**NAME: DAKWOYI SAMUEL SHEKWOYANDU**

**MATRIC NO: 16/ENG02/060**

**COURSE CODE: CSC 410 UPDATES**

**1. Explanation**:

A number of laws are derived which establish relationships between throughput, response time, device utilization, space-time products and various other factors related to computer system performance. These laws are obtained by using the operational method of computer system analysis. The operational method, which differs significantly from the conventional stochastic modeling approach, is based on a set of concepts that correspond naturally and directly to observed properties of real computer systems. Operational methods simply refer to methods of physical observation and measurement, in other words, they are directly measurable or directly measured. Except for measurement errors, the operational laws apply with complete precision to all collections of observational data, and they are similar to fundamental laws found in other areas of engineering and applied science.

2. Exhaustively describe at least eight operational laws that are widely employed in computer system performance evaluation.

1. **Utilization Law**

Utilization law can be used to verify the internal consistency of a set of empirical data collected during some observation interval.

Its formula is given as which can be denoted as

It is derived from the equation which is

The formula above can be simplified into Throughput Mean Service Time

1. **Forced Flow Law**

Forced Flow Law relates the system throughput to individual device throughputs.

According to the Forced Flow law, the following is proven

* In an open model, System throughput is equal to the number of jobs leaving the system per unit time
* In a closed model, System throughput is equal to the number of jobs traversing out to in link per unit time.
* If observation period T is such that Ai = Ci, then the Device satisfies the assumption of job flow balance.
* Each job makes Vi requests for ith device in the system.
* Ci = C0 Vi or Vi = is called visit ratio
* System Throughput (X) = =
* Device Throughput (Xi) = = which can be simplified to Xi = X Vi

Where Ci and C0 = The Jobs completed

T = Total Time

Ai = Number of arrivals

Vi =

1. **Little’s Law**

Little’s Law is a theorem that determines the average number of items in a stationary queuing system based on the average waiting time of an item within a system and the average number of items arriving at the system per unit of time.

The law provides a simple and intuitive approach for the assessment of the efficiency of queuing systems. The concept is hugely significant for [business operations](https://corporatefinanceinstitute.com/resources/knowledge/finance/what-is-corporation-overview/) because it states that the number of items in the queuing systems primarily depends on two key variables, and it is not affected by other factors such as the distribution of the service or service order.

Little’s Law can only be used in queuing systems. Almost any queuing system and even any sub-system can be assessed using the law. In addition, the theorem can be applied in different fields, from running a small coffee shop to the maintenance of the operations of a military airbase.

The Formula for Little’s Law:   
Mean Number in the device = Arrival Rate Mean Time in the device

Which can be denoted as Qi = λi Ri

If the job flow is balanced, the arrival rate is equal to the throughput and we can write:

Qi = Xi Ri

1. **General Response Time Law**

According to General Response Time Law, there is one terminal per user and the rest of the system is shared by all users.

When applying Little's law to the central subsystem, the formula is Q = X R

Where, Q = Total number of jobs in the system

R = system response time

X = system throughput

The General Response Time Formula is R = X1R1 + X2R2 + … + XMRM

When dividing both sides by X and using forced flow law, the formula is

R = V1R1 + V2R2 + … + VMRM

1. **Interactive Response Time Law**

According to Interactive Response Time Law:

* If Z = think-time, R = Response time

The total cycle time of requests is R+Z

Each user generates about T/(R+Z) requests in T

* If there are N users:

System Throughput = =

This can be further simplified into )

Or R =

1. **Bottleneck Analysis**

From forced flow law, the formula is Ui α Di

According to the Bottleneck Analysis,

* The device with the highest total service demand Di has the highest utilization and is called the bottleneck device.
* Note: Delay centers can have utilizations more than one without any stability problems. Therefore, delay centers cannot be a bottleneck device.
* Only queueing centers used in computing Dmax.
* The bottleneck device is the key limiting factor in achieving higher throughput.
* Improving the bottleneck device will provide the highest payoff in terms of system throughput.
* Improving other devices will have little effect on the system performance.
* Identifying the bottleneck device should be the first step in any performance improvement project.

Throughput and response times of the system are bound as follows:

And

Here, D = is the sum of total service demands on all devices except terminals.

These are known as asymptotic bounds

Bottle Neck Analysis Proof:

The asymptotic bounds are based on the following observations:

The utilization of any device cannot exceed one. This puts a limit on the maximum obtainable throughput.

The response time of the system with N users cannot be less than a system with just one user. This puts a limit on the minimum response time.

The interactive response time formula can be used to convert the bound on throughput to that on response time and vice versa.

For the bottleneck device b we have: Ub = XDmax

Since Ub cannot be more than one, we have: XDmax 1

X

With just one job in the system, there is no queueing and the system response time is simply the sum of the service demands:

R(1) = D1 + D2 + … + DM = D

Here, D is defined as the sum of all service demands. ! With more than one user there may be some queueing and so the response time will be higher. That is:

Applying the interactive response time law to the bounds:

R(N) D

Combining these bounds we get the asymptotic bounds

1. **Space-Time Product Throughput Law**

The Space-Time Product Throughput Law states that the throughput is equal to the average amount of memory in use are divided by average space-time product.

The formula is:

Where X = Throughput (i.e. number of job completions per unit time)

M = Average amount of memory in use during the observation interval

Y = Average space-time product (i.e. space-time product completed per job

1. **Space-Time Product Response Time Law**

The Space-Time Product Response Time Law is obtained by a similar relationship between average space-time product and average response time in the asymptotic case.

The formula is:

Where R = response time (i.e. average amount of time in system state per interaction)

N = Number of interactive terminals. It is assumed that N is constant throughout the observation interval.

Y = Average space-time product (i.e. space-time product completed per job

M = Average amount of memory in use during the observation interval

Z = Average think time (i.e. average amount of think time per transition from think state to system state).

3. **The differences between the Forced Flow Law and the Residence Time Law from a systems perspective**

**Forced Flow Law**

Relates the system throughput to individual device throughputs.

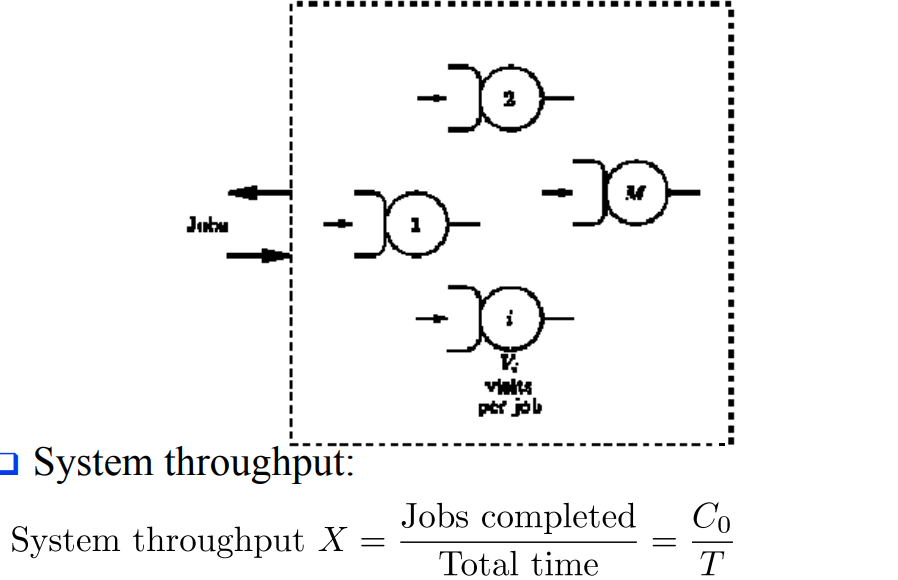
In an open model, System throughput = # of jobs leaving the system per unit time

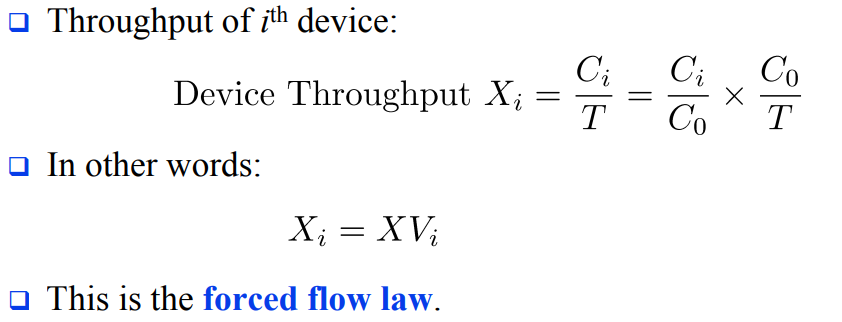
In a closed model, System throughput = # of jobs traversing OUT to IN link per unit time.

If observation period T is such that Ai = Ci⇒ Device satisfies the assumption of job flow balance.

Each job makes Vi requests for ith device in the system

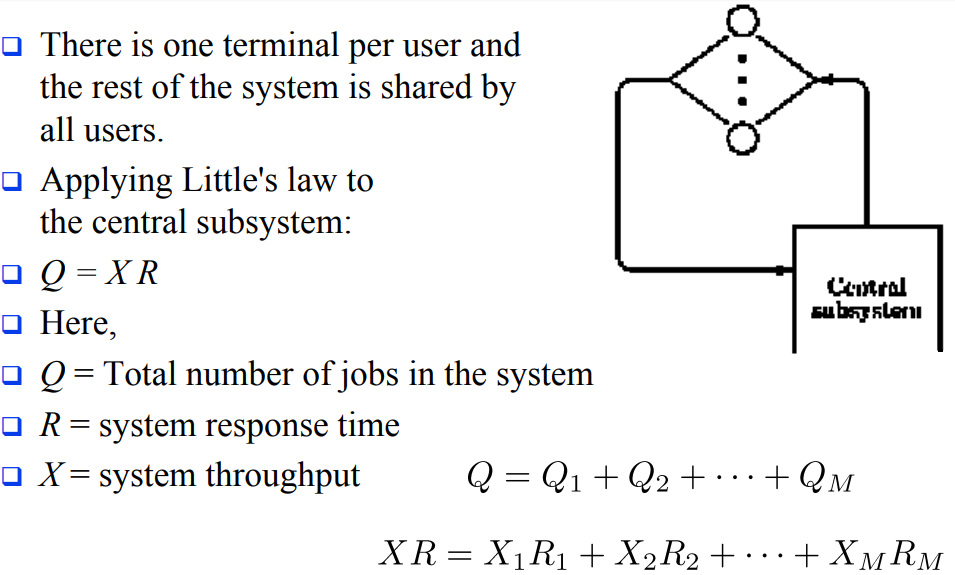
Ci = C0 Vi or Vi =Ci/C0 Vi is called visit ratio

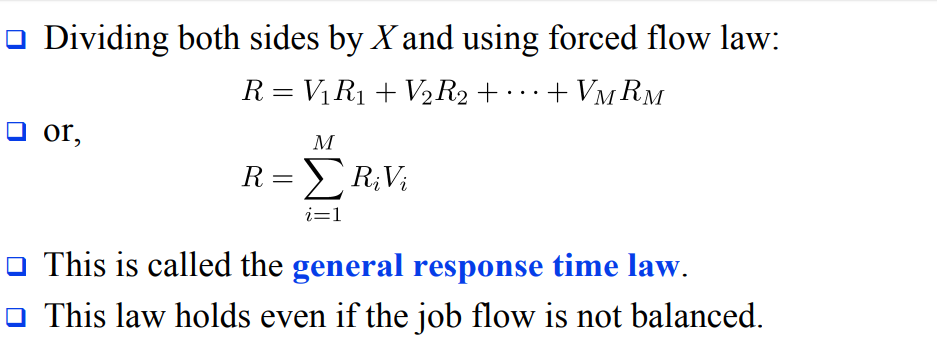




**WHILE**

**Residence Time Law**





**4. QUEUING MODELS**

**M/M/1 queue** represents the queue length in a system having a single server, where arrivals are determined by a [Poisson process](https://en.wikipedia.org/wiki/Poisson_process) and job service times have an [exponential distribution](https://en.wikipedia.org/wiki/Exponential_distribution). The model name is written in [Kendall's notation](https://en.wikipedia.org/wiki/Kendall%27s_notation). The model is the most elementary of queuing models and an attractive object of study as [closed-form expressions](https://en.wikipedia.org/wiki/Closed-form_expression) can be obtained for many metrics of interest in this model. An extension of this model with more than one server is the [M/M/c queue](https://en.wikipedia.org/wiki/M/M/c_queue)

* The (M/M/1) system This is a queuing model in which the arrival is Marcovian and departure distribution is also Marcovian, number of server is one and size of the queue is also Marcovian, no. of server is one and size of the queue is infinite and service discipline is 1st come 1st serve (FCFS) and the calling source is also finite.
* M/M/1 queue is a useful approximate model when service times have standard deviation approximately equal to their means.

**M/M/c/∞/∞ queue**: c servers operating in parallel

* Arrival process is Poisson with rate λ
* Each server has an independent and identical exponential service time distribution, with mean 1/μ.
* To achieve statistical e q, (equilibrium, the offered load (λ/μ) y must satisfy λ/μ < c, where λ/ (cµ) = ρ is the server utilization.

**M/M/c queue** (or **Erlang–C model**) is a multi-server [queuing model](https://en.wikipedia.org/wiki/Queueing_model). In [Kendall's notation](https://en.wikipedia.org/wiki/Kendall%27s_notation) it describes a system where arrivals form a single queue and are governed by a [Poisson process](https://en.wikipedia.org/wiki/Poisson_process), there are *c* servers and job service times are exponentially distributed It is a generalisation of the [M/M/1 queue](https://en.wikipedia.org/wiki/M/M/1_queue) which considers only a single server. The model with infinitely many servers is the [M/M/∞ queue](https://en.wikipedia.org/wiki/M/M/%E2%88%9E_queue).’

**M/M/M/s/N/N queue**: here the arrival distribution of customers follows Poisson distribution and the distribution for service time follows exponential distribution with s number of parallel serves. The number of population and the queuing capacity is limited to N. this situation often happens in queuing for machine repair system where the number of population is equal to the number of machine = N’

**M/M/s/s/n queue**: Consider the loss system (no waiting places) in the case where the arrivals originate from a finite population of sources: the total number of customers is n.

* To be specific, think of the customers being telephone users.
* Assume that the time to the next call attempt by the customer, so called thinking time (idle time) of the customer obeys the distribution Exp (γ).
* Blocked calls are lost – does not lead to reattempts – starts a new thinking time: again, the time to the next attempt ∼ Exp (γ) – the holding time X ∼ Exp (µ)

**QUEUING DISCIPLINES**

Are listed as follows;

1. First come first services (FCFS)

2. First in First out (FIFO)

3. Last in First out (LIFO)

4. Service in Random order (SIRO)

**First Come First Serve (FCFS)** is an operating system scheduling algorithm that automatically executes queued requests and processes in order of their arrival. It is the easiest and simplest CPU scheduling algorithm. In this type of algorithm, processes which requests the CPU first get the CPU allocation first. This is managed with a FIFO queue. The full form of FCFS is First Come First Serve.

As the process enters the ready queue, its PCB (Process Control Block) is linked with the tail of the queue and, when the CPU becomes free, it should be assigned to the process at the beginning of the queue

## Characteristics of FCFS method

* It supports non-pre-emptive and pre-emptive scheduling algorithm.
* Jobs are always executed on a first-come, first-serve basis.
* It is easy to implement and use.
* This method is poor in performance, and the general wait time is quite high.

## Example of FCFS scheduling

A real-life example of the FCFS method is buying a movie ticket on the ticket counter. In this scheduling algorithm, a person is served according to the queue manner. The person who arrives first in the queue first buys the ticket and then the next one. This will continue until the last person in the queue purchases the ticket. Using this algorithm, the CPU process works in a similar manner.

**First in First out (FIFO)**

Stands for "First In, First Out." FIFO is a method of processing and retrieving data. In a FIFO system, the first items entered are the first ones to be removed. In other words, the items are removed in the same order they are entered.

To use a real world analogy, imagine a vending machine where the items are loaded from the back. When someone selects a Milky Way bar from row E5, the machine churns out the candy bar closest to the front. The next Milky Way in line then moves to the front. Therefore, using the FIFO method, the candy bars are dispensed in the order they were placed in the machine.

Computers often implement the FIFO system when extracting data from an array or [buffer](https://techterms.com/definition/buffer). If the first data entered into the buffer must be extracted first, the FIFO method is used. The opposite of FIFO is [LIFO](https://techterms.com/definition/lifo), in which the last data entered is the first to be removed.

**Last in First out (LIFO)**

Stands for "Last In, First Out." LIFO is a method of processing data in which the last items entered are the first to be removed. This is the opposite of LIFO is [FIFO](https://techterms.com/definition/fifo) (First In, First Out), in which items are removed in the order they have been entered.

To better understand LIFO, imagine stacking a deck of cards by placing one card on top of the other, starting from the bottom. Once the deck has been fully stacked, you begin to remove the cards, starting from the top. This process is an example of the LIFO method, because the last cards to be placed on the deck are the first ones to be removed.

The LIFO method is sometimes used by computers when extracting data from an array or data [buffer](https://techterms.com/definition/buffer). When a program needs to access the most recent information entered, it will use the LIFO method. When information needs to be retrieved in the order it was entered, the FIFO method is used.

**Service in Random order (SIRO)**

When the system chooses one of the data to execute at random. Such a system is known as **service in random order (SIRO).**

**5.** Discuss how to resolve some basic queuing problems.

***Answer***

Firstly, queuing problems are the problems that occur when the service rendered doesn’t match the level of demand i.e. waiting for some time till the service is rendered to a customer. These customers might be humans at a bank on a queue or even airplanes ready to take off/ land or jobs waiting to be processed. An example could be when a supermarket doesn't have enough cashiers on a busy morning or when requests reach a system faster than it can process them.

Generally, no two businesses’ queuing problems are the same but some basic queuing problems that exist include; more customers entering the queues than leaving, when the queues are too long and strenuous, where queues are idle and not moving due to many queues and limited service providing stands to ease the queues etc.

We can then solve these various problems by:

1. **Let Customers Know How Long The Wait Is**: The uncertainty of how long it will take to wait is often the cause of queue anxiety. Because of this the customers are impatient and this is a major cause of queuing problems as people want to jump the queues or altogether leave the queue.
2. **Assess and improve your queue management strategy:** How do you currently handle a long line of customers? Think about what works well and what doesn’t. Assessing the tactics used to manage the queue in the particular organization will really help solve the queuing problems being encountered.
3. **Design Your Environment To Be Able To Accommodate Queues:** Studies have shown that one of the most common issues found in lines is queue anxiety. A well-designed queuing area can help organize waiting lines, remove the possibility of queue jumpers and generally ease customer flow management.
4. I**mplement Digital Queuing Software**: Long queues can inspire customer’s irritation even disgust. But anyone can learn how to reduce queues the use of a nifty technology called a queue management system (QMS).Automating the queuing process creates more labor efficient customer lines, decreases the overall amount of walkaways as well as ultimately reducing queue times. When it’s their turn, a teller calls them to the counter to be served. They can see where they are in line by observing HDTVs hung on the walls of the organization and therefore customers are free to sit or wander and maybe grab a coffee across the street as they wait. They’re not corralled into the line like sheep. By giving customers back their time (their autonomy) one enable customers to wait in leisure. Now that’s effective queue management.
5. **Occupy Customers in The Queue**: Boredom in the queue can often lead to longer perceived waiting times. Queue solutions is to provide a distraction to people in the queue and help them continue shopping while waiting, easing up frustrations etc. Display entertaining programming on HDTVs. Prompt customers to answer surveys to report on their experience. Engaging customers is the best way to reduce the tension inherent in queueing. Because it’s typically the psychology behind queueing rather than the queues themselves that makes queues feel unbearable.
6. **Keep The Rules Of Queuing Fair And Consistent:** One of the most important characteristics of any queue problem solving method is the queuing discipline used. Simply put, the queuing discipline is the rule used to decide who goes next in a queue.

Two of the most commonly used rules are:

* First in, First out.
* Last in, First out.

Bottom line, people expect queues to be fair. It’s not like they’re happy to be stuck waiting in line, to begin with. But when everyone abides by the same rules, we can’t help but follow them too.

1. **Reduce Response times:** So when it comes to providing service, be quick as possible. It's not possible to solve every problem immediately, but customers don’t expect that from you. What they do expect is that you give them some kind of response quickly. Having all information at your fingertips is the next step as these steps will help improve the flow of the queues and have less waiting times.

6. A project management infrastructure that would be needed to support a software development consulting team working at a client site

A project management infrastructure, consists of systems of policies, standards, procedures and guidelines that define how project management work is to be performed. From my research, I suggest that there are four key components that are part of a project management framework or infrastructure

1. Portfolio Management System

A Portfolio Management System ensures that the initiation of the project management process is grounded in sound strategic business decisions. A Portfolio Management System has five subsystems: a Solicitation Process (doing the right projects), a Selection Process (stopping the wrong ones), a Prioritization Process (doing them in the right order), a Registration Process (codifying them in a central repository), and an Enterprise Resource Planning Process (staffing them with the right people).

First, a Solicitation Process provides a consistent model for all proponents to follow; in other words, requestors of projects to follow. This model defines how a proponent prepares a business case that will be evaluated by the organization's business decision-makers. Then comes the Selection Process during which time the decision-makers approve those projects that add value to the organization and reject those projects that do not. After certain projects are approved, this same group of decision-makers prioritizes these projects relative to predefined business criteria, thus signifying those projects that will be given higher visibility and support and those that will not. Pertinent information such as project client, project scope, and team members is entered into a centralized database for all to access. In addition, these approved and prioritized projects are staffed (or resourced) relative to all the projects within the portfolio mix and relative to where the project sits within the prioritization ranking.

This part of the infrastructure allows the enterprise to manage the inventory of projects within the enterprise.

1. Process Management System

A Process Management System takes the approved and prioritized project through the Definition, Planning, Execution/Control, and Closeout phases.

The approved project from the Portfolio Management System goes into the Definition phase, which creates a project charter. The project charter becomes the input to the Planning phase, which creates a work plan; that is, schedule, staffing plan, project budget, and so on. The charter and the work plan then become the baseline in the Execution/Control phase of the project process. During this phase, the project team creates status reports and product deliverables. Once the project is over, these outputs from the execution/control phase are the input into the Closeout phase from which lessons learned are documented and archived for reference when starting the project management process all over again.

Various auxiliary processes such as a risk management process, a change management process, a quality assurance and control process, and a vendor/ contractor management process augment the above “core” process.

This component of the infrastructure ensures that the discipline of project management is performed in a consistent and professional manner throughout the entire organization.

1. Organizational Management System

An Organizational Management System is the governance structure defining roles, responsibilities, and authorities and reporting relationships.

From almost the beginning of project management, the applied organization structure that supported a project environment was a matrix structure. A matrix structure consists of representatives from various functional areas working together in an ad hoc team to accomplish certain business objectives producing specified deliverables. These cross-functioning teams work within the constraints of multiple bosses and often multiple priorities; however, they create a better and more “acceptable” product because of everyone's involvement in the project effort.

Today the “Project Office” is the newest version of the matrix project organizational structure. This autonomous department, staffed by project management subject matter experts, becomes the focal point for the project management discipline. As time evolves, the project office gains credibility, builds expertise, grows in self-confidence, and simultaneously increases its responsibility within the organization.

The organization platform of the infrastructure indicates the political interactions among departments and among people within the project community.

1. Performance Management System

A Performance Management System supports the three systems described above. This process sets project management performance objectives for project managers and for project team members and sees that these folks are rewarded for their successes and given development plans to improve their areas of deficiencies. The Performance Management System consists of a performance improvement process in which performance expectations and personal developmental plans are established and agreed upon.

During the appraisal review cycle, typically of 12 months, project managers have interim dialogues with their functional managers, with input from the project client. At the same time, project team members are having interim dialogues with their functional managers, with input from their project managers. The interim dialogues focus on whether or not project players are attaining their performance objectives and whether they are working toward their developmental plan. If they are not, the objectives or the plans need to be changed or the project players need to readdress themselves to these commitments.

As the performance improvement process comes to a close, the performance appraisal review process takes over. In this process, the functional manager of the project player prepares an official review document, with final input from the appropriate project client or project manager. The functional manager then executes the performance appraisal, and the cycle begins all over again.

This piece of the infrastructure sees that the people are guided, directed and rewarded.

The concept of learning cycles to briefly explain how project teams should work in a massive IT project to avoid conflicts

**1.** **Concrete Experience** - a new experience or situation is encountered, or a reinterpretation of existing experience.

**2.** **Reflective Observation of the New Experience** - of particular importance are any inconsistencies between experience and understanding.

**3.** **Abstract Conceptualization** reflection gives rise to a new idea, or a modification of an existing abstract concept (the person has learned from their experience).

**4.** **Active Experimentation** - the learner applies their idea(s) to the world around them to see what happens.

The relationship that exist between Project Life Cycle (PLC) and Software Development Life Cycle (SDLC)

The project life cycle (PLC) focuses on the phases, processes, tools, knowledge and skills of managing a project, while the system development life cycle (SDLC) focuses on creating and implementing the project’s product – the information system. How a project team chooses to implement the SDLC will directly affect how the project is planned in terms of phases, tasks, estimates and resources assigned. The SDLC is really part of the PLC because many of the activities for developing the information system occur during the execution phase. The last two stages of the PLC, closing and evaluating the project, occur after the implementation of the information system. The integration of project management and system development activities is one important component that distinguishes IT projects from other types of projects.