Session 3
System Models

Course Leader: Jishmi Jos Choondal

Objectives

At the end of the lecture, the student will be able to

- Discuss the various models employed to design, develop and analyze Distributed Systems

Contents

- Physical Models
- Architectural models
- Fundamental models

System Models – Introduction

- In this session we bring out the common properties and design issues for distributed systems in the form of descriptive models.
- Each model is intended to provide an abstract, simplified but consistent description of a relevant aspect of DS design.

Design of Distributed Systems

- Physical models: consider the types of computers and devices that constitute a system and their interconnectivity without details of specific technologies
- Architectural Models: describe a system in terms of the computational and communication tasks performed by its computational elements
- Fundamental Models: take an abstract perspective in order to describe solutions to individual issues faced by most distributed systems

Physical Model

- A representation of the underlying hardware elements of a DS that abstracts away from the specific details of the computer and networking technologies
  - Baseline physical model
    - Coordination only by passing messages
- 3 generations of DS
  - Early DS: intranet ,10-100 nodes, homogeneous, local print, file server, email and file transfer
  - Internet-scale DS: internet, more nodes, more heterogeneity, middleware-CORBA, web services
  - Contemporary DS: mobile computing, ubiquitous computing, cloud computing

Generations of DS

Figure 2.1 Generations of distributed systems

Distributed systems:	*Early*	*Internet-scale*	*Contemporary*
Scale	Small	Large	Ultra-large
Heterogeneity	Limited (typically relatively homogenous configurations)	Significant in terms of platforms, languages and middleware	Added dimensions introduced including radically different styles of architecture
Openness	Not a priority	Significant priority with range of standards introduced	Major research challenge with existing standards not yet able to embrace complex systems
Quality of service	In its infancy	Significant priority with range of services introduced	Major research challenge with existing services not yet able to embrace complex systems

Architectural Models

- An Architectural model defines the way in which the components of systems interact with one another and the way in which they are maped onto an underlying network of computers.
- An architectural model of a distributed system first simplifies and abstracts the functions of the individual components of a DS and then it considers:
  - The placement of the components across a network of computers
  - The interrelationships between the components
- The initial simplification is achieved by classifying processes as server processes, client processes and peer processes
- The latter being the processes that cooperate and communicate in a symmetrical manner to perform a task. Example Client-Server Model and peer process model

Architectural Models

- Architecture of system is its structure in terms of separately specified components and their interrelationships
  - Architectural Elements
  - Architectural patterns
  - Associated middleware solutions

Architectural Elements

- What are the entities that are communicating in the distributed system?
- How do they communicate, or, more specifically, what *communication paradigm* is used?
- What (potentially changing) roles and responsibilities do they have in the overall architecture?
- How are they mapped on to the physical distributed infrastructure (what is their *placement*)?

Communicating Entities

- System Oriented entities: Processes
  - Sensor Network : Nodes
  - In DS: Processes, threads
- Problem Oriented Entities
  - Objects
  - Components
  - Web Services

Problem Oriented Entities

- Objects
  - To encourage the use of object oriented approaches in DS
  - Accessed via interfaces with an associated IDL providing specification of the methods defined on an object
- Components
  - Component technology offer problem oriented abstractions for building DS accessed through interfaces
  - Making all dependencies explicit and provide a more complete contract for system construction
- Web services
  - Approach based on encapsulation of behavior and access through interfaces

Communication Paradigms

- Interprocess communication
  - Message passing, Socket programming, multicast communication
- Remote invocation
  - Request-reply protocol, RPC, RMI
- Indirect communication
  - Group communication, publish-subscribe paradigm, message queues, Tuple spaces, Distributed shared memory

Communication Paradigms

Figure 2.2 Communicating entities and communication paradigms

Communicating entities (what is communicating)		*Communication paradigms (how they communicate)*		
System-oriented entities	*Problem-oriented entities*	*Interprocess communication*	*Remote invocation*	*Indirect communication*
Nodes	Objects	Message passing	Request-reply	Group communication
Processes	Components	Sockets	RPC	Publish-subscribe
	Web services	Multicast	RMI	Message queues
				Tuple spaces
				DSM

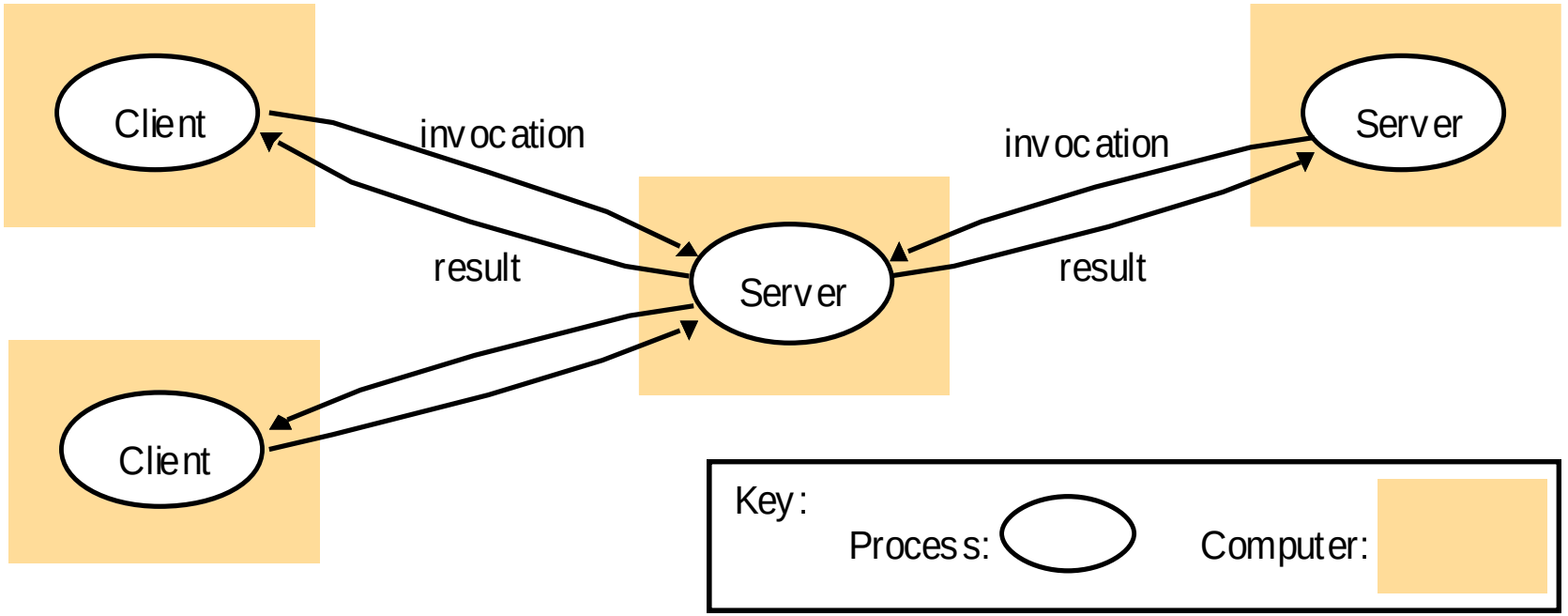
Roles and Responsibilities

- Communicating entities interact with each other to perform useful activity
- Roles of processes are fundamental in establishing the overall architecture to be adopted
- Two important architectural styles from the role of individual processes
  - Client Server
  - Peer to peer

Client Server Model

- It is the most often used and important model in the distributed Systems.
- Client processes interact with individual server processes in separate host computers in order to access the shared resources that they manage.
- Servers may in turn be clients of other servers.
- Example Web Server, Search engines.

Clients Invoke Individual Servers

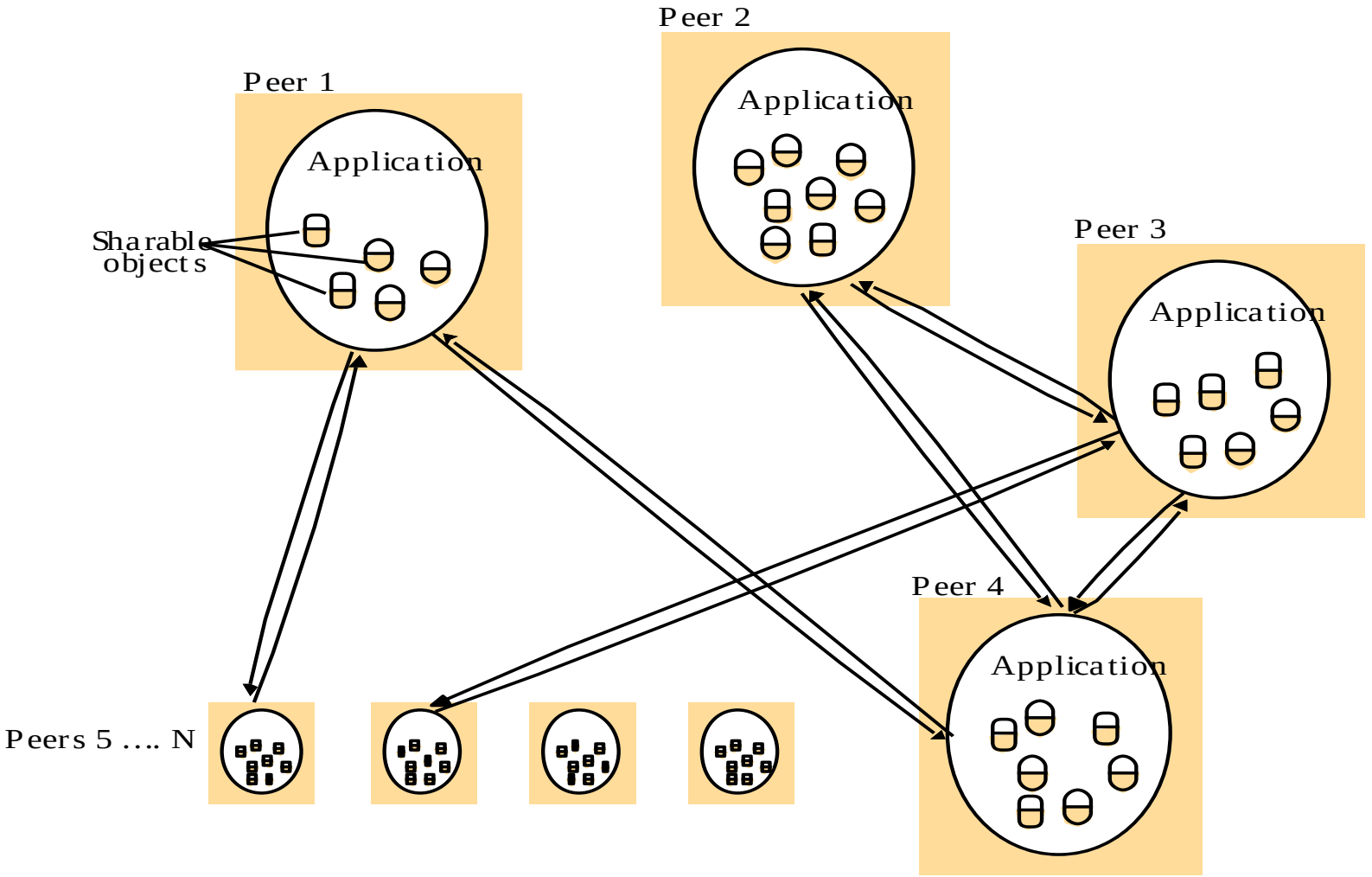


Peer to Peer Model

- In this architecture all of the processes involved in a task or activity play similar roles,
  - Interacting cooperatively as peers without any distinction between clients and server processes to perform a distributed activity or computation.

- The aim of this architecture is to exploit the resources both data and hardware in a large number of participating computers for the fulfilment of a given task.

A Distributed Application Based On Peer Processes



Peer to Peer Model

- Each object is replicated in several computers
  - To further distribute the load
  - To provide resilience in the event of disconnection of individual computers.

- This architecture substantially more complex than client server architecture because of
  - The need to place individual objects
  - The need to retrieve them
  - The need to maintain replicas amongst many computers

Placement of communicating Entity

- Mapping of services to multiple servers
- Caching
- Mobile code
- Mobile Agents

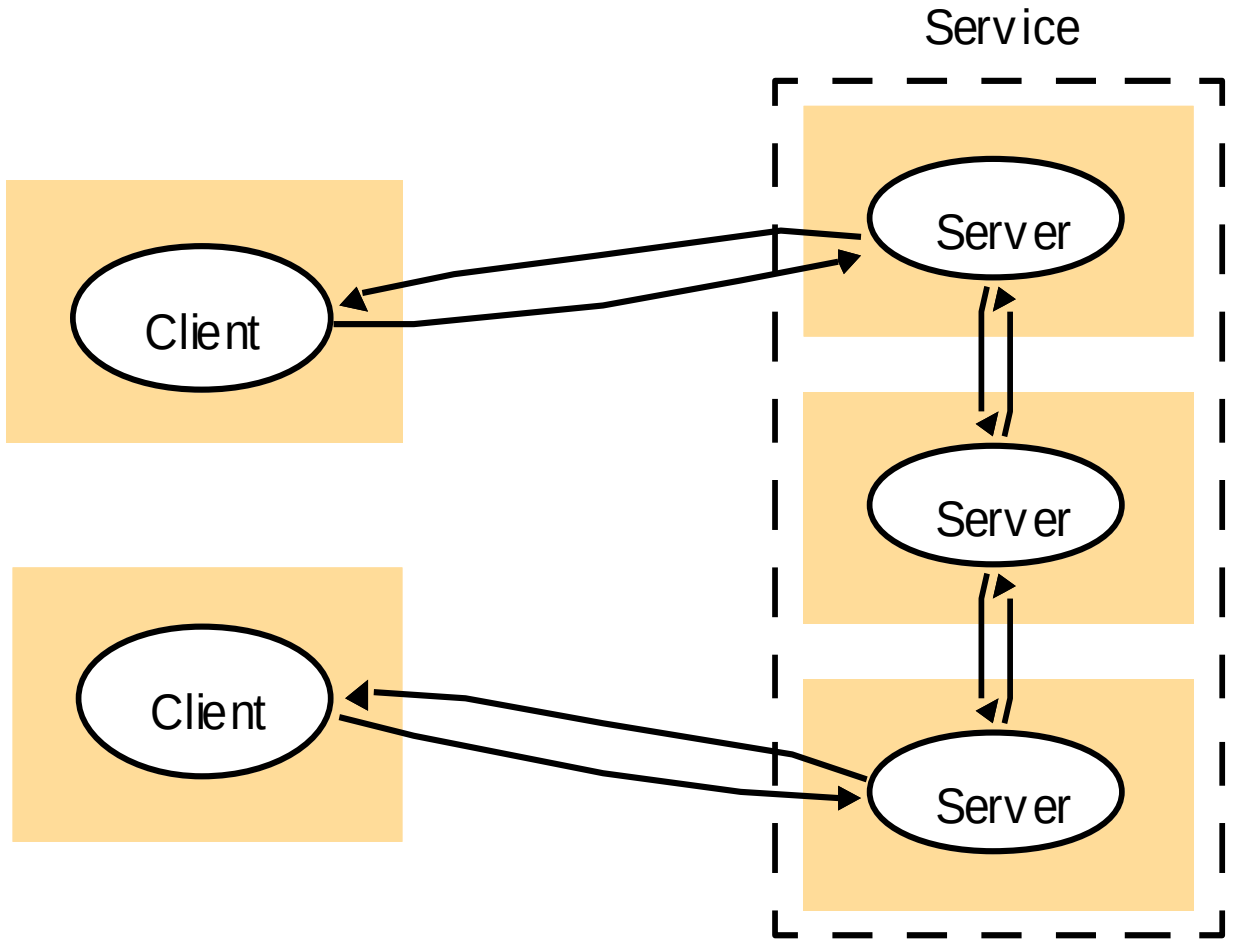
Services Provided by Multiple Servers

- Services may be implemented as several server processes in separate host computers interacting as necessary to provide a service to client processes.

- The servers may partition the set of objects on which the service is based and distribute between themselves or they may maintain replicated copies of them on several hosts. Eg web.

- Replication is used to increase performance and availability and to improve fault tolerance.

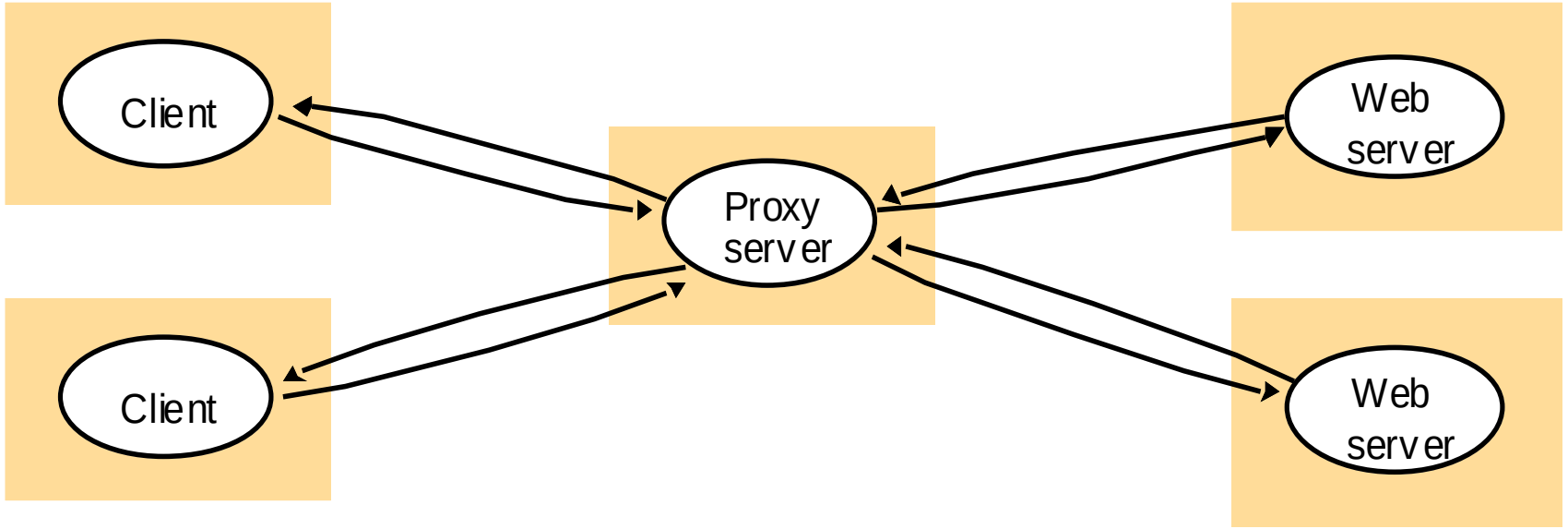
A Service Provided by Multiple Servers



Proxy Servers and Caches

- A cache is a store of recently used data objects that is closer than the objects themselves.
- Web browsers maintain a cache of recently visited web pages and other web resources in the clients's local file system,
  - using a special HTTP request to check with the original server that the cached pages are up to date before displying them.
- Web proxy servers provide a shared cache of web resources for the client machines at a site or across sites.
  - The purpose of proxy servers is to increase availability and performance of the service by reducing the load on the wide area network and webservers.
  - They may be used to access remote web servers through a firewall.

Web Proxy Server

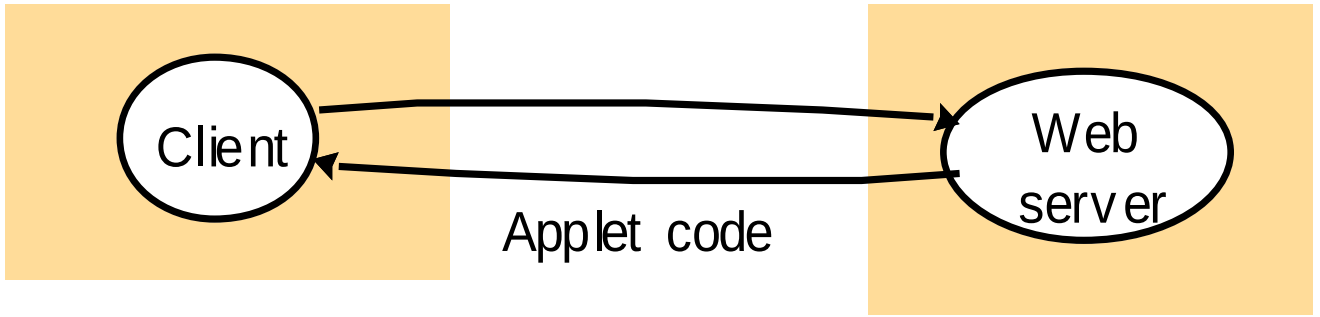


Mobile Code

- Applets are well-known and widely used example of mobile code.
- The user running a browser selects a link to an applet whose code is stored on a web server.
- The code is downloaded to the browser and runs there.
- An advatage of running the downloaded code locally is that it can give good interactive response
  - Since it does not suffer from the delays or variability of bandwidth associated with network communication.

Web applets

a) client request results in the downloading of applet code



b) client interacts with the applet

Mobile Agent

- It is a running program (including both code and data) that travels from one computer to another in a network
  - Carrying out a task on someone's behalf, such as collecting information, eventually returning with the results.

- It may make many invocations to local resources at each site it visits. For example accessing individual database entries.

- Mobile agents might be used
  - To install and maintain software on the computers within an organization.
  - To compare the prices of products from a number of vendors by visiting the site of each vendor and performing a series of database operations.

Mobile Agent

- The applicability of mobile agents might be limited because of
  - Mobile agents are a potential security threat to the resources in computers that they visit.
  - It can themselves to be vulnerable- they may not be able to complete their task if they are refused to access to the information they need.

Mobile Devices and Spontaneous Interoperation

- Unlike mobile agents mobile devices are hardware computing components that move between physical locations and thus networks, carrying software components with them.
- Both clients and servers may exist on mobile devices
- Mobility transparency and Variable connectivity are the various issues related to mobile devices in client server architecture.

Mobile Devices and Spontaneous Interoperation

- Mobility leads to *spontaneous interoperation,* a variation on the client server model in which associations between devices are routinely created and destroyed.

- The process of associating the device with suitable local service is called *service discovery*.

Variations On the Models

- The use of multiple servers and caches to increase performance and resilience.
- The use of mobile code and mobile agents
  - Users need low cost computers with limited hardware resources that are simple to manage
  - The requirement to add and remove mobile devices in a convenient manner.

Summary

- System models: physical model, architectural model and fundamnetal model
- Architectural models of Distributed Systems include Client-Server, Peer to peer and their variations

References and Questions

- https://www.youtube.com/watch?v=L5BlpPU_muY

Thank you