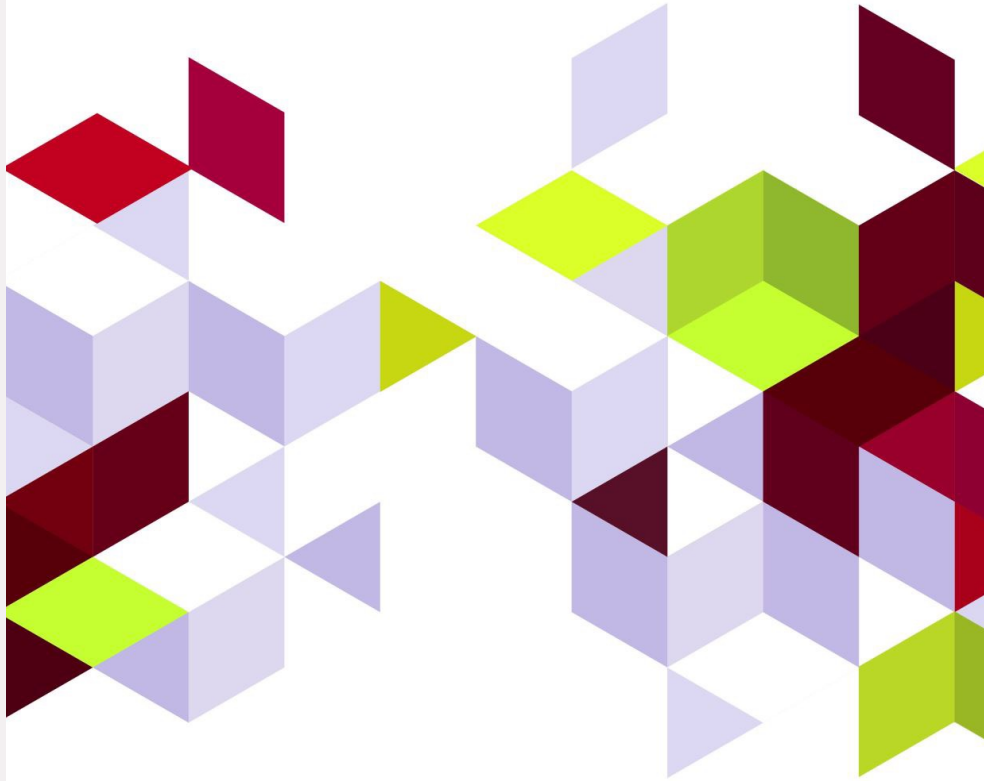
0.1X TO 10X: UNPACKING BUSINESS VALUE IN SOFTWARE ENGINEERING

XTC Berlin – July'23 Edition

Alex Fedorov



LET'S DEFINE TERMS OF BUSINESS VALUE FIRST

Expected Business Value

"Business value is an indicator of the benefits expected to be gained by a proposed change." — agileambition.com

*Example: we **expect** this feature to save our operations 30 minutes every day on average.*

PLANNING → SHIPPING

Shipped Expected Business Value

An indicator of the expected benefits gained by a set of changes that have been made available to the customers.

*Example: The team has developed and **shipped** the feature in **production** that is **expected** to save 30 minutes every day for our operations on average.*

SHIPPING → VALIDATION

Actualized Business Value

A validated measure of the actual benefits gained by a set of changes that was used by the customers.

*Example: We've **measured** how much time operation spends on this and we've improved it only by 20 minutes per day on average with this feature.*

FOCUS OF THE DISCUSSION

- We seldom have control over Actualized Business Value.
- Let's focus on the Shipped Expected Business Value because we have control over it.

TYPES OF SOFTWARE ENGINEERING WORK

- Features — can be with or without Expected Business Value
- Bug fixes — retain existing Actualized Business Value (risk management)
- Tech maintenance work — retain existing Actualized Business Value (risk management)
- Big rewrites — can be with or without Expected Business Value
- Research — can be with or without Expected Business Value

HOW DOES 0.1X LOOKS LIKE?

Software Engineers spend **98%** of their time working on the items that either retain the existing Actualized Business Value, or have no Expected Business Value.

1 unit of work out of 50, on average, has some Expected Business Value.

HOW DOES 1X LOOK LIKE?

Software Engineers spend **80%** of their time working on the items that either retain the existing Actualized Business Value, or have no Expected Business Value.

1 unit of work out of 5, on average, has some Expected Business Value.

HOW DOES 4X LOOK LIKE?

Software Engineers spend **20%** of their time working on the items that either retain the existing Actualized Business Value, or have no Expected Business Value.

4 units of work out of 5, on average, has some Expected Business Value.

HOW DOES 10X LOOK LIKE?

5x is theoretical maximum, assuming the same speed and efficiency.

Software Engineers spend only **2%** of their time working on the items that either retain the existing Actualized Business Value. They don't work on items without Expected Business Value at all.

They are also more than **2x** more efficient in their work.

QUESTIONS TO DISCUSS

- How might we avoid falling into 0.1x trap (or coming anywhere near that)?
- How might we improve to get to at least 4x?
- What does it take to increase efficiency to get to full 10x and beyond?

THANK YOU!