CDP4-COMET

A platform for Collaborative Model-based System Engineering

User Manual

Document type: Manual

Document ID: CDP4-COMET-MAN

Document classification: Proprietary

Approval date: 29/08/2024

Table of contents

1 Introduction

1.1 What is Concurrent Design?

Concurrent Design is a method to accelerate complex engineering projects in a series of working sessions where the client, experts from various domains and system engineers work together to develop an integral consistent design based on shared understanding and rigorous design decisions. The method improves the rigor and relevance of the study, accelerates the process, improves shared understanding and is focused on identifying key trade-offs and challenges early in the design process. Concurrent Design emphasises a quantitative fact-based approach to increase the quality of decision-making despite the uncertainties and interdependencies characteristic for the early phases of design.

1.2 Goal & Origin of Concurrent Design

Concurrent Design was pioneered in Europe the European Space Agency (ESA) in the late nineties. Concurrent Design has its origin in System Engineering. To design a system, we start from the needs of the client. Next, we translate these to functional demands of a system. Following an iterative process, a system is designed. In this process different components of the system are linked or related. Finally, we evaluate the performance of the system as a whole. We then compare the performance with the requirements to verify if the design meets the requirements. Further, we validate the design in comparison to the customer needs and the problem that needs to be solved, to evaluate if the solution indeed resolves the problem. When a system consists of many subsystems and parts, the solution requires extensive collaboration amongst experts. Traditional system engineering approaches such problem with a waterfall approach. Needs are specified in requirements that are implemented in a design by different experts that in turn add a part to the system and hand over their work to a next expert. When the system as a whole does not meet the requirements, the problem has to be resolved. In an integrated system, this implies that many subsystems are affected by the changes, and need to revise the design, resulting in new problems with interdisciplinary consequences. This approach therefore takes a lot of time and is inefficient due to re-work and setbacks in the design. Further, problems are not approached in an integrated matter, which can result in cascading problems throughout the design. Solving these problems can cause a lot of work, and when overlooked, these can cause faults in the system that are discovered late and cause larger problems. The key objective of a Concurrent Design approach is thus to create a more integrated design in which trade-offs are evaluated by the customer and design challenges are solved in a multidisciplinary manner. Such design can serve well as a starting point for Model-based System Engineering (see also 1.7 on MBSE).

1.3 5 pillars of Concurrent Design

Concurrent Design consists of five important elements:

1) A multidisciplinary team
2) A structured process
3) Rigorous methods and techniques for decision making

4) A facility and modelling infrastructure

5) An integrated design model

The Concurrent Design team consists of a group of carefully selected stakeholders that represent key domains of expertise and critical stakeholders. Participants are selected to have the mandate to make decisions in the study with respect to the domain or organization they represent to ensure progress and commitment at the end of the study.

Concurrent Design uses a structured process for each study. This process starts with an initiation phase in which the team evaluates if the project can be supported with Concurrent Design and if the (human) resources to organize and conduct the study can be ensured. Next, in the preparation phase a small 'core team' for the study is set up to decompose the goals and ambitions of the study into a step-by-step process and a roadmap for the study. Furthermore, the team will design the basis for the integrated study model. After this preparation phase the study phase will start. In this phase a sequence of sessions will take place on a (bi) weekly basis. To create an efficient iteration rhythm, each session takes place at the same time, same day, preferably every week. All stakeholders are present at each session to ensure presence and participation of all key domains in the project. The study phase starts with a kick-off session and after 5-8 sessions closes with a joint statement in which commitment towards the results is confirmed by all key stakeholders. The study ends with closure phase.

Concurrent Design has a library of methods and techniques to support decision making ranging from simple dialogue technique to complex mathematical decision models. Depending on the study and the decisions required for the project, these techniques are prepared and used.

For the Concurrent Design approach, a special facility with dedicated equipment is advisable. Co-location is a key success factor for Concurrent Design. For online Concurrent Design a video conferencing tool and shared cloud solution is required. Further, the CDP4-COMET server should be installed to host the integrated design model.

Finally, Concurrent Design uses an Integrated Design Model to support the analysis of the solution on a system level. The model is build based on System Engineering principles, in which the requirements are modelled and compared to different solutions to test feasibility and to validate various solution directions. To build an Integrated Design Model, a modelling platform is required. All key stakeholders should be able to contribute to the model. For this purpose, CDP4-COMET is built (see 2.1 on CDP4-COMET). CDP4-COMET is a platform to support Concurrent Design and Model-based System Engineering. It is designed to enable a team of experts to model a system at different levels of abstraction, and to verify if it meets requirements.

1.4 Application areas of Concurrent Design

Concurrent Design focuses on complex multidisciplinary design challenges and thus is used to design complex systems, both from a technical perspective (the system crosses several technical domains) as well as an organisational perspective (there are multiple stakeholders involved in the design of the system) These complex multidisciplinary problems are also called wicked problems and have one or several complex characteristics.

First, many stakeholders are involved. A Concurrent Design study focusses on a problem that involves different disciplines, domains or areas of expertise. Because there are several disciplines involved, there are different stakes and interests in the design. Further, each discipline brings its own body of knowledge to the problem, which on one hand adds expertise and insight but also increases the number of different perspectives increasing the risk of misunderstanding and misalignment. Creating structure, overview and shared understanding of the design and its key trade-offs is one of the key aspects of Concurrent Design.

Next, complexity can be raised by many standards, requirements and frameworks involved. Often the design is challenged by additional requirements resulting from the need to comply to standards, rules, regulations and laws. Next, systems often involve technical complexity. This means they have many interrelated components that are highly integrated and connected. Further, the system can be networked with other systems as part of a system of systems and can rapidly develop due to ongoing innovation.

Besides technical complexity, complexity can also be raised by organisational complexity where many internal stakeholders are involved in using and operating the system.

Next, a typical characteristic of complex systems is interdependency and interrelatedness. This means that when an aspect of the system is adjusted, other aspects are also affected and will need to change accordingly. This requires Trade-offs and complex decision making. The customer will have to choose which aspects are most important and thus needs to understand the consequences of these choices. This often manifest as a complex puzzle that requires a thorough understanding of the solution and the interrelated impact of choice on the design.

Finally, the surrounding of the system can pose complexity, dynamics in developments, time pressure and political pressure can complicate the design process and limit options.

Systems with this level of complexity are found in many domains of engineering ranging from large systems such as space, maritime, automotive and aviation systems. But also complex production systems, software systems and infrastructure systems can encompass this level of complexity. Even when the complexity balance leans toward the stakeholder complexity rather than the technical complexity, the team can benefit from Concurrent Design.

1.5 Course of a CD Study

A Concurrent Design Study encompasses four phases.

- **Initiation Phase**: Concurrent Design studies start with an initiation phase in which we evaluate if they project can be supported with Concurrent Design and if the key resources to organize and conduct the study can be ensured. In this phase the complexity of the study is assessed to determine if the problem requires a multidisciplinary and model-based approach.

- **Preparation Phase**: During the CD activity preparation, the agenda is further refined, and the team prepares for the first set of sessions and sets up a CDP4-COMET model. The preparation phase typically lasts between 2 to 4 weeks.

- **Session execution phase**: During the session execution phase, a sequence of CD Sessions is held to support an iterative design process. Actions, decisions, and important discussions are recorded in a so-called ADI-log. Depending on the number of sessions and the session rhythm (1 per week / 2 per week), the session execution phase lasts between 4 to 8 weeks.

- **Closure phase**: During the Closure phase, a final report is generated which is one of the deliverables of a Concurrent Design activity. The Closure phase lasts between 1 to 2 weeks.

1.6 Concurrent Design team composition

The Concurrent Design process defines clear roles and responsibilities as well as work that needs to be executed at certain points during the process. The participants involved in a CD Study form a CD Team. The following roles can be distinguished:

- **CD Customer**: The CD customer is the recipient of the work that is being done by a CD team. Typically, this is a project manager or end-user representative that can make decisions on the content of the problem specification, i.e. the requirements.
- **CD Team Leader**: The CD Team Leader is responsible for the overall organisation and execution of a Concurrent Design activity. A critical factor of CD is to discuss and make decisions based on fact, while still being able to work on the basis of assumptions that are made explicit during the work sessions. The role of the Team Leader is to act as facilitator during the co-located work sessions making sure the team transitions from assumptions to facts and makes sounds decision to progress the work.
- **CD System Analyst**: The CD System Analyst keeps track of the Actions, Decisions and Information items in a so-called ADI-Log. The System Analyst stays in close contact with the various team members to make sure preparation work is done and the session content is properly organized.
- **CD Modeller**: The CD modeller supports the team with the creation of a CDP4-COMET model. This model contains requirements, solution architecture and multiple key budgets such as overall mass, power, cost, etc.
- **CD System Engineer**: The CD System Engineer is responsible for synthesizing the information provided by the domain experts and to make sure that an optimal system is designed. The system engineer maintains a holistic view of solutions provided and makes sure the design meets the requirements as set by the CD Customer.
- **CD Domain Expert**: CD Domain Experts are subject matter experts required to design or debate the complex system/process that is the subject of the Concurrent Design activity. The subject matter experts provide information on the basis of which decisions are made to progress with the challenges at hand.

A Concurrent Design Team has to be composed specifically for the study (see 4.3 on team composition).

1.7 Concurrent Design & Model-based System Engineering

While traditional System Engineering made the step towards model-based and lifecycle-focused engineering recently, Concurrent Design has had an integrated design model at the 'roots' of its

approach as early as the 1990s. At that time, the Engineering Model was focused on SWP budgets and key design drivers, but the key premiss of MBSE, which is having one single point of truth that is interconnected to avoid mistakes in 'copying' information from one document to another, has always been a starting point and baseline in the approach. While focussing on early phase design, Concurrent Design also prescribed a lifecycle perspective involving not only engineers and end-users, but also experts on compliance, costs, risk and planning. System engineering has brought rigour and relevance to the engineering process by structuring it in phases and ensuring an iterative approach, with frequent verification and validation creating a loop between requirements and design. Model-based system engineering (MBSE) brings that rigour to the next level.

In a document-based engineering process, requirements are subject to interpretation and complex user requirements are extremely difficult to capture, with lengthy requirements documentation that often includes incomplete, ambiguous and conflicting requirements. Project managers struggle to ensure validation and verification is done correct and precisely, and often fail to identify and analyse alternative solutions to trade-offs identified in the design. Furthermore, as the volume of documentation increases, the manual overhead of maintenance of these artifacts quickly becomes unmanageable. A model-based approach will enable more scrutiny in verification and validation, creating direct links between the design and its requirements. However, it poses a new challenge in involving end-users in the design, as trade-offs are analysed within the complex modelling environment, making it hard for end-users to understand the complexity of the trade-off and the (im)possibilities of the design challenge.

2. CDP4-COMET PLATFORM

2.1 Role & purpose of the Concurrent Design platform

CDP4-COMET is designed to support model-based system engineering using the Concurrent Design method. CDP4-COMET enables modular system engineering in a collaborative approach in which stakeholders can modify design parameters and verify and validate the overall design on system requirements and system overall performance. CDP4-COMET software is open source. With CDP4-COMET the team can build an integrated design model for the study. CDP4-COMET is a collaborative tool in which all key stakeholders have access to the model and ownership of the information regarding their domain of expertise. In this model a decomposition of the system is established in which elements of the system and associated parameters represent factors that enable analysis of the key challenges in the design. A detailed documentation of the requirements of the system enables a rigorous comparison between the design and its requirements to reveal key trade-offs and challenges in the performance of the system. The model enables an early analysis of key design challenges and factors driving the choices in the design. In this way the model serves as a single point of truth for the team.

CDP4-COMET supports the Concurrent Design processes (see 1.1 on Concurrent Design), allowing multidisciplinary teams to work compliant with the Concurrent Design method. This involves customer participation and access to the design information within the CDP4-COMET model. It provides a collaborative working environment for collocated teams through an intranet connection or geographically distributed teams through an internet connection. CDP4-COMET supports a clear

division of tasks and responsibilities according to role assignment and additional settings to determine access rights to the integrated design model.

As any System Engineering model, a CDP4-COMET model is set up from a set of Requirements and a Product Design. These are decomposed to the right level of abstraction to enable verification and validation of the design against the Requirements. The Requirements are described in a requirements specification (see 7 on Requirements), the product is described in a Product Tree (see 9.4 on the Product Tree). To create a Product Design that meets the design challenge, several options may be explored. These options can be modelled as variations within one engineering model, to enable reuse of the overall design. For each option, a Product Tree can be visualised. To build up the design, components are used that are called elements. Elements are defined in the Elements Definition Browser, where we can specify their characteristics using parameters (see 9.2 on Elements). The elements can be used to compose larger elements (Systems consisting of sub-systems, consisting of equipment's, etc.) These levels of abstraction are indicated using categories. Between the elements in the model we can define relations. For instance, we can relate components to requirements to indicate they satisfy a certain requirement. We can also relate components that are interfaced.

2.2 CDP4-COMET Architecture

CDP4-COMET is the main MBSE tool to support multidisciplinary teams to perform Concurrent Design in cooperation with and participation of the customer. It provides all the required integrated design and modelling capabilities to support the multidisciplinary team in creating a joint system model for the full life cycle of the system.

CDP4-COMET provides 7 major software components (for these components, see Starion Group · GitHub):

- CDP4-COMET IME (Integrated Modelling Environment): A windows desktop application that allows a user to interact with the CDP4-COMET information. (see section 8 on logging into CDP4-COMET)
- CDP4-COMET Excel Add-in: A Microsoft Excel integration that allows a user to interact with a subset of the CDP4-COMET information from within excel. The Excel Add-in is distributed together with the CDP4-COMET-IME (see 11 on the Excel plug-in)
- CDP4-COMET Web: a web application that provides various dashboards that provide insight into the status of a model including a simplified 3D geometry.
- CDP4-COMET Server: the application server and database that stores all the information and provides a JSON REST API; CDP4-COMET-EE includes LDAP authentication.
- CDP4-COMET C# SDK: A C# software development toolkit that allows developers to build CDP4-COMET integrations using the C# language
- CDP4-COMET Java SDK: A Java software development toolkit that allows developers to build CDP4-COMET integrations using the Java language
- CDP4-COMET Typescript SDK: A Typescript software development toolkit that allows developers to build CDP4-COMET integrations using the Typescript language.

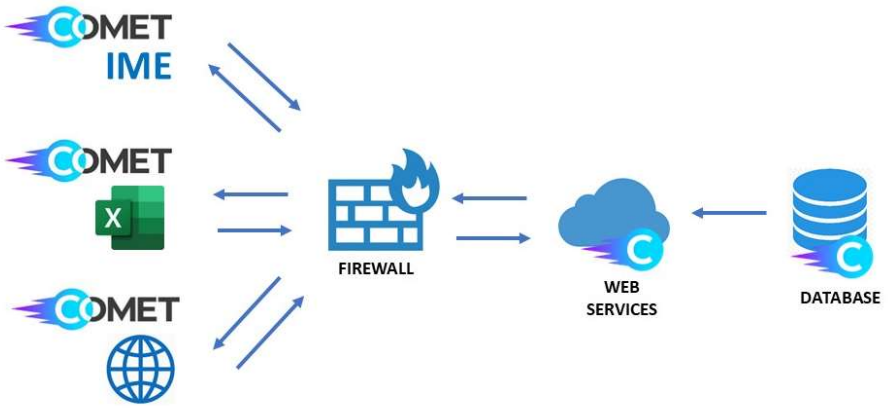


Figure 1: CDP4-COMET Architecture

A typical user will use the CDP4-COMET app to contribute to a model, while turning to the EXCEL plug-in workbook to make calculations based on the data in the model. The web applications can be used by users that do not want to install the desktop application.

2.3 Standards

CDP4-COMET is an implementation of ECSS-E-TM-10-25A Annex A, Annex B and Annex C.

ECSS-E-TM-10-25A is a technical memorandum, under the ECSS-E-10 System Engineering in the engineering branch of ECSS series of documents, defines the recommendations for model-based data exchange for the early phases of engineering design; it is the foundation of CDP4-COMET. This standard describes a meta-model of a system, composed of elements, that are characterized by parameters. Elements implement functions to satisfy requirements. Using this standard, the system design will likely encompass higher consistency, and enables exchange with other modelling methods.

ECSS-E-TM-10-25 is the System Engineering - Engineering Design Model Data Exchange (CDF) Technical Memorandum. is developed by the ESA with contributions from Industry and Academia.

“This Technical Memorandum facilitates and promotes common data definitions and exchange among partner Agencies, European space industry and institutes, which are interested to collaborate on Concurrent Design, sharing analysis and design outputs and related reviews. This comprises a system decomposition up to equipment level and related standard lists of parameters and disciplines. Further it provides the starting point of the space system life cycle defining the parameter sets required to cover all project phases, although the present Technical Memorandum only addresses Phases 0, Phase A and Phase B1”

This draft open standard is intended to provide the basis for:

- “Creating interoperable Concurrent Design centres across the European space community.

- Allowing semantically consistent data exchange between Concurrent Design centres.
- Enabling and supporting joint real-time collaborative design activities involving multiple Concurrent Design centres."

See: ECSS-E-TM-10-25A – Engineering design model data exchange – CDF (20 October 2010) | European Cooperation for Space Standardization

The underlying Data Model is an implementation of the Concurrent Design concepts that are modelled and described in ECSS-E-TM-10-25, reflecting those concepts that are agreed upon and accepted by the Concurrent Design Community. In this way, the CDP4-COMET complies with and supports interoperable Concurrent Design centres. CDP4-COMET allows to connect to other data sources that are compliant with this draft open standard.

2.4 Data Exchanges

RHEA develops so-called Domain Specific Tool integrations with the CDP4-COMET ecosystem. Domain Specific Tools are tools (software applications) used by the various engineering domains that make up a Concurrent Design Team and benefit from an automated link/data exchange with the CDP4-COMET server. The following tool integrations have been developed:

- Capella
- MagicDraw – SysML
- Enterprise Architect – SysML
- STEP-TAS (for integration with ESATAN-TMS)
- STEP-AP244 – STEP 3D CAD (for integration with 3D modelling environments)
- Catia v5
- Ecosim Pro
- Matlab
- ASTOS
- ReqIF

These tool integrations are available in the public domain, except for the ASTOS integration which is only available from ASTOS solutions. The integrations are open source (with the exception of ASTOS) and all available on GitHub. The integration does not include the tools themselves; these will have to be acquired from the respective suppliers.

Figure 2 provides an overview of the typical domains of expertise for Space Mission Design and a non-exhaustive list of engineering tools used by those domains. The figure demonstrates that CDP4-COMET, as implementation of ECSS-E-TM-10-25, can function as a model-based engineering hub. If your design tools are not yet part of this ecosystem, it is possible to exchange data manually via Excel or to create a new plug-in with our SDK kit. Of course RHEA can build custom integrations on request.

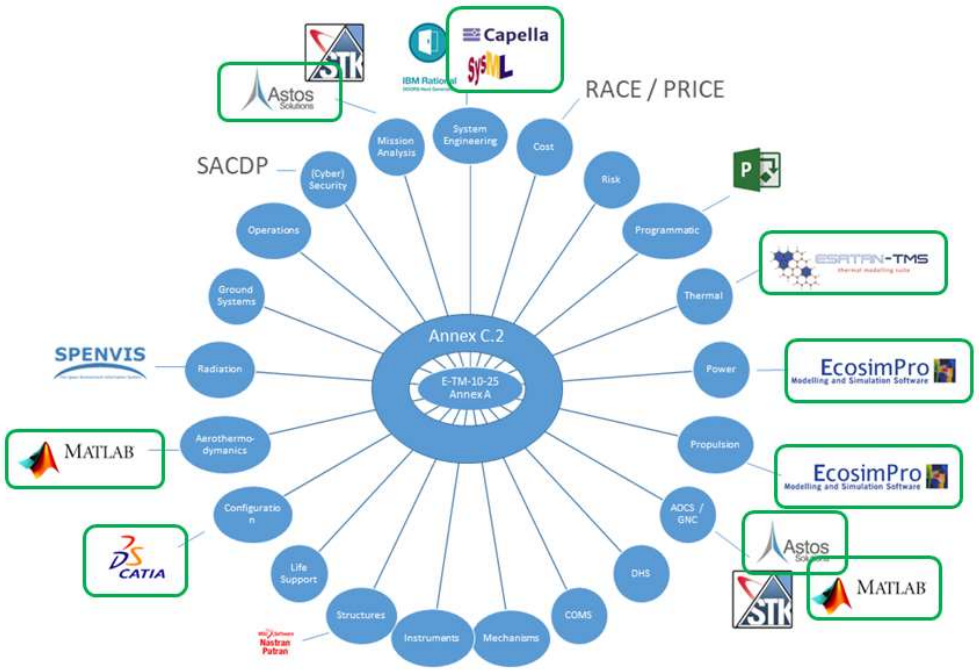


Figure 2: Digital Engineering Hub

2.5 Reusability

In CDP4-COMET models are built in a modular fashion. This enables us to use the same model to create different options, that vary in aspects, but are often the same in structure. The Model, including its options can also be 'frozen' in different iterations of the design, enabling the team to create a history of the design as well as roll-back options. Models can also be saved as template models to enable re-use. Besides the re-use of entire models, the individual Building Blocks in the model can also be re-used.

The design is built up from building blocks that are called Element Definitions. Each element is defined with a name and can be described in text and by properties that are called parameters.

Elements are first created in the Elements Definition Browser (see 9.2 on Element Definitions). This can be seen as a library containing an overview of all elements that can be used in the design. Elements that are specified in the element definitions browser, can be used in the design of an option. Elements are re-usable. They can be specified in much detail and re-use it in different design options, and even in different models. For this purpose, elements can also be defined in a catalogue model, from which they can be copied in study models. Such element can also encompass sub-elements enabling the re-use of entire sub-systems. These items can be copied between engineering models without values, keeping only a specific structure, or with values, e.g. choosing specific equipment from domain or company specific equipment catalogues. When an element is used in an option, it is copied as an 'element usage' - its properties such as name, shortname and its ownership can be changed, without

affecting the original building block (the element definition). The underlying parameters with their values are linked to that of the element definition however. There is a mechanism to change parameter values for specific instances, see the description of using parameter values and the concept of overrides in section 9.6.

2.6 Users, Roles, Domains

Users of CDP4-COMET are stakeholders in the Concurrent Design study. Each stakeholder fulfils a role in the study. Most participants represent a domain of expertise. Other roles are the customer, the CD System Engineer, CD Team Lead, CD System Analyst and CD Modeller (see 1.6 on CD Team roles). Domain experts are representing a Domain of Expertise, for which they manage the data in the engineering model, such as Structural Engineering or Communication. Besides their domain assignment, users also get a team role. The team roles, describing what is expected of them within the CD process. Following ECSS-E-TM-10-25 this is arranged using the notions of Person Role and Participant Role which distinguishes roles on CDP4-COMET server level and on Engineering model level (see 5.5 on Roles).

A study can require many different domains, some up to 20 or more. Ideally one person represents one domain of expertise to ensure ownership and encourage responsibility, but it is possible to assign the same domain to more than one user (see 5.1 on Domains). Note however, that these users can then unintentionally overwrite each other's changes for that domain, if they are connected simultaneously. For practical reasons the supporting roles (e.g. System Analyst) often also get the domain 'System Engineering' assigned to them. Each model has its own team. Based on the CDP4-COMET user log-in name and password, the combinations of model iterations and domains are presented to the user for only those engineering models he/she participates in as a team member. For each of these, access rights are set by the combination of person and participant role.

Domains, Participants and Roles are created and accessible at the CDP4-COMET Directory level (see 5 on the Site Directory), they can be assigned to different models. The roles and rights of each user can be specific for specific models. Teams can also be created in the Team Composition Browser.

2.7 Models and versions of models

While the team typically works in one model, the model can represent the system in many ways.

The model can be built from scratch, but we can also base the model on a template and fill it using a catalogue. Models can be stored as scratch models, template models or a model catalogue, besides regular study models. We call this the Model Kind (see 4.2 on Model Kinds).

A model is captured in different phases. In the preparation phase, the model is only accessible to a small team preparing the study, in the design session phase it is open to all team members. In the reporting phase the model is typically used to retrieve data for reports and after closure of the study, it is classified as completed study (see 4.2 on Study Phases).

Within a model we can model different options. When we want to explore variations in the product design, we capture these as options in the design. These options can be displayed as variations based on the same model. Elements can be excluded or included in specific Options (see 10.7 on Options), leading to these design variations. The elements can also vary between the different options by having option-based parameter values (i.e. possibly different values for the various options).

A model can also be saved as an iteration. The design process typically involves several Iterations, in which the design is further detailed and analysed. To capture these Iterations, they can be saved and frozen. In this way the team has a roll-back possibility but can also evaluate how the design has evolved (see 10.9 on Iterations).

Finally, the design of a system may differ in different States. For instance, an airplane behaves differently during take-off, flight, landing and taxiing. To model the behaviour of the system in these States, we can specify those aspect that vary across States separately (see 10.5 on Finite States).

Setting up all these different variations in the model is the task of the System Engineering. guidance on how to model each variation can be found in this manual. It is important to teach the team the language to indicate each of these model variations.

2.8 Data Sources, Reference data, documents.

The CDP4-COMET app allows to connect to multiple data sources at the same time. As long as the user has the appropriate access rights on various data sources, in principle these can be opened simultaneously. This extends to opening lower-level concepts, e.g. opening multiple engineering models from multiple data sources (servers), opening multiple reference data libraries, or opening multiple iterations for any engineering model. This allows the user to e.g. compare different engineering models both within one data source as well as on different data sources.

In a Concurrent Design activity design parameters are exchanged between the active domains that are together developing an engineering model. These parameters are often (physical) parameters that describe or use quantities that have scales and units associated with them (see 6.2 on Parameters). The parameters are created based on a definition of a Parameter Type. Other concepts are implemented as well to organize and structure the design data in an engineering model. Reference data is that kind of data that needs to be defined only once and can be reused for different kinds of Engineering Models. Reference data is stored in Reference Data Libraries (RDL's) (see 6.1 on RDL's).

The following concepts are managed in Reference Data Libraries (RDLs): Parameter Types (Measurement Units, Measurement Scales, Parameter Types), Categorization (Categories, Rules) and References (Unit Prefixes, Constants, Glossaryes, File Types, Reference Sources).

According to the definitions in ECCS-E-TM-10-25, RDLs in a connected data source can be managed at different levels, typically the following 3 levels:

- A top-level Site RDL: Provides generally used concepts such as generic Parameter Types, SI Quantity Kinds, Measurement Scales and Measurement Units. These are concepts that are used in almost any Concurrent Design Engineering Model independent of the application domain.

- A second-level Site RDL: Holds concepts like Parameter Types, Measurement Scales, Measurement Units and Glossaries for a specific application domain or "family of studies", for example: Space Mission Development, System of Systems Development, Early Concept Development, Payload Instrument Development, Launch Vehicle Development.
- A third-level Model RDL: Contains concepts similar to the second level, but now those that are particular to a specific Engineering Model.

The top-level Site RDL of a data source is available from the installation of a CDP4-COMET server as ECSS-E-TM-10-25 compliant data source. All other RDLs, both second-level Site RDLs as well as Model RDLs usually have a relationship to this top-level RDL. It is also possible to create any number of additional second-level Site RDLs, e.g. to create RDLs for specific types of projects. These are often created as branches from the top-level RDL, although they can also be created from scratch; see below for a description of dependencies between RDLs and their use in engineering models. The creation of new Site RDLs based on an existing one leads to a dependency by inheritance, where e.g. copies between modelling items such as element definitions between different engineering models can only be performed for items that are available in the chain of RDLs. The feature of creating separate Site RDLs should be used with care. It is advisable to keep the definitions at the top level (Generic RDL) to allow easier reuse across engineering models on a CDP4-COMET server.

Next to the Site RDLs, following ECSS-E-TM-10-25 as the third level RDLs given above, for each engineering model, a Model RDL is being created. Adaptations and extensions can be made to the RDL of the specific engineering model without the risk of compromising any other engineering models; see the section below however for dependencies between RDLs and engineering models.

The Reference Data of Measurement Units, Measurement Scales and Parameter Types in the RDL as well as the other concepts, such as categories and the reference sources, are defined at the level of the CDP4-COMET server or can be managed at the level of a specific engineering model. All new items that are created in a Site RDL can be viewed and used in other engineering models, provided that they are within the chain of RDLs. This means that the RDL should be carefully managed. In managing the RDL, care has to be taken to create combinations of measurement units, measurement scales and parameter types that lead to meaningful definitions for parameters that are needed in the design work in the engineering models, and trying to avoid incorrect, ambiguous or duplicate definitions. Editing entries in the RDL can have a large impact on not only the current activity, but also on all the other activities using it. It is advisable to avoid negatively affecting other CD activities and engineering model setups inadvertently by careful management of the concepts in the RDL, such as measurement units, measurement scales and parameter types, categories etc. by a restricted group of expert users.

3. CDP4-COMETAPPLICATION GENERAL FUNCTIONS

3.1 General Navigation

The CDP4-COMET desktop app consists of a shell main window, which we call the Integrated Modelling Environment (IME). In this main window, the appearance of the various Tabs, Browsers and Dialogs are given by the available plugins. The assignment of users to a specific Person Role will determine what rights a user with that specific role has in the CDP4-COMET app and thus what he will

be able to see and do (see 5.5 on Roles). Each user can however adapt for himself how to arrange the views in the CDP4-COMET app. The arrangement of views is fully controllable by the user by a docking mechanism described below. It is advisable to set up the layout of the model that will allow the usage of combined views in interactions, e.g. having the Parameter Types and the Element Definitions Browser on separately docking areas to allow drag-and-drop actions, or having the Element Definitions Browser and the Publication Browser separately docked to allow inspection between these two views. This item provides a generic description on how to open and use these tabs and views in the CDP4-COMET app User Interface.

3.2 Layout, ribbons, tabs

On the top of the screen, a series of **tabs** is visible, like in office applications. The tabs start with Home, View, Directory, Reference Data, Requirements, Model, and Scripting.

In the Home tab we find buttons to access a server and to open a model as well as some general information and settings (see 8 on functions at the Home tab). In the View tab some specific features to customize views are available. In the Directory tab you find the browsers to set up new models and to assign teams, users, roles, domains, etc. (see 5 on the Directory tab) The Reference data is where you find all reference information such as for instance available parameters, categories, a glossary and files (see 6 on the RDL). The Requirements tab leads to the requirements specification and related functions (see 7 on Requirements) and the Model tab is where we find the elements definition and product three as well as the functions to structure the model (see 9, 10 on Model functions). Finally, the scripting tab offers a scripting interface (see 12 on Scripting).

These tabs each reveal a **ribbon** with **buttons**. When clicking a button, one of the browsers opens, in which we find part of the model, directory or reference databases. For most buttons we can open multiple iterations, options or multiple models. When you click the arrow below the button you can choose among these models or model versions, each opening a separate browser. When we open more than one of these browsers, we can arrange them on the screen. When you have multiple screens, you can arrange them on these screens; the separate browsers can be undocked from the general IME application as well. This allows us to use information from one browser to add elements to another browser using drag and drop functions.

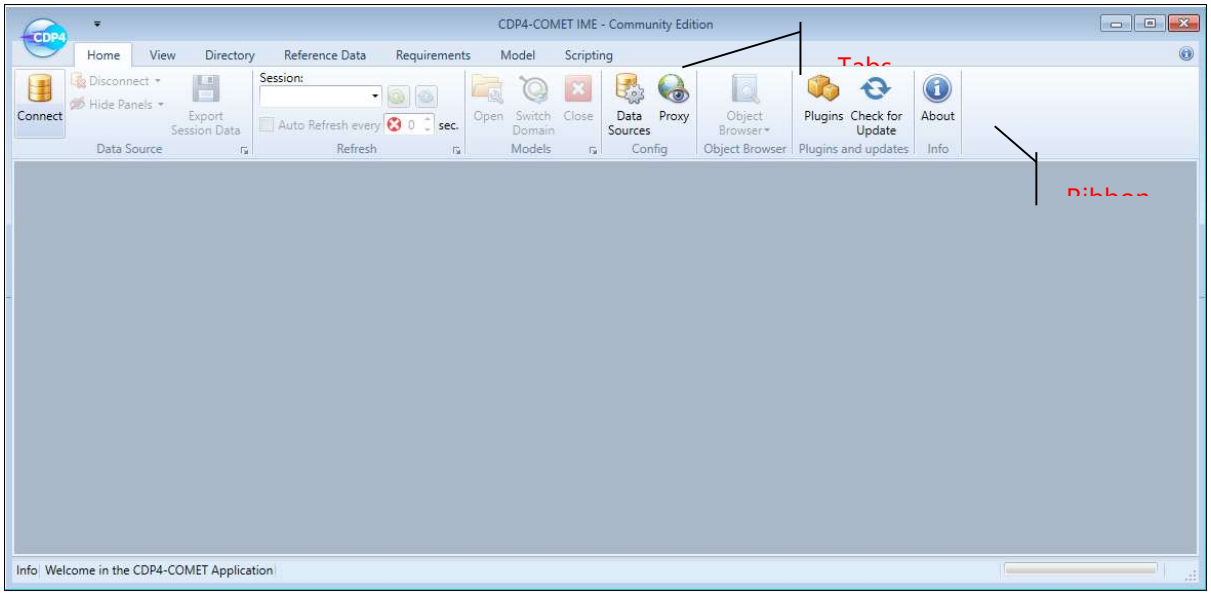


Figure 3: IME with Tabs and Ribbons

These browsers can be moved and docked to the left, top, bottom or right side of the interface by using the separate placing icons or by the cross-icon. The browsers can be moved around in the main interface as a floating window by selecting a browser, and then using drag-and-drop on the required place. It is usually not convenient to actually work with too many views as separate floating windows; it is advisable to have a view as a floating window only for moving the view to the position preferred by a user or placing a specific browser on a second screen.

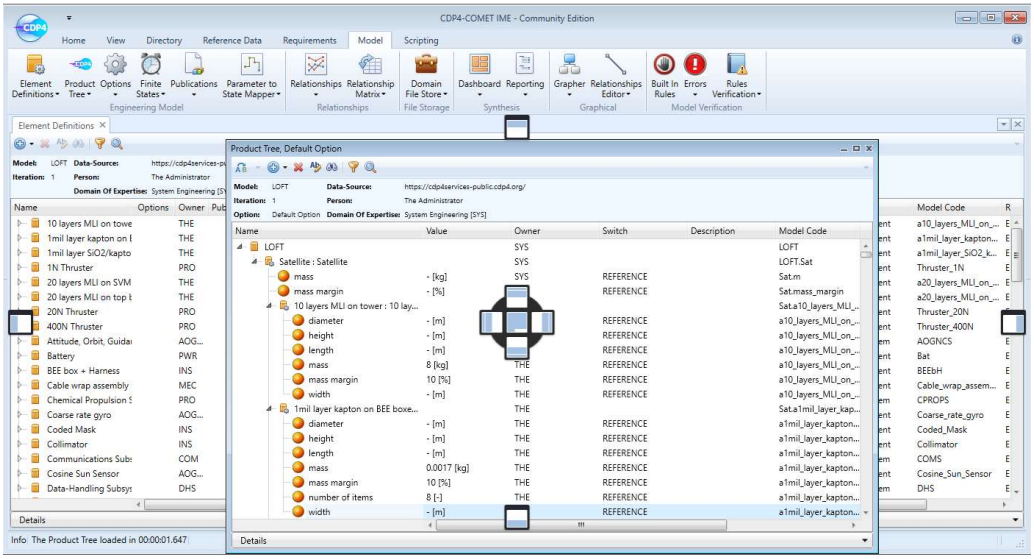


Figure 4: Organizing Browsers on the Screen

Multiple browsers can be placed on top of each other; these will then appear as multiple tabs within a group. When multiple browsers are tabbed, picking up a browser by the header area will allow the user to move it to a new docking location (drag-and-drop to a new preferred location). Browsers can also be arranged by right clicking the tab and selecting 'new vertical tab group' or 'new horizontal tab group' the browsers will then be arranged accordingly on the screen.

Browsers can be placed visibly all the time, or can be hidden by pinning them to the side using the Auto Hide/ Unhide icon on the top right of the browser. Hidden browsers will show up as tabs at the side of the main interface and will fold out when hovering over the tab.

3.3 Tabs and Browsers

These tabs have menu items that are described in their respective dedicated topics. The actual information presented in the Browser will also depend on the specific setup. Most browsers have a tabular lay-out with the rows representing a three structure in which we can drill down in Model Elements in more detail. In the table, most browsers in any case have their content presented with a Name and Short Name column. When concepts cannot be deleted, because they (could be) used in other Iterations or models, you can deprecate them to hide them from the view. In the view tab there is an option to see deprecated items, which also enables you to un-deprecate them if required.

Each Browser has a couple of generic Navigation options.


Add, object depending on selection


Delete or Deprecate


Edit (CTL+E)/ Change/ Set


Filter


Inspect (CTL +I)


Search


Favourites


Highlight Element Usages (right mouse click)


Copy Element Definition (right mouse click)


Export


Check / Open


Toggle Element Long vs Short Name

Some of these functions are both available as buttons on the top of the browser as well as through right-clicking, some are only available when right mouse clicking an object in the Browser. Not all functions are available through all browsers.

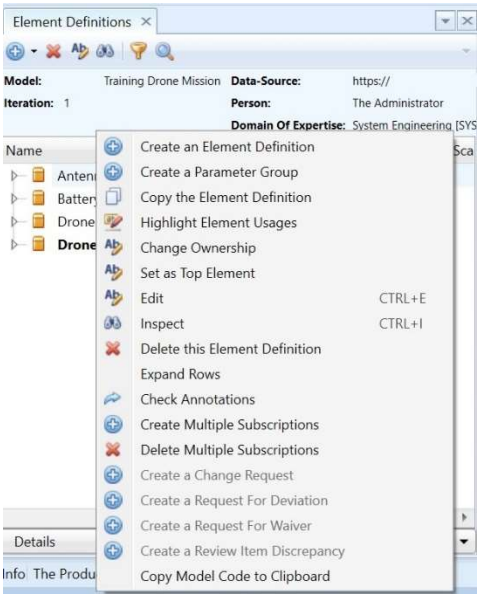


Figure 5: Right Mouse Click Functions

Below these functions the header area of the browser displays the general information about the Data source, the user (username and domain), and if relevant the model with iteration number it is showing. This helps to identify different (versions of) models.

At the bottom of the Browser, you can find a panel that can be opened by the small arrow at the bottom right of the browser. This will reveal details of the selected item in the Browser. In the details you can also find the Model Code of items (for the applicable browsers). The Model Code is generated by CDP4-COMET to refer to an item, for instance in Excel. The short name is used in the Model Code, making it easy to use.

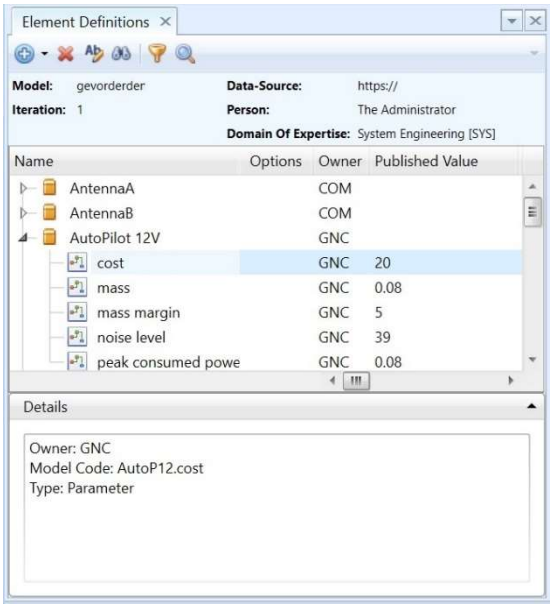


Figure 6: Browser Layout

Note that adding a lower-level item such as a Requirement that is added to a Requirements Specification requires to select the high-level item to which it is added first. The add - options change depending on what is selected in the Browser. Adding in this case can also be done using the right mouse click options.

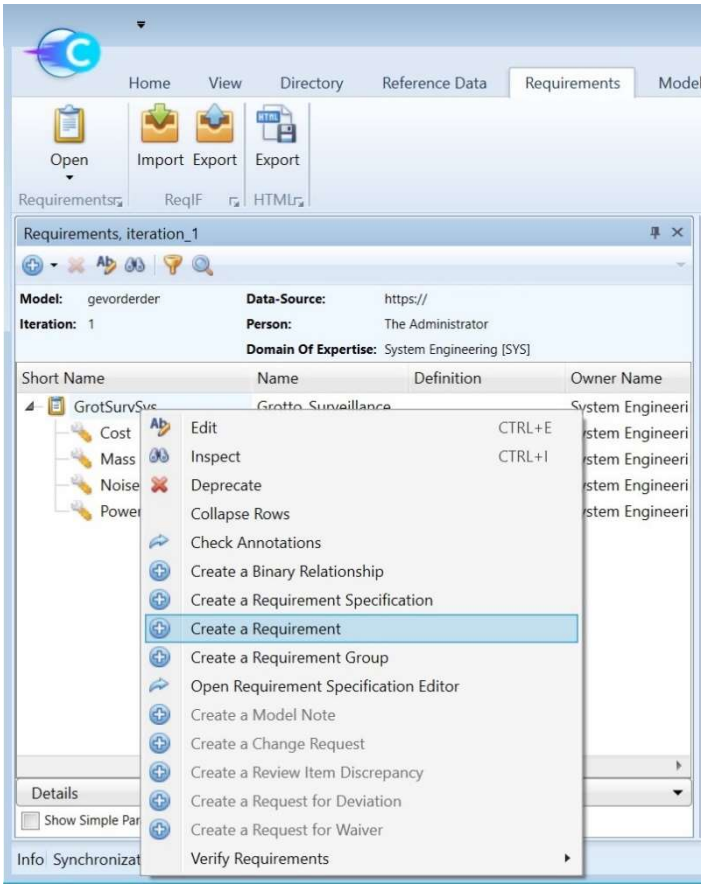


Figure 7: Create Functions

3.4 Dialogs

Typically, when we add or edit something in our model like an element definition, or a requirement, we open a Dialog, in which we can specify this new item. These views have to be closed before the user can continue to work in other views in the CDP4-COMET app.

Dialogs generally have the following tabs:

Basic: generally, Name and Short Name, and a check box for status indication of Deprecated (only for concepts that can't be deleted).

Advanced: generally, the Unique ID of the item and a Revision Number

Many Dialogs have content-specific fields on the Basic tab, and possibly even content-specific additional tabs, that are described in their respective dedicated topics. Additionally, many Dialogs have tabs for the generic concepts of Aliases, Definitions and Hyperlinks. These can be used to help navigate the model and explain concepts added to the model.

When filling in in a Dialog the red marked crosses in fields indicate that something still needs to be filled in or corrected before the item can be changed or created.

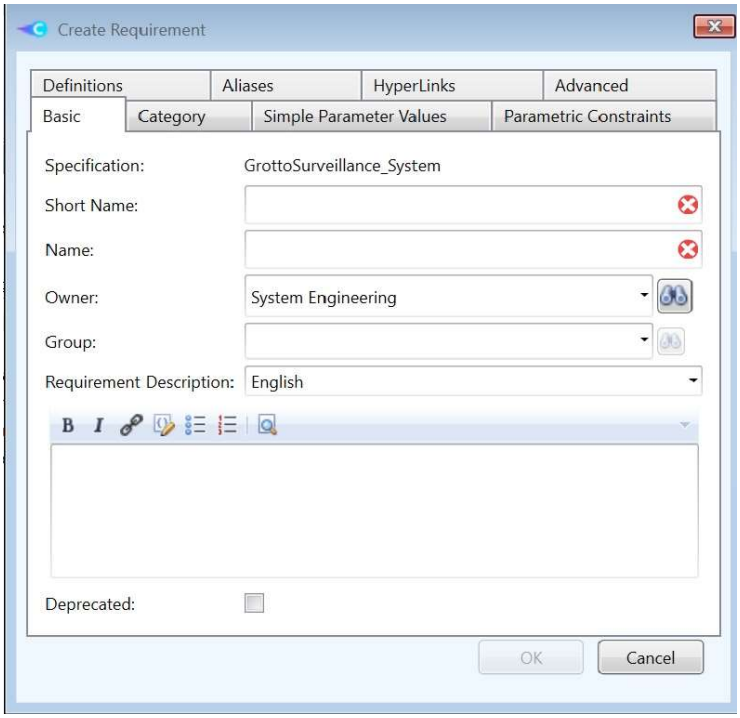


Figure 8: Dialog example

3.5 Log Information

All actions that a user performs are stored in a log file on the CDP4-COMET server. These are listed as entries in the Log Information Browser that is available on the View tab. The log entries provide information on changes to the CDP4-COMET data model, which may be mostly useful for a CDP4-

COMET Database administrator to trace changes to the model, usually related to issues or exceptions. The Log view can be found on the View tab. The Log view provides read-only information. The Log Information Browser can be filtered to show information from a minimum level and higher (trace, debug, info, warning, error, fatal), giving the possibility to leave out specific level entries. This is done by selecting or deselecting the icons for the different levels. Other possible actions are clearing the log entries that are shown in this browser using the Clear Log Panel icon, and saving the entries using the Export Log to a file icon.

To inspect the entire contents of a Data Source you can browse the Object Browser on the Home tab. The Object Browser displays all the data according to the ECSS-E-TM-10-25A containment hierarchy.

3.6 Revision Numbers

In CDP4-COMET, all the concepts in the CDP4-COMET Data Model have a Unique ID. In the CDP4-COMET Back-End a full revision history is implemented for traceability and review purposes. This versioning is visible to the user by the Revision Number. For each change that is performed on a concept this revision number is incremented. The revision number for a concept can be inspected on the Advanced tab of the applicable dialogs.

3.7 Notifications and Feedback

For actions that are performed in CDP4-COMET, user feedback is given in the status bar in the bottom left corner of the CDP4-COMET IME main application window. This feedback mainly consists of very high-level feedback, e.g. that a particular view has been opened, or that a change has been written to the CDP4-COMET server, e.g. when adding parameters, changing values or the value switch, or changing ownership of parameters within an engineering model.

4. ENGINEERING MODEL

4.1 Creating an Engineering Model

To create a new Engineering Model, or to modify the users or status of a model, we go to the Directory tab. Open the Engineering Model browser by selecting the Models icon in the Directory tab. This will show all the available Engineering Models the user is allowed to see according to the permissions, depending on the setup of the Person Role and the Engineering Models to which the user is assigned.

Engineering models can be of the following Model Kind: Scratch Model, Study Model, Template Model, Model Catalogue.

Engineering model setups are given with a Name and the applicable Study Phase in the column under the header Description, with possible choices: Preparation Phase, Design Session Phase, Reporting Phase, Completed Study. More details on kinds and phases are described below.

Engineering models can be expanded using the triangular arrow icon to show the Participants, Iterations, Active Domains and Organisations, each in a separate folder. The Participants are given with the Name, Description and Role. In the description column, the Organization and Domain of Expertise are provided. In the folder of the Iterations, a listing is given of the available iterations. The Active

Domains folder lists the domains that have been selected to be involved in the model. Finally, the Organisations folder lists the Organisations involved.

To create a new Engineering Model, select the Create Engineering Model Setup icon or right mouse click to select Create an Engineering Model Setup.

On the Basic tab in the Dialog, provide the mandatory fields for Name and Short Name. A Model Kind and Study Phase will have to be provided as well. These are preselected as Study Model in Preparation Phase.

Furthermore, it is required to provide the Site RDL (see 2.8 on RDL's) . This will create a new engineering model with default settings and for which the content will be mostly empty, i.e. the model is not based on an existing engineering model with predefined sets of design data already available.

Alternatively, it is possible to create a new Engineering Model-based on an existing engineering model, either another Study Model, or a Template Model, see the description of Model Kinds. This is done by selecting the applicable engineering model in the field of Source Model. If a source model has been selected, the new engineering model will be automatically created in the Site RDL of this source model due to the required RDL dependencies.

The person that has created a new engineering model setup is automatically assigned to it as a Participant, in the Participant Role of Model Administrator. This person can then continue the steps for the development of an Engineering Model.

Further Participants can be assigned in a required Role in creating the CD Team, see the more detailed description there. It is also possible to transfer this by assigning another participant as the role of Model Administrator. Please note that an Engineering Model requires at least 1 Participant.

A new Engineering Model created without a source model automatically contains one Iteration, iteration_1, and no Options. It will also contain only one active Domain of Expertise, namely the default domain of the person that created the Engineering Model. When the model is created based on a source or template model, the initial content and available information in the new Engineering Model is determined by the chosen source model. The new model will be a full copy of the source model, including all the design information, such as the active domains, iterations, element definitions and element usages, subscriptions, values for parameters etc.

It is not possible to edit the source Model and the Site RDL. These are fixed attributes of the model.

Engineering Models can be deleted. Most items in CDP4-COMET are never completely deleted, but they are marked as Deprecated. **Engineering Models are an exception to this.** When an engineering model is deleted from the CDP4-COMET database, it will **actually be removed** from the data that is stored in the CDP4-COMET database from that point onwards.

4.2 Model Kinds and Study Phases

Engineering models can be of different Model Kinds. When creating an Engineering Model Setup, you need to indicate the Model Kind and the Study Phase (see 4.1 on setting up and Engineering Model). Model Kinds can be of the following types:

- STUDY MODEL

assertion that an engineering model (represented by an Engineering Model Setup and an Engineering Model) is the model of a concrete study

- TEMPLATE MODEL

assertion that an engineering model (represented by an Engineering Model Setup and an Engineering Model) is a template model intended to be used as the basis for new study models

Note: A template model may have only one iteration (represented by a combination of Iteration Setup and Iteration).

- MODEL CATALOGUE

assertion that an engineering model (represented by an Engineering Model Setup and an Engineering Model) is a catalogue of model components and/or patterns that can be re-used in the model of a study

Note: A catalogue may have only one iteration (represented by a combination of Iteration Setup and Iteration), and one Option.

- SCRATCH MODEL

assertion that an engineering model (represented by an Engineering Model Setup and an Engineering Model) is a scratch model to be used for preparation, training or experimentation purposes.

These Model Kinds are defined in ECSS-E-TM-10-25 and cannot be extended or edited, but only applied to an engineering model.

The Model Kind of an engineering model can be inspected or edited in the Engineering Model browser.

Engineering models can be in different Study Phases, according to the design process in a Concurrent Design Activity. Model Phases can be of the following types:

- PREPARATION PHASE

assertion that a model pertains to a study preparation phase

- DESIGN SESSION PHASE

assertion that a model pertains to a study design session phase

Note that an engineering model has to be in the design sessions phase to be able to create iterations.

- REPORTING PHASE

assertion that a model pertains to a study reporting phase

- COMPLETED STUDY

assertion that a model pertains to a completed study

A main reason for the Study Phase of engineering models is to provide an indication to the users of its status and by extension also the status of the design information contained in it. The Study Phase is usually loosely linked to the concept of the Model Kind of an Engineering Model.

The Study Phases are mainly applicable to an engineering model that is of the Model Kind Study Model. For this, an engineering model would start in the Preparation Phase to perform preliminary engineering tasks with a core team.

Typically, while performing the Concurrent Design Activity on this model, it will go through the consecutive phases, i.e. the Design Sessions Phase and the Reporting Phase (See 1.5 on the Concurrent Design Process). Note that an engineering model has to be in the design sessions phase to be able to create Iterations (See 10.9 on Iterations).

After the Concurrent Design Activity has ended, it can be set to Completed Study. This will indicate to all users that have access to this engineering model that it has gone through the full design process cycle. It can therefore be expected to contain a full model with identified options, element definitions and usages, defined states, parameters (and their subscriptions) with the corresponding values for which the publications and iterations provide an audit trail for their evolution. Such a Completed Study engineering model may well provide a basis for e.g. Template Models, provide new inputs to existing or newly created Catalogue Models, or copied to create a new Scratch or Study Model.

When creating or editing an engineering model setup, it is possible to provide all combinations of Model Kinds and Study Phase. The combination of Model Kinds and Study Phase is intended to provide an accurate indication of the status of an Engineering Model to other users.

4.3 Creating a CD Team

Once a model is set up, we need to assign participants to the model with a specific domain to ensure all experts can access and modify the model. Participants can have different access rights which are described in the Roles section in the site directory (see 5.5 on Roles). To become a participant, a person first has to be registered in the Persons Browser to obtain a CDP4-COMET username and password (see 5.4 on Persons; a person role is needed as well to allow the user to actually connect). Then they can be added to the model in the Engineering Model Browser (4.1 on setting up an Engineering Model). The Team Composition Browser gives an overview of the team members that perform the Concurrent Design Activity.

In the Engineering Model Browser, expand the Participants folder to inspect the existing participants. Use the create icon or right mouse click to Create a Participant. You can now use the drop-down menu to select and add an existing person to add to the model as participant. You can select their participant role and the default domain for the selected person will be automatically preselected as Domain of Expertise if this domain is set as an Active Domain for the engineering model. It is always possible to change this default domain assignment by ticking or unticking the check boxes for the available domains as required. A person can participate in multiple domains and multiple users can participate in a single domain. If the default domain is not provided for the person, or if this is not an Active Domain in the engineering model, of course it cannot be preselected, and the user has to provide the correct domain assignment for the new Participant.

Using the Team Composition Browser, it is possible to get an overview of the background details of all the persons that are assigned to an engineering model as participants, i.e. representing the Concurrent Design Team for a specific CD Activity. After connecting to a data source, open the Directory ribbon tab and select the Team Composition icon. The list of engineering models with their Names and Short Names is presented in a modal dialog. Select the engineering model for which you want to see the team composition and click Ok.

This will open the Team Composition Browser, consisting of a header bar, and a tab for each user that is assigned to the Engineering Model as a Participant. The tab for each participant in the team can be collapsed to only show the name or expanded to show more details by clicking the Expand or Collapse icons. These are the Domain, Telephone Number (default), Email Address (default), Person Role, Organization, Organizational Unit, and a check box with the setting for Active; see the descriptions of persons (section 5.4) for more detailed descriptions of these fields.

The Team Composition browser layout can be easily adapted by the user. On the top right-hand side of the browser, a list is available of the attributes that are given as details on the tab for each team member; click on the List icon to show or hide these. Any of these attributes can be dragged onto the header bar to sort the team members' tabs accordingly, where these will appear as a group under a vertical grouping tab for the chosen attribute(s). Similar as each team member tab, use the Expand or Collapse icons or alternatively double-click this vertical grouping tab to collapse or expand the team members that belong to it. As a typical example, the team members are sorted according to their participant role, showing the model administrators and the domain experts as easily identifiable groups. Another possible ordering could be based on e.g. the domain or organizations. Each layout can be easily adapted by adding or removing attributes from the header bar. Please note that this layout is not saved; the same or an alternative representation can however easily be applied again when reopening the Team Composition browser for any engineering model.

5. SITE DIRECTORY

5.1 RDL

On the Directory tab, an overview is given of the site-level Reference Data Libraries available on the data source the user is connected to. For a description of the site-level as well as other level RDLs, see section 2.8. In the browser showing the Site RDLs, it is possible to create, edit or deprecate these. Note that it is possible to create new site RDLs based on existing ones by selecting a required RDL, to have all the items in that site-level RDL (already existing, but also items that are added to this required RDL afterwards as extensions) will be available in this new site-level RDL. Managing the content of an RDL is described in chapter 6.

5.2 Manage Domains

The Concurrent Design method embedded in CDP4-COMET requires the system under consideration to be designed by stakeholders each representing one or more domains of expertise. Concurrent Design is a multidisciplinary approach, and it is therefore important that experts take ownership of those aspects of the design that relate to their domain of expertise. CDP4-COMET captures interactions between disciplines within a CD Activity through ownership of requirements, of modelling items (element definitions and element usages) and of parameters, with other domains using these as inputs, and by using options and iterations, providing an audit trail to the design process.

Each domain has a workspace in CDP4-COMET with an associated CDP4-COMET workbook to perform domain specific calculations. A user needs to be assigned to an engineering model as a participant, representing a specific domain see the description on how to create a team (section 4.3). Items added to a model are listed as Owned by the active domain of expertise of the contributor. Only the Owner and roles with administrator rights can edit items and parameters (see section 5.5).

This section describes how to manage domains at the level of the Directory. These domains are thus available for all Engineering models at a specific CDP4-COMET server. In the Engineering model setup you can select the Active Domains for a specific model.

To open up the Domains of Expertise Browser, on the Directory tab select the Domains icon. Note that if you have different CDP4-COMET servers open, you need to select the Directory of the server you want to edit. The Domains of Expertise browser shows a list of all the Domains of Expertise that are available in the selected Site Directory. From this panel domains of expertise can be added, edited, inspected or deprecated. The list will contain a list of typical domains of expertise in the space domain with their corresponding acronyms, consisting of e.g.:

- Cost Engineering - COS
- Guidance, Navigation and Control -GNC
- Power - PWR
- System Engineering – SYS

These domains of expertise are available to be used in all engineering models within that data source in CDP4-COMET. When a user is created, as a Person, it is possible to assign a default domain from this list of all possible Domains of Expertise.

The actual domains of expertise active in a CDP4-COMET model also have to be chosen from this list. If a domain of expertise is needed in a particular Activity that is not yet available in this list, it has to be

created first at the level of the Site Directory. New domains should be an expression of a domain of expertise as much as possible and not as a person or role.

5.3 Manage Organisations

In Concurrent Design, a multidisciplinary team works together on a design problem in a Concurrent Design Activity. The team members will often belong to different organizations, coming from e.g. governmental organizations, industry, research institutes, and/or academia. These organizations can be indicated to provide more context to these team members.

Users of CDP4-COMET are called Persons in E-TM-10-25 (see 5.4 on Persons). To get an overview of the background of all the persons that exist on a data source, it is possible to manage Organizations and show the persons that belong to it. After connecting to a Data Source, open the Directory ribbon tab and select the *Organizations* icon to manage organizations. This will open the Organizations browser that contains a list of all the organizations available in the specified data source. These are given with their *Name* and *Short Name*. When managing Persons, it is possible to indicate that a Person belongs to an organization from this list. For each Organization, the persons that belong to it will then be given in the Organizations browser, with their *Name* and *Username*, as well as their *Person Role*, *Default Email* and *Default Phone*. In the browser Organisations can be added, edited, inspected or deprecated.

5.4 Manage Persons

Within the Concurrent Design Platform, a user account is used to identify a Person working with the application. A user account holds information such as a *Name*, *Username*, CDP4-COMET *Password* and contact details. The name and contact details are visible to other users.

The user account by itself does not yet enable a user to participate in a Concurrent Design activity. The account is linked to a specific engineering domain to allow for a workspace in the current activity option. To allow the user to actually work with CDP4-COMET a user Role is granted. The Role describes the granted functionalities and rights.

Administrators or Persons with a role that has the permission to modify Persons are able to access the Person Browser and create, edit or delete persons. It also allows viewing all persons in the Site Directory of the selected data source.

To get an overview of the Persons open the Person Browser by selecting the Persons icon on the Directory tab. This browser gives an overview of the existing persons with *First Name, Last Name, Organization* and *Description*.

By expanding a Person, the Engineering Models they are assigned to as Participants are listed, together with their Domain acronym(s) and Participant Role. Persons cannot be assigned as Participant to an Engineering Model without a specific Participant Role from this browser; for this, see 4.3 on how to create a team.

To create a new Person, select the Create Person icon or right mouse click to select Create a Person and provide the mandatory fields for *Username*, *Given Name* and a *Surname*. Note that these fields

are only enough to create person as a placeholder that will require further editing and are not enough for a person to be able to login to CDP4-COMET and perform any actions. For this it is in any case required that a Person Role is specified, see the details in the topic on Person Roles, and a Password is set. To set the Password, select the Edit icon behind the field for the Password and provide a valid Password and Password confirmation. A person should also be activated by ticking the Active check box. Unticking the Active check box will take away the access rights of the person, essentially locking the person out of the CDP4-COMET app and all models. We cannot delete persons, but they can be deprecated.

Furthermore it is possible to specify a Default Domain for a person (see 5.1 on domains); see the details on how to create a team for a description of how this default domain is used (see 4.3 on team composition). Other optional fields are Organizational Unit and Organization.

On the Emails, Telephone Numbers and Preferences tabs, it is possible to provide additional information with the standard functionality of Create, Edit, Inspect and Delete. For emails provide a *Type* and the *Email Address*, for telephone numbers provide a *Type* and *Number*, and for preferences provide a *Short Name* and *Value*.

5.5 Manage Roles

Role management is an important step setting up a new activity within CDP4-COMET. A role is a tool to customize the experience a user has while working with CDP4-COMET. A Role is a description of the rights and permissions a user has while using CDP4-COMET. A role can be finely tuned in the form of permissions for a role by setting the required access rights as defined in the data model of ECSS-E-TM-10-25 Annex A. This section gives a background description of the purpose of roles in CDP4-COMET.

Roles will need to be assigned to a specific user, providing him with a role combination that will allow access to CDP4-COMET and determine possible actions a user can perform. These can reflect a user's role in the actual organization, but this is not compulsory.

The roles exist at two levels:

- Person Roles, assigned to a user at CDP4-COMET server/ Site Directory level.
- Participant Roles, assigned to a user, applicable only within the context of a specific CDP4-COMET Engineering Model Iteration level.

This split of Person and Participant Roles allows to have more flexibility and better control of access rights, allowed actions, and accessibility. This control is important to guard the consistency of the overall model. For instance, when all participants can create categories, we cannot guarantee that items are categorized in such a way that the budgets for the model work, or if Parameters that are the same are defined in different ways, we cannot ensure that our budgets are complete. To guard such consistency, a participant role is defined with the rights to add domain specific data but not generic reference data and structuring data.

Note that setting up the Roles and permissions is not straightforward and requires background knowledge of the ECSS-E-TM-10-25 draft open standard due to the relations and required interactions between the various concepts. It is advisable to have the functions for Role management available for a limited group of users only. Once the setup of Roles has been properly done and fine-

tuned, it will not often be required to update or edit these. When you are a novice user, you can use the default roles listed in the roles browser. For an overview of preconfigured or typical roles in a Concurrent Design environment, see the description of typical roles. Please be aware that editing a Role will instantly affect all the Persons on the Data Source that have this role assigned. Changing the permissions should therefore be done with great care to avoid that users are unintentionally limited in their available functionality.

5.6 Creating Roles

To get an overview of the person roles open the Roles Browser by selecting the Roles icon on the Directory tab. This browser gives an overview of the existing Person Roles with Name and Short Name. By expanding a Person role, the assigned Classes with *Name* and *Short Name* are listed, together with the Permission that has been set for each Class. Persons with a role that has permission to modify the Person and/ or Participant Role are able to access the Roles Browser and create, edit or delete Person or Participant Roles.

To create a new Person Role, select the Create Person Role icon or right mouse click to select Create a Person Role. On the Basic tab, provide the mandatory fields. Furthermore, it is required to specify the Permissions on the available Classes. On the Permissions tab, all available classes are listed together with the access rights.

The available choices for the access rights on these classes are the following:

- **NONE**: No actions allowed. All data for the selected class is not visible to the user in CDP4-COMET.
- **MODIFY**: All actions allowed (inspect, create, edit and delete or deprecate).
- **READ**: Read action allowed. All data for the selected class is visible to the user in CDP4-COMET.
- **MODIFY_IF_OWNER**: Actions of inspect and create are always allowed. Edit and delete or deprecate is only allowed for data for the selected class that is owned by the participant. Allowed editing actions exclude changing ownership. (specific for participant roles)
- **MODIFY_IF_PARTICIPANT**: All actions allowed (inspect, create, edit and delete or deprecate), but only data for the selected class within the scope of engineering models to which a person is assigned as a participant is visible and can be modified. (specific for person roles). Applicable to classes that are related to the Iteration level, e.g. Iteration Setup and Engineering Model Setup.
- **MODIFY_OWN_PERSON**: Allowed to inspect and edit own person (e.g. edit password, phone numbers, emails etc.). (specific for person roles)
- **READ_IF_PARTICIPANT**: Read action allowed, but only data for the selected class within the scope of engineering models to which a person is assigned as a participant is visible. (specific for person roles)

On the Permissions tab, all available classes are listed. For details on these classes, please refer to their definition in ECSS-E-TM-10-25 Annex A (sections 4.x). These classes correspond to the modelling capabilities and items available in the various browsers in CDP4-COMET.

See: ECSS-E-TM-10-25A – Engineering design model data exchange – CDF (20 October 2010) | European Cooperation for Space Standardization

6. Reference Data Library: RDL

6.1 Managing Reference Data Library's

The reference data used in the CDP4-COMET model are managed in the Reference Data Library. Reference data are data to structure the model and to ensure and enhance its consistency. Consequently, it is critical that these data are not changed by everyone. Reference Library's should be manged with great care. To understand the conceptual and architectural aspects of RDL's please refer to the architecture section. Below you find the instruction to add, edit and deprecate different RDL data. Persons with a role that has the Permission modify Site Reference Data Library are able to access the Basic RDL to create, edit or deprecate RDL data. Adding, editing, inspecting and deprecating a parameter requires filling out a Dialog. Furthermore, it is required to provide the Container RDL. **It is important to understand the difference between the Container RDL's.** In the RDL reference data is stored either in the Generic RDL which is accessible across all models on the CDP4-COMET server, or in the model RDL, which is specific and accessible in the model only. Careful management of the RDLs is important, as reference data in the Generic RDL cannot be deleted, while using the model RDL makes it difficult to copy elements to other models. It is only possible to edit the Container RDL to promote an item from a model-specific RDL into a higher-level Site Directory RDL. To add an additional Site RDL, see section 5.1. More information about different RDL levels can be found in section 2.8. When an Engineering Model is opened, the data (content) from the required RDLs will be loaded automatically. If a user connects to a CDP4-COMET server without opening a corresponding model, the browsers in the RDL (Parameter Types, Categories etc.) will be empty. Should a user want to inspect or edit a Site RDL independently from a model, it can be opened directly from this Reference Data Library tab.

Do not create 'test parameters' or other fake RDL data, you will not be able to remove these. To test it is possible to use the test and demonstration server. A link and log-in data for the test server can be found here: CDP4-COMET - Starion, System engineering for space, defence & critical infrastructures (stariongroup.eu).

Please note that editing RDL data should be handled with extreme care, as this may have a large impact on all engineering models that use concepts that depend on it.

6.2 Parameters

Various parametric descriptions can be used within the CDP4-COMET model, not only physical quantities, such as e.g. a mass or power, but for example also boolean, composites, date, enumeration or free texts. As a basis, a large set of parameters, with associated measurement units and scales, are already defined as parameter types and available as a starting point in ECSS-E-TM-10-25 Annex B (informative) Space Engineering Reference Data Library. This set can be extended to define any required new parameter, scale or unit according to any one of the supported types.

For each parameter that needs to be created in CDP4-COMET, a Parameter Type definition will have to be available. It requires several steps involving definitions of units, scales and Parameter Types before a parameter can be created. This approach allows or requires a strict management of these definitions in the RDL.

Please read the general implication on managing RDL data before you proceed (see 6.1).

Open the Parameter Types Browser by selecting the Parameter Types on the Reference Data tab. Parameter Types are given with a *Short Name*, a *Name*, *Type*, *Default Scale*, and *Container RDL*.

Optionally, a Category can be applied to the new parameter type definition. To do this, provide the applicable category or categories by selecting these on the Categories tab. Given on this tab are only the categories that can be applied to the specific parameter type that is being created or edited. This is done by including the required parameter types in the list of permissible classes in managing the categories.

6.2.1 Parameter Types

Parameter Types can be of the following types (with their description from ECSS-E-TM-10-25 Annex A given below in italic)

The boolean, date, Date Time, text and Time Of Day parameter type, do not require any further specifications and can be used to create parameters.

- **Boolean**: representation of a boolean valued scalar parameter type, for derived parameters allowed values are TRUE or FALSE.
- **Date**: representation of a calendar date valued scalar parameter type, for derived parameters format value format is “YYYY–MM–DD”.
- **Date Time**: representation of a calendar date and time valued scalar parameter type, for derived parameters format value format is “YYYY–MM–DD-HH:MM:SS”.
- **Text**: representation of a character string valued scalar parameter type, for derived parameters allowed values alphanumerical characters, as well as all special characters.
- **Time Of Day**: representation of a time of day valued scalar parameter type, for derived parameters value format is “HH:MM:SS”.

For Quantity Kind, Array, Compound and Enumeration further actions are needed.

Quantity Kind Parameter Types

Quantity Kind: representation of a numerical scalar parameter type, for numerical parameters mostly representing physical quantities. various parameter definitions can be created as Parameter Types by the combinations of units, scales and number sets (Real, Rational, Natural, Integer). Allowed values and formats depend on the number set (Real, Rational, Natural, Integer), and on possible conditions on the value ranges of the applicable scale, or "–" if no value is supplied For the Quantity Kind Parameter Types, the measurement scales that can be chosen for the quantity have to be indicated on the Possible Scales tab by checking the tick-boxes of the relevant scales. These scales will be available as choices for the scale for parameters that are created based on this parameter type. These scales consist of a combination of a measurement unit and a number set, see the description of managing the measurement scales. After selecting the possible scales, a Default Scale can be selected in the additional field on the Basic tab. Optionally a Dimension Symbol can be supplied as well (expression of powers of the base quantity dimension symbols L, M, T, I, Θ, N and J).

Possible Quantity Kind parameter types are

Simple Quantity Kind: specialization of Quantity Kind that represents a kind of quantity that does not depend on any other Quantity Kind. These 7 SI base quantities are represented in the Parameter Types browser in bold font: time, length, mass, electric current, temperature, amount of substance, luminous intensity

Derived Quantity Kind: specialization of Quantity Kind that represents a kind of quantity that is defined as a product of powers of one or more other kinds of quantity For a Derived Quantity Kind, on the Factors tab, the Quantity Kinds with their corresponding Exponents have to be provided that together give the required correct definition.

Specialized Quantity Kind: specialization of Quantity Kind that represents a kind of quantity that is a specialization of another kind of quantity.

Example: The kinds of quantity "distance", "width", "depth", "radius" and "wavelength" can all be specified as specializations of the "length" Simple Quantity Kind.

For a Specialized Quantity Kind, in the field General provide the existing Quantity Kind for which this parameter type is a specialization.

Array Parameter Type

Array: specialization of Compound Parameter Type that specifies a one-dimensional or multi-dimensional array parameter type with elements (components) that are typed by other Scalar Parameter Types. Multidimensional parameters can be defined in CDP4-COMET as Array Parameter Types. This can be used to define vectors (e.g. position vectors) and matrices (or tensors respectively, e.g. inertia tensor) or any required multidimensional array.

To do this, provide the required dimensions of the array. Then select the component parameter types for each of the components that together form the required array parameter type. This uses the definition of the parameter type as component. The Short Name is generated for the component based on the dimension. Most Parameter Types can be used as a component of the Array Parameter Type, with the exception of Array and Compound Parameter Types.

Compound Parameter Type

Compound: representation of a non-scalar compound parameter type that is composed of one or more other (component) ParameterTypes allowed values dependent on definitions of parameter types used as components. The Compound parameter type is essentially a user-convenient grouping of a predefined set of other parameter types. For this type, the components have to be defined. To do this, select the component parameter types that together form the required compound parameter type. This uses the definition of the parameter type as component. It is possible to use a different Short Name for the component in the compound parameter type definition however, separately from the Short Name of the referenced Parameter Type. Most Parameter Types can be used as a component of the Compound Parameter Type, with the exception of Array and Compound Parameter Types.

Enumeration Parameter Type

Enumeration: representation of an enumeration valued Scalar Parameter Type with a user-defined list of text values (enumeration literals) to select from, for derived parameters allowed values are user defined in the definition of each specific enumeration Parameter Type can contain alphanumerical and special characters

For the Enumeration Parameter Type, the enumeration values have to be defined. To do this, on the Values tab, select the Create Value Definition icon and provide the Name, Short Name and Value. To delete an entry from the list, select the Delete Value Definition icon or right mouse click to select Delete. The values for an enumeration Parameter Type can be deleted, even if these are used. The reason is that contrary to other deletions such as units, deleting an item from the enumeration values list would not lead to possible corruption of any of the engineer models. Note that deleting an enumeration value will invalidate parameters in any of the engineering models that is using this value and may possibly invalidate corresponding user-defined validation or calculation steps. Parameters that have an invalid value, which is indicated by the value checking mechanism in CDP4-COMET can easily be fixed by the user by providing an allowed value. Dependent user-defined functionalities will have to be fixed by the user.

The values in the list on the Values tab will be appearing as a drop-down list in the manual value field of the enumeration parameter created based on the Parameter Type definition.

Sampled Function Parameter Type

A **Sampled Function** Parameter Type allows for a definition of a ParameterType for parameters whose value is a discretely sampled function where each sample consists of a given unique tuple of independent parameter values mapped to a tuple of dependent parameter values. See also a http://mathworld.wolfram.com/Map.html and https://mathworld.wolfram.com/Function.html for a formal mathematical definition.

The Sampled Function Parameter Type can be used to define among others the following: (1) a time series of analysis predictions or measurements, (2) a temperature dependent set of material properties, (3) a frequency spectrum displacement response of a structural item, (4) a mapping of key-value pairs, i.e. the same data structure as a dictionary or hash map in a programming language.

The unique property of this type of parameter type is the ability to create parameters which have unbounded matrices of values. You can supply an optional definition of a degree of interpolation to be used when computing a function value for a domain value that lies in between stored discretely sampled values.

The Independent Parameter Types are an ordered set of Parameter Types for the independent values of the sampled function, i.e. representing its domain (i.e. time). For these you can also supply an optional representation of a period in case of a cyclic function to be taken into account for interpolation. For example, the function could represent the incident albedo flux as a function of mission elapsed time for a spacecraft in a circular orbit. The Dependent parameter types are an ordered set of Parameter Types for the dependent values of the sampled function, i.e. representing its range (in this example the velocity sampled at a certain time). Both Independent and Dependent parts must contain at least 1 parameter type. If Quantity Kinds are used, you must also provide a scale.

It is important to note that after creating a Sampled Function Parameter Type you will no longer be able to modify the underlying parameter type sets as any parameters using it will cease to be valid.

6.3 Measurement Scales

The measurement units can be used in the definition of scales.

Please read the general implication on managing RDL data before you proceed (see 6.1).

Open the Measurement Scales Browser by selecting the Measurement Scales icon on the Reference Data tab. To create a new measurement scale, select the Create Measurement Scale icon or right mouse click to select Create a Scale for each of the available types. On the Basic tab, provide the mandatory fields. On the Basic tab, select the applicable Measurement Unit for which the corresponding Scale has to be created. For any type of Measurement Scale, a Number Set has to be selected as well.

Optionally, in defining a scale, it is also possible to define conditions for the allowed values. It is possible to supply a Maximum and/or Minimum Permissible Value, together with an explanation for this in the Positive and/or Negative Value Connotation. Tick the check box to include this minimum or maximum value.

On the Value Definitions tab, it is possible to add a definition for values that have a special significance for the measurement scale. To do this, select the Create Value Definition icon and provide the mandatory fields.

6.3.1 Measurement Scale Types

Measurement Scales can be of the following types:

Logarithmic scale: Defined by Factor, a Logarithmic Base and Exponent. Further defined by Reference Quantity Kind and an optional Reference Quantity Value

Example: The base 10 logarithmic measurement scale for "sound pressure level" in "decibel", with factor is "10", exponent is "2", reference Quantity Kind is "sound pressure" and reference Quantity Value is "20" on the "µPa" RatioScale. Source: ISO 80000-08

To specify a Logarithmic scale, on the Logarithm tab, provide the Logarithm Base, Exponent, Reference Quantity Kind and Reference Quantity Value

Cyclic ratio scale: Measurements bound to a range and repeat (degrees in a circle, time periods).

Example: Planar angle with modulus 360 degree, therefore an angle of 450 degree is the same as an angle of 90 degree, and -60 degree is the same as 300 degree.

To specify a Cyclic ratio scale, on the Basic tab, provide a value for the Modulus that represents the repeat cycle for the cyclic ratio scale.

Interval scale: Relative scale with arbitrary 0-point (e.g. Celsius-scale).

Example: The Celsius and Fahrenheit temperature scales are common examples of interval scales

Ordinal scale: Describes order, but not relative size or degree of difference between the items measured (e.g. rank order).

Examples: The Beaufort wind force scale with Scale Value Definitions (as defined by the World Meteorological Organization): 0 : "Calm"; 1 : "Light air"; 2: "Light breeze"; 3: "Gentle breeze"; 4: "Moderate breeze"; 5: "Fresh breeze"; 6: "Strong breeze"; 7: "Near gale"; 8: "Gale"; 9: "Strong gale"; 10: "Storm"; 11: "Violent storm"; 12: "Hurricane".

The NASA/ESA Technology Readiness Level (TRL) scale with Scale Value Definitions: 1 : "Basic principles observed and reported"; 2 : "Technology concept and/or application formulated"; 3 : "Analytical and experimental critical function and/or characteristic proof-of- concept"; 4 : "Component and/or breadboard validation in laboratory environment"; 5 : "Component and/or breadboard validation in relevant environment"; 6 : "System/subsystem model or prototype demonstration in a relevant environment (ground or space)"; 7 : "System prototype demonstration in a space environment"; 8 : "Actual system completed and 'flight qualified' through test and demonstration (ground or space)"; 9 : "Actual system 'flight proven' through successful mission operations".

To specify an ordinal scale, on the Value Definitions tab, select the Create Value Definition icon and provide the Name, Short Name and Value. The created value definitions will be presented as possible values for parameter types that are using this ordinal scale. These parameter types should then be created as a Quantity Kind parameter type.

Ratio scale: Interval scale with a true zero point (e.g. mass, length, Kelvin-scale temperature).

Example: The majority of the physical quantities are expressed on ratio scales, e.g.: length (metre, millimetre, kilometre, inch, foot, mile, lightyear); mass (kilogram, gram, nanogram, ton, pound); time (second, hour)

For these categories of scales, the scale indicates the step size between measurement units. A choice of number steps can be made by choosing a different number set. The number sets available are: Real, Rational, Natural and Integer

6.4 Measurement Units

A Measurement Unit is a definition of what you are measuring with a parameter. Units are expressions of the magnitude of associated physical quantities. The Measurement Units can be used in the definition of measurement scales.

Please read the general implication on managing RDL data before you proceed (see 6.1).

Open the Measurement Units Browser by selecting the Measurement Units icon on the Reference Data tab. Measurement Units are given with a *Short Name*, a *Name*, *Type*, and *Container RDL*. You can Add, Edit, Inspect and Deprecate a measurement unit using a Dialog.

To create a new Simple Unit, select the Create Measurement Unit icon or right mouse click to select Create a Simple Unit. On the Basic tab, provide the mandatory fields. Some Measurement Units require further specification as is indicated below per type of measurement unit.

6.4.1 Measurement Units Types

Measurement Units can be of the following types (with their description from ECSS-E-TM-10-25 Annex A):

Simple Unit: Specialization of measurement unit that represents a measurement unit that does not depend on any other measurement unit

Example: "gram", "metre"

Derived Unit: Specialization of Measurement Unit that is defined as a product of powers of one or more other measurement units

Example 1: The measurement unit "metre per second" for "velocity" is specified as the product of "metre" to the power one times "second" to the power minus one.

Example 2: The measurement unit "micrometre per metre" for the Derived Quantity Kind "linear expansion coefficient" is specified as the product of (Prefixed Unit) "micrometre" to the power one times (Simple Unit) "metre" to the power minus one.

To specify a Derived Unit, on the Unit Factors tab, select all applicable Measurement Units together with their Exponent for the combination of measurement units.

Linear Conversion Unit: Specialization of Conversion Based Unit that represents a measurement unit that is defined with respect to another reference measurement unit through a linear conversion relation with a conversion factor

Example: In the definition of the Linear Conversion Unit for "inch" with respect to the reference Unit "metre", the conversion Factor would be 254/10000, because 0.0254 metre = 1 inch.

To specify a Linear Conversion Unit, select the appropriate Reference Unit and Conversion Factor.

Prefixed Unit: Specialization of Conversion Based Unit that defines a measurement unit with a multiple or submultiple Unit Prefix

Example: Measurement units like "kilogram", "millimetre", "nanovolt", "gibibyte" etc.

To specify a Prefixed Unit, on the Basic tab, first select the Container RDL. From all the items in this RDL, select a Prefix (see 6.5 on Prefixes) and applicable Reference Unit (see 6.4 on Measurement Units). The Name and Short Name are generated automatically as a combination from the selected prefix and unit

6.5 Unit Prefixes

To create Prefixed Units, a list of unit prefixes is available.

Please read the general implication on managing RDL data before you proceed (see 6.1)

Open the Unit Prefixes browser by selecting the Unit Prefixes icon on the Reference Data tab. Unit Prefixes are given with a *Short Name*, a *Name*, *Conversion factor*, and *Container RDL*. The list of available prefixes contains the current official SI prefixes. **Although it is possible to extend the list of prefixes with the normal functionality to create an item, this shouldn't be needed in principle.** It is possible, but not advisable to edit Unit Prefixes. Unit Prefixes cannot be deleted, only deprecated.

6.6 Categories

Categories can be created and applied to concepts in the CDP4-COMET model to structure and manage it. According to ECSS-E-TM-10-25, a category is a representation of a user-defined category for categorization of instances that have common characteristics. The categories are defined and managed in the RDL, and can be applied to specific concepts in CDP4-COMET. A key purpose of categories is to structure a budget at different levels of abstraction. Also categories are required to do rule based verification and to create traceability. The use of categories will also allow the easier retrieval of specific information in CDP4-COMET, as well as to allow easier navigation, filtering or sorting for particular domains.

Please read the general implication on managing RDL data before you proceed (see 6.1).

Applying categories to each item created is error-prone. For this purpose, template elements can be created at different levels of abstraction. It is important to teach participants to use categories consistently, and to task the System Engineer or a designated supporting team member to guard the consistent use of categories. Various types of rules can be setup that use one or more defined categories to express relationships between various modelling concepts, see the section on rules itself for more details. For the use of categories in rules, and setup the correct and meaningful relationships between modelling concepts in these rules using the categories, it is important to set these up properly.

To create, edit or deprecate categories, Open the Categories Browser by selecting the Categories icon on the Reference Data tab. This browser contains a list of all the Categories contained in all the RDLs that are opened at that moment for the selected data source. Categories are shown with a *Short Name*, a *Name*, *Super Categories*, and *Container RDL*.

To create a new Category, select the Create Category icon or right mouse click to select Create a Category. On the Basic tab, provide the mandatory fields. Categories can be edited, they cannot be deleted, only deprecated. Optionally, tick the check box for the Abstract field (to indicate that a category is non-physical).

On the Permissible Classes tab, select the Classes to which the category can be applied. At least one permissible class should be selected to allow a new category to be created.

Optionally, it is possible to indicate one or more super categories on the Super Categories tab. The super categories can be used to build up a (hierarchical) set of related categories. As an example, consider the following high-level concepts as used in the ECSS standards:

- Requirements - derives into one or more requirements
- Functions - satisfies one or more requirements, usually expressed as a term or short sentence with a verb
- Products - implements on or more functions, can be derived from one or more requirements, usually expressed as a noun

For functions, there is a (hierarchical) relationship between systems and subsystems:

- System - a set of interrelated or interacting functions
- Subsystem - part of a system, fulfilling one or more functions

There is also a hierarchy within the products:

- Element
- Equipment (or Unit)
- Component (or Part)

By creating a category "Element", "Equipment" and "Component", it is possible to indicate that the Super Category for all of these is "Product". It is then possible to go through the model and look for all items that are of the category "Product". This can be done directly by looking for items that are categorised as "Product", or by having the categories "Element", "Equipment" etc. applied, which have the "Product" as a super category. An example where this is applied or used extensively is in the relationship matrix (see 10.8), or in the reporting editor (see 13).

6.7 Rules

A Rule defines relationships between concepts in ECSS-E-TM-10-25: *representation of a validation or constraint rule for Categorizable Things and relations between them*. Rules can be created to assist in a correct modelling approach according to industry- or organization-specific standards and methods. Applying categories to modelling concepts, such as Element Definitions, Element Usages, Parameter Types or Requirements, and using rules to define the relationships between these concepts, allow to provide a backbone structure to an engineering model, provide real meaning to the objects and parameters used in the modelling, and enable validation of the model.

Please read the general implication on managing RDL data before you proceed (see 6.1).

Applying Rules and categories further provide a powerful mechanism to work with the various concepts in CDP4-COMET, setting up links, providing meaning, and allowing filtering and combining various sources of information. This can help to setup basic calculations for users, up to providing useful entry points for automation to retrieve information, e.g. for the integration of Domain Specific Tools (DSTs). An example for the application of rules is the Relationship Matrix Browser, where it is possible to create and inspect (bidirectional) relationships between modelling items once the correct categories and rules have been setup (see 10.8 on relationships).

Categories are used to define the rules and the relationships between concepts these describe. The rules are defined and managed in the RDL and can be applied to most concepts in CDP4-COMET. Open the Rules Browser by selecting the Rules icon on the Reference Data tab. This browser contains a list of all the Rules contained in all the RDLs that are opened at that moment for the selected data source. Rules are shown with a *Short Name*, a *Name*, *Container RDL* and *Type*. To create a new rule, select the Create Rule icon or right mouse click to select Create a Rule for each of the available types. On the Basic tab, provide the mandatory fields. Further provide the additional required fields for the different rule types.

6.7.1 Types of Rules

Binary Relationship Rule: A Binary Relationship Rule specifies for Binary Relationships that are a member of the relationship Category what are the valid Categories for the related source and target Things

Example: A rule where the relationship Category is Category "Requirement Satisfaction Traces", the source Category is "Architectural Elements" (Element Definition, Element Usage) and the target Category is Category "Requirements".

On the Basic tab, provide the description for the Forward and Inverse Relationship Name. This should give a concise description of the binary relationship that is intended. A Relationship Category has to be provided as well.

Decomposition Rule: A Decomposition Rule specifies for Element Definitions that are a member of the containing Category what are the valid Categories (specified by contained Category) for the type of contained Element Usages. A subcategory of a valid Category is also valid.

Example: A rule where the containing Category is Category "Equipment" and the contained Category is Category "Sub Equipment".

On the Basic tab, provide the containing category. Next, select the contained category (or categories). If applicable, a minimum and/or maximum number of contained items can be indicated

Using decomposition rules, a hierarchical description of the model can be defined, specifying a containment tree of possible categories. The categories by themselves are defined to be applicable to specific modelling concepts, e.g. the categories "SYS", "ELE" and "EQT" have permissible classes Element Definitions and Element usages. The decomposition rules can now specify the hierarchical ordering of these categories, by defining containing and contained categories. As an example, a rule can be setup for the “System” as Containing Category will hold at a next lower level a “Subsystem” as Contained Category: System – Subsystem. Similar for the “Equipment” as Contained Category for the containing category of “Element”: Element – Equipment. This combination of rules can then be used to check and validate the setup of an engineering model, checking if an element usage is placed at the correct level.

Multi Relationship Rule: representation of a validation rule for Multi Relationships that relate (potentially) more than two Categorizable Things. On the Basic tab, select one or more categories in the Related Category overview. A Relationship Category has to be provided as well. The Multi Relationship Rule can be used to verify if concepts in the model are correctly categorized in the context of the multi relation.

Parametrized Category Rule: Rule that asserts that one or more parameters of a given Parameter Type should be defined for Categorizable Things that are a member of the associated Category. On the Basic tab, select the Category to which the rule will apply. Then select the Parameter Types that should be included for modelling items that have the selected category applied. Parameterized rules can e.g. be used to check an engineering model setup for completeness. As an example, it can be used to setup a rule that any element definition or element usage that has the specified category applied to it, e.g. "Equipment" (EQT), should have at least a parameter "mass" associated with it.

Referencer Rule: representation of a validation rule for Element Definitions and the referenced Element Nested Elements. On the Basic tab, provide the referencing category. Next, select the referenced category (or categories). If applicable, a minimum and/or maximum number of referenced items can be indicated

6.8 Constants

Constants are constant variables that can be used in one or more domains. An example of a constant is π. Very often physical, mathematical or other constants are used as parameters in calculations by domain engineers. To keep track of these parameters they can be entered into CDP4-COMET as constants to make them available to all users across engineering models on a CDP4-COMET server.

Please read the general implication on managing RDL data before you proceed (see 6.1).

To create a constant, open the Constants browser by selecting the Constants icon on the Reference Data tab. Constants are given with a *Short Name*, a *Name*, *Value*, *Scale*, and *Container RDL*. Persons with a role that has the permission to modify the Site Reference Data Library are able to access the Basic RDL to create, edit or deprecated Constants. Constants cannot be deleted, only deprecated.

6.9 Glossary

Every domain has its own unique language, and the primary purpose of the activity glossary is to understand and capture that language. The activity glossary provides a dictionary of key terms and definitions.

Please read the general implication on managing RDL data before you proceed (see 6.1).

Open the Glossaries browser by selecting the Glossaries icon on the Reference Data tab. Glossaries are given with a *Short Name*, a *Name*, *Value* and *Container RDL*. Right mouse click on the white space and select Create a Glossary and within a glossary Create a Term. The existing entries can be edited, deprecated, or the details can be viewed. Furthermore, it is possible to add synonyms.

Glossary terms should be understandable by everyone in the activity, including all the domains or disciplines. As well as defining key terms, the activity must resolve synonyms and homonyms. In the activity glossary you should record the preferred term and list any synonyms under the definition. This may involve encouraging some domains or disciplines to become accustomed to different terminology. It is best to set up a glossary in the beginning of the activity, but has to be updated continuously, making sure that it is up to date throughout the full life cycle of the subject of the activity. Texts corresponding to an entry in the Glossary are indicated as blue texts when used in CDP4-COMET, e.g. when defining requirements in the CDP4-COMET Requirements Manager (see 7.1 on Requirements). The definition of the term as given in the Glossary is presented in a tooltip box when hovering over the blue text.

6.10 File Types

File types are the extensions of computer files to indicate their type. For instance, .docx is the file type for MS Word documents. To store documents in the CDP4-COMET model, a library of file types is needed.

Please read the general implication on managing RDL data before you proceed (see 6.1).

To create a File type, open the File types browser by selecting the File types icon on the Reference Data tab. File types are given with a *Short Name*, a *Name*, *Extension*, and *Container RDL*. File types cannot be deleted, only deprecated.

6.11 Reference Sources

Reference Sources are links to documents describing standards and sources of information used in the model. For instance, ECSS-E-TM-10-25A is a reference source. To store Reference Sources in the CDP4-COMET model, a library of Reference Sources is needed.

Please read the general implication on managing RDL data before you proceed (see 6.1).

To create a Reference Source, open the Reference Source browser by selecting the Reference Source icon on the Reference Data tab. Reference Sources are given with a *Short Name*, a *Name*, and *Container RDL*. More information about the Reference Source can be provided, such as version and author information. Reference Sources cannot be deleted, only deprecated.

7. REQUIREMENTS

7.1 Setup Requirements Manager

Open the Requirements Browser by selecting the Open Requirements icon on the Requirements tab. The requirements specifications and Requirements are given with a *Short Name*, a *Name*, *Definition*, and *Owner Name*.

Requirements are managed within a Requirements Specification. This provides a high-level container for the actual requirements with the associated simple values and parametric constraints, which are described further below.

To create a new Requirements Specification, select the Create Requirements Specification icon or right mouse click to select Create a Requirements Specification. On the Basic tab, provide the mandatory fields. You can edit, inspect and deprecate a specification.

Within a Requirements Specification, the Requirements can be placed in Requirement Groups, that are used to group and organize the requirements as in the chapters and sections of a requirements document. From a modelling perspective, requirements groups do not have any further meaning, they are mostly for user convenience (as is the case for the parameter groups for instance) in visually combining specific sets of Requirements.

To create a new Requirement, select the Create Requirement icon or right mouse click to select Create a Requirement. On the Basic tab, provide the mandatory fields. You can edit, inspect and deprecate a requirement or requirement group.

In the description field, several basic editing capabilities are included, e.g. to mark text italic, bold, or to create numbered or bulleted lists. Normally, the description field in the Create or Edit Requirement dialog will be in editing mode. It is possible to look at the result in normal reading mode by clicking

the Preview icon. To go back to editing mode, select the Edit Note icon, which is placed in the top right corner of the Description field. Please note that this fades out after a few seconds and becomes visible again when hoovering over it.

Optionally, a Category can be applied to a Requirement. To do this, provide the applicable category or categories by selecting these on the Categories tab. Given on this tab are only the Categories that can be applied to Requirements. This is done by including Requirements in the list of permissible classes in managing the Categories.

7.2 Simple Parameter Values

From the definition in ECSS-E-TM-10-25, the only mandatory fields for a requirement is the *Short Name*, *Name*, *Owner* and *Description*. In requirements management and requirements engineering, Requirements are usually characterized by other aspects as well.

It is possible to create further attributes for requirements by adding Simple Parameter Values to a requirement. As an example, it is possible to set up requirements with specific attributes, e.g. according to the ECSS-E-10 standards as used in the European space industry. ECSS-E-10 can in this case be used to define the Requirements with attributes such as Requirement Types, severity and verification methods.

In principle, any existing Parameter Type in the RDL can be added to a requirement as a Simple Parameter Value. Usually, dedicated Parameter Types for the required attributes for requirements according to a specific (industry) standard or requirements management method will most likely have to be added, see section 6.2.1 on how to create and manage parameter types.

To create a Simple Parameter Value, create or edit a requirement, and then on the Simple Parameter Values tab, select the Create Simple Parameter Value icon or right mouse click to select Create Simple Parameter Value. Select the parameter type that is required as attribute for the requirement. If possible or desired, an initial value for the simple parameter can already be provided. Alternatively, it is possible to add a Simple Parameter Value by using drag-and-drop from the Parameter Types browser onto a Requirement.

In some cases an alternative to modelling attributes as Simple Parameter Values, in any case for e.g. requirement types, is by defining these as categories (see 6.6). This can be done by defining a category Requirements, which is applicable to the Permissible Class Requirements. The requirement types, such as functional, interface, operational, verification etc., can then be created as separate categories, having Requirements as Super Category. Within the requirements specification, the different types can not only be separated visually by placing them in different requirement groups but can now also be labelled using these categories. It is then possible to go through the model and identify requirements of interest, e.g. by looking for operational Requirements and linking these to other Requirements, e.g. verification requirements or Element Definitions and/or Element Usages. For this linking to be possible, predefined (Binary Relationship) Rules (see 6.7 on Rules) need to be setup, expressing the characteristics of the link, .e.g. a “derived”, “implements”, “satisfies” or “trace” relationship. These links can be created graphically in the relationship editor. Another possibility is to create, inspect and update these links in the Relationship Matrix (see 10.8 on the Relationships Matrix).

7.3 Parametric Constraints

Parametric Constraints can be added to a Requirement. These are used to create a mathematical expression for the requirement definition. This expression can be interpreted by a machine, and thus allows for automatic verification with respect to the design solutions.

To create a Parametric Constraints, create or edit a Requirement, and then on the Parametric Constraints tab, select the Create Parametric Constraints icon or right mouse click to Create a Parametric Constraints, and then click Create Relational Expression. Select the Parameter Type that is required as attribute for the requirement, with the appropriate scale. Next, select the requirement operator for the expression, and the corresponding value.

As a next step, it is possible to link a Requirement to a design Parameter through the Parametric Constraint. To do this, select a Parameter and link it by using drag-and-drop from the Element Definitions or Product Tree browser onto the Parametric Constraint. It is only possible to link a Parameter that is of the same Parameter Type and uses the same Measurement Scale (See 6.2 and 6.3 on Parameters and Scales).

A linked Parametric Constraint and the Parameter get a green link overlay icon in the Requirements Browser and the Element Definitions or Product Tree Browser respectively (see 9.2 and 9.4 on the Elements Definition and Product Tree Browsers).

7.4 Requirements Specification Editor

The Requirements are contained within a Requirements Specification, and organised in groups, as explained in section 7.1.1. These can be opened in a Requirement Specification Editor, which provides an overview of the Requirements Specification in a textual format, so that it reads more like a requirements document. Within the Requirements Editor, the items are presented in text boxes with different background colours. To open the Requirements Specification Editor, right mouse click a Requirements Specification.

The Specification is in light green and Requirement Groups in light red. The specification and requirement groups are given as headers, ordering the requirements. These are given with their Name, Short Name, and Owner. These cannot be edited within the Specification Editor.

The Requirements are given in light purple, and have a Requirement icon. The requirements are given with their *Name, Short Name, Owner* and *Categories*. Within the box, the textual description of the requirements, in their Definition, is given in a reading-friendly textual format. The Simple Parameter Values and Parametric Constraints associated with a requirement are not provided in this browser.

It is possible to edit the Definition. To do this, select the Edit Mode icon to go into edit mode for the Definition field. In edit mode, various formatting functionalities are available to adapt the font, insert links, or create ordered lists.

7.5 Requirement Verification

A checking mechanism is implemented in the CDP4-COMET Requirement Manager to indicate the agreement of a design parameter in the model with the conditions of the Parametric Constraint (see 7.3 on Parametric Constraints). To do this, select Check Requirements when right mouse clicking any of the items in the Requirements Browser. A green background colour is given on the lines that are related to a Parametric Constraint (i.e. the row of the Parametric Constraint itself and upwards to all its parent items) if the conditions are satisfied and red lines if the conditions are not met by the design Parameter. Yellow lines are given if the check cannot be performed to give a meaningful or correct result.

The conditions can in principle be used for most Parameter Types, but if general conditions are given the checking mechanism available in the CDP4-COMET Requirements browser only gives meaningful results for numerical parameters, i.e. for Quantity Kind Parameter (with Integer or Real number sets), Date, Date Time and Time. For Boolean, Enumeration and Text Parameter Types only the condition '=' or '<>' will give a meaningful result. If this is applied, using the Check Requirements verification will indicate the status with the same green lines if the conditions are satisfied and lines if the conditions are violated. If other conditions are used for these Parameter Types, yellow lines will be given on the Parametric Constraint and its parent lines to indicate that the conditions cannot be evaluated properly (see 6.2.1 on Parameter Types).

Yellow lines are also given for Requirements that do not have Parametric Constraints associated with them, or that are in any case not linked to an appropriate design Parameter.

Please note that the value that is being used in the evaluation of the Parametric Constraint is the Actual Value, not the Published Value. This allows assessing the impact of ongoing design work and calculations on the Requirements, and performing several what-if scenarios or checks with the intermediate results, without having to go through the publication cycle first.

7.6 Importing and Exporting Requirements

Requirements can be imported and exported using the ReqIF standard. In addition it is possible to export requirements to html.

Importing and Exporting using ReqIF

For the requirements management in CDP4-COMET, support for Requirements Interchange Format (ReqIF) has been implemented. In this way it is possible to import a set of requirements from or export requirements to other tools that support this exchange format, such as other Model-based Systems Engineering (MBSE) tools or dedicated requirements management tools such as DOORS.

To export a Requirements Specification, on the ReqIF part of the Requirements tab, select the Export icon. This opens the Export to ReqIF File dialog. In this dialog, select the target model, provide the path where the ReqIF file should be saved and select Export.

To import a requirements specification in ReqIF, on the ReqIF part of the Requirements tab, select the Import icon. This opens the Import Requirement Data from ReqIF File dialog. In this dialog, select the

target model, provide the path to the ReqIF file, and optionally provide a saved mapping configuration, see below; the default mapping selection is set to (Auto). Select Import to start the import process.

These are the basic steps to import a set of requirements. Extra dialogs are given to provide additional information about the ReqIF content and its mapping to the ECSS-E-TM-10-25 concepts. If these concepts are not included in the ReqIF file, these dialogs may be empty

Relations for Group
Relations for the Specification
In the final dialog, a preview is given of the result of the import.

In this last step, it is possible to save the settings for the mapping of concepts as applied in the import steps. This settings file is stored locally in a configuration file. When all steps are performed, this will create the requirements specification with all its underlying content as a new item in the Requirements browser. Updating an existing Requirement Specification is not yet supported in the current implementation.

7.7 Exporting to HTML-file

It is possible to create an HTML export of a requirements specification. On the HTML part of the Requirements tab, select the Export icon. This opens the Requirements HTML Export dialog. In this dialog, select the source Engineering Model and the required Iteration from the list of all combinations that are currently open in the CDP4 model. Next, provide the path with a file name where the HTML file should be saved, the Requirements Specification to export and then select Export. This will create the HTML export file in the selected folder. This file can be opened with any suitable internet browser (e.g. Edge, Firefox, Chrome etc.).

8. OPENING A CDP4-COMET MODEL

8.1 Login

The login procedure in the CDP4-COMET app involves several steps, depending on which actions the user wants to perform. Open the CDP4-COMET app by double-clicking the CDP4-COMET icon on your desktop or start CDP4-COMET from the program files. This will open the main CDP4-COMET IME. In any case a user has to connect to a data source to be able to perform any action within CDP4-COMET.

8.2 Connect to Data Source

In the CDP4-COMET app, go to the Home tab and select Connect. This opens the Data Source Selection Dialog. In the drop-down, select the Source Type to connect to, standard this is CDP4-COMET/CDP4 Services. To connect to a CDP4-COMET server provide the correct URL

URL examples:

CDP4-COMET server: http://cdp4services-public.rheagroup.com

Connect with a CDP4-COMET Username and CDP4-COMET Password (see 5.4 on creating users). The correct server URL, as well as your CDP4-COMET Username and CDP4-COMET Password should be distributed by the CDP4-COMET administrator.

In case a Proxy server is required, check the **Use Proxy** check box and provide the required details (this may include the URL, username and password. Contact your network administrator for details).

Once you are connected, the URL is visible as session in the refresh area of the Home tab. You can connect to more than one server, and you can also disconnect. Finally, it is possible to hide all panels and browsers from a specific server.

To try out CDP4-COMET use the server https://cdp4services-public.cdp4.org

And log in with credentials: admin and password: pass

The server will be reset every night.

8.3 Opening a model on a CDP4-COMET Server.

When connecting to a CDP4-COMET Server, the default option is 'connect and open model' this will open a Model browser, where the user can see and open all Models he/she contributes to. If you selected to only connect to the server, or if you want to open a second model, select the Open button on the Home tab.

To be able to work with a specific Engineering Model, the user has to open it, choosing a specific Iteration (see 10.9 on iterations), representing a particular Domain (see 5.1 on domains). It is possible to open multiple Engineering Models in parallel.

8.4 Refreshing Data

In performing Concurrent Design activities, the complete team is continuously making changes to the data model, e.g. changing values, adding objects or Parameters, adding or editing Requirements, uploading documents etc. In the CDP4-COMET app of a team member his own changes are immediately visible. Changes made by other team members are communicated through the CDP4-COMET database. It is therefore very important for users at specific moments to retrieve the latest information of the data model from the database to make sure the user is in line with the state of the complete Model.

Refreshing can be performed automatically. This can be set separately for each CDP4-COMET data source a user is connected to. To enable this functionality, go to the Home tab and in the drop-down on the Data Source group select a CDP4-COMET data source. An appropriate time interval can be chosen by the user for the Auto Refresh, given in seconds [s]. This should be filled in in the AutoRefresh Interval field. When editing domain specific information, it may be practical to switch the AutoRefresh off, to not be distracted by data updates.

Figure 9: Refresh menu

On the Data Source group, it is possible to refresh all the data as well if needed through the Refresh icon to retrieve an updated list of changes from the database. Clicking the Reload icon will reload all the data from the activity. When a refresh, auto-refresh or reload is performed, this is indicated in the feedback message in the status bar. The refresh or reload actions are visible in the Log Information browser. Besides the general refresh functions, for each individual window that is open in the CDP4-COMET Client, manual refresh is possible by using the Refresh icons.

8.5 Opening a JSON File Based Model.

A CDP4-COMET Model can be exported to a JSON File. This is stored as a zip-file on the local computer of a user. A user has to provide an existing CDP4-COMET username. A password has to be provided for the JSON zip-file as well; this doesn't have to be the same one as for the user on the CDP4-COMET server itself, a separate archive password can be chosen as well.

To connect to a JSON File Based data source, the user has to browse to the location where the JSON File is stored. Selecting the browsing icon opens a Windows Explorer view to browse to the correct zip-file. The archive password has to be provided as well. This is the password has been set when exporting the CDP4-COMET model to the JSON File and should be known by the user or provided to him if the JSON File was created by another person.

9. CONTRIBUTING TO A CDP4-COMET MODEL

9.1 Conceptual Modelling in CDP4-COMET

When creating a CDP4-COMET model we build one or more Product Trees (see 9.4) that represent the system design in one or more Options (see 10.7 on Options) in order to meet a set of Requirements (see 7 on Requirements). The System Design consists of Elements with properties that we specify through Parameters (see 6.2 on Parameters). The System is designed as a tree structure. On the top level we define the top element, which represents the entire system. Below we define its key components, usually a platform system, some operation unit and potentially other key systems such as charging stations. Each system is then again composed of smaller subsystems, which are composed of even smaller elements until we end up with nuts, bolts and wires. CDP4-COMET is typically used for a design that is detailed up to equipment level. We can also set the top element to an element at a more detailed level in the Product tree. In this way we can focus the session on a specific subsystem or part of the system. It is not the objective to Model everything there is to know about the system. The Model is built to test if we can design solutions that meet a specific set of key Requirements, we call Design Drivers. For instance: Can we build a faster boat within a budget and with a maximum weight to fit a specific class? In this example, we need to specify each element in the design with its weight,

costs and power usage. This will enable us to create an overview of the overall weight, costs and power available for propulsion. Aspects such as the materials used, the precise location of equipment and the maintenance costs of components are not yet relevant to answer this question, and can be added later.

To specify the design, we thus need to create a decomposition of the system with elements to achieve the right level of abstraction required to answer the questions of the study. We also need to specify these elements with the properties that are relevant to calculate the systems' overall performance with respect to the relevant Design Drivers. For this purpose, we need to specify Requirements, the Elements of the system and the Parameters that indicate their properties. Then we will want to do this for several options to compare the best overall design solution. To avoid modelling the system many times, we need to use Building Blocks in our design.

9.2 Building Blocks, Element Definitions and Element Usages

The Element Definition Browser is where we define the concepts that are used in the model. These are the Building Blocks of our model. We first need to define the Building Blocks. Next, we can use them in our model. Some concepts consist of other concepts, for instance, a communication subsystem can consist of several equipment's such as a radio and an antenna. This can be part of the definition of the concept communication subsystem. We indicate this by nesting the Element Definition of the sub-element in the super element. The nested element is copied as an Element Usage, and listed separately as such, below its super-element. We can make several changes to the Element Definition, and the corresponding usages will automatically change accordingly. However, it is also possible to change only the Element Usage. In that case, the definition will not change. In this way we can efficiently re-use concepts in different options but vary them for specific situations. Changes to an Element Definition that will affect the corresponding Element Usage are:

- Adding, updating or deleting parameters and parameter groups/grouping
- Changing parameter values and/or switch settings (see 9.6)
- Applying or editing state and option dependency of parameters (see 9.6)
- Changing ownership of parameters (see 10.2)
- Setting subscriptions for parameters (see 9.7)
- Applying categories (listed as Element Definition categories on Element Usages) (see 9.8)

Changes to an Element Definition that will not affect the corresponding Element Usage are:

- Renaming the Element Definition

Changes to an Element Usage that will not affect the corresponding Element Definition are:

- Renaming the Element Usages
- Creating overrides for parameters (with possible change of ownership of override, see 9.6)
- Applying or removing categories applied to a specific Element Usage

In most cases it is most efficient and consistent to change and Element Definition, so that all its usages also change. However, there are situations in which we want to change Element Usages. For instance, consider a car with four wheels. We create one Element Definition 'wheel' and add 4 of these to our car. We then need to make these unique, by naming them left front wheel, right front wheel, etc. Now if we add a position parameter to the wheel, this value would be different for each of the four wheels.

However, the mass, diameter, costs, etc. will be the same. In this case we can change the position parameter of the element usage.

 Element

 Element Usage

9.3 Create an Element Definition

To create a new Building Block in our design we open the Element Definition Browser. Here we can add a new Element, which we give a *Name* and *Short Name*. We also define the Owner, which is by default the active domain of the engineer who created it. It is further important to categorize the element in the Category tab. CDP4-COMET will further give the Element a model code based on the Short Name.

To use an Element in the Element Definition of another Element, we can drag the element that defines the sub-element on the super element.

To further specify the Element, we need to add Parameters to it.

It may be useful to create an Element with the same parameters as another Element. In this this case we can copy the Element. In this way we don't have to specify the same parameters for each element we create. The System Engineer may specify templates for systems, subsystems and equipment to pre-define key parameters for each type of Element. There are different copy functions, and each copies different aspects of the data:

- COPY: The Element Definition is copied, including the contained Element Usages and Parameters. The ownership in the target model is set to the Active Domain. The values are set to "-" by default.
- CTRL COPY: The Element Definition is copied, including the contained Element Usages and Parameters, the ownership in the target model is set to the Active Domain. The values are set to values as defined in the source Model.
- SHIFT COPY: The Element Definition is copied, including the contained Element Usages and Parameters. The ownership in the target model is set to the ownership as defined in the source model. The values are set to the default "-".
- CTRL+SHIFT COPY: The Element Definition is copied, including the contained Element Usages and Parameters. The ownership in the target model is set to the ownership as defined in the source model. The values are set to values as defined in the source Model.

9.4 Product Tree

To see an overview of the entire structure of the system, we use the Product Tree. The Product Tree is different for each option. We use it to inspect the result of our design. It is possible to modify the model both in the Element Definition Browser and in the Product Tree, but it is **advised to use the Elements Definition Browser to create new concepts in the design**. This is to avoid confusing

changing the Element Definition with changing the Element Usage. In the Product Tree we see mainly the Element Usages.

In the Elements Definition Browser we can see only one level down. If you want to see the full overview of the System, we need the Product Tree. To see the Product Tree, we need to specify the top Element. Select the Element that represents the Highest Level overall System, and right click to choose set as Top Element. When you now open the Product Tree you see the entire system tree. Elements that are not used in other Elements are not displayed in the Product Tree.

9.5 Add Parameters

To add a Parameter, it is easiest to open both the Element Definition Browser and the Parameter Definition Browser (see 6.2). You can also choose add Parameter when you right click the Element. Find the Parameter you want to use to define the Element and drag and drop it onto the Element Definition. You can remove the Parameter. If you cannot find the Parameter you need to specify you Element, it can be created (see 6.2). Once you added the right Parameters, you need to specify their value to use them zo calculate the performance of the Design.

9.6 Specify Parameter Values

To fill in the value of the Parameter, go to the Element Definition Browser. Find the Element you want to specify and check if you are the Owner of the Element (see 2.6 on Ownership). Usually, only the Owner can edit the value. In a single Option, there are three fields in which you can specify the value of the Parameter for the element: Manual, Computed and Reference value. There is a Switch, to choose which Parameter is the value that needs to be used. You can also see the Published Value. This is the value that is accepted by the System Engineer.

- **Manual**: this is a direct estimate of the value
- **Computed**: this is a value that is calculated based on more detailed information
- **Reference**: this value is derived directly from an element specification, previous project or other reliable source of information.

With the different values you indicate the reliability of the data. A reference value is most reliable, it is fact-based. A computed value is calculated by an expert, the manual value is an estimate. Computed values can be e.g. a single value as a result from a calculation in a report or a formula referencing a cell on a calculation sheet in an excel workbook. It is not possible to define a value as function of other Parameters in the CDP4-COMET app, you have to create a calculation sheet in the Excel plug-in and link the result to the Computed value. (see 11). Other possibilities are doing calculations in a CDP4-COMET report, for which results can be linked to specific parameters and submitted as computed values, or by calculating values in Domain Specific Tools (DST) and feeding the results into CDP4-COMET. To specify the value in the model, simply go to the appropriate value field in the Element Definition Browser table and adjust it. You can also select edit parameter when right mouse clicking the Element.

When a new value is specified, and the SWITCH is set to this value, it becomes blue, which indicates that it is ready for Publication. The System Engineer will inspect the value and Publish it when deemed

correct (see 10.4 on Publishing). A value that is not specified, is indicated with a '-' symbol. This is important to avoid thinking the value is 0.

If a Parameter is Option and/ or State dependent (see 10.5 and 10.7 on States and Options), the table will expand and the value can be specified for each Option and/ or State of the parameter in the Element Definition. For option dependency, tick the check box in editing the parameter. For option-dependent parameters only the value specific for that option will be visible in the corresponding element usage in the product tree. To set the state dependency, select the required actual state list (see 10.5) in the dropdown menu.

For parameters in element usages, the concept of override can be used to provide a value which only applies to that particular Element Usage. This does not affect the value of the parameter in the Element Definition, or in any other possible Element Usage that was created for the parent Element Definition. This mechanism allows e.g. to model 4 wheels of a car with similar properties for some parameters (e.g. size and mass of tires), while making others specific (e.g. the actual position: front left, front right, back left, back right, or with different x-y-z coordinates). For a parameter that is overridden, it is possible to change the ownership as well. This also applies to only the overridden parameter in that specific Element Usage.

9.7 Create Subscriptions

When a Domain Expert uses values of other Domain Experts to perform calculations on system Parameters, he/she needs to Subscribe to this Parameter. To subscribe, right mouse click on the Parameter and select subscribe to this parameter. You can see the Parameters you Subscribe to in you Excel workbook, and you can use them to calculate the value of other Parameters. More importantly, you signify the Owner, that his/her data are used, so the expert is aware of the dependency. You can delete the Parameter Subscription. In the Product Tree, Parameters that have a Subscription by you have a white dot in front of them. Parameters that are subscribed to by other domains (not yours) are marked blue. In this way we can see the parameters that are used by others. Parameters that no one is subscribed to are orange.


A Parameter subscribed to by others


A Parameter you subscribed to


A Parameter without any subscriptions

9.8 Categorize Elements

Some properties of an Element can be assigned through Categories. To assign a Category edit the Element Definition and select the appropriate categories on the Category tab. Note that the use of Categories can have significant impact on (budget) calculations. It is important to Categorize Elements based on the instructions of the System Engineer. To add a Category that does not exist, see the section on Categories in the Reference Data (see 6.6 on Categories).

9.9 Add a Requirement

A domain expert can also bring in constraints to the design, rules or regulations the system needs to adhere to. These can be best specified as Requirements. To add a Requirement, go to the Requirements Specification, in the Requirements tab and select the specification that you want to add to. Use the right mouse click menu or the add icon to add a requirement and specify it's *Name, Short Name* and *Definition*. As all key Elements in CDP4-COMET you can edit and deprecate it, not delete it (see 7 on specifying requirements).

9.10 Add files

If Additional document-based information is required, this can be stored per domain in the Domain File Store. Open the Domain File Store from the Model Tab. Select a domain and select add file. In the Dialogue select the blue plus again and then select the upload icon in the right upper corner. Give your file a name and add it. It is possible to create folders in the domain file to structure the documents.

10. SYSTEM ENGINEERING / MODEL STRUCTURATION

10.1 Preliminary Engineering

While it is recommended to involve all key Domain Experts in the modelling effort, preferably though Concurrent Design sessions, it is also a good practice to to set up the model in a smaller group of key stakeholders. In this first model set up, the model has to be created, and the participants have to be assigned.

System Engineering typically starts from a set of Requirements. The structure (chapters in) of the Requirements Specification is designed, and the high-level System Requirements are specified (see 7 on specifying requirements). Next, a Baseline Model can be created. In a Baseline Model, the main hierarchy is defined. This is the high-level system decomposition to such level that all or most of the stakeholders can find the subsystems they need to contribute to. Also, the Baseline Model can be used to scope the design, what is part of the system and what is not.

At this stage it is also efficient to think about the Budgets that need to be created, and the Parameters that need to be modelled for each element in order to calculate the overall system performance (see 10.10 on Budgets). Using these parameters, template Elements can be created (see 2.5 and 9.3 on re-using Element definitions). When Domain Experts use these templates, it is ensured that the Elements they define are Categorized in the right way and contain the right Parameters. An example of a Template Element is to create an Equipment Template. Equipment should be Categorized as such and should be specified with at least its costs, mass, power consumption and TRL level. These Categories and Parameters can be added to the template, and Domain Experts can copy it to specify each Equipment they add to the model.

With a Baseline Model, the participants can recognize key components in the system and start contributing to the model. Usually, creating the Baseline Model also reveals the need to specify specific new parameters. At this point it is also important to consider who can add new reference concepts (parameters, categories, measurement units, etc.) and how those without these rights need to request the creation of a new reference concept (see 6.1 on managing Reference Data).

Once the Baseline Model is set up, it is also important to think about Options and States of the model. Both can be created at any time, but the earlier they are designed in the model, the less (re)work it is to specify them. When Options are created, those Element Usages that are Option dependent can be identified and made dependent, and similar, State dependency can be indicated (see 10.7 and 10.5 on Options and States).

Finally, the System Engineer can set up preliminary Rules to verify the design. These Rules will be used throughout the design to verify the consistency of the model. The earlier these Rules are implemented, the more they will guide the domain experts in modelling consistently. To verify Rules, often the related Elements need to be Categorized in a specific way, this Categorization can be embedded in the above mentioned Element Templates (See 6.6 and 6.7 on Categories and Rules).

10.2 Change Ownership

It can be required to change the Ownership of an Element Definition, Parameter or Requirement. When you right mouse click these components, there is an option to change the Ownership. Also in the Edit Dialogue the Owner is a property in the Basic tab that can be changed. The System Engineer can be assigned to all Domains. If so, the System Engineer can change his Domain of Expertise using the button on the Home tab to add multiple components from a specific Domain of Expertise. This is more efficient than changing the Ownership of each new component created for a specific Domain (see 2.6 on Domains).

10.3 Copy/move Model Elements

It may be needed to move items in the model to a different place in the hierarchy. For this we can delete the Element Usage and drop it onto another Element Definition (see 9.2) to nest it in the Product Tree (see 9.4) . When we want to re-use an Element Definition, we can copy it using different copy functions. There are different ways to copy a model Element that retain different aspects of its definition:

COPY: The Element Definition is copied, including the contained Element Usages and Parameters. The ownership in the target model is set to the Active Domain. The values are set to the default "-".

CTRL COPY: The Element Definition is copied, including the contained Element Usages and Parameters, the ownership in the target model is set to the Active Domain. The values are set to values as defined in the source Model.

SHIFT COPY: The Element Definition is copied, including the contained Element Usages and Parameters. The ownership in the target model is set to the ownership as defined in the source model. The values are set to the default "-".

CTRL+SHIFT COPY: The Element Definition is copied, including the contained Element Usages and Parameters. The ownership in the target model is set to the ownership as defined in the source model. The values are set to values as defined in the source Model.

10.4 Publish Data

When Domain Experts change the values of their Parameters, or add new Parameters, they turn blue in the system (see 9.6). After each Iteration or Session, the System Engineer inspects the changes and Publishes them. In Concurrent Design, Domain Experts are asked to explain the changes they made to the team. To publish new data go to the model tab and click the Publication Browser icon. Select the appropriate Iteration. The browser shows the changes sorted per Domain with the old and new value as well as the percentage of change. You can select the change of one or more domains and click Publish. The bottom part of the browser contains the history of Publications.

10.5 Manage Finite States

When designing a System in a CDP4-COMET model, it is for one State of the System. To include more than one State, we need to define Finite States for the Model. States of an airplane system are for instance, it's take-off, cruising and landing. In each of these States the power consumption is for instance different. There could be a second set of states, for example design, operation, and maintenance. We could then create combinations of States, where some are not desirable. In our example, we would not want take-off and maintenance to happen at the same time. We therefore distinguish Possible Finite States and Actual Finite States.

States can be used to give a description of this dynamic behaviour, e.g. mission phases or system or equipment modes. The definition consists of the *Name* of the list, with the associated States in that list. States can be applied to Parameters of all Parameter Types, including e.g. compound and array parameter types (see 6.2.1). It is also possible to apply an actual Finite State list together with setting an Option-dependency for a Parameter. It is important to realize that adding states entails fast multiplication of values for the Parameters to which it is applied. Especially when combining States and Options, the list of Parameters that need to be specified can grow fast, creating an unmanageable workload for Domain Experts.

The definitions of each State List needs to be given in a Possible Finite State List. Then combinations of Possible Finite State Lists can be made in Actual Finite State Lists. The States in the Actual Finite State List are generated as all possible combinations from the Finite States in the applicable Possible Finite State Lists. Please note that for a single Possible Finite State List, this has to be created as an Actual Finite State List as well to be able to use it.

The Actual Finite State Lists can subsequently be assigned to Parameters. The Parameter will thus be turned into a Stateful Parameter, with the automatic creation of a value for each combination as indicated in the Actual Finite State List. Actual Finite states can only be applied to Parameters within the scope of an Engineering Model, where these state lists are defined and stored in the Iterations. Over the Iterations, changes can be made to the State Lists, e.g. to reflect changes in the Options that

are under investigation. The State Dependency of Parameters within an Engineering Model is not transferred if an Element Definition containing them is copied to other engineering models.

To get an overview of the finite states open the Finite State Browser by selecting the Finite States icon in the Finite State group of the Model tab. This browser gives an overview of the existing Possible and Actual Finite State Lists with *Name*, *Short Name*, *Owner*, and when expanded *State Kind*, where this last column is only applicable to actual states.

Participants with a role that has the permission to modify the Actual and Possible Finite State List are able to access the Finite State Browser and create, edit or deprecate possible or actual finite states. Note that states cannot be deleted, but instead are deprecated.

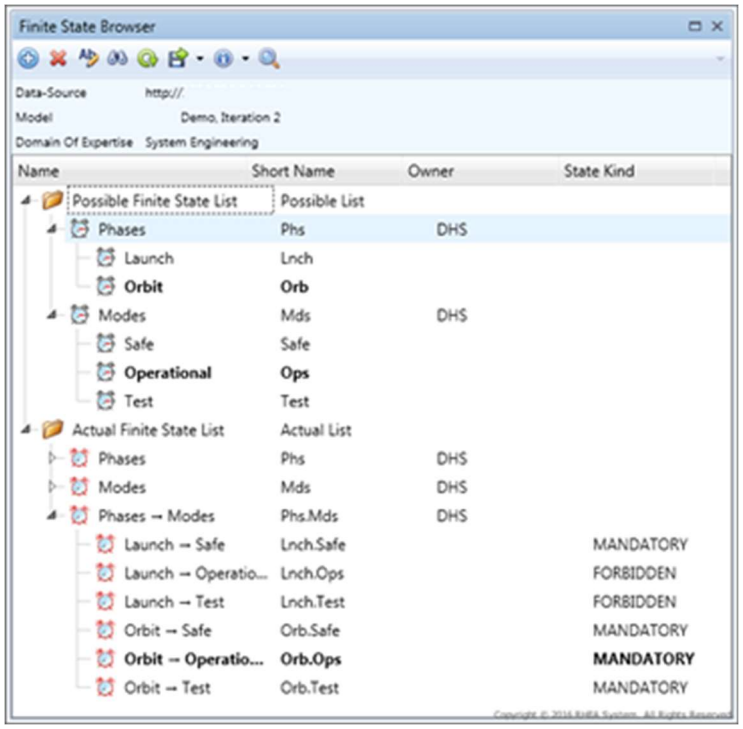


Figure 10: Finite State Lists

To create a new Possible Finite State List, select the Create icon or right mouse click to select Create a Possible Finite State List. On the Basic tab, provide the mandatory fields. Furthermore, the *Possible Finite States* have to be added to the *Possible Finite State List*. To do this, on the Basic tab, select the *Create a new Possible Finite State* icon and provide the *Name* and *Short Name*. One specific *Possible Finite State* should be set to default. You can change the order with the up and down icons. On the Category tab, it is possible to select a *Category* that is applicable to the *Class* of *Possible Finite State List*.

The *Actual Finite States* are derived as all combinations of the *Possible Finite States* in the *Possible Finite State List(s)*. In the *Create Actual Finite State List* modal dialog, select a domain as the *Owner* of

the *Actual Finite State List*. Select the *Add Possible Finite State List* icon to a row with possible finite state lists. All available *Possible Finite State Lists* will become available in the drop-down menu for each row, with the exception of any *Possible Finite State List* that has already been selected in any of the other rows. It is possible to add as many rows as there are *Possible Finite State Lists*. Each row can be edited. To delete a row again, select the row by clicking the box at the front of the row; this box should show a triangle icon. Select the *Remove a Possible Finite State List* to discard a row. Note that The Actual Finite State Lists and actual finite states will be deleted from CDP4-COMET. Please note that this will delete all corresponding values of Parameters that were depending on the Actual Finite State List.

It is furthermore possible to edit the *Actual Finite States* on the Basic tab. These can be edited for their *State Kind*, which can be set to either *MANDATORY* or *FORBIDDEN*. It is possible to change the relative ordering of the lists themselves. You can change the order with the up and down icons. On the Options tab, it is possible for each *Actual Finite State List*, together with the specific settings for its *State Kind* for the *States* to select to which option it is applicable.

10.6 Parameter To State Mapper

The Parameter to State Mapper can be used to Map Parameters that are defined in an Element Definition and that are static, to a state dependent Parameter Override in an Element Usage, where the value of that Parameter Override varies depending on the Actual Finite State.

10.7 Manage Options

Using the same Model, we can compare different design Options. For instance we can Model a car and test the feasibility of the Design Drivers by using two different engines. Choosing a different engine, may required a few other components to be different as well, like the fuel tank, but many other components can be the same for both Options. To model these Options using the same Baseline Model, we create a Product Tree for each option. The Product trees are generated for each option specifically (see 9.4 on Product Tree).

The options are managed in the Options browser. When creating an Engineering Model, Option 1 is created as Default. It is possible to create additional options, editing the name and short name, and select which option is default. It is possible to delete options as well. Please be aware that the option is not set to deprecated, but will be deleted from CDP4-COMET. Please note that this will delete all corresponding modelling data in that option, i.e. all the information in the corresponding Product Tree.

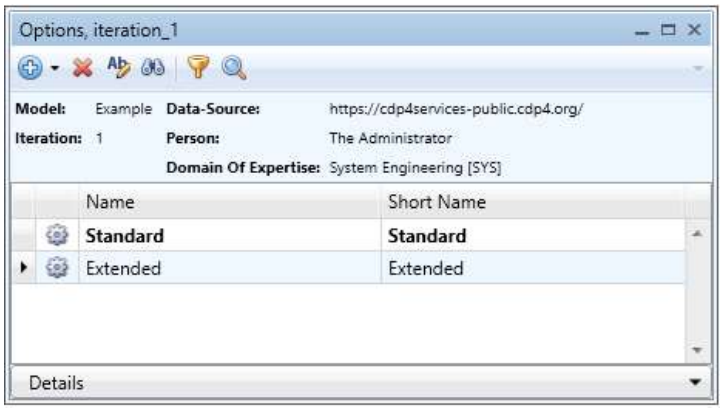


Figure 11: Option browser

The distinction between the Options can be expressed by indicating the desired Option dependency of an Element Usage. If the Option dependency is indicated for an Element Usage, this means that this element usage will be included in the Product Tree for that option when this is generated. By unticking the check box for an Option, the Element Usage will be excluded from it, and the underlying Parameters will not show up on the generated Option sheet for that Option. Setting the Option dependency for Element Usages is thus one of the main mechanisms to create different Product Trees for different Options to explore these as solution directions in a CD Study.

In the Element Definition Browser, an indication is given of the Option dependency of Element Usages. If an element usage is included in all Options, this is indicated with the green circles icon, if used in some of the Options a blue list icon, and if not used in any of the Options a red cross icon. To edit the Element Usage for the Option dependency, clicking on the icon will display a drop-down menu with the available Options. Use the check box to include or exclude the element usage for each specific Option.

	The Element is Option dependent
	Element included in all Options
	Element included in none of the Options

For any new Option that is added to an Iteration of the Engineering Model, the relevant items are updated to include it. The Element Usages will be updated, where these will all be set to be included for the new Option. Make sure to review the list of applicable Element Usages after creating a new Option if this is required.

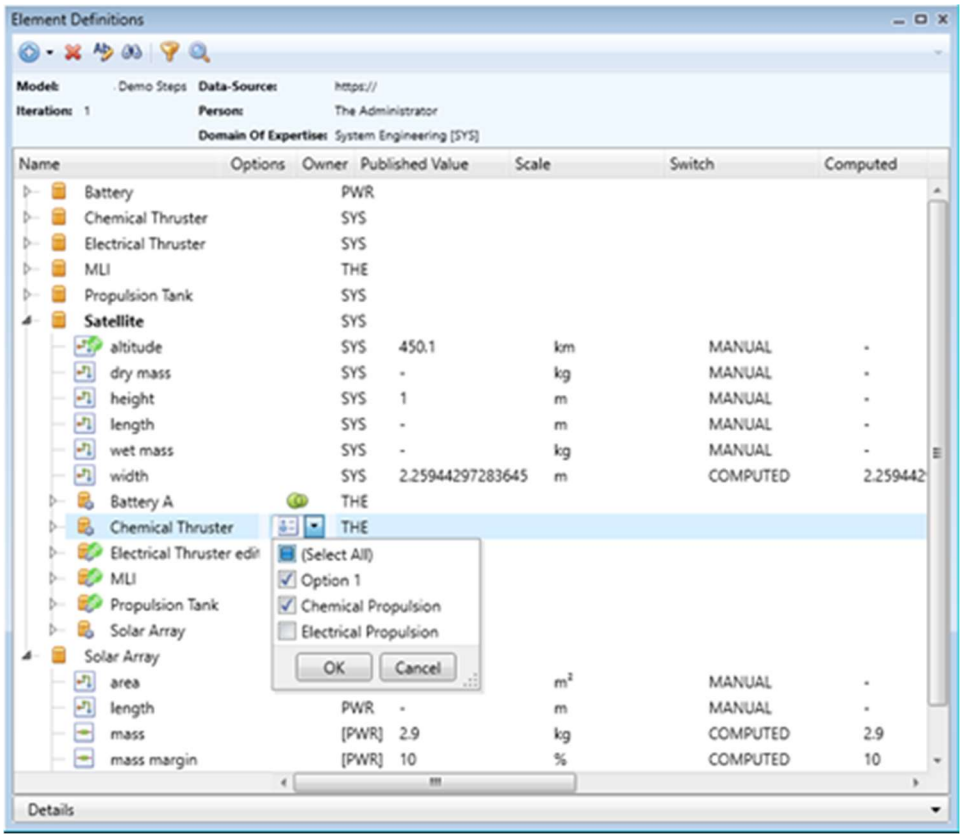


Figure 12: Option Selection

10.8 Manage Relationships

In the Relationships Browser, we can create Relationships using a drag-and-drop function. We recommend using the Relationship Matrix to create and manage Relationships.

The Relationship Matrix allows you to gain an overview of related concepts in the model. This can be used for instance to create a traceability matrix of Requirements that are implemented in the design. Again, the relations can be created in the Relationship Browser. The Relationship Matrix only supports Binary Relationships. Select the X and Y axis of the matrix by selecting the type of concept that needs to be shown (e.g. requirements, element usages), filtering these on specific categories of interest. The relationship matrix will only show a result if the selections can give a valid result, which means that the correct categories that are related to the type of relationship that should be created or viewed are selected, and categorisation and definition of both the relationship rule and the modelling items is done correctly.

Here we can identify the source and target of the relation in the matrix and specify the relation by double clicking the cell; this creates a link in the direction from row to column item. The direction of the relation can be defined by right mouse-clicking the cell and selecting the appropriate arrow. Note that binary relationships depend on a correct choice of this direction.

10.9 Managing Iterations

Once a model has evolved to a certain level of completeness, it may be useful to save this version of the model as a frozen Iteration. A frozen Iteration captures a version of the model as is. It cannot be modified anymore, but it can be used for reference and comparison.

Within the scope of CDP4-COMET an Iteration is basically a snapshot of the complete model that can be created on any required time. By comparing Iterations an assessment can be made of the improvements by the team in reaching an adequate solution. Depending on the time, several of these Iterations may be performed to further enhance the model and explore the feasibility of different design options in more detail. These design iterations can then be checked to see if they fulfil the given set of Requirements.

10.10 Checking Budgets

Once all parameters have been given first values, we can create a first version of various budgets. A budget is an overview of the performance of the entire system. Budgets can be created on various aspects, such as costs, mass, power, data but also availability, technology readiness, etc. To display a budget, we need to create a budget template (see 13). Creating a budget template requires some programming skills. Go to the Report browser on the Model tab. Many editing screens will open, you can ignore these. To open an existing budget, click the Open Report button on the Reporting – Designer tab in the left upper corner. Browse to the report template and open it. In the right upper corner, choose the Preview tab (marked red in fig. 15). Next click the Rebuild Data Source button to reload the report with the latest model data (marked red in fig. 15). You can now see the report.

If the report has parameters, you can fill these out and submit (marked red in fig. 15) them to visualize the report with the given parameters. You can hide the screen where you submit the parameters using the parameters button(marked red in fig. 15).

In the preview tab it is possible to save, export or print the Report in various ways.

If the report calculates system level values, and these are connected to parameters in the model, it is possible to submit these back to the model. For this, after rebuilding the model and inspecting it, use the Parameter Values - Submit button at the right upper corner, next to the Rebuild Data Source button.

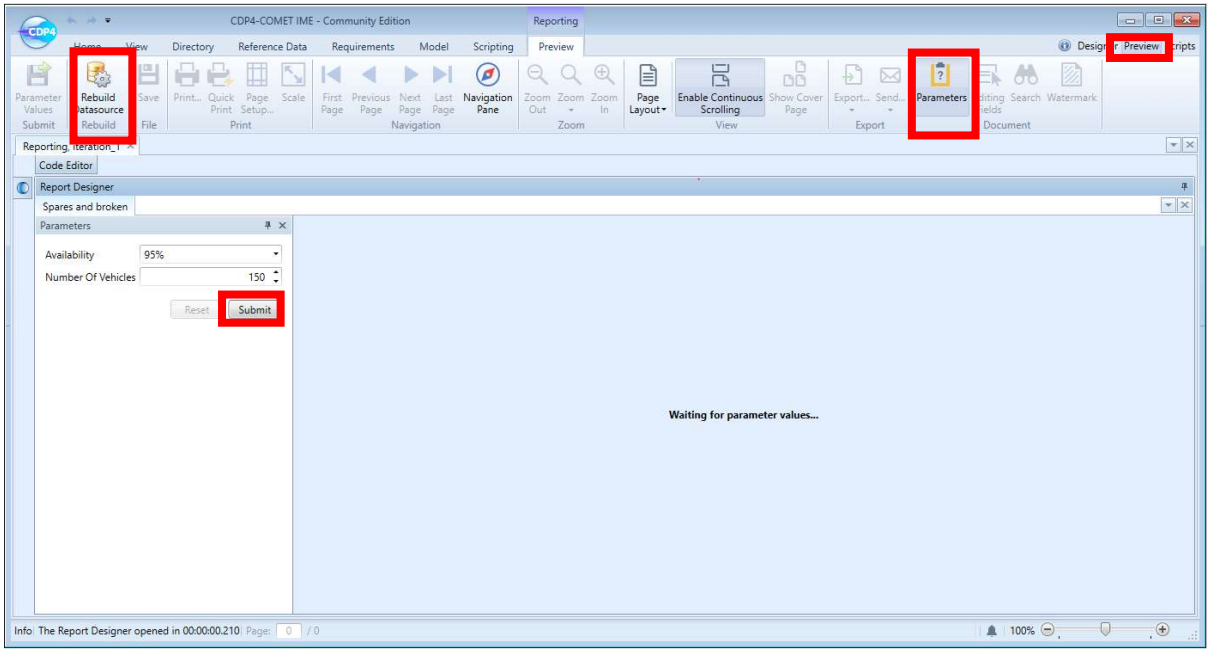


Figure 13: Viewing a Report – Provide additional reporting parameters

10.11 Dashboard

In the Preliminary Engineering the core team identifies key design drivers, parameters that have a large impact on the solution space. It may be interesting to monitor and visualize the evolution of these design drivers during the design process. Are costs moving up or are they decreasing. Is the mass reduced compared to last Concurrent Design session, etc. For this we can customize a dashboard. Open the Dashboard using the button in the Model tab. Drag the parameters that you want to monitor from the Product Tree to the Dashboard screen. This is easiest when you open the Product Tree as a vertical tab group using the right mouse click option for this.

The parameter will now be visualized on the dashboard with an indication of its previous value and the percentage it changed. If you double click the tile of the parameter, a graph will visualize the way the value changed over time. You can add multiple parameters in the dashboard view to create an overview of Design Drivers. (see 9.1 on Conceptual Modelling)

10.12 Graphic representations

To visualize the Product Tree a Graphic representation is available in the Grapher. To view the Graphic Diagram, click on the Grapher button in the Model tab. On the right side you see the Product Tree visualized as a diagram (see 9.4 about the Product Tree). When you click an element in the tree, the actual values of the Parameters are listed on the right side of the screen. This view can be used to present or inspect the model.

To create relations in a graphic way, you can use the Relationships Editor. Click the Relationships Editor button in the Model tab. Existing relation types are listed at the right. In the middle, an empty diagram view is visible. Open the Product Tree next to the Relationships Editor (right mouse click and

select Create New Vertical Tab Group). Now, drag and drop the elements that you want to relate on the Relationships Editor. They will appear as blocks. Select the right relationship type and click the source Element and next the target Element to create the relation. The relation is now also visible in the Relationship matrix (see 10.8 for Relationship Matrix).

10.13 Verification

To verify the consistency of the model, rules can be created that define how components in the model should be consistently related. Examples of rules are that verify that elements categorized as equipment can only be found below sub-system level. Another example is that all equipment should have a mass parameter. Such rules can be defined and verified using the verification tools described below.

In de Build in Rules browser we can find the rules applicable for the model. This is used as a reference source for domain experts that need to adhere to the rules.

In the errors list we find an overview of all instances where aspects are not modelled according to the rules defined or incorrect in the implementation according to ECSS-E-TM-10-25A.

In the rules verification we can create a rule verification list. It serves to provide an overview of all instances where aspects are not modelled according to the user rules defined (see 6.7). This will help the System Engineering to maintain consistency and to structurally verify the model against these rules specified. To do this, create a Rule Verification List, add any required user rule as Rule Verification and choose whether to set it to active or not. The Rule Verification List can then be executed and will check all the active user rules in the list. The results are given below each Rule Verification entry in the list.

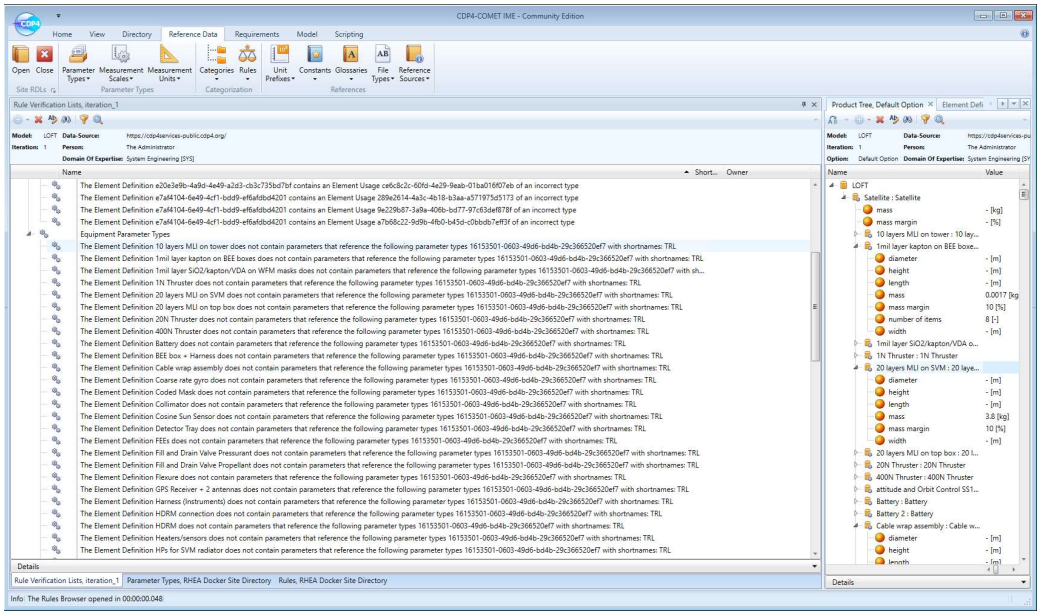


Figure 14: Executed Rule Verification List

11. CDP4-COMET EXCEL WORKBOOKS

11.1 Calculations in Excel

The CDP4-COMET workbooks are the Microsoft Excel workbooks in which relevant domain specific calculations can be made, providing a powerful environment for modelling and calculations together with e.g. the report editor, scripting engine, and interfaces to/from Domain Specific Tools (DSTs) such as CATIA or Matlab. The CDP4-COMET workbooks often are an integral part of the activity and the Concurrent Design process. CDP4-COMET supports Microsoft Excel 2013, 2016 and higher files.

CDP4-COMET workbooks have added functionality which makes it possible to incorporate them into CDP4-COMET.

The CDP4-COMET workbooks can be managed in domain file stores and stored on a connected data source (CDP4-COMET servers).

In this section an introduction is given on the setup and management of CDP4-COMET Workbooks.

11.2 Opening an Excel workbook

When installing the CDP4-COMET app with regular setting you automatically install the EXCEL plug-in. To ensure the plug-in works, make sure the CDP4-COMET version is compatible with your MS EXCEL version (32 or 64 bits).

When you open Excel, use an empty workbook and connect to the CDP4-COMET server with the identical icon as in the CDP4-COMET app.

In your workbook a tab will load with parameter values (parameters sheet), and one for each option in the CDP4-COMET model (option sheet). Further, there are empty calculation sheets.

In your workbook only your own values and the values you subscribe to will load. If you need parameters from CDP4-COMET in your workbook that are not listed, you need to first subscribe to these parameters in the CDP4-COMET app.

In the Excel workbook you can change the values of parameters and make calculations using the values from the model.

Besides these key functionalities, it is possible to inspect many aspects of the CDP4-COMET model using the same functions and buttons as described above.

11.3 General Setup of and Functionality

CDP4-COMET workbooks can contain three different types of sheets

- Parameters sheet
- Generated Option sheets
- Calculation sheets

These workbooks have added functionality which makes it possible to incorporate and use them in CDP4-COMET. When CDP4-COMET is installed, a CDP4-COMET ribbon is added to Microsoft Excel through the CDP4-COMET Office Add-in. Any normal Microsoft Excel workbook can be turned into a CDP4-COMET workbook. This is done by logging in and rebuilding the workbook for a particular engineering model, representing a particular domain. The Excel workbook does not automatically refresh, instead make sure you submit new data and rebuild the database to download new data from the CDP4-COMET server.

Parameters and subscriptions are available and can be edited through these workbooks. Several interactions between the engineering model and the workbooks are possible as well, e.g. subscribing to parameters or deleting existing subscriptions.

11.3.1 Parameters Sheet

The Parameters sheet of a workbook are necessary for CDP4-COMET workbooks to work properly. This sheet is the actual interfaces with the rest of the system. This sheet allows the values of parameters and subscriptions, including all their corresponding possible combinations of overrides, option-dependencies and state-dependencies, to flow into and out of a workbook.

This sheet is automatically rebuilt by CDP4-COMET every time a user requires this. The rebuild action will be performed based on the latest available information on the CDP4-COMET server. Predetermined columns should be edited by the users to reflect the changes in their domain. When these columns are edited these changes still need to be communicated to the CDP4-COMET server, otherwise the next rebuild will overwrite these changes. Editing the other columns of the Parameters sheet is of no use, the changes are not communicated to the CDP4-COMET server and will therefore be lost with the next rebuild. The user will get a notification when trying to edit these columns if the sheet is locked for editing.

In the parameter sheet the values represent the element definitions. When making calculations it is advised to use the data from the option sheets. Here you find the values from the element usages, and these values could contain overrides of the element definitions as well as option dependent

values, which could be different from the element definition values. Using the data from the options sheet is therefore recommended.

11.3.2 Generated Option Sheets

Within an engineering model, options can be created to explore different solution directions. From the various modelling concepts, such as including or excluding element definitions from specific options and differentiating values for parameters by setting option dependency or using overrides, a resulting CDP4-COMET Product Tree can be calculated for each specific option. These Product Trees are given in the CDP4-COMET Workbook after rebuilding, representing the subset of the engineering model that is relevant for and used by a domain. Given are the owned parameters and subscriptions with their resulting values corresponding to their place within the Product Tree. Their place is made clear by inclusion of information on their containing design objects, starting from the top element (which is the only element definition in any Product Tree) going recursively down the tree through its underlying element usages.

In this way, the generated option sheets provide a way for the user to access the required design information as specified by him that is connected to a particular option. This makes this information accessible in an understandable way, as a basis for further processing in calculations or different representations. All the information that is represented on the generated option sheets is given as-is on the moment it has been generated. This information cannot be edited. Changes to the underlying structure as well as to the values that are given on this sheet are managed by the appropriate browsers, used either from within the CDP4-COMET Workbook or in the IME, or through the Parameters sheet. To reflect these changes on the generated option sheets, these will have to be updated.

When changing a parameter value in Excel, make sure to change the switch accordingly and then submit the change to be published on the CDP4-COMET server.

11.3.3 User-defined Calculation Sheets

The Parameters sheet and the generated Options sheets are thus getting their information from the CDP4-COMET server or other connected data source, and can always be restored again. To perform Concurrent Design, users can add any number of additional Excel worksheets as Calculation sheets in their CDP4-COMET4™ Workbook as domain modelling space.

Any changes made to the Calculation sheets in the CDP4-COMET Workbook, including adding new sheets, are only stored in the CDP4-COMET workbook if these are stored as a physical file in the CDP4-COMET server using the domain file store, so changes to these will be lost if the workbook is not saved and uploaded in the correct way or is deleted. Workbook versions can alternatively be managed by another file management or versioning system to keep the correct files available for their corresponding engineering model.

To create a calculation, you can use the Cell name of the CDP4-COMET value in your calculation sheet like you would use a cell variable in excel. The result of your calculation can be named. Select the cell with the result and in the left upper corner you can change the label of the cell from the automatic label (e.g. ‘C12’) to a self-chosen variable name (e.g. Batery.Mass.Calc) . Next you can go to the

parameter you want to assign this value and select the variable from the drop-down list. Do not forget to adjust the switch and submit the value.

11.4 Requirements sheet

Besides the ability to browse the requirements, it is also possible to create a requirement sheet. The requirement sheet contains an overview of requirements including name, definition and simple parameter values. In this way a structured overview of the requirements can be generated.

To create and rebuild the requirements sheet use the requirements button and select sheet instead of the browser.

12. Scripting Engine

Multiple computations can be done in CDP4-COMET using scripts. The module Scripting Engine offer you a development environment specially for that purpose. It is for example possible to manipulate an Engineering Model Setup to calculate its total mass.

The scripting engine module can be found in the view tab. It's possible to open several script files or create a new script. Every script panel contains its own text editor, output console, input console and a panel to see the actual local variables. A script panel can be associated to a session , it's notably compulsory to retrieve data from CDP4-COMET.

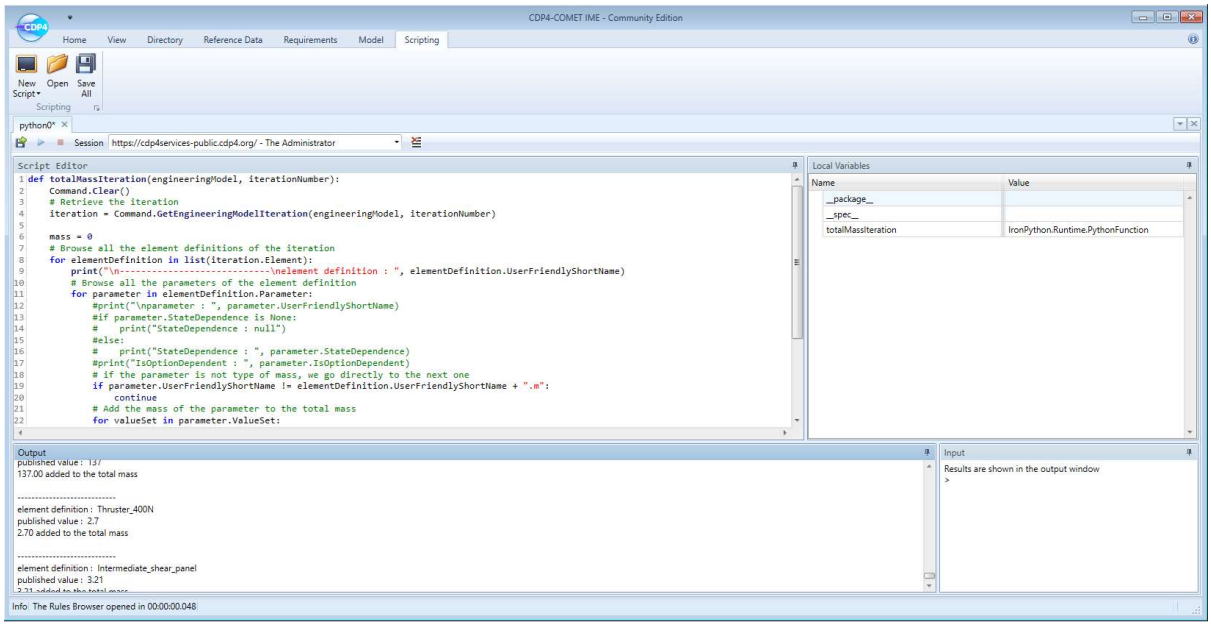


Figure 15: Scripting Engine - Python

The scripting engine contains commands that will help you to interact with it, to clean your development environment or to get data from CDP4-COMET for example. Besides some CDP4-COMET classes and methods are available in the script.

A command has to be used as follow:

`Command.CommandName(parameters)`

`Command.Help()`: List the commands available.

`Command.Clear()`: Clears the output.

`Command.ClearVariables()`: Clears the local variables.

These are the main commands that will help you to get started. The languages supported in the script engine are Python 3.7 and Lua 5.x. Note that Lua Script is not supported.

13. Reporting and Creating Budgets

CDP4-COMET provides the user with the capability to create custom reports, these reports provide an overview based on the data that has been created by a Concurrent Design team. Examples of reports are a Mass Budget and a Power Budget. All ECSS-E-TM-10-25 data that is loaded in the model is available.

13.1 Creating a Report

To open the Reporting browser, select the Reporting icon on the Model tab. Apart from the browser itself, an additional Reporting tab is added to the ribbon to provide access to the extended functionality in this browser.

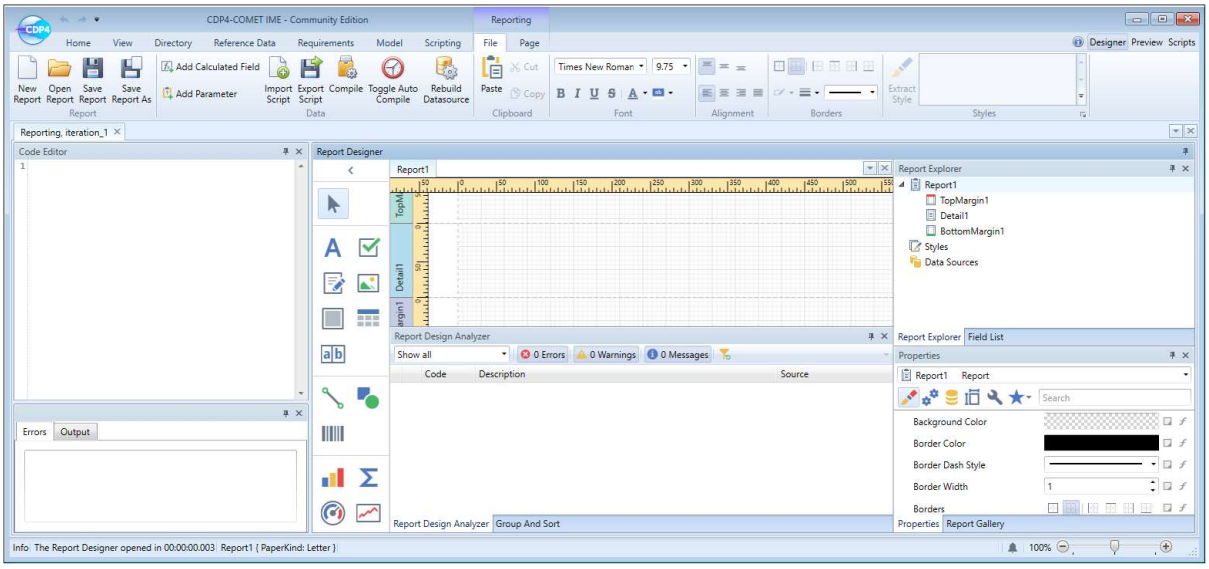


Figure 16 Reporting browser

The Reporting Engine consists of the following parts:

- Code Editor and Status (typically used for data collection)
- Report Designer
- Report Explorer and Properties

The Reporting engine can be used in 3 different modes, to switch between the modes, use the tabs in the upper right-hand corner of the reporting engine. The 3 modes are listed below:

- Designer: Use the designer to create the template of the report
- Preview: Preview the result of the report
- Scripts: Add scripts and formulas for automation of the report

Report Definitions can be created from scratch, saved to disk, or loaded as existing reports. To create a new report from scratch, select the New Report button. Saved report definitions (report templates) have the ".rep4" extension. Resulting reports can be exported to various file formats including PDF.

The Report Designer comes with a built-in script editor that provides end-users with the advanced capability to write and execute scripts at runtime when a report is generated, see the following link: Script Editor in End-User Report Designer for Web | .NET Reporting Tools | DevExpress Documentation

13.2 The Code Editor

The Code Editor provides the entry point to work with the data available in the engineering model. Using C#, the user can create a dedicated model and use this as needed for purpose from the data in the engineering model, e.g. to retrieve parameters from nested elements with specific categories. To setup the code for a report, specific functions and classes are available to access the data in the engineering model. Specific helpers can be created to make this data more easily accessible and available to users in the reporting tool, as is done e.g. for the data in the Option trees. More information on these functions and classes can be found on GitHub at GitHub - STARIONGROUP/COMET-Reports: COMET Report definitions and templates that are used by the COMET Reporting plugin

Several reports are available here as well that can serve as examples or starting points for custom reports.

The Code Editor part of the Reporting plugin contains two panels:
1. Code Editor: used to write C# code
2. Error and Output panel: shows the results of code compilation and the result of (re)building a report's data source using the compiled code.

Once the code is setup, use the Compile icon on the tab to compile the code. The code will be automatically compiled when you open an existing report. The status of this action, together with any

errors, are indicated in the status box. Compilation can also be performed automatically by using the "Toggle Auto Compile" button. When the toggle is in an "On" state, the code will be compiled automatically when the code in the Code Editor is changed.

Click on the Rebuild Datasource button to have the results of the code editor to be reflected in the Report Designer. The code in the code Editor will be compiled and the compiled code will be executed, resulting in a refreshed Field List panel in the Report Designer. The reporting engine automatically executes "Rebuild" when a report is opened. When designing a report, don't forget to Rebuild the Datasource when changes have been made.

13.3 Report Designer

With the setup of a model in the code editor for the report, the next step is in the report designer. This provides a 'canvas' where a report can be (gradually) build up, adapted, refined and extended

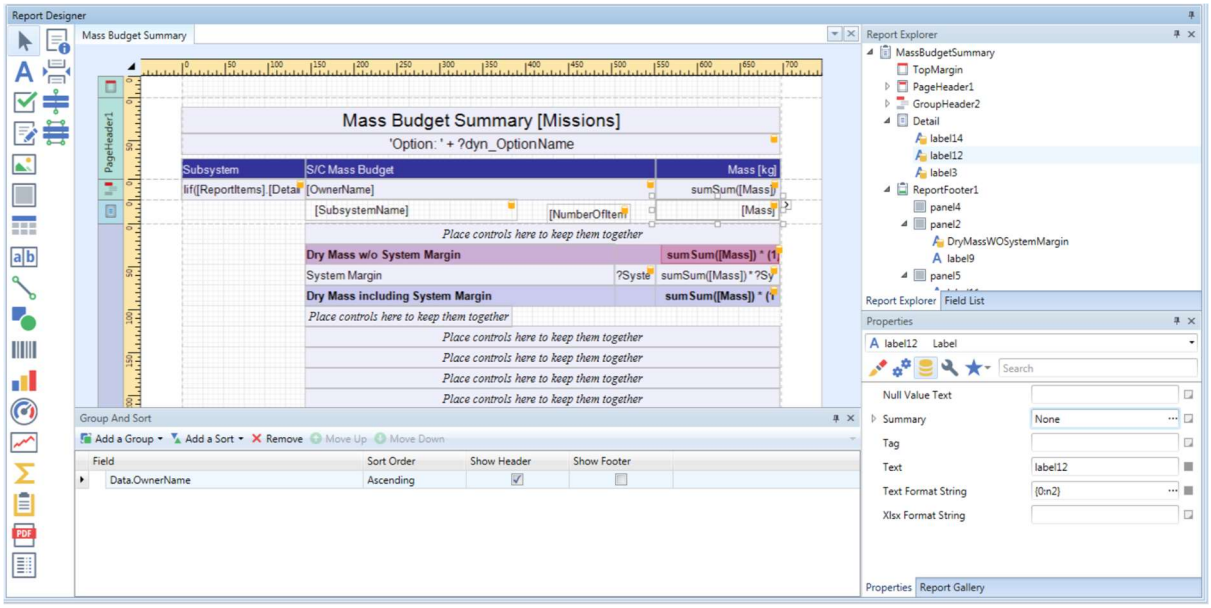


Figure 17 Report designer

The reporting browser makes use of DevExpress, with the available functionality. This includes the possibility to setup your report with a header area, a footer area, and any required structure in the middle. For this, bands can be defined and included. For the overall setup, labels can be used to provide information in the report as required. The required modelling information as defined in the code is available in the report designer in a so-called data source. This data source needs to be created or updated (to get intermediate changes that are made to the engineering model) using the Rebuild DataSource icon. The information can now be used in the report setup.

For (repeating) items in the engineering model, a template setup can be defined here, using the fields that are created for the report as defined in the code editor. The information to be retrieved in this way is usually defined by the (hierarchical) structure of the model, provided by the application of categories.

The structure given in the report designer is then used to fill the actual report dynamically with all the modelling items as found in the engineering model that adhere to the provided structure. In this way it is possible to e.g. collect specific parameters from target categorized element usages, such as "mass" or "number of items" for all subsystems in the model, given in the code by the field "SubsystemName". To use a field in the report, use drag and drop to place a field in the required position in the report.

Each separate box in the report has an additional information dialog that can be activated by clicking the arrow icon on the top right of the box that is currently selected. By clicking the three dots "..." an expression editor dialog opens with additional possibilities to setup custom calculations for this box in the report, using the available fields and a variety of standard operators and functions.

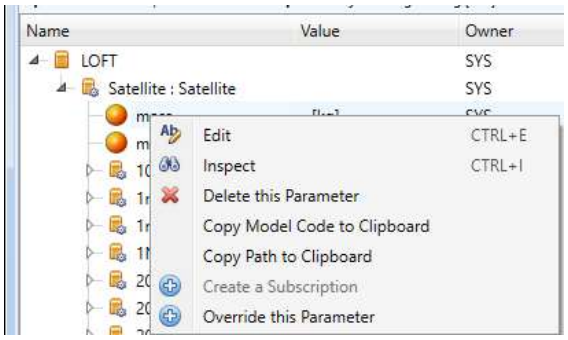
These building blocks are then used to gradually build up a custom report to provide an overview to users of any data of interest. The possible inclusion of various types of charts or figures can make these reports even more informative, insightful and useful. Reports can be saved and adapted as needed, using the Save Report or Save Report As icons on the Reporting tab.

13.4 Linking results back to the model

It is possible to create a link between (calculated) results from the report back into the engineering model. The easiest way to create links between a result in the report and a parameter in the engineering model is by creating a direct link through the report designer. To do this, copy the path of the required parameter by right mouse clicking the product tree using the Copy Path to Clipboard menu item. This path can then be linked to a field label in the report. To do this, select the required label. In the Properties box, select the Data tab. The link is created through the property "Tag": insert the following text:

- "path="
- followed by the copied path (without any spaces)

This method provides a "hard-coded" link, specific for that modelling item in the path. It is also possible however to provide a linking mechanism more dynamically through code in the code editor, making the setup of the report more agile and independent on the actual content (in terms of names of the modelled items), making use instead of the (fixed) backbone structure and categories to provide results at a specific level of interest, making the budget setup more robust.



After running the report, any updated parameter values available in the report can be submitted by using the Submit Parameter Values icon on the Preview tab. This icon is only active when such a link exists. Note that this can only be done for parameters that are accessible by the participant running

the report, i.e. for owned parameters. If this is not the case, a message for this is given, and these values will not be submitted. Any value that can be submitted will be send however. The values for any linked parameters that have changed in value are submitted to the CDP4-COMET server as "computed" value. Please note that if the switch setting of such a linked parameter is not set to computed this means that the parameter will not show up in the publication browser.

14. Licences, Feedback and Contributions

14.1 Licences

The CDP4-COMET-IME Community Edition is provided to the community under the GNU Affero General Public License. The CDP4-COMET Community Edition relies on open source and proprietary licensed components. Some of these components have a license that is not compatible with the GPL or AGPL. For these components Additional permission under GNU GPL version 3 section 7 are granted. See the license files for the details.

The RHEA Group also provides the CDP4-COMET Web Services Enterprise Edition which comes with commercial support and more features. https://www.rheagroup.com/contact/ Contact for more details.

14.2 Feedback

When you have ideas to improve CDP4-COMET, or find issues in working with the software, you can report these on GitHub: Issues · STARIONGROUP/COMET

Other feedback can be sent to us via the Contact page.

14.3 Contributions

Contributions to the code-base are welcome. The CDP4-COMET-IME Community Edition make use of the GitHub - STARIONGROUP/COMET-SDK However, before we can accept your contributions we ask any contributor to sign the Contributor License Agreement (CLA).

Index