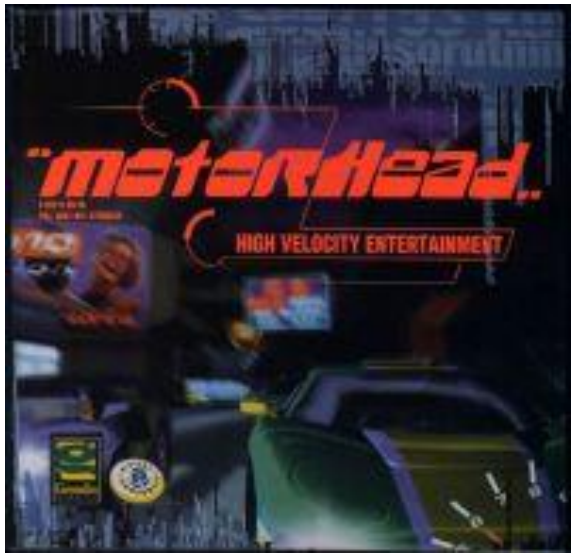
Working in small and large teams

Mikael Kalms, DICE

Me.

My formative years

I've worked on games.



I've worked on games.

I've worked on games.

I've worked on games.

But, on to the real topic.

Working in small teams

Team size: one person… you.

Is this you?

1. “I’m gonna build something”.
2. You build something.

You could analyze things a bit…

Why are you going to do it?
RATIONALE

How are you going to do it?
APPROACH

Who wants the end result?
CUSTOMER

How do you know when you're done?
GOAL

Typical rationales

There are no tools that allow me to do X, so I will build one

The tools that allow me to do X all suck, so I will build yet another one

I want to learn Y, and the process of building X will help me learn

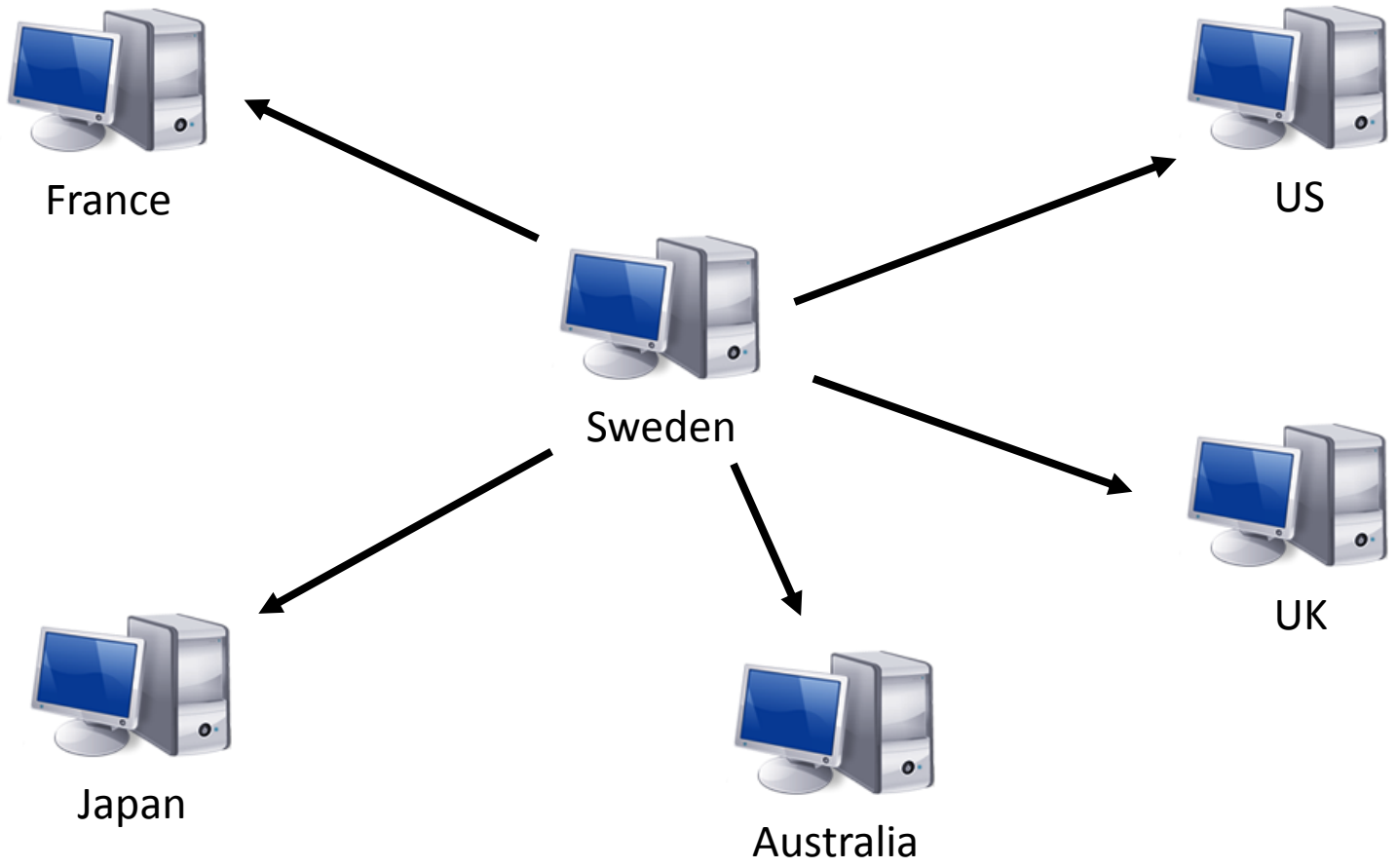
I want to show myself/someone special that I can build X

Having a silly rationale is fine

Just be honest to yourself about it OK?

Let’s get real

Real world scenario



Real world example

1GB file

Up to 100 recipients

File must be modified for each recipient

Old method: Command line tools, FTP file transfer, hours of manual work

It’s pretty crap.

What’s the rationale?

Existing solution manual, error-prone, insecure
No existing solution supported our particular use case
It seemed like a webservice could solve the problem nicely

I wanted to learn how to build webservices
I wanted to add new tools to my toolbox
I wanted to re-think how I work as an individual

-> Helps the company and I learn new stuff at the same time

Who is the customer?

Me and my closest colleagues

Developer = customer makes it easier for the programmer to build the right thing … but it is no guarantee!

What is the approach?

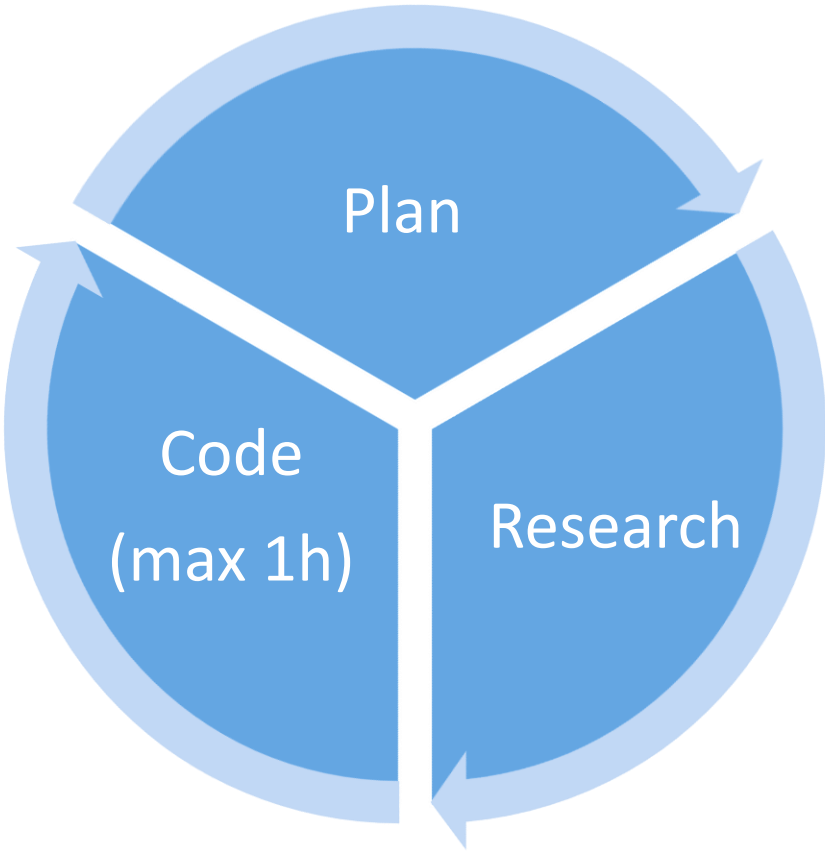
Quick iteration time is king!

Every iteration should **add value**

Change the plan after **every iteration**

Maintain development speed as the project grows larger

Start using the program in production **early**



Short iteration times

Spend as long as you like on planning & research, but code for **max 1 hour**… and then plan again. There are no sprints and no fixed milestones.

Positive side effects:

You ask yourself “am I doing what’s best for the project?” several times per day

You learn to plan in as much detail as the situation requires – no more, no less

You learn to break down tasks into small yet complete chunks

What’s this ‘value add’ then?

Value comes in many forms.

- More features
- Higher quality
- Improved development situation

In short, it makes the product better either directly or indirectly.

If the current task does not add value, why are you doing it?

Maintaining development speed

Don’t get stuck, and don’t go a lot slower as the project grows in size. Which tools do you have in your toolbox?

- Write simple code
- Employ defensive programming practices
- Manage technical debt
- Use automated testing
- Use version control
- Use continuous integration

Start using the program in production early

“No plan survives contact with the enemy.”
— *Field Marshal Helmuth von Moltke*

Break time!

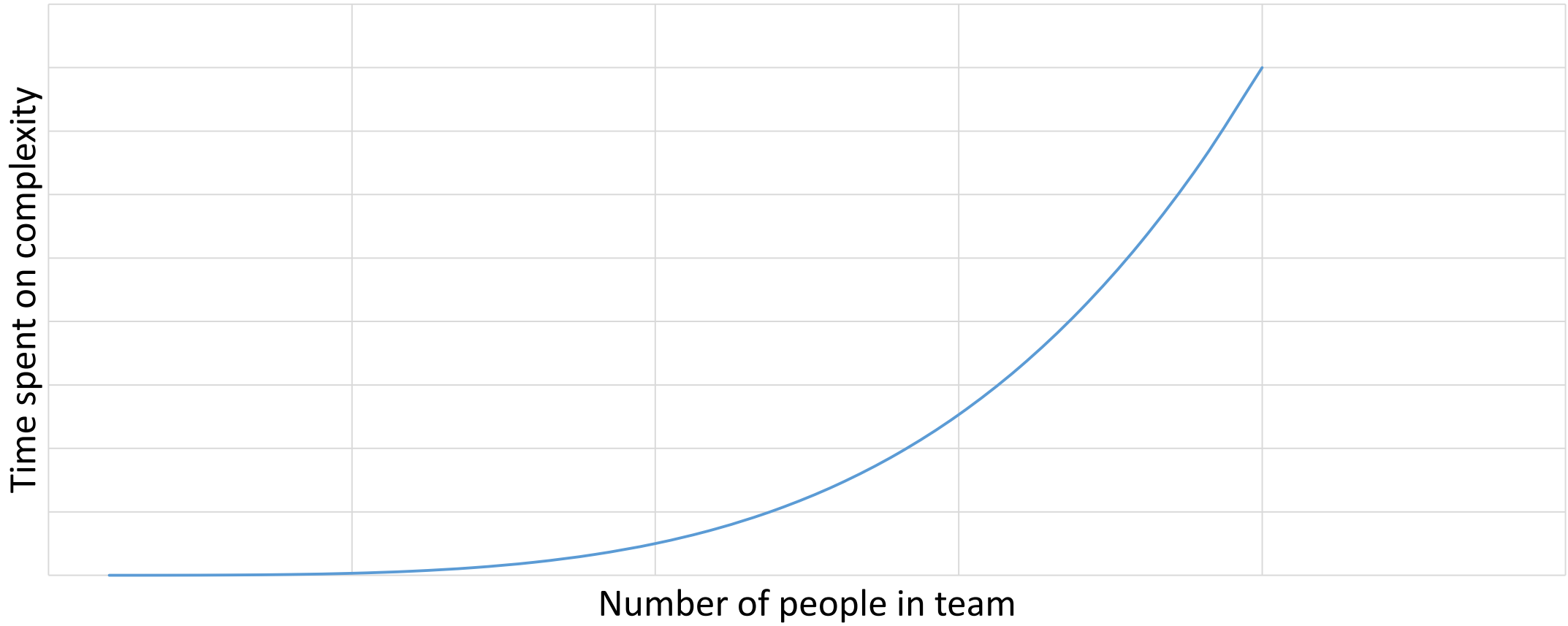
Working in large teams

Team size: 100 people… and then you.

The advantage of large teams

You can build more, larger and better things!

Problem #1 with large teams



Problem #2 with large teams

Knock-on effects

- We don’t know where we are
- We don’t know where we are going
- Course corrections are difficult and take time

How do we avoid those problems?

Create clarity

Create structure

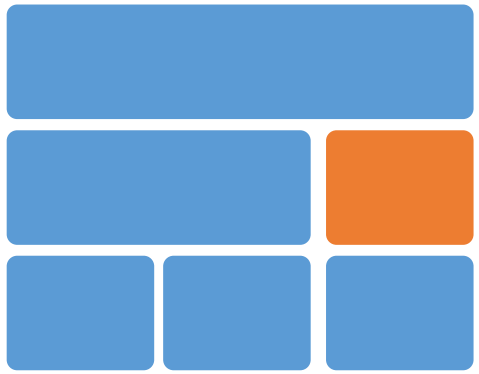
Create processes

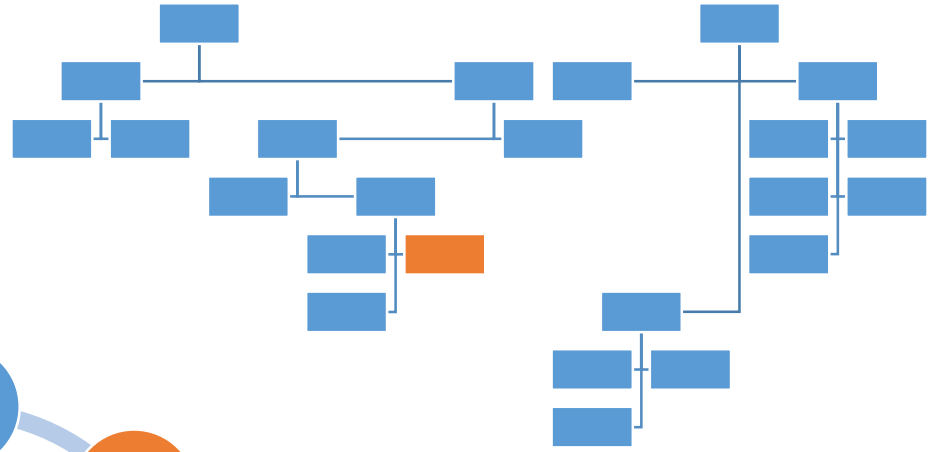
Use technology to your advantage

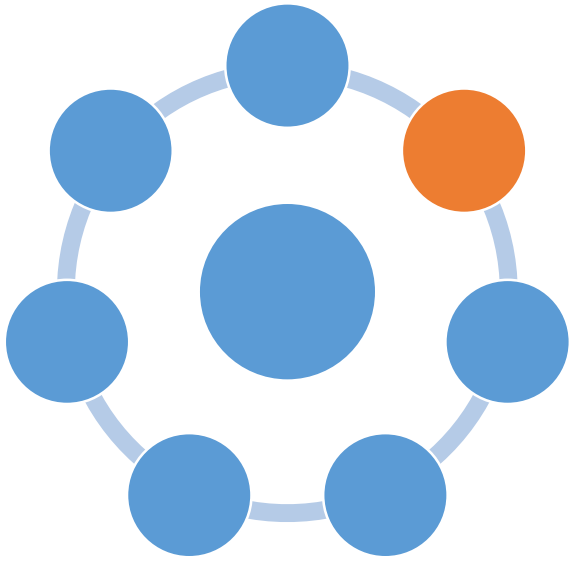
… but not too much of any of the above.

Keep on improving.

What’s your role in the team?







Always strive for clarity

What is the entire team aiming for?

What is my local group aiming for?

What am I aiming for?

Is there a conflict hidden here?

Communication is not important

Relevant communication is, however.

Clarity, structure and process all make it easier to avoid irrelevant communication.

You cannot avoid politics

Embrace it, minimize its impact on your area, or ignore it at your peril.

Fight the good fight

You can make your corner of the organization work better.

Please do. The world needs more heroes.

My three top tips

Solve your stuff, help your colleagues with theirs.

Make yourself replaceable.

Never stop improving.

Questions?

Mikael Kalms

mikael@kalms.org