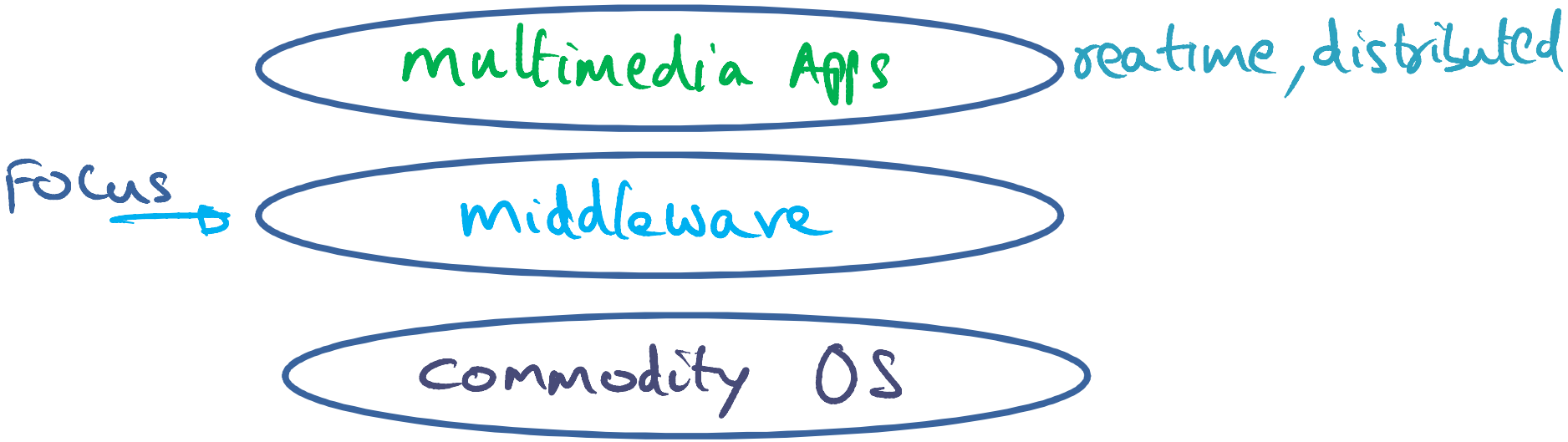
Real Time Computing - PTS

Lesson outline

✓ TS- Linux

→ Persistent Temporal streams



Programming Paradigms

Parallel Program

- Pthreads: API for parallel programs

Programming Paradigms

Parallel Program
- Pthreads: API for parallel programs

Distributed Programs
- Sockets: API for distributed programs

Programming Paradigms

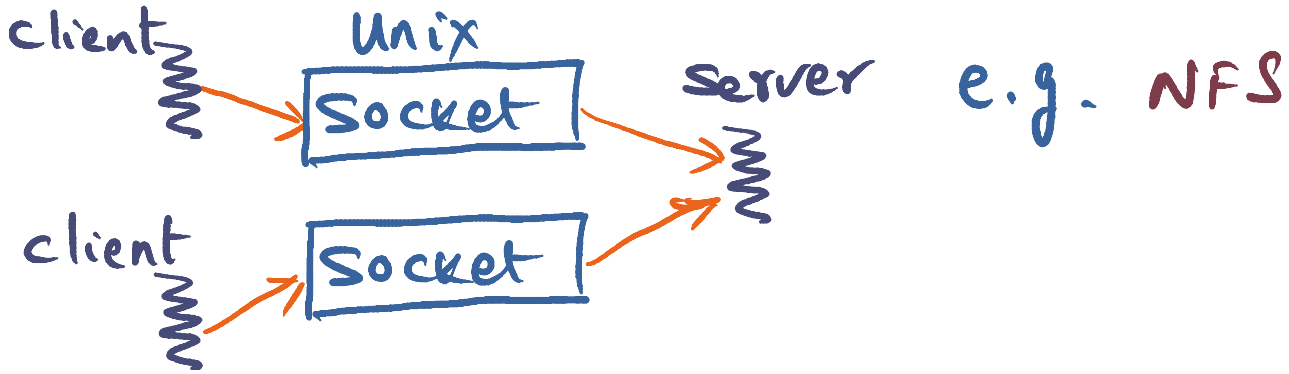
Parallel program

- Pthreads: API for parallel programs

Distributed Programs

- Sockets: API for distributed programs

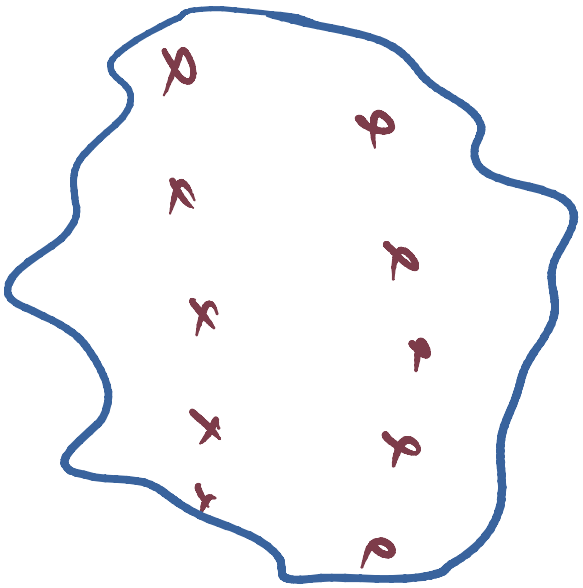
Conventional Distributed program



e.g. NFS

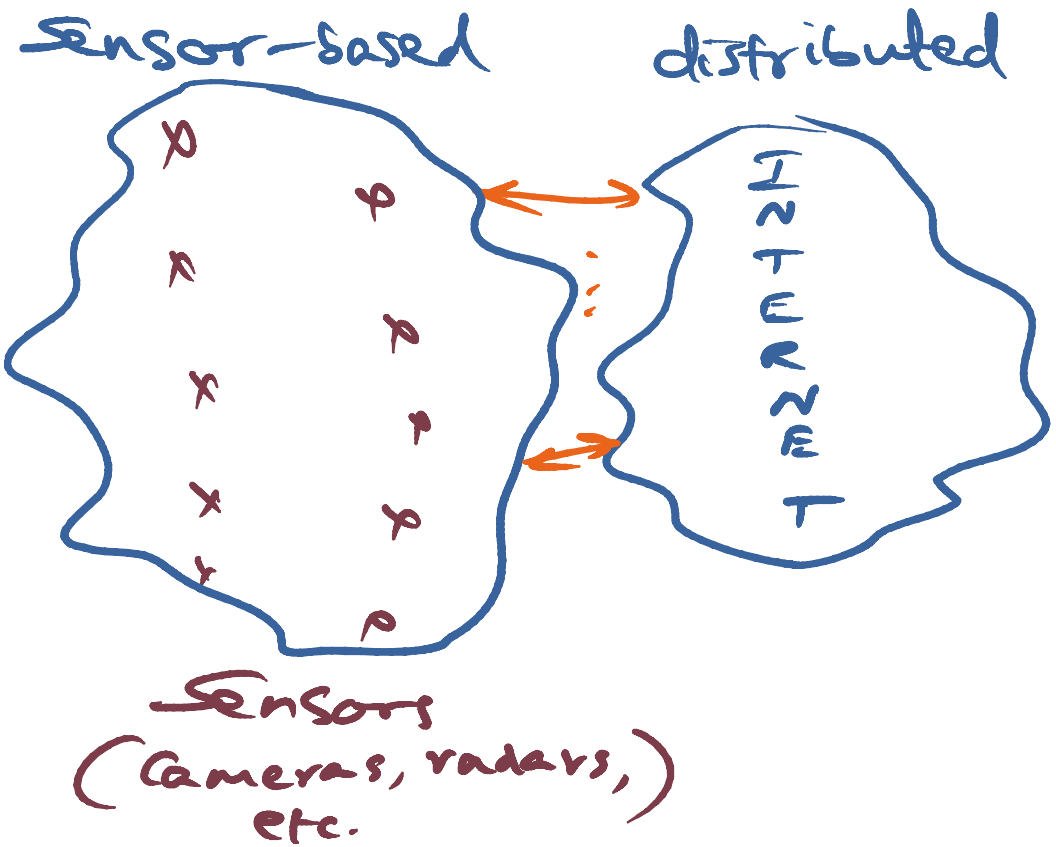
Novel Multimedia Apps

Sensor-based

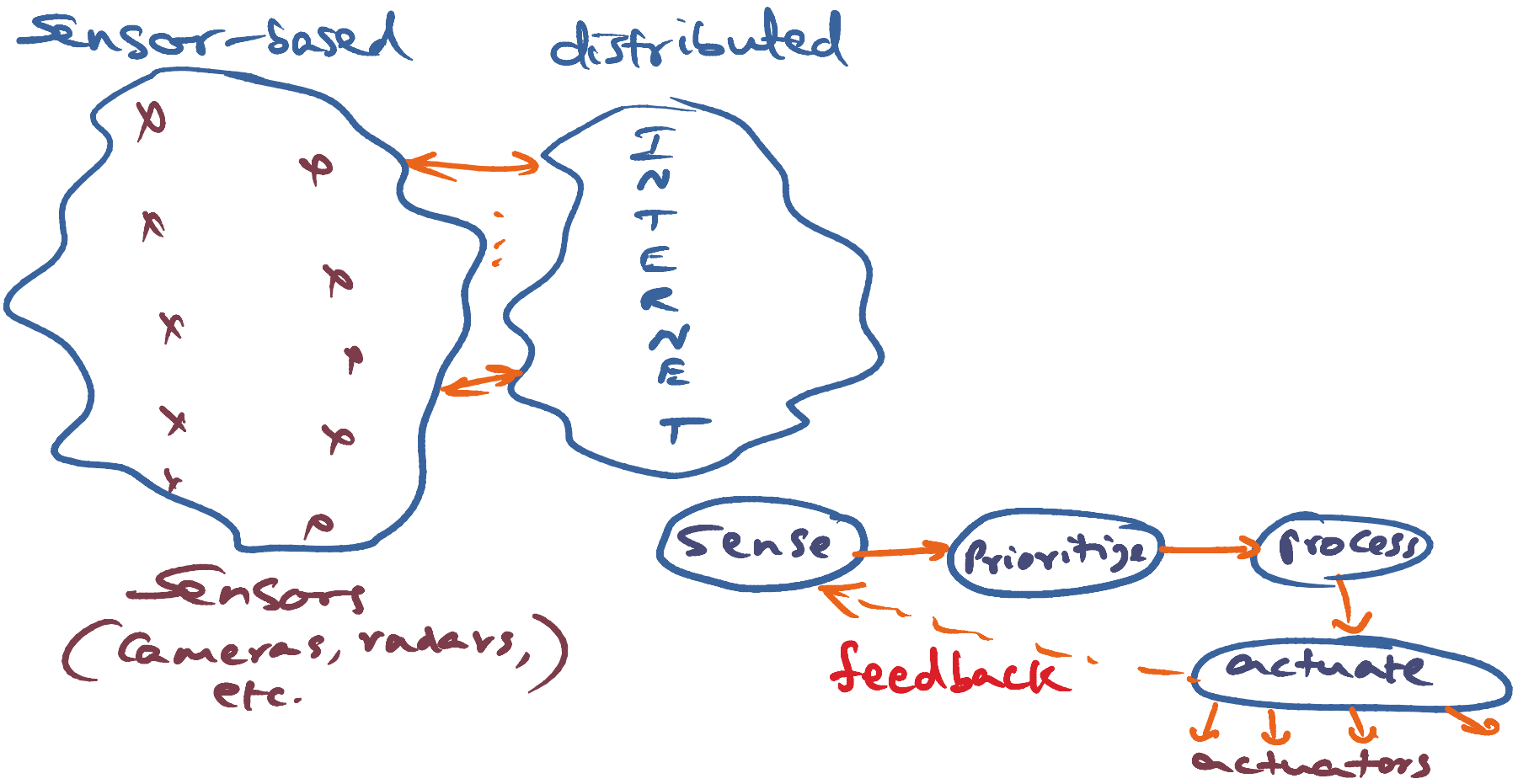


Sensors
(cameras, radars,)
etc.

Novel Multimedia Apps

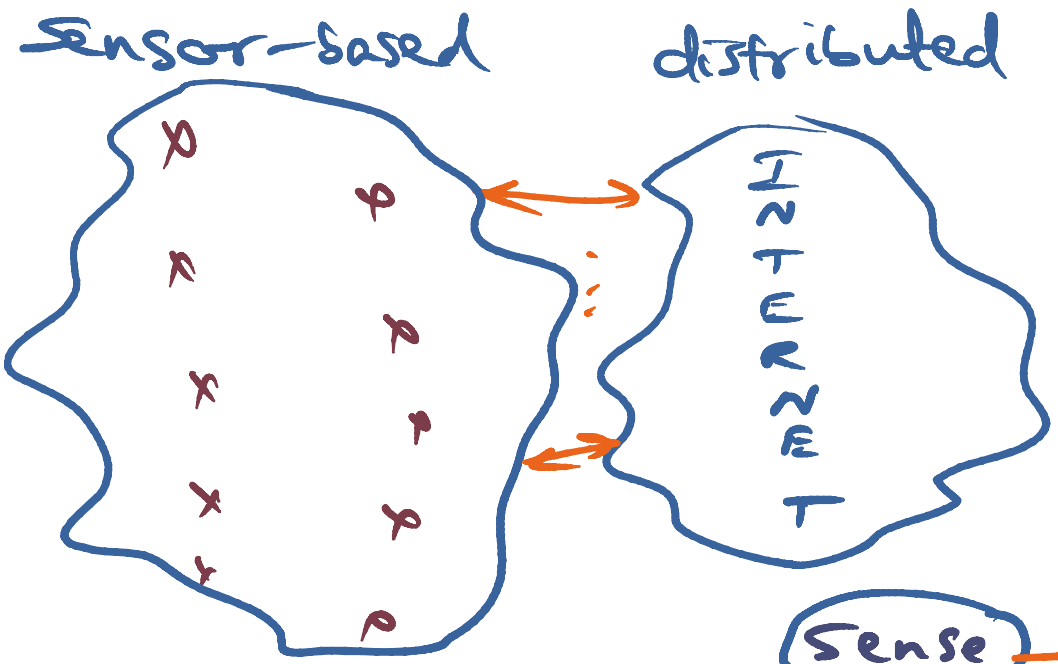


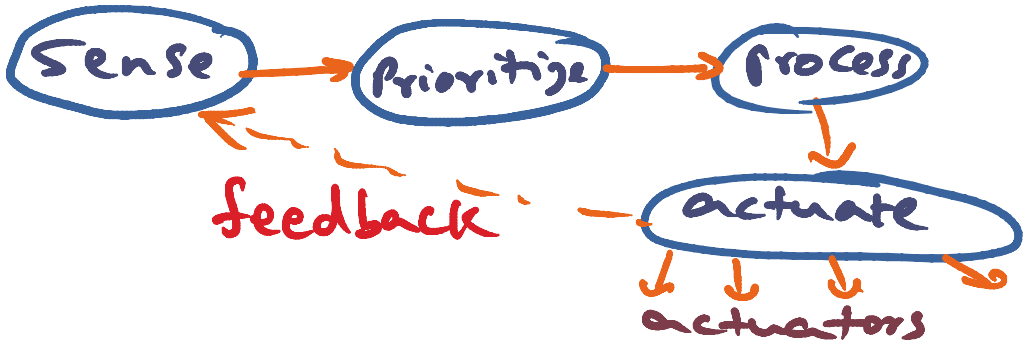
Novel Multimedia Apps



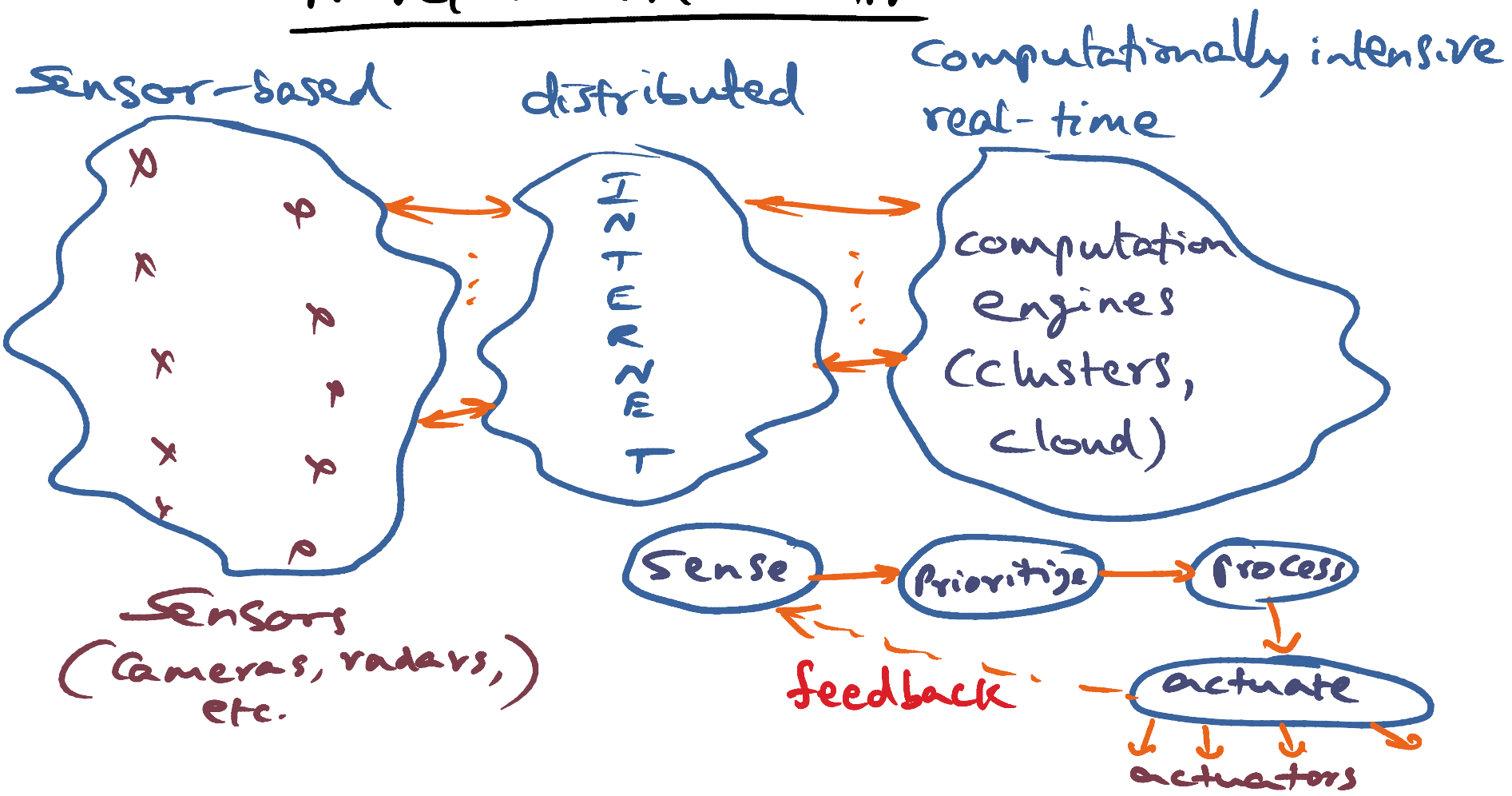
Novel Multimedia Apps

Computationally intensive
real-time





Novel Multimedia Apps



Example: Large-scale Situation Awareness

Example: Large-scale Situation Awareness

Infrastructure Overhead

Example: Large-scale Situation Awareness

Infrastructure Overhead

Cognitive Overhead

Example: Large-scale Situation Awareness

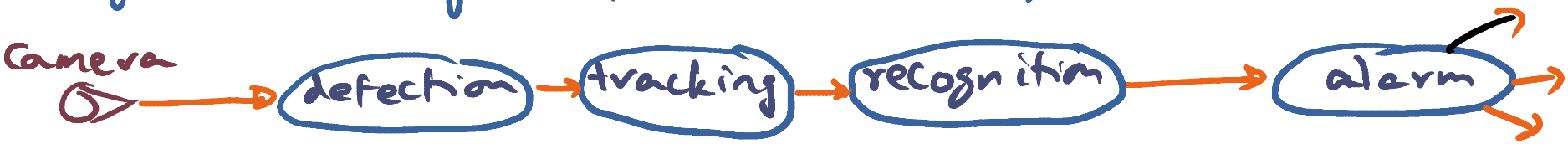
Infrastructure Overhead

Cognitive Overhead

False Positives
False Negatives

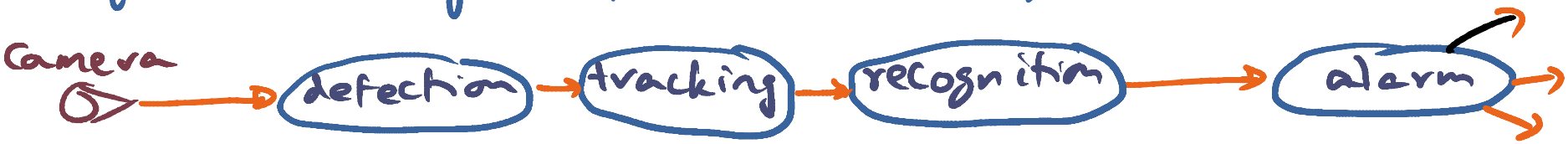
Programming Model for Situation Awareness

Sequential program for video analytics



Programming Model for Situation Awareness

Sequential program for video analytics

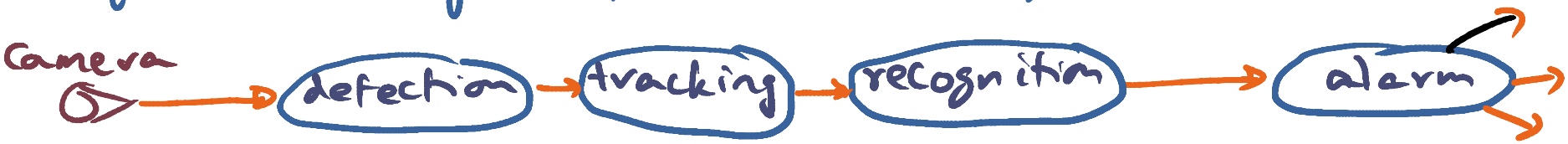


Objective in Situation Awareness Apps

- Process streams for high level inference

Programming Model for Situation Awareness

Sequential program for video analytics



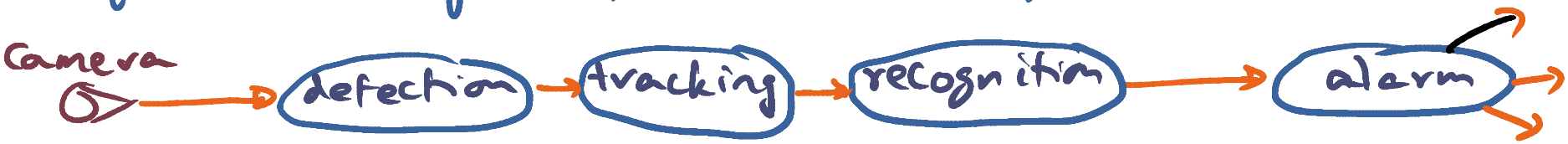
Objective in Situation Awareness Apps

- Process streams for high level inference

!Not watch youtube video! ☺

Programming Model for Situation Awareness

Sequential program for video analytics



Objective in Situation Awareness Apps

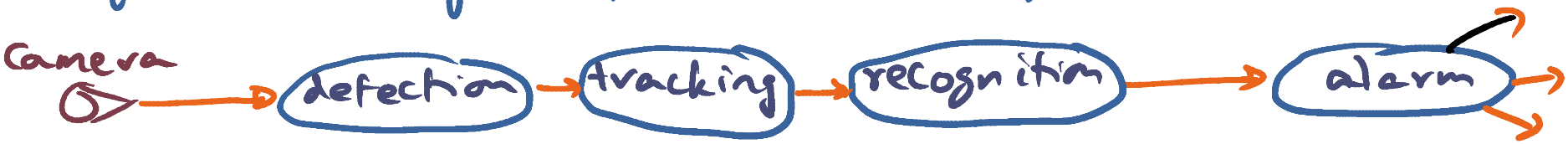
- Process streams for high level inference

!Not watch youtube video! ☺

How do we scale up to 1000's of cameras??

Programming Model for Situation Awareness

Sequential program for video analytics



Objective in Situation Awareness Apps
- Process streams for high level inference

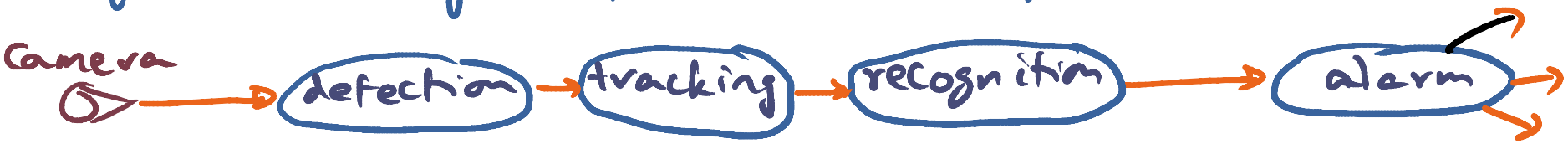
!Not watch youtube video! ☺

How do we scale up to 1000's of cameras??

PTS is just an exemplar of a distributed Programming System for such Apps

Programming Model for Situation Awareness

Sequential program for video analytics



Objective in Situation Awareness Apps
- Process streams for high level inference

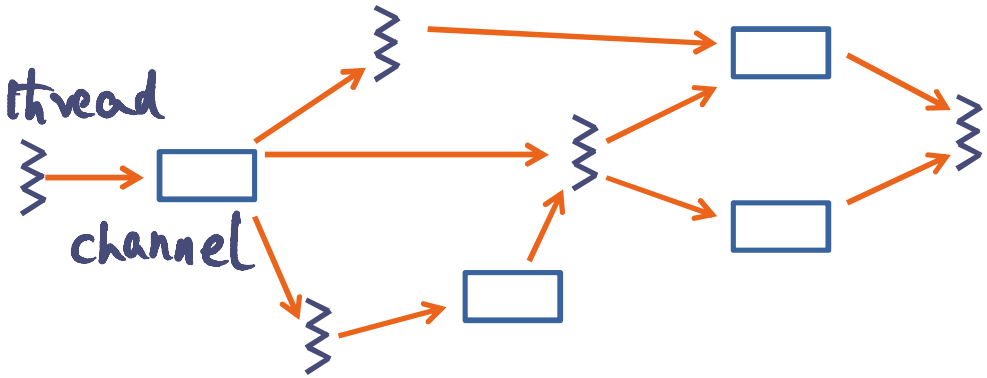
!Not watch youtube video! ☺

How do we scale up to 1000's of cameras??

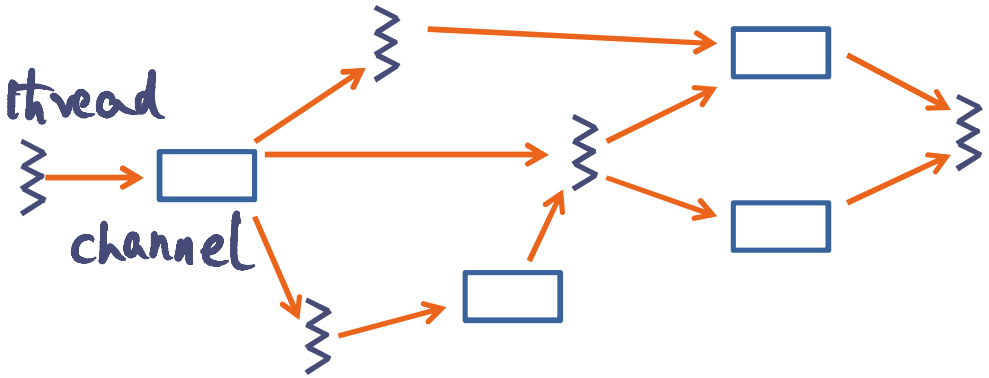
PTS is just an exemplar of a distributed Programming system for such Apps

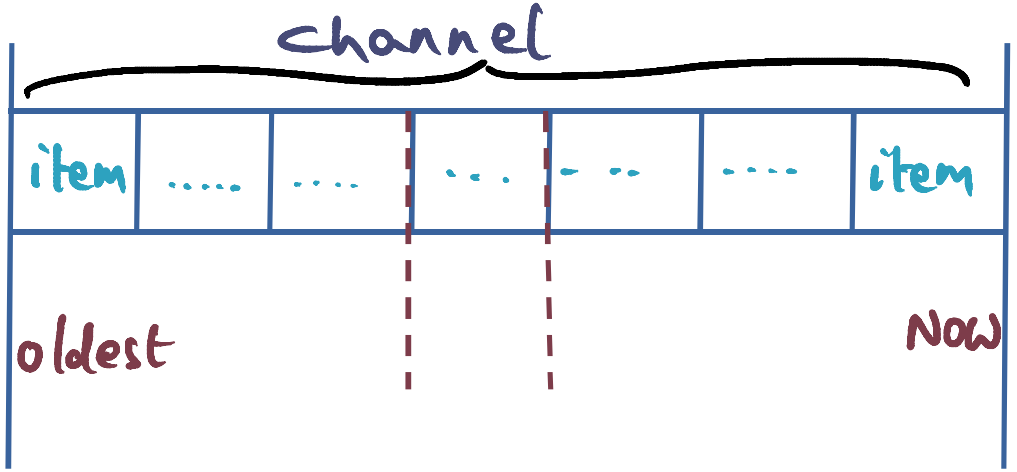
....not the last word!!

PTS Programming Model

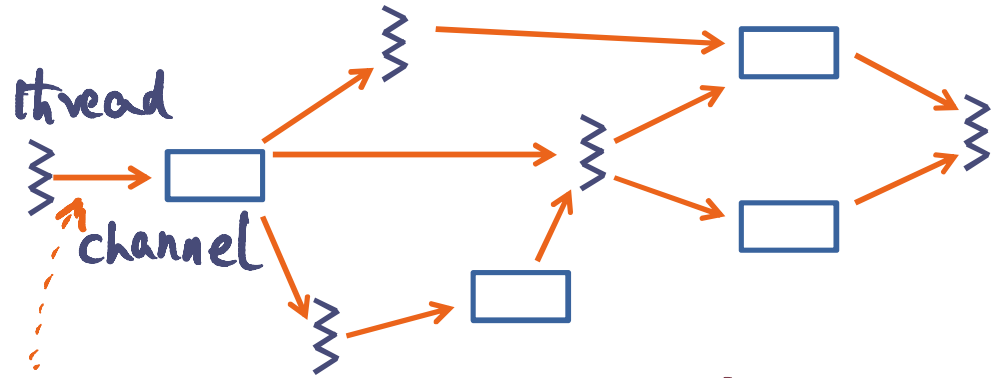


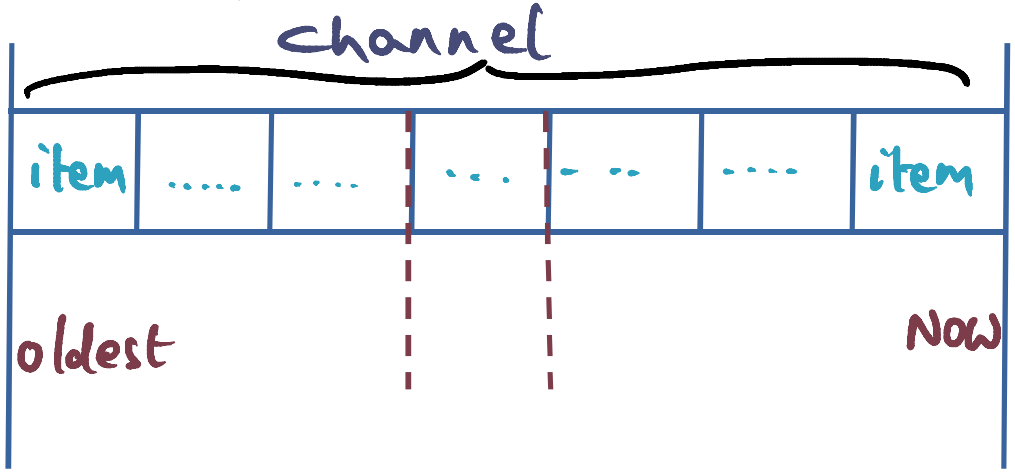
PTS Programming Model



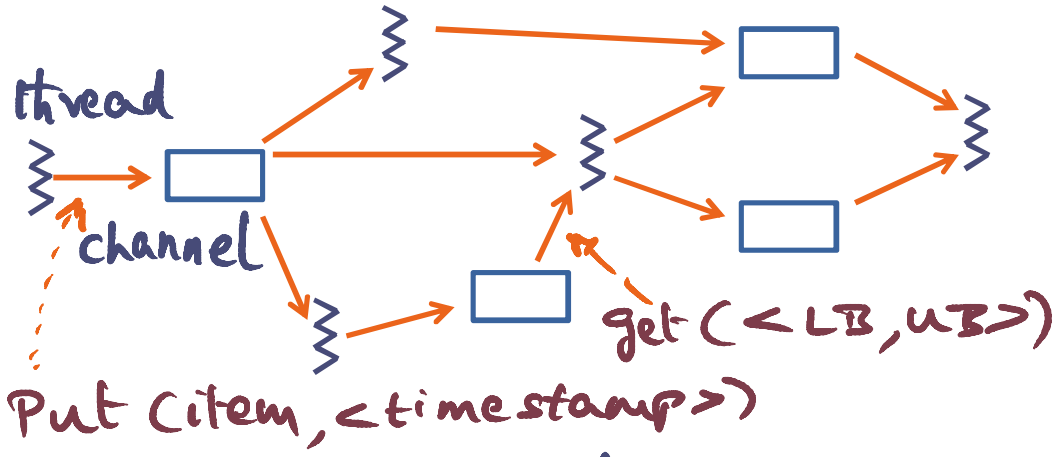


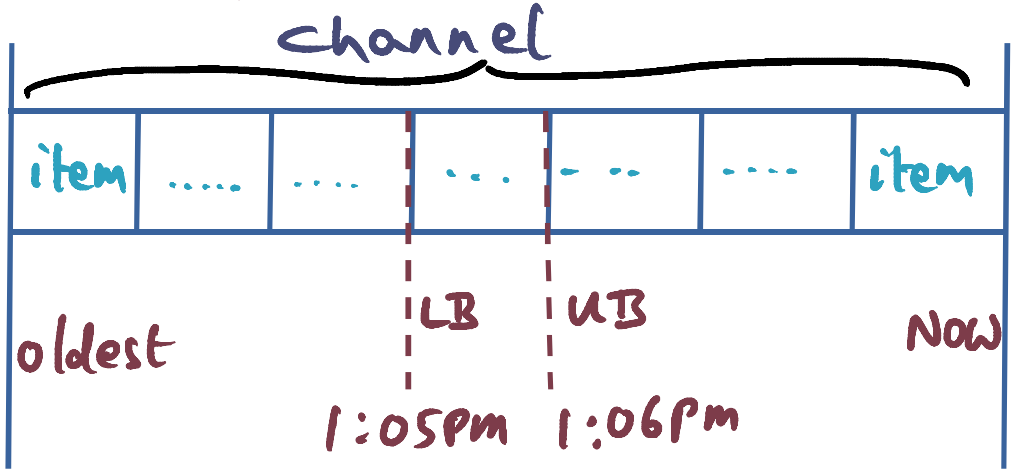
PTS Programming Model



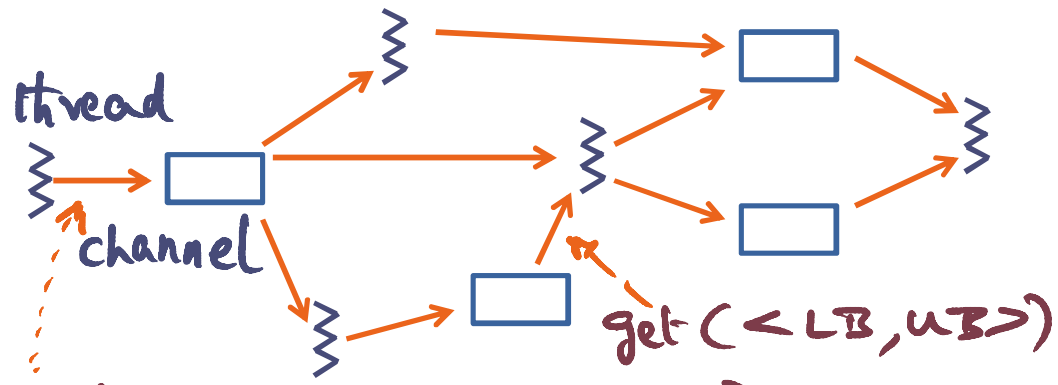


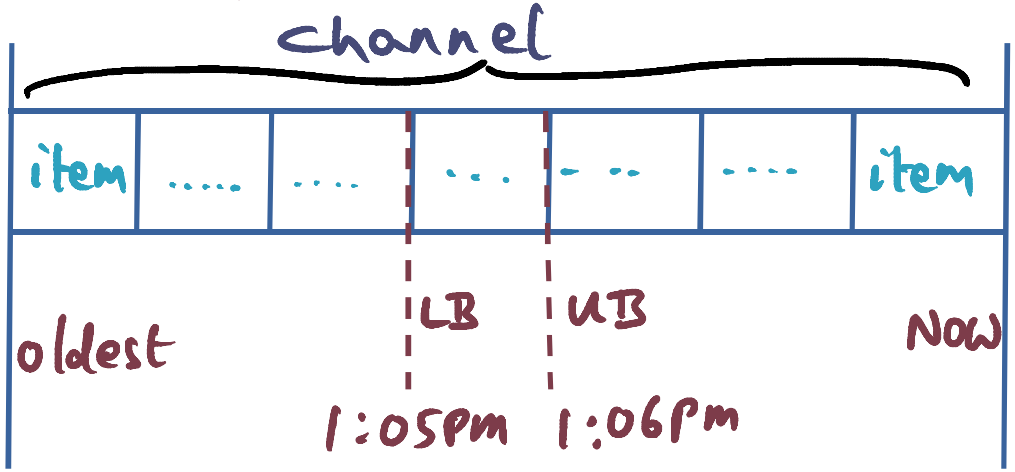
PTS Programming Model





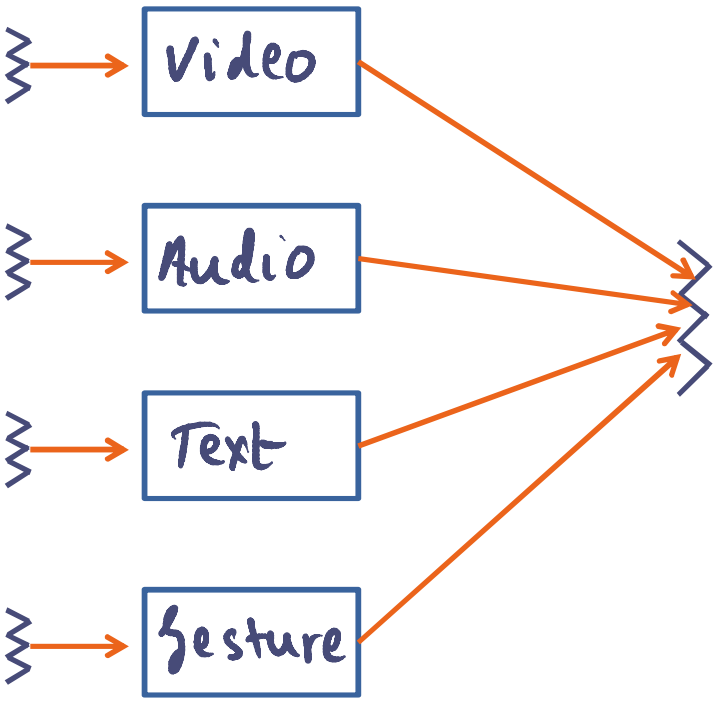
PTS Programming Model



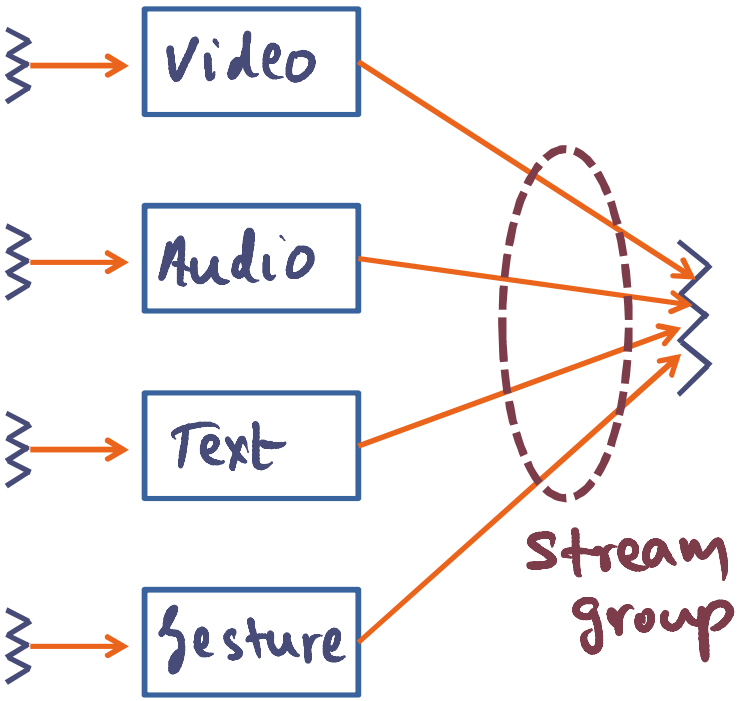



```
Channel ch1 =
    lookup ("Video channel");
While (1){
  //get data
  response r =
    ch1.get (<LB,UB>);
  //Process data
    :
  //Produce output
    ch2.put (item,<ts>);
}
```

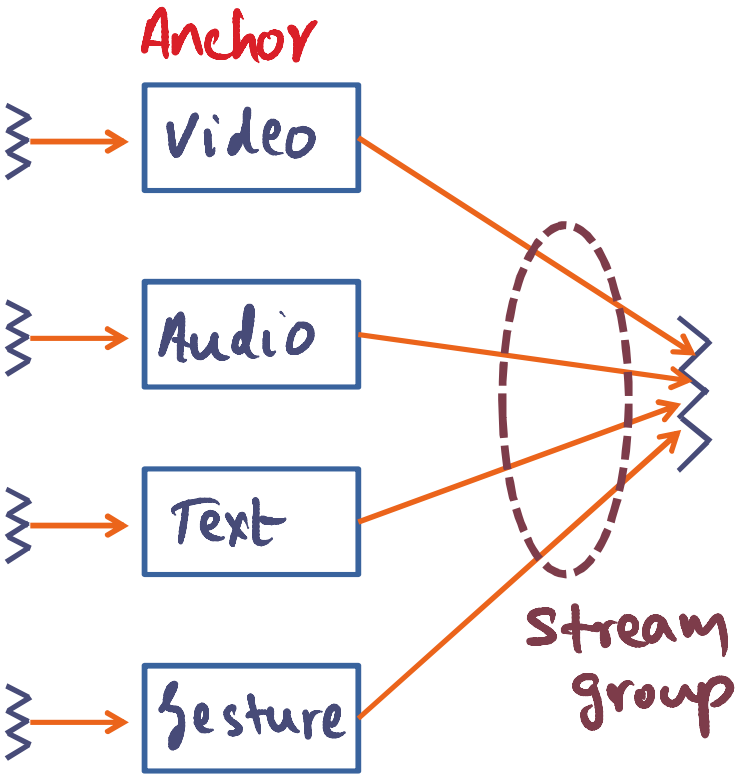
PTS: Bundling Streams



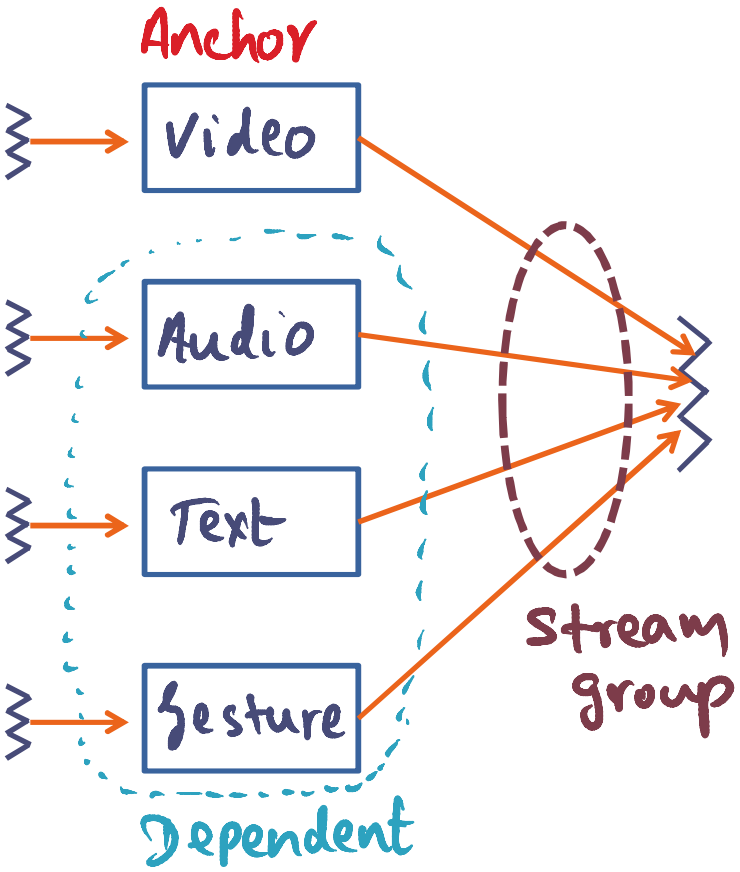
PTS: Bundling Streams



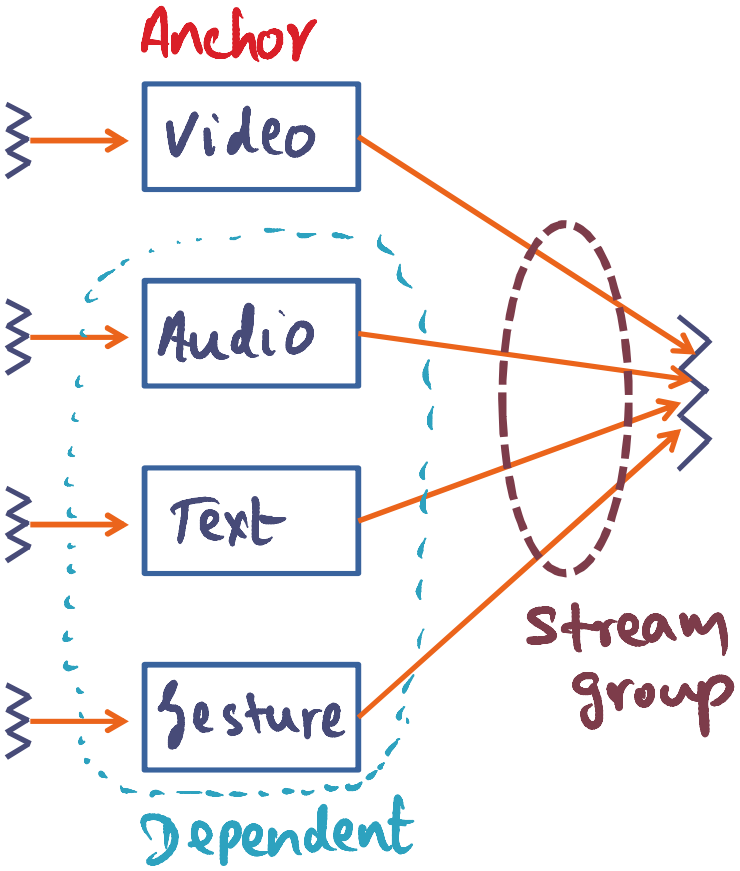
PTS: Bundling Streams



PTS: Bundling Streams



PTS: Bundling Streams

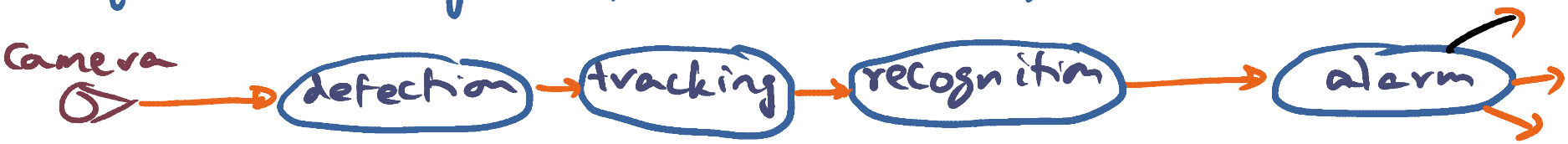


Group get :

- get corresponding time-stamped items from all the streams in the group

Power of Simplicity

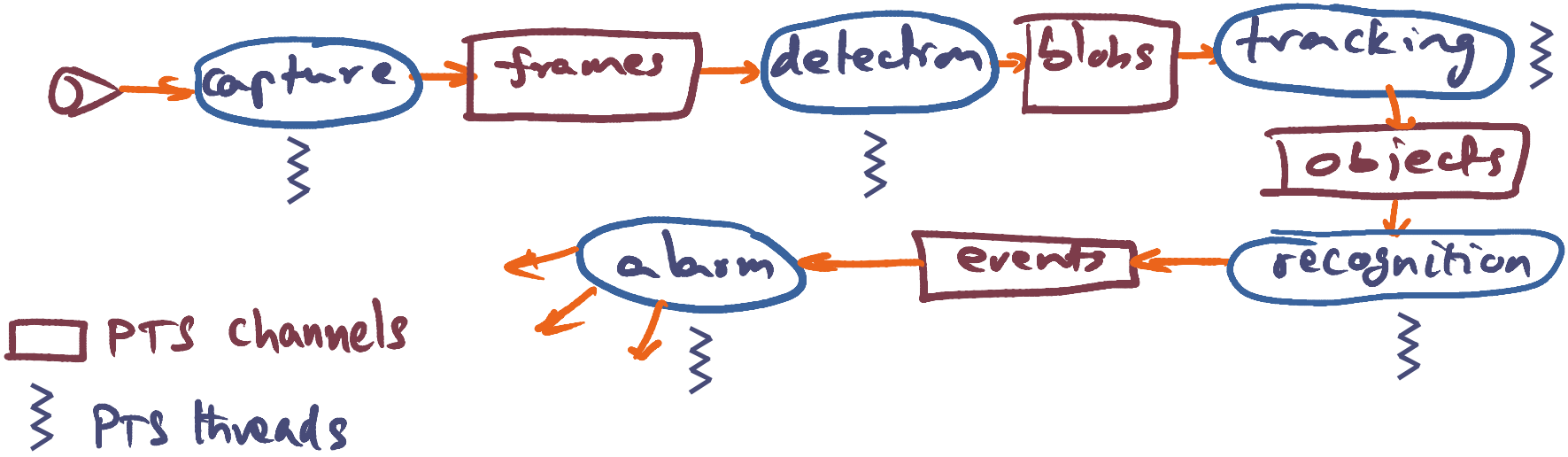
Sequential program for video analytics



Power of Simplicity

Sequential program for video analytics

Distributed program with get/put between modules



PTS Design Principles

Simple Abstractions / interfaces

— channel and get/put

PTS Design Principles

Simple Abstractions / interfaces

— channel and get/put

Do the heavy lifting (systems) under the covers

PTS Design Principles

Simple Abstractions / interfaces
- channel and get/put

Do the heavy lifting (systems) under the covers

PTS channels
- Can be anywhere
- Can be accessed from anywhere
- network-wide unique

PTS Design Principles

Simple Abstractions / interfaces
- channel and get/put

Do the heavy lifting (systems) under the covers

PTS channels
- Can be anywhere
- Can be accessed from anywhere
- network-wide unique

} Similarity to Unix sockets

PTS Design Principles

Simple Abstractions / interfaces
- channel and get/put

Do the heavy lifting (systems) under the covers

PTS channels
- Can be anywhere
- Can be accessed from anywhere
- network-wide unique
- time first class entity

} Similarity to Unix sockets

PTS Design Principles

Simple Abstractions / interfaces
- channel and get/put

Do the heavy lifting (systems) under the covers

PTS channels
- Can be anywhere
- Can be accessed from anywhere
- network-wide unique

} Similarity to Unix sockets

- time first class entity
- Persist streams under App control

Group Activity

Similarity of Unix Socket + PTS channel?

Differences between Unix Socket + PTS channel?

Synchronization causality?

Temporal causality?

PTS Design Principles

Simple Abstractions / interfaces
- channel and get/put

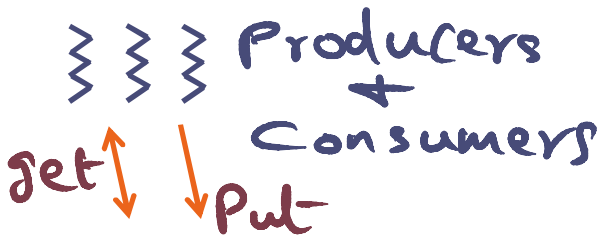
Do the heavy lifting (systems) under the covers

PTS channels
- Can be anywhere
- Can be accessed from anywhere
- network-wide unique

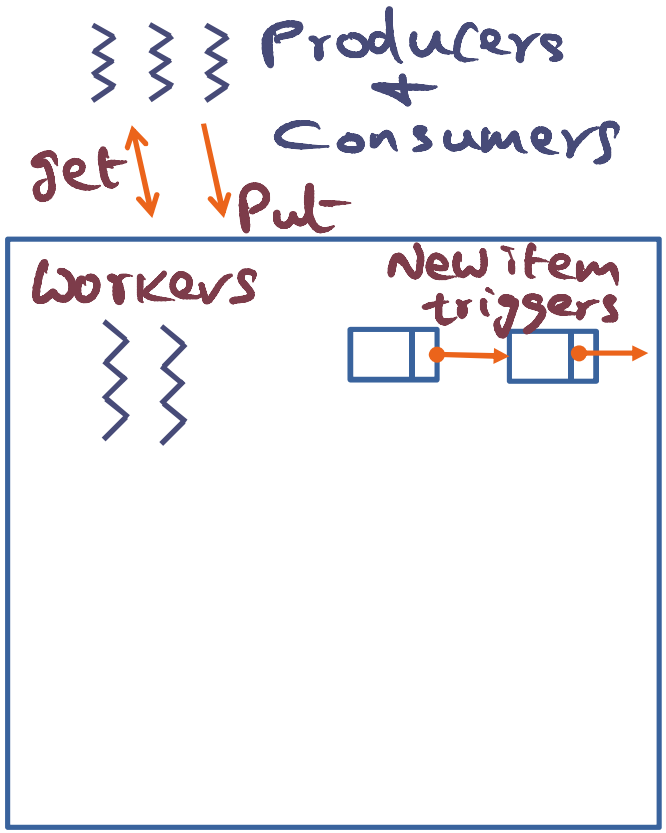
} Similarity to Unix sockets

- time first class entity
- Persist streams under App control
- Seamlessly handle live and historical data

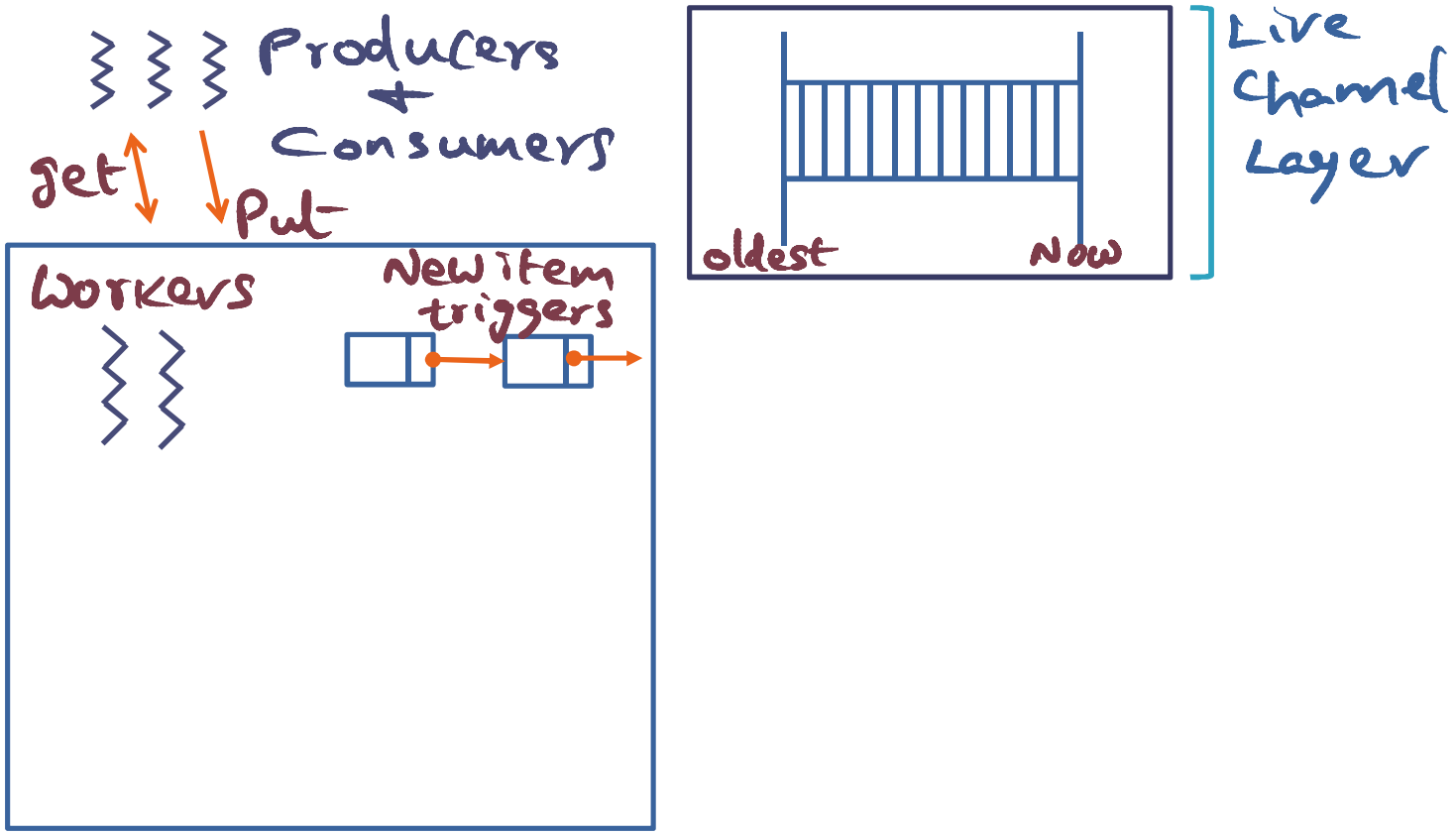
Persistent Channel Architecture



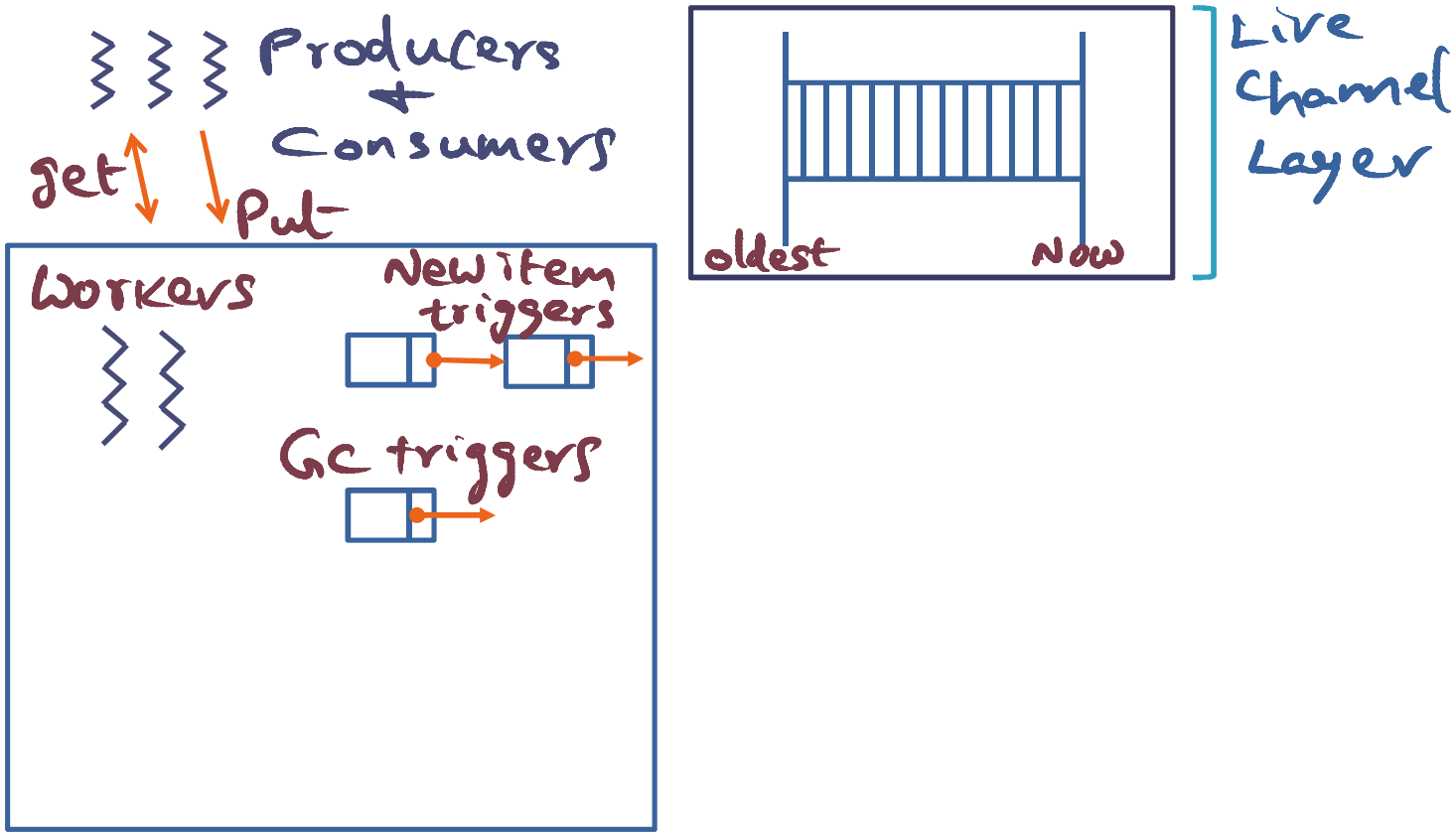
Persistent Channel Architecture



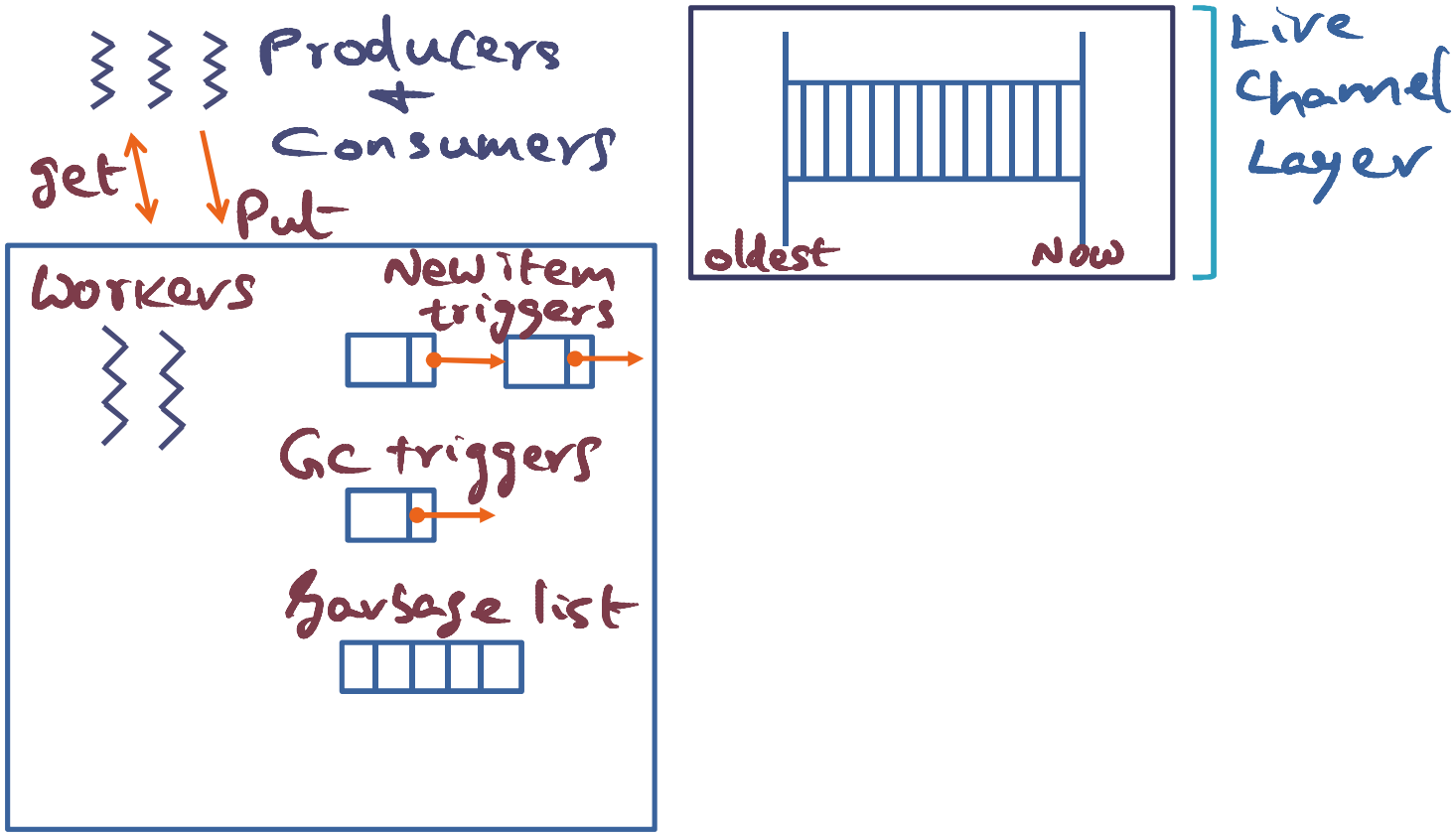
Persistent Channel Architecture



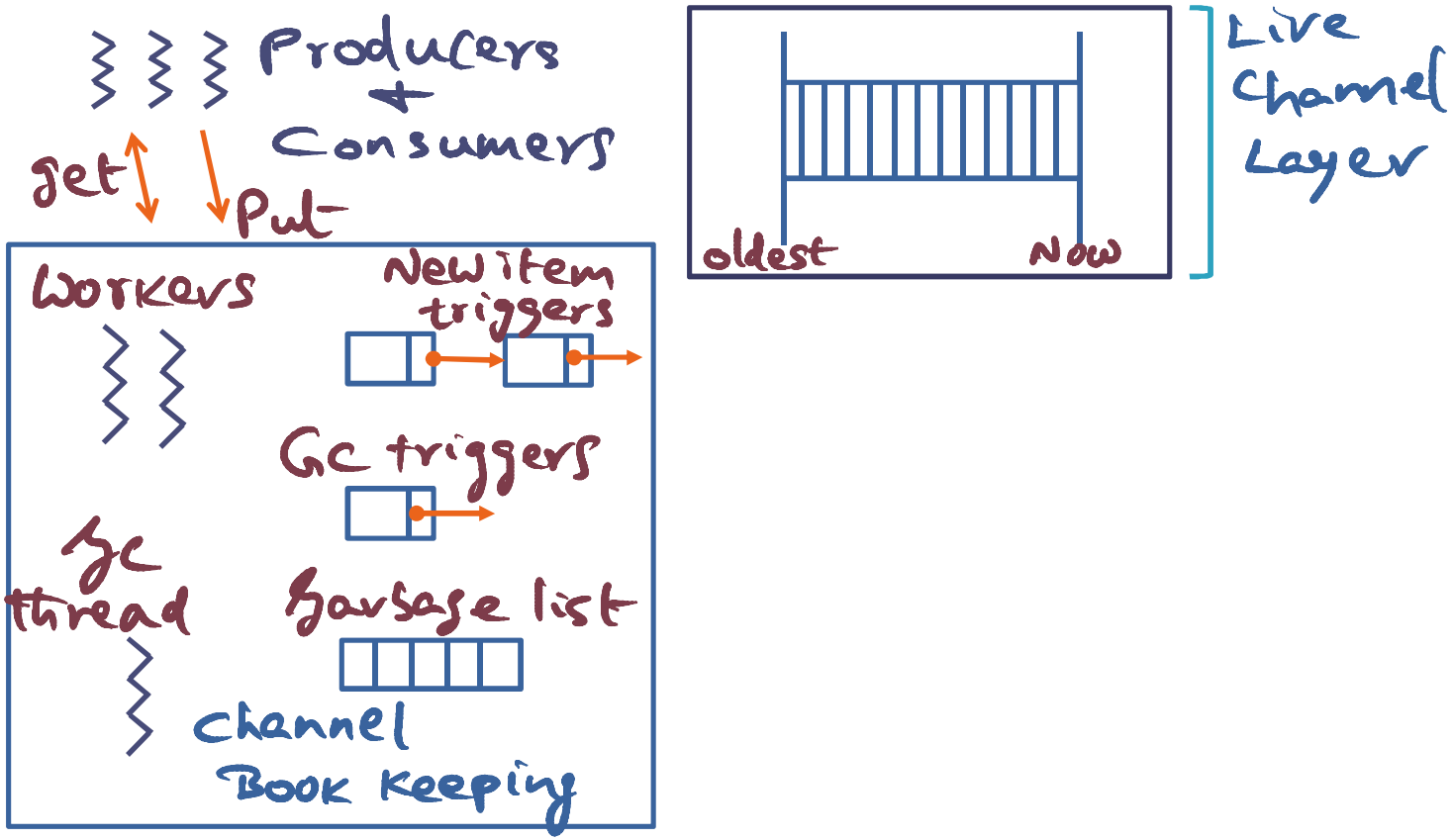
Persistent Channel Architecture



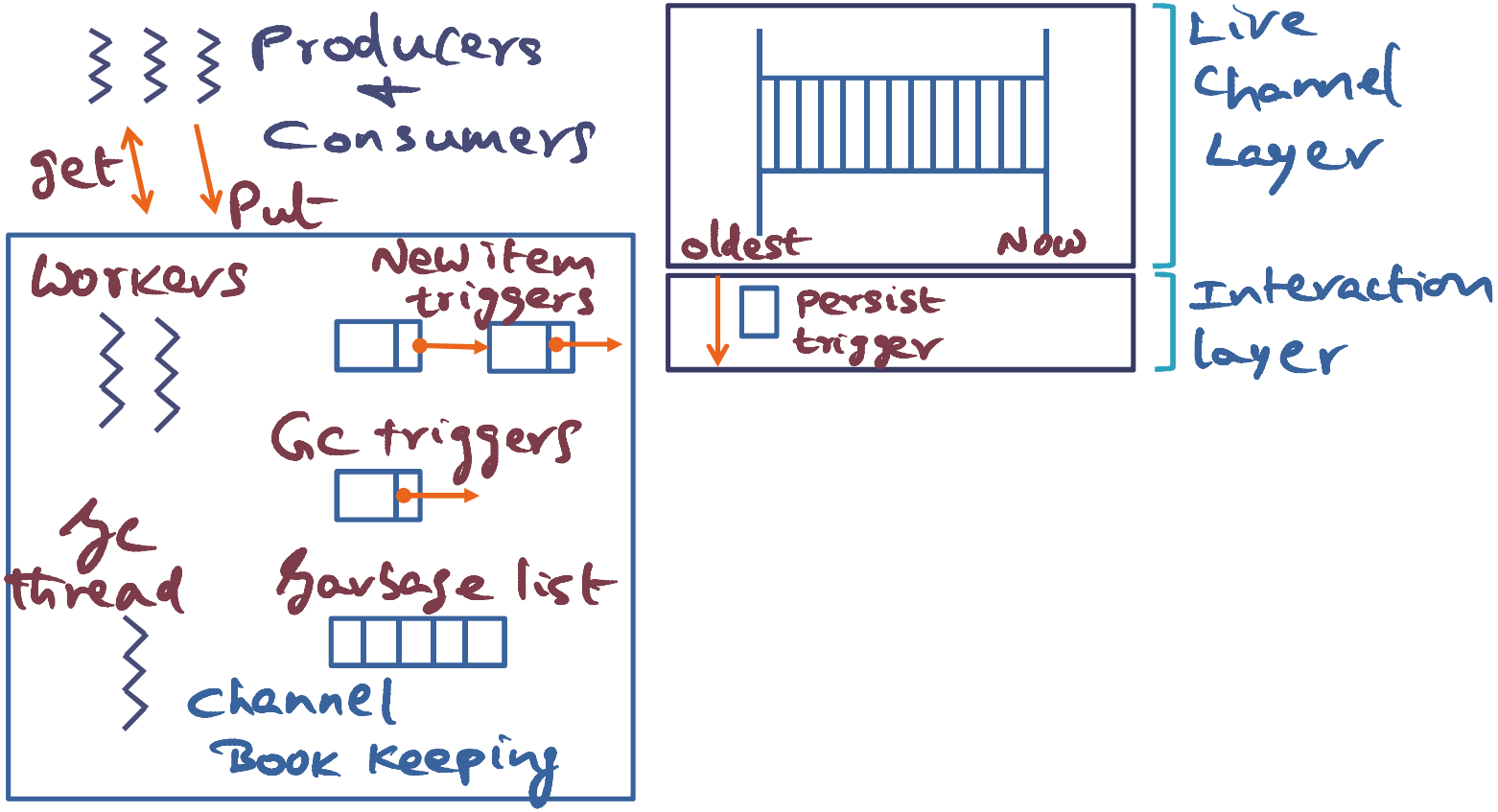
Persistent Channel Architecture



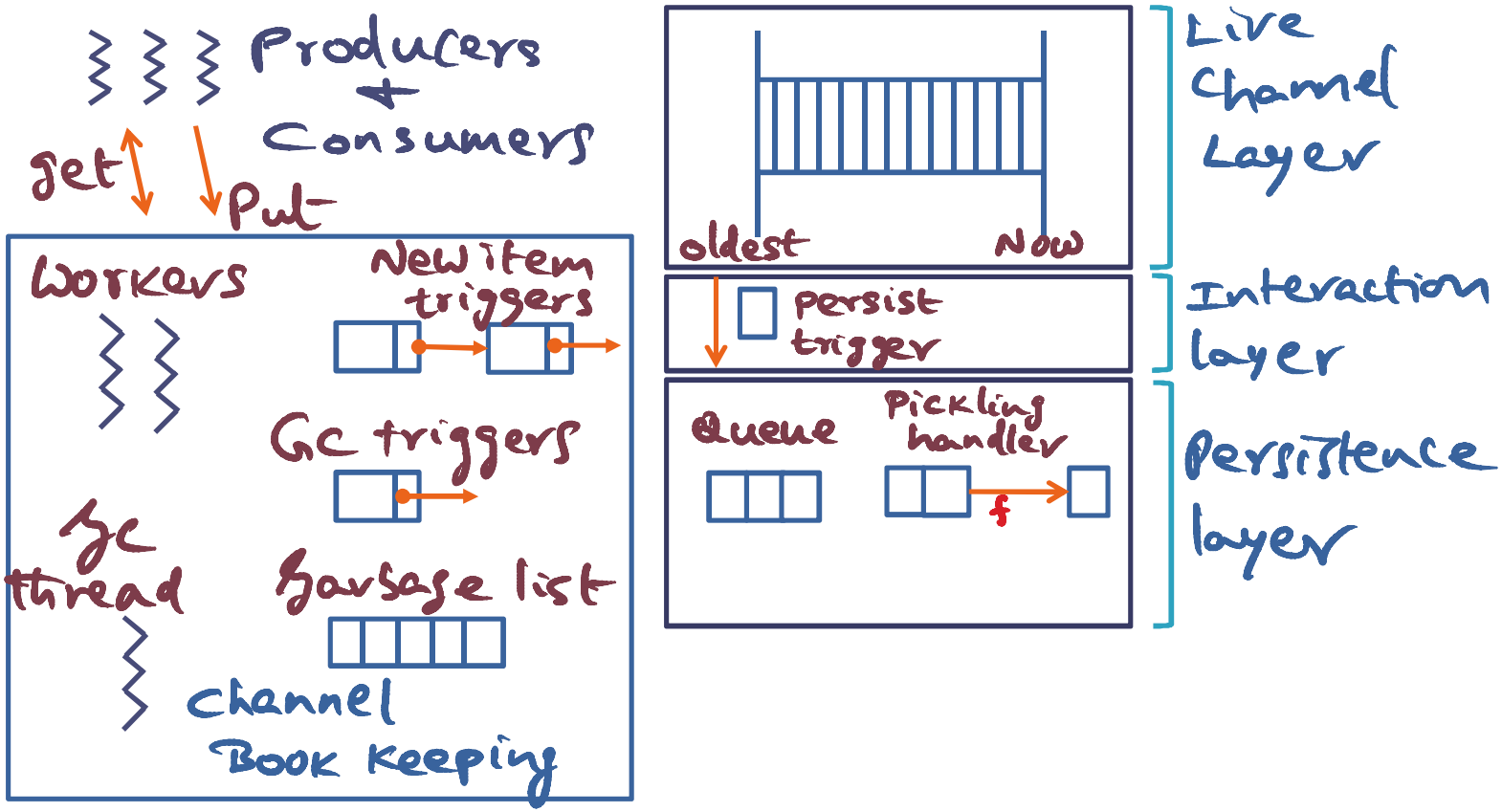
Persistent Channel Architecture



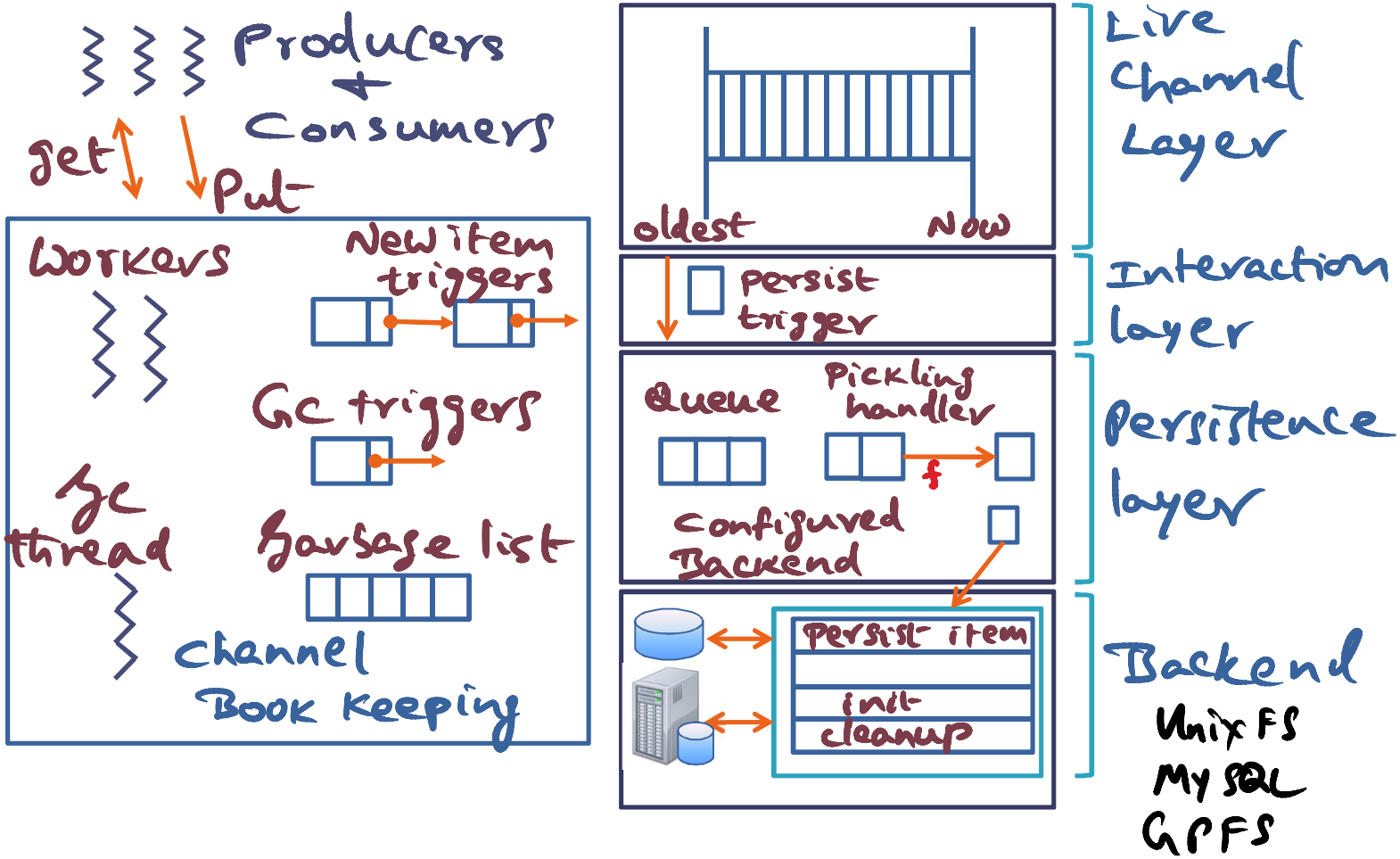
Persistent Channel Architecture



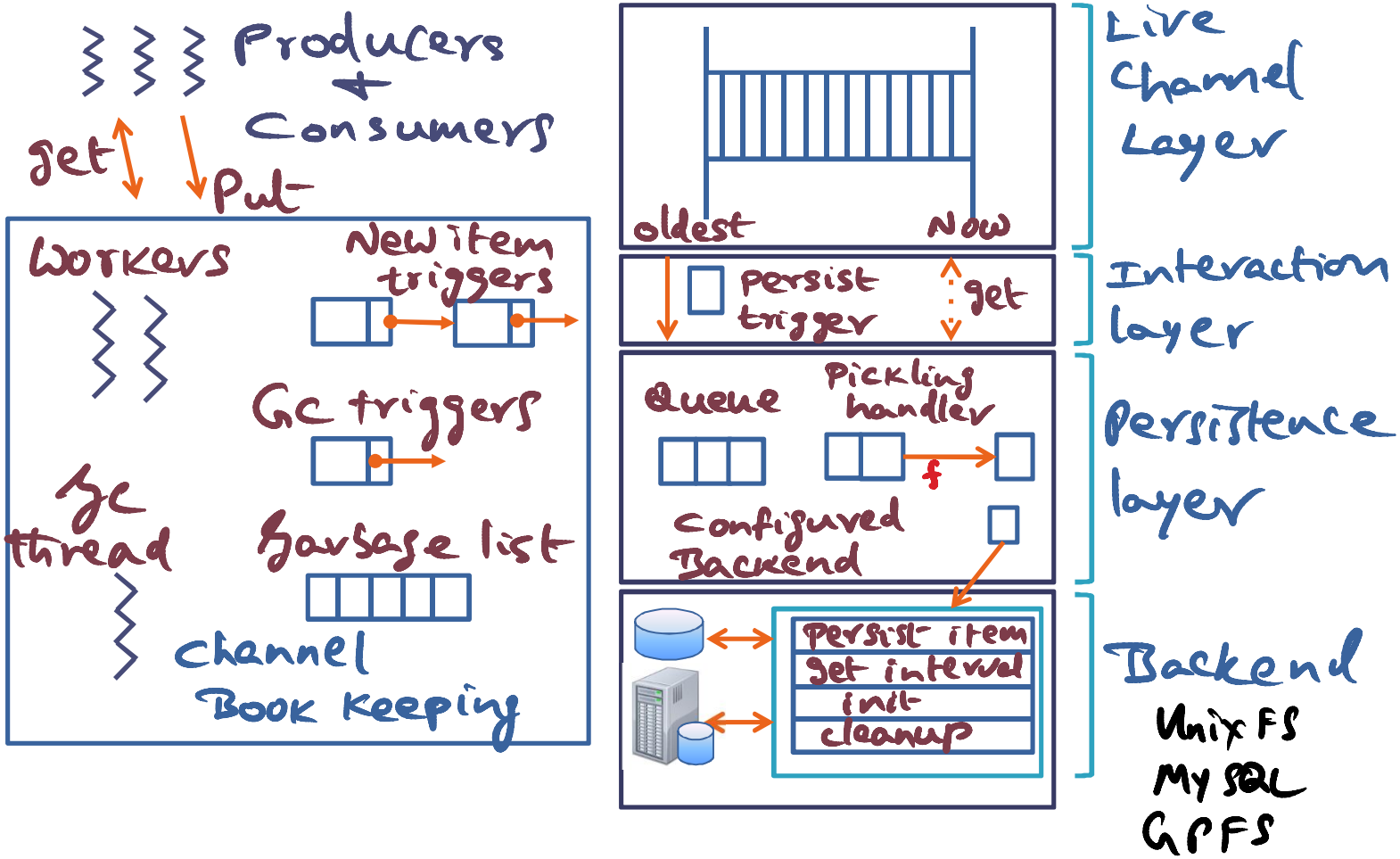
Persistent Channel Architecture



Persistent Channel Architecture



Persistent Channel Architecture



Key takeaways

- Similarity to map-reduce provides a simple and intuitive programming model for the domain expert to develop “big data” processing apps, PTS is providing a simple programming model for the domain expert to develop “live stream analysis” applications.
- Time-based distributed data structures for streams, automatic data management, transparent stream persistence are the unique features of the PTS programming model that facilitate live stream analysis.