PRACTICE OF MAXIMIZING THE WORK NOT DONE

Tampere Goes Agile 28.10.2017

TAMPERE goes AGILE

@VilleTormala
villetormala.com

Simplicity – the art of maximizing the amount of work not done – is essential.

1) CLARIFY YOUR MISSION

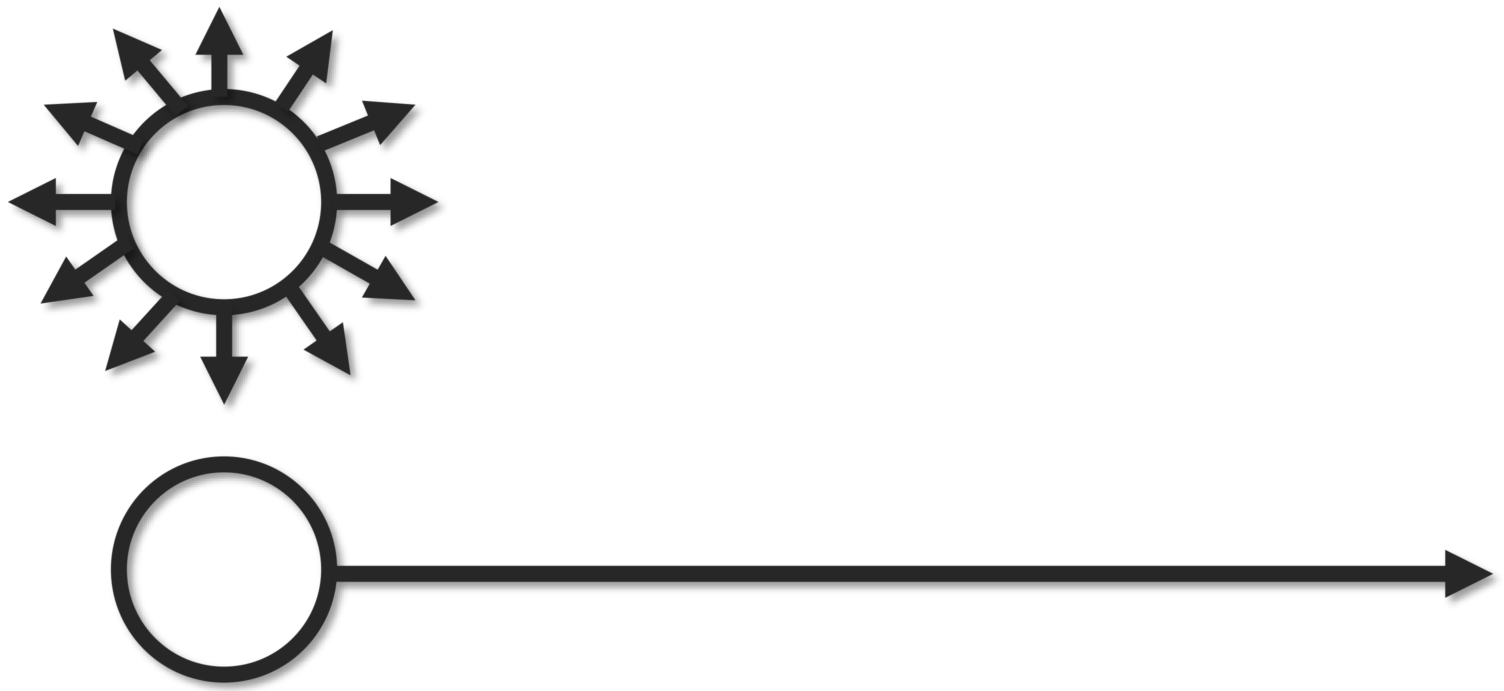
… and repeat

When you know exactly what you are aiming for, you can **use extreme criteria to leave things out.**

Saying "no" is about having a clear criteria.
Only then, what you <u>don't do</u> becomes as important as what you do.

Being purpose and mission driven is not easy

On the other hand, doing little bit of all the 248 things means getting nothing done.



Adopted from: Essentialism: The Disciplined Pursuit of Less, Greg McKeown

2) DO ONE THING AT A TIME

No, you are probably doing more

It's **impossible to optimize value without prober feedback cycle.** That is, it is impossible to YAGNI with a big upfront design.

- Doing the work is supposed to create more knowledge about what to do next, what to leave out and when to stop.
- Work in small pieces, keep making them smaller all the time.
- Optimizing value is about delivering as little software as possible.

It’s not just about “continuous delivery.” it’s about how different ways of working provide different capability and tools for making things simple.

- Focus
- Deciding late and having more options
- Removing unnecessary things
- Having peace of mind and more time

3) INSIST TECHNICAL EXCELLENCE

Spot bad SW development mindset

There needs to be a clear and shared understanding what it means to create a good technical solution.

The idea is to keep things efficient and fun <u>in the long run.</u>

- "Add value" = add more stuff
- "Fix things" = add more stuff
- "Update things" = add more stuff
- "Improve things" = add new framework
- "Refactor" = add more stuff
- "Effective work" = add more stuff faster

Doug McIlroy

The notion of "intricate and beautiful complexities" is almost an oxymoron. Unix programmers view with each other for "simple and beautiful" honors — a point that's implicit in these rules, but is well worth making over.

We used to sit around in the Unix Room saying, 'What can we throw out? Why is there this option?' It's often because there is some deficiency in the basic design — you didn't really hit the right design point. Instead of adding an option, think about what was forcing you to add that option.

Basics of the Unix Philosophy, http://www.catb.org/esr/writings/taoup/html/ch01s06.html

Yes, there is often essential complexity in the problem itself. And yes, there are deadlines.

But often much of the complexity is added by us. We do it by ourselves.

4) EXPERIMENT WITH LESS

Less is more, more is worse.

You need to all the time actively declutter everything. Extra tasks and processes creep in all the time.

Try to see and value simplicity more.

“You just can’t remove wheels from the car to make it better”

Simple end result is at the same time very similar and very different.

Making things simpler is about making then different and better.

MEET OUR NEW HERO

Fumio Sasaki

Tokyo, Japan

Owns three shirts, four pairs of pants, four pairs of socks, and a few other belongings.

“It's easy to keep your kitchen table clean if you don't have much to put on it.”

“Because there aren't many items in the first place, the ones you need are easy to find.”

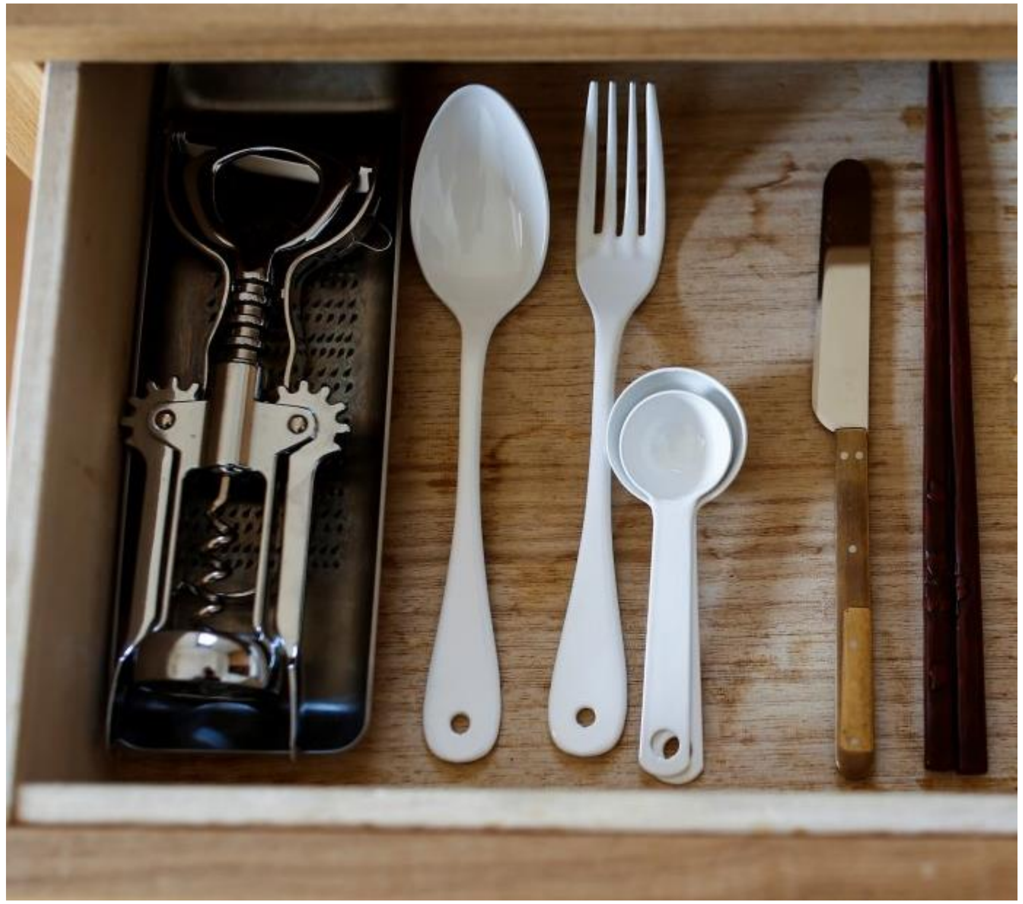
Only the absolutely necessary items. Nothing more.

You never need to think about which pot to use for which dish.

"Sink counters exist to keep stuff on them. All you need is a toothbrush, and that doesn't require one."

"I kept thinking about what I did not own, what was missing"

But sometimes this all means not owning a mop.



This is how Fumio's kitchen drawer looks like.

With every improvement action, **remember that Fumio has only one spoon and fork!**

- Improve by doing less
- Improve by taking something away
- Improve by stopping doing something

Epilogue

Reduce documentation and you will be agile

1) **Clarify your mission.** Doing things is not the same as getting things done.
2) **Do one thing at a time.** YAGNI doesn’t like big up-front design.
3) **Insist technical excellence.** Suck less with your codebase.
4) **Experiment with less.** Remember Fumio.

Simple systems are rare

1. For many professions, us included, **simplicity is the ultimate goal.**
2. **It’s rather easy to create complex solutions.** Even a monkey can do it.
3. **Simplicity and elegance is neither simple nor easy.** It’s difficult.
4. **Simple doesn’t mean less work, it means more thinking.** Often it means more work but it’s worth it in the long run.

There is clear value in simplicity:

1. Simplicity and clarity lead to good design.
2. Simple and small systems are understandable and easy to adopt. And easy to discuss about.
3. …with great potential to be reliable and fast.
4. Simple designs are easier and faster to change.

THANK YOU!

YOU ARE ALL AWESOME!

@VilleTormala
villetormala.com