LPRD

LEAN PEER REFACTORED DEVELOPMENT

syndicats

DISCLAIMER

The information contained in this document is intended for personal use only. Circulation or reproduction of this document is not permitted.

KEY CONCEPTS

PHASES

Implementation is split into **two phases** - approval by stakeholders and approval by the team

- Quick implementation is the top priority for approval by stakeholders
- Maintainability and longevity of the solution are paramount when it comes to team approval

CONCEPTUAL FRAMEWORK

- As opposed to traditional code reviews, refactorings are done by the developer who is reviewing the code, **not** the original author
- Stakeholders and team members communicate with each other *directly* and *quickly*
- There are neither estimates nor arbitrary time frames! Development and deployment are continuous flows
- Requirements are framed informally

ROLES

ROLES IN LPRD

STAKEHOLDER

Stakeholders can file *Requests*, as well as prioritize and approve *Deployables*.

Each stakeholder can make any decision **autonomously**. Thus, every change request is regarded as **binding**.

TEAM-MEMBER

Team-Members have two main responsibilities:

- Translate *Requests* into *Deployables*
- Implement *Deployables*

ENTITIES

REQUEST

- Exclusively filed by *Stakeholders*, Requests contain all the information needed to know exactly **what** behavior of the system needs to be changed.
- Requests are regarded as **binding**, so as soon as a *Team-Member* notices a new Request, time will be invested to translate it into one or more *Deployables*.
- *Stakeholders* are responsible for ensuring that Requests are unique and don't interfere with other Requests or *Deployables*.
- Ideally, Requests are self-contained and not linked to any external source of information.

DEPLOYABLE

- Deployables are derived from *Requests* and created solely by *Team-Members*. They describe **how** to change the system in order to fulfill the original *Request*.

- They are self-contained and include a link to the original *Request*. Any *Team-Member* and any *Stakeholder* must be able to understand the description.

BOARD

- The Board displays every *Request* and *Deployable.* The current status is reflected by their *Lane*.

- It is regarded as the single source of truth. Any communication related to *Requests* or *Deployables* must be documented in the respective Entity on the Board.

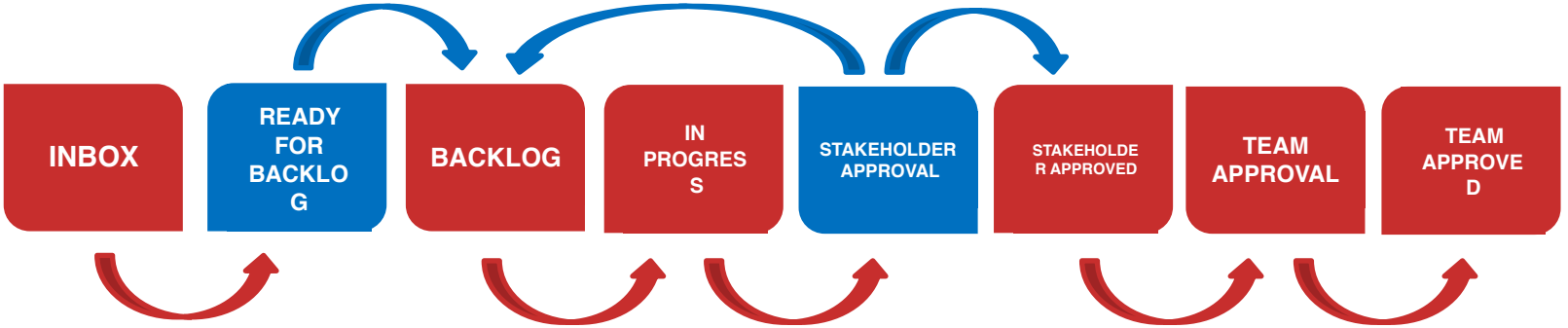
- All *Team-Members* and all *Stakeholders* can access the Board at all times.

LANE

- A Lane represents a group of *Deployables* or *Requests* that share the same status.
- How and by whom Lanes can be filled or emptied is clearly defined for each Lane.

LANES

LANES



Stakeholder responsibilities

Team-Member responsibilities

LANE: INBOX

INBOX

The Inbox contains all *Requests.*
Clearing the Inbox is the first priority of each *Team-Member.*
This Lane should be empty most of the time

IN

Stakeholder create *Requests* directly in the Inbox

OUT

Team-Members can

1. Label Requests as Blocked and ask at least one Stakeholder to clarify
2. Translate Requests to Deployables and transfer them to Ready-for-Backlog
3. Link created Deployables to the original Request and archive the Request

LANE: READY-FOR-BACKLOG

READY-FOR-BACKLOG

All Deployables that resulted from Requsts are waiting to be prioritized by a Stakeholder can be found here.
This Lane should be empty most of the time

IN

Team-Members can create Deployables here

OUT

Stakeholders can move Deployables to the Backlog

LANE: BACKLOG

BACKLOG

The Backlog holds all Deployables that have been prioritized by Stakeholders.
It is strictly sorted by priority from top to bottom starting with the most urgent Deployable.
Stakeholders can change the priority in this lane any time.
There will be a lot of Deployables in this Lane most of the time.

IN

Stakeholders can prioritize Deployables by moving them here from Ready-for-Backlog.

OUT

Team-Members move the top Deployable of the Lane to In-Progress when they start working on it.

LANE: IN-PROGRESS

IN-PROGRESS

Contains all Deployables that are currently being implemented by Team-Members.
There will be a few Deployables in this Lane most of the time.

IN

Team-Members move the top Deployable of the Backlog here when they start implementing it.

OUT

Team-Members move Deployables to Stakeholder-Approval when they are done with the implementation.

LANE: STAKEHOLDER-APPROVAL

STAKEHOLDER-APPROVAL

Contains all Deployables that are ready to be approved by a Stakeholder.
This Lane should be empty most of the time.

IN

Team-Members move Deployables here from In-Progress when they are done with the implementation.

OUT

Stakeholders can:

1. Move a Deployable to Stakeholder-Approved if they agree with the implementation.
2. Move a Deployable back to the Backlog and label it as Rejected if they disagree. Rejections focus solely on the purpose of the Deployable. If new requirements emerge during approval, they have to be filed as a new Request.

LANE: STAKEHOLDER-APPROVED

STAKEHOLDER-APPROVED

Contains all Deployables that were approved by Stakeholders.
There will be many Deployables in this Lane most of the time.

IN

Stakeholders move Deployables to Stakeholder-Approved if they agree with the implementation.

OUT

Team-Members move Deployables to Team-Approval when they are ready to review and possibly refactor the implementation.

LANE: TEAM-APPROVAL

TEAM-APPROVAL

Contains all Deployables that are currently being reviewed or refactored by Team-Members.
There will be very few if any Deployables in this lane most of the time.

IN

Team-Members move Deployables here from Stakeholder-Approved when they are ready to review and possibly refactor the implementation.

OUT

Team-Members move Deployables to Team-Approved when they agree with the implementation or have refactored it.

LANE: TEAM-APPROVED

TEAM-APPROVED

Contains all Deployables that were reviewed and possibly refactored by Team-Members.
There will be a lot of Deployables in this Lane most of the time.

IN

Team-Members move *Deployables* here from Team-Approval when they are done with its refactoring or agree with the implementation.

OUT

Stakeholders can archive Deployables.

LABELS

LABEL: BLOCKED

If a state transition is impossible, *Deployables* are labeled as Blocked regardless of the *Lane* they are in.

Steps

1. Leave a comment explaning why the Deployable/Request is blocked
2. Add the **Blocked** label
3. Assign the person you think is most likely to be able to resolve the blockage

LABEL: REJECTED

A Stakeholder disagrees with the implementation of a Deployable during Stakeholder Approval

Steps

1. Leave a comment explaining why the Stakeholder disagrees
2. Add **Rejected** label.
3. Move *Deployable* back to *Backlog*.

RESPONSIBILITIES

RESPONSIBILITIES (ORDERED BY PRIORITY)

TEAM-MEMBER

1. Translate *Requests* in the **Inbox** into *Deployables*
2. Work/help on *Deployables* in **In-Progress**
3. Pick the top *Deployable* from the **Backlog** and start implementing it
4. Work/help on *Deployables* in **Team-Approval**
5. Pick any *Deployable* from **Stakeholder-Approved** and review/ refactor it

RESPONSIBILITIES (ORDERED BY PRIORITY)

STAKEHOLDER

1. Clarify *Requests*/*Deployables* labeled as **Blocked**
2. Prioritize *Deployables* in **Ready-for-Backlog** and move them to the **Backlog**
3. Approve/Reject *Deployables* in **Stakeholder-Approval**
4. Add new *Requests* to the **Inbox**

Thank you