Homework-4

Mohan Srinivas Naga Satya

NUID: 001821918

Exercise 6.1:

Construct an argument as to why indirect communication may be appropriate in volatile environments. To what extent can this be traced to time uncoupling, space uncoupling or indeed a combination of both?

Indirect communication is defined as communication between entities in a distributed system through an intermediary with no direct coupling between the sender and the receiver(s). Volatile environments refer to the concept where the communication to between the client and server is affected or intercepted by the intercepted by another third-party entity. In addition, the precise nature of coupling varies significantly between systems. Indirect communication avoids this direct coupling and hence inherits interesting properties.

Indirect communication is often used in distributed systems where change is anticipated – for example, in mobile environments where users may rapidly connect to and disconnect from the global network – and must be managed to provide more dependable services. Indirect communication is also heavily used for event dissemination in distributed systems where the receivers may be unknown and liable to change – for example, in managing event feeds in financial systems. Hence, in these volatile environments, the use of indirect communication helps the system maintain a space and time uncoupling allowing the sender and receiver(s) to exist in different places at different times.

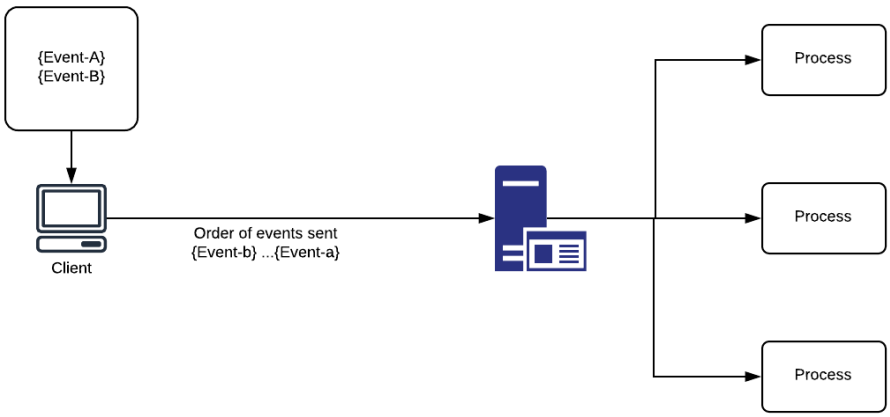
With indirect communication, though there is a performance overhead, the need for space and time uncoupling takes more priority. Following with the mobile communication example above, indirect communication the need for time uncoupling. As the mobile senders keep moving from place to place, there can be multiple receivers based on the geography and time uncoupling needs to be implemented. Also, in the scenario, when the mobile sender propagates the signal, it wouldn't know the exact sender processing the request. These examples demonstrate that space and time uncoupling can be traced to every indirect communication.

Exercise 6.6

In the context of a group communication service, provide example message exchanges that illustrate the difference between causal and total ordering.

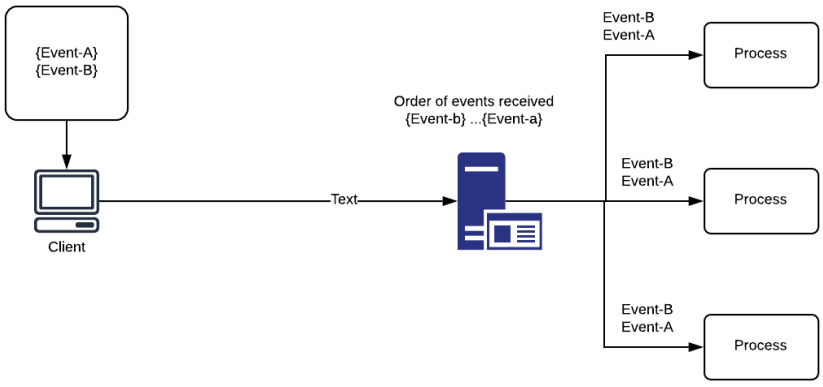
Casual Ordering

In group communication, casual ordering is when the Event A happens before Events B, then the ordering of messages delivered to the receiver processes is preserved in the order of event occurrence.



Total Ordering

In group communication, total ordering is when the message B is delivered to the receiver before message A, then the order of the messages received is preserved and sent to the processes of the receiver in the preserved order.

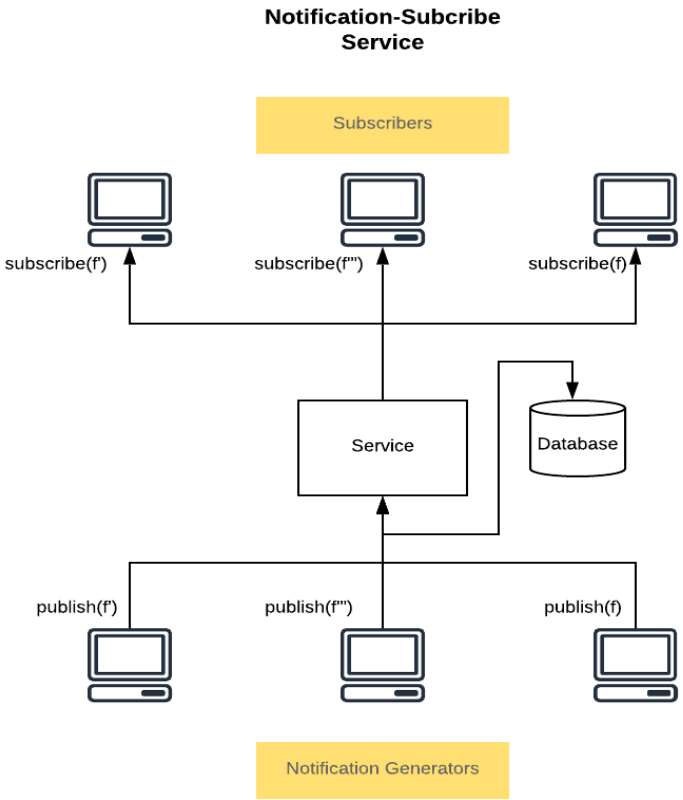


Exercise 6.8

Suggest a design for a notification mailbox service that is intended to store notifications on behalf of multiple subscribers, allowing subscribers to specify when they require notifications to be delivered. Explain how subscribers that are not always active can make use of the service you describe. How will the service deal with subscribers that crash while they have delivery turned on?

The notification mailbox service can be similar to a publish-subscribe service. The key point of the notification-mailbox service would be:

- Notifications
  - The notification for an event *f* generated are stored in a notification-subscribe system. This system utilizes the database it has to store the notifications by category.
- Subscription:
  - The subscription clients then make a request of notifications for an event. And these notifications are then retrieved on-demand and sent to the clients.



Subscribers who are not active throughout the day can request for a delivery of the notifications whenever the client system becomes active. As the notification-subscribe system is on-demand in nature, the notifications are then delivered to the client. The clients can also make a request to service in pre-defined intervals. In doing so, whenever the intervals elapses, all the pending notifications are delivered to the client which are generated during that time-frame.

The service can have handlers which check for the availability of the client before, during and after the delivery. In doing so, the service can make sure the notifications are delivered to the client when the notifications need to be sent to the client for a time-interval. The same system can also be used to check for the existence of the client during the delivery of the notifications. And on non-availability, the transaction can be reversed and be sent to the client again once it become available.

Exercise 6.16

Discuss whether message passing, or DSM is preferable for fault-tolerant applications.

Fault-tolerant applications can be defined based on the services being offered by the applications. Though there is no definite answer about which method is more preferable for the application as whole, the below comparison takes into picture each service being offered and discusses the viability of the methods. But irrespective of the application, the fault-tolerant applications offer following benefits:

- Consistency
    - A fault tolerant would be consistent in delivering non-corrupted data. In that sense, message passing provides the capability to share the resources among multiple data layers, whereas DSM uses shared-memory between different processes and hence, does not provide the consistency required to access different layers of data.
- Persistence
    - Persistence of a fault-tolerant application can be defined as its ability to persist the data even after the client disconnects. The data persistence allows the system to be distributed in nature and allow access to multiple instances of the same application.
    - In that perspective, message passing schemes cannot persist data between sessions as they rely on a database which has its own persistence properties. Whereas, DSM shares the memory between its processes which allows multiple instances or processes to share the same memory db and retain information between sessions.

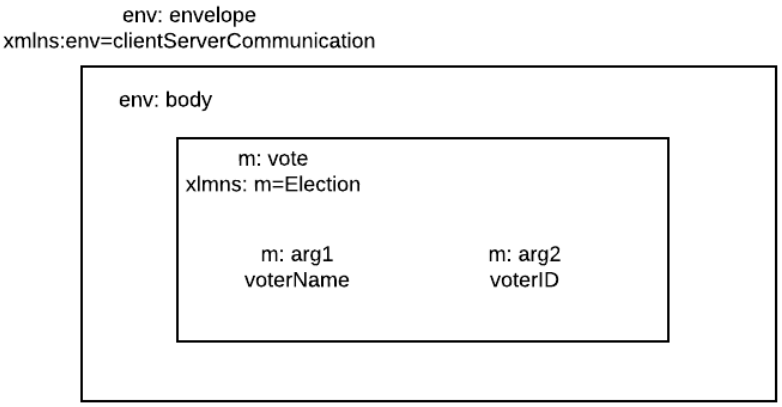
- Privacy or Data Partition
    - Privacy in a fault tolerant system is achieved through the maintaining distinguishable data properties between multiple instances.
    - In that perspective, message passing does not have capability to maintain a shared yet distinguishable data store of its own but DSM can differentiate the data properties based on different instances.

With these properties of a fault tolerant system into consideration, the DSM offers more accountability for the data and its processes in a system and can be leveraged to have fault-tolerant system.

Exercise 9.3

Illustrate the contents of a SOAP Request message and corresponding Reply message in the Election service example of Exercise 5.11, using the pictorial version of XML as shown in Figure 9.4 and Figure 9.5.

SOAP Request Message



Contd…

SOAP Response Message