# *DevOps & Continuous Delivery in the Enterprise*

Eberhard Wolff

Freelancer

Head Technology Advisory Board adesso AG

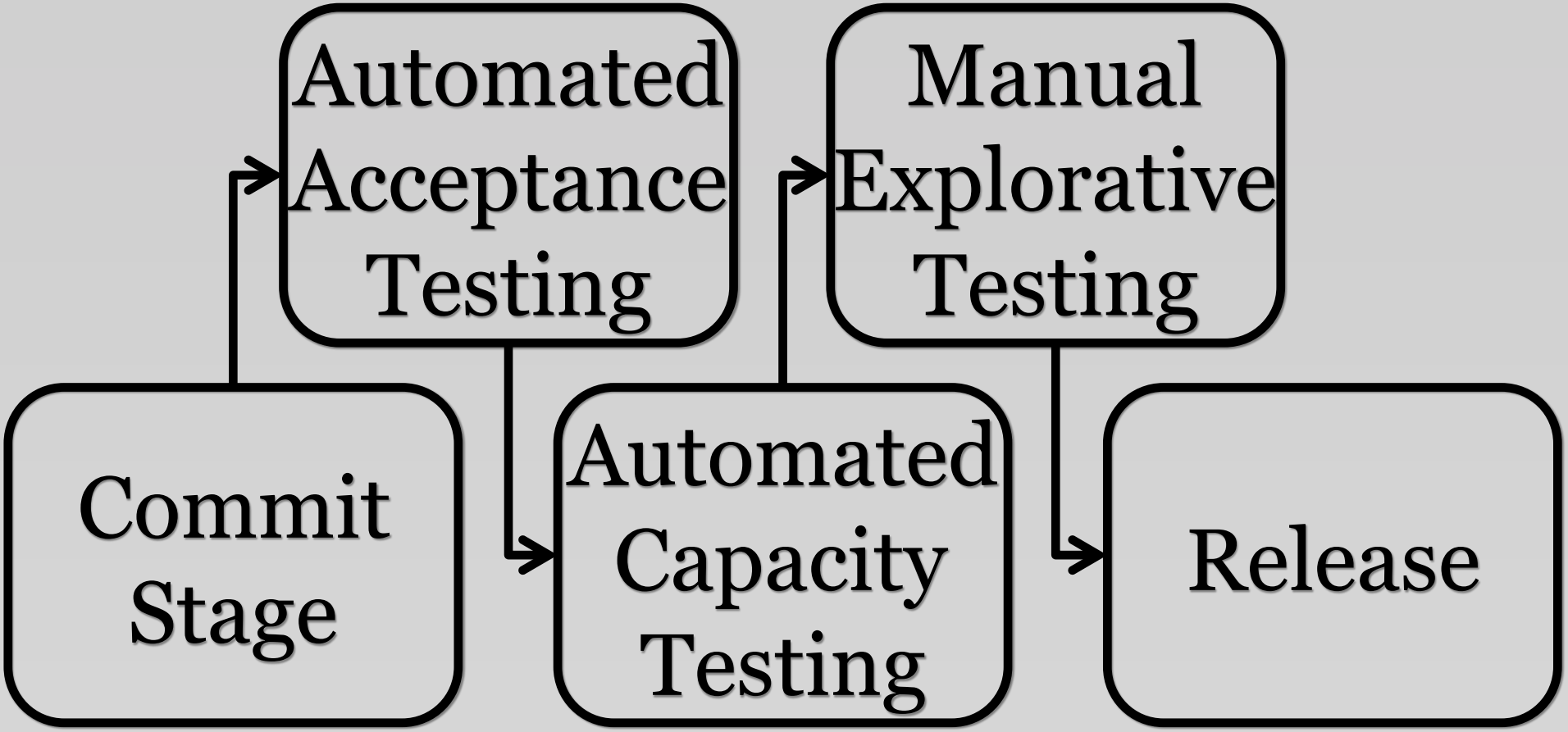
Leseprobe:
http://bit.ly/CD-Buch

# Continuous Delivery DevOps

# Continuous Delivery

DevOps

# *Continuous Delivery: Build Pipeline*



# *Continuous Delivery*

- Pipeline executed several times per day
- Automated provisioning
- Fast Feedback
- Tests, tests, tests…
- Feedback from production

# *Continuous Delivery in Startups*

Eberhard Wolff - @ewolff

# *Lean Startup*

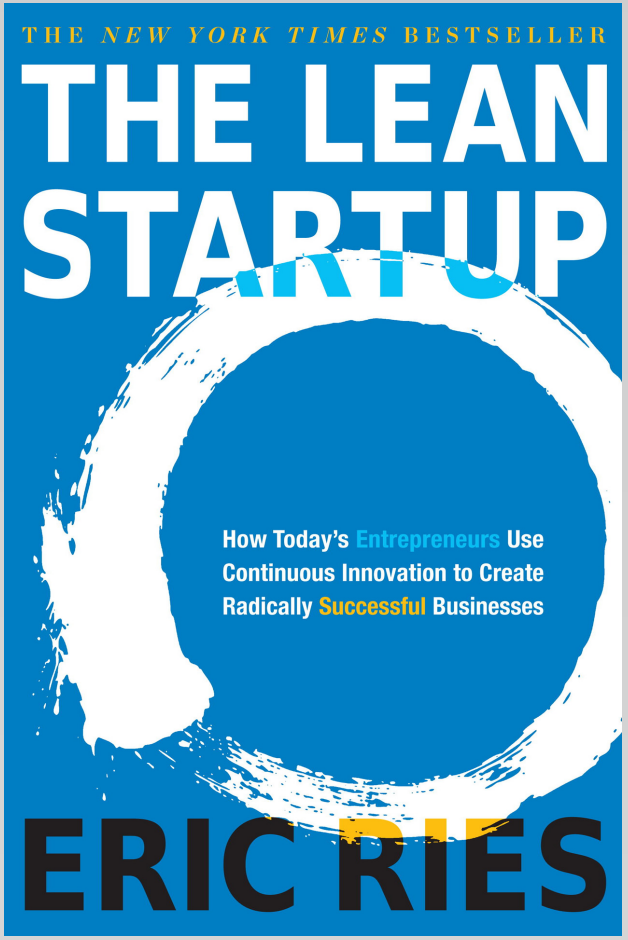
Approach for launching businesses and products

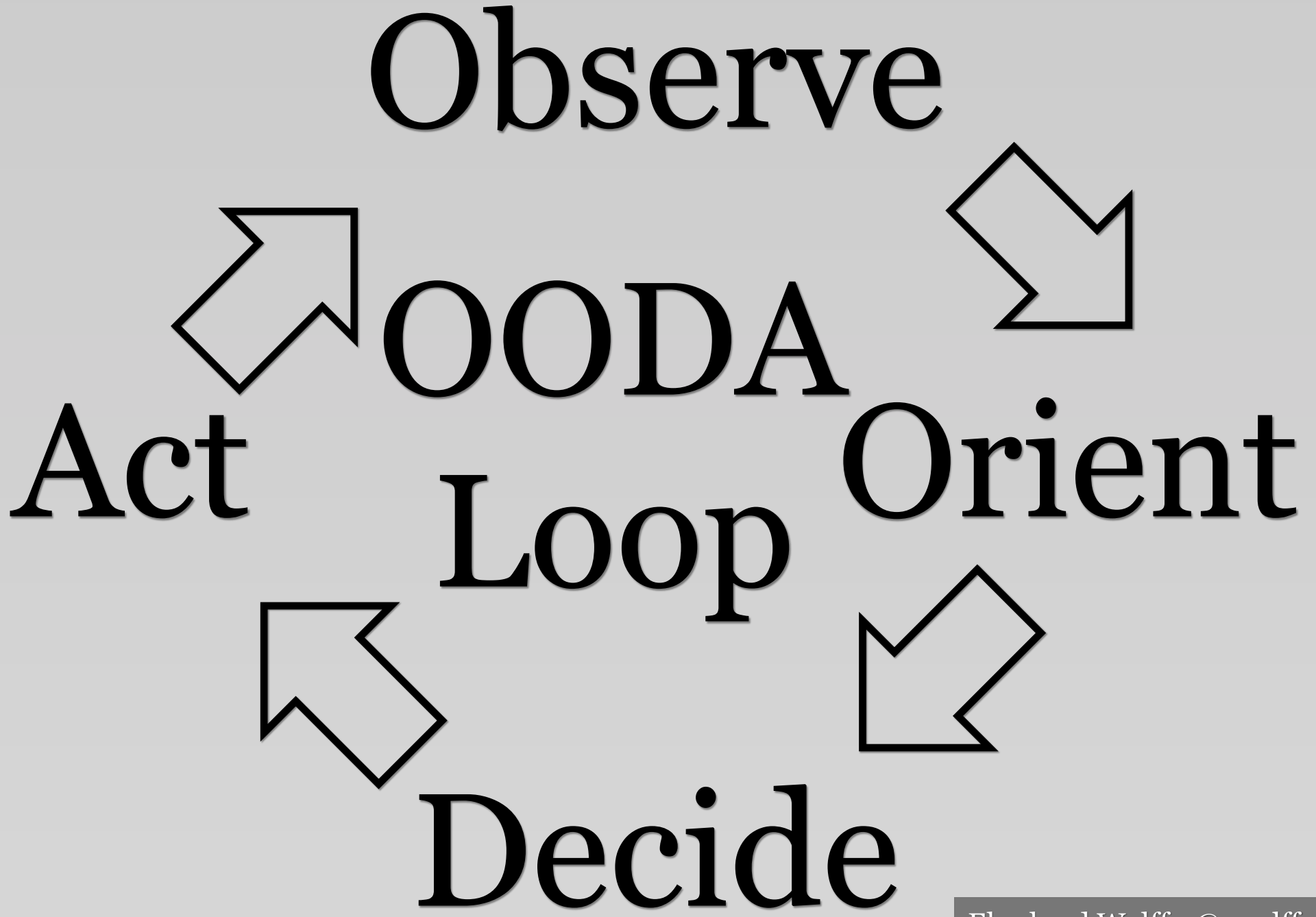
Relies on

- validated learning
- scientific experimentation
- iterative product releases

To

- shorten product development cycles
- measure progress
- gain valuable customer feedback.




Observe
OODA
Loop
Orient
Act
Decide

# *Startups*

- No clear market yet
- Need feedback
- …and change plan accordingly
- Success depends on speed in OODA loop

# *Continuous Delivery and the Market*

- OOAD: act quickly, observer better

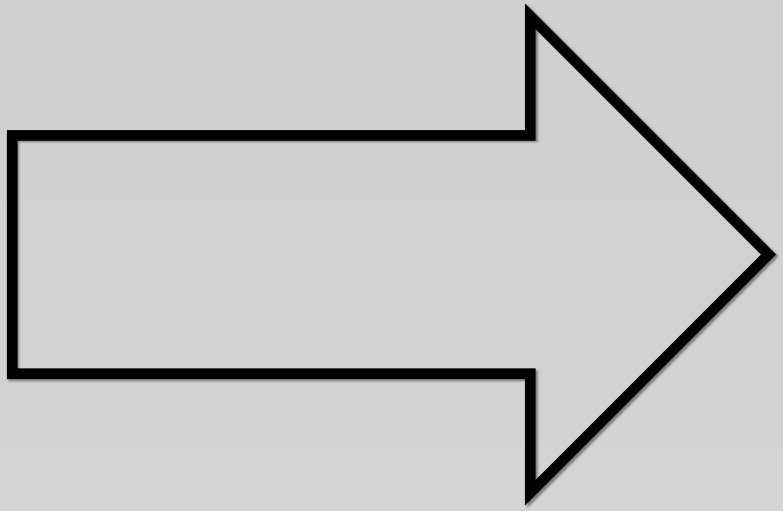
- Act: Faster deployment
- = shorter time to market

- Observe: Better visibility with better monitoring

# *Continuous Delivery in Enterprises*

Eberhard Wolff - @ewolff

Faster!



IT

Time to Market!

*We used to do a release a quarter.*

*We used to do a release a quarter. Now we do one every two quarters*

Faster!

IT

Time to Market!

# *Why not?*

- Managers think in month
- …or even years

- Quick feedback and fast deployment hardly usable

# *So?*

- Adopt Lean Startup in Enterprises
- Lean Startup is not just for Startups
- Quicker innovation
- Gain a competitive edge

- Business strategy
- Outside IT

# *However…*

- Software becomes the product
- Example: Automotive
- Cars: Auto pilot (Tesla, Audi)
- Car sharing instead of buying cars
- DriveNow (BMW), Car2Go (Daimler)
- Uber
- MyTaxi (Daimler)
- Enterprise = Start Up?

*Continuous Delivery is faster deployments*

*Enterprises can deploy quickly!*

# *Quick Deployment in Enterprises*

Severe production problem

- Hot fix
- Deployed in minutes / hours

How?

- Only limited tests, high risk (?)
- Small batch, small risk

# *Lessons Learned from Hot Fixes*

- Speed of deployment: no good reason for Continuous Delivery
- Continuous Delivery = Hot Fix + automated tests

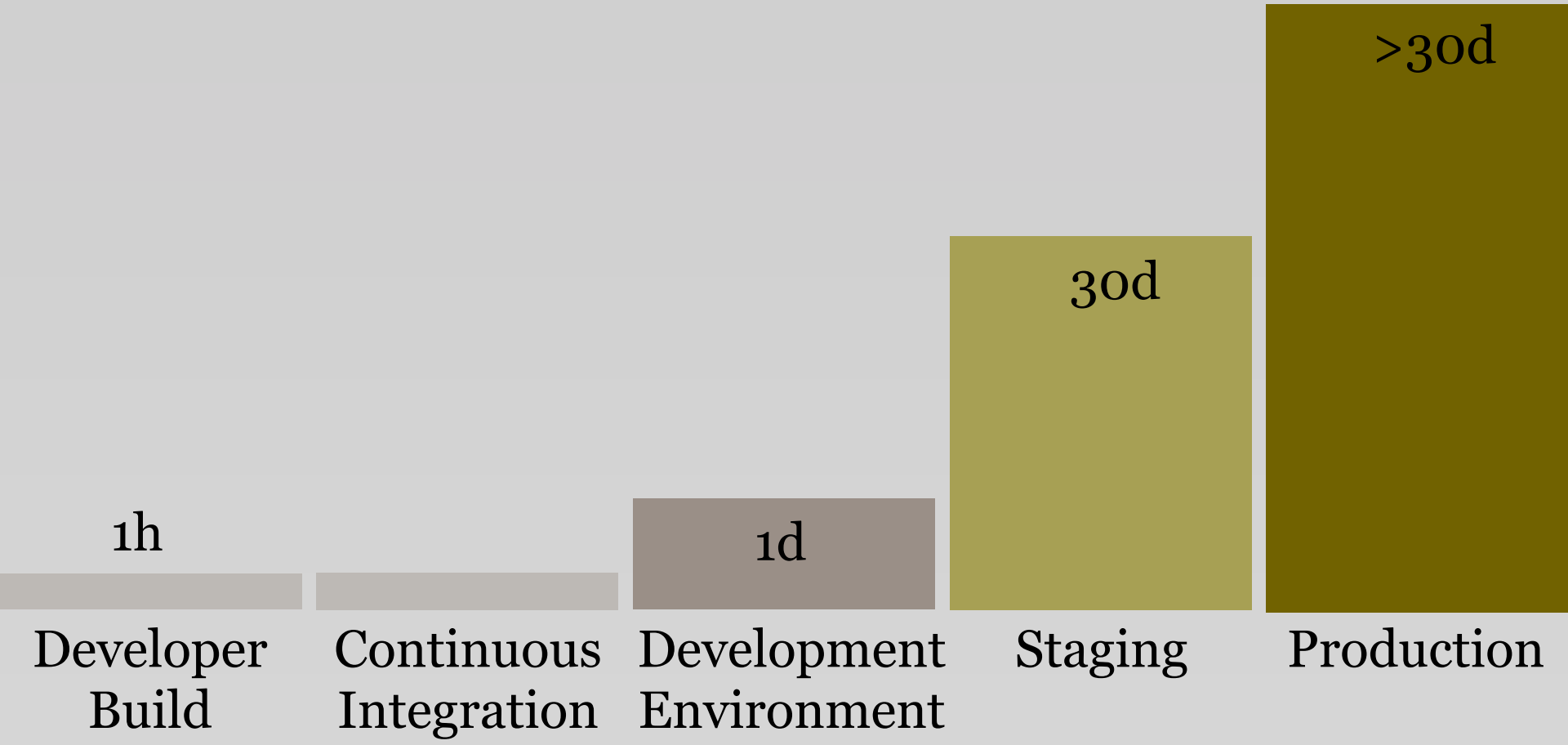
- Different way to handle risk
- Smaller and faster deployments

# *Continuous Delivery Minimizes Risk*

- Risk depends on size of change

- More frequent deployments
  less risk

# *Traditional Deployments: Risk*



# *Continuous Delivery: Risk*

# *Traditional Enterprise Approach*

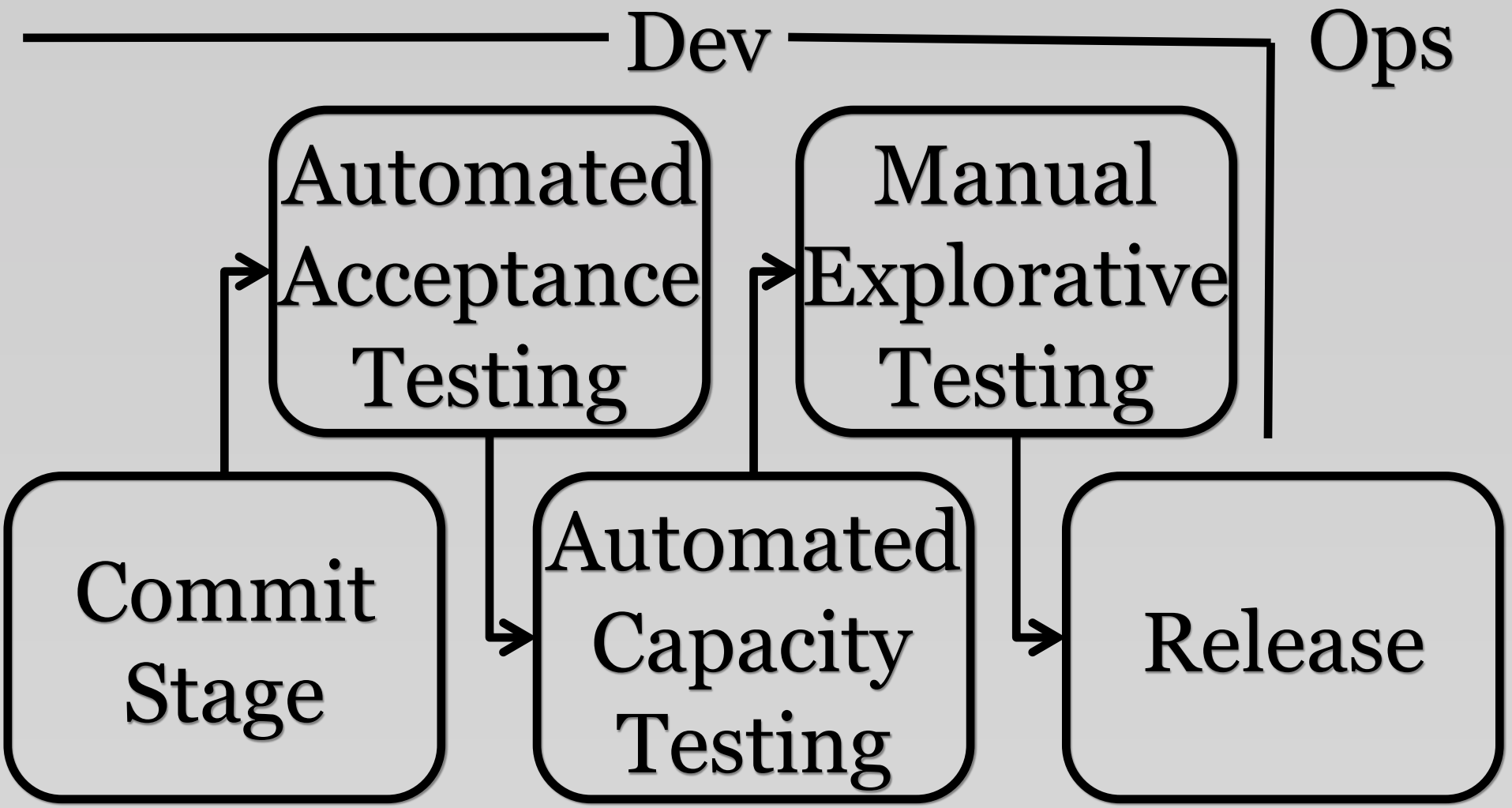
- Low frequency of releases
- Each release scheduled
- …extensively managed
- …precautions if it fails

# *Traditional Enterprise Approach*

Difference to Continuous Delivery?

- No automation
- Not reproducible
- Too infrequent to establish best practices

# *Full Continuous Delivery Pipeline?*



# *Continuous Delivery & Separated Ops*

- Need buy-in from Ops
- Separate organizational unit
- Mistrust
- Used to their ways
- …and tools

- Solution: DevOps (later)

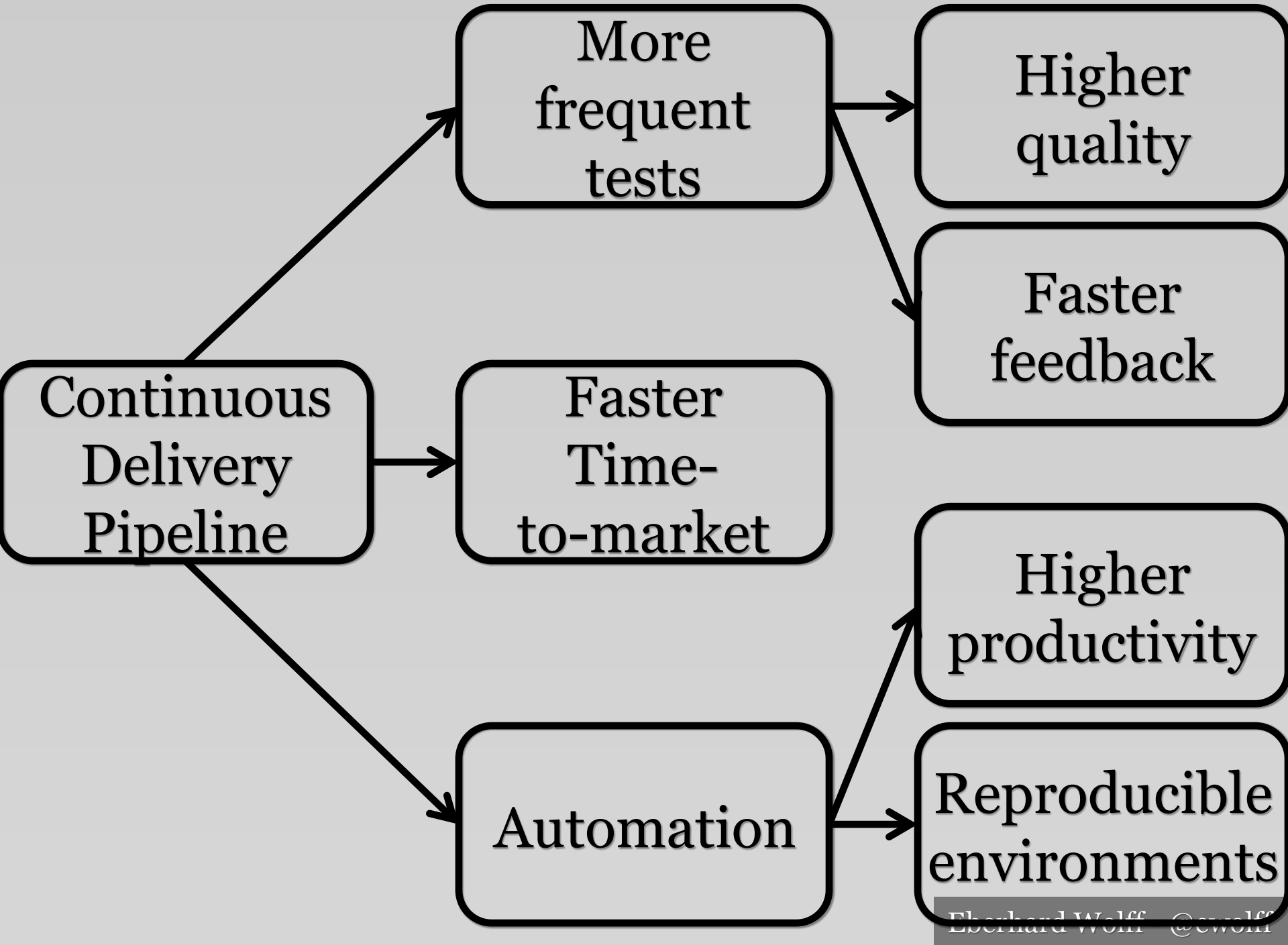
# *Dev-only pipeline useful?*

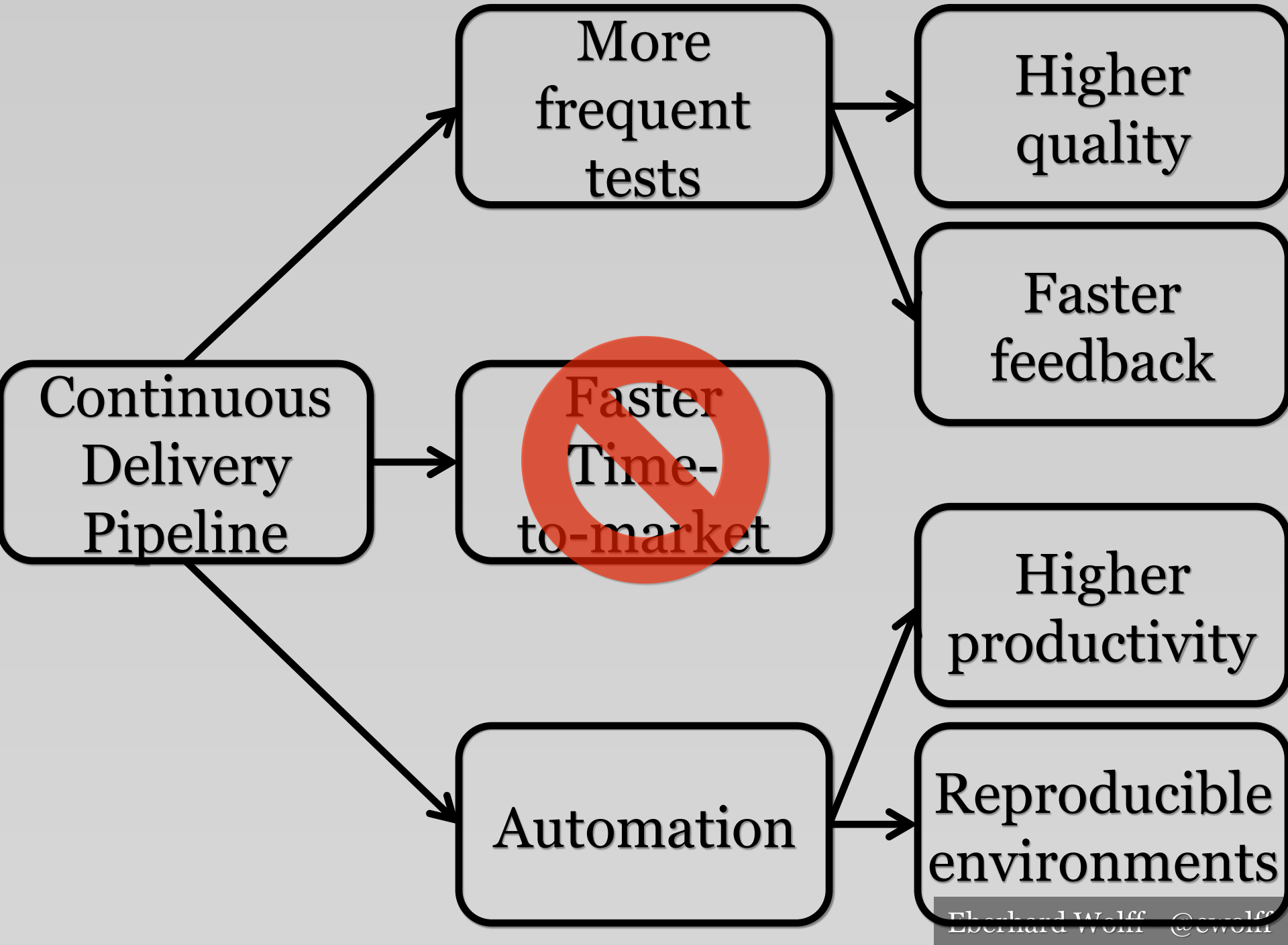
# *Dev-only Pipeline*

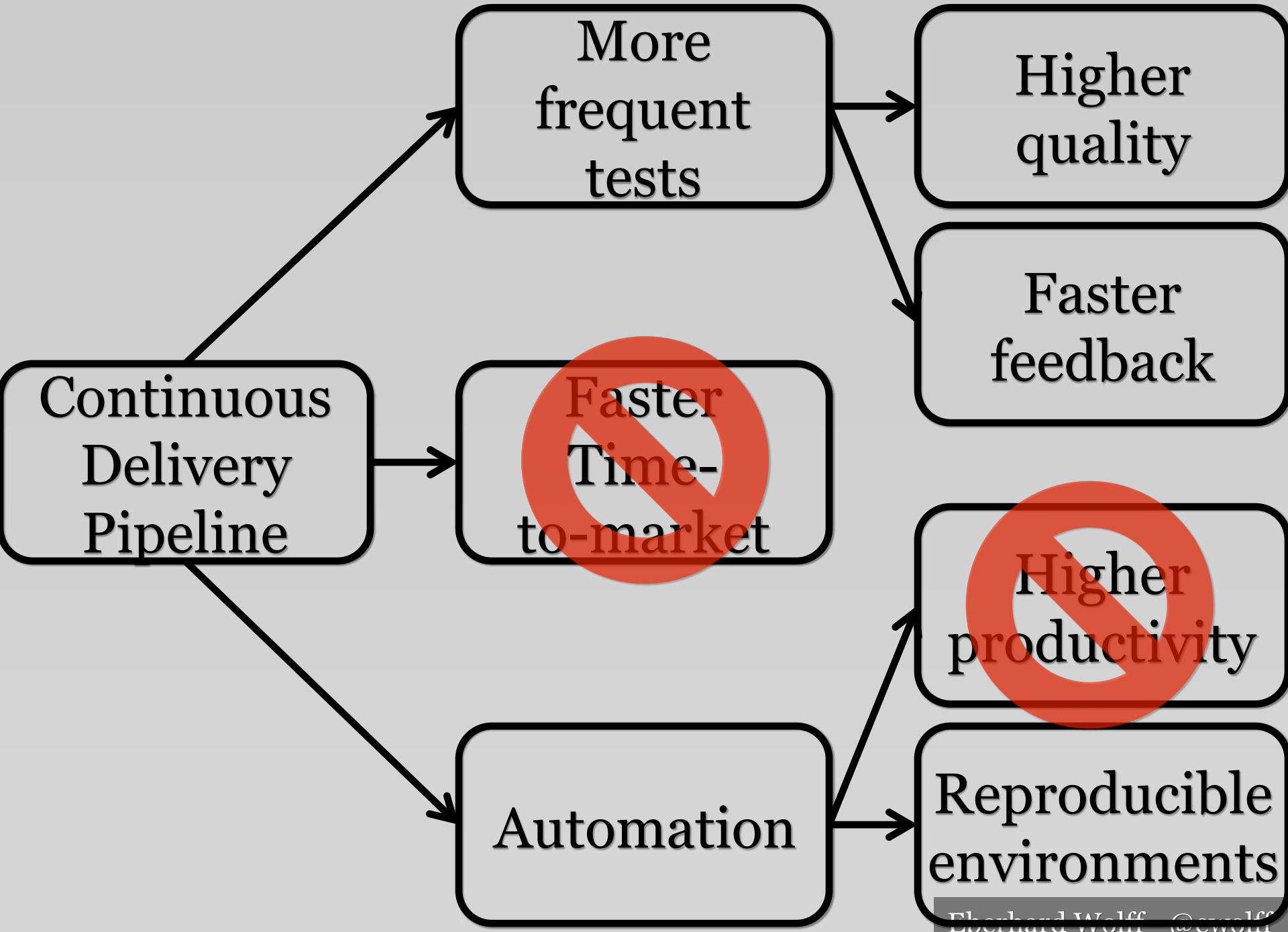
- Much higher quality
- A lot more tests
- Automated infrastructure
- Pays off for testing systems

- Sensible alternative

*Not just productivity gain automation vs. manual*


More frequent tests
Higher quality
Faster feedback
Continuous Delivery Pipeline
Faster Time-to-market
Higher productivity
Automation
Reproducible environments


More frequent tests
Higher quality
Faster feedback
Continuous Delivery Pipeline
Faster Time-to-market
Higher productivity
Automation
Reproducible environments


More frequent tests
Higher quality
Faster feedback
Continuous Delivery Pipeline
Faster Time-to-market
Higher productivity
Automation
Reproducible environments

# *Continuous Delivery in Enterprises: Sum Up*

Dev+Ops+Business

- Time to market
- Lean Startup
- Best Pay Off

Dev + Ops

- Avoid deployment problems
- All the weekends spend on releases

# *Continuous Delivery in Enterprises: Sum Up*

Dev

- Faster feedback
- More tests
- Easier to set up test environments

Can be introduced with little Buy-In

- Just do it!

# Continuous Delivery

DevOps

# Continuous Delivery

# DevOps

# *DevOps*

- Dev: Development
- Ops: Operations

- DevOps = close collaboration between Dev and Ops
- Teams have Ops and Dev people

# *DevOps in Startups*

Eberhard Wolff - @ewolff

# *DevOps in Startups*

- You build it – you run it
- Smaller organizational units
- At the beginning: No clear roles
- DevOps teams by feature
- From the start

# *DevOps in Enterprises*

Eberhard Wolff - @ewolff

*Dev – features*

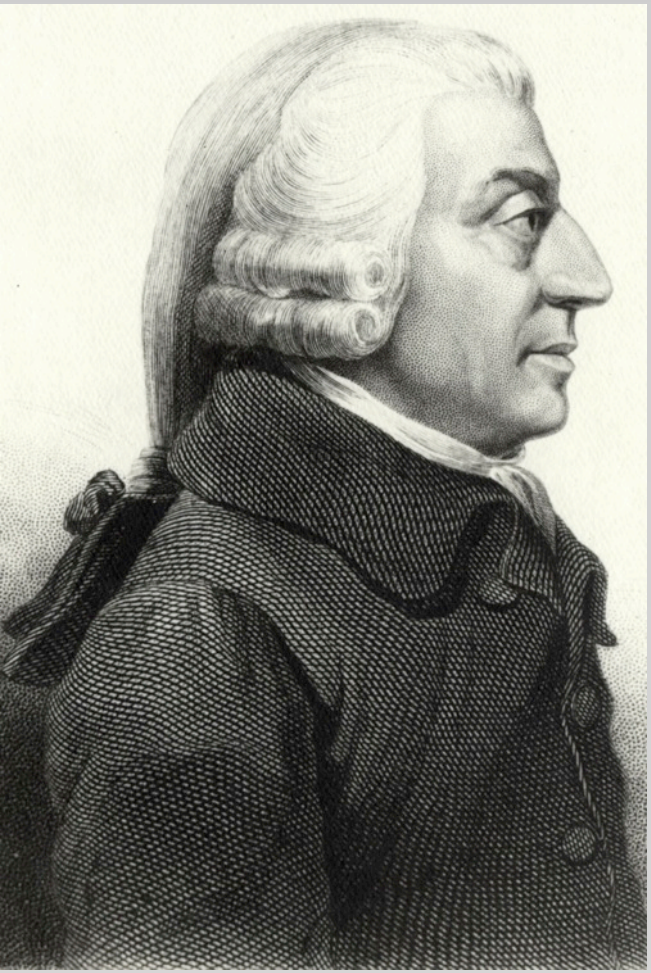
*Ops – cost*

*Dev builds it*
*Ops runs it*

*Why??*

# *Costs*

# *Why Separate Dev and Ops?*



- Adam Smith
- Separation of labor
- Standardization
- Industrialization
- Cost
- Productivity

# *Lower Costs Differently*

- Basic tasks automated by software
- See Continuous Delivery

- IT = automating tasks

# *Goals Might Be Different*

- Flexible IT to support business
- See Lean Startup etc

- Separate Dev/Ops: not be the best option for Enterprises?

# *Let's introduce DevOps in Enterprises!*

# *DevOps in Enterprises*

- Dev and Ops separated directly below CIO
- Need to change the organization
- Fundamentally
- Changing the organization is hard

*DevOps*

=

*Culture*

*Do you need change the organization?*

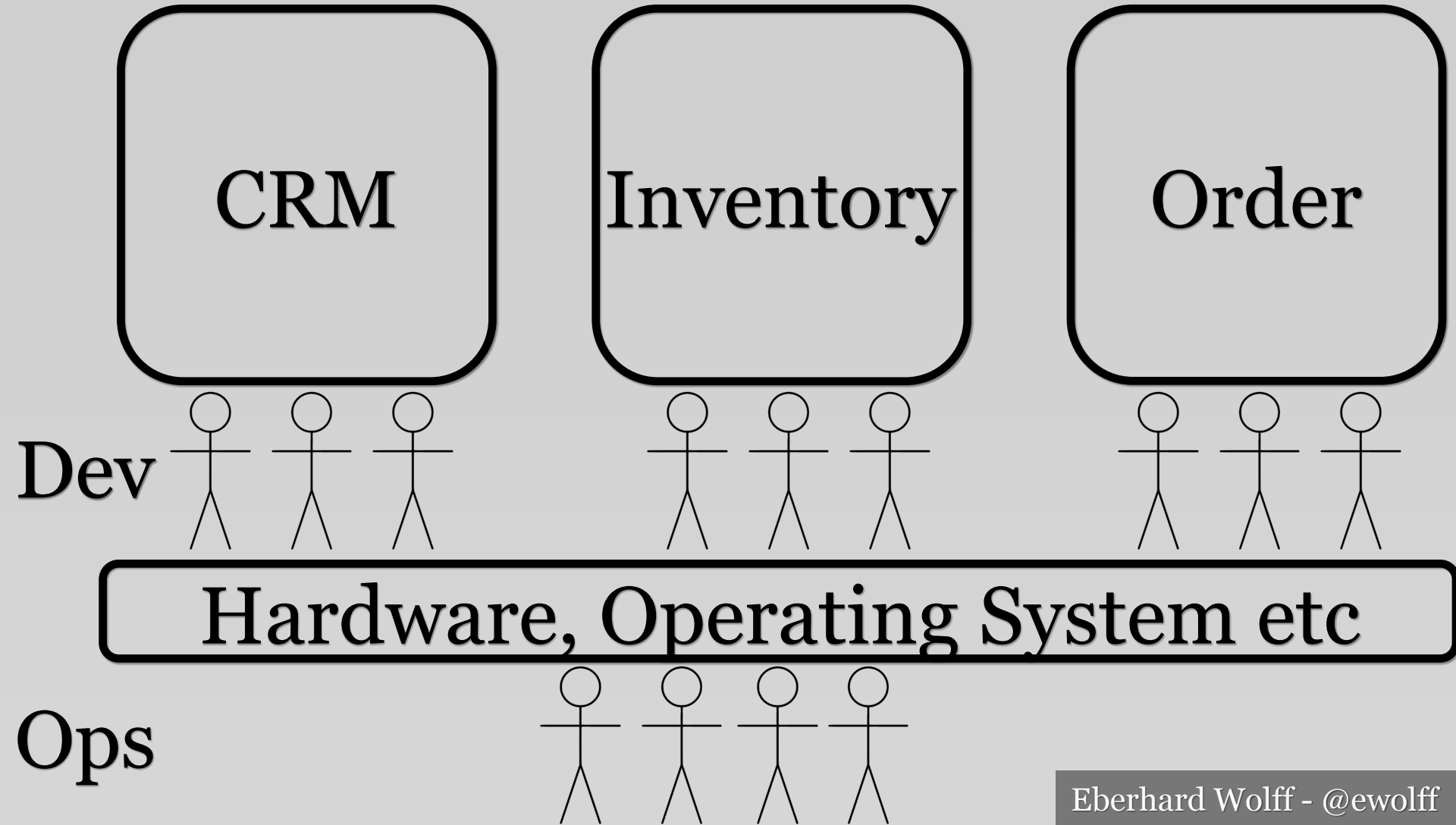
# *DevOps = Culture*

- Let Ops and Dev sit together
- Direct communication between Ops and Dev engineers emerges
- Associate Ops with Dev teams

- Share tools…
- …and knowledge

*What if you change the organization?*

# *Traditional Organization*



# *Problem with Traditional Organization*

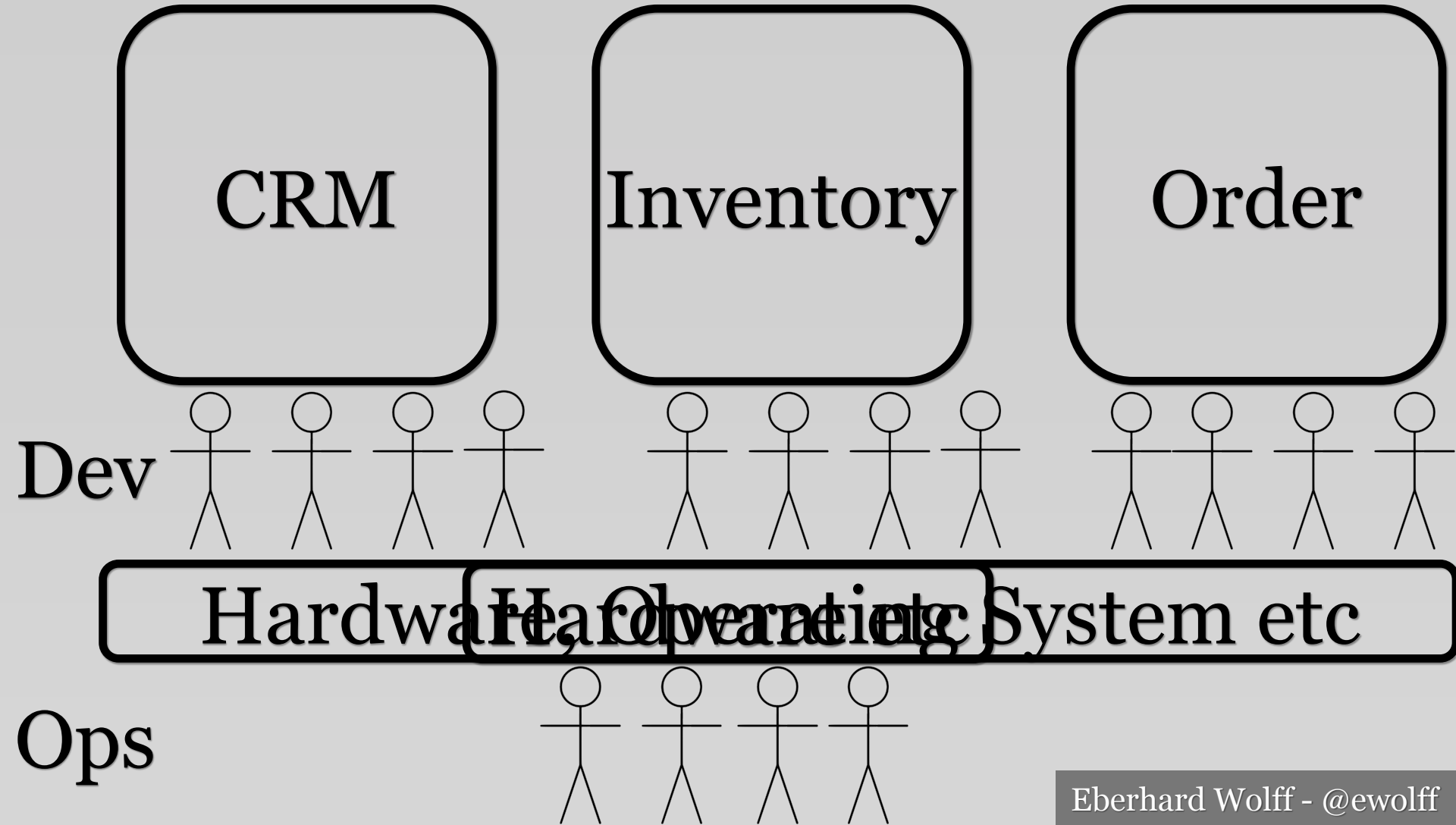
- Customer only cares about applications
- Not infrastructure
- So: Ops not important?
- Different goals
- Dev: Bring out new features quickly
- Ops: Stable platform and applications

# *Problem with Traditional Organization*

- At the end Ops and Dev both serve the customer
- Both own stability and new features

- But not represented in organization
- …or incentives

# *DevOps Organization*



# *Dev Ops = Customer Oriented*

- Team owns feature
- …and stability
- i.e. value of the application to the customer
- Clearly responsibility
- Competitive edge for in-house IT

# *Sum Up*

# *Continuous Delivery in Enterprises*

- Faster time to market
- Only if environment changes accordingly

- Otherwise: Lower risk
- Automation & reproducibility
- Smaller batches for releases

# *DevOps in Enterprises*

- DevOps = Culture
- Encourage communication
- …and sharing tools

- Optional: DevOps = Organization
- Customer Oriented IT
- Optimization for flexibility instead of cost

*Thanks!*