DOI: 10.13009/EUCASS2019-597

8TH EUROPEAN CONFERENCE FOR AERONAUTICS AND AEROSPACE SCIENCES (EUCASS)

DOI: ADD DOINUMBER HERE

CARMEN, The Liquid Propulsion Rocket Engine Simulation Platform, Development Status and Perspectives

Marco Galeotta⋆†*, Florent Ventimiglia*⋆*, Gaelle Usseglio*⋆⋆*, Robin Vermes*⋆⋆
⋆*CNES Direction des Lanceurs*
52 Rue Jacques Hillairet, 75612 Paris, France
⋆⋆*Thales-service*
290 Allée du Lac, 31670 Labège, France
marco.galeotta@cnes.fr · florent.ventimiglia@cnes.fr · gaelle.usseglio@cnes.fr · robin.vermes@cnes.fr
†Corresponding author

Abstract

CNES efforts to develop an in-house tool for the system simulation of liquid propellant rocket engines led to the creation of CARINS in 2002. Since then a great number of improvements and user customizations have been applied to the first versions of the code. This highlighted the need for the harmonization on a common integrated platform with other tools used for the design/simulation of the LPRE. This impulse towards a complete redevelopment of the numerical core of CARINS has been further supported by the increasing complexity of the integration of some of the components used (e.g. MAXIMA, FORTRAN77, SCILAB, JAVA). The strategy implemented since 2016 is to proceed with a gradual improvement of the tool and a harmonization of the programming languages used in order to simplify the maintenance of the code. Python has been chosen for the new model processing and visualization platform. The ODE (Ordinary Differential Equation) solvers selected belong to the LSODE (Livermore Solver for ODE) family and a dedicated integration strategy was developed (based on the CARINS development) and is implemented using a Fortran to Python interface generator (F2PY). Migration of each elementary component from the standard CARINS library to the CARMEN library is on-going, with a complete revision/improvements of its physical modelling. Finally, a dedicated non-regression analysis is performed in order to ensure the validity of the results as well as the non-degradation of the simulator performance. The following steps will be the introduction of the advanced CARINS tools for parametric studies and the engine domain definition and the integration of the CARDIM (engine cycle definition and sub-components modelling) tool in the CARMEN platform.

1. Context

Around the year 2000, the decision was taken at the CNES to develop an in-house tool for the 0D modelling of liquid-propellant rocket engines (LPRE).[8] This led to the development of the first numerical core which later became the first version of CARINS (CAlcul de Réseaux en régime INStationnaire). First versions of the tool allowed the steady-state and transient modelling of pneumatic-hydraulic networks in a graphical user interface. Basic modelling components allowed the simulation of capacitive effects (cavities) and resistive or inductive/resistive effects (valves, pipes and orifices). Specialized elements were developed in order to model LPRE (Liquid Propellant Rocket Engine) components such as combustion chambers, turbo-pumps and regenerative circuits. Heat exchange and thermal evolution of the components were also taken into account (allowing for non-adiabatic simulations). Finally, a dedicated fluids' database was developed for the firsts versions of CARINS based on the BWR (Benedict-Webb-Rubin) state laws and thermodynamic properties.[10] During the last two decades several improvements have been progressively introduced to the CARINS software both at the modelling level (with the introduction of new components and the refinement of old ones) and at the functionality level. New software with a more pronounced focus on engine design was also developed. New functions such as sensitivity analysis, engine domain studies and engine tuning were introduced into a revised tool - CARSTAT (CAlcul de Réseaux en régime STATionnaire)[3]- retaining the numerical core of CARINS (improved with an automatic detection of the steady-state engine mode). The functions solely related to engine design (engine cycle optimization and sub-components dimensioning) were introduced into a new tool named CARDIM.[6] Later developments during the past 5 years tried to unify these tools, used internally for engine design (CARDIM) and modelling

DOI: 10.13009/EUCASS2019-597

(CARINS) and for engine tuning or domain studies (CARSTAT) into a single tool CARFONC (CAlcul de Réseaux en mode FONCtionnement).

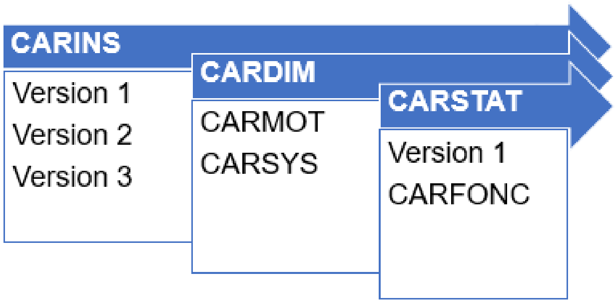


Figure 1: CARINS family software and versions.

However, the increasing complexity related to maintenance of the tools, together with the proliferation of new versions of basic modelling elements or customization of existing ones, lead to a higher risk of model opacity as well as to a decrease in software performance. Furthermore, some major issues with the tools of the CARINS family were highlighted during the years and needed to be treated, among them:

- Use of a multiple programming languages (FORTRAN, C, MAXIMA, JAVA);
- Archaic GUI (Graphical User Interface) functionalities and bugs;
- Clean implementation of external API (Application Programming Interface) and user-defined components.

For these reasons it was decided in 2016 to redevelop the CARINS family of tools.[7] The new software named CARMEN (CAlcul de Réseaux Modulaires ENergétiques) is intended to be an integrated platform where, via a single GUI, the user can perform the LPRE design, modelling and analysis. The roadmap of this development is composed of a new GUI development as well as two numerical core refactoring (for the two main tools: CARFONC and CARDIM). The scheduled time frame for this development is the second half of this decade (2016-2020).

2. The CARMEN Development

Figure 3 shows the roadmap for the CARMEN development (excluding the development of the new GUI). Five key blocks of activities can be identified:

- Phase 0: CARINS numerical core prototyping
- Phase 1: CARINS deployment (first part)
- Phase 2: CARINS deployment (second part)
- Phase 3: CARSTAT deployment
- Phase 4: CARDIM deployment

Phase 0 and 1 have been achieved in 2017 and 2018. Phase 2 is currently on-going and it is scheduled to be achieved by September 2019, leading to a first deployment of the CARINS functionality on the CARMEN platform. For what concerns CARSTAT and CARDIM deployment, the activities are scheduled for the second half of 2019 and 2020 according to the internally allocated budget.

2.1 GUI Development

This activity was considered to be a preparatory phase for the CARMEN development and was completed in 2016 before the decision to engage in the continuation of the project. The current GUI (developed in JAVA) was replaced with a Python based GUI. In parallel to a graphical redesign, the optimization of some basic characteristics as well as bug correction have been performed. Figure 2 shows a prototype of the new CARMEN GUI.

DOI: 10.13009/EUCASS2019-597

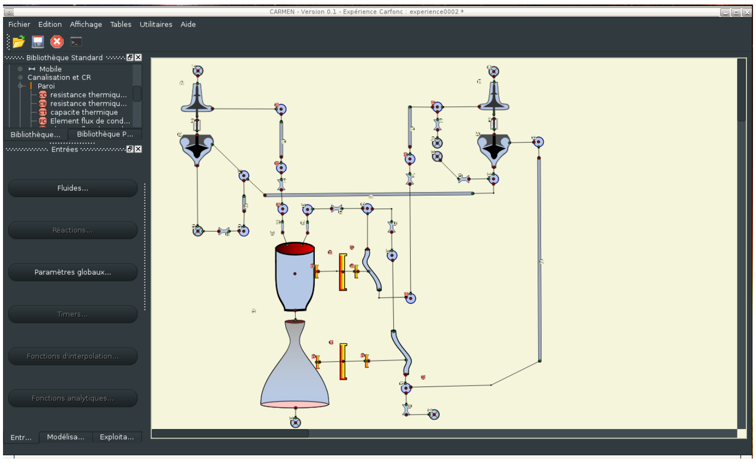


Figure 2: CARMEN GUI

2.2 CARINS numerical code audit and CARMEN prototyping

The first phase of the development was dedicated to the development of an initial CARINS numerical core prototype: GAM (Générateur Automatique de Modèle). Prior to the development of CARMEN, a code audit was carried out to identify the required capabilities for the new software. In addition, the audit allowed for the identification and highlighting of difficulties and limitations in the current GAM. Thus, possible solutions to these problems can be considered in the earliest design stages.

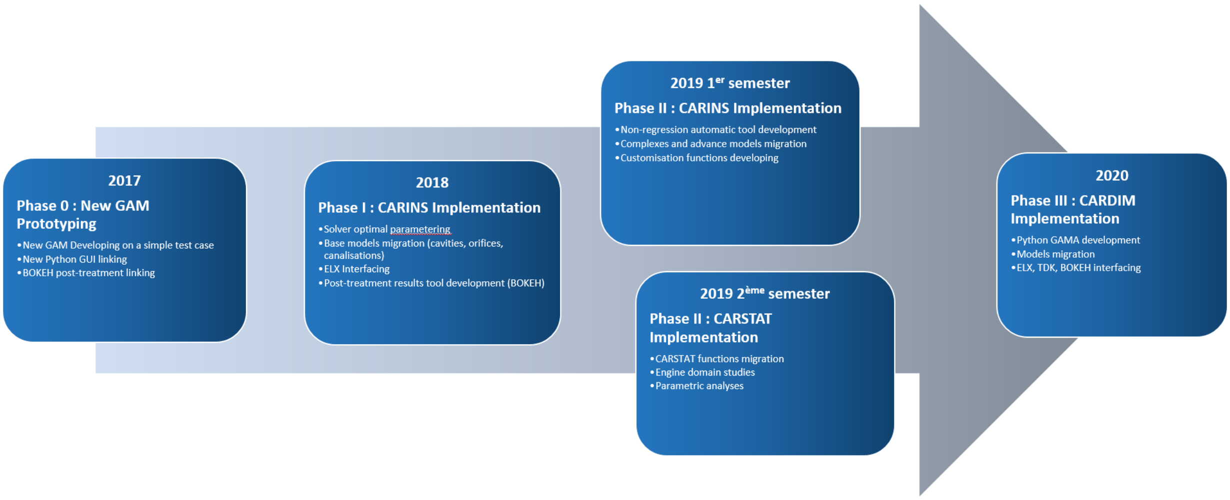


Figure 3: CARMEN Roadmap

Functionalities that are needed are:

- Fast, rapidly converging code;
- Fully customizable software with easy possibilities of adaptation;
- Clarity regarding software characteristics and maintainability;
- Possibility to be used in co-simulation for system studies, optimization and control problems;
- Modern and user-friendly tool.

The main limitations are:

DOI: 10.13009/EUCASS2019-597

- Use of uncommon programming languages (LISP and MAXIMA);
- Progressive difficulty in the maintainability of some languages (Fortran and JAVA) inside the CARINS GAM structure;
- Massive use of non-modular scripts with recursive dependence between main and libraries;
- API for external software non clearly defined and unstable.

Two new configurations were proposed in order to overcome these limitations and to answer to the user needs. Both of them were proposed with a Python architecture but the first with a preliminary dynamic generation of the main code of the simulator (to be run in a second step, this was the current CARINS GAM mode

of operation) and the second without any dynamic code generation. Figure 4 shows the spider diagram with the coverage of the user needs for the 2 configurations. It is important to highlight that in order to guarantee the same performances of the code, the numerical solvers that already showed excellent capabilities in CARINS, were maintained in both configurations (see Section 3.3) Based on these results the configuration without any dynamic code generation, in the following called the "full Python" architecture, was selected.

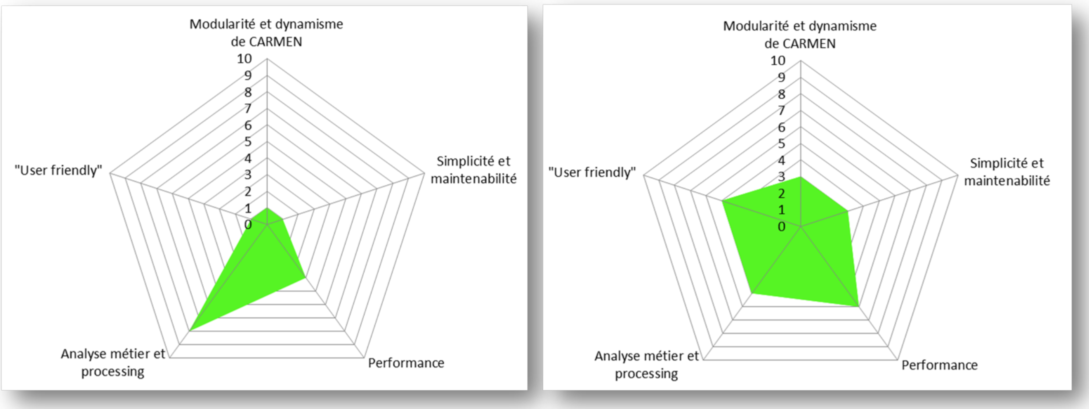


Figure 4: Spider diagram with reference to user needs (right graph: configuration without any dynamic code generation)

2.3 CARINS deployment

The first prototype of the CARINS GAM was developed based on the "full Python" architecture. The main objective was to test a concrete application case in the new modelling environment, as well as the performance of the new GAM. Only a few elementary components were developed in this frame (cavities, orifices and pipes for gaseous flows). The numerical core was linked with the FORTRAN solver (LSODE ODEpack) via the SCIPY package that does not allow for a complete customization of the solver parameters. The results of two simple simulations showed a very good agreement with the CARINS reference data but a different behavior in numerical noise attenuation. Furthermore, for the same cases, a large loss of computational performance was observed. The reasons for these problems can be clearly identified as a non-optimal parametrization of the solver and the strategy of its call. During the first development phase of CARMEN a big effort was devoted to the correction of these issues. First of all, a dedicated interface between the Fortran LSODE solvers and the Python GAM was developed (with use of the F2PY pack) in order to access the complete palette of the solver parameters. Secondly the former integration strategy of CARINS was taken up and redeveloped in a new GAM (see Section 3.3). The rest of the CARINS deployment on the CARMEN platform consisted essentially of the development of the elementary component models (see Section 3.4), the implementation of the fluids thermodynamic tables (see Section 3.5) and others functions (such as timers, function definitions, etc.), and the linking with a tool for the simulation results analysis (see Section 3.6). These 2 firsts phases of development have been almost completed. The on-going activities are mainly devoted to the validation of the elementary components models (see Section 4).

DOI: 10.13009/EUCASS2019-597

2.4 CARSTAT and CARDIM deployment

The next phase (phase 3) consists of the implementation of the CARSTAT functionalities in the CARMEN platform. As this tool shares the numerical core with CARINS, there is no need for the development of a new numerical core. On the contrary the new functionalities of CARSTAT (essentially the engine domain study, sensitivity analysis and engine tuning) will require a complete redevelopment since they are currently coded inside the SCILAB environment (via a shell call to the compiled executable simulator generated by the GAM). This activity is, currently, scheduled for the second half of 2019 or beginning of 2020. The rest of 2020 will be devoted to the deployment of the CARDIM tools in CARMEN. CARDIM functions differently from CARINS and CARSTAT as it computes only the thermodynamics engine cycle without any time evolution of the engine parameters. Its numerical code, GAMA (Générateur Automatique du Modèle Algébrique) solves only a non-linear algebraic system of equations (and not an ODE system). Its transition to a "full Python" architecture will be the objective of the activities until the end of 2020.

3. CARMEN Numerical Code Development

The strategy of the CARMEN numerical core has been largely inspired by the current CARINS one. The main difference is that the CARINS GAM foresees a preliminary phase of dynamic code generation while in CARMEN a Python script allow the direct construction of the ODE system and its integration via a call to the solvers libraries. It is worth to notice that the solvers called by the Python scripts are the same LSODE family solvers used in CARINS (coded in Fortran95) via the F2PY tool. The use of compiled code for the numerical integration of the equations allows for a limitation of the computational performances loss linked to the utilization of a non-compiled language.

3.1 From GUI to Configuration File

The communication between the former GUI in JAVA and the Fortran GAM in CARINS was achieved through a Maxima formatted file. The commonality of the language used for the development of the new GUI and GAM (Python) suggested to suppress the Maxima formatted file creation step as well as the overall use of the Maxima language. A set of python description objects, with simple attributes (lists and dictionaries) seemed more adapted and easier to interface with the GAM. Furthermore, a JSON formatted file is created, storing these simple description objects at the start of the simulation. This configuration file further allows the user to manually re-run the experience (including without the GUI) as well as permitting an easy verification of simulation setting.

3.2 Construction of the ODE System

The new CARMEN GAM is based on a vast library of Python objects. Every object describes an elementary component (cavity, orifice, pipe, etc.) which contains all the equations needed for its modelling (generally a differential and algebraic equations system). Numerical values of all parameters required for the components' description are present as well. Finally, it also contains all the links assuring its connection with the rest of the simulation and the sharing of its variables. At the start of the simulation, each object provides its equations and their own dependencies (which variables are needed from the others objects). The equations are then ordered according to those dependencies and absence of recursive definitions is verified. At the end the obtained ODE system will always be explicit and causal. In case of non-adherence to one of these 2 conditions, the simulation stops with a dedicated warning.

3.3 Numerical Integration Strategy

The obtained ODEs system is integrated by a numerical solver. Due to the nature of the equation systems generated by this kind of problem (possibility of high degree of stiffness and sparse Jacobian matrix) the choice of a linear multistep method has been made. Solvers of the LSODE family are well adapted to our needs.[5] The user has the choice between 2 different solvers of this family: the LSODA solver (well suited for stiff and non-stiff problems with dense Jacobian matrix and with an automatic choice between Adam Moulton and BDF formulas) and the LSODES solver (particularly efficient for stiff problems with a sparse Jacobian matrix). The strategy of calling the solvers is showed in Figure 5. It takes into account event detection and triggering. The solver is re-initialized every time an event is triggered (if its triggering time is known) or detected. In the latter case a research of the instant of detection time via a dichotomy method is done by the GAM (within a tolerance imposed by the user). Reasons for event detections vary from flow-rate inversion, negative pressure, density detection, or others user-defined conditions. At each time step (automatically calculated by the solver based on the relative error imposed by the user) algebraic and differential

DOI: 10.13009/EUCASS2019-597

variables are calculated. Differential variables are then passed to the solver that integrates the differential variables according to the chosen method.

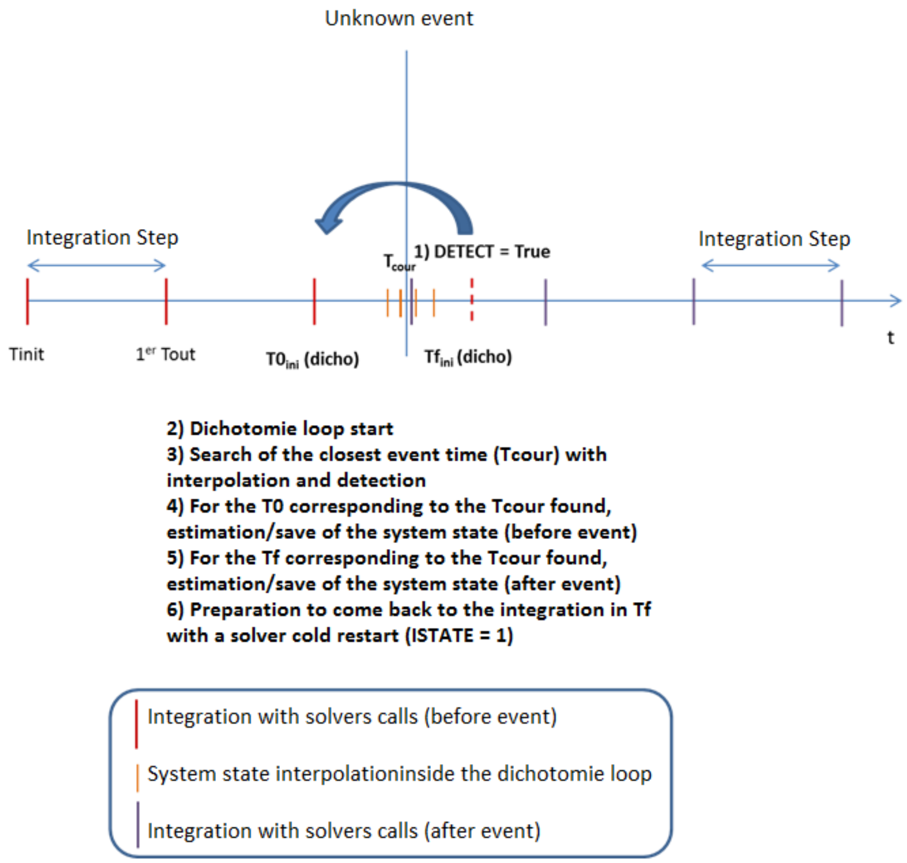


Figure 5: Integration strategy with event detection.

3.4 Elementary Components Modelling

A part of the strategy of the CARMEN development was to verify the numerical core accuracy and performance on simple experiences composed of few elementary components. New elements and functions are progressively introduced and validated with a non-regression verification method. The elementary components that have been, for the moment, introduced in the CARMEN GAM are listed in the Table 1.

3.5 Thermodynamic tables import

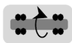
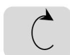
In the former versions of CARINS, the thermo-dynamic properties and fluid laws were part of the GAM with dedicated libraries and functions for a certain number of fluids. BWR state laws were chosen for their high degree of accuracy with reference to experimental data and their stability in the regions close to the fluids saturation curves.[10] Nevertheless, in the last CARINS versions as well as in the CARMEN platform, it was decided to replace the BWR libraries of the GAM with a series of direct call functions to the ELX API. ELX is the CNES internal tool for the calculation of the fluids thermodynamic parameters. It is also based on BWR laws but updated with the most recent values of polynomial coefficients. The decision to use ELX allows the use of an up-to-date database of fluids (more fluids than contained in the CARINS GAM libraries) and simplifies the code maintenance.

3.6 Data results post-analysis

Results of the run of a simulation are recorded in an output file. This file is easily readable but, very often, too large in size to be handle in an efficient way. Furthermore, the user needs are in most cases limited to a quick visualization

DOI: 10.13009/EUCASS2019-597

Table 1: List of elementary components.

Elementary Component	Symbol	Media
Cavity		Perfect Ideal Gas Ideal Liquids Real Fluids
Orifices		Perfect Ideal Gas Ideal Liquids
Pipe		Perfect Ideal Gas Ideal Liquids
Valve		Perfect Ideal Gas Ideal Liquids
Pump		Ideal Liquids
Turbine		Perfect Ideal Gas Real Gas
Shaft		With and without gear
Thermal Capacity		
Thermal Flux		Conductive Convective Radiative
Fluid Boundary Conditions		
Thermal Boundary Conditions		
Mechanical Boundary Conditions		

of specific simulation variables. For these reasons a sophisticated data results post-treatment tool has not been judged essential in this phase of development. The BOKEH API seems to respond well to these needs, as it allows an easy way to visualize time evolutions series and basic functions associated with quick analysis (zooms, super-imposition, off-set...)[2] A specific panel gives control over the graphs that can be created (see Figure 6). Up to two different variable types can be displayed on the same graph. The created graphs are displayed in an html page, which further allows for the export of the results in image format.

4. Non Regression Verifications

The new CARMEN GAM is being validated by means of a comparison of identical simulations carried out in both CARMEN and CARINS. The quantitative evaluation of the degree of similarity of two simulations is based on the

DOI: 10.13009/EUCASS2019-597

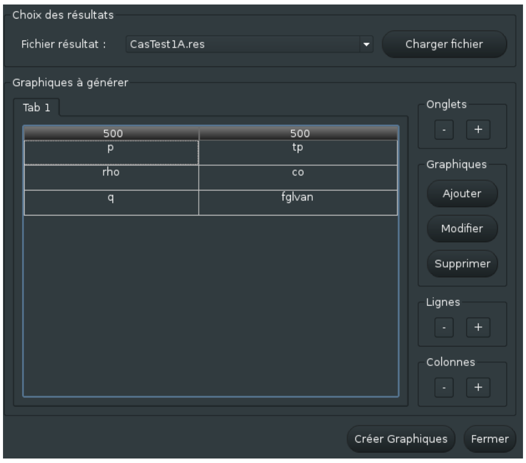


Figure 6: Graphs customization panel.

comparison of the temporal evolutions of all the state parameters of an experience (we referred to state parameters as all the differential variables that compose the ODE system). Since the sampling of the time-series may not be identical, an interpolation on a common time series is mandatory in order to permit the evaluation of the differences between the 2 curves. The common time series is composed of the union of the time series of the 2 simulations. With this method we are sure to “capture” all the densification of time step on the 2 experiences. As for the choice of the norm to use, several possibilities are available from literature, ranging from standard Euclidean norms to integral norms with or without a complexity factor (useful when we compare periodical signals[9]). Other aspects must be taken into account such as the length of the simulation (that requires always to have the same duration of time series), the number of points of the series (normally overcome by the use of integral norms) and the differences in the amplitudes between the parameters (the use of normalized parameters permits a better comparison). Five norms were retained for the present comparison all of them defined on normalized variables, shown in Table 2:

Table 2: Retained norms for comparison.

Distance	Formula
“City Blocks” Distance (CBD)	$\sum_{i=1}^{N} \lvert x_i - y_i \rvert$
Euclidian Distance (ED)	$\sqrt{\sum_{i=1}^{N} (x_i - y_i)^2}$
Infinite Norm (Ninf)	$max_{i=1...N} \lvert x_i - y_i \rvert$
Complex Invariant Distance (CID)	$\frac{max\,(CE(x), CE(y))}{min\,(CE(x), CE(y))} EDE(x, y)$
Dissymmetric Distance (Dissim)	$\int_0^T \lvert x(t) - y(t) \rvert dt$

The first three norms are the standard Lp norm of order 0, 1 and 2. The fourth is the Complex Invariant Distance (CID)[1] while the last, is the Dissymmetric distance (Dissim).

CE(x) is a complexity estimation function and it is simply defined as:

DOI: 10.13009/EUCASS2019-597

$$CE(x) = \sqrt{\sum_{i=1}^{N-1} (x_i - x_{i+1})^2} \quad (1)$$

The CID norm multiplies the Euclidian norm by the complexity factor that is the ratio between the maximum and the minimum complexity of a time series, aiming to enlarge the differences between the Euclidian norms of 2 functions with very different CE. The Dissim norm is just the difference between the integrals of the two-time series (always taken with a positive contribution).

4.1 Automatic script development

To verify effectively the non-regression after an update, an automatic process was developed. The script converts a CARINS experiment into a CARMEN one. This process is iterative and there is a need to update the utility every time a model is added to CARMEN. The converted experience is then launched by CARMEN and sent to the GAM for simulation. A result file is generated. The results are compared with the CARINS experience results; warnings can be raised when the metric conditions are not fulfilled and BOKEH html files are created to allow the user to manually verify the result for each variable.

4.2 Test case definition

Test cases have been defined based on a step by step verification logic, introducing one new element at a time. Every test case is run for incompressible liquid, ideal gas (calorically perfect or not) and real fluid cases. The validation process is on-going. For the moment no major discrepancies have been identified (Turbine and thermal elements are still to be tested).

The first collection of tests includes cavities and orifices (including valves of several type, isolation, control, non-return, safety ...). The second set includes pipes (non- capacitive pipe). The third and fourth sets consists of more complex experiences with cavities, orifices and pipes as well as mono and multi-species simulations. The fifth test group is devoted to the validation of pumps, the sixth to the validation of turbines and the seventh to the validation of turbo-pumps (pumps, turbines and shafts). Finally, the last set of tests (eighth) consists of the validation of thermal elements (capacities and thermal flux estimators). The repartition of the number of test cases for each set is shown in Figure 7 (the grey area represents tests cases still to be run). Less than 5 % of the test cases exhibited significant over-passing of norm thresholds indicating a major issue in the CARMEN simulation. All the others warnings raised by the automatic tool concern small differences in numerical noise and or peaks generated by the interpolation process due to very small time off-sets in the two series (outliers).

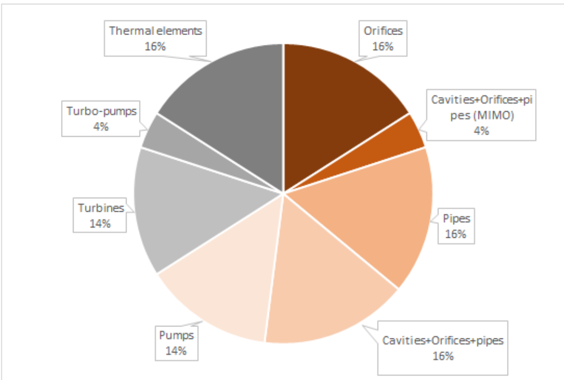


Figure 7: Test case repartition by subject

4.3 Main results

Non-regression analysis is in a well advanced phase of development and allows for the identification of potential problems linked to:

- Errors in the modelling equations;

DOI: 10.13009/EUCASS2019-597

- Recursive or non-causal variable determination;
- Bad-definition of variable saturations;
- ODE system built-up phase.

Norm thresholds have been defined sufficiently low in order to be assured of the coherence between the two time-series in the case of the absence of warning messages. A warning of outdated thresholds from the automatic script requires a "manual" comparison of the distance vector of each variable of the test case under analysis. In figure 8, as an example, a test case with several warnings on exceeded norm thresholds will be shown (pump test case with LH2 media and a transient pump behavior).

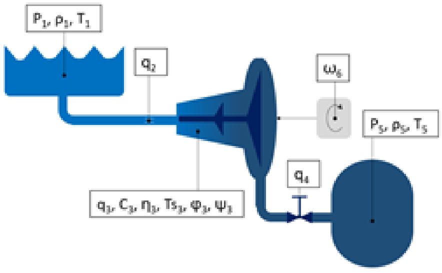


Figure 8: Pump test case.

In this test case the pump is started at 1.4 s and it changes its functional regime from 7000 rad s^{-1} to 8500 rad s^{-1} at 7 s, the evolution of pump characteristics as well as the pressure and temperature in the downstream cavity are observed. As it can be seen in Figure 9, the absolute maximum error in pressure and temperature on the downstream cavity is quite low, less than 4 Pa for pressure (not considering the initial spike due to different saturation conditions on the pump downstream temperature) and less than 10×10^{-6} K for temperature (that is less than 5×10^{-4} % and 5×10^{-6} % respectively). On the contrary on the pump parameters, flow-rate, Δp and torque, the maximum absolute error is significantly larger: around 5×10^{-3} % for the flow-rate and 10×10^{-2} % for Δp and torque (see Figure 8) however localized to only 2 time-frames where the pump is changing its functioning regime. The reason for theses discrepancies can most likely be attributed to a difference in the calculation of the hyperbolic tangent function used to simulated the pump startup (see Figure 11). The comparison of the different norms for all the state variables of the simulation is reported on Figures 12 and 13. It is interesting to note the larger values of calculated norms for temperature and pressure in the downstream cavity compared to the pump variables (when the manual analysis of the curves suggests the opposite). This is due to the fact that the error in the pump parameters is localized to pump transient phases.

With this type of approach, it has been possible to run a large number of test cases and to easily identify any possible deviation in the simulations results. In case of detection of a discrepancy, a detailed and deeper analysis has always been performed and allowed us to fix any potential issue in the model under analysis or in the general functioning of the new numerical core.

5. Perspectives

The current state of the activities for the development of the CARMEN platform has been presented in the previous paragraphs. It is important to underline that the excellent obtained results are the consequence of a big effort sustained by the CNES during the last 2 decades to improve and refine the CARINS family of simulation tools. Without these solid foundations on which the platform relies it would have been impossible to "migrate" the numerical core to the new platform in such a short timeframe. Nevertheless, a large part of the activity remains, in particular:

- The completion of the migration of all the CARINS models and their complete validation
- The implementation of the engine domain and sensitivity analysis tools (CARSTAT)
- The refactoring of the CARDIM numerical core.

Finally, the vast catalog of the CARINS models available at CNES has to be migrated to the new platform, CARMEN. The development of an automatic tool able to convert the old CARINS simulators to the new CARMEN ones has to be foreseen.

DOI: 10.13009/EUCASS2019-597

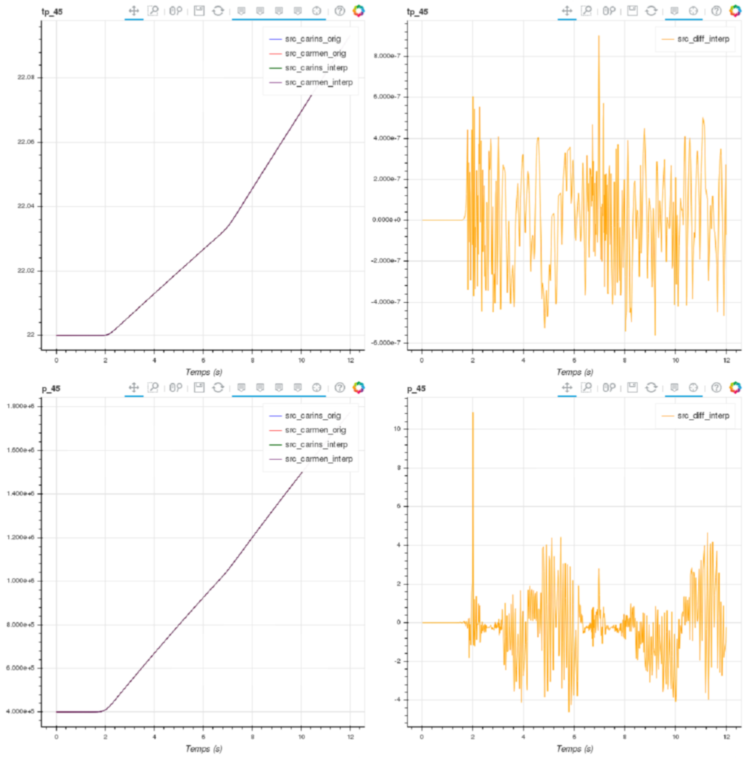


Figure 9: Comparison of pump downstream P and T

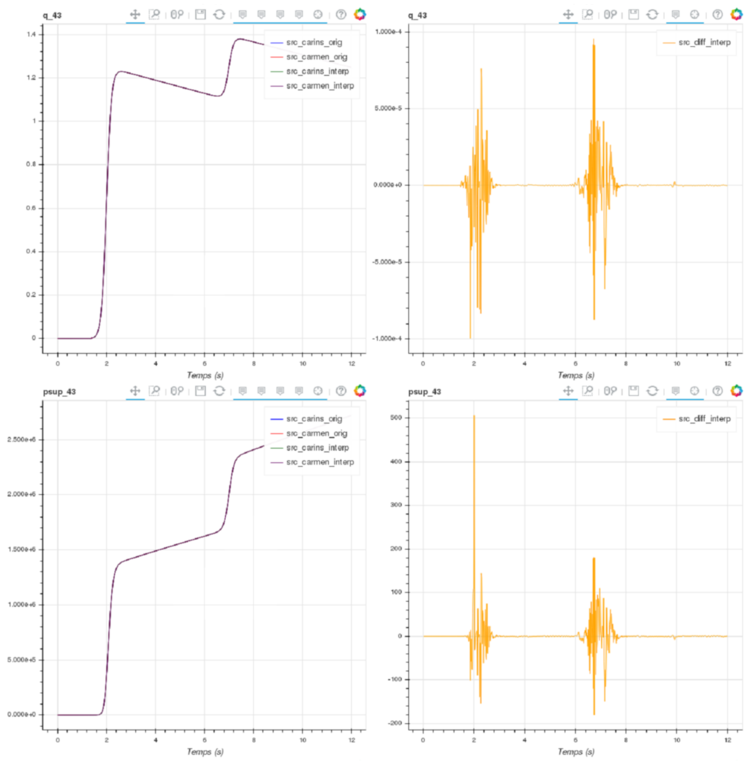


Figure 10: Comparison of pump torque and Δp

DOI: 10.13009/EUCASS2019-597

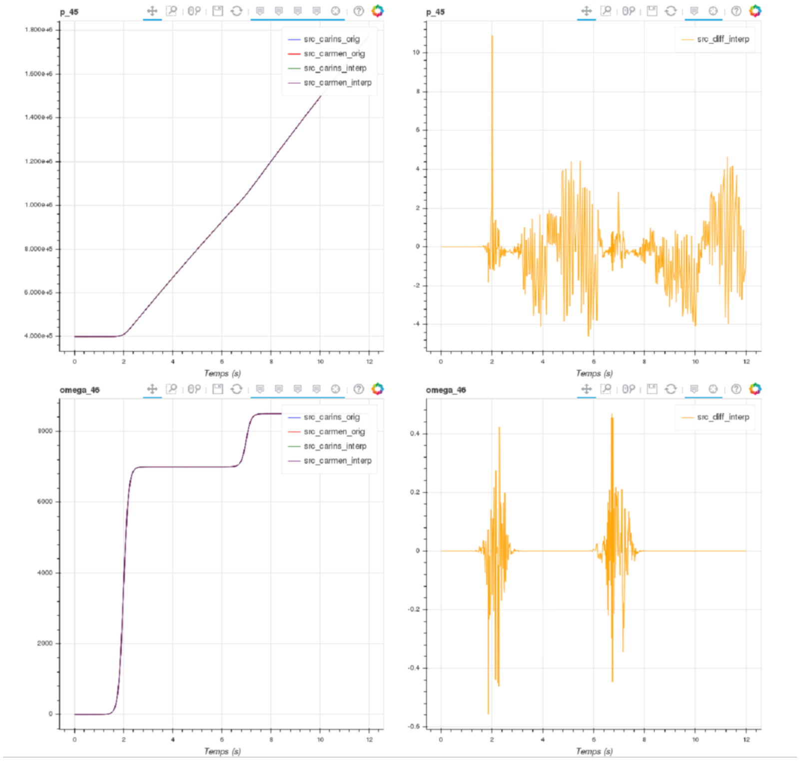


Figure 11: Comparison of pump rotational speed.

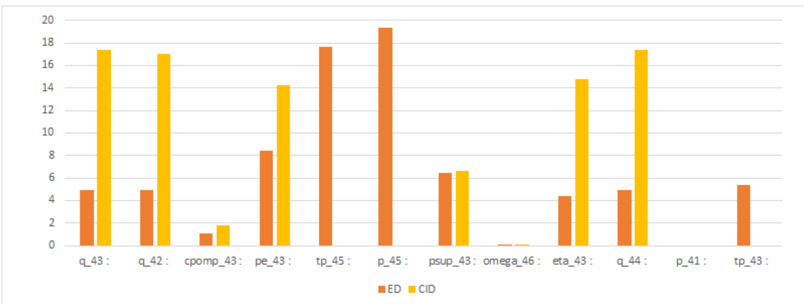


Figure 12: ED and CID norms for experience variables

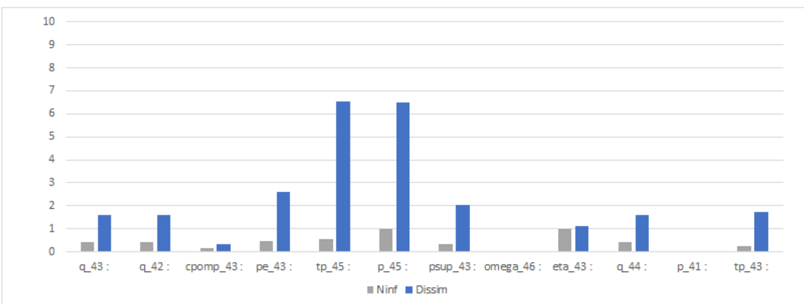


Figure 13: NInf and Dissim norms for experience variables

DOI: 10.13009/EUCASS2019-597

References

[1] Gustavo EAPA Batista, Xiaoyue Wang, and Eamonn J Keogh. A complexity-invariant distance measure for time series. In *Proceedings of the 2011 SIAM international conference on data mining*, pages 699–710. SIAM, 2011.

[2] Bokeh Development Team. *Bokeh: Python library for interactive visualization*, 2018.

[3] E Cliquet, A Iannetti, and J Masse. Carmen, liquid propulsion systems simulation platform. *Progress in Propulsion Physics*, 2:695–706, 2011.

[4] Elisa Cliquet, Gerard Albano, and Gerard Ordonneau. Cnes launchers directorate capabilities in rocket engines transients simulation. ISTS, 2008.

[5] Alan C Hindmarsh and Krishnan Radhakrishnan. Description and use of lsode, the livermore solver for ordinary differential equations. *Lawrence Livermore National Laboratory Report UCRL-ID-113855*, 1993.

[6] Alessandra Iannetti, Gerard Albano, and John Masse. Cardim rocket engine design tool. In *44th AIAA/ASME/SAE/ASEE Joint Propulsion Conference & Exhibit*, page 5116, 2008.

[7] T Jues and Rouzaud S. Carmen, liquid propulsion systems simulation platform, 2016.

[8] V Leudiere, G Albano, G Ordonneau, J Masse, and B Legrand. Carins: A versatile and flexible tool for engine transient prediction¡ development status. In *24th Symposium (International) on Technologies for Space. Tokyo, Japan*, 2004.

[9] Usue Mori, Alexander Mendiburu, and Jose A Lozano. Distance measures for time series in r: The tsdist package. *R journal*, 8(2):451–459, 2016.

[10] W.C. Reynolds. *Thermodynamic Properties in SI: Graphs, Tables, and Computational Equations for Forty Substances*. Department of Mechanical Engineering, Stanford University, 1979.