Analysis of Function

Lecture 05, v03

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

Introduction

- The analysis of function and form are tightly interwoven
  - When employing zigzagging, the architect works in one domain for a period of time and then switches to the other domain
  - Discussing form first does not imply any strict sequence in analysis
  - Discussing form first is simply an educational mechanism used since form is more tangible than function, and therefore easier to present first
- *Function* is what a system does
  - Externally delivered functionality is linked to the primary purpose of the system
  - This external functionality emerges from internal functions
  - Function is about activity, in contrast with form, which is about existence
  - Function involves operations, transformations, or actions
  - *Performance* (how well a system performs) is an attribute of function

Definition of Function

Function is the activity, operation, or transformation that causes or contributes to performance

In designed systems, function is the actions for which a system exists, which ultimately lead to the delivery of value

- Function is executed by form, which is instrumental in function
- Function emerges from functional interaction between entities
- Function is a product/system attribute

Objects Analytically Represent Form

- Systems can be represented with only two classes of abstractions
  - *Objects*
  - *Processes*
- Objects and processes faithfully describe the system's structure, function, and behavior in a single, coherent model in virtually any domain
- An *object* is that which has the potential for stable, unconditional existence for some period of time
- Recall that *form* is the physical or informational embodiment of a system that exists or has the potential for stable, unconditional existence, for some period of time, and is instrumental in the execution of function
- Thus, objects could be used to represent form
- The definition of form contains the additional criteria that the form must be instrumental of function and must exist before function
  - Objects that do not meet these additional criteria are *operands*
  - Example: data passed on an interface (concrete) or a system state (abstract)

Object	Has the potential for stable, unconditional existence for some period of time
Element of Form (Formal Entity)	Is instrumental of function Exists before function
Element of Function (Operand)	Is not instrumental of function Can exist before or after function

Operands

- *Objects*
  - Some objects in a system are elements of form, and some are operands
  - All objects, including operand objects and objects of form, have the potential for stable unconditional existence for some period of time
- Objects of form
  - Must exist before the function
  - Are instrumental in the function
  - Are designed by the architect and supplied with the system
- Objects of function are called *operands*
  - Operands are the parts of function that represent what is changed by the function
  - Operands need not exist prior to the execution of function and are in some way acted upon by the function
  - Operands may be consumed, produced, or modified by the process part of function
  - Operands are usually not supplied by the architect or builder of the system
  - They often appear at the time of operation, usually from other sources
  - Often, architects do not have a great deal of control over the availability nor the properties of operands
  - Both the stakeholder and the architect are not only concerned about the primary functions that the system performs, but also what are the primary operands that are input into and output from the system
    - Goods – tangible operands produced by the system; what the system produces for benefit of the stakeholder
    - Services – intangible actions delivered to the consumer for their benefit

Note: The textbook "Systems Architecture" uses the terms "destroy" to represent the effect on input operands and "create" to represent the effect on output operands In this lecture, we instead use the terms "consume" and "produce"

Definition of Process

Process is a pattern of transformation undergone by an object

Processes generally involve consumption of, production of, or a change in an operand

- A process is a pattern of transformation applied to one or more objects
- Processes generally involve the consumption or production of an operand, or a change in an operand
- Processes are transient and dynamic and take place along a timeline
- Verbs are used to describe processes

Form and Function - Review

- Form has
  - *Entities of form (objects)* and
  - *Formal relationships among the entities (structure)*
- Similarly, function has
  - *Entities of function* and
  - *Functional relationships among the entities (interactions)*
  - The function we see on the surface of a system is a result of the emergence that has taken place among these entities of function within the system, and in the whole product system
- Function is vitally important to the design of systems and system architecture
  - All of the magic of systems and their emergence, and nearly all of the challenge of designing them, is found in the functional domain
- Function is a major system attribute
  - All systems have function
  - In built systems, function must be conceived so that the goals of the system are achieved

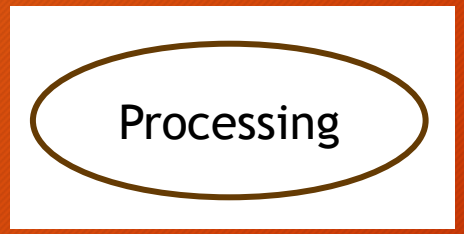
Function is made up of a process and operands

This leads to the rule

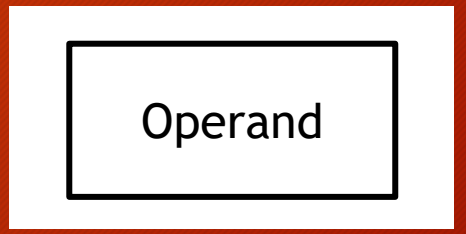
function = process + operands

Depiction of Functionality in OPM

- The Object-Process Methodology (OPM) is a systems modeling paradigm that represents the two things inherent in a system: its objects and processes
- OPM has a set of symbols used to represent objects and processes and their interactions
  - Processes are shown by ovals, with a label indicating the name of the process
  - Operands are shown as rectangles, because they are objects, with the name of the operand written inside
  - Functions are made up of these two OPM elements



Processes are represented in OPM as ovals

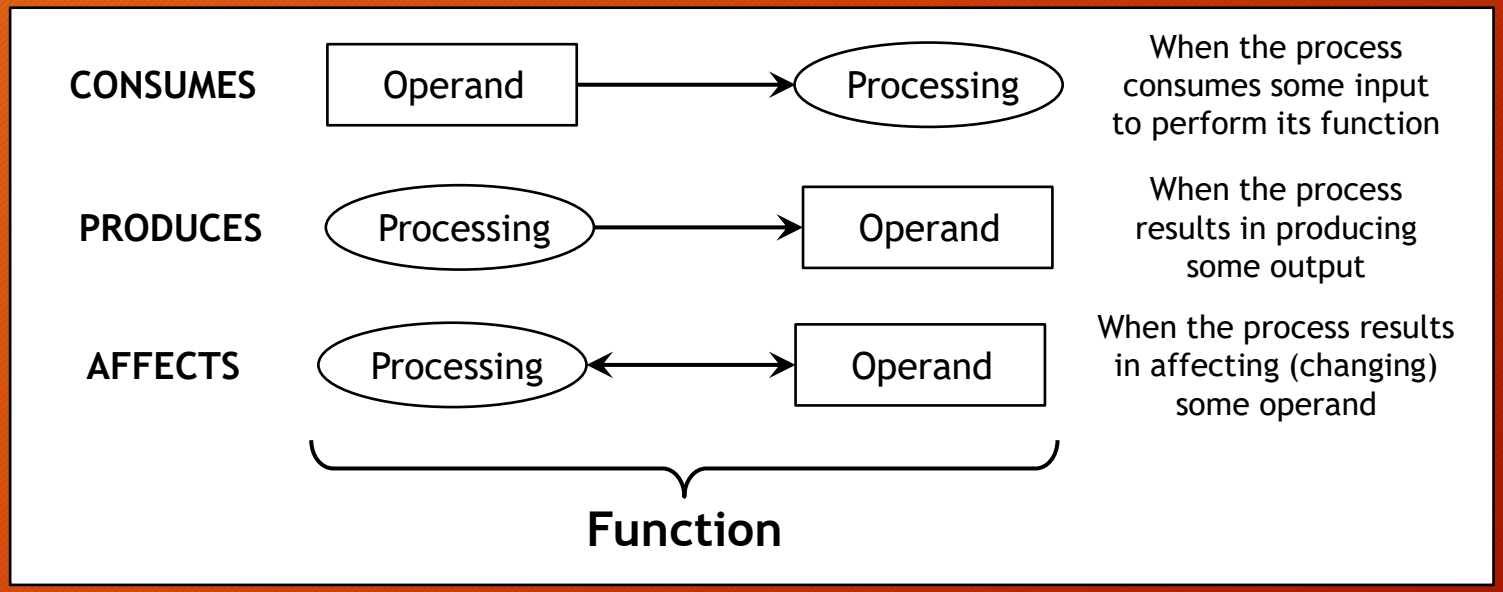


Operands are represented in OPM as rectangles

Depiction of Functionality in OPM (continued)

In OPM, there are three ways of representing the relationships between an operand and a process

- Single-headed arrow running from the operand to the process
  - Implies that the process consumes the operand, as when a factory consumes parts (to make a car)
  - Consumption implies that the abstraction of the operand no longer exists in the original nature and place after the process executes
  - The car parts still exist after they leave the factory, but they have become part of another abstraction, called the car
- Single-headed arrow running from the process to the operand
  - Implies that the process produces the operand, as a factory would produce a car
  - The car did not exist before the process; after the process, it did
- A double-headed arrow implies that the process affects the operand but does not consume or produce it
  - The existence of the operand did not change
  - Rather, some attribute of the operand was changed by the process



When a process consumes an operand, and then produces another operand, this could be considered "transformation", especially when the two operands are actually the same object, just transformed in some way (such as freezing water into ice) Sometimes these kinds of cases can be considered to be a change of state of the same operand

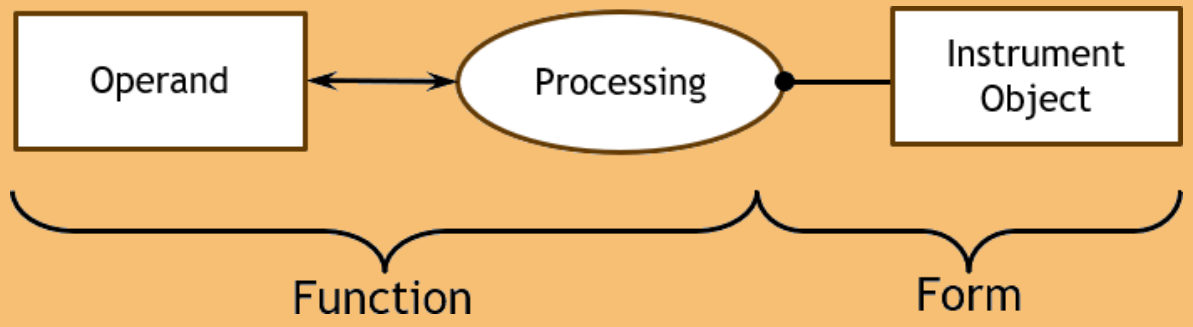
Depiction of Functionality in OPM (continued)

OPM applies a round-headed arrow as the symbol that links the instrument object with the process that it enables

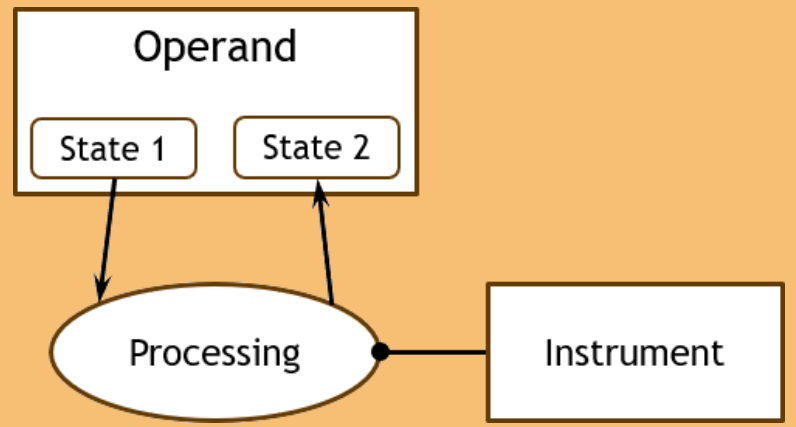
An "instrument object" is an entity of form (a system component) that hosts or implements a process

These diagrammatic features enable the analytical diagramming of form and function

The canonical model of a system using OPM



A more explicit representation of the canonical system is illustrated here, in which the states of the operand are explicitly shown



This view emphasizes that the operand changes its state through the action of the process enabled by the instrument objects

Primary Externally-Delivered Function

- Two aspects of the primary externally delivered value-related function
  - First
    - The functionality must be externally delivered
    - The functionality must cross the boundary of the system and influence something in the context
  - Second
    - The system exhibits a primary function
    - That is the function for which the system was built
- It is important that the function is externally delivered
  - Function only delivers value to the stakeholder when it acts externally to the system
  - It cannot only be internal to the system
- This implies an important rule of systems:
  - Primary externally-delivered function and value are always delivered at an interface on the system boundary
- The form, however, is still responsible for delivering the system functionality
  - Without form, there can be no function

Example: Operational Amplifier (Op Amp)

- Built systems have a primary externally delivered function
- This is the function for which the system is built
- If the built system does not deliver this function, it will be a failure
- Example:
  - Operational amplifiers are built to amplify
  - They may also filter high frequencies
  - But if they do not amplify, they are a failure

An operational amplifier is a DC-coupled high-gain electronic voltage amplifier with a differential input and, usually, a single-ended output

In this configuration, an op amp produces an output potential that is typically 100,000 times larger than the potential difference between its input terminals

Function			Form
Value-related operand	**Value-related attribute and state**	**Value-related process**	**System form**
Output signal	Magnitude (higher)	Amplifying	Operational amplifier

- It is critical to focus on delivery of the primary function in order to not lose sight of it amongst the enormous list of requirements and potential features to deliver
  - Many products are delivered that fail to provide the primary function, providing instead a host of secondary features
  - The architect should try to reason about the primary function, before being distracted by other features

Value-Related Operands

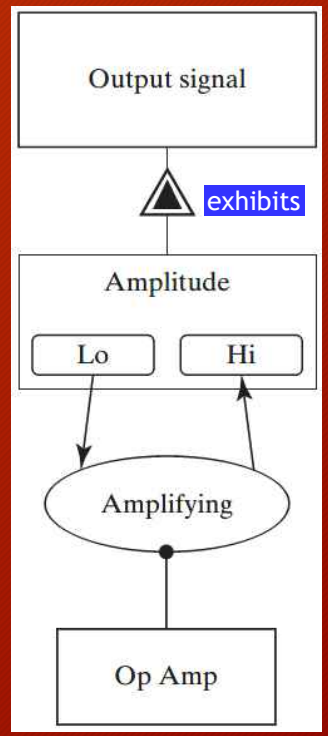
- Value is defined as benefit, worth, importance, or utility delivered at some associated cost
  - Good value can therefore be created by high benefit at modest cost, or by modest benefit at low cost
  - Cost is a measure of the tangible contribution that must be made in exchange for the benefit
- Value is subjective and is judged by an observer who is the customer, beneficiary, or user (not the architect*)
- A system delivers value when it is working properly and executes its primary externally delivered function
  - The benefit delivered by systems is provided by the primary externally delivered function
  - There is usually a single externally delivered function
  - The inability of the system to deliver the primary functionality constitutes failure of the system
  - Often, emergence of the primary system function during operation is far from certain

* To do so, the architect must serve in the interest of the customer, which could generate a conflict of interest

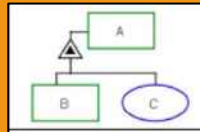
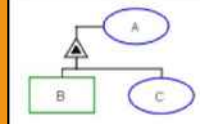
Value-Related Operands (continued)

- Good architecture delivers benefit, first and foremost
  - The objective is the delivery of the primary externally function of the system for which the system was originally defined
  - Accomplished by focusing on the emergence of functions
  - And delivery of that functionality across the system boundary at an interface
- Value-Related Operand
  - It is critical to identify the value-related operands from among all of the operands on which the system acts
  - The value-related operands can be identified by asking which operand the system exists to influence
    - To consume, produce, or affect
  - Often, there are operand objects present that are not directly related to the delivery of value
    - These are simply called “other operands”
- Op Amp Example
  - The op amp clearly exists to produce a high-amplitude output signal
    - “Amplitude” is the value-related attribute of the value-related operand “signal”
  - The op amp system delivers value when the process “amplifying” acts on the operand “signal” in such a way as to change its value-related attribute “amplitude” to the state of “high”

OPM diagram showing the delivery of the Amplifying function



OPM Syntax for Exhibition or Characterization

	**A** exhibits **B**, as well as **C**.	Object B is an attribute of A and process C is its operation (method).
	**A** exhibits **B**, as well as **C**.	A can be an object or a process.

Internal Functionality

- Within a system, there are internal functions and relationships among these internal functions
  - Together these define the functional architecture of the system
  - The externally delivered function, performance, and “ilities” emerge from these internal functions and relationships
- Procedure for Analyzing Internal Functionality
  - Identify the important internal functions or the entities of function
  - Then construct the functional architecture by integrating the entities of function
- Each of these internal functions has an operand part and a process part
- For the operational amplifier
  - Gain is an operand, and Setting is a process
  - The output signal (with Voltage as a state) is an operand, and Increasing is the process

Principal Internal Operands	Internal Processes	System Form
Gain	Setting	Operational amplifier
Voltage	Increasing	

Identifying Internal Functions

There are several approaches to identifying internal functionality

In applying these approaches, first focus on the internal processes that lead to the creation of value

- Reverse engineering of the form
  - Start with the elements of form
  - Reverse-engineer their function by asking what do the system elements do
- Use of standard blueprints
  - Some combinations of processes often go together to achieve an emergent function
    - These are called standard blueprints (also called “patterns”)
  - Sometimes functions just naturally unfold into a set of internal functions that are stable over many years
    - This may be because this is inherently a stable blueprint
    - Or because we use the blueprints of prior internal functions when we develop new systems
    - In either case, the blueprint is useful in understanding new systems
  - Successful businesses have developed standard processes blueprints over decades to improve efficiencies

Identifying Internal Functions (continued)

Further approaches to identifying internal functionality

- Use of metaphors
  - Consider a well-understood system as a metaphor for an unknown system to identify its internal functionality
- Other approaches to analyzing internal functions include the three techniques developed for predicting emergence
  - Precedent
  - Analysis
  - Experimentation
- Often, just observing operations is valuable too
- Also, you can apply domain knowledge and experience

Identifying Internal Functions (continued)

In the case of the centrifugal pump we observe the primary value delivery that is of interest to the stakeholder:

- Take in low pressure fluid at input port
- Output high pressure fluid at output port

There are other obvious features:

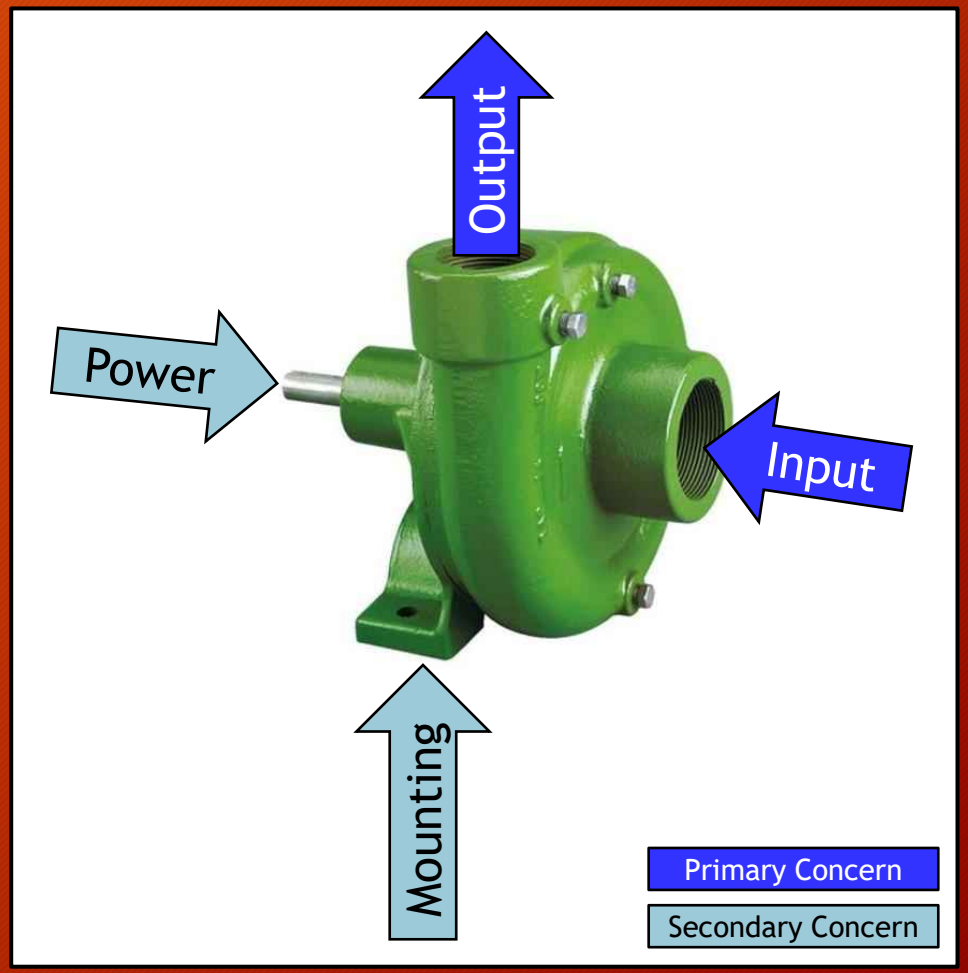
- The pump needs to be powered to operate
- The pump needs to be properly mounted to operate effectively

While these are concerns, they are not the primary concerns of stakeholders (delivery of value)

This is the “black box” view of the pump functionality

From this perspective, we cannot see how the pump transforms low pressure input fluid to high pressure output fluid

“Black-Box” View of Centrifugal Pump



https://www.sprayerdepot.com/products/ace-fmc-200-hydraulic-driven-centrifugal-pump

Process-Operand (PO) Array

- A matrix approach can be used to represent internal functionality
  - Create a Process-Operand (PO) Array that lists the processes on one axis and the operands on the second axis
  - Identify the external inputs (external operands that are consumed by the system) and external outputs (external operands that are produced by the system)
  - Through analysis of the internal processes occurring within the system, identify how it is that the external input operands are transformed into the external output operands
  - In the array, show how each input operand is consumed (c) by a process, that then produces (p) an output operand that then serves as the input operand for the next process in sequence
- This procedure is applied to the centrifugal pump example
  - Two obvious pump functions that can be observed externally are the inflowing and outflowing functions
  - To identify the two principal internal functions of the pump requires some domain knowledge of pump design
  - The pump first invests kinetic energy into the fluid by accelerating it (implemented by the impeller)
  - It then trades kinetic energy for potential energy by diffusing the flow (implemented by the volute chamber), thereby converting high velocity, low pressure fluid flow into low velocity, high pressure fluid flow

Impeller

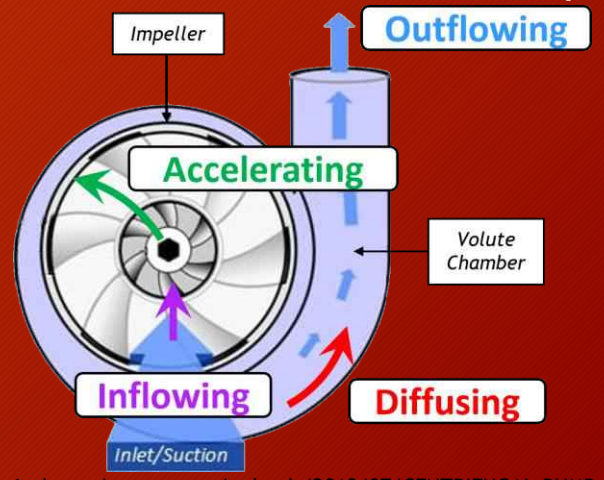
Operands

Process List	Internal low pressure flow	Internal high velocity flow	Internal high pressure flow	External low pressure flow	External high pressure flow
Inflowing	p			c (Input)	
Accelerating	c	p			
Diffusing		c	p		
Outflowing			c		p (Output)

Processes

Value-related operand

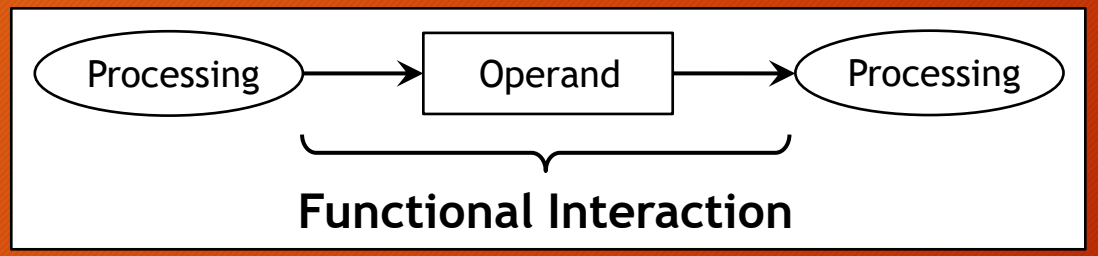
"White-box" View of Pump



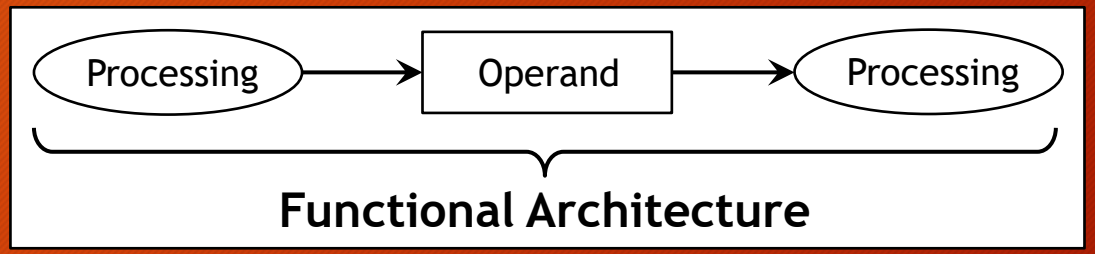
https://learnmechanical.com/wp-content/uploads/2018/07/CENTRIFUGAL-PUMP.jpg

Functional Interactions and Functional Architecture

- Developing the functional architecture requires understanding the relationships among the entities of function (called the functional interactions)
  - This is the key step in analyzing architecture
  - It is through the interaction of the internal functions that the externally delivered function emerges
  - The key idea is that *the exchanged or shared operands are the functional interactions*



- Therefore, *the functions, plus the functional interactions, are the functional architecture*
  - In simple flow-through systems, there is a unique operand that is produced by the upstream process and consumed by the downstream process
  - This produce/consume mechanism constitutes the functional interaction



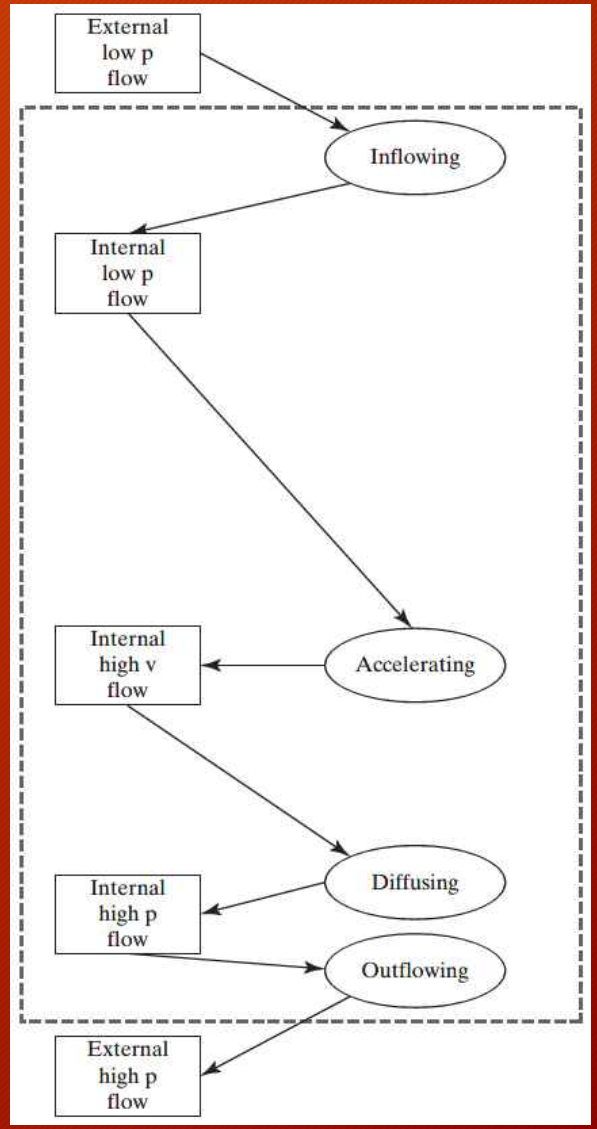
Process for Identifying Functional Interactions

- Start with the diagram of the internal functions
  - This must include the value-related operand
- Identify processes that are obviously missing from the diagram
  - Such as inputting and outputting processes
- Identify any operands that are obviously missing from the diagram of internal processes
  - Such as inputs and outputs
- For each process, ask whether there any other operands that are needed for the function to be completely represented
- For each operand, ask what other processes interact with the operand
- Trace the path from the value-related output
  - Consider whether the desired emergence is likely to occur based on the processes and operands represented

Procedure for Creating the Functional Architecture

- The OPM diagram shown here illustrates the internal functions of the pump
  - The diagram shows that the internal functions fit into a simple flow-through pattern
  - Each process consumes one operand and produces another
- Identify externally obvious inputs, outputs, and functions
  - The Inflowing process and the Outflowing process
  - The external low-pressure flow (input operand) and the external high-pressure flow (value-related output operand)
- Using domain knowledge, identify the internal functions and internal operands
  - The centrifugal pump employs the impeller to accelerate the fluid to a high velocity
  - Between the processes of Inflowing and Accelerating, there must be an internal operand labeled internal low pressure flow
  - The centrifugal pump employs the volute chamber to decelerate the fluid and produce high pressure
  - Between the processes of Accelerating and Diffusing, there must be an internal operand labeled internal high pressure flow
  - With all the operands present, we connect the operands to processes as shown
- Now identify the interactions between the processes
  - The interaction between the Inflowing and Accelerating processes is the exchange of the internal low-pressure flow operand
  - The interaction between the Accelerating and Diffusing processes is the exchange of the internal high-pressure flow operand

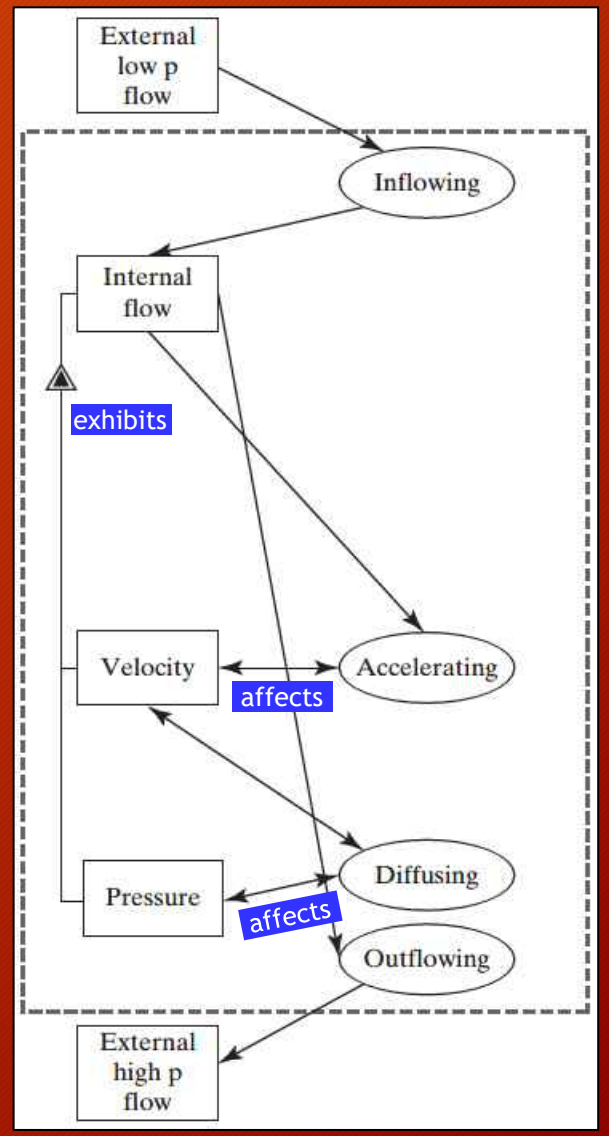
Functional Architecture for the Pump



Procedure for Creating the Functional Architecture (continued)

- Decomposition of form is a more tangible and therefore a more definite process (easy to understand and to implement)
- However, the interpretation of the abstractions for the internal functions of a system is not unique
  - For example, the figure on the previous chart gives the impression of the pump as a simple flow-through system
  - There are often several equally valid ways to define the internal processes and operands
  - The figure to the right illustrates another way of depicting the processes and operands of the pump
- The real question centers on whether emergence is easily understood and predicted
  - We recognize that water is not really consumed by a process, and new water is not really produced
  - What really happens is that the water changes state
  - The figure shows that the internal flow has two important attributes: pressure and velocity
  - The accelerating process changes the velocity (from low to high)
  - The diffusing process changes both the velocity (from high to low) and the pressure (from low to high)

Modified Functional Architecture for the Pump



Revised Process-Operand (PO) Array

- Representing this change of state loses the sense of simple flow-through but more accurately captures the physics of the pumping process
- A revised PO array can be constructed based on this new perspective

Modified Process-Operand Array for the Pump

Process List	Internal Flow			External low pressure flow	External high pressure flow
	Internal flow existence	Velocity state	Pressure state		
Inflowing	p			c (Input)	
Accelerating		a (high velocity)			
Diffusing		a (low velocity)	a (high pressure)		
Outflowing	c				p (Output)

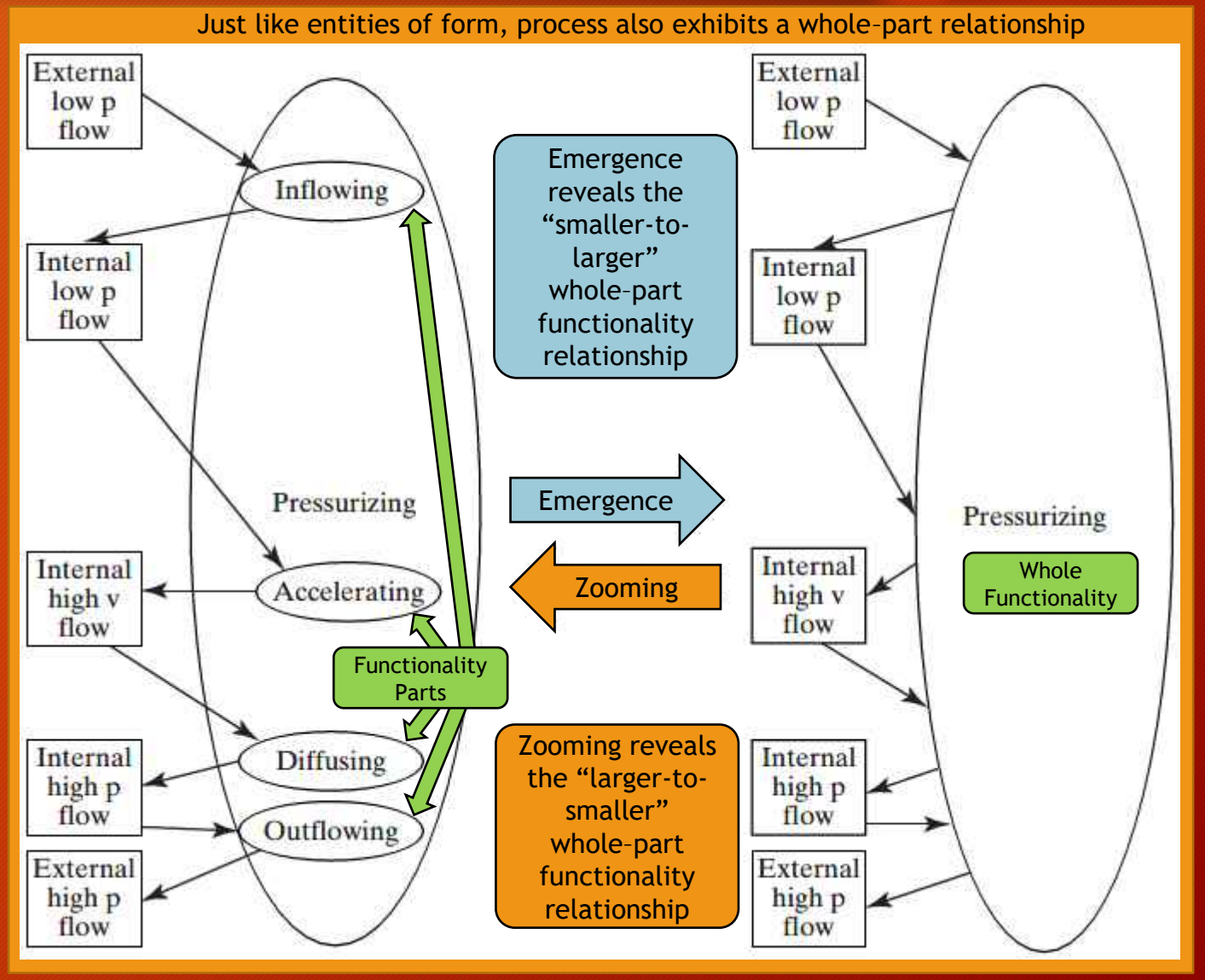
The operands at the top of the array now explicitly show the various states of water in the pump
The "a" under "velocity state" implies that the accelerating process affects (hence "a") the velocity state

Value Pathway

- One of the main features of the functional architecture is the value pathway along which value develops
  - It is along this pathway that benefit and ultimately value develop
  - It often starts with some input and progressively moves through internal operands and processes until it reaches the value-related output
  - In flow-though functional architectures, the value pathway is often readily evident
  - In other types of architectures, more careful reasoning may be necessary to spot it
- There are several classes of operands, processes, and objects of form that are not on the value pathway:
  - Entities of form are not on the value pathway because form is not part of functional architecture
    - Entities of form only provide an instrumental role to the processes
  - Processes and operands that support secondary externally delivered functions are not on the value pathway
  - Processes and operands that don't contribute to the emergence of any desired externally delivered function
    - These are sometimes the result of
      - Unwanted side-effects
      - Poor design
      - Legacy (outdated) designs
    - These can sometimes contribute to gratuitous (unnecessary) complexity
  - Supporting processes and form
    - These are even further from the value pathway

Emergence and Zooming

- Knowing the internal value pathway is a key step in effectively predicting emergence
- In the view on the left (zoomed-in view), all of the internal operands are shown connecting to their associated internal processes
- In zooming, identify not only the internal processes themselves, but also how the external and internal operands interact with those processes
- The right side view shows the pressurizing process emerging, with the internal processes suppressed and the internal and external operands simply attached to the emergent process

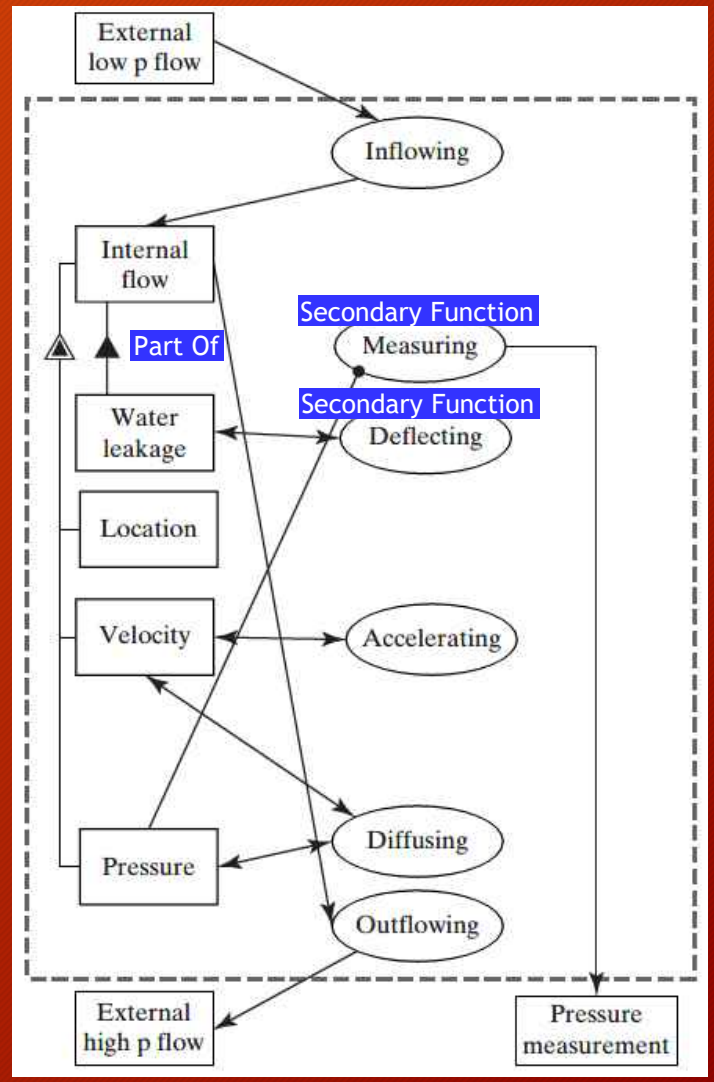


Secondary Value-Related Functions

- Once a system delivers its primary value-related function, there is no reason why it should not also be designed to deliver other secondary value-related functions
- It is important to identify these secondary functions, because they are expected by customers and are a source of competitive advantage
- These other value-related external functions also emerge from the internal processes
- The procedure to address this question is just to repeatedly apply the reasoning for the primary value-related function to these secondary functions
- Centrifugal Pump Example
  - The centrifugal pump could have secondary value-related functions
  - Many water pumps have a “slinger” to sling water that leaks through the rotating seal away so that it does not enter the electric motor
  - The pump could also include a pressure sensor that indicates the outlet pressure of the water, a measure of the property associated with delivery of the primary value
- One (Deflecting) appears along the principal value pathway
- The other (Measuring) on its own pathway

Systems deliver secondary external value-related functions, in addition to the primary value-related function for which the system exists

Internal functions associated with secondary value-related functions



Summary

- Like form, function is a system attribute conceived by the architect
- It is the actions, activities, operations, and transformations that cause or contribute to performance
- Function is made up of a process, which is pure activity, acting on an operand, which is an object transformed by the process
- Built systems have a value-related operand, resulting from a transformation that is associated with the delivery of benefit and eventually value
- The value of a system occurs when the externally delivered function acts on an external operand across the system boundary
- The externally delivered value-related function and other, secondary value-related functions emerge from the internal functional architecture
  - Such emergence usually features a value-related pathway within the architecture
- A complete description of functional architecture includes internal processes and internal operands

Summary of the Features of Form and Function

Form	Function
What a system is (noun)	What a system does (a verb)
Objects + formal structure	Operands + processes
Aggregates (and decomposes)	Emerges (and zooms)
Enables function	Requires instrument of form
Specified at an interface	Specified at an interface
Source of cost	Source of external benefit
When transaction is a good	When transaction is a service

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. Wikipedia (2021). *Operational Amplifier*. Retrieved from https://en.wikipedia.org/wiki/Operational_amplifier
3. Dori, Dov (2002). *Object-Process Methodology, A Holistic Systems Paradigm*, Springer-Verlag Berlin Heidelberg, New York, NY