ISSN 1870-4069

Algoritmos para el control de indicadores clave de desempeño para Smart Cities

Elsa Estrada, Adriana Peña Pérez Negrón, Graciela Lara López, Luis Casillas

Universidad de Guadalajara, CUCEI, Departamento de Ciencias Computacionales, Guadalajara, Jalisco, México
{elsa.estrada, ciela.larea}@academicos.udg.mx,
{adriana.pena, luis.casillas}@cucei.udg.mx

Resumen. En cada ciudad emergen diversos y diferentes tipos de situaciones a las que la ciudad debe adaptarse, lo que implica un cambio continuo durante su proceso de evolución, mismo que involucra un gran número de variables. Este esquema dificulta la integración de metodologías y herramientas para medir los indicadores que permitan la toma de decisiones que lleven al mejoramiento de las condiciones de habitabilidad. La selección de indicadores clave de desempeño de una ciudad conforma un modelo. En este trabajo se presenta el diseño y un prototipo que implementa los algoritmos de un esquema general para el control de indicadores clave de desempeño para las Smart Cities. Este diseño permite además llevar un método sistemático en el quehacer de monitoreo y actualización, así como la comparación de diferentes reportes entre distintos modelos. El método aplicado se desarrolla en tres fases: estudio de modelos de métricas de ciudad, análisis del formato en el Open Data, y el diseño e implementación de algoritmos para procesamiento de objetos JSON en mongodb.

Palabras clave: smart cities, open data para evaluación de ciudades, documentos JSON de indicadores clave de desempeño, NOSQL.

Key Performance Indicators Algorithms for Smart Cities

Abstract. In each city diverse and different type of situations emerge, to which the city has to adapt itself. This implicates a continuous change on its evolution process that involves a number of variables. This scheme difficult methodologies and tools integration to measure the indicators that allows the decision making for the improvement of the living conditions. The selection of the key performance indicators of a city represents a model. In this work are presented the design and a prototype that implement the algorithms of a general scheme for key performance indicators for Smart Cities. Moreover, this design allows a systematic approach for monitoring and updating the key performance indicators, as well as comparing different reports of different models. The

applied approach is developed in three faces: analysis of the metrics model of a city, Open Data format analysis, and the design and implementation of algorithms for processing JSON objects in mongodb.

Keywords: smart cities, open data for cities evaluation, JSON documents of key performance indicators, NOSQL.

1. Introducción

Aunado al incremento de la población en las ciudades va el aumento en la demanda de recursos y servicios, y aparecen fenómenos como la falta de inclusión social e inequidad. Para mitigar estos problemas, las Smart Cities proponen el desarrollo de estrategias de mediciones que soporten la toma de decisiones, lo que supone el manejo de una cantidad indefinida de indicadores.

Chourabi, *et al.* [1], en su estudio: "*Understanding Smart Cities: An Integrative Framework*", definen el concepto de Smart City como una estrategia que emerge para mitigar los problemas a consecuencia del crecimiento de la población urbana y la rápida urbanización, cuyos indicadores clave de desempeño o kpi (por sus siglas del inglés: *key performance indicator*) que impactan en el bienestar son: la administración y organización, la tecnología, el gobierno, política, gente y comunidades, economía, infraestructura construida y medio ambiente natural. Mientras que el trabajo de Chourabi, *et al.* [1] se centra en la caracterización por kpi, Townsend [2] se enfoca en las operaciones y sus beneficios, de tal manera que se refiere a una Smart City como aquella que resuelve sus problemas a través del monitoreo de eventos y consumo de recursos mediante el uso de sensores.

Otro concepto muy relacionado con Smart City es el de "Ciudad Sustentable" [3], que enfatiza como factor preponderante a las Tecnologías de la Información y de la Comunicación (TIC) para la mejora, principalmente, de tres aspectos urbanos: 1) mejora de la calidad de vida, 2) la eficiencia de operación urbana y de servicios, y 3) el abastecimiento de las demandas económicas, sociales y ambientales.

Las Smart Cities surgen como medio estratégico para resolver problemas que afectan tanto a la sustentabilidad como a la calidad de vida, impulsando el desarrollo de instrumentos para monitoreo y evaluación de fenómenos urbanos. Ejemplo de dichos instrumentos son: los modelos de métricas para evaluar ciudades, así como los sistemas autónomos que implementan diversos tipos de componentes que incluyen al internet de las cosas (IoT por sus siglas del inglés: *internet of things*), algoritmos *data science* y sistemas inteligentes para la toma de decisiones, infraestructura SDN (Software Definido por la Red) o los estándares de transferencia 5G.

1.1. Modelos de métricas de ciudades

Las métricas son entonces las herramientas para facilitar la toma de decisiones y mejorar el desempeño en las ciudades de por ejemplo, la calidad de vida y el bienestar. Sin embargo estos son factores abstractos e intangibles de tal manera que no es posible medirlos directamente. Por lo que se apoyan en mediciones que se

expresan en indicadores, por ejemplo el porcentaje de población de 15 años de edad con primaria terminada, o bien el porcentaje de emisiones de gas efecto invernadero.

De acuerdo con Jones Lang Lasalle [4], las mediciones son una de las vías para evaluar la complejidad del cambio urbano y juzgar qué enfoques tienen éxito y cuáles no. Los índices que nos ayudan a evaluar son herramientas útiles para los gobiernos, negocios y organizaciones. Nos ofrecen oportunidades tanto para aprender sobre las fortalezas y deficiencias no detectadas, como para persuadir a la ciudadanía en realizar cambios para incrementar el bienestar. De tal manera que la utilidad de los índices se resume en 5 objetivos: 1) evaluación y diagnóstico, 2) comparación de condiciones, 3) apoyo y persuasión, 4) pedagogía, y 5) generar confianza en el vivir. Las Smart Cities por lo tanto están interesadas en medir la calidad de vida de una ciudad para identificar áreas de mejora y así dar soporte a la toma de decisiones. Con este fin se han presentado por un lado modelos de estructuración de las métricas o kpi, y, por otro lado, los gobiernos han puesto atención en instrumentos para conocer la opinión del ciudadano sobre los factores que producen su bienestar.

Entre los modelos más importantes están el de Boyd Cohen que en 2012 formó un comité compuesto por representantes de IBM, Smart City Expo, Smart Cities Council y el Center for Innovation in Cities, para medir a las ciudades, generando en el año 2014 un modelo de métricas con 6 kpi principales, con 18 indicadores en 62 subdivisiones [5]. De los 62 indicadores del modelo Cohen, 16 de ellos son mapeados a ISO 37120 [6] que abarca 17 kpis con 100 indicadores. El Global Power City evalúa y clasifica las mejores ciudades de acuerdo a 6 kpi que contienen 26 indicadores subdivididos en 70 [7]. Se han propuesto muchos otros modelos como el Índice de Desarrollo Humano [8] o el método de la huella ecológica que sirve para calcular los recursos que una ciudad consume y los residuos que produce.

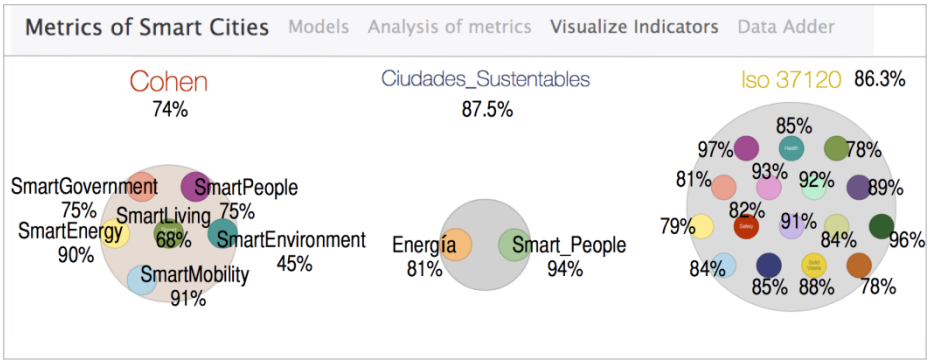

Fig. 1. Estructura general de la plataforma "Visualización de la herramienta Smart City".

En este documento se describe una herramienta que incorpora funciones de carga de datos (*Data Adder*), análisis y visualización de patrones (*Analisys of Metrics*) y un modelo (*Model*) que realiza el desplegado en modo árbol de cada indicador clave de desempeño o kpi con su fuente en la web. En este estudio se presentan los algoritmos que contemplan el almacenamiento de las mediciones para obtener una vista global de las condiciones de calidad de vida que asocian internamente cada kpi, lo que permite

la evaluación de una ciudad desde la perspectiva de diferentes modelos, conformando un modelo extendido. Este trabajo es parte de un proyecto global para medir la calidad de vida de la ciudad de Guadalajara, para el cual se ha desarrollado una herramienta de identificación de patrones entre los sectores territoriales basándose en el análisis de datos del Open Data y en una visualización geo-referenciada [9]. En la Fig. 1 se puede ver la parte gráfica de esta propuesta, aquí se muestra la visualización de kpis y sus valores para tres modelos: Cohen, Ciudades Sustentables e ISO 37120.

En las siguientes secciones se expone el método, su diseño e implementación con resultados desde la perspectiva del funcionamiento.

1.2. Método para el control de kpi (indicadores clave de desempeño)

Debido a la gran cantidad de variables que cada modelo implica, además de su tendencia al crecimiento, se observa la necesidad de una herramienta capaz de administrar sus subdivisiones o capas. Con ello posteriormente se podrá relacionar cada kpi hijo a un conjunto de datos, de tal manera que puedan ser analizados para obtener una medición tanto individual como general. El objetivo de este trabajo es por lo tanto, el de presentar un diseño completo con el desarrollo de algoritmos para la organización de kpis.

El proceso se realizó en tres fases: 1) primero se analizaron los diferentes modelos de métricas de ciudad; 2) posteriormente se examinaron las fuentes de datos locales en Open Data con el fin de identificar los formatos de registro en los archivos, para determinar el tipo de manejador de bases de datos apropiado (SQL o NoSQL), se seleccionó el Open Data por su formalidad y la veracidad en la información que se publica; 3) finalmente, en la tercera etapa se hizo el diseño y la implementación de algoritmos.

1.3. Formato JSON y mongodb

El formato JSON (*JavaScript Object Notation*) sirve para el intercambio y almacenamiento de datos y permite optimizar la velocidad de actualización y hacer archivos ligeros. Este formato utiliza convenciones familiares para los programadores, tales como, por ejemplo, llaves "{}" para agrupación y símbolos para anidar elementos u objetos que a su vez pueden contener a otros objetos [11]. Uno de los campos que más utiliza estos esquemas es el de los negocios, dado que muchas organizaciones buscan obtener ventaja del más mínimo dato. Aunque grandes volúmenes de datos se encuentran almacenados en sistemas diferentes, éstos pueden ser intercambiados integral y velozmente.

Comparado con XML, otro estándar común para el intercambio, el esquema JSON es más fácil de analizar gramáticamente, su formato para la definición de pares llave/valor requiere menor carga de bytes tanto para la transferencia entre sistemas, como para el almacenamiento y su indexación, JSON además posee mejor desempeño, incluye tipos de datos y arreglos, y es más veloz [12].

Aunque aún prevalece la necesidad de manejadores de bases de datos relacionales (RDB por sus siglas del inglés: *relational data base*) en los que existe una única llave para cada registro, con la normalización se decremento la redundancia reduciendo el

costo de almacenamiento [13]. Hoy en día se reconoce información sin una estructura definida en la que es más relevante el análisis científico, su velocidad de transferencia y facilidad de procesamiento, es decir manejadores de documentos JSON.

La aparición del NoSQL (originalmente referido como "*non* SQL" o "*non relational*") lenguaje de consulta para bases de datos no estructuradas, comenzó en el 2007 [14] , período en el cual las exigencias de las organizaciones se dirigen al tratamiento y análisis de grandes cantidades de datos y no a las relaciones normalizadas, estos datos son escasamente definidos en sus tipos, lo que dio a lugar al desarrollo de plataformas orientadas a documentos como mongodb que maneja el esquema JSON, tomando en cuenta la escalabilidad resolviendo el problema del aumento continuo de datos e incompatibilidad de formatos entre sistemas [15]. Es por esto, que en nuestro caso de aplicación, en el que se hace intercambio de documentos entre el sistema local y la web, y el volumen de kpis tiende al Big Data, se seleccionó el formato JSON.

2. Modelo en capas

Una vez hecho el análisis de los modelos aplicados para medir la sustentabilidad de las ciudades, la calidad de vida, el desarrollo humano o la capacidad de producir lo que consume, se observó que estos suelen ser subdivididos para expresar mejor precisión en la medida, se identificó una estructura anidada y la formación de grupos en niveles, que se denominaron capas.

También se encontró que en Open Data no existe un estándar en la definición de los tipos de kpi, ni en sus rangos ni en las unidades de medida, esto es, sin estructura con la forma key-value.

2.1. Interfaz

Un primer prototipo para la implementación de navegación en capas se presentó en [1], en donde la interfaz fue inicialmente orientada al almacenamiento de fuentes de los datos (Fig. 2).

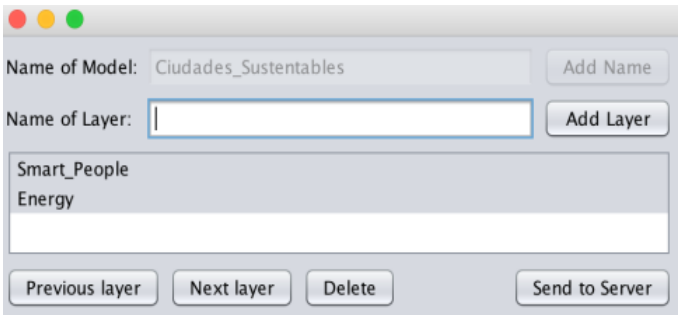

Fig. 2. Interfaz básica para la captura de capas (kpis) extraída de [9].

Dicha interfaz es aquí extendida en cuanto a su funcionalidad, mediante los elementos que se describen en la Tabla 1.

Tabla 1. Funciones de elementos de pantalla.

Elemento	Tipo	Acción
Nombre del Modelo	Caja de Texto	Captura del nombre con el que se identifica al modelo de métricas urbanas
Agregar Nombre	Botón	Guarda en estructura temporal el Nombre del Modelo capturado
Nombre de la Capa	Caja de Texto	Nombre del kpi que se desea anidar
Agregar Capa	Botón	Guarda en estructura temporal el Nombre del kpi anidado en la capa activa con el padre y valor inicializado en null
Capa Anterior	Botón	Regresar a un nivel del modelo
Capa Siguiente	Botón	Navegar a un nivel superior
Eliminar	Botón	Eliminar una capa y sus kpis hijos
Enviar a Servidor	Botón	Convertir a JSON y enviar a servidor mongodb

2.2. Navegador en capas

Para un monitoreo inteligente, por ejemplo, en consumo de energía para determinar la demanda de recursos, el proceso a seguir en esta herramienta es mediante los siguientes pasos: a) identificar el modelo de métricas de ciudades sustentables y agregarlo, b) si existen, agregar las capas o kpis, y c) agregar los indicadores asociados y anidarlos al nivel o capa correspondiente. En la Fig. 3 se hace una comparación de dos modelos: uno internacional, el ISO37120, y uno local nacional mexicano. Se debe hacer un mapeo o selección de los kpis que se desean mantener de cada modelo.

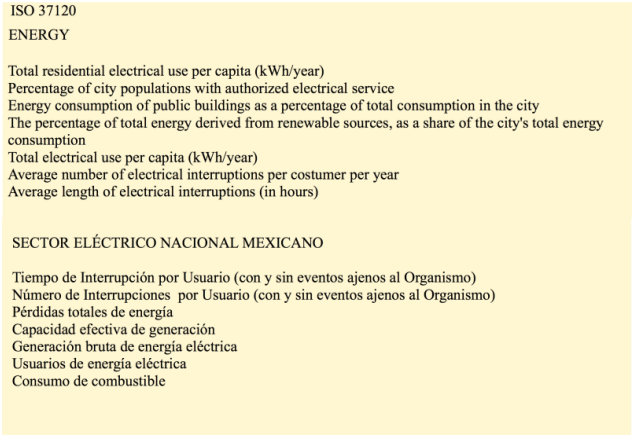

Fig. 3. Listados para kpis de Energía del estándar internacional ISO37120 y del modelo nacional mexicano [10]**.**

En la Fig. 4 se puede ver la pantalla de captura. En este ejemplo se capturó el nombre de un kpi de cada modelo, es por esto que aparecen ambos en el mismo nivel. Al navegar por la Capa Anterior aparecerá el nombre de la capa en la que se han anidado.

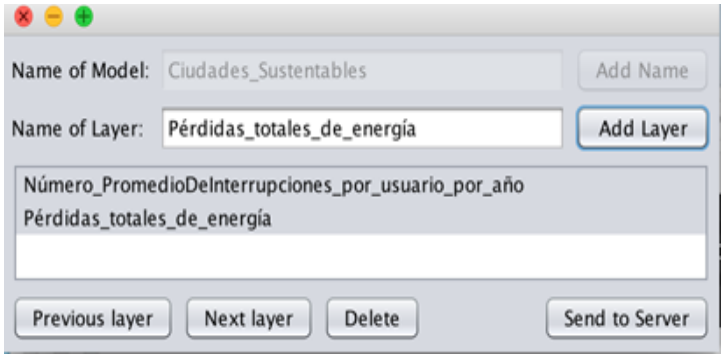

Fig. 4. Dos kpi's añadidos a la rama padre Pérdidas_totales_de_energía.

2.3. Estructura de documentos con kpis

Al ser enviadas las capas al Servidor se generan cuatro documentos: uno para la identificación del nombre del modelo, el cual tiene una variable value con valor null y que queda disponible para almacenar en su momento la medida de porcentaje de Sustentabilidad de una ciudad, a medida que se actualicen los valores de sus diferentes capas (Fig. 5).

En este caso el id Energía tiene un documento padre Ciudades_Sustentables, ver Fig. 6. Al igual que el anterior, para cada documento la variable value inicializada en null, variable servirá para la colocación de la medida de evaluación de Energía cuando sus subcapas o documentos hijos se hayan evaluado. En la Fig. 7 se muestran dos documentos de kpis pertenecientes a Energía, pero ya dentro del modelo Ciudades_Sustentables.

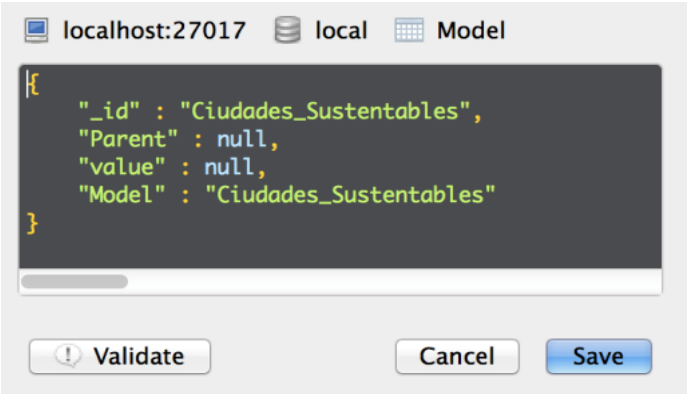

Fig. 5. Objeto JSON para la primera capa del modelo de Ciudades_Sustentables.

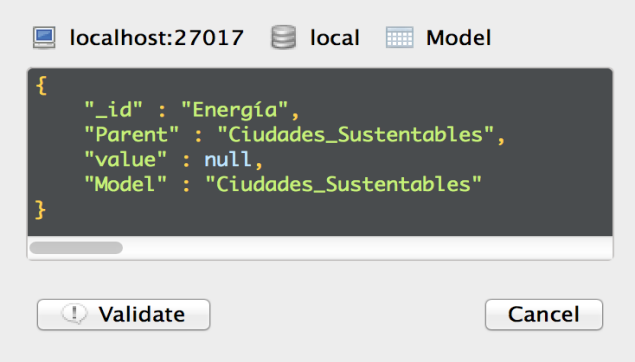

Fig. 6. Objeto JSON para la segunda capa, kpi Energía asociada a la rama padre Ciudades_Sustentables.

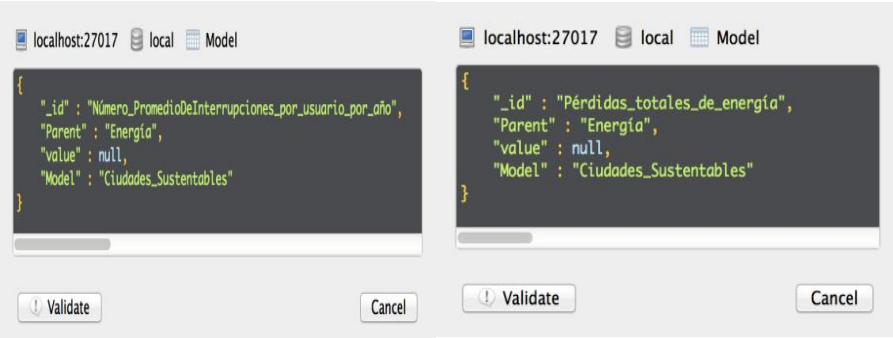

Fig. 7. Dos objetos JSON para la tercer capa, cada indicador asociado a la rama padre Energía.

Veamos otro ejemplo con la captura de la capa Smart People e indicadores clave de desempeño de la subcapa Educación, en la que previamente se fusionaron varios modelos de métricas entre ellos Cohen e ISO37120.

Las métricas de la calidad de la educación PISA aportan indicadores clave de desempeño internacional; además los indicadores clave de desempeño nacionales incluyen la infraestructura y cobertura.

De modo que Educación es evaluada en torno a cuatro aspectos: 1) alumnos en nivel licenciatura y la calidad de universidades a las que asisten, 2) alumnos en nivel de preparatoria y calidad de escuelas preparatorias, 3) alumnos en nivel de escuelas secundaria y calidad de escuelas secundarias, y 4) alumnos en nivel de escuelas primarias y calidad de escuelas primarias. En la Fig. 8 se muestra un árbol con los kpsi considerados, cada uno en sus ramas.

En la Fig. 9 se muestran dos pantallas, de una capa y la subsiguiente, con la interacción de la interfaz para captura del árbol de indicadores de Educación en la rama Smart People, Ciudades_Sustentables, un modelo híbrido ya que mezcla diversos índices, de Educación desde el modelo Cohen, ISO37120, PISA e indicadores de desempeño nacionales.

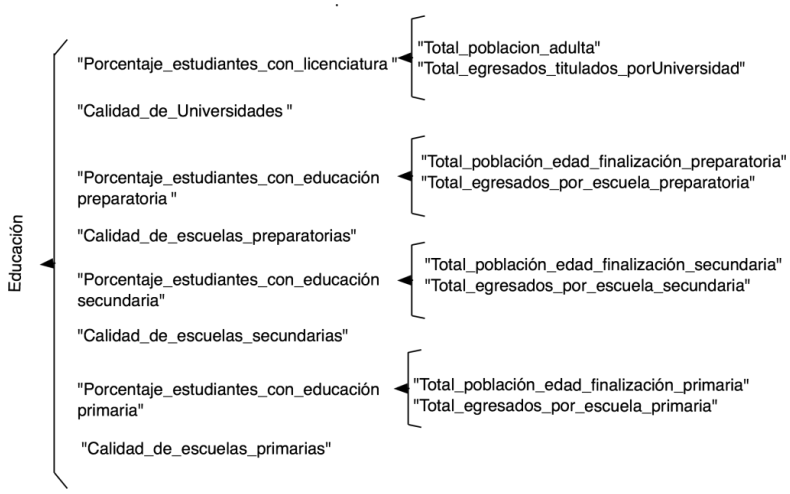

Fig. 8. Kpis para la medida de evaluación de Educación combinando indicadores de Educación de ISO37120 e indicadores de Educación de INEGI.

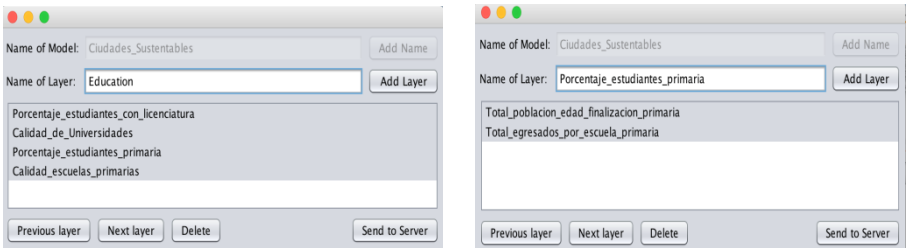

Fig. 9. Captura de kpis para la capa Educación y para su rama Porcentaje_estudiantes_primaria combinando indicadores de ISO37120 e INEGI.

Los documentos JSON resultantes almacenados en mongodb se muestran en la Fig. 10 en tres capas subsecuentes.

2.4. Algoritmos del navegador de capas en el árbol de kpis

El algoritmo para la función Capa Siguiente asigna como padre al kpi seleccionado en la interfaz de la pantalla y actualiza los datos sobre esta (Fig. 11).

```
private void btnNextActionPerformed(java.awt.event.ActionEvent evt) {
    if (listKpiTree.getSelectedIndex() != -1) {
        //Get the actual Kpi
        actualParentKpi = listKpiTree.getSelectedValue();
        //Update the gui
        updateGui();
    }
}
```

Fig. 10. Algoritmo de la función Capa Siguiente.

El botón Capa Anterior busca el padre del kpi seleccionado y extrae el id correspondiente, dichas actividades se ejecutan por el método `btnBackActionPerformed` (Fig. 12), que realiza dos acciones retornar al padre (`returnTree`) y actualizar la interfaz (`updateGui`). El código de `returnTree` dispara a `getParent` con parámetro `actualParentKpi`, quien a su vez busca al el padre de la capa que en la interfaz ha sido seleccionada. `getParent` itera en `kpiList` comparando el hijo o parámetro con alguno de la columna 1 que corresponde al hijo, retornando por ello la columna 0 que corresponde al padre, si no lo encuentra entonces se trata de la raíz cuyo padre es él mismo y es cabecera de la estructura.

```
private void btnBackActionPerformed(java.awt.event.ActionEvent evt) {
    //Return the tree
    returnTree();
    //Update the gui
    updateGui();
}
                private void returnTree() {
                    //Get the actual parent of the actual kpi
                    actualParentKpi = getParent(actualParentKpi);
                }
                private String getParent(String childString) {
                    for (String[] iterateArray : kpiList) {
                        if (iterateArray[1].equals(childString)) {
                            return iterateArray[0];
                        }
                    }
                    return headTree;
                }
```

Fig. 11. Botón Capa Anterior llamando returnTree() quien dispara a getParent.

El algoritmo para la función Enviar a Servidor (Fig. 13), llama al método `convertDocument()` que toma la lista de kpis tipo árbol y los retorna en forma de cadena de caracteres en la estructura de documentos JSON e inserta cada uno mediante el ciclo for en la colección de objetos de mongodb por la conexión realizada en otro método.

```
private void btnSendServerActionPerformed(java.awt.event.ActionEvent evt) {
    for (String temporalString : convertDocument().split("\\n")) {
        DBObject dbObject = (DBObject) JSON.parse(temporalString);
        collection.insert(dbObject);
    }
}
```

Fig. 12. Función Enviar a Servidor.

El algoritmo del método `convertDocument` (Fig. 14) tiene dos actividades, la primera es generar el documento JSON para el objeto raíz, el cual no posee padre, el

id es el nombre del modelo guardado en la variable `headTree`, se asigna a la variable value el valor null reservada para futuro almacenamiento de la evaluación final global.

La segunda actividad es generar los documentos de cada kpi dentro de todas las anidadas. Itera en el contenido de la lista, agregando el id del padre que le pertenece a cada uno y que se encuentra en la misma lista. Añade el nombre del modelo que es `headTree` y la variable value con valor en `null`.

El formato de dos columnas de `kpiList` permite almacenar en la posición 0 el padre, y en la posición 1 el nombre del indicador.

```
private String convertDocument() {
    //A stringbuilder to generate the values
    StringBuilder stringBuilder = new StringBuilder();
    //Append the head
    stringBuilder.append("{'_id' : '").append(headTree).append("','Parent' : null,'value' : null,'Model' :
'").append(headTree).append("'}");
    stringBuilder.append("\n");
    //Iterate the tree
    kpiList.stream().forEach((iterateArray) -> {
      stringBuilder.append("{'_id' : '").append(iterateArray[1]).append("','Parent' :
'").append(iterateArray[0]).append("','value' : null,'Model' : '").append(headTree).append("'}");
      stringBuilder.append("\n");
    });
```

Fig. 13. Método convertDocument**.**

2.5. Interfaz para el modelo de métricas

El esquema general de control de kpis de ciudad debe comprender además la interfaz y funciones para fundamentar cada modelo, con él es posible para el usuario establecer el alcance de las métricas, el alcance de los resultados a obtener al aplicar el modelo, y además producir un marco de referencia que conduzca a la elección de un modelo adecuado a un contexto de ciudad, para ello se propone un diseño preliminar de interfaz que complementa a tal esquema (Fig. 15).

Las funciones principales se concentran en Crear nuevo modelo, Editar y Borrar (New, Edit y Delete), con las opciones para modelos concretos como *Smart Cities*, ISO37120 y modelos de contexto o híbridos que mezclan kpis de distintos autores. En esta interfaz se realiza así la especificación de la fuente del modelo, como autor, año y referencia bibliográfica, se especifica su tipo (reporte, estándar, cuestionario o índice), y la meta, perspectiva y sujeto para especificar la perspectiva desde el cual deberán ser interpretados los resultados.

El elemento `KeyPerformanceIndicators` enlaza a la interfaz de captura para el manejo de capas en la perspectiva de árbol, que es el prototipo funcional expuesto ampliamente en este estudio.

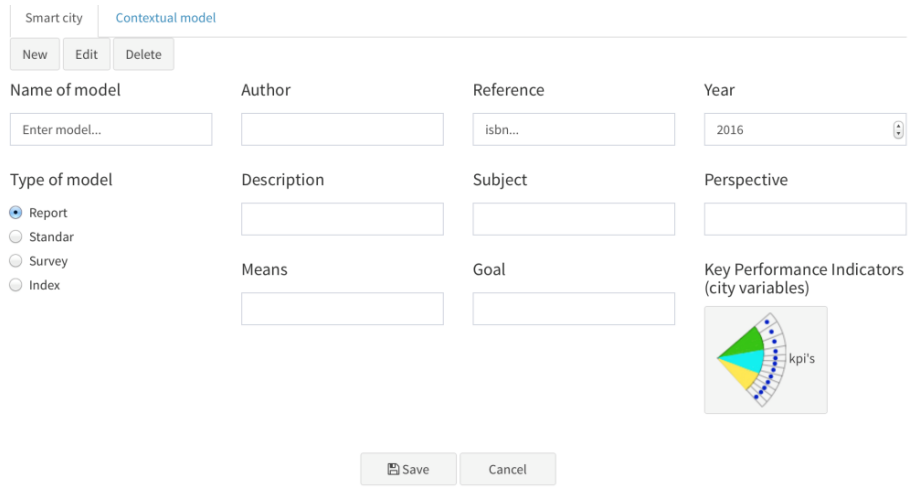

Fig. 14. Interfaz para el Modelo de Métricas.

3. Conclusiones y trabajo futuro

Debido a la gran cantidad de variables que manejan los modelos Smart City es necesaria una herramienta que nos permita mostrar los diferentes indicadores clave de desempeño o kpi de manera ordenada y lógica. En este documento se propone un manejo por capas tipo árbol para los diferentes kpi que maneja un modelo. La herramienta permite la visualización de diferentes modelos para facilitar su comparación. La determinación del modelo es fundamental para conocer mejor el estado de una ciudad.

El método presentado y sus algoritmos permiten además crear un modelo propio para compararlos con los ya existentes o con otros modelos creados.

Como trabajo a futuro se desarrollarán los algoritmos para la plataforma integral "visualization of the Smart City tool", entre ellos se encuentran principalmente los de despacho de recursos y servicios en modo inteligente, que deberán incorporar funciones de monitoreo inteligente basándose en el análisis científico de kpis capturados, para obtener no sólo la evaluación de una ciudad sino también reflejar y ejecutar las acciones para impactar positivamente en la huella ecológica y la sustentabilidad.

Referencias

1. Chourabi, H., Nam, T., Gil-Garcia, J.R., Mellouli, S., Pardo, T.A., Scholl, H.: Understanding Smart Cities: An Integrative Framework. In: 45th Hawaii International Conference on System Sciences, pp. 1–9 (2011)
2. Townsend, A.M.: Smart Cities (2013)
3. ITU: General specifications and KPIs. Pp. 1–34 (2012)
4. Moonen, T., Clark, G.: The Business of Cities 2013. Jones Lang Lasalle IP, INC (2013)

5. Cohen, B.: Boyd Cohen, https://www.smart-circle.org/smartcity/blog/boyd-cohen-the-smart-city-wheel/
6. 37120, S.I.D.: ISO37120
7. Institute for Urban Strategies the Mori Memorial Foundation: Global Power City 2017. (2017)
8. Eclogical Footprint. 1–114 (2016).
9. Estrada, E., Maciel, R., Ochoa, A., Bernabe-Loranca, B., Oliva, D., Larios, V.: Smart City Visualization Tool for the Open Data Georeferenced Analysis Utilizing Machine Learning. International Journal of Combinatorial Optimization Problems and Informatics. 9, pp. 25–40 (2018)
10. SENER ed: Sector Eléctrico Nacional. SENER.
11. International, E. ed: Introducing JSON. ecma international.
12. Gaurav, G., Karanjit, S., Ramkumar, K.R.: A detailed analysis of data consistency concepts in data exchange formats (JSON & XML) Presented at the March 21 (2017)
13. Codd, E.F.: A Relational Model of Data for LArge Shared Data Banks. Presented at the June 15 (1970)
14. Leavitt, N.: Will NoSQL Databases Live Up to Their Promise? (2010)
15. Inc, M.: MongoDB nosql document database.