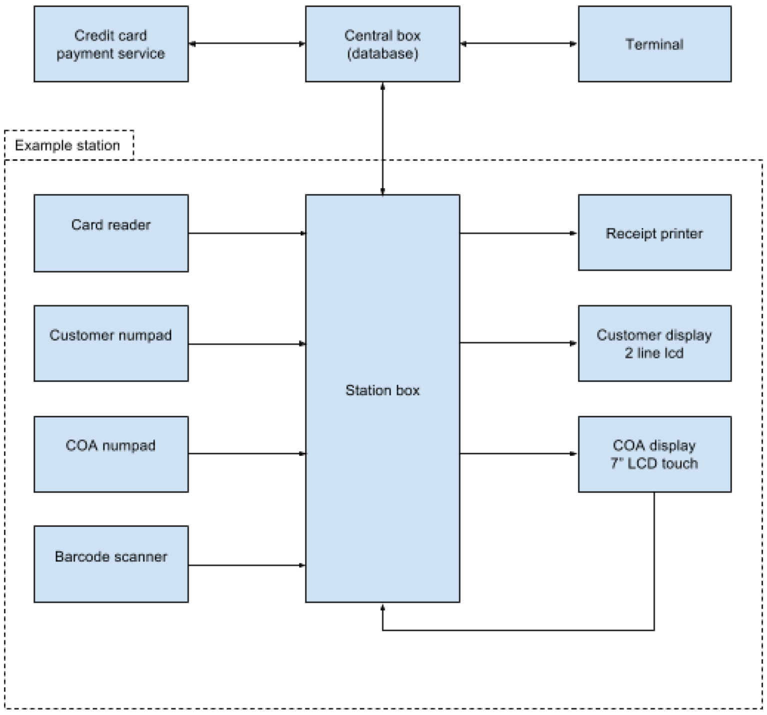
Embedded Software Design

Individual project description for the station box system

This embedded software design project is based on the station box system (as seen below), where the objective is to apply embedded software design patterns and principles in order to develop a fully functional system with respect to requirements specifications that have been defined together with the customer.



The overall objective is work as a team and define a common structure for the implementation of the station box in a way such that the individual components (implemented as tasks) are constrained by a common interface and thus are loosely coupled.

My responsibility is to: (a) design and implement the Task interface, (b) design and implement the Event class, and (c) design and implement the customer display.

The design of the Task interface and Event class must be done in a way that the team can intuitively implement their respective tasks, yet still be constrained and implement any necessary "behind the scenes" functionality, such as communication between tasks; specifically, the biggest hurdle was to decide on how the Task interface handles inter-thread communication (queues, files, shared memory). The proposed solution is to use an event-driver task structures (with a commonly defined Event class) and define a Task interface which constrains both common and required abstract methods for the derived tasks to implement.

The customer display is implemented by following the defined structure and using the Task interface; first, the actual `DisplayDriver` is defined. Then, a `DisplayDriverTask` is defined for the station box to make use of. The `DisplayDriverTask` class encompasses the thread necessary to concurrently interact with the Display hardware. It inherits from the \`Task\` class and uses the `DisplayDriver` class to communicate with the hardware. It provides data structures and high-level methods to easily send data to the display.