Since January 2020 Elsevier has created a COVID-19 resource centre with free information in English and Mandarin on the novel coronavirus COVID-19. The COVID-19 resource centre is hosted on Elsevier Connect, the company's public news and information website.

Elsevier hereby grants permission to make all its COVID-19-related research that is available on the COVID-19 resource centre - including this research content - immediately available in PubMed Central and other publicly funded repositories, such as the WHO COVID database with rights for unrestricted research re-use and analyses in any form or by any means with acknowledgement of the original source. These permissions are granted for free by Elsevier for as long as the COVID-19 resource centre remains active.
PITS: An Intelligent Transportation System in pandemic times

Enrique Brazález\(^{a,b}\), Hermenegilda Macià\(^{b,c}\), Gregorio Díaz\(^{a,b}\), Valentín Valero\(^{a,b}\), Juan Boubeta-Puig\(^{d,*}\)

\(^{a}\)Department of Computer Science, Universidad de Castilla-La Mancha, Campus Univ. s/n, Albacete, 02071, Castilla-La Mancha, Spain
\(^{b}\)Albacete Research Institute of Informatics, Universidad de Castilla-La Mancha, Campus Univ. s/n, Albacete, 02071, Castilla-La Mancha, Spain
\(^{c}\)Department of Mathematics, Universidad de Castilla-La Mancha, Campus Univ. s/n, Albacete, 02071, Castilla-La Mancha, Spain
\(^{d}\)Department of Computer Science and Engineering, University of Cadiz, Avda. de la Universidad de Cádiz 10, Puerto Real, 11519, Cádiz, Spain

**A R T I C L E I N F O**

**Abstract**

The control of the pandemic caused by SARS-CoV-2 is a challenge for governments all around the globe. To manage this situation, countries have adopted a bundle of measures, including restrictions to population mobility. As a consequence, drivers face with the problem of obtaining fast routes to reach their destinations.

In this context, some recent works combine Intelligent Transportation Systems (ITS) with big data processing technologies taking the traffic information into account. However, there are no proposals able to gather the COVID-19 health information, assist in the decision-making process, and compute fast routes in an all-in-one solution. In this paper, we propose a Pandemic Intelligent Transportation System (PITS) based on Complex Event Processing (CEP), Fuzzy Logic (FL) and Colored Petri Nets (CPNs). CEP is used to process the COVID-19 health indicators and FL to provide recommendations about city areas that should not be crossed. CPNs are then used to create map models of health areas with the mobility restriction information and obtain fast routes for drivers to reach their destinations. The application of PITS to Madrid region (Spain) demonstrates that this system provides support for authorities in the decision-making process about mobility restrictions and obtain fast routes for drivers. PITS is a versatile proposal which can easily be adapted to other scenarios in order to tackle different emergency situations.

**Keywords:**

Colored Petri net
Complex event processing
Fuzzy logic
Intelligent transportation system
Pandemic

**1. Introduction**

Every pandemic is unique and can cause different problems in the society and economy, so managing a pandemic is a challenge for authorities and governments around the world. Intelligent Transportation Systems (ITSs) can be helpful for mitigating the effects of a pandemic, since ITSs allow us to make decisions under dynamic changes, uncertainty and limited knowledge. Thus, managing ITSs in pandemic times requires rich information and decision support technologies (Trivedi and Zulkernine, 2020).

Scientific evidences have revealed that the transmission of COVID-19 is related to population mobility. For instance, Kraemer et al. (2020) and Chinazzi et al. (2020) discovered that the drastic measures taken by the Chinese government substantially mitigated the spread of COVID-19. These measures were mainly based on prohibiting the population mobility, among others (Wu et al., 2021). Badr et al. (2020) found a correlation between COVID-19 and mobility patterns in a study performed in USA. In addition, mobility restrictions seriously affect economy as showed by Bonaccorsi et al. (2020). Therefore, a proper balance in the restrictions according to the advance of COVID-19 must be considered, as shown by Gargoum and Gargoum (2021) and by Singh et al. (2021).

The proposed restrictions applied by the different governments at local, national and international level are limited to specific areas to mitigate the negative aspects. The size and population of these areas vary depending on each country and the current health situation shown by their health indicators. Currently, one of the main challenges is to dynamically compute fast routes connecting different areas taking mobility restrictions into account. To address this challenge, in this paper we propose a novel ITS, called PITS, which integrates Complex Event Processing (CEP) (Luckham, 2001), Fuzzy Logic (Zadeh, 1965) and Colored Petri Nets (CPNs) (Jensen and Kristensen, 2009) to provide support for decisions about mobility restrictions and compute fast routes in pandemic times. Remarkably, these technologies are supported by a bundle of software tools which allow us to automatize the whole process, from the system development to the analysis of properties and the decision-making process. In this way, PITS gathers automatically all the health indicators to analyze the current situation of a city. Based on this information, an expert system integrated in PITS

---

\(^{*}\) Corresponding author.

**E-mail addresses:** enrique.brazalez@uclm.es (E. Brazález), hermenegilda.macia@uclm.es (H. Macià), gregorio.diaz@uclm.es (G. Díaz), valentin.valero@uclm.es (V. Valero), juan.boubeta@uca.es (J. Boubeta-Puig).

https://doi.org/10.1016/j.engappai.2022.105154
Received 2 April 2022; Received in revised form 10 June 2022; Accepted 28 June 2022
Available online 8 July 2022
0952-1976/© 2022 The Author(s). Published by Elsevier Ltd. This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/).
helps domain experts to make decisions about the restrictions to be applied in the health areas. Afterwards, fast routes are automatically computed based on these restrictions.

In this system, the CEP technology allows us to analyze and correlate real-time information on each area to automatically provide a set of indicators associated with the COVID-19 transmission level. Since this real-time information could be imprecise, we combine CEP with a Fuzzy approach. Therefore, a two-step process determines the mobility restrictions: (1) an expert system, designed as a Fuzzy Inference System (FIS) (Mamdani and Assilian, 1975), allows us to manage the linguistic variables representing the health risk levels based on such indicators, and (2) a domain expert determines the actions to be taken considering the health risk level obtained. These risk levels should be considered in a dynamic model able to capture the changing restrictions. We propose the use of a formal model to provide mathematical rigor. In this case, a CPN model is able to capture this dynamic behavior and this formalism is supported by several automatic tools that can be used to compute fast routes using real-time techniques, such as the simulation. In addition, the graphic nature of CPNs allows us to have a global and intuitive view of the system and the capability to study different scenarios automatically.

Therefore, the research objectives of this work are:

- To develop an ITS based on the integration of CEP, CPN and fuzzy logic with the aim of addressing pandemic situations, which should be easily adaptable to other situations.
- To provide a set of event patterns to automatically detect risk situations in real time.
- To design a FIS to support domain experts in the decision-making process to adopt an alert level.
- To create a CPN model to compute routes considering which areas are open and closed according to the current health circumstances.

The structure of the paper is as follows. Related works are discussed in Section 2. Section 3 presents a description of the COVID-19 situation in Madrid (Spain), which is the context of this ITS, considering health indicators and alert levels. A background on CEP, fuzzy logic and PNs is presented in Section 4. Section 5 describes the proposal, and Section 6 draws conclusions and future work.

2. Related work

Recurrent problems such as traffic accidents, air and noise pollution and long travel times are in the scope of the ITS (Picone et al., 2015), a cutting-edge area of study whose main aim is to optimize road traffic according to the associated problem. In this way, ITS plays a key role for both improving human life and modern economy (Bekiaris and Nakanishi, 2004). For instance, Lakshmi et al. (2022) focused on Collaborative-ITS (C-ITS), working on intelligent deep learning, based on short-term traffic flow prediction. The objective of this work was to assist in the optimization of signal timing by traffic signal controllers and route adaptation for travelers.

The use of fuzzy logic in the design of ITS has been studied for decades. Palacharla and Nelson (1999) presented a fuzzy reasoning model for dynamic travel time estimation. Naranjo et al. (2003) used logic fuzzy to present an ACC (Adaptive Cruise Control) for inter-vehicle gap keeping. Fuzzy logic is also used by Montero et al. (2012) in controllers for collision avoidance or by Alzamami and Mahgoub (2019) in Vehicular Ad hoc NETworks (VANETs). More recently, Chen et al. (2020) proposed a fuzzy Markov prediction model with the aim of estimating the short-term traffic conditions in VCPS (Vehicular Cyber-Physical Systems) in urban environment. Moreover, the use of fuzzy logic with Neural Networks, known as FNNs (Fuzzy Neural Networks), for the design of ITS has been extended. For instance, Yin et al. (2002) and Quek et al. (2006) applied FNNs to predict the urban traffic flow.

Fuzzy logic has also been used with other formal methods. For instance, Placzek (2013) proposed a traffic model based on fuzzy cellular automata. In the field of eHealth, Calvo et al. (2019) presented a methodology to analyze heart data using fuzzy automata.

Modeling ITS by using PN is not new. Cavone et al. (2018) presented a survey with contributions of PNs and extensions of PNs in the development of ITSs. Ng et al. (2013) reviewed the application of PNs in the scope of urban traffic control, and Qi et al. (2016) showed the use of PNs to model a real-time traffic emergency system for intersections to prevent accidents. CPNs were used by Huang and Chung (2008) to model traffic lights, by Zhou et al. (2013) to control traffic flow, and by Bentaher et al. (2019) in VANETs, who used Petri Nets Vehicular Communication protocol (PNVC). In the field of mobile robot navigation, Kumar et al. (2019) proposed a control approach based on Petri Nets to avoid intercollision among multiple humanoids during their navigation.

To address road traffic optimization, the integration of ITS with the CEP technology is appropriate, as demonstrated the works by Terrososaenz et al. (2012), Eswaran et al. (2015) and Figueiras et al. (2018), due to CEP’s ability to correlate and analyze several thousands of data in real time. In this research line, our preliminary results (Díaz et al., 2020; Díaz et al., 2017), with the aim of controlling air quality in cities, not only integrated an ITS with CEP, but also with CPNs. On the other hand, integrating only CEP and Petri Nets, in Macià et al. (2016), Boubeta-Puig et al. (2019) and Valero et al. (in press) we used Petri Nets formalism as semantic validation of event patterns defined in EPL.

Since the purpose of the present work is to control the spread of COVID-19 in a determined geographic area, we extended our mentioned ITS to control air quality by using fuzzy logic. This fuzzy ITS continues being flexible but with the capability to capture important information to make decisions about traffic regulations. More specifically, this information about health parameters is gathered and processed using a CEP engine. A FIS is then used to help in making decisions about when and how to restrict the traffic in the affected areas. Afterwards, the CPN model allows us to calculate and re-calculate the traffic routes.

With respect to other works tackling different emergency situations using ITS, Ye et al. (2010) evaluated the use of rural transportation infrastructure in evacuation operations under hurricanes. Park et al. (2009) used an Advanced Traveler Management and Information System (ATMIS) as an emergency evacuation system under tsunami, and Chang (2010) presented the available ITS solutions to overcome climate change risks. The main difference with our work is the all-in-one solution based on CEP, fuzzy logic and CPNs, being CEP recently applied to emergency management in the presence of disasters as in Luthra et al. (2019), in airport use cases as in Mljovic et al. (2019) and in nuclear disaster scenarios (Barthe-Delanoë et al., 2018) as in the Fukushima Daiichi nuclear plant.

In the context of the COVID-19 pandemic, Hendrickson and Rilett (2020) indicated several research areas to mitigate the negative effects of this pandemic, such as the use of activity-based travel information as an analysis tool and the complementariness of the transportation and public health systems with the goal of making transportation systems resilient, efficient, reliable, and sustainable in pandemic times. For instance, Bian et al. (2021) studied time lag effects of COVID-19 policies on transportation systems, including a comparative study of New York City and Seattle.

3. COVID-19 in Madrid

In this paper, the geographic area of the Madrid region (Spain) is studied as a proof of concept, but other geographic areas could easily be considered as well. In Madrid, we have considered seven health areas as illustrated in Fig. 1.
Table 1 shows the indicators used to assess health risk levels, which have been obtained from this document. This table is divided into two sections. Section 1 contains the transmission indicators used to evaluate the transmission level in a certain area, while Section 2 contains the public health capacity indicators used to evaluate the occupation level of the assistance services offered by the public healthcare system in that area. Thus, the first section includes the cumulative incidence of COVID-19 cases diagnosed in the last 14 and 7 days for either general population and people aged 65 and over, the percentage of active infection diagnostic tests (PCR and antigen tests) and the percentage of traceable cases, i.e. positive cases linked to a previous patient. In the second section, we have the percentage of hospital beds occupied by patients with COVID-19 and percentage of Intensive Care Unit (ICU) beds occupied by COVID-19 patients.

The assessment of the health risk level is performed by a group of domain experts. The assessment is performed based on this information, but other information can also be considered, such as the proximity of a vacation period – to avoid an increase in population mobility – and a sudden upwards trend in COVID-19 cases. General alert levels, provided in this document, are based on these indicators. There are four alert levels – from 1 to 4 – when at least two indicators from SECTION I and one from SECTION II are: low (alert level 1), medium (alert level 2), high (alert level 3), and very high (alert level 4).

The action protocol defines a set of actions to restrict the mobility according to the alert level: for 1 and 2, recommendation to limit non-essential travels outside the territorial area; for 3, recommendation to the population to leave their home only when necessary, considering restrictions to coming in and out of the territorial area, except for essential activities; for 4, exceptional measures, including the request of the state of alarm, which allows the Government to restrict the fundamental rights of people.

4. Background

This section explains the background on CEP used for defining the event patterns related to COVID-19 indicators, the FIS used to determine risk levels and the CPN formalism.

4.1. Complex event processing

CEP (Luckham, 2012; Ramírez et al., 2021) is a powerful technology for detecting relevant situations in real time through the analysis and correlation of huge amounts of incoming data. These data can be obtained from several data producers, such as sensors and IoT devices (Ortiz et al., 2022).

An event can be described as anything that happens or could happen. A simple event occurs at a point in time, while a complex event is a result of a set of other simple or complex events. From simple or complex events, higher complex events can be derived by matching...
event patterns, which specify the conditions to be satisfied for detecting such relevant situations in real time.

A **CEP engine** is the software needed to automatically detect these event patterns and, subsequently, to generate their associated complex events, which will be sent to the interested data consumers, such as actuators and dashboards.

Fig. 2 illustrates the stages of the CEP technology: (1) the event reception, which consists of obtaining the simple events to be analyzed from an application domain, in that case, health data; (2) the event processing, in which these simple events are analyzed and correlated by the CEP engine, which generates complex events (health indicators) when an event pattern is matched; and (3) the event consumption, where the consumers will then make use of the complex events received.

**Event Processing Languages** (EPLs) are languages tailored for implementing event patterns (Boubeta-Puig et al., 2015). A particular EPL is provided by each CEP engine. As an example, Esper EPL is the language supported by the Esper CEP engine (EsperTech, 2022), one of the most known and widely used open source CEP engines.

### 4.2. Fuzzy Inference System

Fuzzy logic is an extension of classical Boolean logic in the form of a many-valued logic. It can be considered as an approach to the computing based on “degrees of truth”. The first author who introduced this term through fuzzy sets was (Zadeh, 1965). A fuzzy set is a class of objects that uses a membership function which assigns a grade of membership to each object (a real value), ranging between zero and one. In contrast, in classical set theory, the grade of membership to each object can only be 0 or 1. A fuzzy set is then a set of objects with non-well defined boundaries. Fuzzy logic can be used as a mathematical framework to model the uncertainty of human cognitive processes. Furthermore, a fuzzy logic inference system can be used to make decisions, which consists of a collection of fuzzy if-then rules performing logical operations on fuzzy sets, thus providing an inference mechanism that simulates human reasoning procedures in knowledge-based systems.

More specifically, a Fuzzy Inference System (FIS) is a system that uses fuzzy set theory to map inputs (features in the case of fuzzy classification) to outputs (classes in the case of fuzzy classification). This mapping provides a basis from which decisions can be made. Mamdani’s fuzzy inference method (Mamdani and Assilian, 1975) is one of the most commonly FIS used. This method involves the following steps to define a FIS (see Fig. 3): (1) definition of fuzzy sets (fuzzification of each input and output variable), (2) IF-THEN rules (knowledge base), (3) aggregation of the consequent by applying the rule (inference engine), and (4) defuzzification of the output.

There are toolboxes based on Phyton, such as Skfuzzy (Oliphant and Jones, 2020), which facilitate the implementation of a FIS, by implementing such third and fourth steps. By using Skfuzzy, for instance, we only need to define the input/output sets and the if-then rules.

### 4.3. Petri Nets

PNs are a widely used formalism for the modeling and simulation of systems in different domains, because of their mathematical rigour and the graphical representation of the models. Next, PNs are described using a city map as a running example. More detailed information of PNs and CPNs can be found in Peterson (1981) and Jensen and Kristensen (2009), respectively.

A PN is a directed bipartite graph that has two types of node: places (depicted as circles) and transitions (depicted as rectangles). Places and Transitions can be connected by directed arcs, either place-transition (PT) or transition-place (TP) arcs (see Fig. 4). All places $p$ connected with a transition $t$ by an arc from $p$ to $t$ are called the preconditions of $t$, whereas those from $t$ to $p$ are the postconditions places. Places usually represent states or system conditions while transitions are the actions or events that produce changes in the system state. In Fig. 4, places represent the geographic locations associated to the health areas shown in Fig. 1 and transitions represent the actions to move to adjacent areas. Places are annotated by a set of tokens, which represent system states. The number of tokens on each place is called the marking of this place, and the marking of the PN is defined as the marking of all its places. For instance, in the PN model shown in Fig. 4, one token on a place indicates that there is a object/vehicle/person currently at that location.

A transition $t$ is enabled and, thus, can be fired when all its pre-condition places have at least one token at the current marking, and with the firing of $t$ one token is removed from each pre-condition place, and one token is written into each post-condition place of $t$. In Fig. 4, the current marking of place $Nn3$ is one token (depicted in green in the right-hand side of the place), and this place had initially one token (number above the current marking). The transitions enabled at the marking indicated in the figure are depicted in green. These transitions represent the possible movements from place $Nn3$. For instance, the firing of transition $Ntn0W$ removes the token from $Nn3$ and adds a new token in place $Wn2$. Thus, starting from an initial marking, with all the possible sequences of transitions that can be fired, we obtain the so-called reachable markings of PN. The set of all reachable markings from an initial marking is called the state space. Fig. 5 shows the state space of the PN of Fig. 4, as obtained from CPN Tools (Westergaard and Verbeek, 2022). The state space is in general infinite, but in the case of bounded PNs (the number of tokens on each place is bounded) this graph will be finite. However, even when the graph is finite, in many cases it will have a huge size, so different strategies can be applied to reduce the exploration, as explained in Díaz et al. (2020), or we can use simulation techniques for the analysis.

PNs can be extended in several ways. CPNs (Jensen, 1995) is a widely used extension, in which the basic model is extended considering tokens with information (data) and time. In this paper, we use CPN Tools to provide support for the creation, edition, simulation and analysis of CPNs. Thus, this paper makes use of CPN Tools notation.

In CPNs, a place has an associated color set (a data type), which specifies the set of allowed token colors at this place (values). Places are graphically annotated with their corresponding color sets, which are indicated in the bottom right-hand side of them. Tokens can also have a timestamp associated, indicating the time at which they will be
available in order to fire some transitions. Timed tokens are written as $n^{val}@t$, where $n$ indicates the number of tokens we have with value $val$ and timestamp $t$. Symbol ++ is used for the union of untimed multisets, whereas +++ is used for the union of timed multisets. Thus, a timed marking such as $1'5@10++3'2@20$ indicates that we have 1 token with value 5 and timestamp 10 and 3 tokens with value 2 and timestamp 20.

Fig. 6 shows a CPN model with 2 zones interconnected. Each zone is represented by a place, both with the same color set $\text{INTlist}$ (a timed list of integers), representing the zones that have been traversed by a car. Place $Xn1$ has $1'[1]@0++2'[1]@10$ as initial marking, $Yn2$ is unmarked initially. Transitions $XtoY$ and $YtoX$ represent the movements, so a vehicle is represented by one token, which moves from one place to another. Both transitions have a guard to check that the destination zone number is not already in the list (see the predicates in brackets). Guards are used to restrict the firing of transitions, they can only be fired when their associated guards are evaluated as true under the binding used. In particular, the binding used for the firing of transition $XtoY$ is shown in the gray rectangle ($x = [1]$), i.e. variable $x$ is assigned to the list [1], which is the value of the token binded in the $Xn1$ place.
Transitions can have a priority associated, which will be used in the event of a conflict. Arcs have inscriptions, which are expressions to produce multisets of colored tokens. In the case of pt-arcs these are usually variables that are bound to the token values in their preconditions, while for tp-arcs they are used to produce the tokens on the output places. As an illustration, variable \( x \) in the figure will be bound to one token in place \( Xn1 \), so transition \( XtoY \) will only fire if zone 2 is not in the list \( x \), producing the list \( x \) extended with 2 in the \( Yn2 \) place. Transitions can have a delay associated (inscription \( \oplus t \)), so the tokens produced by their firing are delayed by the time indicated \( (t) \).

In this CPN, transition \( XtoY \) can be fired at time 0, and the marking obtained with this firing is that shown in green in the places of Fig. 6. At this marking only transition \( XtoY \) can be fired, because the guard of transition \( YtoX \) is false. A second firing of transition \( XtoY \) at time 10 leads to the marking \( 1' [1,2] @0++2' [1] @10 \) in the \( Yn2 \) place.

5. ITS to control mobility

This section presents our ITS, called PITS, for facilitating the decision-making process on mobility restrictions during pandemic times. This is a novel ITS which integrates CEP, fuzzy logic and CPN as explained through this section.

In our system, we consider the mobility restrictions shown in Table 2 for the five level alerts considered depending on the risk assessments provided by the domain expert. Each Spanish regional government makes its own interpretation of the recommendations described in Section 3 for the different alert levels. In this paper, we have considered a restrictive interpretation of them. As an example, the proposal in that section recommends to limit the mobility for level 2, while we propose to close the mobility during the curfew hours. Thus, the proposal considered in the present work extends the measures indicated in Section 3 in order to reduce the SARS-CoV-2 spread.

Fig. 7 depicts an overview of our proposal which consists of six steps. In step 1, we obtain data from several health areas. In step 2, these data are processed and correlated by the CEP engine, which is responsible for creating the complex events to feed the FIS system. Next, in step 3, the domain expert interprets the alert levels provided by the FIS system, emitting a set of alert levels for the different areas. In step 4, these values are then used in the CPN model to apply the mobility restrictions. Thereby, when a car driver asks for a fast route, in step 5, the CPN model simulation will provide this driver with the route solution in the last step 6. For this purpose, we simulate several cars with the same origin and the same destination, and the route for the fastest one is returned. We remark that the route thus provided will not necessarily be the optimal one, because this depends on the number of initial cars and the number of possible routes.

| Table 2 | Mobility restrictions depending on the risk assessment. |
|---------|-------------------------------------------------------|
| Alert level | Risk assessment | Mobility restrictions |
| 0 | New normality | No mobility restrictions |
| 1 | Low risk | Issue warning |
| 2 | Medium risk | Mobility closed during curfew hours |
| 3 | High risk | Mobility closed all day during weekends and closed during curfew hours on weekdays |
| 4 | Extreme risk | Mobility closed all day |

5.1. Pattern modeling and automatic implementation

Next, a set of event patterns to automatically compute the risk indicators specified in Table 1 is modeled and automatically transformed into code thanks to the use of the MEdit4CEP tool (Boubeta-Puig et al., 2015; Boubeta-Puig et al., 2019).

Fig. 8 shows the CEP domain modeled with the MEdit4CEP tool. This domain consists of two simple event types: AreaBedStatus and AreaTestStatus. AreaBedStatus is used to capture the information about the available hospital beds in a health area, and AreaTestStatus captures the information about the COVID-19 tests performed in a health area. We assume that the information for both event types is registered daily for each health area.

More specifically, AreaBedStatus has the following fields: a timestamp indicating the event date, the health area, number of occupied hospital beds, number of occupied ICU beds, number of beds with COVID-19 patients, and number of ICU beds with COVID-19 patients. AreaTestStatus has the following fields: timestamp, health area, total number of inhabitants in this area, number of inhabitants aged 65 years or more, total number of COVID-19 patients, and number of ICU beds with COVID-19 patients.

Taking now into account the information from Table 1, we have considered the following eight event patterns:

- HOS: daily hospital-bed occupation rate by COVID-19 patients (beds occupied by COVID-19 patients/total beds).
- ICU: daily critical-care-bed occupation rate by COVID-19 patients (critical-care-bed occupied by COVID-19 patients/total critical-care-beds).
- CIT7: 7-day case notification rate per 100,000 inhabitants.
- CIT14: 14-day case notification rate per 100,000 inhabitants.
- CIT7,65: 7-day case notification rate of people aged 65 or older per 100,000 inhabitants.
- CIT4,65: 14-day case notification rate of people aged 65 or older per 100,000 inhabitants.

Fig. 6. CPN modeling two zones interconnected.
• **POS**: 7-day positive test percentage (100 × positive tests/ total tests).
• **TRA**: 7-day traceable case percentage (100 × traceable positive tests/positive tests).

As an example, Fig. 9 shows the HOS pattern modeled with MEdit4CEP. For each `AreaBedStatus` simple event, this pattern will produce a complex event, called HOS, with the following information: timestamp, health area and percentage of occupied COVID-19 beds. This percentage is obtained as the division between the number of beds occupied and the number of COVID-19 beds occupied, and multiplying the result by 100. Listing 1 shows the Esper EPL code automatically generated by MEdit4CEP for this pattern.

Listing 1: EPL code automatically generated by MEdit4CEP for the HOS pattern.

```esper
@Name( "HOS" )
insert into HOS
select a1. timestamp as timestamp ,
a1. healthArea as healthArea ,
((a1. covidBeds / a1. beds) * 100) as covidBedPerc
from pattern [ (every a1 = AreaBedStatus )]
```

Fig. 10 shows the CI7 pattern modeled with MEdit4CEP. For each `AreaTestStatus` event, this pattern checks if there are 6 more subsequent `AreaTestStatus` events that share the same health area. In that case, this pattern will create a new complex event, called CI7, with the following information: timestamp, health area and the 7-day case notification rate per 100,000 inhabitants. This is obtained as the sum of the positive COVID-19 tests per area, divided by the total number of inhabitants, and the result is multiplied by 100,000. Listing 2 shows the Esper EPL code automatically generated by MEdit4CEP for this pattern.

Listing 2: EPL code automatically generated by MEdit4CEP for the CI7 pattern.

```esper
@Name( "CI7" )
insert into CI7
select a7. timestamp as timestamp ,
a7. healthArea as healthArea ,
(((((a2. positCovidTests + a1. positCovidTests) + a3. positCovidTests) + a4. positCovidTests) + a5. positCovidTests) + a6. positCovidTests) +
a7. positCovidTests) / a7. inhabitants ) / 100000 as ci7PositCovidTests
from pattern [ (every a1 = AreaTestStatus ) ->
a2 = AreaTestStatus(a2. healthArea = a1. healthArea ) ->
a3 = AreaTestStatus(a3. healthArea = a2. healthArea ) ->
a4 = AreaTestStatus(a4. healthArea = a3. healthArea ) ->
a5 = AreaTestStatus(a5. healthArea = a4. healthArea ) ->
a6 = AreaTestStatus(a6. healthArea = a5. healthArea ) ->
a7 = AreaTestStatus(a7. healthArea = a6. healthArea ) ]
```
The other event patterns and their implementations are available for downloading from the following Mendeley repository: http://dx.doi.org/10.17632/tmfkm9wgn.1.

5.2. The expert system: COVID-19 FIS

In this subsection, we present the FIS implementation for our PITS system. In Fig. 3 we presented the different elements we need to implement a Mamdani’s FIS. These elements are the input and output fuzzy sets as well as the if-then rules and the configuration parameters. The tool used to implement the FIS is the Skfuzzy Python toolbox. The specific configuration parameters used are the following:

- Defuzzification ⇒ centroid, Rule aggregation ⇒ combine all output membership function using a maximum operator, T-norm (AND) ⇒ minimum; and S-norm (OR) ⇒ maximum. The centroid computes the center of gravity of the fuzzy set along the x-axis (the output fuzzy set). The rule aggregation specifies how the rule outputs must be combined when more than one is applied. The values for the AND and OR operators determine how to obtain the truth degree when we consider several inputs in the same rule.

- The output variable is the alert level. There are five alert levels: “Normal”, “Low”, “Medium”, “High” and “Extremely high”.

Each variable has fuzzy sets associated, and for each membership function we use a trapezoidal function, taking into account the indicator levels indicated in Table 1. Trapezoidal functions are commonly used for membership functions, since they are simple and computationally efficient (Sadollah, 2018).

Fig. 11 shows the fuzzy sets for the input and output variables. As mentioned, the actions to be taken for these levels are more restrictive than those showed in Table 1, to prevent the scenario to get worse.

Table 3 shows the if-then rules for this proposal. As we can observe, an advantage offered by fuzzy logic is that its computing, imitating the human reasoning, allows the system to be easily adapted and extended to other situations of interest by, for instance, updating the FIS rules.

Table 3

| Rule number | Inputs | HOS Conn | ICU Conn | Output alert |
|-------------|--------|----------|----------|--------------|
| 1           | Extreme OR | Extreme OR | Extreme | 4            |
| 2           | High AND not(Extreme) | AND not(Extreme) | 3        |
| 3           | not(Extreme) AND High | AND not(Extreme) | 3        |
| 4           | not(Extreme) AND not(Extreme) | AND High | 3          |
| 5           | Medium AND Medium AND Medium | AND Medium | 2          |
| 6           | Medium AND Low AND Medium | AND Medium | 2          |
| 7           | Medium AND Normal AND Medium | AND Medium | 2          |
| 8           | Medium AND Low AND Low | AND Low | 2          |
| 9           | Medium AND Low AND Normal | AND Normal | 2          |
| 10          | Medium AND Normal AND Normal | AND Normal | 2          |
| 11          | Medium AND Normal AND Low | AND Low | 2          |
| 12          | Low AND Medium AND Medium | AND Medium | 1          |
| 13          | Low AND Medium AND Low | AND Low | 1          |
| 14          | Low AND Medium AND Normal | AND Normal | 1          |
| 15          | Low AND Low AND Medium | AND Medium | 1          |
| 16          | Low AND Low AND Low | AND Low | 1          |
| 17          | Low AND Low AND Normal | AND Normal | 1          |
| 18          | Low AND Normal AND Normal | AND Normal | 0          |
| 19          | Medium AND Medium AND Normal | AND Normal | 2          |
| 20          | Medium AND Medium AND Low | AND Low | 2          |
| 21          | Normal AND Normal AND Normal | AND Normal | 0          |
| 22          | Low AND Normal AND Low | AND Low | 1          |
| 23          | Low AND Normal AND Medium | AND Medium | 1          |
| 24          | Normal AND Low AND Normal | AND Normal | 0          |
| 25          | Normal AND Normal AND Low | AND Low | 0          |
| 26          | Normal AND Low AND Low | AND Low | 0          |
| 27          | Normal AND Medium AND Normal | AND Normal | 1          |
| 28          | Normal AND Medium AND Low | AND Low | 1          |
| 29          | Normal AND Medium AND Medium | AND Medium | 1          |
| 30          | Normal AND Normal AND Normal | AND Normal | 1          |
| 31          | Normal AND Normal AND Low | AND Low | 1          |

Fig. 12 depicts the surface for the alert output variable when the input variables are CI7 and HOS, keeping constant ICU. Five surfaces are then depicted, considering five values for ICU (from 2 to 27.5). The left-hand side graph corresponds to ICU = 2 (normal), and we can observe that for values of HOS and CI17 close to zero the alert level is 0, but the alert level increases as HOS and/or CI17 values increase until reaching an alert value close to 4. In contrast, for ICU = 27.5 (extreme), the alert value will always be 4, independently of the values of HOS and CI7. In this last case, only Rule 1 in Table 3 is applied.

However, when ICU is 2 (first graph), if we consider values 59.7 and 10 for IC7 and HOS, respectively, rules 3 and 19 are applied at the same time with a combined output of 2.65 for the alert level. The domain expert is then in charge of interpreting this value, and decide whether the alert level is 2 or 3, possibly taking into account some other considerations/indicators.
5.3. A CPN model as a map to compute routes

A CPN is used to represent the city map,\(^2\) in a similar way as our running example in Section 4.3. The CPN model proposed to compute available routes consists of 4 pages. This is a mechanism that CPN Tools facilitates to produce complex models in a modular way, splitting the model into pages, the glue being the so-called fusion places, which are places that technically represent a same place, i.e., a place that appears in different pages. These fusion places are identified by a label, indicating their associated fusion set.

In the following subsections we describe these CPN pages of the model.

\(^2\) We use the concept of city map as a geographic area map.

5.3.1. Main page: the map

This page models how a token (a car) moves across the different areas of a map. The color set of these tokens has the following information: final destination, list of traversed zones, and accumulated time for the journey. Furthermore, these tokens are timed, which means that our model is a timed CPN, in which the tokens carry a timestamp indicating the time at which they are available. When a token moves from one area to another, the token age is increased to represent the duration of this step. We use the color set \(\text{Zo}\) for the zone places:

\[
\text{colset Zo}=\text{product INT*INTlist *INT timed;}
\]

These places that represent the different health zones are named as indicated in Fig. 1. Due to a technical problem in CPN Tools when dealing with guards and time, we need a direct control on the time model. Thus, we include a place \(\text{time inf}\) with color set \(\text{INT}\) that captures the current time in the model.
Alert levels are associated to zones, including the places alertZi (i = 1, …, 7), with the color set A defined as follows:

\[
\text{colset A = int with 0..4;}
\]

The token value on each one of these places will then indicate the current alert level in the corresponding zone.

Fig. 13 shows an excerpt of the whole CPN with the movements throughout the city map (see Fig. 14). The movements between zones are represented by transitions. For instance, transition \(WtowNW\) in Fig. 13 represents the movement from zone Wn1 to Wn2. The token \([2,1,0]\)@0 in place Wn1 represents a car whose destination is Wn2 (the zone number is the number at the end of the place names), which is at zone Wn1 (the only one in the list), and has not yet moved. Transition \(WtowNW\) is depicted in green, which means that it is enabled and can be fired at the current time. The guard pass associated to these transitions is defined as follows:

\[
\text{fun pass(j:INT, x:Zo, ai:A, aj:A, t:INT, cij:INT)} =
\begin{align*}
\text{isOpen(j, x) andalso OAalert(ai, t) andalso OAalert(aj, t, cij) }
\end{align*}
\]

This guard pass is used to check that the movement is allowed. The following conditions must be satisfied:

1. The car has not yet reached its final destination (function \(\text{isDest}\)).
2. A car cannot pass twice by the same zone (function \(\text{noReturn}\)).
3. The alert conditions allow the movement, as explained below.

Notice the use of the predefined functions \(\text{mem}\), which checks whether an element belongs to a given list, \(\text{hd}\), which returns the head of a list, and \(\text{rev}\), which returns the list in reverse order. Alert functions are defined as follows:

\[
\text{fun OAalert(a:A, t:INT) = if (a=0 orelse a=1 orelse (a=2 andalso day_hour(t)>q1 andalso day_hour(t)<q2) orelse (a=3 andalso week_day(t)<6 andalso day_hour(t)>q1 andalso day_hour(t)<q2)) then true else false;}
\]

This function implements the mobility restrictions indicated in Table 2 for the five alert levels considered. Functions \(\text{day_hour}\) and \(\text{week_day}\) are used to compute the day and hour of a given value in minutes, using as a reference the first Monday at 0 h 0 min when the protocol is enacted. The constants \(q1\) and \(q2\) determine the time when mobility is allowed depending on the alert level.

fun day_hour(t:INT)=((t div 60) mod 24; fun week_day(t:INT)=((t div 60) div 24) mod 7)+1;

When transition \(WtowNW\) is fired, the token on the Wn1 place is removed, and a new token is written into the Wn2 place, using the function \(\text{traverseSection}\) to produce this new token. This function preserves the first field (destination), extends the list of traversed zones by including zone \(j\), and adds the time to complete this movements to the third field. The definition of this function is then as follows:

\[
\text{fun traverseSection(j:INT, x:Zo, cij:INT) =}
\begin{align*}
(\#1x, (\#2x~@[j]), (\#3x+cij))
\end{align*}
\]

In Fig. 13, the token \((2,[1],0)@0\), will be consumed and a new token \((2,[1,2],4)@4\) will appear at place Wn2, if the travel time is four minutes.

5.3.2. EndDestination Page

Fig. 15 shows the EndDestination CPN page. Once a token (car) has reached its destination, the corresponding \(\text{isDest}\) transition, which has \(P\text{HIGH}\) priority, is fired (see guard \(\text{isDest}\)). This CPN page is used to obtain the fastest route for the movement represented by the token on the corresponding destination zone for this simulation. With the firing of the \(\text{isDest}\) transition, the token is removed from the destination zone and a new token is written into the Destination place, with the list of traversed zones and travel duration (color set D).

\[
\text{colset D = product INTlist*INT;}
\]

There is a control place, named \(\text{end_d}\), which is used to ensure that only the fastest route is stored at Destination, i.e., with this place we only capture the information for the fastest route for the simulation performed.

5.3.3. Time_alerts page

Fig. 16 shows the Time_alerts page of the CPN model. The right-hand side part of this CPN generates the random alert levels for the different areas and the left-hand side part is used for time elapsing, using transitions \(\text{tick}\) (1 min) and new_day (days). The right-hand side part of Fig. 16 consists of a set of alert places \(\text{alertZi}\) for the different areas, and both transitions \(\text{gen_alert}\) and \(\text{set_alert}\), which are used to produce the random alert values using the functions \(\text{genAlert}\) and \(\text{setAlert}\) on a daily basis (p_init is marked with the firing of \(\text{new_day}\)).

fun genAlert():A =
\begin{align*}
\text{let val b = discrete(1,100) in}
\begin{align*}
\text{if (b <= 50) then 1 else if (b <= 85) then 2 else 3 end;}
\end{align*}
\end{align*}

These random values are obtained in two steps in order to avoid huge differences between the alert levels of the health areas, only allowing a maximum difference of two alert levels between areas. In addition, higher alert levels will have smaller probabilities than medium and low levels. In the first step, \(\text{genAlert}\) generates a value between 1 and 3 in place g_init by the firing of the \(\text{gen alert}\) transition. This function establishes a higher probability to the alert level 1 (50%), a probability of 35% for alert level 2 and 15% for level 3. Transition \(\text{set alert}\) is then fired after \(\text{gen alert}\), but at the same time instant (see Fig. 16), to produce a value between 0 and 4 in the alert places \(\text{alertZi}\), using the \(\text{setAlert}\) function. This function increases or decreases in one unit the value obtained by \(\text{genAlert}\).
with a probability of 25%, respectively, otherwise, with a probability of 50% returns the same value. Place $g_{\text{init}}$ in Fig. 16 is marked with one token with value 3, and the alert levels obtained for the different areas vary between 2 and 4, as obtained using the custom distributions shown above. On the left-hand side of the figure time elapsing is modeled. As mentioned previously, CPN Tools only re-evaluates guards when a transition is fired and not when time elapses. Thus, we need a direct control on time to allow the re-evaluation of guards as time advances. Place $\text{init\_time}$ is a timed integer token, whose value is the current model time, whereas $\text{time\_inf}$ keeps this information in an untimed place. This is important to allow the reading of the time without aging it, which would disturb the net behavior.

Notice that both transitions $\text{tick}$ and $\text{new\_day}$ have the lowest priority in the CPN model, so time elapsing can only occur when no other transition can be fired. Regarding the guards, $t<\text{max}$ is used to avoid an infinity ticking behavior, and $t>0$ avoids to produce the alert values twice in time zero. Both transitions are mutually exclusive, $\text{tick}$ is used to advance the time in minutes, except when a new day starts, in which case $\text{new\_day}$ is fired.

5.3.4. Model simulation

Simulation is used in order to produce a fast route, by using a certain number of cars (tokens) traveling from the same origin to a same destination. Thus, the key point is that all of these cars are in a race and the winner is the fastest one. As we increase the number of cars (tokens) in the model we will have a greater probability to obtain the fastest route. Of course, the more possible paths to reach the destination, the more number of tokens we should include to obtain a good result.
However, as we intend to obtain fast routes, but not necessarily the fastest one, we have considered in all cases 100 cars (tokens), and the experiments have demonstrated that with this number we always obtain the fastest route for the case of Madrid. Notice that simulations terminate when no transition can be fired in the CPN model.

Furthermore, before simulating the CPN model, we need to set the travel times (in minutes) between the different areas. These times are shown in Table 4. These values are specified using constants in the model. For instance, shown in Table 4. These values are specified using constants in the model.

Another point of interest is termination. It can be the case that there is no available route to the final destination, due to the current restrictions (connections closed), so our proposal would not be able to provide a solution. In addition, it is also possible that such a route exists, but it cannot be found if few cars have been considered as initial marking.

With respect to the net behavior, for small city maps we can perform some analysis using the reachability graph (markings that are reachable from the initial one), which can be generated using the CPN Tools engine. However, notice that in general, as mentioned in Section 4.3, the reachability graph can have an enormous size, and actually, it can only be constructed for bounded Petri nets. In the case of timed CPNs this problem is worse, so in general, state exploration cannot be used to analyze the net behavior. In our case, we can produce the reachability graph for some specific configurations and only for small city maps. As an illustration, we have generated the reachability graph for this CPN considering only one car and taking all alert levels equal to 1 (low risk) and a time limit of 292 min, starting at time 192. Thus, we are only interested in the routes that are reachable within 100 min from the starting time. With this configuration, the reachability graph has been constructed, and it consists of 2577 nodes and 2580 arcs. In addition, there are 25 dead markings, and 20 of them are terminating routes, from area 3 to area 1. One of these 20 terminating markings is the only obtained as solution by the simulations, which takes 6 min to reach the destination.

5.3.5. CPN external communication

In the previous simulations we have used random values for the alert levels, as produced by functions genAlert and setAlert. Now, the CPN model is fed with the alert levels produced by our FIS system implemented in Python, with the help of the domain expert. This information is provided using the five alert levels defined in Table 2 and encoded by color set A.

Gallash and Kristensen (2001) presented the Comms/CPN library, which allows us to establish a communication between CPN models and external processes. The Python FIS system uses this library to send the alert levels to the CPN model according to the authority decision about the mobility restrictions.

Comms/CPN is designed to act as an interface between CPN models and TCP/IP containing all TCP/IP and socket related primitive functions. The connection strategy is based on a unique identification string and a set of attributes to identify the IP and port used. The library has the capability to either send and receive all types of data, which includes the data types defined by the user (color sets). Encoding and decoding functions are provided with this purpose. The primitives used in an interaction include connect, accepts, send, receive, and disconnect in the external part implemented in Standard ML and Java, while openConnection, acceptConnection, send, receive and closeConnection can be used to model the interaction in the CPN model.

Fig. 17 depicts the implemented model to read the alert levels provided by the FIS system implemented in Python. Constant genType is introduced in the CPN model to decide whether the alert.

### Table 4

| Type | Wn1 | NWn2 | Nn3 | En4 | Cn5 | SEn6 | Sn7 |
|------|-----|------|-----|-----|-----|------|-----|
| Wn1  | 4   | 4    | 2   | 2   | 3   | 8    | 5   |
| NWn2 | 4   | 4    | 6   | 6   | 3   | 6    | 3   |
| Nn3  | 8   | 3    | 4   | 4   | 4   | 6    | 4   |
| En4  | 5   | 4    | 4   | 4   | 5   | 5    | 5   |
| Cn5  | 5   | 5    | 5   | 5   | 5   | 5    | 5   |
| SEn6 | 5   | 5    | 5   | 5   | 5   | 5    | 5   |
| Sn7  | 5   | 5    | 5   | 5   | 5   | 5    | 5   |
levels are produced randomly or by the FIS system. Thus, transition \texttt{t\_connection} has the guard \texttt{genType = 2} to activate the production of the alert levels by the FIS system. This transition is only executed initially (value 0 in \texttt{p\_init}), in order to open the external connection \texttt{Con1}. Once the connection is open, transition \texttt{t\_status2} will be fired to produce the new alert values every day.

All the external interactions follow the structure indicated in the figure. There is an input section, which indicates the input variables used, an output section, which indicates the variables which will contain the results obtained in this interaction, and the action section, in which the actions to be performed in this interaction are defined.

For instance, the interaction associated to transition \texttt{t\_connection} opens a connection \texttt{"Con1"} in the localhost using the port 9000. This connection can now be used by the transitions \texttt{t\_status1} and \texttt{t\_data} to gather the alert values that will be stored in the corresponding alert zone places alertZi, using the variables \texttt{sti} and the receive routine over connection \texttt{"Con1"}, using the \texttt{stringDecode} function. Function \texttt{toInt} is used to transform a string into an integer number.

Finally, notice that the CPN model has been designed to be working continuously, so the connection never closes.

6. Conclusions and future work

In this paper, an ITS, called PITS, was proposed considering different scenarios of mobility restrictions according to the alert levels associated to the spread of the COVID-19.

This novel proposal, integrating CEP, fuzzy logic and CPN, fulfills the following desirable aspects in an ITS for these situations: (1) it is consistent with the pandemic emergency situation, (2) it is flexible so as to modify and adapt the model to new situations, and (3) it is simple, but rich enough to allow us to make decisions and draw conclusions about mobility restrictions.

Health indicator data are assumed to be provided by a set of health stations (hospitals, health primary centers, clinical laboratories, among others). The CEP technology is then used to process the information gathered and determine the official health indicators. Specifically, we used the MEdit4CEP tool to specify the event patterns regarding the indicators: CI7 (cumulative incidence of COVID-19 cases in the last 7 days), HOS (daily percentage of confirmed hospitalized COVID-19 cases with respect to full capacity of the hospital), ICU (daily percentage of confirmed hospitalized COVID-19 cases in ICU beds with respect to full capacity of the hospital in ICU beds), CI14 (cumulative incidence of COVID-19 cases in the last 14 days), POS (percentage of positive in the tests of COVID-19 in the last 7 days) and TRA (percentage of traceable positive cases with respect to all positive tests in the last 7 days). Then, a FIS is applied considering the three first indicators in order to determine a preliminary crisp number. This number is rounded to a risk level according to the domain expert criteria, possibly considering other indicators.

The system, with the help of the domain expert, provides the mobility restrictions for the affected areas. The CPN model of a city map was then defined considering that some connections between the areas could be closed at certain hours, depending on the alert levels. Afterwards, simulation was used to obtain fast routes from the CPN model, by feeding the CPN with a large number of tokens that compete to reach a same destination.

Moreover, this is a scalable proposal, since we could extend the system by including a great number of zones as well as varying the time periods to cross from one zone to another. In addition, it can be adapted to other scenarios and restrictions: a different topology could easily be considered, travel times could also be adapted to the vehicle type, the user might choose the route by avoiding to transit in the most affected COVID-19 areas, among others.

As future work, we plan to include traffic flow data measured by sensor stations and to incorporate queueing theory and machine learning techniques as in Gal et al. (2017) for traveling time prediction. Other health indicators or emergency situations, such as traffic accidents, could be included in our FIS, or in the final decision made by the domain expert to close an area. This decision process can also be performed using group consensus techniques as stated by Pérez et al. (2018). We plan also to extend our proposal by applying these indicators and techniques to face other emergency scenarios, such as air pollution, floods and earthquakes.

CRediT authorship contribution statement

**Enrique Brazález:** Methodology, Software, Writing – original draft. **Hermenegilda Macià:** Conceptualization, Methodology, Software, Writing – original draft, Funding acquisition. **Gregorio Díaz:** Conceptualization, Methodology, Software, Writing – original draft, Supervision, Funding acquisition. **Valentín Valero:** Methodology, Software, Writing – original draft, Writing – review & editing, Funding acquisition. **Juan Boubeta-Puig:** Methodology, Software, Writing – original draft, Supervision, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.
Acknowledgments

This research was supported by the Spanish Ministry of Science, Innovation and Universities and the European Union ERDF funds through the FAME project under grants RTI2018-093608-B-C32 and RTI2018-093608-B-C33, the JCCM project cofinanced with ERDF funds under grant S17/180501/000276, and the UCLM group research grant with reference 2021-GRIN-30993.

Supporting information

The data obtained supporting the findings of this research can be found in the Mendeley repository http://dx.doi.org/10.17632/tmkfmr9wgn.1. In this repository the reader can find all the files that are necessary to replicate the results obtained.

References

Alzamzami, O., Mahgoub, I., 2019. Fuzzy logic-based geographic routing for urban vehicular networks using link quality and achievable throughput estimations. IEEE Trans. Intell. Transp. Syst. 20 (6), 2289–2300. http://dx.doi.org/10.1109/TITS.2017.2767177.

Badir, H., Du, H., Marshall, M., Dong, E., Squire, M., Gardner, L., 2020. Association between mobility patterns and COVID-19 transmission in the USA: a mathematical modelling study. Lancet Infect. Dis. 20 (11), 1247–1254. http://dx.doi.org/10.1016/S1473-3099(20)30553-3.

Barthe-Delanoë, A.-M., Montarnal, A., Truptil, S., Bénaben, F., Pingaud, H., 2018. Towards the agility of collaborative workflows through an event driven approach – Application to crisis management. Int. J. Disaster Risk Reduct. 28, 214–224. http://dx.doi.org/10.1016/j.ijdrd.2018.02.029.

Bekiaris, E., Nakanishi, J., 2004. Economic Impacts of Intelligent Transportation Systems. In: Transportation System to control air pollution in cities using complex event processing and colored Petri nets. In: Rojas, I., Joya, G., Català, A. (Eds.), Advances in computational intelligence – 14th International Conference, ICIC-2018. LNCS, Vol. 10306, Springer, pp. 415–426. http://dx.doi.org/10.1007/978-3-319-59147-6.36.

EsperTech, 2022. Esper - complex event processing. http://www.espertech.com/esper/ (March 2022).

Eswaran, S., Bapat, J., Arifhara, V., 2015. Service driven dynamic hashing based radio resource management for intelligent transport systems. In: Lecture Notes in Computer Science (including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). Vol. 9066, pp. 12–23. http://dx.doi.org/10.1007/978-3-319-17765-6.2.

Figureiran, P., Antunes, H., Guerreiro, G., Costa, R., Jardim-Goncalves, R., 2018. Visualisation and detection of road traffic events using complex event processing. In: ASME International Mechanical Engineering Congress and Exposition, Proceedings (IMECE), Vol. 2. http://dx.doi.org/10.1115/IMECE2018-87909.

Gal, A., Mandelbaum, A., Schnitzler, F., Senderovich, A., Weidlich, M., 2017. Traveling time prediction in scheduled transportation with journey segments. Inf. Syst. 64, 266–280. http://dx.doi.org/10.1016/j.jis.2015.12.001.

Gallais, G.E., Kristensen, L.M., 2001. Comms/CPN: A communication infrastructure for external communication with design/CPN.

Gargoum, S.A., Gargoum, A.S., 2021. Limiting mobility during COVID-19, when and to what level? An international comparative study using change point analysis. J. Transp. Health 20 (101119), http://dx.doi.org/10.1016/j.jth.2021.101119.

Gudmundsson, S., Kong, T., 2020. Engineering Applications of Artificial Intelligence 114 (2022) 105154.

Hendrickson, C., Rillet, L.R., 2020. The COVID-19 pandemic and transportation engineering. J. Transp. Eng. A 146 (7), http://dx.doi.org/10.1061/JTEPBL.000196.

Huang, Y.-S., Chun, T.-H., 2008. Modeling and analysis of urban traffic lights control systems using timed CPN. J. Inf. Sci. Eng. 24 (3), 875–890. http://dx.doi.org/10.1109/JISE.2008.038.

Kraemer, M., Yang, C., Gutierrez, R., Wu, C., Klein, B., Piggott, D., Plessis, L., Faria, N., Li, R., Hanage, W., Brownstein, J., Layan, M., Vespiagnani, A., Tian, H., Dye, C., Pybus, O., Scarponi, S., 2020. The effect of human mobility and control measures on the COVID-19 epidemic in China. Science 368 (6490), 493–497. http://dx.doi.org/10.1126/science.abb4218.

Kumar, P.B., Rawat, H., Parhi, D.R., 2019. Path planning of humanoids based on artificial potential field method in unknown environments. Expert Syst. 36 (2), e12360. http://dx.doi.org/10.10111/j/exy.12360.

Laksymi, K., Nagineni, S., Lydi, E.L., Devaraj, A.P.S., Mohanty, S.N., Pustokhin, I.V., Pustokhin, D.A., 2022. An optimal deep learning for cooperative intelligent transportation system. Comput. Mater. Contin. 72 (1), 19–35. http://dx.doi.org/10.32604/cmc.2022.020244.

Luckham, D.C., 2001. The Power of Events: An Introduction to Complex Event Processing in Distributed Enterprise Systems. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

Luckham, D., 2012. Event Processing for Business: Organizing the Real-Time Enterprise. John Wiley & Sons, New Jersey, USA.

Luthra, M., Koldehoff, B., Höchst, J., Lampe, P., Rizvi, A.H., Kundel, R., Freisleben, B., 2019. InNetCEP: In-network complex event processing for information-centric networking. In: ACM/IEEE Symposium on Architectures for Networking and Communications Systems (ANCS). pp. 1–13. http://dx.doi.org/10.1109/ANCS.2019.8901877.

Macià, H., Valero, V., Díaz, G., Boubeta-Puig, J., Ortiz, G., 2016. Complex event processing and colored Petri nets. IEEE Access 4, 7425–7439. http://dx.doi.org/10.1109/ACCESS.2016.2621718.

Mandami, E., Assilian, S., 1975. An experiment in linguistic synthesis with a fuzzy logic controller. Int. J. Man-Mach. Stud. 7 (1), 1–13. http://dx.doi.org/10.1016/0020-7737(75)80002-2.

Mijlović, V., Tomasević, N., Janev, V., Stanojević, M., Vranes, S., 2019. Emergency management in critical infrastructures: A complex-event-processing paradigm. J. Syst. Sci. Technol. Eng. 28 (1), 37–62. http://dx.doi.org/10.1109/JSSCSET.2019.879393.

Montero, V.M., Pérez, J., Godoy, J., Onieva, E., 2012. A fuzzy aid for rear-end collision warning/avoidance system. Expert Syst. Appl. 39 (10), 9097–9107. http://dx.doi.org/10.1016/j.eswa.2012.02.054.

Naranjo, J., Gonzalez, C., Reviejo, J., Garcia, R., de Pedro, T., 2003. Adaptive fuzzy control for in-wheel-motor e-car keeping. IEEE Trans. Intell. Syst. 4 (3), 132–142. http://dx.doi.org/10.1109/TIFS.2003.821294.

Ng, K., Reay, M., Ali, M., 2013. A review on the applications of Petri nets in modeling, analysis, and control of urban traffic. IEEE Trans. Intell. Syst. 14 (2), 858–870. http://dx.doi.org/10.1109/TITS.2013.2246153.

Oliphant, T.P., Jones, P., 2020. SciPy - skfuzzy. URL: https://pythonhosted.org/scikit-fuzzy/.
Zhou, F., Castillo, L., García-de Prado, A., Boubeta-Puig, J., 2022. Evaluating a flow-based programming approach as an alternative for developing CEP applications in the IoT. IEEE Internet of Things Journal 9 (13), 11489–11499. http://dx.doi.org/10.1109/JIOT.2021.3130498.

Palacharla, P., Nelson, P., 1999. Application of fuzzy logic and neural networks for dynamic travel time estimation. Int. Trans. Oper. Res. 6 (1), 145–160. http://dx.doi.org/10.1111/j.1475-9909.1999.tb00148.x.

Park, S., Anderson, K., Bost, D.L., 2009. Disaster planning and evacuation response – case study of honolulu tsunami warning and intelligent transportation systems route planning. URL: https://trid.trb.org/view/924055.

Pérez, I., Cabrerozio, F., Alonso, S., Dong, Y., Chichiana, F., Herrero-Viedma, E., 2018. On dynamic consensus processes in group decision making problems. Inform. Sci. 459, 20–35. http://dx.doi.org/10.1016/j.ins.2018.05.017.

Peterson, J.L., 1981. Petri Net Theory and the Modeling of Systems. Prentice Hall PTR, Upper Saddle River, NJ, USA.

Picone, M., Busanelli, S., Amoretti, M., Zanichelli, F., Ferrari, G., 2015. Advanced Technologies for Intelligent Transportation Systems. Springer Nature.

Placzek, B., 2013. A traffic model based on fuzzy cellular automata. J. Cell. Autom. 8 (3–4), 261–282.

Qi, L., Zhou, M., Luan, W., 2016. Emergency traffic-light control system design for intersections subject to accidents. IEEE Trans. Intell. Transp. Syst. 17 (1), 180–193. http://dx.doi.org/10.1109/TITS.2015.2466073.

Quek, C., Pasquier, M., Lim, B., 2006. Pop-traffic: A novel fuzzy neural approach to road traffic analysis and prediction. IEEE Trans. Intell. Transp. Syst. 7 (2), 133–146. http://dx.doi.org/10.1109/TITS.2006.874712.

Ramírez, A., Moreno, N., Vallecillo, A., 2021. Rule-based preprocessing for data stream mining using complex event processing. Expert Syst. 38 (8), e12762. http://dx.doi.org/10.1111/exsy.12762.

Sadollah, A., 2018. Fuzzy Logic Based in Optimization Methods and Control Systems and its Applications. IntechOpen, Rijeka, http://dx.doi.org/10.5772/intechopen.73112.

Singh, D., Marinescu, M., Guzmán-Merino, M., Durán, C., Delgado-Sanz, C., Gomez-Barroso, D., Carretero, J., 2021. Simulation of COVID-19 propagation scenarios in the madrid metropolitan area. Front. Public Health 9, http://dx.doi.org/10.3389/fpubh.2021.636023.

Terrono-Saenz, F., Valdés-Vela, M., Sotomayor-Martinez, C., Toledo-Moreo, R., Gomez-Sarmaceta, A.P., 2012. A cooperative approach to traffic congestion detection with complex event processing and VANET. IEEE Trans. Intell. Transp. Syst. 13 (2), 914–929. http://dx.doi.org/10.1109/TITS.2012.2186127.

Trivedi, P., Zulkernine, F., 2020. Intelligent transportation system: Managing pandemic induced threats to the people and economy. In: 2020 IEEE 8th International Conference on Smart City and Informatization (ISCI). pp. 60–67. http://dx.doi.org/10.1109/SCIS50694.2020.00017.

Valero, V., Díaz, G., Boubeta-Puig, J., Macià, H., Brazález, E., 2022. A compositional approach for complex event pattern modeling and transformation to colored Petri nets with black sequencing transitions. http://dx.doi.org/10.1109/TESE.2021.3065584, (in press).

Westergaard, M., Verbeek, H.E., 2022. CPN tools home page. http://www.cpntools.org/ (March 2022).

Wu, J., Wang, K., He, C., Huang, X., Dong, K., 2021. Characterizing the patterns of China’s policies against COVID-19: A bibliometric study. Inf. Process. Manage. 58 (4), 102562. http://dx.doi.org/10.1016/j.ipm.2021.102562.

Ye, Z., Chauhani, J., Booth, J., Ponas, B., 2010. Evaluation of the use of rural transportation infrastructure in evacuation operations. J. Transp. Safety Secur. 2 (2), 88–101. http://dx.doi.org/10.1080/19439962.2010.487633.

Yín, H., Wong, S., Xu, J., Wong, C., 2002. Urban traffic flow prediction using a fuzzy-neural approach. Transp. Res. C 10 (2), 85–98. http://dx.doi.org/10.1016/S0924-090X(01)00004-3.

Zadeh, L., 1965. Fuzzy sets. Inf. Control 8 (3), 338–353. http://dx.doi.org/10.1016/0019-9944(65)90241-X.

Zhou, F., Jing, X., Wang, M., 2013. Urban controlled traffic flow modeling using colored Petri nets. J. Comput. Inform. Syst. 9 (3), 1217–1228.

E. Brasíles, H. Macià, G. Díaz et al.

Enrique Brazález is a PhD student in Advanced Information Technologies at the University of Castilla-La Mancha within the ReTiCS research group. He obtained a Master’s degree in Computer Science in 2018-2019. He has participated in some national conferences and he works in a FEDER project called “Analysis and Development of Models for Air Quality Control, Effects on Vegetation and Cloud Environments”. His field of research is Complex Event Processing and its application in Industry 4.0 environments such as renewable energies and the study of air quality.

Hermenegilda Macià received her degree in Mathematics from the Universidad de Valencia and her PhD Computer Engineering from the Universidad de Castilla-La Mancha (UCLM), Spain. She is an Associate Professor with the Department of Mathematics of UCLM and belongs to the ReTiCS (Real Time and Computer Systems) research group. She has published research articles in reputed journals of Mathematics and Computer Science. Her current research interests include the application of formal methods, such as Petri nets, in various fields of engineering in combination with other technologies (Cyber-Physical, fuzzy logic), formalisms for modeling biological processes or Engineering Education.

Gregorio Díaz is an associated Professor at the Universidad de Castilla-La Mancha within the ReTiCS research group with tenure distinction (2011), published more than 22 journal papers indexed by the JCR index, participated in 42 international conferences, main researcher of 4 European Regional Development Fund (ERDF) projects. His research work is limited to management environments more reliable, secure, and easier to design. He has supervised more than 30 master theses, including 4 in research areas, and 3 PhD theses. He has taught in undergraduate and postgraduate studies awarded with the quality award Euro-Inf Bachelor by EQANIE.

Valentin Valero is a full Professor of Distributed Systems and Operating Systems at the University of Castilla-La Mancha, in the Computer Science School of Albacete, Spain. He received his degree in Mathematics from the Complutense University of Madrid in 1987, and his PhD in Mathematics in 1993 at the Department of Computer Science of the Complutense University of Madrid. Since October 1987 he is a member of the Computer Science Department at the University of Castilla-La Mancha. His current research areas are in the field of concurrency, specifically in formal models for analysis and design of concurrent systems and real-time systems.

Juan Boubeta-Puig received the Ph.D. degree in Computer Science and Engineering from the University of Cadiz (UCA), Cadiz, Spain, in 2014. He is an Associate Professor with the Department of Computer Science and Engineering, UCA. His research interests include real-time big data analytics through complex event processing, event-driven service-oriented architecture, Internet of things, blockchain and model-driven development of advanced user interfaces, and their application to smart cities, industry 4.0, e-health, and cybersecurity. Dr. Boubeta-Puig was honored with the Extraordinary Ph.D. Award from UCA and the Best Ph.D. Thesis Award from the Spanish Society of Software Engineering and Software Development Technologies.