A demonstration of a service oriented virtual environment for complex system analysis

This item was submitted to Loughborough University’s Institutional Repository by the/an author.


Additional Information:

- This paper was published in the International Journal of Complex Systems - Computing, Sensing and Control and reproduced with permission of the publisher

Metadata Record: https://dspace.lboro.ac.uk/2134/26447

Version: Published

Publisher: TSI Press

Rights: This work is made available according to the conditions of the Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International (CC BY-NC-ND 4.0) licence. Full details of this licence are available at: https://creativecommons.org/licenses/by-nc-nd/4.0/

Please cite the published version.
A DEMONSTRATION OF A SERVICE ORIENTED VIRTUAL ENVIRONMENT FOR COMPLEX SYSTEM ANALYSIS

C. E. DICKERSON, S. JI, S. J. CLEMENT
Wolfson School of Engineering
Loughborough University
Loughborough, UK
{c.dickerson, s.ji, s.j.clement}@lboro.ac.uk

D. WEBSTER, D. McKee,
J. Xu
School of Computing
University of Leeds
Leeds, UK
{D.E.Webster, scdwm, J.Xu}@leeds.ac.uk

D. BATTERSBY, N. BEVAN,
N. TURNER, W. SUART
Jaguar Land Rover
Gaydon, UK
{dbatters, nbevan3, nturner, wsuart}@jaguarlandrover.com

ABSTRACT—Distributed virtual simulation is increasingly in demand within the automotive industry. A distributed and networked approach to system level design and simulation stands to benefit from a unifying relational oriented modeling and simulation framework. This will permit innovative use of existing independent simulations for increased concurrency in design and verification and validation. This paper demonstrates an analysis of the vehicle as a complex system through the combination of a relational framework, high level syntax and semantics for representing models and distributed simulation. This promises to provide a rigorous, traceable and agile approach to conceptual vehicle design and analysis.

Key Words: Design, V&V, Cyber-physical Systems, SOA.

1. INTRODUCTION

Original equipment manufacturers for automotive and aerospace vehicles are increasingly taking advantage of modeling and simulation (M&S) to reduce reliance on physical prototypes in the development life-cycle [1]. 'Virtual integration' supports design, simulation, verification and validation between environments; reducing the cost of testing through analyzing virtual solutions. The modern vehicle has become a complex cyber-physical system of systems requiring the integration of complex system and simulation models within its development process.

The ability to conduct a trade-off analysis for potential complex system solutions ideally would be supported by a closed, harmonized and holistic system model for analysis. However, in practice the required models are distributed amongst many pre-existing simulations. A common, formal and reusable framework for structuring design and analysis in such a distributed simulation environment has been lacking.

Individual components of the vehicle, whilst integrated at the physical level, are represented by domain specific simulations often created and governed by independent stakeholders. Therefore, a virtual integration approach must consider the combination of system level behaviors and a distributed systems view of these disparate domain simulations. Understanding the process of vehicle design and verification over a distributed simulation network in a dependable way demands substantial advances in how design models and simulations are modeled compared to the more commonly used approach of tightly integrating simulations on a local environment [2,3].

Our proposed methods are illustrated through an elementary case study. We demonstrate how a relational representation of a vehicle transient drive cycle can be utilized to prepare for
integrated simulation in a distributed network of individual simulators; orchestrated through a service-oriented analysis workflow of integrated simulations.

The remaining structure of this paper is as follows: Section 2 presents an overview of the challenges for complex system M&S. Section 3 describes our proposed M&S framework for complex systems. Section 4 provides a case study to apply our approach to modeling and simulating the effects of driver behavior on vehicle performance. Section 5 outlines the conclusions and future direction of this ongoing work.

2. MODELLING AND SIMULATION FOR COMPLEX SYSTEMS

Design specification in traditional engineering practice uses various methods to specify system elements (components/subsystems). Properties of each element are specified, e.g. by an attribute value and a tolerance on that value. Aggregating these specifications to system level attributes and functions is not always clear in current practice. Relational orientation has been developed to provide a more natural approach to such aggregation and system integration.

In complex systems (and systems of systems), system level analytics typically do not exist; therefore sub-systems are simulated individually. Relational orientation can be especially useful when designing and simulating systems or systems of systems for which there are no reliable and repeatable overarching system analytics.

In order to simulate dynamic system responses, analytics must be executed using their defined mathematical functions in the order in which the system performs its functions. In the simulation process, these (static) analytics must become an executable used for analysis of system response to dynamic change. Therefore in Section 3, time will be introduced along with system architectural elements to include control elements defined and integrated into the system specification.

3. MODELLING AND SIMULATION FRAMEWORK

The M&S framework will be implemented using a Relational-Oriented Systems Engineering and Technology Trade-off Analysis (ROSETTA) framework [4]. This permits translation between mathematical models, analysis of physical systems, and disparate computer simulations. It provides a unified common framework for both design and V&V; filling the gap at the top of the systems engineering V-model and capturing the relationships between system input variables and system objectives or requirements. While similar to the Quality Function Deployment (QFD) House of Quality, ROSETTA replaces expert opinion with mathematical relations. Aerospace and data link applications of ROSETTA are presented in [4] and [5].

3.1 ROSETTA for Complex System of Systems

The central concept is to use available models of the system or its components, e.g. mathematical, simulation or data models to create a static relational structure of design solution space in which the time dependency is not exposed. If a system level model is not available or achievable then lower level models can be used to create the pairwise sensitivities between the attributes of the operating environment and those of the system.

Figure 1 shows an abstract view of a ROSETTA framework. After first identifying the input variables and objectives of the stated problem the static relational structure can defined. The Q matrix is defined first, capturing the relationships between the input variables and the objectives. These could be sensitivities (partial derivatives) of transfer or response surface functions. If there is no coupling between input variables or objective variables, then the transformation matrix Q alone provides the static relational structure. These transformation relationships are sufficient for design and dynamic simulation.
Any coupling between the objective variables is stored in the \( M \) matrix and coupling of the system variables are stored in \( N \). The collective matrices \( M, N \) and \( Q \) together define the static framework.

In the general problem, the partial derivatives at a given point in the design solution space, or estimates of their values can be used to populate the Jacobian matrices of the transformational matrix and of the system matrices. When properly combined using the chain rule, the resulting total differentials give system level directions of improvement for the design variables. These are used in ROSETTA in place of a system level model or analytic when none is available.

### 3.2 Using ROSETTA for System Simulation

When no system level model or analytic is available due to the complexity of the system or system of systems of interest, the process of developing a relational oriented framework for a simulation workflow is shown in Figure 1. To illustrate this, a static framework will be developed in Section 4.2 and extended in Section 4.3 to a dynamic structure by appending time as a parametric to the system matrix in the modeling and analysis case study.

The result will then be a partial differential equation for the total derivatives of the objective variables with respect to time, in which the stable relations are captured in the matrix structure of the framework.

Thus, the key for provision of a unified common framework for both design and V&V is to create a ROSETTA framework of the (static) relational structure of design solution space to which time differentials can be appended for dynamic simulation of candidate solutions. This will be a subject of the case study in Section 4.

### 3.3 Integration of Distributed Simulation into the Modelling and Simulation Framework

As described earlier in the paper, in order to apply the M&S framework to a production engineering environment, it is not possible to assume that the high fidelity domain simulations and subsystem level analytics are contained within a closed execution environment. In practice, these systems will be (physically) distributed across an organization and often developed in independent stovepipes [6]. Integration of these simulations with the M&S framework requires not just network enablement, but also the harmonization of heterogeneous interface specifications and modeling assumptions. The development of domain specific simulations has proven

![Figure 1. The architecture of the system simulation process](image-url)
successful in the automotive sector, however, the networked and distributed integration of these
domain simulations still remains a challenge.

Current state-of-the-art technologies, such as High Level Architecture (HLA) and Data Distribution Service (DDS) [7,8], for integrating heterogeneous distributed simulations do not support the provision of Quality of Service (QoS) so as to guarantee timely and dependable service delivery. Our previous work considers the limited approaches to achieving this through redundancy and proposes new methods for dynamically modelling QoS in service oriented environments [9,10].

3.3.1 VIDAE Architecture

The Virtual Integration Design and Analysis Environment supports cyber-physical engineering through the agile combination of simulation services (including hardware-in-the-loop components) and the application of system analysis methods based upon our M&S framework. This is in response to the need to provide a method for integrating distributed analysis components without requiring a deep understanding of the internal details of each simulation model. VIDAE consists of two parts: the simulation workflow and the DIVIDER [11]; see Figure 2.

3.3.2 Abstract Vehicle Model

In order to provide a domain model for the VIDAE to configure the VIDAE a distributed abstract service model is proposed. This provides a generalized model of the vehicle, most importantly the interfaces between its main subsystems, which can be used as the basis of distributed simulation system and simulation method integration activities. The model is initially constructed in UML (Figure 3) and supports the generation of machine processable notations that the VIDAE can consume and reason over. In this model the logical data flows between components represent a specification for service orchestration whereas the interface assemblies represent the 'physical' data flows through the system.
3.3.3 Simulation Workflow

The simulation workflow consumes the abstract vehicle model as a set of services and allows an engineer to construct a workflow from a subset of these in order to conduct early analysis and testing of vehicle and sub-system designs. By using service orientation the abstraction of each individual service is transparent to the engineer allowing them to focus on achieving the most accurate results as fast as possible. Based upon semantic models of the services, the VIDAE is able to check the compatibility of connected services in addition to alerting the engineer whenever manual intervention is required to resolve mismatch between service inputs and outputs.

Once a workflow is configured (see Figure 4), including appropriate analysis services along with design constraints, the engineer is able to observe the results and sensitivities of the designed vehicle system. The system is comprised of two major subsystems:

• The System Analysis Framework analyses each individual service within the workflow. Firstly it is verified whether the services are compatible with each other given various domain models and QoS constraints. The System Design Framework is utilized to identify the parameter sensitivities (using ROSETTA). These results are then integrated to provide a workflow analysis.

• The System Design Framework seeks to find the most optimum design by minimizing the sensitivity of the entire workflow. The results from running simulations through DIVIDER are integrated and analyzed using the methods described in this paper.

3.3.4 Distributed Virtual Integrated Development

To support the distributed Analysis Workflow we developed the Distributed Virtual Integrated Development Environment (DIVIDER) [11]. This provides a Service-Oriented environment for integrating domain simulations dependably and in real-time. This is underpinned by a powerful workflow technology that adapts to changes in the execution environments to satisfy the QoS requirement for the workflow. In cases where delays are encountered, simulation response time is prioritized over fidelity and lower fidelity simulations can be utilized.
SOA abstraction of the capability of domain simulations from their implementations permits the agile combination of services by the engineer and the rapid interchange between implementations based on a semantic model of their functionality. DIVIDER breaks the concept of a workflow engine into three logical components with different phases of execution: offline, deployment, and online. The concept of a publisher is also introduced to automate the process of service publication. Finally the data schema provides the mechanism by which incompatible services can potentially be integrated.

4. CASE STUDY: SIMULATING EFFECTS OF DRIVER BEHAVIOUR

The automotive domain provides a good example of a system domain that has high fidelity subsystem level analytics, simulations or test data models but no reliable and repeatable overarching system level analytic or simulation. There is no single analytic to support important design trades such as optimizing system design for a key performance parameter, e.g. fuel consumption constrained by regulatory requirements on emissions and CO2. The aim of this section is to demonstrate early research results of how ROSETTA and a Service Oriented Virtual Environment can be used to meet this challenge. In brief, we aim to show how to use high level analytics for (rapid) conceptual modeling with pointers to replacement of analytics with service calls. Much of the power train and vehicle behaviors, however, can only be accurately represented by direct physical measurement. In a Service Oriented Virtual Environment, a more accurate simulation can then be achieved by making service calls to higher fidelity models.

4.1 Analysis in a Driving Course Transient Cycle Test

Governments and agencies have specified extensive tests using drive cycles to assess whether vehicle emissions and CO2 satisfy regulatory requirements [12]. Driving cycles are generally defined in terms of vehicle speed and gear selection as a function of time. Speed profiles consist of \( n \) data rows of time in seconds \( t_i \) (1 < \( i \) < \( n \)) and speed \( v_i \) in km/h (1 < \( i \) < \( n \)). The drive cycle can be performed in either a full-vehicle test or on a rolling road. Figure 5 provides a stylized sample of an EU drive cycle in graphical form.

The drive cycle in Figure 5 is a section of a transient type. Drive cycles can be broadly divided into ‘steady state’ and ‘transient’ drive cycles.
A Demonstration of a Service Oriented Virtual Environment For complex System Analysis

- A steady state cycle is a sequence of constant engine speed and load modes. These are not the focus of NEDC cycles for light-duty vehicle models.
- A transient cycle is a sequence of constant accelerations, decelerations, and speeds in the vehicle speed and engine load are more or less constantly changing.

Driver behavior will affect the level of emissions. The simplest example is the actual accelerations realized in a real or simulated test. This is illustrated in the drive cycle illustrated in Figure 5.

### 4.2 A ROSETTA Framework for a Driving Course Transient Cycle Test

The goal of this section is to specify the mathematical models that govern the dynamic behavior of fuel consumption and emissions. An elementary ROSETTA framework will be developed to structure these as a model of the objectives, a model of the vehicle, and a transformation model between the two. Time integration through the drive cycle can then be accomplished by making calls to simulations or databases as the vehicle traverses the time-velocity waypoints of the test. The key performance parameters of the vehicle are acceleration \( \gamma_1 \) and speed \( \gamma_2 \), where \( \gamma_1, \gamma_2 = \gamma_1(t) \).

Three objective variables have been identified for the emissions problem case study. The first is fuel consumption which is sought to be minimized subject to constraints on emissions. For the purpose of illustration these will be limited to carbon monoxide (CO) and nitrogen oxide (NOx). Let \( z_2 \) and \( z_3 \) be the mass of CO and NOx emitted during the drive cycle measured in g/km. These must meet regulatory constraints, which are specified in g/km.

The fuel consumed is denoted as \( z_1 \). This is nominally measured in liters/100km; but for the purpose of this analysis \( z_1 \) is specified in kg/km. The fuel economy \( z_0 \) is then the derived quantity \( z_0 = 100\kappa z_1 \) where \( \kappa \) is the conversion between kg and liters of the fuel. Note that \( z_1 = z_1(y_1, y_2, t) \) and consequently, time integration of fuel consumption expressions will in general have complicated dependencies on \( y_1 \) and \( y_2 \). These will be discussed further in the simulation of the dynamics of the objective variables during a drive cycle.

Figure 5. Stylized sample of a vehicle acceleration and cruise event, similar to segment seen in the EU ECE Cycle No. 1, showing the effect of changes in driver behaviour

\[ \text{Note: a change of speed of 20 km/h in 10 sec is an acceleration of} \]
\[ \text{a} = 2 \times 10^{-3} \text{m/s}^2(3.6 \times 10^5 \text{sec}^2) \]
\[ = 0.556 \text{m/s}^2 \]
\[ = 0.056 \text{ g} \]
The Jacobian matrix in the central section of Figure 6 specifies all possible sensitivities of the objective variables to vehicle variables. For specified test conditions, each of these partial derivatives can be assigned numerical values that can be stored in an array.

![Figure 6. ROSETTA framework for simulation](image)

The emissions variables $z_2$ (CO) and $z_3$ (NOx) are the result of imperfect combustion and can be regarded as mass fractions of the amount fuel consumed ($z_1$). The lower left section of Figure 6 depicts the sensitivity of CO and NOx to changes in fuel consumed. These sensitivities are typically derived from large databases collected from bench test measurements of an engine under specified load and other conditions. In the lower left of Figure 6, the $M$ matrix is a reduced Jacobian matrix in which the symmetric partial derivatives (i.e. the partials of $z_1$ by $z_2$ and $z_3$) and the negligible or zero derivatives have been ignored. These two couplings will be the only ones considered in the objectives model.

There is one coupling to consider in the relational structure for the vehicle. This is between the vehicle speed and acceleration; i.e. the time derivative of $y_2(t)$ is $y_1(t)$. Specifically, in the sample drive cycle, the relation $y_2 = (t - 50)y_1$ during the constant acceleration segment (speed is acceleration times time) yields the sensitivity $t$ of $y_2$ to $y_1$.

The coupling of $y_2$ and $y_1$ exposes an explicit dependency of these vehicle variables to time. This permits augmenting the structure with time, as indicated by appending the $2 \times 2$ matrix with an exterior row and column for time. As such, the new $3 \times 3$ matrix is not intended to represent three vehicles variables that may have coupling but rather two that are defined parametrically by time, i.e. $y_1 = y_1(t)$ and $y_2 = y_2(t)$.

The time derivatives of $y_1(t)$ and of $y_2(t)$ are discontinuous as modeled in Figure 5. For the analysis in the case study, acceleration is constant and non-zero over the open region $50 < t < 60$; but zero over $t < 50$, and $t > 60$. Strictly speaking time derivative of $y_2(t)$ will contribute a jump function (i.e. jump in $y_1(t)$) and the time derivative of $y_1(t)$ will contribute a Dirac delta function at each transient point. A similar argument applies to transients in the objective functions.

Figure 6 displays the resulting ROSETTA framework that can be used for simulation of the drive cycle test. A traditional simulation would be based on only the transformation matrix for a time stepped simulation over the course of a drive cycle based on the time differentials of the
objective variables. ROSETTA, on the other hand, exposes the coupling in both the objective and vehicle models. This now makes clear how to express the time differentials in terms of the partial differentials. Furthermore, time has been properly factored out of the representation to make explicit the time dependencies distinct from the structural dependencies of the models.

4.3 Simulation Equations from the ROSETTA Framework

The collective equations ($i = 1, 2, 3$) for the simulation of the dynamics of the objectives during a drive cycle are given by:

$$\frac{dz_i}{dt} = \frac{\partial z_i}{\partial z_1} \frac{dy_1}{dt} + \frac{\partial z_i}{\partial y_2} \frac{dy_2}{dt}$$

(1)

It is important to understand that the appending of time to the system matrix does not introduce time ($t$) as a third variable in the system model. Instead, $t$ is the parameter through which the system variables are defined dynamically. The objectives $z_{1-3}$ in equation (1) for the drive cycle expressed separately read:

$$\frac{dz_1}{dt} = \frac{\partial z_1}{\partial y_1} \frac{dy_1}{dt} + \frac{\partial z_1}{\partial y_2} \frac{dy_2}{dt}$$

(2)

$$\frac{dz_{2,3}}{dt} = \frac{dz_1}{dt} \frac{\partial z_{2,3}}{\partial z_1}$$

(3)

Recall that when $i = 1$, $z_i$ is the fuel consumed. The partials with respect to $z_1$ are just factors of 1 and drop out of the equation. For emissions $z_2$ and $z_3$, that expressed in terms of the fuel consumption, the equations further picks up an additional factor ($\frac{\partial z_2}{\partial z_1}$) that accounts for the mass fraction of fuel that is converted to an emission. Other than this factor, simulating emissions is the same as fuel consumption. Each factor in equations (2) and (3) can be computed by independent simulations (e.g. a driving profile, mileage model and emission simulation).

These equations support dynamic simulation through direct integration. For the case of a drive cycle with acceleration, the right hand side of equation (2) captures the rate of change in the fuel consumption of changing from one cruise state to another as well as the change in fuel consumption during the acceleration. For the case of a cruise cycle, the acceleration $y_1$ is zero and the whole right hand side vanishes. The fuel consumption then remains constant over the cycle. In section 4.5, to illustrate how each of the partial derivatives amount to a service call in the VIDAE, we will demonstrate the effect of different driving behaviors in more detail based on the integration of equation (2) with different driving profiles.

4.4 Specification of Analysis Workflow

The purpose of simulation and analysis in the emissions case study is to provide objective evidence for the evaluation of system level behavior and performance in relation to the intended design performance. The equations of the previous subsection are not system level analytics where design solutions are given by the assignments of values to the variables. In fact, due to their differential form, these equations are suited for local rather than global analysis of the design solution space. Nonetheless, the equations can be used for simulation of system level performance in the neighborhood of specific design solutions.

The distinction between the workflow based on ROSETTA and customary discrete event simulation is that the coupling of variables both in the objective model and in the system model can be accounted for when the system simulation is distributed across a number of independent simulations. The verification of the workflow and application to conceptual analysis using
response surfaces permits replacement of the differential operations in the cells of the ROSETTA framework with purely algebraic expressions that admit numerical calculation. The numerical values in the cells of the framework will depend on the state of the system to the extent that there is coupling. In the case of linear responses the partial derivatives in the transformation matrix are simply the coefficients of the linear expressions and these do not change with system state.

4.5 Making Fuel Consumption Service Calls

The implementation of the case study will be concerned with the provision of the computational workflow to a distrusted service oriented simulation environment (VIDAE). The ROSETTA framework in Figure 6 facilitates organizing a series of five calculations that result in the fuel consumption over the drive cycle (Figure 5), and its sensitivity to driver behavior (different accelerations). These steps can be used by VIDAE to define service calls with higher fidelity simulations.

Firstly, let us look at how fuel consumption changes at different part of the drive cycle based on a direct integration of equation (2):

$$\Delta z_1 = z_1(t') - z_1(t) = \int_t^{t'} \left[ \frac{\partial z_1}{\partial y_1} \frac{dy_1}{dt} + \frac{\partial z_1}{\partial y_2} \frac{dy_2}{dt} \right] dt$$

(4)

where \(t\) is the time when the vehicle starts to accelerate, i.e. \(t = 50\) s, and \(t'\) is when the vehicle enters a new cruise phase.

The integration can be further simplified to

$$\Delta z_1 = \int \frac{\partial z_1}{\partial y_1} dy_1 + \int \frac{\partial z_1}{\partial y_2} dy_2$$

(5)

since the infinitesimal \(dt\) cancels out. The first partial derivative represents the sensitivity of fuel consumption to acceleration; while the second partial derivative represents the sensitivity of fuel consumption to speed. The assignments of numerical values to the partial derivatives then become service calls to analytic models or to simulations.

The first service call would be to an analytical model for the sensitivity of the fuel consumption on acceleration, which is given by:

$$\frac{\partial z_1}{\partial y_1} = \frac{m}{\eta_e \eta_t E_d}$$

(6)

where \(m\) is the mass of the vehicle, \(\eta_e\) is the engine efficiency, \(\eta_t\) is the transmission efficiency, and \(E_d\) is the energy density of fuel (J/kg). This dependence can be understood physically through Newtonian mechanics. According to Newton’s 2nd Law, i.e. \(F = my_1\), the energy of the fuel consumed is translated into the work done in moving the vehicle, i.e.

$$W = \int F \cdot ds = \int my_1 ds$$

(7)

Then, the fuel consumption during the acceleration portion can be obtained by

$$\Delta z_{1,a} = \frac{\int \frac{my_1}{\eta_e \eta_t E_d} ds}{\int y_2 dt} = \frac{my_1}{\eta_e \eta_t E_d}$$

(8)

where \(s_a = \int ds = \int y_2 dt\) is the total displacement during the accelerating portion. The simple constant sensitivity as in equation above indicates that acceleration will always require additional fuel consumption. It is worth noticing that once the vehicle reaches the desired cruise speed, the instantaneous acceleration jumps to zero.
The linear dependence of fuel consumption in acceleration is determinable with the assignment of a value to \( y_1 \). For the simulation of a drive cycle, this can be achieved through a call to a driver behavior model; this would become the second service call. The reference acceleration in the drive cycle in Figure 5 is \( y_1 = 0.0556g \). Over and under accelerations of 10\%, i.e. an acceleration of 20km/h in 9 seconds and 11 seconds, will correspond to 0.0629\( g \) and 0.0515\( g \) respectively. The calculated values of \( \Delta z_{1,a} \) for different driving behaviours (values of \( y_1 \)) are displayed in Table I.

**Table I. Changes in the fuel consumption due to the acceleration portion of the drive cycle.**

<table>
<thead>
<tr>
<th>( y_1 )</th>
<th>( \Delta z_{1,a} )</th>
<th>( \Delta z_{1,c} )</th>
<th>( \langle \Delta z_{1,a} \rangle )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0556( g )</td>
<td>0.0585</td>
<td>−0.0767</td>
<td>0.0201</td>
</tr>
<tr>
<td>0.0629( g )</td>
<td>0.0650</td>
<td>−0.0767</td>
<td>0.0266</td>
</tr>
<tr>
<td>0.0515( g )</td>
<td>0.0532</td>
<td>−0.0767</td>
<td>0.0148</td>
</tr>
</tbody>
</table>

The third service call would then be the assignment of a value of the sensitivity of the fuel consumed (\( z_1 \)) to the vehicle speed (\( y_2 \)). This might be from a call to a high level analytic. The fidelity does not demand knowledge of the amount of fuel consumed; rather only its sensitivity to speed. For example, assuming a constant energy usage by vehicle accessories such as air conditioning, a constant rolling resistance, and no hill climbing, the first order sensitivity of fuel consumption to speed under constant acceleration is driven by the engine friction and the drag. The sensitivity of fuel consumption to vehicle speed is understood in the sense of a steady cruise and not to be confused with the additional fuel consumption attributed to acceleration.

For modern vehicles, the proportion of the energy required to overcome the engine friction (not to be confused with engine efficiency) can be approximated by a \( y_2^{-1} \) dependence [13,14]. The energy required to overcome the drag on the vehicle is proportional to \( y_2^2 \). Combining the two factors, the sensitivity of fuel consumption to speed is depicted in equation 9.

\[
\frac{\partial z_1}{\partial y_2} = -k\langle N \rangle \frac{V}{y_2^2} + \frac{\rho C_D A y_2}{\eta_e \eta_t E_d}
\]

(9)

where \( k \) is the fuel energy per revolution per engine displacement needed to overcome engine friction at zero power output, \( \langle N \rangle \) is the average engine speed over a drive cycle, \( V \) is the engine displacement; \( \rho \) is the air density, \( C_D \) is the drag coefficient, and \( A \) is the frontal area.

The two terms in equation 9 are competing. In general, at small vehicle speeds, the engine friction term dominates, yielding an overall negative sensitivity. Hence, accelerating would lead to a better fuel economy. At large vehicle speeds, the negative contribution to the sensitivity expression saturates to zero, hence, \( \frac{\partial z_1}{\partial y_2} \) is dominated by the linearly increasing drag term. The fuel consumption therefore increases.

In the case study, with publically known values of the constants [15], an integration of equation (9) over the accelerating time interval allows the determination of the change in the fuel consumption. The integration result is depicted in Figure 7.

Figure 7 approximated fuel consumption as a function of cruising speeds (not including the constant fuel consumption due to rolling resistance).
Despite the fact that different driving behaviors lead to different accelerating times, \( \int \frac{\partial z_1}{\partial y_2} dy_2 \) produces same results for all three acceleration patterns, as the initial speed and end speed are identical, i.e. \( y_2 = 10 \text{km/h} \) to \( y_2 = 30 \text{km/h} \). The change in the fuel consumption from one cruise speed to another is captured in Table I by the integrated value \( \Delta z_{1,c} \).

Note that in Table I, \( \Delta z_{1,a} \) accounts for the change in fuel consumption solely due to acceleration, \( \Delta z_{1,c} \) accounts for the change in fuel consumption from cruising at 10km/h to 30km/h. Finally, a call is required to calculate the average change in the fuel consumption, \( \langle \Delta z_{1,a} \rangle \), during the acceleration portion. This fourth service call would be a calculation that reads,

\[
\langle \Delta z_{1,a} \rangle = \Delta z_{1,a} + \frac{1}{2} \Delta z_{1,c}
\]

where the second term takes into account the averaged instantaneous cruising fuel consumption during the acceleration phase.

The last service call is to calculate the combined fuel consumption for the drive cycle. Using a fuel consumption of 0.12kg/km initially at 10km/h as read from Figure 7, and together with the change in the fuel consumptions calculated in Table I, it is then possible to obtain the overall averaged fuel consumption of the drive for cycle for each of the driving profiles. The results are illustrated in Table II. Note the total distance travelled during the drive cycle is given by \( s = s_a + s_c \), where the distances for the acceleration and cruise portion will depend on the acceleration.

### Table II. Combined fuel consumption for the drive cycle

<table>
<thead>
<tr>
<th>( y_1 )</th>
<th>( s_a z_{1,a} )</th>
<th>( s_c z_{1,c} )</th>
<th>( \langle z_1 \rangle = (s_a z_{1,a} + s_c z_{1,c})/s )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0556g</td>
<td>0.0078</td>
<td>0.0108</td>
<td>0.0609</td>
</tr>
<tr>
<td>0.0629g</td>
<td>0.0081</td>
<td>0.0112</td>
<td>0.0616</td>
</tr>
<tr>
<td>0.0515g</td>
<td>0.0075</td>
<td>0.0105</td>
<td>0.0604</td>
</tr>
</tbody>
</table>

The result then is that for the drive cycle under consideration and the fuel consumption sensitivity in Figure 7, the under acceleration (0.0515g) exhibits a moderate reduction in the fuel consumption. This conceptual analysis also specifies the order of computation and service calls that could be made to higher fidelity modelling using the VIDAE. These analysis results also
provide a benchmark for validation of results from the VIDAE made through distributed service oriented calls to the higher fidelity simulations.

4.6 Emissions Design Trade-off Analysis

For the three objectives modelled using ROSETTA, the design trade-off is to optimize fuel consumption under the regulatory constraint on the emissions. Equation (3) shows the coupling of fuel consumption to emissions. The solution need not be global over the entire range of the operation of the vehicle, because local solutions can be realized over smaller ranges such as the range of the speeds in the drive cycle that were used in Figure 5. These local solutions will depend on design parameters under the control of the systems engineer.

One such parameter is the airflow into the engine. In general, the emissions objectives can exhibit the same direction of improvement or they can compete with each other. The CO and NOx emissions are competing objectives with respect to airflow rate.

The emissions calculations in equation (3) typically are based on a large data model of engine performance. The complexity of the combustion process requires direct measurement from a test bed. These tests are at discrete system states (e.g. engine load and RPM) based on a design of experiments.

The first partial derivative in equation (3) has already been calculated in the previous section. Only the mass fraction of fuel converted to an emission needs to be calculated based on the data model of measurements from the engine test bed. Figure 8 depicts the normalized mass fractions, i.e. emissions to fuel consumptions ratios, as functions of normalized airflow rate while fixing other design parameters such as the amount of fuel injected.

The normalized mass fraction of the emissions \( \frac{z_2}{z_1} \) can be read directly from the graphs in Figure 8. With a normalization constant, these mass fractions can be translated to the partial derivatives \( \frac{\partial z_2}{\partial z_1} \) in equation (3). The red dots in both graphs indicate the zero normalized engine airflow rates (i.e. minimum).

Knowing the two partials on the right hand side of equation (3), the integral calculation of the emissions, \( z_{2,3} \) over a drive cycle amounts to the multiplication of the fuel consumption and the selected mass fraction translated into a steady state value of \( \frac{\partial z_{2,3}}{\partial z_1} \) through the normalization
constant. However, we will now show the detailed calculation for the purpose of a constrained trade-off analysis.

A regulatory constraint on either emission corresponds to a horizontal line in the graph. In the case of CO this implies a minimum acceptable airflow rate. For example, if normalized constraint is at the midpoint of the vertical axis (0.65g/kg), the normalized airflow rate would need to be greater than 0.185 in order to satisfy the constraint.

For NOx, on the other hand, again using the midpoint (0.45g/kg), the maximum acceptable normalized airflow rate would be 0.465. The design choices on normalized engine airflow rate in this example are then restricted to the range of 0.185-0.465.

The actual selection of the rate in this example would choose the best fuel consumption for airflow rate within this restricted range of the design space. In reality, there would be other constraints to consider as well, such as soot emissions, unburnt hydrocarbons and engine noise. The actual selection of the rate would choose the best fuel consumption that respects all constraints.

In summary, as noted in the beginning of this section, even this elementary drive cycle design trade-off does not admit an overarching system level analytical model. However, modelling the problem using the ROSETTA framework facilitated organizing a series of calculations that resulted in the fuel consumption over the drive cycle, the resulted emission under consideration, and the range of feasible solutions for one of the engine design parameters for fuel optimization constrained by emission regulations. This conceptual analysis further provided the analysis workflow for making service calls to a distributed simulation environment such as the VIDAE.

5. CONCLUSION AND FUTURE CHALLENGES

In this paper we have illustrated how a ROSETTA framework can be utilized to provide analysis of vehicle emissions and performance as it performs a drive cycle. ROSETTA provides a rigorous, traceable framework to structure a workflow for a distributed simulation environment. ROSETTA is seen to provide a framework that extends the system structure model to dynamic simulation in a way that accounts for coupling and provides a verifiable analysis workflow that can be used for orchestration of services. We demonstrated how our methods could be used by means of an automotive case study. A trade-off is structure around a sample drive cycle to minimize fuel consumption and meet regulatory emissions requirements.

A major challenge with service-oriented simulation that we are currently addressing is dealing with the changes in execution environments when providing a real-time integrated simulation capability. This will become more significant when hardware-in-the-loop systems are integrated into the virtual simulation workflow, with a good example of this being a driver in the loop (DIL) simulation, requiring a real-time response.

ACKNOWLEDGEMENTS

This work is sponsored by the Programme for Simulation Innovation (PSi) [16]. A partnership between Jaguar Land Rover and UK EPSRC grant EP/K014226/1.

REFERENCES


ABOUT THE AUTHORS

**Professor Charles Dickerson** is Chair of Systems Engineering at Loughborough University and a Principal Investigator of complex vehicle system analysis in the Programme for Simulation Innovation. His research and design experience includes MIT Lincoln Laboratory, the Lockheed Skunkworks and Northrop Advanced Systems. He is an active member of the IEEE and the Assistant Director for Knowledge at INCOSE.

**Dr Siyuan Ji** received his PhD in Physics from the University of Nottingham in 2015. He recently joined the Programme for Simulation Innovation research group led by Prof Dickerson at Loughborough University as a research associate. He is currently working on mathematically-based engineering for the vehicle as a complex system.

**Dr Stephen Clement** received a PhD in Computer Science from the University of York in 2015. His primary research is in distributed optimization and systems engineering as part of the Programme for Simulation Innovation, a joint sponsored research project by EPSRC and Jaguar Land Rover.

**Dr David Webster** is a Research Fellow working in the School of Computing at the University of Leeds. He has published over 25 peer-reviewed papers in the fields of distributed systems since 2004. David's primary research involvement is currently within the joint EPSRC and Jaguar Land Rover sponsored Programme for Simulation Innovation, focusing on supporting complex vehicle system analysis through distributed service-oriented simulations.

**David McKee** is a PhD student in the School of Computing, University of Leeds, UK. He received the MEng degree in Computer Systems and Software Engineering from the University of York, UK. Currently he is working in the areas of real world time service orientation and large-scale distributed system simulation.
**Jie Xu** is Chair of Computing at the University of Leeds, Head of the Institute for Computational and Systems Science, and Director of the UK EPSRC WRG e-Science Centre. He has industrial experience in building large-scale networked systems and has worked in the field of dependable distributed computing for over 30 years. He is a Steering/Executive Committee member of IEEE SRDS, ISORC, HASE, SOSE, etc. and a co-founder of the IEEE Conference on Cloud Engineering (IC2E). He has led or co-led many research projects to the value of over $30M, and published over 300 research papers.

**David Battersby** is the Technical Specialist responsible for the Systems Engineering of Jaguar Land Rover’s (JLR’s) software platform to meet the ever increasing need for advanced software features. He is the industrial theme leader for analysis of the vehicle as a complex system within the joint JLR, EPSRC funded Programme for Simulation Innovation (PSi). His career has included Systems and Software Engineering within both automotive and defence industries in addition to involvement in a number of high profile industrial/academic research collaborations.

**Neil Bevan** has 30 years experience in the automotive industry within both the supply base and Original Equipment Manufacturers. He is currently the Technical Specialist for Diesel Aftertreatment Systems Integration within Jaguar Land Rover. In this role, he has managed the introduction of Diesel Particulate Filter and NOx Selective Catalyst Reduction technologies into the JLR product range. Neil is a Fellow of the Institute of Mechanical Engineering.

**Niall Turner** graduated in 2005 from Loughborough University with an MEng in Automotive Engineering, and has since spent ten years working with engines in industry. Niall has worked in several areas, including combustion and performance development, fuel economy optimization, engine calibration, cooling system performance testing and engine durability testing, for engine applications ranging from stationary power generation to the automotive industry. Trained in 6 Sigma methods, Niall is currently involved with the simulation of engine performance.

**Will Suart** is the model based design, verification and validation group leader at Jaguar Land Rover. In this role he is accountable for the development and verification methods of mathematical models for controls systems, simulation for component sizing and functional simulation architecture development. His research includes modelling standards and novel simulation methods such as numerical methods, simulation synchronization and coupled model formalisms. He is currently the industrial technical director for the PSi program.