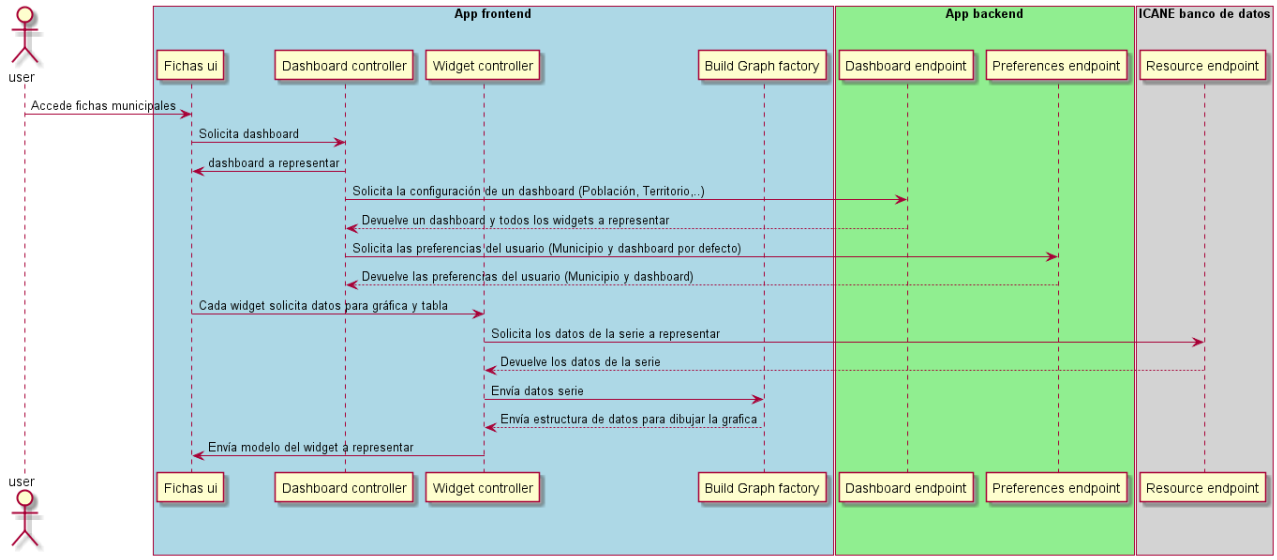

Ilustración 4. Diagrama de secuencia de interacciones principales

En el diagrama anterior se presenta un detalle de los componentes de la aplicación:

- Interfaz de usuario: recoge las interacciones con el usuario.

- Controlador de tablero o *dashboard*: gestiona la configuración de los tableros y sus preferencias.

- Controlador de *widget*: recupera series temporales desde el banco de datos estadísticos del ICANE y devuelve datos para su representación en la interfaz de usuario.

- Generador de gráficos: recibe datos desde el controlador de *widgets* y devuelve estructuras de datos para dibujar el gráfico concreto en la interfaz de usuario.

Justificación de la elección tecnológica

Si bien en el ámbito del *backend* las soluciones basadas en JEE y sus *frameworks* ligeros derivados (como *Spring*) se han mostrado robustas y solventes y están caracterizadas por una gran ubicuidad en el mercado, las opciones en el ámbito de *frontend* son múltiples y con mucha más dispersión, no estando tan claro cuál es la mejor alternativa.

Angular.js

- Presenta una más que aceptable implementación del modelo MVC, ofreciendo las conexiones necesarias entre los componentes de las distintas capas.

- Permite definir una interfaz de usuario de forma declarativa, usando HTML5.

- Ofrece la posibilidad de modificar el comportamiento de los distintos componentes de forma sencilla a través de directivas.

- Permite ordenar y reducir la cantidad de código *Javascript* producida.

- Facilita el uso de pruebas unitarias en el desarrollo, ayudando a garantizar la calidad final del producto.

D3.js

- Permite visualizar datos gráficamente de forma ubicua en la web, sin más requisitos que un navegador moderno.

- Es flexible y se adapta perfectamente con el resto de tecnologías (HTML, CSS, JS, etc.).

- Se presenta como una biblioteca perfecta para la personalización de gráficas.

HTML5, CSS3 y Twitter Bootstrap

Hoy en día no se entiende una nueva aplicación web sin utilizar las nuevas versiones de los estándares. En el ámbito de los estilos y apariencia de las aplicaciones web, el *framework Twitter Bootstrap*[BOOT00] se ha convertido en un estándar de facto. Su uso en este producto ha facilitado el enfoque de un diseño adaptable a distintos tamaños de terminales, dotándolo de características multidispositivo.

Arquitectura de sistemas

Sin duda el mayor reto a la hora de diseñar el sistema en su conjunto fue conseguir ofrecer un rendimiento aceptable sin riesgo de sobrecargas en el banco de datos estadísticos del ICANE. Hay que tener en cuenta que, para la representación de un *widget* con tabla y gráfico, la aplicación cliente realiza tres peticiones al banco: datos, gráfico y metadatos. Esto, multiplicado por unos 50 *widgets* que perfectamente pueden existir en un panel y teniendo en cuenta una consulta de unos 10 usuarios simultáneos (un número razonablemente bajo), supondría una "avalancha" de unas 1500 peticiones al banco de datos en un período muy corto de tiempo. Teniendo en cuenta que el banco de datos es un sistema pensado para ofrecer un rendimiento razonable para grandes consultas pero no excepcional para consultas pequeñas, parece sensato afirmar que se darían serios problemas de carga en el mismo.

Para mitigar estos problemas, se optó por integrar las Fichas Municipales con un acelerador web o servidor de caché (*Varnish*[VAR00]). De esta forma, las peticiones *restful* en formato JSON al banco de datos se recuperarían desde el *backend* del banco de datos tan sólo la primera vez, quedando en caché RAM durante una semana. Esta estrategia proporciona un rendimiento y protección más que aceptables dadas las previsiones de consulta del producto. Los datos quedarían almacenados en caché durante una semana, pudiendo ser refrescados en

cualquier momento (teniendo en cuenta que las series que alimentan a las fichas tienen periodicidades de actualización mensuales o anuales, es más que suficiente). En la siguiente figura se resume los sistemas implicados en su conjunto.

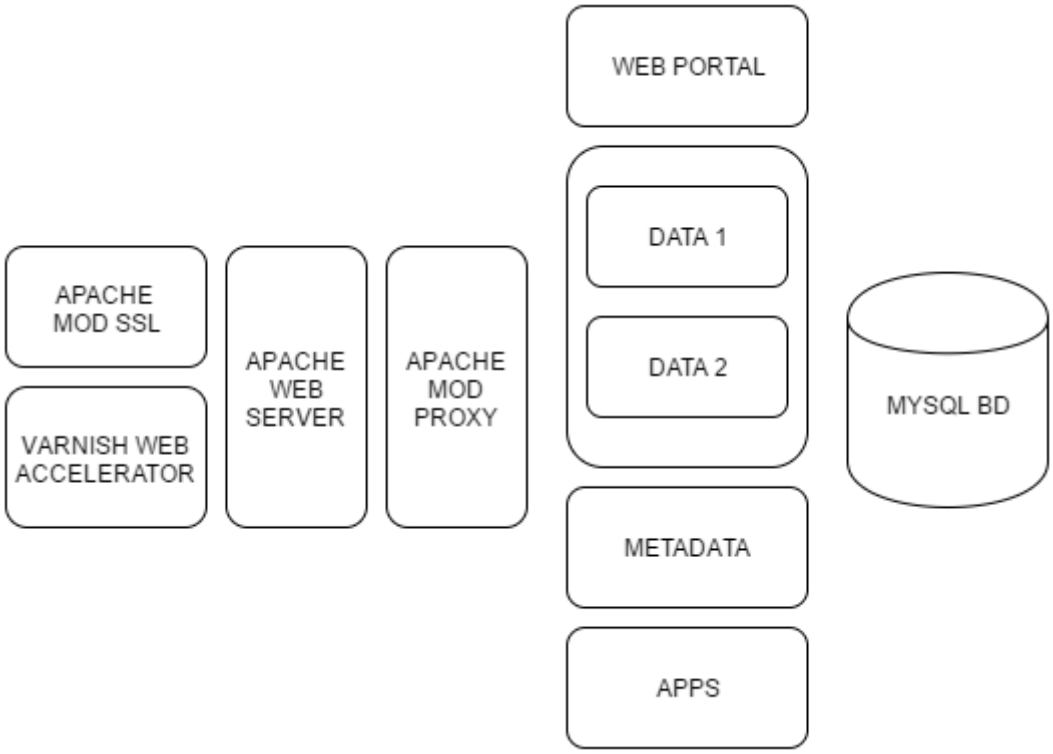

Ilustración 5. Diagrama de sistemas implicados

Diseño de la solución

Para conseguir la flexibilidad y sencillez de mantenimiento deseadas, se opta por un diseño sustentado sobre los siguientes niveles:

- Tableros o *dashboards*: contenedores de *widgets* en base a plantillas. Definen la característica fundamental del producto (ej: fichas o agrupaciones municipales).

- Plantillas o *templates*: configurables y personalizables por los usuarios y/o administradores, determinan el orden y la selección de *widgets* disponibles dentro de un panel dado (ej: economía, población, sociedad, etc.).

- *Widgets*: se trata de la unidad mínima de presentación de información al usuario. Pueden contener datos tabulares, gráficos de diversos tipos o ambos. Los *widgets* se alimentan directamente desde el banco de datos estadísticos del ICANE (tablas, gráficos y metadatos) y son configurados por los administradores en la forma más adecuada para su consumo (ej: selección de datos a representar en tabla, tipo de gráficos, etc.).

- Series: estas entidades establecen una correspondencia directa con el banco de datos estadísticos del ICANE y permiten a los administradores configurar selecciones de datos a bajo nivel que posteriormente serán consumidas en los *widgets*.

Seguridad de la información

Dada la naturaleza orientada al cliente de la aplicación, fue necesario elegir una estrategia de seguridad distinta de las convencionales. Para ello, se optó por un proceso de autenticación y autorización basado en *token*. Básicamente, el usuario se identifica en el sistema, cuyo *backend* devuelve un *token* de acceso que incluye, entre otra información, el rol o perfil de este. Posteriormente, ese mismo *token* se utiliza para validar el acceso del usuario a un determinado recurso de información.

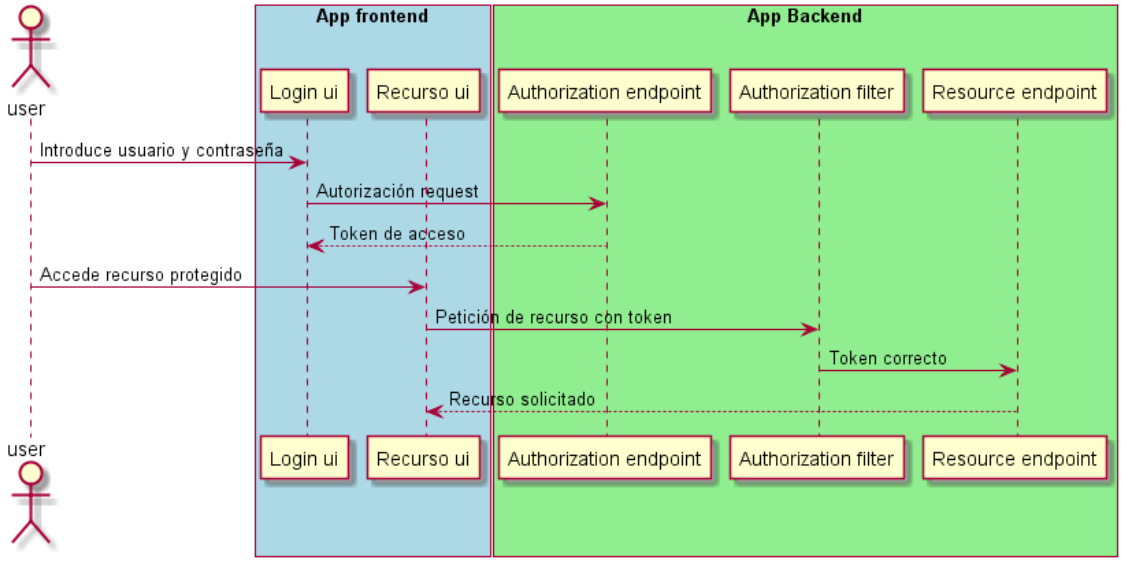

Ilustración 6. Diagrama de secuencia del proceso de autenticación

Naturalmente, para reducir el riesgo de ataques *"man-in-the-middle"* tan sólo se ofrece el acceso a la aplicación a través de HTTPS y utilizando cifrados no vulnerables. En la siguiente figura se representa un hipotético caso de uso en el que un usuario malintencionado intenta usar la fuerza bruta para entrar al sistema:

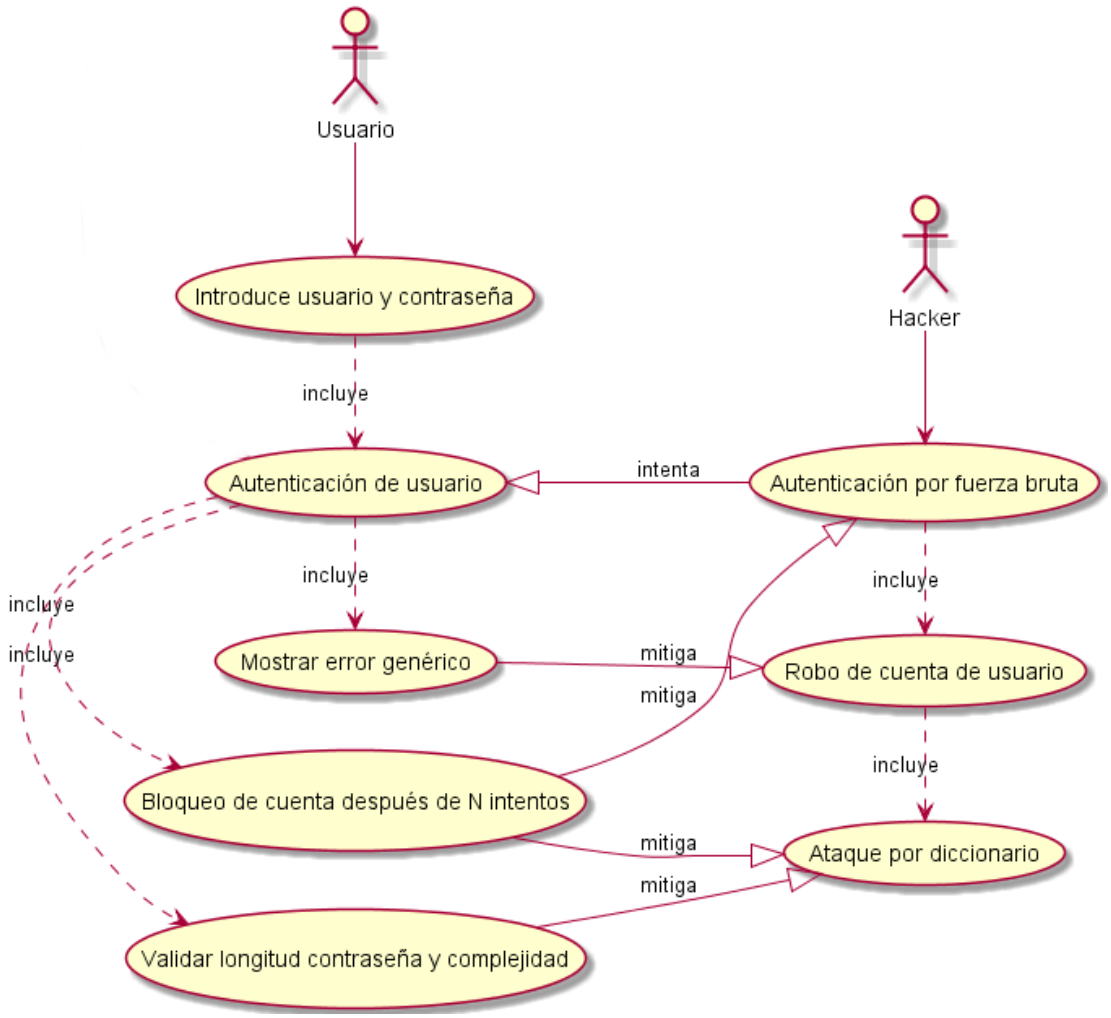

Ilustración 7. Caso de uso de intento de acceso no autorizado

El sistema se bloquea tras un determinado número de intentos fallidos de acceso. Además, para fijar la contraseña, se valida la longitud y complejidad de la misma de manera que dificulte un ataque por fuerza bruta. Tras el registro de un usuario, este recibe un *email* solicitando el establecimiento de dicha contraseña; una vez fijada, se habilita la cuenta del mismo para almacenar sus preferencias.

Demostración funcional

La aplicación de Fichas Municipales de Cantabria permiten la existencia de dos tipos básicos de usuarios, anónimos y usuarios registrados. Para los primeros el uso de la aplicación es más restringido, limitándose a visualizar las Platillas establecidas por defecto, mientras los segundos pueden crearse las suyas propias.

Uso apartado Fichas Municipales

Para todos los usuarios, por defecto la aplicación comienza en el área de *Fichas Municipales*, donde se puede elegir uno de los 102 municipios de Cantabria. Igualmente por defecto, se han generado 4 plantillas temáticas, Población, Economía, Sociedad y Territorio, en las que se dividen el total de Widgets creados.

Exporta toda la ficha como tablas de Excel

Ilustración 8. Selección de Municipio y Platilla

Una vez seleccionado el municipio de trabajo y la plantilla deseada, se visualizarán todos los widgets de la misma, pudiendose exportar todas las tablas a formato Excel.

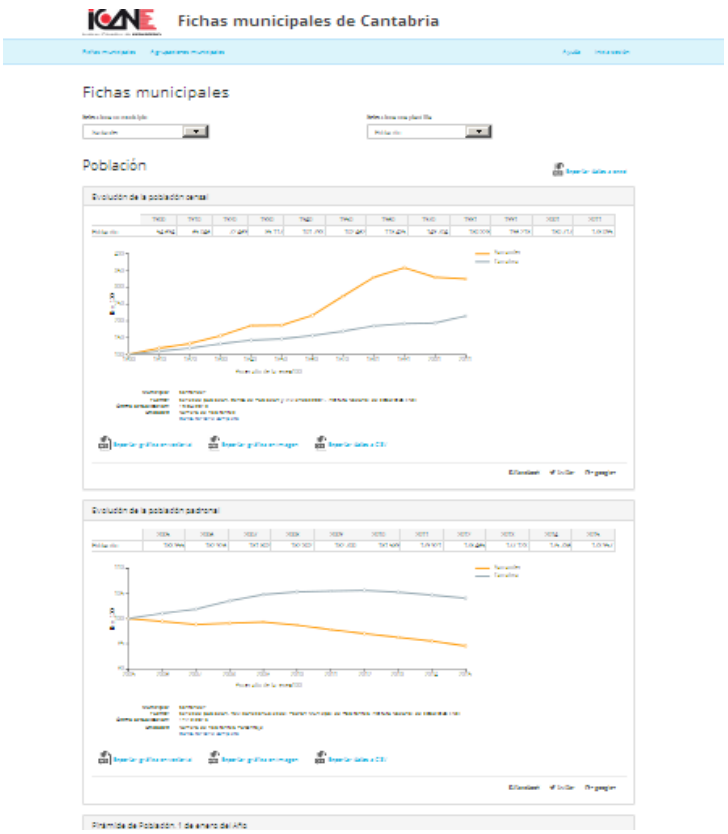

Ilustración 9. Plantilla completa

Un Widget está compuesto como mínimo de una tabla con una serie temporal, y puede contar además con un gráfico. En cada uno de los que componen la plantilla se pueden exportar la tabla a formato CSV, para su uso con otras herramientas informáticas (hojas de cálculo, programas estadísticos, etc.), y el gráfico a imagen PNG o SVG. El widget además incorpora un enlace directo al cubo OLAP de la página Web del ICANE del que procede, así como enlaces para su publicación en redes sociales.

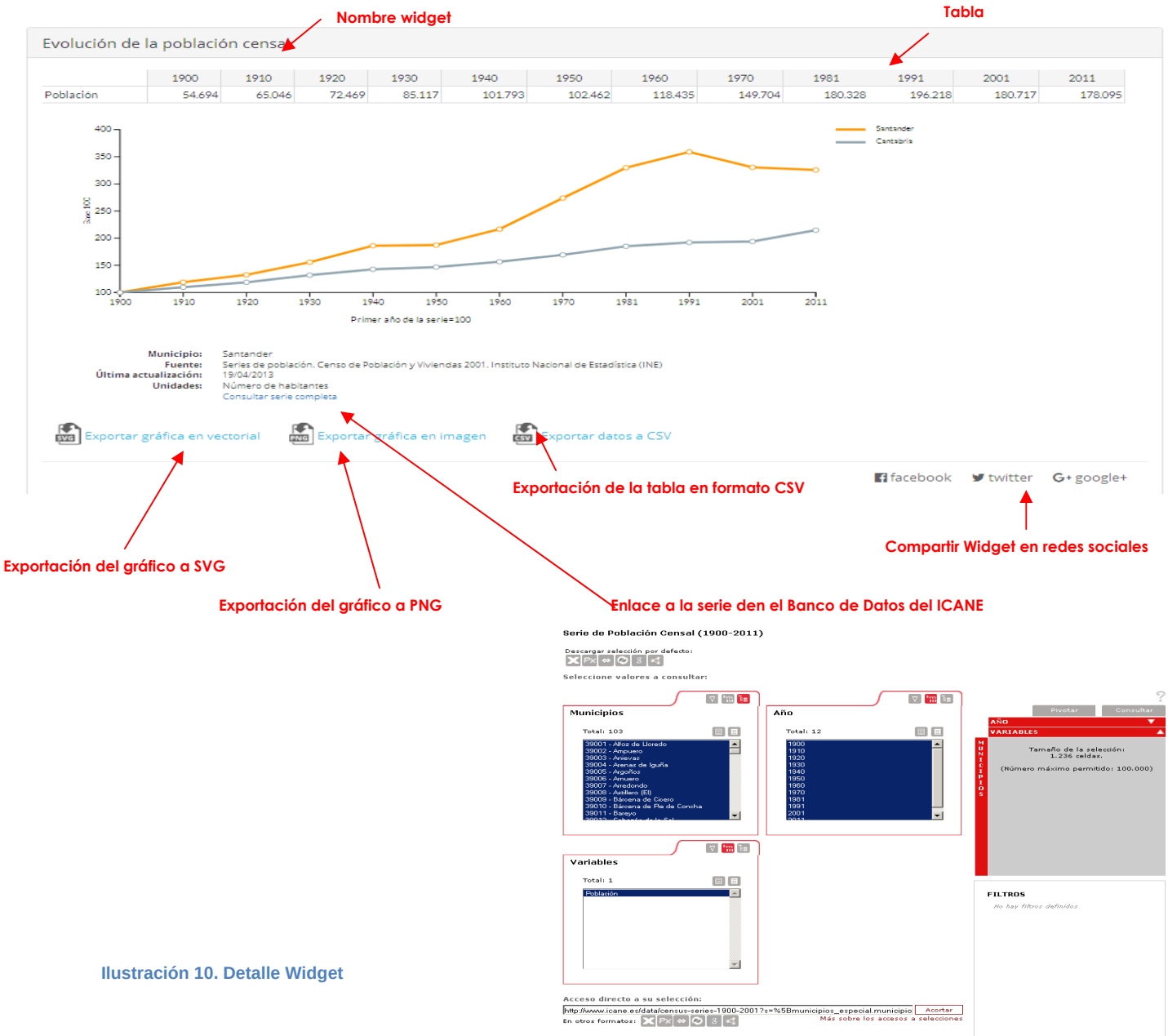

Ilustración 10. Detalle Widget

Uso apartado Agrupaciones Municipales

En el apartado de *Agrupaciones Municipales* se obtiene una ficha igual a la municipal, pero con datos agregados de los municipios seleccionados, siendo el máximo de selección 8. En este caso se ha de seleccionar todos los municipios de los que se quiere obtener datos, así como la plantilla correspondiente. Por defecto existen las mismas 4 plantillas que encontramos en el apartado municipal.

Para elegir los municipios basta seleccionarlos en la lista desplegable, pudiendo eliminarlos simplemente al volver a hacer clic en el nombre. Una vez seleccionados los municipios y la plantilla, para que se carguen los datos es necesario elegir “Pulsa para actualiza la consulta”. Si se desea eliminar toda la selección hacer clic en “Borrar Selección”. El resultado es el mismo que el de *Fichas Municipales*.

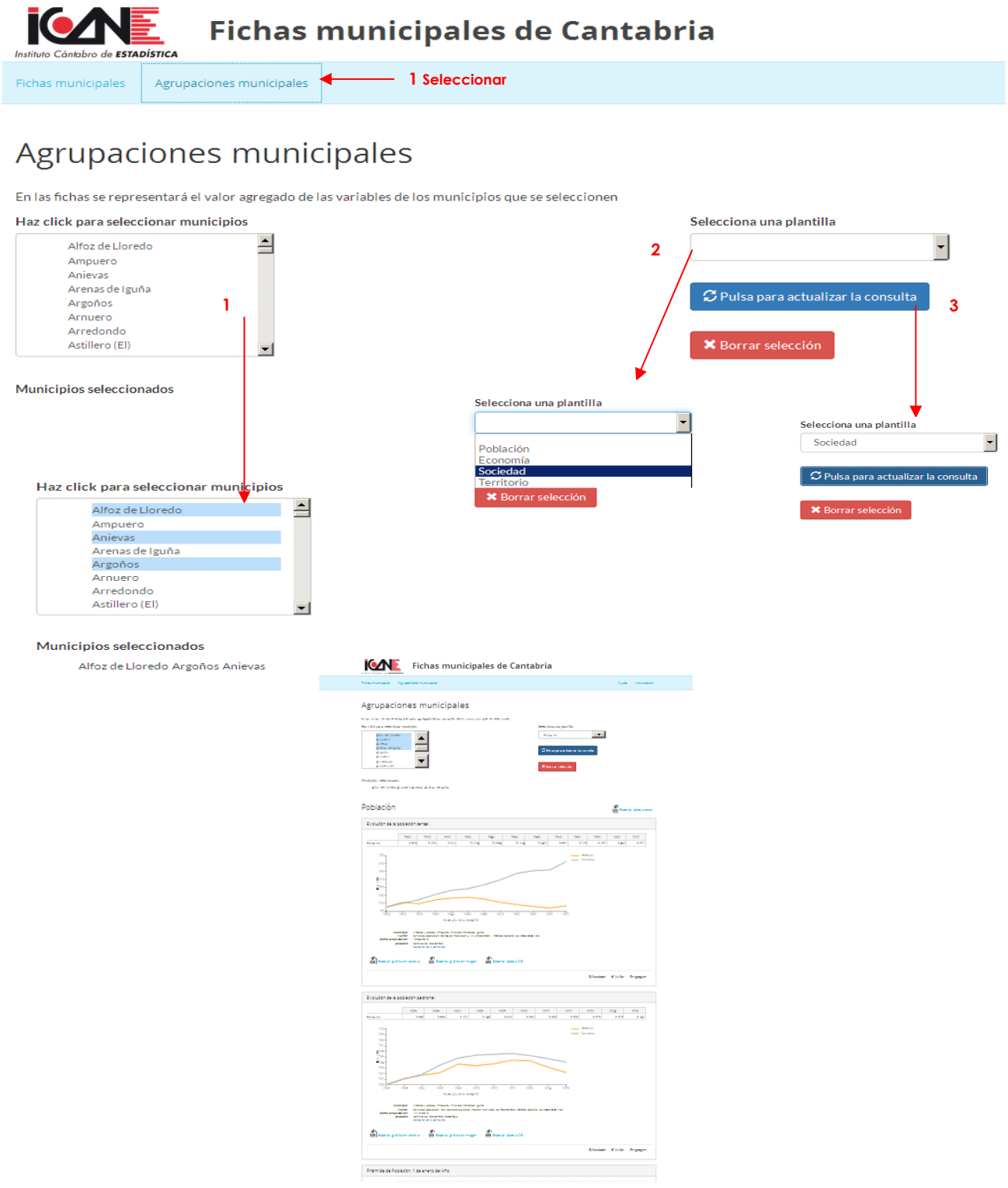

Ilustración 11. Agrupaciones Municipales

Uso apartados como Usuarios Registrados

Para los usuarios registrados el uso de la aplicación es el expuesto anteriormente, pero pueden crear sus propias plantillas, tanto en *Fichas Municipales* como en *Agrupacione*s, en el apartado de *Diseño de Plantillas*. Para generarlas primero se ha de seleccionar en que apartado se quiere incluir, si municipal o agrupaciones, siendo el proceso igual para ambos.

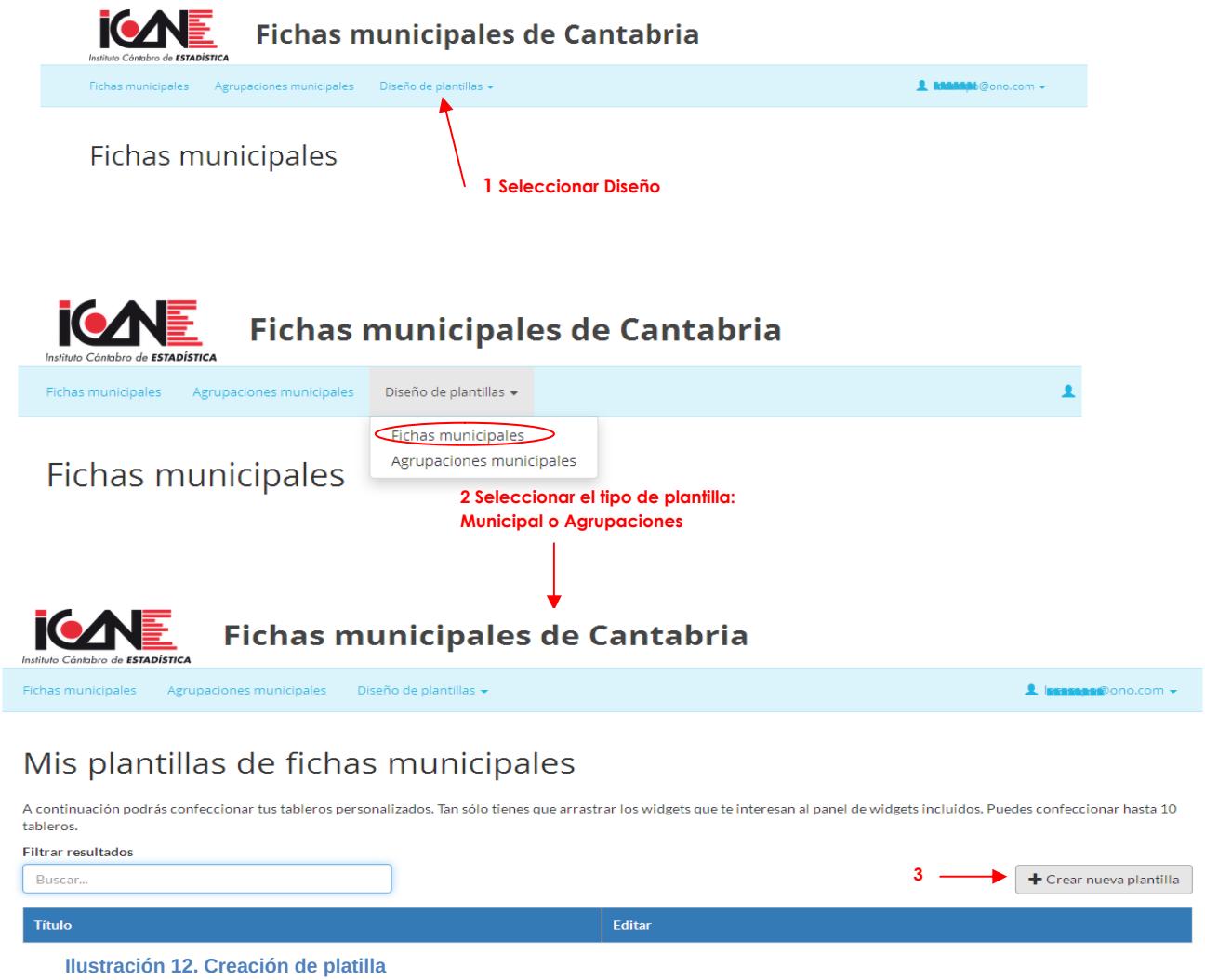

Ilustración 12. Creación de platilla

Una vez creada la platilla el usuario podrá incorporar todos los widgets que le interesen del conjunto de los disponibles, independientemente de la temática.

Ilustración 13. Selección e incorporación de widget a platilla

En el momento que se guarda la platilla, cada vez que el usuario entre en su cuenta, esta aparecerá en el desplegable de *Selecciona una platilla* del apartado para el que se haya generado.

Ilustración 14. Fichas municipales con plantilla de usuario

Métricas e indicadores

Entre las comprobaciones automáticas de calidad de código se incluyen diversas relacionadas con la complejidad, si bien es cierto que todas ellas vienen a representar el mismo concepto aunque con distinta escala. En concreto, se presta especial atención a la complejidad ciclomática de McCabe[MCCAB00], que viene a definir el número de caminos independientes dentro de un fragmento de código y determina la cota superior del número de pruebas que se deben realizar para asegurar que se ejecuta cada sentencia al menos una vez. Se trata de un indicador independiente de la tecnología y lenguaje subyacentes y que se encuentra comúnmente aceptado en la comunidad de desarrolladores de *software*. Un valor de 10 a nivel de método se considera como adecuado, y es el límite que se ha tratado de mantener en el desarrollo de este producto. Este indicador se ha tenido en cuenta tanto en el desarrollo del *backend*, con un *plugin* del gestor de ciclo de vida del proyecto (*Maven*) como en el del *frontend*, con un paquete para *node.js*.

Otras comprobaciones de calidad se han llevado a cabo a través de las listas de comprobación estándar de los analizadores estáticos de código *Findbugs*[FBUG00] y PMD[PMD00], automatizadas con cada despliegue en pruebas y producción.

Las revisiones de accesibilidad no han podido llevarse a cabo con total propiedad dado que las herramientas disponibles en el mercado no están preparadas para aplicaciones *Angular.js*. En los momentos de la redacción de esta ponencia, se está preparando una consulta a la Comunidad de Accesibilidad del Portal de Administración Electrónica de la Administración General del Estado. No obstante, en la medida de lo posible, el código HTML generado se ha validado en el W3C y a través del validador de accesibilidad ofrecido por el PAE.

Por otra parte, la experiencia de usuario se ha medido a través de *Google Page Insights*[GPI00], obteniendo un total de 88 puntos sobre 100. A través de este mismo servicio se puede medir un rendimiento de 56 en dispositivos móviles y 77 en PCs. Son datos mejorables, si bien en la práctica la mayor parte de los retrasos vienen dados por la cantidad de información suministrada al terminal del usuario.

Finalmente, se ha incluido dentro de la propia aplicación un sistema de registro de eventos en archivo con formato de log de servidor Apache, lo que permitirá analizar en detalle las visitas recibidas a las distintas partes de la aplicación aún teniendo en cuenta su naturaleza orientada al cliente. Dicho análisis se llevará a cabo desde el sistema de análisis de tráfico web que el ICANE ha desarrollado para centralizar los procesados de archivos de registro de su ecosistema web, y que estará en producción en los próximos meses desde la redacción de esta ponencia.

Conclusiones y líneas futuras de trabajo

El resultado final del desarrollo expuesto en la presente ponencia ha sido un producto que ha satisfecho la mayor parte de las necesidades de los usuarios y técnicos estadísticos, gozando de una gran aceptación entre los distintos grupos de interés en Cantabria. La aplicación de metodologías ágiles, si bien no ha sido todo lo ortodoxa que debería, ha permitido garantizar la calidad técnica del producto y el ajuste funcional a las especificaciones iniciales proporcionadas por el equipo de trabajo del ICANE. Los riesgos detectados en cuanto a rendimiento de la aplicación se han solventado con pequeñas correcciones y ajustes en el banco de datos estadísticos del ICANE y el uso de un potente servicio de caché. Su diseño adaptable permite una visualización multidispositivo con una única implementación, y su flexibilidad y simplicidad de configuración y actualización lo convierten en un producto fácilmente mantenible que prácticamente no requiere de tiempo adicional por parte de los técnicos de estadística o de informática. Esta facilidad de configuración permite también a usuarios no expertos la configuración de sus propios tableros para representar datos de interés a la carta.

Tal ha sido el éxito del producto, que el ICANE ya ha planteado un proyecto de “generalización” del mismo que permita publicar otros productos en el mismo formato (como puede ser el Anuario Estadístico de Cantabria). Lo que se pretende es desarrollar un “generador” de publicaciones estadísticas en formato “Fichas Municipales”, ya que resulta más fácil de consumir por parte de los distintos grupos de interés y no supone prácticamente un incremento de recursos para su mantenimiento.

Naturalmente, el producto actual presenta aspectos mejorables: aumentar la puntuación de rendimiento en *Google Page Insights*, permitir un enlazado directo a cada uno de los *dashboards* por parte de servicios externos, incluir *widgets* con representación cartográfica, incluir otros tipos de gráficos, etc. La filosofía de los desarrollos incrementales e iterativos encaja con el concepto de producto cuyo desarrollo y mejora nunca terminan, yendo más allá de las limitaciones que supone el concepto de proyecto. El ICANE pretende mejorar también la aplicación de estas metodologías, refinándolas y ajustándolas para optimizar los resultados en cada producto.

Referencias

[CRY00] Crystal Reports, http://www.crystalreports.com/emea/

[ORA00] Oracle, https://www.oracle.com/es/database/index.html

[SUTH00] *The Art of Doing Twice the Work in Half the Time*, Jeff Sutherland.

[BECK00] *Extreme Programming Explained*, Kent Beck.

[POPP00] *Lean Software Development: an agile toolkit*, Mary & Tom Poppendieck.

[FOWL00] *Refactoring*, Martin Fowler.

[GIT00] Git, https://git-scm.com/

[SPR00] Spring framework, https://projects.spring.io/spring-framework/

[HIB00] Hibernate, http://hibernate.org/

[ANG00] Angular.js, https://angularjs.org/

[D3JS00] D3.js, https://d3js.org/

[BOOT00] Twitter Bootstrap, http://getbootstrap.com/

[VAR00] Varnish Cache, https://varnish-cache.org/

[MCCAB00] Complejidad, https://es.wikipedia.org/wiki/Complejidad_ciclom%C3%A1tica

[FBUG00] Findbugs, http://findbugs.sourceforge.net/

[PMD00] PMD, http://pmd.github.io/

[GPI00] Google Page Insights, https://developers.google.com/speed/pagespeed/insights/