

Chapter 11

DISCOVERING REQUIREMENTS

11.1 Introduction

11.2 What, How, and Why?

11.3 What Are Requirements?

11.4 Data Gathering for Requirements

11.5 Bringing Requirements to Life: Personas and Scenarios

11.6 Capturing Interaction with Use Cases

Objectives

The main goals of the chapter are to accomplish the following:

- Describe different kinds of requirements.
- Allow you to identify different kinds of requirements from a simple description.
- Explain additional data gathering techniques and how they may be used to discover requirements.
- Enable you to develop a persona and a scenario from a simple description.
- Describe use cases as a way to capture interaction in detail.

11.1 Introduction

Discovering requirements focuses on exploring the problem space and defining what will be developed. In the case of interaction design, this includes: understanding the target users and their capabilities; how a new product might support users in their daily lives; users' current tasks, goals, and contexts; constraints on the product's performance; and so on. This understanding forms the basis of the product's requirements and underpins design and construction.

It may seem artificial to distinguish between requirements, design, and evaluation activities because they are so closely related, especially in an iterative development cycle like the one used for interaction design. In practice, they are all intertwined, with some design taking place while requirements are being discovered and the design evolving through a series of evaluation—redesign cycles. With short, iterative development cycles, it's easy to confuse the

purpose of different activities. However, each of them has a different emphasis and specific goals, and each of them is necessary to produce a quality product.

This chapter describes the requirements activity in more detail, and it introduces some techniques specifically used to explore the problem space, define what to build, and characterize the target audience.

11.2 What, How, and Why?

This section briefly considers what is the purpose of the requirements activity, how to capture requirements, and why bother at all.

11.2.1 What Is the Purpose of the Requirements Activity?

The *requirements activity* sits in the first two phases of the double diamond of design, introduced in Chapter 2, “The Process of Interaction Design.” These two phases involve exploring the problem space to gain insights about the problem and establishing a description of what will be developed. The techniques described in this chapter support these activities, and they capture the outcomes in terms of requirements for the product plus any supporting artifacts.

Requirements may be discovered through targeted activities, or tangentially during product evaluation, prototyping, design, and construction. Along with the wider interaction design lifecycle, requirements discovery is iterative, and the iterative cycles ensure that the lessons learned from any of these activities feed into each other. In practice, requirements evolve and develop as the stakeholders interact with designs and learn what is possible and how features can be used. And, as shown in the interaction design lifecycle model in Chapter 2, the activity itself will be repeatedly revisited.

11.2.2 How to Capture Requirements Once They Are Discovered?

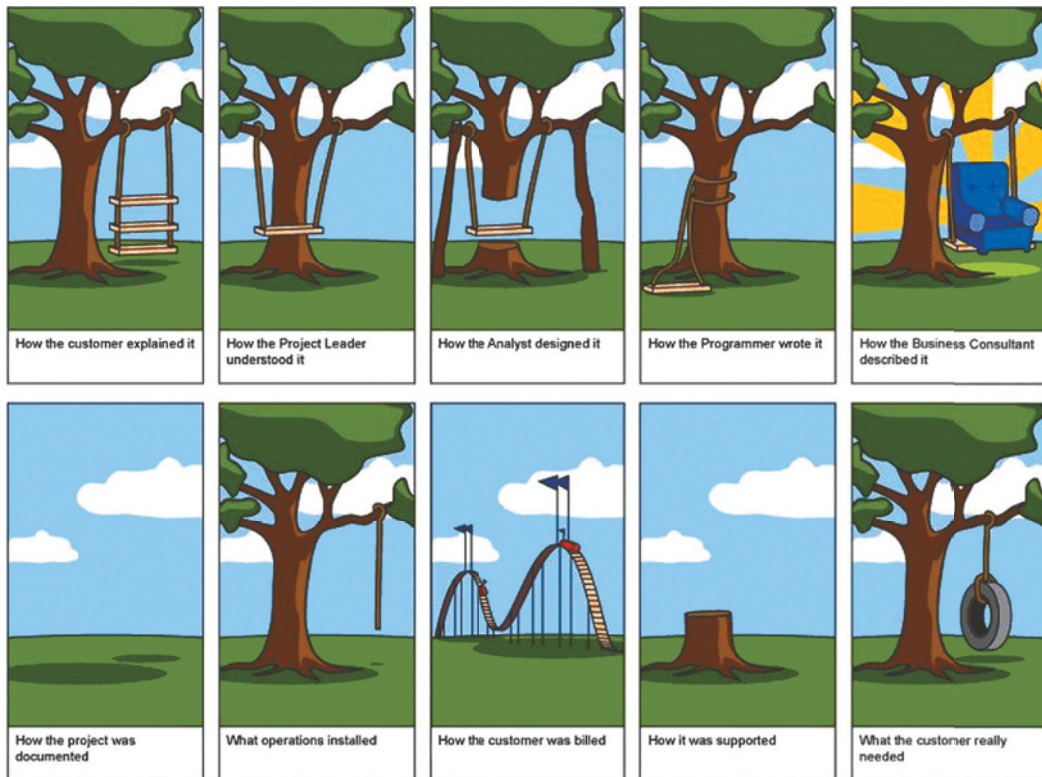
Requirements may be captured in several different forms. For some products, such as an exercise monitoring app, it may be appropriate to capture requirements implicitly through a prototype or operational product. For others, such as process control software in a factory, a more detailed understanding of the required behavior is needed before prototyping or construction begins, and a structured or rigorous notation may be used to investigate the product’s requirements. In all cases, capturing requirements explicitly is beneficial in order to make sure that key requirements aren’t lost through the iterations. Interactive products span a wide range of domains with differing constraints and user expectations. Although it may be disappointing if a new app to alert shoppers about offers on their favorite purchases turns out to be unusable or slightly inaccurate, if the same happens to an air traffic control system, the consequences are far more significant and could threaten lives.

As we discuss in this section, there are different kinds of requirements, and each can be emphasized or de-emphasized by different notations because notations emphasize different characteristics. For example, requirements for a product that relies on processing large amounts of data will be captured using a notation that emphasizes data characteristics. This means that a range of representations is used including prototypes, stories, diagrams, and photographs, as appropriate for the product under development.

11.2.3 Why Bother? Avoiding Miscommunication

One of the goals of interaction design is to produce usable products that support the way that people communicate and interact in their everyday and working lives. Discovering and communicating requirements helps to advance this goal, because defining what needs to be built supports technical developers and allows users to contribute more effectively. If the product turns out to be unusable or inappropriate, then everyone will be disappointed.

User-centered design with repeated iteration and evaluation along with user involvement mitigates against this from happening. The following cartoon illustrates the consequences of misunderstanding or miscommunication. The goal of an iterative user-centered approach is to involve different perspectives and make sure that there is agreement. Miscommunication is more likely if requirements are not clearly articulated.



11.3 What Are Requirements?

A *requirement* is a statement about an intended product that specifies what it is expected to do or how it will perform. For example, a requirement for a smartwatch GPS app might be that the time to load a map is less than half a second. Another, less precise requirement might be for teenagers to find the smartwatch appealing. In the latter example, the requirements

activity would involve exploring in more detail exactly what would make such a watch appealing to teenagers.

One of the goals of the requirements activity is to identify, clarify, and capture the requirements. The process of discovering requirements is iterative, allowing requirements and their understanding to evolve. In addition to capturing the requirements themselves, this activity also involves specifying criteria that can be used to show when the requirements have been fulfilled. For example, usability and user experience criteria can be used in this way.

Requirements come in different forms and at different levels of abstraction. The example requirement shown in Figure 11.1(a) is expressed using a generic requirements structure called an *atomic requirements shell* (Robertson and Robertson, 2013); Figure 11.1(b) describes the shell and its fields. Note the inclusion of a “fit criterion,” which can be used to assess when the solution meets the requirement, and also note the indications of “customer satisfaction,” “dissatisfaction,” and “priority.” This shell indicates the information about a requirement that needs to be identified in order to understand it. The shell is from a range of resources, collectively called *Volere* (<http://www.volere.co.uk>), which is a generic requirements framework. Although not specifically designed for interaction design, Volere is widely used in many different domains and has been extended to include UX analytics (Porter et al, 2014).

An alternative way to capture what a product is intended to do is via user stories. *User stories* communicate requirements between team members. Each one represents a unit of customer-visible functionality and serves as a starting point for a conversation to extend and clarify requirements. User stories may also be used to capture usability and user experience goals. Originally, user stories were normally written on physical cards that deliberately constrained the amount of information that could be captured in order to prompt conversations between stakeholders. While these conversations are still highly valued, the use of digital support tools such as Jira (<https://www.atlassian.com/software/jira>) has meant that additional information to elaborate the requirement is often stored with user stories. As an example, this additional information might be detailed diagrams or screenshots.

A user story represents a small chunk of value that can be delivered during a sprint (a short timebox of development activity, often about two weeks’ long), and a common and simple structure for user stories is as follows:

- As a <role>, I want <behavior> so that <benefit>.

Example user stories for a travel organizer might be:

- As a <traveler>, I want <to save my favorite airline for all my flights> so that <I will be able to collect air miles>.
- As a <travel agent>, I want <my special discount rates to be displayed to me> so that <I can offer my clients competitive rates>.

User stories are most prevalent when using an agile approach to product development. User stories form the basis of planning for a sprint and are the building blocks from which the product is constructed. Once completed and ready for development, a story consists of a description, an estimate of the time it will take to develop, and an acceptance test that determines how to measure when the requirement has been fulfilled. It is common for a user story such as the earlier ones to be decomposed further into smaller stories, often called *tasks*.

Requirement #: 75	Requirement Type: 9	Event/use case #: 6
Description: The product shall issue an alert if a weather station fails to transmit readings.		
Rationale: Failure to transmit readings might indicate that the weather station is faulty and needs maintenance, and that the data used to predict freezing roads may be incomplete.		
Source: Road Engineers		
Fit Criterion: For each weather station the product shall communicate to the user when the recorded number of each type of reading per hour is not within the manufacturer's specified range of the expected number of readings per hour.		
Customer Satisfaction: 3	Customer Dissatisfaction: 5	Conflicts: None
Dependencies: None		
Supporting Materials: Specification of Rosa Weather Station		
History: Raised by GBS, 28 July		

Volere
Copyright © Atlantic Systems Guild

(a)

List of events / use cases that need this requirement

The type from the template

Requirement #: Unique id	Requirement Type:	Event/use case #'s:
Description: A one sentence statement of the intention of the requirement		
Rationale: A justification of the requirement		
Originator: The person who raised this requirement		
Fit Criterion: A measurement of the requirement such that it is possible to test if the solution matches the original requirement		
Customer Satisfaction:	Customer Dissatisfaction:	Conflicts:
Priority: A rating of the customer value	Other requirements that cannot be implemented if this one is	
Supporting Materials:	Pointer to documents that illustrate and explain this requirement	
History: Creation, changes, deletions, etc.		

Degree of stakeholder happiness if this requirement is successfully implemented.
Scale from 1 = uninterested to 5 = extremely pleased.

Measure of stakeholder unhappiness if this requirement is not part of the final product.
Scale from 1 = hardly matters to 5 = extremely displeased.

Volere
Copyright © Atlantic Systems Guild

(b)

Figure 11.1 (a) An example requirement expressed using an atomic requirements shell from Volere
(b) the structure of an atomic requirements shell

Source: Atlantic Systems Guild

During the early stages of development, requirements may emerge in the form of epics. An *epic* is a user story that may take weeks or months to implement. Epics will be broken down into smaller chunks of effort (user stories), before they are pulled into a sprint. Example epics for a travel organizer app might be the following:

- As a <group traveler>, I want <to choose from a range of potential vacations that suit the group's preferences> so that <the whole group can have a good time>.
- As a <group traveler>, I want <to know the visa restrictions for everyone in the group> so that <visas can be arranged for everyone in the group in plenty of time>.
- As a <group traveler>, I want <to know the vaccinations required to visit the chosen destination> so that <vaccinations can be arranged for everyone in the group in plenty of time>.
- As a <travel agent>, I want <up-to-date information displayed> so that <my clients receive accurate information>.

11.3.1 Different Kinds of Requirements

Requirements come from several sources: from the user community, from the business community, or as a result of the technology to be applied. Two different kinds of requirements have traditionally been identified: *functional requirements*, which describe what the product will do, and nonfunctional requirements, which describe the characteristics (sometimes called *constraints*) of the product. For example, a functional requirement for a new video game might be that it will be challenging for a range of user abilities. This requirement might then be decomposed into more specific requirements detailing the structure of challenges in the game, for instance, levels of mastery, hidden tips and tricks, magical objects, and so on. A *nonfunctional requirement* for this same game might be that it can run on a variety of platforms, such as the Microsoft Xbox, Sony PlayStation, and Nintendo Switch game systems. Interaction design involves understanding both functional and nonfunctional requirements.

There are many more different types of requirements, however. Suzanne and James Robertson (2013) suggest a comprehensive categorized set of requirements types (see Table 11.1), while Ellen Gottesdiener and Mary Gorman (2012) suggest seven product dimensions (see Figure 11.2).

Project Drivers	1. The Purpose of the Product
	2. The Stakeholders
Project Constraints	3. Mandated Constraints
	4. Naming Conventions and Terminology
	5. Relevant Facts and Assumptions
Functional Requirements	6. The Scope of the Work
	7. Business Data Model and Data Dictionary
	8. The Scope of the Product
	9. Functional Requirements

Copyright © 2019, John Wiley & Sons, Incorporated. All rights reserved.

Nonfunctional Requirements	10. Look and Feel Requirements
	11. Usability and Humanity Requirements
	12. Performance Requirements
	13. Operational and Environmental Requirements
	14. Maintainability and Support Requirements
	15. Security Requirements
	16. Cultural Requirements
Project Issues	17. Compliance Requirements
	18. Open Issues
	19. Off-the-Shelf Solutions
	20. New Problems
	21. Tasks
	22. Migration to the New Product
	23. Risks
	24. Costs
	25. User Documentation and Training
	26. Waiting Room
	27. Ideas for Solutions

Table 11.1 A comprehensive categorization of requirements types

Source: Atlantic Systems Guild, Volere Requirements Specification Template, Edition 18 (2017), <http://www.volere.co.uk/template.htm>

The 7 Product Dimensions


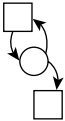




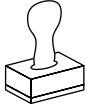
						
User	Interface	Action	Data	Control	Environment	Quality Attribute
Users interact with the product	The product connects to users, systems, and devices	The product provides capabilities for users	The product includes a repository of data and useful information	The product enforces constraints	The product conforms to physical properties and technology platforms	The product has certain properties that qualify its operation and development

Figure 11.2 The seven product dimensions

Source: Gottesdiener and Gorman (2012), p. 58. Used courtesy of Ellen Gottesdiener

To see how the seven product dimensions can be used to discover requirements, see <https://www.youtube.com/watch?v=x9olpZaXTDs>.

In this section, six of the most common types of requirements are discussed: functional, data, environment, user, usability, and user experience.

Functional requirements capture what the product will do. For example, a functional requirement for a robot working in a car assembly plant might be that it is able to place and weld together the correct pieces of metal accurately. Understanding the functional requirements for an interactive product is fundamental.

Data requirements capture the type, volatility, size/amount, persistence, accuracy, and value of the required data. All interactive products have to handle some data. For example, if an application for buying and selling stocks and shares is being developed, then the data must be up-to-date and accurate, and it is likely to change many times a day. In the personal banking domain, data must be accurate and persist over many months and probably years, and there will be plenty of it.

Environmental requirements, or context of use, refer to the circumstances in which the interactive product will operate. Four aspects of the environment lead to different types of requirements. First is the *physical environment*, such as how much lighting, noise, movement, and dust is expected in the operational environment. Will users need to wear protective clothing, such as large gloves or headgear that might affect the choice of interface type? How crowded is the environment? For example, an ATM operates in a very public physical environment, thus using a speech interface is likely to be problematic.

The second aspect of the environment is the *social environment*. Issues regarding the social aspects of interaction design, such as collaboration and coordination, were raised in Chapter 5, “Social Interaction.” For example, will data need to be shared? If so, does the sharing have to be synchronous (for instance, viewing the data at once) or asynchronous (for example, two people authoring a report taking turns to edit it)? Other factors include the physical location of fellow team members, such as collaborators communicating across great distances.

The third aspect is the *organizational environment*, for example, how good is user support likely to be, how easily can it be obtained, and are there facilities or resources for training, how efficient or stable is the communications infrastructure, and so on?

Finally, the *technical environment* will need to be established. For example, what technologies will the product run on or need to be compatible with, and what technological limitations might be relevant?

User characteristics capture the key attributes of the intended user group, such as the users’ abilities and skills, and depending on the product, also their educational background, preferences, personal circumstances, physical or mental disabilities, and so on. In addition, a user may be a novice, an expert, a casual user, or a frequent user. This affects the ways in which interaction is designed. For example, a novice user may prefer step-by-step guidance. An expert, on the other hand, may prefer a flexible interaction with more wide-ranging powers of control. The collection of characteristics for a typical user is called a *user profile*. Any one product may have several different user profiles.

Usability goals and *user experience goals* are another kind of requirement, and they should be captured together with appropriate measures. Chapter 2 briefly introduced

usability engineering, an approach in which specific measures for the usability goals of the product are agreed upon early in the development process and are used to track progress as development proceeds. This both ensures that usability is given due priority and facilitates progress tracking. The same is true for user experience goals. Although it is harder to identify quantifiable measures that track these qualities, an understanding of their importance is needed during the requirements activity.

Different interactive products will be associated with different requirements. For example, a telecare system designed to monitor an elderly person's movements and alert relevant care staff will be constrained by the type and size of sensors that can be easily worn by the users as they go about their normal activities. Wearable interfaces need to be light, small, fashionable, preferably hidden, and not get in the way. A desirable characteristic of both an online shopping site and a robotic companion is that they are trustworthy, but this attribute leads to different nonfunctional requirements—in the former, security of information would be a priority, while in the latter behavioral norms would indicate trustworthiness. A key requirement in many systems nowadays is that they be secure, but one of the challenges is to provide security that does not detract from the user experience. Box 11.1 introduces usable security.

BOX 11.1

Usable Security

Security is one requirement that most users and designers will agree is important, to some degree or another, for most products. The wide range of security breaches, in particular of individuals' private data, that have occurred in recent years has heightened everyone's awareness of the need to be secure. But what does this mean for interaction design, and how can security measures be suitably robust, while not detracting from the user experience? As long ago as 1999, Anne Adams and Angela Sasse (1999) discussed the need to investigate the usability of security mechanisms and to take a user-centered approach to security. This included informing users about how to choose a secure password, but it also highlighted that ignoring a user-centered perspective regarding security will result in users circumventing security mechanisms.

Many years later, usable security and the role of users in maintaining secure practices is still being discussed. Users are now bombarded with advice about how to choose a password, but most adults interact with so many systems and have to maintain a wide variety of login details and passwords that this can be overwhelming. Instead of improving security, this can lead to users developing coping strategies to manage their passwords, which may end up compromising rather than strengthening security. In their study, Elizabeth Stobert and Robert Biddle (2018) identify a password lifecycle that shows how passwords are developed, reused, adapted, discarded, and forgotten. Users are not necessarily ignoring password advice when they create weak passwords or write them down, but instead they are carefully managing their resources and expending more effort to protect the most valued accounts. Chapter 4, "Cognitive Aspects," highlighted issues around memory and passwords and the move toward using biometrics instead of passwords. The need to identify usable security requirements, however, will still exist even with biometrics. ■

ACTIVITY 11.1

Suggest some key requirements in each category (functional, data, environmental, user characteristics, usability goals, and user experience goals) for each of the following situations:

1. An interactive product for navigating around a shopping center.
2. A wearable interactive product to measure glucose levels for an individual with diabetes.

Comment

You may come up with alternative suggestions. These are merely indicative answers.

1. Interactive product for navigating around a shopping center.

Functional The product will locate places in the shopping center and provide routes for the user to reach their destination.

Data The product needs access to GPS location data for the user, maps of the shopping center, and locations of all the places in the center. It also requires knowledge about the terrain and pathways for people with different needs.

Environmental The product design needs to take into account several environmental aspects. Users may be in a rush, or they may be more relaxed and wandering about. The physical environment will be noisy and busy, and users may be talking with friends and colleagues while using the product. Support or help with using the product may not be readily available, but the user can probably ask a passerby for directions if the app fails to work.

User Characteristics Potential users are members of the population who have their own mobile device and for whom the center is accessible. This suggests quite a wide variety of users with different abilities and skills, a range of educational backgrounds and personal preferences, and different age groups.

Usability Goals The product needs to be easy to learn so that new users can use it immediately, and it should be memorable for more frequent users. Users won't want to wait around for the product to display fancy maps or provide unnecessary detail, so it needs to be efficient and safe to use; that is, it needs to be able to deal easily with user errors.

User Experience Goals Of the user experience goals listed in Chapter 1, "What Is Interaction Design?" those most likely to be relevant here are satisfying, helpful, and enhancing sociability. While some of the other goals may be appropriate, it is not essential for this product to, for example, be cognitively stimulating.

2. A wearable interactive product to measure glucose levels for an individual with diabetes.

Functional The product will be able to take small blood samples and measure glucose readings from them.

Data The product will need to measure and display the glucose reading—but possibly not store it permanently—and it may not need other data about the individual. These questions would be explored during the requirements activity.

Environmental The physical environment could be anywhere the individual may be—at home, in hospital, visiting the park, and so on. The product needs to be able to cope with a wide range of conditions and situations and to be suitable for wearing.

User Characteristics Users could be of any age, nationality, ability, and so forth, and may be novice or expert, depending on how long they have had diabetes. Most users will move rapidly from being a novice to becoming a regular user.

Usability Goals The product needs to exhibit all of the usability goals. You wouldn't want a medical product being anything other than effective, efficient, safe, easy to learn and remember how to use, and with good utility. For example, outputs from the product, especially any warning signals and displays, must be clear and unambiguous.

User Experience Goals User experience goals that are relevant here include the device being comfortable, while being aesthetically pleasing or enjoyable may help encourage continued use of the product. Making the product surprising, provocative, or challenging is to be avoided, however. ■

11.4 Data Gathering for Requirements

Data gathering for requirements covers a wide spectrum of issues, including who are the intended users, the activities in which they are currently engaged and their associated goals, the context in which the activities are performed, and the rationale for the current situation. The goals for the data gathering sessions will be to discover all of the types of requirements relevant for the product. The three data gathering techniques introduced in Chapter 8, “Data Gathering” (interviews, observation, and questionnaires), are commonly used throughout the interaction design lifecycle. In addition to these techniques, several other approaches are used to discover requirements.

For example, documentation, such as manuals, standards, or activity logs, are a good source of data about prescribed steps involved in an activity, any regulations governing a task, or where records of activity are already kept for audit or safety-related purposes. Studying documentation can also be good for gaining background information, and it doesn't involve stakeholder time. Researching other products can also help identify requirements. For example, Jens Bornschein and Gerhard Weber (2017) analyzed existing nonvisual drawing support packages to identify requirements for a digital drawing tool for blind users. Xiangping Chen et al. (2018) propose a recommender system for exploring existing app stores and extracting common user interface features to identify requirements for new systems.

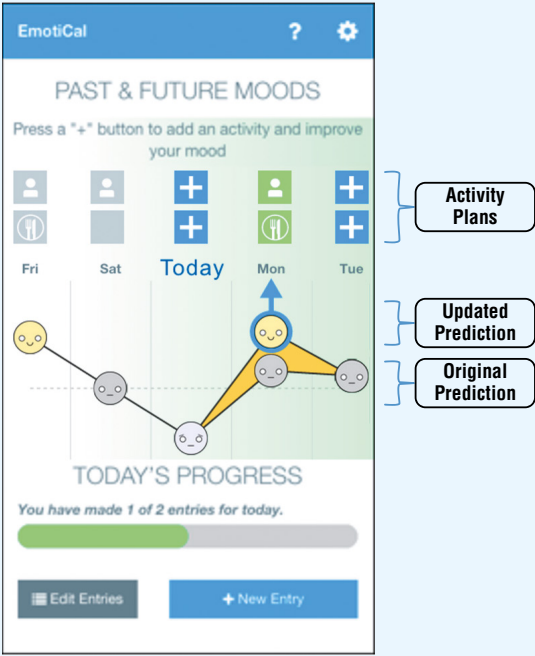
It is usual for more than one data gathering technique to be used in order to provide different perspectives. Examples are observation to understand the context of the activity, interviews to target specific user groups, questionnaires to reach a wider population, and focus groups to build a consensus view. Many different combinations are used in practice, and Box 11.2 includes some examples. Note that the example from Orit Shaer et al. (2012) also illustrates the development of an interactive product for a specialist domain, where users join the development team to help them understand the domain complexities.

BOX 11.2
Combining Data Gathering in Requirements Activities

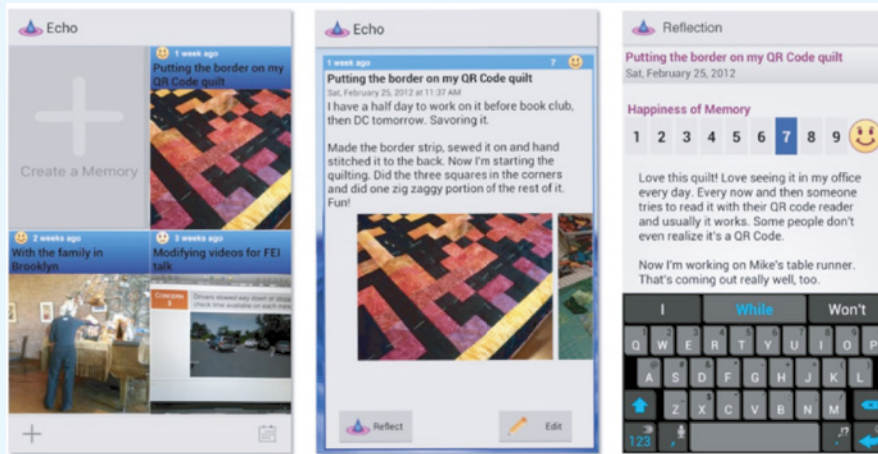
The following describes some examples where different data gathering techniques have been combined to progress requirements activities.

Direct Observation in the Field, Indirect Observation Through Log Files, Interviews, Diaries, and Surveys

Victoria Hollis et al. (2017) performed a study to inform the design of reflective systems that promote emotional well-being. Specifically, they wanted to explore the basis for future recommendations to improve a person’s well-being and the effects of reflecting on negative versus positive past events. They performed two direct observation studies in the field with 165 participants. In both studies, surveys were administered before and after the field study period to assess emotional well-being, behaviors, and self-awareness. The first study also performed interviews. In the first study (60 participants, over 3 weeks), they investigated the relationship between past mood data, emotional profiles, and different types of recommendations to improve future well-being. In the second study (105 participants, over 28 days), using a smartphone diary application, they investigated the effects of reflection and analysis of past negative and positive events on well-being. Together, these studies provided insights into requirements for systems to support the promotion of emotional well-being. Figure 11.3 shows the visualization displayed to emotion-forecasting participants in week 3 of the first study. The leftmost two points in the line graph indicate average mood ratings on previous days, and the center point is the average rating for the immediate day. The two rightmost points indicate predicted mood for upcoming days.



(a)



(b)

Figure 11.3 (a) The image shown to participants in the first field study (b) the smartphone diary app for the second study

In Figure 11.3 (b) The left panel shows the home screen: Participants record a new experience by clicking the large plus sign (+) in the upper left. The center panel shows a completed event record, which consists of a header, textual entry, emotion rating, and image. The right panel shows participant reflection by rating their current emotional reaction to the initial record and providing a new textual reappraisal.

Source: (a) Hollis et al. (2017), Figure 1. Used courtesy of Taylor & Francis and (b) Hollis et al. (2017), Figure 9. Used courtesy of Taylor & Francis

Diaries and Interviews

Tero Jokela et al. (2015) studied how people currently combine multiple information devices in their everyday lives to inform the design of future interfaces, technologies, and applications that better support multidevice use. For the purpose of this study, an *information device* is any device that can be used to create or consume digital information, including personal computers, smartphones, tablets, televisions, game consoles, cameras, music players, navigation devices, and smartwatches. They collected diaries over a one-week period and interviews from 14 participants. The study indicates that requirements for the technical environment needed to improve the user experience of multiple devices, including being able to access any content with any device and improved reliability and performance for cloud storage.

Interview, Think-Aloud Evaluation of Wireframe Mock-Up, Questionnaire, and Evaluation of Working Prototype

Carole Chang et al. (2018) developed a memory aid application for traumatic brain injury (TBI) sufferers. They initially conducted interviews with 21 participants to explore memory impairments after TBI. From these, they identified common themes in the use of external memory aids. They also learned that TBI sufferers do not want just another reminder system but something that helps them to remember and hence can also train their memory, and that their technology requirements were for something simple, customizable, and discreet.

(Continued)

Studying Documentation, Evaluating Other Systems, User Observations, and Group Interviews Nicole Costa et al. (2017) describe their ethnographic study of the design team for the user interface of a ship's maneuvering system (called a *conning display*). The design team started by studying the accident and incident reports to identify requirements for things to avoid, such as mixing up turn-rate meter with rudder indicator. They used Nielsen's heuristics to evaluate other existing systems, and specifically how to represent the vessel on the display. Once a suitable set of requirements had been discovered, sketching, prototyping, and evaluating with the help of users was used to produce the final design.

Ethnographic Study, Interviews, Usability Tests, and User Participation

Orit Shaer et al. (2012) report on the design of a multitouch tabletop user interface for collaborative exploration of genomic data. In-depth interviews were conducted with 38 molecular and computational biologists to understand the current work practices, needs, and workflow of small research groups. A small team of nine researchers investigating gene interaction in tuberculosis was studied for eight weeks using an ethnographic approach, and other labs were also observed. Because the application area was specialized, the design team needed to be comfortable with the domain concepts. To achieve this, biologists were integrated into the development team, and other members of the design team regularly visited biology research group partners, attended courses to teach them relevant domain concepts, and conducted frequent usability tests with users. ■

11.4.1 Using Probes to Engage with Users

Probes come in many forms and are an imaginative approach to data gathering. They are designed to prompt participants into action, specifically by interacting with the probe in some way, so that the researchers can learn more about users and their contexts. Probes rely on some form of logging to gather the data—either automatically in the case of technology probes or manually in the case of diaries or design probes.

The idea of a probe was developed during the Presence Project (Gaver et al., 1999), which was investigating novel interaction techniques to increase the presence of elderly people in their local community. They wanted to avoid more traditional approaches, such as questionnaires, interviews, or ethnographic studies, and developed a technique called *cultural probes*. These probes consisted of a wallet containing eight to ten postcards, about seven maps, a disposable camera, a photo album, and a media diary. Recipients were asked to answer questions associated with certain items in the wallet and then return them directly to the researchers. For example, on a map of the world, they were asked to mark places where they had been. Participants were also asked to use the camera to take photos of their home, what they were wearing today, the first person they saw that day, something desirable, and something boring.

Inspired by this original cultural probe idea, different forms of probes have been adapted and adopted for a range of purposes (Boehner et al., 2007). For example, *design probes* are objects whose form relates specifically to a particular question and context. They are intended to gently encourage users to engage with and answer the question in their own context. Figure 11.4 illustrates a Top Trumps probe; the participant was given six cards and asked to describe objects which were powerful to them and to rate the object's powers using numerical values out of 100 (Wallace et al., 2013).

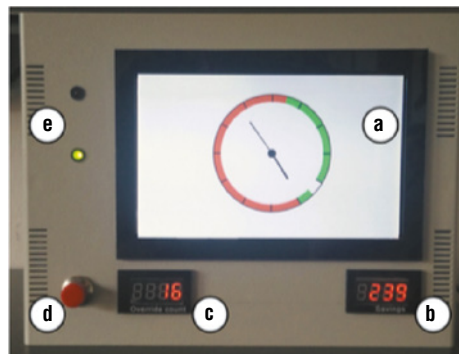


Figure 11.4 Top Trumps probe

Source: Wallace et al. (2013), Figure 6. Reproduced with permission of ACM Publications

Other types of probes include *technology probes* (Hutchinson et al., 2003) and *provocative probes* (Sethu-Jones et al., 2017). Examples of technology probes include tool-kits, such as the SenseBoard for developing IoT applications (Richards and Woodthorpe, 2009), mobile phone apps such as PocketSong, a mobile music listening app (Kirk et al., 2016), and M-Kulinda, a device that uses sensors to monitor movement that was deployed in rural Kenya (Chidziwisano and Wyche, 2018). The last of these, M-Kulinda, worked with the participant's mobile phones to alert participants of unexpected movement in their homes. By doing this, the researchers hoped to provide insights into how sensor-based technology may be used in rural households and to learn about domestic security measures in rural Kenya.

Provocative probes are technology probes designed to challenge existing norms and attitudes in order to provoke discussion. For example, Dimitros Raptis et al. (2017) designed a provocation called “The Box” to challenge domestic laundry practices. The intention was to learn about users' laundry practices and also to provoke users across three dimensions: conceptually, functionally, and aesthetically. Conceptual provocation challenged the assumption that electricity is always available and that its source is not relevant. Functional provocation was provided through an emergency override button that could be pressed if the electricity had been cut off, but its size and color implied that doing so was somehow wrong. Aesthetic provocation was achieved by designing a separate physical box, rather than a mobile phone app, and by designing it to be bulky and utilitarian (see Figure 11.5). This kind of provocation was found to increase participants' engagement.



a) electricity status – 12 hour forecast,
b) savings account, c) override button presses,
d) override button, and e) electricity status at the moment

Figure 11.5 The Box provocative probe

Source: Raptis et al. (2017). Reproduced with permission of ACM Publications

11.4.2 Contextual Inquiry

Contextual inquiry was originally developed in the 1990s (Holtzblatt and Jones, 1993) and has been adapted over time to suit different technologies and the different ways in which technology fits into daily life. Contextual inquiry is the core field research process for Contextual Design (Holtzblatt and Beyer, 2017), which is a user-centered design approach that explicitly defines how to gather, interpret, and model data about how people live in order to drive design ideation. However, contextual inquiry is also used on its own to discover requirements. For example, Hyunyoung Kim et al. (2018) used contextual inquiry to learn about unresolved usability problems related to devices for continuous parameter controls, such as the knobs and sliders used by sound engineers or aircraft pilots. From their study, they identified six needs: fast interaction, precise interaction, eyes-free interaction, mobile interaction, and retro-compatibility (the need to use their existing expertise with interfaces).

One-on-one field interviews (called *contextual interviews*) are undertaken by every member of the design team, each lasting about one-and-a-half to two hours. These interviews focus on matters of daily life (work and home) that are relevant for the project scope. Contextual inquiry uses a model of master/apprentice to structure data gathering, based on the idea that the interviewer (apprentice) is immersed in the world of the user (master), creating an attitude of sharing and learning on either side. This approach shifts the perceived “power” relationship that can exist in a more traditional interviewer–interviewee relationship. Users talk as they “do,” and the apprentice learns by being part of the activity while also observing it, which has all of the advantages of observation and ethnography. Hidden and specific details that people don’t make explicit, and don’t necessarily realize themselves, emerge this way and can be shared and learned. While observing and learning, the apprentice focuses on why, not just what.

Four principles guide the contextual interview, each of which defines an aspect of the interaction and enhances the basic *apprenticeship model*. These principles are context, partnership, interpretation, and focus.

The *context principle* emphasizes the importance of going to the user, wherever they are, and seeing what they do as they do it. The benefits of this are exposure to ongoing experience instead of summary data, concrete details rather than abstract data, and experienced motives rather than reports. The *partnership principle* creates a collaborative context in which the user and interviewer can explore the user’s life together, on an equal footing. In a traditional interview or workshop situation, the interviewer or workshop leader is in control, but in contextual inquiry, the spirit of partnership means that understanding is developed together.

Interpretation turns the observations into a form that can be the basis of a design hypothesis or idea. These interpretations are developed collaboratively by the user and the design team member to make sure that they are sound. For example, imagine that during a contextual interview for an exercise monitor, the user repeatedly checks the data, specifically looking at the heart rate display. One interpretation of this is that the user is very worried about their heart rate. Another interpretation is that the user is concerned that the device is not measuring the heart rate effectively. Yet another interpretation might be that the device has failed to upload data recently, and the user wants to make sure that the data is saved regularly. The only way to make sure that the chosen interpretation is correct is to ask the user and see their reaction. It may be that, in fact, they don’t realize that they are doing this and that it has simply become a distracting habit.

The fourth principle, *focus*, is established to guide the interview setup and tells the interviewer what they need to pay attention to among all of the detail that will be unearthed. While the apprenticeship model means that the master (user) will choose what to share or teach, it is also the apprentice's responsibility to capture information relevant to the project. In addition, the interviewer will have their own interests and perspectives, and this allows different aspects of the activity to surface when all members of the team conduct interviews around the project focus. This leads to a richer collection of data.

Together with the principles that shape how the session will run, the contextual interview is also guided by a set of “cool concepts.” *Cool concepts* are an addition to the original contextual inquiry idea, and they are derived from a field study that investigated what it is about technologies that users find “cool” (Holtzblatt and Beyer, 2017, p10). Seven cool concepts emerged from this study, and they are divided into two groups: four concepts that enhance the joy of life and three concepts that enhance the joy of use.

The *joy of life concepts* capture how products make our lives richer and more fulfilling. These concepts are *accomplish* (empower users), *connection* (enhance real relationships), *identity* (support users' sense of self), and *sensation* (pleasurable moments).

The *joy of use concepts* describes the impact of using the product itself; they are: *direct in action* (provide fulfillment of intent), *the hassle factor* (remove all glitches and inconveniences), and *the learning delta* (reduce the time to learn). During a contextual interview, cool concepts are identified as the user does their activity, although often they only emerge retrospectively when reflecting on the session.

The contextual interview has four parts: getting an overview, the transition, main interview, and wrap-up. The first part can be performed like a traditional interview, introducing each other and setting the context of the project. The second part is where the interaction changes as both parties get to know each other, and the nature of the contextual interview engagement is set up. The third part is the core data gathering session when the user continues with their activities and the interviewer observes them and learns. Finally, the wrap-up involves sharing some of the patterns and observations the interviewer has made.

During the interview, data is collected in the form of notes and initial Contextual Design models and perhaps audio and video recordings. Following each contextual interview, the team holds an interpretation session that allows the whole team to talk about the user and hence establish a shared understanding based on the data. During this session, specific contextual design models are also generated or consolidated. There are 10 models suggested by Contextual Design, and the team can choose the most relevant for the project. Five of these models are linked to the cool concepts: the *day-in-the-life model* (representing accomplishment), the *relationship* and *collaboration models* (representing connection), the *identity model*, and the *sensation board*. Five others provide a complete view of the users' tasks, but they are used only for some projects: the *flow model*, the *decision point model*, the *physical model*, the *sequence model*, and the *artifact model*. The affinity diagram, described in Chapter 2, is produced after several interpretation sessions have taken place. The contextual design method follows this up with an immersion exercise called the *Wall Walk*, in which all of the generated models are hung up on the walls of a large conference room for stakeholders to read and suggest design ideas. For more detail about these models and how to generate them, see Holtzblatt and Beyer (2017).

ACTIVITY 11.2

How does contextual inquiry compare with the data gathering techniques introduced in Chapter 8, specifically ethnography and interviews?

Comment

Ethnography involves observing a situation without any *a priori* structure or framework; it may include other data gathering techniques such as interviews. Contextual inquiry also involves observation with interviews, but it provides more structure, support, and guidance in the form of the apprenticeship model, the principles to which one must adhere, the cool concepts to look out for, and a set of models to shape and present the data. Contextual inquiry also explicitly states that it is a team effort, and that all members of the design team conduct contextual interviews.

Structured, unstructured, and semi-structured interviews were introduced in Chapter 8. Contextual inquiry could be viewed as a form of unstructured interview with props, but it has other characteristics as discussed earlier, which gives it added structure and focus. A contextual inquiry interview requires the interviewee to be going about their daily activities, which may also mean that the interview moves around—something very unusual for a standard interview. ■

11.4.3 Brainstorming for Innovation

Requirements may emerge directly from the data gathered, but they may also involve innovation. *Brainstorming* is a generic technique used to generate, refine, and develop ideas. It is widely used in interaction design specifically for generating alternative designs or for suggesting new and better ideas to support users.

Various rules have been suggested for making a brainstorming session successful, some of which are listed next. In the context of the requirements activity, two key success factors are that the participants know the users and that no ideas are criticized or debated. Other suggestions for successful requirements brainstorming sessions are as follows (Robertson and Robertson, 2013; Kelley with Littman, 2016):

1. *Include participants from a wide range of disciplines with a broad range of experience.*
2. *Don't ban silly stuff.* Wild ideas often turn into really useful requirements.
3. *Use catalysts for further inspiration.* Build one idea on top of another, jump back to an earlier idea, or consider alternative interpretations when energy levels start to flag. If you get stuck, use a word pulled randomly from a dictionary to prompt ideas related to the product.
4. *Keep records.* Capture every idea, without censoring. One suggestion is to number ideas so that they can be referred to more easily at a later stage. Cover the walls and tables in paper, and encourage participants to sketch, mind-map, and diagram ideas, including keeping the flow of ideas, as spatial memory is very strong, and this can facilitate recall. Sticky notes, each with one idea, are useful for re-arranging and grouping ideas.

5. *Sharpen the focus.* Start the brainstorm with a well-honed problem. This will get the brainstorm off to a good start, and it makes it easier to pull people back to the main topic if the session wanders.
6. *Use warm-up exercises and make the session fun.* The group will require warming up if they haven't worked together before, most of the group doesn't brainstorm regularly, or the group is distracted by other pressures. Warm-up exercises might take the form of word games or the exploration of physical items related or unrelated to the problem at hand, such as the TechBox in Chapter 2.

11.5 Bringing Requirements to Life: Personas and Scenarios

Using a format such as those shown in Figure 11.1 or user stories captures the essence of a requirement, but neither of them is sufficient on their own to express and communicate the product's purpose and vision. Both can be augmented with prototypes, working systems, screenshots, conversations, acceptance criteria, diagrams, documentation, and so on. Which of these augmentations is required, and how much, will be determined by the kind of system under development. In some cases, capturing different aspects of the intended product in more formal or structured representations is appropriate. For example, when developing safety-critical devices, the functionality, user interface, and interaction of the system need to be specified unambiguously and precisely. Sapna Jaidka et al. (2017) suggest using the Z formal notation (a mathematically based specification language) and petri nets (a notation for modeling distributed systems based on directed graphs) to model the interaction behavior of medical infusion pumps. Harold Thimbleby (2015) points out that using a formal expression of requirements for number entry user interfaces such as calculators, spreadsheets, and medical devices could avoid bugs and inconsistencies.

Two techniques that are commonly used to augment the basic requirements information and to bring requirements to life are personas and scenarios. Often used together, they complement each other in order to bring realistic detail that allows the developer to explore the user's current activities, future use of new products, and futuristic visions of new technologies. They can also guide development throughout the product lifecycle.

11.5.1 Personas

Personas (Cooper, 1999) are rich descriptions of typical users of the product under development on which the designers can focus and for which they can design products. They don't describe specific people, but rather they are realistic, and not idealized. Any one persona represents a synthesis of a number of real users who have been involved in data gathering, and it is based on a set of user profiles. Each persona is characterized by a unique set of goals relating to the particular product under development, rather than a job description or a simple demographic. This is because goals often differ among people within the same job role or the same demographic.

In addition to their goals, a persona will include a description of the user's behavior, attitudes, activities, and environment. These items are all specified in some detail. For instance, instead of describing someone simply as a competent sailor, the persona includes

that they have completed a Day Skipper qualification, have more than 100 hours of sailing experience in and around European waters, and get irritated by other sailors who don't follow the navigation rules. Each persona has a name, often a photograph, and some personal details such as what they do as a hobby. It is the addition of precise, credible details that helps designers to see the personas as real potential users, and hence as people for whom they can design.

A product will usually require a small set of personas rather than just one. It may be helpful to choose a few, or maybe only one, *primary personas* who represent a large section of the intended user group. Personas are used widely, and they have proved to be a powerful way to communicate user characteristics and goals to designers and developers.

A good persona helps the designer understand whether a particular design decision will help or hinder their users. To this end, a persona has two goals (Caddick and Cable, 2011):

- To help the designer make design decisions
- To remind the team that real people will be using the product

A good persona is one that supports the kind of reasoning that says, “What would Bill (persona 1) do in this situation with the product?” and “How would Clara (persona 2) respond if the product behaved this way?” But good personas can be challenging to develop. The kind of information they include needs to be pertinent to the product being developed. For example, personas for a shared travel organizer would focus on travel-related behavior and attitudes rather than the newspapers the personas read or where they buy their clothes. On the other hand, personas for a shopping center navigation system might consider these aspects.

Steven Kerr et al. (2014) conducted a series of studies to identify user needs and goals in the kitchen as a way to improve the design of technology to assist with cooking. They conducted observations and interviews with three members each of three user groups, identified through background research: beginners, older experts, and families (specifically parents). The interview focused on topics such as cooking experience, meal planning, and grocery shopping. Two researchers attended each home visit. Notes, video, audio, and photographs were used to capture the data, including a think-aloud session during some of the activities.

Personas were developed following both inductive and deductive analysis (see Chapter 9, “Data Analysis, Interpretation, and Presentation”), looking for patterns in the data and commonalities that could be grouped into one persona. Three primary and three secondary personas were developed from the data. Figure 11.6 shows one primary (beginner) persona and one secondary (older expert) persona that they derived from this work. Note that the two types (primary and secondary) in this case have different formats, with Ben being more detailed than Olive.

They also conducted a survey online to validate the personas and to create a list of requirements for new technology to support cooking.

The style of personas varies widely, but commonly they include a name and photograph, plus key goals, user quotes, behaviors, and some background information. The examples in Figure 11.7(a) illustrate the persona format developed and used by the company in Box 11.3, which has been tailored for their purposes. Figure 11.7(b) shows the user journey associated with their personas.

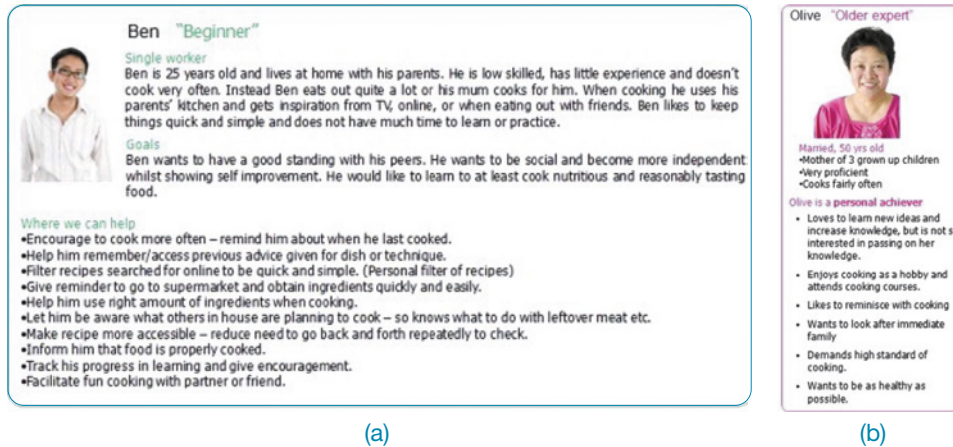


Figure 11.6 (a) One primary (beginner) persona and (b) one secondary (older expert) persona for cooking in Singapore

Source: Kerr et al. (2014). Used courtesy of Elsevier

BOX 11.3

Persona-Driven Development in London

Caplin Systems is based in the City of London and provides a framework to investment banks that enables them to build, or enhance, their single-dealer offering (a platform that integrates services and information for trading in capital markets) quickly or to create a single-dealer platform for the first time.

The company was drawn to use personas to increase the customer focus of their products by better understanding for whom they were developing their system. Personas were seen as a way to provide a unified view of their users and to start building more customer-focused products.

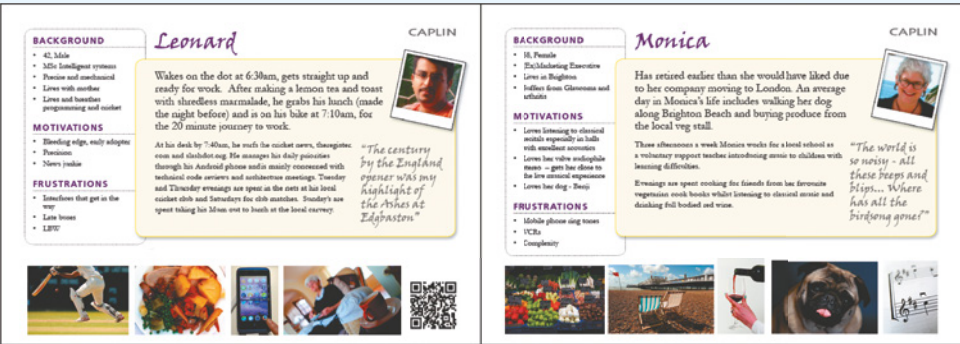
The first step was to run a workshop for the whole company, introduce personas, show how other companies were using them, and have employees experience the benefits of using personas firsthand through some simple team exercises. The following proposition was then put forward:

Should we adopt personas and persona-driven development?

The response was a resounding “yes!” This was a good thing to do. Gaining this “buy in” was fundamentally important in ensuring that everyone was behind the use of personas and committed to the change.

Everyone got excited, and work began to define the way forward. Further workshops were run to refine the first persona, though in hindsight the Caplin team believes that too long a time was spent trying to get the first persona perfect. Now they are much more agile about persona creation.

Eighteen months after the persona breakthrough workshop, the main persona for Caplin Trader, Jack, and his “pain points” were the focus of development, design decisions, and team discussions. Ongoing persona development focused on end users of the software built with Caplin’s technology, and Narrative Journey Maps captured their interactions and helped to define goals/motivations and pain points (see Figure 11.7b).



(a)



(b)

Figure 11.7 (a) Example personas, (b) the narrative journey maps—sad faces show pain points for the persona
Source: Caplin Systems ■

ACTIVITY 11.3

Develop two personas for a group travel organizer app that supports a group of people, perhaps a family, who are exploring vacation possibilities together. Use the common persona structure of a photo, name, plus key goals, user quotes, behaviors, and some background information. Personas are based on real people, so choose friends or relatives that you know well to construct them.

These can be drawn by hand, or they can be developed in PowerPoint, for example. There are also several tailorable persona templates available on the Internet that can be used instead.

Comment

The personas shown in Figure 11.8 were developed for a father and his daughter using templates from <https://xtensio.com/templates/>.

Family traveler



"I want a travel organiser that will offer me a range of potential vacations that suit our needs"

Age: 35
Work: Plumber
Family: Married, two children

Personality



Organised Practical Expects high standard

Goals

- To book comprehensive travel quickly
- To find a trip that meets the needs of the whole family
- To feel supported and guided from the beginning of the booking experience right to the end.

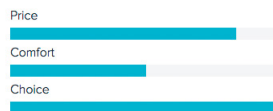
Frustrations

- Wasting time filling in forms
- Too much irrelevant information
- Existing systems tend to be too diverse and complicated

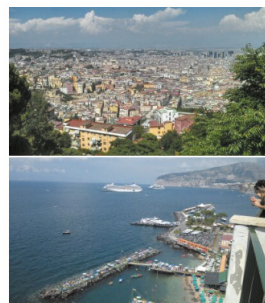
Bio

Will loves to take his family on adventure holidays to explore new challenges. His children, Sky (8) and Eamonn (15) are old enough to take part in several sporting activities and he wants to make the most of this before they no longer want to go on trips with him and his wife, Claire. He likes the fact that choosing travel options is so much easier than it used to be, but is frustrated by the many different sources and disjointed options that this can result in. He wants a travel organiser that can provide clear support for family holidays while offering as wide a choice as possible.

Motivation



Favourite destinations



Young traveler



"I want a travel organiser that will allow all of us to choose the vacation together"

Age: 8
Work: Schoolgirl
Family: Mum Dad and Eamonn (15)

Personality



Energetic Inquisitive Likes reading

Goals

- To find a good vacation without any fuss
- To find a destination with other children her age
- To make sure that the travel time is short

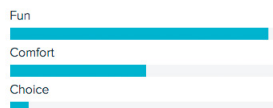
Frustrations

- Sitting around discussing things for too long
- Not getting clear answers to her questions
- Feeling that everything is organised for adults and not children her age

Bio

Sky likes having adventures. She is very energetic and takes part in lots of sporting activities at school, such as gymnastics and swimming. She enjoys playing games with her older brother, Eamonn. Sky is keen to make new friends, but is also happy sitting reading a book, painting or making a model. She likes going to visit new places but expects to see something familiar, such as playground or food that she recognises! The most important thing for her is that she can go on vacation with her family where there will be something for everyone to do - but especially for her and Eamonn.

Motivation



Favourite destinations

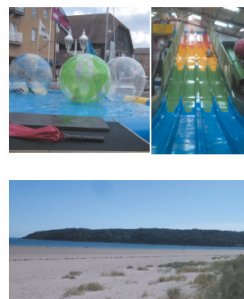


Figure 11.8 Two personas for the group travel organizer ■

This article by Jared Spool explains why personas on their own are not enough and why scenarios also need to be developed:

<https://medium.com/user-interface-22/when-it-comes-to-personas-the-real-value-is-in-the-scenarios-4405722dd55c>

11.5.2 Scenarios

A *scenario* is an “informal narrative description” (Carroll, 2000). It describes human activities or tasks in a story that allows exploration and discussion of contexts, needs, and requirements. It does not necessarily describe the use of software or other technological support used to achieve a goal. Using the vocabulary and phrasing of users means that scenarios can be understood by stakeholders, and they are able to participate fully in development.

Imagine that you have been asked to investigate how a design team working on a large building project shares information. This kind of team includes several roles, such as an architect, mechanical engineer, client, quantity surveyor, and electrical engineer. On arrival, you are greeted by Daniel, the architect, who starts by saying something like the following:

Every member of the design team needs to understand the overall purpose, but we each take a different perspective on the design decisions that have to be made. For example, the quantity surveyor will keep an eye on how much things cost, the mechanical engineer will want to make sure that the design accounts for ventilation systems, and so on. When the architect presents a design concept, such as a spiral staircase, each of us will view that concept from our own discipline and assess whether it will work as envisioned in the given location. This means that we need to share information about the project goals, the reason for decisions, and the overall budget, as well as drawing on our own discipline expertise to advise the client on options and consequences.

Telling stories is a natural way for people to explain what they are doing, and stakeholders can easily relate to them. The focus of such stories is also naturally likely to be about what the users are trying to achieve, that is, their goals. Understanding why people do things as they do and what they are trying to achieve in the process focuses the study on human activity rather than interaction with technology. Starting with current behavior allows the designer to identify the stakeholders and artifacts involved in an activity. Repeated reference to a particular app, drawing, behavior, or location indicates that it is somehow central to the activity being performed and that it deserves close attention to uncover the role it plays. Understanding current behavior also allows the designer to explore the constraints, contexts, irritations, facilitators, and so on, under which people operate. Steven Kerr et al. (2014), who devised the cooking persona Ben introduced in the previous section, also produced a scenario for Ben, which is shown in Figure 11.9. Reflect on the persona and read the scenario alongside to see how the two complement each other to give a fuller picture of the activities related to cooking in which a user such as Ben would engage.

Ben is out with friends, catching up over a meal. A friend asks if he has ever made the dish they are eating. This reminds Ben that he has not cooked for a while.

Later in the week, Ben sees a TV programme on cooking and again he is reminded that he has not cooked for a while. Thus he decides to cook later that week. Ben goes online and Google's "mee soto". He looks through the various sites for pictures that not only look good, but also have few steps and a short time to cook. He spends some time browsing through the sites for other ideas to use in the future.

Next day at work, he remembers to go to the supermarket on his way home. As he is only getting a few ingredients (for a simple recipe), he just remembers what he needs to get. He buys ingredients in whatever size is readily available and is not too concerned about the freshness. Whilst in the shop, he also makes some spontaneous purchases.

On the day he is cooking, Ben checks out a YouTube video on preparing chicken pieces. When he gets home he prints out a recipe and asks his mum for some last minute advice. He had asked her previously about a similar question, but was so long ago that he has forgotten her advice. His mum looks at the recipe and suggests some alterations to it and writes notes on the recipe.

Ben places the recipe near the stove and follows it closely in terms of steps (one process at a time, no preparation) though haphazardly gauges the amount of ingredients to put in. He tries to gauge one portion's worth of ingredients (the recipe is for 3 people), only using some of the chicken, so puts rest back in fridge (he thinks his mum might use it later). He will estimate the time the mee soto needs for cooking, tasting it when he thinks it is ready. He is initially worried about hot oil splashing on his face, so he is hesitant when handling the hot pan.

When ready, he serves the dish and finds out he has made a bit too much. Depending on how much leftover is available, he will either put it in the fridge for later or it will just be wasted.

Afterwards he loads up pictures of the dish on Facebook. He is greatly encouraged when people 'like' his link or leave a comment and it makes him feel good.

Figure 11.9 Scenario developed for Ben, the Beginner persona, for cooking in Singapore

Source: Kerr et al. (2014), Table 1. Used courtesy of Elsevier

The previous scenario describes existing behavior, but scenarios can also be used to describe behavior with a potential new technology. For example, a scenario that might be generated by potential users of a new navigation app for a large shopping center is given here:

Charlie wants to take his elderly mother, Freia, to his favorite home products store, ComfortAtHome. He knows that the store has moved within the shopping center, but he doesn't know where. He also needs to find a route that is suitable for his mother who uses a walker but doesn't like elevators. He opens the navigation app on his smartphone and enters the name of the store in the search feature. Two different branches of the store are listed, and Charlie asks for directions to the one nearest to their current location. A map of the shopping center is displayed, showing their current location, the location of the nearest store, and the suggested route. This route, however, includes a series of steps that are unsuitable for his mother. So, he asks for an alternative route that uses only ramps, which the app displays. They set off, following the new route provided.

Note the following in this limited scenario: the provision of a search facility to find the store's location (but what if the user doesn't know the store name or wants to locate all home products shops), the facility to display a map, and the importance of different options for the navigation routes to accommodate different users. These are all indicators of potential design choices for the new system. The scenario also describes one use of the app—to find a route to the nearest branch of a specific store.

During the requirements activity, scenarios emphasize the context, the usability and user experience goals, and the activities in which the user is engaged. Scenarios are often generated during workshop, interview, or brainstorming sessions to help explain or discuss some aspect of the user's goals. They capture only one perspective, perhaps a single use of the product, or one example of how a goal might be achieved.

The following scenario for the group travel organizer introduced in Activity 11.3 describes how one function of the system might work—to identify potential vacation options. This scenario incorporates information about the two personas shown in Figure 11.8. This is the kind of story that you might glean from a requirements interview:

The Thomson family enjoys outdoor activities and wants to try their hand at sailing this year. There are four family members: Sky (8 years old), Eamonn (15), Claire (32), and Will (35).

One evening after dinner, they decide to start exploring the possibilities. They want to discuss the options together, but Claire has to visit her elderly mother so she will be joining the conversation from her mother's house down the road. As a starting point, Will raises an idea they had been discussing over dinner—a sailing trip for four novices in the Mediterranean.

The system allows users to log in from different locations using different devices so that all members of the family can interact easily and comfortably with it wherever they are. The system's initial suggestion is a flotilla, where several crews (with various levels of experience) sail together on separate boats.

Sky and Eamonn aren't very happy at the idea of going on vacation with a group of other people, even though the Thomsons would have their own boat. The travel organizer shows them descriptions of flotilla experiences from other children their ages, and they are all very positive, so eventually, everyone agrees to explore flotilla opportunities.

Will confirms this recommendation and asks for detailed options. As it's getting late, he asks for the details to be saved so that everyone can consider them tomorrow. The travel organizer emails them a summary of the different options available.

Developing this type of scenario, which focuses on how a new product may be used, helps to uncover implicit assumptions, expectations, and situations in which the users might find themselves, such as the need to plan travel when participants are situated in different locations. These in turn translate into requirements, in this case an environment requirement, which may be expressed as follows:

As a <group traveler>, I want <to be able to share vacation discussions when I am not co-located> so that <the whole group can discuss their choices together under a wide range of circumstances>.

Futuristic scenarios describe an envisioned situation in the future, perhaps with a new technology and new world view. Different kinds of future visions were discussed in Chapter 3, "Conceptualizing Interaction," and one approach that is an extension of the scenario idea and that can be used to discover requirements is design fiction (see Box 11.4).

BOX 11.4

Design Fiction

Design fiction is a way to communicate a vision about the world in which a future technology is situated. It has become popular in interaction design as a way to explore envisioned technologies and their uses without having to grapple with pragmatic challenges. In a fictional world, ethics, emotions, and context can be explored in detail and in depth without worrying about concrete constraints or implementations. The term was first coined by Bruce Sterling in 2005, and its use has gradually increased as different ways of using it have emerged.

For example, Richmond Wong et al. (2017) took a design fiction approach to engage in issues of privacy and surveillance around futuristic sensing technologies. Their design fictions were inspired by a near-future science-fiction novel, *The Circle* by David Eggers (2013). Their design fictions are visual, and they take the form of a workbook containing conceptual designs. They draw on three technologies in the novel, such as *SeeChange*, a small camera about the size of a lollipop that wirelessly records and broadcasts live video. They also include a new prototyped technology designed to detect a user's breathing pattern and heart rate (Adib et al., 2015).

The design fictions go through three rounds. The first round adapted the technologies from the novel, for example, by adding concrete interfaces. As there were no photos in *The Circle*, the authors were able to design interfaces for these technologies based only on the textual descriptions. The second round considered privacy concerns and placed the technologies in an extended world from the novel. The third round considered privacy concerns in situations that went beyond the novel and the designs they had created up to that point in time. They suggested that their design fictions could help broaden the design space for people designing sensing technologies or be used as interview probes in further research. They also reflect that an existing fictional world is a good starting point from which to develop design fictions and this helps to explore futures that might otherwise go unnoticed.

Other examples of design fiction include Eric Baumer et al.'s (2018) consideration of how design fiction can support the exploration of ethics and Steve North's (2017) approach to embrace the perspective of a nonhuman user—in this case a horse.

What's the difference between scenarios and design fiction? Mark Blythe (2017) uses the "basic plots" of literature to suggest that scenarios employ the plot of "overcoming the monster," where the monster is some problem to be solved, while design fiction more frequently takes the form of a "voyage and return" or a "quest." ■

ACTIVITY 11.4

This activity illustrates how a scenario of an existing activity can help identify requirements for a future application to support the same user goal.

Write a scenario of how you would go about choosing a new hybrid car. This should be a new car, not a secondhand car. Having written it, think about the important aspects of the task, your priorities and preferences. Then imagine a new interactive product that supports this goal and takes account of these issues. Write a futuristic scenario showing how this product would support you.

(Continued)

Comment

The following example is a fairly generic view of this process. Your scenario will be different, but you may have identified similar concerns and priorities.

The first thing I would do is to observe cars on the road, specifically hybrid ones, and identify those whose looks I find appealing. This may take several weeks. I would also try to identify any consumer reports that include an assessment of hybrid cars' performance. Hopefully, these initial activities will result in identifying a likely car for me to buy.

The next stage will be to visit a car showroom and see at first hand what the car looks like and how comfortable it is to sit in. If I still feel positive about the car, then I'll ask for a test drive. Even a short test drive helps me to understand how well the car handles, if the engine is noisy, how smooth are the gear changes, and so on. Once I've driven the car myself, I can usually tell whether I would like to own it or not.

From this scenario, it seems that there are broadly two stages involved in the task: researching the different cars available and gaining firsthand experience of potential purchases. In the former, observing cars on the road and getting expert evaluations of them are highlighted. In the latter, the test drive has quite a lot of significance.

For many people who are in the process of buying a new car, the smell and touch of the car's exterior and interior and the driving experience itself are the most influential factors in choosing a particular model. Other attributes such as fuel consumption, interior roominess, colors available, and price may rule out certain makes and models, but at the end of the day, cars are often chosen according to how easy they are to handle and how comfortable they are inside. This makes the test-drive a vital part of the process of choosing a new car.

Taking these comments into account, we've come up with the following scenario describing how an innovative "one-stop shop" for new cars might operate. This product makes use of immersive virtual reality technology that is already in use by other applications, such as designing buildings and training bomb disposal experts.

I want to buy a hybrid car, so I go down the street to the local "one-stop car shop." The shop has a number of booths in it, and when I enter, I'm directed to an empty booth. Inside, there's a large seat that reminds me of a racing car seat, and in front of that there's a large display screen.

As I sit down, the display screen jumps to life. It offers me the option of browsing through video clips of new cars that have been released in the last two years, or of searching through video clips of cars by make, model, or year. I can choose as many of these as I like. I also have the option of searching through consumer reports that have been produced about the cars in which I'm interested.

I spend about an hour looking through materials and deciding that I'd like to experience the up-to-date models of a couple of vehicles that look promising. Of course, I can go away and come back later, but I'd like to have a go right now at some of the cars I've found. By flicking a switch in my armrest, I can call up the options for virtual reality simulations for any of the cars in which

I'm interested. These are really great, as they allow me to take the car for a test drive, simulating everything about the driving experience in this car—from road holding to windshield display and foot pedal pressure to dashboard layout. It even re-creates the atmosphere of being inside the car.

Note that the product includes support for the two research activities mentioned in the original scenario, as well as the important test-drive facility. This would be only a first-cut scenario, which would then be refined through discussion and further investigation. ■

Scenarios may be constructed as textual descriptions, like those shown previously, but they can also use audio or video. For example, Alen Keirnan et al. (2015) used animated scenarios to present early user research findings about wearable emergency alarms for older people. They found that the animated scenarios helped participants to describe problems associated with the use of emergency alarm technology and to discuss and evaluate key emotions and themes (see Figure 11.10 and the following links to videos).



Figure 11.10 Screen captures of the animated scenarios used to explore early user research insights regarding emergency alarm technology for elderly people: I forgot, Dress Code, and Cow Bell

Source: Keirnan et al. (2015), Figure 1. Reproduced with permission of ACM Publications

Here are links to three animated scenarios: “I forgot,” “Dresscode,” and “Cowbell.”

<https://vimeo.com/126443388>

<https://vimeo.com/136821334>

<https://vimeo.com/123466330>

BOX 11.5

Scenarios and Personas

Writing personas and scenarios can be difficult at first, leading to a set of narratives that conflate details of the person with details of the scenario. A scenario describes one use of a product or one example of achieving a goal, while a persona characterizes a typical user of the product. Figure 11.11 captures this difference graphically.

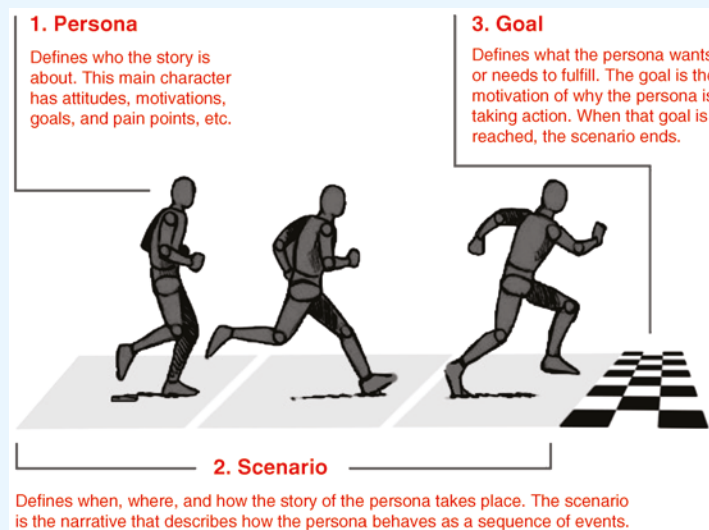


Figure 11.11 The relationship between a scenario and its associated persona

Source: <http://www.smashingmagazine.com/2014/08/06/a-closer-look-at-personas-part-1/>. Reproduced with permission of Smashing Magazine

This image also shows that the scenarios and personas are tightly linked. Each scenario represents a single experience of using the product from the perspective of one persona. Note that this figure introduces the notion of a scenario goal. Thinking about the persona’s goal for the scenario helps to scope the scenario to one use of the product. ■

11.6 Capturing Interaction with Use Cases

Use cases focus on functional requirements and capture interaction. Because they focus on the interaction between a user and the product, they may be used in design to think about the new interaction being designed, but they may also be used to capture requirements—to think through details about what the user needs to see, to know about, or to react to. Use cases define a specific process because they are a step-by-step description. This is in contrast to a *user story*, which focuses on outcomes and user goals. Nonetheless, capturing the detail of this interaction in terms of steps is useful as a way to enhance the basic requirement statement. The style of use cases varies. Two styles are shown in this section.

The first style focuses on the division of tasks between the product and the user. For example, Figure 11.12 illustrates an example of this kind of use case, focusing on the visa requirements element of the group travel application. Note that nothing is said about how the user and product might interact, but instead it focuses on user intentions and product responsibilities. For example, the second user intention simply states that the user supplies the required information, which could be achieved in a variety of ways including scanning a passport, accessing a database of personal information based on fingerprint recognition, and so on. This style of use case has been called *essential use cases*, and they were developed by Constantine and Lockwood (1999).

retrieveVisa	
USER INTENTION	SYSTEM RESPONSIBILITY
find visa requirements	request destination and nationality
supply required information	obtain appropriate visa information
obtain a personal copy of visa information	offer information in different formats
choose suitable format	provide information in chosen format

Figure 11.12 An “essential use case for “retrieve visa” that focuses on how the task is split between the product and the user

The second style of use cases is more detailed, and it captures the user’s goal when interacting with the product. In this technique, the main use case describes the *normal course*, that is, the set of actions most commonly performed. Other possible sequences, called *alternative courses*, are then captured at the bottom of the use case. A use case for retrieving the visa requirements for the group travel organizer with the normal course being that information about the visa requirements is available, might be as follows:

1. The product asks for the name of the destination country.
2. The user provides the country’s name.
3. The product checks that the country is valid.
4. The product asks the user for their nationality.
5. The user provides their nationality.

6. The product checks the visa requirements of that country for a passport holder of the user's nationality.
7. The product provides the visa requirements.
8. The product asks whether the user wants to share the visa requirements on social media.
9. The user provides appropriate social media information.

Alternative Courses

4. If the country name is invalid:
 - 4.1 The product provides an error message.
 - 4.2 The product returns to step 1.
6. If the nationality is invalid:
 - 6.1 The product provides an error message.
 - 6.2 The product returns to step 4.
7. If no information about visa requirements is found:
 - 7.1 The product provides a suitable message.
 - 7.2 The product returns to step 1.

Note that the number associated with the alternative course indicates the step in the normal course that is replaced by this action or set of actions. Also note how specific the use case is about how the user and the product will interact compared with the first style.

In-Depth Activity

This activity is the first of five assignments that together go through the complete development lifecycle for an interactive product.

The goal is to design and evaluate an interactive product for booking tickets for events such as concerts, music festivals, plays, and sporting events. Most venues and events have booking websites or apps already, and there are many ticket agencies that also provide reduced tickets and exclusive options, so there are plenty of existing products to research first. Carry out the following activities to discover requirements for this product:

1. Identify and capture some user requirements for this product. This could be done in a number of ways. For example, observing friends or family using ticket agents, thinking about your own experience of purchasing tickets, studying websites for booking tickets, interviewing friends and family about their experiences, and so on.
2. Based on the information you glean about potential users, choose two different user profiles and produce one persona and one main scenario for each, capturing how the user is expected to interact with the product.
3. Using the data gathered in part 1 and your subsequent analysis, identify different kinds of requirements for the product, according to the headings introduced in section 11.3. Write up the requirements using a format similar to the atomic requirements shell shown in Figure 11.1 or in the style of user stories.

Summary

This chapter examined the requirements activity in greater detail. The data gathering techniques introduced in Chapter 8 are used here in various combinations in the requirements activity. In addition, contextual inquiry, studying documentation, and researching similar products are commonly used techniques. Personas and scenarios help to bring data and requirements to life, and in combination they can be used to explore the user experience and product functionality. Use cases and essential use cases are helpful techniques for documenting the findings from data gathering sessions.

Key Points

- A *requirement* is a statement about an intended product that specifies what it is expected to do or how it will perform.
- Articulating requirements and defining what needs to be built avoids miscommunication and supports technical developers and allows users to contribute more effectively.
- There are different kinds of requirements: functional, data, environmental (context of use), user characteristics, usability goals, and user experience goals.
- Scenarios provide a story-based narrative to explore existing behavior, potential use of new products under development, and futuristic visions of technology use.
- Personas capture characteristics of users that are relevant to the product under development, synthesized from data gathering sessions.
- Scenarios and personas together can be used throughout the product lifecycle.
- Use cases capture details about an existing or imagined interaction between users and the product.

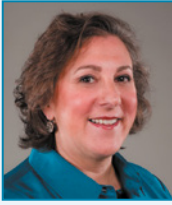
Further Reading

HOLTZBLATT, K. and BEYER, H. (2017) *Contextual Design (second edition) Design for life*. Morgan Kaufmann. This book provides a comprehensive treatment of contextual design—design for life, cool concepts, and all of the models, techniques, principles, and underpinnings for the complete contextual design method.

COHN, M. (2004) *User Stories Applied*. Addison-Wesley. This is a practical guide to writing good user stories.

PRUITT, J. and ADLIN, T. (2006) *The Persona Lifecycle: Keeping people in mind throughout product design*. Morgan Kaufmann. This book explains how to use personas in practice—how to integrate them into a product lifecycle, stories from the field, and bright ideas, as well as many example personas. It also includes five guest chapters that place personas in the context of other product design concerns. *See also* Adlin and Pruitt (2010).

ROBERTSON, S. and ROBERTSON, J. (2013) *Mastering the Requirements Process (3rd edn)*. Pearson Education. In this book, Suzanne Robertson and James Robertson provide a very practical and useful framework for software requirements work.



INTERVIEW with Ellen Gottesdiener

Ellen is an Agile Product Coach and CEO of EBG Consulting focused on helping product and development communities produce valuable outcomes through product agility. She is the author of three acclaimed books on product discovery and requirements including *Discover to Deliver: Agile Product Planning and Analysis*. Ellen is a frequent speaker and works with clients globally. She is Producer of Boston's Agile Product Open and Director of Agile Alliance's Agile Product Management initiative. Visit these websites for resources: www.ebgconsulting.com and www.DiscoverToDeliver.com.

What are requirements?

Product requirements are needs that *must* be satisfied to achieve a goal, solve a problem, or take advantage of an opportunity. The word “requirement” literally means something that is absolutely, positively, without question, necessary. Product requirements need to be defined in sufficient detail for planning and development. But before going to that effort and expense, are you sure they are not only must-haves but also the right and relevant requirements?

To arrive at this level of certainty, stakeholders ideally start by exploring the product's *options*. An option represents a potential characteristic, facet, or quality of the product. Stakeholders, who I like to refer to as product partners, use expansive thinking to surface a range of options that

could fulfill the vision. Then they collaboratively analyze the options and collectively select options, based on value.

Every product has multiple dimensions, seven in fact. Discovering options for each of the seven product dimensions yields a comprehensive, realistic view of the product (see Figure 11.2).

You want to engage diverse stakeholders across the full product lifecycle, from birth to retirement and demise.

So how do you know who are the stakeholders?

Successful teams work hand in hand with their stakeholders as *product partners*, defining value and then actively discovering—and delivering—high-value solutions. This goes beyond feature requests and requirements documents—beyond user stories and product backlogs—beyond the push-pull of competing interests. It's a partnership where the ideas, perspectives, and experiences of three different stakeholder groups converge. The result? Product partners who collaborate to discover and deliver value.

A product partnership includes people from three realms: customer, business, and technology. Each offers a unique perspective and has its own ideas of what is valuable.

The *customer* partners represent users, buyers, and advisers—people or systems that interface with the product, choose to buy it, or influence others to buy it.

They tend to value improved productivity, heightened efficiency, greater speed, entertainment, and similar benefits.

Business partners represent the people in your organization who authorize, champion, or support the product or who provide subject-matter expertise. They find value in improving market position, complying with regulations, achieving a business case, reducing overhead costs, enhancing internal performance, and so on.

Technology partners (your delivery team, internal or third parties) design, deliver, test, and support the product or advise those who do. They may value building a high-quality product, offering smooth, continual delivery, adopting a stable architecture, and the like.

This mix of partners and perspectives is essential, no matter what kind of delivery method you adopt (agile, traditional, hybrid, or another approach). For the partnership to work, these three disparate groups must collaborate to reach their shared goal: discover and deliver value.

How do you go about identifying requirements?

Requirements discovery is highly proactive, interactive, and, well, sometimes hyperactive! You are engaged in eliciting, analyzing, specifying, prototyping, and testing. And in the best practices we've been involved in, you are constantly discovering (aka identifying) product needs. It's not a "one-and-done" activity.

Elicitation includes interviews, existing documentation study, exploratory prototypes, facilitated workshops, focus groups, observation (including apprenticing, contextual inquiry, and ethnography), surveys (and other research-based techniques), and user task analysis (including storyboarding and scenario analysis). There

are a number of specific techniques within each of these general categories, and some techniques overlap. Analyzing involves using lightweight models, often combined with specifications, which are often in the form of acceptance tests or prototypes or both.

It's not enough to get the right people together and ask the right questions. To communicate efficiently and effectively about how to deliver, product partners need a focused way to communicate and make decisions together.

What we've found in our work is that the most efficient and effective discovery mechanism is a collaborative approach called the "structured conversation." In a structured conversation, the product partners first explore possible requirements (options) for their next increment. They do this within and across the seven product dimensions. This enables product partners to collaboratively and creatively explore a range of possibilities. This expansive thinking opens up product innovation, experimentation, and mutual learning.

They then evaluate these many options in terms of value. This means having shared understanding of what value really means at that point in time. Once they have narrowed the list of options through the evaluation process, they confirm how they will verify and validate these candidate solutions with unambiguous acceptance criteria. The validation includes how to test that they delivered the right requirements, and that they achieved the anticipated value from each delivery.

How do you know when you have collected enough requirements to go on to the next step?

I often get asked by clients how I know when I have a complete set of requirements.

(Continued)

I think it's more important to ask whether you are going after the right requirements.

I characterize a “right requirement” as one that is:

1. **Just in time, just enough.** It is essential for achieving the business objectives in this time period.
2. **Realistic.** It is capable of being delivered with the available resources.
3. **Clearly and unambiguously defined.** Acceptance criteria exist that all partners understand and will use to verify and validate the product.
4. **Valuable.** It is indispensable for achieving the anticipated outcomes for the next delivery cycle.

What's the hardest thing about establishing requirements?

People. Seriously. We humans are nonlinear creatures. We are unpredictable, fickle, and (as adults) often inflexible. As requirements seekers, we swim in a stew of complex, ever-evolving human systems that interoperate as we do our requirements work.

To top that off, most products' requirements are fraught with complexity and interdependency; there are truly wicked problems, whereby the problem space overlaps with solution space. As Frederick Brooks said [in his essay *No Silver Bullet*], “the hardest single part of building a software system is deciding precisely what to build.”

You can't make those decisions without trust. And trust is not an easy thing to build.

Do you have any other tips for establishing requirements?

Employ small, tightly wound cycles of requirements-build-release. Use interactive and incremental (aka agile) practices to get feedback early and often on the smallest viable releases.

For successful requirements discovery, you need to keep the focus on value—the *why* behind the product and the value considerations of the product partners. During discovery work, some people view a specific option as a “requirement” for the next delivery cycle, whereas others consider it a “wish list” item for a future release.

Such was the case in a recent release planning workshop. The team wrestled with a particular option, questioning if it could deliver enough value to justify the cost to develop it. The product champion explained why the option was a requirement—without it the organization was at risk for regulatory noncompliance. Once the others understood this, they all agreed it would be included in the release.

In the end, requirements work is human-centric and central to successful product delivery. At the same time, the subject matter and content of product requirements is complex. Thus, requirements work is the hardest part of software and will always be.

To be successful with requirements, engineer collaboration into requirements work.

Personally, I'm excited and grateful for the growing recognition of the value of collaboration and the explosion of interest in collaborative practices in the product and software development community—because collaboration works!