Problems Solved by Microservices

Memi Lavi
www.memilavi.com

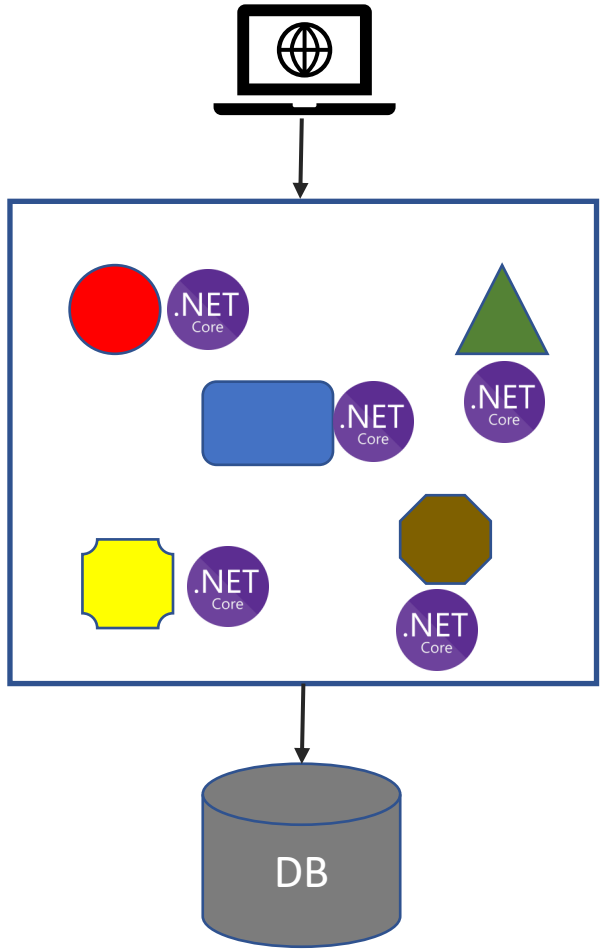
Problems Solved

- We discussed problems caused by Monolith and SOA
- Microservices solve these problems
- Let's see how…

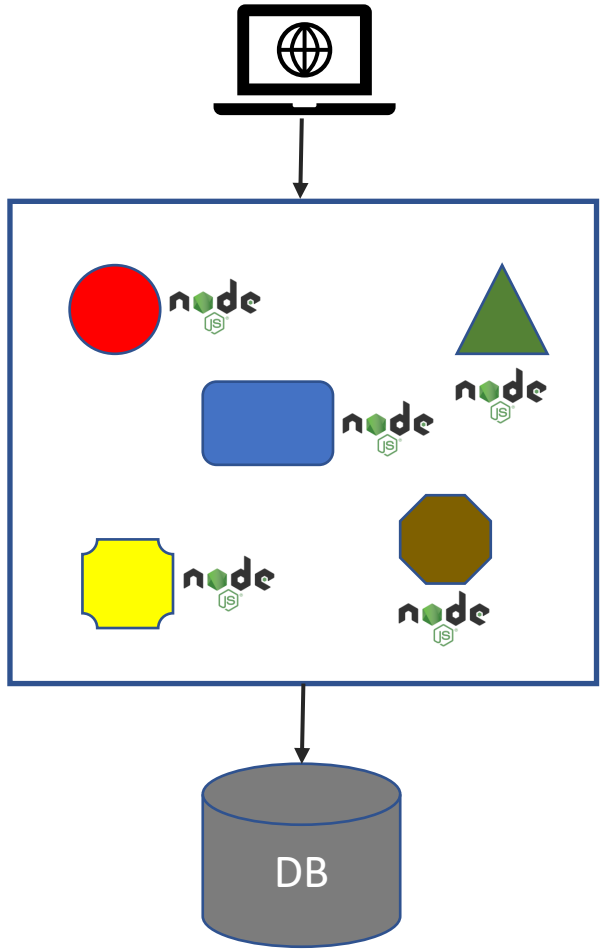
Single Technology Platform

- With monolith, all the components must be developed using the same development platform
- Not always the best for the task
- Can't use specific platform for specific features
- Future upgrade is a problem – need to upgrade the whole app

Single Technology Platform

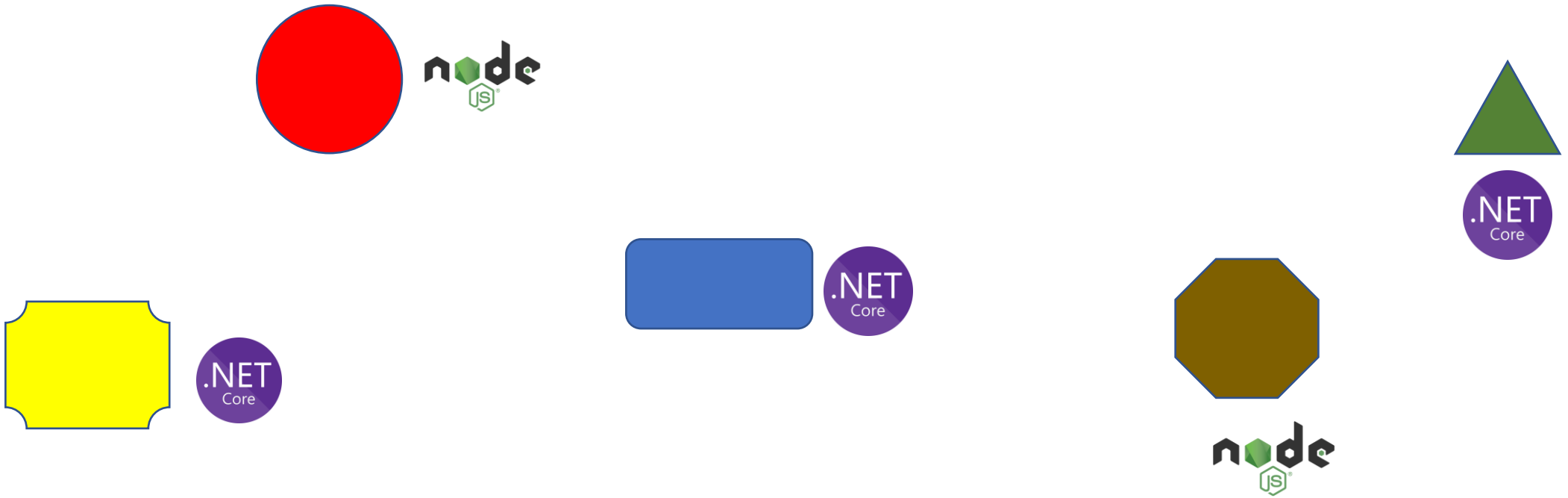


Single Technology Platform



Single Technology Platform

- With Microservices, the Decentralized Governance attribute solves it



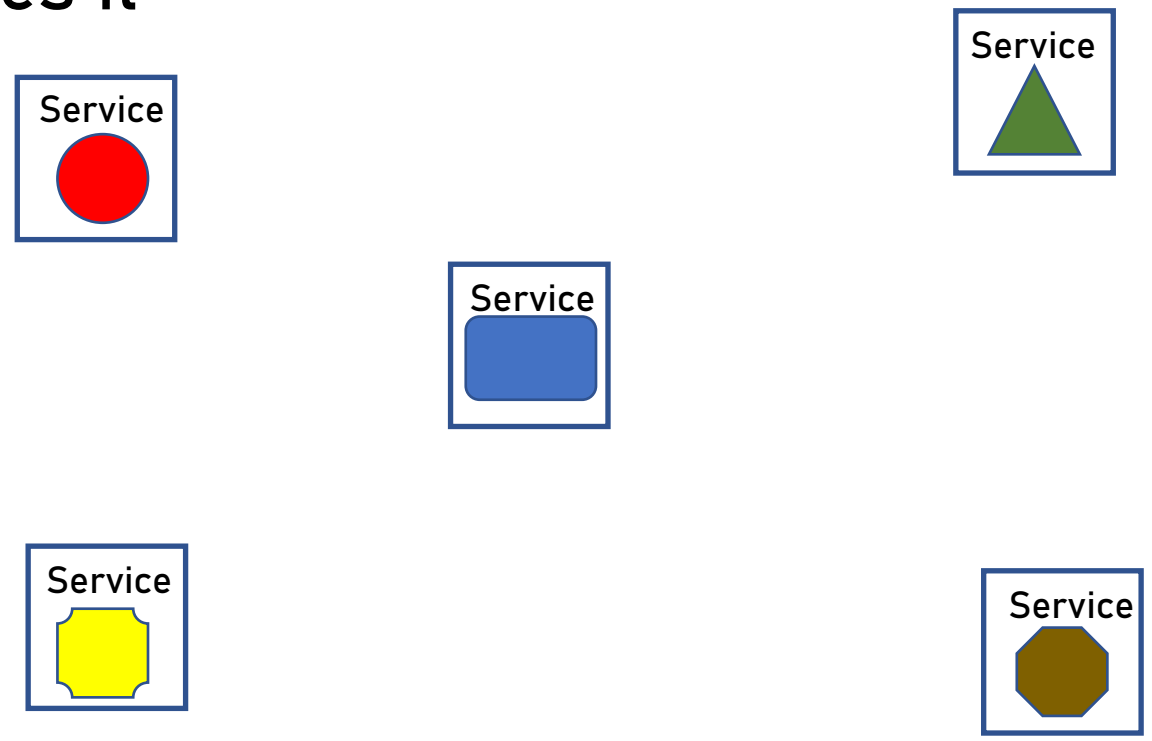
Inflexible Deployment

- With monolith, new deployment is always for the whole app
- No way to deploy only part of the app
- Even when updating only one component – the whole codebase is deployed
- Forces rigorous testing for every deployment
- Forces long development cycles

Inflexible Deployment

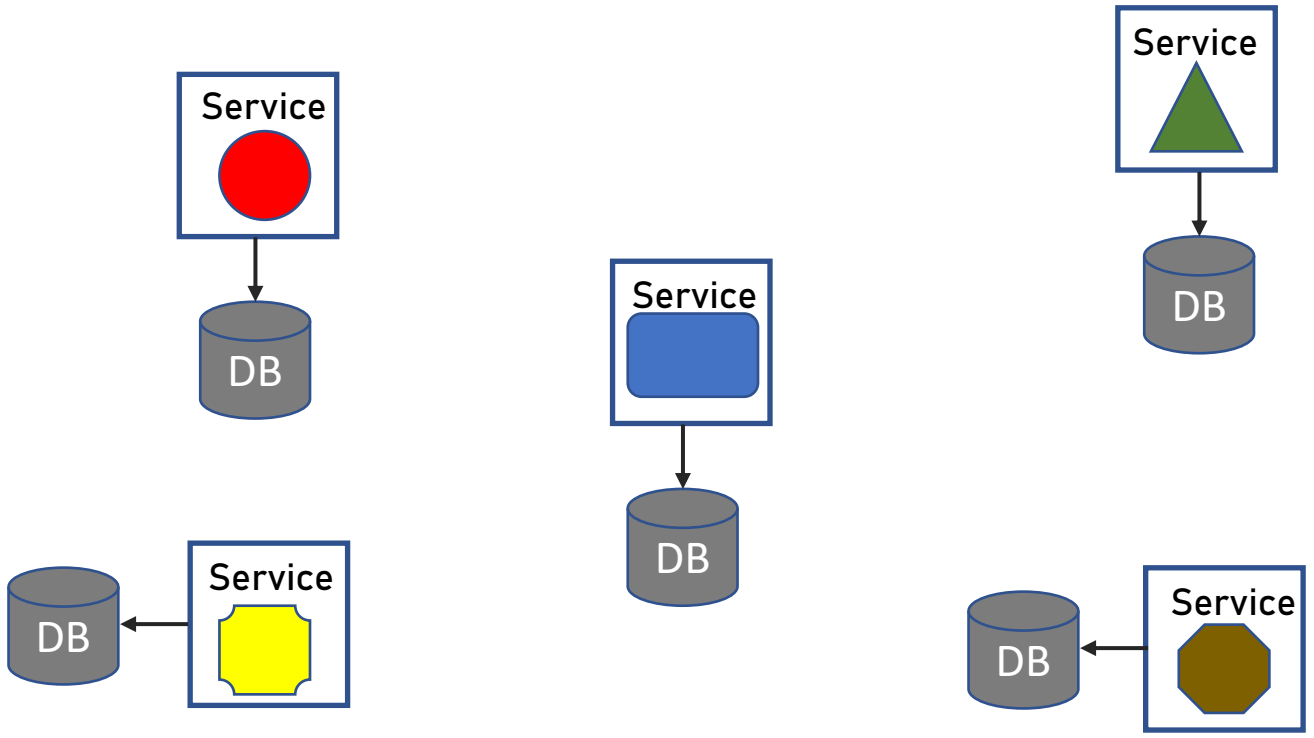
- With Microservices, the Componentization via Services

  attribute solves it



Inflexible Deployment

- Also – Decentralized Data Management

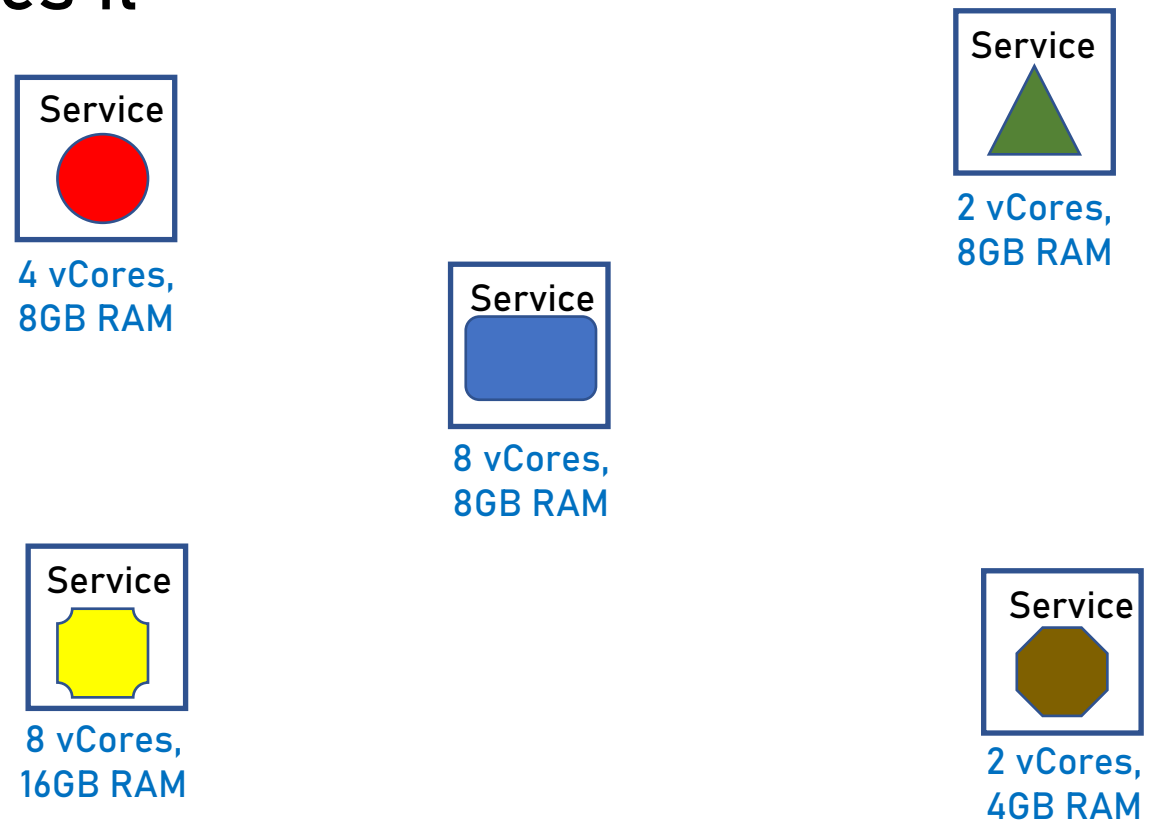


Inefficient Compute Resources

- With monolith, compute resources (CPU and RAM) are divided across all components
- If a specific component needs more resources – no way to do that
- Very inefficient

Inefficient Compute Resources

- With Microservices, the Componentization via Services attribute solves it



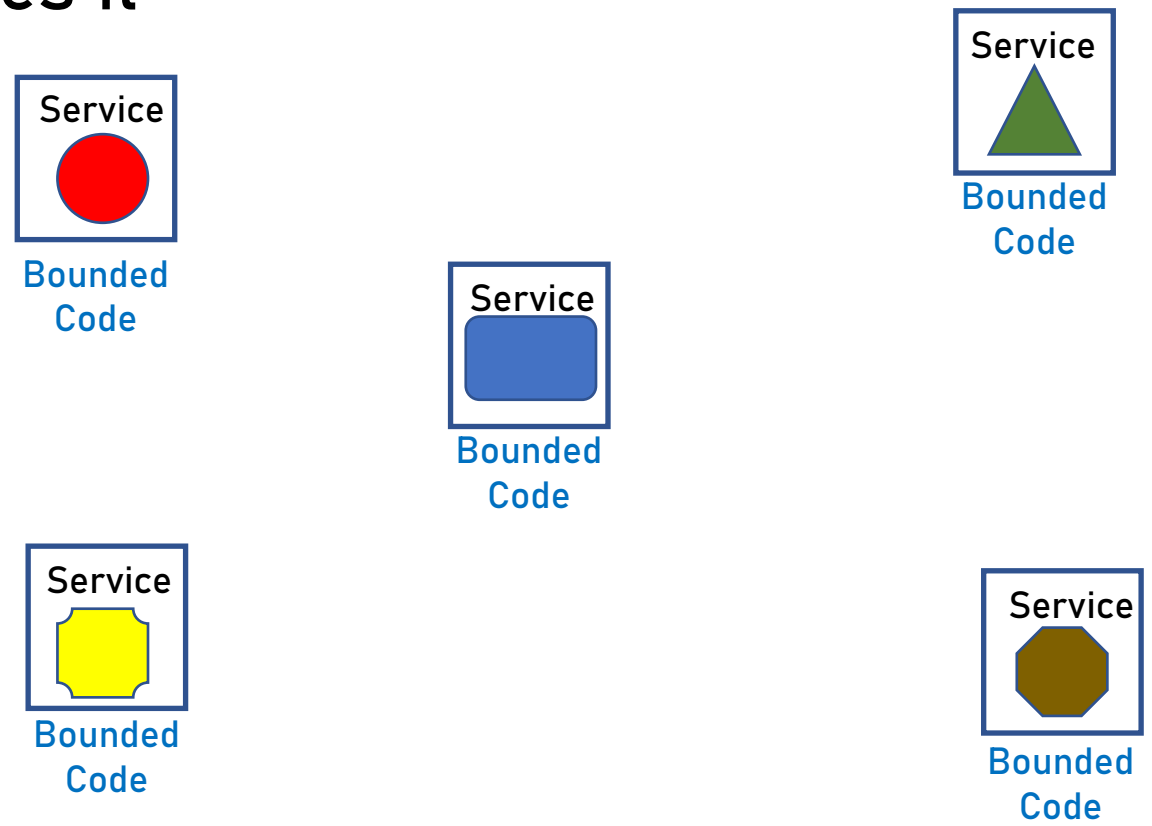
Large and Complex

- With monolith, the codebase is large and complex
- Every little change can affect other components
- Testing not always detects all the bugs
- Very difficult to maintain
- Might make the system obsolete

Large and Complex

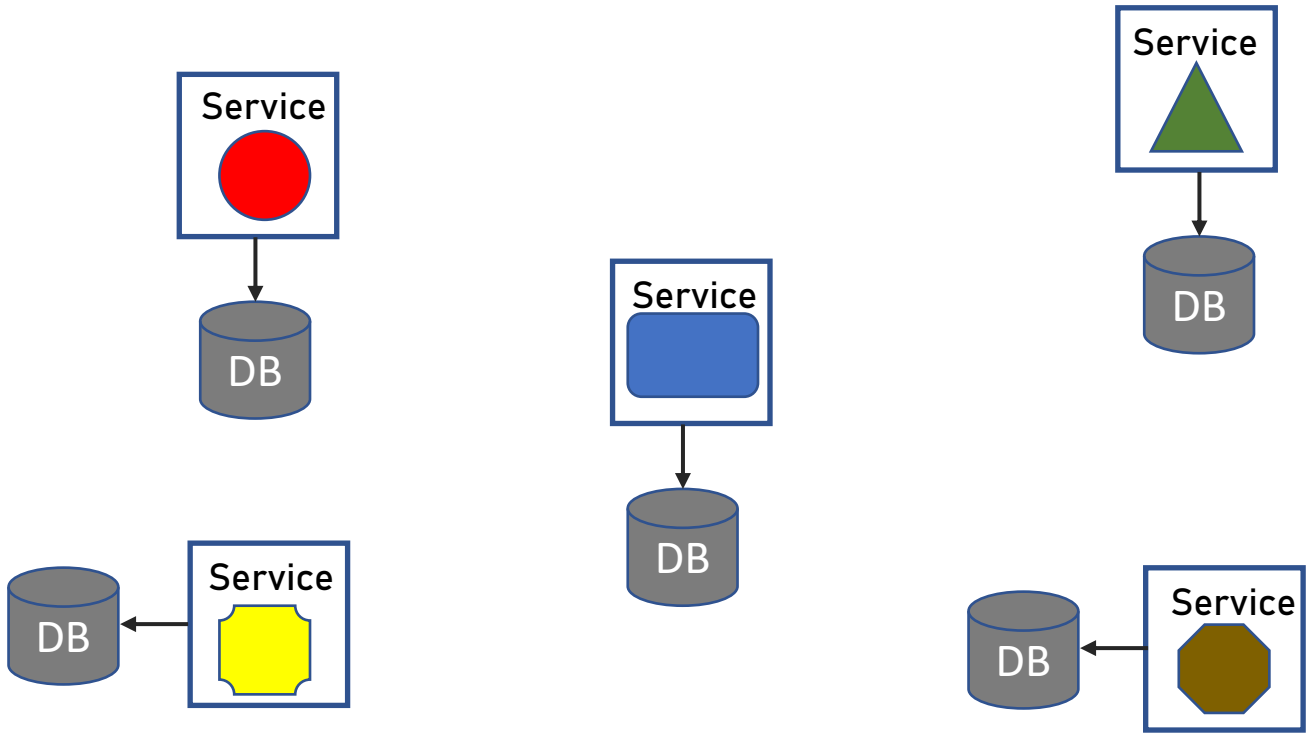
- With Microservices, the Componentization via Services

  attribute solves it



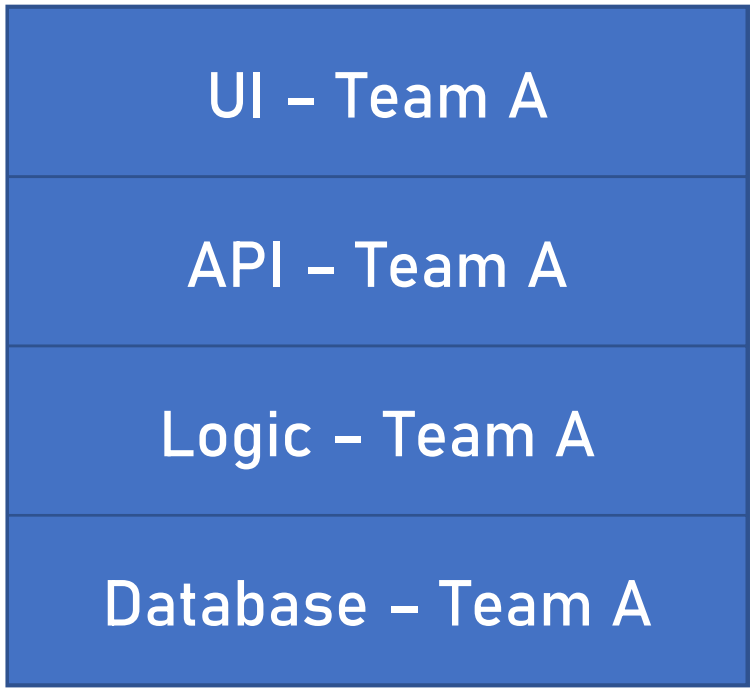
Large and Complex

- Also – Decentralized Data Management



Large and Complex

- And – Organized around business capabilities



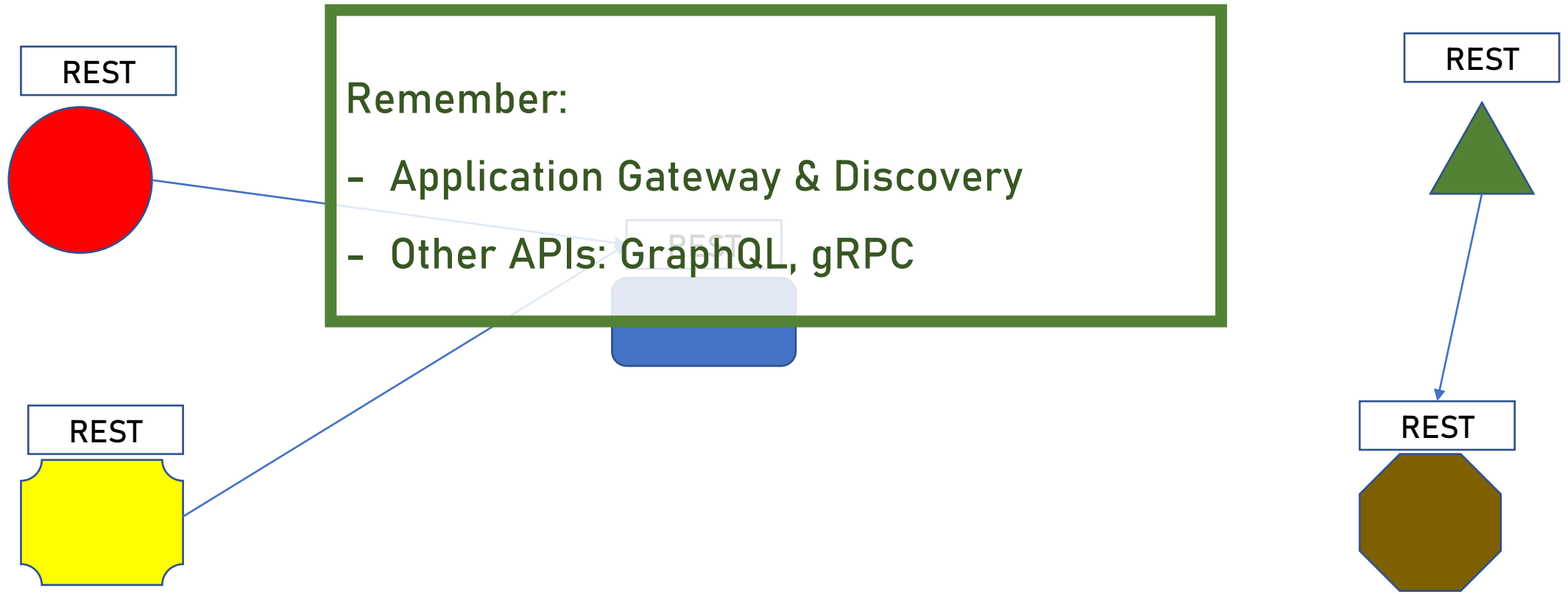
Complicated and Expensive ESB

- With SOA, the ESB is one of the main components
- Can quickly become bloated and expensive
- Tries to do everything
- Very difficult to maintain

Complicated and Expensive ESB

- With Microservices, the Smart Endpoint and Dumb Pipes

  attribute solves it



Lack of Tooling

- For SOA to be effective, short development cycles were needed
- Allow for quick testing and deployment
- No tooling existed to support this
- No time saving was achieved

Lack of Tooling

- With Microservices, the Infrastructure Automation attribute solves it
- Automates testing and deployment
- Provides short deployment cycles
- Make the architecture efficient and effective