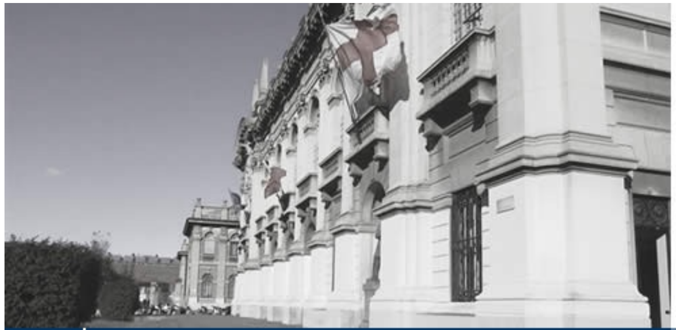


Computing Infrastructures

POLITECNICO DI MILANO





Operational Laws

Prof. Gianluca Palermo

Credits: Hilston, Marzolla, Mirandola, Gribaudo, Zahorian, Lazowska, Ardagna

The topics of the course: what are we going to see today?

HW Infrastructures:

System-level: Computing Infrastructures and Data Center Architectures, Rack/Structure;

Node-level: Server (computation, HW accelerators), Storage (Type, technology), Networking (architecture and technology);

Building-level: Cooling systems, power supply, failure recovery

SW Infrastructures:

Virtualization: Process/System VM, Virtualization Mechanisms (Hypervisor, Para/Full virtualization)

Computing Architectures: Cloud Computing (types, characteristics), Edge/Fog Computing, X-as-a service

Methods:

Reliability and availability of datacenters (definition, fundamental laws, RBDs)

Disk performance (Type, Performance, RAID)

Scalability and performance of datacenters (definitions, fundamental laws, queuing network theory)

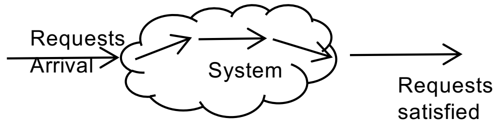
Operational laws

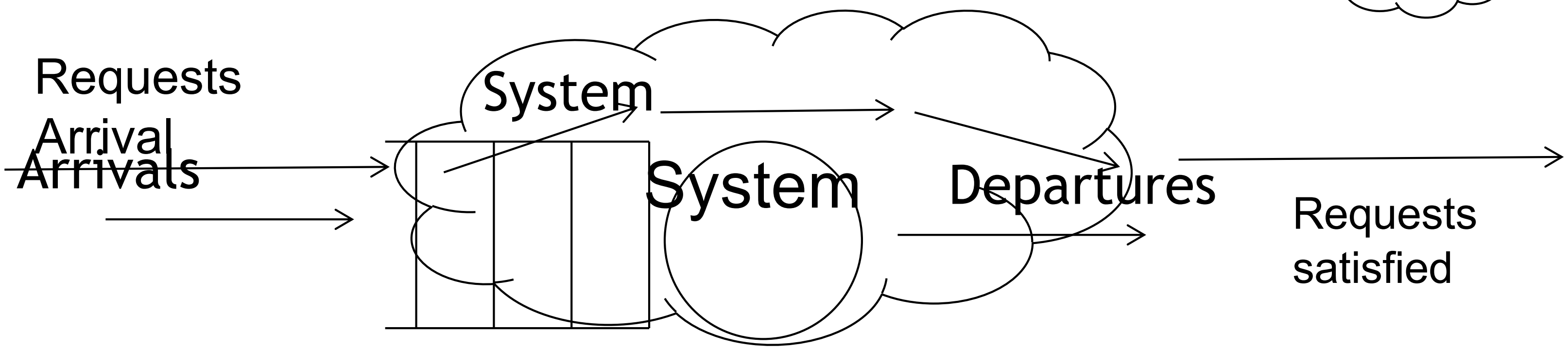
- Operational laws are simple equations which may be used as an abstract representation or model of the average behaviour of almost any system

- The laws are very general and make almost no assumptions about the behaviour of the random variables characterising the system

- Another advantage of the laws is their simplicity: this means that they can be applied quickly and easily

Operational laws



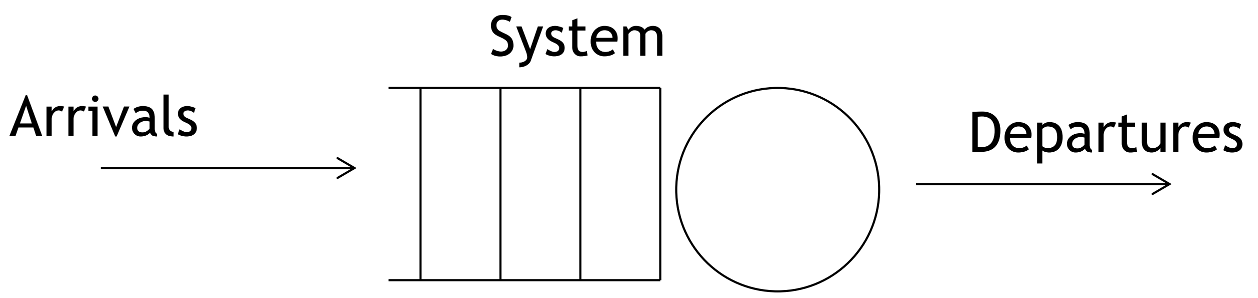


- Operational laws are based on observable variables - values which we could derive from watching a system over a finite period of time
- We assume that the system receives requests from its environment
- Each request generates a job or customer within the system
- When the job has been processed the system responds to the environment with the completion of the corresponding request

Observations and measurements

If we observed such an abstract system we might measure the following quantities:

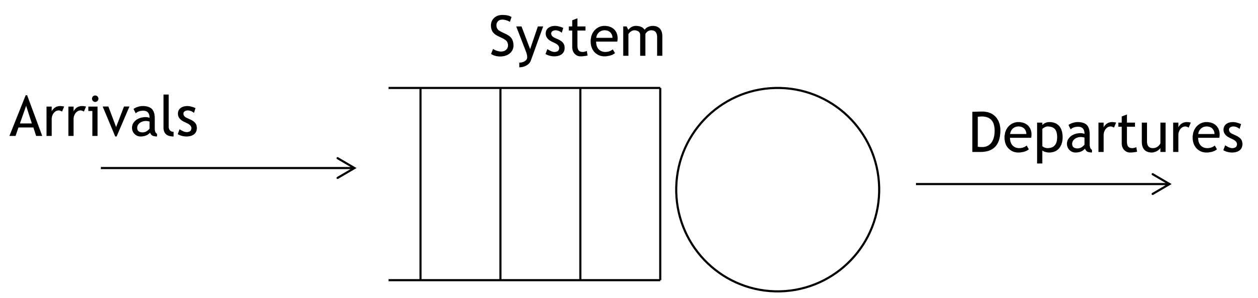
- T, the length of time we observe the system
- A, the number of request arrivals we observe
- C, the number of request completions we observe
- B, the total amount of time during which the system is busy ($B \leq T$)
- N, the average number of jobs in the system



Four important quantities

From these observed values we can derive the following four important quantities:

- $\lambda = A/T$, the arrival rate
- $X = C\ /T$, the throughput or completion rate
- $U = B/T$, the utilisation
- $S = B/C$, the mean service time per completed job

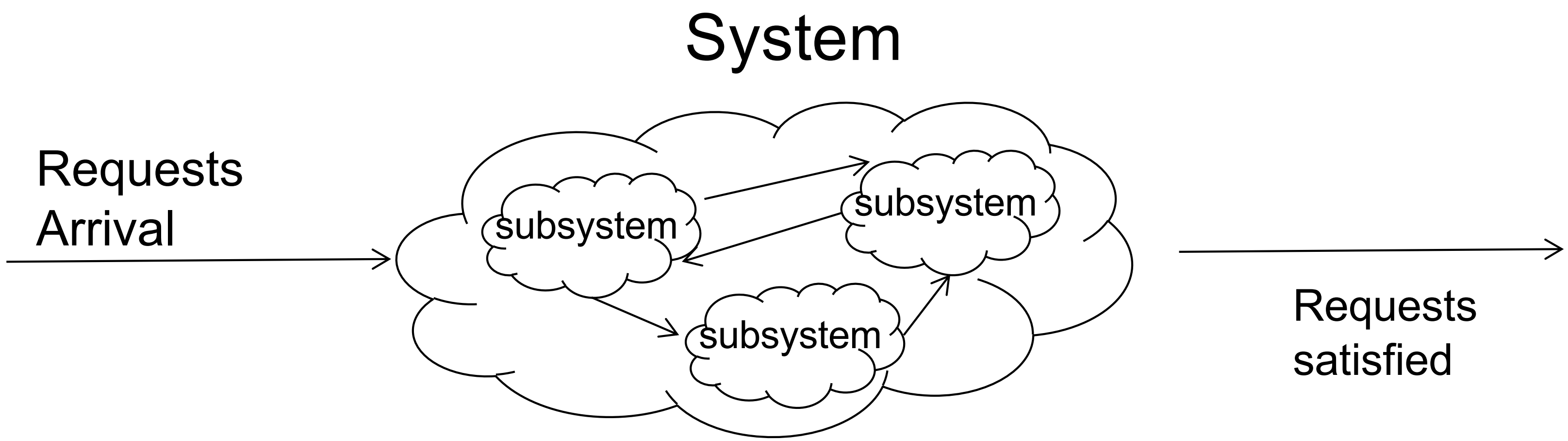


Job flow balance

- We will assume that the system is job flow balanced.
  - The number of arrivals is equal to the number of completions during an observation period, i.e. $A = C$
- This is a testable assumption because an analyst can always test whether the assumption holds
  - it can be strictly satisfied by careful choice of measurement interval
- Note that if the system is job flow balanced the arrival rate will be the same as the completion rate, that is:

$$\boxed{\lambda = X}$$

Operational laws

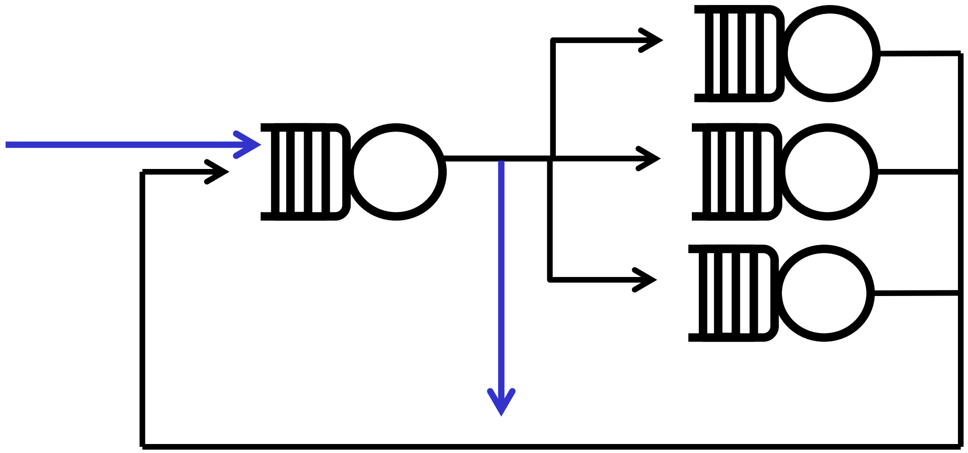


- A system may be regarded as being made up of a number of devices or resources

- Each of these may be treated as a system in its own right from the perspective of operational laws

- An external request generates a job within the system; this job may then circulate between the resources until all necessary processing has been done; as it arrives at each resource it is treated as a request, generating a job internal to that resource

Operational laws



If we observed such an abstract system we might measure the following quantities:

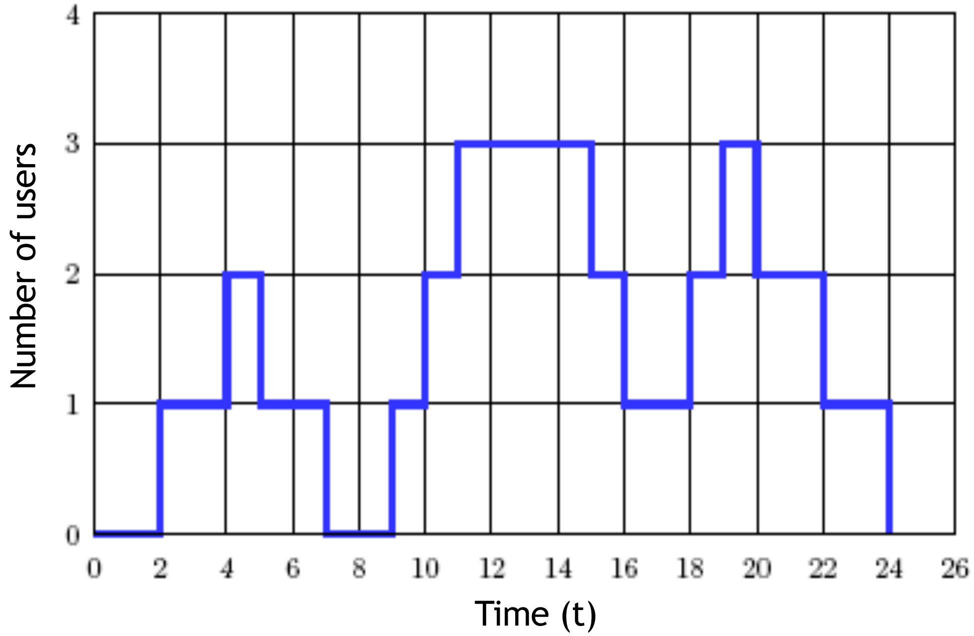
- T, the length of time we observe the system
- A_k, the number of request arrivals we observe for resource k
- C_k, the number of request completions we observe at resource k
- B_k, the total amount of time during which the resource k is busy ($B_k \leq T$)
- N_k, the average number of jobs in the resource k (queueing or being served)

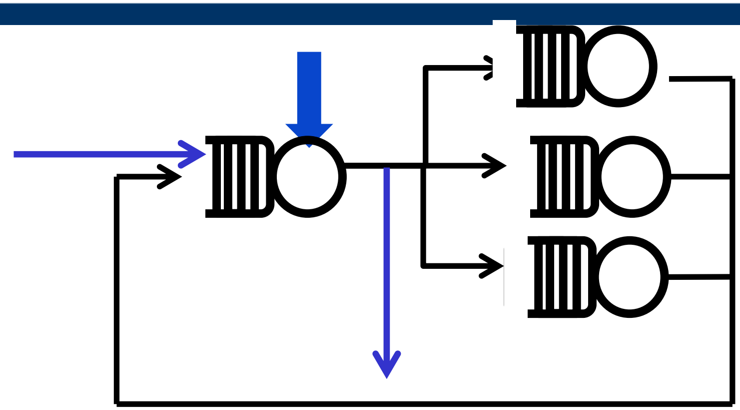
Four important quantities

From these observed values we can derive the following four important quantities for resource k:

- $\lambda_k = A_k/T$, the arrival rate
- $X_k = C_k\ /T$, the throughput or completion rate
- $U_k = B_k/T$, the utilisation
- $S_k = B_k/C_k$, the mean service time per completed job

Example





Let us observe the *k*-th resource and show in the graph the total number of users in k (both waiting for service and actually served)

$T = 26$ s

Arrivals number

$A_k = 7$

Completions number

$C_k = 7$

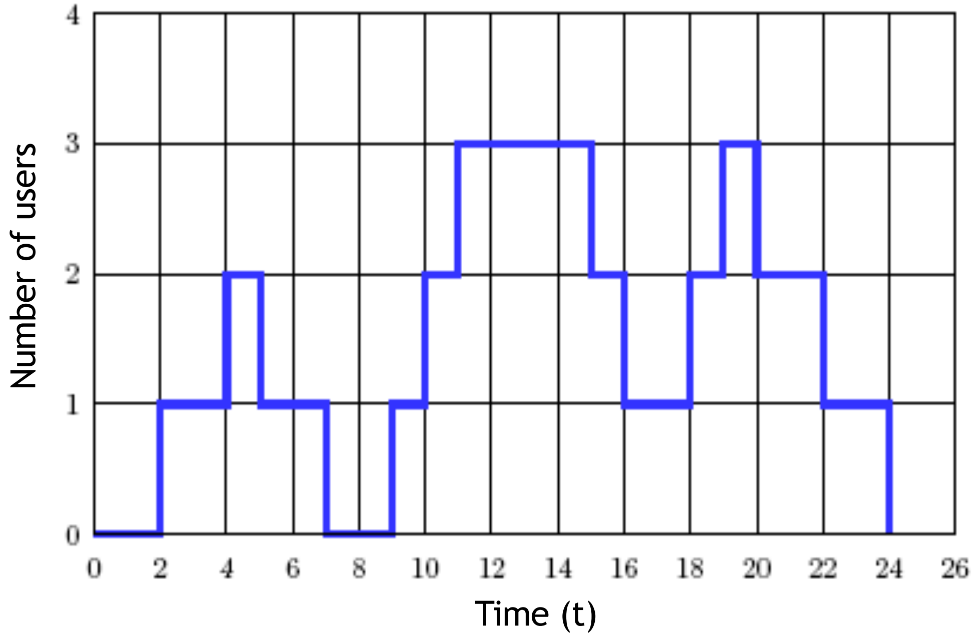
Arrival rate:

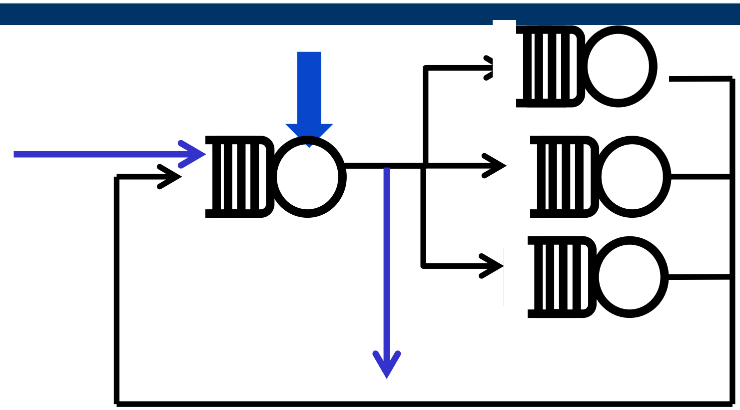
Throughput:

Utilization:

Avg. service time:

Example





Let us observe the *k*-th resource and show in the graph the total number of users in k (both waiting for service and actually served)

T = 26 s

Arrivals number

$A_k = 7$

Completions number

$C_k = 7$

Arrival rate: $\lambda_k = A_k/T = 7/26$ req/s

Throughput: $X_k = C_k/T = 7/26$ req/s

Utilization: $U_k = B_k/T = 20/26 = 0.77 = 77\%$

Avg. service time: $S_k = B_k/C_k = 20/7$ s

Utilization law

Let us recall the following definitions for a resource k:

Throughput: $X_k = C_k / T$

Service time $S_k = B_k / C_k$

Utilization: $U_k = B_k / T$

From:

$$X_k \quad = C_k / T$$

we can derive (utilization law):

$$\boxed{U_k = X_k S_k}$$

Utilization law

Let us recall the following definitions for a resource k:

Throughput: $X_k = C_k / T$

Service time $S_k = B_k / C_k$

Utilization: $U_k = B_k/T$

From:

$$X_k\, S_k = C_k / T \;\; * \; B_k / C_k = \mathrm{B}_k/\mathrm{T} = \mathrm{U}_k$$

we can derive (utilization law):

$$\boxed{U_k = X_k S_k}$$

Derive the utilization of the resource without an active monitor on it

Example: How to calculate the utilization

- Let us consider a resource k serving 40 requests/s, each of them requiring on average 0.0225 s

- From the utilization law we have:

$$U_k = X_k S_k:$$

Little Law

Little's law:

$$\boxed{N = XR}$$

Little law can be applied to the entire system as well as to some subsystems

N= Number of requests in the system

If the system throughput is *X* requests/sec, and each request remains in the system on average for R seconds, then for each unit of time, we can observe on average XR requests in the system

Intuitively: Similar to a pipeline seen in ACA

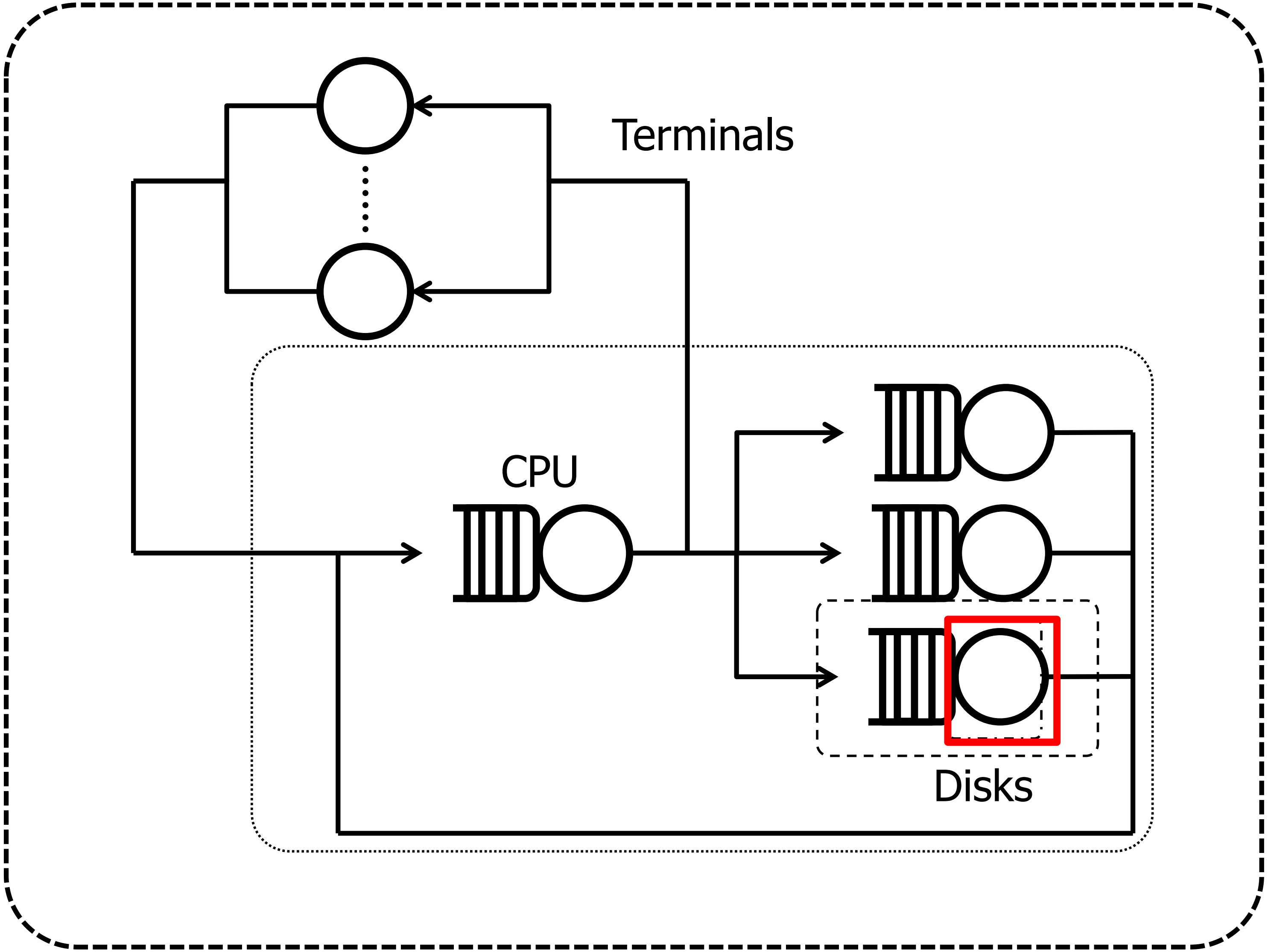
Example: Little Law N=XR

- Consider a disk that serves 40 requests/seconds (X = 40 req/s) and suppose that on average there are 4 requests (N = 4) present in the disk system (waiting to be served or in service)

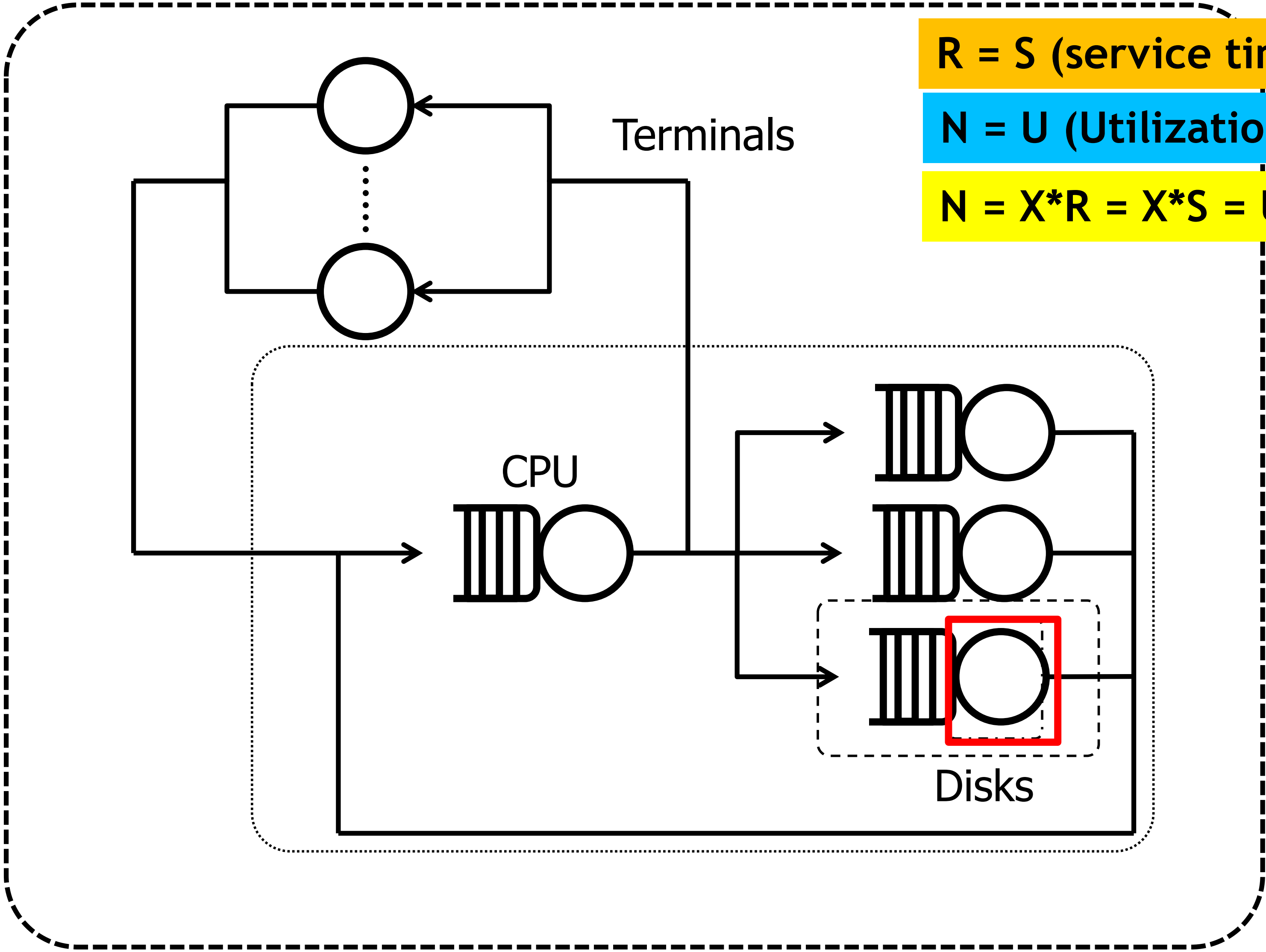
- Little's law tells us that R=N/X
  - the average time spent at the disk by a request must be 4/40 = 0.1 seconds

- If we know S:
  - (e.g.)Each request requires 0.0225 seconds of disk service
  - we can then deduce that the average waiting time (time in the queue) is ***0.0775 seconds***

Application of Little's Law at different levels



Application of Little's Law at different levels

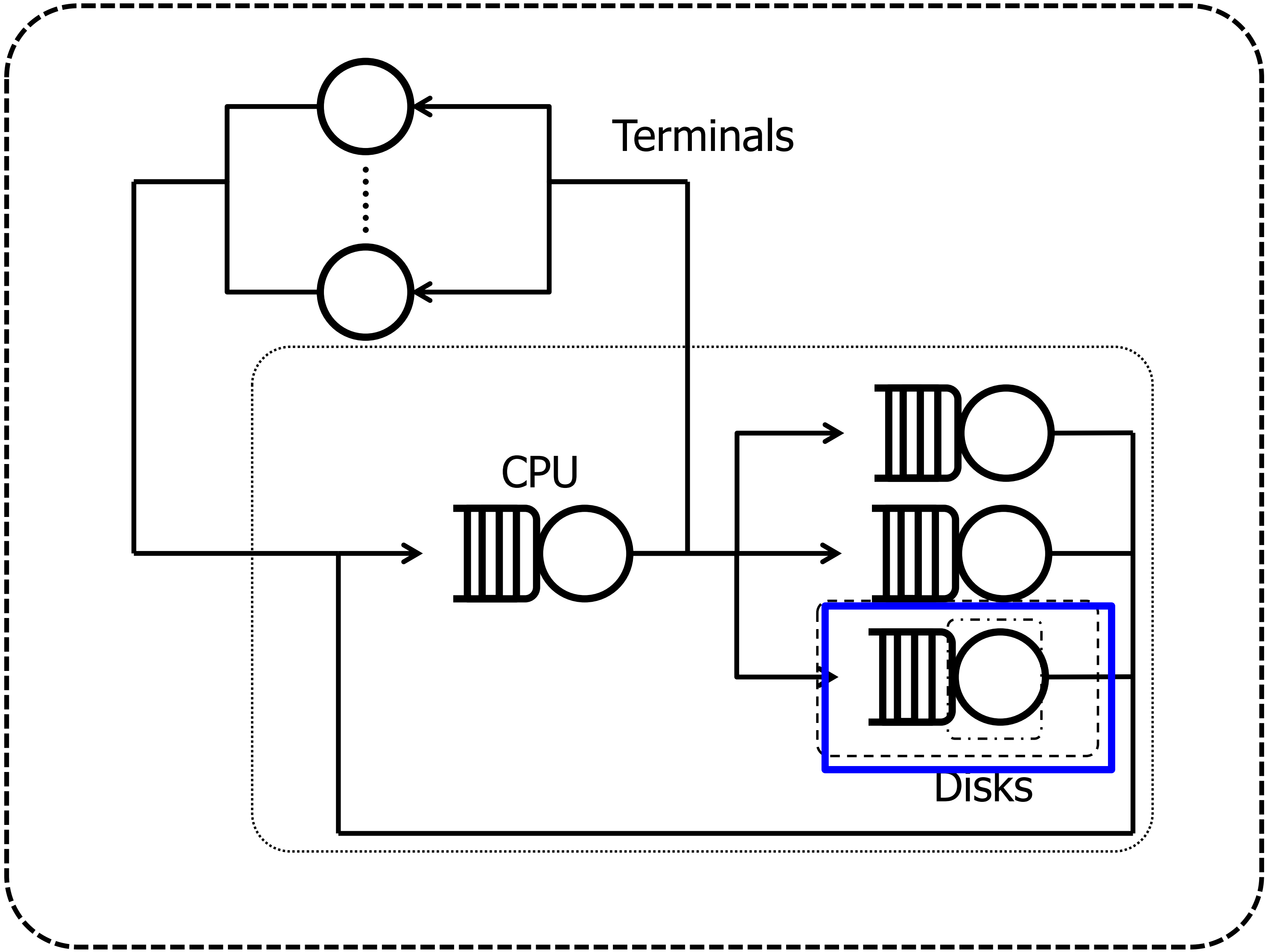


R = S (service time)

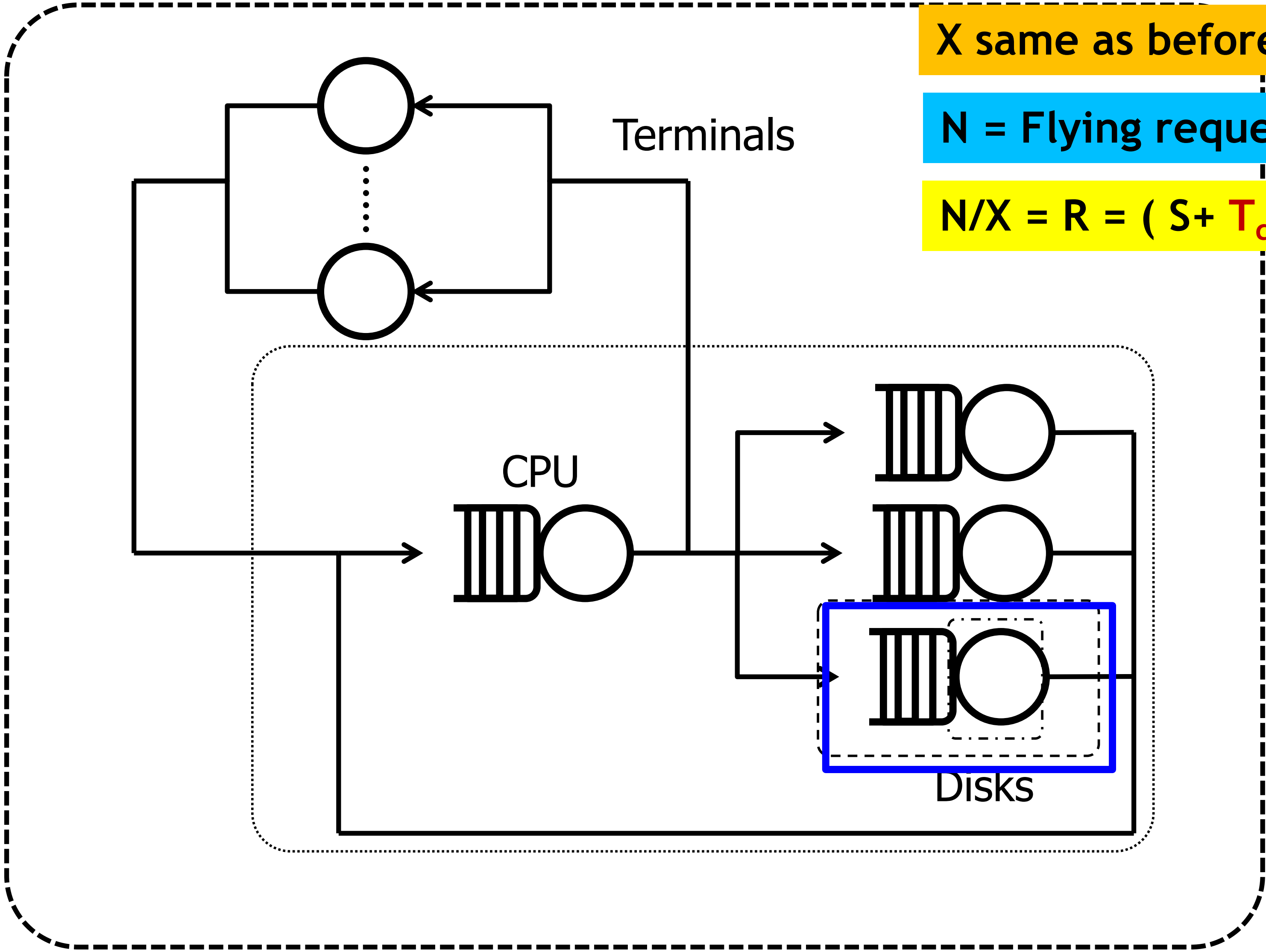
N = U (Utilization)

N = X*R = X*S = U

Application of Little's Law at different levels



Application of Little's Law at different levels

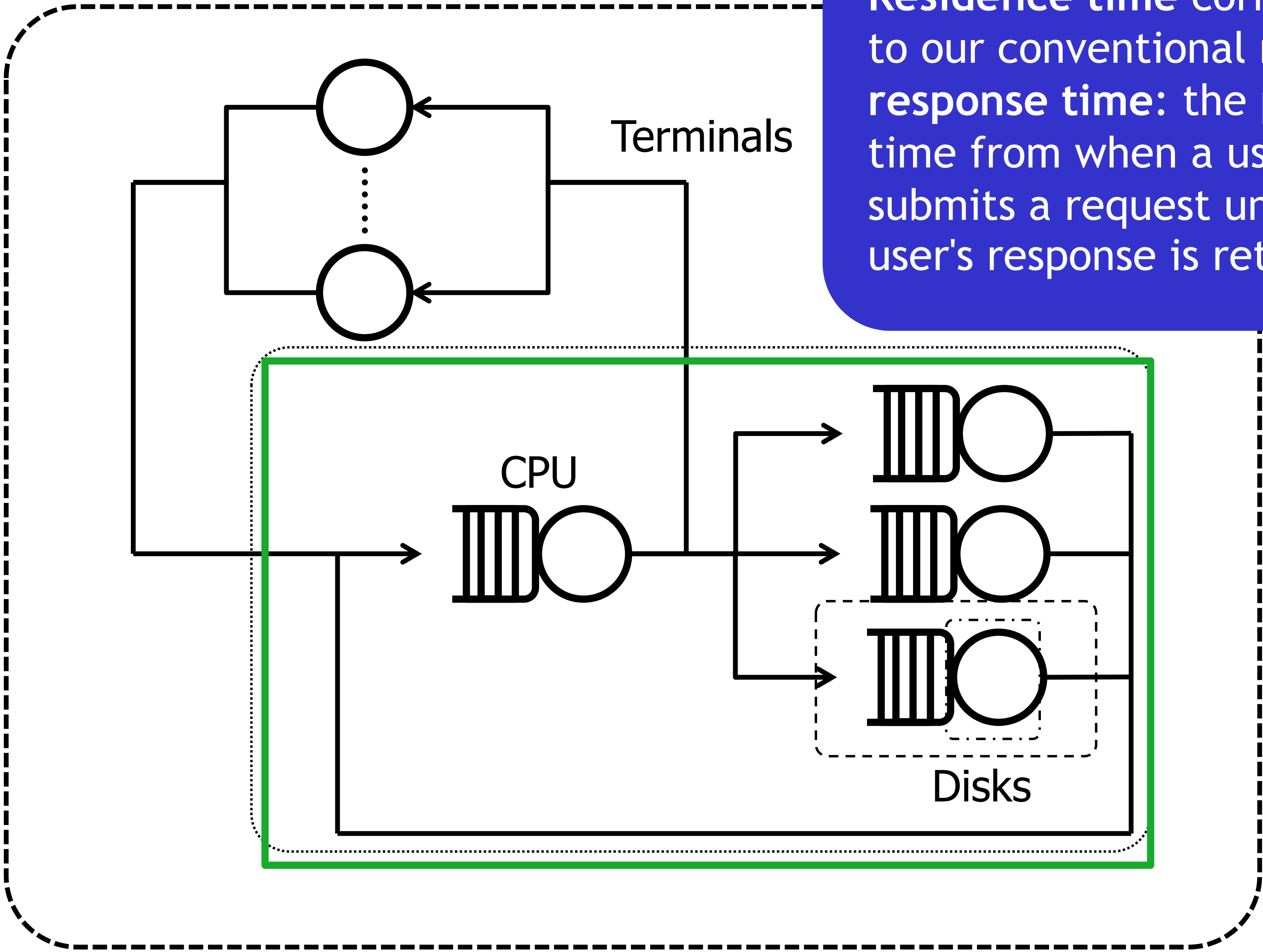


X same as before

N = Flying requests

$N/X = R = (S+ T_{queue})$

Application of Little's Law at different levels



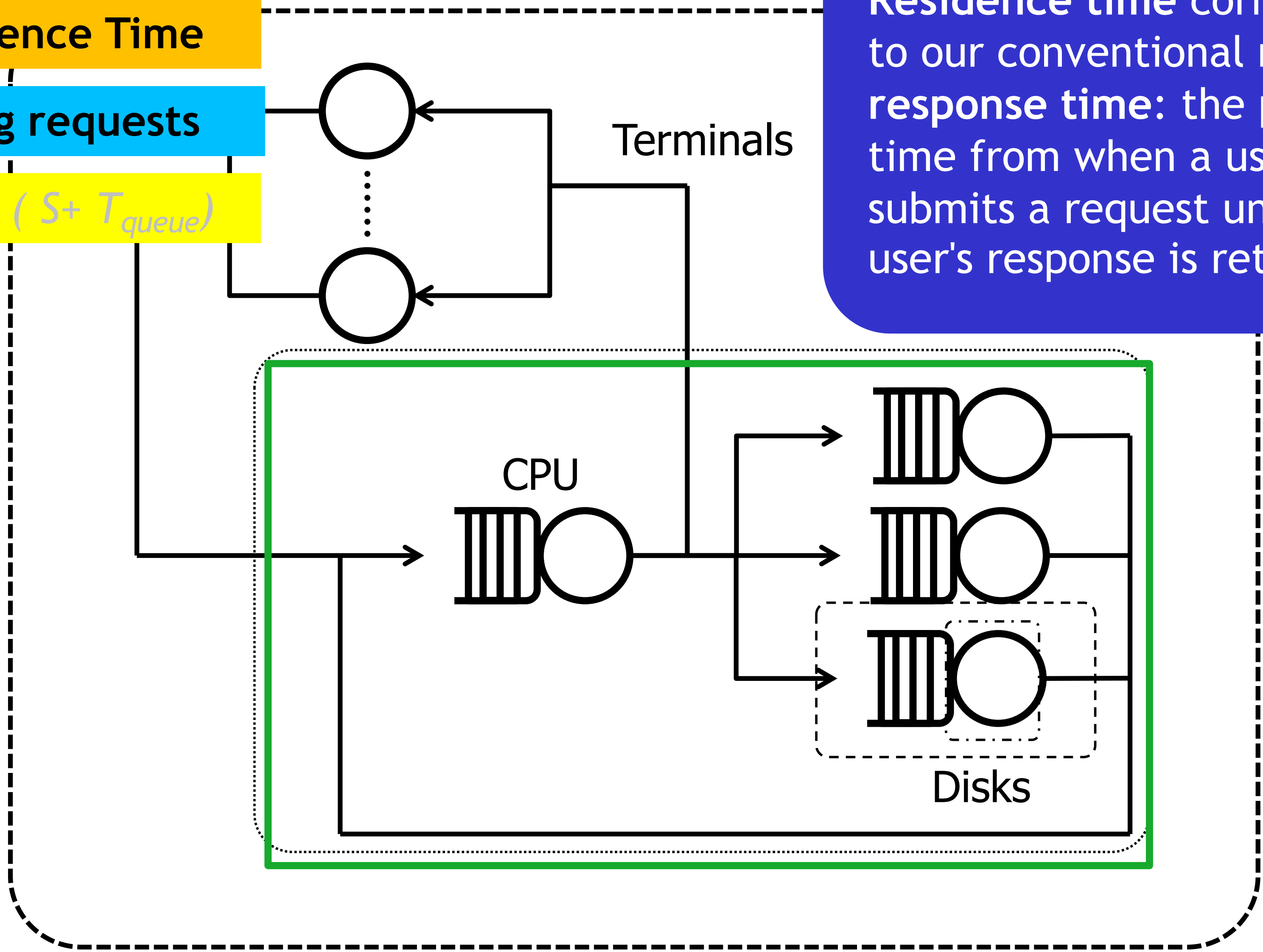
Residence time corresponds to our conventional notion of **response time:** the period of time from when a user submits a request until that user's response is returned

Application of Little's Law at different levels

R = Residence Time

N = Flying requests

$N/X = R$ $= (S + T_{queue})$



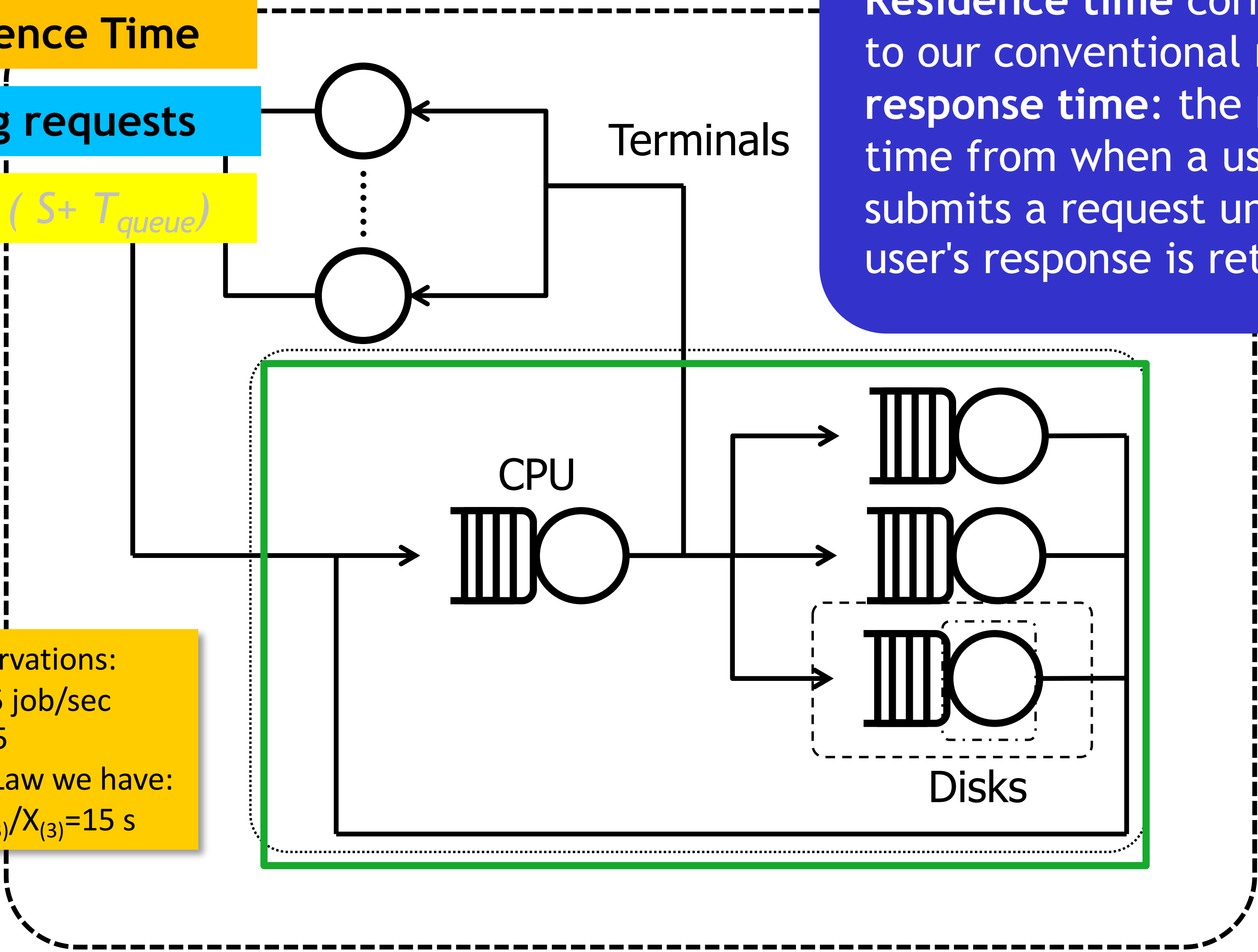
Residence time corresponds to our conventional notion of **response time**: the period of time from when a user submits a request until that user's response is returned

Application of Little's Law at different levels

R = Residence Time

N = Flying requests

N/X = R = ($S + T_{queue}$)



Residence time corresponds to our conventional notion of **response time**: the period of time from when a user submits a request until that user's response is returned

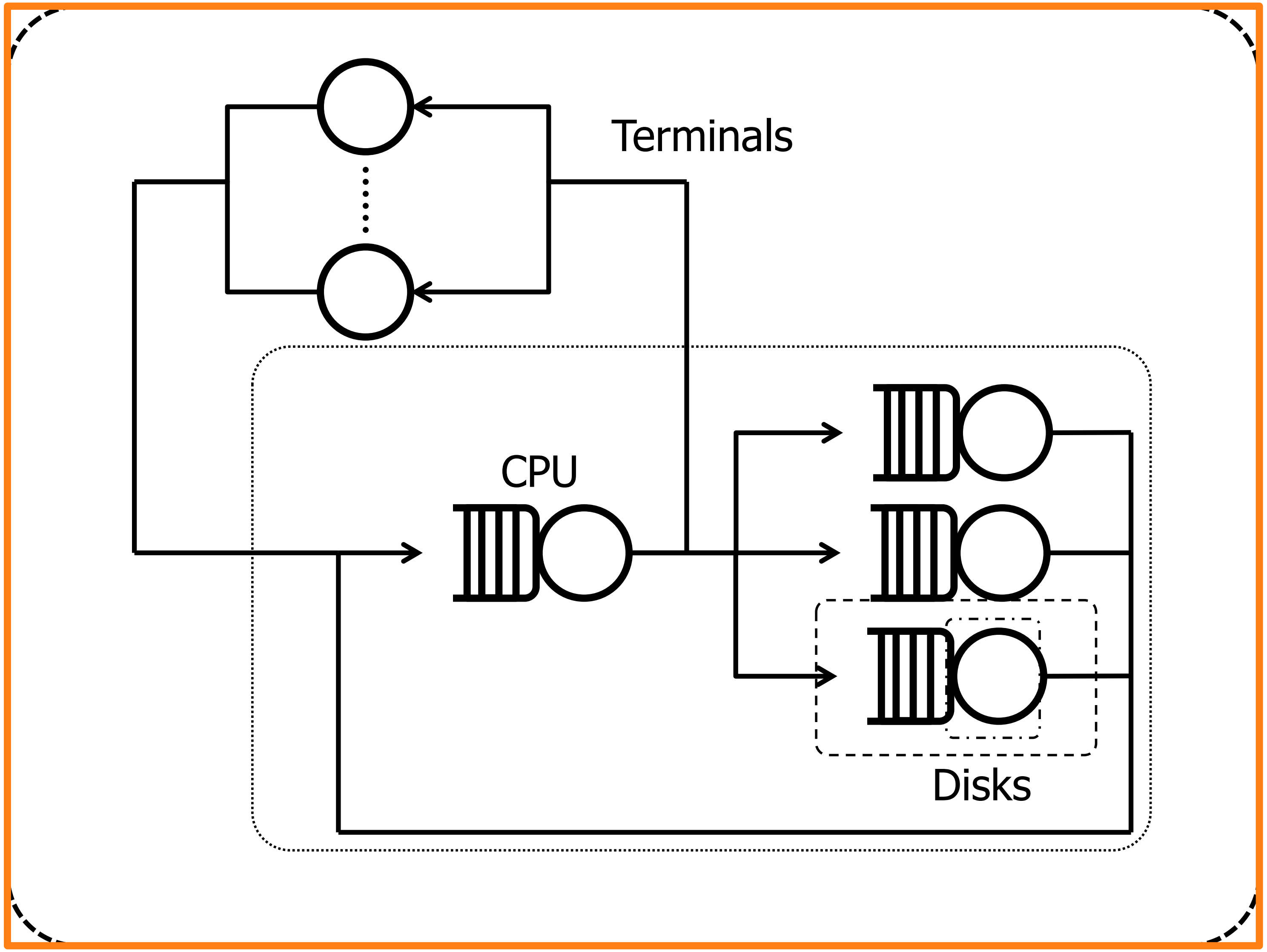
If from observations:

- $X_{(3)}$ = 0.5 job/sec
- $N_{(3)}$ = 7.5

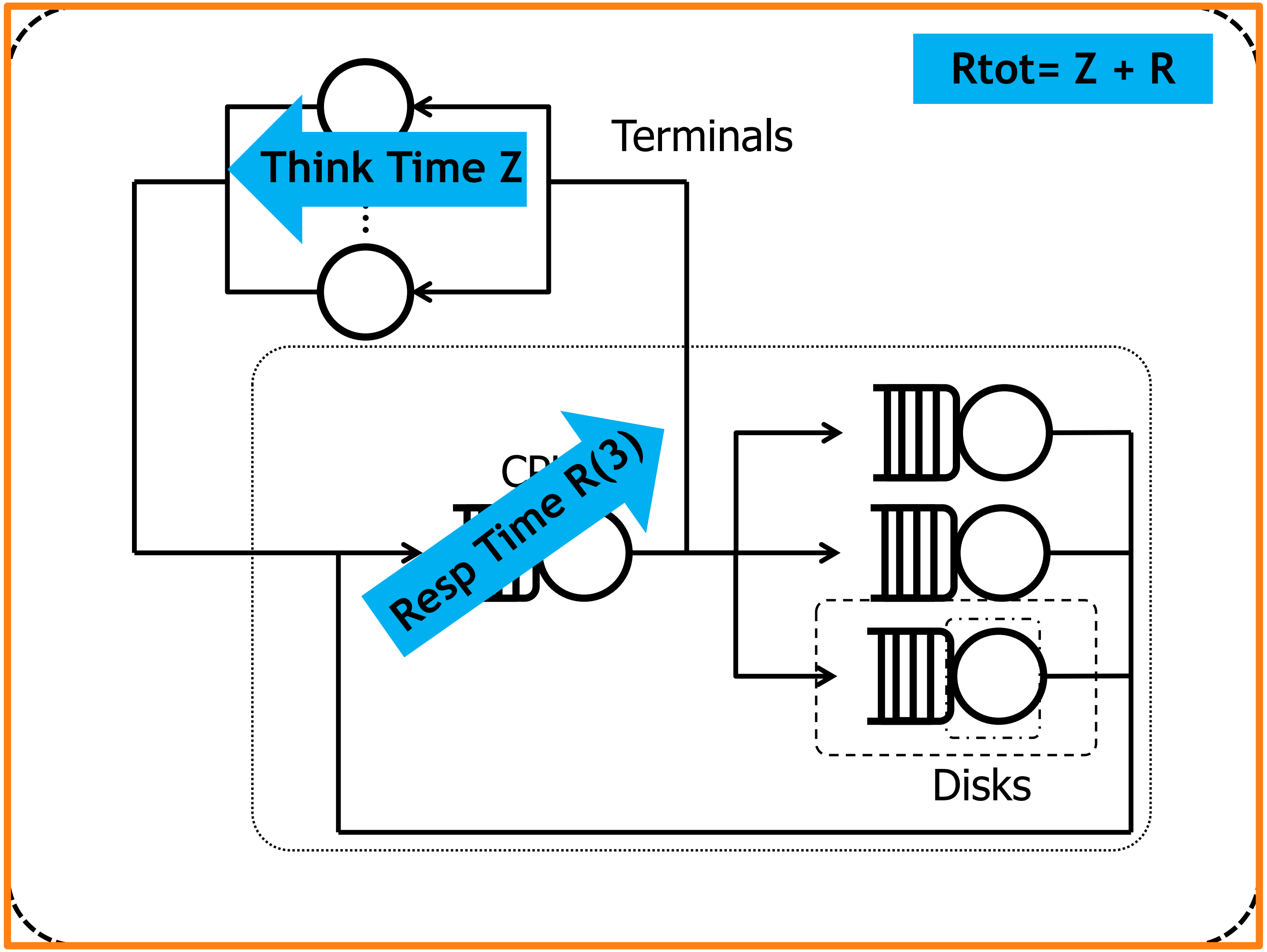
From Little-Law we have:

- $R_{(3)} = N_{(3)}/X_{(3)}$=15 s

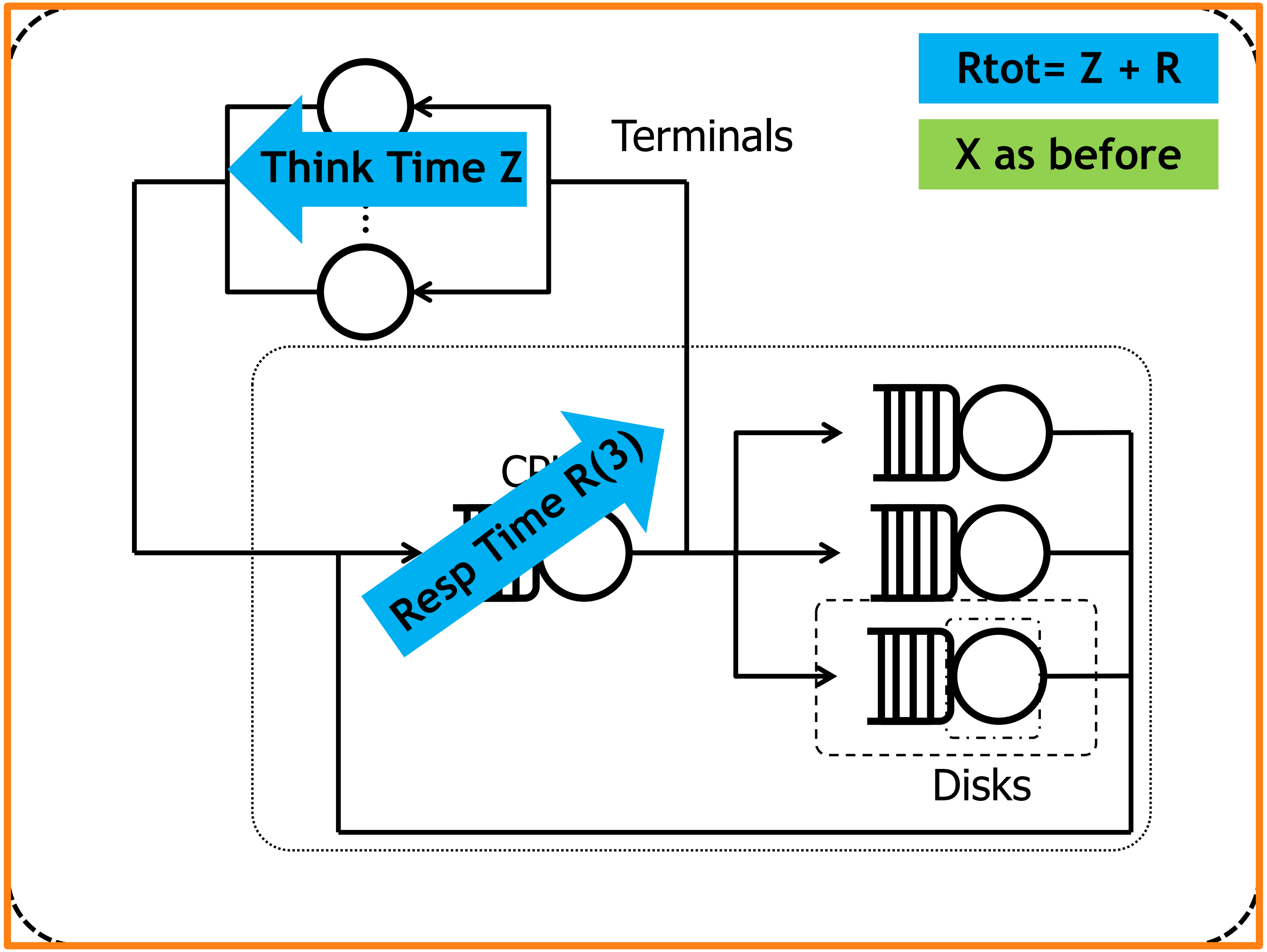
Application of Little's Law at different levels



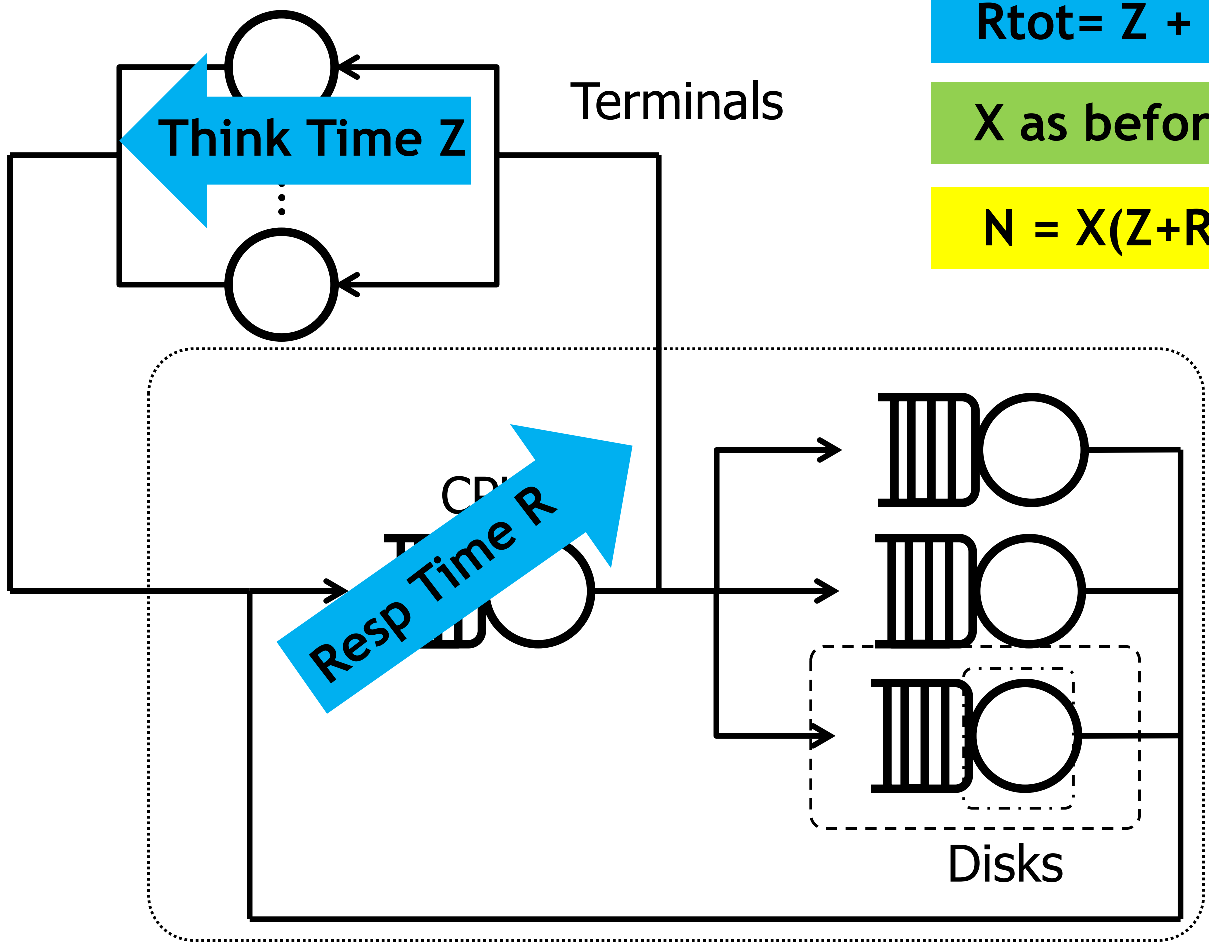
Application of Little's Law at different levels



Application of Little's Law at different levels



Application of Little's Law at different levels



Rtot= Z + R

X as before

N = X(Z+R)

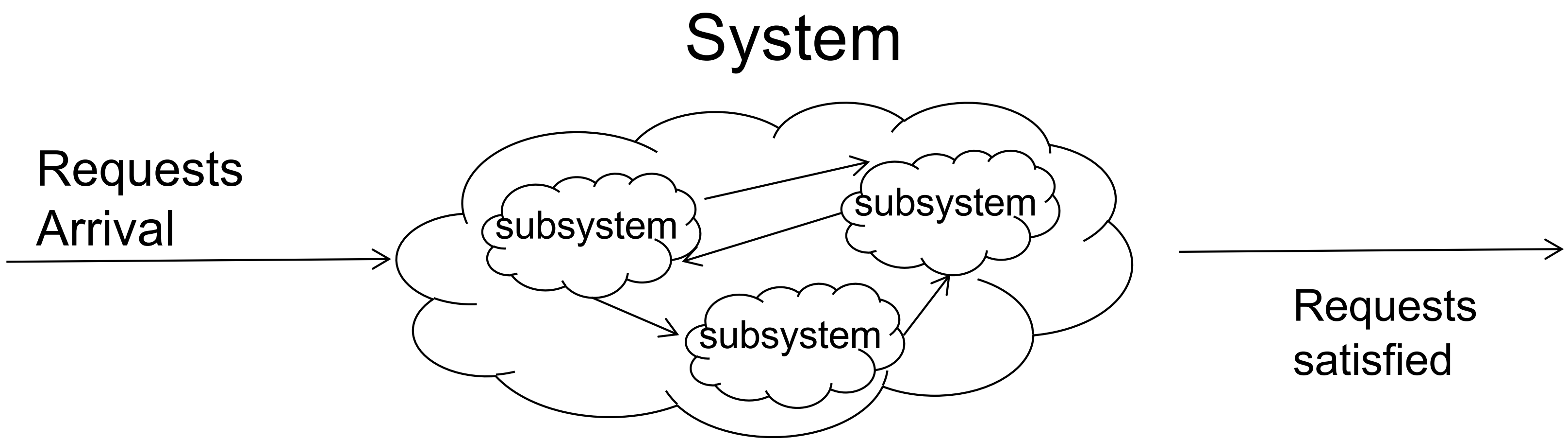
Interactive Response Time Law

- Interactive Response Time Law:

$$\boxed{R = N/X - Z}$$

- The response time in an interactive system is the residence time minus the think time
- Note that if the think time is zero, $Z = 0$ and $R = N/X$, then the interactive response time law simply becomes Little's Law
- **Example**:
  - Suppose that the library catalogue system has **64 interactive users** connected via Browsers, the average **think time is 30 seconds**, and that system **throughput is 2 interactions/second**. What is the response time?
    - The interactive response time law tells us that the response time must be $64/2 - 30 = 2$ seconds

Operational laws



- In an observation interval we can count not only completions external to the system, but also the number of completions at each resource within the system.
  - C_k is the number of completions at resource k
- **We define the visit count, $V_k = C_k/C$**
  - ratio of the number of completions at the k-th resource to the number of system completions

- For example, if, during an observation interval, we measure 10 system completions and 150 completions at a specific disk, then on average each system-level request requires 15 disk operations.

Visits

Note that:

- If $C_k > C$, resource k is visited several times (on average) during each system level request. This happens when there are loops in the model
- If $C_k < C$, resource k might not be visited during each system level request. This can happen if there are alternatives (i.e. caching of disks)
- If $C_k = C$, resource k is visited (on average) exactly once every request

Forced Flow Law

The forced flow law captures the relationship between the different components within a system. It states that the throughputs or flows, in all parts of a system must be proportional to one another.

$$\boxed{X_k = V_k X}$$

The throughput at the k-th resource is equal to the product of the throughput of the system and the visit count at that resource.

Rewriting $C_k = V_k C$ and applying $X_k = C_k / T$, we can derive the forced flow law:

$$C_k = V_k C$$

$$C_k / T = V_k C / T$$

$$X_k = V_k X$$

Utilisation Law - Service Demand

- If we know the amount of processing each job requires at a resource then we can calculate the utilisation of the resource

- To do so, let us assume that each time a job visits the k-th resource the amount of processing, or service time it requires is S_k

- Note that service time **is not** the same as the residence time of the job at that resource
  - in general a job might have to **wait** for some time before processing begin

- The total amount of service that a system job generates at the k-th resource is called the service demand, D_k:

$$D_k = S_k V_k$$

Utilisation Law

- The utilisation of a resource is denoted U_k
  - Percentage of time that the k-th resource is in use processing to a job

- Utilisation Law:

$$U_k = X_k S_k = (X V_k)\ S_k = D_k X$$

- The utilisation of a resource is equal to the product of:
  1. the throughput of that resource and the average service time at that resource,
  2. the throughput at system level and the average service demand at that resource

$$\boxed{U_k = D_k X}$$

e.g. I can derive the resource utilization without direct monitoring of it

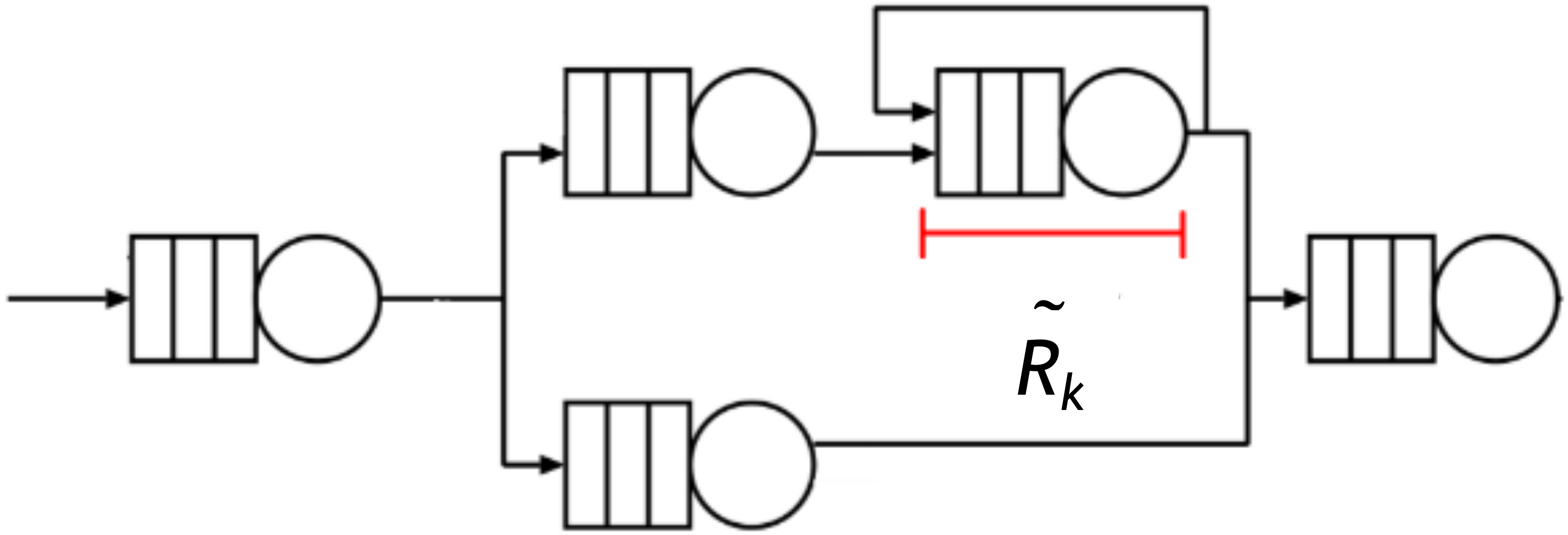
Response and Residence times

- When considering nodes characterized by visits different from one, we can define two permanence times:

  - *Response Time* $\tilde{R}_k$
  - *Residence Time* R_k

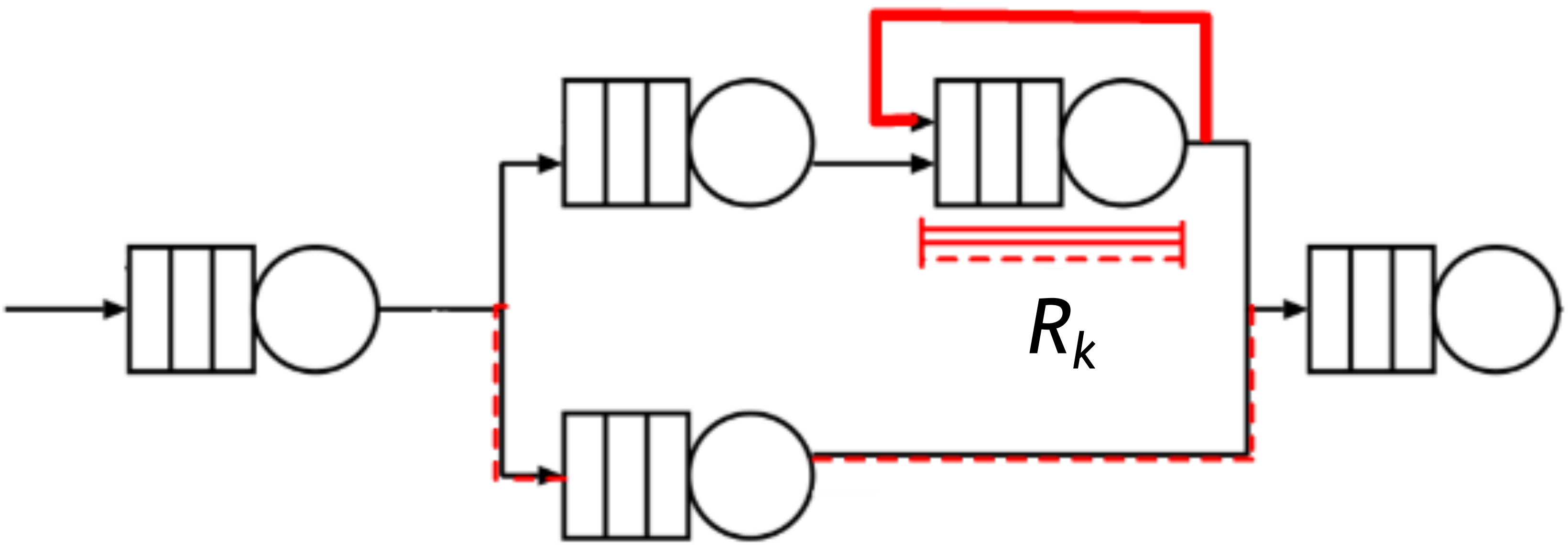
Response and Residence times

The *Response Time* $\tilde{R}_k$ accounts for the average time spent in station *k*, when the job enters the corresponding node (i.e., time for the single interaction, e.g. disk request):



Response and Residence times

The *Residence Time* R_k accounts instead for the average time spent by a job at station k during the staying in the system: it can be greater or smaller than the response time depending on the number of visits.



Response and Residence times

Note that there is the same relation between *Residence Time* and *Response Time* as the one between *Demand* and *Service Time*

$$D_k = v_k \cdot S_k$$
$$R_k = v_k \cdot \widetilde{\mathrm{R}}_k$$

Also note that for single queue open system, *or tandem models*, $v_k = 1$. This implies that average service time and service demand are equal, and response time and residence time are identical

$$v_k = 1 \quad \Rightarrow \quad \begin{array}{l} D_k = S_k \\ R_k = \widetilde{\mathrm{R}}_k \end{array}$$

Operational laws - Conclusions

- Operational laws are simple equations which may be used as an abstract representation or model of the average behaviour of almost any system

- The laws are very general and make almost no assumptions about the behaviour of the random variables characterising the system

- Another advantage of the laws is their simplicity: this means that they can be applied quickly and easily