Petri Net-based Smart Parking Information System

Omar Makke and Oleg Gusikhin
Global Data Insight And Analytics, Ford Motor Company, 22001 Michigan Ave, Dearborn, Michigan, U.S.A.

Keywords: Smart Parking, IoT, Petri Nets, Mobility, Smart City, Digital Twin.

Abstract: In this paper, we propose a Petri Net digital twin solution for smart parking information system to track the occupancy of a parking space while respecting the privacy of the drivers. An edge computing device is deployed to process camera images, and a Petri Net model is generated from the event logs and tracks the occupancy of the parking structure. This type of solution can be enhanced to any desirable level of accuracy. The paper provides preliminary analytics for the parking dynamics in a period of three months. This analysis clearly demonstrates the tangible benefits of the parking information system.

1 BACKGROUND

In the of 2018, the Ford Global Data Insight and Analytics (GDIA) team moved to the new Wagner Place building in downtown Dearborn. Several new retail locations operate out of the first floor of this building. To help with the increased demand for parking in the area, the City of Dearborn constructed a 373-space public parking structure next to the building. This structure, along with other nearby public structures, provide free parking for both Ford employees and shoppers. In addition, GDIA employees can park at nearby Ford campus parking spaces and use corporate shuttles to get to the Wagner Place building.

The parking space consists of a ground parking lot and a 4-level parking structure. As a free parking space, there are no gates installed, and no occupancy information is provided. Drivers usually circle around the ground level while searching for an available parking spot, and if none is found, the drivers usually proceed to go through the structure. It is beneficial to provide information on parking vacancy, especially in the structure, to the residents of Wagner Place to reduce the unnecessary time and fuel spent while scanning the parking space before resorting to an alternative parking location. The parking information will also be very useful to other Ford employees attending offsite meetings at Wagner Place by helping them to decide on whether to use their personal vehicles or to take a corporate shuttle.

To address the parking challenge, the GDIA team decided to explore the opportunity to develop a cost efficient IoT-based parking information solution. Specifically, the team considered leveraging the existing Closed Circuit Television (CCTV) security cameras in the infrastructure which cover the parking space, and connected vehicles for information delivery to the parking tenants using SmartDeviceLink (https://smartdevicelink.com/). The team created a partnership with the City of Dearborn and was able to process real time video streams from the cameras to design a parking information system at minimum cost and effort. One of the critical requirements to utilize the video stream is data privacy. The stream can only be processed on the edge, and no video or images are to be stored or accessed by the Cloud. In addition, the system must not track license plates or other distinguishable features of any private vehicles. Therefore, in order to create the smart parking information service while meeting the budget, technical, and privacy requirements, an edge device is setup in the infrastructure to send anonymous disjointed messages to the digital twin in the cloud where Petri Nets were utilized for several reasons as will be shown in this paper.

The paper is organized as follows. The next section explores the existing technologies and methods used in parking information systems, along with the proposed architecture. Section 3 discusses the Petri Net-based occupancy model and how to implement process discovery methods and automatically deploy parking systems. Section 4 provides the analytics and the implementation results. Section 5 concludes the paper.

Makke, O. and Gusikhin, O.
Petri Net-based Smart Parking Information System.
DOI: 10.5220/0009583403850393
In *Proceedings of the 6th International Conference on Vehicle Technology and Intelligent Transport Systems (VEHITS 2020)*, pages 385-393
ISBN: 978-989-758-419-0

2 SMART PARKING

The demand for parking information systems is rising. In large cities, people spend significant time looking for a vacant parking spot. For example, drivers in the United States spend about 17 hours annually looking for a parking spot, and that number jumps to 107 hours/year in New York City and 85 hours/year in Los Angeles (Cai et al., 2019). This demand created an interest in the topic which ranges from information services to developing new sensor technologies and detection methods as shown in (Lin et al., 2017) where the authors classified the existing approaches along three dimensions: information collection, system deployment, and service dissemination.

Information collection relies on obtaining relevant information from parking sensors or from crowd-sensing. In (Almeida et al., 2015), the authors categorized the sensing into 3 types: Counter-based, sensor-based, and vision-based. The counter-based systems use existing gate-arm counters, inductive loop detectors and similar sensors located at the entrances and exists. The sensor-based systems attempt to sense if spots are vacant or occupied, and can provide information regarding where to find a vacant spot. In this approach, sensors are installed at each parking space. The vision-based systems can use either dedicated cameras configured specifically to support the parking information system or leverage existing surveillance cameras. The latter method can be a cost efficient alternative, although it has a major challenge. The existing CCTV system may not provide full visibility to monitor the occupancy of all parking spots especially during high occupancy times. In most cases however, existing CCTV systems provide sufficient coverage to identify the vehicle movements within critical choke points in the parking space. Tracking vehicle movements between the parking areas and feeding the results into an adequate model helps overcome the challenge. The solution presented in the paper uses this type of approach.

In addition, a combination of counter-based, sensor-based, and vision-based systems can also be used. For example, (Seymer et al., 2019) describes a parking system which utilizes BLE beacons to track the vehicles inside the parking area. The authors also used cameras located at the entrance/exit points of the parking space to account for the potential of drivers disabling their beacons.

2.1 Vision-based Systems

In vision-based parking systems, there are generally two approaches to track the occupancy (Huang and Wang, 2010). The first approach is to identify empty/occupied spots and the second approach is based tracking the vehicles in the parking space. The first approach can be roughly categorized into three major types: car-oriented methods, space-oriented methods, and parking-lot-oriented methods (Màrmol and Sevillano, 2016), or a combination of these methods. Methods using car-oriented approach detect parked vehicles and then derive the number of available spots. Methods using space-oriented approach compare the appearance of the parking place, using a static model prepared in advance, with the current appearance of the parking space. The vacancy is then determined by analyzing the dissimilarity between the appearances. In parking-lot-oriented methods, the parking lot is modeled in a 3D program such as game engines, and the observed image is compared to the 3D model to infer the parking status. For example, in (Huang and Wang, 2010), the authors proposed a probabilistic method which considers lighting variations, shadows, varying perspective distortion on the image, and inter-object occlusion among parked cars. In (Sun et al., 2018), three commercial truck parking detection systems were evaluated, and their accuracy was above 95%, and exceeding 99% in some cases. However, the effect of rain and snow required further analysis. In (Màrmol and Sevillano, 2016), the authors introduced QuickSpot, which is an on-street parking spot detection system based on video analytics. When vision-based systems are used to to identify empty and occupied spots, the required number of sensors and/or cameras is usually a scalar multiple of the number of spots. For example, one camera may be required for every 10 spots as proposed by DeepParking team, which can be found at https://github.com/DeepParking/DeepParking. In some cases where the detection is performed in an opened parking lot, it is possible to reduce the number of required cameras by placing the cameras strategically to obtain a bird view. However, this is not possible in an infrastructure. In addition to the presence of many walls and relatively low ceiling height, vehicles can easily obscure the vision of other vehicles behind them relative to the camera. Thus, many cameras or sensors must be used in a parking infrastructure to cover all the spots. In the second approach, the main idea is to track how many vehicles went into a specific area, and how many vehicles left that area, and the difference between the two indicates the number of available spots.

The technology supporting vision based systems is rapidly evolving. The Tech industry has been working on platforms which assist in providing solutions for smart parking and other applications which require image processing on the edge. For example, NVIDIA now provides DeepStream SDK to achieve high throughput for applications requiring object detection, tracking, and classification (NVIDIA, 2019). Their tracker library offers three options: IOU tracker, Kanade-Lucas-Tomasi (KLT) tracker, and Discriminative Correlation Filter Tracking method. Moreover, Microsoft and NVIDIA now provide an IoT platform for applications which wish to use NVIDIA Jetson devices as an edge computing device, and Azure platform for Cloud computation. Information about this platform can be found at https://github.com/Azure-Samples/NVIDIA-Deepstream-Azure-IoT-Edge-on-a-NVIDIA-Jetson-Nano. Once a vision-based method is selected and deployed on the edge, the critical task is to integrate the individual messages and translate them into meaningful parking occupancy information. The system architecture must be agnostic to the used methods on the edge in order to be extensible.

2.2 Proposed System Architecture

The system architecture for our solution is shown in figure 1. An edge device is placed in the infrastructure and connects to existing CCTV cameras. In the future, additional sensors can be added to the infrastructure and easily integrated with the system. This edge device is capable of processing video streams from several cameras and translates the data to labels. The edge device sends the labels to a digital twin of the parking space in the cloud. The digital twin communicates with the information dispatcher to provide the necessary vacancy information and possible recommendations on where to park within the infrastructure. The information dispatcher obtains the necessary information and transform it for use by the mobile application, so that the mobile application can be designed generically, without any knowledge of the parking infrastructure. The mobile application displays the information it receives on its screen, or alternatively, on the vehicle's head unit by implementing SmartDeviceLink SDK. This architecture can be expanded to allow more sensors and cameras to be used. Wireless devices and cameras can be used as additional sensors for special parking spots, and their bandwidth for can be optimized as necessary (Gholamnejad Davani and Sarhan, 2017) to cover large areas. By adding new devices additional disjointed messages are created. The digital twin plays a critical role to consolidate and integrate the disjointed separate messages from the edge device. The details behind the digital twin modeling is discussed in the next section.

3 Petri Net MODELING

To implement digital twin, Petri Nets (PN) were selected to represent the parking occupancy model. Petri Nets have been proven to be a powerful framework for design, evaluation and control of discrete event systems (Giua and Silva, 2018). In addition to basic PN models, there are numerous PN modifications and extensions which are used to incorporate diverse methods from different domains while sharing a common modeling approach.

A Petri Net structure is a marked bipartite graph formally defined as following (Rozenberg and Engelfriet, 1998):

Definition 1. *A Petri Net structure N is a 3-tuple N = (P,T,F) where $P = \{p_1, p_2, ..., p_n\}$ is a finite set of n places; $T = \{t_1, t_2, ..., t_m\}$ is a finite set of m transitions, $F \subseteq (P \times T) \cup (T \times P)$ is a set of arcs; $P \cap T = \phi$ and $P \cup T \neq \phi$ A marking is a function $M : P \to \mathbb{N} \cup \{0\}$ that assigns a non-negative integer number of token to each place.*

Graphically, a Petri Net is represented as the following. Places are represented by circles and transitions are represented by bars. Places and transitions are connected through directed arcs, and the tokens are represented by dots in the places. See for example figure 2. The dynamic behavior of PN is defined by its transition firings. Transition firings move tokens from their input place to their output place.

Petri Net models have wide range of applications in the industry, specifically in applications related to information integration and IoT-based systems. For example, in (Gusikhin et al., 1996), the authors demonstrate how to integrate plant floor information for scheduling and control in real time. They used modified Petri Nets to represent material flow, and they tied plant floor heterogeneous messages with transitions of a PN model. They used existing process timing information to compensate for potential errors in plant floor information systems. (Zhang et al., 2018) discusses PN application to IIoT, where Coloured Petri Nets are used to assist in managing multiple sensors in 'Plug and Play' manner. In (Yang et al., 2014) Petri Nets have been applied to automatically compose IoT software services.

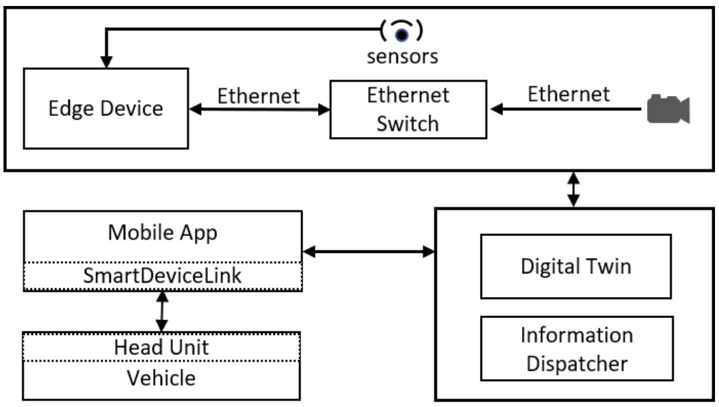


Figure 1: Smart parking system architecture.

Furthermore, among the comprehensive set of analytical and simulation techniques PN provide for evaluation and control of discrete event systems, PN have developed an extensive body of research in state estimation and process mining which can be useful in development of IoT based tracking systems. PN marking estimation research deals with algorithms and methods that estimate the state of a model which combines observable and non-observable transitions and places to determine the optimal set of sensors to achieve observability (Ru and Hadjicostis, 2010) (Ma et al., 2020).

Process mining focuses on automatic construction of PN models by analyzing the events log files (Aalst, 2016). A review of different PN-based process discovery algorithms is provided in (Dongen et al., 2009). Within the scope of IoT tracking applications, automatic generation of a PN model can substantially reduce the time for initial model development and testing. In our work, we leverage α-algorithm (van der Aalst and van Dongen, 2013) to construct the initial parking model and automatically associate IoT messages with respective transitions.

3.1 Modeling Approach

In the context of parking application, the natural interpretations of PN is that the tokens represent vehicles, places represent areas or individual parking spots, and transitions represent entrances and exits of different parking areas. For example, (Lourenco and Gomes, 2008) uses this interpretation and describes an interactive modeling process for a three story parking infrastructure controller.

One of the advantages of PN is that they support evolutionary and hierarchical development of the model. A PN place can represent the entire parking space, specific parking zones, or even individual parking spots. The fidelity of the model is

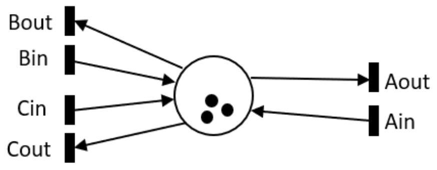


Figure 2: Low fidelity model. The place represents the entire parking structure.

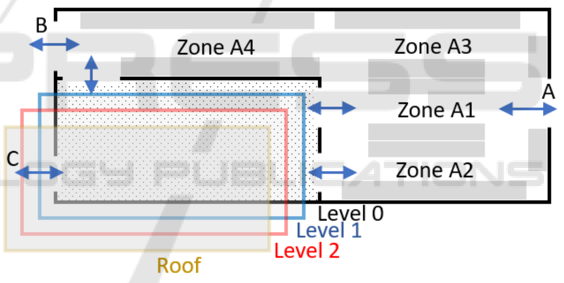


Figure 3: Parking space. It consists of an open area and 4 parking levels, including roof. Grey area represents where vehicles can park.

determined based on the available IoT infrastructure and the application logic requirements. The practical approach is to combine different levels of granularity for different zones, based on the need.

We start by using a low fidelity model of the parking infrastructure. A low fidelity PN model is shown in figure 2. The whole parking space is lumped into one place, and each entrance is represented by two transitions, one for entering vehicles, and the other for exiting vehicles. Despite the simplicity of this model, it can provide important insights into the dynamics of the parking space and offer a useful parking service, as we will show later.

Normally, there are enough CCTV cameras to cover critical choke points in the parking space. By selecting the cameras overseeing the choke points in the infrastructure, a medium fidelity model of the parking structure in figure 3 is created. This PN model is shown in figure 4. In this model, similar to the previous model, each camera detects vehicles transitioning from one place to another. Elaborate models as this one would also allow a guidance service to be provided due to its increased fidelity.

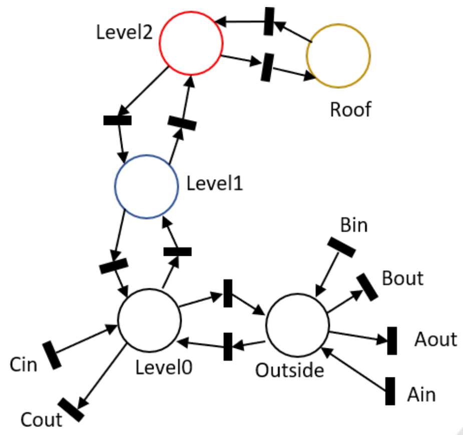


Figure 4: Medium fidelity model. The place represents group of parking spots.

In order to provide an even more detailed model and to identify the vacancy for charging zones or disability zones, the PN model can be further enhanced as shown in figure 5. The requirement for a higher fidelity model may lead to a PN model with some transitions and places that are not supported by existing sensors. The class of PN that deals with this type of models is Partially Observable Petri Nets (POPN) that combine observable and non-observable transitions. POPN research provides the methods to estimate the current markings from the existing observations (Ma et al., 2020) or attempts to find a minimum set of sensors (or additional sensors) required to achieve complete observability (Ru and Hadjicostis, 2010). In lieu of using additional sensors or cameras, existing cameras can sometimes be used to detect vehicles transitioning between multiple defined places by analyzing different regions in the cameras' field of view. Each camera can then trigger several transitions in the PN model.

It is possible to refine the model further to detect the vacancy of each spot individually in extreme cases, at an added cost. We argue that this is only required for special parking spots if the parking space is divided into sufficiently small zones. It would be enough to provide information and guidance to the zone containing an empty spot unless it is a special spot. Nevertheless, as will be discussed later, the PN model is auto-generated, and this type of solution offers a scalable method to provide smart parking service regardless of the desired fidelity. Based on a given budget, the fidelity can be selected, and over time, new zones can be added by refining the PN model.

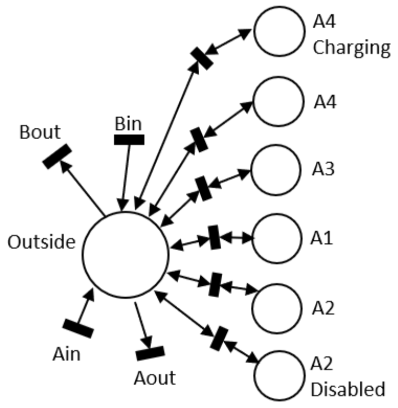
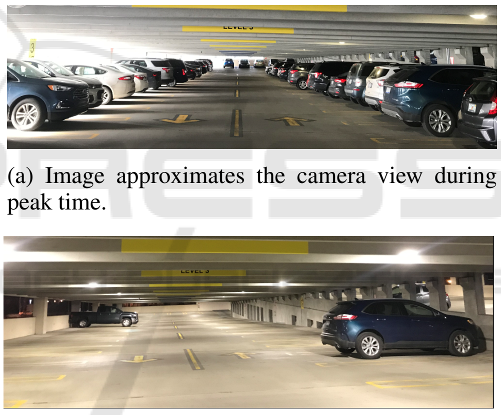


Figure 5: High fidelity model for the ground level. Each zone is assigned a place. Each special spot is assigned a place individually.



(a) Image approximates the camera view during peak time.

(b) Image approximates the camera view at night.

Figure 6: Peak time vs night time images.

An important aspect of the model is initialization. The simplest way to initialize the model is to inspect the parking lot at night when only a handful of cars may be present, and manually input the initial marking once. If an absolute accuracy of the tracking sensors can be guaranteed, this initialization approach may be sufficient. However, in practice, we must account for the potential errors in sensor tracking, when the cameras (or other sensors) may miscount the number of entering or exiting vehicles. Consequently, it is necessary to develop methods to re-initialize the model at regular intervals (e.g. once a day). In this case, the manual process is time consuming and error prone, and may not be adequate. One method to automate the recognition of the initial parking state is using cameras to identify the open parking spots or parked vehicles. As we discussed previously, in case of high occupancy as seen in in figure 6a, it may be very challenging for cameras to identify the occupied spots. At night, however, when the parking lot is mostly empty, the identification is much easier as shown in figure 6b. Consequently, we can initialize the model based on the vision recognition of the occupancy at the times when the parking mostly empty. Alternatively, during peak hours, the parking space gets completely filled. The indication of a completely filled parking space can be detected by examining the overflow vehicle movement pattern as will be shown later in the paper.

3.2 Process Mining

In order to have a versatile and multipurpose cloud component, it is possible to reduce the burden of modeling the parking space by applying techniques in process mining. For example, α-algorithm (van der Aalst and van Dongen, 2013) is a known method to generate a Petri Net model from event logs. To make use of the α-algorithm, we must ensure that the transitions are triggered in the correct temporal order, so that no incorrect places are inserted. For example, if a vehicle is entering the roof, and then another vehicle enters zone A1 in figure 5, it may appear that zone A1 can be entered from the roof level. To remedy this problem, a simple solution would be to track a special vehicle such as a security vehicle as it passes through the zones, and only record these transitions during an initial training phase. Here, we assume that there is only one such security vehicle present at a given time.

The method for recording the log file is as follows. The edge device is configured to associate sensors with REST APIs, so that whenever a sensor detects a vehicle transitioning between places, a defined REST API is called. This makes it easy to add new sensors or detect more zones using existing sensors. Each transition in the PN model is associated with a defined REST API which the edge can call. At the initial stage of deployment, only transitions exist and there are no places. As the sensors trigger the REST APIs, the order of the firing of transitions is saved in a log file in the cloud. This log file is used to auto-generate the PN model using the α-algorithm.

3.3 Information Service Deployment

Once the PN model is generated, it is saved as Petri Net Modeling Language (PNML) file, defined by the standard ISO/IEC 15909 Part 2 (PNML, 2019). This model can then be visualized using existing modeling tools. During the process mining phase, there may be some transitions which never trigger. For example, a security vehicle may never park in a charging station spot, and hence either manual modifications to the model will be required, or statistical analysis can be applied to infer that an additional transition exists between places. Once a PNML file is ready, it does not matter at this point how it was created. It is used to start the digital twin as a PN model.

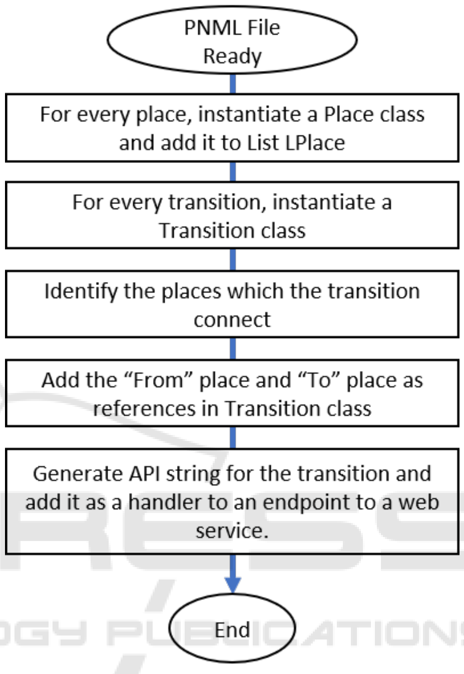

PNML File Ready
For every place, instantiate a Place class and add it to List LPlace
For every transition, instantiate a Transition class
Identify the places which the transition connect
Add the "From" place and "To" place as references in Transition class
Generate API string for the transition and add it as a handler to an endpoint to a web service.
End

Figure 7: Steps to auto-generate the Petri Net model.

Each transition name in the model is extracted from the PNML description, and a REST endpoint is automatically created in the cloud, which is derived from the extracted transition name. A graph structure is created based on the PNML. In this graph, places and transitions are implemented as classes, where the transition class is also the handler for the REST endpoints. By using REST APIs, only parameters related to security would be required, which keeps the model lightweight. Many commercial software products, however, are rigid for this approach, and the endpoints must be defined by this software. For these systems, only one REST endpoint is defined, and a JSON payload is used, containing information about the trigger which fired, and any other security related parameters. In this case, a dispatcher which handles all the transitions analyses the payload and triggers the correct transition. However, in our approach, there is no need for any commercial software, and the service runs with minimal requirements in the cloud. This reduces the deployment costs by eliminating

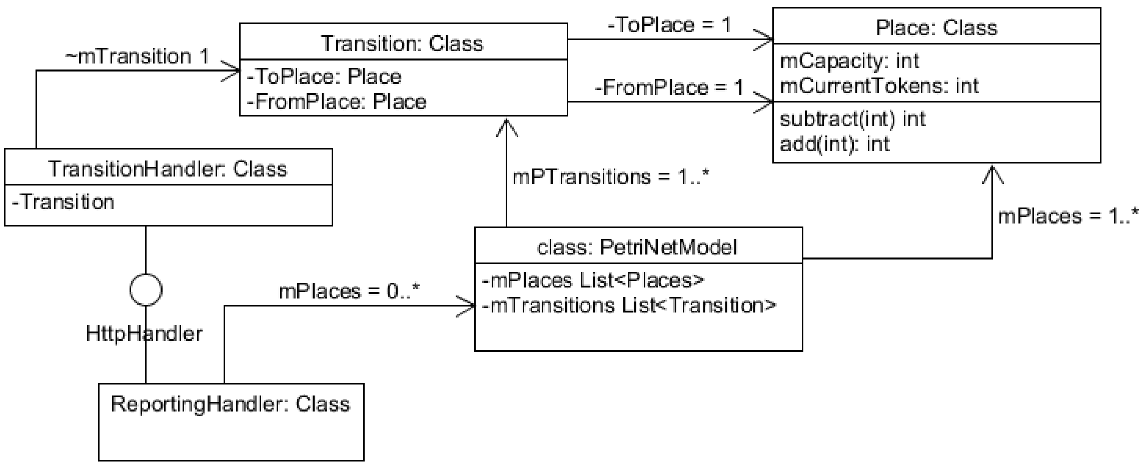


Figure 8: Simplified architecture of the web service.

licensing costs and reducing the hardware and storage space requirements in the cloud. The generation of the Petri Net model is shown in figure 7. The architecture of the cloud service is shown in figure 8. There are two classes which interact with the web service: IoT devices, and the user device, such as a mobile phone, which reports the total available spots in each place.

4 RESULTS

4.1 Analytics

We captured data over a period of three months. To highlight our results, we provide analysis for the parking structure, starting from the second level and up. From our experiments, we found that it takes a vehicle at least 2.5 minutes (average 3 minutes) to drive from the entry point of level-1 all the way to the roof, turn, and drive and back to the same entry point of level-1. By lumping all the places above ground level into one place, we gain insight into the activity within the parking structure. We chose an arbitrary Wednesday in January, after the businesses at Wagner Place opened, and compare that Wednesday with the 3 month average of Wednesdays, where for the most part, these businesses were closed.

Figure 9 shows our analysis. We define the occupancy curve as the *minimum* between capacity (270) and the number of both parked and moving vehicles in a place at a given time of the day. We define the “Overflow” curve as the *maximum* between the capacity (270) and the number of parked and moving vehicles. The curve “3M Average” shows the 3 months average occupancy curve of the structure above ground level. This average never exceeded capacity, and hence there was no overflow. However, for the chosen Wednesday in January, it can be seen that the occupancy has reached capacity, and there was overflow during the day.

It is also possible to infer when the parking structure reaches maximum capacity without knowing the initial marking by analyzing the vehicles entering and leaving the structure. Whenever the signal representing entering vehicles is similar to a 3 minutes lagged signal representing exiting vehicles, and when the volume of vehicles is not close to 0, we infer that capacity is reached. The curve “Overflow Inferred” drops whenever it is inferred that maximum capacity has been reached. The absolute value of this line is not important, and is shown for demonstration. Using this type of detection, variations in the capacity due to construction, or due to vehicles parked incorrectly, can be accounted for, although at a delayed time.

We calculated that on this given day, around 10.75 hours were wasted trying to find a parking spot. Furthermore, for every liter of gas consumed, around 2.3 Kg of CO_2 is produced, and each 10 minutes of idling costs 300 milliliters in wasted fuel for 3 liter engines (NRCan, 2019)(NRCan, 2015). Thus we estimate that the overflow on the selected day contributes to 44.5kg of CO_2 from around 5 gallons of gasoline.

4.2 Information Dissemination

Once the model is generated, the information has to be delivered to the users. SmartDeviceLink has been shown to be an effective solution for connected car features and mobility applications (Yeung et al., 2017). Moreover, it has been shown that it is possible

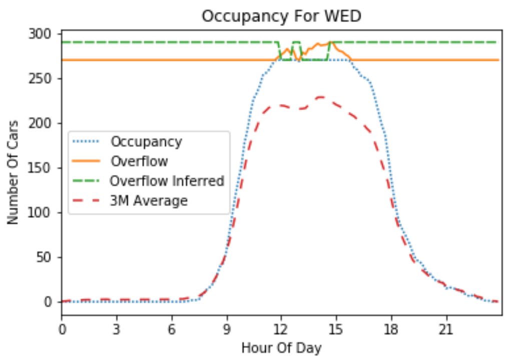


Figure 9: Analysis of activity on a given day.

to create dynamic vehicle applications by relying on SmartDeviceLink (Gusikhin et al., 2018) to integrate the application with the vehicle's head unit. Using similar principles for our smart parking solution, the mobile application implements SmartDeviceLink, and it communicates with the *ReportingHandler* in figure 8, which has access to the places in the Petri Net. The *ReportingHandler* creates a list containing the vacancy in each zone and sends it to the mobile application. The mobile application then presents the information to the users either on the mobile phone, or seamlessly in the vehicle on the head unit as shown in figure 10. If the model changes in the cloud, the mobile application dynamically changes its user interface, and hence reducing the development time. For instance, the current implementation shows the vacancy in the infrastructure as one place. If more places are desired, the cloud can easily send a table to the mobile app, which can simply update the display list to include multiple rows.

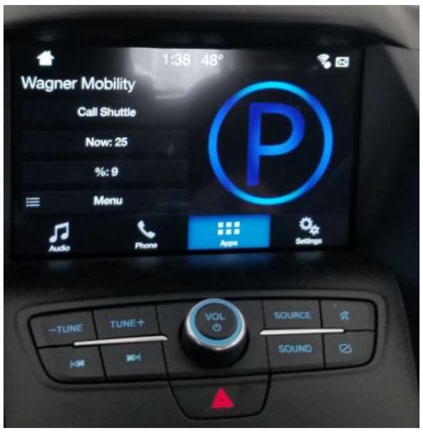


Figure 10: Integration of mobile application with vehicle using SmartDeviceLink.

5 CONCLUSIONS

The paper presents a case study of a cost efficient smart parking information system developed for free public parking space. The development leverages the existing CCTV infrastructure with vision processing on an edge device to ensure data privacy. The cloud backend implements Petri Net-based digital twin to closely track current parking occupancy. The information regarding the parking availability is delivered to the driver's mobile app and is seamlessly integrated with the vehicle's infotainment unit using SmartDeviceLink.

This approach has several benefits. A Petri Net model provides an efficient and effective mechanism to integrate disjointed IoT messages into coherent parking occupancy information. Petri Net graphical formalism allows for clear and concise representation of parking dynamics linked with individual sensors. Petri Nets supports modeling at the different levels of fidelity and thus the model can be designed to meet current budget and technological constrains with provision for future extensions. In addition, we demonstrated how we can leverage PN process discovery algorithms to automatically generate the model and thus reduce the development and debugging efforts. Petri Nets allow us to abstract the development from the specific application or even domain area. The same platform can be used for arbitrary parking spaces, or for any other systems which require tracking, such as inventory tracking.

We also presented and discussed the analytics of parking dynamics over the course of 3 months. The analytics clearly show that in the absence of parking information system there is measurable waste produced from the vehicles overflowing in a full parking structure. Our results demonstrate that even a simple parking information billboard indicating whether parking structure is full or not may save cumulatively over 10 hours of time wasted and around 5 gallons of gasoline on a busy day. These results can be used by cities as a template for business justifications to invest in parking information system infrastructure.

The next development steps include extending PN model to more detailed and granular parking representation and extension of the services to include guidance to the available parking spot delivered through the mobile app. This work requires estimation of vehicle's position within parking structure based on vehicle sensors, such as GPS, accelerometer, speed, and steering angle. Based on this estimation, the digital twin can determine the best route to get to the available spot. One of the key challenges for guiding a vehicle to an available parking spot, in the presence of multiple vehicles, is projecting which parking spot will be available simultaneously while providing the guidance to *that* spot. One of the additional advantages of using PN is that PN formalism offers analytics and simulation techniques to efficiently deal with these types of problems. Additionally, when a guidance service is available, the digital twin may communicate directly with connected vehicles in order to support the coordination of autonomous valley parking.

REFERENCES

Aalst, W. V. D. (2016). *Process Mining*. Springer.

Almeida, P., Soares de Oliveira, L., Jr, A., Jr, E., and Koerich, A. (2015). Pklot - a robust dataset for parking lot classification. *Expert Systems with Applications*, 42.

Cai, B. Y., Alvarez, R., Sit, M., Duarte, F., and Ratti, C. (2019). Deep learning-based video system for accurate and real-time parking measurement. *IEEE Internet of Things Journal*, 6(5):7693–7701.

Dongen, van, B., Alves De Medeiros, A., Wen, L., Jensen, K., and Aalst, van der, W. (2009). *Process mining: overview and outlook of Petri net discovery algorithms*, page 225–242. Springer.

Gholamnejad Davani, S. and Sarhan, N. J. (2017). Experimental analysis of bandwidth allocation in automated video surveillance systems. In *Proceedings of the 25th ACM International Conference on Multimedia*, MM '17, page 1457–1464, New York, NY, USA. Association for Computing Machinery.

Giua, A. and Silva, M. (2018). Petri nets and automatic control: A historical perspective. *Annual Reviews in Control*, 45:223–239.

Gusikhin, O., Lewis, D., and Miteff, J. (1996). Integration of plant floor information for scheduling and control. *SAE Techical Paper Series*.

Gusikhin, O., Shah, A., Makke, O., Smirnov, A., and Shilov, N. (2018). Dynamic cloud-based vehicle apps - information logistics in disaster response. In *VEHITS*, pages 626–635.

Huang, C. and Wang, S. (2010). A hierarchical bayesian generation framework for vacant parking space detection. *IEEE Transactions on Circuits and Systems for Video Technology*, 20(12):1770–1785.

Lin, T., Rivano, H., and Le Mouël, F. (2017). A survey of smart parking solutions. *IEEE Transactions on Intelligent Transportation Systems*, 18(12):3229–3253.

Lourenco, J. and Gomes, L. (2008). Animated graphical user interface generator framework for input-output place-transition petri net models. In van Hee, K. M. and Valk, R., editors, *Applications and Theory of Petri Nets*, pages 409–418, Berlin, Heidelberg. Springer Berlin Heidelberg.

Ma, Z., Li, Z., and Giua, A. (2020). Marking estimation in a class of time labelled petri nets. *IEEE Transactions on Automatic Control*, 65(2):493–506.

Màrmol, E. and Sevillano, X. (2016). Quickspot: a video analytics solution for on-street vacant parking spot detection. *Multimedia Tools and Applications*, 75(24):17711–17743.

NRCan (2015). 2019 fuel consumption guide. https://www.nrcan.gc.ca/energy/efficiency/communities-infrastructure/transportation/idling/4459 Accessed. 2019-12-15.

NRCan (2019). 2019 fuel consumption guide. https://www.nrcan.gc.ca/sites/www.nrcan.gc.ca/files/oee/pdf/transportation/tools/fuelratings/2019FuelConsumptionGuide.pdf Accessed. 2019-12-15.

NVIDIA (2019). Nvidia deepstream. https://developer.nvidia.com/deepstream-sdk Accessed. 2019-12-15.

PNML (2019). Petri net modeling language. http://www.pnml.org/ Accessed. 2019-12-15.

Rozenberg, G. and Engelfriet, J. (1998). *Elementary net systems*, pages 12–121. Springer Berlin Heidelberg, Berlin, Heidelberg.

Ru, Y. and Hadjicostis, C. N. (8/2010). Sensor selection for structural observability in discrete event systems modeled by petri nets. *IEEE Transactions on Automatic Control*, 55(8):1751–1764.

Seymer, P., Wijesekera, D., and Kan, C.-D. (2019). Smart parking zones using dual mode routed bluetooth fogged meshes. In *VEHITS*, pages 211–222.

Sun, W., Stoop, E., and Washburn, S. S. (2018). Evaluation of commercial truck parking detection for rest areas. *Transportation Research Record*, 2672(9):141–151.

van der Aalst, W. M. P. and van Dongen, B. F. (2013). Discovering petri nets from event logs. In Jensen, K., van der Aalst, W. M. P., Balbo, G., Koutny, M., and Wolf, K., editors, *Transactions on Petri Nets and Other Models of Concurrency VII*, pages 372–422, Berlin, Heidelberg. Springer Berlin Heidelberg.

Yang, R., Li, B., and Cheng, C. (2014). A petri net-based approach to service composition and monitoring in the iot. In *2014 Asia-Pacific Services Computing Conference*, pages 16–22.

Yeung, J., Makke, O., Macneille, P., and Gusikhin, O. (2017). Smartdevicelink as an open innovation platform for connected car features and mobility applications. *SAE International Journal of Passenger Cars - Electronic and Electrical Systems*, 10.

Zhang, Y., Wang, W., Du, W., Qian, C., and Yang, H. (2018). Coloured petri net-based active sensing system of real-time and multi-source manufacturing information for smart factory. *The International Journal of Advanced Manufacturing Technology*, 94(9):3427–3439.