



# System Architecture

Lecture 06, v02

John G. Artus

BSEE

MSSE

INCOSE ESEP

# About This Courseware



- The majority of the material presented in this course is sourced from the textbook “System Architecture” by Edward Crawley, Bruce Cameron, and Daniel Selva
- I, John Artus, make no claim of ownership of the material sourced from this textbook
- I, John Artus, am using the material sourced from this textbook, and other indicated sources, as content for this courseware for educational purposes only
- This courseware lecture material has been sourced, interpreted, assembled, formatted, and copyrighted by John G. Artus for use in this educational context
- Anyone may freely access, and reuse this material in an educational context provided the copyright owner, John G. Artus, is recognized as the interpreter, assembler, and formatter of the source material used in the generation of this courseware, and provided that Edward Crawley, Bruce Cameron, and Daniel Selva are recognized as authors of the textbook “System Architecture” from which the majority of the content of this courseware has been sourced

- Form is a system attribute consisting of elements and structure
- Function is a system attribute consisting of entities of function and interactions through operands, collectively forming the functional architecture
- Having learned the roles of form and function, we are now poised to understand the synthesis of form and function known as architecture
- Architecture is not an independent attribute, but the mapping between form and function

## System architecture is

- The embodiment of concept
- The allocation of physical/informational function to the elements of form
- And the definition of relationships among the elements and with the surrounding context

# Principle of Value and Architecture

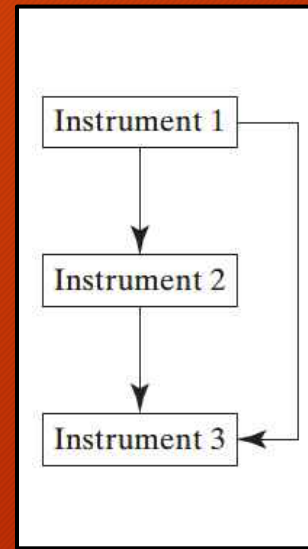
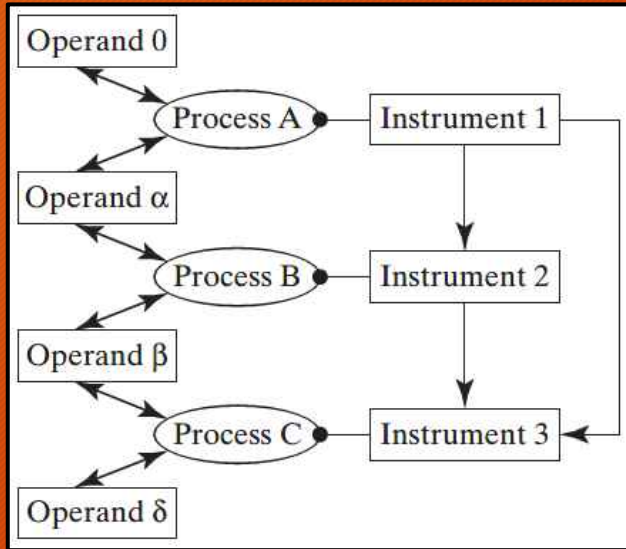
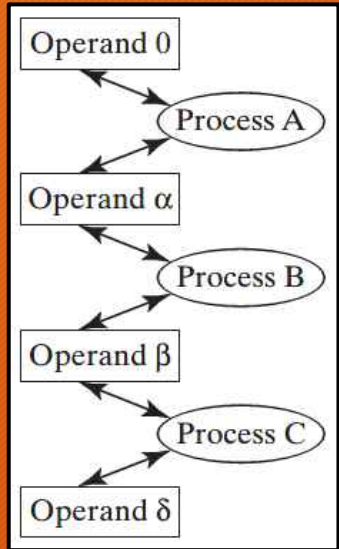


- If we are to deliver value with the system we build, it must have good architecture
  - Value is benefit at cost
  - Architecture is function enabled by form
  - There is a very close relationship between these two statements, because benefit is delivered by function, and form is associated with cost
- Therefore, developing good architecture (desired function for minimal form) will be nearly synonymous with the delivery of value (benefit at minimal cost)
  - Benefit derives from the emergence of primary and secondary value-related function
  - An axiom of lean manufacturing is that parts (form) attract cost

**The goal of the architect is to ensure emergence of value-related functionality at minimum cost**

# Form and Function Combined

- System architecture is the combination of the functional architecture and the elements and structure of form
- The center diagram below represents the simplest, and most commonly assumed architecture of a system
  - A simple flow-through functional architecture with distinct operands created at each stage
  - It further assumes that there is only one instrument linked to each internal process
  - And only one internal process linked to each instrument
  - This is a system that obeys the independence axiom
- Typically, system architectures are not as direct and simple as this depiction



The Independence Axiom states that the independence of Functional Requirements (FRs) must always be maintained

FRs are defined as the minimum number of independent functional requirements that the design must satisfy

When there are two or more FRs, the design solution must be such that each one of the FRs can be satisfied without affecting the other FRs

Functional Architecture

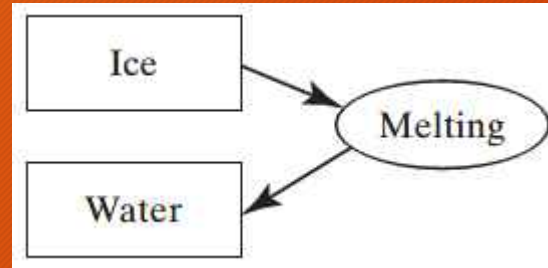
System Architecture

Formal Structure

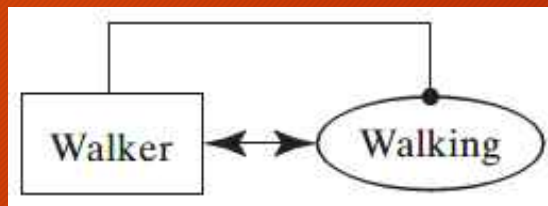
# Mapping of Form and Function - Examples

- No instrument

- The following view of the architecture of ice melting illustrates a case where no instrument object is shown



- This is because, in this case, the environment (not “part” of the system) is the acting instrument of the melting process
- Combined operand and instrument object
  - The following view of the architecture of a person walking illustrates a case of reflexive behavior in which the object on which the process is acting is also performing the function

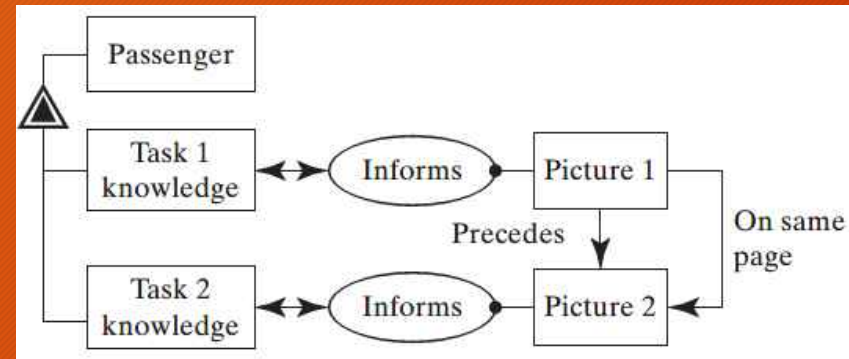


# Mapping of Form and Function - Examples (continued)

- One-to-one mapping affecting the same operand
  - The following view of the architecture of aircraft emergency instructions informing a passenger illustrates a case of one-to-one mapping affecting the same operand (passenger)



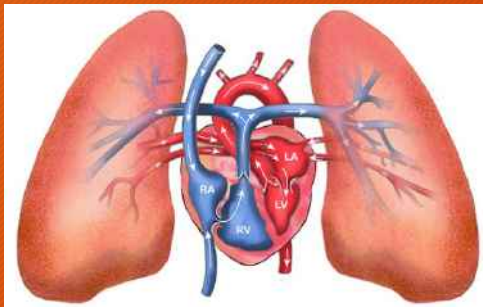
<http://avm104group3.blogspot.com/2015/05/passenger.html>



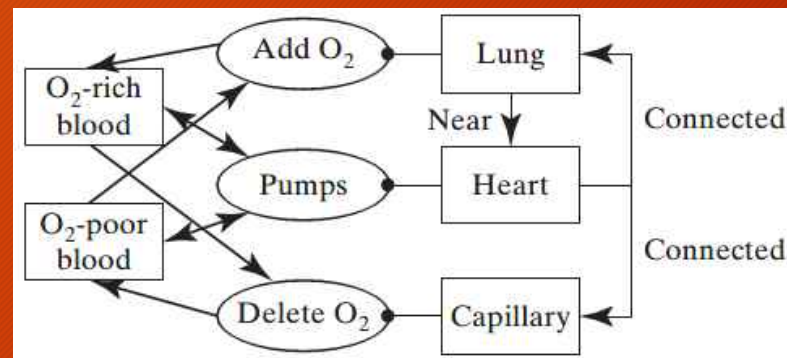
Here, the form-to-process mapping is the simple one-to-one case

However, the operands are not independent but, rather, are attributes of the same operand, a source of extra coupling

- One-to-one mapping with multiple operands
  - The following view of the architecture of the circulatory system in which the multiple operands are affected by multiple processes



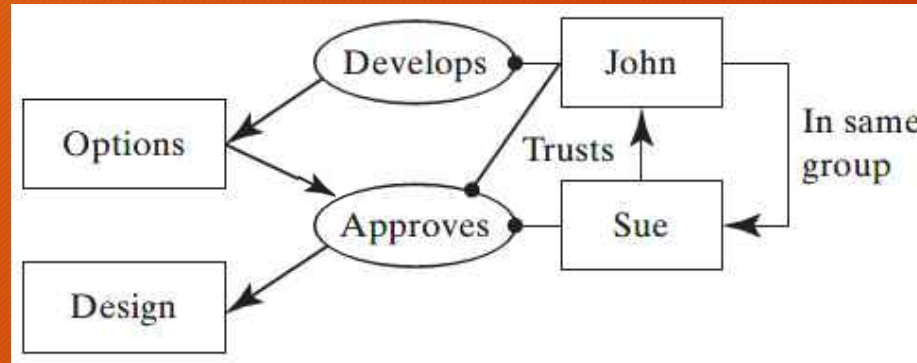
<https://mammothmemory.net/biology/organs-and-systems/the-pulmonary-system/thorax.html>



From these examples, we see that even in systems with one-to-one mapping of instrument to process, complexity can grow on the process-operand side

# Mapping of Form and Function - Examples (continued)

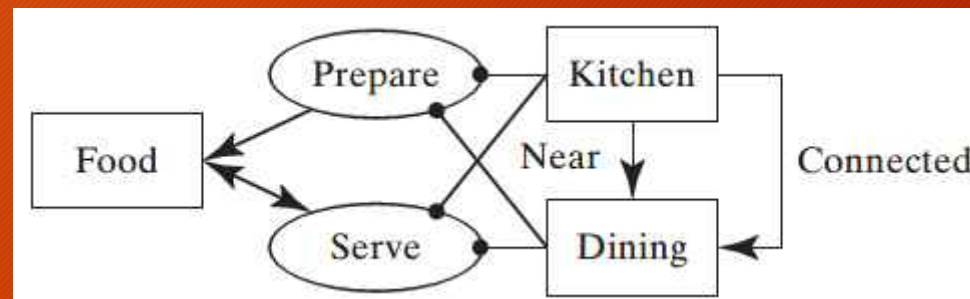
- One-to-many mapping of form to processes
  - The following view of the architecture of a design team illustrates a case of one-to-many mapping in which a single instrument (John) carries more than one process



Here, John is an instrument of two processes

One operand (Options) is subsequently used as an instrument

- Many-to-many mapping of form to process
  - The following view is of the architecture of a food preparation and delivery system in which the multiple instruments are carrying multiple processes



For these kinds of many-to-many mappings, the architect has to really focus on understanding all of the roles of each of the instruments

# Process-to-Form Mapping (Allocation)



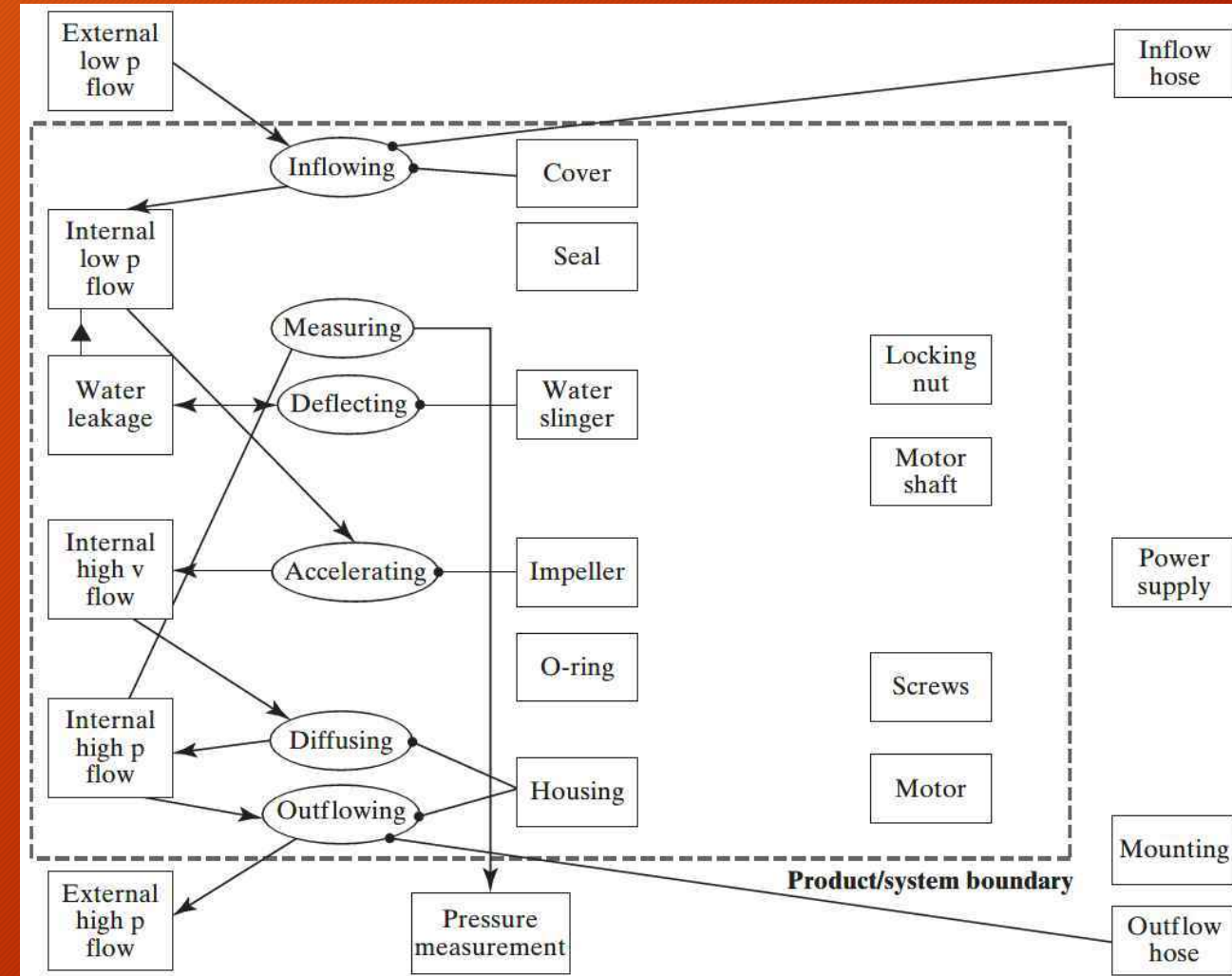
The following procedure describes how to map internal processes to instrument objects

1. Identify all of the important elements of form
2. Identify all of the value-related internal processes and operands
3. Ask what element of form is needed to execute each process
  - i. It may be among the identified elements of form, or it may lead to identifying a new element of form
4. Map (allocate) the internal processes to the elements of form
5. Identify any processes that remain unallocated to elements of form
  - i. Try to reason about what element of form is needed to carry the process, and add it if warranted
  - ii. Or else, reason about the value of the process - is it really needed?
6. Identify any elements of form that have no processes allocated to it
  - i. Try to reason about what process that element of form might carry, and add it if warranted
  - ii. Or else, reason about the value of the element of form - is it really needed?

# Use of Process-to-Form Mapping on Pump Example

- Go process-by-process and identify the link between the internal processes and instrument objects
- Primary value pathway
  - The impeller is the instrument of accelerating the flow
  - The housing is an instrument of diffusing as well as outflowing
- Secondary value delivery functions
  - The water slinger is the instrument of deflecting the flow
  - Originally, there was no instrument of form for the measuring function, so add a sensor
- The mapping of form to process is nearly one-to-one, but the housing carries two processes
- To obtain the desired emergence, the design of this part must skillfully blend these two functions
- We still have many elements of form unassigned to processes

Pump System Architecture with  
Primary and Secondary Value-Related Functions



- The externally delivered function emerges as form is assembled, and this emergence is enabled and informed by the structure of the form
- Recall that structure — the formal relationships among the elements of form — can be of three types
  - Connections indicate how the elements of form are linked or interconnected
  - Location and placement, including
    - Spatial/topological relations
    - Address and sequence, indicate where elements are located
  - Intangible relationships that simply exist, including
    - Membership
    - Ownership
    - Human relationships
- In general, functional interaction takes place because there is a connection of form
- But, some functional interactions can take place in the absence of any connection whatsoever
  - Gravitational and electromagnetic interactions
  - Some types of “ballistic” interactions among particles and fields
- In general, location and intangible relationships do not directly enable interactions and emergence of function
  - Rather, they inform and influence the nature of the interaction or the degree of performance

# Structure of Form Enables and Informs Functional Interactions (continued)



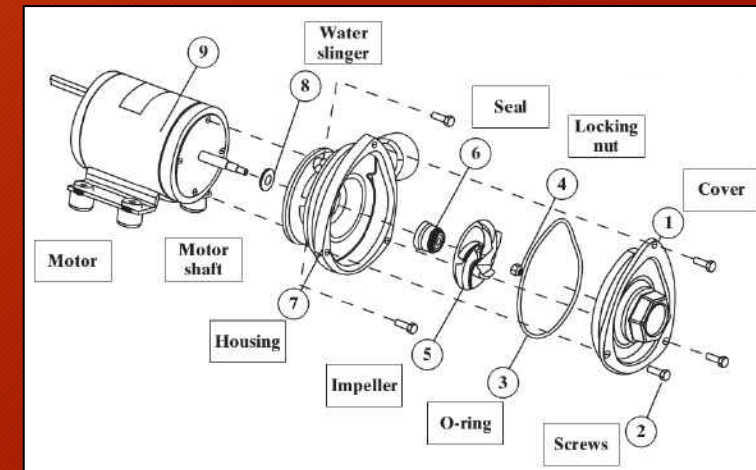
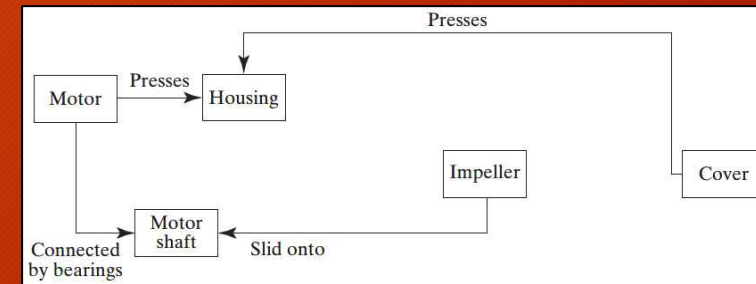
- In the circulatory system, the connections enable the functional interaction: the movement of blood
  - The fact that the lung and heart are nearby informs performance but is not essential to function
  - It is easier to pump large quantities of blood because the heart and lungs are close together
- The passenger safety information card is an information system, where the important feature of structure is sequence
  - The sequence of pictures is implicitly the sequence of instructions
  - If the pictures on the instruction card were reversed in order, the instructions would be invalid or at least confusing, so sequence enables emergence
  - The other spatial/topological relationship — that the individual pictures are on the same page — is a convenience
  - It may improve the reliability of the system (an “-ility”) or how fast a passenger can execute the instructions (performance), but it is not essential to emergence of function
- In Team X the fact that Sue trusts John (an intangible human relationship) probably improves performance
- Both the OPM and matrix representations have a way of showing which element of form is an instrument of which process
  - But neither has such an explicit way of showing which formal structural relationship enables which functional interaction
- This is left to the architect to reason about and document by other means

- The following procedure helps determine how the formal structure supports functional interaction and how it influences emergence
  1. Identify which among the elements of form are likely to be important structural relationships
  2. Identify the functional interactions among the related internal processes
  3. Ask what formal structure is necessary to enable each of the functional interactions
  4. Ask what formal structures would inform the other emergent properties of performance and the “ilities”
  5. Add, remove, or modify the structural relationships as necessary to support important functional interactions

# Formal Structure of the Centrifugal Pump Example

- How well do the structure and the functional interactions match up for the centrifugal pump?
- The important functional interactions for the pump are
  - The internal low-pressure flow is passed from the inflowing process
    - Whose instrument is the cover
  - To the accelerating process
    - Whose instrument is the impeller
- A second important interaction is
  - The internal high-velocity flow is passed from the accelerating process
    - Whose instrument is the impeller
  - To the diffusing process
    - Whose instrument is the housing
- This suggests that there should be some sort of connectivity formal structure that enables these important functional interactions
  - Between the cover and the impeller
  - Between the impeller and the housing,
- Examining the current connectivity structure diagram for the pump (upper right), there are no such connections

Simplified Pump Connectivity Structure



# Formal Structure of the Centrifugal Pump Example (continued)

- The connections in the pump support the interaction among elements of the mechanical parts, not the fluid flow
- The combined spatial/topological and connectivity structural relationships show that the only formal structure that exists between the impeller and the cover is a spatial relationship

Simplified Combined Spatial/Topological and Connectivity Structure

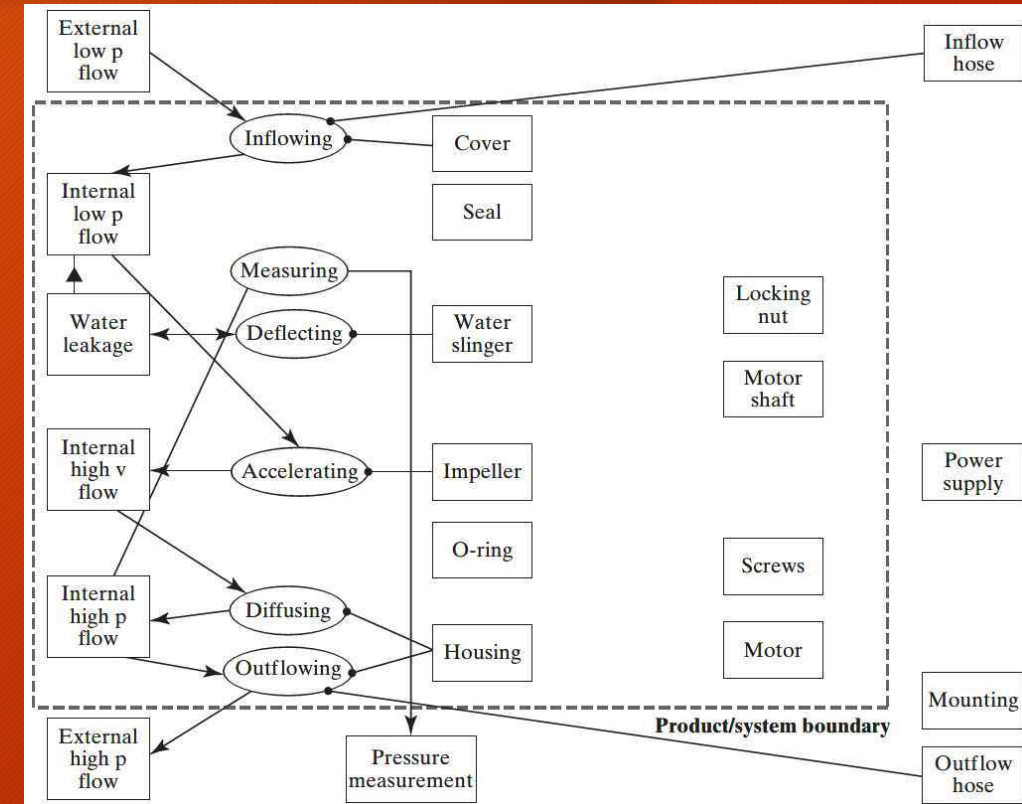
| Object List | Cover | Impeller | Housing | Motor | Motor Shaft |
|-------------|-------|----------|---------|-------|-------------|
| Cover       | X     | S        | SC      |       | S           |
| Impeller    | S     | X        | S       |       | SC          |
| Housing     | SC    | S        | X       | SC    | S           |
| Motor       |       |          | SC      | X     | SC          |
| Motor Shaft | S     | SC       | S       | SC    | X           |

S = Spatial/Topological  
C = Connectivity

- Can a spatial relationship of “close to” support interaction in a fluid flow?
  - Imagine the water stream from a garden hose emerging from the nozzle
  - If the nozzle were close to and pointed at a bucket, the water would be transferred from the hose to the bucket
- In certain cases of such “ballistic” flows, proximity and alignment, which are spatial structural relationships, allow interaction as well
- The formal structure of “close to” also informs performance
- A smaller gap will allow a higher net rise in pressure

# Non-Idealities in System Architecture

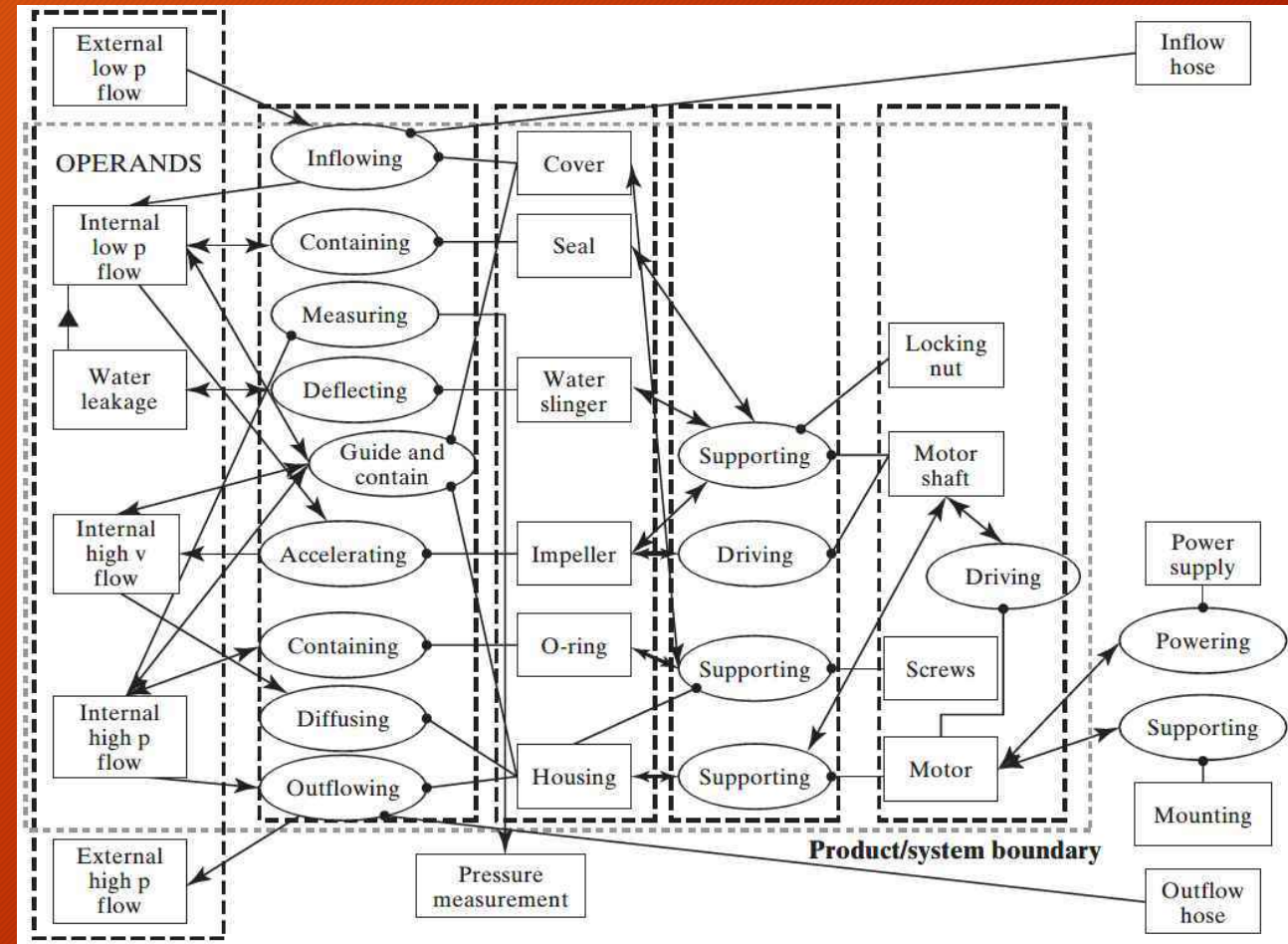
- Looking at the current system architecture for the centrifugal pump, it is seen that some instrument objects are not yet associated with an internal process
  - Seal
  - O-ring
  - Motor
  - Motor shaft
  - Screws
  - Locking nuts
  - Power supply
  - Mounting
- Up to this point, only the idealized value delivery aspects of the pump have been analyzed
  - There is a lot more involved to making a successful system
  - Getting the value delivery pathway right is necessary, but not sufficient
- Other aspects of system function are necessary to develop as well
- These include non-idealities in the value pathway, supporting functions, and interface functions



# Non-Idealities in System Architecture (continued)

- There is a whole class of these non-idealities that are associated with managing the operands
  - Moving them
  - Containing them
  - Storing them while the value processes are taking place
- Another, related class of internal functions provides extra performance or robustness
- When reverse-engineering
  - Examine the instruments that are not associated with the principal and secondary value flows
  - Reason how these either help manage the operands or improve performance or robustness
- Applying this procedure to the current system architecture of the centrifugal pump, the seal and O-ring are identified as unassigned instruments of form that are “close” to the value pathway
  - They touch the water
  - The O-ring helps to contain the water
  - The seal reduces leakage of the water by the shaft
- The cover and housing already support ideal internal functions, but they have an additional non-ideal function
  - They guide and contain the water as it moves through the impeller
- These non-idealities are shown here, in the new system architecture diagram

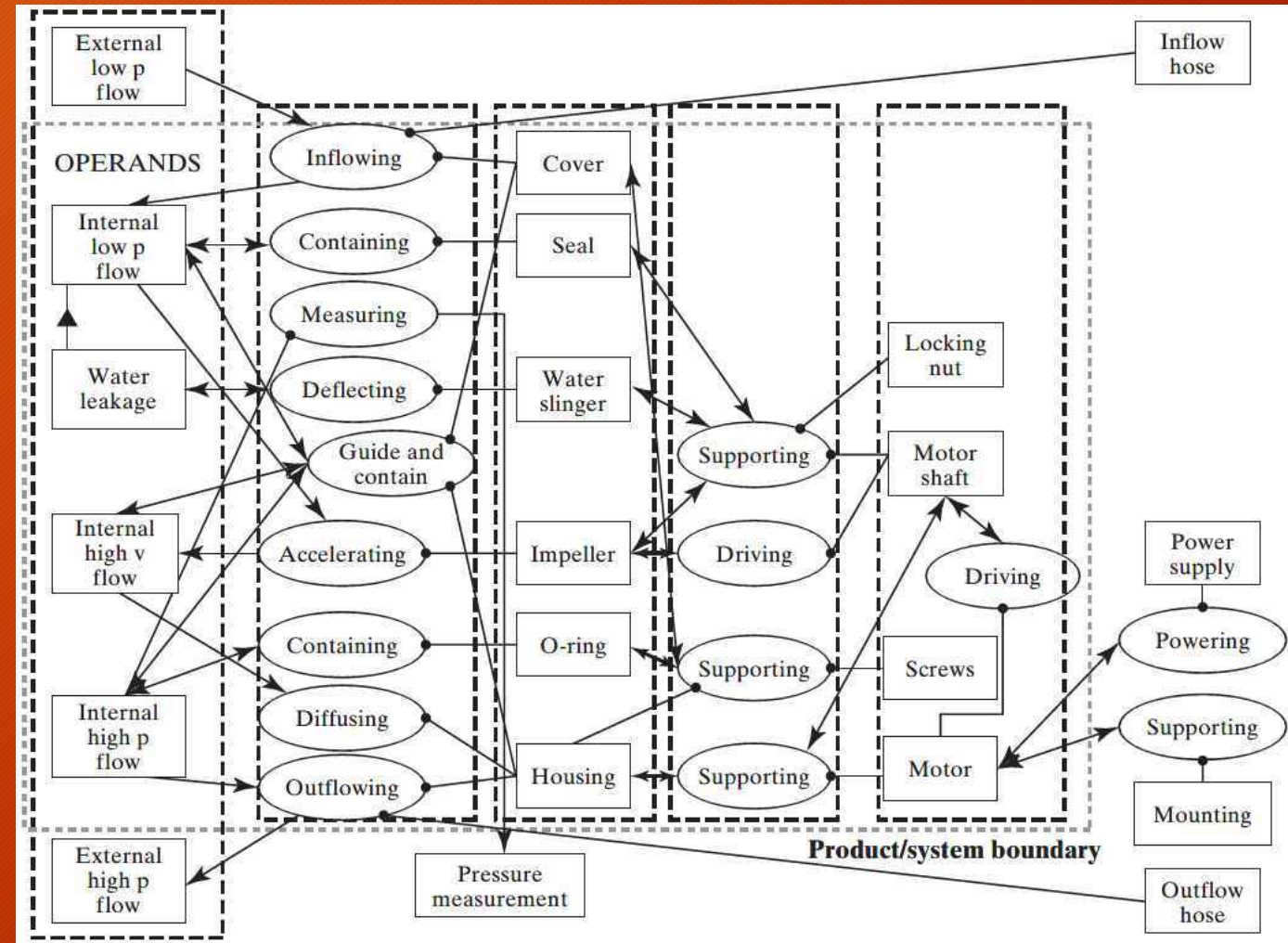
New System Architecture Diagram for Centrifugal Pump





**J.G. ARTUS**  
Systems Engineering

- ## New System Architecture Diagram for Centrifugal Pump



# Supporting Functions and Layers in Architecture



- The objects of form on the value delivery pathway (such as the impeller and cover) themselves have to be physically supported against gravity or other applied loads, powered and controlled
- In general, the architecture of a system can be modeled in these layers
  - Value operands
  - Value processes
  - Value instruments
  - Then several alternating layers of supporting processes and instruments
- When identifying processes and instruments in the supporting layers, reflect on how the value instruments are supported
  - Mechanically
  - Energetically
  - Biochemically
  - Informatically
  - etc

# System Interfaces in Form and Function

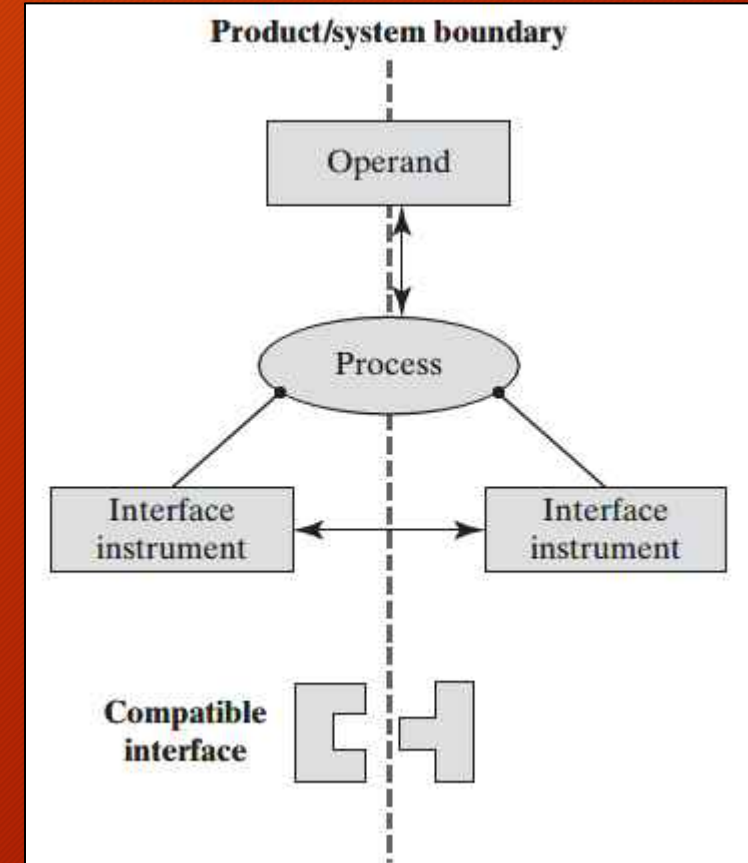


- Boundaries are central to the definition of a system
- The importance of clearly communicating where the system boundary is located cannot be overstated
- The system boundary divides the entities in the product/system from the accompanying systems and defines the entities under the control of the architect
- It identifies the interfaces that must be controlled
- When something is passed into or out of a system, there is a system interface at the boundary crossing
- For the pump, the current system architecture shows five potential interfaces:
  - Inflowing fluid
  - Outflowing fluid
  - External powering
  - External supporting
  - The pressure measurement leaving the product/system

# System Interfaces in Form and Function (continued)

- Like all entities in a system, an interface has form and function (operand and process)
- It is the obligation of the architect to specify the interfaces
  - Citing a standard
  - Writing an interface control document
- There is certain information that must be specified
  - The operand that passes through the boundary, which will be the same on both sides
  - The process of passing the operand, which is usually shared or common at the boundary
  - The two interface instruments, which have some formal relationship
- They can be androgynous
  - Meaning that the interface form is identical on both sides
- Or they can be compatible
  - Meaning that they are different but they somehow fit together

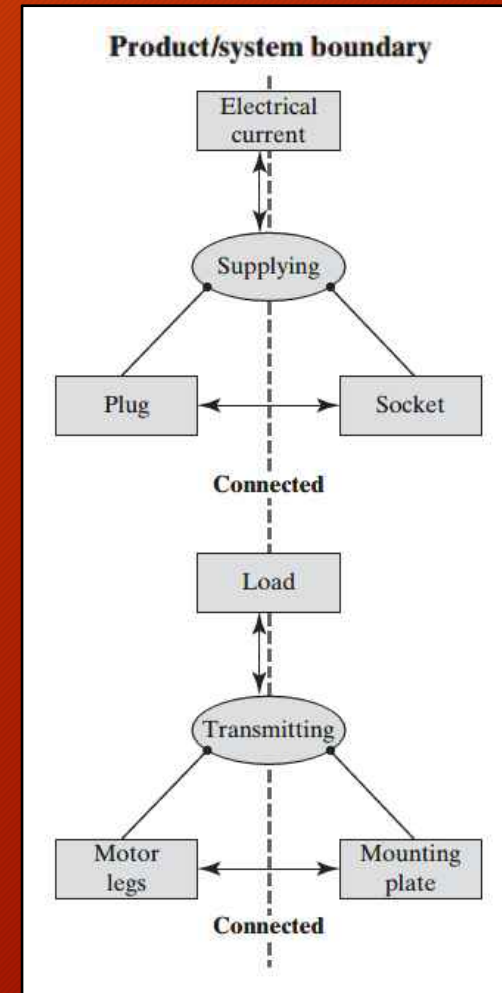
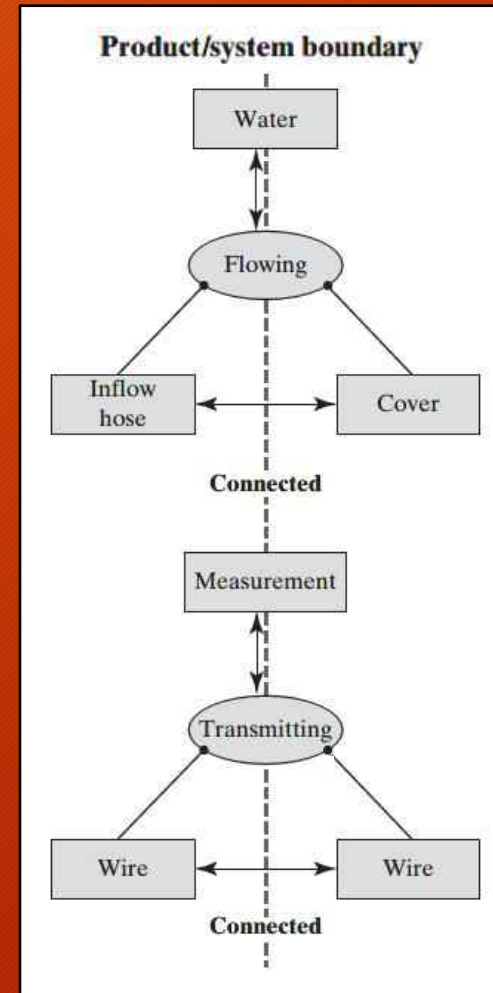
Model of an Interface as a System Boundary



# System Interfaces in Form and Function (continued)

- Four representative interfaces from the centrifugal pump are shown in the interface diagrams to the right
- For the inflowing process
  - The operand is the water
  - Which passes through the boundary unchanged
  - The instruments are
    - Inflow hose
    - Cover, which contains some fitting for the hose
- The outflowing interface is similar
- The powering interface process
  - Passes current
  - Involves the power cable of the motor connecting to a socket
- The measurement signal
  - Is transmitted from an internal wire to an external wire
- The mechanical supporting process
  - Passes the mechanical load from the motor legs to a mounting plate

Models of Four Interfaces at the Pump System Boundary



- The analysis of architecture presented thus far has focused on form and function
  - But function is a somewhat quasi-static view of a system
  - Function is more about what the system can do than about what actually happens when the system runs
- The operational or run time environment is more dynamic
  - Things in the system happen in certain sequences
  - The system interacts dynamically with surrounding systems and people
- It is the difference between describing the system's primary value pathway (its function) and listing, in order, all the steps necessary to actually fulfill stakeholder's needs (its operational behavior)
- Three critical aspects of operational behavior are
  - The operator
  - The system's behavior
  - The operational cost of using the system

# The Operator

- An operator is a person who directly interacts with the system to jointly perform system functions
  - In some cases the operator is so important, that the system simply will not operate without active involvement of the operator
  - For other products, the operator exercises supervisory control over operations
- All systems must accommodate humans any time they touch it
  - Which is normally informed by Specialty Engineers trained in human factors and industrial design
- The human operator is so important that it is considered as a product/ system attribute
- In the case of the centrifugal pump, an operator has some level of supervisory control
  - Someone turns the pump on and off
  - Or at least supervises the mechanism that does so
- Because the pump system is not a rich example of operator interaction, a more simple system is introduced: the two-levered corkscrew
- The operator is essential to the operation of this device
- Operator process:
  - Place the device on top of the bottle
  - Twist the screw into the cork
  - Push down the levers
  - Remove the cork from the screw



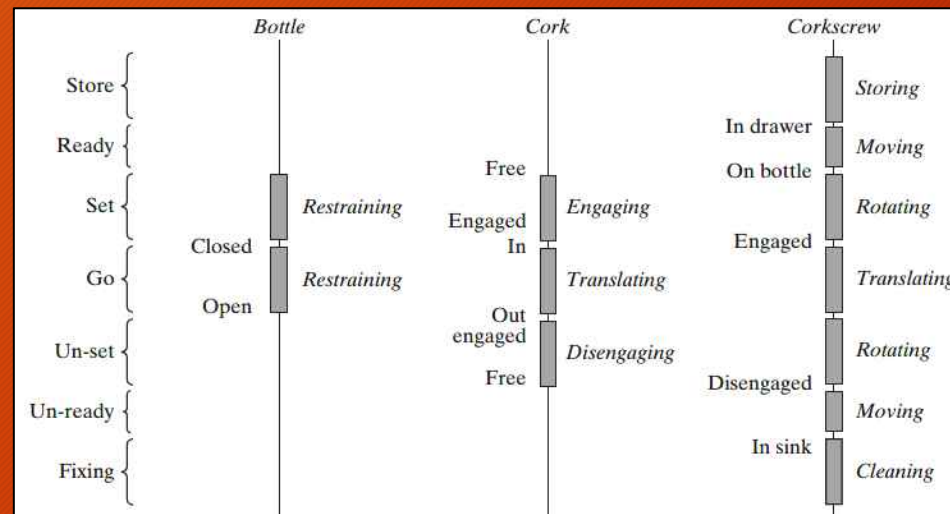
<https://www.pigiame.co.ke/listings/wing-corkscrew-wine-bottle-opener-1866390>

- Behavior is the sequence of functions (and associated changes in state) that form executes in order to deliver value
  - It is important to represent the sequence of events that contribute to external function
  - This sequence of events often results in effecting changes in system state
  - Behavior is a product attribute
- It is important to distinguish between the operations sequence and the dynamic behavior, or timing, of the system
  - Timing includes specific reference to clock time
  - Sequence is based on relationships among actions
- The operations sequence is the total progression of actions or processes that the system undergoes
  - This includes actions associated with the primary and secondary externally delivered function
  - As well as with supporting and interfacing functions
  - From another perspective, sequence is the array of changes in state of the operands of the system
- Sequence describes
  - What step follows what
  - Whether there are overlaps
- Often, the steps must follow in a prescribed sequence
- Sometimes sequence is uncertain or optional

# Operational Behavior of the Corkscrew

- There are various ways to represent the operating sequence of the corkscrew as it engages the cork
  - The simplest is the sequence diagram which indicates the starting and ending states for the processes
- From this diagram, it is immediately apparent that the cork is not the only important object involved
  - The bottle is being restrained
  - The corkscrew is being manipulated
- A more complete representation of the corkscrew sequence is shown by the sequences of the other two system elements
  - Processes are represented on the right side of the line
  - States are shown to the left of the line
- In particular, the corkscrew undergoes several processes and state changes to achieve the desired value function
- As indicated for the corkscrew (far right), these start from storage and end up in maintenance (cleaning)

Sequence Diagram for Corkscrew



- There are a variety of other diagrams that can be used to represent the states and processes of the system, and their conditional interaction
  - These include the three SysML detailed behavioral diagrams
    - Activity diagram
    - Sequence diagram
    - State machine diagram
- In contrast to sequence, dynamic behavior is the detailed timing of steps, start time, duration, overlap, and so on
  - Such behavior is important in real-time systems
  - The issues that typically arise in dynamic behavior include start-up transients and latency of information or elements in a system
- One of the most vexing aspects of real-time systems is timing constraints associated with multiple parallel sequences or threads
  - Dynamic behavior is best represented in the specific sub-domain of the architecture and is left to the architect to explore
    - Multi-threads
    - Non-synchronous events
    - Non-deterministic events
  - The architect is advised to first learn about these issues if they are relevant to the architecture of the product/system

# Operations Cost



- Forecasting operations cost is frequently a consideration when architecting systems
- Operational costs are influenced by
  - The operational behavior
  - The operational concept
  - The details of system operations
- Operational cost is typically expressed per event, per day, or in terms of usage (per mile in a car)
- Like operational behavior and operator, operational cost is a product/system attribute
- Operations cost is built up of a number of components
  - Chief among them is often the cost of the operator and other personnel needed to run the system
  - Consumables are another major category of operational cost
  - Indirect costs of operations
    - Maintenance
    - Nominal upgrades
    - Insurance
- The architect must carefully consider architectural decisions that affect the operational costs
  - These costs are an important factor in the long-term competitiveness of the product/system

# References



1. Crawley, Edward; Cameron, Bruce; Selva, Daniel (2016). *System Architecture: Strategy and Product Development for Complex Systems*, Pearson Higher Education Inc, Hoboken, NJ
2. Suh, Num (2001), *Axiomatic Design, Advances and Applications*, Oxford Press, New York, NY
3. Farid, Amro; Suh, Num (2001), *Axiomatic Design in Large Systems, Complex Products, Buildings and Manufacturing Systems*, Springer International Publishing AG, Switzerland