POLITECNICO
DI TORINO

Politecnico di Torino

Corso di Laurea Magistrale in Ingegneria Aerospaziale

Implementazione del Concurrent Engineering nel processo di progettazione di Space V tramite il software CDP4-CoMET.

Relatore:
Prof. Paolo Maggiore

Co-relatore:
Ing. Franco Malerba
Prof. Patrizia Bagnerini
Ing. Piero Messidoro

Candidato:
Dario Pezzella

A.A. 2024/2025 - Sessione di Laurea Aprile 2025

Sommario

L'evoluzione del progetto di missione e sistemi spaziali e l'adattamento dei processi ingegneristici al contesto, sempre più diffuso, di cooperazione tra Piccole e Medie Imprese comporta la necessità di strutturare e gestire i processi di condivisione delle informazioni progettuali. In questo lavoro di Tesi si analizzano i principi del Concurrent Engineering e come questi possano essere implementati nelle attività di progettazione. In particolare, si presentano le raccomandazioni fornite nell'ECSS-E-TM-10-25A, riguardanti lo scambio di dati ingegneristici basato sui modelli, e l'approccio del Model Based System Engineering. Dunque, si valuta l'applicazione del Concurrent Engineering nel settore spaziale tramite alcuni software sviluppati in questo ambito, individuando in CDP4-CoMET lo strumento adeguato per una gestione efficace e consistente delle informazioni di progetto, attraverso la condivisione di modelli ingegneristici, e per garantire la collaborazione simultanea dei professionisti coinvolti. Si presentano, quindi, le funzionalità del software selezionato, descrivendo nel dettaglio come queste possono essere sfruttate per creare degli spazi virtuali di lavoro dedicati agli utenti. Tra le potenzialità di CDP4-CoMET infatti, risulta fondamentale la possibilità di differenziare l'accesso ai dati, adeguando le funzionalità del software stesso alle responsabilità ricoperte dall'utente. Inoltre si presentano tutti i concetti e le procedure necessarie per la creazione di un modello ingegneristico, ovvero una rappresentazione dello studio di progettazione. Successivamente, si descrive la configurazione del software e la sua implementazione nelle attività di progettazione di Space V, startup che opera nel settore del progetto di sistemi spaziali. L'obiettivo di questo processo, che ha coinvolto direttamente i membri dell'azienda, è massimizzare l'aderenza delle funzionalità del software alle esigenze di progettazione e di cooperazione di Space V, nel contesto delle Piccole e Medie Imprese del settore spaziale. Si descrive la realizzazione di due diversi modelli ingegneristici tramite il software CDP4-CoMET: il primo modello, più semplice e generico, utile come punto di partenza per lo svolgimento dello studio di fattibilità della missione spaziale di interesse di Space V, e il secondo modello, che consiste in uno studio preliminare di progetto della medesima missione, il quale, sfruttando in maniera più completa le funzionalità del software, possa essere un riferimento per future valutazioni progettuali da parte dell'azienda. Infine il lavoro di Tesi si conclude con un processo di generalizzazione del caso di studio di Space V. Nell'ultimo capitolo, infatti, si astrae l'approccio adottato per il processo di implementazione del Concurrent Engineering tramite strumenti di supporto informatico, al fine di individuare una metodologia più generale per l'applicazione di questi principi nell'ambito della cooperazione tra Piccole e Medie Imprese coinvolte nel progetto di missioni e sistemi spaziali. Partendo da un'analisi del contesto di lavoro nel quale interagiscono le PMI, si sottolineano le caratteristiche uniche di questo ambiente e le differenze principali con grandi imprese e agenzie spaziali, per individuare i benefici e le criticità che derivano dall'adozione dei principi del Concurrent Engineering tramite software di scambio dati basati su modelli ingegneristici.

Indice

Elenco delle figure 4

Elenco delle tabelle 6

Lista delle Abbreviazioni 7

1 Introduzione al Concurrent Engineering e alla sua applicazione 11
1.1 Concurrent Engineering . . . 13
1.2 Engineering Design Model Data Exchange . . . 15
1.3 Model Based System Engineering . . . 17
1.4 Stato dell'arte dei Data Exchange Tool . . . 19

2 Funzionalità del software CDP4 - CoMET 23
2.1 Directory . . . 25
2.1.1 User Management . . . 25
2.2 Reference Data . . . 31
2.2.1 Parameter Types . . . 33
2.2.2 Categorization . . . 35
2.3 Requirements . . . 36
2.3.1 Simple Parameter Values . . . 38
2.3.2 Parametric Constraints . . . 40
2.3.3 Import & Export dei Requirements . . . 41
2.4 Model . . . 41
2.4.1 Engineering Model . . . 44
2.4.2 Ulteriori Funzionalità . . . 49
2.5 Scripting e integrazioni con altri software . . . 54
2.5.1 Scripting . . . 55
2.5.2 CDP4-CoMET Web App . . . 56
2.5.3 Excel Add-In . . . 57
2.5.4 Integrazione con Matlab . . . 58

3 Implementazione del software CDP4 - CoMET nelle attività di progettazione di Space V 61
3.1 Studio di Fattibilità . . . 62
3.1.1 Directory . . . 63

3.1.2 Reference Data Library 66
3.1.3 Template Model 72
3.1.4 Requirements 74
3.2 Studio di Spazializzazione dell'Adaptive Vertical Farm 76
3.2.1 Study Model 76
3.2.2 Relazioni e tracciamento dei requisiti 77
3.2.3 Scripting 78
3.2.4 Reporting 85

4 Generalizzazione del caso di studio di Space V per l'integrazione del CE nel processo di progettazione delle Piccole e Medie Imprese 89
4.1 Analisi del contesto operativo 90
4.2 Benefici e criticità nell'integrazione del software 93

5 Conclusioni 99

Elenco delle figure

1.1 International Space Station Adaptive Vertical Farm Proof of Concept. . . 12
1.2 Analisi qualitativa dei benefici dell'applicazione del Concurrent Engineering [1] . . . 14
1.3 Analisi qualitativa delle sfide nell'implementazione del Concurrent Engineering [1] . . . 15
1.4 Diagramma della struttura del SEIM [2] . . . 17
1.5 Architettura di un generico Exchange Data Tool [1] . . . 17
1.6 Esempio di architettura operativa di un prodotto realizzabile tramite Capella. 19
1.7 Interfaccia utente del software Atlas, implementato come estensione di Miscrosoft Excel. . . . 20
1.8 Schermata del software Virtual Satellite[3]. . . . 21

2.1 Finestra per l'autenticazione al server pubblico di test. . . . 24
2.2 Toolbar della sezione Home. . . . 24
2.3 Toolbar della sezione Directory. . . . 25
2.4 Schema delle informazioni che definiscono un utente. . . . 26
2.5 Diritti di accesso del ruolo Concurrent Design Team Member. . . . 28
2.6 Diritti di accesso del ruolo Observer. . . . 30
2.7 Esempio di visualizzazione dei partecipanti ad un modello ingegneristico tramite la funzionalità Team Composition. . . . 31
2.8 Toolbar della sezione Reference Data. . . . 32
2.9 Toolbar della sezione Requirements. . . . 37
2.10 Toolbar della sezione Requirements. . . . 38
2.11 Possibili esiti di verifica di una Parametric Constraint. . . . 41
2.12 Finestra di creazione di Engineering Model Setup. . . . 43
2.13 Toolbar della sezione Model. . . . 43
2.14 Finestra di definizione degli elementi. . . . 44
2.15 Finestra di modifica dei parametri. . . . 45
2.16 Finestra di creazione delle Actual Finite States List. . . . 48
2.17 Finestra dedicata alle Publications. . . . 49
2.18 Finestra di gestione delle Relationships. . . . 50
2.19 Dashboard della massa del sistema. . . . 52
2.20 Interfaccia dedicata al Report Design. . . . 52
2.21 Toolbar per la configurazione delle pagine dei Report. . . . 53

2.22 Interfaccia del Relationships Editor. 54
2.23 Scripting. 55
2.24 Visualizzazione di metodi e attributi nel tooltip. 56
2.25 Model Dashboard della Web App. 57
2.26 Toolbar di Excel dedicata a Comet. 57
2.27 DEHP Matlab Adapter. 59

3.1 Engineering Model Setup per il Template Model. 64
3.2 Esempio di creazione di un partecipante. 65
3.3 Configurazione completa del Template Model. 65
3.4 Creazione del parametro Requirement Verification Method. 66
3.5 Valori possibili per il parametro Requirement Verification Method. 67
3.6 Creazione della categoria High Level per i Requirements. 68
3.7 Creazione della Rule Requirements High to Design Level Decomposition. . 68
3.8 Creazione della Rule Subsystem Parameter Types. 72
3.10 Requirement Specification. 74
3.9 Diagramma a blocchi del Template Model prodotto dal Grapher. 75
3.11 Estratto della matrice di decomposizione dei requisiti di alto livello prodotta da Comet. 78
3.12 Visualizzazione dei Requirements prodotta tramite script. 79
3.13 Esempio di diagramma N^2 generato. 84
3.14 Struttura del Mass Budget nel Report Designer. 86
3.15 Anteprima del Mass Budget Summary Report con dati oscurati dal software. 87

4.1 Ricorrenze nella letteratura scientifica di dichiarazioni di benefici del MBSE correlate con misurazioni[4]. 94

Elenco delle tabelle

2.1 Valori ammissibili per i permessi d'accesso nei Person Role 27
2.2 Valori ammissibili per i permessi d'accesso nei Person Role 28
2.3 Descrizione dei vari tipi di modelli ingegneristici 42

3.1 Parametri introdotti nella Generic Space V Reference Data Library per la valutazione delle prestazioni della camera di crescita. 71
3.2 Costanti introdotte nella Generic Space V Reference Data Library. 71

Lista degli Acronimi

AGR	Agronomics
AS	Active Standards
ASI	Agenzia Spaziale Italiana
AVF	Adaptive Vertical Farm
BIC	Business Incubation Center
CDF	Concurrent Design Facility
CDH	Control & Data Handling
CE	Concurrent Engineering
COTS	Commercial Off-The-Shelf
DEH	Digital Engineering Hub
ECSS	European Cooperation for Space Standardization
EQLR	Equipment Level Requirement
ESA	European Space Agency
HC	Harness and Connections
INCOSE	International Council on System Engineering
IP	Internet Protocol
IRL	Integration Readiness Level
ISS	International Space Station
LDAP	Lightweight Directory Access Protocol
LMN	Illumination
MBSE	Model Based System Engineering

MDL	Middle Locker
MDO	Multidisciplinary Design Optimization
MFA	Autenticazione Multi Fattore
MLR	Mission Level Requirement
MS	Measurement Scale
MU	Measurement Unit
NASA	National Aeronautics and Space Administration
OCDT	Open Concurrent Design Tool
OIDC	OpenID Connection
PMI	Piccole e Medie Imprese
PNRR	Piano Nazionale di Ripresa e Resilienza
PoC	Proof of Concept
PWR	Power
RDL	Reference Data Library
ReqIF	Requirement Interchange Format
RS	Requirements Specification
SDK	Software Development Kit
SEIM	System Engineering Information Model
SERDL	System Engineering Reference Data Library
SEN	Sensors
SH	Shelf Handling
SI	Sistema Internazionale
SLR	System Level Requirement
SML	System Maturity Level
SSL	Secure Socket Layers
SSLR	Subsystem Level Requirement
STR	Structures

SYS	System Engineering
SysML	System Modeling Language
THE	Thermal
TM	Technical Memorandum
TRL	Technology Readiness Level
UML	Unified Modeling Language
VEN	Ventilation
WM	Water Management
WSL	Windows Subsystem for Linux

Capitolo 1

Introduzione al Concurrent Engineering e alla sua applicazione

Il progetto di missione e sistemi spaziali rappresenta uno dei campi più complessi dell'ingegneria moderna, caratterizzato da un continuo rinnovamento tecnologico e spronato dall'ostilità del contesto operativo. Il settore spaziale coinvolge una vasta gamma di discipline, dall'ingegneria meccanica all'elettronica, dalla fluidodinamica all'informatica e il progetto ingegneristico deve garantire un'elevata affidabilità e sicurezza dei sistemi in ambienti estremi, come i carichi di lancio, il vuoto spaziale, le escursioni termiche e le radiazioni cosmiche. Inoltre, data la ridotta scalabilità delle tecnologie di ascesa, risulta di fondamentale importanza valutare attentamente la gestione delle risorse a bordo dei veicoli spaziali e la sostenibilità delle missioni stesse. Oltretutto, le missioni spaziali sono sottoposte a vincoli economici e temporali stringenti, il che promuove l'adozione di metodologie che riducono costi e tempi del processo di progetto di missione, senza compromettere la qualità. Sebbene, in passato, questo settore sia stato dominio esclusivo delle grandi agenzie governative, oggi, nelle opportunità dell'economia spaziale, si inseriscono anche aziende private e Piccole e Medie Imprese (PMI), contribuendo alla crescita dell'ecosistema tecnologico ed economico e allo sviluppo di servizi essenziali per la società, come le comunicazioni satellitari, la navigazione GPS, il monitoraggio ambientale e la gestione dei disastri naturali.
Osservando l'evoluzione del settore spaziale negli ultimi anni, si notano alcune tendenze di grande rilievo che stanno ridefinendo il modo di concepire e realizzare le missioni. Ad esempio, l'ingresso di aziende private come SpaceX, Blue Origin e Rocket Lab ha dato impulso a una nuova era del settore spaziale, incentrata sulla riduzione dei costi di accesso allo spazio [5]. Simultaneamente, l'adozione di approcci agili e l'utilizzo di componenti commerciali standardizzati, Commercial Off-The-Shelf (COTS), ha ridotto significativamente i tempi di sviluppo delle missioni. Di conseguenza, la rinnovata accessibilità all'ambiente spaziale sta richiamando l'interesse di organizzazioni che, altrimenti, non avrebbero ambito a questo settore.
Le PMI stanno assumendo un ruolo di rilevanza crescente [6], collaborando con tutte le

entità del settore, nello sviluppo di soluzioni innovative. Questo modello di collaborazione permette di unire risorse, competenze e prospettive diverse, favorendo la sperimentazione scientifica, lo sviluppo e l'adozione di tecnologie, la raccolta di dati ma anche la formazione e l'educazione.
In questa Tesi di Laurea Magistrale si vuole analizzare il progetto di missione e sistemi spaziali nel contesto delle PMI, con l'obiettivo di implementare la filosofia del Concurrent Engineering (CE) nelle attività di progettazione di Space V.
Space V è una startup nata a Genova nel 2021 grazie a cinque fondatori, tra cui il primo astronauta italiano Ing. F. Malerba, e coinvolta nel programma dell'European Space Agency (ESA) Business Incubation Center (BIC) presso l'incubatore di impresa I3P del Politecnico di Torino. La startup si occupa della progettazione di sistemi modulari dedicati alla coltivazione di vegetali edibili direttamente in contesti spaziali, ed è proprietaria di un brevetto relativo a un sistema di coltivazione verticale terrestre che sfrutta algoritmi di adattività dei volumi delle camere di crescita per incrementare notevolmente la resa [7].
Nella figura 1.1 è mostrato il Proof of Concept (PoC) della Adaptive Vertical Farm (AVF) pensato per dimostrare l'operatività sulla International Space Station (ISS) o altre stazioni orbitanti e presentato in occasione dell' International Astronautical Congress 2024 tenuto a Milano. L'azienda, nell'ambito dello studio di fattibilità e spazializzazione del sistema terrestre, opera a stretto contatto con diverse PMI del settore spaziale: in questo contesto avere un approccio solido e sistematico alla collaborazione risulta fondamentale nella gestione e nella coordinazione delle attività.

Figura 1.1. International Space Station Adaptive Vertical Farm Proof of Concept.

1.1 Concurrent Engineering

Il Concurrent Engineering è una filosofia di progettazione volta alla parallelizzazione e l'integrazione simultanea delle diverse fasi di sviluppo ingegneristico di un prodotto. Le prime formulazioni del CE compaiono negli anni '80 [8], quando settori ad alta tecnologia, come l'aerospaziale e l'automobilistico, iniziarono a implementare i primi sistemi informatici e individuare strategie per ridurre i tempi di sviluppo aumentando al contempo la qualità dei prodotti.
Tradizionalmente, la gestione dei progetti prevede una successione sequenziale delle attività di sviluppo, per cui i team di progettazione trasmettono i risultati delle loro attività solo a valle del processo. Sebbene questa metodologia *a cascata* sia stata largamente consolidata in contesti ad alta complessità, la sua sequenzialità produce alcune limitazioni, legate alla rigidità del metodo e alla poca responsività agli errori, criticità che comportano tempi e costi di rilavorazione impattando negativamente sull'intero ciclo di vita del prodotto.
Il CE si fonda su un insieme di principi chiave[1] che interessano tutti gli aspetti delle attività ingegneristiche. Tra questi, possiamo considerare:

- la coordinazione dei team di esperti nelle varie discipline coinvolte e la condivisione delle informazioni progettuali in sessioni di lavoro simultanee, incrementando la responsività del processo e riducendo l'insorgenza di errori e problemi di integrazione.

- la parallelizzazione delle fasi di sviluppo e validazione consentendo un'individuazione precoce delle criticità del progetto. Inoltre, tramite il coinvolgimento preliminare dei vari stakeholder, come clienti e produttori, si ha una riduzione complessiva del time-to-market.

- l'utilizzo di strumenti di supporto informatico per la gestione e la condivisione di dati in modo solido e strutturato e l'integrazione degli strumenti di modellazione e simulazione, già diffusamente adoperati in ambito ingegneristico, rendendo il processo più fluido ed efficiente.

Questo approccio alla condivisione e parallelizzazione del progetto rende il CE una delle filosofie di riferimento per le agenzie spaziali e le aziende del settore. La riduzione di errori, incomprensioni e discrepanze produce un miglioramento della consistenza e della qualità del processo di progettazione e una conseguente riduzione dei costi e tempi di sviluppo, favorendo così l'efficienza economica del processo stesso. Nella seguente figura 1.2, sono riportati alcuni benefici individuati in un'analisi qualitativa dell'utilizzo del CE.

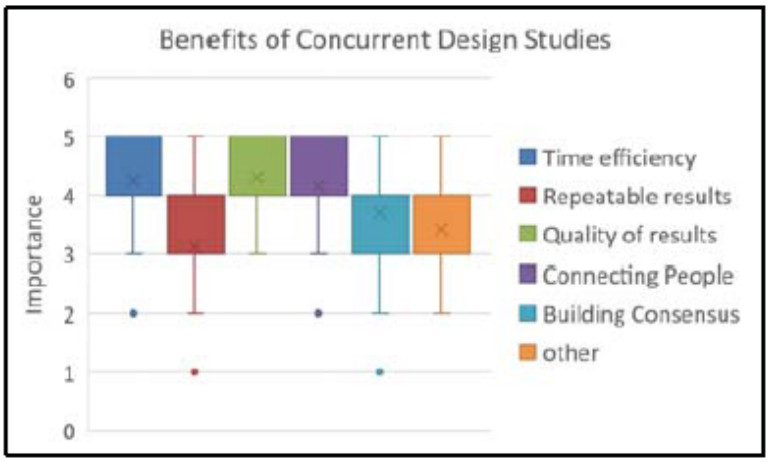

Figura 1.2. Analisi qualitativa dei benefici dell'applicazione del Concurrent Engineering [1]

Le agenzie spaziali, come la National Aeronautics and Space Administration (NASA) e l'ESA, sono state pioniere nell'adozione del CE, riconoscendone i vantaggi strategici per la gestione di progetti complessi, sfruttando ulteriormente l'utilizzo di strumenti di supporto informatico, permettendo, quando necessario, la collaborazione di team remoti. Un esempio significativo dell'applicazione del CE è rappresentato dalla Concurrent Design Facility (CDF) dell'ESA [9], una struttura dedicata alla progettazione collaborativa in cui esperti in ambiti differenti lavorano simultaneamente su un progetto, utilizzando strumenti avanzati per la modellazione e la simulazione. Analogamente, la NASA ha istituito il Goddard Space Flight Center Integrated Design Center [10], che consente di sviluppare concept di missioni spaziali in tempi ridotti grazie all'integrazione immediata delle competenze specialistiche.
Sebbene queste facilities siano di grande importanza, l'attuazione del CE non sempre può, o deve, prevedere la disposizione di infrastrutture fisiche. Infatti, l'implementazione del CE nelle PMI comincia spesso dall'adozione dei sistemi di supporto informatico sia per conformarsi alle procedure delle grandi agenzie spaziali, con cui spesso collaborano, sia per sfruttare a loro volta i benefici della filosofia. Tuttavia, l'applicazione con successo di questi principi non è sempre scontata. Considerato il ruolo centrale della collaborazione, scaturiscono aspetti socio-tecnici dall'interazione, nello spazio di lavoro, di obiettivi tecnici e relazioni umane [11].
Sul piano organizzativo, l'implementazione del CE richiede un cambiamento culturale nelle aziende e nelle agenzie spaziali, che devono adottare un approccio più flessibile e collaborativo rispetto ai modelli tradizionali. Questo cambiamento implica un investimento in formazione e sviluppo delle competenze, affinché gli ingegneri e i project manager possano sfruttare appieno le potenzialità del CE.
Inoltre, l'adozione di strumenti informatici comporta la valutazione di ulteriori aspetti, sebbene già ampiamente diffusi nel contesto ingegneristico moderno, come la sicurezza dei dati, la condivisione di dati sensibili e la gestione della proprietà intellettuale. Garantire

la protezione delle informazioni riservate senza compromettere l'efficienza della collaborazione rappresenta una sfida cruciale, soprattutto nel settore spaziale, caratterizzato da elevati livelli di competizione, requisiti di sicurezza stringenti e talvolta implicazioni strategiche. I risultati di un'analisi qualitativa delle criticità legate all'adozione del CE sono riportati nella figura 1.3 di seguito.

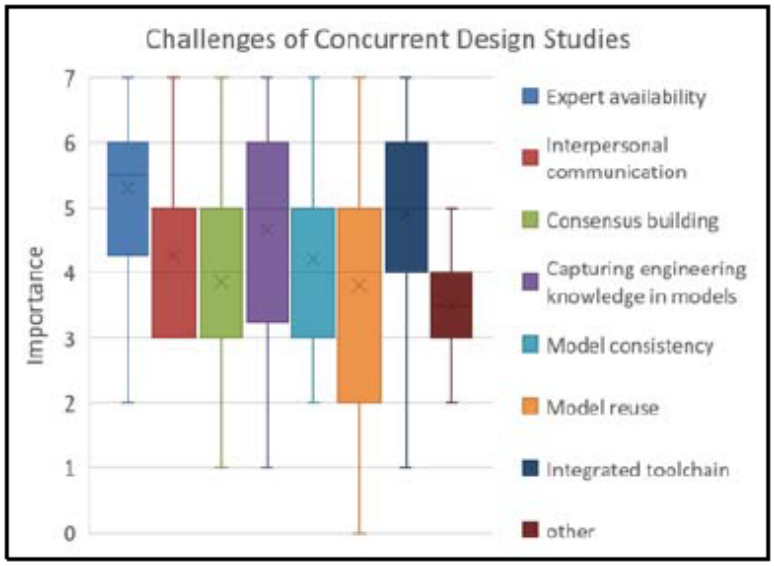

Figura 1.3. Analisi qualitativa delle sfide nell'implementazione del Concurrent Engineering [1]

1.2 Engineering Design Model Data Exchange

È stato analizzato come il CE, per applicare i suoi principi in maniera solida, debba avvalersi di strumenti di supporto informatico per la condivisione dei dati. Questo produce una serie di requisiti dal punto di vista informatico, come ad esempio la gestione solida e consistente dei dati, la responsività dell'architettura del software o la regolazione dei diritti di accesso degli utenti. Dato il crescente interesse nell'implementazione del CE manifestato dai vari enti coinvolti, a partire dagli anni 2000 sono stati sviluppati diversi software che potessero supportare lo scambio di informazioni ingegneristiche. In questo contesto si inserisce l'European Cooperation for Space Standardization (ECSS) formulando in un Technical Memorandum (TM) le raccomandazioni per lo sviluppo di strumenti informatici per lo scambio di dati tramite modelli ingegneristici.

ECSS è un'iniziativa che coinvolge tutte le organizzazioni dei Paesi dell'Unione Europea che operano nel settore spaziale, con l'obiettivo di creare degli standard unici e coerenti che concernono tutte le attività dell'industria spaziale. Questa struttura standardizzata comporta notevoli benefici nell'ambito del progetto di missioni spaziali, poiché omologa la qualità del processo ingegneristico ad alti livelli e promuove la cooperazione tra le organizzazioni . Gli standard vengono formulati in documenti chiamati, appunto, Active Standards (AS) e hanno un valore normativo per gli enti che operano nel settore

spaziale europeo. Gli AS vanno distinti dai TM, questi ultimi infatti individuano delle raccomandazioni, dunque hanno solamente uno scopo informativo e spesso sono utilizzati per la costruzione del consenso, in preparazione alla formulazione di un AS.

Nel documento ECSS-E-TM-10-25A [2], intitolato Engineering design model data exchange, vengono definite le raccomandazioni per lo scambio dati basato su modelli ingegneristici, relativamente alle fasi iniziali (Fase 0 e A) del progetto di missione spaziale. Nel TM viene presentato il System Engineering Information Model (SEIM) e la System Engineering Reference Data Library (SERDL), fornendo, inoltre, una serie di indicazioni sull'architettura del software e sull'utilizzo dei modelli ingegneristici in fase di progettazione. In particolar modo, nel SEIM viene definita la struttura delle classi che dovrebbero comporre il software. In ambito informatico, e in particolare nella programmazione a oggetti, una classe consiste in una rappresentazione di un concetto astratto. Nella descrizione del SEIM quindi, vengono introdotti una serie di concetti per la gestione delle attività di Concurrent Design, come:

- i modelli ingegneristici, rappresentazioni gerarchiche di sistemi dal punto di vista funzionale e fisico. Permettono di svolgere valutazioni ingegneristiche sulle fasi di missione, i modi del sistema e le diverse opzioni di configurazione;
- la definizione e gestione degli utenti tramite attribuzione di ruoli e discipline di competenza;
- la definizione di parametri fisici o concetti ingegneristici, utili per la progettazione.

Il SEIM si configura quindi come un'architettura informatica, a partire dalla quale è possibile sviluppare un software. In figura 1.4 è riportata una schematizzazione di questa struttura. Poiché i TM non hanno valore normativo, le case di sviluppo possono comunque implementare le funzionalità secondo la loro discrezione, ma l'aderenza alla struttura proposta favorisce la compatibilità tra gli ambienti di lavoro e questo è un aspetto importante nell'ottica di cooperazione e scambio di informazioni. Nella figura 1.5 è rappresentata una generica architettura per un software di questo genere.

Ulteriormente, nella formulazione della SERDL, viene illustrato come molte informazioni individuate nel SEIM vadano incluse in librerie da cui attingere in fase di progettazione, per esempio la struttura dei Concurrent Design Parameters. Nell'Annex B del TM viene inoltre fornita una libreria di riferimento contenente le informazioni genericamente utili al progetto di missione spaziale. Ad esempio, sono già definite le discipline tipicamente coinvolte, come System Engineering o Propulsion, e i parametri fondamentali e derivati del Sistema Internazionale (SI).

L'integrazione tra SEIM e SERDL rappresenta un elemento chiave per il consolidamento delle metodologie di Concurrent Engineering nel settore spaziale. La definizione di un'infrastruttura condivisa per la gestione dei dati ingegneristici non solo migliora l'efficienza dei processi progettuali, ma favorisce anche una maggiore affidabilità delle decisioni tecniche e una riduzione dei costi associati alla gestione delle informazioni. In definitiva, l'adozione di tali standard costituisce un passo cruciale verso l'implementazione di un paradigma ingegneristico sempre più integrato e collaborativo, in linea con le esigenze delle moderne missioni spaziali.

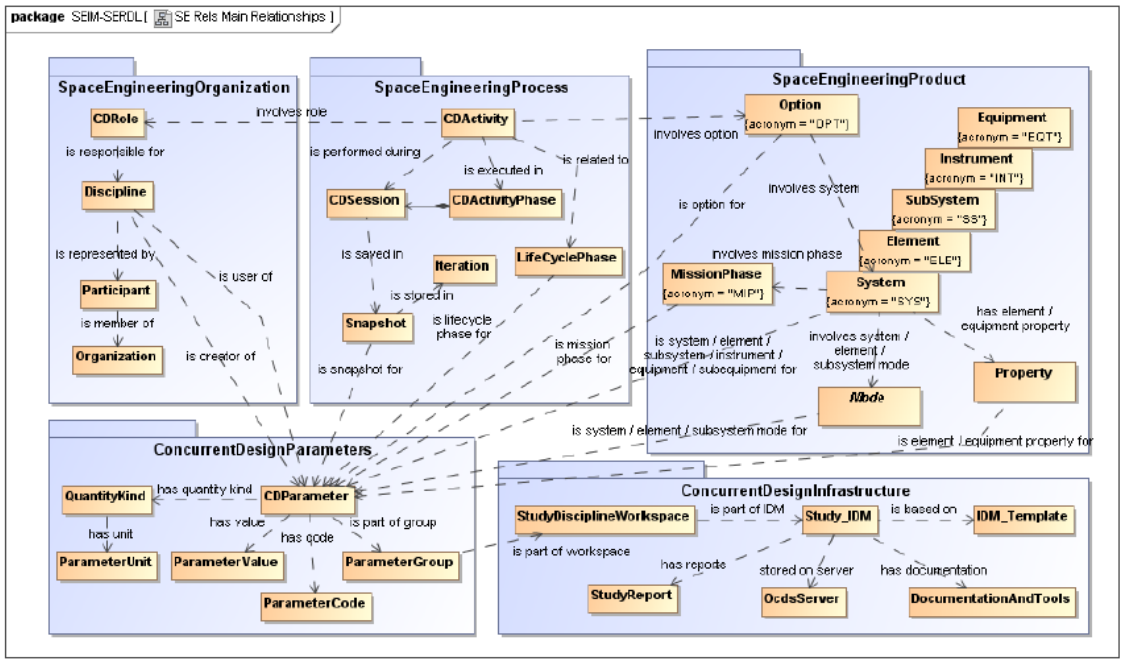

Figura 1.4. Diagramma della struttura del SEIM [2]

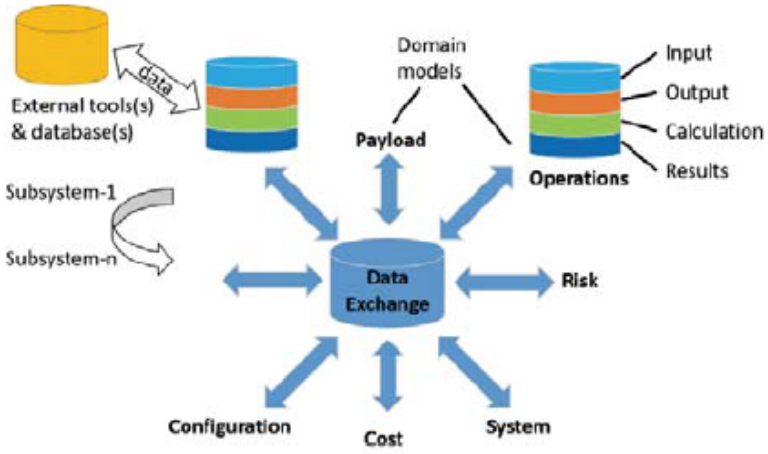

Figura 1.5. Architettura di un generico Exchange Data Tool [1]

1.3 Model Based System Engineering

Il Model Based System Engineering (MBSE) è un metodo di progettazione che prevede l'utilizzo di modelli ingegneristici per supportare le fasi di definizione, analisi, produzione

e validazione di un sistema complesso. Un modello ingegneristico consiste in una rappresentazione concettuale e semplificata di un progetto, utile a facilitare la gestione della complessità e migliorare la comprensione del progetto stesso. Il livello di dettaglio che caratterizza un modello ingegneristico deve dunque essere sufficientemente alto affinché sia significativo, ma non eccessivamente da comprometterne la comprensibilità. Un modello *completo*, e quindi totalmente aderente al sistema in esame, ne presenterebbe la medesima complessità, perdendo dunque la sua utilità.
Sebbene l'utilizzo di modelli sia ampiamente diffuso e consolidato in ambito ingegneristico, si pensi ai modelli matematici, strutturali, o CAD, l'impiego di modelli ingegneristici che affianchino la progettazione in tutti i suoi aspetti comincia negli anni '90 con le prime formulazioni di MBSE. Dalla sua comparsa, il MBSE è stato progressivamente vagliato in vari ambiti ingegneristici, ma si ha un'ulteriore formalizzazione e popolarizzazione dell'approccio da parte dell'International Council on System Engineering (INCOSE) [12].
In passato, l'ingegneria dei sistemi si è basata su documenti testuali e rappresentazioni grafiche per specificare i requisiti e definire le architetture dei sistemi. Questo approccio, sebbene consolidato, presenta alcune criticità, tra cui la possibilità di incoerenza tra documenti, la difficoltà nella gestione e nel tracciamento delle evoluzioni del progetto. Il MBSE si propone di superare queste limitazioni tramite la creazione di un modello consistente che possa essere usato come fonte di riferimento da tutti i team multidisciplinari coinvolti nelle attività di progettazione, per una gestione delle informazioni più dinamica e agevole.
L'implementazione del MBSE nelle fasi di progettazione avviene tramite l'utilizzo di strumenti informatici, come linguaggi di programmazione per la modellazione di sistemi e software a più alto livello. Alcuni linguaggi di modellazione, comunemente utilizzati in diversi ambiti ingegneristici, sono l'Unified Modeling Language (UML), nonostante sia stato sviluppato nell'ambito del software engineering, oppure il System Modeling Language (SysML). Questi linguaggi di modellazione permettono di rappresentare gli aspetti fisici, strutturali, funzionali o comportamentali di un sistema, permettendo lo svolgimento di numerose valutazioni progettuali.
Un esempio di piattaforma software, di alto livello, per la creazione di modelli ingegneristici è invece Capella [13], sviluppato inizialmente da Thales nel 2007. Questo software sfrutta i due linguaggi precedentemente introdotti per creare un ambiente di lavoro ad alto livello in cui il progettista può sfruttare le caratteristiche dei linguaggi di modellazione dei sistemi, con un'interfaccia utente ergonomica. In figura 1.6 è riportata, a scopo esemplificativo, una schematizzazione dell'architettura operativa di un prodotto.
I benefici del MBSE, in termini di solidità e accessibilità delle informazioni progettuali e la gestione più efficace dei dati stessi tramite elaborazioni dinamiche e verifiche automatiche, sinergizzano con i principi del CE. Infatti, se il CE è un insieme organico di approcci sistematici, il MBSE risulta essere uno di questi approcci. I modelli ingegneristici agevolano lo scambio di informazioni e la cooperazione tra i team di progetto, preservando la qualità delle informazioni e dunque la qualità del progetto stesso. Per queste ragioni gli strumenti informatici sviluppati per l'implementazione del Concurrent Engineering spesso sfruttano l'approccio MBSE e dunque lo studio dei sistemi tramite la formulazione di modelli ingegneristici.

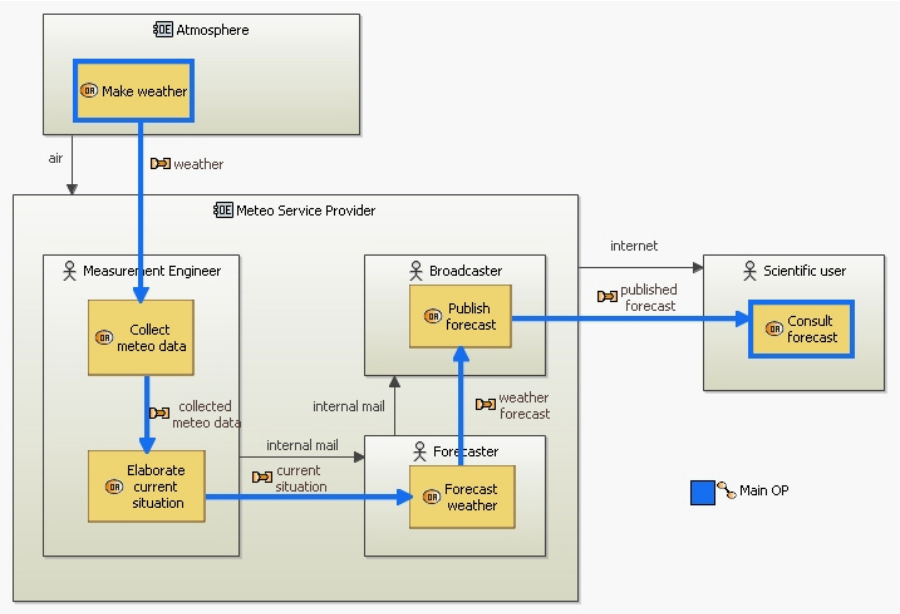

Figura 1.6. Esempio di architettura operativa di un prodotto realizzabile tramite Capella.

1.4 Stato dell'arte dei Data Exchange Tool

Si vogliono analizzare le principali strategie adottate nello sviluppo di questi strumenti informatici per il supporto dello scambio di dati. A partire dai concetti e le raccomandazioni introdotte nel TM analizzato in precedenza (ECSS-E-TM-10-25A), sono stati sviluppati alcuni software, come Atlas, Virtual Satellite, Open Concurrent Design Tool (OCDT) e CDP4-CoMET, che si distinguono negli approcci utilizzati nell'implementazione.
Partendo dagli aspetti in comune, la necessità di differenziare lo scambio di informazioni tra gli utenti a seconda del ruolo che svolgono nelle attività di progettazione, comporta l'utilizzo di un sistema di autenticazione. L'utente quindi dovrà identificarsi, solitamente tramite delle credenziali, al fine di accedere alle informazioni di un determinato progetto. Inoltre, le raccomandazioni fornite nella descrizione del SEIM delineano la gestione dei modelli ingegneristici, prevedendo, ad esempio, la decomposizione dei sistemi in sottosistemi, dispositivi e così via o la valutazione dei modi operativi del sistema.
La differenza principale nella realizzazione di questi software risiede nella metodologia di interazione tra gli utenti ed il database. Queste due soluzioni per l'architettura del software sono state individuate anche da ECSS, che infatti delinea due strategie possibili:

- offline exchange, che consiste sostanzialmente nell'esportazione e importazione di set di dati;

- online exchange, tramite l'utilizzo di un web service che risponda alle richieste degli utenti.

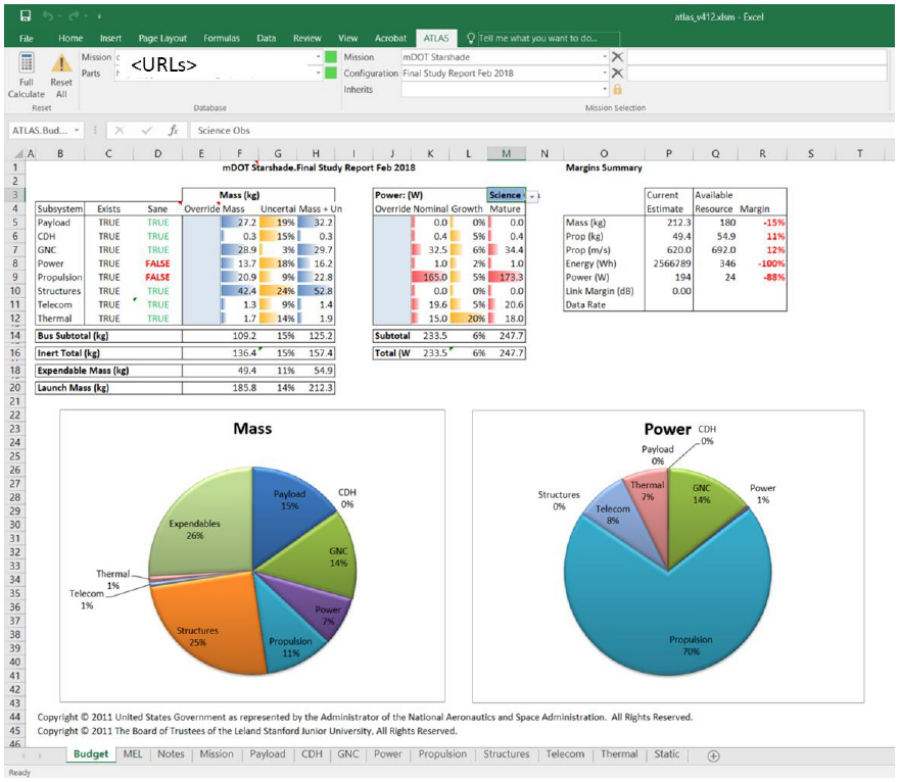

Figura 1.7. Interfaccia utente del software Atlas, implementato come estensione di Miscrosoft Excel.

Tra i primi software realizzati in quest'ambito ci sono Atlas e OCDT[14], sviluppati rispettivamente dalle agenzie spaziali NASA ed ESA. Entrambi questi software non sfruttavano un client dedicato bensì erano stati concepiti come estensione del software Excel (Figura 1.7), di cui integravano le funzionalità di calcolo e di gestione dei dati. Tuttavia, mentre OCDT implementava l'approccio online exchange, sfruttando un web service per il front-end e PostgreSQL per il back-end, Atlas implementava l'approccio offline exchange, utilizzando un sistema di commit delle attività di Concurrent Design a un database centrale. Ad ogni modo, lo sviluppo, e utilizzo, di entrambi questi software è stato accantonato: la mancanza di un client dedicato complicava lo sviluppo delle funzionalità e inoltre le prestazioni del software deperivano velocemente all'aumentare della complessità dei sistemi modellati . Di conseguenza, la NASA si è adoperata allo sviluppo di un nuovo software chiamato Poseidon [15], mentre l'ESA ha adottato il software open source CDP4-CoMET.
CDP4-CoMET [16] è un software sviluppato da Starion Group ed è considerato il diretto successore di OCDT, poiché ne replica piuttosto fedelmente l'architettura. Anche Comet infatti utilizza un web service per la gestione delle richieste degli utenti, e un client dedicato, oltre ad un'estensione di Excel che ne integra le funzionalità. Il software è sviluppato tramite il linguaggio di programmazione C# e supporta la creazione di plugin tramite i

linguaggi C#, Python e TypeScript.
Un'ulteriore alternativa attualmente disponibile, anch'essa open source, è Virtual Satellite [17], creata dal Deutsches Zentrum für Luft-und Raumfahrt, agenzia spaziale tedesca. Questo software è sviluppato tramite il linguaggio di programmazione Java ed è basato su Eclipse [18]. Le funzionalità di Virtual Satellite, di cui alcune raffigurate in figura 1.8 si concentrano sul MBSE mentre gli aspetti di gestione e controllo di versione del database sono affidati a software esterni come Git o Subversion (SVN), sfruttando quindi un sistema di *commit* delle sessioni di lavoro, dove i dati vengono inviati o ricevuti tramite transazioni tra l'utente e il database centrale.

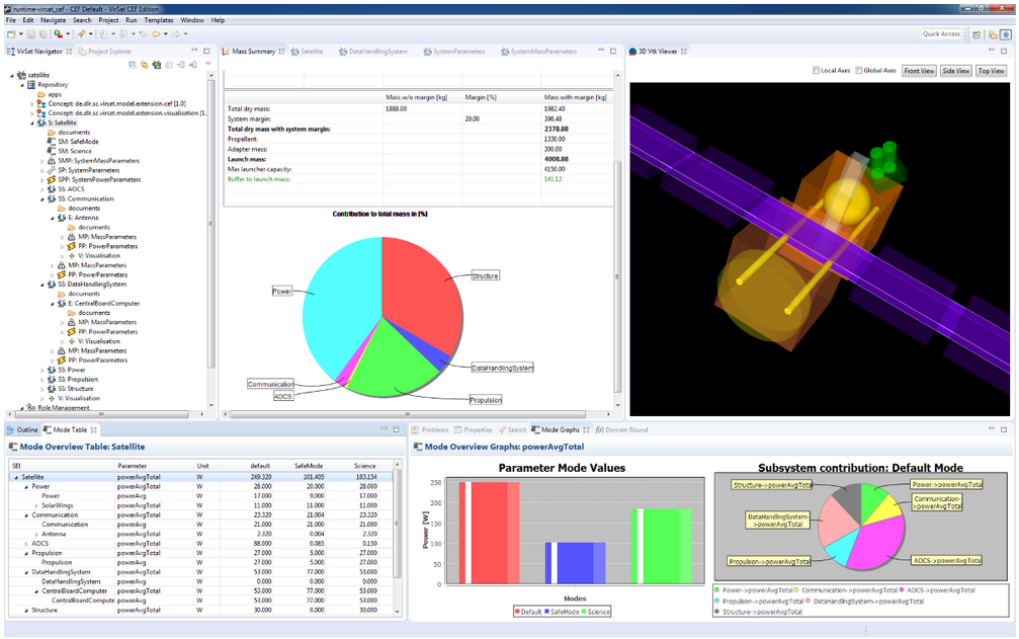

Figura 1.8. Schermata del software Virtual Satellite[3].

Dopo aver esaminato vari software possibili, si è deciso di utilizzare CDP4-CoMET per lo scopo di questa Tesi per una serie di motivi. Innanzitutto questo software, come molti in questo campo, è open source, il che significa che il codice stesso del software è messo a disposizione, in chiaro, per chiunque lo voglia analizzare. Questo certifica la trasparenza del software stesso e la possibilità di poterne valutare il funzionamento ai livelli più profondi. Ulteriormente Comet, tramite la sua architettura che sfrutta un web service per l'elaborazione delle richieste da parte dei client, presenta una responsività molto elevata, permettendo la collaborazione simultanea di un elevato numero di utenti e la condivisione dei dati in tempo reale. Infine, anche l'interesse manifestato dall'ESA [19] per questo specifico software è stato considerato nella scelta; infatti, nell'ottica di adottare uno strumento che permetta la collaborazione, la crescente diffusione dell'utilizzo di questo software ne incrementa la rilevanza.

Capitolo 2

Funzionalità del software CDP4 - CoMET

CDP4-CoMET (Comet) è un software ingegneristico per lo scambio di dati basato su modelli. È stato creato dalla casa di sviluppo software Starion nel 2017 seguendo fedelmente il TM dedicato dall'ECSS, ed è tuttora in continuo aggiornamento. Come già anticipato, il software presenta un'architettura server-client: gli utenti possono utilizzare l'applicazione client[20] per accedere alle funzionalità offerte dal server[21], che risponde alle richieste degli utenti e gestisce il database. Questo tipo di architettura è molto comune nei software di scambio dati poiché permette di parallelizzare le operazioni di numerosi utenti verso un singolo database, che dunque rimane unico e consistente. Ulteriormente, a seconda del tipo di rete utilizzato per la connessione tra server e client, le funzionalità possono essere erogate su una rete locale, limitata a una certa area, ad esempio un ufficio, oppure estese all'accesso da remoto tramite Internet.
L'obiettivo del software è di creare degli spazi di lavoro virtuali dove i professionisti coinvolti nella progettazione possano cooperare proporzionalmente alle loro mansioni e competenze. Al fine di attribuire correttamente i ruoli preposti agli utenti, il software sfrutta un sistema di autenticazione. Nella figura 2.1 è riportata la finestra per l'accesso al server, utilizzando la Source Address del server pubblico fornito dalla casa di sviluppo del software per svolgere test o esplorare le funzionalità. Utilizzando l'indirizzo di un server e le credenziali, nome utente e password, l'utente può accedere a un workspace dedicato, tramite il quale leggere o modificare le informazioni di progetto di sua competenza.

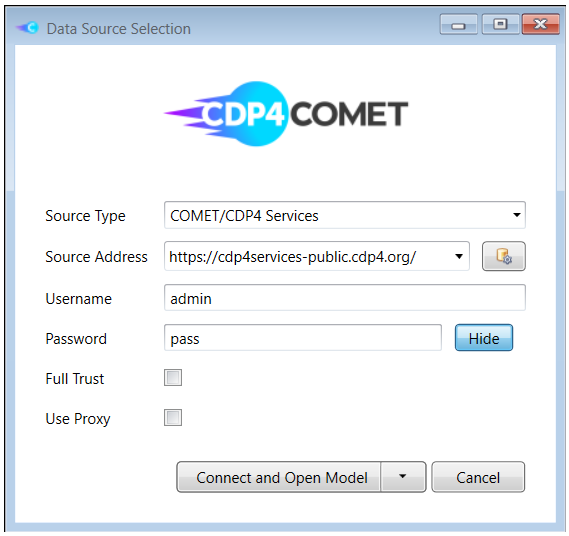

Figura 2.1. Finestra per l'autenticazione al server pubblico di test.

Un aspetto molto importante della struttura del software consiste nella gestione delle informazioni su due livelli, ovvero:

- Site Level: contiene le informazioni più generiche del software, come la gestione degli utenti o le librerie di riferimento. A questo livello sono contenute informazioni che potrebbero essere associate ad uno o più progetti, dunque maggiormente legate all'organizzazione e al settore nel quale questa agisce.
- Model Level: contiene le informazioni specifiche di un progetto. Tutto ciò che può essere definito a questo livello è riferito esclusivamente al modello in considerazione.

Si vogliono scomporre e analizzare tutte le funzionalità del software, integrando la descrizione fornita dal manuale utente [22] con le informazioni ottenute dall'utilizzo del software stesso. Al fine di svolgere questa valutazione, saranno riportate diverse schermate provenienti direttamente da Comet, sfruttando l'interfaccia utente predefinita, per esaminare le funzioni associate ai pulsanti proposti. Questa disposizione può comunque essere personalizzata sulle esigenze dell'utente, creando delle interfacce dedicate che riportano esclusivamente le funzionalità desiderate. Aprendo il client, viene automaticamente mostrata la scheda Home all'utente.

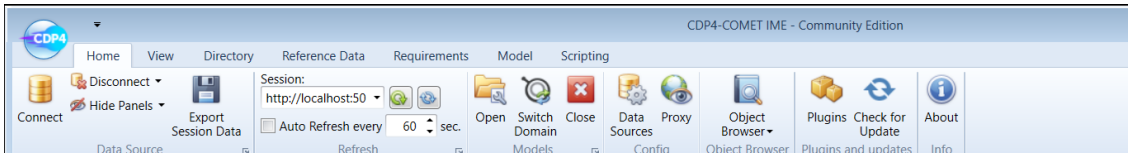

Figura 2.2. Toolbar della sezione Home.

La toolbar disposta in questa schermata (figura 2.2) fornisce funzionalità per la gestione del software. Il pulsante più importante in questa schermata è Connect da utilizzare per svolgere l'autenticazione a un server, utilizzando la finestra precedentemente riportata in figura 2.1. Una volta effettuato il login, diventa possibile l'utilizzo delle altre funzionalità della toolbar. È possibile gestire la sessione attiva, individuata dalle credenziali dell'utente e dalla Source Address, e richiedere manualmente l'aggiornamento dei dati del client o impostare un tempo di richiesta automatico. Inoltre, in questa schermata si possono gestire i modelli aperti e recuperare informazioni relative al funzionamento del software come le Source Address memorizzate, in Data Source, la versione installata o i plugin presenti.

2.1 Directory

Nella scheda Directory è possibile gestire le informazioni generali per il funzionamento del software. In questa schermata infatti si può accedere, da un punto di vista gestionale, a funzionalità come la creazione di modelli ingegneristici o utenti. Nella figura 2.3 è riportata la barra degli strumenti di questa schermata, in cui notiamo la presenza di due sezioni, la prima chiamata nuovamente Directory e la seconda per lo User Management.

Figura 2.3. Toolbar della sezione Directory.

La sezione Directory presenta funzionalità generiche su Models, Reference Data Libraries e Natural Languages. In questa sezione è possibile inizializzare nuovi modelli e librerie, che possono essere ulteriormente caratterizzati e utilizzati nelle sezioni dedicate. Sebbene la creazione di nuovi modelli ingegneristici avvenga tramite la finestra associata al pulsante Models di questa toolbar, la configurazione di nuovi dei modelli ingegneristici verrà descritta nella sezione dedicata (sezione 2.4). Relativamente ai Natural Language, è possibile definire una lingua ordinaria, utile nel caso in cui si vogliano riproporre in diversi linguaggi, come in italiano e inglese, le varie definizioni introdotte nel software.
Per quanto riguarda lo User Management, la possibilità di creare gli utenti e gestire i loro diritti di accesso risulta essere una delle funzionalità centrali del software, quindi viene analizzata in una sezione dedicata.

2.1.1 User Management

La sezione dello User Management permette la creazione degli utenti, rappresentazione di persone fisiche, riportando informazioni. Ulteriormente, il conferimento delle credenziali

per accedere al server ne abilita l'utilizzo dell'utente: fino all'attribuzione delle credenziali, il profilo non può essere utilizzato e ha solo valore indicativo.
Oltre alle informazioni relative alla persona, come email o numero di telefono, si ha la possibilità di definire i Domini e attribuire questi ultimi agli utenti. I Domini sono un aspetto molto importante nella gestione delle informazioni da parte del software, poiché rappresentano la disciplina di competenza dell'utente e definiscono dei veri e propri workspace differenti, nei quali ogni team di progettazione può operare in maniera indipendente, svolgendo le proprie elaborazioni di dati.
Inoltre, è possibile definire delle Organizzazioni e associarle agli utenti al fine di fornire ulteriore contesto alla definizione del profilo. Inoltre, tramite la possibilità di individuare un'organizzazione proprietaria del modello, si può gestire l'accesso di una specifica organizzazione alle informazioni di ogni elemento del modello. Questo aspetto sarà approfondito nella valutazione delle funzionalità relative al modello.
Infine la definizione di un utente si completa con l'assegnazione dei ruoli, che avviene su due livelli separati: Person Role e Participant Role. A questi due tipi di ruoli sono associati dei set di diritti di accesso, che regolamentano l'interazione dell'utente con tutte le funzionalità implementate.

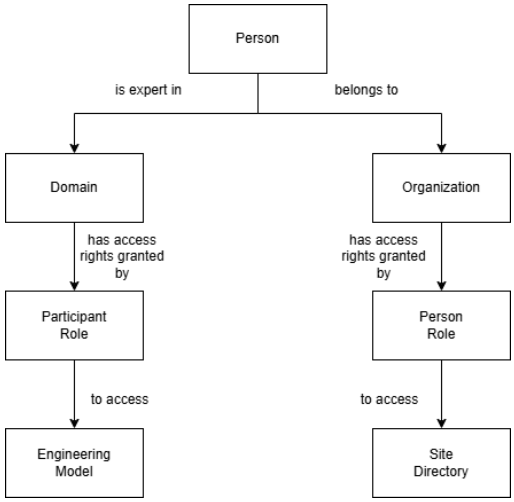

Figura 2.4. Schema delle informazioni che definiscono un utente.

Person Role

I Person Role definiscono i diritti di accesso alle funzionalità relative al Site Level, come la gestione degli utenti, il setup di modelli ingegneristici o la gestione delle Reference Data Library. I valori ammissibili in questi campi sono riportati nella tabella seguente:

Valore ammissibile	Descrizione
NONE	Nessun permesso
READ	Permesso di sola lettura
MODIFY	Permesso di scrittura
READ_IF_PARTICIPANT	Permesso di sola lettura, se l'utente è stato assegnato come partecipante a uno specifico modello ingegneristico
MODIFY_IF_PARTICIPANT	Permesso di scrittura, se l'utente è stato assegnato come partecipante a uno specifico modello ingegneristico
MODIFY_OWN_PERSON	Permesso di modifica dei dati relativi al proprio utente. Questo valore è da attribuire alla classe Person, l'unico contesto in cui è significativo, per consentire agli utenti di modificare le proprie informazioni personali e le credenziali

Tabella 2.1. Valori ammissibili per i permessi d'accesso nei Person Role

Nel software sono implementati dei ruoli predefiniti che coprono le generiche figure coinvolte nel progetto ingegneristico. Per quanto riguarda i Person Roles, si hanno a disposizione:

- Concurrent Design Team Member: è il ruolo preposto per i progettisti. Poichè i Person Role si applicano al Site Level, i suoi permessi sono di sola lettura (READ) o di lettura su partecipazione ad uno specifico modello (READ_IF _PARTICIPANT), in relazione ad aspetti come la gestione dei modelli creati o degli utenti. Invece, è in grado di modificare le Reference Data Library dei modelli a cui è assegnato come partecipante, conferendo la possibilità, ad esempio, di creare nuove grandezze fisiche da attribuire al modello ingegneristico. Questo gli consente di svolgere le valutazioni di progetto, ma non di modificare informazioni relative all'azienda.

- Line Manager: è un ruolo di controllo delle informazioni di Site Level. I suoi permessi sono di sola lettura (READ), ma estesi a tutti gli aspetti del software.

- Site Administrator: l'amministratore è una figura fondamentale per la corretta gestione del software, ad esempio è il ruolo preposto alla creazione degli altri utenti. Il Site Administrator ha diritto di modifica (MODIFY) di tutti gli aspetti del software.

Nella seguente figura 2.5 sono riportati, a scopo esemplificativo, i diritti di accesso associati al ruolo Concurrent Design Team Member.

ClassKind	Access Right
DomainOfExpertise	READ
DomainOfExpertiseGroup	READ
EngineeringModelSetup	READ_IF_PARTICIPANT
IterationSetup	READ_IF_PARTICIPANT
ModelReferenceDataLibrary	MODIFY_IF_PARTICIPANT
Organization	READ
Participant	READ_IF_PARTICIPANT
ParticipantPermission	READ
ParticipantRole	READ
Person	MODIFY_OWN_PERSON
PersonPermission	READ
PersonRole	READ
SiteDirectory	READ
SiteDirectoryDataAnnotation	NONE
SiteDirectoryDataDiscussion...	NONE
SiteLogEntry	MODIFY
SiteReferenceDataLibrary	READ_IF_PARTICIPANT

Figura 2.5. Diritti di accesso del ruolo Concurrent Design Team Member.

È possibile notare come alcuni diritti di accesso riportino il valore NONE. Le classi a cui si riferiscono questi diritti di accesso sono da considerarsi interne al software, dunque non risultano rilevanti, al fine di determinare l'accesso alle informazioni del modello e alle funzionalità del software.

Participant Role

I Participant Role determinano i diritti di accesso a uno specifico modello ingegneristico a cui l'utente è stato assegnato come partecipante. Di seguito sono riportati i diritti di accesso relativi ai Participant Role:

Nell'ambito dei Participant Role i quattro valori ammissibili nei campi sono (2.2):

Valore ammissibile	Descrizione
`NONE`	Nessun permesso
`READ`	Permesso di sola lettura
`MODIFY`	Permesso di scrittura
`MODIFY_IF_OWNER`	Permesso di scrittura, se l'utente appartiene al Dominio proprietario del dato in questione

Tabella 2.2. Valori ammissibili per i permessi d'accesso nei Person Role

I Participant Roles predefiniti sono:

- Observer: presenta permessi di sola lettura (READ), estesi su tutti gli aspetti del modello ingegneristico. Può rappresentare un'autorità di riferimento, ad esempio l'ASI nel caso specifico dell'industria spaziale italiana, che dunque può accedere alle informazioni di progetto senza diritto di modifica.

- Customer: presenta permessi di scrittura (MODIFY) relativamente alla gestione dei Requirements, mentre può visualizzare in sola lettura (READ) gli altri aspetti del progetto.

- Technical Author: similmente al customer, presenta diritti di modifica se appartenente al Dominio proprietario (MODIFY_IF_OWNER) dei Requirements. Ulteriormente presenta lo stesso diritto di modifica su proprietà (MODIFY_IF_OWNER) dei Domain File Store, cartelle di file condivisi sulla base dell'appartenenza al dominio.

- Domain Expert: rappresenta il progettista vero e proprio. Possiede diritti di scrittura su appartenenza al Dominio proprietario (MODIFY_IF_OWNER) su gran parte degli aspetti del progetto, può definire nuovi elementi nel modello ingegneristico e associarvi parametri, configurare Finite States e altre funzionalità che verranno analizzate in seguito.

- Team Leader & Design Authority: presentano lo stesso livello di permessi. Le possibilità di azione sono molto simili a quelle del Domain Expert, con la differenza che il permesso di scrittura (MODIFY) non è vincolato all'appartenenza al Domain proprietario dell'informazione di progetto. Questi due ruoli sono gli unici, insieme al Model Administrator, a poter gestire le Publication, ovvero hanno la possibilità di validare il lavoro svolto all'interno dei workspace del Dominio e approvare i nuovi valori nei parametri di progetto.

- Model Administrator: l'amministratore, come nel caso dei Person Role, è una figura necessaria al corretto funzionamento del software. Presenta i diritti di accesso più completi, ovvero la scrittura (MODIFY) su tutte le classi che implementano il modello ingegneristico, non per competenza ingegneristica ma informatica.

Nella seguente figura 2.6 sono riportati, a scopo esemplificativo, i diritti di accesso associati al ruolo Observer.

ClassKind	Access Right	ClassKind	Access Right
ActionItem	NONE	NestedParameter	READ
ActualFiniteStateList	READ	Note	NONE
Approval	NONE	Page	NONE
Book	NONE	Parameter	READ
ChangeProposal	NONE	ParameterGroup	READ
ChangeRequest	NONE	ParameterOverride	READ
CommonFileStore	READ	ParameterSubscription	READ
ContractChangeNotice	NONE	PossibleFiniteStateList	READ
ContractDeviation	NONE	Publication	READ
DiagramCanvas	NONE	Relationship	NONE
DomainFileStore	READ	RequestForDeviation	NONE
ElementDefinition	READ	RequestForWaiver	NONE
ElementUsage	READ	Requirement	READ
EngineeringModel	READ	RequirementsGroup	READ
EngineeringModelDataAnno...	NONE	RequirementsSpecification	READ
EngineeringModelDataDisc...	NONE	ReviewItemDiscrepancy	NONE
EngineeringModelDataNote	NONE	RuleVerificationList	NONE
ExternalIdentifierMap	READ	Section	NONE
File	READ	SharedStyle	NONE
Folder	READ	Solution	NONE
Goal	NONE	Stakeholder	NONE
Iteration	READ	StakeholderValue	NONE
ModelLogEntry	READ	StakeHolderValueMap	NONE
NestedElement	READ	ValueGroup	NONE

Figura 2.6. Diritti di accesso del ruolo Observer.

Analogamente a quanto osservato per i diritti di accesso stabiliti nei Person Role, anche in questo caso alcune voci presentano il valore NONE poiché rappresentano classi interne del software.

Team Composition

Una volta definiti gli utenti, è possibile visualizzare la composizione dei team di progettazione tramite un pulsante dedicato. In questa schermata sono presentate tutte le informazioni relative agli utenti. Questi ultimi possono essere organizzati in base a voci specifiche, come l'Organization o il Domain a cui sono assegnati. In questa schermata, dunque, si ha un resoconto della creazione degli utenti, valutando i partecipanti a un determinato modello. Nella scheda di ogni partecipante, sono mostrate le informazioni rilevanti, come in figura 2.7, dove sono riportati tre esempi di partecipanti al modello *Spazializzazione di AVF*. Nella parte superiore della finestra infatti, sono riportati i dettagli della sessione in cui la finestra è stata aperta, ovvero Model, Data Source e Person. Queste informazioni sono riportate in molte delle finestre del software. Nell'immagine figurano tre utenti: The Administrator, utente predefinito da utilizzare per l'inizializzazione e la configurazione del modello, appartenente all'organizzazione Space V e attribuito al dominio del System Engineering; l'utente generico Control Authority, che rappresenta appunto un'autorità di controllo coinvolta nel progetto, in questo esempio attribuito all'organizzazione ESA e a tutti i domini attivi nel modello; infine è presente l'utente generico Space V Member, attribuito all'organizzazione Space V e a tutti i domini attivi.

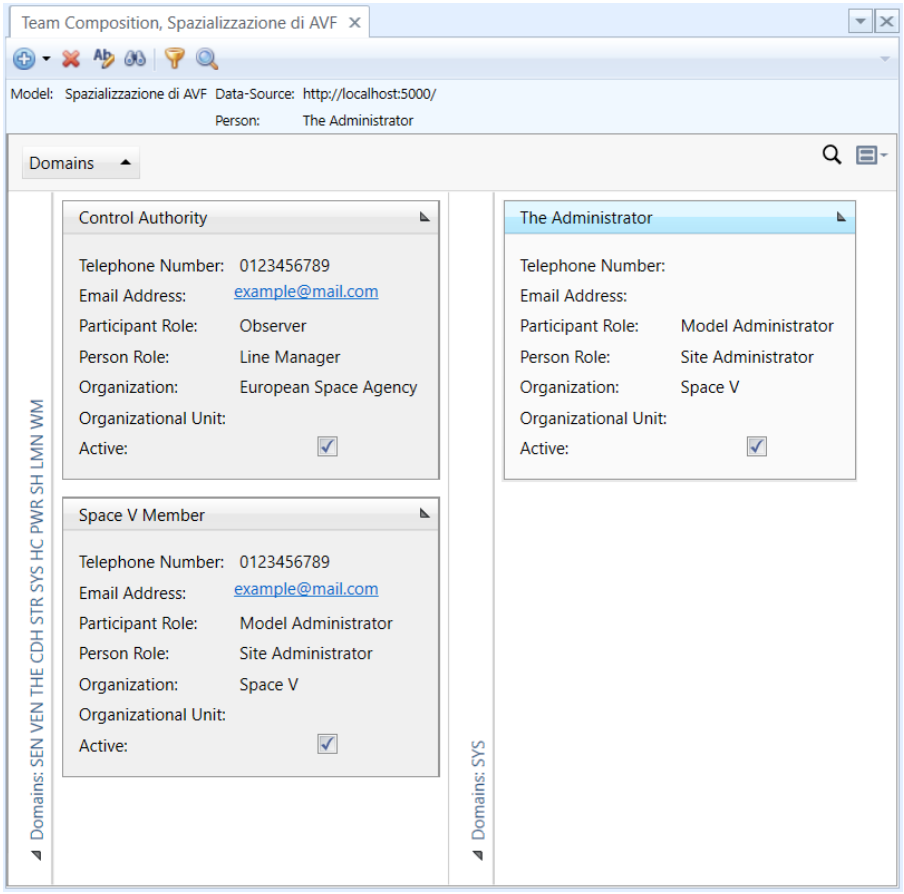

Figura 2.7. Esempio di visualizzazione dei partecipanti ad un modello ingegneristico tramite la funzionalità Team Composition.

2.2 Reference Data

Continuando ad esplorare le schede dell'interfaccia software, si analizza la schermata relativa ai Reference Data (figura 2.8). I Reference Data sono grandezze fisiche e concetti utili alla progettazione ingegneristica, che possono essere attribuiti ad altri oggetti del software. Ad esempio, si possono attribuire parametri ad elementi del modello ingegneristico, come la massa, oppure associare parametri ai requisiti, per riportare il metodo di verifica previsto.

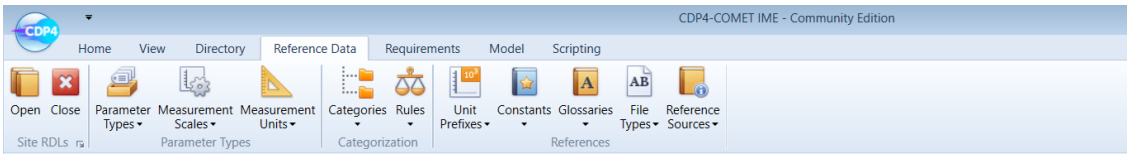

Figura 2.8. Toolbar della sezione Reference Data.

Tramite i pulsanti di questa schermata, si può accedere alle Reference Data Library (RDL). Le RDL sono i contenitori di tutti i concetti implicati nelle funzionalità del software, comprendendo quindi grandezze fisiche, con la relativa scala di misura, o le categorie di elementi del modello ingegneristico, come System, Subsystem o Element. Le RDL possono essere definite a tre livelli all'interno del software:

- Top Level Site RDL: il livello più generico di definizione, nel quale includere tutti i concetti che si prevede debbano essere implicati in tutti i modelli ingegneristici.
- Second Level Site RDL: consiste in una branca di un'altra Site RDL, estendendo quest'ultima con ulteriori concetti specifici di un ambito ingegneristico e di cui si prevede l'utilizzo in tutti i modelli dell'ambito. Questo livello di RDL potrebbe contenere, ad esempio, i Reference Data utili ai progetti di un'azienda o di una divisione aziendale. Le Second Level Site RDL, infatti, possono essere create a partire da una Top Level o da un'altra Second Level.
- Model Level RDL: vengono create automaticamente insieme al modello, estendendo la Site RDL assegnata come riferimento in fase di creazione del modello stesso. I concetti definiti all'interno di questo contenitore possono essere utilizzati esclusivamente nell'ambito del modello ingegneristico associato.

Comet possiede una Top Level Site RDL predefinita, che consiste nell'Annex B del ECSS-E-TM-10-25. Nella RDL predefinita sono presenti i parametri genericamente utili nel progetto di missione spaziale, sono contenuti numerosi parametri come le grandezze fondamentali del Sistema Internazionale e molte grandezze derivate, ma anche altri concetti come il Technology Readiness Level (TRL) oppure il fornitore di un componente. Anticipando brevemente alcune funzionalità secondarie, all'interno delle RDL, è possibile definire:

- Unit Prefixes, prefissi per le unità di misura che specificano l'ordine di grandezza (milli, kilo, giga);
- Constants, definizione di costanti a partire dai parametri esistenti;
- Glossaries, insiemi di termini utili al gruppo di progettazione, per specificare concetti specifici dell'ambito di progettazione o per risolvere casi di omonimia;
- File Types, estensioni di file riconosciute all'interno del software;

- Reference Sources, collegamenti esterni a documenti di riferimento come ISO o ECSS.

Si analizzano dunque due aspetti principali delle RDL: creazione e gestione di parametri e categorie.

2.2.1 Parameter Types

In questa sezione si possono creare nuovi parametri e gestire quelli esistenti. I parametri sono uno strumento fondamentale nella costruzione del modello ingegneristico, poiché permettono di attribuire valori, non necessariamente numerici, agli elementi astratti che compongono il modello stesso. L'utilizzo di questi parametri consente una descrizione completa e personalizzabile degli oggetti implicati nei modelli ingegneristici. I progettisti possono svolgere elaborazioni dei valori dei parametri e condividerli all'interno del gruppo di progettazione, contribuendo all'avanzamento del processo. I parametri infatti, sebbene spesso consistano in grandezze fisiche, vanno intesi in maniera più generale come attributi degli oggetti in considerazione. Volendo valutare le tipologie di parametri a disposizione dell'utente, si individuano innanzitutto delle tipologie più semplici:

- Boolean: parametro che può assumere solamente valori booleani, dunque vero o falso;
- Date: rappresentazione di una data, nel formato YYYY-MM-DD;
- Date Time: rappresentazione di una data completa di orario, nel formato YYYY–MM–DD-HH:MM:SS;
- Text: parametro costituito da una stringa di testo;
- Time Of Day: rappresentazione di un orario, nel formato HH:MM:SS.

Come si può notare, queste tipologie di parametri, non strettamente numerici, sono di grande importanza nello studio di progettazione. Ad esempio i Text Parameter Types possono essere utilizzati per tener conto del produttore o del fornitore di uno specifico componente, oppure possono essere assegnati dei flag booleani per individuare istantaneamente determinati aspetti. La definizione di questi parametri avviene semplicemente tramite l'attribuzione di un nome, un'abbreviazione ed un simbolo. Questi parametri tuttavia non sono sufficienti alle valutazioni tipicamente necessarie nello studio di progettazione di una missione spaziale. Si analizzano, dunque, le restanti tipologie di parametri.

Quantity Kind

I Quantity Kind sono parametri numerici scalari che rappresentano le grandezze fisiche. La creazione di questi parametri prevede la scelta di una o più Measurement Scale (MS), definite dalla combinazione di una scala di misura, un insieme numerico e una Measurement Unit (MU), a sua volta definita semplicemente fornendo un nome e un simbolo.

Per definire una MS quindi, bisogna associare la MU desiderata alla scala di misurazione tra quelle disponibili. Queste sono:

- Ratio Scale per scale di misurazione con uno zero assoluto, come la massa o la temperatura;
- Interval Scale se, invece, possiedono uno zero arbitrario;
- Cyclic Scale, scale cicliche tra due valori prestabiliti, come la rappresentazione degli angoli in gradi (°);
- Logarithmic Scale, scale logaritmiche con una certa base, come i decibel (dB)

Infine si può stabilire l'insieme numerico di riferimento tra Reali, Razionali, Naturali e Interi. Una volta definita la MS desiderata, si può scegliere quale tipologia di Quantity Kind creare. Si hanno a disposizione:

- Simple Quantity Kind, da utilizzare per definire delle grandezze fisiche indipendenti da altre Quantity Kind, come le grandezze del SI. Le 7 grandezze del SI sono riportate nella libreria predefinita in grassetto.
- Derived Quantity Kind, grandezze derivate, appunto, dalla combinazione di altre Quantity Kind. La definizione avviene tramite l'attribuzione di esponenti alle Quantity Kind coinvolte.
- Specialized Quantity Kind, utili a stabilire una specifica accezione di un'altra grandezza fisica. Ad esempio, grandezze come la profondità, la quota o il diametro sono specializzazioni della lunghezza, Simple Quantity Kind del SI.

Compound Parameter Type

I Compound Parameter Type (Compound) sono una rappresentazione di parametri non-scalari. Tramite i Compound, è possibile raggruppare altri parametri a discrezione dell'utente, al fine di creare degli insiemi di parametri tipicamente utilizzati in set. Le coordinate geografiche sono un esempio di parametro di questo tipo, definite dall'insieme di latitudine e longitudine, due Specialized Quantity Kind. Quasi tutti i tipi di parametri possono essere utilizzati nella creazione di un Compound, ad eccezione dei Compound stessi e degli Array Parameter Type.

Array Parameter Type

Gli Array Parameter Type (Array) sono una specializzazione dei Compound: in aggiunta a questi ultimi, gli Array presentano un'indicazione sulla dimensionalità del parametro. Di conseguenza, specificando la struttura di un Array, è possibile creare vettori, matrici e tensori delle dimensioni desiderate. Questo parametro può essere utile a valutare, ad esempio, vettori di posizione o tensori d'inerzia. Nuovamente, i Compound e gli Array stessi non possono essere utilizzati nella definizione di questi parametri.

Enumeration Parameter Type

Gli Enumeration Parameter Type rappresentano degli elenchi di valori possibili a cui è associato, arbitrariamente, un significato. Un esempio è il TRL, che associa un numero da 1 a 9 al livello di maturità di una certa tecnologia nell'ambito spaziale, oppure lo schema di ridondanza di un sistema (Nullo, Attivo o Passivo). Per la definizione di un parametro di questo tipo quindi, l'utente deve stabilire i valori discreti che il parametro può assumere nel suo utilizzo.

Sampled Function Parameter Type

I Sampled Function Parameter Type (Sampled Function) sono dei parametri che associano un insieme di parametri indipendenti ad un insieme di parametri dipendenti. In questo modo è possibile rappresentare delle funzioni matematiche o semplicemente associare due parametri in una relazione chiave-valore. Ad esempio, si possono utilizzare i Sampled Function per descrivere l'andamento con la temperatura di determinate proprietà per un set di materiali, oppure per riportare la risposta in spostamento dello spettro di frequenza di un elemento strutturale. Infine, nella definizione dei Sampled Function, è possibile indicare il grado di interpolazione desiderato per la valutazione dei parametri indicati.

2.2.2 Categorization

Una funzionalità molto utile implementata all'interno del software è la possibilità di creare Categories, da attribuire a qualunque oggetto implementato nel software. Questa funzionalità può essere sfruttata innanzitutto per la gestione dei concetti astratti, andando a raggruppare i concetti del software per caratteristiche comuni. In secondo luogo la struttura organizzativa, definita tramite le categorie, può essere dettagliata tramite la subordinazione tra categorie e la definizione di Rules. Per la definizione di una Category quindi, è necessario selezionare una o più Permissible Classes, le rappresentazioni informatiche dei concetti astratti che compongono il software stesso. Una volta definita una categoria, e quindi selezionata una classe tra quelle possibili, la categoria risulta disponibile per l'attribuzione, nella scheda apposita della finestra di creazione della classe di riferimento. Successivamente alla definizione e attribuzione delle categorie, si possono definire delle Rules che descrivano delle relazioni concettuali tra queste categorie. Prima di procedere con una descrizione più dettagliata delle tipologie di Rules che possono essere definite, si vuole sottolineare come queste funzionalità di categorizzazione possano fornire uno strumento utile a un gruppo di progettazione, al fine di creare una base di concetti solida e condivisa tra tutti i partecipanti. L'insieme di Categories e Rules definite da un gruppo di progettazione, ne descrive quindi la specifica metodologia. Tramite la Rules Validation, che descriveremo trattando il modello ingegneristico, il gruppo di progettazione può verificare che le definizioni del modello siano conformi alle raccomandazioni interne, verificando dunque la consistenza del processo di progettazione e del modello ingegneristico.

Rule Types

Le tipologie di Rules che possono essere definite all'interno del software sono:

- Binary Relationship Rules: specifica la categoria sorgente e quella obiettivo, utili per la creazione di Relationships vere e proprie. Ulteriormente alle due categorie precedentemente individuate, bisogna fornire una categoria di relazioni a cui si vuole attribuire la Rule in stato di definizione, e alla quale apparterranno automaticamente tutte le Relationships costruite a partire da questa Rule. La creazione della Rule si completa inserendo due nomi che siano indicativi per la relazione diretta e inversa.
- Decomposition Rules: sono riferite solamente a categorie che hanno come Permissible Class l'Element Definition. È possibile specificare quali categorie decompongono la categoria selezionata, instaurando di conseguenza un ordine gerarchico tra categorie di elementi. Queste Rules dunque definiscono la struttura ad albero che verrà definita all'interno del modello ingegneristico. Alcune Rules di questa tipologia sono presenti di default nel software, implementando la decomposizione dei sistemi indicata nell'ECSS-E-TM-10-25A.
- Multi Relationship Rules: in maniera del tutto analoga alle Binary Relationship Rules, individua una Referencing Category a cui associare un numero arbitrario di Referenced Categories. Queste Rules servono da criterio per la creazione di Relationship di questa tipologia.
- Parameterized Category Rules: permettono di associare degli specifici parametri alle categorie che possiedono come Permissible Class l'Element Definition. Queste Rules indicano dunque che la definizione degli elementi appartenenti alla categoria indicata dovrebbe sempre contenere i parametri scelti. Ad esempio, una Rule predefinita impone la logica che tutti gli elementi categorizzati come Equipment abbiano associati i parametri di massa e TRL.
- Referencer Rules: servono a specificare un numero minimo o massimo di riferimento per l'implicazione di elementi all'interno del modello ingegneristico. Tramite questo tipo di Rules, ad esempio, si può impedire l'utilizzo di elementi annidati, o limitare il numero di utilizzi annidati.

È importante notare che l'utilizzo delle Rules non vincola le funzionalità del software: gli utenti sono sempre in grado di *violare* le Rules definite. Tuttavia, queste possono essere validate, individuando quali aspetti del modello ingegneristico non risultano conformi alla metodologia stabilita.

2.3 Requirements

Si prosegue con l'analisi della schermata relativa ai Requirements. I requisiti sono un aspetto centrale dello sviluppo di un progetto, soprattutto nelle prime fasi. Una definizione dettagliata dei requisiti, infatti, permette lo svolgimento di un'analisi funzionale del

sistema in esame, consentendo una decomposizione del progetto in elementi funzionali individuati per assolvere a specifiche necessità della missione spaziale. La formulazione dei Requirements, dunque, parte dagli obiettivi della missione e descrive tutti gli aspetti che il sistema, tramite la sua architettura, deve contemplare per il compimento della missione stessa. Le funzionalità dedicate dal software a questo processo del progetto ingegneristico consentono una profonda definizione e gestione dei requisiti, tramite i pulsanti presenti nella seguente immagine (figura 2.9).

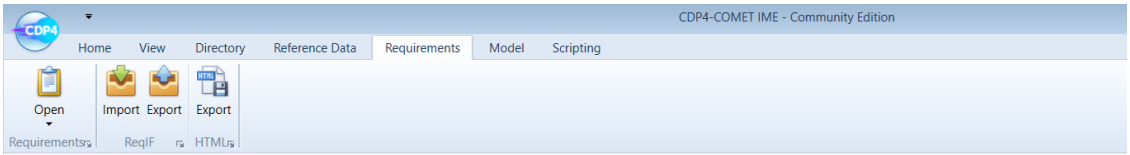

Figura 2.9. Toolbar della sezione Requirements.

Innanzitutto, utilizzando il pulsante Open, viene visualizzata la finestra per la gestione dei requisiti e delle Requirements Specification (RS). Le RS sono *set* che organizzano i Requirements secondo un aspetto del progetto, come ad esempio i Requisiti di Missione o i Requisiti Funzionali, e rappresentano quindi dei contenitori di alto livello per la successiva definizione dei requisiti veri e propri. In aggiunta, all'interno delle RS, è possibile definire dei Requirement Groups per organizzare ulteriormente i requisiti. Le RS possono essere visualizzati singolarmente in un editor specifico che riporta i gruppi definiti, come intestazione delle diverse sezioni, e i dati principali dei requisiti associati. La definizione dei Requirement (figura 2.10), come suggerito dall'ECSS nel medesimo TM, prevede l'attribuzione di un nome, un identificativo, una descrizione e un dominio Owner. La finestra per la formulazione della descrizione presenta diverse funzionalità per la formattazione del testo, come il grassetto, il corsivo, gli elenchi puntati o numerati. In aggiunta, è possibile visualizzare un'anteprima della descrizione, prima di ultimare la stesura. Infine, è possibile attribuire delle categorie ai Requirements consentendo di conseguenza l'applicazione di Rules, il che può essere utile, ad esempio, per tenere traccia della decomposizione dei requisiti o delle loro implicazioni sul modello ingegneristico.

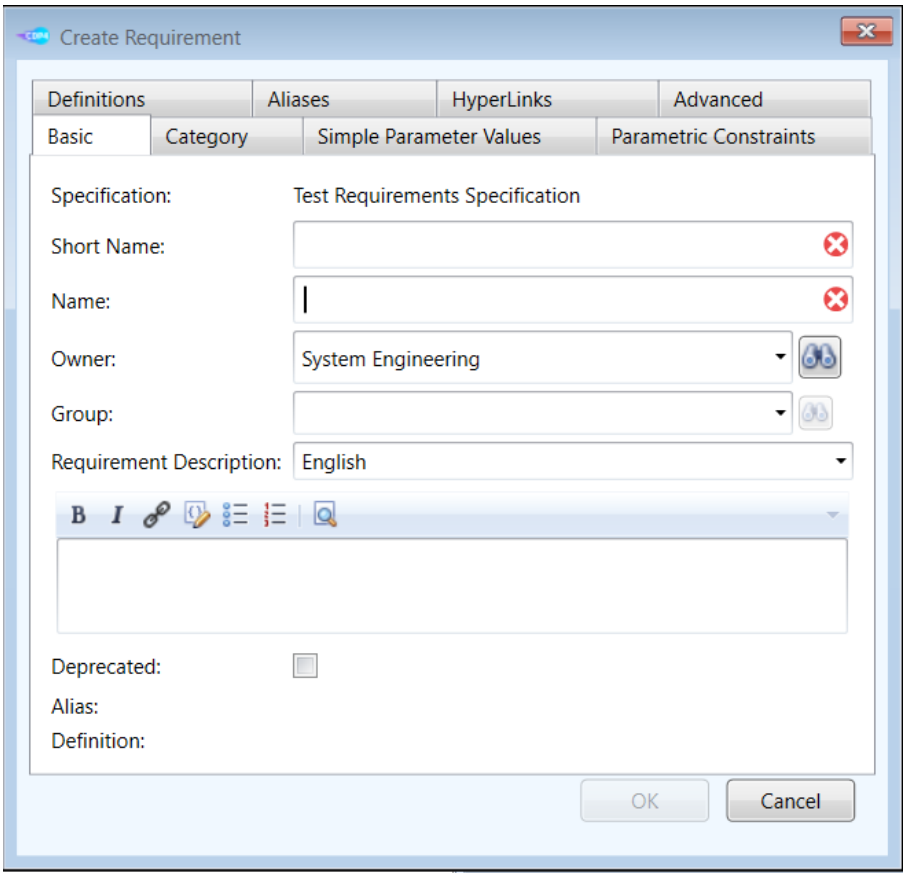

Figura 2.10. Toolbar della sezione Requirements.

2.3.1 Simple Parameter Values

Sebbene i campi previsti per i Requirements siano solamente nome, abbreviazione e descrizione, è possibile attribuire i parametri della RDL ai Requirements sfruttando i Simple Parameter Values. Questa funzionalità, integrata nella finestra di definizione dei Requirements, permette di associare ulteriori parametri al requisito di progetto, come ad esempio un Enumerate Parameter che riporti i possibili metodi di verifica. Sebbene tutti i parametri della RDL possano essere attribuiti ai Requirements, spesso sarà necessario definire dei parametri specifici per la loro caratterizzazione. Un'altra elaborazione che è possibile svolgere sui Requirements consiste nell'utilizzare le Categories e le Rules, precedentemente descritte, per organizzare e correlare i requisiti di progetto. Definendo le categorie necessarie, è possibile creare un raggruppamento a seconda di caratteristiche distintive dei requisiti ed eventualmente organizzare in una struttura gerarchica questi raggruppamenti tramite l'utilizzo di Super Categories, fornendo ulteriore contesto ai requisiti. Sulle categorie definite per i Requirements possono inoltre essere definite delle Rules, creando delle logiche che siano la base di partenza concettuale per la formulazione di relazioni tra i requisiti stessi e tra requisiti ed elementi del modello che ne rispondono. Ad esempio, si

possono definire delle categorie che raggruppano i requisiti in base ai livelli di dettaglio in cui è definito il progetto, dalla missione agli Equipment. Così facendo, si possono definire una serie di relazioni concettuali tra queste categorie, permettendo un tracciamento dettagliato delle varie relazioni logiche tra i requisiti stessi o con gli elementi del modello ingegneristico. Ad esempio, definendo la Rule *implica*, si può implementare una decomposizione dei Requirements che, a partire dai requisiti di missione, implichi i requisiti di sistema, di sottosistema e così via, generando una matrice di relazioni tra requisiti che ne permette innanzitutto la valutazione e che tenga traccia della struttura gerarchica dei requisiti nel corso di tutte le fasi di progettazione. Un'altra elaborazione che è possibile svolgere sui Requirements consiste nell'utilizzare le Categories e le Rules, precedentemente descritte, per organizzare e correlare i requisiti di progetto. Definendo le categorie necessarie, è possibile creare un raggruppamento a seconda di caratteristiche distintive dei requisiti ed eventualmente organizzare in una struttura gerarchica questi raggruppamenti tramite l'utilizzo di Super Categories, fornendo ulteriore contesto ai requisiti. Sulle categorie definite per i Requirements possono inoltre essere definite delle Rules, creando delle logiche che siano la base di partenza concettuale per la formulazione di relazioni tra i requisiti stessi e tra requisiti ed elementi del modello che ne rispondono. Ad esempio, si possono definire delle categorie che raggruppano i requisiti in base ai livelli di dettaglio in cui è definito il progetto, dalla missione agli Equipment. Così facendo, si possono definire una serie di relazioni concettuali tra queste categorie, permettendo un tracciamento dettagliato delle varie relazioni logiche implicate tra i requisiti stessi o con gli altri elementi del modello ingegneristico. Ad esempio, definendo la Rules ", si può implementare una decomposizione dei Requirements che, a partire dai requisiti di missione, implichi i requisiti di sistema, di sottosistema e così via, generando una matrice di relazioni tra requisiti che ne permette innanzitutto la valutazione e che tenga traccia della struttura gerarchica dei requisiti nel corso di tutte le fasi di progettazione. Un'altra elaborazione che è possibile svolgere sui Requirements consiste nell'utilizzare le Categories e le Rules, precedentemente descritte, per organizzare e correlare i requisiti di progetto. Definendo le categorie necessarie, è possibile creare un raggruppamento a seconda di caratteristiche distintive dei requisiti ed eventualmente organizzare in una struttura gerarchica questi raggruppamenti tramite l'utilizzo di Super Categories, fornendo ulteriore contesto ai requisiti. Sulle categorie definite per i Requirements possono inoltre essere definite delle Rules, creando delle logiche che siano la base di partenza concettuale per la formulazione di relazioni tra i requisiti stessi e tra requisiti ed elementi del modello che ne rispondono. Ad esempio, si possono definire delle categorie che raggruppano i requisiti in base ai livelli di dettaglio in cui è definito il progetto, dalla missione agli Equipment. Così facendo, si possono definire una serie di relazioni concettuali tra queste categorie, permettendo un tracciamento dettagliato delle varie relazioni logiche implicate tra i requisiti stessi o con gli altri elementi del modello ingegneristico. Ad esempio, definendo la Rule *implica*, si può implementare una decomposizione dei Requirements che, a partire dai requisiti di missione, implichi i requisiti di sistema, di sottosistema e così via, generando una matrice di relazioni tra requisiti che ne permette innanzitutto la valutazione e che tenga traccia della struttura gerarchica dei requisiti nel corso di tutte le fasi di progettazione.

2.3.2 Parametric Constraints

Una volta definiti i Requirement, e attribuite loro le informazioni necessarie, è possibile imporre la verifica dei requisiti creando delle relazioni con i parametri di progetto del modello. Tramite la funzione Parametric Constraints, il software implementa un algoritmo di verifica basato sulla comparazione di un parametro con un valore assegnato. Nella definizione di un Requirement quindi, si possono definire delle espressioni che associano il parametro desiderato, tra quelli presenti nella RDL, al valore di riferimento stabilito dal requisito. Per parametri di tipo numerico dunque, si può valutare l'uguaglianza o la disuguaglianza, distinguendo i vari casi possibili ($\leq$, $>$, ecc.). Invece, per parametri come stringhe, parametri enumerativi o booleani, l'unica comparazione che produce un risultato significativo è la corrispondenza esatta con il valore di riferimento, dunque si possono imporre esclusivamente comparazioni del tipo *uguale a* o *diverso da*. Dopo aver definito una o più espressioni di questo tipo, si possono configurare relazioni tra le espressioni stesse, utilizzando degli operatori logici. In particolare, tramite gli operatori di congiunzione AND, disgiunzione OR, negazione NOT, e disgiunzione esclusiva XOR è possibile creare delle relazioni di verifica dei requisiti piuttosto efficaci. Ad esempio, nel voler verificare che il TRL di un certo componente sia maggiore o uguale al livello 7, essendo il parametro TRL enumerativo, bisognerebbe effettuare tre comparazioni separate del parametro per i valori 7, 8 e 9 e collegarle tramite un OR. Una volta definita la Parametric Constraints, bisogna associare l'espressione appena formulata allo specifico parametro di interesse. Questa operazione si svolge semplicemente trascinando sull'espressione il parametro da una delle finestre relative al modello ingegneristico, come la finestra della Element Definition o del Product Tree, che descriveremo nel prossimo paragrafo. Svolta questa operazione, il software crea automaticamente una relazione che può essere consultata, modificata o eliminata nell'apposita schermata, sempre relativa al modello ingegneristico. Una volta configurati i Parametric Constraints desiderati, è possibile svolgere la verifica di tutti i requisiti definiti. Questo produrrà tre esiti possibili a cui corrispondono dei colori di riempimento della lista dei requisiti, otteremo:

- riempimento verde, il requisito risulta verificato dal parametro in esame;
- riempimento rosso, il requisito non è verificato dal parametro;
- riempimento giallo, la verifica non ha prodotto un risultato significativo. Questo avviene in generale quando un requisito non possiede un'attribuzione di Parametric Constraints, oppure nel caso in cui il vincolo imposto non sia stato configurato correttamente.

Nella figura 2.11 sono riportati gli esiti della verifica di una Parametric Constraints su due differenti Options. Il requisito impone che la massa del sistema non superi gli 80 kg, in questo esempio la prima Option verifica il requisito mentre la seconda verifica ha esito negativo. Si può notare come un secondo requisito, a cui non è associata nessuna Parametric Constraint, presenti sempre il riempimento giallo.
Infine, anticipando ancora dei concetti che verranno approfonditi descrivendo il modello ingegneristico, la verifica dei requisiti utilizza gli Actual Value, piuttosto che i Published

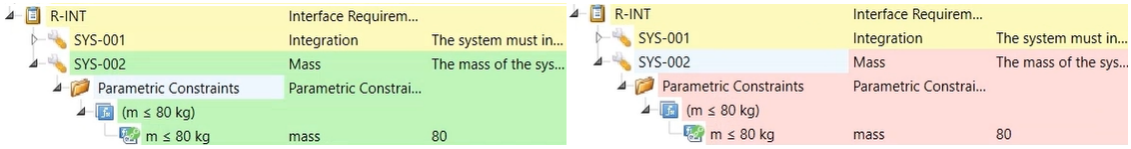

Figura 2.11. Possibili esiti di verifica di una Parametric Constraint.

Value, dove la differenza sarà chiarita in seguito, permettendo la valutazione della verifica dei requisiti prima di proporre una variazione effettiva di un parametro di progetto. Infine, la verifica dei requisiti può essere svolta singolarmente sulle varie Options del modello ingegneristico, consentendo la valutazione e validazione dei requisiti simultaneamente su diverse architetture di sistema, per il compimento di una certa missione.

2.3.3 Import & Export dei Requirements

Il software sfrutta un formato omologato per la definizione dei requisiti chiamato Requirement Interchange Format (ReqIF)[23], piuttosto diffuso in ambito ingegneristico, che consente appunto l'importazione e l'esportazione dei requisiti da e verso altri software che utilizzano lo stesso formato, come ad esempio il già citato Capella oppure il software di gestione dei requisiti DOORS, sviluppato da IBM. Il ReqIF consiste in uno specifico formato di un file XML che permette la compilazione dei dati relativi ai requisiti, riportando le informazioni fondamentali per la loro definizione, e consente di trasportare, all'interno del file di esportazione stesso, gli eventuali parametri interi, reali, booleani, testuali o numerativi correlati ai requisiti. Nel trasferimento dei Requirements, viene preservata la struttura organizzativa dei requisiti, riportando le informazioni relative all'appartenenza a Requirement Specifications ed eventuali Requirement Groups. Infine, le RS possono essere esportate singolarmente in formato HTML tramite il pulsante dedicato. Il file così generato può essere visualizzato tramite un qualunque browser, presentando la lista dei Requirements, e le relative informazioni, contenuti nella RS.

2.4 Model

La schermata sul modello introduce la maggior parte delle funzionalità del software, essendo questo il fulcro centrale del software. Tramite il modello infatti, si può rappresentare uno studio di progettazione, componendo, elemento per elemento, l'albero del prodotto del sistema in esame. La creazione di un modello ingegneristico avviene tramite la finestra consultabile utilizzando il pulsante Models nella toolbar della schermata Directory, da cui si accede alla creazione dell'Engineering Model Setup (figura 2.12). Esistono diverse tipologie di modelli definiti nel ECSS-E-TM-10-25A, e disponibili all'interno del software, che modificano le funzionalità disponibili in quel modello. In fase di creazione di un modello ingegneristico infatti, bisogna selezionare una tipologia di modello tra quelli riportati in tabella 2.3.

Model Kind	**Descrizione**
`Study Model`	Definisce un modello ingegneristico che rappresenta uno studio di progettazione.
`Template Model`	Definisce un modello ingegneristico volto a essere utilizzato come punto di partenza, ovvero Source Model, per altri modelli. Questo tipo di modello non permette la creazione di Iteration oltre quella predefinita.
`Catalogue Model`	Definisce un modello ingegneristico volto a essere utilizzato come catalogo di elementi, o insiemi di elementi, utili per altri modelli. Questo tipo di modello non permette la creazione di Iteration e Option oltre quella predefinita.
`Scratch Model`	Definisce un modello ingegneristico solitamente utilizzato come bozza, in preparazione di uno studio vero e proprio o per esplorare le funzionalità del software.

Tabella 2.3. Descrizione dei vari tipi di modelli ingegneristici

Successivamente, selezionando un modello nel campo Source Model, il nuovo modello viene generato come una copia del modello di partenza. Così facendo, la Site RDL riporterà automaticamente l'impostazione del Source Model, per preservare eventuali dipendenze presenti nel modello di partenza. Alternativamente, è possibile selezionare una Site RDL di riferimento da cui attingere per le definizioni di parametri, Category e Rules. Con la creazione del modello ingegneristico, viene automaticamente generata una nuova model RDL specifica del modello in questione. In questa RDL, a livello del modello, si possono definire oggetti che risultano essere utili solamente nel campo di studio dello specifico progetto. Infine, per completare la creazione del modello ingegneristico, è necessario selezionare una Study Phase tra quelle disponibili, ovvero:

- Preparation Phase;
- Design Session Phase;
- Reporting Phase;
- Completed Study.

Sebbene le fasi possano essere attribuite a qualunque tipo di modello ingegneristico, solitamente il loro utilizzo si addice solo agli Study Model. Queste fasi di studio, ad eccezione della Design Session Phase, costituiscono solamente un'indicazione per gli utenti della fase di avanzamento dello studio in questione, suggerendo un conseguente livello di completezza e affidabilità dei dati. La Design Session Phase invece, è l'unica fase di un modello ingegneristico in cui è consentito creare nuove iterazioni, ammesso che anche il Model Kind lo consenta. Le Iteration consentono di congelare nel tempo delle istantanee

di un modello ingegneristico. Creare una nuova Iteration, infatti, impedisce la futura modifica di quella di partenza, che rimane disponibile solo come riferimento per eventuali comparazioni. Tramite le iterazioni di progetto, è possibile tenere traccia dell'evoluzione del processo e creare dei punti di ripristino delle informazioni, per garantire agli utenti di svolgere elaborazioni in totale libertà, senza il timore di compromettere il lavoro precedente.
In aggiunta a queste informazioni, si possono selezionare i domini attivi e le organizzazioni coinvolte tra quelli definiti nella Site Directory. In questo modo si possono assegnare i partecipanti ai domini contrassegnati come attivi, dedicando loro degli ambienti virtuali di lavoro indipendenti, e distinguere l'organizzazione o le organizzazioni proprietarie del modello dagli altri enti partecipanti, conferendo la possibilità di differenziare ulteriormente la condivisione delle informazioni, non solo in base al ruolo, ma anche a seconda dell'organizzazione di appartenenza.

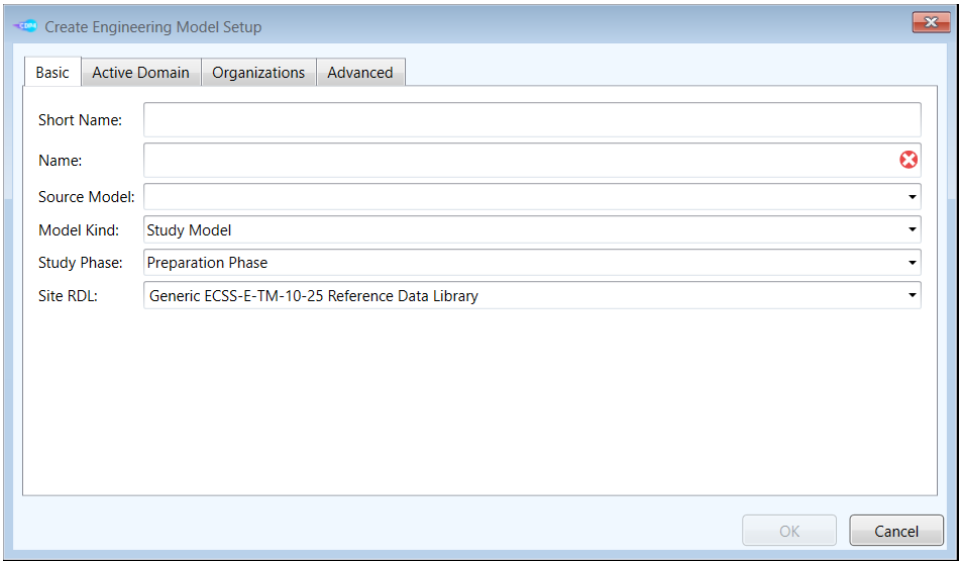

Figura 2.12. Finestra di creazione di Engineering Model Setup.

Dunque, dopo aver inizializzato il *setup* del modello ingegneristico, si possono analizzare le funzionalità previste nella toolbar di questa sezione (figura 2.13).

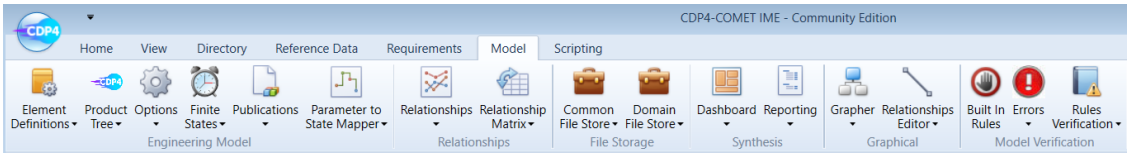

Figura 2.13. Toolbar della sezione Model.

2.4.1 Engineering Model

Una volta inizializzato il modello ingegneristico, si possono sfruttare le funzionalità dedicate, per creare il modello vero e proprio. La creazione del modello ingegneristico si basa sulla definizione (Element Definition), ed il successivo utilizzo (Element Usage), di building blocks per svolgere numerose tipologie di valutazioni. Avremo modo di definire dei *blocchi* che modellano uno specifico elemento o dal punto di vista funzionale, come i Subsystem, o fisico del prodotto, come gli Equipment. In seguito alla definizione, gli elementi possono essere utilizzati per comporre degli schemi a blocchi che rappresentano la missione o il sistema di interesse. La creazione del modello ingegneristico tramite definizione ed utilizzo dei building blocks comporta due importanti vantaggi: innanzitutto gli elementi, dopo averli definiti, possono essere utilizzati più volte all'interno di uno stesso Product Tree, inoltre gli utilizzi possono essere specializzati a seconda delle esigenze di progettazione. Ad esempio, nel design di un'automobile, si può definire una sola volta l'elemento ruota e utilizzarlo quattro volte nello schema a blocchi, specializzando per ogni utilizzo il parametro posizione in anteriore destra, anteriore sinistra e così via.

Element Definition & Usages

Per definire un elemento, è necessario conferire solamente un nome e una sua abbreviazione nella finestra in figura 2.14, tuttavia possono essere attribuite molte altre informazioni. Innanzitutto, l'Element Definition viene automaticamente attribuita al Dominio dell'utente che svolge l'operazione. Un utente con sufficiente livello di permessi, come Team Leader o Design Authority, può modificare il dominio Owner dell'elemento, assegnandolo arbitrariamente. Invece, gli utenti con livello di permesso inferiore a quelli sopra citati, hanno la possibilità di leggere l'Element Definition, visualizzando le informazioni ad esso attribuite, o di modificare i dati solo se appartenenti al dominio Owner dell'elemento.

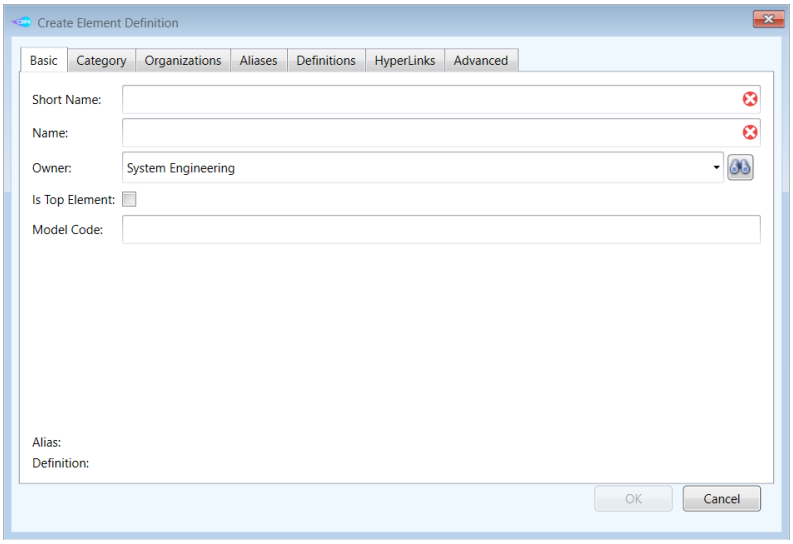

Figura 2.14. Finestra di definizione degli elementi.

Durante la definizione di un elemento, è possibile spuntare la casella Is Top Element per stabilire l'elemento di partenza nell'albero del prodotto e includere automaticamente l'elemento contrassegnato come Top Element nel Product Tree. Così facendo, diventa possibile trascinare un elemento, dalla finestra dell'Element Definition al Top Element nella finestra del Product Tree, creando un Element Usage della rispettiva Element Definition selezionata. Con questo processo è possibile implicare tutti gli elementi nelle relazioni di subordinazione desiderate, componendo lo schema a blocchi del Top Element indicato.
A partire dallo Short Name dell'elemento, viene generato il model code dello stesso. Quando gli elementi vengono utilizzati all'interno del Product Tree, applicandoli in relazioni con il Top Element o suoi subordinati, anche i model code si compongono di conseguenza, concatenandosi con punti (e.g. Sys.Subsys.Equip), e ripercorrendo l'architettura del sistema. Ulteriormente è possibile attribuire una categoria, sia all'Element Definition che al successivo Element Usage, consentendo l'applicazione di Rules. Nel software sono implementate alcune Decomposition Rules predefinite che riportano i criteri di subordinazione suggeriti nel TM ECSS, organizzando la struttura di categorie che compongono l'architettura di un sistema fino al livello di dettaglio dei Subequipment. Infine, è possibile specificare, per ogni definizione, quali organizzazioni tra quelle partecipanti al modello, oltre quella proprietaria, possono visualizzare l'Element Definition.

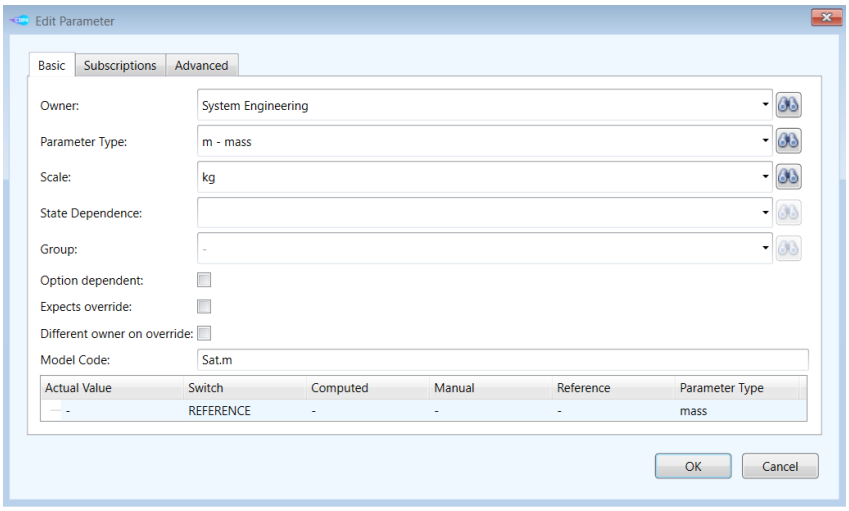

Figura 2.15. Finestra di modifica dei parametri.

Una volta definito un elemento, è possibile selezionare i parametri d'interesse dalla RDL e attribuire questi ultimi agli elementi, semplicemente trascinando un parametro dalla finestra Parameter Types nella schermata Reference Data, alla finestra di Element Definition. Una volta attribuito un parametro, è possibile modificare alcuni aspetti della sua definizione, come riportati in figura 2.15. Innanzitutto si può selezionare che tipo di valore utilizzare per il parametro tra:

- Manual, valore inserito manualmente;

- Computed, valore ottenuto tramite un'elaborazione interna al software o tool esterni come Excel o Matlab, non può essere impostato manualmente;
- Reference, valore impostato manualmente e concettualmente associato ad una fonte di riferimento.

È possibile indicare al software quale di questi tre valori considerare come attivo, selezionando il tipo di valore nel campo Value Switch del parametro. Così facendo, l'Actual Value del parametro sarà considerato uguale al tipo di valore indicato nel Value Switch. Anche ai parametri, similmente agli elementi definiti, è associato un model code, generato dal model code dell'elemento a cui il parametro è riferito, a cui viene concatenato, sempre tramite un punto, il simbolo del parametro stesso. Infine, nella schermata dell'Element Definition, i parametri attribuiti ad un elemento possono essere sovrascritti con la funzione Parameter Override. È possibile permettere la sovrascrittura e eventualmente la modifica dell'Owner del parametro insieme ad essa, spuntando le opportune caselle nella finestra di creazione o modifica del parametro stesso.
Dopo aver modellato gli elementi che compongono un sistema e averli implicati in relazioni gerarchiche, subordinandoli congruentemente al loro scopo, possono essere svolte ulteriori operazioni. Aprendo la finestra del Product Tree, viene visualizzato il Top Element dell'albero. Espandendo il menù a tendina associato al Top Element, vengono mostrati sia gli eventuali parametri attribuiti all'elemento sia gli ulteriori elementi annidati. In questo modo può essere visualizzata tutta l'architettura del sistema, così come è stata definita. Inoltre, nella finestra del Product Tree, è rappresentato un simbolo accanto ai parametri, un cerchio il cui colore indica la proprietà del parametro o la presenza di Subscriptions. Infatti, in fase di elaborazione dei dati, ad esempio tramite l'uso di Excel, un partecipante al modello ingegneristico ha a disposizione, nello spazio di lavoro virtuale dedicato al suo dominio, solamente i parametri di cui quel dominio è Owner, in modo che gli utenti possano elaborare e modificare solamente gli aspetti di loro competenza. Dato il carattere multidisciplinare di questi sistemi però, è molto frequente che i processi di valutazione interni ad un dominio necessitino di informazioni provenienti da altri aspetti del progetto, non di diretta competenza del dominio di appartenenza. Al fine di importare un parametro all'interno dello spazio di lavoro virtuale di un dominio, si utilizzano le Subscription: un utente può creare una Subscription su qualsiasi parametro di cui non sia già Owner, la Subscription verrà visualizzata in una pagina specifica dell'Element Usage, dove sono riportati tutti i domini sottoscritti ad uno specifico parametro. Per segnalare la presenza o meno di Subscription su un certo parametro, il simbolo apposto può essere del colore:

-  arancione, se nessun dominio è sottoscritto al parametro;
- blu, se almeno un dominio è sottoscritto al parametro;
-  bianco, se il dominio di appartenenza dell'utente è sottoscritto al parametro.

Questa funzionalità segnala ai vari team di progettazione la dipendenza vicendevole tra le elaborazioni svolte, incentivando la responsabilità degli utenti nelle valutazioni e nelle proposte di cambiamento dei valori associati ai parametri.

Options

Nella progettazione ingegneristica, può sorgere l'esigenza di valutare più configurazioni alternative di un sistema per rispondere, con soluzioni differenti, ai requisiti di progetto. Per questa necessità, nel software Comet possono essere create diverse Options. Queste opzioni, dunque, consistono in configurazioni alternative dello schema a blocchi, che permettono lo svolgimento di valutazioni simultanee, a diversi livelli di dettaglio, dell'architettura del sistema. Infatti, risulta possibile considerare valori diversi per uno specifico parametro, a seconda dell'opzione considerata, oppure si può rivalutare completamente il product tree, ad esempio nel caso in cui si considerino diversi Concept of Operations, definendo e utilizzando elementi differenti per la specifica Option. Infatti, soprattutto nelle fasi iniziali di progettazione, è utile poter valutare attentamente configurazioni operative sostanzialmente diverse, così da effettuare trade-off e identificare l'architettura di sistema più adatta alle condizioni operative, all'obiettivo di missione, ai tempi e ai costi di sviluppo e realizzazione della missione spaziale. Le Options vengono create all'interno delle Iterations, quindi, nel caso di uno Study Model in Design Session Phase, è possibile creare delle Options per valutare aspetti puntuali del progetto, svolgendo comparazioni all'interno della singola Iteration. Ad esempio, nella verifica delle Parametric Constraints, è possibile applicare l'algoritmo di validazione separatamente sulle opzioni, consentendo di valutare preliminarmente se la configurazione individuata rispetta i vincoli di progetto prodotti dai Requirements. Ogni modello ingegneristico possiede, sin dalla creazione, almeno un'Option di default, mentre le opzioni successive possono essere generate nella finestra dedicata. Le nuove opzioni includono tutti gli Element Usage precedentemente definiti; l'utente, quindi, può rimuovere le implicazioni di elementi non rilevanti per la valutazione da svolgere oppure creare nuovi Element Usage, specificandone l'implicazione solo nelle opzioni desiderate.

Finite States

In aggiunta alle opzioni, risulta utile poter valutare l'andamento dei parametri di progetto, al variare di stati discreti che si prospettano per il sistema in esame. Con stati discreti si intende, in generale, delle fasi ben distinguibili che producano un cambiamento apprezzabile dei parametri di progetto. Ad esempio, si possono considerare valori diversi dei parametri per analizzare l'andamento delle prestazioni del sistema, come la potenza elettrica richiesta, il calore dissipato o le risorse utilizzate dal computer di bordo, a seconda delle fasi operative previste per il sistema. Alternativamente, si potrebbe considerare la variazione dei costi e tempi associati alle diverse fasi della vita del prodotto.
Per utilizzare questa funzionalità, bisogna innanzitutto definire una Possible Finite States List, ovvero un elenco di stati alternativi possibili. Queste liste vengono definite compilando i tipici campi Name, Short Name e Owner e successivamente definendo i singoli Possible Finite State, tramite Name e Short Name. I Possible Finite State possono essere ordinati a discrezione dell'utente e può essere selezionato uno stato come predefinito per quella lista. Una volta definita almeno una Possible Finite State List, è possibile creare una Actual Finite State List, tramite la finestra riportata in figura 2.16. Le Actual Finite State List fanno riferimento a un dominio Owner, e vengono definite includendo le liste di

stati possibili nell'ordine opportuno. Una Actual Finite State List viene quindi definita dalla combinazione degli stati possibili delle Possible Finite States Lists selezionate: ogni stato della prima lista viene combinato singolarmente con gli stati della seconda lista, e così via per ogni lista di stati possibili inclusa. Potrebbe capitare, tuttavia, che nella considerazione di più liste di stati possibili per la creazione di una Actual Finite State List, alcune combinazioni debbano essere scartate poiché configurano degli stati discreti privi di senso in quanto irrealizzabili. A questo fine, è possibile selezionare lo State Kind per ogni Actual Finite State generato, contrassegnando come MANDATORY gli stati che si intendono valutare nel progetto o FORBIDDEN per le combinazioni prive di significato. Infine, selezionando gli specifici parametri all'interno dell'Element Definition o del Product Tree, è possibile impostare la dipendenza di quel parametro da una lista di Actual Finite State, selezionando l'Actual Finite State List nel campo State Dependence. Così facendo, il parametro verrà visualizzato con un menù a tendina che riporta i valori assunti a seconda del Finite State.

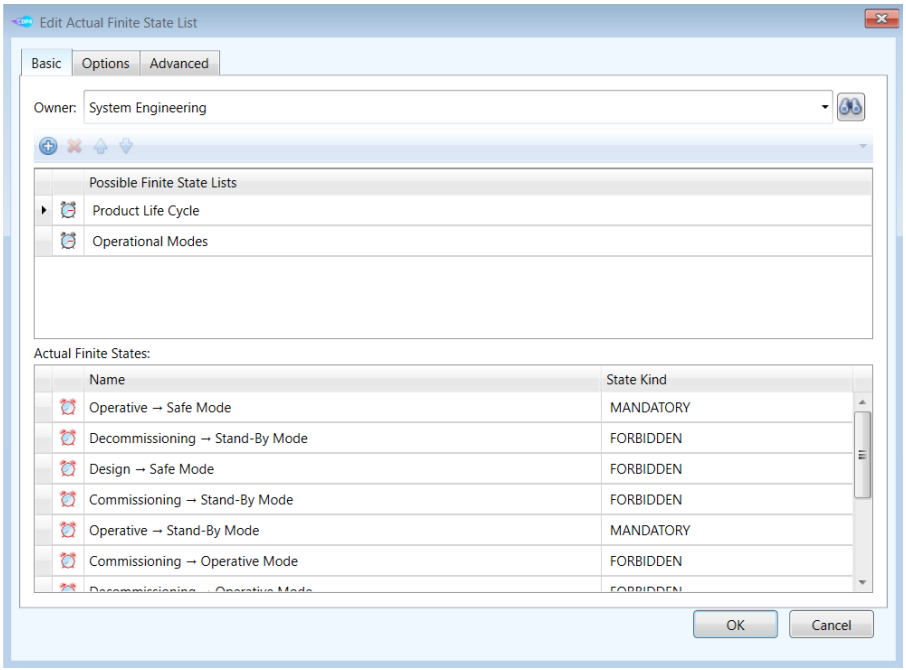

Figura 2.16. Finestra di creazione delle Actual Finite States List.

Publications

Si procede infine con l'analisi della funzionalità delle Publications. Al fine di mantenere un elevato livello di consistenza nel modello ingegneristico, ed evitare ambiguità sui valori attribuiti ai parametri, nel software Comet è integrato un sistema di pubblicazione dei valori dei parametri. Quando un partecipante inserisce o modifica il valore di un parametro di progetto, questa modifica avviene, inizialmente, nel workspace del Dominio che ha elaborato il parametro. Il nuovo valore infatti risulta essere attribuito al campo Actual

Value, e potrebbe quindi essere diverso dal valore contenuto nel campo Published Value. Infatti, l'Actual Value non risulta condiviso con gli altri domini, e quindi gli altri team di progettazione, finché non viene approvato da un utente con un sufficiente livello di permessi. Nella finestra delle Publications, vengono presentati all'utente i cambiamenti proposti dai vari partecipanti al modello ingegneristico, fornendo anche un'indicazione percentuale sul cambiamento proposto. L'utente abilitato può scegliere quali cambiamenti approvare tra quelli proposti da uno specifico dominio e, nella sezione apposita della finestra delle Publications, verrà tenuta traccia dell'evoluzione di questi parametri di progetto in uno storico che riporta data e ora della pubblicazione. Fintantoché un valore è in attesa di essere pubblicato, dunque finché l'Actual Value è diverso dal Published Value, il parametro è riportato in blu in grassetto nella visualizzazione dell'Element Definition o del Product Tree, evidenziando che su quel parametro è presente una Publication da accettare. Si ricorda che l'algoritmo di verifica dei vincoli di progetto associati ai Requirements sfrutta gli Actual Value, piuttosto che i Published Value, consentendo ai progettisti di svolgere delle valutazioni preliminari di verifica dei requisiti.

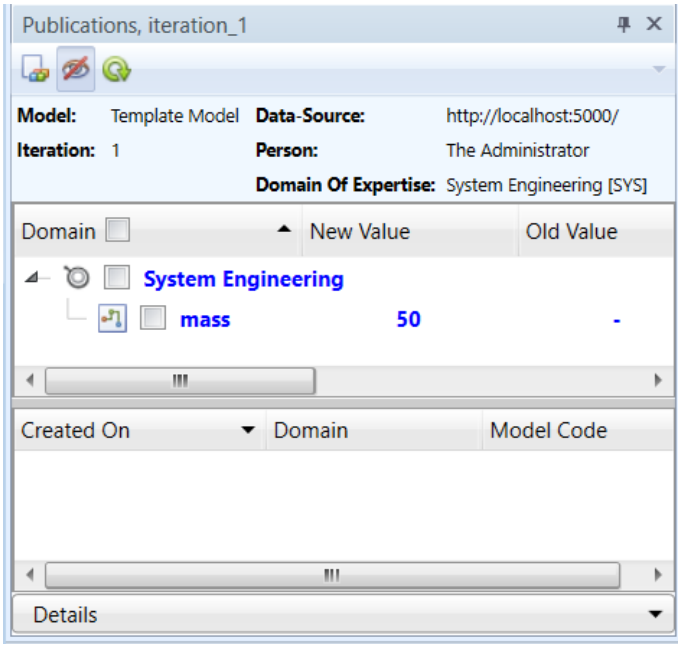

Figura 2.17. Finestra dedicata alle Publications.

2.4.2 Ulteriori Funzionalità

Dopo aver descritto le funzionalità principali e i concetti su cui si basa la creazione e gestione di un modello ingegneristico, si analizzano le restanti features del software relative al modello ingegneristico.

Relationships

Tramite il pulsante Relationships, si possono consultare e gestire tutte le relazioni definite all'interno del modello ingegneristico. Come per le Categories, le relazioni possono essere create a partire da tutte le classi che costituiscono l'implementazione del modello ingegneristico del software. Questa funzionalità presenta diverse applicazioni, ad esempio, definendo relazioni tra gli elementi del modello ingegneristico e/o i requisiti di progetto, è possibile tracciare le differenti implicazioni che caratterizzano il progetto. Una relazione può essere creata in numerosi modi: dalla finestra delle Relationships (figura 2.18) è possibile aprire la finestra di dialogo dedicata per la creazione, e compilare i campi richiesti, oppure trascinare direttamente gli elementi desiderati nella sezione apposita *Create a Relationship* della finestra.

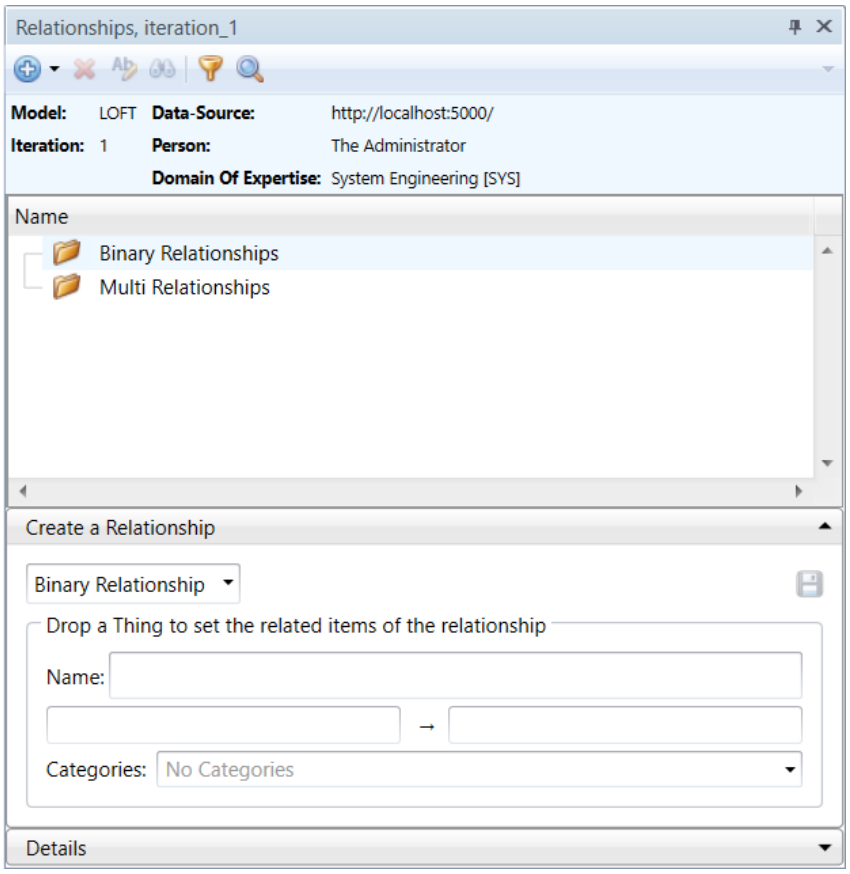

Figura 2.18. Finestra di gestione delle Relationships.

Un altro approccio alla creazione delle Binary Relationships è l'utilizzo della Relationships Matrix. Questa funzionalità permette all'utente di generare delle matrici selezionando i dati da inserire nelle righe e nelle colonne e la Binary Relationship Rule da utilizzare nella creazione. Per determinare gli oggetti da considerare nelle righe e colonne

della matrice, è possibile selezionare, nuovamente, le classi implementate nelle funzionalità del modello ingegneristico, applicando ulteriori filtri come categorie e i domini di appartenenza. In figura Ad esempio, categorizzando i requisiti come Alto Livello e Basso Livello, è possibile implementare una semplice decomposizione tramite le relazioni. La Relationships Matrix può essere usata sia per creare le relazioni, intervenendo nella cella di intersezione dei due oggetti da correlare, oppure come visualizzazione delle relazioni precedentemente definite. Inoltre, le Relationships Matrix possono essere esportate tramite il formato xlsx, per essere utilizzate anche su Excel, rendendole disponibili ad ulteriori elaborazioni tramite le funzionalità dei fogli di calcolo.
Analogamente alle relazioni già descritte, possono essere create delle Multi Relationships che associano un numero arbitrario di oggetti tra loro. Anche questo tipo di relazione può fare riferimento ad una Rule che ne specifichi la categoria. Le Multi Relationships possono essere create tramite l'apposita finestra di dialogo e la sezione *Create a Relationship*, tuttavia non sarà possibile associare a queste relazioni delle Relationships Matrix per la modifica e la visualizzazione. Infine, l'ultimo metodo di creazione delle Relationships consiste nel Relationship Editor che verrà descritto approfondendo le funzionalità grafiche.

File Storage

Rimanendo ancora tra le funzionalità relative al modello ingegneristico, vi è la possibilità di creare dei File Storage, delle cartelle condivise per l'immagazzinamento di file rilevanti. I File Storage possono essere creati su una base comune o specifica per Dominio, differenziando di conseguenza l'accesso ai file condivisi tramite questa funzionalità. Ogni utente con un l'apposito permesso potrà caricare file dal suo dispositivo rendendoli disponibili agli altri utenti. Inoltre, i file possono essere organizzati in cartelle per facilitare la gestione e la consultazione. Al fine di caricare un file all'interno di un File Storage, bisogna selezionare un File Type tra quelli a disposizione nella RDL considerata nel modello ingegneristico.

Synthesis

Nel software Comet avremo a disposizione delle funzionalità per l'elaborazione e la visualizzazione dei parametri di progetto e delle altre informazioni associate al modello ingegneristico. La più semplice di queste funzionalità è la Dashboard (figura 2.19); in questa finestra si possono trascinare parametri, tra quelli utilizzati nella Element Definition o nel Product Tree, e visualizzare il valore pubblicato e la variazione percentuale dall'ultima pubblicazione. Inoltre, la Dashboard appena generata, verifica l'esistenza di Parametric Constraints associate al parametro, colorando la Dashboard stessa del colore corrispondente all'esito della verifica della Parametric Constraints Alternativamente, può essere visualizzato un grafico che riporta l'andamento del parametro in questione al variare delle Revisions che ha subito, ovvero le modifiche svolte, rappresentando dunque l'evoluzione del parametro nel corso del progetto.

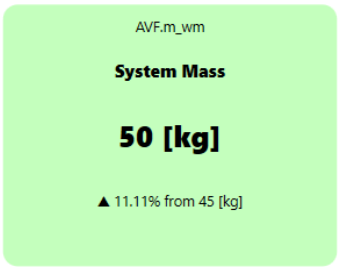

Figura 2.19. Dashboard della massa del sistema.

Un altro strumento molto versatile, per l'elaborazione e la visualizzazione dei dati, consiste nel Reporting. La funzionalità sfrutta un particolare formato di archivi, l'estensione rep4, che devono presentare all'interno un file Datacode.cs, ovvero uno script in C# per l'acquisizione e l'elaborazione delle informazioni del modello ingegneristico, e un file Report.repx, dove è definito lo schema per la generazione del documento di report, che descrive la visualizzazione delle informazioni tramite l'integrazione del DevExpress Reports, che sfrutta il framework .Net distribuito da Microsoft. Utilizzando il pulsante Reporting nella sezione Model, l'utente viene indirizzato all'interfaccia dedicata al Report Design (figura 2.20).

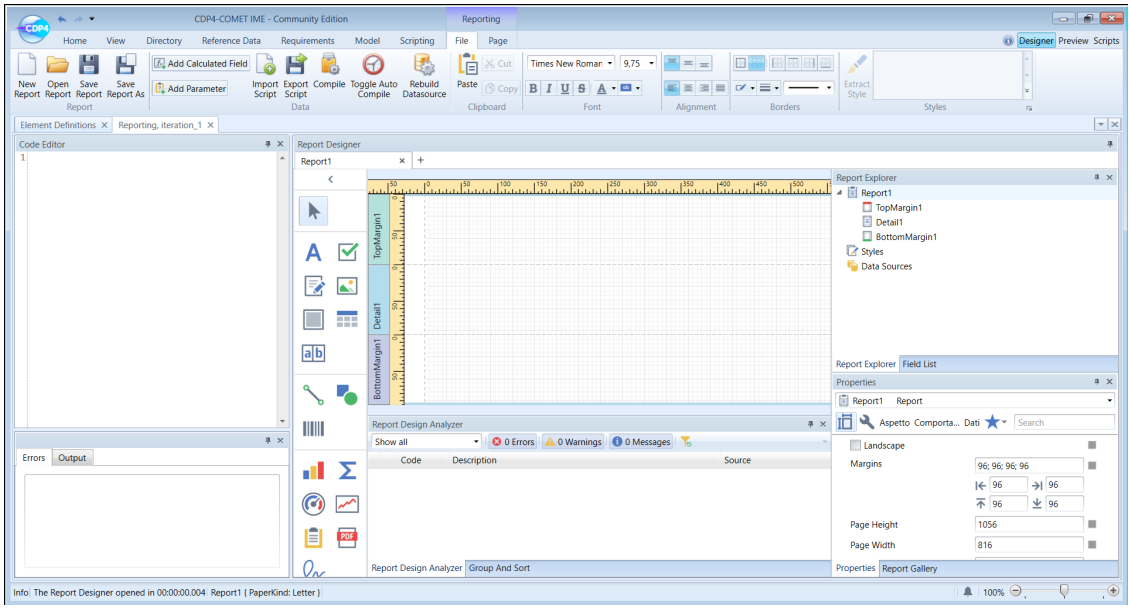

Figura 2.20. Interfaccia dedicata al Report Design.

In questa interfaccia, l'utente dispone di tre pulsanti Designer, Preview e Script, con cui impostare la schermata desiderata. L'interfaccia relativa al Designer presenta una serie di funzionalità dedicate alla formulazione e gestione dei Report, suddivise in due

sezioni, File e Page. Nella sezione File, è possibile gestire gli archivi di Report tramite i pulsanti della toolbar: si hanno a disposizione funzionalità per la creazione, l'importazione, l'esportazione e il salvataggio dei file con estensione rep4. Inoltre sono presenti pulsanti per la gestione e compilazione del codice della Datasource, l'aggiornamento dei dati dal modello ingegneristico e l'impostazione delle caratteristiche relative al documento da generare, come il font e la dimensione dei caratteri. La finestra del Reporting è composta da due sezioni principali, il Code Editor, in cui scrivere il Datasource, la relativa sezione di output ed errori prodotti dalla compilazione del codice ed il Report Designer, che integra le suddette funzionalità di DevExpress Reports per l'organizzazione e la visualizzazione delle informazioni prodotte dallo script, fornendo degli strumenti di alto livello per la compilazione del file Report.repx. Nella sezione Page, invece, i pulsanti della toolbar (figura 2.21) permettono di configurare la visualizzazione delle pagine del Report, impostando margini, dimensioni, orientamento, colore e watermark. Infine, utilizzando il pulsante Preview, è possibile visualizzare un'anteprima del Report ed esportare il file in diversi formati.
Dunque, la funzionalità Reporting permette di generare documenti di riepilogo dinamici, ovvero che si aggiornano automaticamente con i progressi nello studio di progettazione.

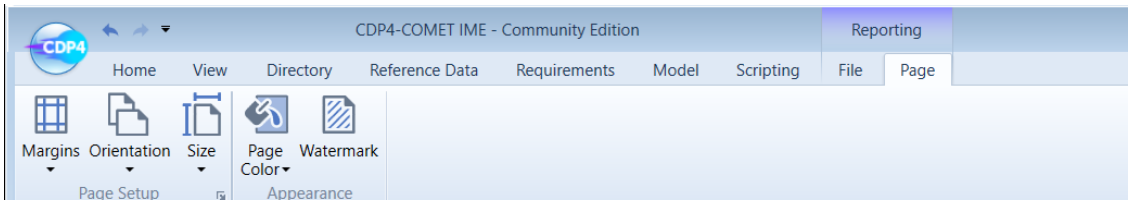

Figura 2.21. Toolbar per la configurazione delle pagine dei Report.

Graphical

Il software presenta alcune funzionalità grafiche che assistono i progettisti nelle attività di lavoro. Innanzitutto è possibile generare automaticamente dei diagrammi a blocchi a partire dalle opzioni definite, rappresentando l'architettura del Top Element costruita nel Product Tree. L'utente può selezionare la tipologia di diagramma da generare da una lista di tipi possibili ed esportare l'immagine prodotta.
Invece, il Relationships Editor consiste nell'ultima funzionalità disponibile per la creazione di relazioni. La possibilità di creare delle Binary Relationship Rules, come già accennato (sec 2.2.2), consente di specificare dei criteri su cui impostare delle Relationships. Queste Rules possono essere utilizzate graficamente in questa finestra (figura 2.22) per correlare qualsiasi tipologia di oggetto del software. Trascinando infatti oggetti come l'Element Definition di un componente o un Requirement nell'apposito editor grafico, vengono visualizzati dei riquadri che rappresentano gli oggetti selezionati. Utilizzando la Rule opportuna, è possibile congiungere graficamente i due oggetti, svolgere questa azione creerà la relazione, che potrà essere consultata nella finestra delle Relationships.

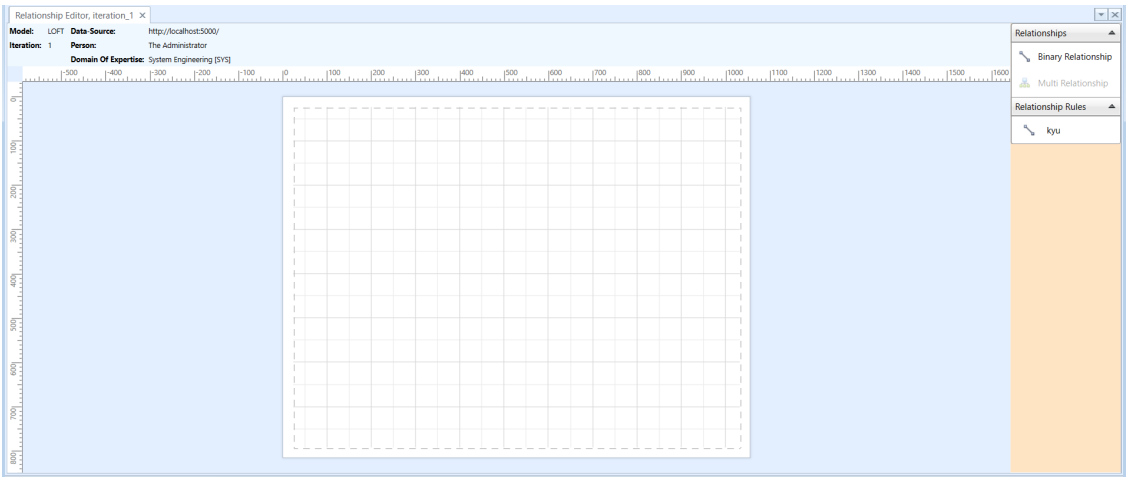

Figura 2.22. Interfaccia del Relationships Editor.

Model Verification

Tra le ultime funzionalità legate al modello ingegneristico, si descrivono quelle relative alla sua validazione. Innanzitutto è possibile ispezionare le Built-in Rules, qui è presente una Rule predefinita che verifica la validità degli ShortName nella definizione degli elementi. Ulteriormente è possibile verificare che non siano presenti errori all'interno degli oggetti utilizzati nel modello. Infine, tramite il pulsante della Rules Validation, è possibile creare delle liste di Rules da verificare sul modello, inserendo le singole Rules a discrezione dell'utente. In questo modo, si può valutare la congruenza del modello con gli approcci stabiliti all'interno del gruppo di progettazione. L'insieme delle Rules formulate infatti, costruisce una struttura, concordata all'interno del gruppo di progettazione, che chiarisce i concetti ingegneristici coinvolti nelle attività di progettazione, producendo una convenzione unica e condivisa. Ad esempio, stabilendo, tramite una Parameterized Category Rule, che tutti gli elementi categorizzati come Equipment debbano avere attributo il parametro di massa, è possibile verificare che questa logica di progettazione sia rispettata da tutti gli elementi definiti nel modello ingegneristico. L'utente rimarrà infatti svincolato dalle logiche definite dalle Rules, che non limitano le funzionalità del software, tuttavia, verificando le Rules desiderate, vengono evidenziati gli elementi che non rispettano il criterio in questione. Di conseguenza, la validazione delle Rules permette rapidamente l'individuazione degli oggetti del modello ingegneristico che esulano dalle logiche definite, consentendo di valutare, caso per caso, se è opportuno omologare l'oggetto alle Rules stabilite o se la violazione è accettabile.

2.5 Scripting e integrazioni con altri software

Infine, completando l'analisi delle funzionalità del software, l'ultima schermata accessibile è quella relativa allo Scripting. In questa sezione è possibile utilizzare i linguaggi di

programmazione Python 3.7 e Lua 5.x per compilare ed eseguire degli script attraverso i quali svolgere svariati tipi di elaborazioni. Ulteriormente, l'installazione del software client CDP4-CoMET IME comporta anche l'installazione di un add-on per Microsoft Excel. Grazie a questo add-on, viene creata una nuova scheda dedicata a Comet all'interno dell'interfaccia di Excel, integrando molte funzionalità di Comet alle tipiche possibilità dei fogli di calcolo. Infine si vuole analizzare un'applicazione standalone, creata sempre dalla casa di sviluppo Starion, chiamata DEHPMatlab, che consente di associare variabili da un workspace di matlab a parametri di progetto definiti in un modello ingegneristico di Comet. Queste integrazioni di Comet con altri tool comunemente utilizzati in ambito ingegneristico, vanno a costituire il Digital Engineering Hub (DEH), un ambiente di lavoro virtuale nel quale le esigenze e competenze multidisciplinari coinvolte nel progetto possono interfacciarsi tra di loro, permettendo lo scambio di dati tra i software specifici dei vari Domain of Expertise.

2.5.1 Scripting

Nella sezione dedicata allo scripting, il software propone alcuni pulsanti per creare, importare e salvare gli script. La finestra dedicata (figura 2.23) presenta diversi riquadri: lo script editor consente la scrittura e l'esecuzione del codice, in Local Variables vengono riportate tutte le variabili inizializzate nel codice e i valori a loro assegnati, in Input si possono inserire comandi da eseguire e infine in Output si ottengono i dati riportati dallo script, come print o eventuali errori di compilazione. Ulteriormente, è presente un selettore da utilizzare per indicare la sessione attiva: questo fa sì che lo script possa svolgere processi che sarebbero comunque permessi ai ruoli dell'utente che esegue il codice.

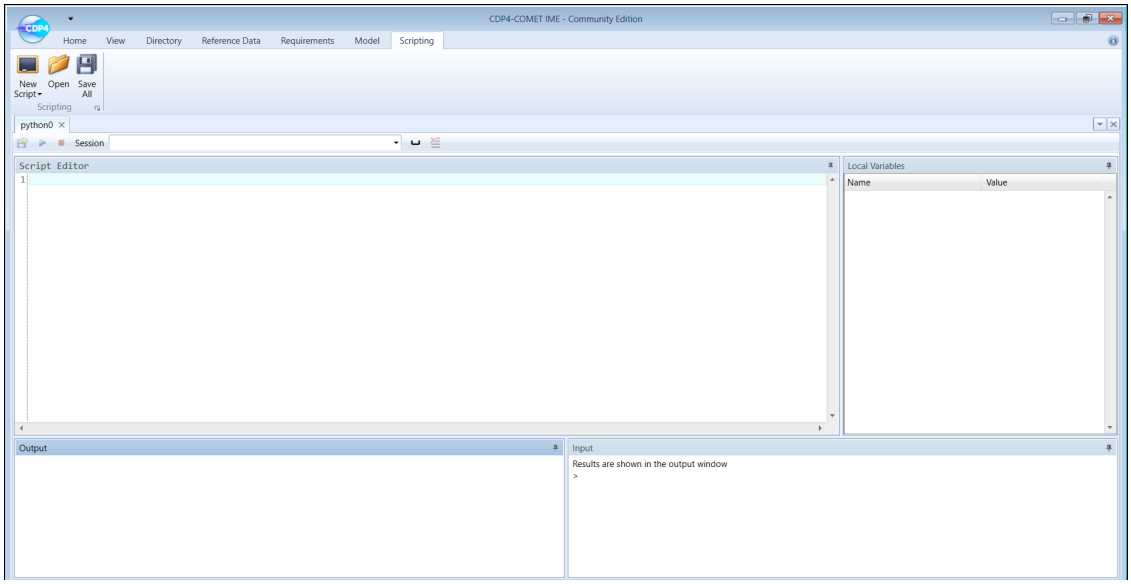

Figura 2.23. Scripting.

Questa funzionalità consente di creare degli script che, acquisendo le informazioni dal modello ingegneristico, sfruttino le potenzialità della programmazione a oggetti per elaborare i dati. I risultati ottenuti possono essere inseriti nuovamente all'interno del modello ingegneristico, oppure visualizzati o esportati per ulteriori valutazioni. Utilizzando lo scripting, si possono creare delle automazioni all'interno del modello ingegneristico, come ad esempio il calcolo della massa di un sottosistema a partire dalle masse dei componenti, aumentando considerevolmente la dinamicità del modello e la consistenza dei dati condivisi. Lo sviluppo di script si basa su comandi built-in implementati nel software: eseguendo Command.Help() nella sezione di Input, vengono mostrati tutti i comandi disponibili con una breve descrizione. Inoltre, formulare l'assegnazione di oggetti ed eseguire il codice, fa sì che l'editor stesso suggerisca, in un tooltip, gli attributi e i metodi disponibili, come nella figura 2.24.

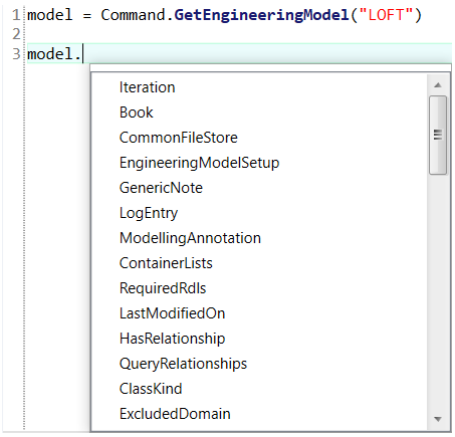

Figura 2.24. Visualizzazione di metodi e attributi nel tooltip.

2.5.2 CDP4-CoMET Web App

Tra le funzionalità offerte dalla casa di sviluppo Starion, è presente anche il software per l'hosting di una Web App di Comet[24]. Questo componente software aggiuntivo, permette di erogare la maggior parte delle funzionalità di Comet tramite servizi web, dunque accessibili con un qualsiasi browser, piuttosto che il client dedicato. L'obiettivo principale della web app è quello di fornire delle panoramiche sull'avanzamento del progetto. Nonostante ciò, è possibile utilizzare gran parte delle funzionalità descritte in questo capitolo, oltre a funzionalità esclusive, come la Model Dashboard riportata nella seguente figura 2.25.

In questa schermata è possibile valutare lo stato del progetto tramite una dei grafici che riportano delle panoramiche, ad esempio sulla completezza dei valori attribuiti ai parametri oppure sulla presenza di valori in attesa di pubblicazione. Una seconda funzionalità esclusiva della Web App è il 3D Viewer, dove è possibile visualizzare il modello ingegneristico. Questa funzionalità necessita l'attribuzione, nella definizione degli

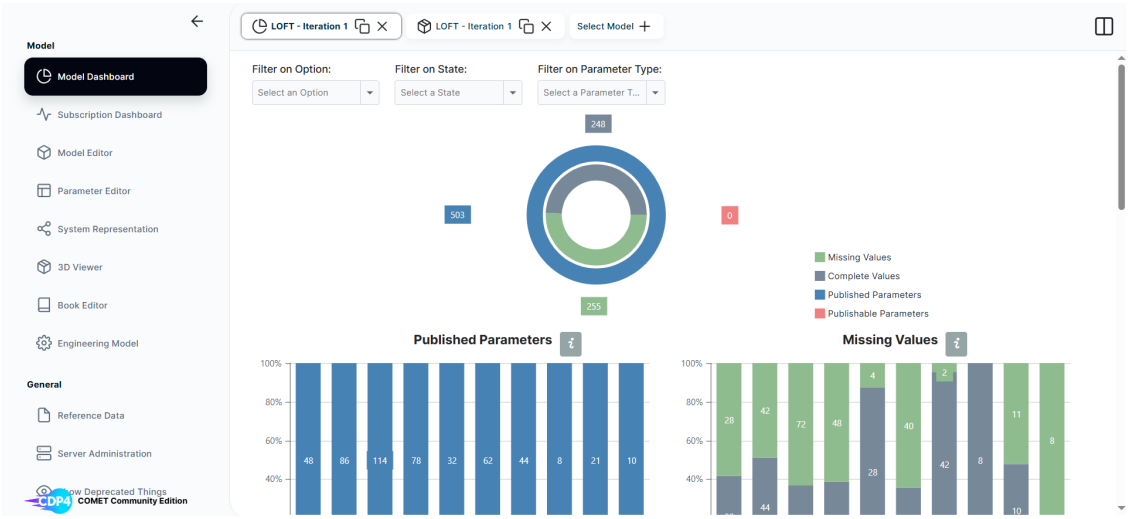

Figura 2.25. Model Dashboard della Web App.

elementi del modello, di informazioni sulla geometria e la posizione dell'elemento. In questo modo, vengono essere visualizzati i *blocchi* che compongono il sistema, permettendo anche valutazioni sulla configurazione dei componenti o sull'harness dei cablaggi.

2.5.3 Excel Add-In

L'integrazione di Comet in Excel permette di creare cartelle di lavoro, importando i dati da un modello ingegneristico. Tramite la toolbar in Excel introdotta da Comet, riportata in figura 2.26, si può aprire la schermata di autenticazione di Comet per il server e accedere a molte funzionalità.

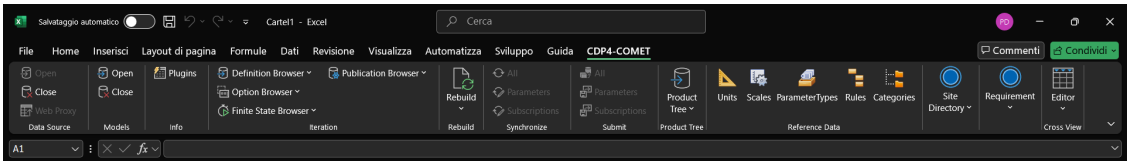

Figura 2.26. Toolbar di Excel dedicata a Comet.

Utilizzando il tasto Rebuild, viene creata automaticamente la cartella di lavoro Excel a partire dal modello ingegneristico selezionato in seguito all'autenticazione. Vengono quindi generati automaticamente dei fogli di calcolo, in particolare: un foglio chiamato Parameters che riporta l'Element Definition, con i rispettivi parametri, di tutti gli elementi di competenza del dominio dell'utente, ovvero quelli di cui il dominio è Owner o per cui esiste una Subscription. Inoltre, viene generato un foglio di calcolo per ogni

Option definita nel modello ingegneristico, che riporta l'architettura del sistema specifica per quella opzione di progetto. Dunque, il tasto Rebuild importa in Excel, su richiesta dell'utente, il workspace dedicato al suo dominio di appartenenza, permettendogli di sfruttare tutte le potenzialità di Excel nell'elaborazione delle informazioni, partendo dai dati del modello ed eventualmente inserendo nel modello i risultati ottenuti. Infatti, selezionando le celle nel foglio di calcolo Parameters, è possibile impostare delle relazioni con altre celle e utilizzare le funzioni di Excel per definire un'espressione che elabori il valore da attribuire al parametro. Una volta configurata la relazione desiderata, Excel produce automaticamente il valore di output a partire dagli input forniti, evidenziando il nuovo valore individuato nella casella. A questo punto, tra i pulsanti disponibili nell'integrazione di Comet in Excel, l'utente può utilizzare Submit per trasferire i cambiamenti al server. Dopo questa operazione, su Comet saranno disponibili i nuovi valori importati da Excel e sarà possibile valutare la Publication dei cambiamenti realizzati.

2.5.4 Integrazione con Matlab

Un'altra integrazione molto utile nel campo dell'elaborazione dei dati è il DEHP Matlab Adapter[25]. A differenza dell'add-on precedentemente descritto, questa risulta essere un'applicazione esterna sia a Matlab che a Comet che permette di collegare i due ambienti di lavoro. Questo software è open source ed è disponibile nella repository Github di Starion. Avviata l'applicazione, viene visualizzata l'interfaccia riportata in figura 2.27, in cui è possibile autenticarsi a un server Comet tramite l'indirizzo e le credenziali utente ed aprire un modello ingegneristico. Successivamente, il software procederà a localizzare l'installazione di Matlab per aprire la MATLAB Command Window. A questo punto, è possibile caricare script matlab dal dispositivo; questi possono essere eseguiti direttamente tramite l'applicazione, elaborando le variabili nel workspace associato a Matlab. Le variabili vengono distinte tra input e output dello script, permettendo di mappare le variabili del workspace di Matlab ai parametri del modello ingegneristico, associando non solo i valori ma proprio gli oggetti parametro e variabile. Infatti, se eseguendo nuovamente lo script i valori dovessero cambiare, i due oggetti rimarrebbero ancora associati. È possibile salvare le configurazioni di associazione, per utilizzarle quando necessario. Questa associazione di variabili di Matlab a parametri di progetto di Comet può avvenire sia importando da Comet verso variabili di input dello script, sia esportando variabili di output elaborate dal codice verso i parametri degli elementi.Infine, bisogna utilizzare il tasto Transfer per ultimare la transazione dei valori selezionati verso Comet, importandoli effettivamente all'interno del workspace del dominio di appartenenza dell'utente. Anche in questo caso, per rendere disponibili i nuovi valori all'interno dell'intero gruppo di progettazione, e dunque tra tutti i Domain of Expertise, è necessario pubblicare le proposte di modifica dei parametri.

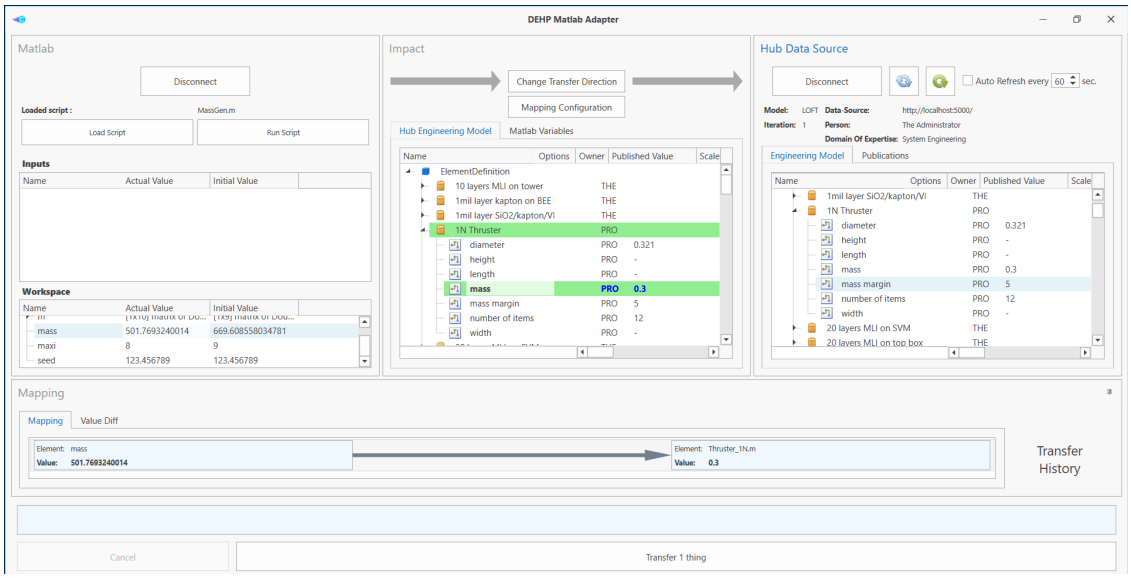

Figura 2.27. DEHP Matlab Adapter.

Capitolo 3

Implementazione del software CDP4 - CoMET nelle attività di progettazione di Space V

In questo capitolo si descrive il lavoro svolto nell'integrazione del software nei processi di ideazione e sviluppo di Space V. La startup opera in un contesto ingegneristico fortemente collaborativo, a stretto contatto con diverse PMI del settore spaziale, dove la partecipazione di organizzazioni esterne si integra nei vari livelli di progettazione, dallo sviluppo dei componenti alla configurazione dei sottosistemi. Comet risulta essere uno strumento fondamentale, che può agevolare la condivisione delle informazioni e, al contempo, aumentare la consistenza dei dati, migliorando la qualità dello scambio stesso.
Poichè l'interesse attuale di Space V è quello di svolgere uno studio di fattibilità, l'implementazione del software consiste nella realizzazione di un Template Model, utile come punto di partenza per la realizzazione di futuri studi di progettazione, e la creazione di uno Study Model, basato su precedenti analisi di spazializzazione del PoC terrestre, al fine di analizzare e valutare le potenzialità del software nel supporto delle attività di progettazione.
Per valutare le effettive esigenze nelle attività e per individuare l'approccio per l'implementazione delle funzionalità del software, sono stati coinvolti direttamente i membri della startup Space V. Dunque, sono state presentate le funzionalità di Comet, tramite l'utilizzo di un modello *DEMO* che potesse documentare gli scopi principali del software. Quindi, sono stati valutati gli approcci più opportuni per far aderire le funzionalità del software alle esigenze interne a Space V e di cooperazione esterna con le PMI coinvolte.
L'implementazione del software comincia dall'installazione delle sue componenti. Risulta infatti necessario dotarsi dell'applicazione desktop CDP4-COMET IME, che implementa, per gli utenti, le funzionalità descritte nel capitolo precedente. Tuttavia, come già accennato, il software ha bisogno anche di una componente server che gestisca le richieste di informazioni degli utenti. Per questo motivo, la casa di sviluppo del software fornisce l'indirizzo di un server pubblico di test a cui accedere tramite Internet e con credenziali predefinite. Questo consente di familiarizzare con le funzionalità del software prima di iniziare a svolgere vere e proprie attività di progettazione. Il server di test, infatti, viene

ripristinato quotidianamente permettendo all'utente di esplorare il software senza la possibilità di compromettere il modello di esempio. Questa peculiarità però non consente in nessun modo di preservare i dati da una sessione a un'altra. Per svolgere lavori di più lungo termine, come quello descritto in questo capitolo, è necessario eseguire l'hosting della componente server del software. Il server CDP4-COMET Web Services necessita di almeno una CPU e 4GB di RAM ed è raccomndata la presenza di 50GB di spazio libero su disco. Per quanto riguarda il sistema operativo, il server richiede l'ambiente Linux per essere eseguito correttamente. È possibile, tuttavia, utilizzare il Windows Subsystem for Linux (WSL) per implementare la compatibilità del server con macchine con sistema operativo Windows, senza dover utilizzare macchine virtuali Linux. Ad esempio, è possibile utilizzare il software Docker per il dispiegamento del server. Docker è un software di gestione di Container, ovvero istanze di spazi virtuali per l'allocazione di risorse per la virtualizzazione a livello di sistema operativo. Disponendo di questo software, l'hosting del server è possibile semplicemente importando l'immagine distribuita direttamente da Starion. È sufficiente, infatti, clonare la repository, contenente CDP4-COMET Web Services, direttamente da Github ed eseguire il comando docker compose up nella directory appena clonata. In seguito, sarà possibile utilizzare l'app desktop Docker per gestire lo stato del server.
Sebbene l'utilizzo di WSL sia sconsigliato, e si dovrebbe preferire una distribuzione Linux, per mancanza di una macchina dedicata su cui installare il sistema operativo, e per la semplicità dell'utilizzo dei Container, è stato utilizzato il software Docker per l'erogazione dei Web Services.
La Concurrent Design Facility dell'ESA fornisce ulteriori raccomandazioni per l'installazione del software e la configurazione del server. Ad esempio, è consigliabile l'utilizzo di un SSD per l'archiviazione, al fine di garantire ottime prestazioni in termini di velocità di accesso ai dati. Inoltre, è consigliato eseguire regolarmente dei backup per l'eventuale ripristino dei dati in caso di anomalie, scaricando tutti i dati del database e salvandoli in un file esterno. Infine, è fortemente consigliato eseguire il server dietro un reverse proxy che possieda un certificato SSL al fine di garantire la sicurezza e la validità delle informazioni.
Nell'ambito di questo lavoro di Tesi, il server è stato eseguito tramite Docker su una rete locale, consentendo agli altri utenti connessi alla medesima rete di individuare correttamente l'indirizzo del server. In questo modo è stato possibile cooperare, quando necessario, per la definizione e la verifica dei dati sin dalle prime fasi della configurazione. Chiariti i requisiti, i processi e le raccomandazioni utili al dispiegamento del server, si descrive l'utilizzo di Comet nell'ambito di progettazione di Space V.

3.1 Studio di Fattibilità

L'obiettivo attuale di Space V, come già anticipato, è quello di svolgere uno studio di fattibilità sulla proposta di missione spaziale formulata. Si vuole valutare il livello di maturità tecnologica di numerose e differenti tipologie di dispositivi, approfondendo la letteratura scientifica, per individuare le tecnologie che possano assolvere alle funzioni desiderate e valutare trade-off delle possibili soluzioni, rispettando vincoli di fattibilità

sia tecnologici sia economici. Al fine di svolgere questo studio, è stato realizzato un Template Model che contiene informazioni generiche relative al progetto e in cui sono stati predisposti gli aspetti rilevanti. Dunque, il Template Model potrà costituire un punto di partenza per le attività di progettazione.

3.1.1 Directory

Per impostare correttamente le informazioni del software, si parte dalla formulazione dei concetti della Site Directory.
Iniziando dallo User Management, sono stati definiti i Domain Of Expertise utili per l'ambito di applicazione di Space V. In particolare sono stati creati i domini:

- Agronomics (AGR): per le consulenze relative all'agronomia;
- Harness and Connections (HC): per le valutazioni relative alla configurazione dei cablaggi del sistema;
- Illumination (LMN): per le valutazioni relative all'illuminazione per la crescita delle piante;
- Sensors (SEN): per le valutazioni relative al sottosistema di sensoristica atto al controllo ambientale;
- Shelf Handling (SH): per le valutazioni relative al sottosistema di movimentazione dei ripiani;
- Ventilation (VEN): per le valutazioni relative alla ventilazione delle piante;
- Water Management (WM): per le valutazioni relative all'apporto di acqua e nutrienti alle piante.

Questi domini personalizzati sono stati integrati con altri domini predefiniti al fine di individuare i domini attivi nel setup del modello ingegneristico. Sono stati ulteriormente inclusi: Control & Data Handling (CDH), Power (PWR), Structures (STR), System Engineering (SYS) e Thermal (THE). I domini ricoprono le funzioni che il sistema dovrà svolgere nell'ambito della missione. Per questo motivo, i domini attivi nell'ambito di un modello ingegneristico sono spesso associati a un sottosistema, poiché anche la definizione di questi ultimi scaturisce dall'analisi funzionale del sistema e della missione che si intende svolgere. Infatti, nel caso in questione, solo System Engineering e Harness and Connections non si riflettono direttamente su un sottosistema, bensì sono domini responsabili dell'integrazione dei sottosistemi nel sistema, con l'obiettivo di favorire una visione olistica. Per quanto riguarda il dominio dell'agronomia, questo non verrà utilizzato come dominio attivo, poiché da questo campo di studi non derivano valutazioni ingegneristiche, piuttosto sarà attribuito agli esperti di questo settore, congiuntamente ad altri domini, per poter disporre delle informazioni necessarie per consulenze.
Successivamente, sono state definite, all'interno del software, le organizzazioni: Space V, Agenzia Spaziale Italiana e il *placeholder* Generic Small Medium Enterprise. Questo

sarà utile per selezionare la startup come organizzazione proprietaria del modello ingegneristico e le altre come partecipanti, oltre ad attribuire agli utenti l'appartenenza alle rispettive organizzazioni. Infine sono stati definiti degli utenti *placeholder* che possono essere sfruttati in futuro per l'accesso oppure come riferimento per la creazione di altri utenti. È stato creato un utente chiamato Space V Member, ed è stato assegnato il ruolo con i permessi più completi, ovvero Site Administrator, come Person Role. All'utente è stato attribuito il Dominio Default del System Engineering, ovviamente l'appartenenza all'organizzazione Space V e delle credenziali di default per l'accesso al server. È stato creato un secondo utente, con lo scopo di simulare un'autorità di controllo coinvolta nel progetto. È stato attribuito il Person Role di Line Manager, affinché possa accedere alle informazioni relative all'azienda senza poterle modificare, e l'appartenenza all'organizzazione Agenzia Spaziale Italiana (ASI). L'ultimo utente, invece, rappresenta un fornitore e gli è stato assegnato il ruolo di Concurrent Design Team Member, permettendogli di accedere esclusivamente alle informazioni dei progetti a cui partecipa. Inoltre, è stato associato all'organizzazione Generic Small Medium Enterprise.
Infine, è stata definita la Generic Space V Reference Data Library, una RDL a livello Site, che contiene i concetti ingegneristici utili a Space V, nel suo specifico ambito di progettazione. Questa RDL è stata creata come estensione della RDL predefinita Generic ECSS-E-TM-10-25 Reference Data Library, dunque include il riferimento agli oggetti contenuti al suo interno, oltre a quelli definiti appositamente.
Disponendo di tutti gli elementi necessari per la creazione dell'Engineering Model Setup, è stato inizializzato il Template Model come riportato in figura 3.1.

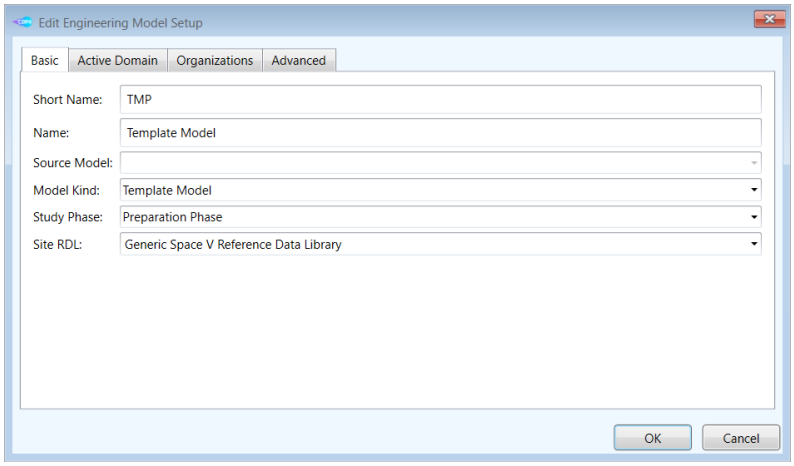

Figura 3.1. Engineering Model Setup per il Template Model.

Oltre alle informazioni visualizzate nell'immagine, sono stati impostati gli Active Domain precedentemente individuati, le organizzazioni definite appositamente e sono stati coinvolti gli utenti creati come partecipanti. In particolare: all'utente Space V Member è stato assegnato il ruolo di Model Administrator ed il dominio attivo del System Engineering, come mostrato in figura 3.2; all'utente Control Authority è stato assegnato il ruolo di Observer e tutti i domini attivi; infine, all'utente Supplier è stato assegnato il ruolo

Technical Author e il dominio Illumination.

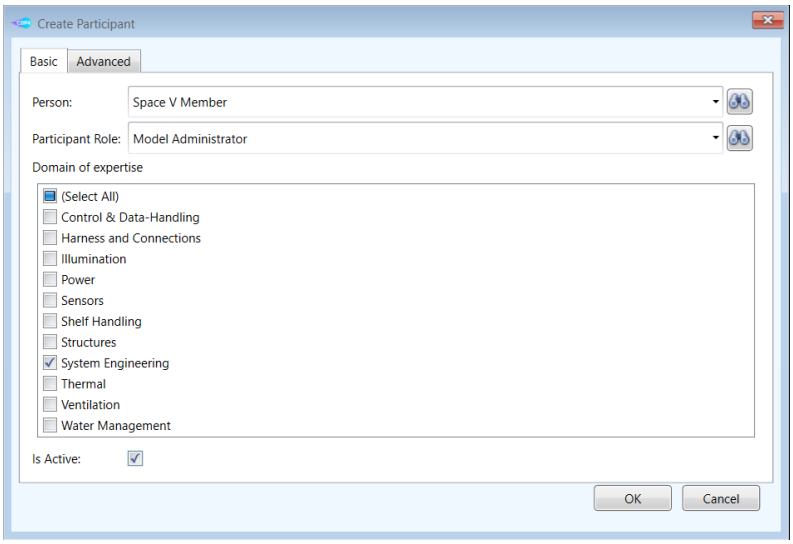

Figura 3.2. Esempio di creazione di un partecipante.

La configurazione del modello ottenuta è mostrata nella sezione Directory nella finestra Models, ed è riportata in figura 3.3

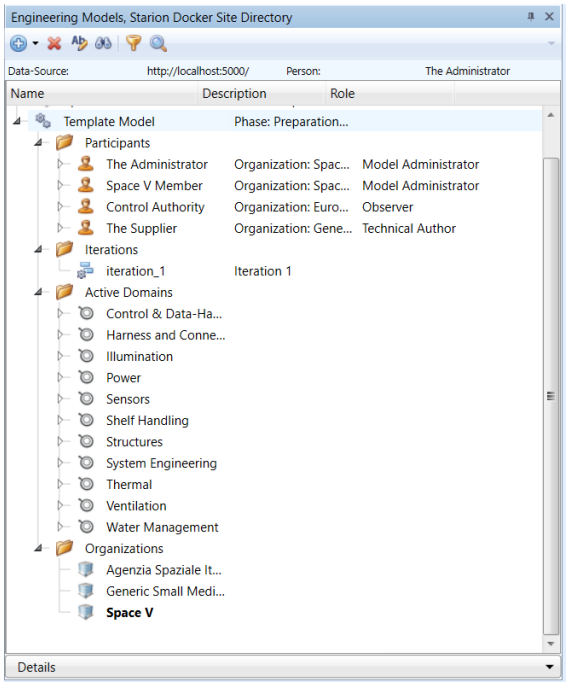

Figura 3.3. Configurazione completa del Template Model.

3.1.2 Reference Data Library

Si procede a caratterizzare lo spazio di lavoro di Comet introducendo gli aspetti della missione e le metodologie di progetto di Space V nella RDL creata appositamente *Generic Space V Reference Data Library*.

Reference Data per l'analisi dei requisiti

I primi concetti inseriti nella RDL sono stati individuati al fine di formulare i Requirements e di elaborare il tracciamento di questi in tutte le fasi di progettazione. Sono stati definiti i parametri Requirement Verification Method, riporato in figura 3.4, e Requirement Product Level, e tre Categories, con le relative Rules, per strutturare l'organizzazione dei requisiti. In particolare, il parametro Requirement Verification Method verrà associato ai requisiti al momento della loro definizione, consentendo di selezionare da un elenco predefinito, il metodo di verifica appropriato per ciascun requisito.

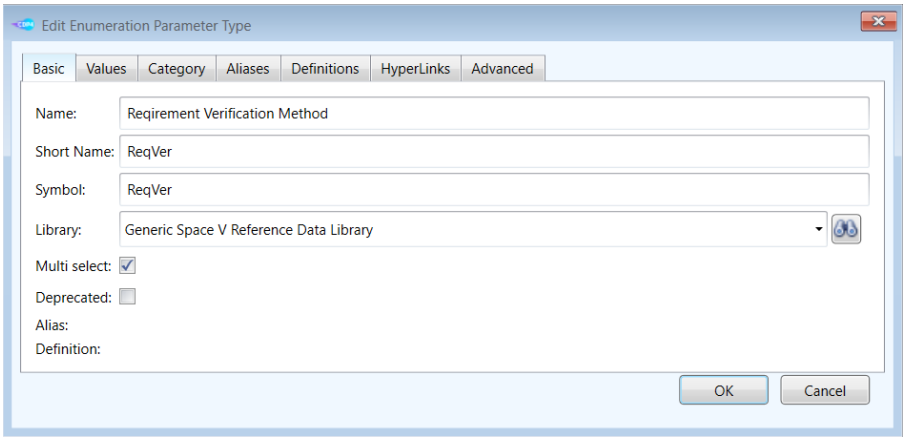

Figura 3.4. Creazione del parametro Requirement Verification Method.

Il Requirement Verification Method è infatti un parametro di tipo enumerativo con quattro valori (figura 3.5) associati:

- Review, la validazione del requisito avviene nella revisione del progetto preliminare;
- Analysis, la validazione del requisito avviene in una fase di design dettagliato del progetto, quando verranno svolte analisi numeriche approfondite;
- Inspection, la validazione del requisito avviene tramite l'ispezione visiva, in una fase di produzione del sistema;
- Test, la validazione di questo requisito avviene nella fase di validazione del sistema stesso.

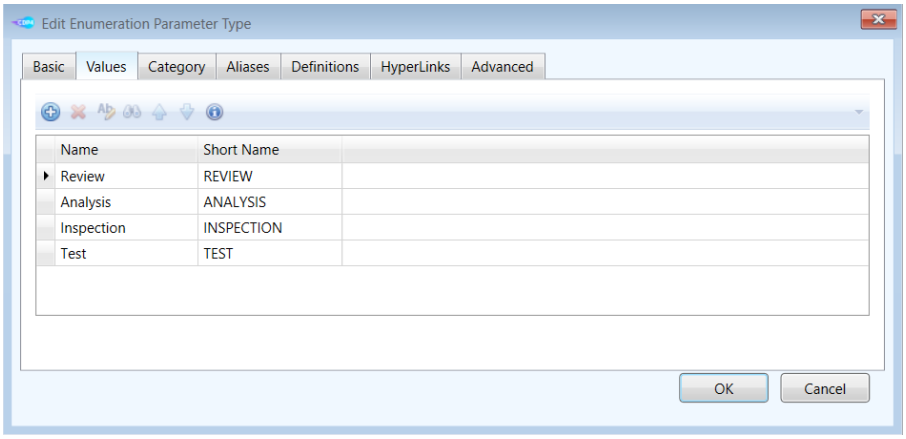

Figura 3.5. Valori possibili per il parametro Requirement Verification Method.

Attribuendo questo parametro ai Requirements, si potrà tracciare la verifica dei requisiti con l'avanzare delle fasi del progetto.
Inoltre, per fornire ulteriore contestualizzazione dei requisiti di progetto, potrà essere utilizzato il parametro enumerativo Requirement Product Level, con i possibili valori:

- Mission Level Requirement (MLR);
- System Level Requirement (SLR);
- Subsystem Level Requirement (SSLR);
- Equipment Level Requirement (EQLR).

Questi valori potranno essere assegnati al requisito per fornire un'indicazione, di rapido accesso, del soggetto di riferimento del requisito stesso.
In seguito, sono state definite le categorie necessarie per creare la struttura per la decomposizione dei Requirements. Sono state create tre categorie da assegnare ai requisiti di progetto, a seconda del livello di definizione di questi ultimi, ovvero:

- High Level, i requisiti fondamentali del progetto ingegneristico che scaturiscono direttamente dal mission statement e dai vincoli di missione ad esso associati;
- Design Level, sono generati dai requisiti di alto livello e descrivono, in maniera generica, le funzionalità e le caratteristiche del sistema e degli elementi che lo compongono;
- Detail Level, sono prodotti dai Design Level e descrivono nello specifico una prestazione o una caratteristica del sistema e degli elementi che lo compongono.

Nella figura 3.6 è riportata la creazione della categoria High Level, a scopo esemplificativo. Nella scheda Permissible Classes, è stata selezionata la voce Requirements affinchè queste categorie possano effettivamente essere applicate agli oggetti di questa tipologia.

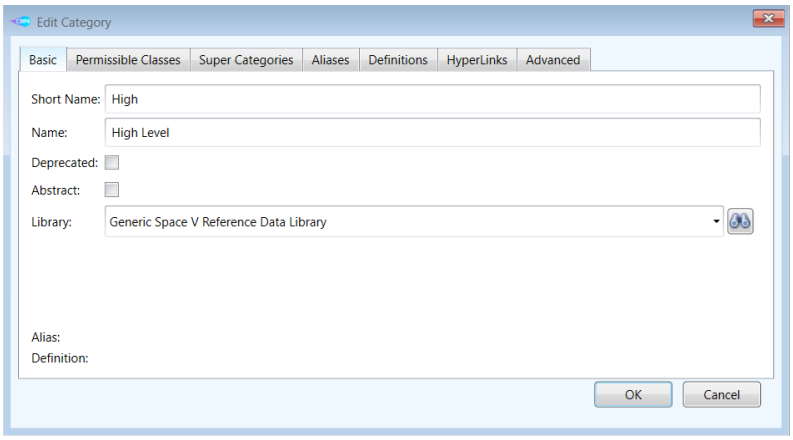

Figura 3.6. Creazione della categoria High Level per i Requirements.

A queste categorie sono state associate due Binary Relationship Rule che rappresentano la relazione padre-figlio tra la categoria Source e la categoria Target, utili a scomporre, quindi, gli High Level in Design Level e a loro volta i Design Level in Detail Level. Dopo aver definito i requisiti di progetto e applicato queste categorie, è possibile implementare un tracciamento della decomposizione dei requisiti tramite le Relationships, correlando tra loro i Requirements utilizzando la Rule di decomposizione opportuna. Inoltre, le due Rules sono state categorizzate come Requirements Decomposition Relationships, una categoria creata appositamente per queste Rules, così che anche le Relationships definite a partire da queste Rules siano categorizzate in questo modo. Questa operazione verrà svolta per lo Study Model. La creazione della Rule Requirements High to Design Level Decomposition è riportata in figura 3.7.

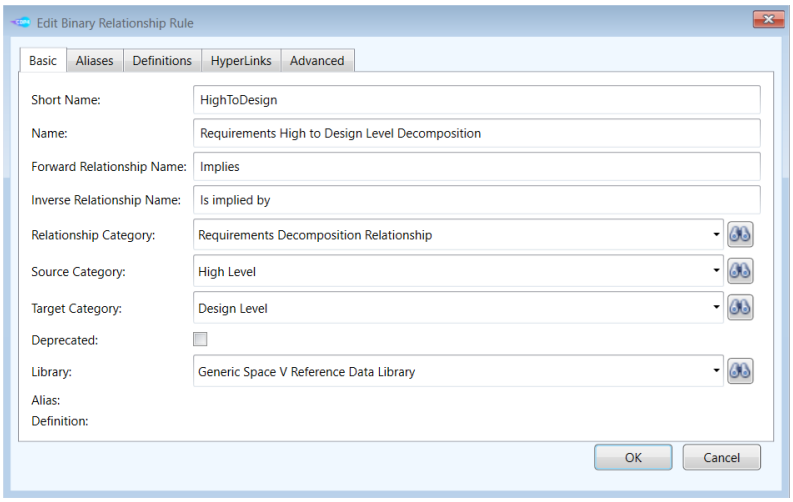

Figura 3.7. Creazione della Rule Requirements High to Design Level Decomposition.

System Maturity Level

Sono stati introdotti i parametri necessari per la valutazione del System Maturity Level (SML)[26], un indice che rappresenta, appunto, il livello di maturità di un insieme di N elementi caratterizzati da TRL. A seconda degli elementi considerati, potrà essere valutato il SML di un intero sistema o dei singoli sottosistemi. L'elaborazione del SML si basa sull'attribuzione dell'Integration Readiness Level (IRL), ad ogni coppia di elementi possibile all'interno dell'insieme considerato. L'attribuzione degli IRL avviene in maniera del tutto analoga all'attribuzione del TRL, ovvero tramite un numero da 1 a 9 che indica il raggiungimento, in maniera comprovata, di certi livelli di integrazione tra due specifici elementi. In questo modo sarà possibile costruire una matrice NxN che fornisca gli IRL precedentemente individuati, riportando il valore massimo 9 se si sta considerando l'integrazione di un elemento con sé stesso o tra due elementi non correlati. La matrice così prodotta sarà simmetrica, con tutti i valori della diagonale pari a 9. L'indice ricercato si ottiene valutando il prodotto della matrice precedentemente costruita per il vettore contenente i TRL degli elementi considerati, dunque a sua volta di dimensione N. Prima di svolgere il prodotto, i valori di TRL e IRL vengono normalizzati, dividendo il valore massimo della scala, ovvero 9. Si ha che:

$$\begin{bmatrix} IRL_{11} & IRL_{12} & \dots & IRL_{1N} \\ IRL_{21} & IRL_{22} & \dots & IRL_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ IRL_{N1} & IRL_{N2} & \dots & IRL_{NN} \end{bmatrix} \begin{bmatrix} TRL_1 \\ TRL_2 \\ \vdots \\ TRL_N \end{bmatrix} = \begin{bmatrix} SML_1 \\ SML_2 \\ \vdots \\ SML_N \end{bmatrix}$$

Il valore del SML si ottiene dalla media algebrica delle componenti del vettore, rapportate a N:

$$SML = \frac{1}{N} \sum_{i=1}^{N} (SML_i/N)$$

Il valore di SML risulta essere compreso tra 0.1, perché TRL e IRL non possono essere uguali a zero, e difficilmente raggiunge valori elevati come 0.9, poiché spesso si accettano tecnologie e integrazioni non completamente mature. Per integrare questa analisi nel modello, sono stati definiti i parametri IRL, parametro enumerativo totalmente analogo al TRL, ed il parametro SML, Specialized Quantity Kind del parametro fraction. Tuttavia, poiché questo tipo di analisi risulta troppo specifica per la fase attuale del progetto, non sono state svolte ulteriori implementazioni.

Prestazioni delle camere di crescita

Sono stati definiti diversi parametri utili alla modellazione dell'andamento della crescita delle piante e alla valutazione della resa delle camere di crescita. Vengono presentati di seguito i parametri introdotti, riportando Simbolo, Nome, Measurement Scale e Quantity Kind del parametro.

Symbol	**Name**	**Unit**	**Quantity Kind**
A	Fraction of PPF absorbed by canopy	-	Specialized Quantity Kind
t_A	Time for full canopy cover	d_AE	Specialized Quantity Kind
t_Q	Onset of canopy senescence	d_AE	Specialized Quantity Kind
t_M	Maturity time (harvest readiness)	d_AE	Specialized Quantity Kind
CQY	Canopy Quantum Yield	$\frac{\mu mol_{C_{fixed}}}{\mu mol_{Ab_{PPF}}}$	Derived Quantity Kind
CUE	Carbon Use Efficiency	-	Specialized Quantity Kind
DCG	Daily Carbon Gain	$\frac{mol_C}{m^2 \cdot d}$	Derived Quantity Kind
DOP	Daily Oxygen Production	$\frac{mol_{O_2}}{m^2 \cdot d}$	Derived Quantity Kind
OPF	Oxygen Production Fraction	$\frac{mol_{O_2}}{mol_C}$	Derived Quantity Kind
CGR	Crop Growth Rate	$\frac{g}{m^2 \cdot d}$	Derived Quantity Kind
BCF	Biomass Carbon Fraction	-	Specialized Quantity Kind
TCB	Total Crop Biomass	$\frac{g}{m^2}$	Derived Quantity Kind
TEB	Total Edible Biomass	$\frac{g}{m^2}$	Specialized Quantity Kind
XFRT	Fraction of daily carbon gain for edible parts	-	Specialized Quantity Kind
t_E	Onset of storage organ formation	d_AE	Specialized Quantity Kind
H	Photoperiod	$\frac{h}{d}$	Specialized Quantity Kind
MWC	Molar Mass of Carbon	$\frac{g}{mol}$	Derived Quantity Kind
VPD	Vapour Pressure Deficit	kPa	Specialized Quantity Kind
VPSAT	Saturation Vapour Pressure	kPa	Specialized Quantity Kind
VPAIR	Vapour Pressure	kPa	Specialized Quantity Kind

Continua nella pagina successiva...

Symbol	Name	Unit	Quantity Kind
T_{LIGHT}	Mean atmospheric temperature during light cycle	°C	Specialized Quantity Kind
RH	Relative Humidity	-	Specialized Quantity Kind
P_{GROSS}	Gross Canopy Photosynthesis	$\frac{\mu mol_C}{m^2 \cdot s}$	Specialized Quantity Kind
P_{NET}	Net Canopy Photosynthesis	$\frac{\mu mol_C}{m^2 \cdot s}$	Specialized Quantity Kind
DPG	Day Phase Duration in Growth Chamber	$\frac{h}{d}$	Specialized Quantity Kind
g_C	Canopy Surface Conductance	$\frac{mol_{H_2O}}{m^2 \cdot s}$	Derived Quantity Kind
g_S	Canopy Stomatal Conductance	$\frac{mol_{H_2O}}{m^2 \cdot s}$	Derived Quantity Kind
g_A	Atmospheric Aerodynamic Conductance	$\frac{mol_{H_2O}}{m^2 \cdot s}$	Derived Quantity Kind
DTR	Daily Canopy Transpiration Rate	$\frac{L_{H_2O}}{m^2 \cdot d}$	Derived Quantity Kind
yield	Crop Yield	-	Specialized Quantity Kind

Tabella 3.1: Parametri introdotti nella Generic Space V Reference Data Library per la valutazione delle prestazioni della camera di crescita.

La definizione di questi parametri ha implicato, il più delle volte, la definizione della relativa Measurement Unit e Measurement Scale come, ad esempio, la Prefixed Unit µmol o la Derived Unit $\frac{\mu mol}{m^2 s}$.
Inoltre, sono stati definiti dei parametri ed associati dei valori costanti:

Symbol	Name	Value	Unit
P_0	Atmospheric Pressure at Sea Level	101.325	kPa
ρ_W	Water Density	1	$\frac{kg}{L}$
MM_C	Molar Mass of Water	12.001	$\frac{kg}{mol}$
MM_w	Molar Mass of Water	18.01528	$\frac{g}{mol}$

Tabella 3.2. Costanti introdotte nella Generic Space V Reference Data Library.

Rules per gli elementi del modello

Per assodare la definizione di alcune categorie di elementi, sono state definite delle Parameterized Category Rules che possano implementare gli approcci alla progettazione adottati da Space V, formulando nel software le ipotesi assunte in fase di realizzazione del modello. È stata creata la Rule Subsystem Parameter Type, riportata in figura 3.8, ed è stata deprecata la Rule predefinita Equipment Parameter Type, per crearne una personalizzata. Queste Rules stabiliscono i parametri che dovrebbero sempre essere associati agli elementi categorizzati rispettivamente come Subsystem ed Equipment. In particolare, per i Subsystem sono stati individuati i parametri System Maturity Level (SML) e fornitore (supplier) e sono stati definiti appositamente i parametri massa con margine (m_wm), specializzazione del concetto di massa, e analogamente potenza consumata nominativa con margini (P_mean_wm) e potenza di picco con margini (P_peak_wm). Per quanto riguarda gli Equipment, si è stabilito che debbano possedere i parametri: fornitore (supplier), potenza consumata nominativa (P_mean), potenza di picco (P_peak), margine di massa (mass_margin) e numero di oggetti (n_items), oltre ai parametri di massa (m) e TRL, precedentemente previsti dalla Rule predefinita.

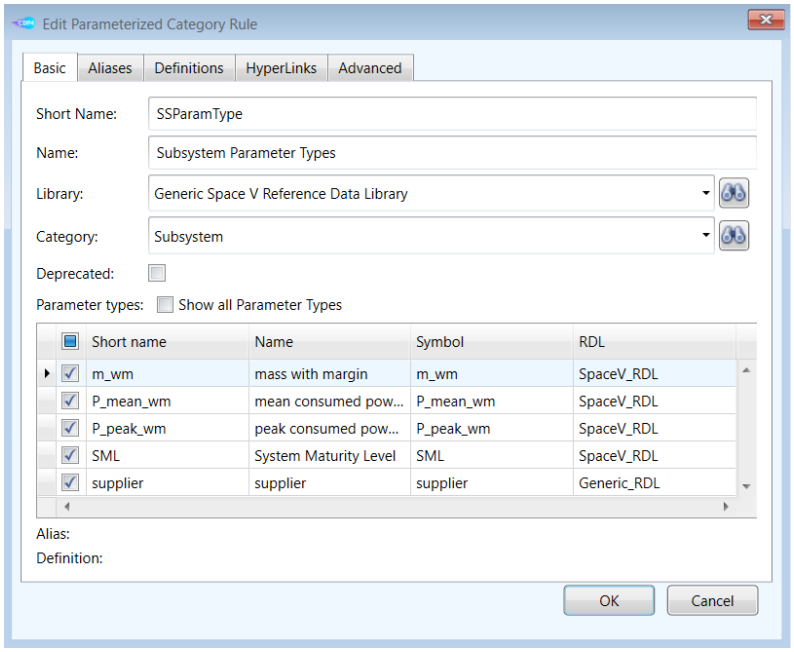

Figura 3.8. Creazione della Rule Subsystem Parameter Types.

3.1.3 Template Model

Nel Template Model si è scelto di rappresentare il progetto della missione spaziale con il grado di dettaglio di definizione dei sottosistemi. Quindi sono stati creati gli elementi: Adaptive Vertical Farm Demonstration Mission (MIS), categorizzato come Product, e Adaptive Vertical Farm (AVF) categorizzato come System. L'elemento MIS è stato

impostato come Top Element, di conseguenza sarà l'unico ad essere automaticamente implicato nel Product Tree, e sono stati attribuiti dei parametri all'Element Definition utili per valutazioni di carattere generico sull'intero sistema, ovvero:

- le tre dimensioni, lunghezza, larghezza e altezza;
- massa con margine;
- volume di ingombro;
- potenza consumata nominale con margine e potenza consumata di picco con margine;
- System Maturity Level.

Poichè la missione analizzata consiste nell'integrazione del sistema all'interno di un sistema ancora più complesso, ovvero una stazione spaziale, sono state definite tre categorie e creati i rispettivi elementi, al fine di includere, nel modello della missione, le interfacce tra la stazione spaziale ed il sistema. Sono state definite le categorie: Space Station, Space Station Interfaces e Space Station Interfaces Modules e sono state assegnate rispettivamente agli elementi International Space Station (ISS), EXPRESS Rack e Middle Locker (MDL). La modellizzazione delle interfacce così ottenuta, consente di valutare le specifiche tecniche di un singolo modulo, in questo caso il MDL. All'elemento categorizzato con Space Station Interfaces Modules sono stati attribuiti gli stessi parametri di summary assegnati anche al sistema, e altri parametri utili alla valutazione dell'integrazione di alcuni sottosistemi, come il data rate ($kbit/s$), per dimensionare il sottosistema Control and Data Handling, la portata d'acqua (m^3/s) e il flusso di calore dissipato (W), per le valutazioni relative al Thermal Subsystem, e la portata d'aria per il Ventilation Subsystem. Le due portate volumetriche di aria ed acqua sono state inserite appositamente nella RDL generica di Space V e sono state definite come Specialized Quantity della generica portata in volume.
Per continuare lo sviluppo dell'architettura del sistema, sono stati definiti: *Subsystem Placeholder*, *Element Placeholder* ed *Equipment Placeholder*, ed è stata attribuita la categoria da cui prendono il nome. A questi elementi *placeholder* sono stati attribuiti i parametri desiderati, come da definizione delle rispettive Parameterized Category Rules, così che possano essere utilizzati nelle successive attività di progettazione. Copiando l'Element Definition del *placeholder*, viene generato un elemento, identico anche nei parametri attribuiti, a cui è possibile modificare Name e Shortname. Eventualmente può essere svolta l'operazione *Change Ownership* per modificare il dominio a cui l'Element Definition è attribuita, spuntando la casella *Include Contained Items* se si vogliono modificare anche i parametri contenuti nell'elemento. Svolgere la definizione degli elementi tramite copia di un *placeholder* garantisce la corretta categorizzazione degli elementi e l'attribuzione dei parametri necessari, aumentando la consistenza degli elementi definiti con gli approcci stabiliti. In questa maniera, sono stati definiti i sottosistemi che compongono l'architettura del sistema e sono stati attribuiti i relativi domini di competenza. Per quanto riguarda gli scopi di questo modello, l'Element Definition risulta completata e si è potuto procedere con l'utilizzo degli elementi nello schema a blocchi. Partendo

dalla Missione, impostata come Top Element, è stato subordinato sia AVF, il sistema, sia l'International Space Station con la suddivisione in Interfaces e Modules presentata precedentemente. In questo modo si tiene conto, in parallelo rispetto all'architettura del sistema, delle informazioni di accomodamento fornite dalla Stazione Spaziale in cui si intende operare. Relativamente al System, sono stati subordinati i vari sottosistemi, completando l'architettura prevista per il Template Model. Utilizzando la funzionalità Grapher, è possibile produrre automaticamente il diagramma a blocchi per l'architettura composta in una Option, riportato in figura 3.9.

Mentre le definizioni degli elementi sottosistema sono state attribuite ai vari domini competenti, l'utilizzo degli elementi nel Product Tree è stato associato al System Engineering. Di conseguenza l'Owner della Element Definition risulterà diverso dall'Owner dell'Element Usage. Questo comporta che un progettista appartenente al dominio specifico di un certo sottosistema ne potrà modificare la definizione, ma dovrà essere sufficientemente autorizzato per modificare l'utilizzo dell'elemento nel Product Tree o sovrascrivere dei parametri nell'Element Usage con valori diversi rispetto all'Element Definition. Infatti, mentre la modifica della Element Definition si ripercuote sull'Element Usage, finché questo non viene sovrascritto, il Parameter Override di un Element Usage non risale a monte sulla Element Definition.

3.1.4 Requirements

Per quanto riguarda la gestione dei Requirements, per lasciare sufficiente libertà agli utilizzi futuri del template, sono state definite le Requirement Specification, riportate in figura 3.10.

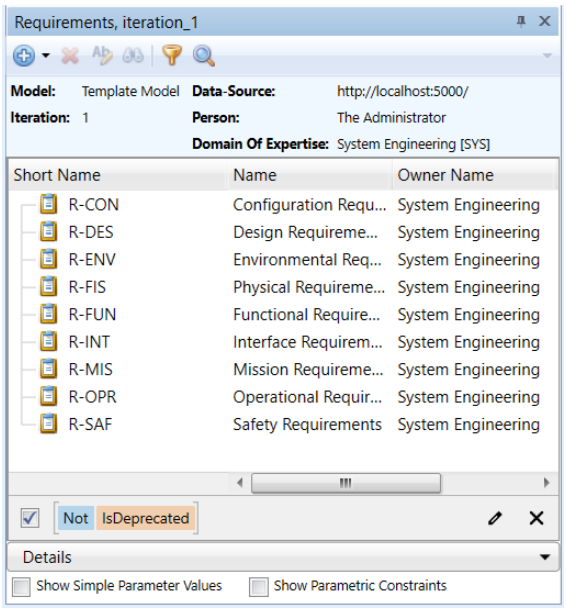

Figura 3.10. Requirement Specification.

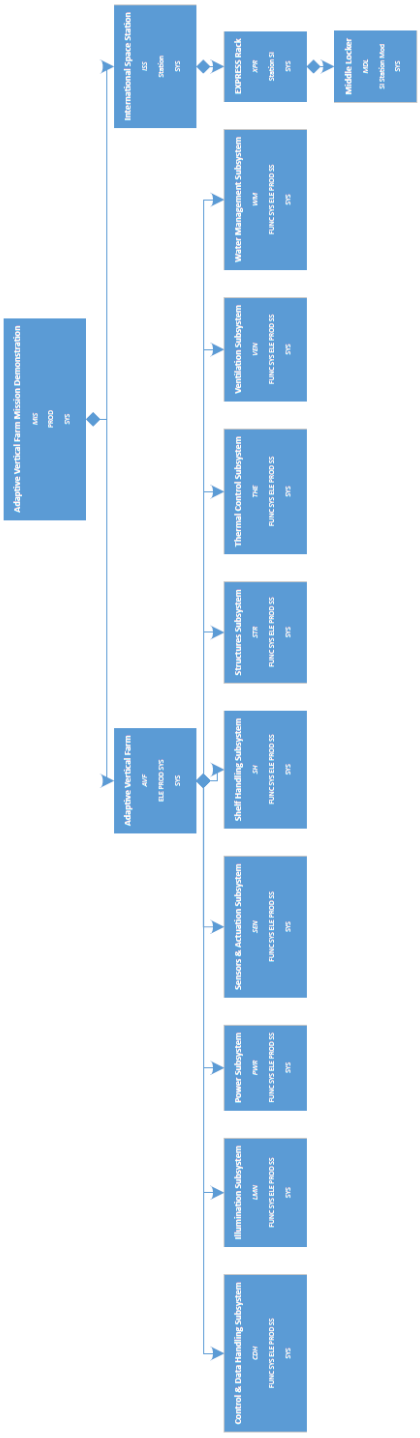

Figura 3.9. Diagramma a blocchi del Template Model prodotto dal Grapher.

Sarà possibile creare i requisiti singolarmente, nei futuri utilizzi del template, sfruttando queste Specification già individuate, oppure importarli direttamente da altri software di gestione dei requisiti.

3.2 Studio di Spazializzazione dell'Adaptive Vertical Farm

Al fine di approfondire la documentazione e le informazioni di riferimento per Space V, e di esplorare e dimostrare ulteriormente le funzionalità del software, è stato realizzato uno Study Model a partire dalla Tesi di Laurea Magistrale dell'Ing. B. Simonassi[27], attualmente System Engineer dell'azienda. Nella Tesi sopra citata viene svolto lo studio di spazializzazione del prototipo terrestre di Space V. Il prototipo ha dimostrato i benefici dell'applicazione dei principi di adattività ai volumi di crescita [28], innanzitutto in ambiente terrestre. Di conseguenza, è stata necessaria la realizzazione di uno studio di spazializzazione, per adeguare il funzionamento del sistema a condizioni operative significativamente differenti, relative al contesto spaziale.

3.2.1 Study Model

La creazione di uno Study Model che possa rappresentare questo studio di spazializzazione, è avvenuta sfruttando il Template Model precedentemente descritto. Così facendo, è stata assegnata automaticamente la Generic Space V Reference Data Library come Site RDL e sono state riprodotte tutte le caratteristiche descritte nella sezione precedente. Come primo passo, sono stati inseriti i requisiti di progetto definiti nello studio di spazializzazione, utilizzando i parametri e le categorie presenti della RDL e le Requirement Specifications inserite nel Template Model. Ad ogni Requirement è stato assegnato un Nome, uno Short Name, una descrizione e uno o più valori per il parametro Requirement Verification Method, riproducendo la loro formulazione nello studio di spazializzazione. Ulteriormente, dopo un'analisi approfondita di questi ultimi, è stata attribuita una delle categorie, tra High Level, Design Level e Detail Level, ed è stato assegnato un valore per il parametro Requirement Product Level.
Successivamente, utilizzando i *placeholder* degli elementi funzionali e degli Equipment, è stato ricreato lo schema a blocchi dello studio di spazializzazione, ottenendo così una descrizione più completa , e con maggiore livello di dettaglio, dell'architettura del sistema. Una volta copiati i *placeholder* e modificati adeguatamente, attribuendo Name, Short Name e Owner, sono stati utilizzati gli Element per comporre ulteriormente l'architettura dei sottosistemi, e successivamente sono stati inseriti gli Equipment relativi agli Element. Agli Equipment di ogni sottosistema è stato attribuito l'Owner del relativo dominio di competenza, tranne per quelli che rappresentano elementi di connessione o distribuzione, come cavi, tubi o elementi di fissaggio. Per questi elementi, è presente l'apposito dominio Harness and Connections (HC), i cui partecipanti valuteranno le problematiche prodotte dall'adattività sugli elementi di connessione. Già da queste fasi iniziali infatti, la stima della massa di questi elementi si baserà sulla valutazione della massa del sottosistema a cui appartengono. Gli Equipment infine, sono stati caratterizzati con valori di riferimento ottenuti da stime o relativi a prodotti simili a quelli considerati, in modo da permettere delle valutazioni preliminari del progetto e di disporre di dati da manipolare all'interno

del modello ingegneristico. Per gli Equipment, sono stati quindi inseriti i valori di massa, numero di oggetti e margine di massa, al fine di valutare il budget di massa del sistema e dei sottosistemi e inoltre i valori di margine di potenza, potenza nominale consumata e potenza di picco consumata, per una valutazione preliminare del budget di potenza dell'architettura proposta. I valori sono stati inseriti nel campo *Reference* del rispettivo parametro, di cui è stato modificato conseguentemente anche il Value Switch, così da impostare il valore inserito come Actual Value e generando conseguentemente le relative Publications dei nuovi valori. In seuito le Publications sono state accettate, conferendo il valore adeguato anche al campo *Published* del rispettivo parametro.

3.2.2 Relazioni e tracciamento dei requisiti

I requisiti di progetto, definiti nello studio di spazializzazione, sono stati integrati nel modello ingegneristico al fine di implementare delle relazioni di decomposizione tra i livelli individuati dalle categorie. Come già presentato, oltre alle informazioni fondamentali, ai requisiti è stato assegnato il Requirement Verification Method, a seguire, i Requirements sono stati riorganizzati ed esportati in un foglio Excel, tramite l'add-in di Excel. Il foglio è stato un ulteriore strumento di visualizzazione dei requisiti di progetto, utile nella fase di rassegna per l'attribuzione della categoria.
In seguito, è stato utilizzato il Relationships Editor, per generare graficamente le relazioni di decomposizione dei requisiti. Trascinando i requisiti, a partire da quelli di livello più alto, questi possono essere importati nell'editor, dove vengono raffigurati come riquadri. Tramite la sezione apposita nella finestra del Relationships Editor, è possibile selezionare la Rule sulla base della quale si intende generare la relazione. Dalla decomposizione dei requisiti, ne risulta che alcuni requisiti di alto livello necessitano di ulteriori sviluppi progettuali per essere maggiormente approfonditi e destrutturati, mentre tutti i requisiti categorizzati come Design Level e Detail Level sono tracciati nello schema di decomposizione prodotto. Questa analisi di decomposizione dei requisiti è valutabile tramite le apposite Relationships Matrix. Nello specifico, sono state create, e salvate, tre configurazioni di matrici, in modo da poter facilmente accedere alle due matrici che visualizzano la decomposizione da High Level, riportata in figura 3.11 a Design Level e da Design Level a Detail Level e una terza matrice che riporta il tracciamento globale dei requisiti. Le prime due matrici presentano sulle righe i requirements *padre*, del livello superiore, e nelle colonne i requirements *figli*, del livello inferiore. La terza matrice invece è quadrata e riporta tutti i Requirements sia sulle righe che sulle colonne, producendo una matrice di dimensioni notevoli, ma in cui sono evidenziati i requisiti non tracciati nella decomposizione.

In aggiunta all'utilizzo delle matrici, per visualizzare la decomposizione e il tracciamento dei requisiti, è stato realizzato anche uno script, che sfrutta la stessa struttura di categorie. Lo script acquisisce tutti i Requirements formulati all'interno del modello, valutandone la categoria per individuare i requisiti di alto livello, categorizzati appunto come High Level. Dopo aver individuato i requisiti, vengono ispezionate le relazioni a loro attribuite. Tra le relazioni, si ricercano quelle categorizzate come Requirements Decomposition. Le relazioni di questa categoria hanno come Source il Requirement in

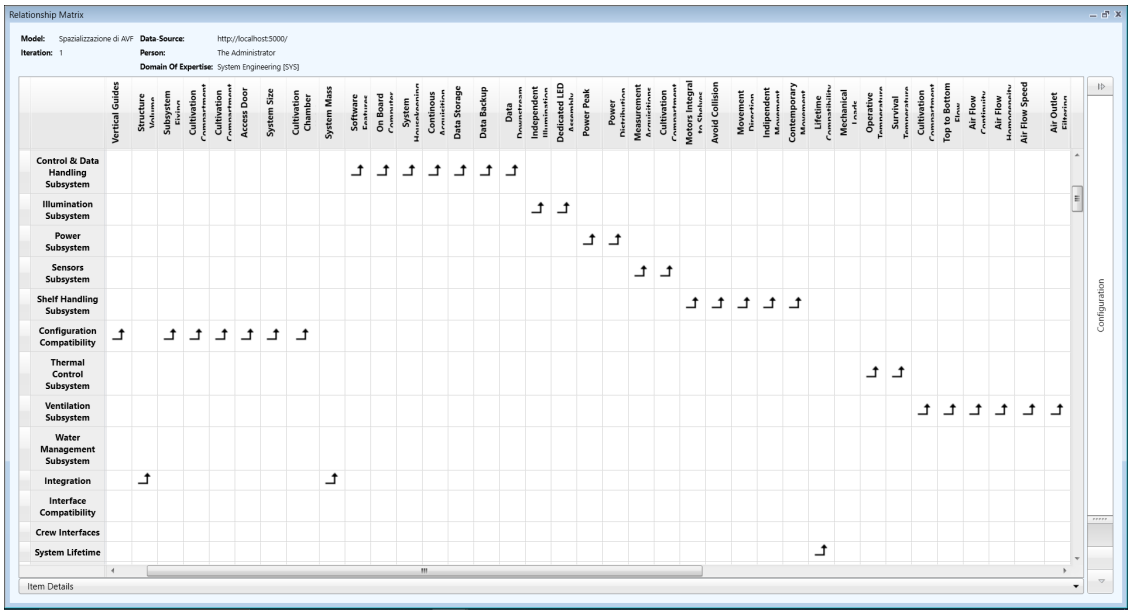

Figura 3.11. Estratto della matrice di decomposizione dei requisiti di alto livello prodotta da Comet.

esame e come Target gli ulteriori requisiti in cui il requisito High Level si decompone. Realizzando questo processo per ogni High Level Requirement e reiterando la procedura su ogni Design Level Requirement, per ricavare i Detail Level Requirements associati, si ottengono delle strutture ad albero in cui ogni High Level Requirement, unica *radice* dell'albero, si decompone prima in Design Level e poi in Detail Level. Dopo aver generato le strutture ad albero, lo script produce un file html, visualizzabile tramite un qualunque browser, in cui vengono presentate le strutture appena generate, con la possibilità di espandere i singoli requisiti, visualizzando la descrizione associata. Dunque, nella sezione Scripting, una volta aperto il file opportuno, è possibile utilizzare la funzione *GenerateRequirementsDecompositionTrees(engineeringModelShortName, iterationNumber=None, output_dir=None)* per produrre il file html. Questa funzione richiede un argomento per essere eseguita e due argomenti opzionali, ovvero: il nome del modello ingegneristico che si intende analizzare e il numero dell'iterazione desiderata e un indirizzo in cui generare il file html di output. Nella figura 3.12 è riportato uno degli alberi di decomposizione generato.

3.2.3 Scripting

Come descritto precedentemente, lo scripting è un potente metodo di automazione delle funzionalità e di verifica, elaborazione e valutazione delle informazioni contenute nel modello ingegneristico. La fonte principale di documentazione sull'implementazione degli scripts, risulta essere la presenza dei tooltip descritti nel capitolo precedente, dunque è

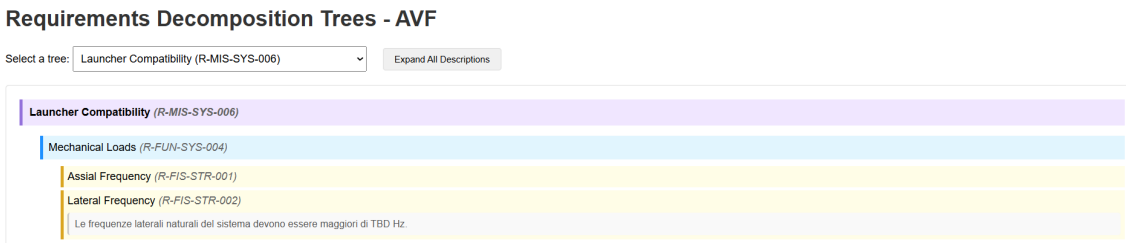

Figura 3.12. Visualizzazione dei Requirements prodotta tramite script.

stato creato un primo script in cui è definita la funzione *InspectObject(object)* che semplicemente elenca i metodi e gli attributi dell'oggetto fornito come argomento. Tramite questa funzione, e congiuntamente con l'analisi del Comet Software Development Kit (SDK)[29] e delle informazioni fornite direttamente dall'interfaccia di scripting, è stato possibile individuare gli approcci corretti per l'implementazione delle funzionalità ricercate. La funzione *InspectObject()* ed altri estratti di codice sono stati pubblicati nella repository Github Scripts-for-CDP4-Comet[30].
Quindi è stato formulato un secondo script, chiamato *ModelComputation*, in cui sono definite diverse funzioni, a disposizione per gli utenti, per compiere elaborazioni sul modello ingegneristico. In un secondo momento, è stato svolto un processo di ottimizzazione, definendo delle funzioni *helper*, unificando e strutturando i processi simili all'interno dello script. Così facendo, è stato possibile creare una serie di funzioni secondarie che possano essere applicate genericamente all'interno dei processi necessari allo svolgimento delle funzioni di più alto livello. Questo approccio aumenta la manutenibilità del codice e la possibilità di implementare nuove funzioni, migliorando la gestione di script complessi.

Calcolo della massa del sistema e dei sottosistemi

La prima funzione implementata consente di automatizzare il calcolo della massa complessiva del sistema e dei sottosistemi. La funzione *CalculateSystemMass(engineeringModelShortName)* richiede un argomento, ovvero una stringa contenente lo Short Name del modello ingegneristico che si intende analizzare. Opzionalmente, possono essere inseriti una serie di argomenti per richiedere alla funzione informazioni specifiche oppure particolari operazioni. Gli argomenti opzionali sono:

- iterationNumber, un intero che indichi l'iterazione desiderata. Se non viene inserito nessun argomento sarà considerata l'Iteration attiva;
- optionShortName, una stringa con lo Short Name dell'Option da considerare. Se non viene inserito nessun argomento sarà considerata l'opzione di default;
- updateValues, un valore booleano per richiedere o meno la scrittura dei risultati dei rispettivi parametri di massa del sistema e dei sottosistemi. Inserendo *False*, o nessun argomento, lo script si limiterà a calcolare e fornire in output le masse degli

elementi presenti. Con *True* invece, i valori calcolati saranno inseriti nel campo *Computed* dei rispettivi parametri di massa.

- actualFiniteStateListShortName e actualFiniteStateShortName, due stringhe contenenti lo Short Name dell'Actual Finite State List e dell'Actual Finite State, per il quale si vogliono considerare i parametri. Nel caso in cui i parametri del modello richiesti nelle elaborazioni dipendano da Finite State, questi argomenti devono necessariamente essere inseriti quando si chiama questa funzione. In tal caso infatti, l'elaborazione deve essere svolta considerando tutti i parametri a parità di Finite State.

Lo script, partendo dall'Iteration del modello indicata, acquisisce l'elemento, o eventualmente gli elementi, categorizzati come System. In seguito, utilizzando l'opzione fornita per individuare l'architettura del sistema, si acquisiscono gli elementi annidati nel sistema e categorizzati come sottosistema. In questo modo, verranno considerati solamente i sottosistemi definiti nel modello ed effettivamente utilizzati nel Product Tree. Ogni sottosistema viene immagazzinato in un dizionario, utilizzando il suo ID unico come chiave, per mantenere l'oggetto a disposizione per successive elaborazioni dello script. In maniera analoga a quanto svolto per il sistema, vengono acquisiti gli elementi annidati che compongono l'architettura del sottosistema, nella specifica Option indicata dall'utente. Dagli elementi annidati viene acquisita l'Element Definition associata, e vengono selezionati gli elementi categorizzati come Equipment. Quindi, è possibile valutare la somma delle masse, considerando la massa di un singolo elemento, il margine di massa e il numero di elementi e inizializzando dei valori di default per gestire l'evenienza di parametri nulli. Dopo aver calcolato le masse dei sottosistemi, la funzione calcola la massa del sistema sommando queste ultime e applicando un ulteriore margine di massa, determinato dal parametro System Mass Margin (sys_mass_margin), parametro attribuito al sistema. Infine, vengono visualizzati i risultati del calcolo ed eventualmente i valori vengono salvati nei parametri del modello, se richiesto dall'utente e se differenti dai valori già in memoria nel server. La funzione restituisce, in output, la massa del sistema prima e dopo l'applicazione del margine di massa globale.

Calcolo della massa degli elementi di connessione

La seconda funzione *CalculateHarnessAndConnections(engineeringModelShortName)* utilizza gli stessi argomenti descritti per la funzione precedente. Lo scopo di questa funzione è individuare gli Equipment, nell'architettura dei sottosistemi, che rappresentano elementi di connessione, come cavi, tubi o organi di giunzione. Per questa tipologia di elementi, infatti, la massa viene stimata come una percentuale della massa totale del sottosistema a cui appartengono. Innanzitutto, è stato definito il parametro *harness and connection margin* ed è stato attribuito a questi particolari elementi. Questo parametro riporta il valore percentuale da applicare alla massa del sottosistema, per calcolare la massa dell'elemento stesso. Poiché questi elementi sono stati attribuiti al Domain di Harness and Connections, questa ipotesi del modello viene utilizzata come discriminante per distinguere gli elementi la cui massa va calcolata a partire dal sottosistema di appartenenza, tuttavia, al fine di individuare gli elementi, potrebbe essere utilizzata anche una categoria

ad hoc. Anche in questo caso, il salvataggio dei valori avviene solo se necessario e previa richiesta dell'utente. Questa funzione restituisce un dizionario nel quale si associano gli Id univoci dei sottosistemi, come chiave, e le liste di elementi del dominio HC individuati per quel sottosistema, come valore.

Calcolo della potenza richiesta dal sistema e dai sottosistemi

È stata definita una funzione per la valutazione preliminare della potenza nominale e di picco richiesta dal sistema. Si è ipotizzato di considerare una configurazione in cui tutti gli elementi, che richiedono alimentazione elettrica, sono accesi contemporaneamente. Si vogliono elaborare le somme delle due potenze, di picco e nominale, dei singoli Equipment appartenenti ad ogni sottosistema, e valutare il consumo globale del sistema. Nello script, quindi, è definita la funzione *CalculateSystemPower(engineeringModelShortName)*, che richiede ancora gli stessi argomenti descritti precedentemente. L'implementazione, infatti, risulta analoga: si individua l'elemento categorizzato come System e si acquisisce la sua architettura descritta nell'opzione fornita, ottenendo i Nested Elements. Da questi ultimi, si risale all'Element Definition associata, per individuare, prima gli elementi categorizzati come Subsystem e, reiterando il processo, gli Equipment. Vengono dunque considerati gli Equipment che compongono ogni sottosistema, acquisendone potenza di picco, potenza nominale, margine di potenza e numero di oggetti, poiché tutti gli elementi che richiedono alimentazione sono considerati accesi in questa analisi. I valori appena ottenuti vengono elaborati per individuare il consumo nominale e di picco dei singoli sottosistemi. Infine si calcolano i consumi dell'intero sistema, applicando un ulteriore margine di potenza complessivo. Analogamente alle funzioni precedenti, i risultati delle elaborazioni vengono presentati nella sezione di Output, ed eventualmente salvati nel modello. La funzione restituisce i quattro valori delle potenze calcolate per il sistema, prima e dopo l'applicazione del margine globale.

Analisi dei costi

È stata realizzata una funzione per implementare la stima preliminare dei costi di progettazione del sistema. Questa analisi dei costi, contenuta nello studio di spazializzazione, consiste nella suddivisione dello sviluppo del progetto in quattro fasi: Studi Avanzati, Formulazione, Progettazione e Implementazione. Per ogni fase, vengono considerate cinque voci di costo: la durata in mesi, il costo della modellazione, il costo delle verifiche, il costo dei test ed il costo dei prototipi. Una volta stimati dei valori, per ogni fase, di durata e di spesa percentuale delle altre voci di costo, si può calcolare l'impatto di ognuna delle fasi considerate, sul costo complessivo di sviluppo. Questo valore si ottiene elaborando i vari "pesi" delle cinque voci di costo in ogni fase. Il calcolo del peso di una voce di costo avviene tramite delle costanti, anch'esse percentuali, che rappresentano, invece, l'impatto di una singola voce di costo sul costo complessivo della fase. Si calcola quindi la percentuale della durata di una singola fase, rispetto alla durata totale della progettazione, e si moltiplica per la costante associata alla voce di costo, ottenendo il peso della durata. Analogamente per le restanti voci di costo, già formulate in percentuale, si moltiplica il

valore per la costante associata, determinando il peso della voce di costo in una fase. Infine, sommando tutti i valori ottenuti a parità di fase di progetto considerata, si ottiene il costo percentuale di ogni fase. Per implementare questa analisi, sono stati definiti innanzitutto dei parametri che rappresentano le voci di costo. I parametri sono: *design phase duration*, specializzazione del parametro *duration*, misurato in mesi, e *cost of models*, *cost of verifications*, *cost of tests*, *cost of prototypes* e *cost of product development phase* come specializzazioni del parametro *fraction*, misura percentuale. In seguito è stato creato il Parameter Group *Voices of Costs* all'interno della Element Definition categorizzata come Product, nel caso specifico la missione, e sono stati inseriti in questo gruppo i cinque parametri delle voci di costo precedentemente introdotti, mentre il parametro di riepilogo *cost of product development phase* è stato attribuito al Product ma fuori dal gruppo, così da risultare visivamente separato. Dunque, è stata definita l'Actual Finite State List "Product Development Phases" che contiene le quattro fasi del progetto precedentemente considerate, ed è stata applicata la State Dependence sui parametri appena descritti. In questo modo, è stato possibile inserire nel campo "Reference" del rispettivo parametro, il valore opportuno per ogni voce di costo ed ogni fase di progetto. Successivamente è stata realizzata la funzione *ProductCostsAnalysis(engineeringModelShortName)* che richiede come unico argomento necessario lo Short Name del modello ingegneristico di riferimento. Possono essere inseriti ulteriormente gli argomenti iterationNumber, optionShortName e updateValues con le stesse finalità delle funzioni precedenti e inoltre il parametro booleano showVisualization, per richiedere la visualizzazione dei risultati in un grafico. Il software recupera le informazioni degli elementi categorizzati come prodotto, all'interno del modello indicato valutando l'iterazione attiva o quella fornita dall'utente e analogamente l'Option. Nella definizione della funzione, è presente lo Short Name dell'Actual Finite State List definita per questa analisi e un dizionario che associa gli Short Name delle voci di costo alle costanti necessarie per il calcolo. Viene quindi svolta l'elaborazione precedentemente descritta per ogni fase di progetto calcolando il costo percentuale di ogni fase, che viene visualizzato testualmente in Output. Il parametro opzionale showVisualization determina se verrà chiamata un'ulteriore funzione per la visualizzazione dei dati appena prodotti. In tal caso, utilizzando le librerie .Net, viene generato un grafico a torta riportante il costo percentuale di ogni fase di progetto. La funzione restituisce un dizionario annidato, che utilizza come chiavi i nomi degli elementi categorizzati come Product, e in cui viene immagazzinata tutta la struttura dell'analisi dei costi appena svolta. Per ogni prodotto infatti, saranno riportati tutti i valori delle voci di costo associate al prodotto nel modello ingegneristico, i rispettivi pesi calcolati, il peso totale di una singola fase e un indice di correttezza della valutazione. Infatti si valuta che la somma delle percentuali individuate totalizzi il 100% dei costi, riportando il risultato della verifica all'utente.

Diagramma N^2

Infine sono state utilizzate le funzionalità dello scripting per automatizzare la generazione di diagrammi N^2. Il diagramma N^2 è una rappresentazione grafica delle relazioni tra i sottosistemi di un sistema. Si è proceduto integrando nel modello la schematizzazione formulata nello studio di spazializzazione. Sono state create quattro categorie applicabili

alle Relationships, corrispondenti alle quattro tipologie di interfacce previste dallo schema, ovvero:

- Mechanical and Physical Interfaces
- Functional Interfaces
- Electrical and Data Interfaces
- Thermal Interfaces

Queste categorie sono state impiegate per definire le relative Binary Relationships Rules, formulate per applicarsi agli elementi della categoria Subsystem, sia in qualità di Source che di Target della Relationship. Utilizzando queste quattro Rules nel Relationships Editor, sono state create le relazioni che rappresentano le interfacce del diagramma N^2. La visualizzazione di queste relazioni, tuttavia, risultava di difficile consultazione poiché la maniera migliore per visualizzare questo diagramma consisteva nell'utilizzare le Relationships Matrix, riportando i sottosistemi sia sulle righe che sulle colonne di una matrice che dunque risulta simmetrica. Ulteriormente, avendo definito quattro categorie differenti per le Relationships, si sarebbero dovute visualizzare le differenti interfacce in quattro matrici separate. Al contrario, definendo una sola categoria, si sarebbe persa l'informazione sulla tipologia di interfaccia visualizzata. Dunque è stata realizzata una funzione per generare delle visualizzazioni del diagramma N^2 associato all'architettura del sistema in esame. La funzione *GenerateN2Chart(engineeringModelShortName)* richiede come argomento lo Short Name del modello ingegneristico, specificando, facoltativamente, il numero dell'iterazione e l'opzione da considerare. La funzione parte dall'architettura dell'opzione fornita per individuare i sottosistemi che compongono il sistema. Successivamente si recupera l'Element Definition dei singoli sottosistemi e si acquisiscono le relazioni applicate a questi elementi e appartenenti alle quattro categorie di relazioni di interfaccia. Si costruiscono dunque quattro matrici, una per ogni tipologia di interfaccia, che contengono valori booleani nelle celle, True se esiste una relazione di quel tipo che rappresenti l'interfaccia. Le matrici vengono popolate solo nella parte triangolare superiore, così da posizionare i simboli nella rispettiva cella per segnalare la presenza dell'interfaccia. Vengono quindi utilizzate le stesse librerie .Net, sfruttate anche nella funzione precedente, per generare dinamicamente il diagramma N^2 a partire dalle informazioni recuperate dal modello ingegneristico. Una versione puramente esemplificativa del diagramma è riportata in figura 3.13. Questa funzione non restituisce nessun valore in output.

Funzioni *helper*

Si presentano alcune funzioni *helper* frequentemente utilizzate nell'implementazione delle funzionalità appena descritte. Queste funzioni sono state estrapolate dalle prime versioni dello script per ottimizzarlo sotto diversi aspetti. In particolare, la loro creazione consente di ridurre la ripetizione del codice, diminuendo il rischio di errori, migliorando la manutenibilità e facilitando l'aggiunta di nuove funzionalità. Raggruppando i processi ricorrenti in porzioni di codice unificate, si ottiene un'implementazione più modulare ed efficiente.

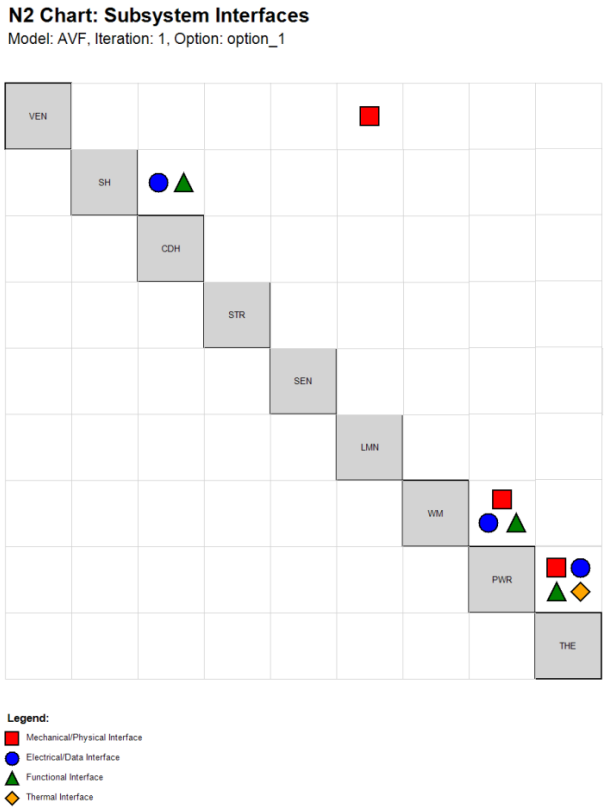

Figura 3.13. Esempio di diagramma N^2 generato.

Un esempio di funzione di questo tipo è *member_of_category(object, categoryShortName)* che verifica che l'oggetto fornito in argomento possieda la categoria specificata tramite Short Name, restituendo un booleano che indica l'appartenenza.
Un aspetto fondamentale di molte delle funzioni descritte è la capacità di gestire, quando richiesto, la transazione dei nuovi valori dei parametri verso il server. Per garantire la persistenza dei dati all'interno del modello ingegneristico, è infatti necessario configurare correttamente la transazione e richiedere l'esecuzione della scrittura finale nella sessione attiva dell'utente. Questa procedura è implementata nella funzione *update_element_parameter(iteration, element_definition, param_suffix, option_index, computed_value, actualFiniteStateListShortName=None, actualStateShortName=None)*, insieme a una serie di controlli per la corretta scrittura dei dati. In questo modo, la modifica verrà effettivamente recepita dal server, e il nuovo valore sarà utilizzabile all'interno delle funzionalità del software, come ad esempio le Publications, e condiviso opportunamente con gli altri utenti. Un ultimo esempio di funzione *helper* è la funzione *get_value_index(parameter, option_index, iteration=None, actualFiniteStateListShortName=None, actualStateShortName=None)* utilizzata molto frequentemente nell'ambito dello script *ModelComputation*. Questa funzione ha il compito di individuare il corretto

indice per l'acquisizione o la scrittura dei valori dei parametri. I parametri contengono infatti una lista di Value Set, ciascuno dei quali rappresenta un insieme di valori (Computed, Manual, Published, ecc.). Ogni parametro possiede un Value Set per ogni funzionalità che lo richiede, ovvero, possono assumere valori diversi nell'ambito delle Option, dei Finite States o di combinazioni di questi. La lista di Value Set, inoltre, non presenta un'indicizzazione rigorosa e affidabile per individuare il valore corretto, quindi, è possibile valutare gli attributi ActualOption e ActualState per ottenere l'opzione e l'Actual Finite State associati al Value Set considerato. In aggiunta, è possibile valutare l'attributo booleano *Parameter.IsOptionDependent*, per individuare preliminarmente la dipendenza del Value Set da opzioni, e consultare *Parameter.StateDependence*, che restituisce l'Actual Finite State List del Parameter, se presente. Fornendo in input lo Short Name di una Actual Finite State List e di un Actual Finite State, la funzione restituisce l'indice, o l'elenco di indici, associato al parametro o alla lista di parametri forniti come argomento. Questa funzione non è destinata a rispondere a esigenze di alto livello degli utenti e, pertanto, difficilmente verrà invocata manualmente. Piuttosto, essa fornisce un accesso sistematico ai dati del modello. Inoltre, per segnalare eventuali errori nel recupero dell'indice del Value Set, la funzione restituisce un valore negativo anziché un indice maggiore o uguale a zero. Le funzioni che si avvalgono di questo *helper* verificano l'assenza di valori negativi prima di proseguire: in caso contrario, l'errore viene propagato lungo la catena di funzioni, interrompendo l'elaborazione richiesta. Questo meccanismo previene la generazione di artefatti e l'esecuzione di elaborazioni errate o prive di significato qualora l'individuazione del valore corretto del parametro fallisca.

3.2.4 Reporting

Le funzionalità del software dedicate alla generazione dei report possono essere utilizzate per monitorare lo stato di avanzamento del progetto e sintetizzare alcune caratteristiche e prestazioni del sistema. A tal fine, sono stati sviluppati due report che forniscono una panoramica del Mass Budget e del Power Budget, sia a livello di sistema che di sottosistemi. La realizzazione di questi report è avvenuta attraverso un'analisi congiunta dei CDP4-Report[31] e CDP4-Model[32], disponibili sul repository GitHub di Starion. Questo studio ha permesso di comprendere i principi di funzionamento del software e di elaborare una strategia per sfruttare i metodi preposti all'acquisizione e all'analisi delle informazioni del modello ingegneristico.
L'implementazione dei Report risulta simile a quella degli script e del tutto analoga tra Mass Budget e Power Budget. Nel Datasource sono state definite le classi necessarie all'esportazione dei dati verso il Report. In primo luogo, è stata realizzata la classe Variables che contiene, come attributi, i Nested Elements. In seguito, è stata creata una classe estendendo *OptionDependentDataCollector* ed è stato definito il metodo *CreateDataObject()* che restituisce una matrice in cui le righe contengono i parametri relativi ai Nested Element, ottenuti tramite la classe Variables precedentemente formulata. In seguito, si definisce la struttura di categorie per indicare la decomposizione gerarchica desiderata per l'architettura del Top Element del Product Tree, producendo conseguente la matrice con tutti i parametri degli elementi considerati. A questo punto si acquisisce separatamente il margine globale, di massa o potenza, attribuito all'elemento categorizzato come

System, e si aggiunge come colonna alla matrice, riportandone il valore. Ulteriormente si aggiungono delle colonne in cui saranno immagazzinati i dati delle elaborazioni. A questo punto, bisogna creare una classe, estendendo *DataCollectorRow*, per l'acquisizione dei valori dei parametri di interesse, nelle righe della matrice precedentemente generata. Si utilizzano dei metodi built-in per imporre i valori dei parametri da visualizzare nel Report, dunque:

- massa, numero di oggetti, margine di massa e margine di massa di sistema per il Mass Budget;
- potenza consumata nominale, potenza consumata di picco, numero di oggetti, margine di potenza e margine di potenza di sistema per il Power Budget.

Infine, bisogna definire una classe che estenda *ReportingParameters* per permettere all'utente che genera il report di inserire, o richiedere, informazioni specifiche. In questa classe, è stato definito il parametro *OptionName*, in modo tale che l'utente possa selezionare un'Option per svolgere le valutazioni dei due Budget. Il file Report.repx riporta semplicemente i dati forniti dal Datasource, e in particolare, vengono riportate le sezioni da rappresentare nel documento che sarà generato. La visualizzazione del Report.repx tramite l'editor integrato in Comet è riportata in figura 3.14.

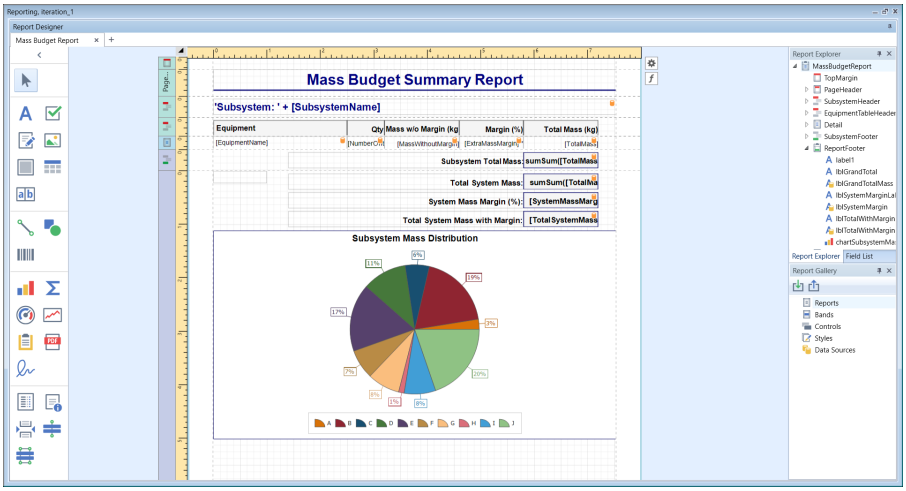

Figura 3.14. Struttura del Mass Budget nel Report Designer.

Il Report rappresenta i dati ottenuti suddividendoli in gruppi in base al *SubsystemName* associato. Dunque, dopo aver presentato il nome di un sottosistema nell'intestazione, segue una sezione di dettaglio in cui vengono riportati i valori dei singoli Equipment appartenenti al sottosistema, mostrando tutti i valori individuati nel Datasource. Alla sezione di dettaglio segue un piè di pagina relativo al gruppo, che quindi presenta un riepilogo della grandezza in esame, specifico per il sottosistema considerato. Questa sezione di Report appena descritta viene ripetuta per ogni sottosistema. Infine è presente un piè di pagina complessivo, in cui vengono rappresentate le grandezze complessive del

sistema, con e senza margine globale. I Report, per come sono stati formulati, calcolano al momento i valori di riepilogo per i sottosistemi e per il sistema, invece che acquisire eventuali valori già attribuiti al modello. In questo modo, risultano essere un'ulteriore verifica, indipendente, della correttezza dei dati elaborati. Nella seguente figura 3.15 è riportato un estratto del Mass Budget, dove, per non divulgare informazioni sensibili della startup Space V, i dati sono stati oscurati tramite le funzionalità del software. Questo documento, infatti, corrisponde a quello che otterrebbe un partecipante appartenente a un'organizzazione diversa da quella proprietaria e privo di autorizzazioni per la visualizzazione di qualsiasi elemento.

Mass Budget Summary Report

Subsystem: Hidden Element Usage

Equipment	Qty	Mass w/o Margin (kg)	Margin (%)	Total Mass (kg)
Hidden Element Usage	1	0,00	0,00	0,00
Hidden Element Usage	1	0,00	0,00	0,00
Hidden Element Usage	1	0,00	0,00	0,00
Hidden Element Usage	1	0,00	0,00	0,00
Hidden Element Usage	1	0,00	0,00	0,00
Hidden Element Usage	1	0,00	0,00	0,00
Hidden Element Usage	1	0,00	0,00	0,00
Hidden Element Usage	1	0,00	0,00	0,00
Hidden Element Usage	1	0,00	0,00	0,00
Hidden Element Usage	1	0,00	0,00	0,00
Hidden Element Usage	1	0,00	0,00	0,00
			Subsystem Total Mass:	0,00
			Total System Mass:	0,00
			System Mass Margin (%):	0,00
			Total System Mass with Margin:	0,00

Figura 3.15. Anteprima del Mass Budget Summary Report con dati oscurati dal software.

Capitolo 4

Generalizzazione del caso di studio di Space V per l'integrazione del CE nel processo di progettazione delle Piccole e Medie Imprese

A seguito della descrizione dell'implementazione delle funzionalità di Comet nei processi di progettazione di Space V, si vogliono generalizzare gli approcci e le procedure precedentemente riportate, per elaborare considerazioni e indicazioni, per PMI del settore spaziale che intendono integrare i principi del CE tramite Comet, o altri software di scambio di dati basati su modelli ingegneristici. La divulgazione di questi principi incentiva l'adozione di metodologie strutturate di lavoro, migliorando la qualità della cooperazione tra i vari attori del settore. Infatti, l'efficacia dell'applicazione del CE è direttamente proporzionale alla diffusione di tali strumenti informatici, che facilitano l'interoperabilità e il coordinamento tra le diverse organizzazioni coinvolte nella progettazione di sistemi spaziali.

Sebbene l'utilizzo di strumenti digitali per supportare le metodologie del CE sia consolidato nell'ambito delle grandi Agenzie Spaziali, l'utilizzo di questi strumenti nel contesto delle PMI risulta essere piuttosto innovativo.
È possibile osservare [33] la similitudine tra le grandi aziende spaziali ed i *cluster* di PMI nella progettazione di sistemi spaziali. Con *cluster*, si intende un insieme di PMI, accomunate da alcune caratteristiche, come la localizzazione geografica e/o l'ambito ingegneristico, che collaborano congiuntamente per la realizzazione di un progetto, condividendo conoscenze e competenze relative alle differenti discipline ingegneristiche coinvolte nella progettazione. All'interno dei *cluster*, ogni PMI contribuisce fornendo un tassello fondamentale alla realizzazione del progetto, partecipando a una o più fasi, dall'ideazione alla validazione.

In questa analogia, i team di progettazione delle grandi aziende o agenzie spaziali, gruppi di esperti nei vari ambiti ingegneristici, sono sostituiti dalle PMI che si affiancano cooperando e condividendo, non solo informazioni relative al progetto, ma anche approcci e metodologie derivanti da esperienze, ambiti operativi e amministrazioni differenti.
Di conseguenza, gli strumenti e le attuazioni pratiche, che tipicamente vengono adottati per implementare il CE nell'ambito di grandi organizzazioni, vanno rivalutati e ridimensionati al contesto delle PMI. Ad esempio, la realizzazione di *facilities*, come la CDF dell'ESA, che possano ospitare fisicamente le attività di progettazione, vengono totalmente soppiantate dall'utilizzo di strumenti informatici performanti e specializzati nella gestione delle sessioni simultanee di Concurrent Design, tramite un'amministrazione granulare delle partecipazioni.
Mentre negli ultimi anni l'utilizzo di strumenti, come OCDT e Atlas, era quasi esclusivo delle Agenzie Spaziali Internazionali, che li avevano creati a partire dalle proprie esigenze, lo sviluppo recente di software dedicati ad implementare il CE in maniera strutturata, soprattutto a seguito della pubblicazione dell'ECSS-E-TM-10-25A, consente l'adozione di questi principi anche nel contesto delle PMI, erogando una serie di funzionalità improntate alla gestione di contesti di cooperazione più complessi. Caratteristiche come l'importazione ed esportazione di varie tipologie di informazioni associate ai modelli o la possibilità di accedere a database unici e consistenti, risultano essere di importanza strategica nella gestione della partecipazione, anche da remoto , ai processi di sviluppo di sistemi altamente complessi, come quelli spaziali, che necessitano lo svolgimento parallelo di numerose valutazioni, implicando un vasto spettro multidisciplinare di conoscenze.

4.1 Analisi del contesto operativo

Si intende svolgere un'analisi del contesto operativo delle PMI del settore spaziale, individuandone le peculiarità e le principali differenze con i processi di una grande azienda o di un'Agenzia Spaziale. Questa analisi è utile ad individuare e descrivere gli aspetti più rilevanti dell'ambito operativo, soffermandosi in particolar modo sulle questioni legate alla collaborazione tra organizzazioni distinte. In questo ambiente particolarmente interconnesso, infatti, si vogliono valutare le esigenze degli enti tipicamente coinvolti e come l'adozione dei principi del Concurrent Engineering possa essere una soluzione a queste necessità.
Partendo dall'analogia considerata nell'introduzione di questo capitolo, le principali differenze tra i contesti operativi di grandi e piccole imprese derivano dalla sostituzione dei team di progettazione con le PMI, organizzazioni a sé stanti, modificando profondamente le relazioni che si instaurano tra le PMI appartenenti ai *cluster*.
Considerando un generico *cluster*, ci si aspetta che ci sia un gruppo ristretto, se non un'unica PMI, che, a valle di una formulazione preliminare del progetto di un sistema, abbia aggregato il *cluster* stesso, a seconda delle necessità individuate nel design preliminare. Il gruppo promotore del progetto spesso ne detiene la proprietà intellettuale, ad esempio, tramite brevetti di elementi fondamentali del sistema in studio. Tuttavia, all'interno di un *cluster*, possono essere presenti numerose PMI che intervengono, più o meno incisivamente, nelle attività di progettazione. Solitamente infatti, una parte delle

PMI partecipa al progetto fornendo, o adattando, un proprio prodotto al contesto operativo del sistema in esame. Questo tipo di interazione, tra cliente e fornitore, risulta essere molto comune per le PMI in ambito spaziale, in cui queste imprese provvedono ad uno specifico prodotto, o servizio, che sviluppano internamente, partecipando al gruppo di progettazione in maniera del tutto analoga a come agirebbero nella collaborazione con una grande organizzazione.
Nei processi di progettazione dei *cluster* di PMI tuttavia, alcune aziende possono essere incaricate di sviluppare interamente un sottosistema o di occuparsi dell'integrazione dei sottosistemi in un sistema complesso, partecipando intensamente nelle varie fasi di design. In questi caso, vengono a configurarsi delle partnerships tra le PMI più strettamente coinvolte, andando a definire a priori i termini della cooperazione in maniera più precisa e strutturata. Le partnerships avvengono secondo una formulazione contrattuale, che delinea chiaramente, e nello specifico, tutti gli aspetti relativi alla collaborazione. Nelle partnerships viene stabilito l'allocamento delle risorse al progetto da parte dei contraenti, definendo le varie responsabilità e mansioni a cui le singole PMI sono chiamate a rispondere. Inoltre, devono essere stabiliti gli obiettivi da raggiungere nell'ambito della cooperazione e la durata e le condizioni necessarie per un eventuale rinnovo o cessazione del rapporto. La definizione a priori di tutti gli aspetti della collaborazione è necessaria per chiarire, fin dal principio, i rapporti tra i contraenti, e conciliare differenti interessi e necessità delle entità coinvolte nella realizzazione del progetto.
Un aspetto cruciale nella definizione della partnership riguarda la gestione della proprietà intellettuale nell'ambito delle PMI[34]. La gestione di trademarks, brevetti o trade secrets richiede un accordo chiaro tra le parti, al fine di garantire che i diritti sui risultati della collaborazione siano equamente distribuiti e rispettino le necessità strategiche delle PMI coinvolte. Ad esempio, è fondamentale stabilire se la proprietà intellettuale sviluppata congiuntamente debba essere condivisa, assegnata a una specifica entità o licenziata secondo termini concordati.

In questo ambiente fortemente interconnesso, dove la progettazione di elementi fondamentali di un sistema viene spesso delegata completamente ad organizzazioni esterne, la formulazione di missioni spaziali è incentrata, solitamente, in ambiti più applicativi.
Mentre le Agenzie Spaziali, disponendo di risorse cospicue, possono dedicarsi alla creazione di programmi spaziali decennali che coinvolgono migliaia di specialisti al fine di ricerca e sviluppo, l'attività di progettazione dei *cluster* di PMI è spesso finalizzata all'applicazione di modelli di business nel settore spaziale, occupandosi dell'erogazione di servizi o della realizzazione di prodotti per il settore. È raro infatti che piccole o medie organizzazioni intraprendano la realizzazione di missioni puramente scientifiche, o si cimentino nell'ideazione e nello studio di tecnologie all'avanguardia, bensì il loro apporto nello sviluppo tecnologico consiste maggiormente nell'applicazione innovativa di tecnologie già esistenti.
Nel loro contesto operativo, i *cluster* di PMI agiscono in un ambito molto più prossimo all'applicazione dello studio progettuale, preferendo l'impiego di tecnologie già sviluppate per il settore spaziale o l'adattamento di tecnologie esistenti alle necessità dell'ambiente

spaziale, e ricorrendo alla progettazione *ex novo* di componenti e dispositivi di nuova concezione solo se indispensabile. Progettare un componente a partire da specifiche esigenze di progetto, e ottenere la certificazione di questo nuovo componente per l'uso in sistemi spaziali, è un processo molto lungo e costoso.
Di conseguenza, le PMI specializzate nello sviluppo verticale di un prodotto tendono a partecipare ai cluster principalmente per adattare e fornire la propria soluzione, piuttosto che per occuparsi della progettazione di sottosistemi o della configurazione complessiva del sistema.

Ne consegue che i principi del Concurrent Engineering non abbiano gli stessi effetti benefici su aziende diverse, poiché questi si focalizzano sulla gestione di progetti complessi, non solo in termini di architettura del prodotto, ma soprattutto dal punto di vista di integrazione delle differenti discipline ingegneristiche.
Si consideri, ad esempio, una PMI che sviluppi internamente un prodotto con un carattere fortemente *verticale*, ovvero che coinvolga un numero contenuto di discipline ingegneristiche, indipendentemente dalla complessità del prodotto stesso. In questo caso, le attività di progettazione dell'organizzazione sono difficilmente parallelizzabili: a causa delle poche discipline coinvolte, risulta spesso necessario procedere in maniera sequenziale nella progettazione, implicando i risultati ottenuti in precedenza nelle valutazioni successive. In un ambiente operativo del genere, i vantaggi dell'approccio parallelo del CE potrebbero essere sovrastati dalle criticità che si generano dall'adozione di metodologie differenti da quella consolidata.
Tuttavia, l'adozione di strumenti informatici che supportino il progetto dei sistemi, può giovare anche ad organizzazioni che operano tramite queste modalità, permettendo, innanzitutto, l'acquisizione di familiarità con questa tipologia di strumenti, ma soprattutto omologando le attività di progettazione agli approcci collaborativi, facilitando l'inserimento dell'organizzazione stessa in un ambiente più ampio.
Una situazione ben diversa si verifica quando una o più PMI, appartenenti ad un *cluster*, si occupano della realizzazione di sistemi che necessitano un approccio multidisciplinare. In questo caso, la progettazione ingegneristica ne guadagna notevolmente in termini sia qualitativi, grazie, ad esempio, ad una gestione consistente delle informazioni di progetto, sia in termini quantitativi, prospettando riduzioni di tempi e conseguentemente dei costi di sviluppo. La gestione in parallelo delle attività infatti permette l'individuazione preliminare di aspetti anche molto specifici del sistema, anticipando, quando possibile, valutazioni più avanzate, con lo scopo di prevenire eventuali rilavorazioni e ridurre il numero di iterazioni necessarie alla convergenza del progetto verso le caratteristiche e le prestazioni desiderate.
Nella progettazione ingegneristica, e in particolare nello sviluppo multidisciplinare dei sistemi spaziali, si osserva frequentemente un forte accoppiamento tra fenomeni fisici e prestazioni del sistema. Questo complica ulteriormente l'analisi e la valutazione delle prestazioni, richiedendo l'impiego di metodologie numeriche avanzate e sviluppate appositamente per affrontare queste problematiche. Questa branca dell'ingegneria, che prende il nome di Multidisciplinary Design Optimization (MDO), è uno degli argomenti centrali

della progettazione ingegneristica moderna, e si concentra sullo studio di funzioni obiettivo create appositamente per ottimizzare un aspetto del prodotto.
Infatti, sebbene le analisi numeriche e le simulazioni digitali vengano condotte tramite software specializzati per ciascuna disciplina ingegneristica, la necessità di accoppiare diverse analisi impone l'integrazione e la comunicazione tra questi Domain Tools, richiedendo un flusso continuo di dati tra i vari software o, in alcuni casi, l'implementazione di piattaforme dedicate, come avviene con Comet.
Inoltre, nel contesto dei cluster di PMI, la competenza nell'uso di un determinato Domain Tool, ad esempio per l'analisi dello scambio termico di un sistema, può risiedere in un'organizzazione diversa da quella responsabile della configurazione dell'intero sistema e della gestione del System Engineering. Questa distribuzione delle competenze introduce ulteriori sfide nella gestione delle informazioni e del progetto, accentuando la necessità, particolarmente sentita dalle PMI, di adottare una piattaforma software unica, che permetta e gestisca il flusso di dati in maniera consona, facilitando da un lato l'acquisizione delle informazioni dal modello ingegneristico, fornendo una base consistente e unica per le informazioni, dall'altro lato condividendo i risultati delle analisi ottenute tramite i diversi Tools, rendendoli disponibili ai progettisti per le successive valutazioni di carattere MDO.
Dunque, l'adozione dei principi del Concurrent Engineering attraverso software basati su modelli ingegneristici apporta benefici significativi alla gestione di progetti caratterizzati da un'elevata multidisciplinarità. L'impiego di questo tipo di software infatti risulta sempre più diffuso nell'approccio alla progettazione, non solo in campo spaziale. Iniziative come Agile 4[35] per il progetto aeronautico, o la Transizione 4.0 prevista all'interno del Piano Nazionale di Ripresa e Resilienza (PNRR)[36], sono un concreto esempio di come tutto il mondo dell'ingegneria dei sistemi complessi stia producendo nuovi approcci e strategie per il miglioramento del processo stesso di progettazione, convertendo le metodologie tradizionali tramite l'impiego dei progressi dei sistemi informatici, come la potenza computazionale disponibile oggigiorno, il cloud computing e l'intelligenza artificiale.

4.2 Benefici e criticità nell'integrazione del software

Dopo aver analizzato il contesto operativo delle PMI, individuando le esigenze alle quali risponde l'applicazione del Concurrent Engineering, si vuole valutare l'impatto prodotto dall'implementazione di CDP4-CoMET, o di software analoghi, sui processi di progettazione.
L'adozione di strumenti innovativi in procedure consolidate è incentivata dai benefici che l'innovazione apporta, ma comporta, inevitabilmente, delle criticità a cui prestare attenzione. In particolar modo nel settore spaziale, è essenziale considerare aspetti come l'adattamento ai flussi di lavoro esistenti, la curva di apprendimento per gli utenti e la necessità di garantire l'interoperabilità con altri sistemi già in uso.
Come già descritto nell'introduzione generale al Concurrent Engineering e nell'analisi dell'ambito operativo delle Piccole e Medie Imprese nel settore spaziale, i software che implementano lo scambio di dati basato su modelli ingegneristici rappresentano gli strumenti più efficaci per applicare concretamente questa filosofia al processo di progettazione.
L'adozione di tali software produce una serie di vantaggi legati alla parallelizzazione e

alla digitalizzazione degli studi di progettazione. Infatti, la diffusa disponibilità di infrastrutture Internet capaci di elaborare rapidamente grandi quantità di dati, sia tramite rete cablata che wireless, unita alla crescente potenza di calcolo dei dispositivi moderni, consente di implementare, con costi contenuti, sistemi informatici in grado non solo di supportare lo sviluppo del progetto, ma anche di agevolare la collaborazione da remoto. Questo genera un miglioramento del processo sia dal punto di vista qualitativo, in quanto garantisce una gestione più agevole ed efficiente del lavoro di progettazione, svincolando, almeno le fasi più concettuali, da vere e proprie infrastrutture fisiche come gli uffici, sia dal punto di vista quantitativo, espandendo notevolmente le possibilità di collaborazione. Il superamento di barriere, innanzitutto geografiche, offre nuove opportunità per l'aggregazione dei *cluster* di PMI, nei quali si potranno selezionare gli enti partecipanti favorendo, ad esempio, l'affinità di obiettivi e ambizioni, potendo attingere ad un bacino più ampio di organizzazioni.
Sebbene queste osservazioni siano valide in generale nell'ambito della cooperazione online, a questi vantaggi si sommano quelli prodotti dall'utilizzo di modelli ingegneristici per la realizzazione e per lo scambio di dati, integrando in maniera sinergica i benefici tipici del Model Based System Engineering, riassunti nuovamente in figura 4.1. L'archiviazione delle informazioni non avviene più in documenti statici, ma attraverso la loro attribuzione a un modello ingegneristico, sviluppato appositamente per questo scopo.

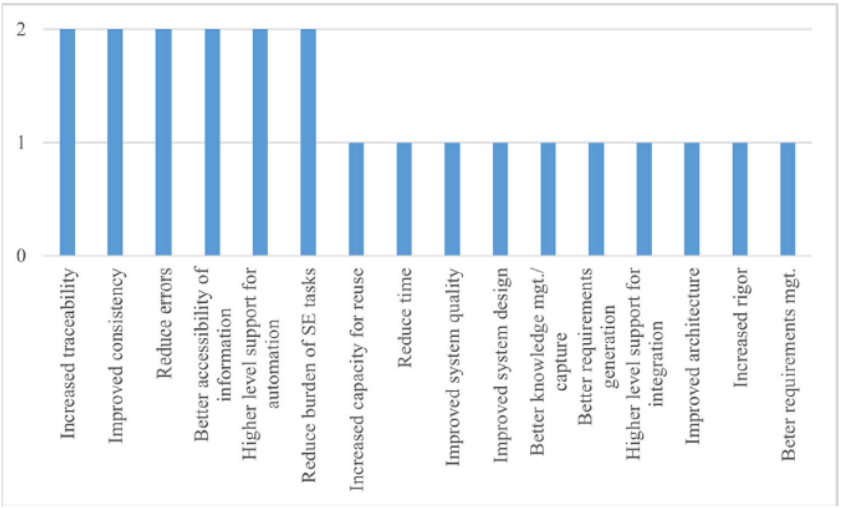

Figura 4.1. Ricorrenze nella letteratura scientifica di dichiarazioni di benefici del MBSE correlate con misurazioni[4].

L'uso di software *General Purpose*, come i fogli di calcolo Excel o i file di testo Word, sebbene sia ampiamente diffuso nel settore, non rappresenta una metodologia efficace per progetti ingegneristici a elevata intensità collaborativa. Anche con l'implementazione di piattaforme di condivisione di file tramite Internet, proprio per il carattere generico di questi software, si avverte la necessità di gestire mansioni e responsabilità, validare le informazioni, automatizzare elaborazioni complesse, e molte altre funzionalità presenti in

software come CDP4-CoMET. Software specializzati che possano gestire adeguatamente i rapporti di collaborazione tra le organizzazioni, ulteriormente arricchiti con le capacità tipiche del MBSE per la progettazione di sistemi complessi, risultano essere uno strumento potente per lo svolgimento dei processi di design.
Inoltre, è frequente che i software specializzati integrino le funzionalità di software *General Purpose*, ad esempio per generare output automatici, garantendo la compatibilità con metodologie *document-centric* più tradizionali. Migliorando la qualità dello scambio dei dati attraverso sistemi strutturati, ci si attende una serie di benefici, tra cui l'univocità delle informazioni, l'individuazione precoce di criticità e la riduzione delle rilavorazioni. Questi aspetti risultano ancora più rilevanti nel contesto delle PMI, dove l'ottimizzazione delle risorse disponibili è cruciale.
Oltre a fornire un importante strumento per la gestione e realizzazione di un progetto, l'adozione di software di questa tipologia, stimola l'organizzazione, o l'intero *cluster* di PMI, a formulare e chiarire le metodologie e gli approcci interni previsti per lo svolgimento del design. Infatti, un fattore fondamentale per il successo della collaborazione, è la condivisione di una struttura di concetti chiara e comune. La definizione di convenzioni accettate e specializzate per l'ambito in cui si intende operare è un passo necessario per permettere il corretto svolgimento delle attività di design collaborativo. Nelle funzionalità di Comet, ad esempio, la possibilità di creare delle Reference Data Library e poter definire parametri con la loro relativa scala e unità di misura, oppure Category che rappresentano concetti ricorrenti nella progettazione, o ancora, Rule che implementano criteri di decomposizione gerarchica dell'architettura del prodotto o dei requisiti di progetto, sono tutte funzionalità che digitalizzano logiche e concetti tradizionalmente utilizzati nell'ambito ingegneristico. Il software quindi, non solo si configura come contenitore delle informazioni tecniche del progetto, ma anche della metodologia del gruppo di progettazione, producendo ulteriori benefici.
In primo luogo infatti, sarà possibile implementare delle verifiche degli approcci stabiliti, così da accertarsi che le attività vengano effettivamente svolte osservando le logiche richieste. In aggiunta, depositando le metodologie all'interno del software, queste vengono effettivamente svincolate dai professionisti che le mettono in atto, preservandone la definizione indipendentemente da chi le utilizza.
Nelle metodologie di un'organizzazione, infatti, rientrano anche le *lesson learned* ed il *know-how* accumulati con l'esperienza e dunque frutto del lavoro precedente: gestire la conoscenza prodotta è di fondamentale importanza per le PMI, e soprattutto per le startup, nelle quali l'ambiente lavorativo risulta molto dinamico[37]. Quindi questi software possono supportare anche i processi di Knowledge Management all'interno di un'azienda[38], garantendo la persistenza e la continuità delle metodologie, indipendentemente dagli individui, e creando un fondamento solido e condiviso sul quale basare la cooperazione.
La possibilità di definire una metodologia interna all'azienda tramite software come CDP4-Comet, o Virtual Satellite, presenta un ulteriore aspetto da analizzare. Infatti, l'utilizzo di software sviluppati su una struttura formulata dall'ECSS, anche se in un Technical Memorandum, introduce le aziende a concetti e approcci che vanno oltre l'implementazione del software stesso.

La concettualizzazione dei processi di progettazione ingegneristica, che costituisce l'ontologia di questi software, proviene infatti da una fonte comune, producendo analogie significative nella realizzazione di questi software. Ci sono infatti diversi esempi di corrispondenze, come l'Element Definition e il rispettivo Element Usage, o Element Occurence per VS; o la gestione degli utenti tramite ruoli e Domain, o Discipline; o la creazione di parametri di progetto tramite Quantity Kind; che fanno notare come questi software producano delle strutture informatiche simili, che possono risultare compatibili o facilmente integrabili.
L'utilizzo di software di questo tipo, quindi, permette di formulare e contenere gli approcci di un gruppo di progettazione e, a questo scopo, propone una struttura concettuale condivisa. Adoperare funzionalità simili produce conseguentemente analogie, soprattutto formali, negli approcci stabiliti dalle aziende nell'ambito di questi software, facilitando la comparazione e l'integrazione delle differenti strategie di design.
Infine, oltre ad abilitare tecnologicamente la cooperazione remota in maniera agile, l'adozione di Comet da parte dell'ESA costituisce una forte dichiarazione di interesse verso questi nuovi strumenti open source, suggerendo che questi possano diventare presto degli standard effettivi per lo sviluppo del progetto spaziale, o dei requisiti necessari per la partecipazione ad ambiti di collaborazione internazionale.

L'adozione di software per il Concurrent Engineering, pur presentando numerosi vantaggi, introduce anche delle criticità che devono essere attentamente considerate.
Nonostante l'implementazione di questi software segua le indicazioni individuate dall'ECSS, l'utilizzo di questi software non permette in nessun modo di validare e certificare le metodologie ed il progetto in sé. Le necessità di ogni gruppo di progettazione sono infatti troppo specifiche e differenti per essere svolte tramite processi totalmente standardizzati, senza limitare eccessivamente le possibilità degli utenti.
Di conseguenza, l'utilizzo di questa tipologia di software non fornisce supporto nel processo di certificazione dei prodotti spaziali, che dunque rimane un aspetto oneroso dello sviluppo di sistemi nel settore. Tuttavia, la loro flessibilità consente di personalizzarne l'impiego, adattando le funzionalità disponibili alle esigenze specifiche di ogni ruolo coinvolto nel progetto. Infatti, nell'ampio spettro di mansioni coinvolte nel design, potrebbe verificarsi che alcune funzionalità siano troppo dettagliate per un ruolo, ma non sufficientemente specifiche per un altro.
Bisogna sempre tenere a mente che il software, e il modello ingegneristico prodotto, è al servizio dell'utente e mai viceversa. Se si perde di vista l'obiettivo finale, la creazione di *Digital Twin*[39] rischia di diventare un'attività eccessivamente complessa, perdendo il suo scopo originale, con un impatto negativo sulla gestione delle risorse. Individuare correttamente i processi in cui applicare questi strumenti non è sempre banale e dipende fortemente dalle caratteristiche specifiche del contesto operativo. La riconversione di metodologie consolidate, infatti, deve essere sempre guidata dallo scetticismo ma non deve farsi ostacolare da pregiudizi o fattori culturali. Inoltre, l'adozione di questi strumenti deve tenere in conto dei vincoli economici e temporali ai quali questi processi di sviluppo sono ovviamente sottoposti.
Insieme a questi aspetti, bisogna prevedere un'opportuna preparazione per l'adozione

di questi strumenti. Essendo software piuttosto strutturati, il loro utilizzo non risulta sempre intuitivo e bisogna quindi disporre di percorsi formativi per introdurre i partecipanti al gruppo di progettazione ai concetti e alle funzionalità da adottare nel loro lavoro. Questo impone l'investimento di risorse economiche proporzionali al tempo dedicato alla formazione di competenze nell'utilizzo dei software, che vanno periodicamente aggiornate integrando le novità introdotte dallo sviluppo del software stesso. Infatti, essendo strumenti relativamente recenti, software come Virtual Satellite o Comet vengono frequentemente aggiornati per risolvere problemi esistenti o per introdurre nuove caratteristiche del software. Questa necessità di aggiornamento periodico, sia del software che delle competenze di chi lo utilizza, può essere accettata, sfruttando i benefici introdotti nelle nuove release dei software, oppure può essere mitigata decidendo arbitrariamente se e quando adottare nuove versioni. Un'ulteriore difficoltà è rappresentata dalla carenza di documentazione per le funzionalità più avanzate, che spesso risultano essere le più utili per la modellazione di sistemi complessi. Approcciare a funzionalità come lo Scripting di Comet potrebbe risultare piuttosto ostico per un utente inesperto nella programmazione ad oggetti, data anche la mancanza di descrizioni esaustive dello specifico ambiente di sviluppo. Tuttavia, ci si aspetta che con il tempo e l'aumento nella diffusione di questi software, si accresca e arricchisca anche il bagaglio di informazioni disponibili, facilitando l'utilizzo delle funzionalità.
Un costo aggiuntivo da considerare per l'adozione di questa tipologia di software nasce dal bisogno di un'infrastruttura hardware in grado di erogare i servizi necessari per il corretto funzionamento di questi strumenti. In ogni caso, questi servizi non richiedono specifiche hardware particolarmente esigenti, permettendo, ad esempio, di accorpare l'erogazione di questi servizi su eventuali macchine già a disposizione dell'organizzazione. Infatti è usuale che le aziende siano dotate di un'infrastruttura hardware per servizi, come siti Internet o domini personalizzati di mail, fondamentali nello svolgimento moderno dei processi di design di sistemi. Tuttavia, mentre per un servizio secondario come un sito Internet, un'azienda potrebbe valutare di affidarsi a terze parti, per servizi che invece riportino informazioni sensibili sull'organizzazione e su dettagli tecnici di un progetto, potrebbe essere opportuno possedere fisicamente la macchina hardware, al fine di gestire internamente l'erogazione del servizio, gli accessi alle informazioni e la struttura stessa dei dati. In aggiunta ai costi associati al funzionamento del server e all'erogazione dei servizi, bisogna considerare anche i costi di manutenzione e gestione sia della macchina che esegue il servizio, sia dei dati in sé che il software produce. Infatti, per una corretta amministrazione di questi servizi, è una buona pratica prevedere backup periodici e il monitoraggio dello stato della macchina e della struttura dei dati immagazzinati.
L'amministrazione di un server e l'erogazione di servizi via web comporta una serie di aspetti di gestione dei dati e della sicurezza informatica fondamentali da valutare nel contesto operativo moderno [40]. La protezione delle informazioni deve essere garantita attraverso l'individuazione delle minacce e delle vulnerabilità relative allo specifico sistema informatico, e deve avvenire tramite l'adozione degli standard stabiliti dagli enti certificatori. Ad esempio, la serie ISO/IEC 27000 definisce i *Sistemi di Gestione per la Sicurezza delle Informazioni*[41], stabilendo una panoramica dei concetti chiave e chiarendo i requisiti dei sistemi di gestione della sicurezza e le norme da osservare in questo

settore.
Gli stessi software considerati inoltre, suggeriscono degli accorgimenti per migliorare il livello intrinseco di sicurezza, prima di un eventuale adozione di sistemi esterni. Innanzitutto, come già presentato nella descrizione del deployment del server nel capitolo 3, è opportuno prevedere l'utilizzo di un server proxy per filtrare e gestire il traffico verso il server. Tramite un proxy che faccia da intermediario tra client e server, si possono nascondere informazioni sensibili come l'Internet Protocol (IP) del server, ma soprattutto monitorare il traffico di dati in entrata e in uscita dal server, migliorando il monitoraggio delle comunicazioni. Ulteriormente, bisognerebbe prevedere l'utilizzo di un certificato Secure Socket Layers (SSL) che possa cifrare e autenticare la trasmissione di dati tra client e server, riducendo i rischi legati all'intercettazione dei dati. Infine, è possibile prevedere dei servizi aggiuntivi che possano implementare dei protocolli di autenticazione come Lightweight Directory Access Protocol (LDAP) oppure OpenID Connection (OIDC). Questi protocolli, da utilizzare in combinazione con i dispositivi di sicurezza precedentemente introdotti, consentono di fornire servizi di autenticazione più solidi, permettendo, ad esempio, un sistema unico per accedere a diversi servizi collegati, oppure abilitando l'autenticazione multi fattore (MFA) che migliora significativamente la sicurezza del protocollo.
In definitiva, l'adozione di software per il Concurrent Engineering richiede un'attenta valutazione delle criticità connesse alla loro implementazione. L'integrazione efficace di questi strumenti dipende da un equilibrio tra le esigenze operative, i vincoli economici e la necessità di garantire sicurezza ed efficienza dei processi di progettazione.

Capitolo 5

Conclusioni

Ripercorrendo questa Tesi Magistrale, si individua l'esigenza di supportare i processi di progettazione ingegneristica in un ambiente fortemente collaborativo. Questa esigenza, già da alcuni decenni, ha suscitato l'ideazione di metodi e principi che, nel loro insieme, vanno a costituire un vero e proprio approccio filosofico alla progettazione. Si è descritto il ruolo del Concurrent Engineering nelle attività ingegneristiche, evidenziando le principali vie per l'attuazione dei suoi principi, e individuato il software CDP4-CoMET come principale strumento.
Quindi si è approfondita la descrizione del software CDP4-CoMET, per far sì che possa essere un utile riferimento sulle funzionalità per le future attività dei membri di Space V, e per chiunque sia interessato all'utilizzo del software. La descrizione delle funzionalità è stata estratta dal manuale utente e integrata con l'esperienza raccolta e le informazioni fornite dalla casa di sviluppo nelle repository Github.
In seguito, le funzionalità del software sono state analizzate per rispondere alle esigenze di progettazione della startup Space V, e sono stati creati due modelli ingegneristici per implementare gli approcci individuati. Il primo, un Template Model, più semplice e generale, è stato prodotto come un possibile punto di partenza per futuri studi di progettazione. Il secondo modello, invece, si basa su un precedente studio preliminare, da cui sono state acquisite le informazioni per svolgere alcune analisi tipiche del progetto di missione spaziale. Tramite la digitalizzazione dello studio considerato, questo modello vuole essere un riferimento iniziale per l'azienda, sull'utilizzo delle funzionalità del software.
Infine, si generalizzano gli approcci di adozione di software per supportare il CE, analizzando il contesto operativo dei *cluster* di PMI e individuando analogie e differenze nell'applicazione del CE nelle grandi aziende. Quindi, sono stati evidenziati i benefici e le criticità dell'impiego di strumenti informatici per lo scambio dati tramite modelli nella cooperazione tra PMI.
Purtroppo, la valutazione dell'impatto dell'adozione di questi approcci nelle attività di lavoro di Space V richiederebbe un'analisi ben più duratura di un lavoro di tesi, per contemplare le evoluzioni delle diverse fasi di progetto. Dunque, al momento, non è possibile valutare gli effetti dell'implementazione del CE.

In ultima analisi, si propongono alcuni aspetti che potrebbero essere sviluppati in

implementazioni future. Innanzitutto, le funzioni definite negli script potrebbero essere ampliate per includere nuove elaborazioni, come la valutazione del SML, e ulteriormente ottimizzate, individuando eventuali criticità o comportamenti inaspettati del codice in condizioni di funzionamento specifiche. Inoltre, l'architettura del client CDP4-CoMET IME prevede l'utilizzo di Event Listener per l'esecuzione di operazioni automatiche interne, al verificarsi di certe condizioni. Utilizzando queste classi nell'implementazione degli script, si potrebbero abilitare questi ultimi ad automatizzare pienamente svariati aspetti del progetto ingegneristico, ad esempio calcolando la massa di un sistema, ad ogni modifica di un parametro di massa di un elemento annidato, aggiornando i valori di riepilogo in tempo reale con le modifiche degli utenti.
Infine, risulta mancare all'interno del software una funzionalità dedicata alla gestione dei rischi. Potrebbe essere sviluppato un plug-in che implementi, in una classe apposita, l'oggetto Risk e una schermata dedicata per la gestione. La definizione dell'oggetto Risk potrebbe essere molto simile a quella, già presente, di Requirements. Infatti, rappresentando i rischi con una descrizione, dei parametri predefiniti per riportare, ad esempio, probabilità e severità e garantendo all'utente la facoltà di associare parametri e categorie dalle RDL, si potrebbero generare dei Risk Register per svolgere ulteriori analisi di mitigazione, o quantomeno, abilitare gli utenti alla formulazione di script che possano utilizzare la rappresentazione del concetto di rischio.

Bibliografia

[1] A. Golkar D. Knoll, C. Fortin. Review of concurrent engineering design practice in the space sector: state of the art and future perspectives. *International Journal of Innovation and Technology Management*, page 393–414, 2011.

[2] ECSS Executive Secretariat. ECSS-E-TM-E-10-25A Engineering design model data exchange (CDF), Oct 2010.

[3] Deutsches Zentrum für Luft-und Raumfahrt. Virtual satellite website, . URL https://www.dlr.de/en/sc/research-transfer/software-solutions/virtual-satellite.

[4] Kaitlin Henderson and Alejandro Salado. Value and benefits of model-based systems engineering (mbse): Evidence from the literature. *Systems Engineering*, 24(1):51–66, 2021.

[5] Harry W. Jones. The recent large reduction in space launch cost. In *International Conference on Environmental Systems (ICES)*, Ames Research Center, 2018.

[6] Paolo Castelnovo, Gelsomina Catalano, Francesco Giffoni, and Matteo Landoni. The outcomes of public procurements: an empirical analysis of the italian space industry. *The Journal of Technology Transfer*, 49(1):367–399, 2024.

[7] Patrizia Bagnerini, Mauro Gaggero, Marco Ghio, and Franco Malerba. The adaptive vertical farm as an efficient tool for the cultivation of multiple crops in space. In *2023 IEEE 10th International Workshop on Metrology for AeroSpace (MetroAeroSpace)*, pages 231–236. IEEE, 2023. ISBN 1665456906.

[8] Josip Stjepandic, Nel Wognum, and Wim J. C. Verhagen. *Concurrent engineering in the 21st century: Foundations, Developments and challenges. Introduction of the Book*, page 1–17. Springer, 2015.

[9] Massimo Bandecchi, B Melton, and B Gardini. The ESA/ESTEC concurrent design facility. 01 2000.

[10] Gabriel Karpati, John Martin, Mark Steiner, and K Reinhardt. The integrated mission design center (imdc) at nasa goddard space flight center. In *IEEE Aerospace Conference*, 2002.

[11] Bryan R. Moser and Ralph T. Wood. *Complex Engineering Programs as Sociotechnical Systems*, pages 51–65. Springer International Publishing, Cham, 2015. ISBN 978-3-319-13776-6. doi: 10.1007/978-3-319-13776-6_3. URL https://doi.org/10.1007/978-3-319-13776-6_3.

[12] International Council on Systems Engineering. Incose international symposium, 1991. ISSN 2334-5837.

[13] Eclipse Foundation. Capella | open source mbse tool, github repository, . URL https://github.com/eclipse-capella/capella.

[14] Hans Peter de Koning, Sam Gerené, Ivo Ferreira, Andrew Pickering, Friederike Beyer, and Johan Vennekens. Open concurrent design tool—esa community open source ready to go. In *6th International Conference on Systems and Concurrent Engineering for Space Applications, Stuttgart*, 2014.

[15] Brittany Wickizer, Timothy Snyder, James DiCorcia, Ronald Evans, Roland Burton, and David Mauro. A new concurrent engineering tool for the mission design center at nasa ames research center. In *2021 IEEE Aerospace Conference (50100)*, pages 1–12. IEEE, 2021.

[16] Starion Group. Cdp4-comet: a powerful mbse platform for concurrent design, . URL https://www.stariongroup.eu/services-solutions/system-engineering/concurrent-design/cdp4-comet/.

[17] Deutsches Zentrum für Luft-und Raumfahrt. Virtual satellite, github repository, . URL https://github.com/virtualsatellite/VirtualSatellite4-Core.

[18] Eclipse Foundation. Eclipse, github repository, . URL https://github.com/eclipse.

[19] Paloma Maestro Redondo, Peter Dubock, and Gwendolyn Kolfschoten. A tool to support concurrent reviewing of concurrent engineering products. 2024.

[20] Starion Group. Cdp4-comet-ime, github repository, . URL https://github.com/STARIONGROUP/COMET-IME-Community-Edition.

[21] Starion Group. Cdp4-comet-webservices, github repository, . URL https://github.com/STARIONGROUP/COMET-WebServices-Community-Edition.

[22] Starion Group. Cdp4-comet user manual, . URL https://www.stariongroup.eu/wp-content/uploads/2024/04/CDP4-COMET-User-Manual-2024.pdf.

[23] M Jastram. How the reqif standard for requirements exchange disrupts the tool market. *Requir Eng Mag.*, Volume 13, 2019. URL https://remagazine.ireb.org/articles/open-up.

[24] Starion Group. Cdp4-comet-webapp, github repository, . URL https://github.com/STARIONGROUP/COMET-WEB-Community-Edition.

[25] Starion Group. Dehp matlab adapter, github repository, . URL https://github.com/STARIONGROUP/DEH-MATLAB.

[26] Brian J Sauser, Jose E Ramirez-Marquez, Devanandham Henry, and Donald DiMarzio. A system maturity index for the systems engineering life cycle. *International Journal of Industrial and Systems Engineering*, 3(6):673–691, 2008.

[27] Bianca Simonassi. Studio preliminare di spazializzazione della serra verticale adattiva di space v come payload per la stazione spaziale internazionale (iss). uso di flow 3d per il dimensionamento del sistema di fertirrigazione basato sui porous tube plant nutrient delivery system (ptpnds). Tesi secretata. Fulltext non presente, Luglio 2024. URL http://webthesis.biblio.polito.it/32274/.

[28] Echrak Chnib, Patrizia Bagnerini, Mauro Gaggero, and Ali Zemouche. Parameter estimation of a dynamic growth model for lettuce in an adaptive vertical farm. In *2024 IEEE 20th International Conference on Automation Science and Engineering (CASE)*, pages 1169–1174, 2024. doi: 10.1109/CASE59546.2024.10711477.

[29] Starion Group. Cdp4-comet-sdk, github repository, . URL https://github.com/STARIONGROUP/COMET-SDK-Community-Edition.

[30] D. Pezzella. Scripts-for-cdp4-comet, github repository. URL https://github.com/DarioPezzella/Scripts-for-CDP4-Comet.

[31] Starion Group. Comet-reports, github repository, . URL https://github.com/STARIONGROUP/COMET-Reports.

[32] Starion Group. Cdp4-comet-models, github repository, . URL https://github.com/STARIONGROUP/CDP4-COMET-MODELS.

[33] A. CORALLO M. LAZOI, F. CECI and G. SECUNDO. Collaboration in an aerospace smes cluster: innovation and ict dynamics. *International Journal of Innovation and Technology Management*, page 393–414, 2011.

[34] Simona Gabor. Intellectual property management: an important tool for small and medium enterprises. *Anale. Seria Stiinte Economice. Timisoara*, 19:282, 2013.

[35] AGILE 4.0 Consortium. AGILE 4.0 – Towards Cyber-Physical Collaborative Aircraft Development, 2025. URL https://www.agile4.eu/.

[36] Ministero delle Imprese e del Made In Italy. Transizione 4.0. URL https://www.mimit.gov.it/index.php/it/pnrr/progetti-pnrr/pnrr-transizione-4-0.

[37] Susanne Durst and Stefan Wilhelm. Knowledge management in practice: insights into a medium-sized enterprise's exposure to knowledge loss. *Prometheus*, 29(1): 23–38, 2011.

[38] Kevin C Desouza and Yukika Awazu. Knowledge management at smes: five peculiarities. *Journal of knowledge management*, 10(1):32–43, 2006.

[39] Götz Philip Brasche, Josef Eichinger, and Juergen Grotepass. *The Role of Digital Twins for Trusted Networks in the "Production as a Service" Paradigm*, pages 181–203. Springer International Publishing, Cham, 2023. ISBN 978-3-031-21343-4. doi: 10.1007/978-3-031-21343-4_7. URL https://doi.org/10.1007/978-3-031-21343-4_7.

[40] Bilge Yigit Ozkan and Marco Spruit. Cybersecurity standardisation for smes: The stakeholders' perspectives and a research agenda. *Research Anthology on Artificial Intelligence Applications in Security*, pages 1252–1278, 2021.

[41] Georg Disterer. Iso/iec 27000, 27001 and 27002 for information security management. *Journal of Information Security*, 4(2), 2013.