www.dwmkerr.com

Team Size

Can you fit your team around a large table?

Yes! → You might not need microservices yet

Challenges around deployment, development, operations etc can probably be most easily handled by **good communication** and **good design** - microservices may be a solution to a problem you are not really suffering from!

No! → Microservices might help you

If your team is large, or you have many teams, you might not be able to enforce strong boundaries between components with good design alone. Enforcing boundaries by separating components into isolated services may help.

State

Are you dealing with a primarily stateless system?

Yes! → Consider serverless!

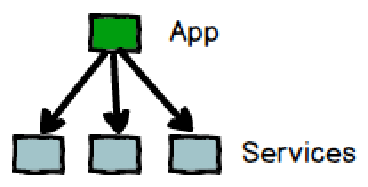
If you are mostly stateless, you may be able to skip microservices and go straight to serverless, at least for parts of your system.

No! → Microservices will come with complexity

This doesn't mean don't use microservices, but be aware that they will not be trivial to implement of manage, particularly as the system **changes over time**.

Consumers

Are you building a solution for a single app or service?



Yes! → Be careful - might have fuzzy domains!

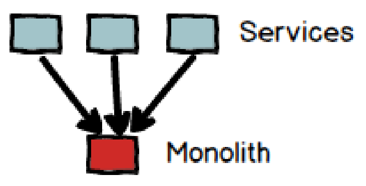
If everything you are building is for a single consumer, you may find that when you build features, you might be updating many services at once. Microsevices might be valid, but be extremely careful in how you design your domains!

No! → Microservices may be very valuable

If you are designing for many diverse consumers, microservices may be a very sensible pattern to look into, to allow you to bring new features to new consumers quickly.

Dependencies

Do you have monolithic dependencies?



Yes! → Performance might be an issue

Independently scalable services are unlikely to be a benefit in this case, as you are dependent on the performance of your dependencies. So some of the key benefits may be unacheivable. You might also have less well defined boundaries to your services

No! → Microservices may be very valuable

If you are not constrained by monoliths downstream, you might be able to achieve the high degree of independence required for effective scaling of microservices.

Expertise

Got container / orchestration / devops experts?

Yes! → Microservices may be very valuable

If you've got the chops, it might be worth looking into microservices! You have the skills to deal with the complexities which will arise and can probably capitalise on the benefits.

No! → Consider testing the water first!

If you don't have the expertise, or are already struggling with devops, this might be too much of a jump. Perhaps consider a small simple service as a spike or proof of concept. Learn the skills on projects which are not mission-critical first!