Analogy for difference between usages of interface and abstract class - I

When I was in the Air Force, I went to pilot training and became a USAF (US Air Force) pilot. At that point I wasn't qualified to fly anything, and had to attend aircraft type training. Once I qualified, I was a pilot (Abstract class) and a C-141 pilot (concrete class). At one of my assignments, I was given an additional duty: Safety Officer. Now I was still a pilot and a C-141 pilot, but I also performed Safety Officer duties (I implemented ISafetyOfficer, so to speak). A pilot wasn't required to be a safety officer, other people could have done it as well.

All USAF pilots have to follow certain Air Force-wide regulations, and all C-141 (or F-16, or T-38) pilots 'are' USAF pilots. Anyone can be a safety officer. So, to summarize:

- Pilot: abstract class
- C-141 Pilot: concrete class
- ISafety Officer: interface

Analogy for difference between usages of interface and abstract class - II

- The abstract class inheritance is used when the derived class shares the core properties and behaviour of the abstract class. The kind of behaviour that actually defines the class.
- On the other hand interface inheritance is used when the classes share peripheral behaviour, ones which do not necessarily define the derived class.
- For eg. A Car and a Truck share a lot of core properties and behaviour of an Automobile abstract class, but they also share some peripheral behaviour like Generate exhaust which even non automobile classes like Drillers or PowerGenerators share and doesn't necessarily defines a Car or a Truck, so Car, Truck, Driller and PowerGenerator can all share the same interface IExhaust.