

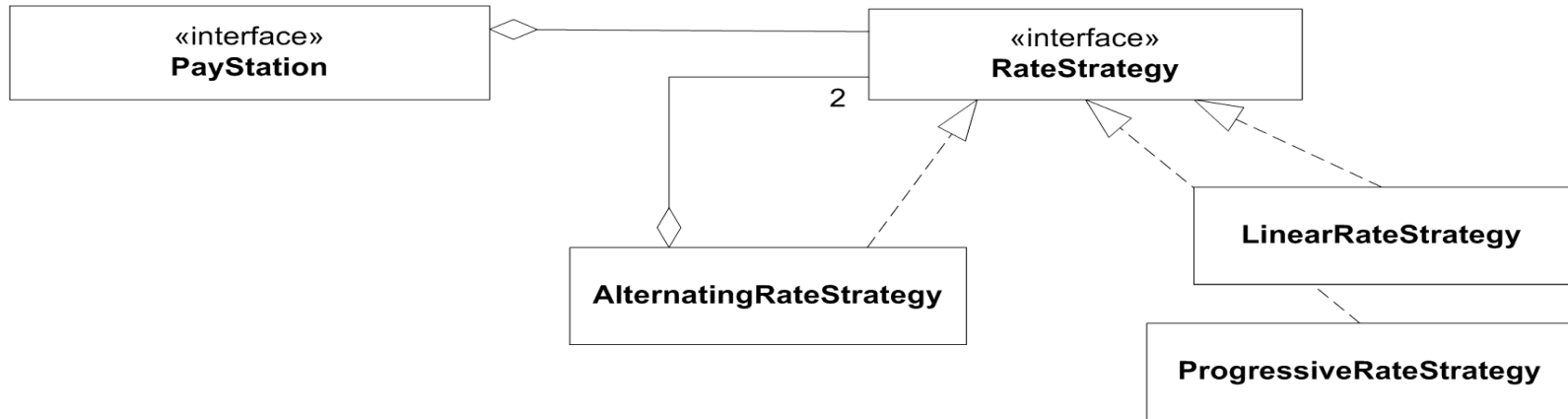


AARHUS UNIVERSITET

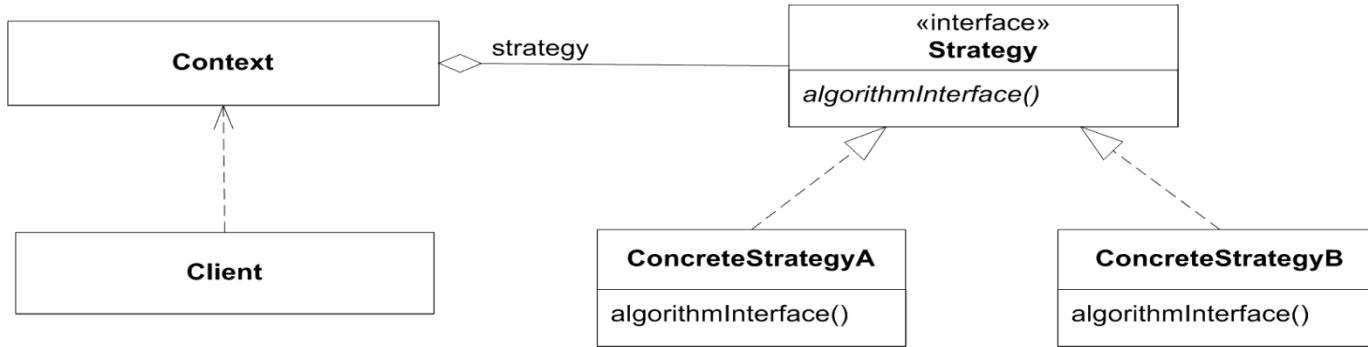
Software Engineering and Architecture

What Patterns are ... and what not...

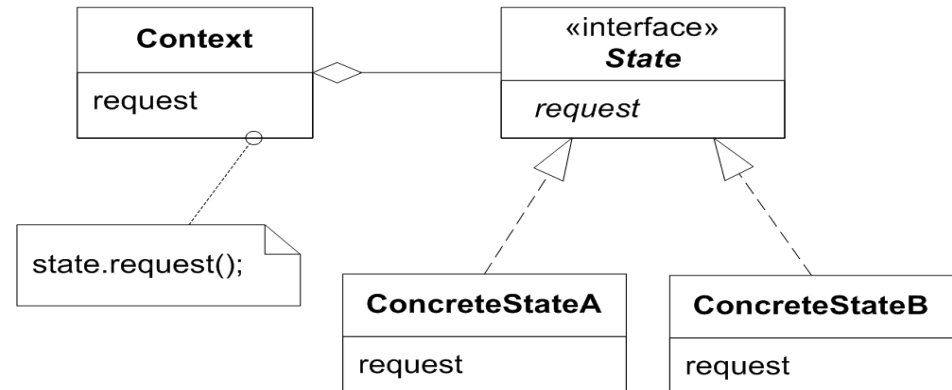
Let us compare this...



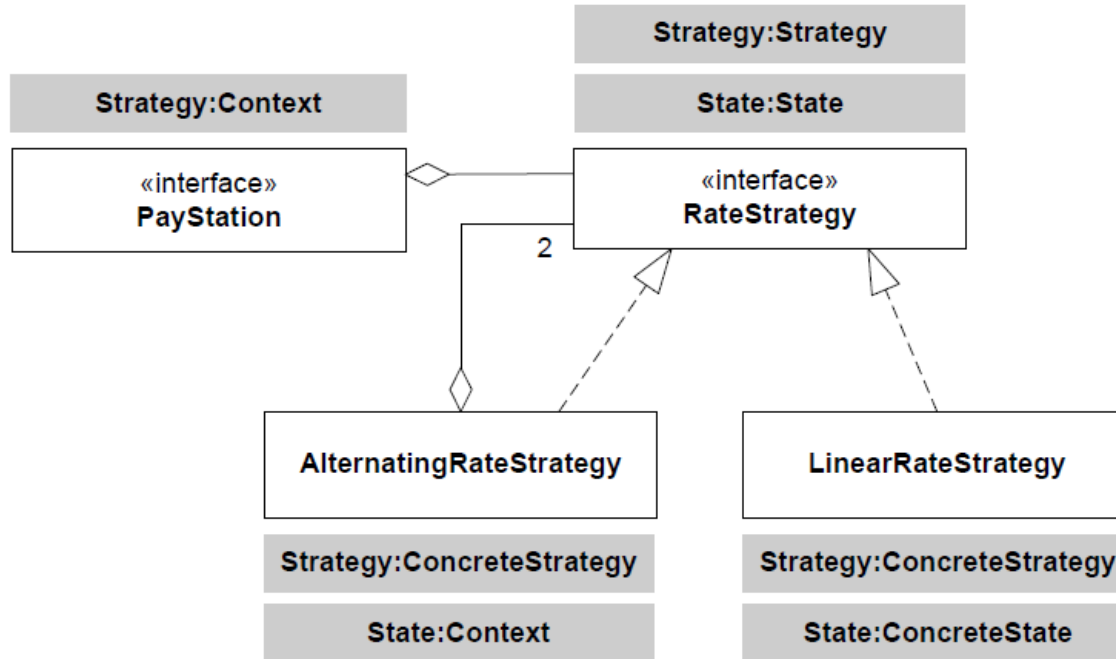
If these diagrams are correct, then something is wrong



One interface named State, one named Strategy...



- UML diagrams cannot express **roles** – and patterns express roles, **not** classes!



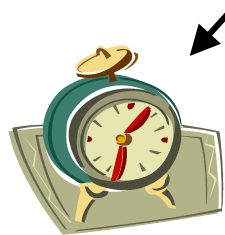


Revisiting

rate calculation request



Team leader



1. check clock

2. delegate to expert



Rate Policy Expert
Linear



Rate Policy Expert
Progressive

The Team Leader

- The AlternatingRateStrategy instance
 - calculates rates = **Concrete Strategy**



Rate Policy Expert
Alternating rates

- changes behaviour depending on = **Context**



Context for state changes

- **The same object plays roles in two patterns!**

- The essence of design patterns is at a higher level of abstraction than what you may see in e.g. UML class diagrams.

Definition: Design pattern (Role view)

A design pattern is defined by a set of roles, each role having a specific set of responsibilities, and by a well-defined protocol between these roles.

- You are *not* restricted by naming, method naming, classes - *only* by the roles and the protocol !
- often a single abstraction plays multiple roles !