Dragon Architecture - DRAFT

The Dragon Architecture is the code name for the Architecture of a Mission Enterprise for Digital Thread approaches. It leverages concepts like Quantitative Engineering, Software Defined Systems and Engineering Information Science built on COTS platforms that emphasize Internet-based standards and protocols for User Experience, Scale and Interoperability.

Quantitative Engineering is the transformation of traditional engineering practices into formalized executable representations such that the engineering work being done can now be performed in an entirely quantitative manner.

Software Defined Systems is an extension of Quantitative Engineering where by the engineering models are cohesive enough to be used to represent the complete product or system of products. This representation becomes the embodiment of requirements and design of the system or product.

Both of these are based on the Executable Systems Engineering Methodology - ESEM.

The Dragon Architecture centers around the concept of Formal Explicit Qualification, which is rooted in the utilization of the Change Package framework and Digital Threads.

It's important to emphasize that the Dragon Architecture maintains a complete neutrality towards the specifics of the modeling methodology adopted. Its sole purpose is to elucidate the manner in which models can be seamlessly integrated with both documents and software systems.

Crucially, Dragon doesn't mandate the existence of a centralized model. Rather, it hinges on the ability to harness models for establishing connections with code. It remains incumbent upon individual projects to determine the approach, if any, they wish to adopt in this regard.

An illustration of the Dragon concept is the engineering document (e.g. System Design Descriptions, Interface Control Documents), which stands as an exemplar of a model-based document. However, it's imperative to note that engineering document serves merely as an instance and not a comprehensive representation.

Often the realm of control software and their models is disconnected from systems engineering. This underscores the pivotal role and inherent worth of generating code and tests bolstered by explicit traceability. Behavioral models, such as those depicting scenarios, exemplify this paradigm.

The decision to model scenarios and the methodologies employed therein is distinctly a project-specific prerogative, separate from the framework set forth by Dragon. The crux of Dragon's contribution lies in delineating the precise process through which these models undergo unequivocal traceability within the qualification trajectory.

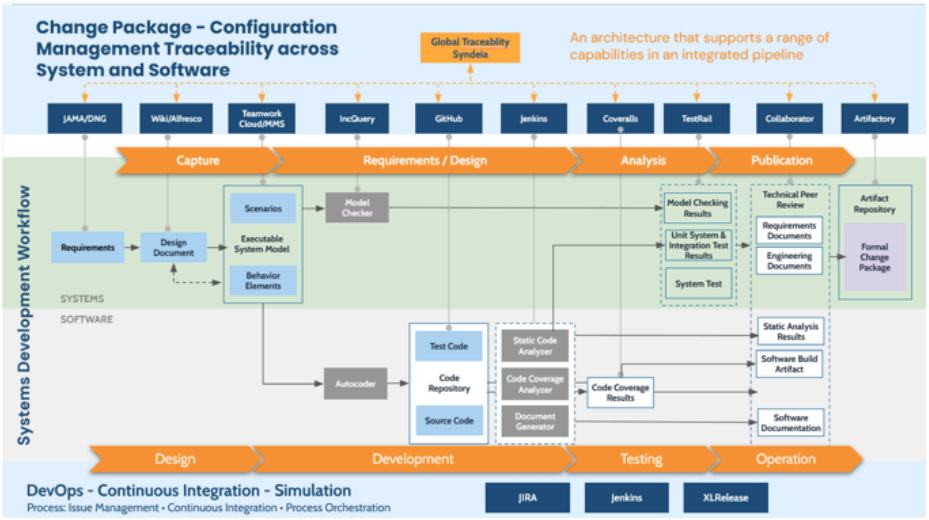
It is pertinent to highlight that the entirety of Dragon adheres to RDF compatibility, making it adaptable to any Model-Based representation. The framework refrains from imposing any requirements specific to representation, be it in relation to models, documents, or any other aspect.

Short IEEEAC2023-Towards a Model-Based Product Development Process from Early Concepts to Engineering Implementation

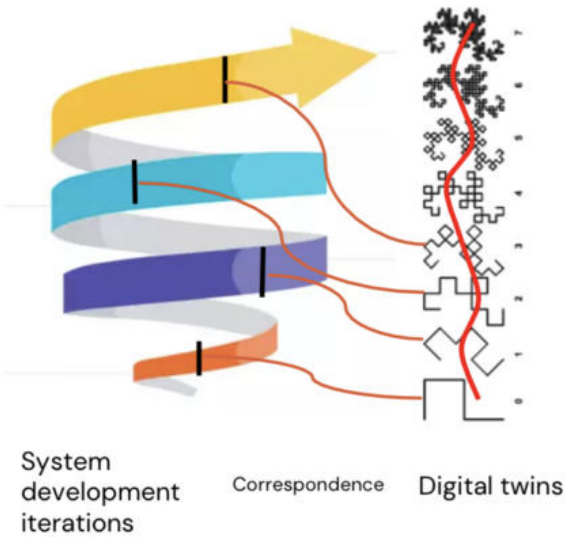
Towards a Model-Based Product Development Process from Early Concepts to Engineering Implementation-v3c.pdf

Dragon notional functional flow

This figure shows how different technologies could be leveraged without prescribing or precluding any.

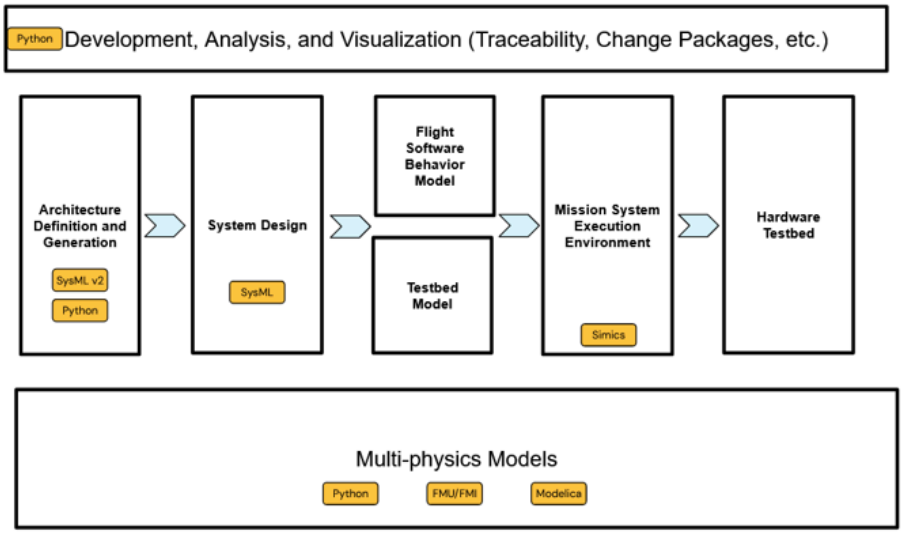


The Dragon functional flow supports the different phases of a Model-Based Development Method such as ESEM.



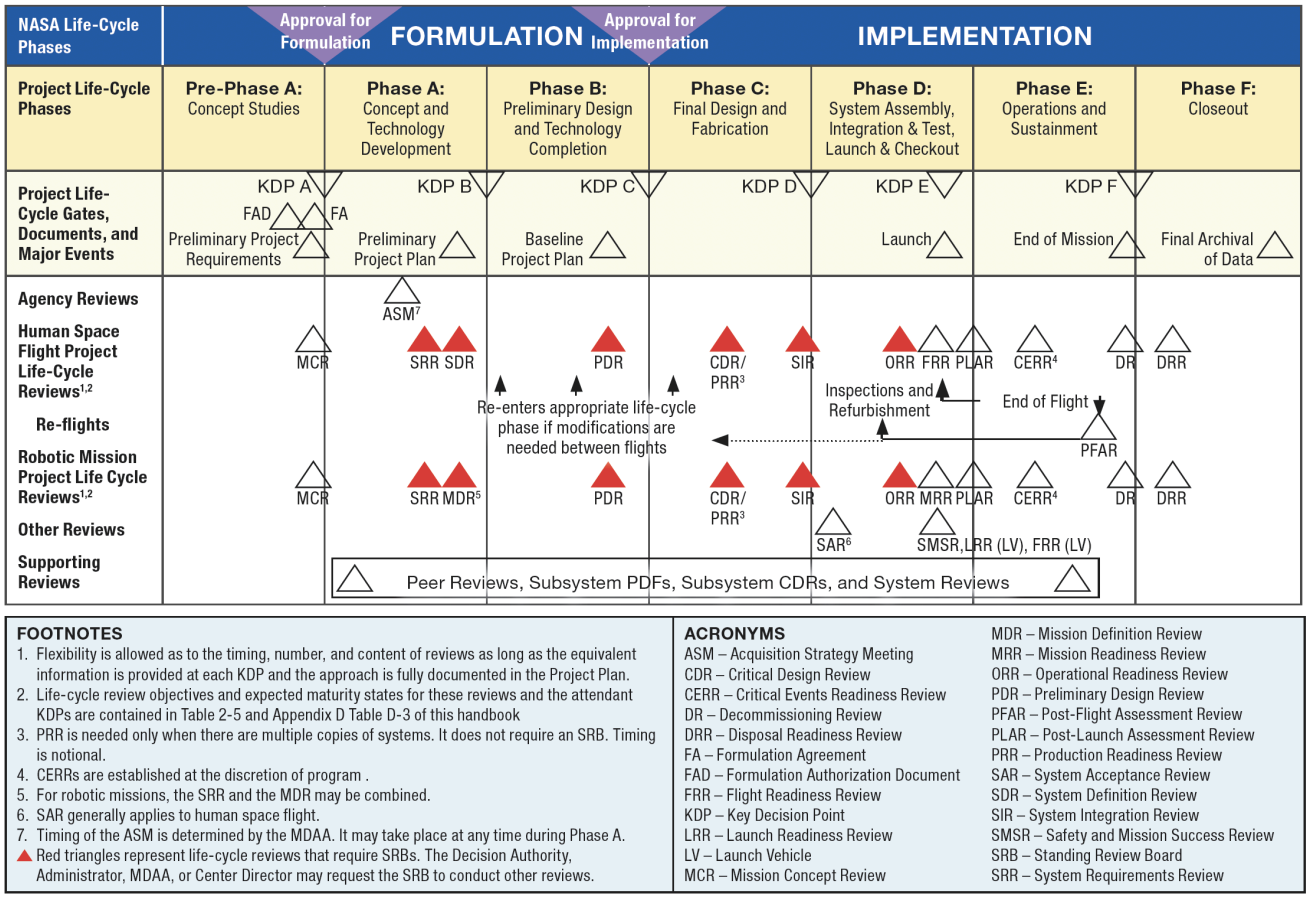
Dragon Digital Twin Pipelines

A notional digital twin development pipeline.



Mission liefcycle

The Dragon Architecture, Digital Twin Pipelines, and ESEM support different missions life cycles such as the NASA Life-Cycle Process:



FOOTNOTES

1. Flexibility is allowed as to the timing, number, and content of reviews as long as the equivalent information is provided at each KDP and the approach is fully documented in the Project Plan.
2. Life-cycle review objectives and expected maturity states for these reviews and the attendant KDPs are contained in Table 2-5 and Appendix D Table D-3 of this handbook
3. PRR is needed only when there are multiple copies of systems. It does not require an SRB. Timing is notional.
4. CERRs are established at the discretion of program .
5. For robotic missions, the SRR and the MDR may be combined.
6. SAR generally applies to human space flight.
7. Timing of the ASM is determined by the MDAA. It may take place at any time during Phase A.

▲ Red triangles represent life-cycle reviews that require SRBs. The Decision Authority, Administrator, MDAA, or Center Director may request the SRB to conduct other reviews.

ACRONYMS

ASM – Acquisition Strategy Meeting
CDR – Critical Design Review
CERR – Critical Events Readiness Review
DR – Decommissioning Review
DRR – Disposal Readiness Review
FA – Formulation Agreement
FAD – Formulation Authorization Document
FRR – Flight Readiness Review
KDP – Key Decision Point
LRR – Launch Readiness Review
LV – Launch Vehicle
MCR – Mission Concept Review
MDR – Mission Definition Review
MRR – Mission Readiness Review
ORR – Operational Readiness Review
PDR – Preliminary Design Review
PFAR – Post-Flight Assessment Review
PLAR – Post-Launch Assessment Review
PRR – Production Readiness Review
SAR – System Acceptance Review
SDR – System Definition Review
SIR – System Integration Review
SMSR – Safety and Mission Success Review
SRB – Standing Review Board
SRR – System Requirements Review

FIGURE 3.0-1 NASA Space Flight Project Life Cycle from NPR 7120.5E

source: https://www.nasa.gov/sites/default/files/atoms/files/nasa_systems_engineering_handbook_0.pdf

Languages, Standards and Technologies

Dragon leverages formal languages, open Standards, and a variety of technologies

Language	Function	IDE Technology	Standard
SysML (v2)	System Design	Jupyter	OMG (Object Management Group)
Python (v3)	System Development, Analysis, and Visualization	Jupyter	Python Software Foundation
Modelica	System Design and Analysis	Modelon Impact	The Modelica Association

SysML (v1)	Systems Design and Analysis	Cameo Systems Modeler OpenMBEE	OMG
BPMN (v2)	Process Design and Analysis	Cameo Systems Modeler	OMG
UAF	Systems Architecture and Analysis	Cameo Systems Modeler	OMG
FMU/FMI	Simulation	Cameo Systems Modeler Modelon Impact pyfmi	The Modelica Association