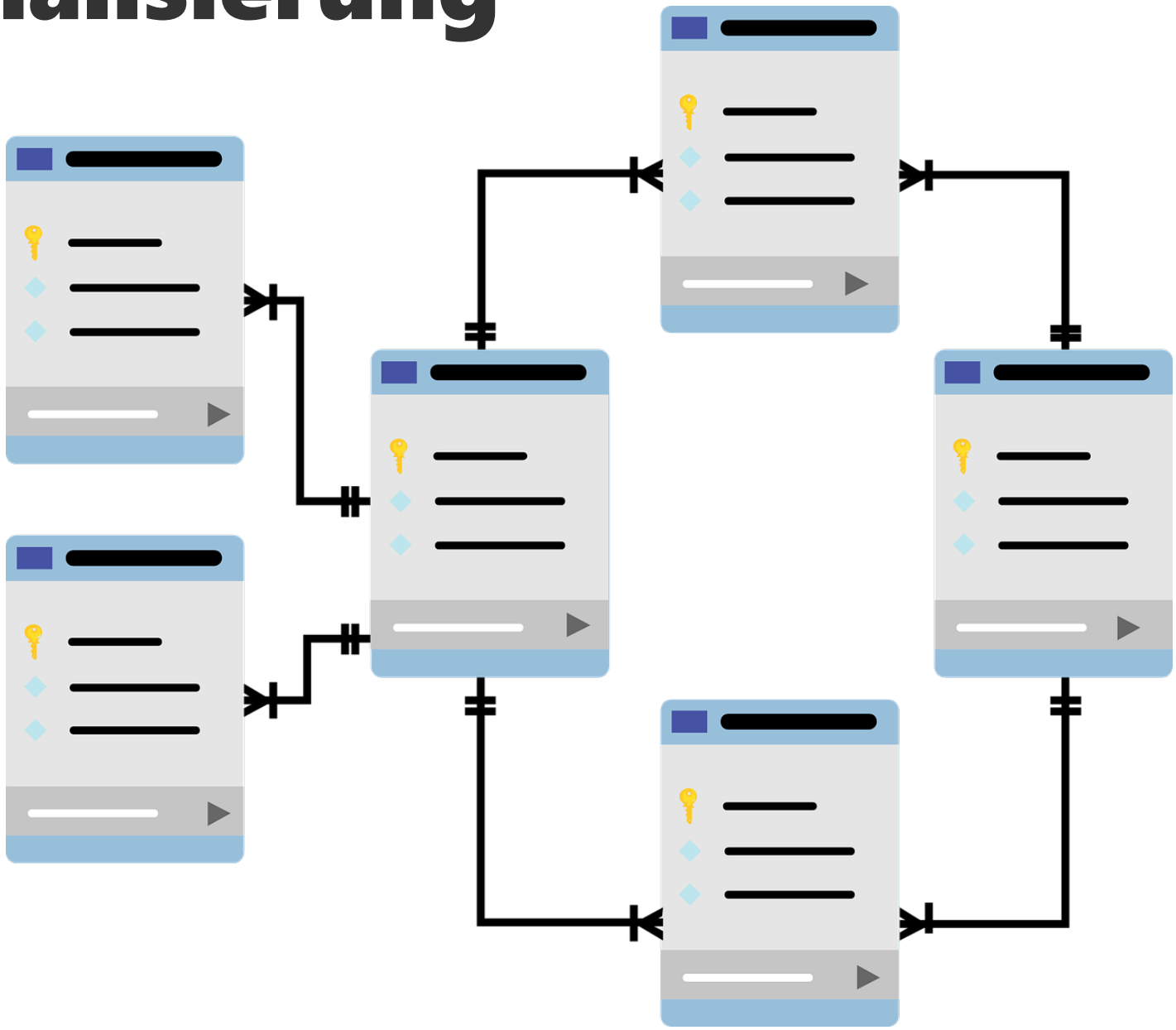
Voll Normal!

Unbürokratische Datenmodelle

Perspektive

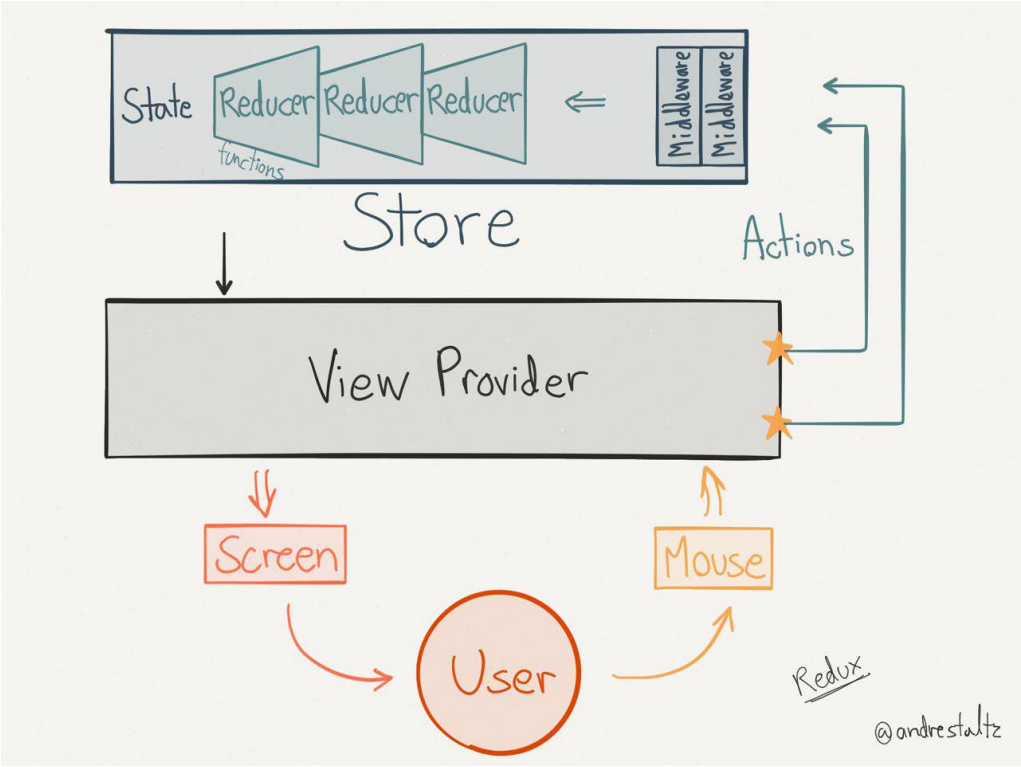
Normalisierung



Wikipedia

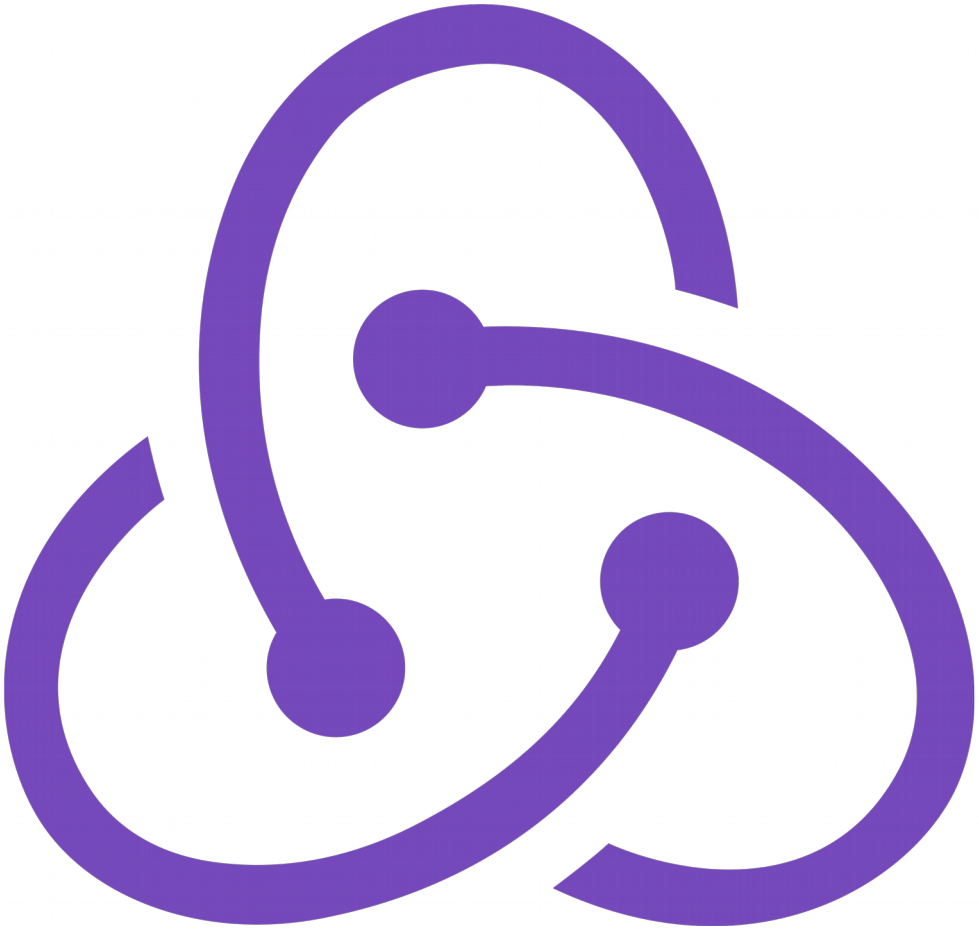
„Database normalization is the process of structuring a relational database in accordance with a series of so-called normal forms in order to reduce data redundancy and improve data integrity.“

Unidirectional UI



https://staltz.com/unidirectional-user-interface-architectures.html

Redux



Normalizing State Shape

- Duplizierte Daten updaten ist schwer
- „trying to update a deeply nested field can become very ugly very fast“
- Geschachtelte Daten vertragen sich nicht gut mit der Kombination Observability/Immutability

Coupling

https://redux.js.org/recipes/structuring-reducers/normalizing-state-shape

Performance

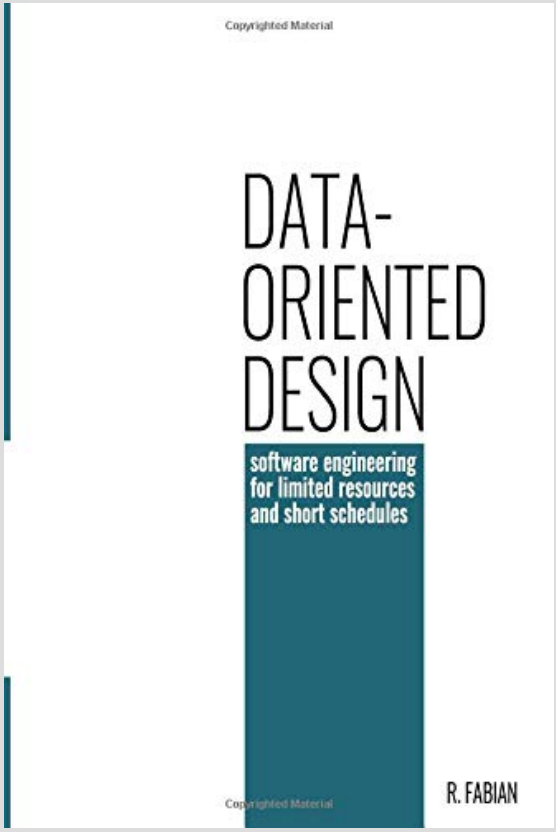
„For maximum rendering performance in a React application, state should be stored in a normalized shape [...]“

[Haha, „maximum“...]

C++?!

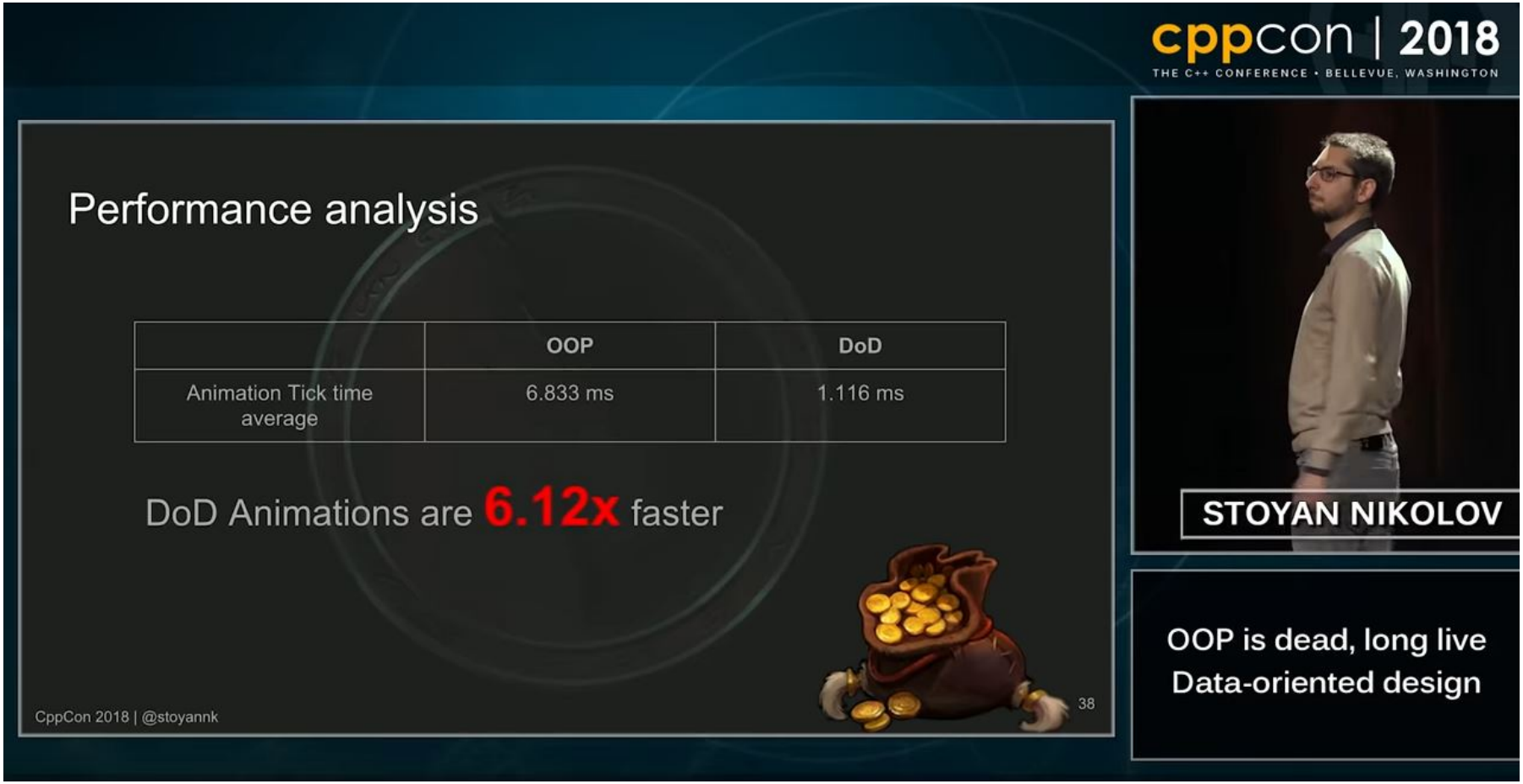
https://isocpp.org

Data-oriented design



http://www.dataorienteddesign.com

OOP is dead?



https://youtu.be/yy8jQgmhbAU

Wieder normal?

Also?

- Einfacher zu Programmieren
- Korrektheit ist leichter
- Kuppelt weniger
- Ist performanter!

Meta-Architektur

- Gute Typen!
- Werkzeuge für den Rest
- Der C++ Hammer
- Kapselung hier wichtig!

Vokabular

Datenmodell

- Komposition (structs?)
- Nur Daten in Containern
- Zusammenhänge durch Handles (IDs, ABCs)
- Gehört „der Applikation“
- Nicht kapseln!

Transformationen

- Zustandslos*
- N Tabellen auf M Tabellen
- OOP / Polymorphismus
- Behavioural Patterns
- Die meinste Komplexität?

Und nun?

- Einfach starten!
  structs, Funktionen, `std::vector`
- OOP Libs als Daten sehen? (Qt?)
- Daten statt Funktionsaufrufe
- „Tiefe“ Patterns einschränken (hierarchical structures, DI, Callbacks)