**Ronan Incremental Delivery Benefits:**

1. **Multiple chances to deliver value to customers.**Customers prefer needs met today rather than month from now. This nicely aligns with your business desire to make revenue now too.
2. **Adapt through Learning.** Each time software is delivered to customers is another change to validate assumptions and find better opportunities than previously known.
3. **Prioritize Scope Better.**Feedback from actual users rather than theoretical future users helps build a better product with more focused investment.
4. **Less Delay Risk.** If you slip 10% on a 12 month project it has a much bigger delay impact than 10% on a 3 month project.
5. **Less Waste.** After each release you can stop. The max amount of waste is all the work done up until a release. A 12 month development cycle has 12 months of potential waste built-up until value is realized.
6. **Business Agility.** Not only can you limit waste but you can focus resources where the new business reality dictates.

**Natalia – why choosing incremental delivery lifecycle?**

1. **Requirements are well defined** – this will allow to divide the project into smaller phases which we can deliver in increments. From those smaller phases we will pick key increments that we will present to customer to gain valuable feedback and ensure needs are met
2. **Phased development** – since development is done sequentially, each increment is building on the previous one. This doesn’t mean that we must cross out parallel development. We can have two developers working on two different increments simultaneously. This will ensure deadline is met and if we are ahead of time, we will have more time for testing and quality control.
3. **Feedback** – incremental delivery ensures we will get customer feedback as the product will be presented to customer at the key phases. This approach allows us to make changes at earlier stage if needed. Even thought Incremental Delivery doesn’t promote frequent change of requirements, quicker customer feedback will reduce waste of resources by picking up on the issues sooner.
4. **Development Progress** – it is more visible how much has been done and what’s left to do as each increment is completed and delivered. This allows to give an idea where the team is at with the development, it also can be reassuring for customer.
5. **Testing** – testing is involved with each increment, which ensures high quality of software is delivered to customer.
6. **Product availability** – since work is delivered in increments, working software can be delivered quicker which can provide value to customer sooner. It surely won’t be a fully working product with first delivery, although some functionality can be available to customer (eg. signing up new newsagent users)
7. Incremental Delivery **supports changes to some extent**, allowing modifications to the project scope between increments when required to address evolving business needs. Increments are planned in advance, although if business require changes, these future increments can be adjusted accordingly, or priority can be reordered.

**Why not Agile?** Mainly because the requirements are well-defined, and we don’t want to involve customer constantly. Instead, we’d rather present what we have when it’s at key stages.

**Why not Waterfall?** Lack of flexibility – waterfall is highly sequential where each phase is dependant on the previous one. You cant have two developers working on separate phases as everything needs to happen one after another. Late Feedback – customer is involved at the end of the project which can lead into expectations not being met and can potentially make it very costly to revise/fix.

**Why not V-Model?** Mainly, I feel that the team is too small. It also would add extra complexity to the project that isn’t necessary. It’s very rigid and like Waterfall, doesn’t promote for potential change if the need arises.

Ivan

Waterfall:

Why?

Static requirements, ensures quality software, have design and planning phase(2). Not customer centric.

Why not?

Takes longer than other approaches to develop but makes sure it meets requirements and works for specified software , less flexible , more expensive to if requirements will eventually change.

Incremental delivery

Why?

Flexible, adaptable, delivering software in small chunks and testing (Agile) value in early stage.

Why not?

Customer centric, lack of organization, not complex

Incremental development:

V model

Why not?

Too much tests, long delivery time , requires bigger team .