electronics

MDPI

Article

NOVA Mobility Assistive System: Developed and Remotely Controlled with IOPT-Tools

Filipe Moutinho [1,2,*], Rogerio Campos-Rebelo [1,2,3], Carolina Lagartinho-Oliveira [1,2], Edna Moreira [1], Bruno Almeida [1] and Luis Gomes [1,2]

[1] NOVA School of Science and Technology, NOVA University Lisbon, 2829-516 Caparica, Portugal; rcr@uninova.pt (R.C.-R.); ci.oliveira@campus.fct.unl.pt (C.L.-O.); ee.moreira@campus.fct.unl.pt (E.M.); bm.almeida@campus.fct.unl.pt (B.A.); lugo@fct.unl.pt (L.G.)

[2] UNINOVA, Center of Technology and Systems, 2829-516 Caparica, Portugal

[3] Polytechnic Institute of Beja, School of Technology and Management, 7800-295 Beja, Portugal

* Correspondence: fcm@fct.unl.pt

Received: 12 June 2020; Accepted: 14 August 2020; Published: 17 August 2020

Abstract: In this paper, a Mobility Assistive System (NOVA-MAS) and a model-driven development approach are proposed to support the acquisition and analysis of data, infrastructures control, and dissemination of information along public roads. A literature review showed that the work related to mobility assistance of pedestrians in wheelchairs has a gap in ensuring their safety on road. The problem is that pedestrians in wheelchairs and scooters often do not enjoy adequate and safe lanes for their circulation on public roads, having to travel sometimes side by side with vehicles and cars moving at high speed. With NOVA-MAS, city infrastructures can obtain information regarding the environment and provide it to their users/vehicles, increasing road safety in an inclusive way, contributing to the decrease of the accidents of pedestrians in wheelchairs. NOVA-MAS not only supports information dissemination, but also data acquisition from sensors and infrastructures control, such as traffic light signs. For that, it proposed a development approach that supports the acquisition of data from the environment and its control while using a tool framework, named IOPT-Tools (Input-Output Place-Transition Tools). IOPT-Tools support controllers' specification, validation, and implementation, with remote operation capabilities. The infrastructures' controllers are specified through IOPT Petri net models, which are then simulated using computational tools and verified using state-space-based model-checking tools. In addition, an automatic code generator tool generates the C code, which supports the controllers' implementation, avoiding manual codification errors. A set of prototypes were developed and tested to validate and conclude on the feasibility of the proposals.

Keywords: mobility assistive system; wheelchairs; mobility scooters; design automation tools; remote monitoring; model-driven development; petri nets

1. Introduction

According to the World Health Organization (WHO) [1], about 15% of the world population presents some form of disability, including 2–4% suffering from a severe disability such as quadriplegia, depression, or blindness [2]. This percentage has been increasing due to the rapid spread of chronic diseases, such as heart disease and stroke, cancer, and diabetes; as well as due to the increase of average life expectancy in developed countries. In general, these people have difficulties that are associated with their physical and psychological condition, including discrimination characterized by fewer educational and professional opportunities, dependence, and restricted participation in social activities due to their little autonomy and mobility, among others [3]. As such, people with disabilities

rely on health care and life support systems to overcome these difficulties, and to become more active and part of the community. For that, assistive technologies and devices [4] have been proposed to help these people in their daily lives, including 75 million people that, according to WHO, need to use a wheelchair [5].

Power Wheelchairs (PWs) and scooters are used by people with physical disabilities, as a way to improve their autonomy. In the market, it is possible to find PWs with different functionalities, aiming to meet the different customer needs. For example, some users cannot handle the traditional joystick to control their PW and need to use other control mechanisms, such as head or chin joysticks. Beyond the users' capabilities, the environment in which the wheelchair is used influences on how it is operated. Although PW users benefit from more mobility, their accessibility is not always guaranteed, it depends on the obstacles and barriers present in the cities [6,7], ranging from public accommodations and transport systems to public roads and sidewalks. It is common for people in wheelchairs to move on the side of the road as a way to get around obstacles. Furthermore, accidents involving PW and scooters have been reported in several countries [8,9], namely, PW that fall on their side, or users who fall to the floor sometimes due to the difference in ground level, and also due to the collision with cars and buses at junctions and intersections. A study in the USA reported that the mortality rate of pedestrians in wheelchairs is 36% higher than the overall pedestrians, with about half of accidents occurring at intersections and 38.7% in places without traffic control devices [10].

Several authors defend the update from PWs to Smart Wheelchairs (SWs) to overcome some of these problems [11,12]. SWs are wheelchairs equipped with embedded systems capable of acquiring and processing data from sensors that are available in the wheelchair. The sensors data, which can be used to constrain the operation and behavior of the SW, provide information about the user, the SW, and the environment. In the same way, smart city solutions have been contributing to support mobility and accessibility in cities [13–15]. In this context, several authors propose the use of crowdsourcing solutions, in which users and non-users of wheelchairs provide data about urban and architectural features of cities [16–18]. However, these proposals have some limitations. Firstly, the existing SWs are very expensive and they often need to be adapted to its user singular needs, which further increases the costs. On the other hand, the information shared by crowdsourcing may not be enough, and it can quickly become obsolete.

The existence of complementary systems, such as those that support the communication between road infrastructure elements and vehicles [19–21], may have a key role to play in the future, as a way of preventing accidents that involve wheelchairs and scooters and assisting the mobility of its users. In this sense, we argue that road signs, information panels, and other infrastructures should collect data from the environment, with information about the state of a street or a road (such as road traffic, flow of people on the street, mobility scooters on the road, wheelchairs leaving or entering transport, etc.), and provide it to wheelchairs and other vehicles. As such, we propose a novel Mobility Assistive System (MAS), named NOVA-MAS, which supports the acquisition of data from the environment, its control, and the dissemination of information between infrastructures and between infrastructures and their users. The infrastructures will be able to exchange data with users and it will also allow for the dissemination of information along public roads, ensuring updated and reliable information to support appropriate/safe decisions by their users. This will make road signs and information panels naturally dynamic, updating their information based on the data received from other infrastructures and vehicles, such as smart wheelchairs with multiple sensors. NOVA-MAS can be a complement to crowdsourcing platforms, supporting the dissemination of local and volatile information. By local and volatile information, we mean information that only matters in that location and at that moment.

The development and management of such systems is a challenging task. They require an architecture that supports (1) data collection, processing, and transmission; (2) specific controllers that change their behavior according to real-time information; and, (3) remote operation capabilities to allow their monitoring, execution control, and reprogramming. To support these functionalities, we also propose a suited model-driven development approach, combined with the use of a specific

Petri net class (the Input-Output Place-Transition Petri net class—IOPT-nets) [22] and its associated tool framework (IOPT-Tools) [23]. Petri nets [24] are a graphical and intuitive modeling formalism, with well-defined execution semantics, that has been widely proposed (1) to model and analyze road traffic, at macroscopic or microscopic level [25–30]; (2) to model, analyze, implement, and optimize, traffic sign controllers [30]; and, (3) to model and validate an interactive road sign system [31]. However, none of these Petri net classes and associated tools support the complete development flow of such systems, as IOPT-Tools framework does.

IOPT-nets have a formal definition and a well-defined execution semantics, enabling simulations and model-checking techniques, for verification purposes, and automatic code generation, for implementation purposes. This make IOPT-nets suited for safety critical systems development, such as connected and dynamic infrastructures, where a failure may cause or not prevent a road accident. IOPT-net models can be created in the IOPT editor [23]. IOPT simulation tools support token player animations and timing diagrams [23]. IOPT model-checking tools [32] include a state-space generator and a query engine, which are used to define queries and apply them to the generated state-spaces, to check the controller's proprieties, such as the occurrence of deadlocks, live-locks, or other undesired situations. IOPT automatic code generators support code generation in C [33] (used in this work), VHDL, Instruction List, and Simulink System block, ensuring that the systems will be free from manual codification errors. Because the IOPT C code generator, automatically generates controllers with run-time monitoring capabilities, and an IOPT debugger tool [34,35] is available, the infrastructures can be remotely monitored and controlled without any effort from the system developers. Therefore, as illustrated in Figure 1, with the use of IOPT-nets and IOPT-Tools it is possible develop infrastructure controllers, supporting: (1) the control of dynamic/variable road signs, and their remote monitoring and control; (2) infrastructures interaction; and, (3) infrastructures interaction with users/vehicles.

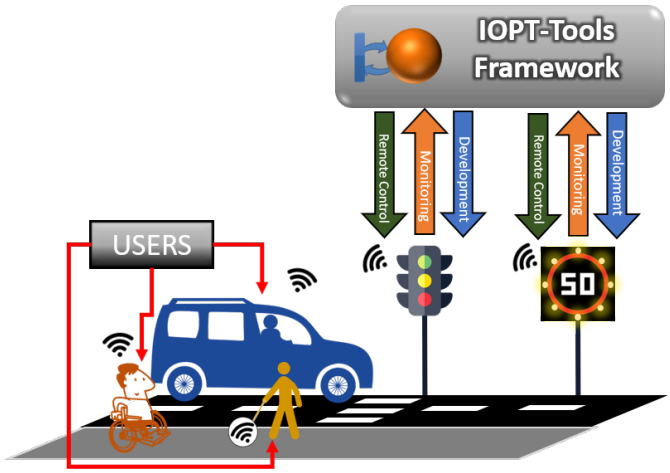


Figure 1. Mobility Assistive System (NOVA-MAS) overview.

The paper has the following structure. IOPT-nets and IOPT-Tools are presented in Section 2. Subsequently, after the related work (Section 3), the proposed mobility assistive system and its development approach are described (Section 4). The prototypes and performed tests are presented in Sections 5 and 6. Finally, the conclusions are presented in Section 7.

2. IOPT-Nets and IOPT-Tools

Petri nets [36,37] are a very well-known modeling formalism used in several areas to describe systems' behavior, benefiting from a rigorous execution semantics. Petri nets also benefit from a simple graphical representation, allowing for explicit modeling of key system's characteristics, such as sequence, concurrency, conflict, synchronization, sharing of resources, among others.

This simple graphical representation (which supports interaction between different agents that are involved in the system development) is complemented by a mathematical representation (allowing formal analysis of the models supported by a solid body of knowledge), as well as by a textual representation, in particular by the Petri Net Markup Language (PNML). The PNML, defined by the ISO IEC standard 15909 Part 2, supports the integration with computational frameworks.

Having the controller behavior modeling in mind, it is of paramount important to include explicit dependencies of the environment under control, namely input and output signals and events. These constraints (among others) lead to the definition of non-autonomous classes of Petri nets. Non-autonomous classes of Petri nets have been widely used in the area of controller design having strong links with automation systems, and they have been widely reported in the literature, namely as in [38].

The Input-Output Place-Transition Petri net class (IOPT nets) was selected as the underlying non-autonomous Petri net class to be used in this paper. The main characteristics of IOPT nets are briefly summarized below; a detailed presentation can be found in [22].

The IOPT net class is a non-autonomous extension of the well-known Place-Transition net class [39] emphasizing the interaction between the in- and output signals and events from the environment and the (autonomous part of the) Petri net model. In this sense, an IOPT-net transition can have associated with it a guard (which is a Boolean expression depending on input signals), a set of input events, as well as a priority. A transition is enabled to fire based on its input places marking (as common in Petri nets) if also ready to fire whenever the associated guard is evaluated to true and associated events occur. Priorities that are associated with transitions can be used to allow a deterministic firing whenever a conflict situation occurs (supporting an automatic conflict resolution modeling strategy). Output signals can be updated depending on place marking or transition firing. Output events are associated with transitions and are generated by their firing. Finally, test (or read) arcs are also available as an effective aid to produce compact controller's models.

The IOPT-Tools framework [23] is a model-based development set of tools, supporting the whole flow for distributed controllers development, while using a web-based strategy. IOPT-Tools are publicly available at http://gres.uninova.pt/IOPT-Tools/ having as ultimate goal to start with the edition of the IOPT-net model and to support the generation of the implementation code without actually writing a line of code. The whole process is supported by a browser, which means that heterogeneous platforms can be used, ranging from PCs, smartphones, tablets, or any other equipment supporting an up-to-date browser to run.

The IOPT-Tools framework is composed by several web-tools addressing the different phases of the development flow, starting with the IOPT net model editing where both autonomous part of the model (places, transitions, and arcs) as well as the non-autonomous part (in- and output signals and events, guards, priorities) can be edited and stored using PNML format. A simulator tool supports an interactive token-player that also allows the production of timing diagrams, where it is possible to interactively validate the net model. Additionally, the verification of the model proprieties can be accomplished by either direct inspection of the state-space automatically generated (which also provide information on deadlocks and place bounds, and can be downloaded for inspection by other dedicated tools), or through the use of a query editor and further analysis of the query results.

Complementing the referred web-based tools, several code generators are available allowing automatic generation of implementation code amenable to be directly deployable into specific platforms, either hardware or software. Currently, available code generators can produce C, VHDL, Instruction List, and Simulink System block, which have been used with a large number of implementation platforms, ranging from PCs, Raspberry-Pi, Arduino, FPGAs, and PLCs, to mention a few.

Of special interest, a debugger tool [34,35] allows for the monitoring and control of the code running in a remote controller. This tool, which combines the token-player simulator with an HTTP

client, connects to the remote controller HTTP server, also automatically generated by the code generator tool, to support remote debugging, control, and monitoring.

3. Related Work

The term smart city has been widely defined. In [13], a smart city was defined as a city that uses information and communication technologies (ICTs), integrated with traditional infrastructures, to provide technical assistance and share information. ICTs, such as the Internet of Things (IoT) and Internet of Everything (IoE) [40] technologies, have great impact on smart cities. Bluetooth Low Energy (BLE) [41] has been used in smartphones and smartwatches for people monitoring, and in BLE beacons (applied to indoor location systems [42]). Wi-Fi is used for monitoring [43,44] and control systems [45], as well as in security, for user authentication [46], among other examples. The aim is to improve the city's environment and promote people's quality of life. In [47], the authors remind us that it is important not to forget people with disabilities and the elderly.

A set of related works and technologies, which can support mobility assistive systems, are presented in this section. Next, works to assist the navigation of power wheelchairs on public roads is presented. Subsequently, works to support communication between infrastructure elements and vehicles are presented. Finally, the presented works are compared with this paper's proposals.

Some web and mobile applications have been proposed in order to provide information about obstacles, barriers, and routes, for wheelchair users to take into account. Wheelmap [48] and EasyGo [18] provide information regarding places that are accessible, based on data provided by users. These apps can also provide routes, where EasyGo informs about the barriers and facilities that appear on the way using the user's GPS location, as well as EasyWheel [49], which provides routes around obstacles. WEMAP [16] intends to improve these offers, generating door-to-door routes, using crowdsourcing data and information from local businesses' websites. mPASS system [50], WheelShare [51], and Wegoto [52] collect crowdsensing data, taking advantage of the sensory and computational resources of smartphones. When users are on the move, the data provided by accelerometers and gyroscopes is related to their physical location using GPS. In [51], the smartphone transmits the data to a server via Wi-Fi. Moving wheels [53] and SmartWheels [54] also use sensors to identify urban features from users' movement. The authors of WheelScout [42] use Ibeacons to also provide routes within buildings. In general, these works have similarities and improve the accessibility of wheelchair and scooter users; however, the works still do not consider their safe interaction with other individuals and vehicles present on the roads.

Over the past few years, several actions have been taken to increase road safety. Many of these actions are in the scope of Intelligent Transportation Systems (ITS) [55], whose main focus is the development of applications that increase the convenience and safety of vehicle users [56], such as in traffic management [57] or pedestrian detection [58]. Most of the existing applications are based on the recognition of traffic signs and obstacles. ITS also focuses on the development of infrastructure elements and vehicles, in order to make possible the communication between them. The need to create complementary and alternative systems, to also consider assistive vehicles on public roads, arises.

CVIS (Cooperative Vehicle-Infrastructure Systems) [59], COOPERS (Co-operative Networks for Intelligent Road Safety) [60], and SAFESPOT (Cooperative Systems for Road Safety) [61] were three major European projects that dealt with the interaction among vehicles and infrastructure elements [62]. The CVIS objective was the creation of a wireless network between vehicles and infrastructure elements that supports their collaboration to increase road safety and efficiency. CVIS core technology is CALM (Communications Air-interface, Long and Medium Range) (ISO TC204 WG16) [63]), which provides V2V (Vehicle to Vehicle), V2I (Vehicle to Infrastructure), and I2I (Infrastructure to Infrastructure) communications. COOPERS project purpose was the development of new safety services based on a two-way communication between infrastructure elements and vehicles. Finally, SAFESPOT project cooperated with CVIS and COOPERS projects and had complementary objectives. All of these works use wireless technologies to support communication between the road elements. Infrastructure

elements are often used as service providers and vehicles as consumers, except in the CVIS project, where these roles are not defined. This topic has been studied and funded since a few years ago, but it is still a very current topic with high interest from the scientific community, as evidenced by the recent journal papers [64–66] and new versions of the ITS-G5 standard [67] produced by ETSI (European Telecommunications Standards Institute) Technical Committee ITS. CVIS, COOPERS, and SAFESPOT projects took place about 10 years ago; however, many scientific contributions have been made since.

Model-driven development approaches have been proposed to support intelligent transport systems, mainly their simulation and verification, such as in [68] and [69]. The model-driven development of simulations for intelligent transport systems is proposed in [68], where a framework with guidelines and tools is proposed to specify and simulate these systems. ITS are specified while using a modeling language, which are then transformed into simulation models that support the simulation code generation. With this approach, it is possible to avoid mismatches between the system model and its simulation code, which may exist when these tasks are manually performed. A model-driven verification and validation approach, for computed-based control systems, was presented in [69] and applied to railway signaling. Model-driven development approaches have also been used for implementation purposes, such as in [70], where a Model-Driven Architecture (MDA) approach was used.

To conclude this section, the proposed mobility assistive system (NOVA-MAS) and its development approach are briefly compared with the related work. The NOVA-MAS is a complementary system to crowdsourcing applications and smart wheelchairs and may be integrated with them. The authors of this paper do not know any similar work that promotes the propagation of information on road infrastructures and vehicles, in order to make the routes of wheelchair users safer, considering other individuals and vehicles on the public road; or any similar development approach, applied to connected and dynamic road infrastructures. The proposed model-driven development approach simultaneously supports the specification, simulation, model-checking, implementation, and remote control and monitoring, through automatic code generation.

4. NOVA Mobility Assistive System

The proposed mobility assistive system (NOVA-MAS) and its development approach are presented in this section. The NOVA-MAS has a Service-Oriented Architecture (SOA) and it is composed of Wi-Fi beacons, named NOVA-WiFi-Beacons (NWBs) and Wi-Fi clients, named NOVA-WiFi-Clients (NWCs). NWBs are embedded in city infrastructures, whereas NWCs are embedded in vehicles or can be mobile applications.

4.1. Architecture

NWBs interact with each other, with NWCs, and with Remote Control and Monitoring Systems (RCMSs). Each NWB is a SOA component, which is a service provider and a service consumer. It is a provider of other NWBs, NWCs, and RCMSs; and a consumer of other NWBs. Each NWB receives requests from: (1) NWCs (requesting data); (2) other NWBs (sharing data); and, (3) RCMSs (sending commands and requesting data). Additionally, each NWB sends requests to other NWBs, to share data. An interaction scenario with three NWBs (NWB1, NWB2, and NWB3), nine NWCs (NWC1 to NWC9), and one RCMS, is presented in Figure 2, where each arrow connects one consumer (making requests) with one provider (replying).

Two message exchange patterns can be used: request/reply and publish/subscribe. NWCs should request data from NWBs using a request/reply message exchange pattern, while interactions between NWBs or between RCMSs and NWBs can be done using either request/reply or publish/subscribe pattern.

The proposed architecture for NWBs, composed by six modules, is presented in Figure 3:

- NOVA-WiFi-Client Provider (NWCP)—responsible for receiving and sending data to NWCs;
- NOVA-WiFi-Beacon Provider (NWBP)—responsible for receiving data from other NWBs;
- NOVA-WiFi-Beacon Consumer (NWBC)—responsible for sending data to other NWBs;

- NOVA-WiFi-Beacon DataBase (NWBDB)—containing data about the NWB, its neighbors, and about the last connected NWCs;
- RCMS Provider (RCMSP)—responsible for receiving commands and sending data to RCMSs; and,
- Dynamic Beacon Controllers (DBCs)—controls the dynamic/behavior of the infrastructure (for example, the lights of a traffic light sign) and the NWCP, NWBP, and NWBC modules.

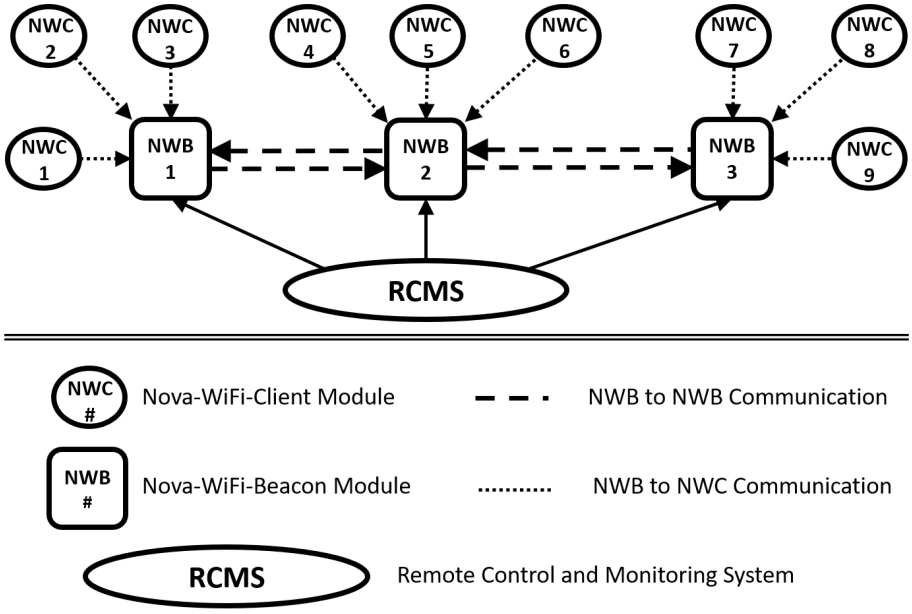


Figure 2. Interaction scenario between NOVA-WiFi-Beacons, NOVA-WiFi-Clients, and a remote control and monitoring system.

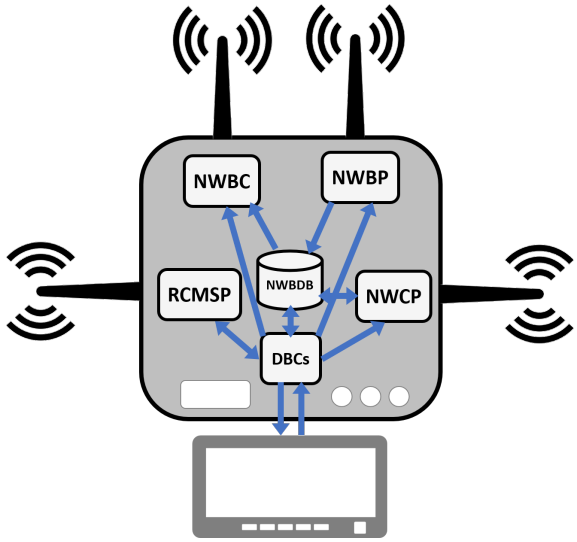


Figure 3. NOVA-WiFi-Beacon architecture.

The NWCP module is a wireless access point for NWCs, requesting data from the NWB and simultaneously providing its own data, for example, the wheelchair current position. The NWCP module saves the received data in the NWBDB and sends data from the NWBDB. The NWBP module is an access point for other NWBs that send data, for example, warning that a mobility scooter is on the road 50 m away. The NWBP module saves the received data in the NWBDB. The NWBC module is a client of other NWBs and connects to their NWBP modules. NWBC sends data from the NWBDB. The RCMSP module interacts with the DBCs module, writing the received commands and

reading the current controller state and IOs (inputs and outputs). Finally, the DBCs module controls the infrastructure behavior and the NWCP, NWBP, and NWBC modules. The DBCs module of each NWB must be developed according to the intended behavior for that NWB.

4.2. Development Approach

To create networks of NWBs and NWCs, it is necessary to develop the NWBs controllers with the required Wi-Fi communication nodes, complemented, or not, by other communication technologies. When considering the selected technologies, all of NWBs using those technologies must have similar network communication nodes (in the NWCP, NWBP, and NWBC modules), regardless of the NWB type, information, and specific behavior (for example: a crosswalk light sign or a pedestrian detection sensor). These communication nodes (their implementation code) can be previously developed and included in a library, to be used in the NWCP, NWBP, and NWBC modules generation. Currently, IOPT-Tools automatically generated code does not include these modules; however, as these are very similar, they can be manually included.

The DBCs module of each NWB must be developed according to the intended behavior for its infrastructure element, for example, to control a traffic light sign or the speed limit displayed in a variable message road sign. Such signs are safety-critical systems, given that their malfunction can cause road accidents. The behavior of each NWB depends not only on its data, but also on its neighbors' data. This means that each NWB may have a singular behavior. These unbounded number of NWB behaviors, associated with the fact that these NWBs are usually safety-critical systems, justifies the use of model-driven development approaches, supported by model-checking tools (to ensure proper specifications) and automatic code generators (to avoid manual codification errors and accelerate the development). In order to develop DBC modules, it is proposed the model-driven development approach presented in Figure 4, supported by design automation tools (the IOPT-nets and IOPT-Tools are presented in Section 2).

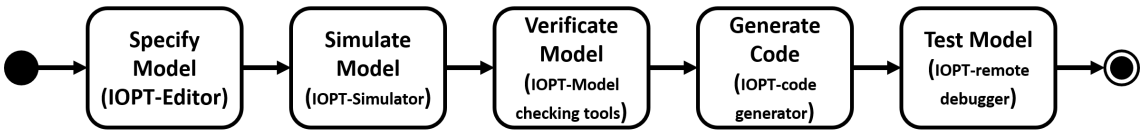


Figure 4. Development approach using the Input-Output Place-Transition (IOPT) framework.

The development approach proposed in this paper includes the following steps:

1. specify the DBCs module through a set of IOPT-net models (in the IOPT model editor), which includes the models of the sub-systems that control:
   - NWCP, NWBP, and NWBC modules; and,
   - the NWB dynamic behavior, for example, to control the lights of a traffic light sign;
2. simulate the created/updated models in the IOPT simulator tools, and update the created models, if required;
3. using the IOPT model-checking tools: write a set of queries about the models; generate their state-spaces, check the queries result to verify if the models have the desired proprieties; and update the created models, if necessary;
4. using the IOPT to C code generation tool: automatically generate the implementation code, which includes the DBCs module and an HTTP server that supports the remote control and monitoring (implements the RCMSP module); and,
5. during tests: monitor and influence the controllers' execution through the IOPT debugger tool; update the created models, if necessary, and repeat the previous steps.

Each NWB must have a database to store its own data and data from its neighbors. To allow data storage from fixed, temporary, or even from moving NWBs, the NWBDB module of each NWB

should allow the storage of data of "all" types of NWBs. This means that all NWBs may have similar NWBDB modules, enabling their replication. IOPT-Tools are not required to develop NWBDB modules; however, in each NWB, the interface between the DBCs module and the NWBDB module should be automatically generated. Currently, this generation is not supported, but it is intended to be.

5. Prototypes

A set of prototypes was developed and tested to validate the proposed mobility assistive system (NOVA-MAS) and the model-driven development approach. The low cost computing platforms used for these prototypes were the ESP8266-01 [71], the Raspberry Pi 2 (RPi2) [72], and the Raspberry Pi 3 (RPi3) [72].

5.1. Development Approach

The proposed model-based development approach is supported by the IOPT-Tools framework. Models are created in the IOPT model edition tool, as presented in Figure 5, which contains the model of the controller of the NWCP module. The second step of the approach is the model simulation and its correction/optimization. The IOPT simulator is shown in Figure 6, presenting the model of the controller of the NWBP module. In the presented simulation, the module is initialized and it has two connections from other NWBs.

Because, through simulations, it is not possible to ensure that the models are free from specification errors, in the third step of the proposed approach, the models' proprieties are verified using the IOPT model checking tools, which include a state-space generator tool and a query engine. The generated state-space graph, of the controller of the NWBP module, is partially presented in Figure 7. It is possible to verify, for example, the controller total number of states/nodes and the existence of deadlocks (this model has 253 possible states and is deadlock free). Given that, controllers usually have large state-spaces, often with millions of states, and it is not possible for humans to visually inspect it. The IOPT-Tools offer query tools to specify queries about the state-space to overcome this limitation. For example, in Figure 8, two queries are specified, in order to check whether the module accepts five or more connections from other NWBs. The query results (Figure 9) show that the controller can have five simultaneous connections, but not more than five (the first query has matching states, but the second query does not), as desired.

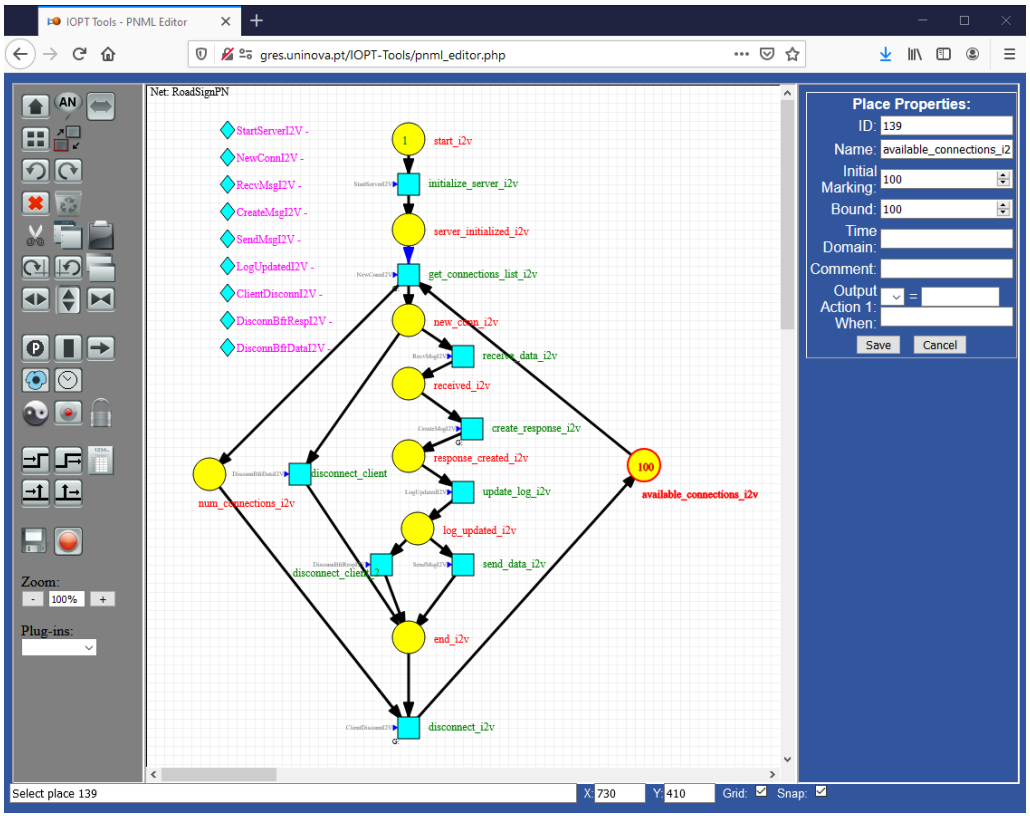


Figure 5. The IOPT edition tool with the model of the controller of the NOVA-WiFi-Client Provider (NWCP) module.

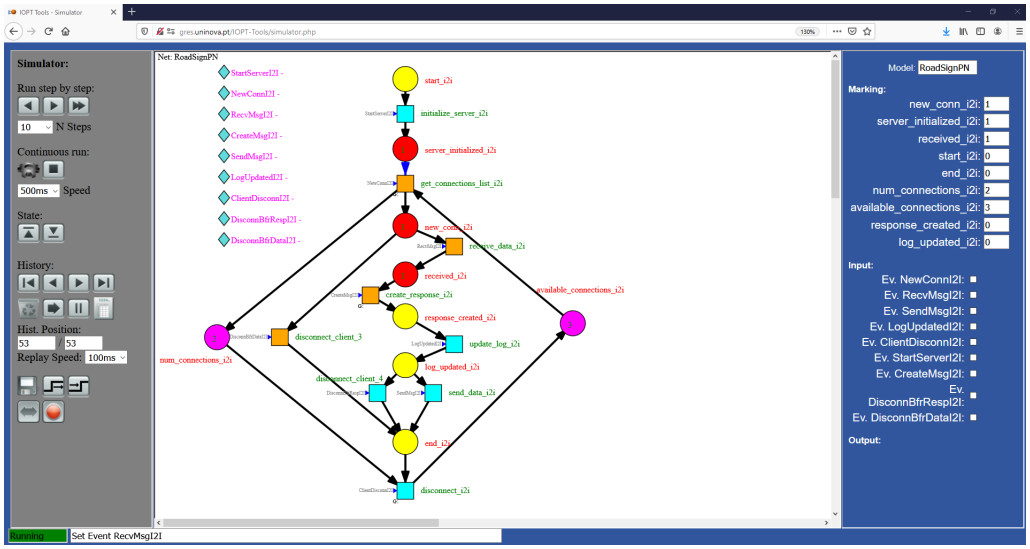


Figure 6. The IOPT simulation tool with the model of the controller of the NOVA-WiFi-Beacon Provider (NWBP) module.

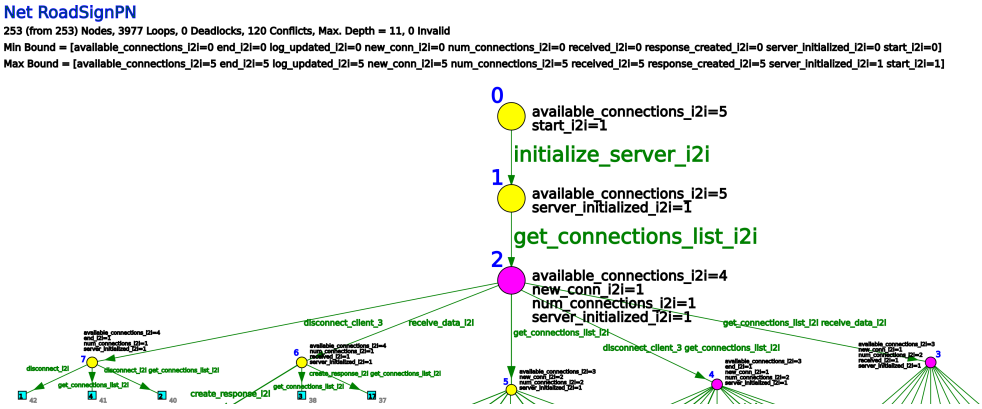


Figure 7. State-space of the controller model of the NWBP module.

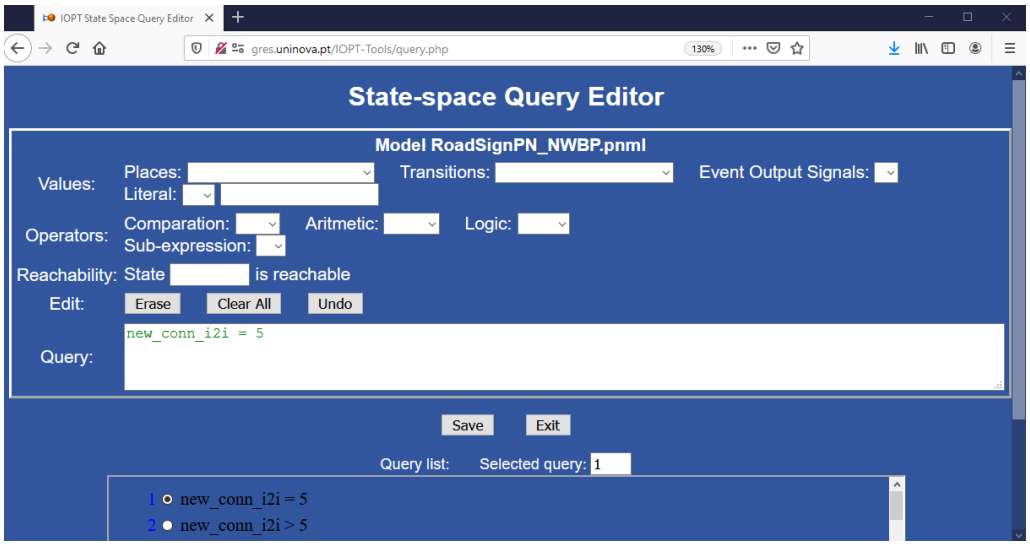


Figure 8. IOPT query edition tool with the queries performed to the controller model of the NWBP module.

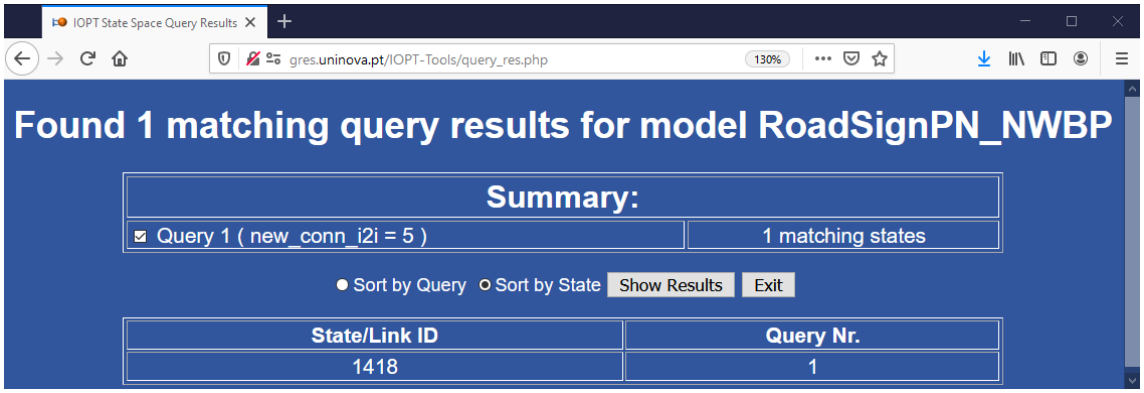


Figure 9. Query results of the controller model of the NWBP module.

After model validation through simulations and model-checking, the modules code is automatically generated while using the IOPT C code generator. Using an automatic code generator, it is possible to ensure that the implementation code conforms with the validated model. To generate the implementation code, it is only required to press the C code button that is shown in Figure 10. As previously mentioned (Section 4), the automatic code generator additionally generates an HTTP server that supports the debugger tool.

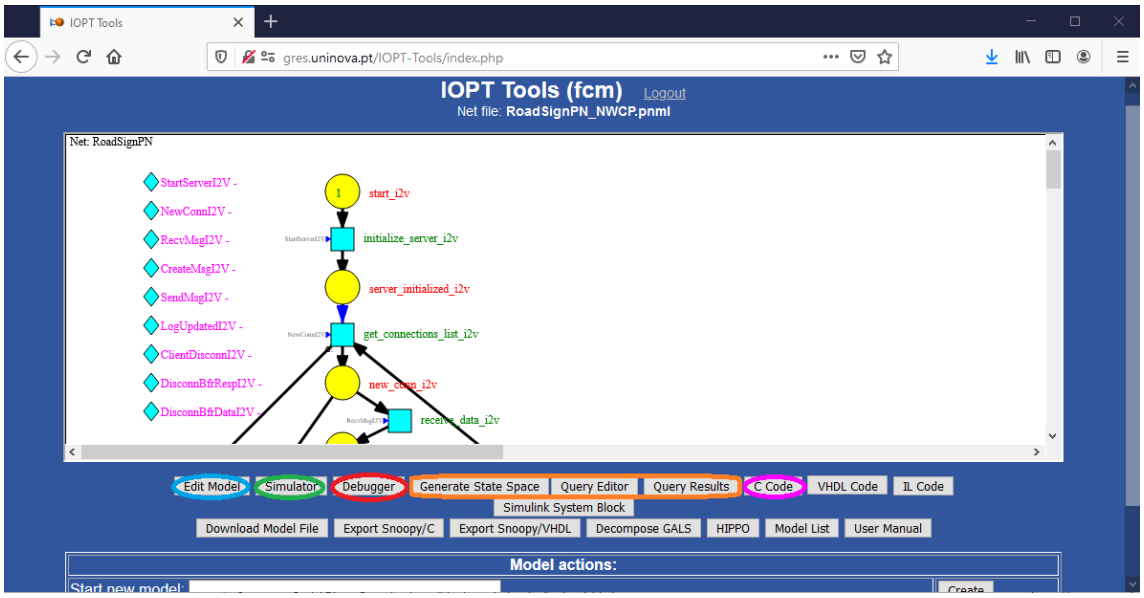


Figure 10. IOPT-Tools main interface.

The Debugger button from the interface presented in Figure 10 opens the debugger tool. This tool supports not only the system tests (the fifth step of the development approach), but also its remote control and monitoring (implements the RCMSP module) during its normal and maintenance operations. To connect the debugger tool (that has an interface similar to the simulator tool) to each NWB, it is required to introduce its IP address, Port, and password, as presented in Figure 11.

Figure 11. IOPT debugger tool connection interface.

5.2. Implementation

The NOVA-WiFi-Client (NWC) prototype was implemented on an ESP8266-01 with a GPS module (MT3329) connected via serial port, as presented in Figure 12 circuit. The USB to TTL converter module and the push buttons are required to program the ESP module. Arduino IDE (Integrated Development Environment) was used to program the board, using the "Generic ESP8266 Module" add-on [73] for Arduino IDE. The implementation code was written in C/C++. The NWC prototype, which is a service consumer, has the following operating cycle: (1) obtains the GPS coordinates; (2) searches for NOVA-WiFi-Beacon (NWB) networks; (3) connects to the one with the strongest signal; (4) sends vehicular own data (type, position, velocity, direction, ...) through an HTTP POST in JSON format; (5) receives POST reply with NWBs data; and, (6) presents received data to the vehicle user. The NWC does not connect twice to the same NWB, in a short period of time.

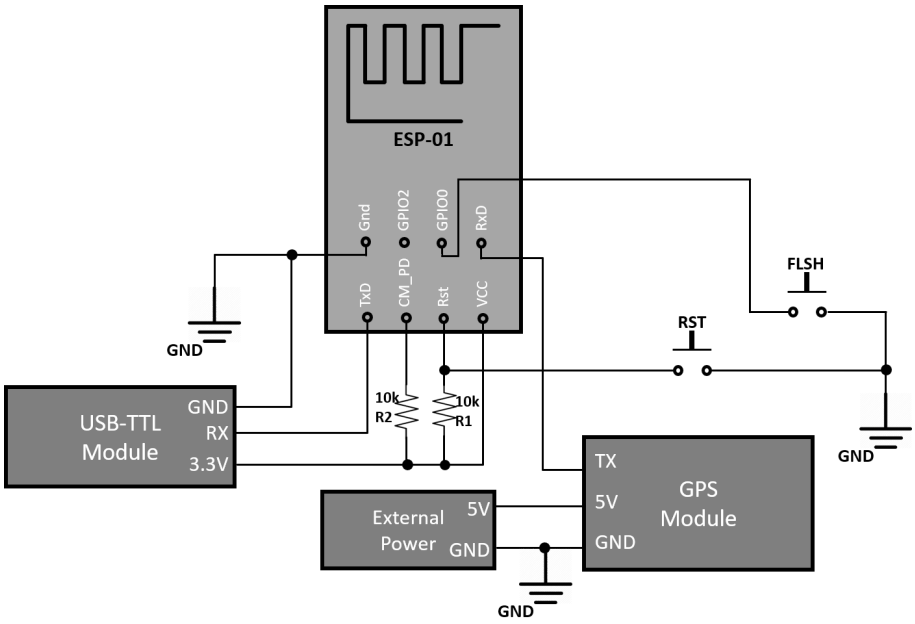


Figure 12. NOVA-WiFi-Client (NWC) prototype circuit.

NWB prototypes were created while using three different computing platforms. The first platform was an ESP8266-01, the second platform was a RPi2, and the last platform was a RPi3. The prototypes implemented in the first two platforms were only used to validate the interaction between NWCs and NWBs, whereas the prototypes that were implemented in the third platform were used to validate the proposed architecture and development approach. Given that the RPi2 does not have built-in Wi-Fi, like RPi3, in the second platform, an USB Wi-Fi adapter (SMC Networks EZ Connect N 150Mbps Wireless USB2.0 Adapter (SMCWUSBS-N3)) was used. Any of the prototypes provide Wi-Fi networks with a specific prefix name and have the following operating cycle: (1) receive connection; (2) wait HTTP POST and store the received data; (3) reply (in JSON format) with their own and also neighbors data (stored in its database); and, (4) disconnect.

The described operation cycles was named in this work as complete mode. However, a simple mode was considered to ensure that NWCs can obtain NWBs information, even when they cannot exchange messages. In the simple mode, the Wi-Fi network names contain NWBs information (their GPS location, direction, and specific data), for instance, to indicate a potential hazard.

A set of NWBs prototypes were implemented in several RPi3. In each RPi3, the built-in Wi-Fi interface supports the interaction with NWCs, and an USB Wi-Fi adapter supports the interaction with other NWBs. The following USB Wi-Fi adapters were used: (1) SMC Networks EZ Connect N 150 Mbps Wireless USB2.0—this adapter supports Wi-Fi b/g/n and WPA and WPA2 encryption standards; (2) XAVi XW501B 54Mbps USB2.0—an adapter that supports Wi-Fi b/g and WPA encryption standards; and, (3) Belkin Wireless G USB Adapter—another adapter that supports Wi-Fi b/g. The built-in Wi-Fi interface of RPi3 was configured as an access point using hostapd to configure the broadcast network—SSID, channel, Wi-Fi mode, network keyword, and so on, and the isc-dhcp-server. This is responsible for configuring the Dynamic Host Configuration Protocol (DHCP) server that is in charge of configuring the network to the endpoints—assigning IP addresses to clients, defining network masks, Domain Name System (DNS) servers. The USB Wi-Fi adapter, used to communicate with other NWBs, was configured as a mesh network, while using the B.A.T.M.A.N. (Better Approach to Mobile Ad-hoc Networking) advanced module [74] that implements the B.A.T.M.A.N. routing protocol. With this protocol, each node only has information about the best next-hop, so that the data reaches the remaining nodes.

The communication modules were developed using IOPT-Tools and Python, while using standard libraries [75], such as socket, threading, json, select, and xml.etree. The control part of the

communication modules was modeled through IOPT-nets, which were validated and automatically translated into C code using the IOPT-Tools. The interaction between the C processes and the Python processes was done through named pipes.

Figure 5 presents the model of the controller of the NWCP module. The model specifies, using the Petri net place *available_connections_i2v*, a NWCP module that can receive up to 100 connections from NWCs. Whenever the event NewConnI2V occurs, the transition *get_connections_list_i2v* (the second from the top) fires if the server is initialized (place *server_initialized_i2v* contains one token) and there are available connections. When it fires, one token is created in place *new_conn_i2c* and the NWCP module can receive data, create response, update log, send data, and disconnect. The model of the controller of the NWBP module (presented in Figure 6), which is also a provider, has similar behavior. The IOPT-net model of the controller of the NWBC module, which is a NWB consumer (making requests and waiting for responses), is presented in Figure 13. These three controllers are implemented in the DBCs module, running at the same time concurrently, being able to interact with each other whenever necessary. In this way, they can be seen as sub-models of the DBCs global model, which also includes the model that specifies the dynamic behavior of the specific NWB.

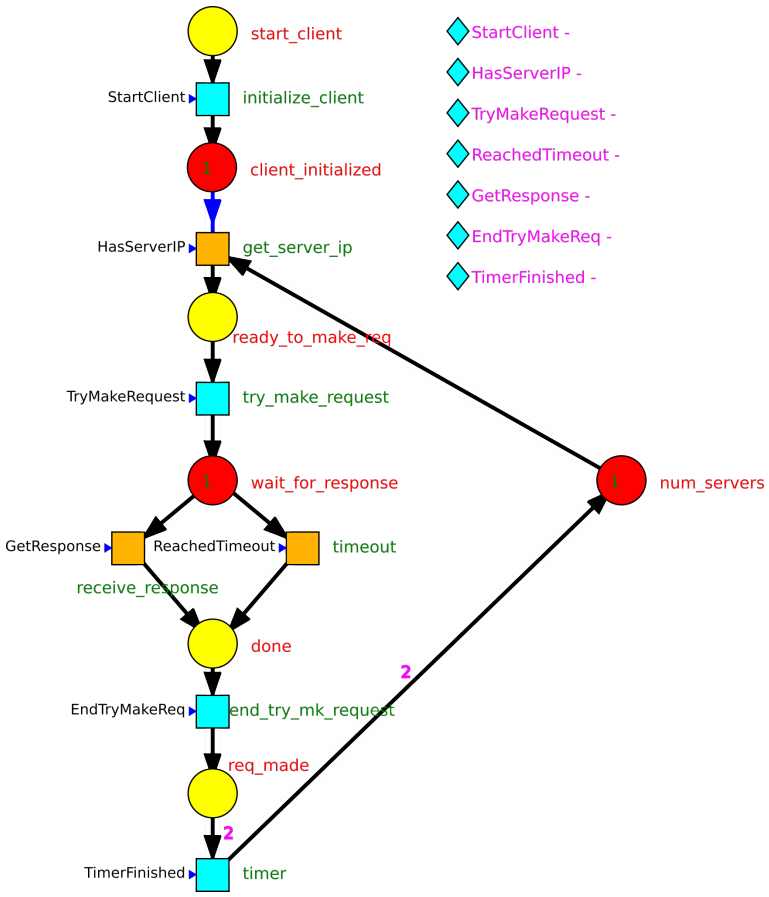
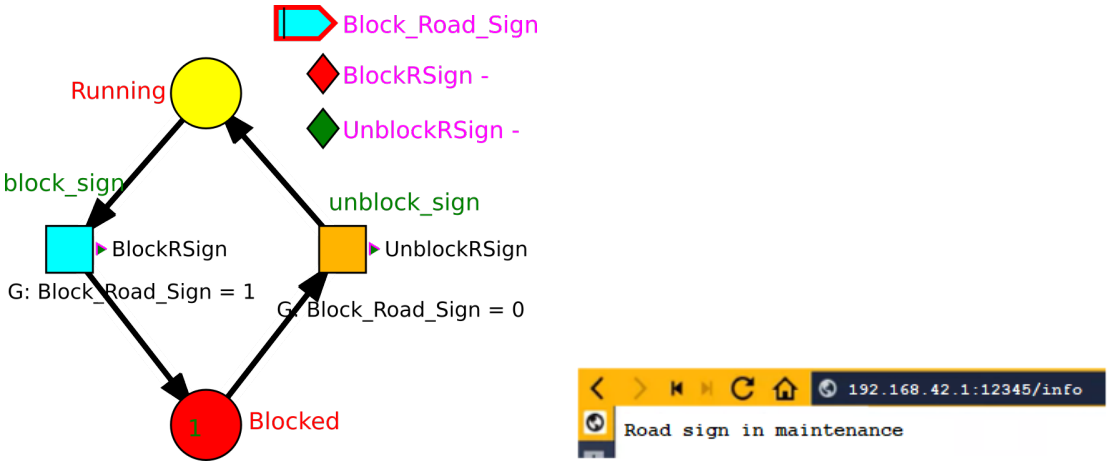


Figure 13. The IOPT-net model of the controller of the NWBC module.

The Dynamic Beacon Controllers (DBCs) and the Remote Control and Monitoring System Provider (RCMSP) were developed using IOPT-Tools. For that, the IOPT-net models were created in the IOPT model editor, specifying the behavior of the system, and then used in the IOPT validation and automatic generation tools, supporting simulation, proprieties verification, automatic DBCs generation and automatic RCMS generation. The interaction between the RCMS/DBCs process and the communication modules processes was also done through named pipes. It was necessary to include the string.h, used to compare the received information with the set of messages defined for the interaction between processes; and fcntl.h, which contains the functions required to create, read,

and write to named pipes. Figures 13 and 14 are snapshots taken from the IOPT Debugger tool, during the remote monitoring. In Figure 13, the controller of the NWBC module is waiting for the response of another NWB, whereas in Figure 14 the DBCs is in maintenance mode. It is important to note that through the IOPT debugger tool it is possible, not only to monitor the controllers, but also to change (control) their inputs, for instance, to switch from maintenance mode to normal mode, during its execution.



(**a**) The IOPT-net model (**b**) The associated response

Figure 14. The maintenance mode controller of the DBC module.

6. Tests and Results

An application scenario with a pedestrian crossing, in a double curve road city, is presented in this section, as illustrated in Figure 15. In both directions, the road has the pedestrian crossing after a blind curve and a variable message panel before it. Next to the crosswalks there are two pedestrian crossing road signs. The road—not only the pedestrian crossing, but also the roadside—is often used by pedestrians in power wheelchairs and mobility scooters, which are very low speed vehicles, sharing the road with cars and other high-speed motor vehicles.

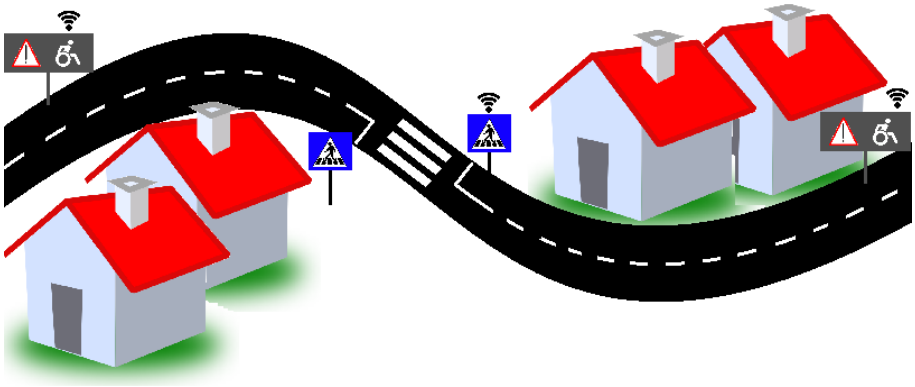


Figure 15. Application scenario with a double curve road city and four road signs.

The variable message panels and the road signs should be able to receive data from vehicles (of different types) and provide data to them. Vehicles (wheelchairs, mobility scooters, ...), with the developed NWCs, are able to provide their type, location, and speed, to the infrastructure elements with the developed NWBs (wireless communication). The panels' visual messages must change according to the number of vehicles in that area and their specific locations and speeds, providing information/alerts to the drivers of high-speed motor vehicles. The NWBs not only receive information

from vehicles, but they also provide it to them (via wireless), which means that the vehicles (low and high speed vehicles) can be aware of unexpected events in advance.

In this application scenario, not only the communication between vehicles (with NWCs) and infrastructure elements (with NWBs) is wireless, but also the communication between infrastructure elements. Additionally, it is considered that the two variable message panels cannot directly communicate due to distance and buildings, and the communication can be indirectly done through a pedestrian crossing road sign.

In the initial tests, the ESP8266-01 platform was compared with the RPi2 platform for the NWBs. With both prototypes implemented in the ESP8266-01 platforms (NWCs and NWBs), it was tested the connection time and the average of five measurements was 4.41 s. Subsequently, the NWB prototype was replaced by the one implemented in the RPi2 platform and the average of five measurements was 3.68 s.

After the initial tests, it was decided to test the interaction of the NWB prototype, implemented in the RPi2 platform with or without a password, with vehicles moving at different speeds. It is important to note that, without a password, the NWB can only provide public information and cannot rely on all of the information that it receives. The NWC prototype was installed on a vehicle and the NWB prototype was placed next to a road sign. The vehicle crossed the NWB at each speed a set of times. In the table presented in Figure 16, it is shown the number of tests performed at different speeds (50 km/h, 70 km/h, 80 km/h, and 100 km/h) and the average times taken by the vehicle, in the complete mode, to: obtain the GPS coordinates; search networks and choose the NWB with the strongest Wi-Fi signal; connect to the NWB; communicate (exchange messages and disconnect); and, the total time. With a password the average total time of 15 measurements, taken at 0 km/h, was 8.28 s and without a password the average total time, taken at 0 km/h, of 15 measurements was 7.23 s. The maximum speed tested was 100 km/h, which is the double of the speed limit in the Portuguese urban areas.

Password	Velocity (Km/h)	Nr. Of Tests	GPS (ms)	Search & choose Wi-Fi (ms)	Connection (ms)	Communication (ms)	Total (ms)
YES	0	15	1190.80	2418.20	4481.20	189.67	8279.87
YES	50	10	1033.10	14010.70	4982.90	214.40	20241.10
YES	70	10	1040.50	2721.50	4889.50	179.40	8828.90
YES	80	10	1107.90	2503.90	4707.40	200.90	8520.10
YES	100	10 (2 failed)	1084.38	2291.38	5016.75	182.00	8574.50
NO	0	15	1126.93	2407.40	3450.73	243.07	7228.13
NO	50	15	1128.67	8661.73	4466.73	185.80	14442.93
NO	70	15	1110.00	6200.80	4220.73	204.87	11736.40
NO	100	15 (1 failed)	1099.79	4157.57	4099.00	400.57	9756.93

Figure 16. Results from tests in real road.

The success rates of the tests performed with the RPi2 platform, not only in the complete mode, but also in the simple mode, are presented in Figure 17. In the complete mode, vehicles search for NWB networks, connect, exchange messages, and disconnect, whereas, in the simple mode, vehicles search for NWB networks and extract NWBs data from their network names. As presented in Figure 17, the success rate was 100% on tests performed on the road at speeds up to 80 km/h. In the complete mode, on tests at 100 km/h, with and without a password, the success rate was 80% and 93.3%, respectively (the communication failed in two of the ten tests with a password and failed in one of the fifteen tests without password). In the simple mode, the success rate was always 100%, regardless of the vehicle speed.

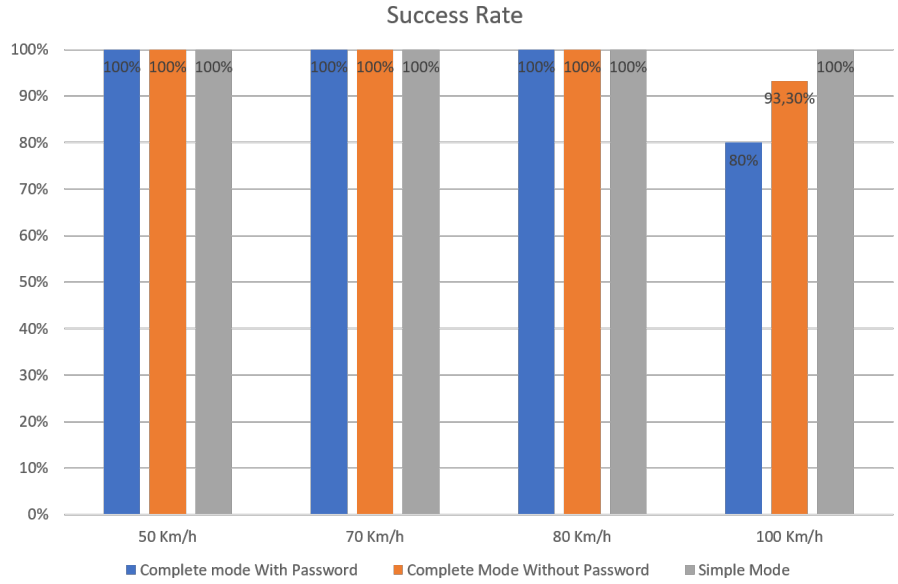


Figure 17. Test results of NOVA-WiFi-Beacons (NWBs) interacting with NWCs moving at different speeds.

Subsequently, the NWB prototype implemented in RPi2 was compared with the prototype implemented in RPi3, with regard to their interaction with vehicles. Fifteen tests were made with each prototype. The results are presented in Figure 18. With the RPi2, the average time to search for networks, connect, exchange messages, and disconnect (complete mode without including the time to obtain the GPS coordinates) was 6.101 s; whereas, the average time with RPi3 was 5.759 s. The minimum, maximum, and median times are also presented. The prototypes implemented in RPi3 had better results, as expected. The presented communication times and success rates are acceptable for low-speed vehicles, such as wheelchairs, and even for high-speed vehicles, such as cars, given that: (1) in urban areas at 50 km/h (13.9 m/s) the vehicle travels about 80 m during the communication with RPi3, which is less than the usual outdoor range of a typical Wi-Fi network operating on the 2.4 GHz band [76]; and, (2) roads signs are able to share their data with their neighbors that may provide data to vehicles at a greater distance from the point of interest. However, these results can be significantly improved if high-performance hardware platforms are used, instead of the current ones.

	Raspberry Pi 2 (RPi2)	Raspberry Pi 3 (RPi3)
Min (ms)	5674	5484
Max (ms)	8065	6040
Average (ms)	6101	5759
Median (ms)	5877	5738

Figure 18. Statistical results of the complete mode tests in RPi2 and RPi3.

The NWB was designed to handle 100 simultaneous connections from vehicles, as shown Petri net place *available_connections_i2v* from Figure 5; however, the maximum number of simultaneous connections that each NWB (for example a road sign) can handle, changes with the NWB

implementation platform. The ESP8622-01 supports a maximum of four connections at a time; whereas, Raspberry Pi 3 can handle about 15 simultaneous connections. In the NOVA-MAS deployment, more powerful Wi-Fi access points should be used, handling dozens of simultaneous connections, such as usual in Wi-Fi equipment [76].

The interaction between NWBs was also tested with success. Three NWB prototypes (NWB1, NWB2, and NWB3) were used in the tests. The NWB1 and NWB3 could find the NWB2 network and vice-versa, but the NWB1 could not find the NWB3 network, or the opposite. In the performed tests, all of the NWBs were able to directly (for instance NWB3 to NWB2) or indirectly (for instance NWB1 to NWB3 through NWB2) interact.

Finally, it is important to note that, during the tests, the NWBs were monitored and controlled though the debugger tool of the IOPT-Tools [34,35]. The debugger tool is a web-based tool that runs directly on common web browsers. The Petri net models from Figures 13 and 14 are snapshots taken from the browser during the tests. The debugger tool is not only for testing purposes, but also to remotely monitor and control systems execution.

7. Conclusions

A set of NWB prototypes, with the proposed architecture, was developed during this work while using the proposed model-driven development approach and tools, online available at http://gres.uninova.pt/IOPT-Tools/. The developed NWB prototypes interact with each other, with NWCs, and with remote control and monitoring systems. As far as we know, no other work has proposed a similar mobility assistive system and the use of design automation tools to support its development, which simultaneously: (1) enable the specification of the NWB behavior through graphical models and use them to automatically generate the simulation, validation, and implementation codes; (2) support not only NWBs simulation, but their verification through model-checking tools; and, (3) ensure that each part of the NWBs, which were specified by models, can be remotely monitored and controlled. The use of model-checking based verification tools and automatic code generators can ensure that the NWBs' implementation code conforms to certain desired proprieties, avoiding specification and manual codification errors. This is a main benefit of using the proposed approach and tools. The other major benefit of using this modeling formalism and associated tools is the rapid prototyping of NWBs and the remote operation capabilities without development effort.

The interaction between NWBs and NWCs, among NWBs, and between NWBs and RCMS was tested. In summary, at speeds below 100 km/h, all of the tests were 100% successfull, in both simple mode and complete mode, while using a password or not. When tests were carried out at 100 km/h, in simple mode there was still 100% success rate, but some flaws were detected in the complete mode. Thus, with a password, two of the ten tests failed, and without a password, one test failed in the fifteen that were performed. The interaction among NWBs and between NWBs and RCMS was made without failures.

Three different low-end computing platforms were used to implement the NWB prototypes. Among the used platforms (ESP8266-01, Raspberry Pi 2, and Raspberry Pi 3), the one with better results was the Raspberry Pi 3. The NWC prototype was implemented in an ESP8266-01. All of the used platforms had integrated antennas and, even so, supported the interaction between NWBs and NWCs moving at speeds up to 100km/h. High-performance hardware platforms with external antennas would definitely improve the results presented in Section 6.

NOVA-MAS can be seen as a complementary system to the crowdsourcing applications presented in the literature review, as well as to the existing smart wheelchairs. With this novel mobility assistive system, the public road users and vehicles (with NWCs) can share their own data (current location, speed, etc.) to infrastructures (with NWBs) and obtain, from infrastructures, updated information about their environment, which can alert users to unexpected events and prevent accidents. Neither crowdsourcing nor smart wheelchairs allow for the dissemination of local and volatile

information (information that only matters in that location at that moment) between infrastructures and their users/vehicles.

As future work, it will be of interest to extend the IOPT tool framework and study the application of other tools. The IOPT-Tools could be extended, with a new tool, to automatically generate interfaces that support the interaction between the automatically generated NWB processes, and the NWB Python processes that handle the communication. The IOPT-Flow tool framework [77], which was proposed to support and accelerate the development of cyber-physical systems, should also be considered.

Author Contributions: Conceptualization, F.M., R.C.-R., C.L.-O., E.M., B.A. and L.G.; Investigation, F.M., R.C.-R., C.L.-O., E.M., B.A. and L.G.; Methodology, F.M., R.C.-R., C.L.-O. and E.M.; Software, E.M. and B.A.; Supervision, F.M., R.C.-R. and L.G.; Validation, E.M. and B.A.; Writing—original draft, F.M., R.C.-R., C.L.-O., E.M. and L.G.; Writing—review & editing, F.M., R.C.-R., C.L.-O., B.A. and L.G. All authors have read and agreed to the published version of the manuscript.

Funding: This work was partially financed by Portuguese Agency "Fundação para a Ciência e a Tecnologia" (FCT), in the framework of project UID/EEA/00066/2020.

Conflicts of Interest: The authors declare that there is no conflict of interest regarding the publication of this paper.

References

1. WHO | Disability. 2011. Available online: https://www.who.int/disabilities/world_report/2011/report/en/ (accessed on 26 May 2020).
2. World Report on Disability. 2011. Available online: https://www.who.int/news-room/fact-sheets/detail/disability-and-health (accessed on 26 May 2020).
3. CBM | How to Make Cities Accessible and Inclusive. 2017. Available online: https://www.cbm.org/fileadmin/user_upload/Publications/How_to_make_cities_accessible_and_inclusive_Web_FINAL.PDF (accessed on 26 May 2020).
4. What Are Some Types of Assistive Devices and How Are They Used? 2018. Available online: https://www.nichd.nih.gov/health/topics/rehabtech/conditioninfo/device (accessed on 26 May 2020).
5. Assistive Technology. 2018. Available online: https://www.who.int/news-room/fact-sheets/detail/assistive-technology (accessed on 26 May 2020).
6. Moiseev, V.V.; Sudorgin, O.A.; Naberushkina, E.; Raydugin, D.S. Inclusive Properties of the City and Urban Citizenship of People with Disabilities. In Proceedings of the 2019 5th International Conference on Social Science and Higher Education (ICSSHE 2019), Xiamen, China, 23–25 August 2019; Atlantis Press: Beijing, China, 2019; pp. 136–139. [CrossRef]
7. Koontz, A.M.; Bass, S.R.; Kulich, H.R. Accessibility facilitators and barriers affecting independent wheelchair transfers in the community. *Disabil. Rehabil. Assist. Technol.* **2020**. [CrossRef] [PubMed]
8. Bækgaard, E.S. Mobility Scooter Accidents—Need for Preventative Action? *Clin. Med Rev. Case Rep.* **2017**, *4*, 1–4. [CrossRef]
9. Carlsson, A.; Lundälv, J. Acute injuries resulting from accidents involving powered mobility devices (PMDs)—Development and outcomes of PMD-related accidents in Sweden. *Traffic Inj. Prev.* **2019**, *20*, 484–491. [CrossRef] [PubMed]
10. Kraemer, J.D.; Benton, C.S. Disparities in road crash mortality among pedestrians using wheelchairs in the USA: Results of a capture–recapture analysis. *BMJ Open* **2015**, *5*. [CrossRef] [PubMed]
11. Leaman, J.; La, H.M. A Comprehensive Review of Smart Wheelchairs: Past, Present, and Future. *IEEE Trans. Hum. Mach. Syst.* **2017**, *47*, 486–499. [CrossRef]
12. Viswanathan, P.; Wang, R.; Kenyon, L.; Foley, G.; Miller, W.; Bell, J.; Kirby, R.; Simpsson, R.; Mihailidis, A.; Adams, M.; et al. Smart Wheelchairs in Assessment and Training (SWAT): State of the Field (Age-Well NCE Position Paper). In Proceedings of the 2015 IEEE 13th International Conference on Industrial Informatics (INDIN), Cambridge, UK, 22–24 July 2015.
13. Hersh, M.; Ferreira, M.G.G.; Ramirez, A.R.G. Introductory Chapter: The Role of Assistive Technologies in Smart Cities. In *Assistive Technologies in Smart Cities*; IntechOpen: London, UK, 2018; Chapter 1. [CrossRef]
14. Disability in Smart Cities: Assessing Assistive Technologies and Urban Accessibility. 2019. Available online: https://www.oxfordurbanists.com/magazine/2019/1/11/disability-in-smart-cities-assessing-assistive-technologies-and-urban-accessibility (accessed on 26 May 2020).

15. Smart Cities for All: A Vision for an Inclusive, Accessible Urban Future. 2017. Available online: https://smartcities4all.org/wp-content/uploads/2017/06/Smart-Cities-for-All-A-Vision-for-an-Inclusive-Accessible-Urban-Futur...-min.pdf (accessed on 26 May 2020).
16. Liu, Z.; Glassey, N.; Sokhn, M.; de Gaspari, E. Crowdsourcing-Based Mobile Application for Wheelchair Accessibility. *J. Technol. Pers. Disabil.* **2017**, *5*, 1–15.
17. Weld, G.; Jang, E.; Li, A.; Zeng, A.; Heimerl, K.; Froehlich, J.E. Deep Learning for Automatically Detecting Sidewalk Accessibility Problems Using Streetscape Imagery. In Proceedings of the 21st International ACM SIGACCESS Conference on Computers and Accessibility, ASSETS'19, Pittsburgh, PA, USA, 28–30 October 2019; Association for Computing Machinery: New York, NY, USA, 2019; pp. 196–209. [CrossRef]
18. Panta, Y.R.; Azam, S.; Shanmugam, B.; Yeo, K.C.; Jonkman, M.; De Boer, F.; Alazab, M. Improving Accessibility for Mobility Impaired People in Smart City using Crowdsourcing. In Proceedings of the 2019 Cybersecurity and Cyberforensics Conference (CCC), Melbourne, Australia, 8–9 May 2019; pp. 47–55.
19. Khan, U.; Lee, S. Multi-layer problems and solutions in VANETs: A review. *Electronics* **2019**, *8*, 204. [CrossRef]
20. Mullapathi Farooq, S.; Hussain, S.; Sk, K.; Ustun, T.S. Certificate based security mechanisms in vehicular Ad-Hoc networks based on IEC 61850 and IEEE WAVE standards. *Electronics* **2019**, *8*, 96. [CrossRef]
21. Talavera, E.; Anaya, J.; Gómez, O.; Alonso, F.; Naranjo, J. Performance Comparison of Geobroadcast Strategies for Winding Roads. *Electronics* **2018**, *7*, 32. [CrossRef]
22. Gomes, L.; Barros, J.P.; Costa, A.; Nunes, R. The Input-Output Place-Transition Petri Net Class and Associated Tools. In Proceedings of the 2007 5th IEEE International Conference on Industrial Informatics, Vienna, Austria, 23–27 June 2007; Volume 1, pp. 509–514. [CrossRef]
23. Pereira, F.; Moutinho, F.; Gomes, L. IOPT-tools—Towards cloud design automation of digital controllers with Petri nets. In Proceedings of the 2014 International Conference on Mechatronics and Control (ICMC), Jinzhou, China, 3–5 July 2014; pp. 2414–2419. [CrossRef]
24. Reisig, W. *Petri Nets: An Introduction*; Springer: New York, NY, USA, 1985.
25. Yaqub, O.; Li, L. Modeling and Analysis of Connected Traffic Intersections Based on Modified Binary Petri Nets. *J. Adv. Transp.* **2013**, *2013*. [CrossRef]
26. Riouali, Y.; Benhlima, L.; Bah, S. Extended Batches Petri Nets Based System for Road Traffic Management in WSNs. *J. Sens. Actuator Netw.* **2017**, *6*. [CrossRef]
27. Vázquez, C.R.; Sutarto, H.Y.; Boel, R.; Silva, M. Hybrid Petri net model of a traffic intersection in an urban network. In Proceedings of the 2010 IEEE International Conference on Control Applications, Yokohama, Japan, 8–10 September 2010; pp. 658–664. [CrossRef]
28. Wang, J.; Yan, J.; Li, L. Microscopic Modeling of a Signalized Traffic Intersection Using Timed Petri Nets. *IEEE Trans. Intell. Transp. Syst.* **2016**, *17*, 305–312. [CrossRef]
29. Ahmane, M.; Abbas-Turki, A.; Perronnet, F.; Wu, J.; Moudni, A.E.; Buisson, J.; Zeo, R. Modeling and controlling an isolated urban intersection based on cooperative vehicles. *Transp. Res. Part Emerg. Technol.* **2013**, *28*, 44–62. [CrossRef]
30. Ng, K.M.; Reaz, M.B.I.; Ali, M.A.M. A Review on the Applications of Petri Nets in Modeling, Analysis, and Control of Urban Traffic. *IEEE Trans. Intell. Transp. Syst.* **2013**, *14*, 858–870. [CrossRef]
31. Kombe, T.; Ele, P.; Offole, F.; Miasse, H. Modelling an Interactive Road Signs System, Using Petri Nets. *Transp. Telecommun. J.* **2017**, *18*. [CrossRef]
32. Pereira, F.; Moutinho, F.; Gomes, L. Model-checking framework for embedded systems controllers development using IOPT Petri nets. In Proceedings of the 2012 IEEE International Symposium on Industrial Electronics, Hangzhou, China, 28–31 May 2012; pp. 1399–1404. [CrossRef]
33. Campos-Rebelo, R.; Pereira, F.; Moutinho, F.; Gomes, L. From IOPT Petri nets to C: An automatic code generator tool. In Proceedings of the 2011 9th IEEE International Conference on Industrial Informatics, Lisbon, Portugal, 26–29 July 2011; pp. 390–395. [CrossRef]
34. Pereira, F.; Melo, A.; Gomes, L. Remote operation of embedded controllers designed using IOPT Petri-nets. In Proceedings of the 2015 IEEE 13th International Conference on Industrial Informatics (INDIN), Cambridge, UK, 22–24 July 2015; pp. 572–579. [CrossRef]

35. Pereira, F.; Gomes, L. A JSON/HTTP communication protocol to support the development of distributed cyber-physical systems. In Proceedings of the 2018 IEEE 16th International Conference on Industrial Informatics (INDIN), Porto, Portugal, 18–20 July 2018; pp. 23–30. [CrossRef]
36. Girault, C.; Valk, R. *Petri Nets for Systems Engineering: A Guide to Modeling, Verification, and Applications*; Springer: Berlin/Heidelberg, Germany, 2002.
37. Reisig, W. *Understanding Petri Nets: Modeling Techniques, Analysis Methods, Case Studies*; Springer: Berlin/Heidelberg, Germany, 2013.
38. Hanisch, H.M.; Lüder, A. A signal extension for Petri nets and its use in controller design. *Fundam. Inform.* **2000**, *41*, 415–431. [CrossRef]
39. Reisig, W. *A Primer in Petri Net Design*; Springer: Berlin/Heidelberg, Germany, 1992.
40. Khaddar, M.A.E.; Boulmalf, M. Smartphone: The Ultimate IoT and IoE Device. In *Smartphones from an Applied Research Perspective*; Mohamudally, N., Ed.; IntechOpen: Rijeka, Croatia, 2017; Chapter 7. [CrossRef]
41. Spachos, P.; Plataniotis, K. BLE Beacons in the Smart City: Applications, Challenges, and Research Opportunities. *IEEE Internet Things Mag.* **2020**, *3*, 14–18. [CrossRef]
42. Harriehausen-Mühlbauer, B.; Roth, J. WheelScout—Barrier-Free Navigation. In *Proceedings of SAI Intelligent Systems Conference (IntelliSys) 2016*; Springer: Cham, Switzerland, 2018; pp. 1056–1063. [CrossRef]
43. Bassoli, M.; Bianchi, V.; Munari, I.D. A Plug and Play IoT Wi-Fi Smart Home System for Human Monitoring. *Electronics* **2018**, *7*, 200. [CrossRef]
44. Hlaing, W.; Thepphaeng, S.; Nontaboot, V.; Tangsunantham, N.; Sangsuwan, T.; Pira, C. Implementation of WiFi-based single phase smart meter for Internet of Things (IoT). In Proceedings of the 2017 International Electrical Engineering Congress (iEECON), Pattaya, Thailand, 8–10 March 2017; pp. 1–4.
45. Walia, N.K.; Kalra, P.; Mehrotra, D. An IOT by information retrieval approach: Smart lights controlled using WiFi. In Proceedings of the 2016 6th International Conference—Cloud System and Big Data Engineering (Confluence), Noida, India, 14–15 January 2016; pp. 708–712.
46. Shi, C.; Liu, J.; Liu, H.; Chen, Y. Smart User Authentication through Actuation of Daily Activities Leveraging WiFi-Enabled IoT. In Proceedings of the 18th ACM International Symposium on Mobile Ad Hoc Networking and Computing, Mobihoc'17, Chennai, India, 10–14 July 2017; Association for Computing Machinery: New York, NY, USA, 2017. [CrossRef]
47. Suryotrisongko, H.; Kusuma, R.C.; Ginardi, R.H. Four-Hospitality: Friendly Smart City Design for Disability. *Procedia Comput. Sci.* **2017**, *124*, 615–623. [CrossRef]
48. Mobasheri, A.; Deister, J.; Dieterich, H. Wheelmap: The wheelchair accessibility crowdsourcing platform. *Open Geospat. Data Softw. Stand.* **2017**, *2*, 1–15. [CrossRef]
49. Menkens, C.; Sussmann, J.; Al-Ali, M.; Breitsameter, E.; Frtunik, J.; Nendel, T.; Schneiderbauer, T. EasyWheel—A Mobile Social Navigation and Support System for Wheelchair Users. In Proceedings of the 2011 Eighth International Conference on Information Technology: New Generations, Las Vegas, NV, USA, 1–13 April 2011; pp. 859–866.
50. Mirri, S.; Prandi, C.; Salomoni, P. Personalizing Pedestrian Accessible way-finding with mPASS. In Proceedings of the 2016 13th IEEE Annual Consumer Communications Networking Conference (CCNC), Las Vegas, NV, USA, 9–12 January 2016; pp. 1119–1124.
51. Edinger, J.; Hofmann, A.; Wachner, A.; Becker, C.; Raychoudhury, V.; Krupitzer, C. WheelShare: Crowd-Sensed Surface Classification for Accessible Routing. In Proceedings of the 2019 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops), Kyoto, Japan, 11–15 March 2019; pp. 584–589.
52. Mourcou, Q.; Fleury, A.; Dupuy, P.; Diot, B.; Franco, C.; Vuillerme, N. Wegoto: A Smartphone-based approach to assess and improve accessibility for wheelchair users. In Proceedings of the 2013 35th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), Osaka, Japan, 3–7 July 2013; pp. 1194–1197.
53. Civitarese, G.; Mascetti, S.; Butifar, A.; Bettini, C. Automatic Detection of Urban Features from Wheelchair Users' Movements. In Proceedings of the 2019 IEEE International Conference on Pervasive Computing and Communications (PerCom), Kyoto, Japan, 11–17 March 2019; pp. 1–10.
54. Mascetti, S.; Civitarese, G.; El Malak, O.; Bettin, C. SmartWheels: Detecting urban features for wheelchair users' navigation. *Pervasive Mob. Comput.* **2020**, *62*, 101115. [CrossRef]

55. Dias, J.; Rodrigues, J.; Soares, V.; Caldeira, J.; Korotaev, V.; Proenca, M., Jr. Network Management and Monitoring Solutions for Vehicular Networks: A Survey. *Electronics* **2020**, *9*, 853. [CrossRef]
56. Botte, M.; Pariota, L.; D'Acierno, L.; Bifulco, G.N. An Overview of Cooperative Driving in the European Union: Policies and Practices. *Electronics* **2019**, *8*, 616. [CrossRef]
57. Zambrano-Martinez, J.; Calafate, C.; Soler, D.; Lemus, L.; Cano, J.C.; Manzoni, P.; Gayraud, T. A Centralized Route-Management Solution for Autonomous Vehicles in Urban Areas. *Electronics* **2019**, *8*, 722. [CrossRef]
58. Liu, W.; He, X.; Huang, Z.; Ji, Y. Transmission Capacity Characterization in VANETs with Enhanced Distributed Channel Access. *Electronics* **2019**, *8*, 340. [CrossRef]
59. EUROPA—Cooperative Vehicle-Infrastructure Systems | TRIMIS—European Commission. 2015. Available online: https://trimis.ec.europa.eu/project/cooperative-vehicle-infrastructure-systems (accessed on 26 May 2020).
60. EUROPA—Co-Operative Networks for Intelligent Road Safety | TRIMIS—European Commission. 2013. Available online: https://trimis.ec.europa.eu/project/co-operative-networks-intelligent-road-safety (accessed on 26 May 2020).
61. EUROPA—Cooperative Systems for Road Safety | TRIMIS—European Commission. 2015. Available online: https://trimis.ec.europa.eu/project/cooperative-systems-road-safety (accessed on 26 May 2020).
62. Toulminet, G.; Boussuge, J.; Laurgeau, C. Comparative synthesis of the 3 main European projects dealing with Cooperative Systems (CVIS, SAFESPOT and COOPERS) and description of COOPERS Demonstration Site 4. In Proceedings of the 2008 11th International IEEE Conference on Intelligent Transportation Systems, Beijing, China, 12–15 October 2008; pp. 809–814.
63. ISO/TC 204—Intelligent Transport Systems. 2017. Available online: https://www.iso.org/committee/54706.html (accessed on 26 May 2020).
64. Erskine, S.K.; Elleithy, K.M. Real-Time Detection of DoS Attacks in IEEE 802.11p Using Fog Computing for a Secure Intelligent Vehicular Network. *Electronics* **2019**, *8*, 776. [CrossRef]
65. Ding, F.; Ma, Z.; Li, Z.; Su, R.; Zhang, D.; Zhu, H. A Terminal-Oriented Distributed Traffic Flow Splitting Strategy for Multi-Service of V2X Networks. *Electronics* **2019**, *8*, 644. [CrossRef]
66. Du, D.; Xin, J.; Wu, X.; Tan, Y.; Zeng, X.; Huang, S.; Li, Y. 3D Spatial Characteristics of C-V2X Communication Interference. *Electronics* **2019**, *8*, 718. [CrossRef]
67. Intelligent Transport Systems (ITS). *ITS-G5 Access Layer Specification for Intelligent Transport Systems Operating in the 5 GHz Frequency Band*; European Telecommunications Standards Institute: Sophia Antipolis, France, 2020. Available online: https://www.etsi.org/deliver/etsi_en/302600_302699/302663/01.02.00_20/en_302663v010200a.pdf (accessed on 26 May 2020).
68. Fernández-Isabel, A.; Fuentes-Fernández, R. Analysis of Intelligent Transportation Systems Using Model-Driven Simulations. *Sensors* **2015**, *15*, 14116–14141. [CrossRef]
69. Flammini, F.; Marrone, S.; Mazzocca, N.; Nardone, R.; Vittorini, V. Model-Driven V&V Processes for Computer Based Control Systems: A Unifying Perspective. In *Leveraging Applications of Formal Methods, Verification and Validation. Applications and Case Studies*; Margaria, T., Steffen, B., Eds.; Springer: Berlin/Heidelberg, Germany, 2012; pp. 190–204.
70. Scippacercola, F.; Pietrantuono, R.; Russo, S.; Zentai, A. Model-driven engineering of a railway interlocking system. In Proceedings of the 2015 3rd International Conference on Model-Driven Engineering and Software Development (MODELSWARD), Angers, France, 9–11 February 2015; pp. 509–519.
71. Esp-01 WIFI Module. 2015. Available online: https://ecksteinimg.de/Datasheet/Ai-thinker%20ESP-01%20EN.pdf (accessed on 26 May 2020).
72. Raspberry Pi. 2019. Available online: https://www.raspberrypi.org/ (accessed on 26 May 2020).
73. Welcome to ESP8266 Arduino Core's Documentation! 2017. Available online: https://arduino-esp8266.readthedocs.io/en/latest/ (accessed on 26 May 2020).
74. Wiki-batman-adv-Open Mesh. 2018. Available online: https://www.open-mesh.org/projects/batman-adv/wiki/Wiki (accessed on 26 May 2020).
75. The Python Standard Library. 2018. Available online: https://docs.python.org/3/library/index.html (accessed on 26 May 2020).

76. Banerji, S.; SinghaChowdhury, R. Wi-Fi & Wi-MAX: A Comparative Study. *Indian J. Eng.* **2013**, *2*, 1–5.
77. Pereira, F.; Gomes, L. The IOPT-Flow framework pairing Petri nets and data-flows for embedded controller development. In Proceedings of the IECON 2016—42nd Annual Conference of the IEEE Industrial Electronics Society, Florence, Italy, 24–27 October 2016; pp. 4832–4837. [CrossRef]