LAUNCH VEHICLE MULTIBODY DYNAMICS MODELING FRAMEWORK FOR PRELIMINARY DESIGN STUDIES

Paul Acquatella B.

German Aerospace Center (DLR)
Institute of System Dynamics and Control
82234 Wessling, Germany
`paul.acquatella@dlr.de`

ABSTRACT

Launch vehicle dynamics modeling is quite challenging mainly because of the highly interconnected disciplines involved: propulsion, aerodynamics, structures, mechanisms, and GNC among others. Discipline experts perform their respective design often independently and with separate dedicated tools. Consequently, during launcher preliminary design studies, numerous iterations are required in order to keep mission objectives synchronized.

These preliminary design efforts can potentially be reduced by using a multidisciplinary launch vehicle model integrated in one single tool. Because this allows to reduce the number of iterations and the associated costs, a launch vehicle multibody dynamics modeling framework is a key technology to aim for.

Dedicated developments of multidisciplinary modeling tools for launch vehicle multibody dynamics have been presented in the relevant literature. However, none fully profits from an object-oriented, equation-based, and acausal modeling language like MODELICA. As yet, such an approach is still missing. It is therefore the objective of this paper to introduce such an alternative approach employing this modeling framework.

This framework enables object-oriented and physics-based modeling of subsystems and components related to most key analyses of a launcher system. These include among others: launcher configuration, staging and separation dynamics, end-to-end trajectories, performance, controllability and stability. Moreover, all this can be done within a single simulation environment.

The paper gives an overview on the first building blocks leading to an integrated and multidisciplinary tool for launcher preliminary design studies. Particularly, its easiness of implementation is demonstrated along with the benefits of this approach.

***Index Terms*—** launcher systems, multi-domain modeling, multibody dynamics, ascent, performance, trajectory.

Nomenclature

m	instantaneous mass
`c.m.`	instantaneous center of mass
$\mathbf{I}$	inertia dyadic of the system
$\mathbf{r}$	inertial position of `c.m.`
$\mathbf{v}$	inertial velocity of `c.m.`
$\mathbf{a}$	inertial acceleration of `c.m.`
$\boldsymbol{\omega}$	inertial angular velocity of main body (R)
$\boldsymbol{\alpha}$	inertial angular acceleration of main body (R)
P	generic particle of a variable mass system
$\mathcal{B}$	boundary of a variable mass system
$\mathbf{r}_p$	position from `c.m.` to P
$\mathbf{v}_r$	velocity of P relative to the main body (R)
$\mathbf{n}$	unit vector normal to $\mathcal{B}$
ρ	density
S_x	cross-sectional area of nozzle exhaust exit
P_z	atmospheric pressure
h	altitude
v_e	effective exhaust vehicle
I_{SP}	specific impulse
g_0	standard acceleration due to gravity
q	dynamic pressure
M	Mach number
$\mathbf{v}_{rel}$	relative speed
S_r	reference area
l	characteristic length
α	angle of attack
β	sideslip angle

1. INTRODUCTION

For the several architectures and configurations to consider and optimize at preliminary design studies, several launch vehicle models with varying levels of scope and complexity are necessary.

In that sense, launch vehicle dynamics modeling is quite challenging mainly because of the highly interconnected disciplines involved: propulsion, aerodynamics, structures, mechanisms, and GNC among others. Discipline experts perform their respective design often independently and with separate dedicated tools. Consequently, during launcher preliminary design studies, numerous iterations are required in order to keep mission objectives synchronized.

Preliminary design efforts could potentially be reduced by using a multidisciplinary launch vehicle model integrated in one single tool. Because this allows to reduce the number of iterations and the associated costs, a launch vehicle multibody dynamics modeling framework is a key technology to aim for.

Early efforts on the subject of launch vehicle dynamics modeling were carried out by NASA during the 60's and 70's given the importance to study stage launch vehicle separation [1, 2, 3]. This led to the development of their generalized trajectory simulation, guidance design, and optimization software *Program to Optimize Simulated Trajectories* POST [4], and its more recent follow-up, POST2. For multibody dynamics, TREETOPS [5, 6] was conceived based on Kane's equations, and followed by the more recent CLVTOPS, both featuring capabilities for multiple flexible body dynamic simulation, separation analysis, and liftoff clearance analysis [7].

On the European side, early efforts on multibody dynamics for space applications were also carried out for over 30 years by the European Space Agency (ESA) with their *Dynamic and Control Analysis Package* DCAP [8, 9, 10]. It provides capabilities to model, simulate, and analyze the dynamics and control performances of coupled rigid and flexible structural systems subject to structural and space environmental loads. More recent efforts for developing and consolidating knowledge in launcher dynamics [11, 12], led ESA to develop a launcher multibody dynamics simulator using DCAP as a backbone [13]. This tool has been adapted to meet typical requirements of the ESA Concurrent Design Facility (CDF) environment.

Many other proprietary and commercial tools, like ASTOS developed by Astos Solutions GmbH, are relevant to the launcher modeling and simulation literature, but the extensive list of tools and solutions is not covered here.

Noticing that multidisciplinary modeling is becoming increasingly important for launch vehicle design and simulation, and that none of these previous dedicated developments fully profits from an object-oriented, equation-based, and acausal modeling language like MODELICA; the objective of this paper is to introduce an alternative approach employing this modeling methodology. This approach comes with the first building blocks leading to an integrated and multidisciplinary launcher vehicle dynamics modeling tool.

A brief description of MODELICA as a modeling methodology is given; then an object-oriented and physics-based modeling framework is introduced; followed by a basic mathematical description of a launcher multibody dynamics model; and finally an application example is presented, outlining the key benefits of this approach.

2. MODELING METHODOLOGY

2.1. MODELICA

MODELICA [14, 15, 16, 17, 18] is a modern object-oriented, equation based ***modeling language*** well suited to model complex physical systems containing, e.g., mechanical, electrical, power, hydraulic, thermal, control, or process-oriented subsystems and components.

Models in MODELICA are described using differential, algebraic, and discrete equations which are then mapped into a mathematical description form called hybrid DAE (Differential Algebraic Equations). A DAE system on its implicit form is generally expressed as

$$\mathbf{F}\big(\dot{\mathbf{x}}(t), \mathbf{x}(t), \mathbf{u}(t), \mathbf{y}(t), \boldsymbol{\rho}, t\big) = 0 \qquad (1)$$

where $\dot{\mathbf{x}}$ are the state derivatives, $\mathbf{x}$ the state variables, $\mathbf{u}$ the inputs, $\mathbf{y}$ the algebraic variables, $\boldsymbol{\rho}$ the parameters and constants, t the time variable, and the dimension $dim(\mathbf{F}) = dim(\mathbf{x}) + dim(\mathbf{y})$. Systems are then solved and simulated by MODELICA simulation environments. When these systems are represented in the DAE implicit form, they can be solved directly by a DAE solver such as DASSL. Alternatively, the system can be sorted out according to specific inputs and outputs and mapped into an explicit ODE (Ordinary Differential Equation) form by solving for the derivatives and the algebraic variables, and then subsequently solved numerically by an ODE solver. The process and details of MODELICA's code compilation is out of the scope of this paper.

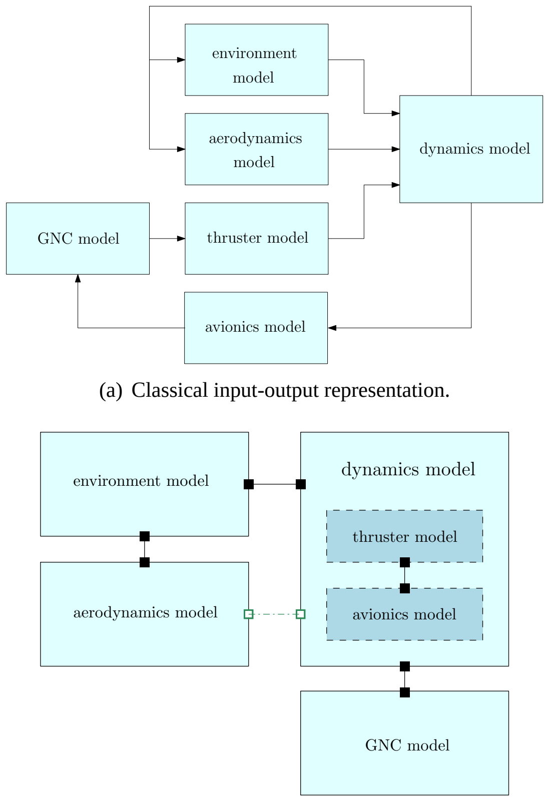
2.2. Main features [18]

In contrast to imperative languages, in which statements and algorithms are assigned in explicit steps, MODELICA is ***declarative***, meaning that declarations are given through equations. These declarations most often describe model's first-principles at their lowest levels without explicit orders or *how* to compute them, hence why MODELICA is said to be ***equation based***. By means of specialized algorithms, these declarative models are translated into efficient computer executable code. This allows ***acausal*** modeling capabilities that give better reuse of classes since equations do not specify a certain data flow direction. This is therefore one of the most important features of the language.

MODELICA is *domain neutral*. In other words, it has ***multidomain*** modeling capability, meaning that model components corresponding to physical objects from several different domains can be described and connected. This interaction between components is defined by means of physical ports, called connectors, and the interconnection is given accordingly to their physical meaning. This meaning is typically represented by flow variables, which describe quantities whose values add up to zero in a node connection (Kirchhoff's first rule); and by non-flow (or potential) variables, which in contrast remain equal (Kirchhoff's second rule).

MODELICA is an ***object-oriented*** language. This helps to model systems and their physical meaning within an object-oriented structure, facilitating the reuse of component models and the evolution of the structure itself. Thus, object-orientation is primarily used as a ***structuring*** concept which exploits the declarative feature of the language, as well as the re-usability of models.

MODELICA has a strong software component model with constructs for creating and connecting components in a ***modular*** fashion. Systems' individual components are defined separately as objects, and their interconnection is given accordingly to their physical meaning. Thus the language is ideally suited as an ***architectural*** description language for complex physical systems.



(a) Classical input-output representation.

(b) Acausal approach, or energy exchange representation.

Fig. 1. Classic approach vs. acausal approach.

3. MODELING FRAMEWORK

A framework for the physical modeling of conventional and non-conventional launch vehicles is presented here. In contrast to the classical signal-based approach, where systems are mainly considered and modeled as signal processors with a fixed causality, this approach employs an acausal approach where systems exchange energy, see Figure 1. In there, the connectors in the acausal approach represent a physical interaction where an energy balance is applied.

3.1. Main features

The framework consists of a structured and *object-oriented architecture* which enable combinations of several sets of system and subsystem models, themselves built and composed into components and interfaces corresponding to different physical domains (mechanical, electrical, structural, control, etc.) and therefore described from their first principles with the MODELICA language.

Refering to Figure 2, given a particular study definition (3-DOF/6-DOF performance, stability and controllability, optimization, etc.) of a preliminary design phase, the first step of the framework is to obtain all necessary data and specific requirements of the study in order to properly generate a particular launch vehicle model. Once the key subsystems and disciplines interacting are properly identified, a multidisciplinary launch vehicle model integrated in one single tool is used to generate study results. For this reason, this tool is quite *versatile*.

In this sense, subsystems of a launch vehicle, as well as the launch vehicle system itself can be modeled within a single simulation environment, and without necessarily implementing coupling interfaces to other specialised tools. This allows the capability of performing end-to-end launch vehicle trajectory simulations as it will be shown in the application example.

To provide application-specific capabilities, the generic functionality of the framework can be tailored and extended by additional user-specific code. For instance, the framework may include databases, pre-processing and post-processing scripts, several MODELICA libraries, interfaces to commercial software like MATLAB&SIMULINK (available for instance in DYMOLA), combination of multibody and

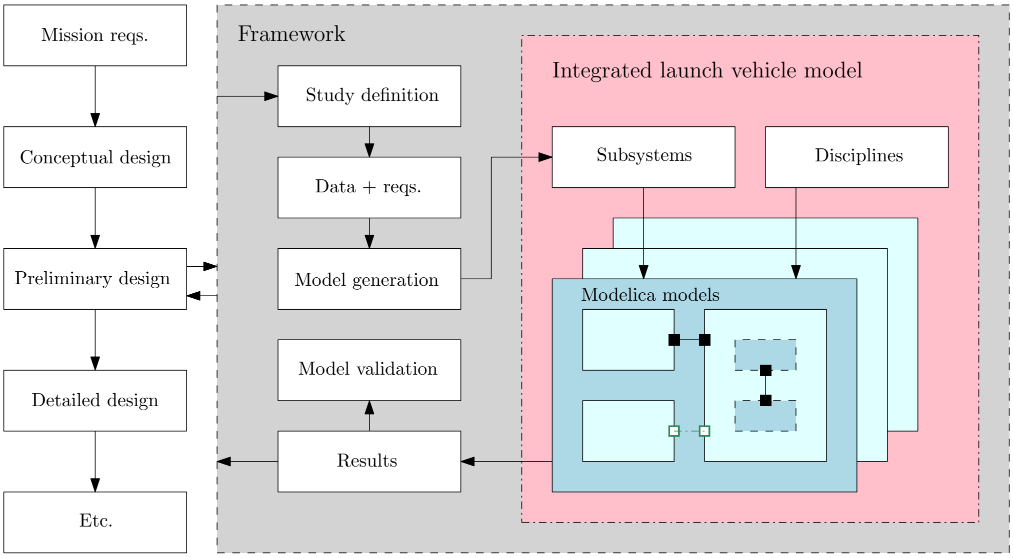


Fig. 2. Overall picture of the framework.

FEM [23], and application programming interfaces (APIs) to other tools.

The framework implementation is based upon the extension of the *DLR Space Systems Library*, introduced in [19], in order to enable object-oriented and physics-based modeling of subsystems and components related to launch vehicle system dynamics.

The main feature of the library is the **World** component. It defines basis coordinate systems such as the Earth Centered Inertial (ECI) and the Earth Centered Earth Fixed (ECEF) coordinate systems, and manages calendar and Julian times. Most notably, it provides capabilities to instantiate multiple gravity models of different kinds of complexity, up to the most precise EGM96 gravity model [26]. Moon and sun perturbation terms to the gravity models are also available. The library also contains state-of-the-art space environment models like the NRLMSISE-00 atmospheric density model [27].

This library builds upon the *Modelica Standard Library* [20], the *Modelica MultiBody Library* [21], the *DLR Flight Dynamics Library* [22], the *DLR Flexible Bodies Library* [23], the *DLR Visualization Library* [24] and the *DLR Optimization Library* [25].

4. MULTIBODY DYNAMICS MODEL

Typically, a multibody system is described by a collection of bodies and their interactions.

The interactions, representing physical coupling of the bodies, can be described as rigid connections between frames (Section 4.1); joints representing motion constraints (Section 4.2), useful for meaningful physical joint models (prismatic joints featuring, e.g., spring-damper actuators); or even special elements describing more complex dynamic behavior like joint motion and separation dynamics (Section 4.3).

Bodies are represented by their physical properties (mass, moments of inertia, etc.) and a collection of frames located at special points of interest (center of mass, joint locations, reference points, etc.). Their translational and rotational dynamics are described depending on the physical nature of the system and their components, for instance, Newton-Euler equations of motion in the case for rigid body models. Here, variable mass systems are described by Kane's equation as obtained by Eke [28] (Section 4.4).

4.1. Frames

Recalling the concept of acausal connectors of Figure 1-(b), a *frame* connector from MODELICA's *Multibody Standard Library* [21] is a coordinate system fixed to a model component with a cut-force and a cut-torque as flow variables, and with a position and an orientation object as non-flow variables. Subsequently, mechanical components can be interconnected together rigidly at this frame.

The dynamics of a frame A is completely described by its *generalized position* $\hat{\mathbf{r}}_A$, *velocity* $\hat{\mathbf{v}}_A$, *acceleration* $\hat{\mathbf{a}}_A$, and *force* $\hat{\mathbf{f}}_A$, respectively

$$\hat{\mathbf{r}}_A = \begin{bmatrix} \mathbf{r}_A \\ \mathbf{R}_A \end{bmatrix}, \qquad \hat{\mathbf{v}}_A = \begin{bmatrix} \mathbf{v}_A \\ \boldsymbol{\omega}_A \end{bmatrix},$$
$$\hat{\mathbf{a}}_A = \begin{bmatrix} \mathbf{a}_A \\ \boldsymbol{\alpha}_A \end{bmatrix}, \qquad \hat{\mathbf{f}}_A = \begin{bmatrix} \mathbf{f}_A \\ \boldsymbol{\tau}_A \end{bmatrix},$$

where $\mathbf{r}_A$, $\mathbf{v}_A$, and $\mathbf{a}_A$ are the absolute position, velocity, and acceleration of the frame A with respect to an inertial frame; $\mathbf{R}_A$, $\boldsymbol{\omega}_A$, and $\boldsymbol{\alpha}_A$ the attitude direction cosine matrix, absolute angular velocity, and angular acceleration of the frame A with respect to an inertial frame; and $\mathbf{f}_A$, $\boldsymbol{\tau}_A$ the resulting forces and torques at frame A [29].

For rigidly interconnected frame connectors, say frames A and B, and as mentioned in the modeling methodology section, the kinematic quantities related to the non-flow variables $\hat{\mathbf{v}}_A$ and $\hat{\mathbf{v}}_B$ are equal to each other, whereas the flow variables, cut-forces and cut-torques $\hat{\mathbf{f}}_A$ and $\hat{\mathbf{f}}_B$ in this case, sum up to zero [29, 21]. This is due to a power P balance constraint considering that no energy is stored:

$$\sum P = \mathbf{0} = \hat{\mathbf{f}}_A^{\mathsf{T}}\hat{\mathbf{v}}_A + \hat{\mathbf{f}}_B^{\mathsf{T}}\hat{\mathbf{v}}_B \tag{2}$$

4.2. Joints

Specific joint interconnections in multibody dynamics are very useful to interconnect mechanical systems featuring a non-rigid and physically-meaningful joint motion.

For that, consider a *generalized joint coordinate* $\mathbf{q}$ allowing certain motions between two frames A and B, and its associated *generalized joint force* $\boldsymbol{\lambda}$. Because of the newly allowed motion, additional relationships between the connected frames are necessary. These are given as functions of $\mathbf{q}$ (and possibly $\dot{\mathbf{q}}$) and in terms of the relative quantities between the frames [29].

The corresponding description between the connected frames A and B can be determined similarly as before from a power balance constraint because no energy is stored in such an ideal joint

$$\sum P_i = \mathbf{0} = \hat{\mathbf{f}}_A^{\mathsf{T}}\hat{\mathbf{v}}_A + \hat{\mathbf{f}}_B^{\mathsf{T}}\hat{\mathbf{v}}_B + \boldsymbol{\lambda}_A^{\mathsf{T}}\dot{\mathbf{q}} \tag{3}$$

In that sense, the dynamics of a the joint is also completely described by its related generalized quantities. Since the elements of $\dot{\mathbf{q}}$ are independent from each other, the last expression leads to a constraint equivalent to d'Alembert's principle, see [29].

4.3. Automatic joint loads computation

For launch vehicle staging and separation dynamics, joint models for both physical connection and separation between bodies are required.

This can be done with MODELICA by automatic joint loads computation [35], which is applied to each of the connected bodies prior to their physical separation and released for their subsequent and independent motion. This is the principle behind the *Constraint Force Equation* (CFE) methodology, developed by NASA for similar kinds of studies [30, 31, 32].

The CFE methodology is a highly intuitive method consisting in the computation of joint loads, namely internal forces and torques, caused by joint constraints; along with their application as external forces and torques on each body independently. In consequence, the CFE joint model simply augments the external loads of the system [31] as shown in Figure 3. The constrained equations of motion of two rigid bodies (A and B) connected by a single joint (point $\overline{A}$ in body A and point $\overline{B}$ in body B) [30, 31] are

$$m_A\,\ddot{\mathbf{r}}_A = \mathbf{f}_A^{ext} + \mathbf{f}_A^{con}, \tag{4a}$$
$$\mathbf{I}_A\,\dot{\boldsymbol{\omega}}_A + \boldsymbol{\omega}_A \times \mathbf{I}_A\,\boldsymbol{\omega}_A = \boldsymbol{\tau}_A^{ext} + \rho_A \mathbf{f}_A^{con} + \boldsymbol{\tau}_A^{con}, \tag{4b}$$

where ρ_A is the position vector from the mass center of A to point $\overline{A}$, the point at which the constraint force is applied. The similar equation applies for body B, giving so far 12 equations out the 24 unknowns. Another set of six equations can be obtained as

$$\mathbf{f}_A^{con} + \mathbf{f}_B^{con} = \mathbf{0} \tag{5a}$$
$$\boldsymbol{\tau}_A^{(con)} + \boldsymbol{\tau}_B^{(con)} + (\mathbf{r}_{\bar{B}} - \mathbf{r}_{\bar{A}}) \times \mathbf{f}_B^{con} = \mathbf{0} \tag{5b}$$

where $\mathbf{r}_{\bar{A}} = \mathbf{r}_A + \rho_A$ and $\mathbf{r}_{\bar{B}} = \mathbf{r}_B + \rho_B$. For relative translation and rotation constraints and $\mathbf{e}$ being unit-vectors of the corresponding (A or B) body-frame, it is required that:

$$(\mathbf{r}_{\bar{A}} - \mathbf{r}_{\bar{B}}) \cdot \mathbf{e}_A = 0 \tag{6a}$$
$$\mathbf{e}_A \cdot \mathbf{e}_B = 0 \tag{6b}$$

To couple Eqs. (6) with the equations of motion, these must be differentiated twice with respect to time so that the resulting relationships involve the unknown linear and angular

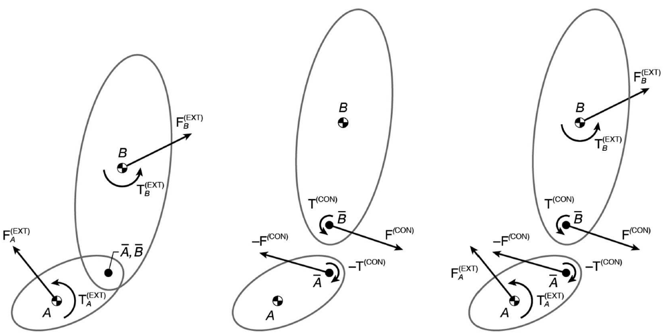


Fig. 3. CFE diagram. Illustration credits: [30].

accelerations. In other words, the six missing equations are given by the following *generalized constraint* equations of the joint, $\ddot{\mathbf{g}} = \mathbf{0}$, where $\mathbf{g}$ represents the non-differentiated constraints in Eqs. (6).

To improve the accuracy of the joint loads solution, which is sensitive to computational error and initial joint misalignment, the generalized constraint equations are augmented with the *Baumgarte stabilization* [33, 34, 30, 31] as:

$$\ddot{\mathbf{g}} + 2\eta\dot{\mathbf{g}} + \eta^2\mathbf{g} = \mathbf{0},\ \eta > 0 \tag{7}$$

As demonstrated in [35], the manual differentiation of Eqs. (6) and their coupling with the equations of motion can be avoided altogether in MODELICA since this is done automatically by the declarative feature of the language.

4.4. Dynamics of variable mass systems

Launch vehicles are systems involving considerable changes in motion as well as in mass (and therefore inertia). The extra loads due to the variable mass effects must be included in the formulation of the dynamic equations of motion.

Consider for instance a solid rocket motor, a system that loses mass while subject to dynamical motion, and which at any given instant of time is a mixture of both a solid rigid part (R) and a fluid part (F) due to products of combustion. These are delimited by the boundary $\mathcal{B}$.

The dynamic equations of motion for these kind of systems as obtained by Eke [28], and established with Kanes's formalism, are summarized here. In [28], it is claimed that these are identical to those obtained by other authors using a Newton-Euler formulation.

The translational equations of motion are given by

$$m\mathbf{a} = \mathbf{f}^C + \mathbf{f}^L + \mathbf{f}^{thr} + \mathbf{f}^{ext} \tag{8}$$

with

$$\begin{aligned}
\mathbf{f}^C &= -2\int_{\mathcal{B}} \rho(\boldsymbol{\omega} \times \mathbf{v}_r)\mathrm{d}V,\\
\mathbf{f}^L &= -\frac{^{(R)}\mathrm{d}}{\mathrm{d}t}\int_{\mathcal{B}} \rho\mathbf{v}_r\mathrm{d}V,\\
\mathbf{f}^{thr} &= -\int_{\mathcal{S}} \rho\mathbf{v}_r(\mathbf{v}_r \cdot \mathbf{n})\mathrm{d}S,
\end{aligned}$$

where $\mathbf{f}^C$ is the Coriolis force, $\mathbf{f}^L$ the system's linear momentum decrease rate relative to the closed surface $\mathcal{B}$, $\mathbf{f}^{thr}$ the thrust vector force, and $\mathbf{f}^{ext}$ the sum of all external forces about the current center of mass of the system, respectively. The left superscript on time derivatives indicates that the derivative is to be taken while the reference frame is kept fixed.

Concerning the thrust vector force, whenever $\mathbf{v}_r \cdot \mathbf{n}$ can be approximated relatively well at the nozzle exit plane, the surface integral can be evaluated in closed form [28]. Using the effective exhaust velocity $v_e = I_{SP}g_0$, a model of the thrust force considering atmospheric losses is given by

$$\mathbf{f}^{thr} = \dot{m}I_{SP}g_0 - S_xP_z(h) \tag{9}$$

The attitude equations of motion are given by

$$\mathbf{I}\,\boldsymbol{\alpha} + \boldsymbol{\omega} \times \mathbf{I}\,\boldsymbol{\omega} + \Big(\frac{^R\mathrm{d}\mathbf{I}}{\mathrm{d}t}\Big)\boldsymbol{\omega} = \boldsymbol{\tau}^{C_1} + \boldsymbol{\tau}^{C_2} + \boldsymbol{\tau}^{H} + \boldsymbol{\tau}^{thr} + \boldsymbol{\tau}^{ext} \tag{10}$$

where

$$\begin{aligned}
\boldsymbol{\tau}^{C_1} &= -\int_B \rho[\mathbf{r}_p \times (\boldsymbol{\omega} \times \mathbf{r}_p)](\mathbf{v}_r \cdot \mathbf{n})\mathrm{d}S\\
\boldsymbol{\tau}^{C_2} &= -\int_B \rho\big[\boldsymbol{\omega} \times (\mathbf{r}_p \times \mathbf{v}_r)\big]\mathrm{d}V\\
\boldsymbol{\tau}^{H} &= -\frac{^R\mathrm{d}}{\mathrm{d}t}\int_B \rho(\mathbf{r}_p \times \mathbf{v}_r)\mathrm{d}V\\
\boldsymbol{\tau}^{thr} &= \int_S \rho(\mathbf{r}_p \times \mathbf{v}_r)(\mathbf{v}_r \cdot \mathbf{n})\mathrm{d}S
\end{aligned}$$

$\boldsymbol{\tau}^{C_1}$ is the so-called jet damping, $\boldsymbol{\tau}^{C_2}$ is due to the Coriolis effect and can be neglected for axisymmetric motion as well as for negligible internal flow, $\boldsymbol{\tau}^{H}$ represents the rate of decrease of the system's angular momentum inside $\mathcal{B}$, $\boldsymbol{\tau}^{thr}$ the moment of the thrust vector about the mass center, and $\boldsymbol{\tau}^{ext}$ the sum of all external moments about the current center of mass of the system.

Notice that if $\mathbf{v}_r$ is zero everywhere, then the Newton-Euler equations of motion for a rigid body are recovered. In general, depending on the nature of the propulsion system and its corresponding shape or assumed burn profiles, these terms can be further simplified and further evaluated in closed form, see [28]. In this way, these loads can be included explicitly in the formulation of the dynamic equations of motion of the corresponding element of the vehicle so that their effect can be included in dynamic analyses.

To conclude the main mathematical formulations, aerodynamic forces and moments can be generally expressed in the body-axis frame as

$$\mathbf{f}^{aero} = -q\ S_rC_i(h, \mathbf{v}, \alpha, \beta, ...), \tag{11}$$

$$\boldsymbol{\tau}^{aero} = q\ S_rlC_j(h, \mathbf{v}, \alpha, \beta, ...), \tag{12}$$

where C_i (for $i = C, Y$, and L) and C_j (for $j = l, m$, and n) are the aerodynamic drag, side force, and lift coefficients, respectively. Finally, the expressions for the dynamic pressure, Mach number, and relative speed are given:

$$\begin{aligned} q &= \tfrac{1}{2}\rho\, \mathbf{v}^2 = \tfrac{1}{2}\gamma P_z(h) M^2, \\ M &= \left|\mathbf{v}_{rel}\right| / \mathbf{v}_s(h), \\ \mathbf{v}_{rel} &= \mathbf{v} - \boldsymbol{\omega}_e \times \mathbf{r} \end{aligned}$$

5. APPLICATION EXAMPLE

An application example for a 3-DOF open-loop point-mass launcher model featuring stage separation dynamics is presented here.

Separation dynamics is simulated with the automatically obtained joint loads satisfying the CFE constraints. The release device is simulated with a linear cutting charge model, and the separation mechanism with the use of retro-thrusters. Properties for this launcher model are taken from the VEGA launcher users's manual as shown in Table 1. Parameters not available were assumed with best guesses.

Table 1. VEGA User's Manual Data (2006)

Property	Stage 1	Stage 2	Stage 3
Length [m]	11.2	8.39	4.12
Diameter [m]	3	1.9	1.9
Gross mass [kg]	95 796	25 751	10 948
Propellant mass [kg]	88 365	23 906	10 115
Thrust (S/L) [kN]	2261	1196	225
Isp (Vac) [s]	280	289	295
Burn time [s]	106.8	71.7	109.6
Ignition time [s]	0	115	195
Separation command [s]	108	188	-

At $t = 106.8\ s$, the first burn is completed and the first stage is separated at $t = 108\ s$. Then after a few seconds, at $t = 112\ s$, giving enough time for clearance aspects, retro-thrusters are actuated to further separate the first stage from the remaining composite. The sequence is similar for the second stage, where the retro-thrusters are commanded at $t = 190\ s$, a few seconds after the second stage separation.

Figure 4 presents the stages' altitude (normalized), relative velocity (normalized), and acceleration during their connected motion as well as during their subsequent separate flight motion.

Results shows that the automatically obtained joint loads satisfying the CFE methodology constraints successfully models the launcher system during its connected flight motion. This demonstrate the capabilities as well as the ease of use and implementation under the proposed framework by taking advantage of MODELICA's modeling methodology.

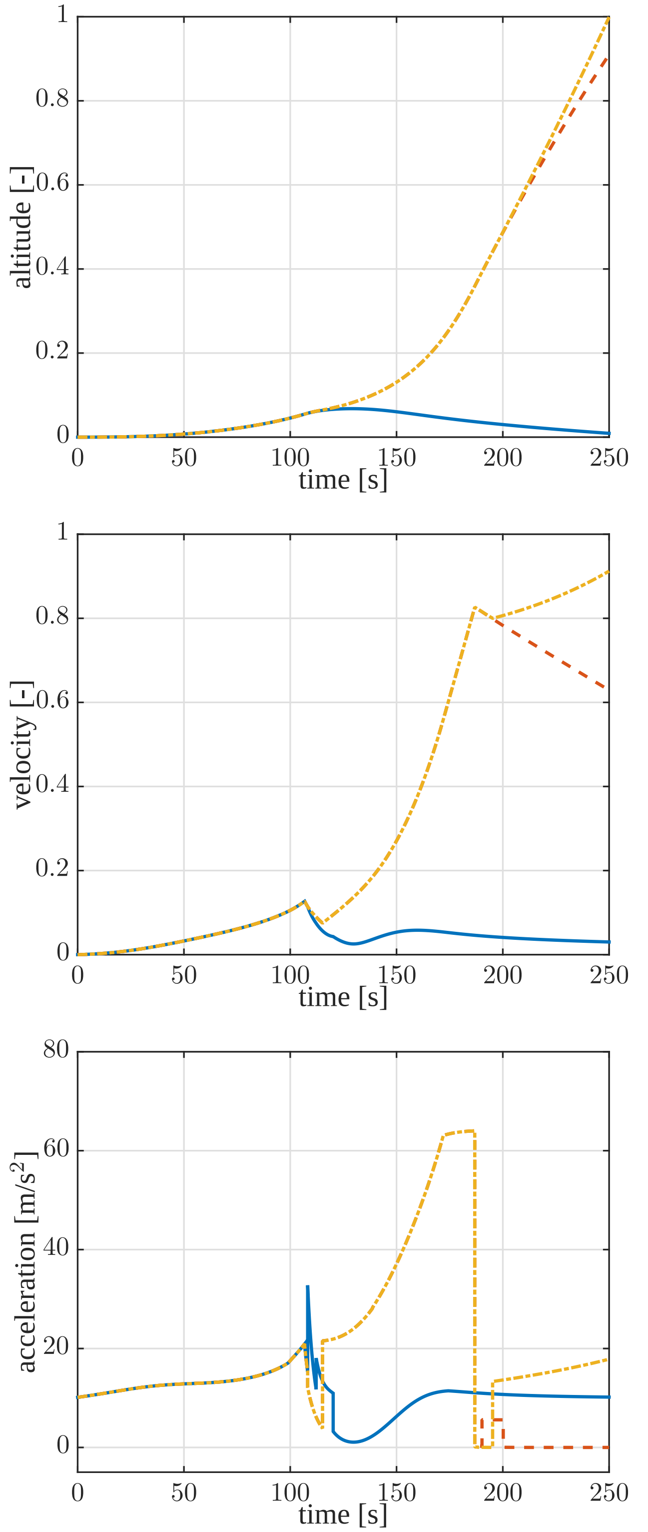


Fig. 4. Application example results. Stage 1 (blue), stage 2 (red), stage 3 (orange).

6. SUMMARY AND OUTLOOK

The objective of this paper was to present an object-oriented and equation-based acausal modeling approach as the first building blocks leading to an integrated and multidisciplinary tool for launcher vehicle dynamics modeling with MODELICA.

Based on MODELICA language as the modeling methodology, we provide a framework which enable object-oriented and physics-based modeling of subsystems and components related to most key analyses of launch vehicle system dynamics. To demonstrate its benefits, a launch vehicle multibody dynamics model is described and implemented within this framework as described with introductory mathematical formulations. Its easiness of implementation is done with an application example.

Future work will be dedicated upon extension of this framework by adding more capabilities, featuring for instance the interconnection of flexible bodies, dedicated algorithms for GNC sizing and design, and most importantly, for optimization studies concerning trajectory, stage sizing, and performance among others.

Moreover, this launch vehicle modeling and simulation framework could in fact support a vast number of use cases across a launcher program life cycle. These may include not only preliminary design phases, but also activities concerning detailed system design, software and component verification and validation, etc.

7. REFERENCES

[1] Decker, J. P. *Experimental Aerodynamics and Analysis of the Stage Separation of Reusable Launch Vehicles.* NASA-SP-148, January 1967.

[2] Decker, J. P., and Gera, J. *An Exploratory Study of Parallel-Stage Separation of Reusable Launch Vehicles.* NASA TN D-4765, October 1968.

[3] Decker, J.P., and Wilhite, A. W. *Technology and Methodology of Separating Two Similar Size Aerospace Vehicles Within the Atmosphere.*AIAA Paper 1975-29, Jan. 1975.

[4] Bauer, G.L., Cornick, D.E., and Stevenson, R. *Capabilities and Applications of the Program to Optimize Simulated Trajectories (POST).* NASA CR-2770, 1977.

[5] *User's Manual for TREETOPS, A Control System Simulation for Structures With a Tree Topology.* NASA Contract NAS-36287, Marshall Space Flight Center, April 1990.

[6] Yates, R. M. *TREETOPS Structural Dynamics Controls Symulation System Updgrade.* NASA Contract NAS8-40194, Marshall Space Flight Center, August 1996.

[7] Orr, J. *Space Launch System Flight Control.* Aerospace Control and Guidance Systems Committee (ACGSC). Portland, ME. 10-12 Oct. 2012.

[8] Franco M., R. *Enhanced Dynamics and Control Analysis Package (DCAP).* Proceedings of ESA WPP-041, ESA Workshop on Spacecraft Guidance Navigation and Control. September, 1992.

[9] Franco M., R., Dumontel, L., Portigliotti, S., anb Venugopal, R. *The Dynamics and Control Analysis Package (DCAP) - A versatile tool for satellite control.* ESA Bulletin, Nr. 87. August, 1996.

[10] Portigliotti, S., Dumontela, M., Baldesi, G., and Sciacovelli, D. *DCAP (Dynamics and Control Analysis Package) an Effective Tool for Modelling and Simulating of Coupled Controlled Rigid Flexible Structure in Space Environment.*

[11] Baldesi, G., Sciacovelli, D., and Thirkettle, A. *Simulation tool for generic launcher flight dynamics-control interaction analysis.* Proceedings of the 6th Int. Symp. on "Launcher Technologies: Flight Environment Control for Future and Operational Launchers". Munich, Germany. November, 2006.

[12] Baldesi, G., Toso, M. *ESA launcher flight dynamics simulator used for system and subsystem level analyses.* Proceedings of the 11th Intl. Workshop on Simulation & EGSE facilities for Space Programmes (SESP 2010). ESTEC. September, 2010.

[13] Baldesi, G., Toso, M. *EEuropean Space Agency's launcher multibody dynamics simulator used for system and subsystem level analyses.* CEAS Space Journal (2012) 3:2748. DOI 10.1007/s12567-011-0023-9.

[14] Elmqvist, H., Mattson, S.E., and Otter, M. *Modelica - An International Effort to Design an Object-Oriented Modeling Language.* Summer Computer Simulation Conference, pp. 333-339, 1998, ISBN 1-56555-149-4.

[15] Mattson, S.E., Elmqvist, H., and Otter, M. *Physical system modeling with Modelica.* Control Engineering Practice, v. 6, pp. 501-510, 1998.

[16] Tiller, M. *Introduction to Physical Modeling with Modelica.* The Springer International Series in Engineering and Computer Science, 2001.

[17] Otter, M. and Olsson, H. *New features in Modelica 2.0.* Proceedings of the 2nd International Modelica Conference. Oberpfaffenhofen, Germany. March, 2002.

[18] Fritzson, P. *Principles of Object-Oriented Modeling and Simulation with Modelica 2.1.* John Wiley & Sons, Jan. 2004

[19] Reiner, M. and Bals, J. *Nonlinear inverse models for the control of satellites with flexible structures.* Proceedings of the 10th International Modelica Conference. Lund, Sweden. 2014, pp 577-587.

[20] Modelica Association, editor. *Modelica - A Unified Object- Oriented Language for Physical Systems Modeling Language Specification Version 3.2.* 2010.

[21] Otter, M., Elmqvist, H., and Mattsson, S.E. *The New Modelica Multibody Library.* Proceedings of the 3rd International Modelica Conference. Linköping, Sweden. 2003, pp 311-330.

[22] Looye. G. *The New DLR Flight Dynamics Library.* Proceedings of the 6th International Modelica Conference. Bielefeld, Germany. 2008, pp 193-202.

[23] Heckmann, A., Otter, M., Dietz, S., and Lopez, J. *The DLR flexible bodies library to model large motions of beams and of flexible bodies exported from finite element programs.* Proceedings of the 5th International Modelica Conference. Vienna, Austria. 2006, pp 85-95.

[24] Bellmann, T. *Interactive simulations and advanced visualization with Modelica.* Proceedings of the 7th International Modelica Conference. Como, Italy. 2009, pp 541550.

[25] Pfeiffer, A. *Optimization library for interactive multi-criteria optimization tasks.* Proceedings of the 9th International Modelica Conference. Munich, Germany. 2012, pp 669680.

[26] Lemoine, F., Kenyon, S., Factor, J., Trimmer, R., and others. *The Development of the Joint NASA GSFC and the National Imagery and Mapping Agency (NIMA) Geopotential Model EGM96.* NASA/TP-1998-206861, NASA Goddard Space Flight Center. July, 1998.

[27] Picone, J. M., Hedin, A. E., Drob, D. P. and Aikin, A. C. *NRLMSISE-00 empirical model of the atmosphere: Statistical comparisons and scientific issues.* Journal of Geophysical Research: Space Physics, volume 107, number A12, 2002.

[28] Eke, F. O. *Dynamics of Variable Mass Systems.* NASA CR-1998-208246, NASA Ames Research Center, January 1999.

[29] Otter, M., Elmqvist, H., and Cellier, F.E. *Modeling of Multibody Systems with the Object-Oriented Modeling Language Dymola.* Nonlinear Dynamics 9: 91-112, 1996.

[30] Toniolo, M., Tartabini, P. V., and Pamadi, B. N. *Constraint Force Equation Methodology for Modeling Multi-Body Stage Separation Dynamics.* 46th AIAA Aerospace Sciences Meeting and Exhibit, 2008, 10.2514/6.2008-219.

[31] Tartabini, P. V., Roithmayr, C. M., Toniolo, M. D., Karlgaard, C. D., and Pamadi, B. N. *Modeling Multibody Stage Separation Dynamics Using Constraint Force Equation Methodology.* Journal of Spacecraft and Rockets, Vol. 48, No. 4, July-August 2011, pp 573-583.

[32] Pamadi, B.N., Tartabini, P.V., Toniolo, M.D., Roithmayr, C.M., Karlgaard, C.D., and Samareh, J. A. *Application of Constraint Force Equation Methodology for Launch Vehicle Stage Separation.* Journal of Spacecraft and Rockets, Vol. 50, No. 1, January-February 2013, pp 191-205.

[33] Baumgarte, J. *Stabilization of Constraints and Integrals of Motion in Dynamical Systems.* Computer Methods in Applied Mechanics and Engineering, Vol. 1, No. 1, 1972, pp. 1-16.

[34] Ascher, U.M., Chin, H., Petzold, L.R., and Reich, S. *Stabilization of constrained mechanical systems with DAEs and invariant manifolds.* Journal of Mechanics of Structures and Machines, Vol. 23, 1994, pp 135-157.

[35] Acquatella B., P., and Reiner, M. J. *Modelica stage separation dynamics modeling for End-to-End launch vehicle trajectory simulations.* Proceedings of the 10th International Modelica Conference. Lund, Sweden. 2014, pp 589-598.