69th International Astronautical Congress (IAC), Bremen, Germany, 1-5 October 2018.

IAC-18,D1,4B,8,x43716

Enabling a Conceptual Data Model and Workflow Integration Environment for Concurrent Launch Vehicle Analysis

Philipp M. Fischer[a]*, Meenakshi Deshmukh[a], Aaron D. Koch[b], Robert Mischke [d]
Antonio Martelo Gomez[c], Andreas Schreiber[d], Andreas Gerndt[a]

[a] *DLR German Aerospace Center, Software for Space Systems and Interactive Visualization, Lilienthalplatz 7, 38108 Braunschweig, Germany, philipp.fischer@dlr.de, meenakshi.deshmukh@dlr.de, andreas.gerndt@dlr.de*
[b] *DLR German Aerospace Center, Space Launcher System Analysis, Robert-Hooke-Straße 7 , 28359 Bremen, Germany, aaron.koch@dlr.de*
[c] *DLR German Aerospace Center, System Analysis Space Segment, Robert-Hooke-Straße 7, 28359 Bremen, Germany, antonio.martelo@dlr.de*
[d] *DLR German Aerospace Center, Intelligent and Distributed Systems, Linder Höhe, 51147 Köln, Germany, robert.mischke@dlr.de, andreas.schreiber@dlr.de*
* Corresponding Author

Abstract

Concurrent Engineering (CE) and Model Based Systems Engineering (MBSE) have increased the efficiency of spacecraft, and satellite design in particular. Early design of satellites in Concurrent Engineering Centers (CEC) has almost become business as usual. However, such progress has still to be achieved for the design of launchers. Applying the same approaches as used for satellites has not led to the same amount of improvement, yet. To address this, DLR initiated the project Concurrent Launch Vehicle Analysis (CLAVA) to investigate the shortcomings and to improve the efficiency of conceptual launcher design and analysis. From an MBSE point of view, investigations show that concurrent modelling requires new Conceptual Data Models. In contrast to designing satellites, they are focused on a much more physical abstraction rather than a functional one. Regarding simulations, it has become clear that the conceptual design phase of launchers requires far more computationally intense simulations in a sequential order. With this knowledge, it is possible to outline a new process for CE studies allowing for concurrent design phases and sequential simulation phases. For this, an adjusted architecture of tools is required as well. The data model used for satellite studies within DLR's Concurrent Engineering Facility (CEF) does not fit to the requirements of launcher design and has been adapted. Additionally, DLR's aeronautics divisions have already made substantial progress in increasing the efficiency of their simulations. They employ automated simulation workflows using a parametric model for information exchange between integrated tools. This approach has been adopted and integrated. This paper outlines how this approach is combined with CE and MBSE concepts used for satellites and addresses the specific requirements of launcher design. It provides details about the database used during CE sessions, and how its information is transferred into the parametric data model used to run the required simulations. The conceptual data model of this database has been adapted to the physical representation of launchers; these changes will also be discussed. Furthermore, the general idea of the workflow and the design of the parametric model will be presented. The paper concludes by providing an outlook of how DLR intends to continue on this work, and further refine the developed tools and processes into daily CE and CEF application.
Keywords: Concurrent Engineering, Launcher Analysis, MBSE, CDM, RCE, Virtual Satellite

Acronyms/Abbreviations

Concurrent Launch Vehicle Analysis (CLAVA), Virtual Spacecraft Design (VSD), Concurrent Engineering (CE), Concurrent Engineering Facility (CEF), Concurrent Design Facility (CDF), Remote Component Environment (RCE), German Aerospace Center (DLR), European Space Agency (ESA), European Space research and Technology Centre (ESTEC), European Cooperation for Space Standardization (ECSS), Conceptual Data Model (CDM), Model Based Systems Engineering (MBSE), Common Parametric Aircraft Configuration Schema (CPACS)

1. Challenges in Concurrent Launcher Design

Concurrent Engineering (CE) has become a very efficient tool for early phase spacecraft design. So far it has been applied very efficiently in early phases of satellite design at various different design centers such as the Concurrent Engineering Facility (CEF) of the German Aerospace Center (DLR) in Bremen/Germany and the Concurrent Design Facility (CDF) of the European Space Agency (ESA) at the European Space research and Technology Centre (ESTEC) in Noordwijk/Netherlands. At DLR's site, several studies have been successfully executed. Supported by special

tooling and a well-defined process, the overall maturity of these studies has grown. Still, some of these studies concerned launchers. These launcher studies in particular highlighted that studying satellites and launchers cannot be handled the same way.

To better understand these differences and to improve the concurrent launcher design, DLR decided to establish the project Concurrent Launch Vehicle Analysis (CLAVA). The goal of this project is to improve the early launcher design by means of Model Based Systems Engineering (MBSE). Some major Results are: First, designing a launcher involves much more coupled simulation than designing satellites. Here sequential execution of simulations is needed and is naturally contradictive to the idea of concurrency. Second, the engineers prefer a much more physical abstraction for launchers rather than a functional abstraction as used for the system model of satellites during CE studies. Together with the currently used CE process, the data model based on Virtual Satellite, as well as the current simulation tooling used in launcher design, DLR has not yet reached the anticipated level of maturity for running highly efficient launcher design studies in the CEF.

As a consequence, some adjustments are needed. Since launcher design is highly dependent on the simulation result, the efficiency of this sequential process needs to be improved. This has been achieved by using DLR's Remote Component Environment (RCE) software for orchestrating the execution of the individual simulation tools. For the transfer of simulation results between tools, a new parametric model, called clavaModel has been used. It is inspired by the Common Parametric Aircraft Configuration Schema (CPACS). On the side of CE, a first draft of a new process has been created that allows for concurrent work as well as the sequential execution of simulations. On the side of the system model, which is an important asset for CE studies to enable a true MBSE approach, a new Conceptual Data Model (CDM) has been defined for Virtual Satellite. This CDM is the new language for the engineers supporting them specifically in the task of launcher design. Thus Virtual Satellite will be the primary design tool for engineers to enter the initial design into the parametric data model. RCE will then perform the overall simulation by passing the data model through the defined workflow of individual simulation tools.

This paper provides relevant background information to CE in satellite design as well as early launcher design. Based on this state of the art, a short comparison will highlight the shortcomings, on why launcher design is not yet as efficient as satellite design in CE. Starting from that knowledge, an overview to the resulting architecture of Virtual Satellite, RCE and the data models is given. The achievement of the new CDM will be discussed in detail, before an outlook for the next steps is provided and the results are summarized.

2. General Background to MBSE and CE

MBSE addresses the vision of using models instead of documents as central resource for system engineering activities already starting at conceptual design phases. [1] For the space industry, this has been adopted to create data bases with a central data model that interchanges information with domain relevant processes and domain models. The information stored in this central model is growing in size and maturity throughout the phases of designing, developing and operating a spacecraft. [2] Applying these MBSE data bases triggers positive side effects such as reuse of designs and information. [3]

The goal of CE is to achieve simultaneous design of a product regarding all its processes as well as improving the design upfront the production and thus reducing later costs. CE can be tracked back to the early days of automotive line productions and has ever since evolved. Beginning from a pure team based approach, digital models such as Computer Aided Design (CAD) have been introduced and needed a central platform for exchange such as the International Organization for Standardization (ISO) "Standard for the Exchange of Product Model Data" (STEP). [4] An ascoiated expected reduction in cost is already visible during design time. [5] The European Space Agency (ESA) once stated a reduction of design time from 6-9 months down to 3-6 weeks. [6]

2.1. State of the Art in Satellite Design

Concerning the data bases, various different ones exist such as Virtual Spacecraft Design (VSD) or the Open Concurrent Design Tool (OCDT) by ESA, RangeDB by Airbus Defence and Space, or Virtual Satellite by DLR. [7] [8] [9] [2] All these data bases aim to implement MBSE into some stage of the lifecycle of a spacecraft. All of them provide some sort of CDM implemented in some sort of tool. Such a tool can be used by the engineers to design a system model. [10] There are several upcoming standards such as the ECSS-E-TM-1023, ECSS-E-TM-10-25 and EGS-CC, which define the capabilities of the underlying CDMs. [11] [12] [13] These data bases have in common that they allow the engineers to break down the complexity of the designed system into a functional decomposition of subsystems, components and further elements. This is generally known as product structures. At each of these elements within a product structure engineers can store information. In most of these data bases the actual type of information can be stored and extended by so called engineering categories. Where in most of these data bases the product structures are fixed to product, configuration, assembly trees, etc., Virtual Satellite

allows changing and adopting these product structures to the actual use cases in the same fashion as provided by engineering categories. Such tailoring is captured in a so called concept. A further important aspect of the product structures are configuration control capabilities. It means that a component, such as a reaction wheel, is modelled in the product tree, with an assigned engineering category defining its mass to 5 kilograms for example. Every single instance of that reaction wheel within the configuration or assembly trees inherits the mass of this previously defined component. [2] Another important aspect of these data bases is collaboration. Engineers can work together on a common design since different domains can combine their information into an overall one. The data bases handle these concurrent data changes by either providing merge capabilities or rights management. The rights management of e.g. Virtual Satellite assigns one user to a sub system or component. Only that user can change information of that component at a time. [10]

A system model stored in such a data base also allows for a controlled data exchange with other domain models, e.g. for the configuration of simulators or for the purpose of visualization. [14] In the case of interchanging with a simulator data base, it is interesting to see that the simulator data base is not much different from the system data base, but still contains information which is not relevant for the system itself but only for the simulation. For example, the execution speeds/frequencies of simulation models are purely relevant for the configured simulators. [15]

While data bases like RangeDB or VSD address the development phases of a spacecraft, OCDT is focusing on the early phase of conceptual spacecraft design. Virtual Satellite is addressing both. Nevertheless, both OCDT and Virtual Satellite are just additional tools for this early design phase. At DLR and ESA this phase is usually conducted in so called concurrent engineering sessions. [10] To efficiently conduct these sessions it also needs a process and an interdisciplinary team in addition to the model. The studies themselves are usually performed in special offices such as DLR's CEF or ESA's CDF. [16] [17] Other facilities work in a quite similar manner, usually applying some sort of data model as well. [10] [18] So far, more than 60 studies have been performed in DLR's CEF. Most of them were focused on spacecraft, but some launcher studies were of interest as well. [19]

The process that is applied in DLR's CEF includes some upfront preparation as well as some post-processing around the actual CE sessions. Usually, these sessions are held within one or up to three weeks of intensive work, where the engineers iterate towards a common design. Virtual Satellite supports them to start off by decomposing the system into sub-systems and required equipment, which is then followed by phases of collecting important measures such as the overall mass and power consumption and further properties if needed. [10] Simulations usually play a secondary role and often focus on the mission trajectory. [20] Some other simulation work is focusing on system concept simulators to estimate the overall system performance during CE studies in ESA's CDF. [21]

The CDM of Virtual Satellite is designed to make it as easy as possible for the engineers to get used to the tool and to model the system. The software takes care of distributing all relevant information within the study team. It provides a simple product structure of just one tree for the decomposition. Measures such as the mass and power consumption are stored in predefined parameters and updated automatically. [10]

2.2. State of the Art in Launcher Design

MBSE in the context of launcher design is a rare occurrence. Instead, a strong focus is on multidisciplinary design optimization (MDO). [22] Several MDO techniques, used at ONERA, are suggested to be applied within a concurrent engineering process. [23] NASA's Jet Propulsion Laboratory has developed a flight portal, based on the Open Model Based Engineering Environment. [24] This portal contains launch system data. It is, however, geared towards mission planners intending to launch a spacecraft and not meant for launcher design itself. Another framework dedicated to mission optimization is using data mining techniques on data bases for initializing the investigated launch vehicle in the design process. [25]

2.3. Workflow Automation in Aeronautics and Space

Within the aeronautics domain at DLR a workflow engine has been applied for a multi-disciplinary design and automation (MDAO) process of future aircraft designs. Such a workflow consists of simulation tools that are sequentially executed and that interchange information via a data model called CPACS. CPACS stands for Common Parametric Aircraft Configuration Schema and is developed since 2005. [26] [27] This data model has also been foreseen as backbone for workflows in CE studies in the Integrated Design Laboratory (IDL). The IDL is another concurrent engineering facility at DLR with a focus on preliminary aircraft design. [28]

The execution of such workflows can be performed using RCE (Remote Component Environment), an open source, workflow-driven, distributed integration environment developed at DLR. It supports the design and execution of scientific and engineering workflows. It is especially suited for multidisciplinary collaboration where different groups or organizations benefit from integrating their specific tools into larger simulation workflows. So far, it has been primarily used in

preliminary aircraft design, but also in the optimization of thermal management of spacecraft. [29] A first integration with a system model of Virtual Satellite was achieved as well [30].

Airbus Defense and Space presents the application of such an approach for spacecraft budgeting and sizing. They use their data base RangeDB to feed into workflows executing various different simulation tools before feeding back analysis results into the system model. [31]

3. Evolving CE from Satellite to Launcher Design

As it can be seen, CE promises an increase in efficiency concerning the design of new spacecraft and satellites in particular. Even though launchers have already been designed in CE processes, the focus is still on the level of a functional conceptual design. Sure this is enough for a satellite, but the design of a launcher requires more advanced analysis on a physical level, e.g. the overall aerodynamics performance or detailed analysis of the trajectory and transfer capacities to target orbits. With this in mind, the challenges for integrating this into a concurrent engineering process become clearer. In fact, there are two important aspects to be addressed:

First, satellites are designed purely on a functional decomposition. This means, there will be sub-systems such as power or propulsion. For each sub-system, exactly one engineer will be in charge of the design, which means selecting the adequate components and sizing their masses, power consumptions, etc. Launchers instead need a much more physical approach rather than the functional one. A launcher needs to be decomposed into stages, fairings, noses, thrusters and boosters. Here it becomes difficult to assign just one engineer for exactly one sub-system. As an example, designing the fairing needs the involvement of the structural engineer and the one responsible for the aerodynamics. Either it requires the engineers to work sequentially, or there is a need of a data model that supports the engineers in working concurrently.

Second, satellite design does not involve a lot of intensive simulations. The majority of the simulation is focusing on the mission trajectory. This trajectory is more of a requirement to the needed launcher for the mission rather than a boundary condition for the design of the satellite. Designing a launcher, the trajectory becomes the dominant result of the study. To get a precise and representative estimate of the trajectory, it requires executing several independent simulation tools where the results of one are often the inputs to another. As a matter of fact, most of these simulations cannot be executed concurrently and completing the necessary number of iterations might require hours rather than minutes or seconds.

To adapt CE for launcher design, several aspects must be adjusted. Based on these two essential differences, and the knowledge of CE in satellite design some changes need to be done regarding the process, the overall tool architecture, and the data model.

3.1. Proposal for an Adjusted CE Process

The process is an essential part of successful concurrent engineering. Where in satellite design the data model and small involvement of simulations allow for quick iterations, the process needs to be adjusted for launcher design. Here it is important to accept that simulations have to be executed one after the other, and that results of a simulated launcher will only be available to the design team after several hours.

Fig. 1 shows the adjusted process that tries to split the daily activities into two major blocks, one of

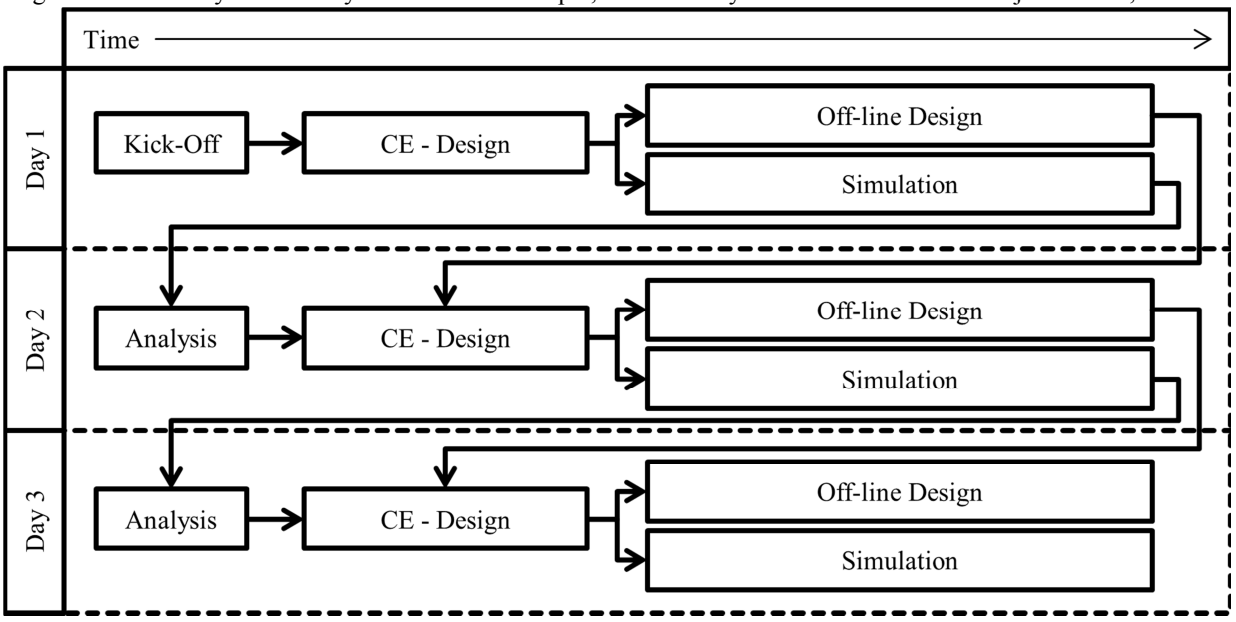

Fig. 1 An adjusted CE process allowing for concurrent work and sequential simulation phases.

concurrent design and one of simulation. At times of simulation, there is also some off-line design time considered. This design time provides time to the engineers to go back to their offices, continue with general work or prepare their domain work for the next day's CE session. The simulation is intended to start at the end of the day, thus it can be executed at night with the aim to provide results for the next morning. The next day usually starts with a quick analysis of the simulation results, before the engineers enter the next CE design session. The first day, of course, starts with some kick-off work, to brief the design team about the mission objectives, and further general aspects of their CE study.

3.2. Overview of Required Architecture

Similar to satellite design, the process dictates the kind of tooling that is needed. Where satellite design just asks for one central data model it is still not enough for the launcher design. One of the major differences is based in the amount of simulation which takes a much more dominant part in the overall study. Nevertheless, the coupling of simulations and the automated execution are following a defined workflow that has been successfully evaluated and applied within aeronautics of DLR.

Fig. 2 depicts the overall needed architecture. Same as in satellite design, Virtual Satellite is used as the system engineering tool and supports the concurrent design sessions. Based on the research results of RCE and CPACS in aeronautics, the simulation workflow is modelled and executed using RCE. The parametric data model, clavaModel, used to exchange the needed information between all simulation tools is based on concepts from CPACS. The simulation models are provided by the individual experts and institutes. With RCE, it is possible to execute the simulation tools on distributed systems, which is beneficial when using high performance computing resources, or confidential tools. In the case of the latter, RCE allows these codes to be used within the overall workflow without requiring them to be actually moved or installed across legal entities. Similar to what has been shown by Airbus Defence and Space of coupling a simulator to the system model, the system model of Virtual Satellite will not contain simulation relevant information. This information will be stored and processed in the parametric data model. Still, the system model of Virtual Satellite has to feed the system design to the parametric model and has to extract relevant results from it.

3.3. The Role of Virtual Satellite and the System Model

Virtual Satellite has already served well in satellite design and other CE studies. Its main advantage is that it is easy to use, even by unexperienced study participants. Every engineer is executing an instance of Virtual Satellite on the computer in front of them. Here they create a system model during the CE sessions. The model is refined every day. Virtual Satellite handles the data exchange for the engineers. But due to the intrinsic differences to launcher design, the CDM, providing the language to model a system, needs to be adjusted. It needs to be possible that different disciplines can work on the same part, e.g. the aerodynamics and structural experts working individually on the fairing, but their work needs to integrate into one common system model. This system model of the whole launcher is then used to

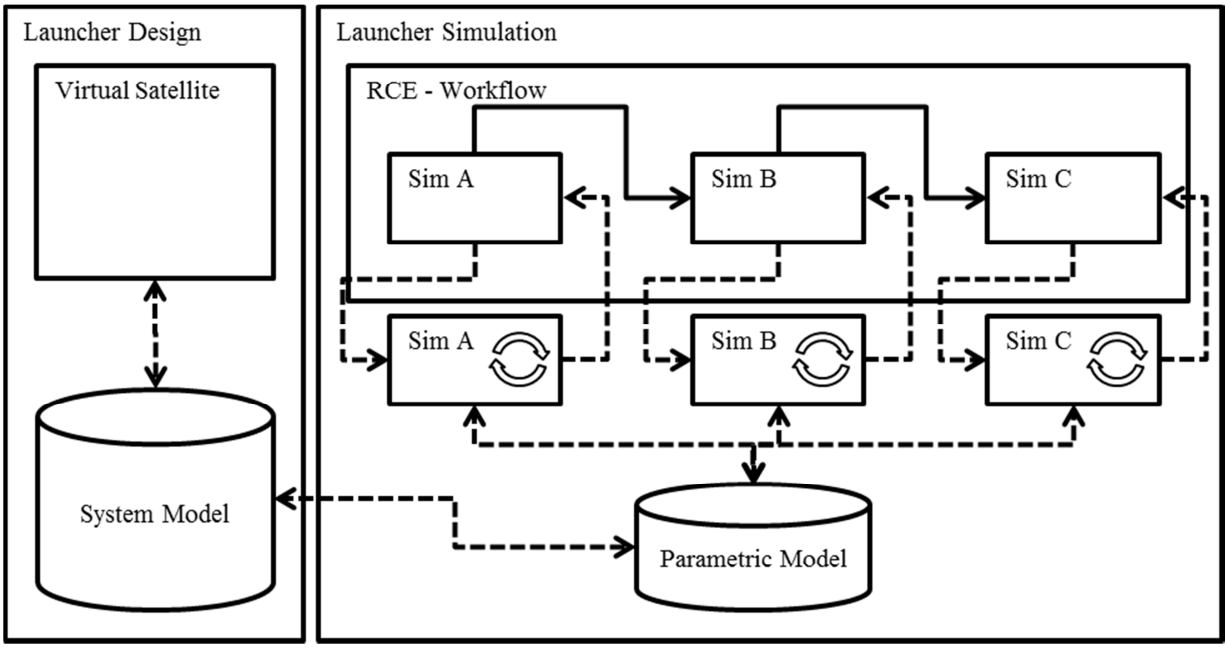

Fig. 2 The architecture overview connecting Virtual Satellite, the system model and the RCE simulation workflow.

update the parametric data model for running the next simulations.

3.4. The Role of RCE

The core idea of RCE is that complex simulation workflows are usually comprised of many individual steps, each addressing a part of the problem domain. These individual steps are usually represented by a program or script file with a command-line interface, commonly referred to as "tools". Within RCE, users can easily make their tools available as standardized workflow parts by using a graphical editor, in which they define the data inputs, outputs, and various execution settings that define how incoming data should be processed. This approach of including simulation tools into the overall workflow is indicated by Fig. 2.

An essential aspect of RCE is that it is distributed. Each instance of the program can choose to accept connections from other instances, which allows them to form collaborative networks. The most typical setup consists of several high-powered compute nodes, one or more end-user (client) machines, and one or more communication hubs in-between that serve as common entry points into the network.

Once a tool has been made available on a machine, e.g. Sim A, that machine's owner can choose to publish this tool to other machines in the network. On each RCE instance that has access to this tool, it can be used as if it is present on that machine. However, a published tool never actually leaves the machine it was originally installed on. This is especially useful for confidential or experimental tools. Using RCE, these tools can still be made available for scientific collaboration without handing out the actual software.

Using these distribution features, users can easily create and execute workflows that transparently combine any of the published tools within their network, using a single graphical workflow editor. This feature has already been highly useful in various research projects, and has also been essential for the integration of distributed simulation toolchains in CLAVA.

A special strength of RCE is the high adaptability of workflows. Beyond simple toolchains, optimization and convergence loops can also be defined, and also nested to achieve complex multi-disciplinary optimizations. This flexibility is especially useful in a project like CLAVA, where the overall design process is still an area of scientific experimentation. For example, a design variable (e.g. propellant loading) that has been manually provided so far may be considered for automatic optimization in a new process model. Using RCE, such changes can usually be performed quickly. Due to the stability of the standardized tool interfaces, it is also easy to keep different versions of a workflow to test and switch between different approaches as needed.

3.5. The Role of the Parametric Data Model

The parametric data model creates a common understanding for storing model information used in various simulation tools. Developing the clavaModel has required discussion and consent among the different disciplines involved in launcher design and analysis. Experts from system analysis, system dynamics and control, mechanical engineering, aerodynamics and propulsion have participated in defining the model parameters. This effort has also fostered the development of common definitions and a common understanding of launcher design.

The role of RCE is to orchestrate the workflow, toolchain and all the simulations. The parametric data model is used to transport information across the simulation tools, whereas the system model from Virtual Satellite is considered to provide and initialize system engineering relevant information to the clavaModel.

4. Enabling Concurrent Launcher Design

The previous chapters described the differences between satellite and launcher design, as well as what needs to be changed to improve efficiency of launcher design for concurrent engineering. This chapter will focus on the changes that were needed to achieve this goal. Two of the major fields of work were the CDM of Virtual Satellite to provide a language to the study engineers that is suitable for launcher design as well as the parametric data model to support the RCE based workflow of simulations.

4.1. Implementation of a new Conceptual Data Model

The CDM of Virtual Satellite allows for concurrent work on the design. In satellite design whole sub-systems are assigned to a single person, and only this person is responsible for designing the sub-system. This does not work for launcher design. As discussed the launcher design is focused on physical aspects rather than functional ones. This requires that various different persons responsible for different tasks, such as the structure or thermal design can contribute to the same part of the spacecraft. This is quite contradictive to how the CDM has been used so far, since two persons cannot work on the same part at the same time. A way out of this dilemma is to make use of the inheritance mechanisms of the CDM which allow inheriting properties from one component to another one, as well as the flexibility in defining new product structures.

As described, the data model has to provide some sort of representation for the launcher/rocket that is investigated during the study. Therefore, new product structures need to be defined as shown in Fig. 3. The final assembly is represented as a product tree representing the *Rockets*. *Rockets* can contain one or even more objects of the type *Rocket*. A *Rocket* itself

consists of *Bodies*, such as a main-stage, a booster etc. whereas a *Body* consists of one or several *Stages* and a *Fairing*.

A *Rocket*, designed by these elements, is usually assigned to the system engineer. The System Engineer is allowed to change the overall design and to define the properties of the parts. The properties are stored in predefined engineering categories such as one for structural or thermal properties. These categories can be assigned to a stage to e.g. define its mass.

In most cases, these values are not highly specific to the actual launcher but specific to the part. Therefore it is required that already defined parts can be reused. Hence, these parts are defined in another product tree called *Part Library.* This *Part Library* is used to define *Fairings* or *Stages* which are frequently reused or even preassembled *Bodies* out of these *Fairings* and *Stages.* These predefined parts are assigned to one engineer responsible for it. The system engineer can now reuse such a part by e.g. defining that the main stage of the rocket is inheriting from a main stage in the library. All defined properties are now copied over. In case needed, the system engineer can also override and change an inherited value.

To finally allow for the various different domains of e.g. thermal, structural, etc. to contribute to the same part at the same time, the responsible engineers combines information from so called *Part Information.* They are contained in the product tree called *Domains* and are structured into *Domain Information.* This means for example, the thermal engineer is responsible for a *Domain Information* called thermal. Here the engineer defines a *Part Information* for each *Body*, *Stage* or *Fairing* needed and adds the relevant thermal properties to it. The part engineer now inherits from the *Part Informations,* which means that e.g. a *Fairing* is now inheriting from the *Part Information* of thermal and structural and all other domains of interest.

As an instance example in Fig. 4 we can consider a *Domain Information* for the geometry and the structural domain. Both of them define a *Part Information* for an Ariane 6 fairing. The structural engineer adds a weight of ~2700 kg to the *Part Information* object, whereas the geometry engineer defines the contour of the fairing. The Ariane 6 fairing in the library is now inheriting from both, thus defining both properties of the contour as well as the weight. Both properties are then forwarded into the launcher where this fairing is actually used. The structural engineer for instance can now simply adjust the mass and this value automatically propagates through the model.

4.2. Implementation of the Parametric Data Model

The new parametric data model for launchers, named clavaModel, is based on CPACS heritage. It therefore exhibits similarities. The most apparent one is the usage of XML Schema to define the various parameters of the model. It is, however, a completely independent model with its own features. Most importantly, the clavaModel offers more than the schema. It consists of the clavaSchema and the clavaLibrary. The latter is a program library that gives access to the parameters via a C++ and a Python library. The Python version is generated based on the C++ code by using Boost.Python. [32] Contrary to the usage of CPACS, simulation tools are not supposed to directly access the XML file. Instead they are required to access the stored data via the clavaLibrary. The intention is threefold: (1) Many tools need to access the data. It is therefore better to invest in a common library rather than having to write a specific XML interface for every single tool. This approach also allows better handling of different versions of the clavaModel, while it is still under development. (2) At the start, the data in the XML

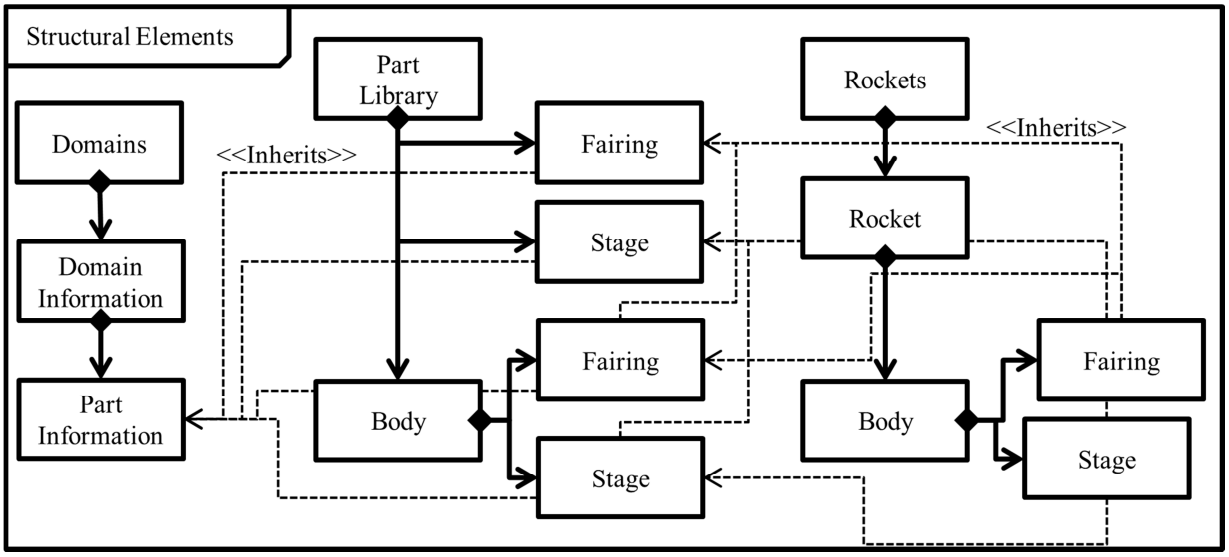

Fig. 3 The Conceptual Data Model enabling Concurrent Engineering of launchers.

file is read completely so that subsequent access is handled based on objects in memory rather than file access. A new file is only generated once a tool finishes its data manipulations and calls the write operation. Accessing the objects in memory is much more efficient than directly reading and writing the file multiple times, especially as this requires search operations. Potential memory problems associated with large files are avoided by storing large data sets, like aerodynamic data, in binary NetCDF files. These are only accessed when required. (3) It allows consistency checks across the entire model. This is only noticed by the user in case of errors. It is also easier to prevent the XML file to become corrupted by false usage when standardized and tested write functions are used. The aforementioned points are based on lessons learned from using CPACS.

The rocket data in the clavaModel is organized hierarchically, as depicted in Fig. 5. It contains five levels of building blocks. Depending on the available data, a rocket can be modelled up to any level. The different building blocks have been designed to share similarities. Therefore, they contain standard nodes like *structure*, *geometry*, and *propulsion*. The library takes care, in case a lower level is subsequently added, that these nodes are shifted on the next level. By example it means that, a body can have a contour assigned under the geometry node. If, however, the body is constructed from a fairing and several stages, then each of those building blocks on level 3 will contain a contour. The combination of those contours then constitutes the contour of the body. Another example is the usage of the function getMass() from the clavaLibrary. It exists on every level. Calling this function on the rocket level will always procure the total mass, regardless of the number of levels used. In addition to the rocket data, the clavaModel contains other relevant information, like trajectory data. An overall goal of the model design is to avoid redundancy, which helps to ensure a consistent design.

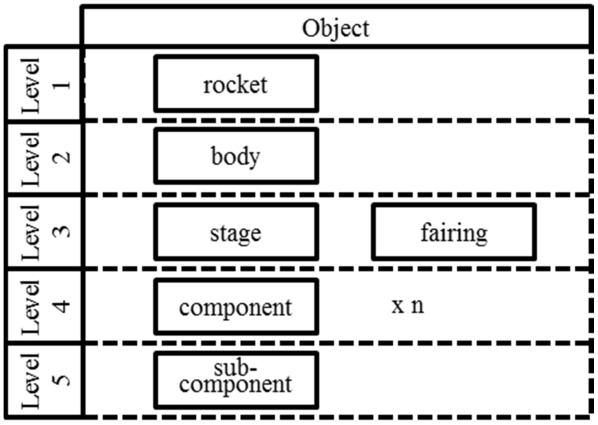

Fig. 5 Object hierarchy of the clavaModel

4.2.1. Implementation of the RCE Workflow

The application of RCE is analogous to the CPACS based workflows. Every tool receives an XML file as input. The tool itself needs to have a C++ or Python interface to make use of the clavaLibrary. It will first execute the read operation, then perform its own calculations, modify the data in memory, and finally call the write operation. The new XML file resulting from the last operation will then be handed over to the next tool.

4.3. Interaction with the Parametric Data Model

An important part of the overall architecture is the data exchange from the Virtual Satellite System Model to RCE and the parametric data model. This feature has been implemented using the App interface of Virtual

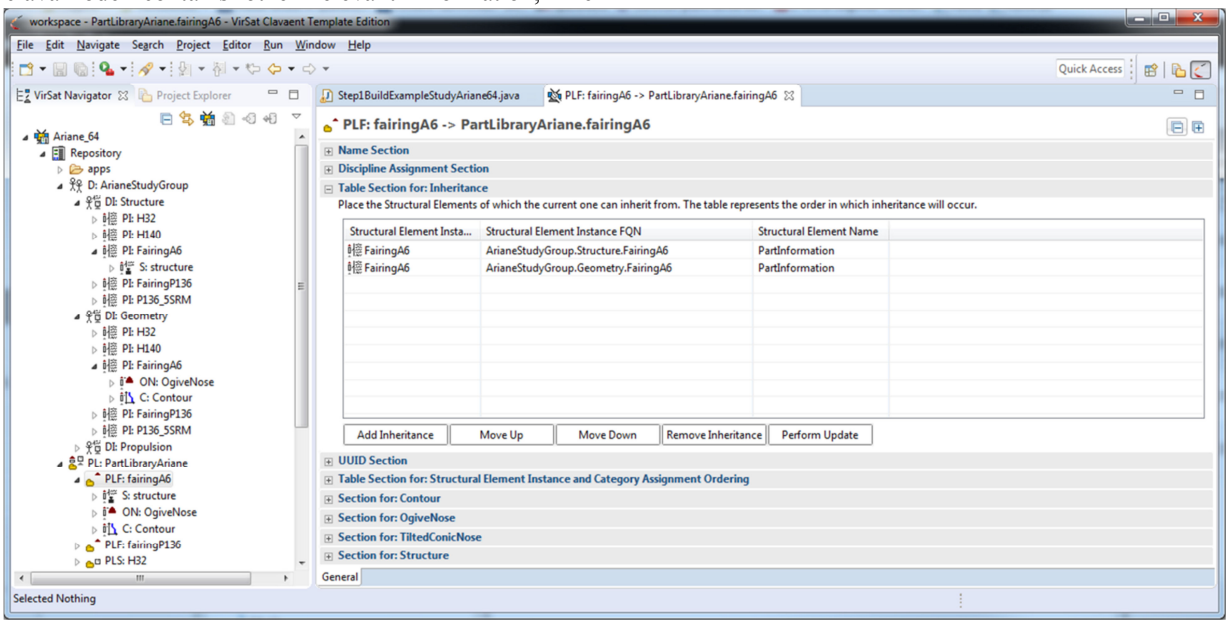

Fig. 4 The data model based on Virtual Satellite with a system model based on the new underlying CDM.

Satellite. This App interface provides a simplified Java based access to the data model. Every user is able to write their own App to either read from or write to the data model. In total, two Apps were implemented, of which one exports the *Part Library* and the *Rockets* into the parametric XML based data model. The second App is extracting simulation results, such as the maximum mass to geo transfer orbit (GTO), from the parametric data model and copies them to the correct place in the system model.

In this prototype, the App is writing the XML itself. In future implementations it is supposed to use the described library. The export makes additional use of inheritance information, and knowledge about overridden values. Usually a rocket part such as a body is defined in the library and only reused in the rocket. This is achieved by inheriting all values. This is also expressed within the parametric XML model, where a body of the rocket is referencing to one in the library section as indicated by the *idref*-attribute shown in Fig. 6. In case a value of that body is overridden in the system model, the export detects it, and rather than exporting the *idref* to the part library it is providing an inline definition of the body in the rocket section of the XML.

The workflow and process suggests that once a system design has been established, the system engineer can call the export app to initialize the parametric data model. Now, this data model can be used to execute the simulation workflow. Once finished the second script reads the results back into the original system model.

5. Outlook

The whole approach is still in a prototype stadium. The next steps need further thorough analysis in some real life scenarios. Even though the requirements to this approach are derived from real CEF studies it has not yet been fully tested in such an environment. Virtual Satellite has so far been used for designing spacecraft in the CEF same as RCE in executing simulation workflows for aircraft design. Still, they have not been used together for design studies. Even though, the clavaModel exists which can be interchanged with the CDM of Virtual Satellite, RCE and Virtual Satellite have not yet been executed together as design tools for a real study. As a consequence this displayed approach remains in the frame of concept. The next steps have to focus on improving and fine-tuning the integration of all tools and data models. This requires to continuously designing launcher related system models in Virtual Satellite and exporting them to the parametric data model using the clavaLibrary as well. It also needs to be verified, that such a data model can be understood by all simulation tools and automatically be executed in an RCE workflow. For some simulation tools interfaces need to be programmed to couple them to the library of the parametric data model. As soon as this integration is finalized, it is time to move into some real CEF studies to optimize the overall design process.

6. Summary

CE is not particularly new to designing spacecraft. It has also been applied to designing launchers. Nevertheless, it turned out that it needs an adapted approach compared to what is done in satellite design. One of the main reasons is the more physical view and analysis of the system model of a launcher. Such a model contains more interdependencies, and analyzing it requires efficient sequential execution of simulations. This requires a change in several respects.

First of all, a change in the CEF process is needed, which allows for simulation time same as times of non-

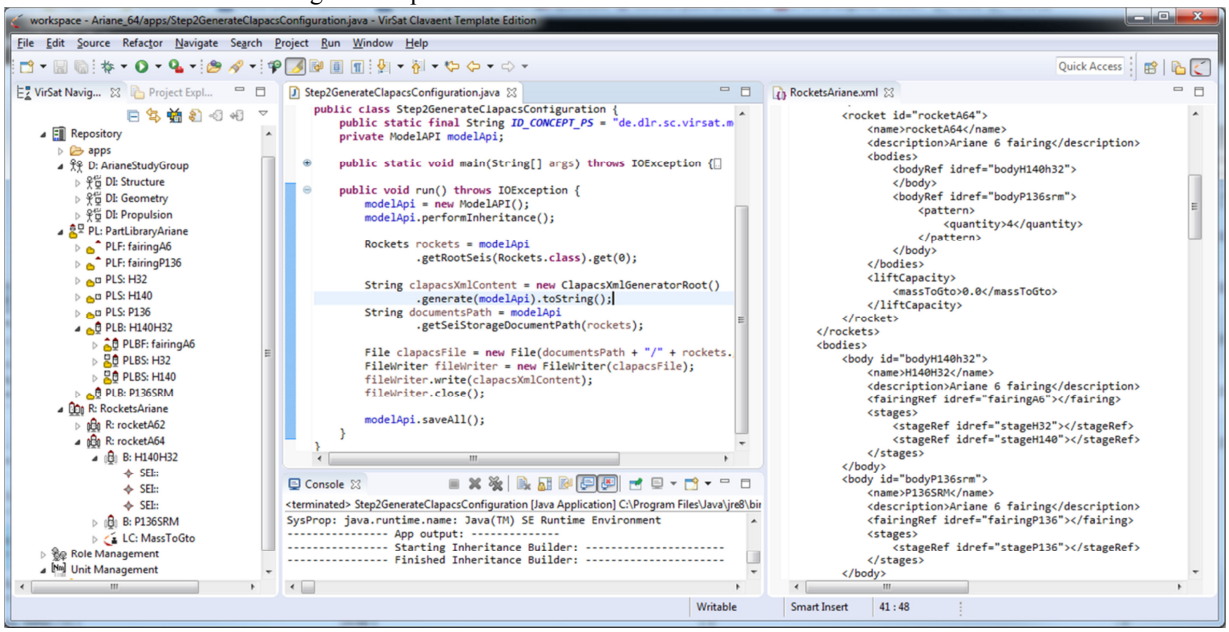

Fig. 6 The system model, the app for export and the parametric data model making use of inheritance.

concurrent work. This helps the design team to work on the system model in the morning, and start the simulations in the afternoon and evening.

The second change addresses the system model for the CEF. It has to provide the required physical representation of a launcher. This is achieved by defining a new product structure. This structure allows the domain experts to individually model their relevant aspects of the launcher parts, before they get integrated into a part library. This is possible by using the inheritance mechanisms such as the ones of the data models of VSD or Virtual Satellite.

The third change is the introduction of the parametric data model called clavaModel. It is tailored to launchers. By lessons learned from aircraft design, this data model can be read and written by all relevant simulation tools through a library. Obviously these tools need to be adjusted as well before they can be integrated into an automated workflow being executed in RCE.

By now the first versions of all three parts are developed and are in place. A thorough evaluation of these tools and the new process is part of ongoing work. Nevertheless, the individual parts are developed and designed on lessons learned from current best practices in spacecraft and aircraft design. Thus minor adjustments to both process and tools are expected rather than fundamental changes. Now, it is about refining the developed tools and processes within daily CE and CEF work.

References

[1] INCOSE, "Systems Engineering Vision 2020 - Version 2.03," International Council on Systems Engineering INCOSE, 2007.

[2] P. M. Fischer, D. Lüdtke, C. Lange, F.-C. Roshani, F. Dannemann and A. Gerndt, "Implementing model-based system engineering for the whole lifecycle of a spacecraft," *CEAS Space Journal,* 12 July 2017.

[3] C. Lange, J. T. Grundmann, M. Kretzenbacher and P. M. Fischer, "Systematic reuse and platforming: Application examples for enhancing reuse with model-based systems engineering methods in space systems development," *Concurrent Engineering: Research and Application,* 6 November 2017.

[4] H. H. Jo, H. R. Parsaei and W. G. Sullivan, "Principles of concurrent engineering," *Concurrent Engineering: Contemporary issues and modern design tools,* pp. 3-23, 1993.

[5] M. Lawson and H. M. Karandikar, "A Survey of Concurrent Engineering," *Concurrent Engineering: Research and Applications,* no. 2, pp. 1-6, 1994.

[6] D. D. Domizio and P. Gaudenzi, "A Model for Preliminary Design Procedures of Satellite Systems," *Concurrent Engineering: Research and Applications,* vol. 16, no. 2, pp. 149-159, June 2008.

[7] ESA, "Virtual Spacecraft Design," 2013. [Online]. Available: http://www.vsd-project.org/. [Accessed 20 4 2016].

[8] H. P. de Koning, S. Gerené, I. Ferreira, A. Pickering, F. Beyer and J. Vennekens, "Open Concurrent Design Tool - ESA Community Open Source - Ready to Go!," 2014. [Online]. Available: http://esaconferencebureau.com/docs/default-source/14c08_docs/open-concurrent-design-tool---esa-community-open-source-ready-to-go!.pdf. [Accessed 27 01 2017].

[9] H. Eisenmann, C. Cazenave and T. Noblet, "RangeDB the product to meet the challenges of nowadays System Database," in *Simulation and EGSE for Space Programmes (SESP)*, Noordwijk, Netherlands, 2015.

[10] P. M. Fischer, M. Deshmukh, V. Maiwald, D. Quantius, A. Martelo Gomez and A. Gerndt, "Conceptual Data Model - A Foundation for Successful Concurrent Engineering," *Concurrent Engineering: Research and Applications,* 2017.

[11] ECSS Secretariat, "ECSS-E-TM-10-23A - Space engineering - Space system data repository," ESA-ESTEC Requirements & Standards Devision, Noordwijk, Netherlands, 2011.

[12] ECSS Secretariat, "ECSS-E-TM-10-25A - Space engineering - Engineering design model data exchange (CDF)," ESA-ESTEC Requirements & Standards Division, Noordwijk, Netherlands, 2010.

[13] M. Pecchioli, A. Walsh, J. M. Carranza, R. Blommestijn, M.-C. Charmeau, M. Geyer, C. Stangl, P. Parmentier, H. Eisenmann, J. Rueting, P. Athmann, W. Bothmer, I. Krakowski, P.-Y. Schmerber, F. Chatte, P. Chiroli and M. Poletti, "Objectives and Concepts of the European Ground Systems Common Core (EGS-CC)," in *Simulation & EGSE for Space Programmes*, Noordiwjk, Netherlands, 2012.

[14] M. Deshmukh, R. Wolff, P. M. Fischer, M. Flatken and A. Gerndt, "Interactive 3D Visualization to Support Concurrent Engineering in the Early Space Mission Design Phase," in *5th CEAS Air and Space Conference*, Delft, Netherlands, 2015.

[15] P. M. Fischer, H. Eisenmann and J. Fuchs,

"Functional Verification by Simulation based on Preliminary System Design Data," in *6th International Conference on Systems & Concurrent Engineering for Space Applications (SECESA)*, Stuttgart, Germany, 2014.

[16] M. Bandechi, B. Melton, B. Gardini and F. Ongaro, "The ESA/ESTEC Concurrent Design Facility," in *Proceedings of the 2nd Concurrent Engineering Conference (EuSEC)*, Munich, Germany, 2000.

[17] O. Romberg, A. Braukhane and H. Schumann, "Status of the Concurrent Engineering Facility at DLR Bremen," in *German Aerospace Congress*, Dresden, Germany, 2008.

[18] C. Iwata, S. Infeld, J. M. Bracken, M. McGuire, C. McQuirk, A. Kisdi, J. Murphy, B. Cole and P. Zarifan, "Model-Based Systems Engineering in Concurrent Engineering Centers," in *AIAA SPACE 2015 Conference and Exposition*, Pasadena, USA, 2015.

[19] A. Martelo, S. S. Jahnke, A. Braukhane, D. Quantius, V. Maiwald and O. Romberg, "Statistics and Evaluation of 60+ Concurrent Engineering Studies at DLR," in *68th International Astronautical Congress (IAC)*, Adelaide, Australia, 2017.

[20] A. Braukhane and V. Schaus, "Model-based Engineering applied in Concurrent Engineering: Concurrent Engineering," in *Space Studies Program (SSP), International Space University (ISU)*, Straßbourg, France, 2013.

[21] S. Kranz, D. Vicente, G. G. Borja, A. Matthyssen and M. Fijneman, "System Concept Simulation for Concurrent Engineering," in *6th International Conference on Systems & Concurrent Engineering for Space Applications (SECESA)*, Stuttgart, Germany, 2014.

[22] M. Balesdent, N. Bérend, P. Dépincé and A. Chriette, "A survey of multidisciplinary design optimization methods in launch vehicle design," *Structural and Multidisciplinary Optimization,* vol. 45, no. 5, pp. 619-642, 2012.

[23] L. Brevault, M. Balesdent and S. Defoort, "Preliminary study on launch vehicle design: Applications of multidisciplinary design optimization methodologies," *Concurrent Engineering: Research and Applications,* vol. 26, no. 1, pp. 93-103, 2018.

[24] C. Singh-Derewa and P. Srivastava, "Launch: A Model Based Systems Engineering Platform for Rapid Collaboration on NASA Launch-Flight System Integration," in *68th International Astronautical Congress (IAC)*, Beijing, China, 2016.

[25] J.-I. Shu, J.-W. Kim, J.-W. Lee and S. Kim, "Multidisciplinary mission design optimization for space launch vehicle based on sequential design process," *Journal of Aerospace Engineering,* vol. 230, no. 1, pp. 3-18, 2016.

[26] J. Jepsen, D. Böhnke and B. Nagel, "Beschleunigung des geometrischen Erstentwurfs durch wissensbasierte Methoden," in *Deutscher Luft- und Raumfahrtkongress*, Stuttgart, Germany, 2013.

[27] Deutsches zentrum für Luft- und Raumfart, "CPACS," [Online]. Available: http://www.cpacs.de/. [Accessed 9 9 2018].

[28] A. Bachmann, J. Lakemeier and E. Moerland, "An Integrated Laboratory for Collaborative Design in the Air Transportation System," *Concurrent Engineering Approaches for Sustainable Product Development in a Multi-Disciplinary Environment,* vol. 1, pp. 1009-1020, 2013.

[29] D. Seider, A. Basermann, R. Mischke, M. Siggel, A. Tröltzsch and S. Zur, "Ad hoc Collaborative Design with Focus on Iterative Multidisciplinary Process Chain Development applied to Thermal Management of Spacecraft," in *4th CEAS Air & Space Conference*, Linköping, Sweden, 2013.

[30] D. Seider, M. Litz, A. Schreiber, P. M. Fischer and A. Gerndt, "Open source software framework for applications in aeronautics and space," in *IEEE Aerospace Conference*, Big Sky, Montana, USA, 2012.

[31] S. Estable, "Application of the 'Federated and Executable Models' MBSE Process to Airbus Orbital Servicing Missions," in *Phoenix Integration- International Users Conference*, annapolis, Maryland, USA, 2018.

[32] D. Abrahams and S. Seefeld, "Boost.Python - 1.64.0," boost C++ Libraries, 04 2017. [Online]. Available: https://www.boost.org/doc/libs/1_64_0/libs/python/doc/html/index.html. [Accessed 29 08 2018].