Topologies: do less

- General advice: where possible, design a topology with a single purpose
- As more functionality gets mashed into single topology, problems like conflict of configuration pop up

Experience report

- By choosing sound abstractions, we are getting performance for free as the library matures
    - Certainly, if you were to start using Kafka streams today, a lot of the pain points mentioned in this talk would not exist!
    - Streams API constantly improving (new features)
- Streams DSL almost meets expectations
    - You **will** need to tweak configuration (of which there is many) to meet your use case
    - You **will** need to dig into implementation details
    - Can be difficult to reason about resulting topology, despite functional DSL
    - We have been able to write extremely concise, functional code that will continue to meet our scalability concerns as we grow.
- Future challenges: complexity in “migrations”/breaking data changes