Demystifying Complexity

A Guide to Running a Successful Guidewire Upgrade

by

ACINI

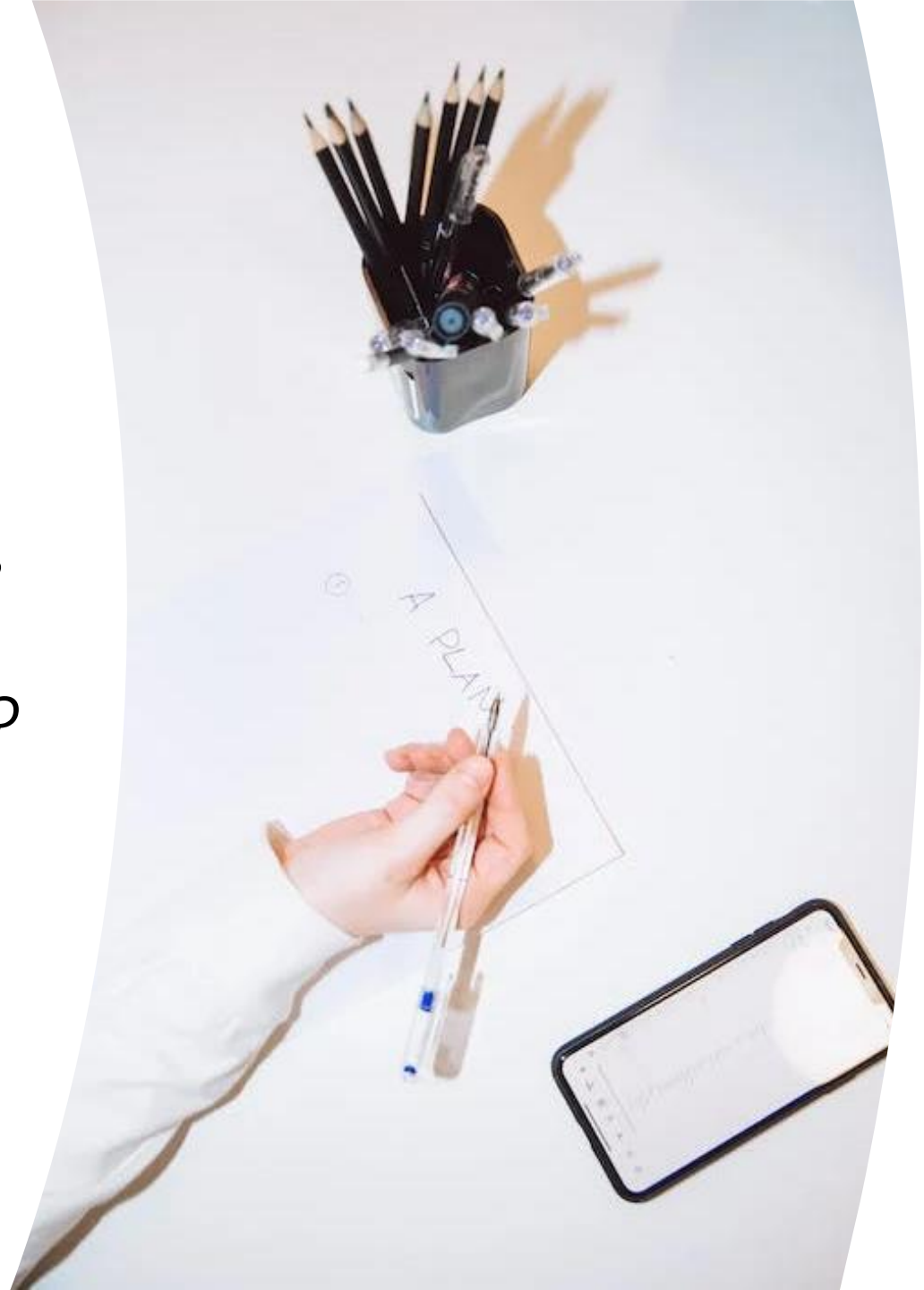
Welcome to Acini Guidewire Upgrade Guide!

This guide contains a collection of best practices gathered by our team, that has supported insurers around the world in their successful Guidewire upgrades.

Our crew members will **Guide** you on your **Guide**wire Upgrade adventure.

Sit back, relax and enjoy the journey!

Q: OK - where should I start ?



A: You should start by planning this project and placing it with a high enough priority in the roadmap of all the ongoing and planned projects in your organization

Q: How can I give high enough priority to a project that my business doesn’t like so much?

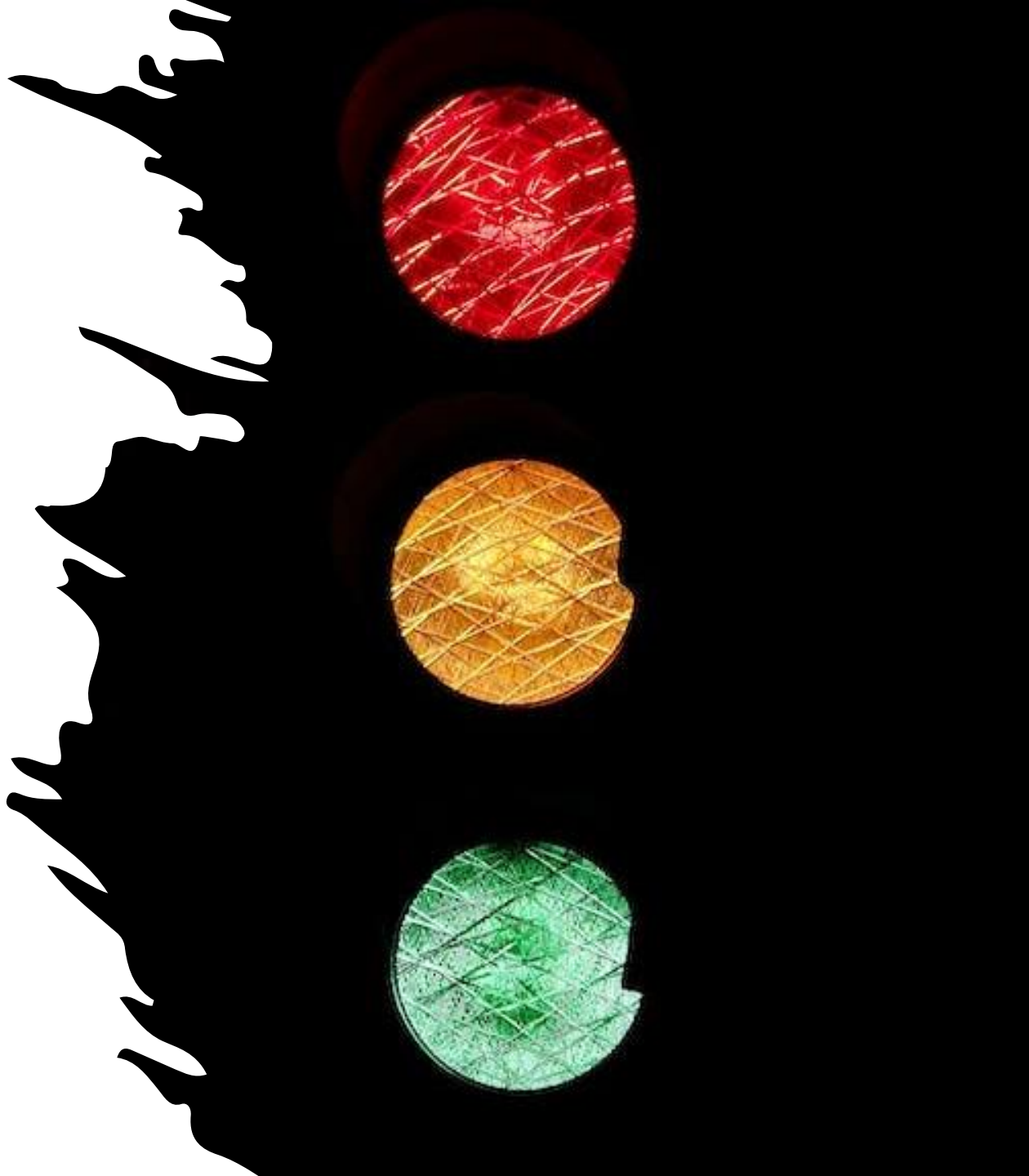
A: Why don’t they like it? Highlight the benefits they can achieve:

- Increased security
- End of lifecycle of software / hardware
- Improved performance
- New integration options
- Some new functionalities
- Decreased risk of reimplementation

Last but not least - you don't want your platform to become a modern legacy

Q: My organization is afraid that they will need to stop BaU – is that true?

A: That's not true – we will come back to this question later when explaining the project schedule. [(21)]

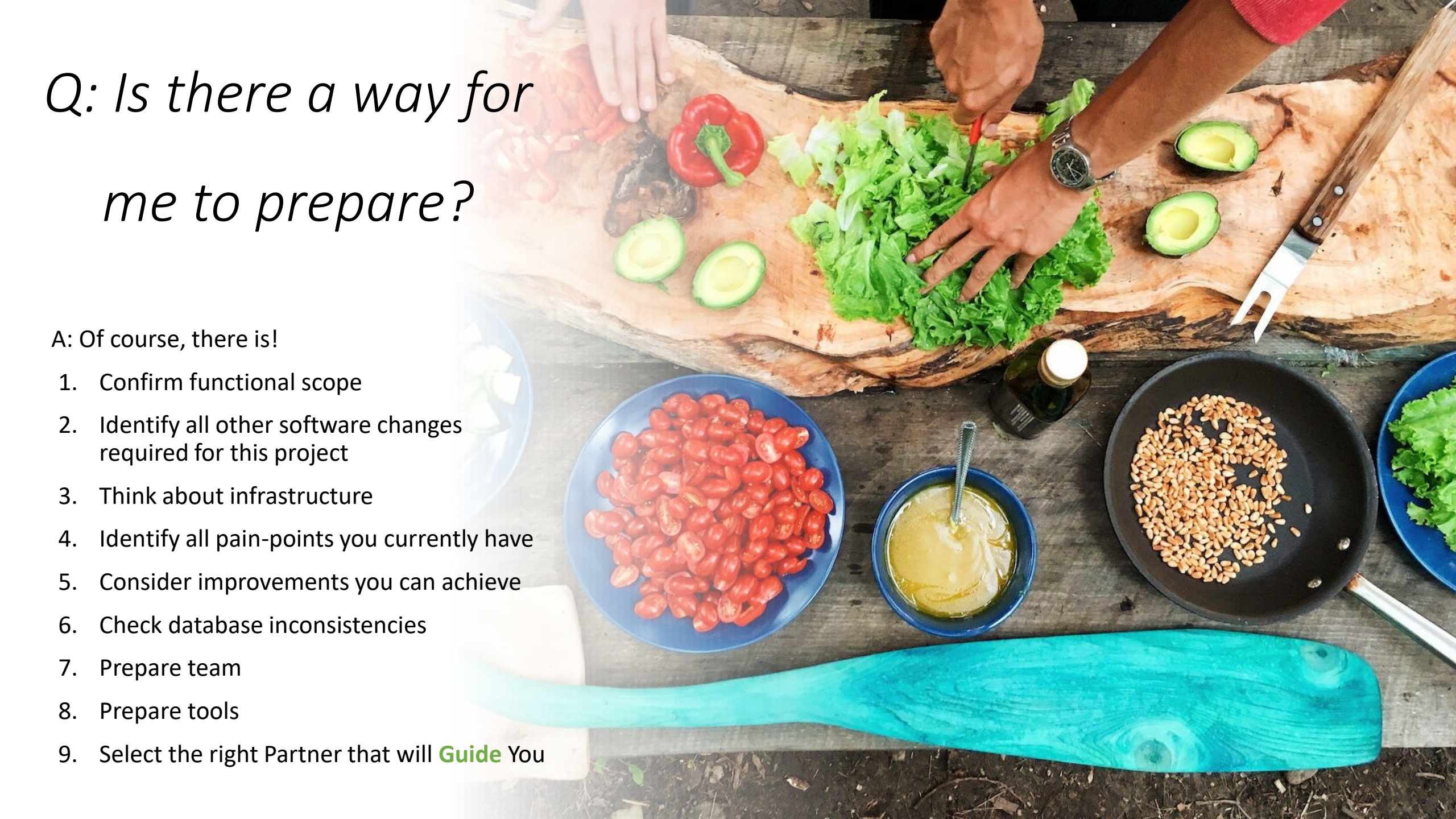


Q: Will it be a difficult project?

A: It will certainly be a different project than the initial implementationn.

- On one hand, it will be easier than the implementation because if you take the approach, we recommend you can avoid complicated discussions about the expected functionality
- On the other hand, even though you have a team of people experienced in working with the Guidewire platform, in all probability none of them has done such a project.

Q: Is there a way for me to prepare?



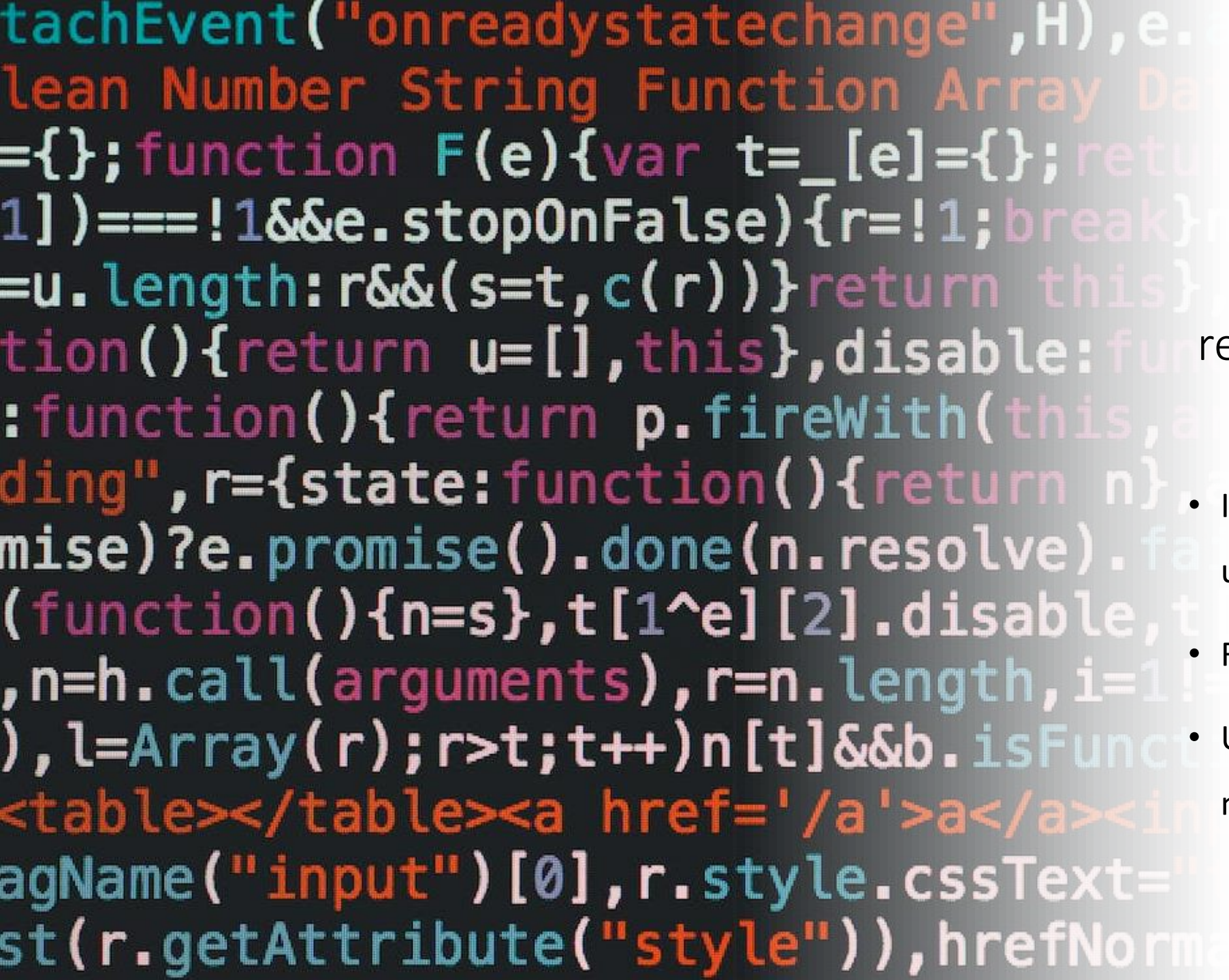
A: Of course, there is!

1. Confirm functional scope
2. Identify all other software changes required for this project
3. Think about infrastructure
4. Identify all pain-points you currently have
5. Consider improvements you can achieve
6. Check database inconsistencies
7. Prepare team
8. Prepare tools
9. Select the right Partner that will **Guide** You


Too much of this all at once -
can you explain step by step?

1. Confirm functional scope

- Take technical first approach [(19)]
- Review new functionalities
- Plan next phases for implementing the interesting ones



2. Identify all other software changes required for this project

- Integration layer may require upgrade
- Review the technical debt list
- Upgrade may generate / uncover new technical debt

3. Think about infrastructure

- You may consider moving to another Data Center
- How about moving to Cloud?
- You may consider moving to Guidewire Cloud to achieve even more benefits from the upgraded version (more details might be provided on request)
- It’s recommended to create new infrastructure
- If you jump between the major version then most likely you need to update other components, like
  - App server
  - DB Server
  - Java version
  - OS

4. Identify all pain-points you currently have

- If you’re already aware of some pain-ponts, consider improving / fixing them either before or during the upgrade, instead of moving identified pains to the new version
- Be transparent – don’t try to hide them and make improvements in meantime

5. Consider some improvements you can achieve

- CI/CD
- Automated tests
- Unit tests coverage
- Plan remediation of customized functionalities into OOTB ones

6. Check data inconsistencies

- It’s strongly recommended to run DB Consistency checks
- But it’s not required to fix all of them
- You should review / assess each one

7. Prepare the team

- Run trainings for them
- Familiarize them with the plan & methodology
- Invest in them – it might not be your last GW Upgrade project
- Build the Team!

8. Prepare tools

- Adapt the methodology
- Prepare the backlog tool
- Tools for remote work

9. Select the right Partner that will Guide You

- Project is done by people – not by brand
- Select people you trust
- Select people who have done it before
- Select people eager to train your team

Q: How should I organize the project itself

A: Each project is characterized by three aspects:

- Scope
- Time
- Budget

And this project is not different – these are high-level aspects that need to be planned carefully

Q: Building on the scope discussion, upgrading to the latest version with my customizations intact should be easy, correct?

A: Upgrading isn't straightforward. New versions bring enticing features, but integrating and adapting them can be complex. We suggest a two-phase approach:

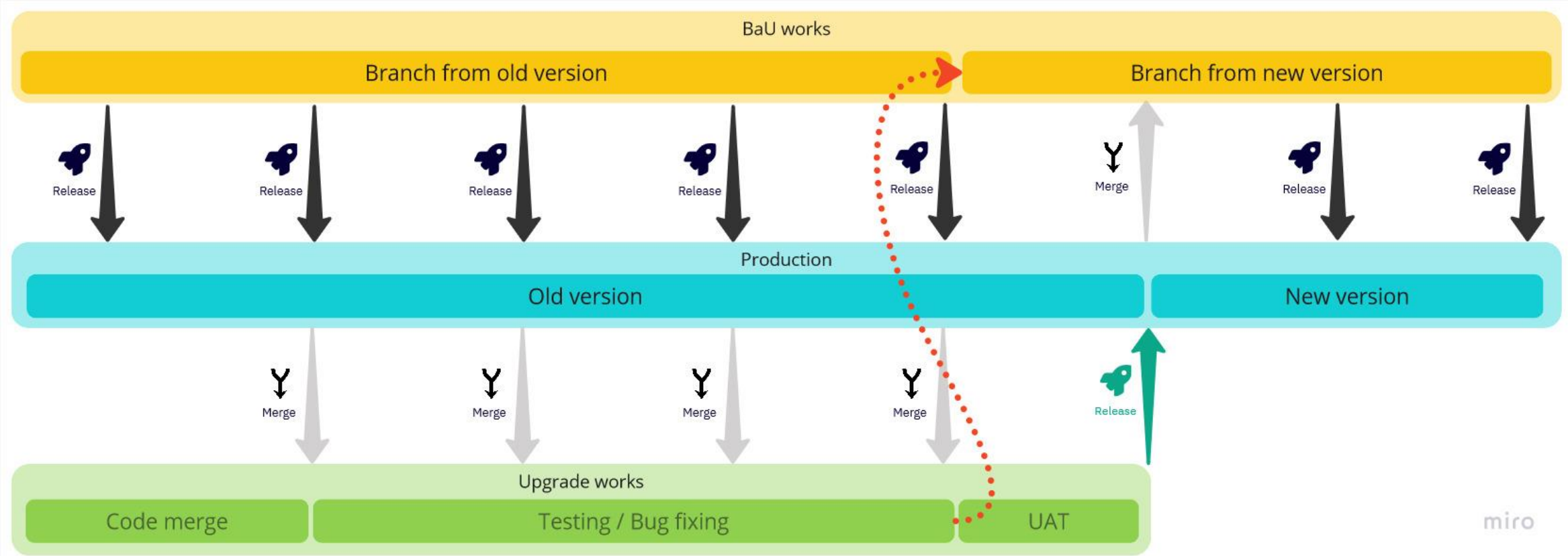
- Technical Upgrade
- Additional enhancements (functional and technical).

This reduces the immediate risk, allows you to have some benefits earlier and lets you explore new features at a manageable pace

Q: Can you explain the Technical Upgrade phase?

A: In this phase, we aim to seamlessly merge your current application version with the latest one available. We'll transfer all your customizations to the newest out-of-the-box (OOTB) version. Additionally, we may choose to disable certain new features to keep the focus elsewhere for now.

This phase can also be used for any necessary remediation or clean-up, especially for significant items in the technical debt, like migrating from a custom implementation of multicurrency functionalities that weren't present in the historical version to OOTB structures.

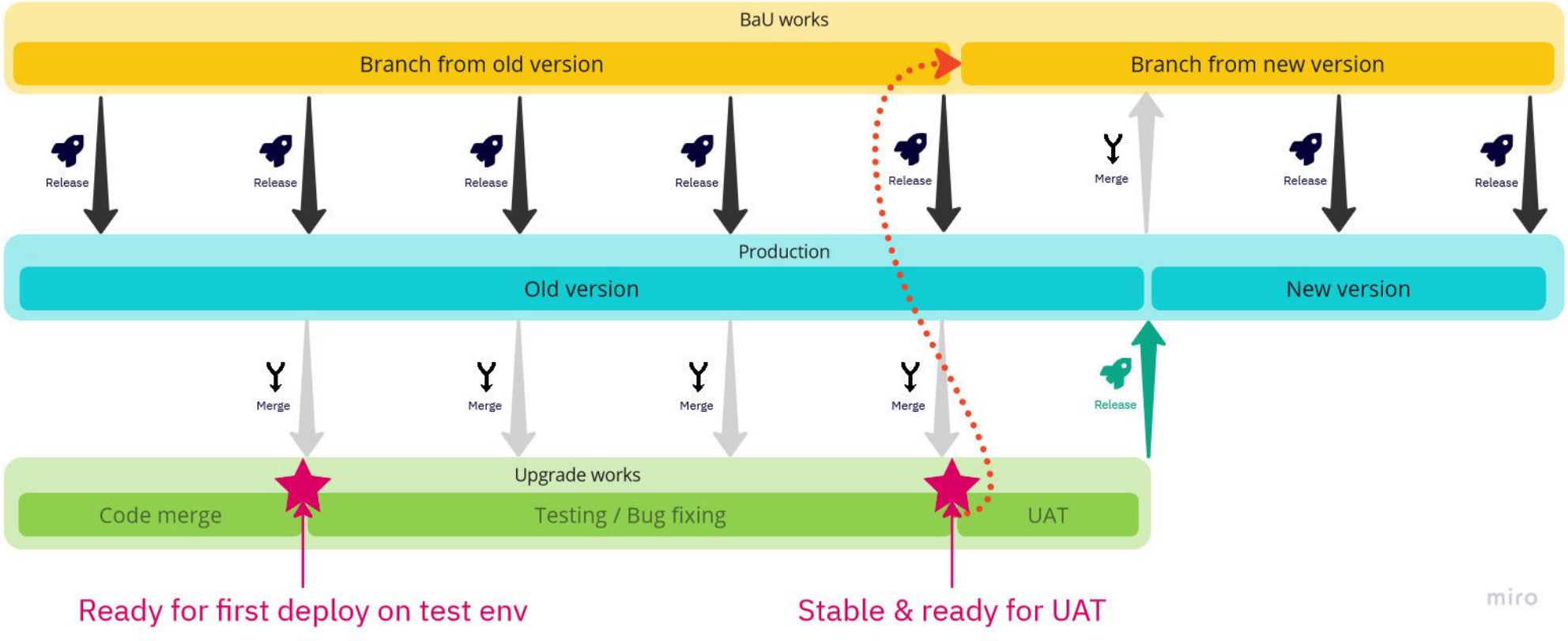


Q: Seems like a lot of technical work with my codebase. You mentioned earlier that I don't have to freeze ongoing BaU changes?

A: That's correct. We recommend keeping a regular sync between Upgrade and BaU branches.

- Initially, syncs will be from BaU to Upgrade.
- Once the upgraded version is stable for UAT tests, it becomes the candidate to go live.
- From then on, all BaU changes should be implemented on a new branch derived from the Upgrade branch.

Let’s take a look at the graphic above!



Q: So, we have one milestone for a stable version in the schedule. Can we set additional milestones?

A: Absolutely, we have another significant milestone before the stable version. This is when the project delivers an application ready for deployment to the initial environment.

While this version may still have some bugs, and certain functionalities might intentionally be reduced or switched off, it allows the testing team to start their work ahead of the stable version milestone.

Q: With two milestones we can divide the project into three stages, right?

A: We suggest breaking down the project into three main steps for the technical update

- **Code Merge** - This is where the application is launched for the first time, as explained on slide 22 and simultaneously the QA team should prepare all test cases to be executed for the 2nd stage
- **Testing / Bug Fixing** - This phase gets the application ready for acceptance, as detailed on slide 21
- **UAT** - This is the important phase for testing before full deployment.

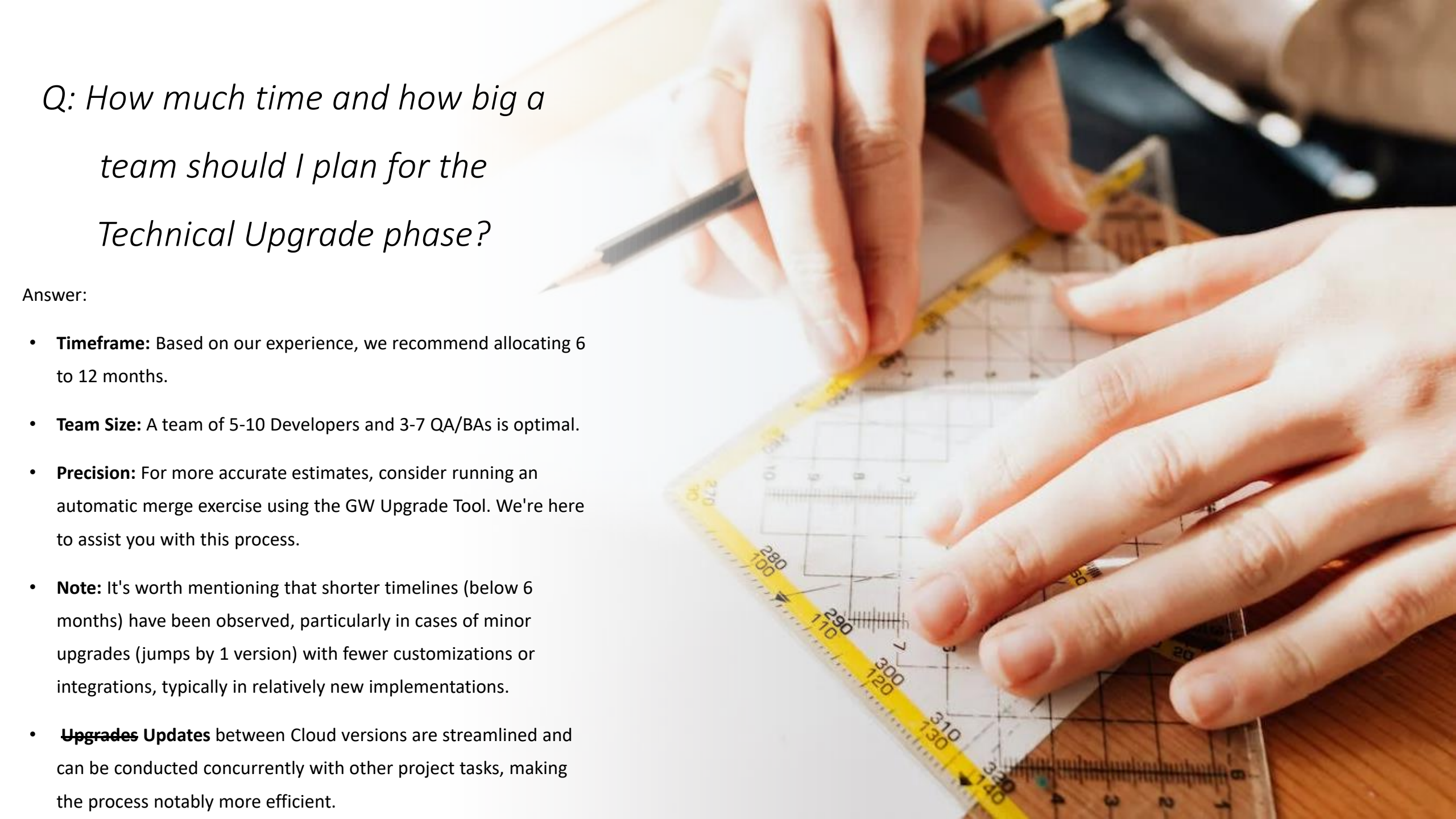
Following this, as mentioned on slide 19, you can consider a stage for additional improvements.

In addition, the overall project planning should be handled during the mobilization phase, discussed on slides 7 - 17

Q: How much time and how big a team should I plan for the Technical Upgrade phase?

Answer:

- **Timeframe:** Based on our experience, we recommend allocating 6 to 12 months.
- **Team Size:** A team of 5-10 Developers and 3-7 QA/BAs is optimal.
- **Precision:** For more accurate estimates, consider running an automatic merge exercise using the GW Upgrade Tool. We're here to assist you with this process.
- **Note:** It's worth mentioning that shorter timelines (below 6 months) have been observed, particularly in cases of minor upgrades (jumps by 1 version) with fewer customizations or integrations, typically in relatively new implementations.
- ~~**Upgrades**~~ **Updates** between Cloud versions are streamlined and can be conducted concurrently with other project tasks, making the process notably more efficient.



Q: Is there an alternative approach for this kind of a project?

A: While an alternative is reimplementation from scratch, it tends to be notably more complex and costly mainly due to the following reasons:

- **Intensive Functional and Architectural Discussions**: Reimplementation, much like the initial setup, requires extensive discussions about expected functionality and architectural choices
- **Data Migration Complexity:** It involves a significant effort in migrating existing data to the new instance, a process not needed in an upgrade scenario.

Q: Are there any pros of reimplementation?

A: Certainly, starting from scratch offers several advantages. Here are a few examples:

- **Cost Considerations:** It helps alleviate some of the increased costs mentioned on the previous slide. This approach allows for easier integration of new features available in the latest versions, such as APD or Autopilot.
- **Utilizing APD for Product Redesign:** This can lead to long-term benefits.
- **Optimizing the Current Implementation:** Valuable lessons learned from the initial implementation can be applied to enhance your current setup.
- **Technical Debt Cleanup:** It presents an opportunity to address any accumulated technical debt.

Q: What will be your recommendation for me, then?

- If your current implementation meets your organization's needs and you're generally satisfied, we suggest proceeding with the upgrade.
- However, if you have serious issues or are unsatisfied, you might consider making a business case for re-implementation. It's worth noting that this is not a common choice, so it should be reserved for cases where you're truly dissatisfied with the current setup.

Q: Are you the right Partner to help us?

ACINI

A: Absolutely, we’re the ideal Partner to help you!

- Our team is not only certified but also extensively experienced in Guidewire projects
- We've successfully supported insurers with highly customized implementations worldwide.
- While we may not be the largest company, we are big enough to run this kind of project and you'll be our top priority Customer, ensuring your project receives our utmost attention.
- Feel free to seek opinions from our satisfied customers to verify our track record.

* Picture presents part of Acini team

Q: I need more info

A: OK, please contact me!

https://www.linkedin.com/in/rafaldurasiewicz/

rafal@acini.pl

+48 696 466 665

https://calendly.com/acini-rafal/30min