Software Development Organization Practices VS Product Maturity Phases

Krzysztof Kobus
kk@glass-spider.io

Motivation

- Most startups require software development team (in house or outsourced) to develop product or platform supporting their business
- Product maturity relates to achieved product-market fit, funding stage and available financial resources
- Organization of software development remarkably depends on maturity of the product

Product Maturity Phases

- Prototype
- MVP – Minimum Viable Product
- Launched Product (first regular users / customers)
- Established Product (plenty of regular users / customers)

Software Development Organization Practices

- Source code standards (code formatting, variable naming conventions)
- Source version control (branching)
- Test driven development (unit tests, functional tests)
- Frequency of releases
- Code reviews (pull requests)
- Pair programming
- Common code ownership
- Constant refactoring and simplification
- Focus, core functionality, error situations
- Technical documentation (source code, wiki)
- Formal work organization (Scrum)

Software Development Organization Practices

- Well defined tasks, milestones

Prototype

Just idea – no team, no funding, high risk

- Creative way to demonstrate the concept
- Make it cheap – no other rule

Universal Practices

- Common code ownership
- Source code standards (code formatting, variable naming conventions). Few key rules.
- Constant refactoring and simplification
- Documenting source code
- Well defined milestones (what goal to achieve)
- Pair programming – exclusively for targeted problem solving
- Frequent team updates on goals to achieve

MVP – Minimum Viable Product

Still just an idea of product formula – no or little funding, no traction, high risk

- Frequent “releases” – every code commit is new “release”
- Very basic source version control
- Focus on core functionality, accept bugs and limitations at scale
- Very limited and only fundamental documentation (wiki)
- No formal work organization like Scrum

Launched Product

Basic formula identified, some funding, traction, medium-high risk

- Source version control (branching, tags)
- Frequent releases – regularly tagged / branched
- Basic test-driven development techniques (unit tests, functional tests)
- Focus on core functionality and typical error situations
- No formal work organization like Scrum – too frequent scope changes
- Well established fundamental documentation (wiki)
- Code reviews (pull requests) for changes of high risk of regression only
- Backlog of well-defined tasks

Established Product

Formula identified, decent funding, traction, low risk

- Full code reviews, if needed also by two engineers
- Less frequent releases – regularly tagged / branched
- Well established full documentation (wiki)
- Sophisticated source version control usage
- Formal work organization like Scrum
- Sophisticated test-driven development (unit tests, functional tests)
- Focus on stability, fully serviced error situations

Krzysztof Kobus

CTO & Founder, Glass-Spider

kk@glass-spider.io

www.glass-spider.io

www.linkedin.com/in/krzysztofkobus/

www.flat6labsbahrain.com/mentor
/krzysztof-kobus/

© Copyright Krzysztof Kobus 2020