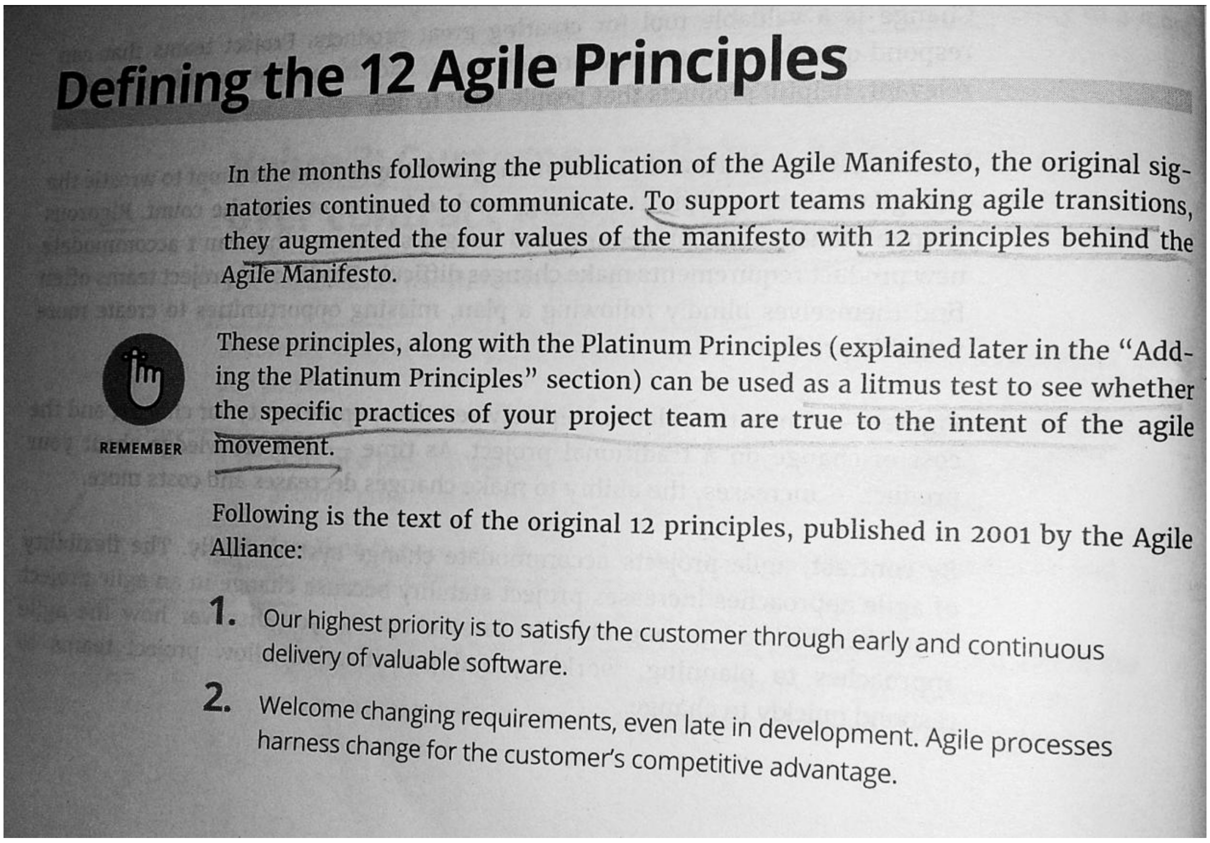


## Defining the 12 Agile Principles

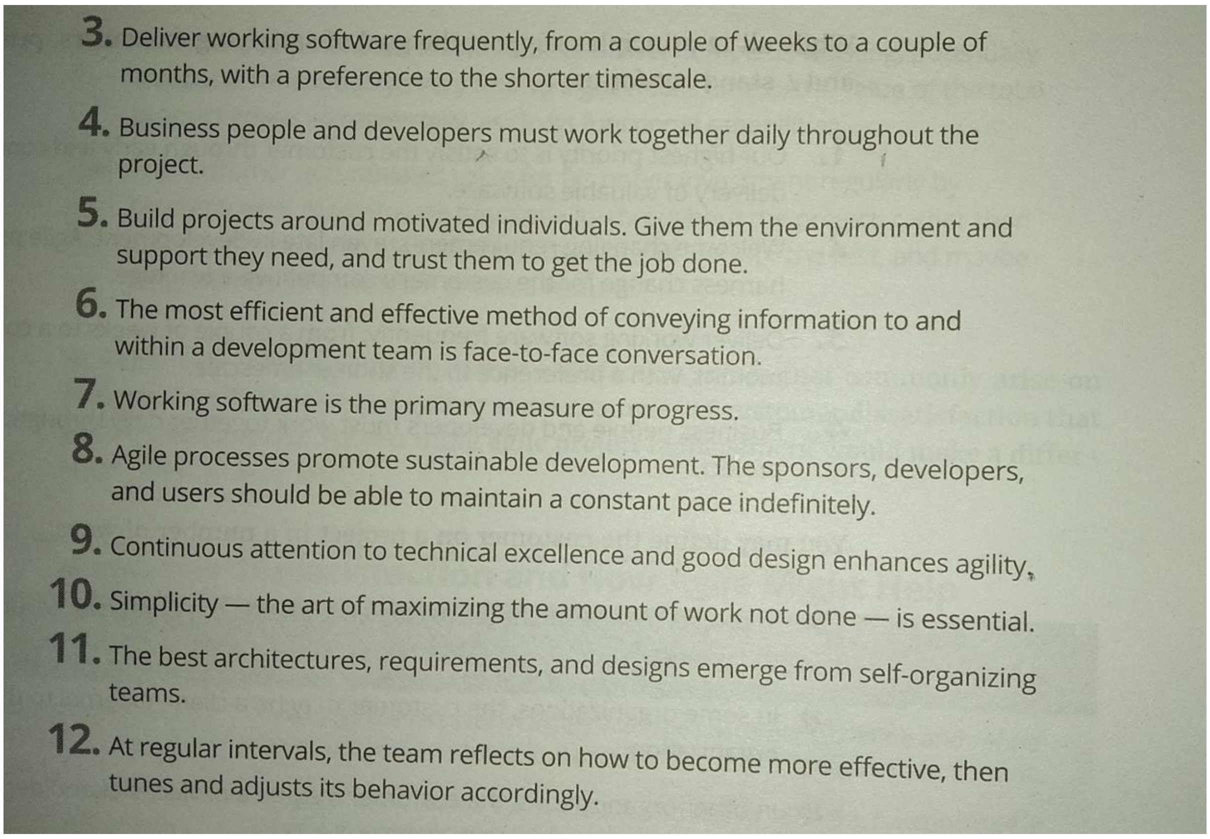
In the months following the publication of the Agile Manifesto, the original signatories continued to communicate. To support teams making agile transitions, they augmented the four values of the manifesto with 12 principles behind the Agile Manifesto.

REMEMBER

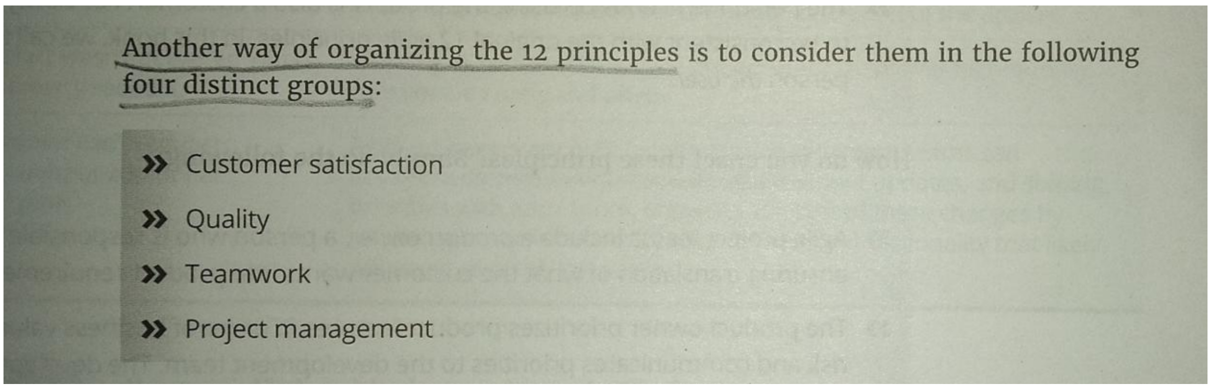
These principles, along with the Platinum Principles (explained later in the "Adding the Platinum Principles" section) can be used as a litmus test to see whether the specific practices of your project team are true to the intent of the agile movement.

Following is the text of the original 12 principles, published in 2001 by the Agile Alliance:

1. Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
2. Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.



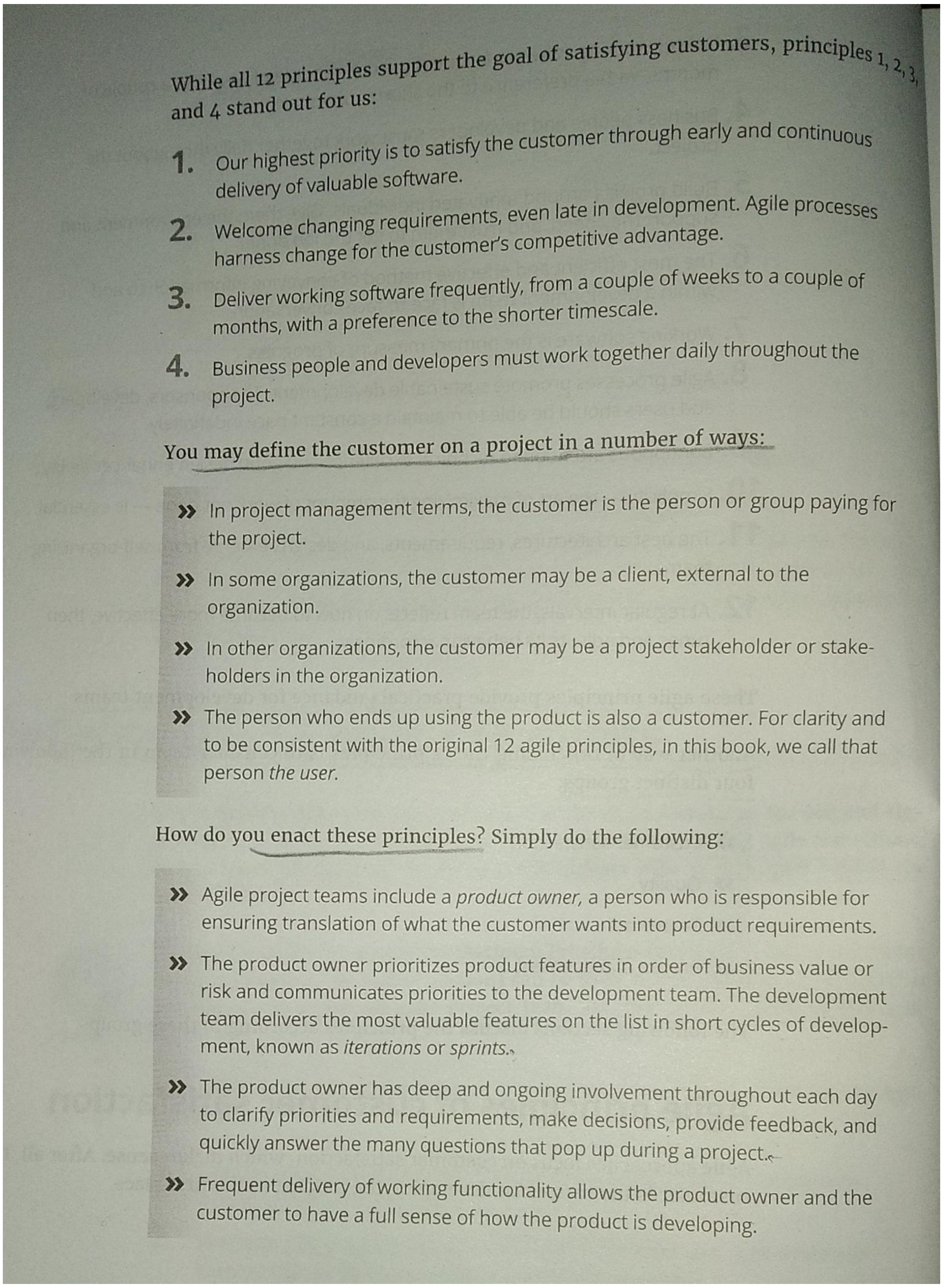
3. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
4. Business people and developers must work together daily throughout the project.
5. Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
6. The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
7. Working software is the primary measure of progress.
8. Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
9. Continuous attention to technical excellence and good design enhances agility.
10. Simplicity — the art of maximizing the amount of work not done — is essential.
11. The best architectures, requirements, and designs emerge from self-organizing teams.
12. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.



Another way of organizing the 12 principles is to consider them in the following four distinct groups:

- Customer satisfaction
- Quality
- Teamwork
- Project management

## Agile Principles of Customer Satisfaction



While all 12 principles support the goal of satisfying customers, principles 1, 2, 3, and 4 stand out for us:

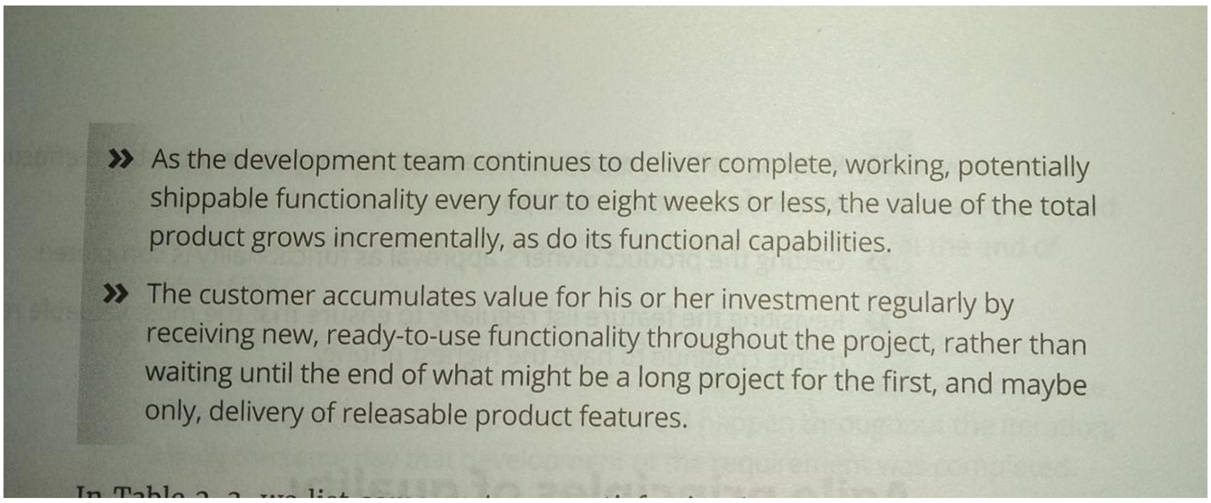
1. Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
2. Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
3. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
4. Business people and developers must work together daily throughout the project.

You may define the customer on a project in a number of ways:

- » In project management terms, the customer is the person or group paying for the project.
- » In some organizations, the customer may be a client, external to the organization.
- » In other organizations, the customer may be a project stakeholder or stakeholders in the organization.
- » The person who ends up using the product is also a customer. For clarity and to be consistent with the original 12 agile principles, in this book, we call that person *the user.*

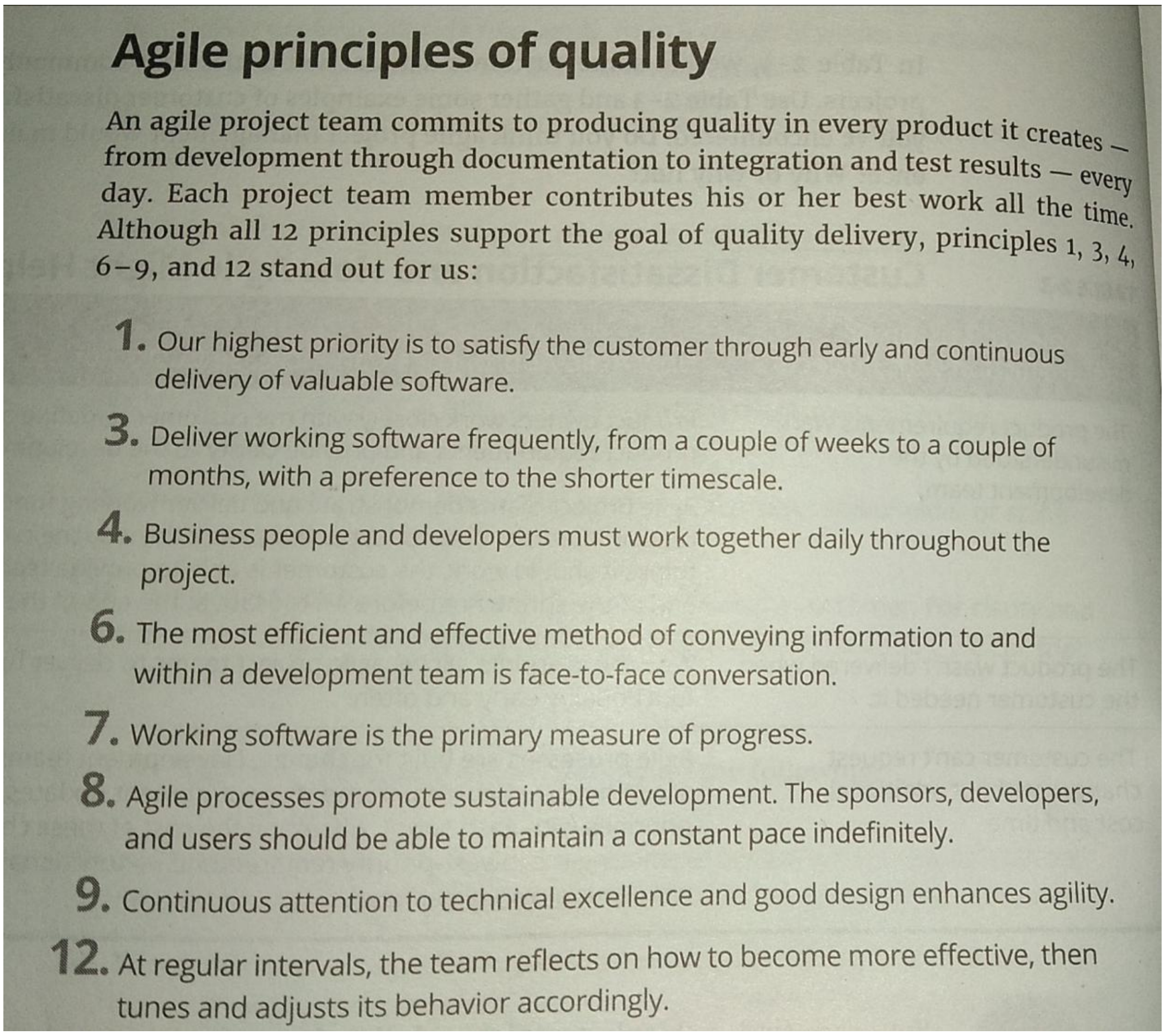
How do you enact these principles? Simply do the following:

- » Agile project teams include a *product owner,* a person who is responsible for ensuring translation of what the customer wants into product requirements.
- » The product owner prioritizes product features in order of business value or risk and communicates priorities to the development team. The development team delivers the most valuable features on the list in short cycles of development, known as *iterations* or *sprints.*
- » The product owner has deep and ongoing involvement throughout each day to clarify priorities and requirements, make decisions, provide feedback, and quickly answer the many questions that pop up during a project.
- » Frequent delivery of working functionality allows the product owner and the customer to have a full sense of how the product is developing.



- As the development team continues to deliver complete, working, potentially shippable functionality every four to eight weeks or less, the value of the total product grows incrementally, as do its functional capabilities.
- The customer accumulates value for his or her investment regularly by receiving new, ready-to-use functionality throughout the project, rather than waiting until the end of what might be a long project for the first, and maybe only, delivery of releasable product features.

In Table 2-3, we list

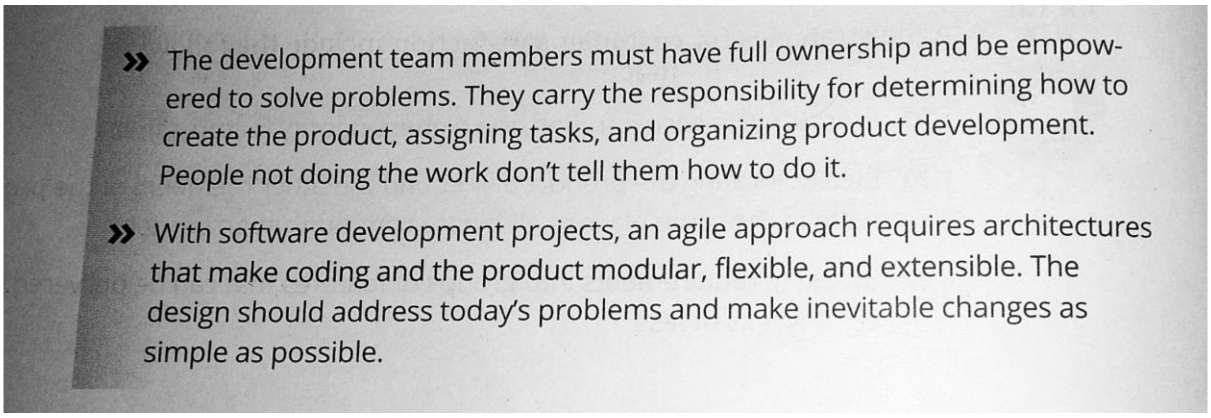


## Agile principles of quality

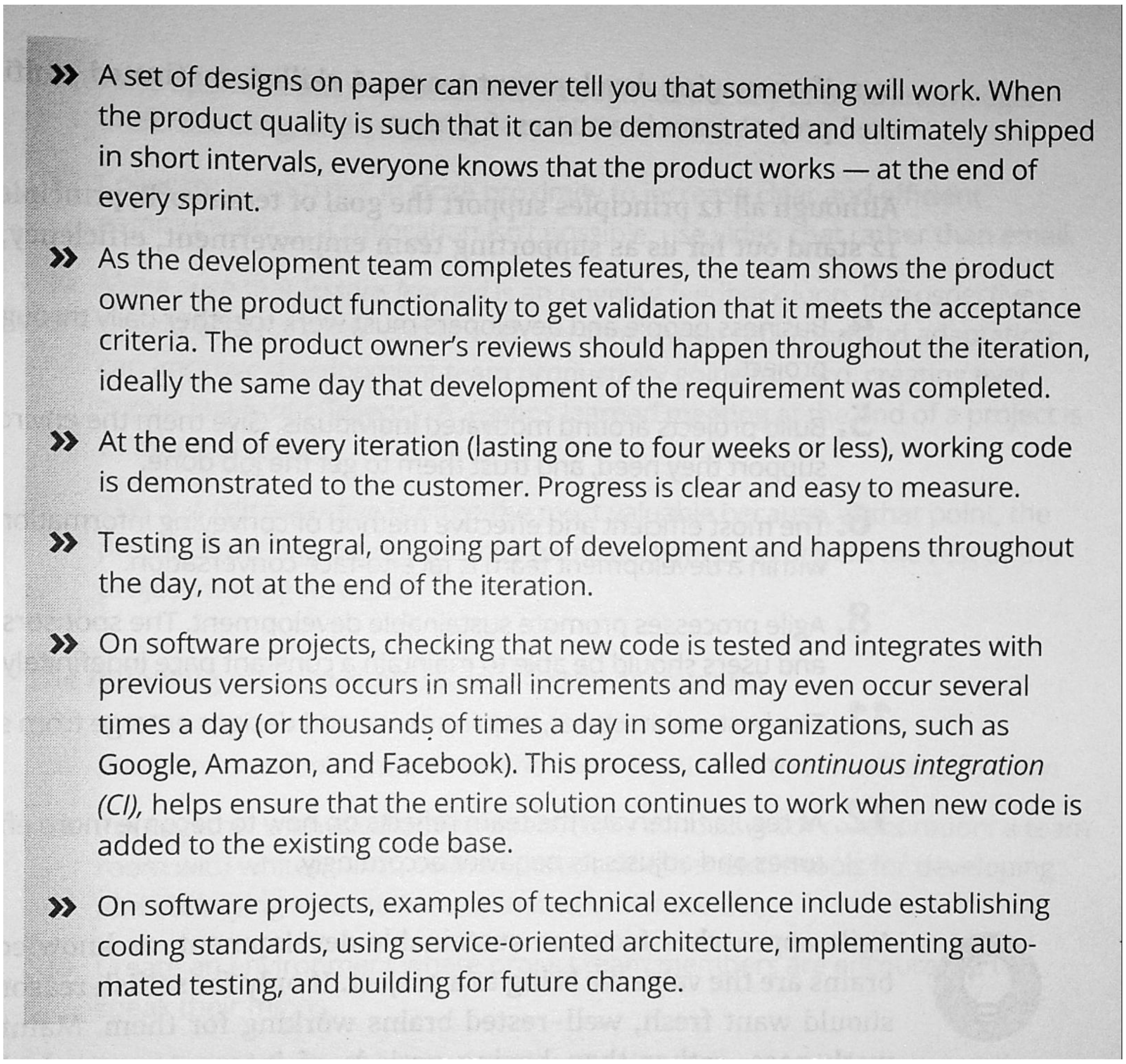
An agile project team commits to producing quality in every product it creates — from development through documentation to integration and test results — every day. Each project team member contributes his or her best work all the time. Although all 12 principles support the goal of quality delivery, principles 1, 3, 4, 6–9, and 12 stand out for us:

1. Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
3. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
4. Business people and developers must work together daily throughout the project.
6. The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
7. Working software is the primary measure of progress.
8. Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
9. Continuous attention to technical excellence and good design enhances agility.
12. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

**How To apply Principles for Quality on a Day to Day basis:**

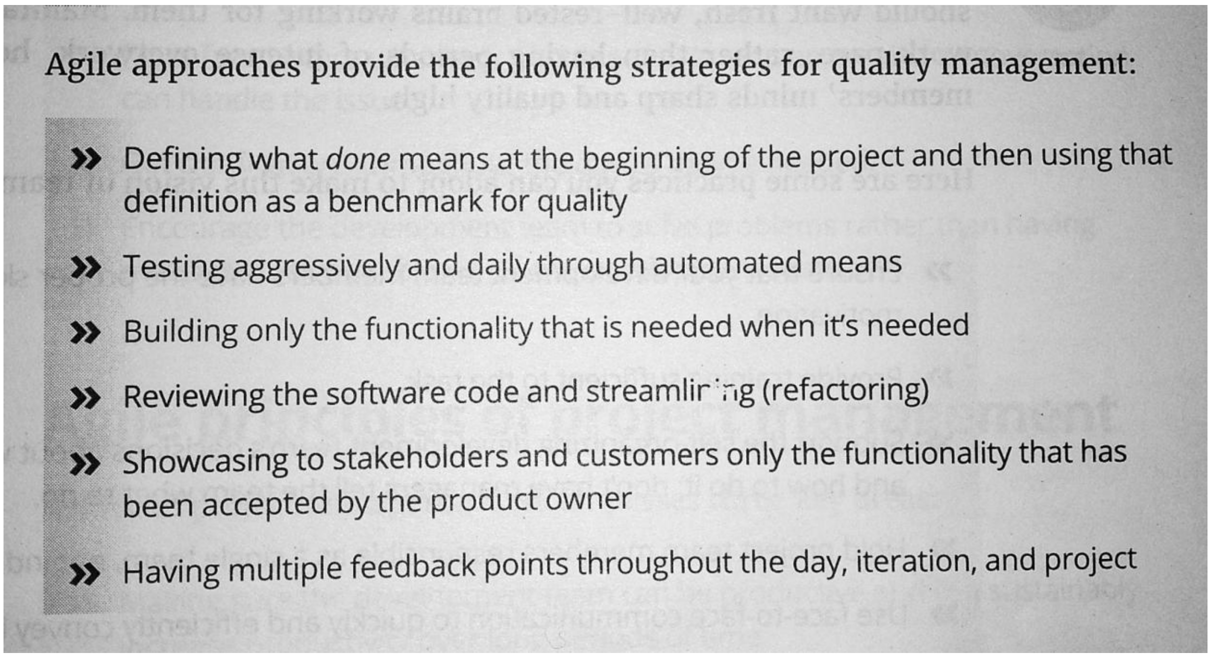


- The development team members must have full ownership and be empowered to solve problems. They carry the responsibility for determining how to create the product, assigning tasks, and organizing product development. People not doing the work don't tell them how to do it.
- With software development projects, an agile approach requires architectures that make coding and the product modular, flexible, and extensible. The design should address today's problems and make inevitable changes as simple as possible.



- A set of designs on paper can never tell you that something will work. When the product quality is such that it can be demonstrated and ultimately shipped in short intervals, everyone knows that the product works — at the end of every sprint.
- As the development team completes features, the team shows the product owner the product functionality to get validation that it meets the acceptance criteria. The product owner's reviews should happen throughout the iteration, ideally the same day that development of the requirement was completed.
- At the end of every iteration (lasting one to four weeks or less), working code is demonstrated to the customer. Progress is clear and easy to measure.
- Testing is an integral, ongoing part of development and happens throughout the day, not at the end of the iteration.
- On software projects, checking that new code is tested and integrates with previous versions occurs in small increments and may even occur several times a day (or thousands of times a day in some organizations, such as Google, Amazon, and Facebook). This process, called *continuous integration (CI),* helps ensure that the entire solution continues to work when new code is added to the existing code base.
- On software projects, examples of technical excellence include establishing coding standards, using service-oriented architecture, implementing automated testing, and building for future change.

## Strategies for Quality Management by Agile



Agile approaches provide the following strategies for quality management:

- Defining what *done* means at the beginning of the project and then using that definition as a benchmark for quality
- Testing aggressively and daily through automated means
- Building only the functionality that is needed when it's needed
- Reviewing the software code and streamlining (refactoring)
- Showcasing to stakeholders and customers only the functionality that has been accepted by the product owner
- Having multiple feedback points throughout the day, iteration, and project