Identifying mismatches between microservice testbeds and industrial perceptions of microservices

Vishwanath Sehagiri† *, Darby Huye‡ *, Lan Liu‡, Avani Wildani†, Raja R. Sambasivan‡
†Emory University, ‡Tufts University

Introduction

- Microservice architecture: de-facto way to build distributed apps in industry
  - Goal: increase deployment velocity and reduce coordination across teams
  - Applications designed as loosely-coupled services that:
    - each provide distinct functionalities
    - interact via language-agnostic protocols

- Problem: little known about industrial microservices apart from above
  - E.g., Communication methods, service sizes, topological characteristics
  - E.g., How different organization’s architectures vary
  - Depresses ability to perform impactful research in this area

- This Work: A User study with microservice developers to characterize the design space of industrial microservice architectures
  - Identify where existing open-source testbeds’ design choices are too narrow

Existing open-source testbeds

- Open-source microservice testbeds adopt a narrow set of design choices
  - E.g., DeathStarBench, TrainTicket, BookInfo
  - Likely not representative of:
    - the variety of designs present in industry
    - any single industrial microservice architecture
    - Yet, used to inform much research and development on microservices
- Research using them may be useful to narrow or ill-defined microservice designs

Methodology

Systematization of existing testbeds design axes and choices

- Performed via analysis of their codebases and published literature

Interviews with real microservice developers

- Recruited 12 total participants
  - Initial participants via social media posts (e.g, Reddit, Twitter)
  - initial participants suggested other participants (snowball sampling)
  - 32 total questions designed to probe industrial microservice designs

Expanded systematization via analyses of participants’ responses

- Identified designs not present in testbed-only systematization

(Some) key mismatches

- Communication:
  - Testbeds use single, uniform communication protocol
  - Industrial architectures use multiple ones that differ in use of serialization, REST vs. RPC, and performance sensitivity

- Topology:
  - Industrial architectures’ topologies extremely varied
  - Some grow organically w/o a prescribed shape
  - Testbeds’ topologies prescribed to be hierarchical
  - Cycles common in industry but not in testbeds

- Service reuse:
  - Testbeds have very limited service reuse
  - Industrial architectures can exhibit significant reuse
  - One participant said this was key reason for microservices

- Other observations about participants’ responses:
  - They disagreed on what constitutes a service
  - Could not agree on scope of a single service

(Some) of our expanded design axes

<table>
<thead>
<tr>
<th>Axis</th>
<th>Category</th>
<th>Industry possibilities</th>
</tr>
</thead>
<tbody>
<tr>
<td>Communication</td>
<td>Protocol: HTTP, RPC, both</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Manner:  Sync, Async, both</td>
<td></td>
</tr>
<tr>
<td>Topological</td>
<td>Structure: Hierarchical, non-hierarchical, star</td>
<td></td>
</tr>
<tr>
<td>characteristics</td>
<td>Cycles?: Endpoint, service, none</td>
<td></td>
</tr>
<tr>
<td>Service reuse</td>
<td>Service defn: Business use case, single team, etc.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Within apps: Yes, No</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Across apps: Yes, No</td>
<td></td>
</tr>
</tbody>
</table>

Conclusions & future work

- Started this effort because of concerns research using existing testbeds may force us to use invalid assumptions
- Found industrial architectures vary greatly from testbeds
- Implications for microservice optimization and tooling
  - E.g., services with cycles should be scaled together
  - E.g., in-network serialization not always applicable
  - E.g., aggregate analyses depend on reuse characteristics

Future work:

- Broader microservice testbeds
- Categorization of different type of microservice architectures