THREAD-SAFE QUEUE API FOR EMBEDDED SYSTEMS

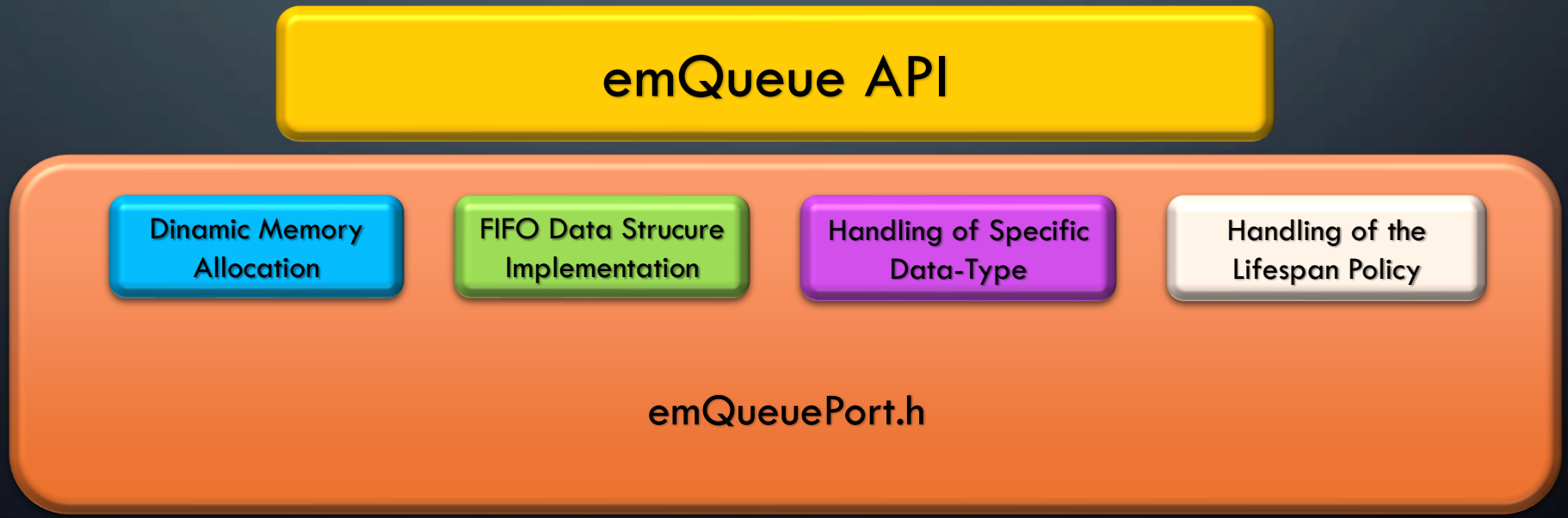
IMPLEMENTATION OF MULTIPLE QUEUES WITH LIFESPAN POLICY

Authors:
Corrado Luca Eugenio
Catalano Luca

EMQUEUE API

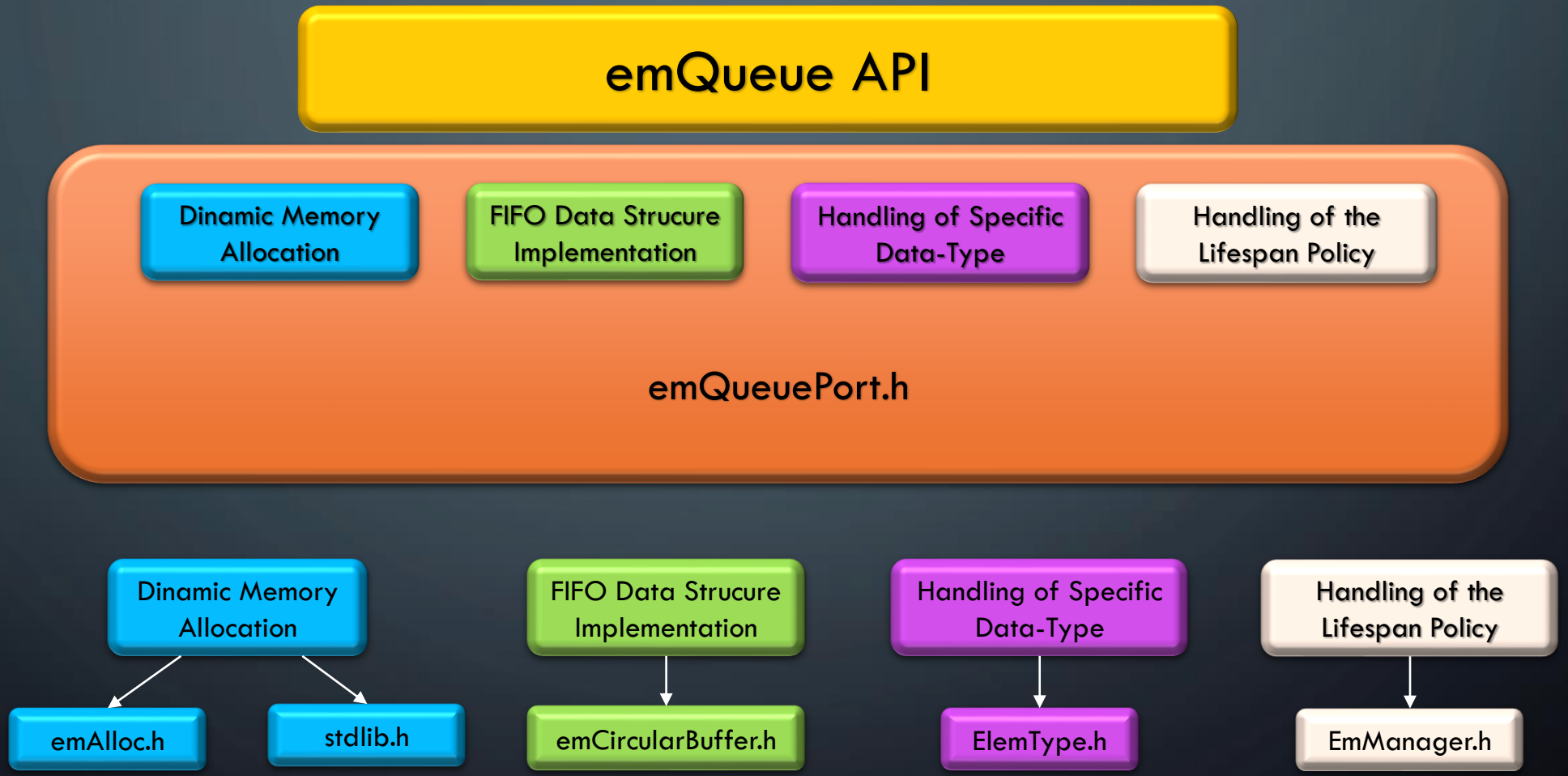
DIRECTORY ORGANISATION

- emQueue.c ⟶ functions implementation
- emQueue.h ⟶ functions declaration
- emQueuePort.h ⟶ declaration of necessary porting functions



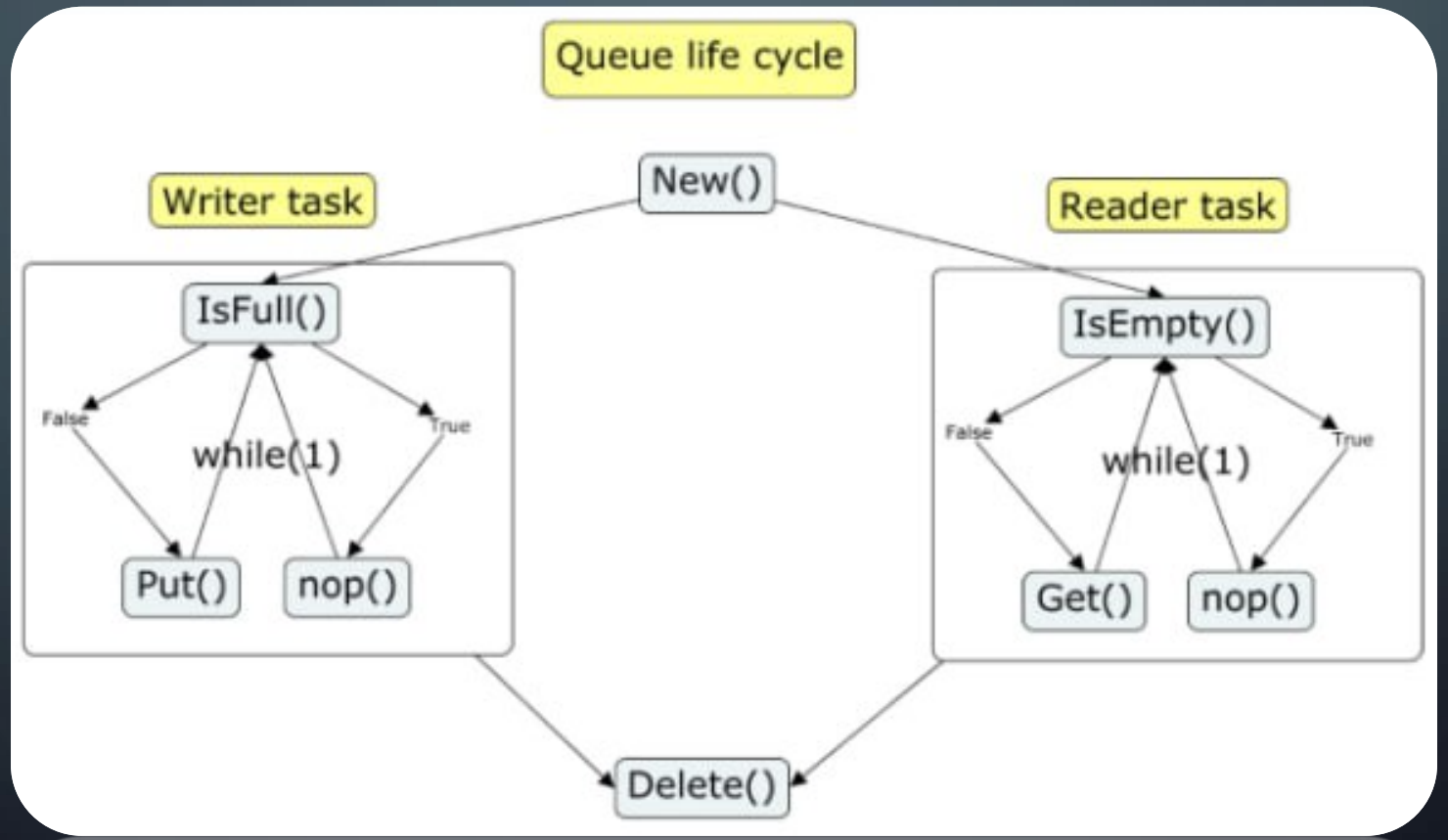
EMQUEUE API

PORTING FUNCTIONS IMPLEMENTATION



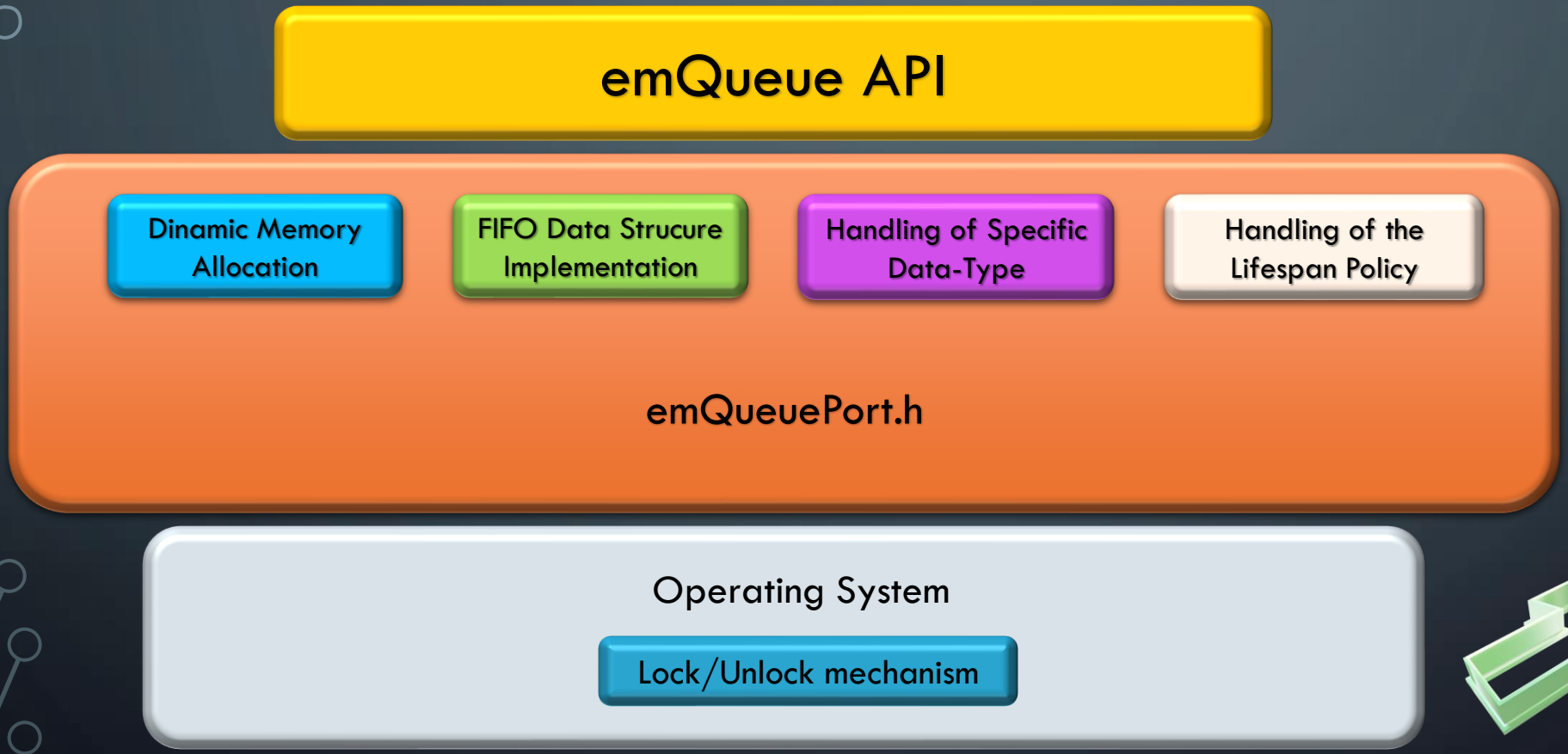
EMQUEUE API

emQueue life cycle



EMQUEUE API

Thread-safe implementation



In porting headers it is possible to configure the library to use **lock/unlock mechanisms** such as semaphores and mutexes

EMQUEUE API

MULTIPLE QUEUES WITH LIFESPAN POLICY

What are multiple queues

- They are data structures, defined by an array, with a predefined size, each with a specific priority

Why multiple queues

- Any item with a shorter lifespan will be associated with the higher priority queue

How are implemented

- A pointer to an array of references to the various priority queues is defined (void**dataStruct)

EMQUEUE API

MULTIPLE QUEUES WITH **LIFESPAN POLICY**

What is the lifespan policy and how it works ?

- The lifespan policy establishes how long an item remains valid
- The value of the lifespan establishes the priority queue to refer to
- The *"lifespan variable"* is contained in a data structure called dataLifespan
- The *"priority"* is based on the relationship between the lifespan value contained in the structure and the *"step variable"*, given by the maximum value of the lifespan divided by the maximum number of queues with priority

EMQUEUE API

MULTIPLE QUEUES WITH LIFESPAN POLICY

AGING FUNCTION

Guarantees an item to move to a higher priority queue

The shift is based on the remaining ***time slack***, which is the difference between the lifespan and the clock time

DELETE FUNCTION

Guarantees the deletion of an item from the multiple queues

The index in the queue is incremented, the head index and the tail index are updated, and eventually the data is overwritten

EMQUEUE API

MULTIPLE QUEUES WITH LIFESPAN POLICY

What about the elements for which the lifespan policy is not considered ?

FIFO queue managment

- The last queue of the set of multiple queues whose number is defined beforehand, is managed in FIFO mode: the value of the policy variable is considered.
- If it is set to 1, the the lifespan will be calculeted, multiple queues will be managed according to the priority obtained and the *aging* and *delete* functions will be used
- If it is set to 0, **there is no lifespan policy, so all elements will have access to the last queue in FIFO mode**