CHAPTER 3

The System Requirements

In this chapter we look at:

- Different types of requirements
- The stages of requirements elicitation, specification and validation
- Interviews
- Questionnaires
- Structured meetings
- How to document requirements
- Fagan inspections

After studying this chapter and working through the exercises you will be able to:

- Explain what is meant by the term 'requirements engineering'
- Give details of the stages of requirements elicitation, specification and validation
- Describe some of the different techniques used at each stage
- Plan an initial interview with a client
- Design a simple questionnaire
- Document requirements

Introduction

The first task for a system developer in any development project is to find out as much as possible about the client organization and its problems. There are many ways of doing this, such as observing the company at work and studying relevant documents, but often the most effective way is to talk directly to the clients. In this

chapter we include an interview with Harry and Sue of the Just a Line Company that gives further details about their problems and what they hope a software system can do for them.

3.1 Background

Anecdotal evidence suggests that errors in requirements may account for approximately 50 per cent of the total cost of debugging a software system, yet it is only relatively recently that serious research has been carried out on the subject of system requirements. Many of the traditional system development methods, which are generally underpinned by a standard life cycle model, merely pay lip service to the problems of identifying, describing and validating the client's requirements for the system.

Chapter 2 describes the initial problem definition of a system. In the past this was agreed and signed off by the client, who would then frequently have little more to do with the development process until delivery and installation of the final system. Needless to say, this method of development often led to unsatisfactory systems and unhappy clients. Today, ***requirements engineering*** is recognized as a crucial stage in the development of software. Each year more and more requirements methods become available, often as expensive commercial products, involving computer-based tools, training programmes and extensive documentation. On the academic side, requirements engineering has become a major research area, supported by journals, specialist groups and international conferences.

3.2 What are requirements?

The problem of definition

Requirements engineering is generally regarded as having three distinct, but overlapping, stages: ***requirements elicitation, specification and validation***, but one of the main problems in this area is that there is no consensus of opinion as to what is meant by these terms. Some developers and writers talk about requirements, others about constraints, while some make a distinction between system requirements and software requirements. For the purposes of this chapter we shall assume that *system requirements* refers to the client's needs and wishes, whereas *software requirements* covers constraints put on the system development, such as hardware, software and design methods. When capturing system requirements the developer works with clients and users (in the case of Just a Line, this means Harry and Sue), identifying their needs, wishes and available resources, and must produce documents that can be understood by them as well as by computer professionals. When capturing software requirements the developer works from the system requirements documents, producing software requirements that must be understood by software designers and programmers. We shall also

assume that the expression *requirements engineering* covers the three phases of elicitation (identifying requirements), specification (describing them in an appropriate language or notation) and validation (checking with the client that the description accurately records his or her needs and wishes).

Evolving requirements

Modern approaches to requirements engineering differ significantly from traditional development methods in that they do not assume that a requirements specification document is agreed by the client and then remains cast in stone for the duration of the software development project. System developers today are fully aware that requirements are dynamic and evolve constantly during development of a software system. Organizations themselves are constantly changing; although the basic business, such as selling cards for Just a Line, will remain the same, the company's scope and objectives will change. For Just a Line, for example, the Internet and the World Wide Web will have a huge impact on the way that they do business. Moreover, development of the software system itself has an effect on the organization; in-depth discussions about the company and its problems often lead to fresh ideas and to new ways of working which will, in turn, have an effect on the original system requirements. In the Just a Line interview, which you will find later on in this chapter, Sue admits to 'an overall feeling of being disorganized'. It is likely that close examination of current working procedures during development of the new software system will make both Harry and Sue aware of where these can be improved and this will have an effect on their original list of requirements.

Different types of requirements

In the past, requirements engineering meant defining ***functional requirements***: what the system was to do, what its inputs and outputs were and how these were linked. Correct functional requirements are still considered essential for successful software development today, but in recent years developers have also come to realize the importance of ***non-functional requirements***. These can be defined as the attributes of the system as it performs its job and can be divided into non-functional requirements of the system and non-functional requirements arising from external sources. Non-functional requirements of the system will include:

- Usability. Does the system attract or put off its intended users? Is the right level of help provided? Are options clearly displayed and easy to follow? Does the system fit in with the user's preferred way of working?
- Performance. Does the system respond quickly enough for the user's needs? Can it cope efficiently with the required volume of transactions? What will happen in the case where the volume of data exceeds the specified capacity of the system?
- Reliability. Can the client have confidence that the system will behave consistently as expected?

- Security. How easy is it for non-authorized users to access the system and read or modify confidential data?

Non-functional requirements that arise from external sources include methods of operation, such as the client's existing procedures, physical constraints, such as the layout of the accommodation available, and international quality control standards, such as ISO/9001.

3.3 Stages in engineering the system requirements

Requirements elicitation

Requirements engineering begins with the task of finding out as much as possible about the client's organization, their current problems and what they would like the new system to do for them. This seems deceptively simple, since it involves sifting through large amounts of information and deciding what exactly is relevant. It is actually extremely difficult for the system developer to be sure that he or she has a complete and accurate understanding of what the clients want. Good communication skills, both oral and written, are essential for requirements elicitation, since nearly all methods of fact-finding depend on communication with clients and users. Requirements elicitation covers several different types of activity, such as observation of the users at work, and a study of relevant documents and user questionnaires, but the most effective way of getting information is simply to talk to the people involved in the system.

Interviews

A useful interview is one that has been prepared thoroughly. Both the developer and the interviewee should be clear about the purpose of the interview and what they each want to get out of it. A plan for the interview should be prepared in advance by the developer, identifying the purpose of the interview, listing any documents that are to be made available, and setting out a draft agenda. A plan, prepared by Barnes of D&B Software, for the first interview with Sue and Harry Preston of Just a Line is shown in Figure 3.1.

The developer should have established relevant details about the interviewee, such as his or her background, position in the organization, length of time with the company and special skills, such as level of computer expertise. It is part of the developer's job to put the interviewee at ease, particularly if the interview takes place away from the interviewee's place of work. It is worth spending some time chatting to the interviewee in general terms and very important to listen carefully to what he or she has to say, even if it does not appear to be directly relevant. The ability to listen attentively and identify important and relevant information is one of the essential skills for a system developer during requirements capture. Although direct questions are needed to control the interview, a lot of information can also be discovered by smiling, nodding encouragingly and making the

D&B Software – Interview Plan			
System: *Just a Line*		**Project reference:** JaL/MB/02	
Participants: Sue Preston (*Just a Line*) Harry Preston (*Just a Line*) Mark Barnes (Developer)			
Date: 10/4/02	**Time:** 14.30	**Duration:** 45 minutes	**Place:** Prestons' house
Purpose of interview: Preliminary meeting to identify problems and requirements regarding a software system for the *Just a Line* card retail company.			
Agenda: • current problems, particularly customer ordering, and any other concerns • current procedures • initial ideas • follow-up actions			
Documents to be brought to interview: • current catalogue • any documents relating to current procedures			

Figure 3.1 Plan for the first interview with Sue and Harry from Just a Line

interviewee feel that what he or she is saying is important. The developer should direct the interview, but must not dominate it.

On the next few pages you will find the first interview between the system developer and Harry and Sue of Just a Line. In any conversation of this type we can find several different kinds of information. Some of these are listed below:

- Information which is already structured in lists or forms.
- Information about company procedures; how certain tasks are carried out at present.
- Measurements such as the number of customers or the average size of an order.
- Problems that the client has identified in the current system. Definite requirements for the new system.
- Information that is not stated directly, but where there are definite 'vibes'. An example of this might be where the clients complain that they are always rushed when the supplier's order comes in, whereas what is actually happening is that the supplier always delivers late.

As you read the interview with Harry and Sue, try to identify examples of the different kinds of information that tell the system developer about Just a Line.

Interview with Sue and Harry from Just a Line

HARRY: Nice to see you. Traffic a bit heavy, was it?

MARK BARNES (SYSTEM DEVELOPER): Yes, dreadful. I'm sorry I'm a bit late, I got badly held up leaving the motorway.

SUE: Don't worry, it was good of you to phone and let us know. Now, a cup of tea perhaps. I'll put the kettle on and we can tell you a bit about the company.

MB: Yes, I've heard a lot of good things about your cards, and I was wondering if you could show me some samples later on. I'm very interested to hear how you got the idea for the business.

SUE: Of course we'd love to show you the cards and we're always keen to get comments on possible new lines.

MB: Well, just to put you in the picture about this meeting – what I'd like to do is get a good idea of the company and what you'd like from a computer system. I'd like to cover what you do, how you work, what you see as your current problems and how you think a software system could help. Is that OK with you?

HARRY: Fine, Sue will be quite happy to talk Just a Line all night.

MB: Right, perhaps you could start by describing the main jobs you have to do to run Just a Line. What about the first contact with the customers, for example?

SUE: Well, we meet them when we take the orders, or speak to them at least, if it's a phone order – it usually is. That's a real pain. Most of the customers are so chatty, they ask all about what we do and what we stock, and we have to explain all about the different sorts of cards and designs and all that. We try to persuade them to come into the shop to see for themselves, but they don't always want to. Then we get on to the personalized bit and we have to talk them through all the various typefaces and colours and what have you. People are always so friendly and interested and they usually end up ordering quite a few cards, but it's terribly awkward to deal with it all over the phone.

MB: What happens when they ask you detailed questions? Do you have all the information in your heads, or do you have to go and look things up all the time? And perhaps you can tell me how you keep information about card designs, prices, etc.

SUE: We've got a list of the card designs we offer on a regular basis – there's a copy stuck on the wall by the phone. We can usually remember what the design or picture is like by the name, so we just describe it to them as best we can, and quite often people have seen samples on our flyers in places like the library anyway.

MB: Do you get many comments from customers about your ordering procedures – either good or bad?

SUE: Well, our local customers do like the personal touch, but I'm sure a lot of people find it very frustrating that our processes aren't more streamlined.

MB: I think it might be a good idea to ask customers to fill in a simple questionnaire about ordering cards from Just a Line. What do you think?

HARRY: Yes, that's a good idea. It would give us more information about how the customers feel about the way we handle that side of things.

MB: Good, we can get that organized. Now, what about the prices? Is that a separate list?

SUE: Yes, we keep that by the phone as well, so that it doesn't get lost. The price list gets updated more often than the design list – in fact here's an old one; you can keep it.

MB: I see. I suppose this is how you hold information about suppliers, too. I see it's got their name and address on it, as well.

SUE: Yes. Well, at the moment there's only one supplier. We're thinking about using a different one who does recycled paper. We're very keen on that. But we may be able to persuade our present supplier to sell it as well, which would be a lot simpler than dealing with two suppliers. Anyway, eventually the customers tell us what they want to order and we take it down, in triplicate, on our order forms using carbon paper.

MB: Three copies?

SUE: The top two copies go out with the order. The customer keeps one as a delivery note, the other comes back with the customer's payment. We keep the third one until the signed copy comes back with the payment. Everything's cash on delivery at the moment, although we realize we're going to have to start thinking about credit accounts and that sort of thing.

MB: What about pricing? Do you work out the cost when you take the order?

SUE: Sometimes, because customers want to know how much it's going to cost. But quite a lot of people just leave it to me to work out the cost later. That's better really.

MB: Then what happens?

SUE: To the orders? Well, they go in the order file – our whole filing system is just one big filing cabinet with three drawers and orders go in the top drawer. They go in the drawer in the order we receive them, unless there's a very urgent one. We generally tell the customers to allow a maximum of 28 days before they get the cards.

MB: Then, when the cards are ready you deliver them?

SUE: Yes, if it's local – or else we use the post.

MB: Well, I've got a reasonably good picture of how you handle orders. Just one other thing – have you thought about selling cards via the Internet, you know with a website to display the designs, and online ordering?

SUE: Yes absolutely, I think it's a brilliant idea. I'm not sure that Harry's so keen though.

HARRY: Well, I'd just rather get a straightforward system going here first. I know the Internet seems to be a perfect way of selling the cards, but it would mean a huge amount of change in the way we do everything and I'm not sure that we're ready for that.

SUE: But Harry, you know how important it is to keep ahead of the game. I think our designs are great, but they won't sell themselves without some sharp marketing.

MB: Well, let's leave that for the moment; we can always come back to it. What we can do is design this system so that we can develop Internet facilities later on if you decide that's the way you want to go. What about dealing with stock? How do you organize that?

SUE: We usually reorder about once a month. Our current supplier delivers free if we can order more than £500 worth of stuff. At first we couldn't always afford to make that saving, but things are going reasonably well now, so we usually manage to. It's complicated. We can't just order one particular line if we've got a run on it, but we usually manage to organize a large enough order each month. Fortunately, we can get what we order delivered within a week. When it's obvious we're getting low on a few lines, I check to see if we can order enough to get the free delivery. We swore we'd never run into debt with the supplier, and so far we've managed to keep to that.

MB: How do you actually decide what to order?

SUE: It's a bit hit and miss sometimes. I look in the store where we keep all the cards and make a list of what's getting low, particularly with popular lines. Then I have a look through the orders from customers and make an estimate of what we need. The idea is to order so that everything runs down at about the same rate. I've pretty well got the hang of it by now.

MB: OK . . . so you have to keep checking manually that you have enough cards of each line in stock to cope with current orders and what you think customers will order in the near future? As you say, it sounds as if it can be a bit hit and miss, but it gives me a good idea of how a new system will be able to help. We can make sure that the system keeps track of all sales and alerts you when an item is running low. By the way, what happens if you do run out of something and can't meet a customer's order within the 28 days?

SUE: We just tell the customers we haven't got what they want at the moment. They usually don't mind as long as we tell them when they order. If we find out later, then we phone them. They're mostly very nice about it. I get Harry to tell them, he's much better at that sort of thing than I am and he can often persuade them to order something else instead. We always try to remember to say everything is 'subject to availability' when they place the order, but it's easy to forget.

MB: So, you send out orders to your suppliers a bit irregularly, but roughly once a month. What happens when the stuff arrives? How do you pay?

SUE: They send an invoice with the goods. We check everything's there and sign for it. Then they invoice us. Then we pay them.

MB: What is it you sign, when they deliver?

SUE: It's the bottom copy of their invoice.

MB: You have to record customer payments, too, of course?

SUE: Yes. When we get a payment we take our copy of the order out of the top drawer of the filing cabinet and put it in the second drawer. We call that one 'past orders'. We make a note of the payment in our cash book. Harry usually banks the money weekly, or daily if a lot has come in.

MB: Cash book? What else goes in there?

SUE: Everything, I'm afraid! We record all the money going in and out. So, as well as customer payments, we record payments we make to our supplier, and all our running expenses, like petrol, postage, phone calls, and so on.

MB: Well, I think I've got some idea about how you run the company, now. My main impression is that we definitely need to work on the customer ordering side. I'm sure we could make that a lot more efficient. What I'd like to do is go away and sort it all out a bit in my mind, then come back to you and check I've got things right. In the meantime, are there any other problems you feel we haven't covered?

SUE: Not really, it's just an overall feeling of being disorganized. That worries me, especially now that some of the big stores seem to be interested in the cards. I do think we've got a good product to sell and I don't want to mess it all up by seeming to be unprofessional.

MB: Well, that's just the sort of problem we deal with. I'll be in touch again soon and I'm sure we'll be able to get you organized. Now could I have a look at the cards? I'm really keen to see them after all you've told me.

HARRY: Great, that's terrific. If you'd like to order some I could even do you a discount.

Questionnaires

Apart from interviews with clients and users, the most useful form of requirements elicitation is often a questionnaire. This is particularly effective when a small, well-defined amount of information is needed from a large number of people, especially if they are widely scattered. It could be used, for example, in the Just a Line case study to find out what the company's customers think about the Just a Line method of ordering cards.

As with interviews, it is essential to prepare questionnaires thoroughly, including testing on a small sample of people to ensure that the questionnaire is easy to understand, simple to fill in and that it will produce useful results. It is the responsibility of the system developer to make sure that people who fill in the questionnaire are aware of its purpose and how their answers will be used. A variety of question types may be used, including multiple choice, short answer and extended answer questions, but the main priority must be to ensure that all questions are as clear and straightforward as possible. If a question does not contain enough information, the person filling in the questionnaire will not understand what is required, but if it contains too much information, nobody will bother to read it.

Figure 3.2 shows an extract from a questionnaire on Just a Line's ordering methods. The purpose of the questionnaire is to help the system developer find out

Just a Line Ordering Service – Customer Survey

We are intending to move to a new computerized ordering system for our cards in the near future. It would be a great help to us if you could spare a few minutes to give us your opinions of our current ordering system and any suggestions you have for improving it. Please answer the questions below and return the form to us in the enclosed pre-paid envelope.

1. How many times have you bought cards from Just a Line?

   Once only [ ]

   2–5 times [ ]

   6 times or more [ ]

2. Have you experienced any problems with the Just a Line ordering system?

   Yes [ ] please explain briefly below

   No [ ]

3. For each of the statements (a)–(e) shown below, circle the number that is closest to your own view, where 1 means that you agree strongly with the statement and 5 means that you strongly disagree.

(a) The current system works well.	1	2	3	4	5
(b) The staff are always friendly and helpful.	1	2	3	4	5
(c) It is easy to choose and order cards.	1	2	3	4	5
(d) I would like to see more information about my order.	1	2	3	4	5
(e) I feel that a computerized ordering system would be more efficient.	1	2	3	4	5
(f) I would like to view card designs on the web.	1	2	3	4	5
(g) I would like to order cards on the web.	1	2	3	4	5

4. Please note below any other comments you have on the current system.

5. Please note below any suggestions for the new system.

Your name: _____

Your address: _____

Thank you for completing this questionnaire.

Figure 3.2 Just a Line ordering service – customer survey

what Just a Line's customers think about the way ordering of cards is handled at present and to elicit ideas from them on how the process might be improved.

Structured meetings

Just a Line is a relatively small, simple system and requirements elicitation could be carried out using just interviews and questionnaires. However, in the case of large, complex systems, successful requirements elicitation often involves structured meetings with ***stakeholders*** who may include not only clients and users, but also user managers, marketing staff, project managers and software trainers. Such meetings often take up the time of many people and so must be carefully planned with a well-defined purpose and clear agenda. Effective management during the meeting is also important in order to avoid unnecessary conflict or deviation from the main purpose.

One particular type of structured meeting, which originated in Scandinavia, is known as the future workshop. Future workshops come from a development approach called Participatory Design which views users of the system as experts in the problem domain and aims to include them as active collaborators in the development process. A future workshop is generally organized and run by two facilitators who spend some time beforehand familiarizing themselves with the client organization, its processes and any existing hardware and software.

The workshop itself is divided into three separate stages: critique, fantasy and implementation. During the critique stage, participants in the workshop focus on current problems in the organization; brainstorming rules apply, so speaking time is limited, statements do not have to be justified and criticisms are not allowed. The aim of the fantasy stage is to imagine the perfect future system, without considering any constraints. This, again, is carried out using brainstorming to generate statements about the future system; at the end of this part of the workshop, participants vote to rank the statements, in order to identify those that have most support. In the final part of the workshop, the implementation stage, participants discuss ways in which the fantasy view may be realized, taking recognized constraints into account. After the workshop, the facilitators develop part of the proposed implementation as a prototype in order to validate and refine the system requirements.

Requirements specification

Whereas requirements elicitation involves an expansion of the developer's knowledge about the problem domain and the client's wishes, requirements specification involves sifting through the information to filter out the important and relevant issues and record them in an appropriate form. This may be narrative English, diagrams or a mixture of the two. The techniques described in Chapters 4, 5, 6 and 7 are commonly used to record requirements. Alternatively, in certain ***safety-critical or security-critical systems***, it may be appropriate to describe some or all of the requirements using a formal, mathematical language such as the ***Z specification language***. It is beyond the scope of this book to discuss such

languages, but you can find references in the bibliography to books that describe them in detail. One of the most effective ways of recording requirements for a software system is rapid ***prototyping***, where the developer builds an unpolished version of all or part of the system. Clients and users can then get a feel for what the new system will be able to do and what it will look like. Rapid prototyping allows clients to see how their requirements translate into a computer system; it is particularly useful when requirements are uncertain. You will find a discussion of prototyping in Chapter 12 of this book.

Whatever language or method is chosen for the specification of requirements, certain information must be provided and the requirements specification itself must have certain qualities. For each separate requirement the following information should be included:

- A number or code that uniquely identifies the requirement.
- The source of the requirement.
- The date this version of the requirement was suggested.
- A brief, natural language description of the requirement.
- Priority of the requirement (essential (E), desirable (D) or optional (O)).
- A list of related requirements.
- Alternatives to the requirement, if any.
- Related documents, diagrams or tables.
- If the new requirement involves a change to a previous one, then this must be fully documented, together with the reasons for the change and the effects it will have on other system requirements.

No.	Source	Date	Description	Priority	Related Reqs.	Alternative Reqs.	Related Docs.	Change Details
4.9	Meeting with Sue & Harry Preston.	10/4/02	System must keep track of sales and alert when an item is running low.	Essential	2.9		Sue's lists of current stock and estimates of what needs to be ordered	

Figure 3.3 Specification of a requirement from the Just a Line system in tabular form

The specification is a cornerstone of a system development project, since it encapsulates the shared understanding and intentions of all the stakeholders. The specification may be used as a vehicle for communication between developers, users and other stakeholders; it may also form the basis of a legal contract between developer and client, and it is the document that guides the programmers in their implementation of the system. Much has been written recently about the desirable qualities of a requirements specification. One of the most useful sources is the IEEE Recommended Practice for Software Requirements Specifications from the IEEE

Standard 831–1993. The Standard describes qualities that are essential for a good requirements specification document, including correctness, consistency and understandability. Many of these qualities are simply common sense, but the Standard is useful as a check-list.

Requirements validation

Although it is a time-consuming process to check that a requirements specification has all the qualities mentioned above, it is technically feasible for the system developer to confirm that the requirements specification document is actually of the desired quality. What is much more difficult to ascertain is whether the requirements expressed in the specification are really what the client wants and needs. The situation is further complicated by the fact that the client may not know what he or she wants, or that what is wanted may be completely different from what is needed. The process of checking that the requirements as specified are a true representation of the client's needs and wishes is known as validation.

Validation of requirements is essential from the earliest stages of requirements engineering. During interviews with clients and users there should be constant feedback to ensure that the developer has fully understood what is being said. This is also useful in that it helps the interviewee to feel that what he or she is saying is helpful and relevant. The extract below from the Just a Line interview illustrates these points.

SUE: It's a bit hit and miss sometimes. I look in the store where we keep all the cards and make a list of what's getting low, particularly with popular lines. Then I have a look through the orders from customers and make an estimate of what we need. The idea is to order so that everything runs down at about the same rate. I've pretty well got the hang of it by now.

MB: OK . . . so you have to keep checking manually that you have enough cards of each line in stock to cope with current orders and what you think customers will order in the near future? As you say, it sounds as if it can be a bit hit and miss, but it gives me a good idea of how a new system will be able to help. We can make sure that the system keeps track of all sales and alerts you when an item is running low.

Initial validation is also carried out by taking notes during the interview and later producing a written summary for the interviewee. The developer should always ask permission to take notes and be prepared to show the interviewee what is in them. The written summary should be produced shortly after the interview, so that the interviewee remembers what was said, but has had time to think about it and can check that the developer has understood the important points. A summary of the initial interview with Sue and Harry from Just a Line can be seen in Figure 3.4.

Different methods of requirements elicitation can be used in conjunction to validate requirements. This may be carried out by comparing answers on a particular topic from a questionnaire with comments on the same topic that have been obtained during interviews. A client's account of certain business procedures may also be checked by observation of how the procedures are carried out in

D&B Software – Interview Summary			
System: *Just a Line*		**Project reference:** JaL/MB/02	
Participants: Sue Preston (*Just a Line*) Harry Preston (*Just a Line*) Mark Barnes (Developer)			
Date: 10/4/02	**Time:** 10.30	**Duration:** 45 minutes	**Place:** Prestons' house
Purpose of interview: Preliminary meeting to identify problems and requirements regarding a system for the *Just a Line* card retail company.			

No.	**Item**	**Action**
1.	Phone orders are time consuming and complicated, both for Just a Line and probably customers.	Produce simple questionnaire on current ordering procedures and ask customers to fill it in with next order.
2.	Designs often have to be described over the phone.	Consider setting up a website to display designs.
3.	Prices have to be updated manually on a central list.	
4.	Possible move to more than one supplier.	New system must cater for multiple suppliers.
5.	Currently no facilities for customer credit.	
6.	Total cost of order cannot always be given over the phone.	
7.	Sometimes orders cannot be fulfilled because of stock problems.	New system must keep track of sales and alert when stock of an item is running low.
8.	''Overall feeling of being disorganized''.	New system must streamline procedures to make company more efficient. Arrange follow-up meeting with Sue and Harry (in about 10 days time).

Figure 3.4 Summary of the initial interview with Sue and Harry Preston of Just a Line

practice. Other techniques may also be introduced to aid the validation process. The technique of prototyping, discussed in Chapter 12, is useful at all the stages of requirements capture. When used for validation, a prototype allows the client and users to get some feeling for how their ideas would work once implemented in a computer system. The structured modelling techniques, described in Chapters 4, 5, 6 and 7, are designed to be user-friendly and to facilitate validation. Since the client's original requirements are generally expressed in natural language, one of the most effective methods of validating the requirements specification is simply to talk through it with the client and users.

One very popular way of validating requirements is to carry out a ***Fagan inspection***. Fagan inspections aim to uncover defects in the output from any stage in the software development process; they are a systematic and structured way of checking documentation, such as the requirements specification. Each inspection

usually concentrates on a small component of the documentation and lasts up to two hours. Inspections are carried out by small teams of people consisting of a moderator or chair, a note-taker, the person who produced the component and one or more people to inspect it. A typical Fagan inspection has six separate stages:

- Planning, when the team is selected.
- Overview when the component is presented to the team.
- Preparation, when each team member studies the component individually.
- Meeting, when the team reviews the component together.
- Rework, to remedy the defects that have been identified and agreed by the team.
- Re-inspection, which may be carried out by the chair, rather than by the whole team.

Although it is never possible to prove conclusively that a requirements specification describes exactly what the client wants and needs, it is essential that both the system developer and the client are happy that the requirements as stated have been thoroughly validated. The validation process can be seen as the application of ***quality assurance*** as applied to the requirements specification, leading to a well-founded belief on all sides that the specification is an accurate description of the system requirements.

Summary

Requirements engineering is one of the most important stages in the system development process, since getting the requirements wrong can (and often does) cause serious problems. Requirements engineering is generally considered to have three distinct, but overlapping stages. These are requirements elicitation (where the developer tries to find out as much as possible about the client's problems and what they want from the new system), requirements specification (where the requirements are recorded as fully and as accurately as possible), and requirements validation (where the developer checks that the clients' requirements have been understood and described correctly).

Exercises and topics for discussion

What can you remember?

You will find all the answers in the chapter

a) What are the three stages of requirements engineering?
b) What is meant by 'evolving requirements'?
c) What is the difference between functional and non-functional requirements?
d) Give two examples of non-functional requirements.

e) What happens during requirements elicitation?

f) What is one of the most important skills in requirements elicitation?

g) Name three of the techniques used in requirements elicitation.

h) What happens during requirements specification?

i) Name two ways of describing requirements.

j) What happens during requirements validation?

k) What is an interview summary used for?

l) What is a Fagan inspection?

3.1 Find an example from the Just a Line interview of each of the different types of information listed in Section 3.3

3.2 What other questions could the system developer usefully have asked Harry and Sue?

3.3 You have been asked by the town council to develop a system to regulate traffic in a local car park. Design a questionnaire that will help you to find out how the car park is used at present and how people feel that it could be improved.

3.4 Organize a future workshop with a small group of people to identify initial requirements for one or more of the following systems:

- A cash machine to be situated on a university campus.
- A system to handle orders from a charity's Christmas catalogue.
- A public information system for your local area.
- A system to control traffic lights at the junction near a village school.

The workshop should cover the three stages of critique, fantasy and implementation, as described in this chapter in the section on structured meetings.

Further reading

Institute of Electrical and Electronics Engineers, Inc. (1994) *IEEE Recommended Practice for Software Requirements Specifications*, New York.

Kotonya, G. and Sommerville, I. (1997) *Requirements Engineering: Process and Techniques*, John Wiley and Sons, Chichester.

Macaulay, L. (1996) *Requirements Engineering*, Springer-Verlag, London.

Pfleeger, S.L. (2001) *Software Engineering, Theory and Practice*, 2nd edn, Prentice-Hall, Upper Saddle River, N.J.

Sommerville, I. and Sawyer, P. (1997) *Requirements engineering A good practice guide*, John Wiley and Sons, Chichester.