**Interface Concept Template**

**1. Introduction**

* **Purpose**: This document defines and documents the interfaces within the system and between the system and external entities. It aims to ensure clear understanding, interoperability, integration, and effective communication among all interacting elements.
* **Scope**: This artifact covers all forms of interactions at shared boundaries, including physical, logical, electronic, environmental, and human-machine interfaces, involving system elements, subsystems, or external systems.

**2. Definition of Interfaces**

* **What is an Interface?**: An interface is a **shared boundary** where, or across which, two or more systems or system elements interact. It is viewed as a boundary, not a tangible physical "thing," and its fundamental aspects are functional, defined by inputs and outputs. Interactions can be direct (actual connection) or indirect (design features affecting another system) and involve the exchange of energy, matter, or information.
* **Types of Interfaces**: Interfaces can be categorized by their nature:
	+ **Functional Interfaces**: Define inputs and outputs of functions and how tasks are allocated.
	+ **Physical Interfaces**: Bind two system elements, such as mechanical (structure, loads), electrical (power, signals), fluid (thermal control), and radio frequency (RF) connections.
	+ **Informational Interfaces**: Handle the exchange of data, commands, and messages, including their format, identifiers, rates, and communication protocols (e.g., APIs).
	+ **Human-Machine Interfaces (HMI)**: Points where users, operators, or maintainers interact with the system.
	+ **Environmental Interfaces**: Where the system interacts with its physical surroundings.
	+ **Internal Interfaces**: Interactions occurring between system elements within the System of Interest (SoI) boundary.
	+ **External Interfaces**: Interactions between the SoI and external systems or the environment, which can include interfacing, interoperating, or enabling systems.

**3. Key Interfaces and Interactions**

* **Interface Identification**: This section identifies all relevant interface boundaries and interactions, ideally initiated early in the preliminary lifecycle concept definition phase. Techniques such as external interface diagrams, context diagrams, or N2 diagrams can be used.
* **Specification of Interactions**: For each identified interface, describe the specific exchanges that occur across the boundary, including:
	+ **Nature of Exchange**: What information, signals, materials, or commands are exchanged.
	+ **Conditions**: Under what conditions these exchanges take place.
	+ **Performance Characteristics**: Expected performance attributes (e.g., speed, accuracy, format).
	+ **Definition**: A common agreement concerning a specific interaction, which should be explicitly recorded (e.g., in an Interface Control Document or IDD). These definitions are statements of fact and evolve as the design matures.

**4. Interface Requirements**

* **Explicit Statements**: Provide explicit, testable statements defining the expected interactions across each interface boundary. These are functional/performance requirements.
* **Content**: For each requirement, specify:
	+ **Direction and Nature**: Input/output, supplier/receiver roles.
	+ **Object or Data**: The specific data, material, or energy being exchanged.
	+ **Performance/Quality**: Any quantitative or qualitative performance and quality characteristics.
* **References**: Include clear references (pointers) to where the detailed interface definition is documented (e.g., a specific section in an ICD). Interface requirements are defined and allocated throughout the system's design process.

**5. Interface Management and Control**

* **Process Overview**: Describe the approach to Interface Management (IM), which is a continuous set of activities throughout the system lifecycle. IM facilitates and manages the identification, definition, design, and control of interactions across system or system element boundaries.
* **Documentation and Baselines**:
	+ **Interface Control Documents (ICDs)**: The primary documents or repositories for recording agreed-upon information defining interactions across interface boundaries. Other terms include Interface Definition Document (IDD) or Interface Agreement Document (IAD).
	+ **Baselines**: Establish baselines for interface requirements, definitions, architecture, and design.
	+ **Control**: Outline procedures for managing changes to interfaces, ensuring compliance, and disseminating interface information to all relevant stakeholders. Model-Based Systems Engineering (MBSE) tools can significantly enhance IM by automating generation and consistency checks.

**6. Relationship to Other Concepts**

* **System Context and System of Interest (SoI)**: This artifact defines how the SoI interacts with its environment and external elements. The system context, depicted via Block Definition Diagrams (bdd) and Internal Block Diagrams (ibd), clarifies how structural elements interface and information flows.
* **Concept of Operations (ConOps) and Operational Concept (OpsCon)**: The ConOps, developed early, provides a high-level view of system usage, which helps identify external and internal interfaces. The OpsCon provides more detailed user-oriented usage descriptions. Human Systems Integration (HSI) considerations influence the ConOps and early design of human-system interactions.
* **System Architecture (Functional and Physical)**: The Interface Concept informs and is informed by the system's architecture. Functional architecture defines transformative processes and input-output tasks, while physical architecture arranges physical elements and their interfaces as a concrete solution.
* **Requirements**: Interface requirements are derived from lifecycle concepts, defining what the SoI needs to do and how well, including interactions with external systems.
* **Integration and Validation**: Well-defined interfaces are essential for combining system elements into a cohesive whole (integration) and for enabling systems to exchange and use information (interoperability). ICDs and approved interface changes serve as inputs for product verification and validation, particularly for test objectives and plans.

**7. Importance and Value**

* **Reduces Integration Risk**: Clear interface definitions prevent costly misunderstandings and integration failures later in the project lifecycle.
* **Enables Parallel Development**: Allows development teams to work on different components independently, confident that their interfaces will align during integration.
* **Supports Verification and Validation (V&V)**: Well-defined interfaces are critical for effectively testing and verifying that components and subsystems work together as intended.
* **Facilitates Change Management**: Changes to one component or system can be managed and communicated effectively if interface definitions are well documented and controlled.
* **Ensures Interoperability**: Critical for achieving the ability of two or more systems to exchange and use information effectively.

**8. Risks and Challenges**

* **Unidentified/Unmanaged Interfaces**: A significant project risk that can lead to incompatible system elements, system failure, or project cost/schedule overruns.
* **"To Be Determined" (TBDs) / "To Be Resolved" (TBRs)**: Effectively managing these placeholders in interface definitions is crucial to avoid costly changes and rework in later phases.
* **Supplier-Developed Interfaces**: Requires clear agreements and the use of common, configuration-managed definitions across all interacting systems, especially when components are outsourced.
* **Legacy System Integration**: Integrating with existing systems requires careful consideration of their predefined interfaces and operational needs.
* **Increasing Complexity**: The growing complexity of modern, software-intensive systems increases the number of internal and external interactions, making interface management more challenging and requiring semantic interoperability across modeling tools.

**9. Recommended Representations**

* **Narrative Text**: For providing context, rationale, and detailed descriptions of interfaces and their interactions.
* **Tables**: To summarize interface types, characteristics, requirements, and examples.
* **Diagrams**:
	+ **System Context Diagrams / External Interface Diagrams / Boundary Diagrams**: To visually represent the System of Interest (SoI) and its interactions with external systems.
	+ **Block Definition Diagrams (BDD) and Internal Block Diagrams (IBD)**: To illustrate how structural elements interface and how information flows between them.
	+ **N2 Diagrams (Coupling Matrices)**: For systematically analyzing and depicting inputs and outputs between system functions or physical elements.
	+ **Activity Diagrams / Sequence Diagrams**: To detail the time-ordered interactions and workflows involving specific interfaces.
	+ **Interface Control Document (ICD) / Interface Definition Document (IDD)**: Formal documents detailing the interface definitions.
	+ **Data Dictionary / Application Program Interface (API) specifications**: For defining data, commands, and messages exchanged at software interfaces.