Data-Oriented Design

Friday, 24 May 2024 14:03

"Data dominates. If you've chosen the right data structures and organised things well, the algorithms will almost always be self-evident. Data structures, not algo programming." - Rob Pike

Design Philosophy:

- If you don't understand the data, you don't understand the problem.
- All problems are unique and specific to the data you are working with.
- Data transformations are at the heart of solving problems. Each function, method and work-flow must focus on implementing the specific data transfo solve the problems.
- If your data is changing, your problems are changing. When your problems are changing, the data transformations needs to change with it. If we are c algorithms as data keeps changing, sometimes it creates cascading changes across entire codebase which is very painful. This is when we should st decoupling the code from data changes so that cascading changes are minimised.
- Uncertainty about the data is not a license to guess but a directive to STOP and learn more. You can write code for the transformations/algorithms tha about the input and output of.
- Solving problems you don't have, creates more problems you now do.
  We are writing code for today, we are going to design an architect for tomorrow. Don't add more code to the code you need today than you do.
  The more lines of code you write, probability of more bugs increases. So, write only as much as required.
- If performance matters, you must have mechanical sympathy for how the hardware and operating system work.
- Minimise, simplify and REDUCE the amount of code required to solve each problem. Do less work by not wasting effort.
- Code that can be reasoned about and does not hide execution costs can be better understood, debugged and performance tuned.
- Coupling data together and writing code that produces predictable access patterns to the data will be the most performant.
- Changing data layouts can yield more significant performance improvements than changing just the algorithms.
- Efficiency is obtained through algorithms but performance is obtained through data structures and layouts.

Resources:

Data-Oriented Design and C++ - Mike Acton
Efficiency with Algorithms, Performance with Data Structures - Chandler Carruth