Solution to Exercise 3a

OO: Applet Design, Java Card RMI, Optimizations

There are three distinct transaction-related problems in the provided applet:

1. In the method `verfyPin()`, a transaction is incorreclty used because the two fields [`pin_tries`, `PIN_VERIFIED`] are unrelated and need not be mutually consistent. `PIN_VERIFIED` is a transient field, which would not participate in the transaction anyway. Not only is the transaction unnecessary, it creates a vulnerability: If PIN verification fails, an exception is thrown. This causes the method to exit prematurely, never allowing the transaction to complete. When the `process()` method exits, the JCRE will rollback the changes, including the incremented `pin_tries`. This circumvents velocity checking, allowing a brute-force attack on the PIN.

2. In the method `updatePin()`, a transaction is used because the two fileds [`PIN`, `pin_size`] must be mutually consistent. However, `Util.arrayCopyNonAtomic()` is used to copy the `PIN` contents. This method, as the name implies, does not participate in the transaction, potentially causing an inconsistent state.

3. The constructor calls the mehod `updatePin()`, which uses a transaction. The constructor is called during `install()`, where a (system) transaction is already in progress. Thus, it is not only unnecessary, but will cause an exception to be thrown, as Java Card does not support nested transactions.