DEMONSTRATIONS

Outcome

A demonstration is an opportunity to show working software primarily to a customer or a customer representative (e.g. a Product Owner) in order to gather feedback.

Function

Running demonstrations facilitates learning both for the developer and the customer, which results in higher quality products. The development team learn whether they have understood the requirement correctly and delivered what was wanted.

This last part is essential, because a demonstration helps the customer learn what they want. This includes:

- Things the customer asked for but when seeing decides are not right – "I know I asked for that but it's not what I want".
- Things the customer didn't know they wanted, sometimes known as 'emergent opportunities'.

Benefit

Demonstrations engage the customer more fully in the development process, giving an opportunity to see the effects of requirements and changes. It provides a better sense of control and greater involvement.

The faster the feedback, the faster the learning – this should make the process of design more efficient. The requirements, algorithms, and the code are still fresh in the developers mind; thus, if an issue is found, it is easier for them to understand what to change and estimate the amount of work that will be needed to address it. Also, when demonstrations are performed as soon as the features are developed, fewer extra features will have been added so the system will be easier to change.

Who

Scaling Factors

Difficulty

Implementation

Preparation

- Demonstrations should be kept short, no more than an hour. Otherwise, they become tiring for the attendees who need to understand what is demonstrated and be able to provide feedback.
- Run through what you want to demonstrate. For a longer, more complex, demonstration it may help to ensure you have an agenda to cover all the points you are concerned about. You may want to use the Acceptance Criteria as a basis.
- Make sure you can access the right environment. To keep the organisation and the cost of demonstrations to a minimum, they are usually performed from a development environment. Make sure you have tried the software to be demonstrated on the appropriate computer / bandwidth. Saying, 'well, it worked perfectly on my machine' is not a good excuse.
- Set up any relevant data (test data, anonymised production-like data...).
- Invite all the relevant people. Some demonstrations can be more about architecture decisions or very technical points and will require the feedback of an architect or a member of the support team for instance.
- If you are demonstrating to more than a few people and/or for a longer time, it may be worth grabbing a room. If your team is geographically distributed, get video-conference or screen sharing set up.

Run the demonstration

1. Have a very brief presentation of the elements (user stories, requirements...) that will be demonstrated and what type of data is going to be used (test data? production data?...).
2. Run the demonstration: requirements involving User Interface changes might be highly visual. More technical changes might require a more static demonstration using documents, log or message dumps.
3. Allow the viewers to provide feedback. Question them to be sure you fully understand the nuances. Watch body language – impatience, boredom or confusion - as well as articulated comments.
4. Write down all the comments and suggestions received.

Follow up

5. Define a list of actions from the comments and suggestions. Some of them may become new requirements or change requests. This is an important step. You must manage the expectations of the customers and product owners as to when they will be achieved.
6. Prioritise these actions with the team, including any business representative (e.g. Product Owner).
7. Keep the sources of the changes informed of the progress. This can be done by organising new demonstrations for the bigger changes.

Potential Pitfalls

- How you approach the demonstration. A demonstration is not a sales presentation to showcase how good your work is, nor a chance to celebrate what you've achieved. It is a working meeting to gather feedback – being told you need to make changes is a positive outcome.
- Demonstration length and frequency. A demonstration should take less than an hour and preferably be scheduled according to a regular cadence – this stops lots of wasted effort organising suitable diary time. If you have too much to fit into a single demonstration, it's best to organise two slots rather than rushing customer comments.
- Demonstrations are not a substitute for talking to the customer in the first place.
- Don't leave a demonstration too near to when you perform a release. Demoing early gives you more time to respond to feedback and lowers the cost of any required change.
- Don't assume you don't need a demonstration because the requirements look really clear or are really detailed. Remember, demonstrations elicit what customers don't yet know or can't articulate in advance. Demonstrations allow them to change ideas at a relatively low cost.
- Manage the expectations of the attendees. Not all changes proposed during a demonstration can be made immediately because of deadlines, costs or technical issues. The demonstrations should always involve the Project Manager/Team Leader to moderate the discussions and help decide whether a change will be performed or whether it will become a new requirement or change request.

If you want to learn more, consider reading:
Agile Software Requirements: Lean Requirements Practices for Teams, Programs, and the Enterprise by Dean Leffingwell
The Lean Start-up by Eric Ries