Akamai

FASTER FORWARD

Safety Thinking in Cloud Software: Challenges and Opportunities

Kevin Riggle, Senior Lead Security Researcher

Who is Akamai?

- World’s largest Content Distribution Network (CDN)
- ~25% of all traffic on the Web
- 200k servers in 110 countries
- Servers on all seven continents

Who am I?

- Joined Akamai in 2012
- Information Security department (Infosec)
- Manager
- Adversarial Resilience team
  - Infosec's part of our product review process

- kriggle@akamai.com
- kevinr@free-dissociation.com
- Twitter: @kevinriggle
- #stamp2016

Specific Examples but General Principles

- Feedback if you’ve encountered these problems
- Sharing and collaboration if you’re currently having these problems
- Opportunities for academic work

- What is cloud software?
- Systemic Pressures
- Fundamental Problems & Gotchas
- Opportunities

Cloud Software

Amazon

Facebook

Google

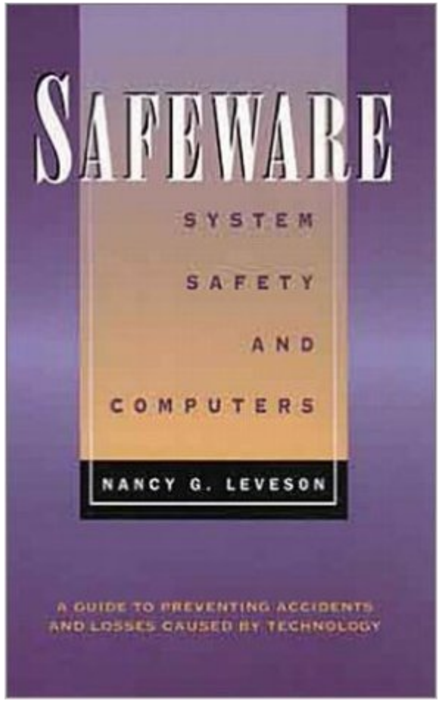
Microsoft

Akamai

Internet Software Services

(Software as a Service/SaaS)

not Embedded Software

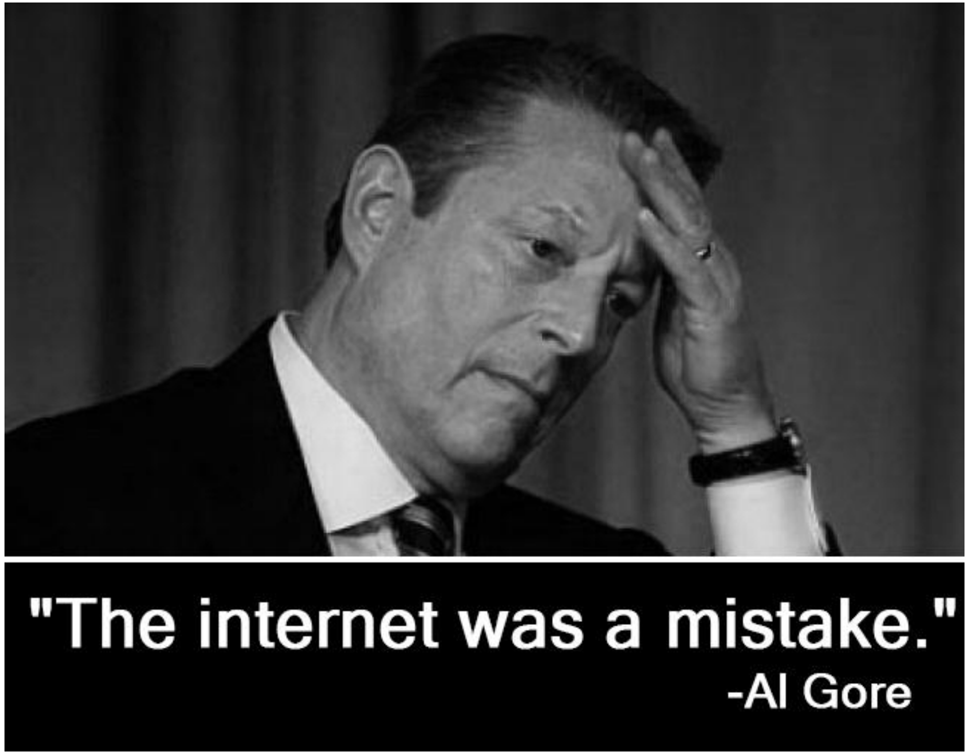

SAFEWARE
SYSTEM SAFETY AND COMPUTERS
NANCY G. LEVESON
A GUIDE TO PREVENTING ACCIDENTS AND LOSSES CAUSED BY TECHNOLOGY

not Retail Software

Internet-scale, heterogeneous distributed systems

“A distributed system is one in which the failure of a computer you didn’t even know existed can render your computer unusable.”

--Leslie Lamport


"The internet was a mistake."
-Al Gore

Systemic Pressures

Software is lean

Software is fast

Akamai’s accident review team:

4 people

50 accidents reviewed/year

Akamai’s design safety team:

11 people

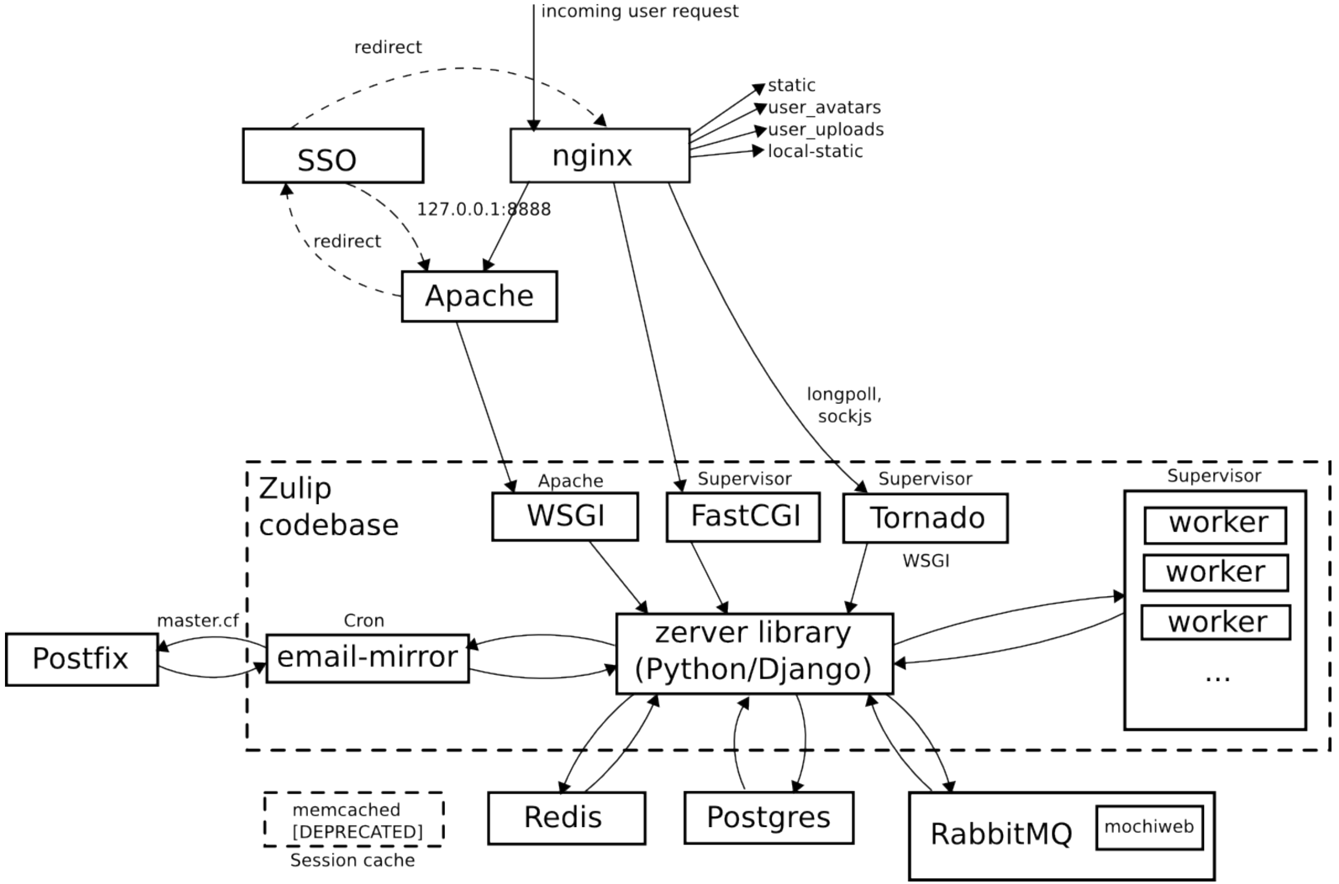
Platform: 50 reviews/year

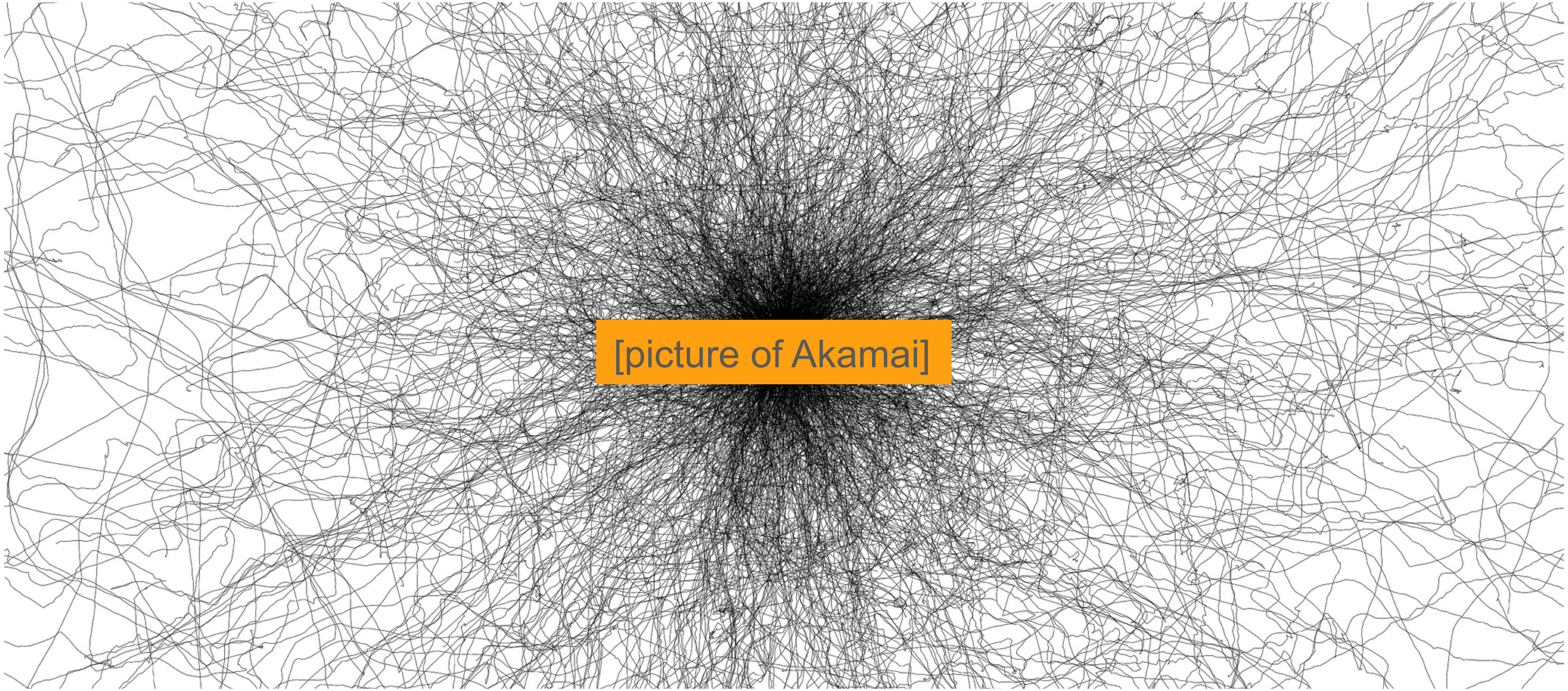
2 business days, 30 pages

Products: 50 meetings w/ 130 agenda items

Weeks (usually)

Software is complicated


incoming user request
redirect
static
user_avatars
user_uploads
local-static
SSO
nginx
127.0.0.1:8888
redirect
Apache
longpoll,
sockjs
Zulip
codebase
Apache
WSGI
Supervisor
FastCGI
Supervisor
Tornado
WSGI
Supervisor
worker
worker
worker
...
master.cf
Cron
Postfix
email-mirror
zerver library
(Python/Django)
memcached
[DEPRECATED]
Session cache
Redis
Postgres
RabbitMQ
mochiweb

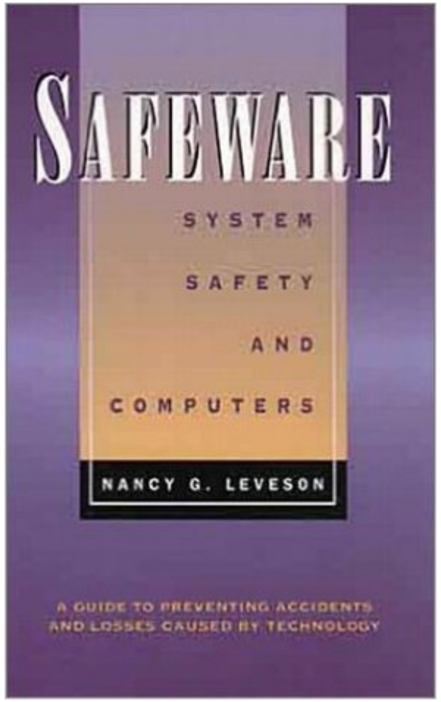


https://flic.kr/p/dcEBQ5

Few constraints on how software is composed

With a few notable exceptions,
software has never had a safety culture

(Especially outside embedded)


SAFEWARE
SYSTEM
SAFETY
AND
COMPUTERS
NANCY G. LEVESON
A GUIDE TO PREVENTING ACCIDENTS
AND LOSSES CAUSED BY TECHNOLOGY

Kitty Hawk: 1902

NTSB founded: 1926

Hacker mentality

Still people in the industry who reject testing

Still people in the industry who reject documentation

Still people in the industry who reject planning

New programs need to be lightweight

New programs need to demonstrate clear value that we can’t get any other way

I don’t need to persuade the hackers

I do need to persuade the people who are bought in already who have to work with and for the hackers

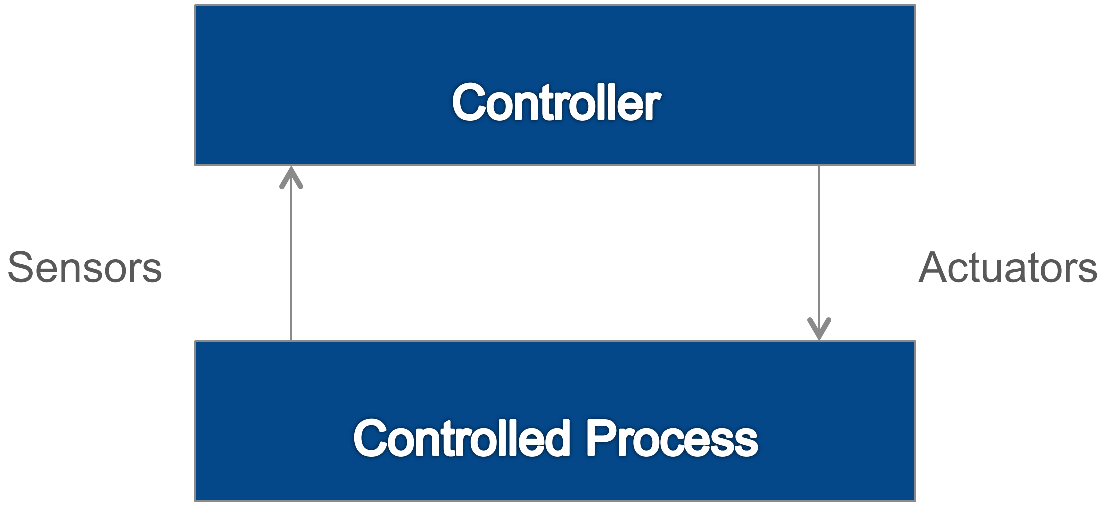
Management wants the benefit…

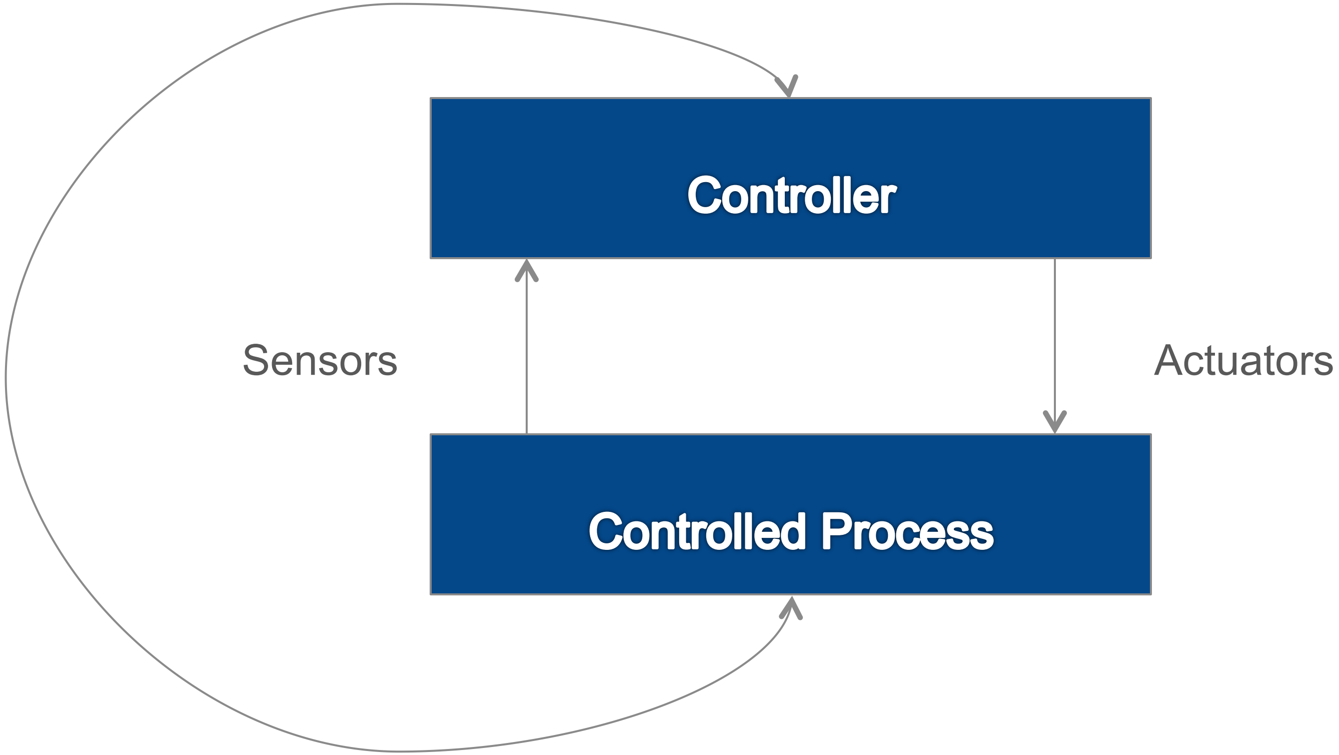
…but doesn't want to pay.

80% of the benefit, 20% of the cost

Fundamental Problems & Gotchas

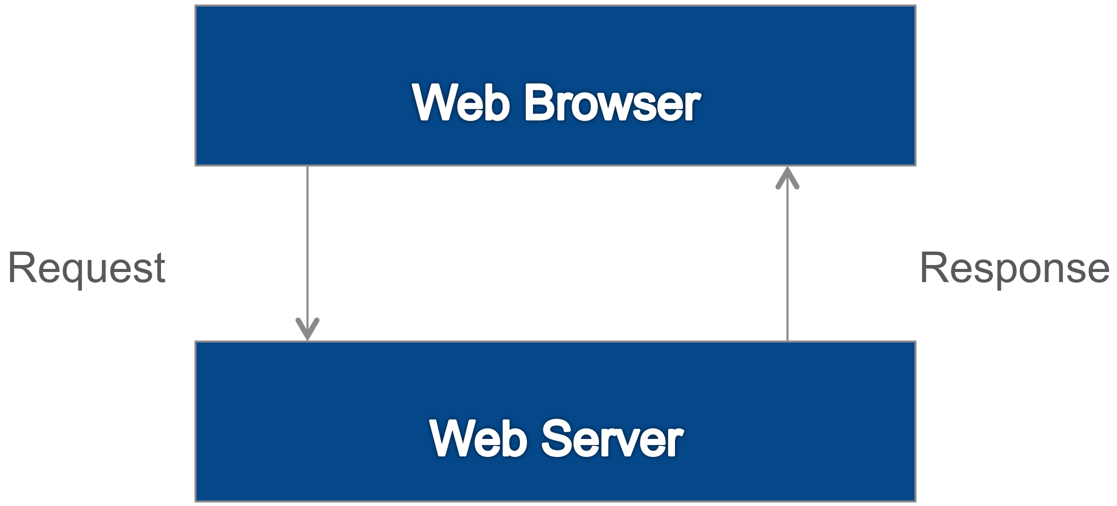
Weird Loops

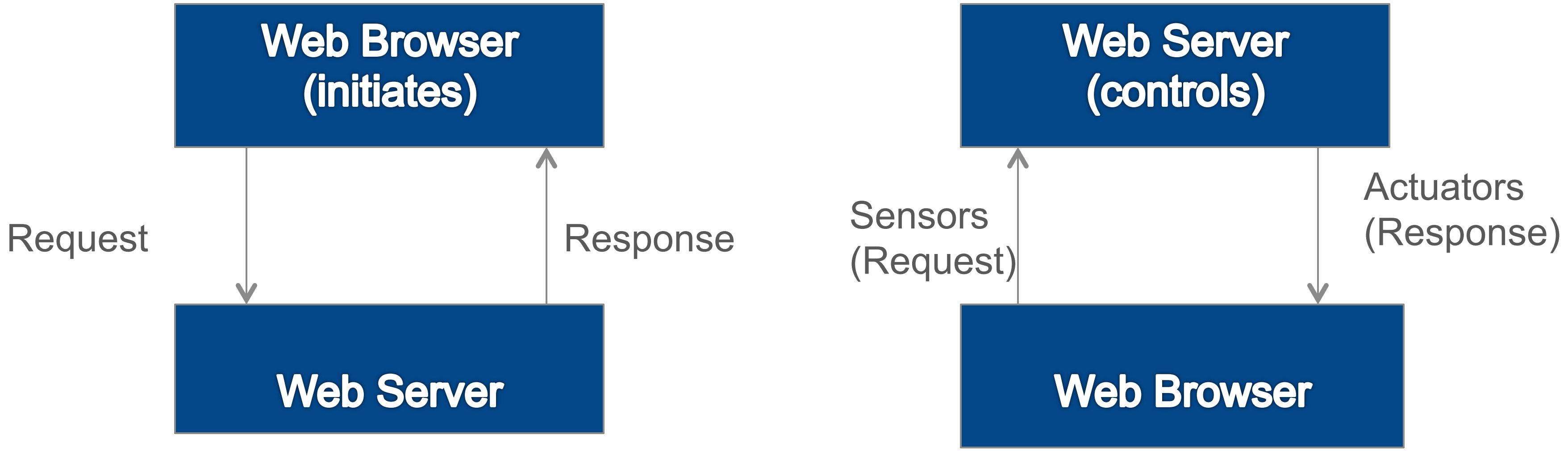

Controller
Sensors
Actuators
Controlled Process


Controller
Sensors
Actuators
Controlled Process

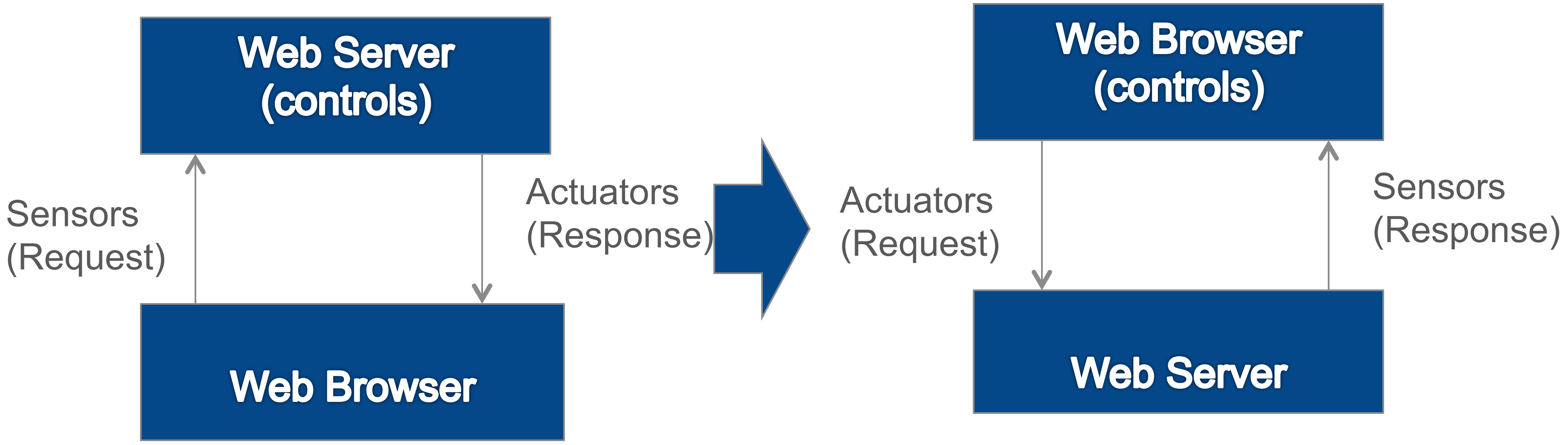
Physical Decomposition
doesn’t match Control Decomposition

Functional/Process Decomposition
doesn’t match Control Decomposition

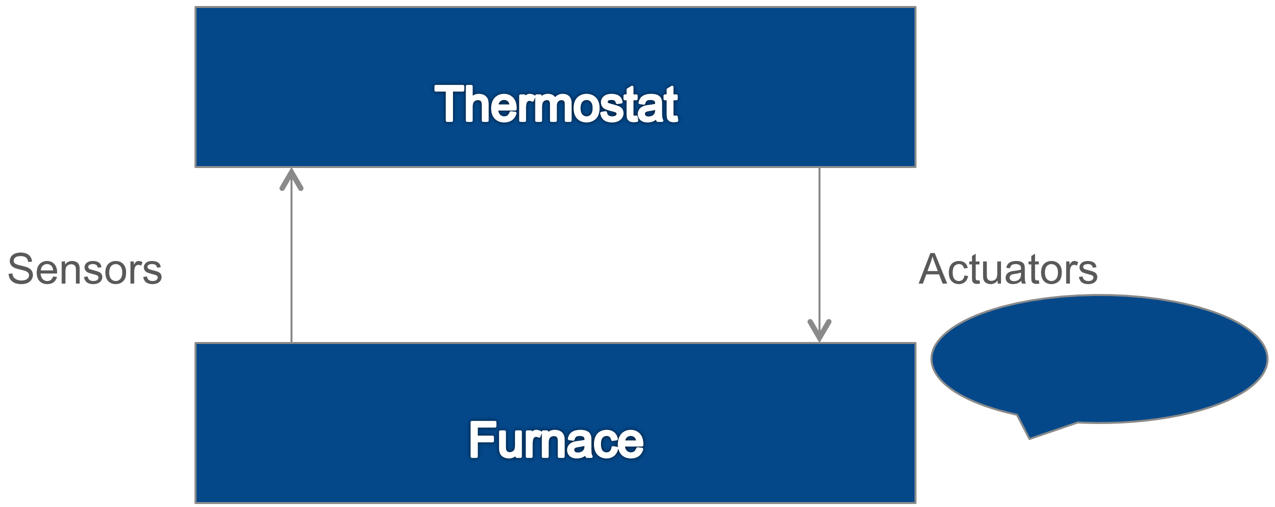

Web Browser
Request
Response
Web Server


Web Browser (initiates)
Request
Response
Web Server
Web Server (controls)
Sensors (Request)
Actuators (Response)
Web Browser

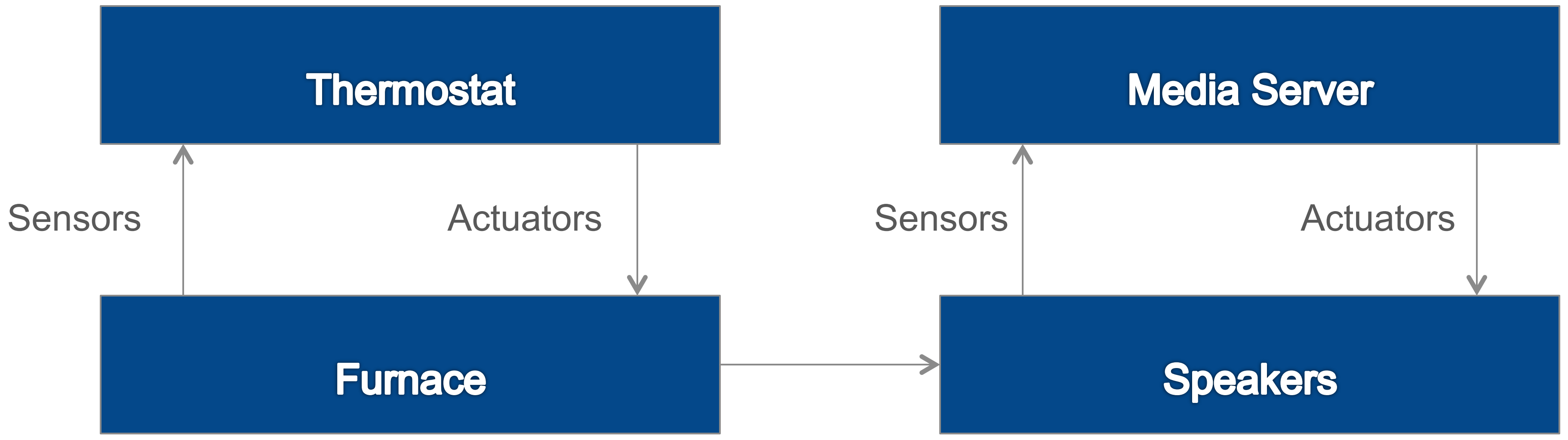
Surprise Inversion of Control


Web Server (controls)
Sensors (Request)
Actuators (Response)
Web Browser
Web Browser (controls)
Actuators (Request)
Sensors (Response)
Web Server

Rich Control Languages


Thermostat
Sensors
Actuators
Furnace

Dynamic Reconfigurability


Thermostat
Media Server
Sensors
Actuators
Sensors
Actuators
Furnace
Speakers

???????

Surprise Your System is Turing Complete

time+96
64 64 64 64 64 64 64 64 64

All is not lost

Security moment

STECA

Nix

Chaos Monkey

Reproducible builds

Non-Turing complete control languages

Software sucks…

…but I’m convinced that this shows us the way to build ***safe systems from sucky software***…

…if we can figure out how to make it practical to apply.

- Feedback if you’ve encountered these problems
- Sharing and collaboration if you’re currently having these problems
- Opportunities for academic work

80% of the benefit, 20% of the cost

Questions?

Thank You

Twitter: @kevinriggle
#stamp2016