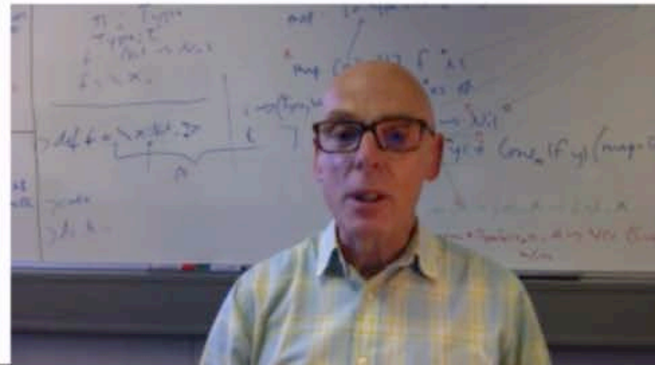

University of
Kent

Building robust systems

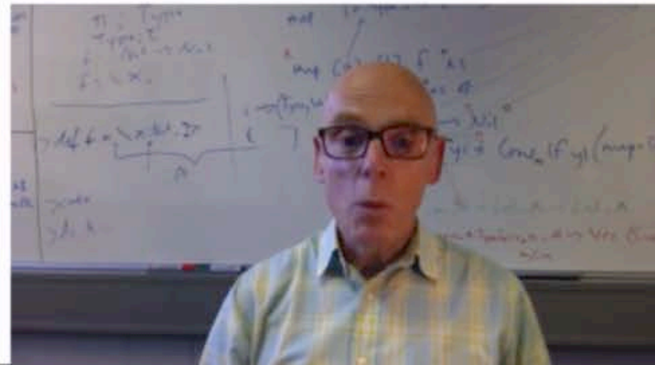


Message-passing concurrency

Different concurrent processes "share nothing" …

… and only communicate by passing messages.

No worries about shared state, thread safety, …



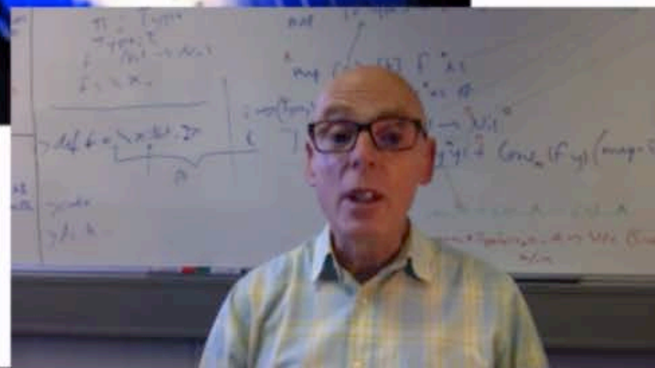
What could go wrong?

Software error

Data mismatch

Network partition

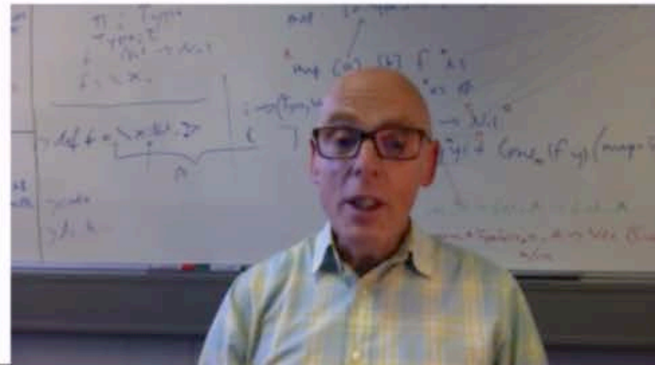
Hardware failure



Let it fail!

We can't avoid parts of a system failing, …
… so we must handle failure somehow.

Let it fail! is the key design principle …
… simplifies how to deal with all kinds of errors
… by separating concerns.



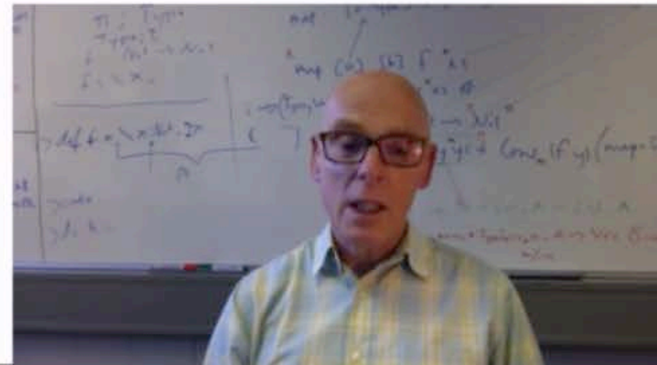
A beautiful design

Mailboxes and message handling

Process errors and trapping exits

OTP generics

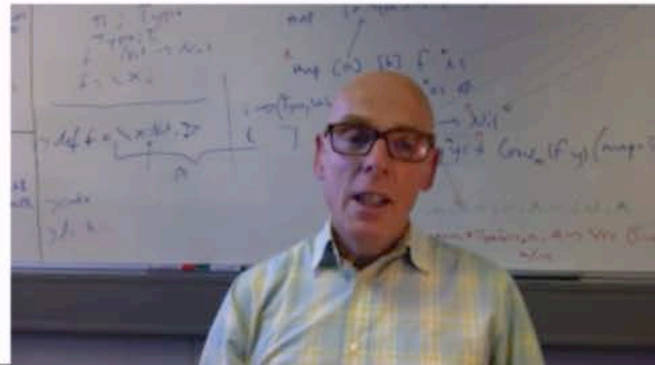
Concurrency and distribution



Exceptionally ...

Erlang does have an exception handling mechanism too.

We'll cover this briefly … it's nothing exceptional :-)



Fixing problems

Can't shut down just to install a software upgrade.

Erlang allows hot code loading into a live system.