International Journal on Software Tools for Technology Transfer
https://doi.org/10.1007/s10009-022-00689-5

GENERAL

Special Issue: TACAS 2021

On the road with RTLola

Testing real driving emissions on your phone

Sebastian Biewer[1] · Bernd Finkbeiner[2] · Holger Hermanns[1] · Maximilian A. Köhl[1] · Yannik Schnitzer[1] · Maximilian Schwenger[2]

Accepted: 4 November 2022

Abstract
This paper is about shipping runtime verification to the masses. It presents the crucial technology enabling everyday car owners to monitor the behaviour of their cars in-the-wild. Concretely, we present an Android app that deploys RTLOLA runtime monitors for the purpose of diagnosing automotive exhaust emissions. For this, it harvests the availability of cheap Bluetooth adapters to the On-Board-Diagnostics (OBD) ports, which are ubiquitous in cars nowadays. The app is a central piece in a set of tools and services we have developed for black-box analysis of automotive vehicles. We detail its use in the context of real driving emission (RDE) tests and report on sample runs that helped identify violations of the regulatory framework currently valid in the European Union.

1 Introduction

Far more than 600 million cars have entered the streets worldwide [26] in the last decade. With very few exceptions, each of them is equipped with a standardized On-Board-Diagnostics (OBD [41]) interface. Five years ago, it surfaced that many of the cars out there do not adhere to the regulatory framework with which they are supposed to comply. A number of undeniable proofs of tampered emission cleaning systems in passenger cars [6,15,34] are known by now. This scandal was made possible by legislatory frameworks that imposed very few and precisely defined emission tests, to be carried out under laboratory-like conditions on chassis dynamometers upon type approval of a new car model [7,45].

Since then, there has been a growing understanding that measurements of emission and fuel or battery consumption should best be carried out in non-artificial contexts. As a result, the first test framework for testing on public roads, the *real driving emissions* (RDE) test, has been developed [42,44] and is being rolled out for car model approval in Europe and other entities of jurisdictions.

The RDE regulation specifies the conditions under which a car trip qualifies as a valid RDE test. These conditions refer to the trajectory driven, duration, altitudes, speeds, as well as the dynamics of the driving profile [42]. By combining the information available at the OBD port and the position of the car, it is possible to cast RDE testing into a runtime monitoring [29,31,46] problem. Indeed, we have shown in earlier work [25] how to formalize the RDE regulations in RTLOLA [4,22], a real-time extension of the stream-based specification language Lola [16]. Lola combines the ease of use of rule-based specification languages with the expressive power of heavy-weight scripting languages or temporal logics. The eponymous framework generates runtime moni-

✉ Sebastian Biewer
biewer@depend.uni-saarland.de

Bernd Finkbeiner
finkbeiner@cs.uni-saarland.de

Holger Hermanns
hermanns@depend.uni-saarland.de

Maximilian A. Köhl
mkoehl@cs.uni-saarland.de

Yannik Schnitzer
s8yaschn@stud.uni-saarland.de

Maximilian Schwenger
maximilian.schwenger@cispa.saarland

1 Saarland University, Saarland Informatics Campus E1 3, Saarbrücken, Germany

2 CISPA, Saarland Informatics Campus E1 1, Saarbrücken, Germany

Published online: 30 January 2023

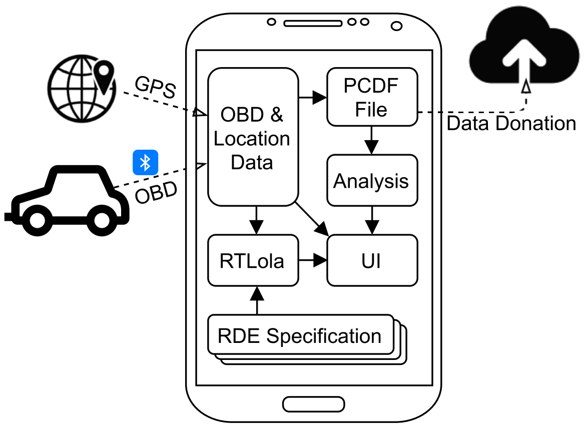


Fig. 1 LolaDrives

tors for such specifications, some of which were successfully deployed, for instance, on unmanned aircraft [4,43].

An official RDE test requires a calibrated *portable emissions measurement system* (PEMS) to be connected to the car's exhaust pipe while driving the test, so as to correctly quantify the amount of exhaust emissions induced. The purchasing costs of a PEMS are in the order of €250,000 which is close to unaffordable even in a research context. However, many car models expose a variety of diagnosis data through OBD, and an OBD-to-Bluetooth adapter can be purchased for around €10. Which types of data are exposed depends on the type of engine, emission cleaning system, and other components in use. There are several minimal combinations of OBD data giving good approximations of emitted gases. All combinations in particular rely on the car model exposing the sensor readings of an NO_x sensor deployed at the rear of the exhaust pipe. Such a sensor is typically part of the exhaust cleaning control loop of systems using selective catalytic reduction.

Contribution This paper introduces LolaDrives, an Android app enabling car owners to carry out real driving emission tests with little investment. Prerequisites to use this app are (i) an Android phone, (ii) an OBD-to-Bluetooth adapter, and (iii) a car model that does indeed expose the needed values via OBD. If the latter is not the case, the app can still serve the user as a convenient personal monitoring and logging device for the many quantities exposed while driving. We provide a detailed account of LolaDrives, its look-and-feel, and its inner working. Moreover, we explain the details of our *Car Data Platform* (CDP), a set of tools and services for car-related, data dependent research. We also discuss our approach to testing LolaDrives during code development.

A structural overview of LolaDrives is depicted in Fig. 1. At the core of the app is an Android version of the RTLola engine [22]. The engine is strictly separated from the data acquisition and the RTLola RDE specification. This separation makes it possible to reuse the approach in other runtime monitoring contexts, be it of espresso machines via USB, or drones via Wi-Fi. In both cases, it is especially the specification in RTLola that needs to change, not the engine. Car sensor data are acquired via Bluetooth from the OBD device, and combined with location data provided by Android's GPS service. The data streams are recorded for later diagnosis in a *Portable Car Data Format* file (PCDF). Anticipating future application scenarios involving crowd sourcing car data, we advertise the app as part of the car data platform, which includes an upload facility for donating drive records. While driving, the app's user interface (UI) displays diagnostic information to the user, both regarding the correct execution of an RDE test drive and the car's emission data.

Notably, the lack of any calibration and the unknown precision of the data exposed by the car manufacturer via OBD make it, in a legal sense, impossible to consider the RDE test results reported by LolaDrives as anything more than indicators of the car's RDE behaviour.

This article is an extended version of a conference publication [9] where a preliminary version of the app was showcased. We here present the latest version of LolaDrives, which has been extended by on-device analytics and a more intuitive user interface. Moreover, we explain the details of our Car Data Platform, a set of tools and services for car-related, data-dependent research. We also provide insights into how to test LolaDrives without having a real car at hand. Finally, we present our findings based on four additional on-the-road experiments with LolaDrives.

2 Context

In 2015, the Diesel Emissions Scandal unveiled an uncomfortable truth about the work of many passenger car manufacturers. Millions of diesel-powered cars were equipped with tampered emission cleaning systems; during official test situations they performed commendably, but in real driving situations they polluted the environment most of the time. Among the early discoveries and most severe cases is Volkswagen. Their cars were equipped with engine control units provably [15,40] containing software components to detect whether the car is undergoing an official emissions test according to the, then effective, admission regulations. These regulations [45] define a driving cycle called *New European Driving Cycle* (NEDC) with the weakness of enforcing tests with a constructed and very unnatural driving behaviour largely containing constant speed phases and repeating patterns—it can be detected easily.

Research and the vehicle regulations consortia have each taken up these weaknesses to suggest solutions. A product of research is a breach of the (proprietary) software running in Volkswagen cars, which allowed gaining valuable insights into their defeat devices [15]. These devices were so precise that even small deviations from the NEDC came with a significant increase of emissions. Small input deviations leading to large output deviations—this observation was taken up by D'Argenio et al. [17] and was turned into a set of formal definitions for *clean* behaviour of systems. Systems that are not clean would be called *doped*, and the usage of defeat devices in diesel cars becomes an instance of *software doping*. The theory is amenable to model-checking [14,23] and testing and has been successfully used to find cases of software doping in practice [7,8,11,19]. The official homologation process has been adapted to better cover real driving behaviour; the NEDC has been replaced by the *Worldwide Harmonised Light Vehicles Test Cycle* (WLTC), which eliminates constant speed phases and pattern repetition, and represents a more dynamic driving behaviour. In addition, a relatively large set of driving cycles is potentially considered for real driving emission tests carried out in-the-wild under presumably realistic conditions. The RDE regulation specifies broad certification conditions for tests conducted under real-word conditions, on public roads and during working days. An (informal) specification document [42] spells out precise preconditions a trip, i. e., a trajectory driven with a car, has to satisfy in order to count as a valid RDE test. These preconditions comprise constraints on the route, allowed altitudes and speeds, and on the dynamics of the driving profile. An RDE test must comprise three modes, the urban, the rural, and the motorway mode covering different speed ranges and each making up approximately one-third of the total trip distance. Table 1 provides an overview of the constraints for all three modes.

In an official RDE test, a calibrated portable emissions measurement system is connected to the car's exhaust pipe and to the OBD interface. It measures the amount of several gases and particles emitted by the car and combines this information with the information received from the OBD interface. As mentioned, the costs of a PEMS are in the order of €250,000.

Nevertheless, without a PEMS it is still possible to access the OBD interface, the use of which is documented in the official regulation [41]. The amount of data offered through OBD depends on the type of engine, emission cleaning system and other components of the car. There are several minimal combinations of OBD data, which can be combined to get a good approximation of emitted gases. Köhl et al. successfully performed RDE tests with an Audi A7 solely using OBD data and indeed observing excessively high emission values, thus indicating a violation of the regulation [28]. One of their major contributions is a formalization of the informal RDE specification in the official regulation. This formalization is written in the Lola specification language [16] with syntactic sugar for discrete aggregation windows [28]. For the present work, we translated this specification to RTLOLA, which is an extended variant of the discrete specification language Lola and the input language. In prior work, RTLOLA was used to monitor log data of networks [21] and drones [3,22]. An automatic synthesis onto a field programmable gate array [5] enabled a flight test with RTLOLA as a monitor of an autonomous drone of the German Aerospace Center [4]. LOLADRIVES brings RTLOLA technology to Android devices for the purpose of RDE testing. It is, however, not the first ever verification and validation tool to run on mobile devices. For example, APHzip [33] is an app for construction and minimization of continuous probability distributions.

2.1 RTLola

RTLOLA [4,22] is a stream-based specification language for real-time properties. An RTLOLA specification is a collection of input stream, output stream and trigger declarations. Input streams represent data sources such as sensors or information retrieved over the OBD interface. Each output stream declaration details how to filter and refine input data to obtain relevant statistical information. Trigger declarations use this information to indicate when the system under observation reaches an undesired state such as a violation of a safety margin or a transgression of the permitted NO_x emission.

Table 1 Some constraints for the three modes of RDE tests [25]

	Urban	Rural	Motorway
Ratio Range [%]	$[29, 44]$	$[23, 43]$	$[23, 43]$
Speed Range [km/h]	$[0, 60]$	$]60, 90]$	$]90, 160]$
Distance [km]	≥ 16	≥ 16	≥ 16
Additional Constraints	stop percentage between 6% and 30% of urban time; average velocity in range $[15, 40]$km/h		$>$ 100km/h for at least 5mins
Temperature [K]	moderate: $[273, 303]$; extended: $[266, 273[$ or $]303, 308]$		
Relative Altitude [m]	start and end point altitudes must not differ by more than 100		
Absolute Altitude [m]	moderate: < 700; extended: $]700, 1300]$		
Speed Limit [km/h]	145 ($]145,160]$ for at most 3% of motorway time)		

The RTLOLA toolkit generates a monitor for a given specification. The monitor receives input data from the system. These data may be asynchronous, i. e., different input sources can produce data at different points in time. Upon reception, the monitor computes output stream values and checks for satisfaction of trigger conditions. The result can then be fed back to the system or displayed to the user.

The RTLOLA language can most easily be understood by example. Consider the following specification:

```
input velocity: Float32
output is_urban: Bool := velocity ≤ 60
```

The specification consists of one input stream and one output stream. The input stream carries the velocity of the system in km/h as a 32-bit wide floating point number. Let us consider the system to operate in an urban environment when its velocity is below 60 km/h. To this end, the Boolean output stream `is_urban` indicates exactly this condition.

To extend the specification, suppose the system is not allowed to travel more than 10 km in an urban environment within 20 min. In this case, the specification can be extended by two output streams and a trigger:

```
  output urban_velo :=
      if is_urban then velocity else 0.0
  output urban_dist @10Hz :=
      urban_velo.aggregate(over: 20min,
          using: integral)
  trigger urban_dist > 10
      "Travelled more than 10km in urban
          env."
```

The first stream, `urban_velo`, carries the velocity of the system provided it operates in an urban environment, or zero otherwise. The second one integrates the urban velocity for 20 min to compute the distance travelled as a sliding window aggregation. Notice the annotation `@10Hz`, which transforms the stream into a *periodic* one. This prompts the monitor to compute the output stream only 10 times a second and is mandatory for streams with a sliding window aggregation operation such as the integral. This mandate allows the monitor to employ an efficient algorithm for aggregation [30]. In particular, if the aggregation additionally is a list homomorphism, the monitor does not have to store input values received within the time frame. This reduces the memory footprint of the monitor drastically. Details can be found in earlier work on RTLOLA [4,38].

Other output streams are *event-based*, i. e., they are computed when the monitor receives new input values. However, some streams depend on more than one input stream. When the monitor receives only an update for a subset of input streams due to asynchrony, it re-computes the output streams for which all relevant inputs were updated. Since the `urban_velo` stream only depends on `velocity`, it will be evaluated upon every reception of this input.

To finalize the example, the trigger declaration states that the `urban_dist` should remain below 10. Specifiers may provide a human-readable explanation after the condition. This string can then be displayed to users.

Beyond this brief introduction into RTLOLA, we refer the reader to earlier work [22,38] for the full syntax and type system of RTLOLA.

3 From regulation to specification

The distinguishing feature of LOLADRIVES relative to all other apps on the market is the ability to monitor the progress of an RDE test while driving. For this, it harvests the RTLOLA monitoring framework bringing formally rigorous runtime monitoring techniques to the end user and every-day use cases.

While RTLOLA targets a broad audience, that audience is still intended to be expert users rather than the general public. To leverage RTLOLA, one has to provide a formal specification capturing the intended behaviour, supply input data, and interpret the monitor's output. LOLADRIVES reduces these tasks to minimal action points for end users. The specification for RDE tests is fixed [28] and baked into the app. The app takes care of supplying the relevant input data, and the app visualizes the monitor's output.

While the app takes care of those details for the end user, it has been a major undertaking to formalize the RDE regulation [42] itself in the RTLOLA framework. We were able to build on top of our earlier efforts [25,28] formalizing the RDE regulation. For the purpose of LOLADRIVES, we extended this previous work. We here elaborate on (a) the formalizing of the RDE regulation and (b) the changes made in order to bring this formalization to the end user.

Regulation The RDE regulation has been issued by the European Commission [42] in order to make exhaust emissions tests more realistic. To this end, it meticulously describes conditions a trip driven on public roads has to satisfy in order to count as a valid RDE test. The regulation itself mostly relies on natural language, however, as we shall demonstrate, the individual conditions translate naturally into a stream-based specification language such as RTLOLA.

Some of the conditions apply universally, e.g., the ambient temperature must range between 273 K and 303 K throughout the whole trip. For others, the RDE regulation differentiates between three modes characterized by the speed of the car: *urban*, *rural*, and *motorway*. Table 1 shows an overview over both the universal conditions and the conditions for the individual modes. Driving in each mode can be interrupted by short periods of driving in another mode, e.g., when changing the motorway the data collected by virtue of the driving speed may count towards the rural or urban environment.

While modes may be interrupted, each one needs to occupy a specific share of the total distance.

The three modes To identify the mode a given data record belongs to, we introduce a Boolean stream for each mode being true if and only if the speed of the car is in the respective range as required for the given mode. For instance, for the rural segment:

```
output is_rural := (60.0 < v) && (v <= 90.0)
```

This directly reflects §6.3 of the regulation [42, ANNEX IIIA] which says: "Rural operation is characterized by vehicle speeds higher than 60 and lower than or equal to 90 km/h." Note that in contrast to other EU regulations, such as domestic market trade laws, the RDE regulation lends itself well to formalization as it clearly defines RDE tests in terms of mathematical concepts. The whole regulation assumes that records of test data come in synchronously at a fixed frequency of at least 1 Hz. Hence, the stream-based specification language RTLOLA is a perfect fit for formalization.

Now, for each of the modes it is required to determine the distance driven in that mode. The ratio d_m/d of the distance d_m driven in a given mode m and the distance driven overall d need to be within a specific interval for each mode (see Table 1 and §6.6 of the regulation [42, ANNEX IIIA]): "The trip shall consist of approximately 34 % urban, 33 % rural and 33 % motorway driving classified by speed as described in points 6.3 to 6.5 above. 'Approximately' shall mean the interval of ±10 percentage points around the stated percentages. The urban driving shall, however, never be less than 29 % of the total trip distance." So, we define a stream for the total distance d:

```
output Dd := v / 3.6 * 1.0
output d @1Hz := Dd.aggregate(over: 2h,
    using: sum)
```

Here Dd is the distance driven since the last data record. Assuming that data are provided with a fixed frequency of 1 Hz, we calculate the distance in m from the velocity v in km/h by dividing by 3.6 to obtain m/s and then multiplying the result with 1.0 s. To obtain the total distance d, we use RTLOLA's aggregation functions and simply take the sum of Dd over the last 2 h, i. e., the maximal duration of a test.

Obtaining the distance travelled in a specific mode is also straightforward: Remember that we have a Boolean stream for each of the modes. We first define an auxiliary stream for each mode whose value is Dd when the car is in the respective mode and 0 otherwise. For instance, for the rural mode, we define r_d_a:

```
output r_d_a := if is_rural then Dd else 0.0
```

Using aggregation functions again, we obtain the distance travelled in the rural mode with:

```
output r_d @1Hz := r_d_a.aggregate(
    over: 2h, using: sum
)
```

So, §6.6 of the RDE regulation translates in part to the following condition for the rural mode:

```
    0.23 <= (if d > 0.0 then r_d / d else
        0.0)
&& (if d > 0.0 then r_d / d else 0.0) <=
    0.43
```

Doing this for all the other modes enables us to compute whether the ratio condition defined in §6.6 is satisfied or not. Analogously to the distance ratios, one defines streams for the remaining conditions.

Driving Dynamics A more complex part of the RDE regulation[1] concerns the driving dynamics. The intuition is simple: Too aggressive driving leads to highly increased emissions; hence, at least for testing, it would not be fair for the manufacturer to have their car evaluated based on unrealistically aggressive driving. Likewise, too restrained driving is unrealistic as well. Hence, the RDE regulation specifies lower and upper bounds on the driving dynamics. The driving dynamics is defined as the product of speed and acceleration:

```
output dyn := v * a / 3.6
```

v is the speed in km/h and a the acceleration in m/s^2. The resulting stream dyn captures the dynamics in $\frac{m^2}{s^3}$. For an analysis of the dynamics in a rural environment, the r_dyn stream mirrors dyn besides discarding values not acquired in a rural environment.

The requirements for the lower bound of the dynamics consider the relative positive acceleration (RPA). The RPA is the sum of the positive values of the dynamics. The regulation defines a dynamics $dyn = \frac{va}{3.6}$ as *positive*, if the acceleration a is at least to 0.1 m/s^2. As shown below, our RTLOLA specifications compute the RPA by first generating an output stream rpa_va that copies dyn but replaces all values computed with a non-positive acceleration by zero. An output stream rpa_agg computes the sum of these values, and output stream rpa computes the RPA by dividing this sum by the length of the trip. The specification snippet below shows the computation of the RPA for the rural mode (hence, each stream name is prefixed with r_):

```
output r_rpa_va :=
      if a >= 0.1 && is_rural then dyn else
         0.0
output r_rpa_agg := r_rpa_va.aggregate(
      over_discrete: 7200, using: sum
).defaults(to: 0.0)
output r_rpa := if r_d > 0 then r_rpa_agg
    / r_d else 0.0
```

[1] See Appendix 7a of ANNEX IIIA [42].

The threshold for the RPA depends on the average velocity. If v_{avg} is the average velocity; then, the regulation requires that the RPA is above $-0.0016 \cdot v_{avg} + 0.1755$ if $v_{avg} \leq 94.05$ and above 0.025 otherwise.

The requirements for the upper bound for the dynamics consider the 95th percentile of the dynamics values. The following specification snippet shows the computation of the 95th percentile for the parts of the RDE test that belong to the rural mode:

```
output r_pctl_dyn @1Hz := r_dyn.aggregate(
  over_discrete: 7200, using: pctl95
).defaults(to: 0.0)
```

The percentile aggregation is computed for up to 7200 time steps, i.e. for a duration of 2 h given the (regulation enforced) fixed frequency of 1 Hz. The computation uses the values of the rural dynamics output stream `r_dyn`. Analogous streams `u_pctl_dyn` and `m_pctl_dyn` are constructed for the urban and, respectively, motorway mode. As for the RPA, the concrete threshold depends on the average velocity v_{avg}: the 95th percentile of the dynamics must not be greater than $0.136 \cdot v_{avg} + 14.44$ if $v_{avg} \leq 74.6$ km/h and no greater than $0.0742 \cdot v_{avg} + 18.966$ otherwise.

We can now encode the validity of the (rural) dynamics as a Boolean stream:

```
output r_is_dynamics_valid :=
  (if r_avg_v <= 94.05 then
    r_rpa > (-0.0016 * r_avg_v + 0.1755)
  else
    r_rpa > 0.025 )
&&
  (if r_avg_v <= 74.6 then
    r_pctl_dyn <= (0.136 * r_avg_v + 14.44)
  else
    r_pctl_dyn <= (0.0742 * r_avg_v +
        18.966) )
```

Checking emissions After establishing that a trip is indeed valid according to the conditions of the RDE regulation, the exhaust emissions have to be checked. If a trip is not valid, the emissions are irrelevant and the test has to be repeated. To calculate the emissions, one first needs the *exhaust mass flow* (EMF), i. e., the mass of exhaust gas emitted per time unit. Based on the EMF in g/s and the measured particles per million, one can then compute the emissions in g/s. From this, the amount of NO_x emission in g/km can be computed by dividing the total amount of NO_x (in gram) emitted during the test by the total trip length. For NO_x and diesel fuel, the respective equations are:

```
output nox_mass_flow :=
  exhaust_mass_flowp * 0.001586 * nox_ppmp
output nox_mass_aggregated :=
  nox_mass_flow.aggregate(over: 2h, using:
      sum)
output nox_per_kilometer :=
  if d > 0.0 then
    nox_mass_aggregated / (d / 1000.0)
  else
    0.0
```

Appendix 4 of the RDE regulation provides a table of factors for computing the emissions. For NO_x and diesel fuel, the necessary factor is 0.001586.

It remains to sum up all the emitted gases over the whole trip in order to calculate the amount of gases emitted per kilometre. For each of the gases, we introduce a stream indicating whether the limit for the respective gas has been exceeded:

```
output nox_exceeded :=
  nox_per_kilometer > 0.168
```

The exact threshold, here 168 mg/km, depends on the emission class of the vehicle.

Finally, we use a trigger to indicate when the trip is valid but the emission limits have been exceeded constituting a violation of the RDE regulation, i. e., that the vehicle did not pass the test (where the Boolean output stream `is_valid_test` is a conjunction of Boolean streams such as `r_is_dynamics_valid` that must necessarily be true for a regulation conforming RDE test):

```
trigger is_valid_test && nox_exceeded
```

As soon as the condition is violated, the trigger goes of notifying the user of the violation.

A vision As we have demonstrated, there is a mostly obvious correspondence between individual sections and paragraphs of the RDE regulation and the respective RTLOLA formal specification. Instead of writing the RDE regulation in natural language, one could have equally well written it in a suitable formal specification language such as RTLOLA. The full specification is available online.[2] It also contains further comments stating which parts of the specification relate to which parts of the regulation.

For the future, we suggest legislators take advantage of formal specification languages to define what is allowed behaviour and what is not. Formal languages enable not only a rigorous and unambiguous definition of allowed behaviour but furthermore the usage of tools, e.g., to synthesize runtime monitors or verify a system for compliance with a regulation.

Dynamic specification While the specification of the RDE regulation itself is fixed, the app may be connected to a variety of cars, not all of which provide the relevant data. For instance, we need the exhaust mass flow (EMF) which is usually measured directly by the PEMS. In case the car does not come equipped with an EMF sensor and we do not have a PEMS at our disposal, we may still be able to calculate the EMF from other data. This has already been

[2] https://www.loladrives.app/scientific-background/.

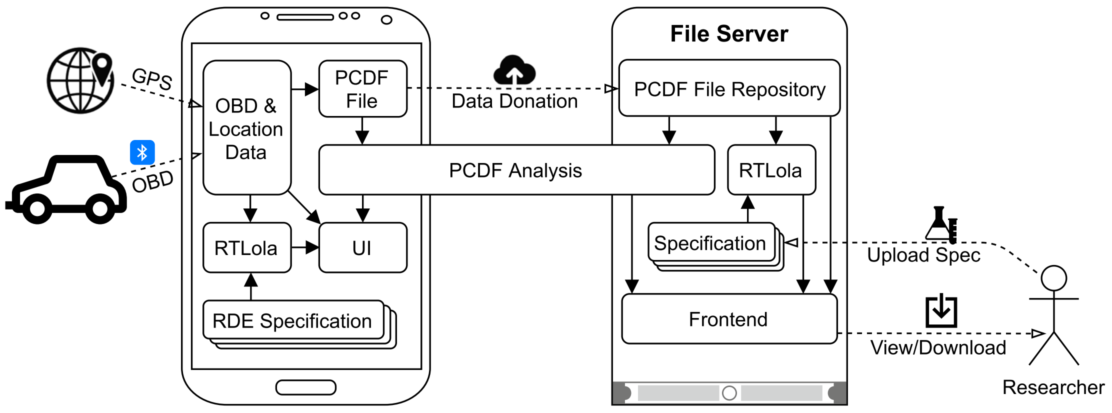


Fig. 2 Components of the car data platform

demonstrated in the previous work and has been the nucleus of the lightweight and low-cost variant of the RDE test procedure [28] LOLADRIVES relies on. Clearly, having the layperson end user change or modify the RTLOLA specification to account for different car configurations is infeasible. Hence, LOLADRIVES automatically adapts the specification to a specific car. To this end, it queries the car for the supported sensors and then automatically pieces together the best specification for the car. This specification then includes the necessary formulae to compute values such as the EMF from data the car actually provides.

4 Car data platform and LOLADRIVES

It is in the nature of Software Doping analysis, that, from the user perspective, systems under investigation are black-box systems, where "observing the system" is the only possibility to gain insights into the software controlling the system [8]. For cars, this raises three concrete questions: 1) how can we observe what the car is doing, 2) how can we accumulate sufficiently many observations to draw conclusions, and 3) how can we address the above two questions in a cost-efficient manner? Our answer to all three questions is CDP, the *Car Data Platform*. It combines several car-related tools and services.

Figure 2 summarizes the main components that belong to the CDP. At the core is the *Portable Car Data Format* (PCDF) to encode car-related diagnostics data in a well-defined way [36]. The central place to collect PCDF files is a data server, which provides interfaces to submit new files, to analyse existing files, and to view analysis results. New files are typically submitted by instances of the mobile app LOLADRIVES, which run in-the-field connected to the diagnostics interface OBD of a car. LOLADRIVES enables and encourages users to donate their PCDF files. CDP offers a repository of analyses for PCDF files [12] that is used by LOLADRIVES instances as well as the server. A server frontend provides researchers with convenient access options regarding the analysis results computed for donated data.

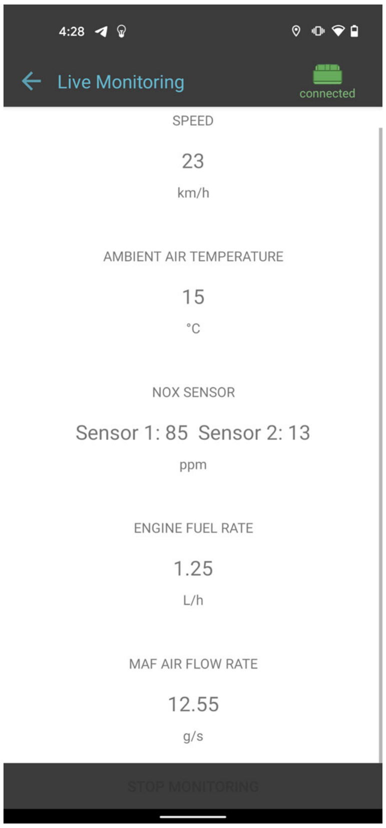
4.1 LOLADRIVES

LOLADRIVES is an Android application publicly available in Google's Play Store [1]. It is compatible with many Bluetooth OBD adapters to access diagnostic data from cars. The app supports two main diagnosis modes: real-time diagnostics monitoring and RDE test guiding.

Diagnostic monitoring In diagnostics monitoring mode, the user selects a set of diagnostic parameters (e.g. vehicle speed, ambient air temperature, etc.), for which real-time values are shown on the screen (see Fig. 3). Monitoring is supported for all cars with combustion engine built since 2005[3].

RDE Testing In RDE test mode, the app constantly analyses the driving behaviour of the user to check whether it satisfies the RDE constraints. The constraints [42] require the driver to equally partition the test into an urban, rural and motorway mode, and to adhere to realistic acceleration and deceleration behaviour. LOLADRIVES displays the most critical RDE parameters (that the driver can influence) by visualizing the evaluations of the RTLOLA streams presented in Sect. 3. This allows the test personnel to easily detect and understand constraint violations. Figure 4 shows the RDE feedback view of LOLADRIVES. From top to bottom, it shows the total time, which must be between 90 and 120 min to finish the test, and the total distance travelled (corresponding to the RTLOLA stream `d`). The next line indicates the current state of the conditions for a valid RDE test drive disregarding emission data.

[3] Some electric vehicles are supported, too. However, it is legally not enforced that electric vehicles expose the OBD protocol at an OBD interface, but they may.



SPEED
23
km/h

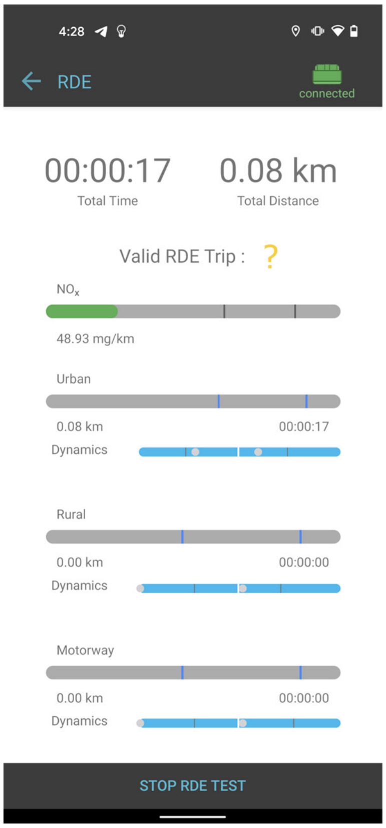
AMBIENT AIR TEMPERATURE
15
°C

NOX SENSOR
Sensor 1: 85 Sensor 2: 13
ppm

ENGINE FUEL RATE
1.25
L/h

MAF AIR FLOW RATE
12.55
g/s

Fig. 3 Diagnostics monitoring view displaying the most recent diagnostics data



00:00:17 Total Time
0.08 km Total Distance

Valid RDE Trip : ?

NO_x
48.93 mg/km

Urban
0.08 km 00:00:17
Dynamics

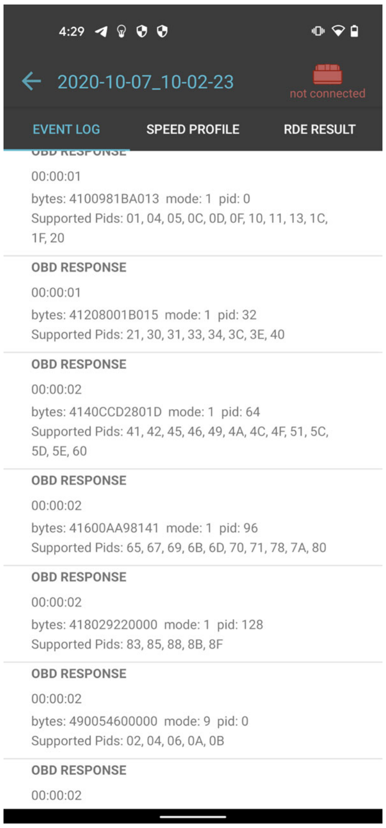
Rural
0.00 km 00:00:00
Dynamics

Motorway
0.00 km 00:00:00
Dynamics

Fig. 4 RDE test guide displaying the current state parameters of the test drive

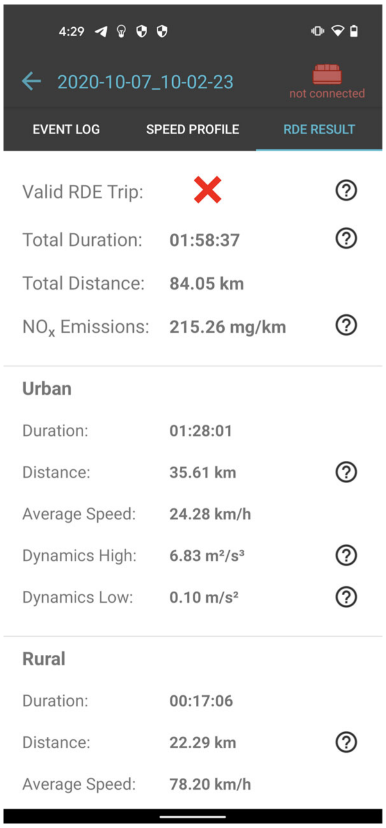
In the screenshot, the drive is still in progress and inconclusive, indicated by the question mark. Instead, the UI can also indicate success or failure. The latter verdict can occur far before the time limit is reached, caused by an irrecoverable situation such as transgression of the 160 km/h speed limit. We remark that currently LOLADRIVES does not always detect if for a test the RDE constraints are irrecoverably violated. For example, if a test has run for 119 minutes, but there are still at least 3 km remaining to drive in the motorway mode to cover the 23 % share of the total trip distance; then, this would require the driver to drive faster than 160 km/h for the remaining minute. Since this is forbidden by the regulation, the RDE constraints are in this moment irrecoverably violated. In the converse case in which the indicator reports a successful drive, this concerns the trip up until this moment. Together with the regulatory constraints, this implies that the current verdict can alternate between success and inconclusive from minute 90 to 120 and also jump to failure. As there is no specific point in time when the test ends, the app continues to compute statistics until the tester manually stops it or the 120-min mark is reached. Beneath the status indicator is the green NO_x bar displaying the total NO_x emissions (RTLOLA `nox_per_kilometer` stream). The two markings denote the permitted thresholds of 168 mg/km for cars admitted before 2021 and 120 mg/km for cars admitted in 2021 or later.

The next three UI groups represent the progress in each of the distinct modes: urban, rural, and motorway. Each group consists of two horizontal bars. The grey progress bar displays the distance covered in the respective mode (e.g. RTLOLA `r_d` stream for the rural mode). The vertical blue indicators denote lower and upper bounds as per official regulation, for an expected trip length configured by the user. We remark that the configured trip length is solely used to determine the initial position of the distance indicators. In particular, when the user drives the car so that the distance would cross the upper bound of a mode, then LOLADRIVES instead increases the upper bound as necessary to avoid an overstepping and updates the distance bound indicators for the other two modes accordingly. The blue bar below the grey one illustrates two different metrics for the driving dynamics (e.g. RTLOLA `r_rpa` and `r_pctl_dyn` streams for the rural mode). Both dots need to eventually remain in the middle of the bar below/above their thresholds. A more aggressive acceleration behaviour shifts the dots to the right and a passive driving style to the left. The RDE test guide is available for cars with compatible diagnosis characteristics. Necessary diagnostic parameters to check the RDE constraints are



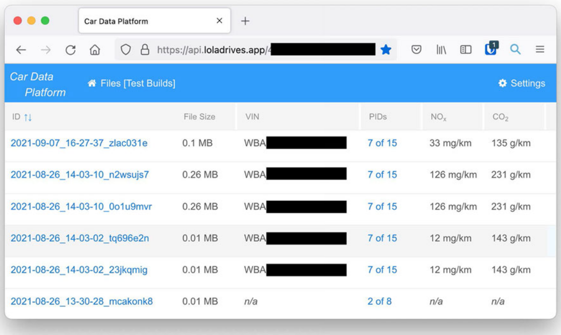
```
OBD RESPONSE
00:00:01
bytes: 4100981BA013  mode: 1  pid: 0
Supported Pids: 01, 04, 05, 0C, 0D, 0F, 10, 11, 13, 1C,
1F, 20

OBD RESPONSE
00:00:01
bytes: 41208001B015  mode: 1  pid: 32
Supported Pids: 21, 30, 31, 33, 34, 3C, 3E, 40

OBD RESPONSE
00:00:02
bytes: 4140CCD2801D  mode: 1  pid: 64
Supported Pids: 41, 42, 45, 46, 49, 4A, 4C, 4F, 51, 5C,
5D, 5E, 60

OBD RESPONSE
00:00:02
bytes: 41600AA98141  mode: 1  pid: 96
Supported Pids: 65, 67, 69, 6B, 6D, 70, 71, 78, 7A, 80

OBD RESPONSE
00:00:02
bytes: 418029220000  mode: 1  pid: 128
Supported Pids: 83, 85, 88, 8B, 8F

OBD RESPONSE
00:00:02
bytes: 490054600000  mode: 9  pid: 0
Supported Pids: 02, 04, 06, 0A, 0B

OBD RESPONSE
00:00:02
```

Fig. 5 All recorded files can be inspected data packet by data packet



Valid RDE Trip:	❌
Total Duration:	01:58:37
Total Distance:	84.05 km
NO_x Emissions:	215.26 mg/km
Urban	
Duration:	01:28:01
Distance:	35.61 km
Average Speed:	24.28 km/h
Dynamics High:	6.83 m²/s³
Dynamics Low:	0.10 m/s²
Rural	
Duration:	00:17:06
Distance:	22.29 km
Average Speed:	78.20 km/h

Fig. 6 For recorded RDE tests, an RDE summary view is shown RDE specific parameters



ID ↑↓	File Size	VIN	PIDs	NO_x	CO_2
2021-09-07_16-27-37_zlac031e	0.1 MB	WBA[redacted]	7 of 15	33 mg/km	135 g/km
2021-08-26_14-03-10_n2wsujs7	0.26 MB	WBA[redacted]	7 of 15	126 mg/km	231 g/km
2021-08-26_14-03-10_0o1u9mvr	0.26 MB	WBA[redacted]	7 of 15	126 mg/km	231 g/km
2021-08-26_14-03-02_tq696e2n	0.01 MB	WBA[redacted]	7 of 15	12 mg/km	143 g/km
2021-08-26_14-03-02_23jkqmig	0.01 MB	WBA[redacted]	7 of 15	12 mg/km	143 g/km
2021-08-26_13-30-28_mcakonk8	0.01 MB	*n/a*	2 of 8	*n/a*	*n/a*

Fig. 7 Server Frontend: list of internal test files

vehicle speed, fuel type and the ambient air temperature. Furthermore, LOLADRIVES needs access to the location services of the phone to check RDE conditions that are concerned with the altitude of the car. To compute the amount of emitted NO_x, the exhaust mass flow and relative amount of NO_x as measured in the exhaust pipe are necessary. The exhaust mass flow can be approximated from the mass air flow into the engine and the fuel rate [28]. Instead of the fuel rate, the air–fuel ratio can also be used. In case, neither is available, an expected air-fuel ratio based on the fuel type can be used.

Drive History In both monitoring and RDE testing modes, the data received from the car are stored in a PCDF file. All recorded files can be inspected in the "History" section of LOLADRIVES. It is possible to inspect the raw data received from the car (Fig. 5), but also results of the analyses once it is available (e.g. RDE results in Fig. 6). Every analysis has individual requirements about the set of diagnostics parameters that must be available in the record.

4.2 File server

To collect PCDF files at a central place, CDP provides a file server with an easy to use API to submit new files and to get analysis results. The server provides two distinct repositories, one for internal usage by the developers and one for official usage of the PlayStore version protected by a strong privacy policy. Figure 7 shows the web frontend for the internal repository.

Internal files can be downloaded, and several analysis results can be inspected. Accessing and submitting data is protected by authorization tokens. The implementation of the server is an interplay of individual components, deliberately held flexible so that components can be replaced, removed or extended by other components in the future.

4.3 Technical set-up

The Car Data Platform is a collection of tools and services for car-related research. It is designed in a modular way to allow for a flexible development in the future. Modules are encapsulated in software packages, some of them are publicly available online. The implementation of the analyses shared by LOLADRIVES and the file server, and the PCDF core for easy handling of PCDF files is written in Kotlin and published online [12,36]. Maven artefacts are available for integration in other projects. An analysis worker (written in Kotlin) is running on the server to regularly check whether new files have been uploaded and to run the analyses on new files in the background. The file server backend is an npm package written in TypeScript and Express. The frontend is also an npm package providing a react UI to show the contents on the server.

The entire RTLOLA toolkit is written in Rust and available online[4]. The Rust compiler uses LLVM as a backend, which enables compilation for Android devices. Moreover, the implementation of the interpreter[5] contains both a standalone interpreter and a library. The library exposes a C-compatible interface, which can in turn interface with Java Virtual Machine (JVM)-based languages such as Kotlin. This enables linking of RTLOLA and LOLADRIVES. The latter is written in Kotlin and also freely available online [37].

Car simulator By the nature of the app's functionality, the testing of new features of LOLADRIVES requires it running on a phone, connected via Bluetooth to an OBD adapter, which is plugged into a car being driven by a human. This overhead makes testing quite inefficient. We therefore simplified the test procedure by instead constructing a physical car simulator, to which the OBD adapter can be connected. The simulator consists of two parts: 1) a regular PC software to parse and prepare PCDF files for simulation and 2) an Arduino board attached to a CAN bus shield. The Arduino board serves as a "diagnosis storage device" to which the PC repeatedly writes diagnostic data. The CAN bus shield is connected to the OBD adapter; it reads the diagnostics data from the diagnosis storage via the OBD protocol. The PC transmits each event in the source PCDF file to the board in the same order and with the same delay as it was recorded. PC and Arduino communicate via a protocol based on *Consistent Overhead Byte Stuffing* (COBS) [13]. The PC software is written in Kotlin and the code on the board is written in the typical C++-based Arduino language. Figure 8 shows the simulator in action.

Privacy An important feature of LOLADRIVES and CDP is the support for data donation; users can opt-in to upload the files recorded by LOLADRIVES during monitoring or RDE test mode. But PCDF files may contain personal data, for example, the vehicle identification number or GPS coordinates. Collecting personal data is regulated by data protection laws; in our case, the General Data Protection Regulation (GDPR). The GDPR concedes every EU citizen the rights to receive a copy of their data (in a machine readable format), to have it corrected, or deleted. All privacy policies in the EU must educate these rights to the users. LOLADRIVES does so in full. Moreover, our privacy policy [10] explains that data uploads are automatically deleted after at most 15 years (counting 5 years for doing research with it and 10 years data retention time after publication recommended by the German Research Council DFG [18]). Data donations are voluntary. Refusing or withdrawing consent does not restrict the available features of LOLADRIVES in any way.

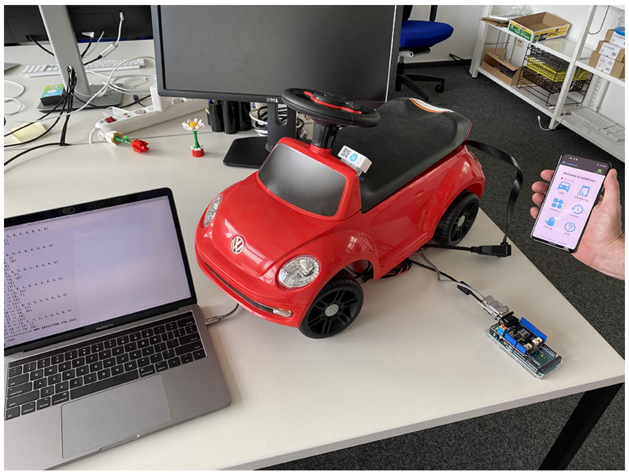


Fig. 8 OBD simulator

5 Demonstration

This section discusses the user perspective on LOLADRIVES. After a general overview, we report on the use of LOLADRIVES for conducting RDE test drives with two rented vehicles (the precise car model being unknown upfront).

Overview The preparation of the test requires the user to plug the OBD-adapter into the OBD-port of the car. After starting car and app, LOLADRIVES receives data packets and determines the sensor profile of the car, assuming phone and adapter are paired via Bluetooth. As the provided diagnostics data suffices to evaluate the RDE constraints, the app selects the appropriate RTLOLA specification and initializes the RTLOLA monitor. LOLADRIVES then starts filtering and visualizing the data output and trigger notifications provided by the monitor, as explained in Section 4.

[4] https://rtlola.org.

[5] https://crates.io/crates/rtlola-interpreter.

Test drive The technical framework and visual feedback of the app were tested in two experiments. The first experiments involved two RDE test drives that were both conducted with an Audi A6 Avant 45-TDI hybrid diesel, which was admitted in 2020 under the Euro 6d-TEMP-EVAP-ISC (DG) regulation with an NO_x threshold of 80 mg/km under lab conditions and 168 mg/km for RDE conditions. We denote this car as *A20* and the RDE tests as *A20.1* and *A20.2*. The second experiment involves four RDE tests that were conducted with the successor of the above car—an Audi A6 50-TDI hybrid diesel admitted in 2021 under the (moderately stricter) Euro 6d-ISC-FCM (AP) regulation enforcing the same NO_x threshold of 80 mg/km under lab conditions and a smaller 120 mg/km threshold for RDE conditions.[6] We denote this car as *A21* and the RDE tests done with this car as *A21.1* to *A21.4*.[7] Among the diagnosis parameters available within these cars are vehicle and engine speed, ambient temperature, engine fuel rate and mass air flow. The A20 car has two NO_x-sensors—one in front and one behind the emission cleaning system in the exhaust pipe. The A21 car has three NO_x-sensors—presumably one in front, one between components of the emission cleaning system and one behind it. With this set of sensors, the car is compatible for RDE tests with LolaDrives. We configured LolaDrives to assume an expected trip length of 83 km for the visual guidance.

Test drives A20.1 and A21.4 meet all conditions to be considered as a valid RDE test. Test drives A20.2 and A21.1 did not experience sufficiently much accelerations in the urban mode to be a valid RDE test; in A21.2 there was a malfunctioning of the OBD adapter and the test was forced to end before reaching the minimal 23 % share of the total trip length in the motorway mode; and test drive A21.3 is invalid, because we failed to comply to the maximum altitude difference of 100 m between start and end points. In all cases, LolaDrives correctly confirmed the satisfaction and violation of the RDE criteria. For the valid tests A20.1 and A21.4, we measured 215 mg/km and, respectively, 30 mg/km of NO_x emissions. Hence, test A20.1 reveals a violation of the RDE regulation, while the result of test A21.4 is conforming to it. A comprehensive overview for all measured emissions is shown in Table 2. In this table, the distances and the total amounts of emitted NO_x are computed directly by LolaDrives; all other values have been computed using a custom RTLola specification that was applied to the trip recordings after they were uploaded to the CDP by LolaDrives's data donation feature. Notice that the NO_x value for A21.3 is significantly higher than for the other A21 drives. Additional diagnostics data

[6] We determined the precise car model and the variant of the Euro 6d norm using the registration certificate of the car and the German Wikipedia [48,49]

[7] The records A21.1, A21.2 and A21.4 have already been used in other work [11]; the RDE aspect was not discussed there.

Table 2 Aggregation of the emission data based on the CDP

	Drive A20.1			Drive A20.2			Drive A21.1		
	Distance [km]	NO_x [mg/km]	CO_2 [g/km]	Distance [km]	NO_x [mg/km]	CO_2 [g/km]	Distance [km]	NO_x [mg/km]	CO_2 [g/km]
Urban	35.61	138	221	37.42	111	250	43.54	31	221
Rural	22.29	303	155	27.46	82	170	29.17	11	121
Motorway	26.15	245	153	25.33	103	176	28.73	25	166
Total	84.05	215	183	90.24	100	205	101.44	23	183
	Drive A21.2			Drive A21.3			Drive A21.4		
	Distance [km]	NO_x [mg/km]	CO_2 [g/km]	Distance [km]	NO_x [mg/km]	CO_2 [g/km]	Distance [km]	NO_x [mg/km]	CO_2 [g/km]
Urban	28.75	42	230	37.96	76	227	36.46	36	219
Rural	30.88	19	164	29.49	222	199	25.29	21	142
Motorway	15.66	17	137	43.01	41	145	34.42	29	150
Total	75.29	27	184	110.46	101	188	96.17	30	174

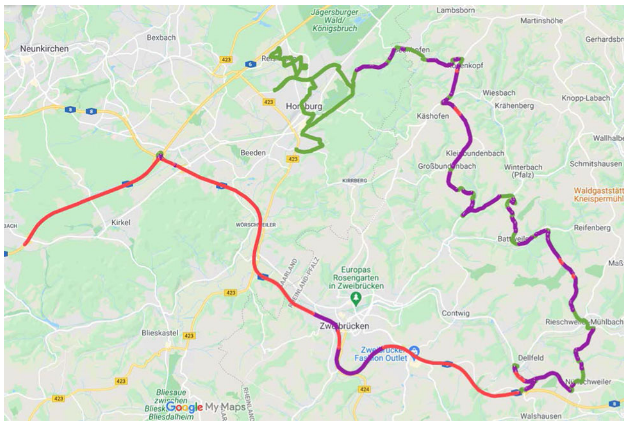


Fig. 9 Map with the A20.2 test route highlighted

recorded made it evident that the car had been cleaning its diesel particulate filter and NO$_x$ adsorber during the test. We are, however, not certain if this is the reason for the higher emissions. Anyway, if A21.3 were a valid RDE test, the overall NO$_x$ emissions are still below the threshold of 120 mg/km defined in the regulation.

Figure 9 shows the route of test drive A20.2. The first half of the time constituted the urban segment (green). The next 30-40% of the test mainly consisted of the rural segment (purple) followed by the motorway segment (red). The map shows that the rural and motorway segments are regularly interrupted by other segments when the driver had to slow down for traffic reasons; the three phases are solely defined by the vehicle speed. As a result, depending on external circumstances, the driver cannot freely choose their environment, potentially exceeding the distance thresholds for a different segment on accident. It is, therefore, advisable to start with the urban environment and progress to the next environment as early as possible.

6 Conclusion

Since the Diesel Emissions Scandal in 2015, driving a diesel car comes with some uncomfortable feelings about how much the car actually does pollute the environment. With LOLADRIVES, we aim at more perspicuity and transparency and provide car owners a tool to investigate the ecological footprint of their car and drivings. To this end, LOLADRIVES pushes runtime verification technology into cars and phones of everyday users. The app is available in *Google Play* [1]. In the context of ecological responsibility, this kind of research plays an important role [27]. The car data platform constitutes a crowd-sourcing initiative for car data with the intention to enable researchers to conduct large-scale analyses of emission data beyond a single trip and a single car model.

The current state of the CDP is a proof of concept that can be further developed into several, not mutually exclusive directions. LOLADRIVES was launched in the middle of a global climate crisis. Diesel cars, but also gasoline cars aggravate the situation. Among the most promising modalities to reduce the impact of individual mobility on the climate crisis are electric vehicles. Electric vehicle technology promises a decarbonization process that is deemed necessary with respect to the ongoing climate crisis [35,39,50].

Unfortunately, electric vehicles are not legally enforced to provide OBD interfaces. Instead, manufacturer-specific protocols [47] are in place for acquiring diagnostic data from these cars, and some of the protocol specifications have been leaked to the public [2]. Clearly, emission data of electric vehicles is irrelevant, but there are other diagnostics data that are of substantial importance for research and for users. Indeed, a measurable success of the decarbonization approach discussed hinges on two quantifiable characteristics: *energy efficiency* (the power consumption of the car must be as low as possible), and *sufficiency* (electricity savings by refraining from what is possible but not necessary) [35, Chapter 9][32]. For the latter, a first step is to make users aware of which changes in their driving behaviour have positive or negative effects on the energy consumption of their car. For example, fast accelerations typically consume more energy than moderate accelerations. In this regard, we envision new CDP components that provide app-based feedback to the drivers about their driving behaviour and recommendations on how to change their driving behaviour to become a *sufficient* driver.

We have already initiated work on further CDP components, for example an iOS-version of LOLADRIVES, and a "CDP core", a library of multiple CDP components used by all CDP applications (mobile and server). The PCDF *analysis* component (with non-stream-based analyses) from Fig. 2 is the first part of this library. More analyses are planned for the future.

A recent approach [11] harvests the data collected with LOLADRIVES to improve test input selection for (model-based) software doping tests. There, a car is used on the road while RDE-related diagnostic data is recorded. Once sufficiently many recordings are available, the data are used to model an emissions prediction function for the car. This prediction function can be combined with a probabilistic falsification technique to find test cycles that likely reveal software doping of the car. To verify this, the resulting test cycle must be driven on a chassis dynamometer with an official emissions measurement device attached to the exhaust pipe.

For stream-based analyses with RTLOLA, we are planning a native integration into the server frontend. Researchers can then upload an RTLOLA specification and let the server run the analysis on selected files. For the public repository (protected by privacy laws), we are planning to accept only specifications that do not reveal data of individual donators, which will require non-interference [24] and differential privacy [20] analyses of specifications.

Acknowledgements We thank everyone who tested or used LOLADRIVES, or will do so in the future. Thanks to Marc Großjean for his support when we became lost in the jungle of privacy policies. We also thank Lorenz Hetterich, who built the car simulator for testing LOLADRIVES. Many thanks go to Florian Schießl, Felix Freiberger and Gregory Stock, who jointly keep our technical infrastructure running, and who have spent a lot of effort to make the CDP possible. We also owe special thanks to Sarah Sterz, who designed the gorgeous LOLADRIVES logo and produced great tutorial videos for loladrives.app! This work is partly supported by DFG grant 389792660 as part of TRR 248 – CPEC, by the European Research Council (ERC) grants 683300 (OSARES), 695614 (POWVER), and 966770 (LEOpowver), and by the Key-Area Research and Development Program Grant 2018B010107004 of Guangdong Province.

Funding Open Access funding enabled and organized by Projekt DEAL.

Declarations

Legal Attribution Android, Google Play and the Google Play logo are trademarks of Google LLC.

References

1. LolaDrives web page. https://loladrives.app
2. DDT4All: (2021) https://github.com/cedricp/ddt4all
3. Adolf, F., Faymonville, P., Finkbeiner, B., Schirmer, S., Torens, C.: Stream runtime monitoring on UAS. In: RV 2017, LNCS, vol. 10548, pp. 33–49. Springer (2017). https://doi.org/10.1007/978-3-319-67531-2_3
4. Baumeister, J., Finkbeiner, B., Schirmer, S., Schwenger, M., Torens, C.: RTLola cleared for take-off: monitoring autonomous aircraft. In: CAV 2020, LNCS, vol. 12225, pp. 28–39. Springer (2020). https://doi.org/10.1007/978-3-030-53291-8_3
5. Baumeister, J., Finkbeiner, B., Schwenger, M., Torfah, H.: FPGA stream-monitoring of real-time properties. ACM Trans. Embedd. Comput. Syst. **18**(5s), 88:1-88:24 (2019). https://doi.org/10.1145/3358220
6. BBC: Audi chief Rupert Stadler arrested in diesel emissions probe. BBC, https://www.bbc.com/news/business-44517753 (2018). Online; accessed: 2020-10-15
7. Biewer, S., D'Argenio, P., Hermanns, H.: Doping tests for cyber-physical systems. In: D. Parker, V. Wolf (eds.) Quantitative Evaluation of Systems, In: 16th international conference, QEST 2019, Glasgow, UK, September 10-12, 2019, Proceedings, Lecture notes in computer science, vol. 11785, pp. 313–331. Springer (2019). https://doi.org/10.1007/978-3-030-30281-8_18
8. Biewer, S., D'argenio, P.R., Hermanns, H.: Doping tests for cyber-physical systems. ACM Trans. Model. Comput. Simul. (2021). https://doi.org/10.1145/3449354
9. Biewer, S., Finkbeiner, B., Hermanns, H., Köhl, M.A., Schnitzer, Y., Schwenger, M.: Rtlola on board: testing real driving emissions on your phone. In: J.F. Groote, K.G. Larsen (eds.) Tools and algorithms for the construction and analysis of systems. In: 27th international conference, TACAS 2021, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2021, Luxembourg City, Luxembourg, March 27 - April 1, 2021, Proceedings, Part II, Lecture Notes in Computer Science, vol. 12652, pp. 365–372. Springer (2021). doi: https://doi.org/10.1007/978-3-030-72013-1_20
10. Biewer, S., Hermanns, H.: LolaDrives (App) Privacy Policy. https://www.loladrives.app/app-privacy-statement/
11. Biewer, S., Hermanns, H.: On the detection of doped software by falsification. In: E.B. Johnsen, M. Wimmer (eds.) Fundamental Approaches to Software Engineering - 25th international conference, FASE 2022, Held as part of the European joint conferences on theory and practice of software, ETAPS 2022, Munich, Germany, April 2-7, 2022, Proceedings, Lecture Notes in Computer Science, vol. 13241, pp. 71–91. Springer (2022). https://doi.org/10.1007/978-3-030-99429-7_4
12. Biewer, S., Schnitzer, Y.: PCDF analyser (2021). https://github.com/udsdepend/pcdf-analyser
13. Cheshire, S., Baker, M.: Consistent overhead byte stuffing. In: C. Diot, C. Huitema, S. Shenker, M. Steenstrup (eds.) Proceedings of the ACM SIGCOMM 1997 conference on applications, technologies, architectures, and protocols for computer communication, September 14-18, 1997, Cannes, France, pp. 209–220. ACM (1997). https://doi.org/10.1145/263105.263168
14. Coenen, N., Finkbeiner, B., Sánchez, C., Tentrup, L.: Verifying hyperliveness. In: I. Dillig, S. Tasiran (eds.) Computer Aided Verification - 31st international conference, CAV 2019, New York City, NY, USA, July 15-18, 2019, Proceedings, Part I, Lecture Notes in Computer Science, vol. 11561, pp. 121–139. Springer (2019). https://doi.org/10.1007/978-3-030-25540-4_7
15. Contag, M., Li, G., Pawlowski, A., Domke, F., Levchenko, K., Holz, T., Savage, S.: How they did it: an analysis of emission defeat devices in modern automobiles. In: 2017 IEEE Symposium on Security and Privacy, SP 2017, San Jose, CA, USA, May 22-26, 2017, pp. 231–250. IEEE Computer Society (2017). https://doi.org/10.1109/SP.2017.66
16. D'Angelo, B., Sankaranarayanan, S., Sánchez, C., Robinson, W., Finkbeiner, B., Sipma, H.B., Mehrotra, S., Manna, Z.: Lola: runtime monitoring of synchronous systems. In: TIME 2005, pp. 166–174. IEEE Computer Society Press (2005). https://doi.org/10.1109/TIME.2005.26
17. D'Argenio, P.R., Barthe, G., Biewer, S., Finkbeiner, B., Hermanns, H.: Is your software on dope? - Formal analysis of surreptitiously "enhanced" programs. In: programming languages and systems - 26th European symposium on programming, ESOP 2017, Proceedings, LNCS, vol. 10201, pp. 83–110. Springer (2017). https://doi.org/10.1007/978-3-662-54434-1_4
18. Deutsche Forschungsgemeinschaft / German Research Foundation: Guidelines for safeguarding good research practice – code of conduct. https://www.dfg.de/download/pdf/foerderung/rechtliche_rahmenbedingungen/gute_wissenschaftliche_praxis/kodex_gwp_en.pdf

19. Dimitrova, R., Gazda, M., Mousavi, M.R., Biewer, S., Hermanns, H.: Conformance-based doping detection for cyber-physical systems. In: A. Gotsman, A. Sokolova (eds.) Formal techniques for distributed objects, components, and systems - 40th IFIP WG 6.1 international conference, FORTE 2020, Held as part of the 15th international federated conference on distributed computing techniques, DisCoTec 2020, Valletta, Malta, June 15-19, 2020, Proceedings, Lecture notes in computer science, vol. 12136, pp. 59–77. Springer (2020). https://doi.org/10.1007/978-3-030-50086-3_4
20. Dwork, C.: Differential privacy: a survey of results. In: M. Agrawal, D. Du, Z. Duan, A. Li (eds.) Theory and Applications of Models of Computation, 5th international conference, TAMC 2008, Xi'an, China, April 25-29, 2008. Proceedings, Lecture Notes in Computer Science, vol. 4978, pp. 1–19. Springer (2008). https://doi.org/10.1007/978-3-540-79228-4_1
21. Faymonville, P., Finkbeiner, B., Schirmer, S., Torfah, H.: A Stream-Based Specification Language for Network Monitoring, pp. 152–168. Springer, Cham (2016)
22. Faymonville, P., Finkbeiner, B., Schledjewski, M., Schwenger, M., Stenger, M., Tentrup, L., Torfah, H.: StreamLAB: stream-based monitoring of cyber-physical systems. In: CAV 2019, LNCS, vol. 11561, pp. 421–431. Springer (2019). https://doi.org/10.1007/978-3-030-25540-4_24
23. Finkbeiner, B., Rabe, M.N., Sánchez, C.: Algorithms for model checking HyperLTL and HyperCTL*. In: D. Kroening, C.S. Pasareanu (eds.) Computer Aided Verification - 27th international conference, CAV 2015, San Francisco, CA, USA, July 18-24, 2015, Proceedings, Part I, Lecture Notes in Computer Science, vol. 9206, pp. 30–48. Springer (2015). https://doi.org/10.1007/978-3-319-21690-4_3
24. Goguen, J.A., Meseguer, J.: Security policies and security models. In: 1982 IEEE symposium on security and privacy, Oakland, CA, USA, April 26-28, 1982, pp. 11–20. IEEE computer society (1982). https://doi.org/10.1109/SP.1982.10014
25. Hermanns, H., Biewer, S., D'Argenio, P.R., Köhl, M.A.: Verification, testing, and runtime monitoring of automotive exhaust emissions. In: LPAR, pp. 1–17 (2018). https://doi.org/10.29007/6zxt
26. International Organization of Motor Vehicle Manufacturers: 2005-2019 sales statistics http://www.oica.net/category/sales-statistics
27. IPCC (Intergovernmental Panel on Climate Change): Climate change 2014: synthesis report. contribution of working groups i, ii and iii to the fifth assessment report of the intergovernmental panel on climate change (2014)
28. Köhl, M.A., Hermanns, H., Biewer, S.: Efficient monitoring of real driving emissions. In: C. Colombo, M. Leucker (eds.) Runtime verification - 18th international conference, RV 2018, Limassol, Cyprus, November 10-13, 2018, Proceedings, Lecture notes in computer science, vol. 11237, pp. 299–315. Springer (2018). https://doi.org/10.1007/978-3-030-03769-7_17
29. Lee, I., Kannan, S., Kim, M., Sokolsky, O., Viswanathan, M.: Runtime assurance based on formal specifications. In: H.R. Arabnia (ed.) Proceedings of the international conference on parallel and distributed processing techniques and applications, PDPTA 1999, June 28 - Junlly 1, 1999, Las Vegas, Nevada, USA, pp. 279–287. CSREA Press (1999)
30. Li, J., Maier, D., Tufte, K., Papadimos, V., Tucker, P.A.: No pane, no gain: efficient evaluation of sliding-window aggregates over data streams. SIGMOD Rec. **34**(1), 39–44 (2005). https://doi.org/10.1145/1058150.1058158
31. Moosbrugger, P., Rozier, K.Y., Schumann, J.: R2U2: monitoring and diagnosis of security threats for unmanned aerial systems. Formal Methods Syst. Des. **51**(1), 31–61 (2017). https://doi.org/10.1007/s10703-017-0275-x
32. Princen, T.: The logic of sufficiency. MIT Press, Cambridge (2005)
33. Pulungan, R., Hermanns, H.: A construction and minimization service for continuous probability distributions. Int. J. Softw. Tools Technol. Transf. **17**(1), 77–90 (2015). https://doi.org/10.1007/s10009-013-0296-8
34. Riley, C.: Volkswagen's diesel scandal costs hit $30 billion. CNN Business (2018). https://money.cnn.com/2017/09/29/investing/volkswagen-diesel-cost-30-billion/index.html. Online; accessed: 2020-10-15
35. Schneidewind, U.: Die große Transformation: eine Einführung in die Kunst gesellschaftlichen Wandels. S. Fischer Verlag (2018)
36. Schnitzer, Y.: PCDF core (2021). https://github.com/udsdepend/pcdf-core
37. Schnitzer, Y., Biewer, S.: LolaDrives Android (2021). https://github.com/udsdepend/loladrives-android
38. Schwenger, M.: Statically Analyzed Stream Monitoring for Cyber-Physical Systems. Dissertation, Saarland University (2022)
39. Skea, J., Shukla, P., Kılkış, Ş.: Climate change 2022: Mitigation of climate change (2022)
40. Taylor, M.: EU's highest court deals Volkswagen yet another Dieselgate emissions cheat blow. Forbes (2020). https://www.forbes.com/sites/michaeltaylor/2020/12/17/eus-highest-court-deals-volkswagen-yet-another-dieselgate-emissions-cheat-blow/. Online; accessed: 2021-09-10
41. The European Parliament and the Council of the European Union: Directive 98/69/ec of the european parliament and of the council. Official Journal of the European Communities (1998). http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=CELEX:31998L0069:EN:HTML
42. The European Parliament and the Council of the European Union: Commission Regulation (EU) 2017/1151 (2017). http://data.europa.eu/eli/reg/2017/1151/oj
43. Torens, C., Adolf, F., Faymonville, P., Schirmer, S.: Towards intelligent system health management using runtime monitoring. In: AIAA Information systems-AIAA Infotech @ Aerospace. American Institute of Aeronautics and Astronautics (AIAA) (2017). https://doi.org/10.2514/6.2017-0419
44. Tutuianu, M., Bonnel, P., Ciuffo, B., Haniu, T., Ichikawa, N., Marotta, A., Pavlovic, J., Steven, H.: Development of the world-wide harmonized light duty test cycle (wltc) and a possible pathway for its introduction in the European legislation. Transport. Res. Part D Transp. Environ. **40**, 61–75 (2015). https://doi.org/10.1016/j.trd.2015.07.011
45. United Nations: UN Vehicle Regulations - 1958 Agreement, Revision 2, Addendum 100, Regulation No. 101, Revision 3 — E/ECE/324/Rev.2/Add.100/Rev.3 (2013). http://www.unece.org/trans/main/wp29/wp29regs101-120.html
46. Watanabe, K., Kang, E., Lin, C., Shiraishi, S.: Runtime monitoring for safety of intelligent vehicles. In: Proceedings of the 55th annual design automation conference, DAC 2018, San Francisco, CA, USA, June 24-29, 2018, pp. 31:1–31:6. ACM (2018). https://doi.org/10.1145/3195970.3199856
47. Wikipedia: Keyword protocol 2000 (2021). https://en.wikipedia.org/wiki/Keyword_Protocol_2000.Online;accessed:2021-09-13
48. Wikipedia: Abgasnorm — wikipedia, die freie enzyklopädie (2022). https://de.wikipedia.org/w/index.php?title=Abgasnorm&oldid=223638095.Online;accessed:2022-05-14
49. Wikipedia: Audi A6 C8 — wikipedia, die freie enzyklopädie (2022). https://de.wikipedia.org/w/index.php?title=Audi_A6_C8&oldid=221632578. Online; accessed: 2022-05-14
50. World Commission on Environment and Development: Our Common Future. Oxford University Press (1987)