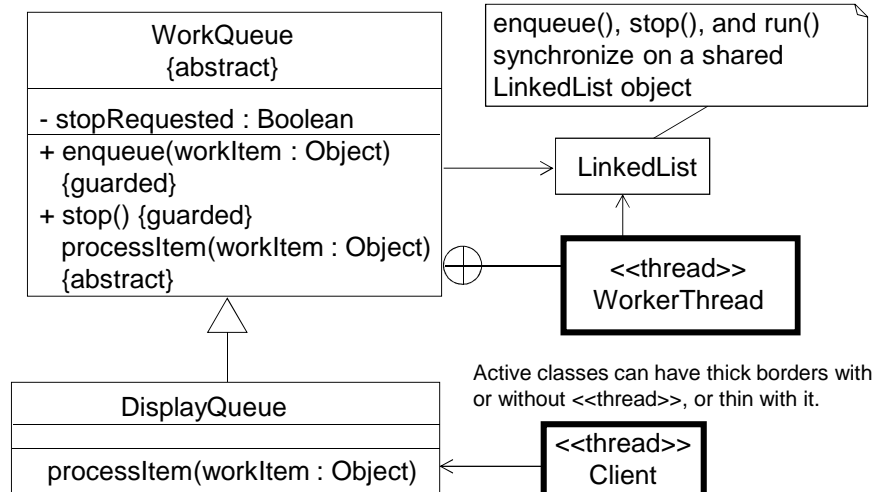

SYSC 3303 Real-Time Concurrent Systems

Assignment 4

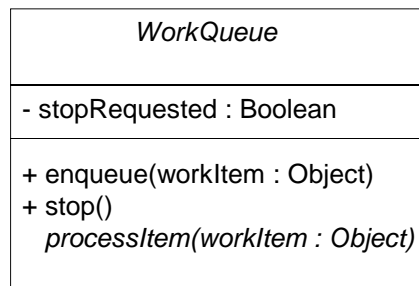
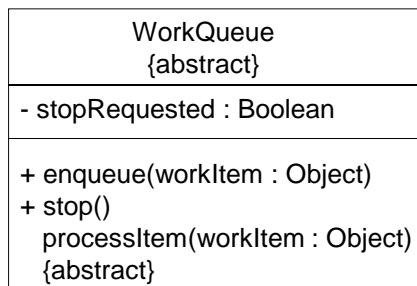
- Copyright © 2014 L.S. Marshall, Systems and Computer Engineering, Carleton University
- Revised April 21st, 2014

Class Diagram (Required)



Notes about Class Diagram

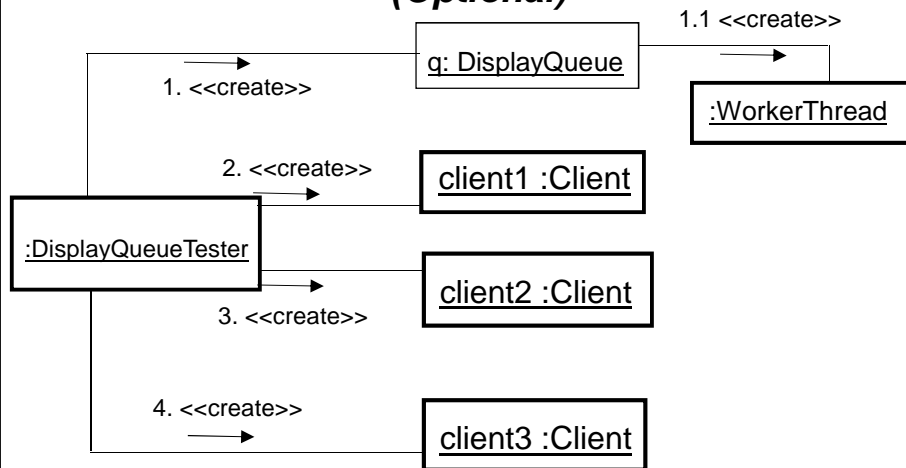
- Abstract things can be specified by rendering their names in *italics*, instead of using the {abstract} property



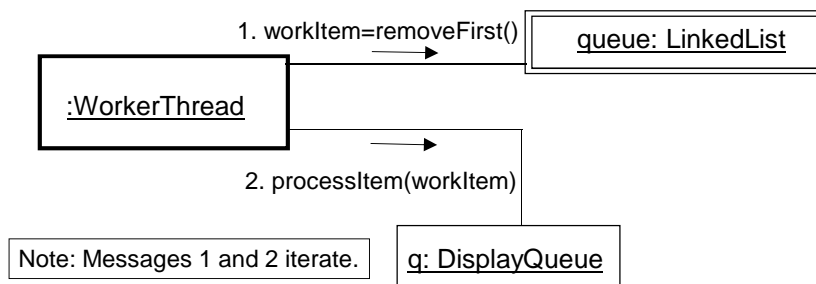
Collaboration Diagrams

- The assignment asked you to draw one collaboration diagram for each active object
- In addition, you could also combine the diagrams into one larger one
- You did not have to show the diagrams including the commented out `displayQ.stop()` in Client's run method, but you could have. (Slide 9 shows a version with this code uncommented.)

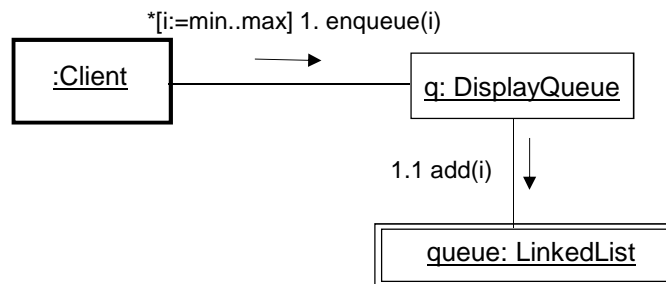
DisplayQueueTester Collaboration Diagram (Optional)



WorkerThread Collaboration Diagram (Required)

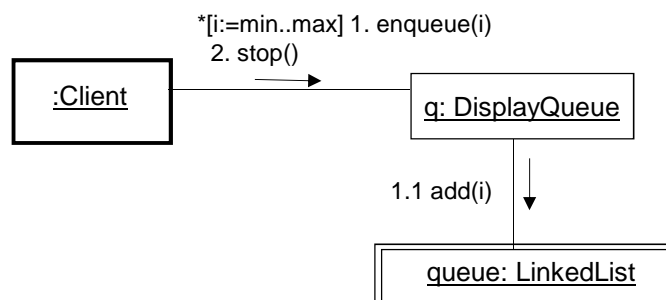


Client Collaboration Diagram (At Least One Required)



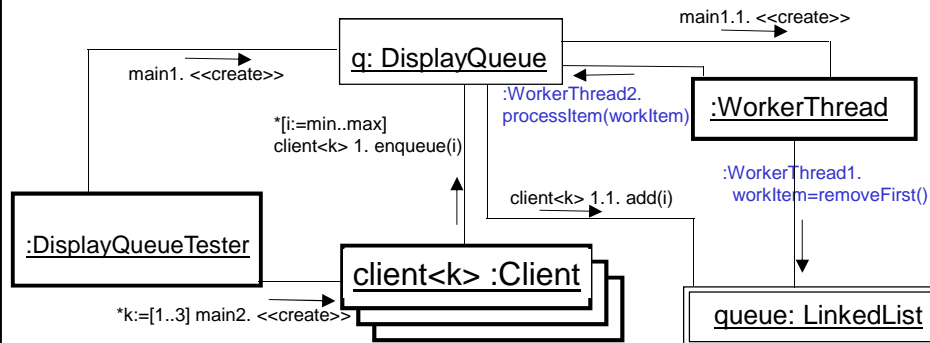
Note: The same collaboration diagram applies for client1, client2, and client3.

Client Collaboration Diagram with stop() shown (Optional)



Note: The same collaboration diagram applies for client1, client2, and client3.

Combined Collaboration Diagram (Optional)



Note: The three client threads could each be shown separately.

Note: WorkerThread Msgs 1 & 2 iterate.

Use Case Maps (UCMs)

- Scenarios (you did not have to include these in your solution):
 - client i creates a work item and adds it to queue
 - worker thread consumes a work item
 - Note: we could also consider this to be one scenario: client creates a work item, and worker consumes it as per slide 27 of the UCM package, giving the resulting UCM on slide 16 of this package.

Use Case Maps (cont.)

Scenario 1: client i creates a work item and adds it to the queue

Responsibilities:

- Client:
 - generate next item
 - add it to queue

Use Case Maps (cont.)

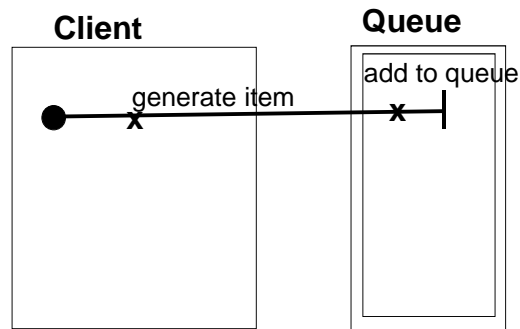
Scenario 2: worker thread consumes a work item

Responsibilities:

- Worker Thread:
 - get next item to be consumed
 - process item

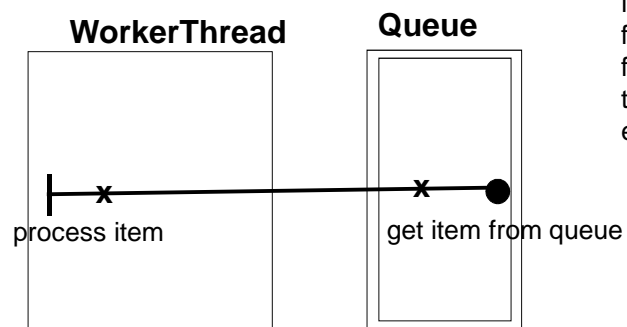
Use Case Map: Map 1 of 2

Note: You had to include one UCM of the system or one for each scenario in your solution.



Note: we are **done** after max-min+1 items have been generated and added to the queue.

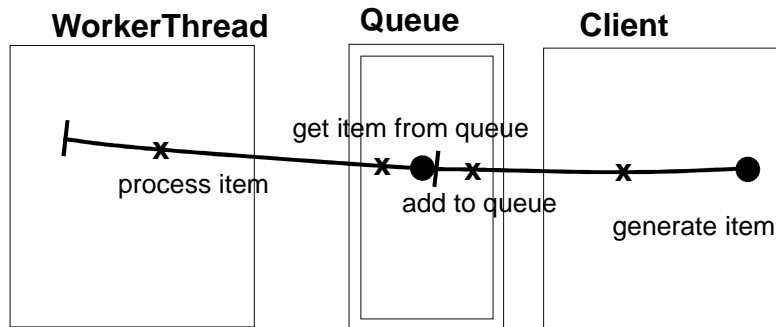
Use Case Map: Map 2 of 2



Note: Precondition for getting an item from the queue is that the queue is not empty.

Note: we are **done** when stop is requested.

Use Case Map: Combined Map



Note: Precondition for getting an item from the queue is that the queue is not empty.

Note: Worker Thread is finished when stop is requested. Client is finished after max-min+1 items have been generated and added to the queue.