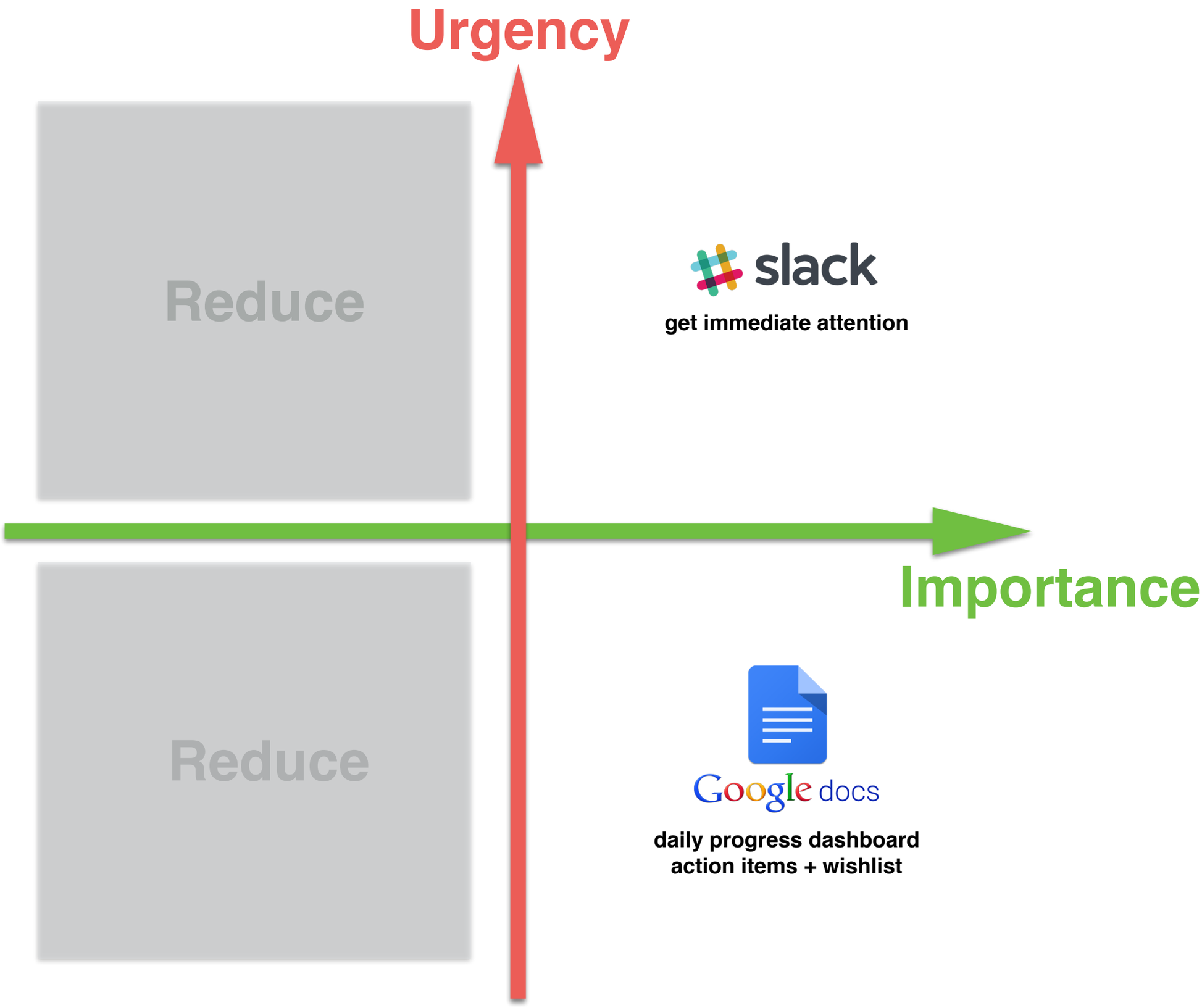
Lean + Agile

Collaboration Guidelines and Schematics

Communication Strategy



Workload Balance and Optimization

Phase 1

Subteam 1 — EDA

CS WZ

Subteam 2 — Pipeline

SO YF

Phase 2

Individual Iterations

CS

WZ

SO

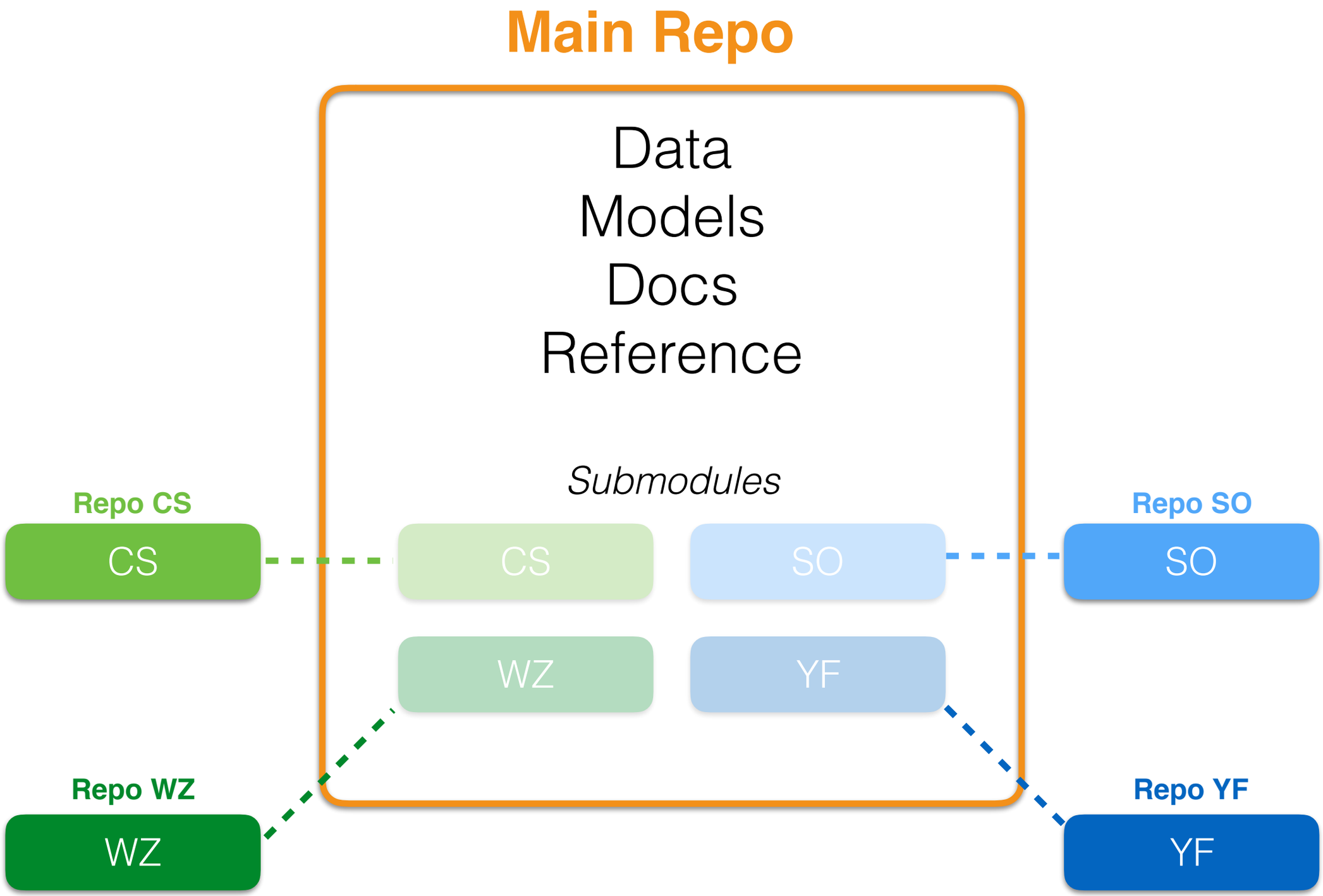
YF

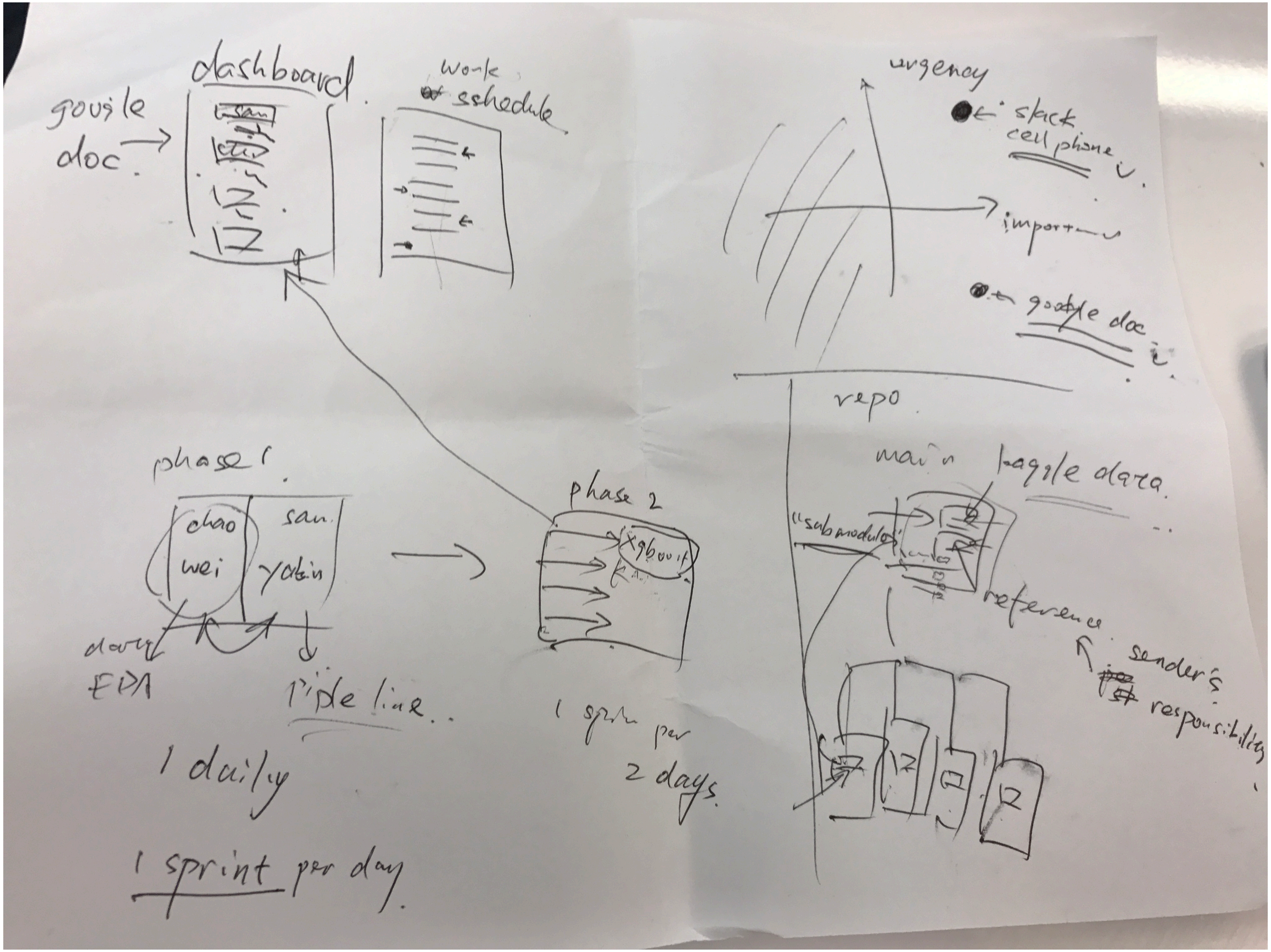
Wrap Up

Model Stacking

All

GitHub Repository Structure




dashboard
doc
work schedule
urgency
slack
cell phone
importance
google doc
repo
main
kaggle data
"submodule"
reference
sender's
responsibility
phase 1
chao
wei
san
yabin
EDA
pipeline
phase 2
xgboost
1 sprint per 2 days
1 daily
1 sprint per day

1) Effective communication strategy

"Continuous Improvement" + "Eliminate Waste"

General observation:

The four of us often have different schedules. We are at different places especially during the weekends. We only have 2 weeks on the project.

What do we want?

Yes -- early feed back, knowledge share, ability to make fast response
No -- frequent meetings and interruptions

Yes -- informative dashboard so everyone knows other people's progress
No -- complicated/tiring documentation system

Yes -- version control and file sharing system
No -- sending files in unorganized fashion through multiple channels

Proposed solution:

a) **Important** and **Urgent** info goes to **Slack**. We exchange our cell numbers just in case.

b) **Important** but **Non-urgent** info goes to **Google Docs**. Google Docs = progress dashboard + key strategy reference + ticket pool (wishlist)

Each of us would write in our own sections. At the end of each day, type at least 1-2 sentences describing the progress. Feel free to write more if needed. As soon as we enter the iterative computational stage, start making and updating a picture to provide quantitative visual update.

The high level project plan is stored here. Feel free to choose a color arrow to mark where you are.

Record your non-urgent wishlist here. After each sprint (1-2 days) we go over the wish-lists and decide what to work on for the next sprint.

c) **File sharing** and **version control** with **GitHub**. We would make **one main project repository**, while each of us create our own **individual** repo within our own accounts. We use the "**submodule**" method to link our individual repos to the main repo. Kaggle source data will be stored in the main repo; all files shared through all other channels will be copied to a "reference" folder in the main repo too.

c) **Unimportant communication** should generally be **limited**.

d) Avoid important discussion when certain members are missing -- we don't want to waste time saying the same things many times. We would assign relatively independent tasks to Wei.

2) Work load balance and optimization

"Just In Time" + “Regular reflection & adaptation” + “Simplicity”

General observation:

Feature engineering and pipeline building are both adaptive and continuous effort throughout the 2-week project. Wei and Chao would physically see each other after 6pm; Sam, Yabin and Chao overlap during the day.

What do we want?

Start data digestion and pipeline building both on day 1. Enable fast-track solution delivery, effectively optimize computational schedules. Minimize overlapping effort while guarantee knowledge backup.

Proposed solution:

Two phased project cycle --

a) **Phase 1, two sub-teams, daily sprints.** During the initial stage, we divide our team into two sub-teams. Chao and Wei, Sam and Yabin. One team will focus more on Data cleaning, EDA and feature engineering, the other more on code structure design, pipeline building and testing. There will be frequent communication within a sub-team, but only one scheduled daily end-of-sprint communication for the full team.

b) **Phase 2, four parallel working ants, 1-2 days sprints.** After the initial phase, we would sync our understanding of data characteristics and algorithm pros and cons. We each pick one algorithm and move into four parallel data crunching tasks. Sprints are now longer. We would record our daily progress in the google doc, but only have a scheduled meeting every 1 or 2 days.