Synthesizing Failure: Simplicity in Modeling

Aaron Bedra

Disclaimer

So you need to **build** a new thing

You’re going to use a **modern** system design approach

The team comes back with a **design** that is…

Multi region

Containerized

Multi zone

k8s deployed

Eleventy billion nines!*

Multi cloud

Replicated

*Some items more expensive than they appear

Why?

Because we **don't know** how or when something will **break**

We spend **time**, **money**, and **talent** on making things unbreakable so we don’t have to understand how they break

But they still do

What if instead of building **unbreakable** things we designed **gracefully failing** systems?

What if those systems modeled **failure** as part of the **domain**?

What if the **failure** modeling included **business impact** analysis?

What if you could **reduce** infrastructure **complexity** and suffer no material **loss of revenue**?

But **how?**

Modern architecture is…

10% **innovation**

20% **learning**

20% **compromise**

50% ego

The **process**

When something fails, **encode** how to **handle** it

try/catch is **no longer** the end of your journey

Failure is an **algebraic data type**

Temporary | Semantic | Terminal

When it’s **temporary**, dead letter the action and **retry**

Queues are liberating

When it’s **semantic**, decide if there’s **enough information** to continue and continue while alerting

If you can’t, it’s **terminal**

If it’s terminal, **shut down**

Yes, **shut down**

It tells other consumers to queue up and **try again later**

Most systems can afford to be

eventually consistent

Most can afford some ambiguity in execution with simple **reconciliation** tactics

When you understand **reconciliation**, you understand what's **most important**

Failure synthesis as a design technique is powerful

When people understand **business impact**, they tend to make better choices

Next time you ask a team to build something, re-evaluate how **failure** impacts **design**

Use failure synthesis to **simplify** it

References

- aaronbedra.com/post/failure_synthesis/
- lexi-lambda.github.io/blog/2019/11/05/parse-don-t-validate/
- www.howtomeasureanything.com/
- www.amazon.com/Waltzing-Bears-Managing-Software-Projects/dp/0932633609
- www.youtube.com/watch?v=AnZ0uTOerUI