FIELD OPS GUIDE

futurice

Transparency.
Caring.
Trust.

go.futurice.com/field-ops-guide

FIELD OPS GUIDE

Mission Briefing

Mission Briefing

In the beginning of a mission, check all the items on this list, succeed and live to see another adventure. You should be able to answer **yes** to each of the following questions, and only say **no** with a good reason.

However, if the mission situation gets critical, start eating this guide from the first section towards the last. Just remember to read & comprehend the pages before consuming them.

The guide is structured by project phase:

- Sales & Negotiation
- Beginning of a Project
- Project Execution
- Release & Delivery

Sales and Negotiation

Have the proposal scope & estimates been validated by the experts in that field?

WHAT? The work estimates presented in the proposal have been validated by someone whose skillset matches the job. For example, the amount of design work needed has been checked by an actual designer – preferably someone whose CV is in the Proposal.

The same applies to the implementation estimates & the tech competency.

WHY? If experts are not consulted when creating the work estimates, there is a higher risk that the estimates do not reflect the actual needs of the project. Ideally the estimates should be co-created with the experts that would likely be working on the project, lowering the risk of creating impossible deadlines or making it difficult for the project to achieve satisfactory results.

Does everyone understand what are the expected Deliverable(s)?

WHAT? Everyone, including all relevant stakeholders in the Client organization, knows the type of the expected delivery and its possibilities for further use: is it **a concept** created in a Service Vision Sprint project, **a prototype or a proof-of-concept**, or **a launchable MVP** of the service? Does everyone understand which of these (or some other goal) we are talking about?

WHY? Skyscrapers are not built on the foundations of sand castles. If the Team is building a throwaway prototype, everyone else needs to understand that it is not suitable as the basis for future development. Everyone needs to have a clear image about the expected capabilities of the deliverable, so that they can plan their future projects and investments accordingly.

What the end result **does not do** is as important to understand as what it **does do**. When everyone under-

stands the difference between the types of planned outcomes, they don't get surprised after the project: "this isn't the thing we bought, what the heck?"

Has it been validated that the problem is actually worth solving?

WHAT? It is proven (not only by the Client) that the end user has the problem we think they have – i.e. "this is a problem". We also know that solving the problem will resonate with the Client and the end users – i.e. "this is a problem worth solving".

WHY? One of our strengths is that we critically challenge our clients and their ideas for projects. If it seems that doing this project would not create value for the Client, **we need to have the courage to say no** to the money offered and to try to find an actual problem worth solving. Doing this will strengthen the partnership in the long run.

If the problem is not an actual issue for the end users, doing the project is a total waste and money is best used elsewhere. If the value of the solution is smaller than the cost of solving it, it does not make sense to allocate time & money to the project. Remember to **Ask why!**

Does everyone understand the project contract model?

WHAT? It is clear for everyone whether the project is **time & material** or if it is **fixed price & scope** – or some-

thing else. If the model is time & material, does everyone, especially the Client, understand that none of the features are guaranteed simply by the contract?

WHY? If this is unclear for either of the sides, you might end up in a difficult situation where it's not obvious what has or hasn't been agreed upon, and committed to, in any point of the project. Expectation management!

Does everyone have a shared assumption of the recognizable major risks in the project?

WHAT? You have discussed about the major risks in mind that might endanger the project. You have a mutual awareness about the risks – either party is not keeping any of the risks in mind a secret.

WHY? You have to map the major risks in order to be prepared for them. When both parties have knowledge about the known risks, their probability of materializing can be reduced more efficiently.

Beginning of a Project

Does everyone know why we are doing this project?

WHAT? Everyone – including the project team and **all** the stakeholders in Client organization – knows why

this project is done. If you ask anyone they will give you the same answer directly. Is this project being done e.g. because of legislation, is the Client trying to build new business, or why exactly?

WHY? When all the people in the project know why we are doing the thing we are doing, everyone is able to make decisions that help push the project to a glorious outcome.

Does everyone understand their role in the project?

WHAT? The Team members know what they are expected to contribute to the project, and how the responsibilities are shared between the different parties involved in the project.

The Client has appointed a Product Owner who is able to prioritize features and make decisions.

WHY? The Team can only assume a responsibility of the project if it is clear for everyone what is expected of them. Having the roles and responsibilities clearly agreed helps the people better understand the scope of their work, which makes it easier to focus on **creating impact**. It is also easier to identify staffing needs when roles are made explicit instead of implicit.

The availability of proper ownership from the Client's side can make or break the project. Being in frequent contact with a PO enables the team to concentrate on the right things, gives direction and motivates people.

The PO also makes communication more efficient by bridging the Team with different facets of the Client organization.

Does everyone know where project tasks are tracked and how progress is updated?

WHAT? Everyone knows where to look for the current up-to-date progress of the project. It can be Trello, JIRA, a wall in your project war room, anything – as long as such a thing exists.

Everyone must know what the next & upcoming tasks are, and what is currently in progress. All this information must be found in a single place accessible for everyone. Everyone knows how the tracking process works and how they can update the status of their tasks so that others can follow the progress.

WHY? Good situational awareness is the key element of successful execution. If the task statuses in the project management tool don't match the reality, it leads to false conclusions of the situation.

Are communication channels established between stakeholders?

WHAT? The Team and Client have agreed upon a primary communication channel, e.g. a chatroom where everyone involved in the project has access.
The Team is aware of, and has got the contact information of other relevant parties or stakeholders (prefer-

ably met them face-to-face) and is able to reach them on a short notice without the need for a proxy. This is especially important if there are people responsible for some part of the project delivery outside of the immediate reach of the Team.

WHY? Having a shared agreement on the communication channels is a prerequisite for efficient communication, makes things more transparent between parties, and promotes trust. Information is more likely to be shared and less likely to become siloed, and potential worries and problems are brought up earlier. Fewer and clearer communication channels helps people stay in sync.

Having met the people you are building the project with makes it easier to communicate textually. Knowing how to reach people even if you are not in contact on a daily basis pays off when quick actions need to be done.

Project Execution

Minimum frequency for asking these questions is **every week**.

Does everyone know possible hard deadlines & why they exist?

WHAT? Everyone has a shared understanding about deadlines that must be met, and knows what are the

actual **hard deadlines** (such as a big advertisement campaign for the product starting next month) and which are **soft deadlines** (such as the product owner having a holiday and it'd be nice to have the thing completed before that). What are the consequences of missing the deadline?

WHY? Hard deadlines affect how the product owner and the project team should prioritize tasks and raise flags early if need be. It helps to know what things are essential for a successful execution.

Does everyone know how much time & budget is left?

WHAT? Everyone knows and has access to information on

1. How much budget there was in the first place?
2. How much of it has already been spent and how much is left?

WHY? In order to plan your work efficiently and the level of detail where you can operate, you have to know at what point of the project you are currently at. The Client might get in trouble if the original budget is overrun unexpectedly.

When you know the available budget, you can more easily **prioritize** what will be done and what will be left out.

Does the Team have a reasoned estimate about the remaining amount of work?

WHAT? The Team knows what is required to be delivered for the next milestone, and they have made an estimate of the work required for meeting that milestone.

WHY? If the Team does not know how many tasks they have and how much work delivering them requires, it is impossible to track the progress towards that milestone.

If you don't know how much work is left, you can't efficiently **say no** to new requests made by the Client since you don't know how much flexibility you have available on your delivery capability.

It is common for developers to be optimistic, but if as a team you don't have an estimate at all about the remaining work to be done, it can be hard to realize that you actually **might already be in trouble** and only going deeper. Quite often the Client would happily take no for an answer if that is the reality.

Does the Team continuously write, update and run tests?

WHAT?

1. Tests are written for new features.
2. Tests are continuously run – preferably by automation.
3. Tests that break on new feature development are retrofitted.

4. Changing the behaviour of core business logic does break existing tests.

WHY? Having tests in place reduces the risk of regression and enables effective collaboration. It also makes onboarding new users easier since they get feedback from the tests in case they break something that was working before they touched the code.

Hardly ever is there budget for writing tests afterwards, nor do the tests help the development at that point, so make appropriate testing part of your standard mode of operation.

Are the work contributions systematically peer reviewed?

WHAT? Designs & code are continuously peer reviewed before passing them forward. Everything made in the project goes through at least two pairs of eyes.

WHY? Peer reviewing helps spread the knowledge across the team and validates the quality of solutions. It increases the possibility of detecting issues early when it's still cheaper to address them.

Does the Team have a documented plan for the system architecture?

WHAT? The Team has a rough documented plan about the architecture of the solution. It is pre-planned, not something that just emerges during the implementation.

The Team is also aware of constraints and limiting factors set by technology choices. This applies equally to newly adopted technologies and libraries, and any legacy architecture already in place.

WHY? Changing the architecture afterwards is really expensive and burdensome. When you plan the architecture beforehand, you can iterate & validate it more easily and more cheaply. A well-thought-out architecture makes it easier to reason about solution ideas, much easier to understand how the system works, and should also make it easier and cheaper to implement new features in the late game.

As a rule of thumb, more parts means more things to maintain, fix, and patch. The less code you can make do with, the better.

Does all the produced code follow the same style conventions?

WHAT? When you look at the code written in the project you don't know which team member has written it. The coding style and the way to structure code is the same between the developers. When choosing the code style for the project, also look beyond the project itself and preferably pick a style that is framework, platform or language style and follow these standards.

WHY? It is easier to understand each others' code when the style is identical – you don't have to switch between "style contexts" when browsing the code. It is

faster to write new code when you **don't have to think** about how the code should look like, and it is easier to reason about the style in peer reviews – it's not my style or your style, it's just the well thought (project!) style of the code. With the right tools, you can conveniently automate this and make it a non-issue.

Does everyone have the same situation awareness?

WHAT? The Team is not hiding anything from the Client – and vice versa. All unpleasant surprises are shared between both parties.

It is not enough that you have a clear situation presented in the project management tool if everyone is not actively using the tool.

WHY? Keeping bad news under the radar harms the-project execution. You can't mitigate things you are not aware of. It is always easier to deliver a small piece of bad news early than it is to attempt to hide it only to see it surface later.

Continuously sharing the situational awareness between the Client and the project team creates **transparency** & an atmosphere where surprises can be flagged easily and escalated further when there is a need to do so.

Can a new team member get started without help from current project team?

WHAT? A new project member can jump on the team by looking at **written down instructions** on how to setup the working environment, preferably as a README.md on the “front page” of the project repository. Furthermore, the setup process should be automated to as near to a “one-click experience” as possible.

WHY? If these things have not been written down but are only in the minds of the current project team you have a **bus factor problem** – if you get too many casualties, hidden knowledge is lost forever, and onboarding new team members becomes painful and expensive.

This is not only a concern of the tech people in the Team: are the design assets & materials available in a designated place where a new designer can pick them up and start working?

Resuming work on the project after a quiet season is an order of magnitude easier with proper instructions in place.

Release and Delivery

Especially important in projects where we are delivering a pre-defined scope instead of working as “bandwidth” as part of a Client’s project team.

Have all the relevant technical information been documented and accessible to the Client?

WHAT? The Client has access to a **Minimum Viable Documentation** that includes:

- overall architecture description, what parts does the system/delivery consist of
- list of integrations and dependencies to other systems and third-party services
- hosting information: list of services including IP addresses and other technical details
- information on how backups and upgrades are handled
- other “good to know” type information

WHY? Having the basic information available in one place makes it possible to maintain and further develop the system. It prevents vendor lock-in and empowers the Client. Overall, the solution is more **robust and fault-tolerant** when potentially anyone can understand the big picture and plan changes to how the system works.

It is also easier for new people to join the project and gain an understanding of the whole. Being familiar with different aspects of the system makes it possible to reason about changes and prevents knowledge silos from forming.

Is it clear who is in charge of operating the system?

WHAT? There is a defined person/group of people who have the responsibility of **monitoring** the health & status of the system.

There is a contact point to these people if someone else notices something weird in the system.

WHY? If no-one is in charge of the system and it happens to go down, we would only have an ad-hoc "hopefully the sales person answers their phone" procedure for getting the system back online. Some valuable operation time in production might be lost and end users could get pissed.

Does someone get an automatic notification if something is wrong with the live system?

WHAT? There are automatic checks in place that notify the people responsible for monitoring the system if something unexpected happens.

WHY? Without automatic checks in place failures could **go unnoticed** for long periods of time and could cause low service utilization, lost ROI or angry end users who are unable to use the system.

Are all the running expenses handled by the Client?

WHAT? When active project development mode is over, all running expenses are handled directly by the owner of the service, the Client.

WHY? Not only might charges like these annually generate thousands of euros worth of expenses, but manually proxying them to another party to be paid also causes administrative overhead.

When credit cards expire (and they do), the credit card holder might not even be aware of it. To avoid any easily **avoidable downtime** in the services, the responsibility of the running costs should be where the direct interest for keeping the service up is – the Client.

SALES AND NEGOTIATION

- [ ] Have the proposal scope & estimates been validated by the experts in that field?
- [ ] Does everyone understand what are the expected Deliverable(s)?
- [ ] Has it been validated that the problem is actually worth solving?
- [ ] Does everyone understand the project contract model?
- [ ] Does everyone have a shared assumption of the recognizable major risks in the project?

BEGINNING OF A PROJECT

- [ ] Does everyone know why we are doing this project?
- [ ] Does everyone understand their role in the project?
- [ ] Does everyone know where project tasks are tracked and how progress is updated?
- [ ] Are communication channels established between stakeholders?

PROJECT EXECUTION

- [ ] Does everyone know possible hard deadlines & why they exist?
- [ ] Does everyone know how much time & budget is left?
- [ ] Does the Team have a reasoned estimate about the remaining amount of work?
- [ ] Does the Team continuously write, update and run tests?
- [ ] Are the work contributions systematically peer reviewed?
- [ ] Does the Team have a documented plan for the system architecture?
- [ ] Does all the produced code follow the same style conventions?
- [ ] Does everyone have the same situation awareness?
- [ ] Can a new team member get started without help from current project team?

RELEASE AND DELIVERY

- [ ] Have all the relevant technical information been documented and accessible to the Client?
- [ ] Is it clear who is in charge of operating the system?
- [ ] Does someone get an automatic notiication if something is wrong with the live system?
- [ ] Are all the running expenses handled by the Client?