Table of Contents

[1 Introduction 3](#_Toc452985453)

[1.1 Primary Goals for Interface Modeling 3](#_Toc452985454)

[2 Definitions 3](#_Toc452985455)

[2.1 Component 3](#_Toc452985456)

[2.2 Interface 3](#_Toc452985457)

[2.3 Interface-end 4](#_Toc452985458)

[2.4 Interface Layer 4](#_Toc452985459)

[2.5 Connection 4](#_Toc452985460)

[2.6 Interface Medium 5](#_Toc452985461)

[2.7 Port 5](#_Toc452985462)

[2.8 Interface constraints 5](#_Toc452985463)

[2.9 Interface-end compatibility 5](#_Toc452985464)

[3 Limitations of current SysML interface modeling (i.e. ports, connectors, item flows) 5](#_Toc452985465)

[4 Requirements from UML 4SE RFP 7](#_Toc452985466)

[5 SE Interface Needs 7](#_Toc452985467)

[5.1 Interface Specification, Interface Realization and Textual Requirements 7](#_Toc452985468)

[5.2 Interface Specification Capture 7](#_Toc452985469)

[5.3 Items Flowing or Exchanged 8](#_Toc452985470)

[5.4 Transfer Agreement 8](#_Toc452985471)

[5.5 Interface Decomposition 9](#_Toc452985472)

[5.5.1 Interface-end Level Decompositions 9](#_Toc452985473)

[5.5.2 Interface-End Layered Decomposition 9](#_Toc452985474)

[5.5.3 Flow Items and Flow Property Decomposition 9](#_Toc452985475)

[5.5.4 Connector Decomposition 10](#_Toc452985476)

[5.6 3D Geometrically Representation 11](#_Toc452985477)

[5.7 Interface Re-Use Libraries 11](#_Toc452985478)

[5.8 Views and Diagram Representation 11](#_Toc452985479)

[5.9 Interface Compatibility 12](#_Toc452985480)

[5.10 Free form Interface Tables 12](#_Toc452985481)

[6 Interface Examples 14](#_Toc452985482)

[6.1 Tractor/Implement Example 14](#_Toc452985483)

[6.2 PC Mouse Example 14](#_Toc452985484)

[6.3 Power Plug Example 14](#_Toc452985485)

[6.4 Simple Two Plate Example 15](#_Toc452985486)

[6.5 Door Hinge Example 15](#_Toc452985487)

[6.6 Layered Interface Example 15](#_Toc452985488)

[6.7 Hybrid SUV 15](#_Toc452985489)

[7 Questions 15](#_Toc452985490)

# Introduction

This document includes some considerations to be addressed when developing the SysML v2 requirements for modeling interfaces. The intent is not to provide a solution but to express the needs of the Systems Engineers and others users of SysML v2.

The solution may be provided completely by SysML or partially, in fact many of these capabilities already exist in SysML. Some of these capabilities may require the integration between SysML tools and other domain specific tools. However, SysML is intended to provide an integrated solution within the Systems development Environment.

## Primary Goals for Interface Modeling

1. The information captured in the model includes equivalent information that is generally contained in an interface specification document and interface design document (e.g. IRS, IDD, ICD, ...)
2. The interface concepts are consistent with the behavior, structure, and other concepts of the language
3. The concepts of interface specification and interface realization are distinct such that the model can clearly capture how interface specifications can be realized.
4. Ensure a consistent approach to model a diverse range of interfaces (e.g. electrical, mechanical, software, user IF), and include the ability to model Modelica-like physical interface concepts and flow based concepts
5. Ensure ability to support nested interfaces and reusable interfaces
6. Ensure the ability to readily model different interface viewpoints that address a broad range of interface concerns

# Definitions

## Component

(1) An entity with discrete structure, such as an assembly or software module, within a system considered at a particular level of analysis. (ISO/IEC 1998)

(2) One of the parts that make up a system. (IEEE 2008)

[3, SEBoK Glossary]

A component is the generic term for the level of decomposition at which system elements are fully specified, and for which design disciplines can implement them. [3, similar to SEBoK Glossary Discussion] Note that a component can correspond to any level of the system hierarchy.

## Interface

An interface is a shared boundary between two functional units (component), defined by various characteristics pertaining to the functions, physical signal exchanges, and other characteristics. (ISO/IEC 1993)

'An interface provides the means for systems and system elements to interact, which may include the exchange of information, material, forces, and energy. To specify an interface, one must specify the Interface-end specification on the components on either side of the interface, the items that are exchanged, the constraints and/or rules that govern the exchange, and the mechanism used to connect the interface-ends, i.e. the connection or interface medium. (A Representative Application of a Layered Interface Modeling Pattern”, Shames, Sarrel, Friedenthal

An interface definition sometimes refers to one side of an exchange, but more generally, and for the SE needs definition, an interface refers to both sides of the exchange and the connection mechanism between them.

## Interface-end

An interface-end defines one side of the realized interface for a single component (generally corresponds to a SysML Port). An Interface-end specification provides the specification that the interface-end must conform to.

An interface-end can be decomposed into interface layers.

## Interface Layer

An interface layer is an abstraction approach to help deal with interface complexity where each interface layer provides specific functionality associated with the interface. An interface-end will typically have at least two-layers, i.e. the application and physical layers, however there are often additional layers such as described in the ISO seven level communication specification and the four layers associated with the Internet Protocol Suite (TCP/IC). The number of layers included in the model is dependent on the specific concerns being addressed. Sometimes, only the application layer is modeled, and other times only the physical layer is modeled.

A Protocol Stack is a set of layers that transforms items to enable their exchange, such as for purposes of communication. A fundamental principle of an interface layer is that the layer below is independent of how the layer above is encoded. For example, an electrical signal of less than 0.5 volts may correspond to a software bit = 1 or a software bit = 0 at the next higher layer, and this software bit may correspond to a switch being in the on position or the switch being in the off position at the next higher layer. One cannot tell from looking at the electrical signal whether the switch position is on or off or whether the bit is 0 or 1. This information is specified by the transformation between layers.

## Connection

1. A connection can correspond to a logical or physical connection.
2. A physical connection is a connection between two interface-ends that join directly together without an interface medium between them such as the connection between a plug and receptacle, or the connection between two other mechanically mated parts.
   1. Other synonyms Junction, Joint, Seam, connection point, union, matching faces, interconnect, fit-together
   2. A connection may still impose constraints on the interface that are not imposed by either interface-end, such as the maximum electrical resistance between the mating contacts between a plug and receptacle.
3. A logical connection is a connector between interface-ends that abstracts away the interface medium.

## Interface Medium

A component used to connect 2 or more interface-ends such as a pipe, harness, bus, or network.. As noted above, a logical connection abstracts away the interface medium by showing the connector and the 2 interface-ends without presenting the interface medium.

1. The logical connection can be elaborated through decomposition or allocation to model the interface medium.
2. This interface medium may adapt, transform and/or constrain the exchanged items from one Interface-end to the other.
3. The Interface-ends on either side of the interface medium are not required to be directly compatible but each end of the interface medium must be compatible with the corresponding component interface-ends. For example, a pipe can have a physical connection on one end that is entirely different than the physical connection on the other end.

## Port

1. Port is not used in this document since it is an overloaded term. The term Interface-end is used in its place.

## Interface constraints

1. Interface constraints can be applied to any aspect of the including interface-ends, the items exchanged and the connection. These constraints can include behavioral constraints such in a protocol, limits on items exchanged, Modelica constraints as another kind of behavioral constraint, etc.

## Interface-end compatibility

1. Interface means conformance to interface constraints

# Limitations of current SysML interface modeling (i.e. ports, connectors, item flows)

1. The inability to easily create different abstraction levels and views of the interface.
   1. An example is when two nested parts are connected via a connector, the model should derive the connector between the higher level parts without showing the nested parts.
   2. Another example is the ability to easily show nested connections and flows between parts by double clicking on the composite connector or composite port.
2. The ability to capture mechanical/geometric interconnections using an IBD is difficult and the result is not very intuitive.
   1. Maybe another diagram type, i.e. Mechanical, like electrical schematics and mechanical representations that reference defined blocks, parts and Interface-ends and connectors
   2. Have ability to show geometric shapes with respect to defined coordinate systems, and projections (exploded views) with dotted lines showing mating points.
   3. Include the ability to rotate 3D objects so as to adjust the view from all perspectives.
   4. Examples:
      1. Show a bolt passing through a hole in another part.
      2. A mechanical part connecting to another mechanical part via a flange, mechanical interlock, bonded interface, etc.
3. Limited ability to specify and enforce compatibility constraints between interface-ends.
4. Limited libraries containing common and industry standard interfaces are required.
   1. The intent is to minimize the amount of effort to add an interface and to manage consistency across the industry in such interfaces.
5. In constructing an interface (Interface-ends, connectors, item flows), too much redundant information needs to be entered. Much of this information can be derived from existing content and actions. Examples include:
   1. When connecting to ports on internal parts, a proxy port on the owning block representing port of the internal part should be derived vs. creating a new elements and the connector
   2. Most Item flows can be derived directly from flow properties already defined within the Interface-end
   3. An Interface-end can be derived when a connector connects the Interface-end to another block
6. Full Port and Part over lapping semantics
   1. A Full Port is intended to represent a part located on a boundary, therefore a Full Port should have the same semantics as a part that has been moved to a boundary.
   2. Often both the part and the full port need to be represented in different views. This requires both constructs to be created and maintained. A part is created when a composition relationship is created but not a full-port.
   3. A part is considered to be a part when it is viewed within the boundaries of its parent. A part becomes a full-port when it is dragged to the boundary of the parent. The concrete syntax may be enhanced to describe what ports of the part are viewed as being internal and external to the host, but the syntax should be the same as a part.
7. Tabular views of the interface
   1. Tables are often a simple and intuitive means for entering and viewing interface related information
   2. An example is specifying the layers of an interface and their transformations, such as a table that captures the logical interface, the application layer (software/message), the electrical signals, and the physical connector.
   3. The content of the cells is distributed across different parts of the model
8. Distinguish the difference between a connector that represents a logical and physical connection. Note: A direct point of interface implies the Interface-ends must be compatible whereas the other may not.
9. Internal Connectors – There is a need to be able to show a connector that connects two external ports of a component within the component.
10. Lack of clear distinction between interface specification and interface realization. The closest concept to this in SysML is the proxy port can be thought of as an interface specification, and the full port can be thought of as an interface realization.
11. Execution semantics - There are many aspects of execution semantics that are not intuitive such as the inability to distinguish 2 different proxy ports on a component of the same type. Another fundamental limitation is the inability to represent the flow of pins and streaming outputs. Most execution is currently limited to the exchange of signals.

# Requirements from UML 4SE RFP

1. UML for SE shall provide the capability to model the part of a system that provides access between the system’s behaviors and properties, and its environment.
2. UML for SE shall provide the capability to model the boundary of a system, which includes the set of all ports (interfaces) that connect the system to its environment.
3. UML for SE shall provide the capability to model the logical and physical connections between ports, and the associated interface information.
   1. Note 1: A default capability should enable connections to be shown without designating the ports. In this case, the ports are implicit.
   2. Note 2: The primary function of some “connecting” components is to connect the outputs from one system to the inputs of another system via its ports. These components may include wires, networks, mechanical couplers, etc. In some cases these components are abstracted from the model to avoid detail.

# SE Interface Needs

## Interface Specification, Interface Realization and Textual Requirements

1. Have the ability to show how the interface specification refines textual requirements and how the interface realization satisfies the textual requirements.

## Interface Specification Capture

1. For each interface capture the information associated with the interface specification including:
   1. The specification for each interface-end on the component
   2. The specification for each connector
   3. The specifications for the being exchanged items across the interface
   4. The specification for the interface medium (optional)
   5. The constraints associated with the above including protocol constraints, Modelica like constraints, physical connection constraints, and message constraints.
2. For each interface capture the realization for the interface specification including:
   1. The interface-ends
   2. One or more connectors that connect the interface-ends
   3. The items being exchanged between the interface-ends
   4. The interface medium that enables the exchange of the exchanged items
   5. The way in which the constraints are satisfied (e.g., interface-end state machine that conform to a protocol constraint)
3. Maintain consistency between each abstraction level of the interfaces
4. Specify properties that characterize the interface.
   1. Properties can be added to the interface-ends or the connectors.
   2. These properties and constraints may need to be added to characterize how they impact the transfer. Examples include:
      1. The resistance, inductance and capacitance (impedance) of an electrical wire, such as a coax cable or twisted pair.
      2. Propagation delays
      3. Error rates
   3. Additional implementation constraints may apply to various types of properties such as structure, data, thermal, loads, geometric, etc.
   4. If a specification is referenced, these properties may specify what variants of the specification are used
      1. E.g. Ethernet 802.3 100 BASE-T – The cable may be specified as [CAT5](https://en.wikipedia.org/wiki/Category_5_cable) copper cabling with two twisted pairs
      2. Liquid transfer– The medium may be specified as ½ copper tubing
      3. Electrical power transfer may require a specific wire size and type.
   5. If not, these properties establish just the limits and constraints of the intended realization of a new design and not the design detail selections.
5. Indicate which constraints apply to specifying the compatibility constraints between interface-ends and have a mechanism to indicate when these compatibility constraints are not being satisfied.

## Items Flowing or Exchanged

1. Define each type of item flowing or exchanged through the interface (categories of types include messages, data, energy, mass and force).
2. Define the direction of each item flowing, i.e. in, out or bi-directional
3. Define associated properties and constraints with each item.
   1. For example, definition, frequency, voltage level, message size, performance values, priority, amount of energy or mass, etc.
   2. These items are usually domain specific and multiple domains may need to control the values of these properties
4. Each entity flowing may be a structure of entities and therefore can be decomposed into multiple sub-entities
   1. For example a data message may be composed of multiple fields or files.
5. Select a system view that tracks the flow of selected items through a set of interfaces
6. The item may be represented differently as it is transposed through interface layers

## Transfer Agreement

1. Specify the rules that govern the exchange of items flowing (e.g., protocol)
2. For each interface-end specify one or more transfer agreement (i.e. protocol) that will be used to interact across the interface to manage the transfer of the data, power or mass from one interface-end to another. These transform agreements can include behavior and properties.
   1. Examples - Ethernet, USB, AC/DC, voltage rise times, propagation delay, maximum pressure, range of voltage levels, etc.).
3. If Interface layers are define, a transfer agreement may also exists between peer interface-end layers.
4. Specify transfer properties that may limit or constrain the transfer.
   1. The agreement may be specified by referencing an external specification.
   2. If a specification is referenced, these transfer properties would specify the variants available in the specification.
   3. For example Ethernet provides the protocol that will be used and also permits multiple transfer speeds. So a transfer property may specify the speed variant such as 10BASE-T or 1000BaseT.

## Interface Decomposition

The interface-ends, the connector and the items flowing may each be decomposed.

### Interface-end Level Decompositions

1. Show how an interface-end (port) on the higher level is decomposed into a series of interface-ends on the next lower level. This type of decomposition would result in multiple connectors on the lower layer.
   1. For example: A power plug and receptacle interface may appear at a higher level as a single port but at a lower level be shown as three pins, hot, neutral and ground.
2. Specify the physical interface layer of an interface-end defines the physical connection characteristics of the interface-end.
   1. Generally is the lowest layer
   2. This may be represented as a property stating a reference to an external specification.

### Interface-End Layered Decomposition

1. An interface-end may consist of multiple interface layers, referred to as stacks.
2. Show the sending interface-end pushes the flowing item down the stack its interface layers and the receiving interface-end pushes the item up the stack.
3. Show how each layer performs a particular type of transformation on a flow or exchange.
4. Show how each intermediate layer interacts with a higher and a lower interface layer
5. Show how an interface layer may also interact with a peer layer of a connected interface-end
6. There generally is at least an application and physical layer.
   1. Show how the sending physical layer performs the final transformation (if any) and provides the physical connection to the peer physical layer(s) of the connected interface-end(s).

### Flow Items and Flow Property Decomposition

1. Show Flow items on the Connector.
2. Show Flow Properties as a property in an interface-end.
3. Show how these items that flow may be recursively decomposed to represent more details at lower abstraction levels.
   1. A single flow item on a higher level could represent an aggregate flow into lower level flows.
   2. A single flow item could be decomposed adding more structural details to each flow, e.g. a message being decomposed into multiple fields.
   3. For Example: The surface where a wheel is bolted to wheel hub, a bottle top that snaps or screws onto a bottle, the point at which an airplane wing is attached to the fuselage.

### Connector Decomposition

1. Categorize Interface Flow Decompositions as Connection Decomposition or Interface Medium Decomposition
2. Internal Connector – An internal connector connects two ports of a component (part) internal to the component.
   1. This internal connector may be a connection or and interface medium. If there is only a need to specify constraints than it can be a connection.
   2. However, it typically alters or transform the input flowing item before presenting it as an item flowing out of the component and therefore in this case be shown as an interface medium.

#### Connection Decomposition

1. At any level of decomposition a connector indicates the two interface-ends are joined directly together and an interface medium is not available the next lower level.
   1. This implies the interface-ends on this connection must be compatible at all layers.
   2. At lower levels of decomposition the connection may be refined by adding additional properties and constraints but the connection continues to be a direct connection.
   3. Examples of Connections: The surface where a wheel is bolted to wheel hub, a bottle top that snaps or screws onto a bottle, the point at which an airplane wing is attached to the fuselage.

#### Interface Medium Decomposition

1. Define an interface medium that captures the characteristics of the connection between two interface-ends.
2. At the highest level of decomposition show the interface medium as a single line, i.e. a connector, between two interface-ends.
   1. Depending on the maturity of development, lower level interfaces may not yet be specified. Therefore, in these early development stages the higher level interfaces can specify a very abstracted interface.
3. At the next lower level of decomposition, provide the ability to replace the connector with another entity.
   1. This entity may alter, i.e. transform, and constrain the transferred items.
   2. The entity provides additional interfaces between this new entity and the original interface-ends.
   3. These new interfaces at each new interfacing point could then be decomposed as a new entity at the next lower level.
   4. Example 1, see the three decomposition levels of the Tractor/Implement Example in the SECM model. The tractor interface to an Implement (attachment) may initially be shown as a single connection (Level 1) between the tractor and the Implement. As the development matures the interface may be decomposed into a mechanical connection interface (3 point hitch), a mechanical power transfer interface and a hydraulic interface (level 2), Level 3 provides more detail of each of the three connections.
   5. Example 2 – See the three decomposition levels of the Power Plug Example in the SECM model.
   6. Example 3 - Interface medium could be a bus structure, i.e. where multiple interface-ends connect to a single connector.

## 3D Geometrically Representation

1. Show a new SysML diagram that reveals the mechanical/physical geometrical representation of the specified interfaces and their host elements.
   1. This diagram has the ability to show shapes, **geometry** and projections (exploded views) of model elements defined in the SysML model.
      1. Mechanical entities shown are a geometrical representation of defined model elements.
      2. Capable of showing element names, such a parts and port names, as they are defined within the SysML model.
   2. This three dimensional view can be rotated in all three dimensions to allow viewing of all sides.
   3. Need a capability to examine adjustable Cross Section views, both horizontal and vertical from all three dimensions to view within objects
2. Discussion Points
   1. The intent of the geometric diagrams is to allow SEs to reveal early conceptual physical views of the proposed product. The intent is not to replace the detailed design captured in a mechanical CAD tools but to provide a means of communicating these ideas both to all the stakeholders, especially the customers, and to the detailed design teams.
   2. This is especially useful in the early stages of development.
   3. These mechanical/physical diagrams are much like the free-form drawings we have always included in all our traditional specifications. The biggest difference is that the elements in these diagrams are linked to elements defined in the SysML models, so if some aspect of an element is changed it is reflected correctly in this new diagram and other existing SysML diagram.

## Interface Re-Use Libraries

1. Reusable partially pre-specified standard interfaces and Interface-ends need to be made available to minimize the amount of effort to specify a new interfaces.
2. These standard interfaces need to include fixed and variable properties with defaults assigned.
   1. One of these properties could include diagram icons where appropriate, include performance parameters, etc.
3. Examples include a set of Ethernet 802.3 interfaces, USB standards, wall sockets and plugs, 3 Point Hitch,

## Views and Diagram Representation

1. A view can consist of multiple entities with different decomposition levels for each entity.
2. Show icons for specific port types, i.e. logical, Ethernet, USB, 3 Point Hitch, etc.
3. Show the specified direction of a payload.
4. A system wide interface view may consist of selected interfaces and selected items flowing across these interfaces. The intent is to be able to track select items flowing through the system as it transverses multiple interfaces and are transposed by various entities of the system.
5. Provide interface views that allow the ability to revert back and forth between the higher and lower interface levels.
   1. For example, a particular view may require the Tractor/Implement example interface to be shown very abstractly thus only showing the single connection interface level. Lower levels would show each of the three arms, the PTO and hydraulic interfaces.

## Interface Compatibility

1. When interface-ends of two entities are connected together via a connector, compatibility checks are needed to ensure the payload, physical characteristics, protocol, and medium are compliant to their constraints.
2. Provide the ability to specify compatibility checks between two connected interface-ends. This includes:
   1. Providing the ability to specify compatibility of properties within the two entities that are interacting through the interfaces.
   2. Providing interface compatibility such as constraints on the features of the interfaces, e.g. physical connector compatibility,

## Free form Interface Tables

1. Tables provide an easy and intuitive means of viewing and entering information.
2. Simple free form interface tables have been used extensively in traditional document based specifications to capture the values for property values and flow items.
3. Tables may also be used to capture an array of acceptable property values.
4. Typically interface tables have the flowing characteristics
   1. The top row provides the column titles and usually includes the units for the column.
   2. Column may identify and interface layer.
   3. Each row represents a different state of the interface.
   4. The cells of columns that represent interface layers **display the values as they are transformed from one interface layer to another**.
   5. Cell values can be a string value or a numerical value. Numerical values may be a nominal value, a value range and can include a tolerance, accuracy, permissible error, uncertainty or a distribution characteristic.
5. Example: See PC Mouse Example in SECM, Momentary push button switch and Scroll Wheel
   1. The tables below provide examples of interface layer transformations.
   2. The two table capture the interface definition resulting from a contact closure of a momentary push button switch and a scroll wheel being moved as they transposed from an abstract application layer to an electrical representation layer.
      1. The rows represent the states of the switch (idle/Pushed and ) and Positive and Negative increments)
      2. The columns represent the definition for each interface layer from the logical application layer to the electrical signal layer.

|  |  |  |  |
| --- | --- | --- | --- |
| **Button Position** | **Contact** | **Binary Value** | **Electrical Value** |
| Idle | Open | 1 | 3.5 volts |
| Pushed | Closed | 0 | 0 volts |

*Table 1- Push Button Layer Transformation Table*

|  |  |  |
| --- | --- | --- |
| **Scroll Wheel Movement Description** | **Binary Value** | **Electrical Value (volts)** |
| Positive Increment | 1 | 3.5 |
| Negative Increment | 0 | 0 |

*Table 2= Scroll Wheel Movement Signal Description*

* 1. The table below provides examples of interface layer transformations and valid data values.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Message Name**  **(Block Name)** | **Direction (To or From Mouse)** | **Total Bytes in Message** | **Information** | **Valid Binary Data Range** | **Bit Assignments** | | | **Voltage Levels**  (See Electrical Signal Values Table) | |
| **Number of bits** | **Byte Offset** | **Bit Offset** | **Binary Value 0** | **Binary Value 1** |
| Position Message | From | 4 | Message ID | 1 | 8 | 0 | 0 | VOL | VOH |
| X Position Change | 0 to 255 | 8 | 1 | 0 |
| Y Position Change | 0 to 255 | 8 | 2 | 0 |
| Reserved | 0 | 8 | 3 | 0 |
| Button Message | From | 2 | Message ID | 2 | 8 | 0 | 0 |
| Button 1 | 0 or 1 | 1 | 1 | 0 |
| Button 3 | 0 or 1 | 1 | 1 | 1 |
| Button 3 | 0 or 1 | 1 | 1 | 2 |
| Reserved | 0 | 5 | 1 | 3 |
| Scroll Wheel Message | From | 2 | Message ID | 2 | 8 | 0 | 0 |
| Scroll Wheels Increments | 0 to 255 | 8 | 1 | 0 |
| Get Report Message | To | 2 | Message ID | 4 | 8 | 0 | 0 |
| Data Requested | 0 to 255 | 8 | 1 | 0 |
| Set Report Message | To | 4 | Message ID | 5 | 8 | 0 | 0 |
| Set Data A | 0 to 255 | 8 | 1 | 0 |
| Set Data B | 0 to 255 | 8 | 2 | 0 |
| Set Data C | 0 to 255 | 8 | 3 | 0 |

* 1. The Electrical Signal Values table below provides the specifications for input and output electrical signals after they are transformed form a binary representation.

Table 1- Electrical Signal Values

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Binary Value** | **Symbol** | **Parameter** | **Conditions** | **Min** | **Typ.** | **Max** | **Units** |
| 0 | VIL | Input Low Voltage | CMOS |  |  | 0.9 | V |
| 1 | VIH | Input High Voltage | CMOS | 2.3 |  |  | V |
| 0 | VOL | Output Low Voltage |  |  |  | 0.4 | V |
| 1 | VOH | Output High Voltage |  | 2.4 |  |  | V |

# Interface Examples

The following examples are captured in the SECM. All the examples will be used to demonstrate some modeling difficulties with the existing SysML spec. They are not just limited to interface modeling but will also be used to demonstrate other concepts.

## Tractor/Implement Example

See the Tractor/Implement Example contained in the SECM model.

Specifies the interfaces used between a farm tractor and an implement. This includes the 3-point hitch, a PTO (Power Take-Off) and a hydraulic connection. Illustrates interface decomposition and some of difficulties with specify mechanical type interfaces.

A three interfaces can be mostly specified by referencing industry standards.

Referenced specification include ISO 730-1:2009 Agricultural wheeled tractors -- Rear-mounted three-point linkage -- Categories 1N, 1, 2N, 2, 3N, 3, 4N and 4.

## PC Mouse Example

See the PC Mouse Example contained in the SECM model.

It illustrates interface decomposition, interface layering and some of difficulties with specifying mechanical type interfaces. It also includes a USB Communication Interface.

USB, short for Universal Serial Bus, is an [industry standard](https://en.wikipedia.org/wiki/Technical_standard) developed in the mid-1990s that defines the cables, connectors and [communications protocols](https://en.wikipedia.org/wiki/Communications_protocol) used in a [bus](https://en.wikipedia.org/wiki/Bus_%28computing%29) for connection, communication, and power supply between [computers](https://en.wikipedia.org/wiki/Computer) and electronic devices.[[2]](https://en.wikipedia.org/wiki/USB#cite_note-2) It is currently developed by the [USB Implementers Forum](https://en.wikipedia.org/wiki/USB_Implementers_Forum). Referenced specification can be found at [Universal Serial Bus Revision 3.1 Specification](http://www.usb.org/developers/docs/usb_31_010516.zip).

## Power Plug Example

See the Power Plug Example contained in the SECM model.

This model provides the levels of abstraction and the use of ports to provide the interface-end specification. It illustrates interface decomposition, and some of difficulties with specifying mechanical type interfaces.

## Simple Two Plate Example

## Door Hinge Example

This picture capture on a diagram shows a mechanical drawing of a hinge and how a language used in mechanical engineering, Decomposed Connectivity Network, also captures a perspective hinge. SysML was not used.

## Layered Interface Example

## Hybrid SUV

A future effort after the above simpler efforts are complete.

# Questions

1. When you’re specifying a physical interface how is conjugate interrupted?