Which paradigm is best?

the paradigm paradox

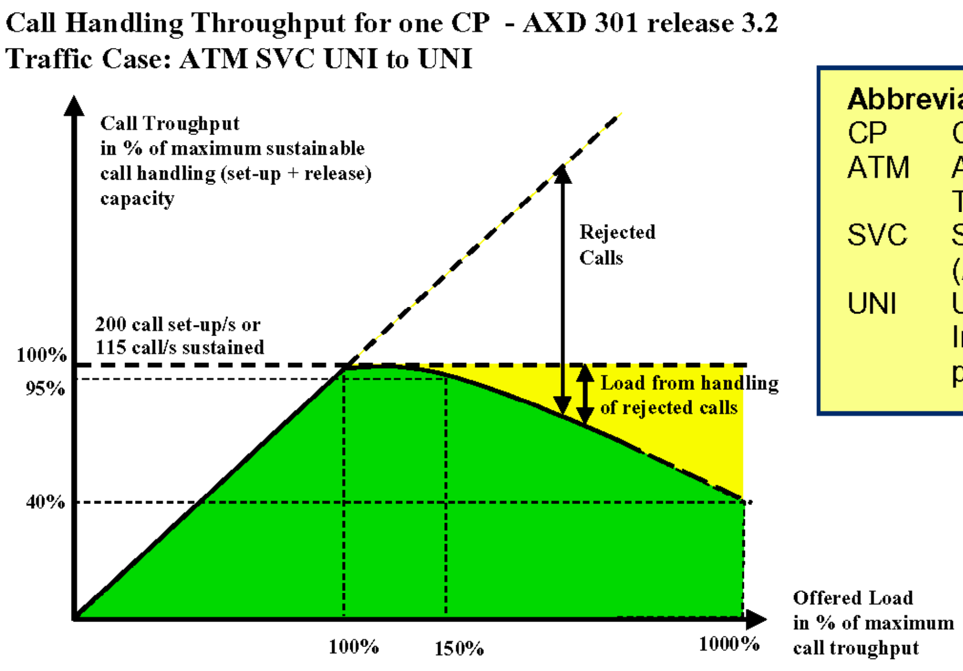
- Each is best for a particular kind of problem
  - None is best overall: *"more is not better or worse, only different"*
- The conventional boundaries between paradigms are completely artificial (they exist only for historical reasons)
  - Java is only object-oriented ⇒ **too limited**
  - Scala is functional, object-oriented, and actor-based ⇒ **better**
- A big program almost always needs several paradigms
  - This is why you need to know multiple paradigms
- A good language should support several paradigms
  - This is hard for industry languages (Scala and Erlang are moving in the right direction; Java and C++ are bogged down by legacy code)
  - In this course we have used Oz: a research language that supports many paradigms (Oz ideas are slowly moving to industry…)

A large example

Ericsson AXD 301 ATM Switch: >1 million lines of Erlang

✓ **Erlang**: Concurrent and independent by default, asynchronous messages, multi-agent programs

✗ **Java**: Sequential and monolithic by default, synchronous RMI, shared-data programs



Abbreviations:

CP	Control Processor
ATM	Asynchronous Transfer Mode
SVC	Switched Virtual (ATM) Channel
UNI	User-Network Interface signaling protocol

- Object-oriented programming is the wrong paradigm for Internet programming!
  - Important: isolation, concurrency, asynchronous messages, higher-order programming
  - Unimportant: inheritance, classes, methods, UML diagrams, monitors