ABSTRACT
This paper introduces the EAGLE model-based design simulator framework for designing and testing guidance navigation algorithms for planetary probe entry / descent, landing and terminal-phase algorithms. EAGLE has been developed to aid systems engineers involved in the design, maturation, and evaluation of these algorithms throughout the design phases of the space system engineering life-cycle. This paper will describe the design goals and the principal elements of the software. The work-flow for maturing the technology readiness level of the algorithms will also be discussed.

1. INTRODUCTION
Entry, descent, and landing (E/DL) is one of the most critical parts of a surface planetary mission following the launch sequence. It is also a mission phase that is most reliant on spacecraft autonomy with little scope for human intervention. For this reason, E/DL is subject to high levels of simulation-based evaluation during each of the design phases of the space system engineering lifecycle (SSEL) [1].

Fig. 1 shows the SSEL for a given space system from mission analysis need (phase 0) to system disposal (phase F.). Design of a given space system takes place in phases A, B and C. The mission needs and identification of possible system concepts takes place in phase 0.

Each design phase uses a different level of simulation fidelity, with the fidelity increasing as the mission concept and associated design matures. Simulation is used in the subsequent phases but is not used to increase the maturity level of the space system under consideration. For example, in phase E, operational simulators are used to perform operator training and in-orbit validation if the system under consideration is a space vehicle. These simulators are space systems in their own right and follow their own SSEL.

The Entry and Guided Landing Environment (EAGLE) is an environment based on MATLAB® / Simulink® and other commercial-of-the-shelf (COTS) and open-source software (OSS) for systems engineers to create mission-specific design and verification / validation simulators for E/DL applications in a repeatable manner.
The initial design goal of EAGLE is to provide a platform for the design and evaluation of GNC algorithms for Mars EDL (such as the ESA Mars EDL Demonstrator) and Lunar DL (such as the ESA Lunar Lander) vehicles. However, it can be applied to other applications such as Earth, Venus and Titan EDL. Additionally, it can be applied to non-ED/L spacecraft applications such as aerobraking and asteroid rendezvous / landing.

Although primarily aimed at the SSEL design phases for GNC, EAGLE can also be used to create simulators (and simulation components) for evaluating candidate mission concepts during phase 0 and phase D (where specialized simulators are required for hardware / software integration and test.)

EAGLE provides a framework for creating coupled multi-body vehicle and vehicle / environmental simulators using a model-based design (MBD) philosophy. This approach requires mathematical models (either first-principles models, data-centric models such as look-up tables and polynomial approximations, or combinations of the two) to be integrated so that a single mathematical model is obtained. The MBD philosophy then requires this model to be simulated with the results being used to drive mission and design decisions.

The target users include:

1. **Mission Systems Engineers:**
   Mission Systems Engineers perform feasibility studies and design the mission concepts and architectures. Additionally, mission systems engineers select the mission system candidate(s) that is (are) most likely to satisfy the science, cost, and schedule constraints imposed by the programme.

2. **Systems Engineers:**
   Systems engineers perform integrated system design and system trade-offs to obtain solutions that are optimal with respect to the mission goals and constraints. They flow down mission and system requirements to subject matter experts (SMEs) such as GNC engineers.

3. **GNC Engineers:**
   GNC Engineers develop guidance, navigation and control (GNC) algorithms, hazard detection and avoidance (HAD) algorithms. Additionally, GNC engineers specify / select GNC related hardware (such as sensing and actuation subsystems) and place requirements on the on-board computing resources.

4. **Other Subject Matter Experts:**

SMEs such as specialists in decelerator technologies, propulsion systems, power systems and other hardware engineers can provide subsystem and component models and integrate them into EAGLE.

3. **EAGLE CAPABILITIES**

The EAGLE framework is used to create simulators of appropriate fidelity for the design, testing and maturation of GNC and related algorithms. It supports the V-model [2] of system engineering as shown in Fig. 2.

![Fig. 2: EAGLE supports the V-model of system engineering.](image)

The V-model is supported through automated testing at every stage of the system design process (test suites are executed corresponding to the different design stages on left hand side of the diagram.) Additionally, the flight software verification and validation activities are supported by the following (sequential) simulation modes:

1. **Model-in-the-loop (MIL):** Native Simulink blocks are used to represent the environment, the vehicle and the GNC algorithms / other flight software.

2. **Software-in-the-loop (SIL):** The blocks representing the flight software are converted to code for the host platform (upon which EAGLE executes) using TargetLink®. This code is compiled and linked with Simulink and replaces the corresponding native Simulink blocks.

3. **Processor-in-the-loop (PIL):** The blocks representing the flight software are converted to code for the target platform (a flight-representative processor.) The vehicle and environment models are converted to code using TargetLink and compiled on a dSpace® real-time, hardware system. A network then connects the processor and the real-time vehicle / environment model to enable feedback control.

4. **Hardware-in-the-loop (HIL):** This is the same as the above but with flight hardware (such as sensors) replacing elements of the vehicle model.
An important part of flight software is its ability to respond to the effects of various in-flight contingencies (such as the occurrence of a failure mode or an unexpected environmental condition.) For this reason, EAGLE provides the user with the ability to trigger faults in the vehicle model’s subsystems and to change parameters that represent environmental variables. These faults and parameter changes can also be triggered from MATLAB scripts permitting automated testing of robustness and fault detection and recovery (FDIR) algorithms.

EAGLE is also able to support phase D and phase E of a spacecraft’s lifecycle by permitting the generation of ESA’s simulation portability (SMP) standard compliant code (for SMP1 and SMP2) through the Mosaic [3] target for Real-Time Workshop®. The automatically generated code can be compiled and linked into other frameworks such as real-time operational simulators for training, in-orbit validation, and vehicle anomaly-handling activities.

4. ARCHITECTURE AND DESIGN
EAGLE consists of the following elements (see Fig. 3):
1. The simulation kernel,
2. Analysis tools,
3. Design tools,
4. Graphical User Interface.

Fig. 4: The EAGLE blockset.
The blockset is arranged into six sub-libraries:
1. Dynamics: 3 / 6 DOF (Euler angle and quaternion computation), ablating rigid body, mass properties evolution, tank dynamics, fuel slosh, flexible structures, parachutes and airbags.
2. Environment: Atmospheric (simple model, Earth models: NRL-MSISE00, GRAM2007, Mars models: EMCD, MarsGRAM2005), ground interactions and gravity (Nth order harmonics.)
4. Actuators: On/off thrusters (suitable for pulse-width modulation control and timed control) and continuously variable thrusters.
5. Logic / Control: Flight regime modes (ballistic entry, parachute deceleration, powered descent, landing) and GNC / HDA models.
6. Sensors: Landing LIDAR (provided through PANGU2 [4]), accelerometers / gyros, camera, pressure measurement, Doppler radar (provided through PANGU), star tracker.

2 PANGU: Planet and Asteroid Natural Scene Generation Utility software created by the Space Systems Research Group, the University of Dundee.
Comprehensive documentation for each block is provided through the Simulink on-line help facility. New blocks can be integrated into the appropriate sub-library provided that they adhere to the EAGLE block interface standard. Blocks undergo a strict qualification procedure by which simulated data is compared with actual data (for example, from the manufacturer of the hardware being modelled or from actual space missions.)

**Analysis tools:** Post-simulation analysis of results can be made through a number of different tools. These include statistical (such as landing dispersion as shown in Fig. 5) and dynamic analyses (such as dynamic pressure as shown in Fig. 6.)

![Fig. 5: Landing dispersion in Cartesian coordinates.](image)

**Design tools:** Since the EAGLE simulation kernel is implemented in MATLAB and Simulink, many tools can be employed for design. The Control System Toolbox™ (upon which EAGLE relies to represent linear systems) can be used to design control loops. The data objects within EAGLE can also be used with the Robust Control Toolbox™.

The current revision of EAGLE interfaces with the Worst-Case Analysis Tool (WCAT) from the University of Leicester. This tool can reduce the computational effort in determining the robustness to parameter variations. Robustness is normally determined through computationally intensive Monte Carlo methods.

**Graphical User interface:** Following the creation of a simulator using the Simulink interface, all further interaction with the simulator is achieved through the EAGLE GUI (see Fig. 7), which is a Java-based application built using the Eclipse software development environment. The user has full control over the parameters used in the simulator. The benefits of separating the GUI from the simulator are:

1. Non-users of MATLAB and Simulink can execute simulations and to analyse their results,
2. It ensures that the simulator structure is not changed accidentally,
3. The functions of the GUI and the simulator / simulator elements are separated which permits each to be maintained separately.

![Fig. 7: The EAGLE graphical user interface.](image)

**5. EAGLE WORK-FLOW**

EAGLE provides an iterative simulation workflow (see Fig. 8) that starts with the simulation needs of phase 0 and phases A studies. These studies provide data products for the simulation needs of subsequent system engineering phases; culminating in the validation of flight software.

To support the above workflow, EAGLE provides four macro-levels of simulation fidelity that are mapped to the following simulation classes:

1. System concepts simulator (SCS),
2. Mission performance simulator (MPS),
3. Non-real-time functional engineering simulator (NRT-FES),
4. Avionics test bed (ATB).

EAGLE provides an iterative simulation workflow (see Fig. 8) that starts with the simulation needs of phase 0 and phase A studies and provides data products for the simulation needs of subsequent SSEL phases, culminating in validated flight software.

To support the above workflow, EAGLE provides four macro-levels of simulation fidelity that are mapped to the following simulation classes:

1. System concepts simulator (SCS),
2. Mission performance simulator (MPS),
3. Non-real-time functional engineering simulator (NRT-FES),
4. Avionics test bed (ATB).

Fig. 8: The EAGLE simulation development workflow. These simulation fidelity levels are described below:

SCS: The SCS is built in a desktop computing environment. The purpose of the SCS is to determine the feasibility of the selected mission from the vehicle’s final trajectory correction manoeuvre to the lander touch-down point. A key feature of the SCS is that it operates much more quickly than wall-clock time. This is achieved using relatively simple spacecraft, environment, and algorithmic models (for example, point mass / single body representations of the spacecraft and a simple atmospheric model.) This simulation speed permits mission concepts to be evaluated, accepted / rejected, or modified quickly.

MPS: Like the SCS, the MPS is built on the desktop. The MCS is a refinement of the SCS and is used to elaborate the spacecraft and environmental models. Aspects of the MCS include full 6 DOF modelling of all the spacecraft bodies, a realistic atmospheric model, and fault injection capability. After the necessary refinements and elaborations have been performed, the MCS provides a platform for the definition of a GNC architecture. The MCS permits parameter trade-offs and robustness tests (through Monte Carlo methods) of the GNC concept to be performed.

NRT-FES: The NRT-FES represents another increment in simulation fidelity and is derived from the MCS. It is used to focus on specific mission phases and events (such as parachute / airbag / landing leg deployments) and also to confirm GNC robustness in a full fidelity simulation. The NRT-FES may be bypassed, with the MPS being converted to the ATB directly.

ATB: The ATB is used to evaluate the GNC algorithms in a real-time environment that represents the spacecraft computing resources as closely as possible (LEON 3 processors.) The ATB is especially useful in determining if the GNC algorithms satisfy timing, memory, and fault recovery constraints. The fidelity level of the spacecraft (dynamics, sensors, actuators, etc.) and the environment is equivalent to that of the MPS. The spacecraft and environment models are converted to real-time code and compiled and linked to run on the dSpace platform. This processor-in-the-loop architecture as shown in Fig. 9 provides end-to-end simulation capability.

Fig. 9: The architecture of an EAGLE ATB simulator.

6. CONCLUSIONS AND FURTHER WORK
The EAGLE simulation framework has been presented. The need for such a framework and the intended audience has been described. The primary capabilities of EAGLE and its architecture were also presented. Finally, the workflow for maturing the GNC algorithms was presented.

EAGLE is being used (and improved) on several other projects, funded both internal to and external from Scisys. Recently completed work permits the modelling of the power generation (battery charging, solar-electric generation with eclipse effects) and usage of various subsystems. This permits the system engineer with another tool for requirements generation and compliance.
Work is also under way to extend the number of gravity models available in the EAGLE blockset. Recently, a panel method for high-fidelity representation of gravity fields of irregularly shaped bodies was developed at SciSys and this is being integrated with EAGLE currently. This method is useful for asteroid rendezvous and also for modelling local gravitational effects in the vicinity of vehicle landing sites.

7. ACKNOWLEDGEMENTS
SciSys is grateful for the support of ESTEC contract 21286/07/NL/EK under which EAGLE is being developed for ESA. The authors are also grateful for the help provided by Michelangelo Russo and Benoit Pigneur of SciSys during the preparation of this paper.

8. REFERENCES