Dianne Lopez
Kailie Chang
Heriberto Juache
CPSC 351 - 01
10/7/19

Design of Sender and Receiver

Modularization

Two Modules:

- Sender: sends data over the shared memory segment
- Receiver: receives the data from the sender

Concurrency

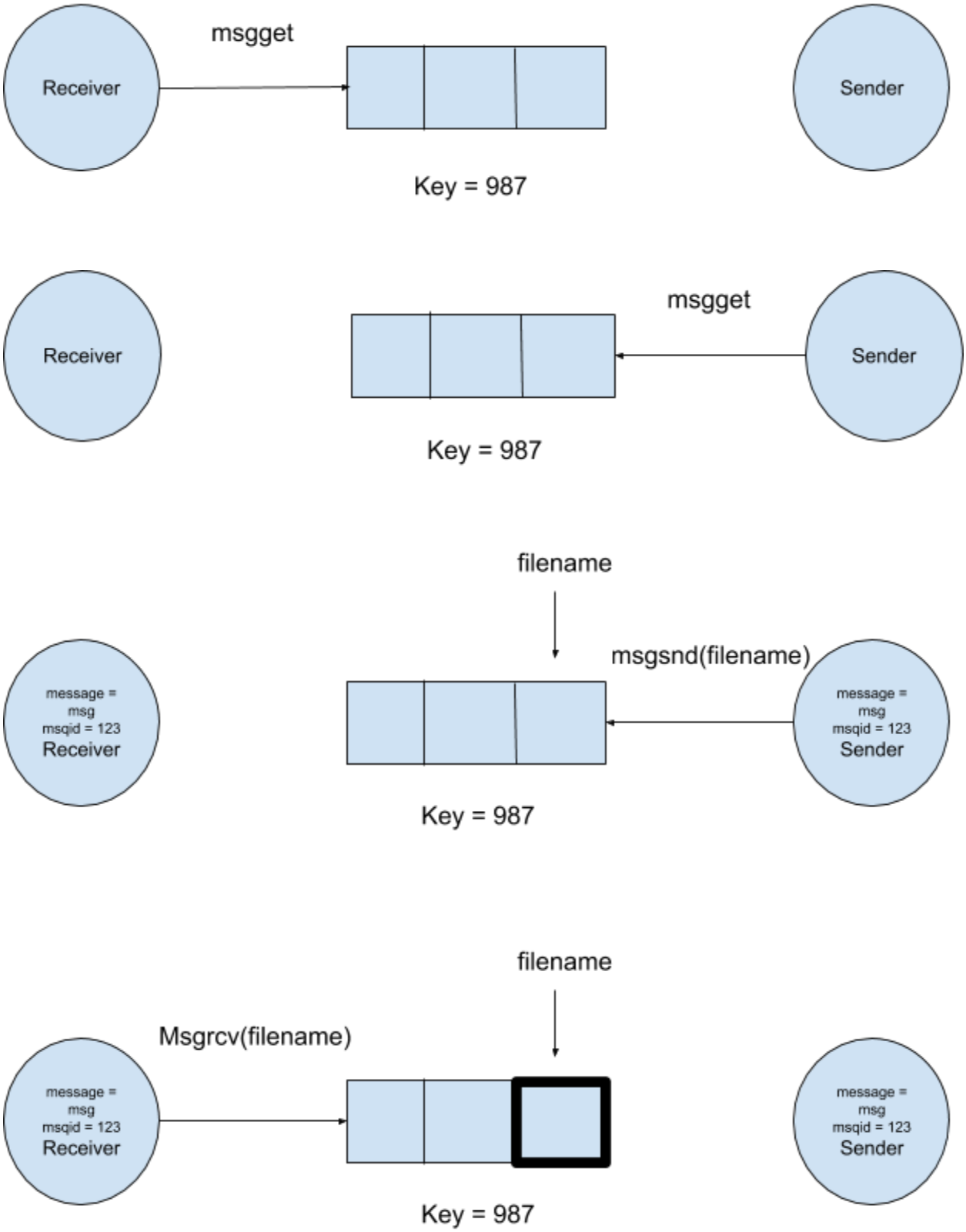
In this application, the sender and the receiver are the modules that are executing in parallel. The receiver is executed first to notify the sender that it is ready to receive messages. The sender is then executed after with an input file to send. Doing this concurrency of the modules allows both sender and receiver to be executed parallel to each other to ensure that the information set and received are the same.

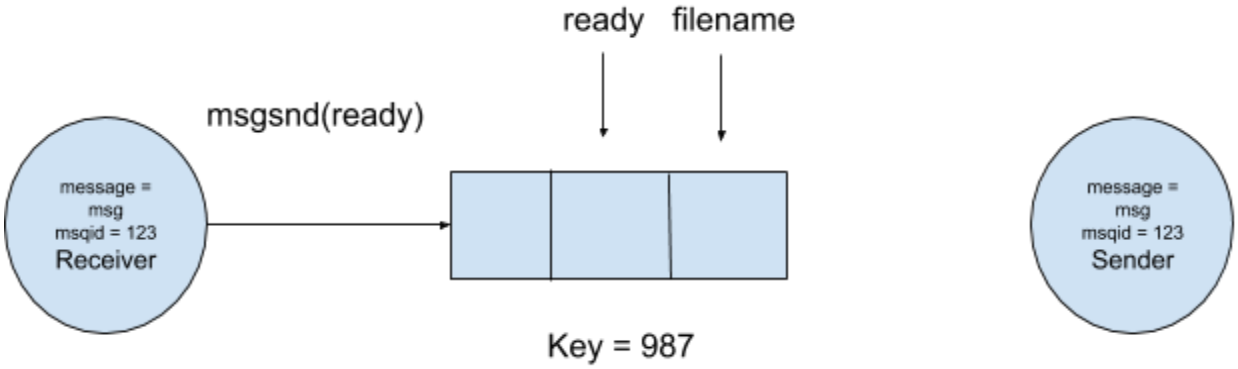
Coupling and Cohesion

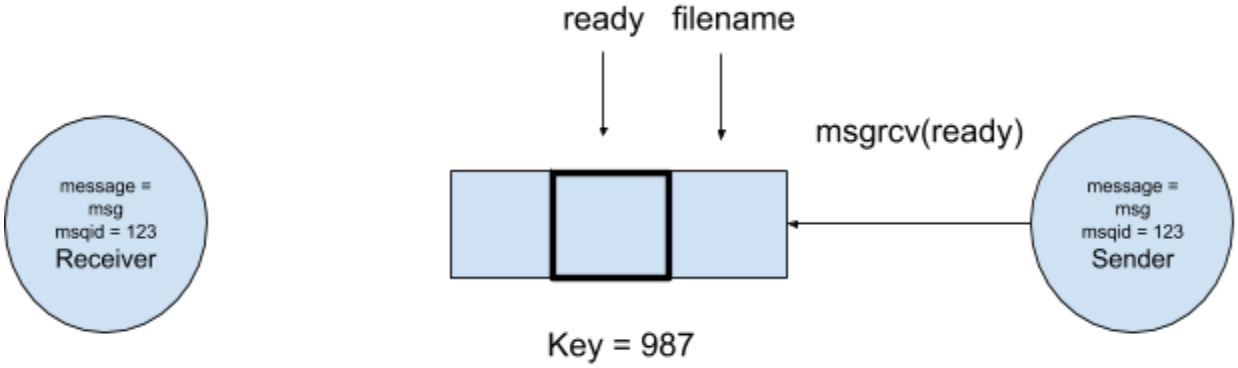
The cohesion between the sender and receiver is functional in terms of the modules being cooperative. The modules being cooperative means that information can be shared/exchanged easily, tasks are broken up into subparts and executed concurrently on multiple processors, system functions are divided into separate processes, and multiple tasks can be done simultaneously.

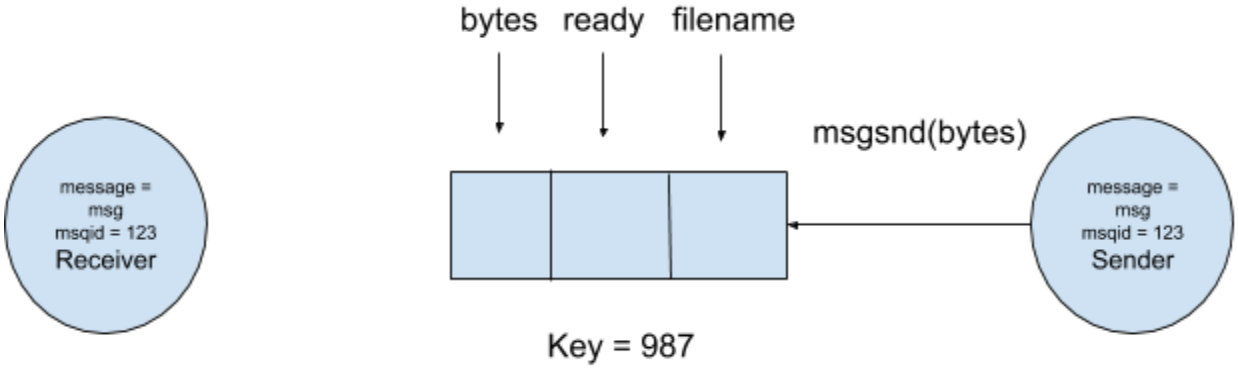
The coupling between the sender and receiver modules is very high. This means that they have a strong relationship with one another and dependent on each other. Since they are passing data between one another, this is considered Data Coupling.

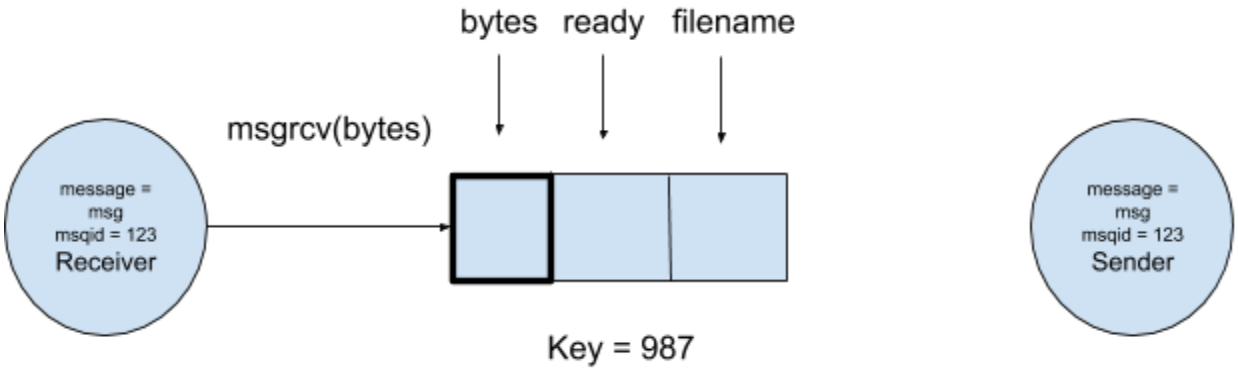
Design Verification




ready
filename
msgsnd(ready)
message =
msg
msqid = 123
Receiver
message =
msg
msqid = 123
Sender
Key = 987


ready
filename
msgrcv(ready)
message =
msg
msqid = 123
Receiver
message =
msg
msqid = 123
Sender
Key = 987


bytes
ready
filename
msgsnd(bytes)
message =
msg
msqid = 123
Receiver
message =
msg
msqid = 123
Sender
Key = 987


bytes
ready
filename
msgrcv(bytes)
message =
msg
msqid = 123
Receiver
message =
msg
msqid = 123
Sender
Key = 987