EPIC Platform and Technology Solution

INTRODUCTION AND PROJECT OVERVIEW

The European Platform for Intelligent Cities (EPIC) project (grant reference 270895), was one of seven projects funded by the European Commission under the 2010 Competitiveness and Innovation Programme (CIP) objective "Open innovation for future Internet-enabled services in smart cities". As a CIP ICT Pilot B project, the aim of the project was not to undertake fundamental research, but instead to further develop, extend and integrate existing technologies and solutions to create a proof-of-concept solution. This would be tested rigorously and made ready for market, supported by a clearly defined roadmap for implementation and commercial exploitation.

The overall aim of the EPIC project was to develop a flexible, extensible, future-proof cloud computing platform maximising the use of open-standards. The platform would host, manage and deliver a diverse range of smart-city applications to citizens and businesses, deliver smart-city data services to support innovation among SME developers and improve efficiency in city administration. A schematic of the platform and its constituent Pilots and components is shown in Figure 1:

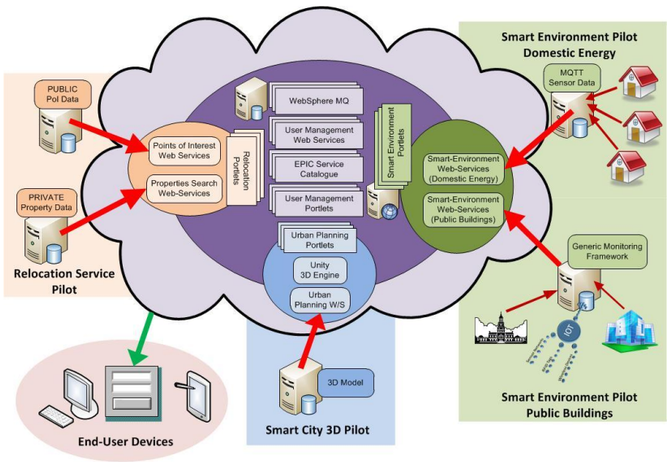


Figure 1: EPIC Platform and Pilot Schematic

The EPIC project defined a number of core principles to ensure that the lessons learned could be replicated and applied widely post-project:

Cloud Platform: The use of a cloud platform allowed the EPIC project to demonstrate and test the attractiveness and viability for city administrations of combining 'software as a service' (SaaS) model with virtualisation of the platform hardware through 'platform as a service (PaaS), as a future path for evaluation and delivery of increasingly smart city services.

Open-Standards: A core principle used throughout the entire project was to maximise the use of open-standards for the communication of all data between the platform and the applications to ensure that the lessons learned from EPIC could be readily replicated and applied by others post-project.

Device Agnostic Solutions: The architecture employed to deliver user interfaces ensured that the end-user applications were delivered to be as device agnostic as possible, i.e. with a minimised reliance on the hardware specifics of the end-users' devices, which could be personal computers; tablet computers; smart-phones, etc.

"User-in-the-Loop" Development and Test: Three very different pilot applications were implemented on the platform and tested with citizens engaged through the European Network of Living Labs (ENoLL) to ensure that the platform could be tested across a diverse range of current and future application scenarios, The three pilots were initially developed separately in collaboration with the cities of Brussels, Issy-les-Moulineaux and Manchester, to address the local needs of citizens and businesses. An iterative "user in the loop" development methodology was adopted to ensure that the applications were truly addressing local needs and that the look and feel and functionality were appropriate.

Design for Re-Use: Following development and testing in their "home" cities, the three Pilots were combined into an integrated smart-city application and recreated in the city of Tirgu-Mures, Romania for testing with their citizens to test that the platform and service model of EPIC could be readily re-used and that pilots developed using web-services and portlets could be readily re-deployed in a new city.

Role of the Pilots within EPIC

The three very different pilots developed within the EPIC project had several purposes within the validation of the concept of the EPIC platform architecture for the delivery of smart city services:

- The pilots demonstrated that highly functional and engaging smart-city application could be developed either inside or outside of the EPIC platform environment and could be subsequently integrated successfully into the platform
- That portlets running in a standard browser window could deliver highly functional and engaging user-interfaces to a variety of devices in a browser and hardware
- That presentation of smart-city data as web-services allowed the pilots to consume shared smart-city data from the platform, accessed by web-services exposed by the data owners
- The user-centric pilots allowed comprehensive user-testing to be undertaken within diverse stakeholder groups to validate the user acceptability of the functionality and look-and-feel that could be delivered using the web-services and portlet architecture used within EPIC
- The repurposing of three separately developed pilots to create a single smart-city solutions for Tirgu-Mures demonstrated both the ease of reuse and repurposing of existing web-services and portlets and demonstrated the potential for third-party developers to consume platform data web-services to create new and engaging smart-city applications

The Relocation pilot allows users planning to relocate to Brussels to search for properties that satisfy their personal requirements, in terms both of the accommodation and price and the proximity to public transport, schools and other facilities. This Pilot demonstrates how public data on points-of-interest provided by CIBG (IT-department of the Brussels Region) can be combined with private data on available properties obtained from ImmoWeb and maps obtained from Google maps to create highly-interactive solutions built on web-services and portlets. The large-screen version is also complemented by a version suitable for a mobile device for use during visits to Brussels.

The SmartCity 3D (Urban Planning) pilot demonstrated how a 3D model of a city can be used to explore the city; view rich media about key points of interest and understand the services offered by the SME community. The application also provided tools for SMEs to customise their information and to inter-communicate using Social Media feeds. SmartCity 3D also demonstrated how a commercial 3D games engine (in this case Unity) could be readily integrated into the EPIC platform to provide the 3D visualisation functionality required in a client browser.

The Smart Environment pilot was developed on two parallel threads to maximise alignment with the needs of smart-cities, namely to encourage citizens to understand and hence reduce their personal energy usage (domestic energy consumption) and to demonstrate how the city administration is reducing energy used in public buildings (energy monitoring for public-buildings). The smart environment pilot was the

 © EPIC Project Consortium, 2013

most complex of the three pilots in terms of data acquisition and data processing required. Technical testing of the Smart Environment Pilot covered several parts of their operation: data acquisition; processing and storage; separate web-services for domestic and public buildings and the performance of the domestic energy dashboard and the public buildings dashboard.

The monitoring of **domestic energy consumption** demonstrates how energy data can be extracted from external data repositories and rendered onto a personal energy dashboard via the EPIC platform. This approach potentially allows energy data to be retrieved from many different external sources and aggregated via the EPIC platform. This approach parallels the use of open POI data and commercial propriety data in the Relocation pilot, where the data is externally owned but is consumed by and rendered to the user via the EPIC platform, through a range of value-added web-services that could be offered to SME software developers as "components" for new smart-city applications.

The **energy monitoring for public-buildings** has made use of the data-feeds already in existence from fiscal smart-meters (used by utility companies for billing purposes) and sub-metering implemented within commercial Building Management Systems (BMS). The logic here was not to add yet more sensors but to maximise use of data already being collected within building management systems by aggregating key data into an off-platform generic monitoring framework (GMF). Again the data was accessed using web-services running on the EPIC platform and energy dashboards were delivered using portlets delivered by the EPIC platform.
The three pilots were deployed iteratively in the three pilot cities and tested within closed and open user groups. The user testing as well as evaluation of the Brussels **Relocation pilot** was conducted from April 2012 to February 2013, starting with two cycles of closed user group testing and followed by two cycles of open user group testing. Similarly, the **Urban Planning pilot** was tested and subsequently evaluated from April to November 2012.

The **Smart Environment Pilot** was tested by the closed user group in January and February 2013, followed by the two cycles of open user group testing and the subsequent evaluation in March and April 2013.

All pilots were tested together within **the integrated scenario** in Tirgu Mures. The user testing of 'Find a House' pilot (relocation as model), a 'Discover Tirgu Mures' pilot (Urban Planning as a model) and a 'Smart Environment' pilot (Smart Environment as a model) started in February and finished in April 2013, followed by the evaluation.

OVERVIEW OF THE PLATFORM AND PILOT FUNCTIONALITY

A platform for smart-cities such as that developed within the EPIC project may have to operate in a variety of modes and with varying degrees of support for legacy applications and future applications development so it was important that a wide range of such paradigms were tested within the project. The key functionality that should be provided by the EPIC platform that was embedded within the pilots is explained below:

Access to Data via Web-Services: Increasingly cities are responding to their open-data obligations by making city data available through the city web-site or other routes. Whilst the exposure of raw data may seem an attractive option due to convenience, this is not an ideal route for technical, governance and financial reasons. A better strategy is to expose city open-data through well configured standard web-services, which also contain the meta-data necessary to provide context, which can be readily consumed by third-party developers. The use of web-services for city data access also provides future potential for stratifying access to data services and revenue generation, allowing free-to-access data services to be augmented by to pay-to-use data, where additional data in the pay-to-use web-services provides commercial advantage to applications developers who will sell their applications into a smart-city market place. This approach can be extremely useful where third-party data owners wish to make their data available to developers on a licensed basis. The EPIC model is that the data owner exposes their data through web-services running on their database servers. The EPIC platform hosts web-services that consume the data owner web-services, exposing the third-party data to the EPIC platform user. In this way the ownership of data resides with the data owner; the data owner is free to provide web-services through other channels; the data owner can license the web-service to the EPIC platform and the EPIC platform is responsible for providing access to and licensing of the EPIC platform web-service to the third-party developers.

Implement User-Interfaces as Portlets: Almost all mobile computing devices have a standard web-browser application such as Internet Explorer, Firefox, Chrome, Safari, etc that can display content delivered in the HTML form used for standard web-pages. These browsers typically also provide a Java virtual machine (JVM) that can execute code written in Java either installed on the user's device as an applet or delivered by the remote server as a servlet. Portlets are pluggable user interface software components managed and displayed in a web portal to create a user-interface as a series of non-overlapping portlets. Although created from a series of individual portlets, the user sees only a single interface application interface which is delivered by a portal server. Portlets also give great flexibility to developers, allowing different portlets to be updated as required rather than the refreshing of an entire web-page and there are well-developed standards in place for portlets that ensure re-usability across different browsers and portal servers.

Re-Use Existing Software Applications: Many software developers have applications that they may wish to make available to users through a cloud-based smart-city platform, so it was critical the EPIC project tested this paradigm, rather than assuming all smart-city applications were developed specifically for or on the smart-city platform. Furthermore, a wide variety of web-applications are increasingly available to programmers through a web application programing interface (API). For example, Google provides access to interactive maps through the Google Maps JavaScript API, which allows developers to embed interactive Google maps in their own web-pages or portlets. The JavaScript Maps API V3 is a free service for any web site that is free to consumers but must be licensed for commercial use.

Smart-City Monitoring Capability: In addition to providing access via web-services to historic data, a smart-city platform will be responsible increasingly for the collection and aggregation of real-time data from the city's "vital signs", including traffic flow; air quality; weather; CCTV; web-cams and citizen-generated data from social media, etc. The exponential growth of smart-devices and smart-sensors, including wireless sensors, radio-frequency identification (RFID) tags and IP-enabled devices is driving the "Internet of Things" (IoT) revolution, where management and ad-hoc networks of autonomous devices can generate and share data across the public internet. Whilst standards for IoT are still evolving, certain standards including Message Queue for Transducer Telemetry (MQTT) have been developed; placed into the public domain and have wide support from commercial software products and from open-source tools.

The inclusion of each element into the three individual pilots and subsequently the integrated pilot is shown in Table 1.

 © EPIC Project Consortium, 2013

Pilot	Web-Services	Public Data	3rd Party Data	Portlets	Web APIs	Software Re-Use	Sensor Data
Relocation (Brussels)	X	X	X	X	X		
Urban Planning (Issy)	X	X	X	X		X	
Smart Environment (Manchester)	X		X	X			X
Integrated (Tirgu Mures)	X	X	X	X	X	X	X

Table 1: Pilot Element Integration

EPIC PLATFORM TECHNOLOGY DESCRIPTION

The EPIC platform can be considered as a generic system model or system architecture for a smart-city service delivery platform , which could be implemented using a variety of commercial or open source software that conform to the open source standards used in the project. The key functional elements, given the reliance on web-services and portlets, are for a process-server or application server to deliver the data web-services and for a portal-server to deliver the user-interface as portlets. In addition, the use of MQTT for sensor data transport required an MQTT message broker to handle the publish and subscribe requests for collecting sensor data. In terms of platform service components, common web-services and portlets were needed for user management, to be shared by all pilots and a service catalogue was required to present application functionality and expose web-service functionality to developers. This functionality was all hosted within a cloud computing environment, as shown schematically in Figure 1.

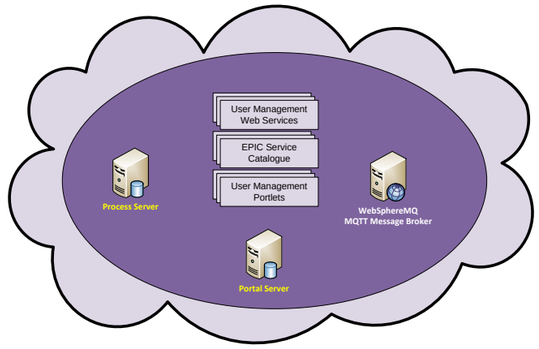


Figure 2: EPIC Platform Architecture Requirements

From the outset, the intention of the EPIC project was to leverage proven commercial-strength assets for the platform, to ensure that maximum development time could be focused onto the Pilot applications by the development teams, rather than potentially time consuming task of creating a platform from open-source components. However, in accordance with the aim of using open-standards and ensuring vendor neutrality, the commercial assets used for the platform embraced fully open-standards, to ensure that the required platform functionality could be delivered post-project using open-source or commercial assets. Because IBM was the key technical partner included specifically in the consortium for their platform expertise, the EPIC platform was constructed using IBM assets, as explained further below.

Cloud Platform: Even within a cloud computing application, the EPIC platform software had to be physically hosted somewhere on physical hardware, even if the machines on which it executed were virtualized. One attraction of a cloud computing approach is that the virtualized machines needed for execution can expand or contract across physical hardware in response to user demand, rather than having servers specified for the highest user demand remaining idle for a large proportion of time. The project was a collaboration of partners spread over many countries in Europe and there was no main, central partner in the project with the physical hosting capacity for the project. The EPIC platform was therefore implemented on the IBM SmartCloud Enterprise, which enabled the project to obtain its computing needs as a service through both “platform as a service” to the consortium and “software as a service” delivered to the developers and end-users.

Application Server: To avoid confusion it is important to differentiate between Application Servers and Web-Servers. Web Servers serve HTTP protocol level content to deliver web-pages to end-users. Application servers couple the operating system and applications to extend the function of the operating system by integrating core business systems such as databases and transaction processors and extending them to the Web. Put simply, in a system such as EPIC, the application server deals with data and executes the business logic and the web-servers presents the results from the application server to the user as web-pages. Java 2 Enterprise Edition (J2EE) is the standard defining Application Servers which enables developers to create reusable, platform-independent modules. J2EE allows independent development of applications that can be deployed on any J2EE Compatibility Test Suite (or CTS) compliant application server and allows mission-critical, J2EE applications to be developed and executed across an enterprise and beyond. The IBM WebSphere Application Server was used as the software core of the EPIC platform to serve a J2EE application software framework.

Portal Server: A Portal Server is a set of frameworks and tools to deliver a unified “web experience” by marshaling individual portlets to create one unified presentation. The portal server delivers a highly personalized user-experience, taking into account user-role, personal settings such as language and user-device settings such as screen size, etc. From the developer perspective, the portal server manages themes and skins to provide a unified look across all portlets, the equivalent of cascading style sheets (css) for web-pages. In a portlet sever, pages consist of layouts and page areas. Layouts include branding elements pages and navigation and each page has its own layout and any row/column layout is possible. Each portlet is a separate application and can be developed independently with different modes, configuration and instance data. Portlets can support multiple devices such as smartphones and tablets in addition to standard PCs and can offer unique views for each device, while sharing common business logic.

There are well-developed open-standards for Java portlets (JSR168 and JSR268) that ensure portlets developed to these standards in one environment are portable to different portal servers. The Web Services for Remote Portlets (WSRP) standard defines how portlets communicate with portal servers, irrespective of implementation method. Portlets created with development tools compliant with these standards can run on any Portal Server that is certified to the standards.

The EPIC platform was configured by IBM with the following components:

- 1 x Portal Server
- 2 x Business Process Manager Servers (with 1 x WebSphere MQ for data collection)
- 1 x DB2 database server
- 1 x WSRR Server (for the Service Catalog)

 © EPIC Project Consortium, 2013

Security: User profiles are managed through the WebSphere Member Manager, which stores user data in an LDAP database. Authentication is forms-based, through WebSphere Security and 3rd party proxy servers and support for Java 2 Security. Built-in Authorization system protects portal resources, including pages and portlets and single sign-on services include a credential vault and persistent back-end connections. Hierarchical management of access rights scales for large resource hierarchies is supported and portlet viewing permissions can be assigned by resource, by group or by user, giving flexibility both to the development teams who are creating web-services and portlets and to those managing the user-testing.

Database: IBM DB2 is a relational model database server which provides an open database environment that runs on a wide variety of computing platforms. DB2 is used by the platform for internal storage of data and was also used to store User Management data in a database structure designed by the EPIC development team. In a database environment, the support for programming language independent standards such as Structured Query Language (SQL) for the building of queries and the widely implemented Open Data Base Connectivity (ODBC) programming language middleware API ensures that EPIC specific database queries developed could be readily supported to another commercial database such as Oracle or Microsoft SQL Server or to an open-source tool such as MySQL.

MQTT Message Broker: Message Queue Telemetry Transport (MQTT) is a lightweight event and message-oriented protocol allowing small low-power devices to communicate asynchronously across bandwidth constrained networks to remote systems in a highly efficient manner and is referred to an a machine-to-machine or M2M protocol. MQTT is supported by an open-standard and the current version, MQTT v3.1 is widely used within Internet of Things applications. MQTT employs a "publish-subscribe" architecture where the device that wishes to send data "publishes" a message to the MQTT broker. Other devices that wish to receive data "subscribe" to topics and messages on the message broker. MQTT is designed to be compact and fast and is ideally suited to small low-power sensors and other devices that spend much of their time in a low-power state, "waking" periodically to send data to the broker. The user of MQTT ensures that the sensor needs no knowledge of which devices might be subscribing to its data messages. The subscription is handled by the message broker, the publisher needs only to publish to the broker. MQTT messages are transmitted across the internet using the TCP/IP protocol.

MQTT is widely supported by many industry-standard software products and by open-source tools such as Mosquitto. Within the EPIC project, the WebSphere MQ Telemetry add-on was used to handle the transmission of data for the Energy Monitoring Pilot due to simple integration with the other WebSphere components. WebSphere MQ is the industry's most widely deployed messaging middleware, featuring inherent reliable delivery and transaction control, native, high-speed handling of any type of message or file and end-to-end advanced security.

RELOCATION PILOT

The Relocation Service combined interactive queries to public and private data sources with data overlays onto Google maps, to create a personalised Relocation Service for end-users with a large-screen application and a complementary mobile component. From a platform perspective this demonstrates how web-services and applications running on the EPIC platform can access off-platform data exposed by owner-managed web-service. The platform provides a significant value-add to the native web-services by filtering, transforming and re-presenting selected data to a variety of platform clients, including portlets that consume SOAP/XML web-services and mobile clients that more readily consume JSON format web-services. The development path used also demonstrated how applications created outside of the EPIC platform using open-source tools such as Eclipse and NetBeans could be readily migrated onto the EPIC platform environment.

From an end-user perspective, the function of the Relocation Service was designed to help individuals and families planning to relocate to the city to find an ideal place to live by helping them to:

- Understand the different areas of the city and identify those areas that might best fit their lifestyle needs.
- Find available housing within these preferred areas, either for sale or for rent, that satisfy their requirements in terms of price / size / rooms / proximity to local services, etc.
- Evaluate available properties from multimedia descriptions and the property location within the surrounding neighbourhood including proximity to schools, healthcare, leisure, parks, shops, public transport, etc.
- Discover and engage with institutions, organisations and facilities targeted at the integration and support of incoming citizens, including identifying the administrative tasks needed to complete their move to the city and engagement with city services.

Data Aspects of the Relocation Service

The Relocation Service is based on a service-oriented J2EE architecture which makes extensive use of portlets and web-services as per the EPIC platform architecture description. As explained previously, the portal server delivers the portlets which present the user interface. The web-services, which deliver data to the portlets execute on the process server and on the remote database servers in which the property data

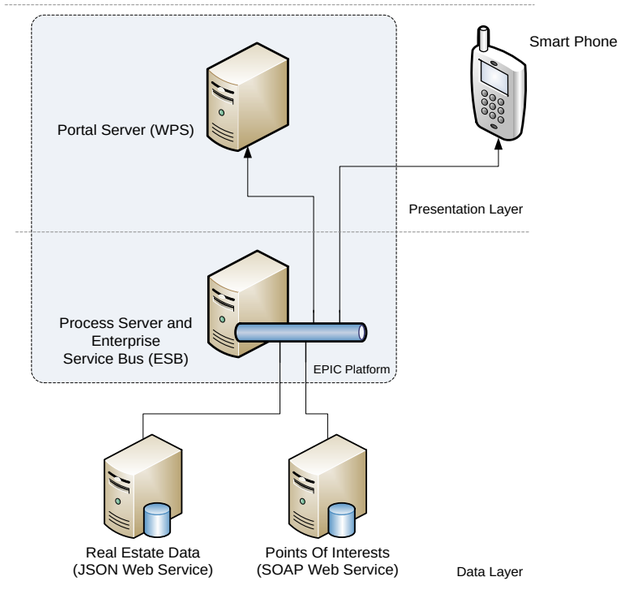


Figure 3: Data Architecture of Relocation Service

and POI data is stored by the data owners. Considerable data cleaning, filtering and data transformation is carried out by the EPIC web-services which consume the external web-services providing the raw data to EPIC. It is the EPIC platform services that provide the added-value to the raw-data services available from the external data providers and it is the combination of the private property data with the public points–of-interest data that creates the value to the end-user application. The data architecture for the Relocation Application is shown in Figure 3.

One important principal that was tested by this architecture was that of combining public and private data, by exposing this to the platform applications in a way that protected the ownership of the data but also provided the potential for future, revenue-generating services to be developed. Public points of interest (PoI) data is often available from city administrations under and open-data obligation, and commonly includes the locations of key facilities and services such as transport, healthcare, education, museums, galleries, etc. This helps provide a local context on the location within which a property is situated. Data on properties available for sale or for rent is typically not available as open-data, as this is the core commercial asset owned by Real Estate Agencies, although information on available social housing might be available from the city administration.

 © EPIC Project Consortium, 2013

A key principal of EPIC was that data owners should retain ownership of their data and that their commercial interests should be protected fully. This was done by the data owners implementing on their database servers a web-service that exposed to the EPIC platform only the data that they wished to make available in the format that was most convenient to them.

The EPIC platform is responsible for fetching property and PoI data via the external web-services and for filtering, cleaning and re-formatting this data as required by the end-user applications. This ensures that the end-users can make their data in the format most convenient for them and that the EPIC platform can transform the data into the form most available either for the large-screen or mobile applications.

To illustrate this point, the property data for Brussels was exposed by the Real Estate Agency in JSON format. This data is filtered, cleaned and then presented by the EPIC platform in both a SOAP/XML format which is readily consumed by the portlets that provide the large-screen interface, but also presented by the platform in JSON format which is most readily consumed by the smartphone application which provides the "on the ground" application for the end-user. This transformation is shown diagrammatically in Figure 4 and demonstrates the power of the EPIC platform to act as a data intermediary, to transform the data provided by the data owner as a sing web-service into a variety of formats suited to different data consumers.

Property Description Translation service

Regional Real Estate Agencies generally provide free text descriptions of properties in their regional language and for Brussels the descriptions were mainly in French. To make the descriptions understandable on a pan-European level it is necessary to undertaken some form of automated translation of the EPIC platform, to remove from the data provider the obligation to provide property description data in multiple languages. Whilst users of the Relocation Service could, of course, choose to use other translation tools such as Google Translate, the underlying problem is actually more complex than simple word-by-word translation, as equivalence of words may not provide equivalence of meaning. For example, the term duplex apartment is commonly used to mean an apartment with accommodation over two floors. However the term duplex house is commonly used to mean a dwelling having two apartments with separate entrances for two households. One particular challenge for truly automated translation of property descriptions would the creation of ontology for property descriptions to ensure there is a shared vocabulary between users creating property descriptions. This was outside the scope of EPIC but would be required to provide more accurate translation of property descriptions, ensuring that meaning and context was retained across languages. The translation service is an integral element of the platform web-services that take the property data in French from the real- estate data provider and present this to the SOAP or JSON web-service in English to the user. The translation is performed on-the-fly and allows the French property descriptions to be immediately accessible to those who cannot read French but can read English.

User-Perspective in Relocation Service

It was identified by users and developers that there were two potential use-cases for the Relocation Service. For remote exploration of the city, users would rely on a web application, typically using a large-screen computer or tablet allowing a very rich interface with high degrees of functionality. The web application was envisioned as the primary interface to initiate one's search for the best place to live in the smart city. Less restricted by the screen-size limits imposed by mobile devices, regular web applications can provide more feature rich interfaces and functionalities. Once potential locations had been identified by the user,

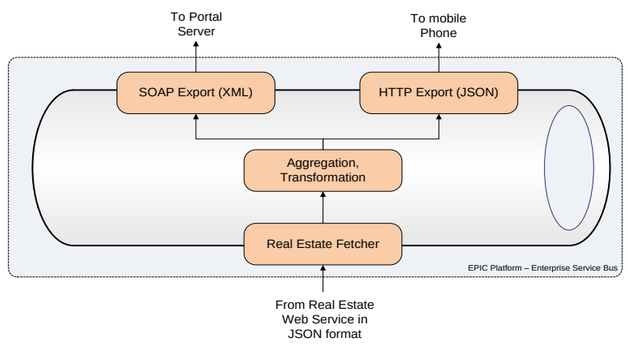


Figure 4: Transformation of Brussels Real Estate Data

these could be saved as "My Favourites" and displayed in a mobile application that was created for use "on the ground" once the user had travelled to Brussels and begun to explore the city.

Remote Exploration of the City

The main functionality of the Relocation Service application involves a preliminary evaluation of properties available for sale or for rent by user selection of four parameters that must be satisfied including location by municipality; proximity to different points of interests (transport, utilities; security and healthcare; education); type of housing and price, as shown in Figure 5.

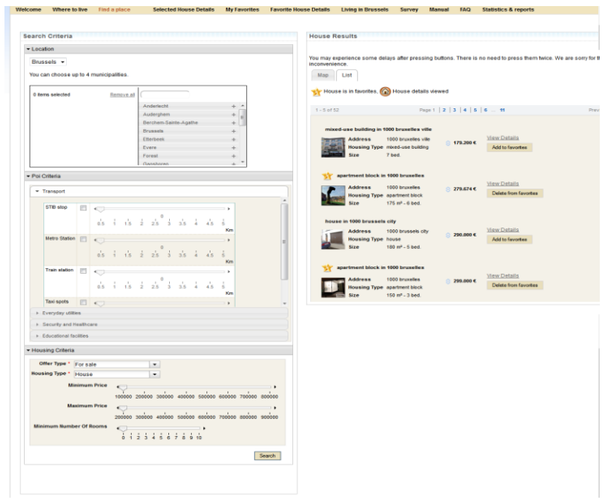


Figure 5 : Initial Property Search Screen

Selecting the map view tab displays the available properties on a map, as shown in Figure 6.

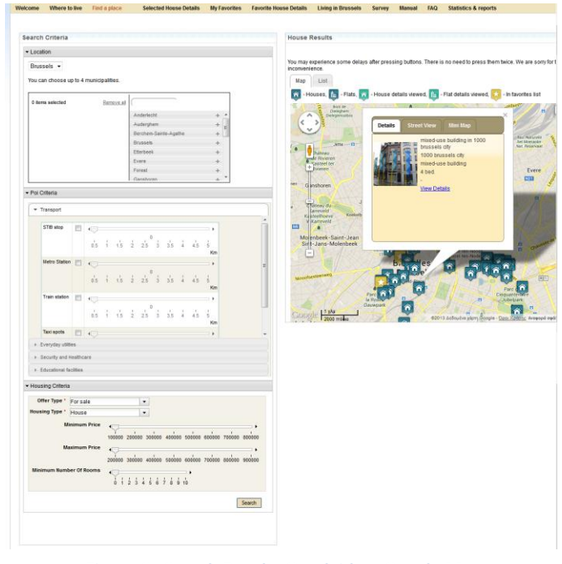


Figure 6 : Search Results Overlaid on Google Map

Clicking on individual properties displayed on the map exposes a more detailed view of the selected property selected by the user for further evaluation as shown in Figure 7.

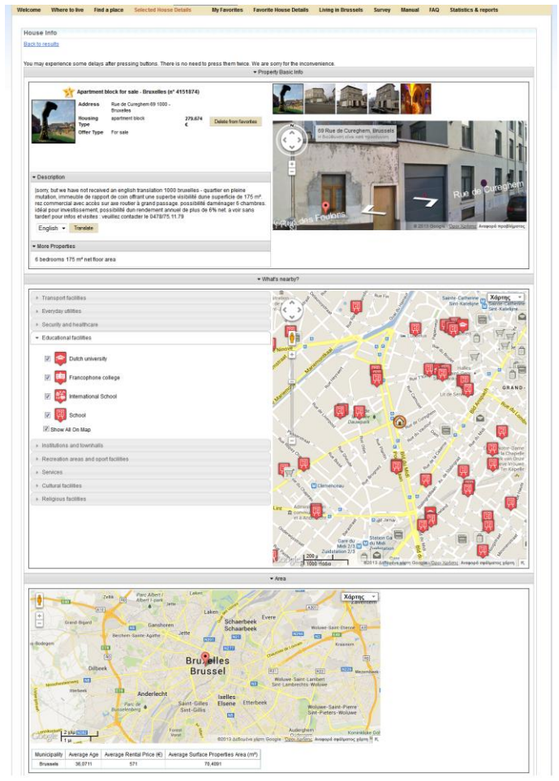


Figure 7 : Detailed view of a property

Statistical information about properties in the city could also be displayed, as shown in Figure 8, including average age, rental price and floor area of the properties within the selected area, to help inform the user.

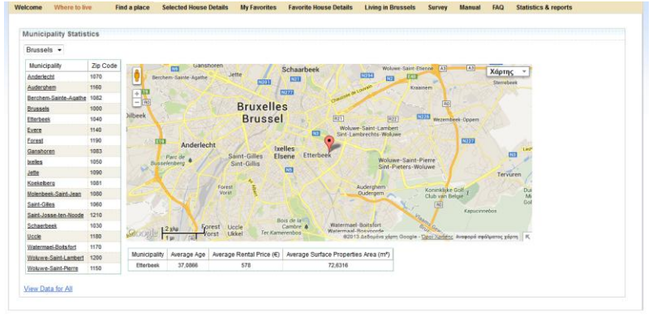


Figure 8 : Statistical Information

My Favourites

The My Favourites window provides the user with a mechanism to store details of the properties and associated points of interest for retrieval in subsequent sessions either via the large-screen application or the mobile application, as shown in Figure 9.

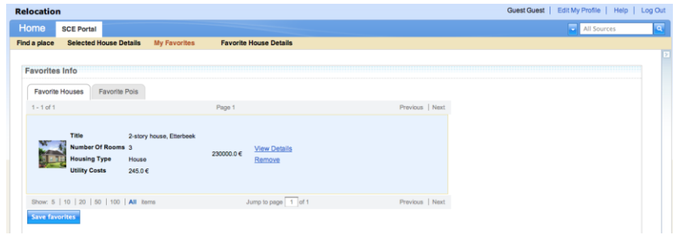


Figure 9: Favourite properties and points of interest

Living in Brussels

Moving to a new city involves much more than simply finding somewhere to live that satisfies the user's requirements. A number of legal and administrative matters related to housing, healthcare, education and other city services must be carried out to integrate the user into the new location which can be difficult for the new city dweller, especially when these are distributed across multiple departments in the city administration. To facilitate this process, the Living in Brussels section gives the user a clear overview of these institutions and facilities, their purpose and contact details. For ease of browsing this section is grouped in different categories.
These categories hold information related to "Residence Formalities", "Working in Brussels", "Housing", "Transport and getting around", etc as shown in Figure 10.



Figure 10: Living in Brussels links and resources

The Mobile Application

The remote virtual exploration of the city in terms of areas to live, available properties and points of interest in normally carried out remotely by the user, prior to travelling to the city. The mobile application is designed to complement the large-screen application by storing favourites identified during the remote exploration. Local exploration involves the user travelling to the city and undertaking an on-the-ground exploration visiting the places that were saved as favourites while using the web application. The mobile application also allows users to find new places from within the application itself and to discover the neighbourhood and its points of interest in more detail.

Although the mobile device could consume the web-application portlets delivered by the EPIC platform, it is unlikely that the web-application, developed for a large screen, would be well-suited to the smaller screen and would also require that large volumes of data are delivered to the mobile device across a typically cellular data connection, introducing significant roaming charges for the user when off their native network.

It was decided therefore that a native application would be developed for smartphones running iOS or Android, the two most common smartphone OS currently in use. The user interface would be developed as a native application but the mobile application would still consume the EPIC platform web-services to deliver property data and PoI data as per the web application, demonstrating the ease of re-use. This was necessary because the available property data could be changing in real-time as properties that were available were sold or rented and new were added by the Real Estate broker. It was therefore not appropriate to export into the mobile platform a snapshot of the property data available at the start of the visit and use this as a local property data cache.

Once again the power of the EPIC platform was demonstrated by consuming into the mobile app the property and POI data available from the platform. By default, EPIC web services are made available as SOAP/XML web services, but SOAP is not ideal for mobile applications, due to the size of the data transfer involved. Therefore an alternative format of web-service was presented by the EPIC platform in the form of RESTFul/JSON web-services which also included the necessary authentication for the mobile client. This was needed because the web-application authentication was integrated into the welcome portlets. Because the mobile device was not consuming portlets but simply consuming platform web-services, a separate authentication mechanism was required.

For ease of development and to provide a cross-platform development environment, the mobile application was built using Appcelerator

Titanium, allowing the same source code to be cross-compiled fir iOS or Android application, with only minimal changes.

Example screen-shots from the device emulator are shown in Figures 11a and 11b.

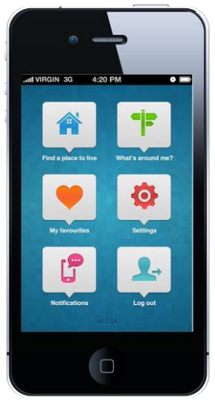


Figure 11a: Mobile App Welcome Screen

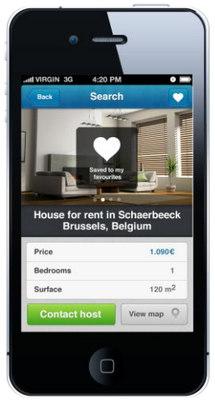


Figure 11b: Mobile View of Property Detaisl3

Smart City 3D

Introduction

Smart City 3D is an innovative web based information and 3D application that provides an intelligent service to citizens and small medium enterprises (SME). This application gives cities a tool for communicating with the public, for organizing consultations around an urban project with architects and civic associations and for taking decisions used by elected officials and technical services of the city. It also supports the development of social and economic activities.

The service enables the user to virtually fly over and to move around the digital 3D model of the city and to enter major sites. The user can opt for a general discovery of the city or choose a specific topic of interest (Leisure and Culture, Sustainable Development, Urban Planning Projects, Public Services, Enterprises).

After selecting the main topic, the points of interest (POI) in the city associated with it are highlighted on the 3D map. The user then can simply fly over the city and choose the POI they wish to receive more detailed information on. By choosing the desired item, the user has access to a media centre providing general information, pictures and videos. He/she can also take a virtual tour of the place or access its web site, for more detailed information.



Figure 12: Data Layering in Smart City 3D

Collected information from multiple sources like city administration data, public open data and wiki data are packed together within the time line to provide cohesive information about the history and city development. The Administrative tool in the application lets both the city and SME monitor, modify, update and delete the information, which keeps the information updated.

SOAP web services are implemented in the smart city 3D application to transmit all the information in a safe manner as well as to provide smart city service to other EPIC pilots to use in their application.

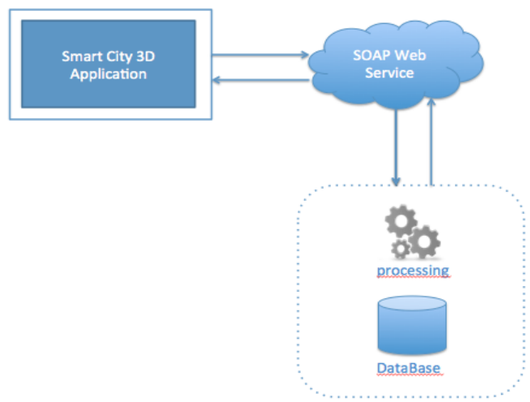


Figure 12: Smart City Service

Web Service:
Web services are implemented in the Smart City 3D application to transmit both the internal and external data between the EPIC portal and the external server in a safe manner. It also lets other pilots in the EPIC portal to use Smart City 3D service and vice versa. Web Sphere Portlet factory builders such as service provider and service consumer make it very simple to implement all these web services in the EPIC Portal.

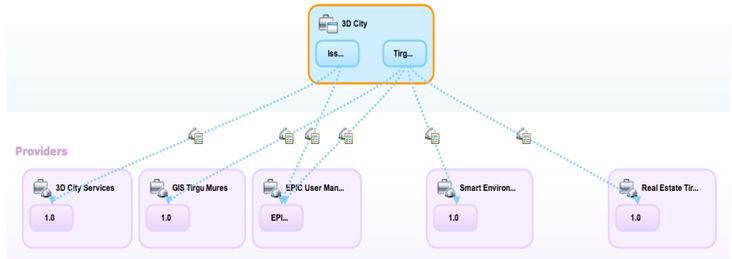


Figure 13: Web Services consumed by Smart City 3D Application.

Smart City 3D application consumed various web services like 3D city, GIS Tirgu Mures, and User Management, Smart Environment and Real estate Tirgu Mures.

Smart City 3D service:
Depending on available sources of data, Smart City 3D is able to offer within EPIC on a 3D mapping, an interactive and multimedia-modelling city to:

- Display geo information enriched with media in space and time
- Display dated urban projects (time line)
- Propose different levels of representation, from symbolic to realistic view.
  - Thematic display through layers or categories or by the introduction of a consultation filter
  - Design urban elements (lighting, trees, structures...) or places from LOD1 to LOD4 anywhere in the 3D model.
- Interpret data and transform it into information to give them meaning and be understood by different public without technical, specialist or expert support
- Manage dynamic data in real-time such as mobile phone, energy, transportation flows...
- Propose templates, forms and models for dashboard or any kind of representation

User Functionality of SmartCity 3D Service:

With this application, the city can a) communicate with its internal team to develop a strategy, b) evaluate urban projects with its partners and service providers, and c) inform citizens on issues, projects and local initiatives. This application can also be used to monitor projects, to initiate consultations, and to train administrative and technical teams to the vision and the development strategy owned by the official elected.

Three different views are presented, based on roles assigned to users of the EPIC platform, namely the citizen view, the city view and the enterprise view.

Citizen View:

A typical end-user connects to the EPIC platform and accesses the Urban Planning service in order to find information about the city's layout and its major sites as well as planned changes to the built infrastructure. The service enables the user to "fly" virtually over the city by exploring the digital 3D model of the city and to enter major sites presented in the model. The user can opt for a general discovery of the city, or they can choose a specific topic of interest. After selecting the main topic, the points of interest in the city associated with it are highlighted on the 3D map. The user then can simply fly over the city and choose the POI to have information about. By clicking on the desired item, the user has access to a media centre providing general information, pictures and videos and they can also take a virtual tour of the place or access its web site, for more detailed information.

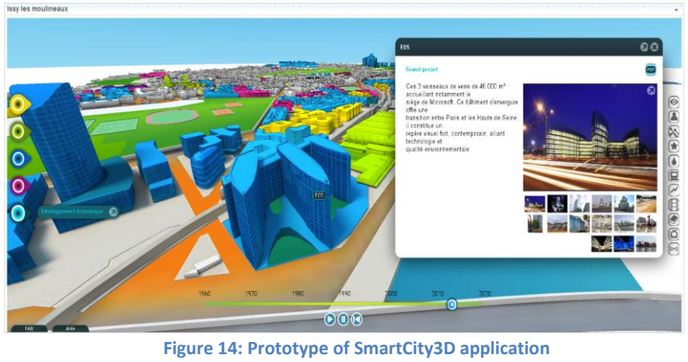


Figure 14: Prototype of SmartCity3D application

By choosing the topic "Enterprises", the user can view all the companies registered in the city (in a 3D aerial view). Again, additional information is available, in this case provided by the respective company itself. This information might contain photos, presentation videos, latest job offers or a URL to a web-site from which further information be obtained.

City View:

The Urban Planning service is very useful for the city administration, allowing it to maintain up-to-date information (content and media) on the platform via a customised administration tool.

At the same time, the service offers the city administration a detailed view of the local economical fabric, in order to follow its activity and stay up-to-date with the latest news and events in the sector. It will also allow the city administration to train internal teams to the vision and the development strategy for optimizing "communication" and "decision support".

Enterprise View:

In addition to supporting citizens to explore the 3D city model and to help the city administration communicate development timelines and economic information about the city, SmartCity 3D also supports the SMEs within the city to be more economically active and competitive. By helping them to promote their services and products and also to intercommunicate with other business within the city. A regular SME accesses the section "Enterprises", wanting to inquire about similar products and services in the city or simply just to get an idea about other companies located in the city. After selecting the topic, all the city's SMEs are highlighted on the 3D city map. By clicking on one particular SME, information such as photos, presentation video, or the official website for further contact appears. The authentication procedure is launched if the user wishes to get in direct contact with a particular enterprise.

A private communication system (possibly in the form of a forum and accessible via the user's personal account credentials) will be put in place in order to allow fast exchange of information and to enhance the communication flow among the SMEs

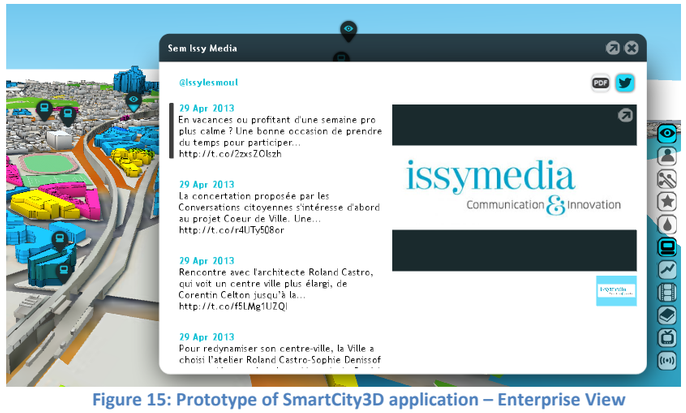


Figure 15: Prototype of SmartCity3D application – Enterprise View

Smart City 3D Implementation:

Smart city 3D application is now developed and deployed for two cities in Europe through the EPIC project. In the future, smart city 3D application can be built for any city in the world to provide a web-based interactive 3D view. . In order to provide smart and detailed information to the citizen and business people, Smart City 3D application consumed EPIC services, some other external services and sources of open data and geo-localized social networking data like GIS data of the city, Wiki to provide more useful information like transport, medical around the city.

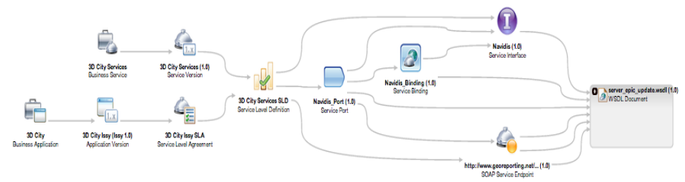


Figure 16: Service Catalogue for Smart City 3D Service

Within EPIC, SmartCity 3D has developed a unique "3D immersive" visualization concept to discover a city, its surroundings and its projects in a form this is intuitive, informative and entertaining.

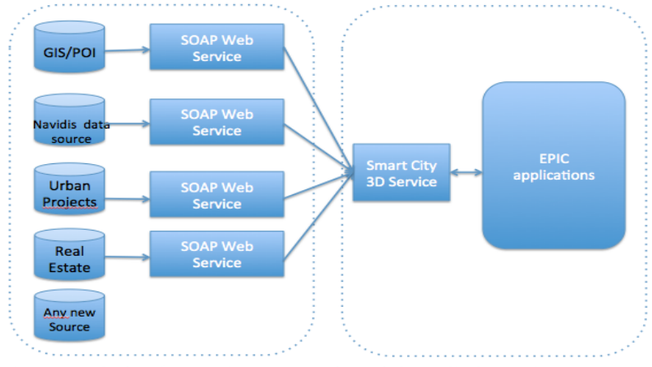


Figure 17: POI data from various sources through SmartCity3D service

Building SmartCity 3D for Other Cities

A particular focus of the EPIC platform was the support for re-use of web-services, portlets and user management services for other cities wishing to access smart-city services from the platform or for developers involved in developing such services. A key focus of the EPIC platform web-services was the abstraction of the details of the underlying property data and POI data from the end-user and developer, by presenting a standardised interface irrespective of the nature of the source data exposed by the data owner. However, for the SmartCity 3D

service, there is a degree of localisation required due to the basic exploration tool taking the form of a 3D city model.

A solution can be proposed to any City based on a standard 3D model build with available data and displaying geolocalised POI dispatched on 3 layers and coming from e.g. Open Street Map & Wikipedia and which will include the following steps:

1. Define the city requirements
2. Identify the data available :
    a. Data gathering from the city (GIS, Cadaster, Urban model library)
    b. Data gathering from other places (Open Data and any other external Sources)
3. Propose and validate the 3D Model
4. Define user interfaces and user interactions specifications
5. Develop additional features if required
6. Get it integrated into the SmartCity 3D Framework
7. Test and validation with the City
8. Publishing for the web and for any other devices

Once the integration is completed, a prototype is presented to the City client. Successive tests are performed to optimize the application and make any required changes leading to the complete recipe of the application.

A Platform and Technical Perspective of Smart City 3D

SmartCity 3D was a very different pilot to Relocation and Smart Environment, in terms of both end-user functionality and platform requirements. Unlike Relocation and Smart Environment, Smart City 3D was not developed entirely within the EPIC project or built specifically for the EPIC platform, but was built from NAVIDIS pre-existing framework which has been enriched, consisting of an "Editor" and a "Player" running on the game engine Unity 3D which "*is a game development ecosystem: a powerful rendering engine fully integrated with a complete set of intuitive tools and rapid workflows to create interactive 3D content; easy multiplatform publishing*".

This application ensures that it could run on the EPIC platform and re-use EPIC platform services such as user-management and web-services. Furthermore during the redeployment into Tirgu-Mures, Smart City 3D consumed web-services exposed by other Pilots, including Tirgu-Mires POI's and public building energy data made available as a web-service by Smart Environment.

The Smart City 3D Framework

- is scalable
- can act as a middleware within EPIC (remaining adaptable to any type of infrastructure)
- has its own set of program interface to implement
    - designed customer interfaces
    - model and representations of static or dynamic data
    - set of interactions on dot, line or volume
    - for adding enriched features
    - for aggregations of services developed elsewhere
- supports various devices like PC, MAC, Touchscreen, Tablet, Smartphone

The successful implementation of Smart City 3D demonstrated that an existing application could be readily integrated into the EPIC platform and readily consume platform web-services including user-management to handle authentication and roles-assignment. This was critical to maximise the market acceptability and relevance of EPIC as a versatile smart city platform which was capable of supporting both pre-existing applications to run within the platform in addition to new applications developed specifically for the platform.

Smart Environment Service

Given the importance of carbon reduction and energy sustainability to aspiring smart-cities, the Smart Environment pilot looked at two key aspects of energy usage, namely electricity used by citizens in their homes and the electricity and gas used in public buildings. The aim of this pilot was to make visible to citizens and the city administration real-time and historic information on energy usage within citizens' homes and public buildings operated by the city administration.

The Smart Environment pilot had a level of technical complexity not found in the other pilots due to the requirement to collect real-time electrical power data from homes and energy data from the smart-meters in public buildings and offered much less of the user-interaction found in Relocation and SmartCity 3D.

The BCU and IBM technical teams worked closely together to design and implement a complete end-to-end system for collecting and storing the energy data updates received from each home every six seconds. Smart Environment was the only pilot where Internet of Things (IoT) data connectivity protocols were implemented, through the use of the well-known and open-source Message Queuing Telemetry Transport (MQTT) protocol v3.1. The requirement to collect, store and post-process significant volumes of energy data required the design of a back-end database that would be suitable for both domestic energy data and the public buildings energy data and be readily extensible for other sensor data types. The architecture of the database, termed the Generic Monitoring Framework (GMF) allows the storage of a wide range of sensor data. The GMF was developed using the open-source database MySQL and was implemented off the EPIC platform on a server located at Birmingham City University. Web-services to insert data into and extract data from the GMF were developed on the GMF server and exposed to the EPIC platform through "helper" web-services hosted on the EPIC platform. This design follows the standard EPIC solution architecture, where 3rd party data is stored off-platform and exposed by 3rd party web-services running off-platform, accessed by web-services running on the EPIC platform. A schematic of the resulting smart environment pilot can be found is Figure 18.

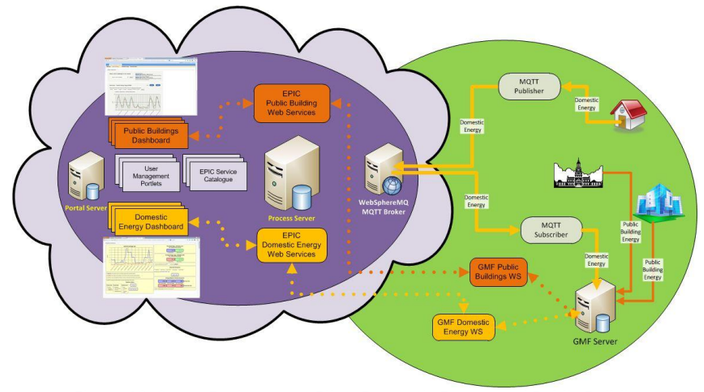


Figure 18: Smart Environment Pilot: Domestic Energy and Public Buildings

Collection of Domestic Energy Data

For domestic energy, the project team was required to develop an in-home solution to capture domestic energy data and relay this to the GMF for storage. The well proven commercially available EnviR monitor manufactured by Current Cost Ltd was used for the in-home capture of electrical power, using a wireless inductive clamp meter attached to the consumer unit and a stand-alone LCD monitor for display of energy data. Because this had no in-built connection to the Internet, the BCU and IBM technical teams worked together to implement an internet data bridge using a combination of customer-firmware in a commercial mini-router and the MQTT protocol for data transfer, supported by the installation on the EPIC platform of the IBM MQTT message broker, WebSphereMQ. MQTT, which is well-proven open-standard protocol for internet scale communications, ideally suited to low-power embedded sensors and other devices. MQTT is a Publish-Subscribe or "Pub-Sub" architecture, where a message broker sits between the data sources, namely the publishers and the data consumers, namely the subscribers. Pub-sub architecture is ideal for large-scale networks where the publisher is not

required to have knowledge of the subscriber, but simply has to publish the data to the message broker which receives the data-flows from the publishers and relays these to the appropriate subscribers.

The clamp meter does not measure energy consumed but measures the electrical power flowing into the household, proving a reading of instantaneous power in Kw once per six seconds. The power readings are published MQTT messages to the WebSphereMQ message broker on the EPIC platform once every six-seconds, equating to 14,400 power-usage updates per home per day. Within the scope of the EPIC project, it was planned that the homes of at least 10 closed-group users and 50 open-group users would be allocated the necessary monitoring equipment for connection to the smart-environment pilot. A schematic of the domestic energy collection is shown in Figure 19.

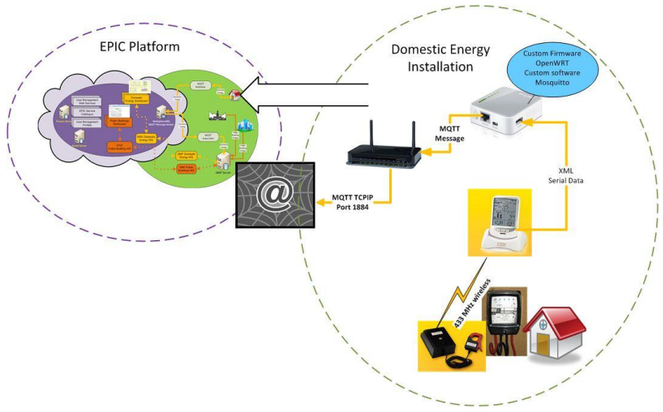


Figure 19: EPIC Domestic Energy Monitoring Schematic

The EnviR monitor from Current Cost Ltd be positioned anywhere within wireless range to display instantaneous power and historic cumulative energy usage data. The EnviR base unit receives a power update at 6-seconds intervals, 14,400 power readings per day per unit. The EnviR has no internet connectivity, so the BCU and IBM technical teams re-flashed the firmware of a commercial TP-Link mini-router to create an MQTT publisher to send power data to the WebSphereMQ message broker. The user does not need to leave a PC switched on but their broadband connection must remain on, or data is lost. A typical installation of the clamp and transmitter in a domestic installation is shown below.



Figure 20: Domestic Installation of Inductive Current Clamp and Wireless Transmitter



Figure 21: CurrentCost EnviR Display Unit and re-Flashed TP-Link Mini-Router Internet Bridge

The TP-Link MR-3020 uses the Open-WRT firmware, a Linux distribution for embedded devices http://openwrt.org) which means that the router firmware can be modified and additional code added to change the functionality of the device. To act as the gateway between the serial data output from the EnviR and the IP address of the WebSphereMQ broker, the firmware was required to receive the serial data; parse it to extract key information; add a UTC timestamp and the MAC address of the router which was in effect the siteID the serial and package this data as an MQTT message and publish it to the broker. The TP-Link was connected between the EnviR and the user's broadband router or home network using the serial-USB cable supplied with the EnviR and a standard Ethernet cable. The software to parse the incoming serial data was written in C language and could be readily tested on a PC before being included in the image for the TP-Link. Mosquitto is a freely downloadable and well-proven open-source MQTT v3.1 broker (www.mosquitto.org) and could be readily included into the TP-Link firmware image to provide the MQTT publishing required.

WebSphereMQ is a telemetry add-on to WebSphere and is a well proven IBM commercial product and add-on to WebSphere. This was installed on the EPIC platform and configured by IBM to accept the EPIC energy data messages published by the TP-Link. Testing of the end-to-end domestic energy data publishing solution, i.e. EnviR to WebSphereMQ, was accomplished by monitoring the messages received by the WebSphereMQ broker to ensure that these were received correctly and intact.

WebSphereMQ to Generic Monitoring Framework Communication

The GMF subscribes to EPIC energy messages received by the message broker, and these are relayed by the broker as they are received. MQTT defines three levels of Quality of Service (QoS), which controls how diligently the broker/client will try to ensure that a message is received:

- QoS 0: The broker/client will deliver the message once, with no confirmation.
- QoS 1: The broker/client will deliver the message at least once, with confirmation required.
- QoS 2: The broker/client will deliver the message exactly once by using a four step handshake.

Messages may be sent at any QoS level, and clients may attempt to subscribe to topics at any QoS level. Higher levels of QoS are more reliable, but involve higher latency and have higher bandwidth requirements and may involve significant local caching of messages on publisher or broker where connectivity is disrupted.

For domestic energy monitoring, data messages are created every six seconds by each EnviR and in the context of the application, namely understanding cumulative energy consumption, the non-delivery of individual data messages is not consider problematic and guaranteed delivery of messages is not mission critical . Domestic energy usage is calculated by integrating a stream of six-second power readings over a five-minute period, nominally fifty readings, to calculate the average energy used in that period, so the effect of a few missing readings is minimal. It is also important, given the high volume of MQTT messages, that if the remote GMF server were to go off-line for any reason, the WebSphereMQ broker is not required to queue potentially tens of thousands of messages until connection to the GMF is restored, which it would be required to do if a guaranteed delivery QoS was used. Furthermore, from prior experience it is known that many domestic broadband connections may be rather unstable, and subject to periodic disruptions. Again, due to limited internal memory, if the connection between the TP-Link and Broker is disrupted then the six-second data updates from the EnviR are discarded until the data connection to the broker is restored. Many of the TP-Links have continued to send data reliably for several months, and have recovered from drop-outs in the user's broadband without user intervention or resetting.

Public Building Energy Data Collection

Most public buildings have a building management system (BMS) that controls all of the heating, ventilation and air-conditioning (HVAC) systems within the building and monitoring energy usage. Furthermore public buildings normally have fiscal smart-meters installed by the utility companies which relay data on electricity, gas and water consumption

normally at 30 minute intervals. BMS systems also commonly implement sub-metering so that, for example, total electricity usage can be split into that used for different purposes and within different zones.

The public buildings element of Smart Environment therefore aimed not to install new monitoring equipment, but to make use of existing data flows accessible via the BMS. Four Manchester buildings were monitored, namely two Manchester City Council buildings including the Town hall and two enterprise academies, all of which were connected to the Manchester BMS network. The reporting abilities of most modern BMS enable the scheduling and generation of reports that can sent to specified email recipients. The Manchester BMS system created a separate daily report for each building, and this was sent as an email with an MS Excel attachment and a dedicated email mailbox epicdata@bcu.ac.uk was created at BCU to receive the scheduled reports.

When an email was received, the attachment was removed and processed and the data records were inserted automatically into the GMF database. A software module - Data Harvesting and Ingest Module (DHIM) was used to harvest incoming emails from the epicdata@bcu.ac.uk mailbox and ingest directly to the GMF using a GMF web service 'The storeSensorValue'. The DHIM uses IMAP to connect and retrieve emails from the epicdata mailbox and emails are processed based on known compression and mapping requirements resulting from different email encoding, data file compression or BMS data format. This is shown schematically in Figure 22.

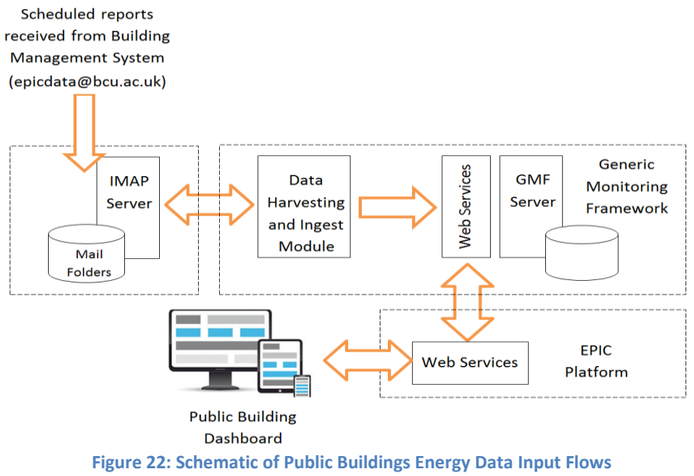


Figure 22: Schematic of Public Buildings Energy Data Input Flows

GMF Energy Data Web-Services

As per the agreed architecture for EPIC, the GMF in essence represented a third-party off-platform data source similar the POI data or Real-Estate data. Therefore data was exposed to the EPIC platform through web-services running on the GMF server which were invoked by web-services running on the EPIC platform. This was a direct analogue of the architecture for POI and property data, although both the GMF and EPIC web-services were created by the BCU technical teams. Due to the differences in domestic energy data (instantaneous power readings in Kw) and public building data (total energy consumed in 30-minute time frames), separate sets of web-services were created for domestic and public building energy data streams.

The domestic energy web-services had not only to calculate and present the smoothed power-profile for the specified property, but also to calculate the total energy used across the specified time-period by integrating the power curve. The domestic web-service was also required to calculate the average smoothed power profile of all properties to provide the red trace on the domestic dashboard and integrate this to calculate the average energy consumed. Domestic energy data was a particular technical challenge due to the large volumes of data, as 14,400 power readings were received per house in each 24 hour period. The domestic energy web-services created are shown in Figure 23.

Domestic Energy Web Service	Function
getSensorAveragesOverPeriodByID	Averages power readings from specified sensor.
getDomesticSensorAverages	Averages power readings from all matching domestic sensors.
getTotalConsumedOverPeriodBySensorID	Calculates energy consumed by integrating power profile for specified sensor.
getAverageConsumedOverPeriodAllDomesticSensors	Calculates energy consumed by integrating power profile for all domestic sensors.

Figure 23: Domestic Energy Web Services

The public buildings web-services were much simpler than those for domestic energy, primarily because the data received was the amount of energy used in the preceding 30-minute interval, as this data was calculated directly by the smart-meters or BMS sub-metering. The main requirement of the web-services was therefore to allow retrieval of data from a given sensor (meter) over a given range, but an option was provided to provide this as the energy consumed per hour or the energy consumed in the previous week, to allow meaningful comparisons between different buildings with user controlled granularity of data. The set of web-services created for Public Bu8idlngs as shown in Figure 24.

Public Buildings Energy Web Service	Function
getWeeklySensorDataOverRange	Returns total energy used in specified week period for specified building.
getHourlySensorDataOverRange	Returns a list of hourly energy usage figures over specified period for a given building.

Figure 24: Public Building Energy Web Services

Domestic Energy dashboard

The domestic energy dashboard allows the user to view their electrical power profile and cumulative energy use over a variety of periods from 1-hour to 90-days. The graph window shows the instantaneous power demand and displays the peaks that correspond to the use of high-load appliances such as electric kettles, electrical showers, cookers, etc. This helps the user to identify the effect of using different appliances. By entering per-unit costs for their electricity, the dashboard also provides an estimated cost of the energy usage over his display period, but this is intended only as a guide and is not a bill-estimator. In addition to the personal power profile, the user can also view the average power profile and the average energy usage of properties of a similar type for comparison with their personal energy usage. An example screen-capture of the domestic energy dashboard is shown in Figure 25: Example Screen Grab of Domestic Energy Dashboard. The 8 Kw load between 10:04 and 10:19 corresponds to the use of an electric shower, with the rise to 11 kW in the middle corresponding to the boiling of a kettle, adding an additional 3 Kw to the 8 Kw shower loading.

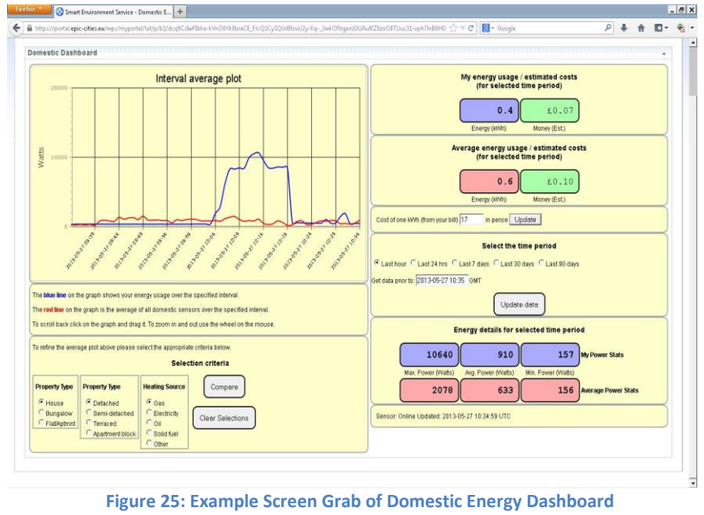


Figure 25: Example Screen Grab of Domestic Energy Dashboard

Public Buildings Dashboard

Modern public and commercial buildings typically have a building management system (BMS), with data accessed from the system "head-end" which allows detailed drill-down into the building controls network. The public building dashboard in Smart Environment is not designed to replicate the BMS functionality, but to provide a live extension to the energy certificates commonly found in public buildings. This helps a city to demonstrate to its citizens how they are achieving energy reduction, particularly by comparing between buildings or between different operating years of the same building. The public buildings dashboard is a separate portlet that takes a rather different form to the Domestic Energy dashboard, as shown in Figure .

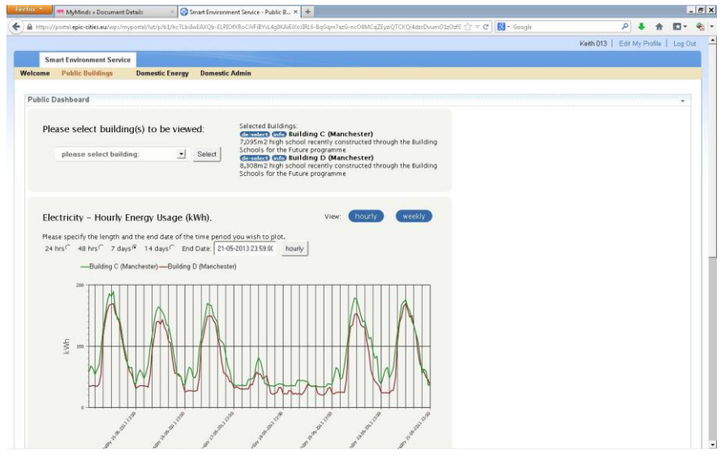


Figure 26: Example Public Building Dashboard

Repurposing Smart Environment for Tirgu-Mures

Energy monitoring for public buildings in Tirgu Mures was made difficult because the buildings to be monitored did not have smart meters and were not connected to a BMS which could generate the daily energy usage exports. The interest of Tirgu-Mures was to compare the electrical energy used by three schools which had 3-phase electrical supplies as is normal for commercial buildings, other than the single-phase used in domestic premises.

The domestic monitoring hardware was therefore modified, as the EnviR transmitter could handle 3 separate current clamps, one per phase of a three-phase supply. The firmware of the TP-Links was modified to handle the data from the three inductive clamps (one per phase) and the Tirgu Mures sensors were identified as Public Buildings in Tirgu Mures. The data from the Tirgu Mures public buildings was transported using MQTT and stored in the GMF but not combined with the other domestic data.

The Manchester public building data comprised energy-used in each 30-minute timeslot, yet the Tirgu Mures public building data was in the same format as the domestic energy, namely the electrical power readings received at 6-second intervals via the MQTT transport. The Public Buildings web-services were therefore modified to become very smart web-services to handle both Manchester and Tirgu Mures data in a manner transparent to the user, rather than creating separate web-service variants for the Tirgu Mures public buildings. If the data for Manchester sensors was requested this was returned by the web-service in the normal way, from the 30-minute energy reading supplied by the Manchester BMS. If data was requested from Tirgu Mures public buildings sensors, the web-service processed the necessary 30-minute energy consumption figures from the 6-second Tirgu Mures public buildings power readings in the GMF by integration over the requested period. This allowed, for example, a Tirgu Mures building to be compared with Manchester building, which would not have been possible had separate dashboards with separate web-services been created for Manchester and Tirgu Mures.

The modification Public Buildings web-service demonstrated very clearly the advantages of using web-services to expose third-party data. It was not necessary for the user of the web-service to know that the underlying data formats of Manchester and Tirgu-Mures buildings were different, as the complexity of processing the Tirgu-Mures data has handled transparently to the user, who was presented with data in the same form irrespective of source.

The Integrated Scenario and Tirgu-Mures Implementations

As explained at the outset of the project, the pilot architecture utilising the EPIC platform, web-services and portlets to deliver the end-user application was designed to allow simple reuse of web-services and to accelerate the creation of new user-interfaces by modifying existing portlets. This was verified by the integration of the three separate pilots to create a unified smart-city solution for the city of Tirgu-Mures in Romania. The successful and rapid integration and redeployment validated the EPIC platform architecture and the methodology used for implementation of the pilot applications.

Tirgu-Mures Data Sources

The first requirement for each Pilot was to obtain data that was relevant to Tirgu-Mures, rather to the cities of Brussels, Issy-les-Moulineaux and Manchester for which the three separate pilots has been initially created.

Points-of-Interest (PoI): The Tirgu-Mures city administration maintained a comprehensive database of points of interest with the required GIS data that could form the basis for the Tirgu-Mures PoI data. A web-service was developed on the T-M servers which exposed the PoI data as a web-service and a new web-service of the EPIC platform was developed to present the Tirgu-Mures web-service data a form to the presentation of Brussels PoI data, allowing simple re-use by "plugging in" the T-M data.

Property Data: The city administration in Tirgu-Mures maintained a register of city owned properties that were available for rent. Again a web-service was developed by the data-owners to expose this data. A T-M property web-service was then developed on the EPIC platform which presented property data in a format identical to the Brussels property data, permitting seamless re-use by the consuming portlets that created the T-M Relocation interface.

3D City Map: A partial 3D city map was created by Navidis from 2D data and building heights for key Tirgu-Mures buildings, allowing a pseudo 3D city map to be created or a limited are of Tirgu-Mures.

Domestic Energy Data: The hardware used for the Manchester homes, namely the CurrentCost EnviR and TP-Link was re-used without change for the Tirgu-Mures and data was transferred to the GMF repository and retrieved by the domestic-energy web-services.

Public Building Data: The schools in Tirgu-Mures did not have smart-meters, so no BMS data was available. The domestic energy monitoring hardware used for homes was modified to measure 3-phase electrical power and fitted with three current clamps, one per phase. The data was tagged as Tirgu-Mures Education data and stored in the GMF as a six-second power reading. The web-service to retrieve public building data was significantly extended to ensure that, when data from T-M buildings was retrieved, the filtering and integration needed to calculate a "pseudo 30 minute energy usage" value was carried out. The public building energy data web-service therefore retrieved data, processed and output public building data in a consistent format, irrespective of the way the data was stored in the GMF.

Relocating to Tirgu-Mures: The relocation application for Tirgu-Mures readily was created from the new web-services and modified portlets, to replicate the functionality of the Brussels relocation application including a mobile component. The consistency of data format resulting from the new PoI and Property web services made the reworking of the portlets very simple. This pilot also consumed domestic energy data web-services for Tirgu-Mures, overlaying this onto the domestic properties for which data was available.

SmartCity 3D Tirgu-Mures: This was readily repurposed by Navidis using the new 3D map and the Tirgu-Mures PoIs were used to provide key points of interest in the city. This pilot also consumed data from the public-buildings web-service to overlay energy data onto the public buildings in the 3D model for which energy data was available.

Smart-Environment Tirgu-Mures: The portlets for Tirgu-Mures were unchanged although language support for Hungarian and Romanian was added through the regionalisation settings of the Portal Server. Because there was no geographic information contained in the portlets and the energy web-services retrieved data in the same format, minimal changes were required to adapt the Manchester portlets for Tirgu-Mures.

CONCLUSIONS

The aim of this project was to validate the concept and design of the EPIC platform architecture for the delivery of smart city services through the development, integration and redeployment of three smart-city pilot applications.

The EPIC platform utilised well accepted open standards for platform services and data-delivery web services (SOAP and RESTful); user-interface portlets (JSR-168 and JSR-268) and the transport of sensor data (MQTT), to allow simple re-use and re-purposing of the pilots developed within EPIC.

The platform and the three pilots developed within EPIC were designed to explore a wide range of functionality that would be required by a platform that supported a broad range of smart-city services ranging from the support of citizens and the city administration through to the provision of smart-city data as managed web-services to SME software developers.

The Relocation pilot demonstrated how a secure and managed combination of public data and private data exposed through the data owner's web-services could be filtered, combined and transformed to provide a value-laden rich data stream of points-of-interest and property for sale or rent, to be displayed on a map of the city. Relocation also demonstrated how the EPIC platform could transform data presented by the owner as an unfiltered SOAP/XML web-service could be filtered and cleansed and represented both as SOAP/XML format readily consumed by portlets and in the JSON format preferred by the native applications running on mobile devices.

The Urban Planning pilot through SmartCity3D, demonstrated how an existing application developed outside the EPIC platform could be ported into the EPIC platform. Once integrated, SmartCity3D used EPIC platform web-services for user-management and consumed the Points-of-Interest web-service and Energy web-services to provide city landmarks and public building energy data on an interactive 3D city map.

The Smart-Environment pilot demonstrated how the Internet-of-Things protocol MQTT could be used to transport reliably at six-second intervals energy readings from homes through the EPIC platform to an off-platform database. Smart Environment also showed how the smart-meter data available with the BMS used in public buildings could be repurposed to provide visibility to citizens of the effective of carbon reduction measures employed by the city administration.

Through the development, operation, user-testing and redeployment of these pilots, The EPIC project defined and validated a number of principles which can be used widely post project, and validate the platform architecture as follows:

- Cloud Platform: the project demonstrated the viability of using a cloud approach for the evaluation and delivery of smart city services, minimising hardware cost and providing elasticity to meet highly variable user demands
- Open Standards: the principle of open standards, which was core to the project, will enable the lessons learned from the EPIC platform to be readily replicated and applied by others.
- Device Agnostic Solutions: the structure developed during the project has a minimised reliance on specific hardware, enabling the platform to be hosted by devices of the end users.
- Design for Re-Use: the varying pilots demonstrated how web-services and portlets enable easy re-deployment for any new city
- "User-in-the-Loop" Development and Test: Three very different pilot applications were implemented on the platform and tested with citizens engaged through the European Network of Living Labs (ENoLL) to ensure that the platform could be tested across a diverse range of current and future application scenarios.

 © EPIC Project Consortium, 2013