

“The future of teamwork coordination” - A project for Aalto Software Project course (CS-C2130)

## 1. Introduction

Different groups across the globe have a constant need for managing their workstream and supporting their team coordination. Many tools have been developed for the purpose, but they tend to be cumbersome and unnecessarily feature-rich, while lacking certain features that a seasoned Agile coach would immediately do on a whiteboard.



Not all teams have the luxury of their own space and wall, and thus a digital tool is needed for coordinating a dispersed team. Even those that do, may want to have a digital system for workflow management due to organizational benefits, or simply wanting to have access to the workflow while outside the team room.

Wouldn't it be lovely if there was a free tool that anyone could take immediately into use, that contains just the necessary features with an easy-to-use experience? We estimate that creating such a tool would not be a great amount of work, if we maintain relentless focus on the essential and resist the feature creep.

## 2. Project Goals

In this project we aim to develop the first version of just such a tool. This version will contain a set of basic features, the kind of things that an Agile professional would immediately create using a whiteboard, and a couple of basic project management tools derived from that.

The end result is a graphical, simple workstream management tool, for an individual or a team. Think of a simpler Trello designed by an Agile veteran of software development. Think of the abstraction levels and most important reports of Jira with a user friendly interface.

The software will be open source and created using technologies that are easy to run and modify for convenience of future admins and users. The underlying data structures will enable expanding the tool and its reporting capabilities.

### Differentiators from existing tools

- Swimlanes
- Note corner text (size, caretakers etc.)
- Free and open source
- Simple accessibility/login
- Convenient integration of upper level Kanban and lower level team iterations (larger items in decision process before moving to implementation and being split to smaller, executable actions)
- Convenient basic scope/progress reports for projects

### The overall problem statement user story

As a team facilitator who understands applying Agile methods,  
I want a tool that provides elegantly the basic functionality of a team wall, while enabling remote access, mobile access, and software-based reporting,

In order to:

- More effectively work with teams where members are frequently not in the team room
- Provide remote transparency to team situation
- Use one tool for project work management, prioritization, grooming, planning, and teamwork coordination (to avoid unnecessary synchronization work)
- Retain possibility to have swimlanes, visual sizes of items, and WIP limits (per state & person)
- Gain automatic measurements on lead times, throughput, and similar basic lean metrics
- Gain a release burn-up report
- Motivate team members to use such a nifty tool
- Motivate managers/leaders to guide their organizations with user-friendly, simple tools

## 3. Technologies

Complete freedom to choose, as long as Product Owner can be convinced that the choice is:

- Free
- Widely used
- Easy to support/develop
- Easy to run for any plausible team
- Supports different user terminals
  - Desktop first, mobile: simple use cases

Suggested:

- Container or two for public cloud with automatic installation script for AWS
- Front: React
- Back: Python - Django
- Event logging: flat text files

## 4. Requirements for the students

The team is expected to be interested or enthusiastic about collaborative teamwork and learning practical Agile. This topic is probably of moderate difficulty.

## 5. Legal issues

### Intellectual Property Rights (IPR)

The IPRs of the Results shall be governed according to the principles stated in an open source software license.

### Confidentiality

No confidential information will be shared with the students.

## 6. Client

The client is Aalto University. The client will be represented by **Towo Toivola** (visiting lecturer from Futurice) and **Jari Vanhanen** (Project sponsor, Aalto University).

Towo Toivola will act as Product Owner for the team. His expertise in Agile software development and software quality is extensive. However his expertise in UI technologies is limited. Towo is committed to guiding the team as in Scrum, and perhaps a bit more, with greater emphasis on the beginning of the project. Towo is likely to also share real-world experience and wisdom with the team.

Aalto University may support the team with accounts/expenses and the like. Futurice may provide space for occasional meetings.

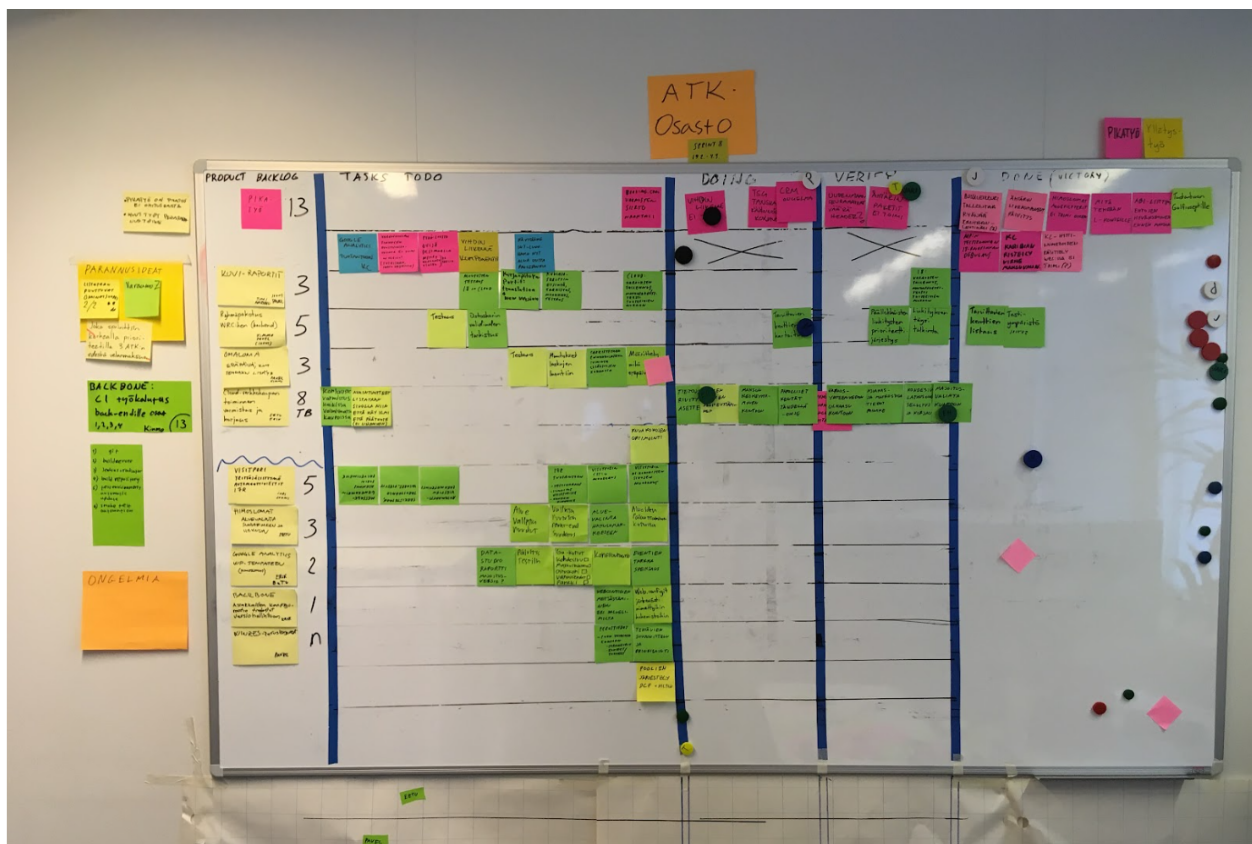
Contact information:

Towo Toivola - Product Owner

[towo.toivola@futurice.com](mailto:towo.toivola@futurice.com)

+358 40 590 2112

## 7. Additional information



Some guidance on the approach for the project

- High attention to visualizing details: Make situation easy and fast to grasp
- Use an instance of the system for guiding work on the system as early as we can
- Less, but better
- Is of great use for a Kanban board, a Scrum Sprint board, and a Product Backlog (with maturity steps and planning alike)

Here is the top of the current backlog as a taster of the project

- "Hello world" via a cloud service
- "Hello world" via Git and CI
- Automated unit tests
- Picture of a board
- A named board created by user
- A sticky with text can be created
- I can define states for stickies
- I can move stickies between states
- ...

## Main functionality

### Work management (initial backlog)

- M: I can create boards with names
- M: I can create stickies with titles
- M: I can groom a story to stickies
- M: I can define states for stickies
- M: I can move stickies between states
- M: I can set a size that shows clearly for stickies (input as a number)
- M: I can remove stickies from system
- M: I can remove stickies from board
- S: I can set a stickie to be one of several colors
- M: I can define swimlanes with names for a board
- S: I can add acceptance criteria to stickies (checklist)
- S: I can set WIP limits for states (number of stickies)
- S: I can set WIP limits for states (total size of stickies)
- M: I can access the system from laptop as well as phone
- S: I can see who is in the team
- S: I can see what different team members are currently working on (with WIP limit - magnets)
- S: I can add comments to stickies
- S: There exists a log of changes made to stickies (who, when, what change)
- M: An owner can be defined for a sticky
- S: Supporting team members can be defined for a sticky
- S: I can get sticky to appear on board A as well as B
- S: I can define a sticky on board A to be a swimlane for board B
- NTH: indicate remaining work vs estimated work for an item (like 3/8)

## Work planning

- Velocity calculations

## Reporting

- S: Data can be easily gathered for throughput of a board in a given timeframe
- NTH: Burn-up chart for a defined scope (a number of stickies with certain tag?) can be shown
- S: Cumulative flow diagram of stickies in different states as function of time

## Performance

- M: operation for less than 100 stickies must be immediate
- S: operation for less than 1000 stickies must be immediate
- S: if something takes a longer time, user must be immediately informed that the system is working on whatever was requested

## Business support

- Reports on # users, # actions, # boards..
- Promoting the original creators for fame and glory
- Gathering feedback on the tool

## Security

- This is a low priority aspect of quality for this system

## Interoperability

- Layers of GUI, business logic, and data storage should be separated by clear APIs to enable future configurations of different implementations for each layer.
- Must be usable from Mac, Android, iOS, Windows.
- Should be deployable to work in public and private networks

## Reliability

- System should not lose data in the event of an error
- System should automatically restart in case of an error
- System should handle unavailability of some layer gracefully

## Supportability

- Deploying of an instance of the tool should be trivial
- Error situation: logs should be created and stored automatically, alert should be sent automatically (email?)
- State of the work must be transferable to another instance in a convenient way

## Testability

- An API needed that can be used to create/manipulate boards and stickies and to fetch current situation in a format that is computer and human readable

## Implementation guidance

### User experience

- 1) Fast, convenient, and reliable user experience is the key to value with this tool. We want to make using this as handy as possible, otherwise users will get frustrated instead of delighted.

### Interoperability (future concern)

- 1) At some point we want partial **integration with Jira**. Thus, create a layered approach that allows us to easily add alternate data storages.

### Supportability

- 1) System is to be implemented with **modern cloud methods** that enable **trivial deployment** and managing the entire system configuration as code in version control.

Please maximise the work not done, while maximising value.

## Other potential features

(most of these would just be a piece of text that shows somewhere around the workflow)

- Legend?
- DoDs?
- Team member list? Availabilities?
- Team agreements?
- Wall rules?
- Meeting notes?