INFORMATION MODEL FOR CITY MONITORING BY SMART LIGHTING

Mila Ilieva-Obretenova

University of Mining and Geology "St. Ivan Rilski", 1700 Sofia, milailieva@abv.bg

ABSTRACT. City monitoring includes a variety of services like informing for parking places, pedestrian planning, situational awareness (access to photos, audio, video), environmental planning, traffic planning. This could be achieved by improving the Smart Lighting technology. Smart Lighting was designed for energy efficiency. The new aim is safety for the citizens. It is reached by embedding different sensors and cameras in street lighting.
This paper offers an information model for city monitoring through Smart Lighting. The model is designed for user interface developers, city authorities, and citizens. The hypothesis is as follows: the description of the objects containing the data should be represented as a software model. It is sufficient that the software of each component be detailed with elements up to the level of service logic. The model should possess the following features: The components containing the data are represented as physical entities. For each component, software is represented as a set of functions. Functions are grouped according to the areas of open system interconnection (OSI): configuration, security, maintenance, accounting, and performance. The focus is on the configuration area because it contains the data on the execution of the service logic. Software components in the configuration area are detailed up to attributes and operations that direct the logic to the next element of its execution. Design methodology for the information model includes defining of classes of managed objects for service components. The result is an information model design for city monitoring by Smart Lighting. This is a model of management services and their components. The object-oriented method is used. The classes of managed Objects are defined according to managed units. The Guidelines for the Definition of Managed Objects (GDMO) from the Network Management Standards are followed. The UML (Unified Modeling Language) is used for the model description. On this stage, objects are represented with names, "Part of"- relationships, associations, attributes, and operations. On the next stage, new attributes and operations will be added to the objects managed. With this level of defining, the model is a good base for the development of user interface. The model is also applicable for monitoring the processes in an underground mine. Lighting with sensors and cameras could be applied there and the same services could be provided: employee planning, collecting information for a specific location, environmental planning, retrieving event information.

Keywords: Information model, city monitoring, Smart Lighting, safety for the citizens

ИНФОРМАЦИОНЕН МОДЕЛ ЗА НАБЛЮДЕНИЕ НА ГРАД ЧРЕЗ УМНО УЛИЧНО ОСВЕТЛЕНИЕ (SMART LIGHTING)

Мила Илиева-Обретенова

Минно-геоложки университет "Св. Иван Рилски", 1700 София, milailieva@abv.bg

РЕЗЮМЕ. Наблюдението на един град включва разнообразие от услуги като информиране за свободни паркоместа, насочване на пешеходци, информация за дадено място (достъп до снимки, видео), наблюдение на параметрите на околната среда, информиране за трафика. Това може да се постигне чрез усъвършенстване на технологията Smart Lighting. Първоначалната цел на Smart Lighting е постигане на енергийна ефективност. Новата цел е сигурност за гражданите. Тя се постига чрез вграждане на различни сензори и камери в уличните лампи.
Статията предлага информационен модел за наблюдение на град чрез Smart Lighting. Моделът е предвиден за разработчици на потребителски интерфейс, органите на реда и гражданите. Хипотезата е следната: описанието на обектите с данните трябва да представя модел на софтуер. Софтуерът на всеки компонент е достатъчно да се детайлизира с елементи до ниво логика на услуга. Моделът трябва да притежава следните качества: Компонентите с данните се представят като физически единици. За всеки компонент се представя софтуер като множество от функции. Функциите се групират в съответствие с областите за взаимодействие на отворени системи (OSI): конфигурация, защита, поддържане, таксуване и технически характеристики. Фокусираме върху област конфигурация, защото тя съдържа данните за изпълнението на логиката на услугите. Софтуерните компоненти в област конфигурация се детайлизират до атрибути и операции, които насочват логиката към следващия елемент при нейното изпълнение. Методологията за проектиране на информационния модел включва дефиниране на класове управлявани обекти за компонентите на услугите. Резултатът е проект на информационен модел за наблюдение на град чрез Smart Lighting. Това е модел на услугите за управление и техните компоненти. Използван е обектно-ориентиран метод. Класовете управлявани обекти са дефинирани в съответствие с управляваните единици. Следвани са Препоръки за дефиниране на управлявани обекти (GDMO) от стандартите за управление на мрежи. За описание на модела е използван UML (Унифициран език за моделиране). На този етап компонентите на услугите за управление са представени чрез имена, отношения „Част от", асоциации, атрибути и операции. На следващия етап ще се добавят нови атрибути и операции към управляваните обекти. С това ниво на дефиниране моделът представлява добра основа за разработване на потребителски интерфейс. Моделът е приложим и за наблюдение на процесите в подземна мина, където могат да се приложат лампи със сензори и камери, и могат да се предложат аналогични услуги като насочване на служителите, събиране на информация за дадено място, наблюдение на параметрите на средата, информиране за събития.

Ключови думи: информационен модел, наблюдение на град, Smart Lighting, сигурност за гражданите

Introduction

City monitoring includes a variety of services, like informing for parking places, pedestrian planning, situational awareness (access to photos, audio, video, and other media), environmental planning, traffic planning, etc. This could be achieved by improving Smart Lighting technology. Initially, Smart Lighting was designed for energy efficiency. The new aim is safety for the citizens. It is achieved by embedding different sensors and cameras in street lighting (Velinov, 2018). Nowadays, a few cameras are located in big cities in Bulgaria: Sofia "Orlov most", Sofia "Serdika" Underground station, Plovdiv "International Fair", Varna "Hristo Botev Boulevard", Bourgas "Stefan Stambolov Boulevard", Berkovitsa

"City center Berkovitsa", Pamporovo "Ski track Snezhanka", Pamporovo "Ski track Studenets", and Chepelare "Mechi chal" (https://nova.bg/camera , 2018). Smart Lighting services offer a lot of details for the users (https://developer.currentbyge.com/cityiq , 2018).

The Parking Planning service provides parking metadata that is collected from intelligent nodes installed on all street lights along public roadways or in parking lots. Each time a vehicle enters or leaves a parking space, a timestamped event is created and stored in the cloud. This data can be obtained from a single sensor or a group of sensors and can provide valuable insights about parking activity in a given region. The data can also be used to improve traffic and parking convenience.

The Pedestrian Planning service provides pedestrian metadata that is collected from intelligent nodes installed on all street lights along public roadways or in parking lots. As pedestrians pass by these sensors, the service creates and stores a timestamped event, counting each event. This data can be obtained from a single sensor or a group of sensors, to provide insights to city officials and local businesses about pedestrian movement in a given region. The data can also be used to enhance safety and convenience.

The Situational Awareness service provides access to images, video, and other media collected from intelligent nodes installed on all street lights along public roadways. You can retrieve media on demand (streaming data) or historical media data. Examples of use cases for these images and videos include enhancing situational awareness, improving traffic flow, and verifying where vehicles are parked. The media can be requested using a date range beginning with near real-time and ending based upon data availability. This data can be obtained on-demand directly from a single sensor or node, as needed, to provide insights to city officials that enhance awareness. This service can be used in concern with other "Intelligent Cities" services to address a variety of use cases.

The Environmental Planning service provides environmental sensor data collected from intelligent nodes installed on all street lights along public roadways. This data can be obtained from a single sensor or a group of sensors, as needed, to provide timely environmental information to city officials that directly affect urban dwellers and their activities.

The Traffic Planning service provides vehicle metadata collected from intelligent nodes installed on all street lights along public roadways. As vehicles pass by these sensors, information such as speed, direction lane use, and volume (counts) are collected and stored with a time-stamp in the cloud. The data can be viewed from any number of nodes in a given region and can present both historical and near real-time information. The density, range, and flexibility of this sensor data can provide insights into traffic patterns at different intervals. Along with facilitating the planning for future safety and convenience enhancements, the data can reveal opportunities to optimise traffic flows.

Scope

The paper aims to offer an information model for city monitoring by Smart Lighting. The model is designed for user interface developers, city authorities, and citizens.

Thesis

The model should include descriptions of management services in Smart Lighting and their components.

Hypothesis

The description of the management services in Smart Lighting and their components should include software definitions. Each component`s software is sufficiently defined with elements of the level of managed object. The model should possess the following features: management service components are represented as physical entities; software as a set of functions is represented for each component; functions are grouped according to OSI (ISO/IEC/IS, 1989) areas: configuration, security, maintenance, accounting, and performance; focusing on a configuration area because it contains the data for service logic execution; software components in a configuration area are detailed up to attributes and operations that direct the logic to the next element of its execution.

Methodology

The methodology for the design of an information model for city monitoring by Smart Lighting includes definitions of managed objects classes for services provided to subscribers and providers. Definitions are represented verbally and by UML (Unified Modelling Language) diagrams (Fowler, 2004; Gentleware, 2018). UML diagrams are classified into two types: behavior diagrams and structure diagrams. Class diagrams, a type of structure diagrams, are appropriate for the description of city monitoring management. A class diagram describes types of objects in the system and the different kind of static relationships between them. Diagrams also show "Part of"-relationships, features, and operations of classes and the limits of the way in which the objects are connected. The "Part of"-relationship is shown with a rhombus and a line. Features are one term, but they are represented with two quite different notations: attributes and associations. The notation for an attribute describes a distinct feature like a text (second row) in a rectangle, envisaged for a class. The association is a directed line between two classes and its direction is from class-source to class aim. A feature's name is set on the aimed end of the association with its majority. The end-aim of the association is connected to the class which is the feature's type. The majority of a feature is a note for how many objects could complete the feature. Operations are actions which a class could realise. Obviously, they correspond to the methods of a class. There is a distinction between Operation and Method, however. The operation is the name used for a method – a declaration of a procedure. The method is a code which the procedure consists of. Operation and Method are different by polymorphism. The model has the impact of a similar model for telecommunications management (Magedanz, 1994).

Results

In this section, the designed model for management in Smart Lighting is represented: a model of management services and their components. The object-oriented method is used. The classes of managed objects are defined according to managed units. The Guidelines for the Definition of Managed Objects (GDMO) (ISO/IEC/IS, 1989) from the Network Management Standards are followed. The UML is used for model description. At this stage, objects are represented with names, "Part of"-relationships, associations, attributes, and operations.

Managed Objects classes for service configuration in Smart Lighting

Managed Object (MO) **SConfiguration** represents a service configuration needed for the provider or tailored on special order for a subscriber. MO **SComponent** contains all parts of a service which must be distributed on the network infrastructure. MO **Template** represents the description of the entry format for each service in a Smart Lighting network. MO **Resource** represents the description of a specialised resource (camera, sensor) used by a specific service. MO **Program** represents the description of a program with service logic. MO **TriggerInfo** represents the description of a trigger needed for service execution direction. MO **SDFData (Service Data Function Data)** represents an entry for a specific service subscriber. MO **SubsConfiguration** represents the subscriber's requirements for the configuration of the provided services. MO **ParkPlanning** contains the description of the Parking Planning service. MO **PedPlanning** contains the description of the Pedestrian Planning service. MO **GetMedia** contains the description of the Situational Awareness service. MO **EnvPlanning** contains the description of the Environmental Planning service. MO **TrafficPlanning** contains the description of the Traffic Planning service. Fig. 1 shows the UML diagram of Managed Objects classes for service configuration in Smart Lighting.

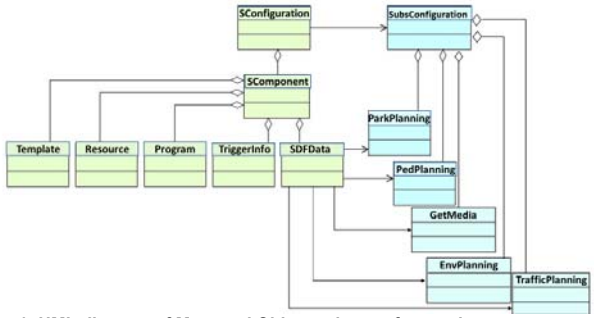

Fig. 1: UML diagram of Managed Objects classes for service configuration in Smart Lighting

Attributes and operations of Managed Objects classes for service configuration in Smart Lighting

MO **Resource** has the following attributes:
Attribute **Camera** describes the unique Identifier for each camera. Attribute **Sensor** describes the unique Identifier for each sensor.

MO **Program** has the following attributes:
Attribute **AssetUID** describes the Identifier for a unique asset. This is a key attribute by which information query for a specific asset is performed. Attribute **LocationUID** describes the Identifier for the unique location in an observed area. This is a key attribute by which an event query in a specific area is performed. Attribute **EventType** describes the Identifier of type events. This is a key attribute by which a pedestrian query by number, direction, and speed is performed. Attribute **TimeStamp** describes the actual data and time when the event occurs. Attribute **GeoCoordinates** describes the GPS coordinates of the four angles of the parked vehicle. Attribute **ObjectID** describes the unique Identifier (plate number) of a specific vehicle. Attribute **VehicleType** describes the type of specific vehicle i.e. vehicle, truck, bus, building machine, agricultural machine. Attribute **DirectionUnit** describes the measurement unit for pedestrian direction, i.e. degree. Attribute **SpeedUnit** describes the measurement unit for pedestrian speed, i.e. m/s. Attribute **Unit** describes the unit, by which specific magnitude is measured, i.e. K, HPa, south-southwest, %. Attribute **PowerOf10** describes the factor of 10, i.e. -1 (10^{-1}). The measure in K multiplies with 10^{-1}, i. e. 2700 x 10^{-1} K=270 K, 270 K-273 15=- 3,15° C. Attribute **ImagePollUrl** describes the link which shows photos captured from a specific device. Attribute **VideoPollUrl** describes the link which shows video captured from a specific device. Attribute **Status** describes the status in response to a request for video or photos. The status could be INPROGRESS when the request is in process or SUCCESS when the request is completed. Attribute **MediaID** describes the Identifier used for a specific type media or the Identifier for a specific node.

Attribute **MediaType** describes a specific media type: IMAGE provides a list of images in JPG, PNG, and GIF formats. Depending on the position of the camera, it will return the best resolution image available: either 320x240 for black and white or 1920x1080 for color; VIDEO provides a list of videos in H.264 format. Attribute **MediaLogID** describes the Identifier for each entry in the list. Attribute **MediaFileName** describes the file name of the captured media. Attribute **MediaTimeStamp** describes the time when the media was captured from the node. Attribute **ExternalRefID** describes the Identifier of an external pointer. Attribute **EntryTimeStamp** describes the time when the captured media was entered in the database. Attribute **Size** describes the maximum number of records to return per page; if none specified, the default value of 2 is used automatically. Attribute **Number** indicates the page number, the default is 0.

MO **Program** has the following operations:
Operation **Counter_direction** calls the procedure for computing of a pedestrian's direction in interval west, northwest, north, northeast or east direction (270° - 0° - 90). Operation **Counter_dir_pedCount** calls the procedure for counting the serial number of a pedestrian who moved in a specific direction (the same as in Counter_direction). Operation **Counter_dir_speed** calls the procedure for computing the average speed of a pedestrian in a specific direction (the same as in Counter_direction). Operation **Direction** calls the procedure for computing a pedestrian's direction in interval west, southwest, south, southeast, and east direction (270° - 180° -90°). The procedure calculates the direction opposite the one in Counter_direction. Operation **PedCount** calls the procedure for counting the serial number of a pedestrian who moved in the direction opposite the one in Counter_direction. Operation **Speed** calls the procedure for computing the

average speed of a pedestrian opposite the one in Counter_direction. Operation **Counter_dir_vehicleCount** calls the procedure for counting the serial number of vehicles in a specific direction. (As with the pedestrians, the movement is calculated in two directions, the serial number of the vehicles in a specific direction is counted, and the average speed in this direction is calculated.) Operation **VehicleCount** calls the procedure for counting the serial number of vehicles in the direction opposite the one in Counter_dir_VehicleCount.

Operation **Max** calls the procedure for determining the maximum value of a specific magnitude for a defined period, i.e. humidity, wind direction, pressure, and temperature. Operation **Mean** calls the procedure for computing the mean value for a defined number of digits. Operation **Median** calls the procedure for determining the median in a defined interval of values. (The median is the value separating the higher half of a data sample from the lower half.) Operation **Min** calls the procedure for determining the minimum value of a specific magnitude for a defined period. Operation **Sort** calls the procedure for sorting values of attributes: Asset UID, Media File Name, Media Time Stamp, Media Type. Operation **Duration** calls the procedure for counting the duration of an event. Operation **SortDir** calls the procedure for sorting in an ascending or a descending order. Operation **Ignore Case** calls the procedure for determining whether the request is interrupted or not (for service Get Media). Operation **getAssetUIDrtd** calls the procedure for capturing different events registered by a unique device in real time. Operation **getLocationUIDrtd** calls the procedure for capturing different events for a unique location in real time. Operation **getAssetUIDhd** calls the procedure for capturing different events registered by a unique device as historical data. Operation **getLocationUIDhd** calls the procedure for capturing different events for a unique location as historical data.

MO **TriggerInfo** has the following attributes:
Attribute **Pkin** indicates the event of parking enter. Attribute **Pkout** indicates the event of parking escape. Attribute **Pedevt** indicates a pedestrian appearance. Attribute **Tfevt** indicates the event of traffic appearance (vehicle). Attribute **Humidity** indicates the event of humidity counting in an exact moment. Attribute **Orientation** indicates the event wind orientation counting in an exact moment. Attribute **Pressure** indicates the event of pressure counting in exact moment. Attribute **Temperature** indicates the event of temperature counting in an exact moment. Attribute **Click** indicates the event of click on a link.

MO **SConfiguration**, MO **SComponent**, MO **Template**, MO **SDFData**, MO **SubsConfiguration**, MO **ParkPlanning**, MO **PedPlanning**, MO **TrafficPlanning**, MO **EnvPlanning,** and MO **GetMedia** are described only by name and number.
Table 1 shows theattributes of MO **Resource.** Table 2 shows the attributes and operations of MO **Program.** Table 3 shows the attributes of MO **TriggerInfo.**

Table 1.
Attributes of MO Resource

Resource
Camera[1] : String Sensor[1] : String

Table 2.
Attributes and operations of MO Program

Program
AssetUID[1] : String LocationUID[1] : String EventType[8] : String TimeStamp[1] : String GeoCoordinates[4] : List ObjectID[1] : String VehicleType[5] : String DirectionUnit[1] : String SpeedUnit[1] : String Unit[4] : String PowerOf10[1] : String ImagePollUrl[1] : String VideoPollUrl[1] : String Status[2] : String MediaID[1] : String MediaType[2] : String MediaLogID[1] : String MediaFileName[1] : String MediaTimeStamp[1] : String Size[0..1] : int Number[1] : int
Counter_direction(…): void Counter_dir_pedCount(…): void Counter_dir_speed(…): void Direction(…): void PedCount(…): void Speed(…): void Counter_dir_vehicleCount(…): void VehicleCount(…): void Max(…): void Mean (…): void Median(…): void Min(…): void Sort(…): void Duration(…): void SortDir(…): void IgnoreCase(…): void getAssetUIDrtd(AssetUID, EventType): EventType1, EventType2 getLocationUIDrtd(LocationUID, EventType): EventType1, EventType2 getAssetUIDhd(AssetUID, EventType, StartTime, EndTime): void getLocationUIDhd(LocationUID, EventType, StartTime, EndTime, size, page): void

Table 3.
Attributes of MO TriggerInfo

TriggerInfo
Pkin[1] : String Pkout[1] : String Pedevt[1] : String Tfevt[1] : String Humidity[1] : String Orientation[1] : String Pressure[1] : String Temperature[1] : String Click[1] : String

The chosen granularity for the description of the managed objects gives an idea of the work volume which should be completed in the development phase. The proposed model represents an abstract description of service management in Smart Lighting. Only the functional area Configuration is considered here. In comparison with other models, i.e. for telecommunications management, attributes and operations for the managed objects are also listed here. The disadvantage could be found in the limited number of details.

Conclusion

The paper represents the design of an information model for city monitoring by Smart Lighting. The model corresponds to the responsibilities of the Service Subscriber and Service Provider actors. The classes of Managed Objects are defined which represent the managed resources for the network elements following the object-oriented method. The classes of Managed Objects are organised in a hierarchy which shows the correlation between them and relates to easier implementation. At this stage, the software for management services is detailed with elements up to the level service components. The chosen classes of Managed Objects are defined by name, "Part of"-relationships, associations, attributes, and operations. At the next stage, new elements could be added to the functional area Configuration and details could be developed for other functional areas. Nevertheless, the model is a good basis for prototype development of user interface.

The model is also applicable for monitoring the processes in an underground mine where lighting with sensors and cameras could be installed and where similar services could be provided, like employee planning, information capturing for specific location, environmental planning, events information, etc.

References

Fowler, M. UML Distilled: A Brief Guide to the Standard Object Modeling Language. 3rd ed. Addison-Wesley Professional, 2004.

Gentleware AG, Poseidon for UML CE 8.0, www.gentleware.com (accessed 2018).

https://nova.bg/camera (accessed 2018).

ISO/IEC/IS 7498-4 CCITT Recommendation X 700: Information Processing – Open Systems Interconnection – Basic Reference Model – Part 4: Management Framework, 1989. - 19-30.

Magedanz, T. An integrated management model for intelligent networks, München, Wien: Oldenburg, 1994.

Redefining the future, https://developer.currentbyge.com/cityiq (accessed 2018);

Velinov, K. et al., Web-based Database for Luminaries and Application in Street Lighting, Proceedings, VII Balkan Conference on Lighting BalkanLight 2018, Sofia, 2018. - 201-204.