TM354-14J Software engineering

Tutor-marked assignment TMA 01

Contents

Session

Introduction

Submitting your assignment

This tutor-marked assignment (**TM354 TMA 01**) must be submitted by 12 noon (UK local time) on **3 December 2014**.

This module requires all assignments to be submitted electronically. To submit an assignment, please follow the link(s) from your StudentHome page to the online TMA/EMA service.

Ensure that you create and submit your answers in a file with an acceptable file format. Acceptable file formats are those with extensions of .doc, .rtf or .docx. If your word processing software creates a file with a different extension (e.g. .odt) you should use 'save as' then select an option to provide a file with the correct format; otherwise your tutor may not be able to open your TMA. It is generally advisable when including diagrams created in word processor applications to 'group' the elements of the diagram together to ensure that your tutor will see what you intended upon opening the file. Alternatively, or in addition to your main submission, you are permitted to submit a .pdf file for *supporting documents only* providing that it is zipped with your main assessment files before submission. You must *not* submit your main assignment text as a .pdf file. Your tutor will not mark this if you do.

If you foresee any difficulty with submitting your assignment on time, you should contact your tutor well in advance of the cut-off date.

For further information about policy, procedure and general submission of assignments please refer to the Assessment Handbook, which can also be accessed via your StudentHome page.

Mark allocation

The weighting of this assignment is 33 per cent of the continuous assessment score.

This TMA assesses Block 1, as detailed in Table 1.

Table 1

Question	Marks	Units
1	20	1, 2
2	20	2
3	20	1, 3, 4
4	25	1, 3, 4
5	15	1, 3, 4

Question 1 (20 marks)

In this question you will consider the principles and practices of software development – both plan-driven and agile. For part (a) you will need to listen to an extract from Audio 1, which was first introduced in *Study guide 5* and is repeated here for convenience.

Audio content is not available in this format.

Audio 1 Requirements in agile projects

View transcript

a. Software projects are often associated with long and complex documentation. Agile projects are often associated with minimalist documentation.

   Listen to Audio 1, which gives a balanced perspective on documenting requirements. In particular, concentrate on the extract between 10 minutes 19 seconds and 19 minutes 56 seconds.

   i. Give two reasons why documentation may be important in a software project.
   ii. Suggest two principles that should guide the production of documentation in a software project.

   (*4 marks*)

b. The manifesto for agile software development sets out four values:
   - individuals and interactions over processes and tools
   - working software over comprehensive documentation
   - customer collaboration over contract negotiation
   - responding to change over following a plan.

   One particular form of agile development is Scrum. The Mountain Goat Software website provides a brief introduction to Scrum.

   Taking each of the agile values listed above as a heading, explain how the principles of Scrum development are relevant to each value. You will need to use your judgement in relating what you read on the website to the four values.

   Your answer must be mainly in your own words, with any quotations from third-party sources (including the one given above) clearly indicated by quotation marks and an appropriate reference.

   Write no more than 400 words, excluding the headings.

   (*12 marks*)

c. In software development what are *models* and in what ways do they contribute to the development process?

   (*4 marks*)

Walton e-shop

Questions 2, 3, 4 and 5(a) make use of an imaginary ecommerce application, the Walton e-shop, which is introduced in Box 1.

This example will also be used in TMA 02 and TMA 03 and runs as a unifying thread throughout the assignments.

Box 1 Walton e-shop

Walton Ltd is a start-up company planning to launch an online shop selling a range of products, such as electrical equipment, garden equipment, kitchenware, and so on. Instead of purchases being delivered to the customer's address, Walton will have a series of depots in major towns and the customer will be able collect their purchase the next day from a depot convenient to them. Although there are already many online retailers, Walton believes that this ready-next-day model will allow it to gain a significant position in the market.

A software system is required that will support Walton's business in the ways listed below. The new system will be known as the Walton shopping system (WSS).

Website browsing

Any member of the public can browse the products on offer. Products are organised into categories, such as electrical, garden equipment, and so on. Customers can view a picture and description of each product, see the price, read reviews of it by purchasers, and see the average star rating purchasers have given it.

Customers can also browse a list of collection depots, to help them locate a convenient choice. If a customer selects a depot, the system displays the address.

Registration

Only registered shoppers can place orders. To register online, shoppers must choose a username and password and supply the following information:

- billing address
- email address
- credit or debit card details.

Shopping

Once a registered shopper is logged into the system, an open order is created to which the shopper can add items from any category. To add an item to their order a shopper clicks on it and enters the quantity they require of the item. While the order remains open the quantities can be modified at any time. If the shopper decides to remove an item from their order, they do so by setting the quantity to zero.

Checking out

Once a shopper is satisfied with their order, they proceed to checkout, where they are asked to choose a depot for collecting their purchases. They are then invited to place their order and may accept or decline. Once an order is placed the customer is issued with a unique order number and sent an email confirming the order. A warehouse system, separate from the WSS, generates a corresponding invoice and schedules the items to be shipped to the chosen depot.

Once the shopper collects and signs for their purchases at the depot, the staff there record this on the system. The shopper’s card account is debited by the amount on the invoice and the order is marked as completed.

Returning an item

If goods are faulty or unsuitable. the shopper can return them. To do this they must log into the system and select the returns facility. The system will ask the shopper to provide the unique order number. If the number provided is invalid, the shopper is informed and the process terminates. Otherwise the shopper is requested to say which item they are returning, and write the reason for the return in not more than 256 characters. They are asked to confirm that they want to return the item, or instead to cancel the return procedure. The user provides a response. If they cancel, the process terminates. Otherwise the system generates a unique return number and communicates it to the shopper online. An email confirming the return number is sent at the same time.

The shopper can then return the item to the same depot they collected it from originally, quoting the return number. A depot assistant records the return, the system credits the customer’s card account with the amount of the item, and a refund note is emailed to the customer.

Writing reviews

Logged-in shoppers can submit written reviews of items they have purchased, and assign them star ratings on a scale of 1 to 5.

Viewing purchase history

Logged-in shoppers can browse a history of the orders they have placed.

Managing the system

The system is administered by Walton staff at its head office.

Administrators can:

- add or remove a category
- add or remove a product
- change the price of a product

When adding a category, the administrator must provide a category name which is different from the name of any existing category.

When adding a product the administrator must select a category and then provide the product details, consisting of name, picture description and price. Some products already have unique identification numbers in a standard format. If this is the case, the administrator provides the existing number. Otherwise the administrator notifies the system and the system generates a unique identification number for the product. The overall term Walton identification number (WIN) is used to describe both existing numbers and ones generated by the system.

Senior administrators can perform the tasks listed above, and in addition can add or remove a depot.

When adding a depot the senior administrator must provide a unique depot name and the address. When removing a depot, the senior administrator must provide the depot name.

Users of the system

Users of the system are members of the public browsing the site, registered shoppers, the warehouse system and Walton staff at collection depots and at head office.

Question 2 (20 marks)

In this question you are asked to complete selected sections of the Volere template for the Walton shopping system (WSS) (see Box 1).

The full template is long and it would not be practical to complete it all, which is why we only want you to fill in the parts we have indicated in the simplified version below.

a. *Purpose of the product*. What is the reason for building the product and how will it bring business advantage to Walton e-shop?

   (*2 marks*)

b. *Stakeholders*. Who are the stakeholders?

   (*2 marks*)

c. *Users of the product*. Who are the users?

   (*2 marks*)

d. *Naming conventions and definitions*. Give two terms from the description in Box 1 that might be added to the project glossary.

   (*2 marks*)

e. *Scope of the product*. What are the boundaries of the system?

   (*3 marks*)

f. *Functional requirements*. Identify three functional requirements for the WSS.

   (*3 marks*)

g. *Non-functional requirements*. Invent an example of a non-functional requirement that might be associated with one (or more) of the functional requirements you identified in part (f) above. For this example,
   - state the non-functional requirement and the functional requirement(s) it relates to
   - say what category of non-functional requirement it belongs to
   - provide a suitable fit criterion for the requirement.

   (*3 marks*)

h. *Project issues*. Choose any one of the project issues 18–27 listed in Section 6.1 of Block 1 Unit 2 and invent an example that could apply to the WSS. Only write a short answer.

   (*3 marks*)

Question 3 (20 marks)

In this question you will be interacting online with other members of your tutor group, using the ShareSpace environment. Before you start this question, make sure you have read the material about ShareSpace in the *Accessibility guide*.

You will need to make an early start on this question in order to allow time for the online interactions with your fellow students. So start at least two weeks before the cut-off date of the TMA.

a. Draw an activity diagram to model the process of returning an item, as described in ‘Returning an item’ in Box 1. You may produce the diagram using any tool you have available, or by hand (in which case you will need to photograph or scan it). Your diagram should take the process as far as the point where a return number has been communicated to the shopper and confirmed by email.

   Once you have completed your diagram, make it available to other students in your tutor group using the ShareSpace slot called ‘TMA 01 - My initial model’, which you will find via the My Work tab. You should do this at least 2 weeks before the TMA cut-off date to allow your fellow students time to comment.

   Copy this diagram into your assignment submission, labelling it ‘My initial model’.

   (*8 marks*)

b. Once you have uploaded your diagram to ShareSpace, look at the activity diagrams from other students, which you will find via the My Tutor Group tab. Use the ShareSpace comments facility to suggest improvements that could be made to the models. Focus your attention on the models that the diagrams represent, rather than details of how the diagrams are presented. Try to comment on diagrams that have not yet received many comments from other students, and that differ from your own diagram.

   Provide comments on at least two diagrams. Aim to write about 100 words of comment per diagram. Do this *at least a week* before the TMA cut-off date, to allow your fellow students time to make use of your comments.

   If you really cannot think of any suggestions for improvement, give some comments on two diagrams, discussing possible modelling alternatives that might have been considered in each case.

   For the purpose of this question, select two diagrams on which you have made comments. Copy both sets of comments into your assignment submission, labelling each of them clearly ‘My comments to [your fellow student’s name]’

   (*6 marks*)

c. Read the comments made by fellow students on your own initial model. Then introduce any changes you wish to make to your model and post a new version in ShareSpace, using the ShareSpace slot called ‘TMA 01 - My updated model’. Copy your updated diagram into your assignment submission, labelling it ‘My updated model’.

   In ShareSpace, add a comment to your new diagram explaining how and why you have improved your original model. If you did not make any changes, explain why. Aim to write about 100 words.

   Copy your comments into your assignment submission, labelling them ‘My comments on improving my model'.

(*6 marks*)

Question 4 (25 marks)

In this question you will develop a use case model for the Walton shopping system (WSS).

a. List the different actors, choosing appropriate role names for them, and identify and justify any possible relationships between actors.

   (*3 marks*)

b. List the use cases, choosing suitable names. You should aim to identify between 15 and 20 use cases.

   (*7 marks*)

c. Draw your corresponding use case diagram. You may produce the diagram using any tool you have available, or by hand (in which case you will need to photograph or scan it).

   Your diagram should include all the actors and use cases you identified in parts (a) and (b) above. It should also show their associations. You may need to identify relationships between use cases and/or between actors.

   (*8 marks*)

d. Suppose the WSS is to be developed incrementally, starting with a functioning core and adding a series of reasonably independent ‘chunks’. Suggest what should form the core and what other increments should follow, and in what order. Give brief reasons for your answers.

   (*4 marks*)

e. Give one example of a feature from the WSS that you think it would be important to prototype, giving your reasons. Suggest what form the prototype could take and who should test it.

   (*3 marks*)

Question 5 (15 marks)

In this question you will derive functional requirements from the use case in Box 2.

Box 2 Description of the add-product use case

Identifier and name	UC10 *add product*
Initiator	*Administrator*
Goal	A new product is added to the online shop
Precondition	None
Postcondition	A new product will have been added
Assumptions	The initiator is an administrator

Main success scenario

1	The administrator makes a request to add a product.
2	The administrator selects the category.
3	The system requests a name, picture, description and price.
4	The administrator provides a name, picture, description and price.
5	The system requests an identification number for the product.
6	The administrator provides the identification number.
7	The system adds the product.

Extensions

2.a.1	A new category is required. The administrator informs the system.
2.a.2	The system requests a category name.
2.a.3	The administrator provides a category name.
2.a.4	The system adds the category.
6.a.1	No identification number. The administrator informs the system.
6.a.2	The system generates an identification number.
6.a.3	The system issues the identification number to the user.

For simplicity we are assuming that:

- at step 2.a.3 the category name entered does not duplicate a category name already on the system
- at step 6 the number entered does not duplicate an identification number already on the system.

a. Section 7.2 of Block 1 Unit 4 illustrates how detailed functional requirements for a system can be derived from a textual description of a use case. Use the same approach to specify *five* detailed requirements, corresponding to the first five steps of the main success scenario described in Box 2.

(*10 marks*)

b. Once detailed functional and non-functional software requirements have been drawn up they must be checked for completeness. Outline briefly (in two or three sentences) how you would do this.

(*5 marks*)

Transcript

Interviewer So Suzanne, James – welcome to IEEE Software and SE Radio.

Suzanne Robertson Thank you.

James Robertson Thank you for having us.

Interviewer And how about first of all introducing yourselves to the listeners?

Suzanne Robertson Well, my name's Suzanne Robertson and I'm a Principal of the Atlantic Systems Guild. It's an organisation that's been in existence for – how long, James? – 27 years, I think now. And there are six of us in it and one of my partners in fact is James, who is sitting next to me.

James Robertson As you've no doubt gathered, I'm also a Principal of the Atlantic Systems Guild and we can portray some of those as we go through the interview.

Interviewer Excellent. Well, what we want to talk about today is looking at the intersection between agile, which has been around now for well over a decade, and its intersection with requirements. So I thought I'd start asking about when in your requirements work both of you became familiar or aware of agile projects and its potential impact on requirements work.

Suzanne Robertson Well, it was about, actually about ten years ago. Wasn't it when Kent Beck wrote the book about extreme programming?

James Robertson I think it was a little earlier than that, but – at least, I think Kent Beck wrote the book a little earlier than that, but without checking let's say ten years.

Suzanne Robertson OK. Well, I think probably it came into the mainstream – you know, people – he wrote very well. He had some very good ideas and it captured a lot of people's attention and they said, 'Oh, great. This means that we can just work in a small group and we can just build things.' And actually, one of the big mistakes I think that people made – not that Beck really intended them to do this – but one of the big mistakes that people made was to say, 'Oh, if we work closely with another person we don't really need to do any requirements work because we're there and we can build software.' It was very software focused. And those are my earliest memories of it. What about you? Can you remember that?

James Robertson I think my earliest memories are, well, reading Extreme Programming but also the – I don't want to use the word, but hysteria. The extreme zealotry that surrounded the introduction of agile techniques. This has been very much a facet of software engineering: as soon as some new paradigm comes along, everybody – not everybody, but a lot of people leap onto the bandwagon and throw everything away. We don't do what scientists do, which is stand on the shoulders of giants. As an industry, we tend to throw everything away and start again, and this happens – Suzanne and I have seen it happen four or five times in our working lifetime.

The problem with agile – and I'm talking here about 'big A' Agile – following a method, whether it's XP or Scrum or Crystal or many of one of the other flavours of it, is that the baby got thrown out with the bathwater. The problem was seen as requirements were all wrong. We had to have this giant requirement document that had to be written and complete and then thrown over the transom to the development team. And this was taking a long time for anything to happen and requirements were pointed at as being the villain of the piece. And so the early Agilists threw out requirements and I think that was a huge mistake. I think we are gonna come back to that in a – in a big way later on.

So my early – you know, if you're asking my early, you know, responses to agile was that this is living dangerously, because we've gotten rid of something that's valuable whereas really there's room for two things here. I think that would be my –

Interviewer Well, did you find that your clients were expressing that view or were they taking a more pragmatic middle road once they started to move to agile?

Suzanne Robertson Well, the clients were either confused and following some kind of a step-by-step method, or being driven by – sometimes by managers who had very little knowledge about what agile was about. It was just 'if you're agile, that means you can get things done much more quickly'. And consequently a lot of the teams that I was working with, and had been working with, just got an awful lot of pressure to be agile and really that translates to 'get this done more quickly'.

And I think the more intelligent views – more intelligent approaches have been when people have said, 'Well, hang on. What does it actually mean and why is it a good idea?' Because it is a good idea. It's a terrific idea. So what is it? And you then, you know, you make the abstraction and you think, well, what it really is is you want to produce value. And you want to produce it because we build systems for people, and systems – I'm talking about them in the wide, you know, not just software but systems. And we want to be able to produce this value and we want to be able to do it quickly and we want to be able to turn on a sixpence when businesses change – and they change more and more often.

And I think the people who really understand what agile is all about understand that. They don't say 'being agile means you do Scrum' or 'being agile means you don't do requirements' or whatever it is that they pick up from the patchwork quilt.

Interviewer So it's understanding the motivation, the rationale for agile rather than the mechanics of it.

James Robertson Oh, absolutely. I think that’s, you know, my point about ‘big A’ Agile and ‘little a’ agile. We’re very much in favour of ‘little a’ agile. I think if you follow too slavishly a method, it can lead you down the wrong road no matter what the – what the method is. And that was my experience with clients who were early adopters of it. They felt that if they simply did everything that the method – ‘big A’ Agile method said to do that their problems would be over. Well, it turns out that doesn’t work. It doesn’t work that way. We can’t – we can’t work following a slavish method, but if we’re flexible and not rigid – in other words, ‘little a’ agile – then it does bring benefits.

Suzanne Robertson Well, if people understand what ‘little a’ agile is then they can pick the techniques that suit particular situations. Like if they say, ‘OK, what we’re really trying to do is to understand the problem well enough, as quickly as we can, to decide what to do to give people the most value.’ Sometimes that will mean using a particular technique or building a particular type of model. Sometimes it’ll mean going through different sorts of cycles using prototypes, using simulation, all the things that we know about as being useful techniques and approaches. But the problem is that because it was new and because it was presented as a technique or an approach, that’s how people got to know about it. So they thought about it as a technique rather than as what’s behind it, and that’s what we were – we were meaning by – by ‘little a’ agile.

James Robertson Yes, most of my clients now are moving away from ‘big A’ Agile and to ‘little a’ agile, when they’re getting more value out of better thinking I suppose to –

Interviewer So how does ‘little a’ agile manifest itself in projects today?

James Robertson By the adoption of things outside of the original ‘big A’ Agile camp. For example, something that is near and dear to our hearts is systemic thinking. This is not mentioned in most of the textbooks but it’s a very important part of what we do. Innovation, for example, is not mentioned anywhere that I’ve been able to find, but nevertheless is absolutely vital if we’re going to be building useful things. And I think the – I don’t know who first said this but, and to paraphrase it, you can build something any way you want. You can build software any way you want, but if it doesn’t do something useful then by definition it’s absolutely useless. And something that – in the early adoption of agile, the emphasis was on doing things quickly. Well, I’m also gonna say, you can build something as quickly as you want to but if it doesn’t do what’s needed to be done, it’s absolutely useless. And so I think clients or customers and software builders are now coming to the realisation that it’s better to take a little bit longer to do something and get it right as opposed to build something totally useless on time and on budget.

Suzanne Robertson I think another thing that you see when people are really coming to grips with 'little a' agile is that they're not driven by any rigid process – not at all. What they're looking for is to come to grips with a problem as quickly as they can so that they can make some decisions about which bits to do what with – which bits to go further with. And they might – it might be a team who work in a particular way that they like or they might be experimenting with something else, but it's not driven by any process or, you know, any rigid process.

Interviewer I think one of – you often hear about agile is that either the stakeholders really can't express their requirements or by the time they've been able to express them the requirements have changed. We're living in such a dynamic world that it's almost pointless to document the requirements in any way that we might consider traditional. What's your view on that?

James Robertson I think requirements change much more slowly than people say they do. The problem is that the requirements are never gathered correctly in the first instance and so when they're presented back to the users, the users are saying 'no, that's not what we meant'. And that's taken to be a change in the requirement. Well, it's not a change in the requirement. The basic business need is still there. It just hasn't been expressed properly.

One of the things that I noticed about early 'big A' Agile adoption was the emphasis was on building something. 'Let's go and build some software.' And even this macho stuff about having iteration times timed in hours, if not minutes, in order to keep on producing software and keep on producing it and keep on producing stuff. This is producing a solution, OK, but it's not necessarily a solution to the right problem. Just because we deliver up a piece of software, it does not mean you're delivering value and this almost chanting of a mantra that went on about constant delivery of customer value software is not necessarily valuable. If it doesn't do what's needed then it's useless. It's not valuable at all. And so I've seen more and more, now, the stepping back from that constant delivery of software into getting a better understanding of what the real problem is, and I think that's extremely healthy.

Suzanne Robertson Well, also the other thing that you – you mentioned the D word – documentation. I think there's a lot of misunderstanding about that as well. I think that people think about if you do requirements, it means you've got to produce a lot of formally written documentation in large documents, blah, blah, blah. But it's not that at all. What it is is you want to leave a trail, and if you're working in a co-located environment there's nothing to stop you leaving your trail on the walls and on flip charts. We've got video cameras, we’ve got – we’ve got cameras all over the place. Everybody’s got them. We’ve got all sorts of technology that we can use to free us from having to sit down and say, ‘Well, now it’s time to do the documentation.’

I mean, you should only do a translation into another form if you need to do that translation in order for whoever it is your clients communicate with to understand it. So if you’re outsourcing or something like that, you probably need to spend more of your budget on translation. But you shouldn’t do it if you don’t need to.

Interviewer So the reported demise of more traditional requirements techniques is premature?

Suzanne Robertson Well, I don’t know, because I’m wondering what you mean by more traditional requirements techniques.

Interviewer Documenting requirements in a way that makes them measurable – that you have got a specification that can be analysed for various properties, completeness and correctness.

Suzanne Robertson But I think you see that you can do that in a co-located team with white walls. I mean, I know that’s extreme but you can do it. The key thing – and this is the big key – is to tell the difference between the form and the content, and the content is what you’re trying to communicate. Now if you’ve got a team, for example, and they are talking to each other about requirements, let’s say, and they say, ‘Well, we’ve got 125 requirements’ – if they haven’t got a shared mental model of what a requirement is, they’re just going to lose the meaning. So you’ve got to have some kind of shared mental model and it doesn’t have to be imposed by anybody else as long as that team understands that. If you’ve got that then you can trace the requirements. You can make them measurable. You can say whether you’ve got this bit of requirement – whether you’ve got a rationale – whether you haven’t. But it just might be in a different form.

Interviewer So agile brings us new forms of requirement. The content remains but the form changes.

James Robertson Absolutely. I think that requirements now – we’re accepting that some requirements can be verbally communicated, that they can be sketched and scribbled on the back of story cards and so forth. There is nothing wrong with that at all. I think it comes back partly to what are you calling a requirement? And when you say traditional requirements, if you think about traditional requirements as being the business analyst goes and talks to the customers, writes down whatever they say, translates that into some form and gives it the developers – yeah, that’s been gone for quite some time. If you’re talking about requirements in terms of – let me call this ‘little r’ requirements, in terms of actually finding out what the real problem is and having some way of communicating that, it doesn’t matter whether it’s written or not, then that's the way we see requirements going today.

There are some needs. You mentioned documentation a moment ago. There is a need for documentation still. What we have learnt from the agile methods that documenting what a piece of software does is completely useless because the code is the best documenter of functionality. However, it's not important to document that simply because we can read the code to find out what it's doing. Documenting why something exists is very important because in five years' time, when the software still lives – and let's face it, software is living longer and longer and longer. I sometimes do a brief survey in my requirements courses and find that there is software belonging to organisations represented in the room that's older than the students represented in the room. So knowing why something exists is important. So that kind of documentation has not gone away, or at least the companies that were developing software without keeping that kind of documentation realised that they do need it after all – the rationale behind something. Which is why we have it on our show cards to always document the reason behind something.

And I think it comes back to when you talk about requirements - or we were talking earlier about requirements changing. If you do discover the rationale for a requirement, or even a rationale for a use case as a rationale for a whole piece of work, you find that that concentrates everybody's mind on exactly what the problem is. And you find the problem doesn't change but proposed solutions to that problem do change. And so rather than demise I'm going to say no, requirements work is morphing into something different with an emphasis on rationale, reasoning behind things, with an emphasis on discovering the real work as opposed to the emphasis being on traditionally writing the large document.

Suzanne Robertson Large document – yeah.

James Robertson Mind you, having said all of that, there is still a need for a large document if you're outsourcing. And let's face it, most organisations today are outsourcing. You need to write the complete specification because when you send that off to India or Russia or Indonesia or wherever you're sending it to, they do need a complete specification. Very few outsourcers are gonna work iteratively with you.

Interviewer I can see how that can work for functionality, functional requirements. What about quality requirements and usability and reliability? I've seen them scribbled on the backs of story cards and so on. Do you think those have to be documented in different ways in agile?

James Robertson I do, yes, because the qualitative requirements – what we call the non-functional requirements – quite often exist across several businesses or end user cases, across a number of different transactions, whatever you – partition you want to use. So attaching them to one story card is possibly not the best thing to do, simply because if there's another similar story they may not pick up the same non-functional requirements. So it's almost as if the qualitative requirements have to be separately documented because they are an overarching quality of the whole development effort as opposed to one particular –

Suzanne Robertson Yes, but they also need to be connected to the relevant bits of functionality and they also need to be connected to the people who really understand that particular type of non-functional requirement. So it's like now you've got the thread that goes from the functionality to the non-functional requirements to the particular stakeholders who are the experts. And we're seeing this more and more that because our systems can do many more things and can support many more non-functional requirements, we're seeing more and more internal consulting stakeholders, people who are usability experts who just focus on those – on that type of requirement and they're experts in that. And if there is some kind of formal way of writing those things down, or making them available to people anyway, that means that you then start to open the door to reuse, to really open the door to reuse because you don't have to go and discover those usability requirements and write their measurements down every single time. But that doesn't happen unless you've actually built up that sociology in the organisation.

Interviewer OK. So taking agile with a small a, how do you think your consulting and requirements practices have changed over the last decade? How have you modified or changed what you teach or train people?

Suzanne Robertson Well, now because people are becoming more aware of a wide picture – not enough, but they are becoming – they're starting to realise that we have to be system thinkers. Because of that we can actually get them to start thinking about the requirements and really at the beginning of doing a piece of work – and I really mean at the beginning, before you actually start to say 'OK, let's go and find out all the atomic requirements'. Why not identify just that really simple model that we've used, the scope, stakeholders and the goals. If you can make that visible as the highest level requirements, but you're actually making them measurable, and at that point identify the pieces well enough to start prioritising. And this is where the 'a' – the 'little a' comes in, because the earlier you can start prioritising the pieces the more agile you're able to be, because you can say, 'Oh look, you know, we've done this – this quick analysis. We can identify pieces and we've got 50 pieces and we're going to go through quick prioritisation: whereabouts is the major value, where are the difficulties, risks, costs and …' But you can do this quite quickly –

Interviewer So would this be on sketched requirements – requirements that are not perfectly formed as we expect them to be?

Suzanne Robertson Well, it's connected back to – the functional side of it is connected back to the scope. The non-functional side is connected back to the stakeholders. So even though you don't know everything downwards in the detail, you do know what you're talking about at the high level. In other words, high level doesn't mean 'I don't really know what I'm talking about. Maybe I'll find out.' High level means we understand this – we understand what we think it is now so that we will know if it changes. And that means you can then – you can be very agile. You can say, 'OK, we are gonna work on this piece and we're going to deliver something' or 'We're going to work on this piece because we think it's the riskiest thing and if we can't do this we may as well, you know, stop doing the project.' It's those things that people can be – can be used to be much more agile.

James Robertson I think the changes I've seen is that the 'big A' Agile folks are coming much more round to looking very much at the problems faced. And this is where – if you want to talk about traditional requirements and traditional agile coming together, it's an investigation of the problems faced to find out what is it we're really trying to do here. And what we're trying to do is not to build a piece of software. What we're trying to do is improve the business. And there's a subtle, very important and very important difference between them – between the two of them.

The other one is, as I mentioned, prioritisation – looking at a problems face – cutting it up any way you like, but carving it up into pieces and then prioritising the pieces. Because a lot of what we do is not important, or a lot of what's been done is not important. We've built software to carry out functionality that's only used once every five years or isn't that important, doesn't happen very often, adds very little value to the owner of the software. So prioritisation I think is causing people to think about – much more closely about what they're doing and not develop as much software, but only develop for the important – the important things. How little can they actually get away with here?

Interviewer That's interesting. That's important. You mentioned systemic thinking. To me systemic thinking involves some kind of modelling, or walking through something, but it involves artefacts. And all sorts of models and artefacts aren't typically things you associate with 'big A' Agile, say. They tend to eschew that kind of modelling and they see it as documentation. Do you use those sorts of techniques at all in your approaches?

James Robertson Oh, absolutely.

Suzanne Robertson Very much.

James Robertson Systemic thinking is hugely important because – and it doesn't matter what your method is saying, 'you can't build a diagram' is rubbish simply because systemic thinking is difficult enough if you try to do it all inside your head or on the back of a card. It's not gonna work, OK? Typically you do need some kind of model just to see the effect of what you're doing. And building a piece of software –

Sort of back up a moment here. I was talking just a little while ago about the problems facing – investigative problems faced – this is something we are seeing more and more. People thinking about the work area that they're trying to improve and not just the software, OK. The early 'big A' Agile went straight into a piece of software and had no idea really whether it would solve the right problem, because they were just looking at this very narrow focus. Well, focus is broadening. We're looking at a piece of work, but then systemic thinking says you've got to look at that piece of work – if we change that piece of work, what happens to the rest of the organisation that it's going into? If it's a simple simplistic piece of software – like a game, for example – you know you don't care what happens around that because the person is gonna buy the game or not. What else they do in their life, you're not concerned with. If you go and put in a new inventory control system into a car manufacturer, you can feel very concerned about what effect that has on manufacturing, on finance, on personnel, on everything else. And if we don't think about all these other things then we've got the potential for catastrophe.

Suzanne Robertson But that points to something that I've seen people more and more ready now to go back to things that have proved to be useful in the past and connect them to this idea of being intuitive, being agile, being – being able to look at things in hierarchies just as, you know, good systems thinkers can do. So, for example, people are now saying, 'Well, process models work in certain cases. Data models work, state models work, systems flow, dynamics models work.' But they're also starting to understand how you can connect those so that you're being consistent and you can – you can choose which ones to use in what situations, but you can connect them in to the overall picture of the project that you're doing. And that harks back to something I said earlier where people can choose the techniques, the procedures, the visualisations that are most appropriate to what they're doing. But it does rely on a really important thing about a systems thinking school, which I mentioned earlier – being able to distinguish between what is there because of a form and what's there because of a content. Not that form's bad, but just to be able to understand which is which.

Interviewer So do you see your role in part nowadays as helping a project choose the right techniques and tools to use, in a sort of pick and mix agile shop? There's a lot of choice out there and you may be overwhelmed by the choice. Is that a role that you see for yourselves?

Suzanne Robertson Oh, we do, we do that quite a bit with people, you know, help. Obviously you've got to learn a little bit about that, you know, that first high-level analysis. And then from there, because we've seen so many different approaches used, we can say to people, 'Why don't you start with – oh, I don't know – a data model, for example, because this is a very data-heavy problem and you're going to get much more insight more quickly if you start with data.' Well, these people might never have done that before because they were used to starting with worrying about processes or functions or whatever they were used to starting. So it's trying to help people to free themselves from 'this is the way we always do it'.

James Robertson Also freeing teams from believing that the model or the technique is the purpose of it. The purpose is to do some work, to improve the owner's business in some way. Getting into wars about techniques or diagrams and so forth – a lot of what I found myself doing is saying to people, 'You can build a model of data any way you like and if three different people are using three notations that's fine, because essentially they're all talking about the same thing, OK?' And there's no one model that's going to be notably superior to another. In fact, you can actually do it in a tabular form if you want to. It doesn't really matter.

Suzanne Robertson But getting people to realise that is hard if they haven't had a lot of experience, because it does rely on – on having done a fair amount of project work so that you can understand these different abstractions. We've been focusing a lot on helping people to become better abstracters, because I think everybody can be better at abstraction – it's just that quite often you're not taught that way. You're taught, you know, a particular way to do it rather than what's – what's behind it. And even if you were taught, you need a little bit more experience before you can say 'ah, this is really the same as this' or 'this will work better now'.

So we've been developing some models. I mean, our brown cow model is very useful as a way of helping people to see you can make, you know, there are four different viewpoints that you can take of anything that's useful. And traditionally in business analysis they say the 'as is' and 'to be' – well, that's not really enough. You need the 'as is' and how it's done and the 'to be' and 'how it'll be', but you also need what's behind it. And that's really looking at the product that's in the essence and all of that. So it's models and things like that that we've helped people to use that as a starting point to help them to be more agile, because if you know that you can take four different useful viewpoints – you can take lots more, but four different useful ones – then it doesn't matter where you start. And that's what we – that's what makes people agile.

James Robertson Yes, abstraction is one of those incredibly useful abilities to be able to see something independently of its implementation; in other words, the underlying reason behind it. We often refer to this as the essence of the problem. And once you can see the essence, see the reason behind something, it does help enormously to come up with better implementations. My earlier criticism of 'big A' Agile rushing into an implementation was that they were not looking at the underlying problem – in other words, were not taking any abstractions at all – and shouldn't be producing a piece of software. To complain about change in requirements is – by not looking at the abstraction, you build a piece of software, it's probably not going to meet the real need. And so when a change is asked for in that piece of software that's taken to be a change in requirement. Well, not at all. The original requirement is the same. And if people take a little bit of time to abstract away from implementations then you can get to see the real problem, and I think that helps enormously.

Suzanne Robertson But you see, if you're really agile, supposing that you're trying to understand the requirement and you're having trouble and people are having trouble telling you or you just don't get it, then it's an intelligent thing to do to build some kind of simulation or to build a prototype, to use that as a way of trying to figure out what is it you're really trying to do. And that's really – if you can do that sort of thing then you're really being agile.

Interviewer Are you beginning to see clients build solution abstractions – modelling architectures now? Because that was something that seemed to disappear in the 'big A' Agile in the early days. Moving straight from user storage to codes – where was the architecture? Is that reappearing in projects?

James Robertson Very, very slowly. For a reason I cannot explain, architecture is unpopular. It's one of those necessary things that nobody wants to talk about. And so architecture, yes, it is reappearing and people are understanding the need for it and so on. But again, architecture is not doing itself any favours with things like TOGAF 9 and only five people on the planet can understand it properly. So it's – I'm exaggerating here, I hasten to add. But wrapping it up in this arcane kind of presentation is not doing it any favours. So for that reason it's unpopular.

Interviewer Let's look at innovation, then. You mention that as one of the drivers, if you like, for projects nowadays and why we're developing software. How do you think innovation sits with agile? I can see how proponents of agile could argue it's responsive and dynamic. What are your experiences?

James Robertson We were actually arguing about this earlier on today. I'll present my version, which is no doubt the correct version of this –

Suzanne Robertson And then I'll argue with you.

James Robertson The most valuable innovations come early in the project and the most valuable innovations of those are the ones that affect the problems faced as a whole. Now, I'll hasten to add here, innovation is not coming up with some new technological gizmo with a touch-sensitive blast screen or a new iPad or anything else like that. Innovation is fresh thinking: thinking differently about the problem area. And the most valuable innovations are the ones where the whole problem is changed because of some innovative fresh thinking.

One example that springs to mind simply because I'm doing it at the moment is the First Direct bank here in the UK. First Direct – brand new bank, or was a brand new bank about five years ago. They started by saying, 'Let's look at the whole problem of banking and what is it that people like and what is it that people don't like.' So rather than coming at it from a traditional banking perspective, they said 'we want to look at banking from the point of view of a customer', OK – which is really quite an innovation when you think about it, because most banks look at banking from the point of view of the bank. And First Direct came up with a completely branch-free, paper-free way of banking. It's all done on the internet or over the telephone. And importantly, in order to make this work – because anybody can implement that – in order to make this work, their fresh thinking or innovative thinking was about the people they hired for the bank, and they found them all in the service industry. If you'd worked in traditional banking you're not able to work for First Direct. You had to come from an airline or a hotel or something to do with customer service, and that's what they were selling. They said banking is not about money, it's about a customer service. And when you think about it, that's perfectly correct.

Suzanne Robertson But the contretemps we had earlier on with – you said, 'Ah yes, well, you can't – being agile doesn't mean you are going to be innovative.' What I maintain is that if you are 'little a' agile, you are developing and applying skills that make it possible for you to say, 'OK, this is where we are now.' So that other people quickly – quickly is the key word here – so that other people can understand what you're talking about. And then if you're really agile, you say, 'We're not going any further. We're gonna question all these things. We're gonna question who's here but ah, yes, couldn't it be this and who are we going to hire to do this? Why are we really doing it in the first place?' So now you've got to have some kind of starting point because if people just brainstorm like mad, what ends up is they have lots of ideas that float around and that's that. But if you're really agile you can get a starting point quickly, something that you can build on, disagree with, replace. And that's why I maintain that 'little a' agile makes you. If you really, really have got it, it will make you more innovative.

James Robertson Well, if you think about ‘big A’ Agile and, you know, a time boxed sprint one iteration, it doesn’t really leave you time for innovation because you’ve got to – you’re committed to building a piece of software by a given date. That’s what you’ve gotta do.

Suzanne Robertson That was the core of our – of our disagreement, because you were talking about ‘big A’ is not innovative and I was talking about ‘little a’ is innovative but we were just saying agile. Consequently we had to talk it through.

Interviewer Let me try and mediate the domestic dispute in front of me for the listeners, in case they’re worried about the consequences of this interview on your harmony. I think this time boxing is an interesting issue, certainly from looking at creativity and innovation, the process of incubation, mulling over ideas, thinking things through. It’s possible that doesn’t often fit well with the need to be doing something and being seen to do something.

James Robertson Well, innovation is not a rigid process. It’s one of those bizarre things that we don’t know how long it takes to do it. We don’t know how long ideas sit around before they come to fruition. The iPad as one of the most successful consumer devices in recent history took quite a few years because it was kicked around, it changed, it morphed from one thing to the other. Some of the ideas from the iPad were put into the iPhone that was developed first and then later the iPad itself came along. These things can’t be timed, OK? Now in terms of value, the iPad is staggeringly valuable to Apple, OK? It’s also incidentally something that nobody knew they wanted and this is – just sort of take a little diversion here to let me say with our requirements, people don’t know what they want. Nobody listening to this – I’m willing to bet nobody listening to this broadcast knew they wanted an iPad before they saw one. If they’d been asked before they knew of its existence, ‘Do you need a tablet-sized device to carry around and manipulate purely with your fingers, no keyboard?’ the answer would have been ‘No’.

Suzanne Robertson But you’re talking about an invention.

James Robertson Well, that’s an invention, yeah …

Suzanne Robertson As opposed to an innovation –

James Robertson I’m using it as an illustration of ‘we don’t know what we want’. And so relying upon traditional requirements-gathering techniques of sitting down and saying to customers, sitting down and saying to the users, ‘What do you want?’ – it doesn’t work. Similarly, in ‘big A’ Agile the role of the product owner doesn’t work because one person from the business cannot possibly know what is needed. He or she may know vaguely what they think might be a solution but it’s – it’s proven to be flawed. Most clients are moving away from having a single product simply because ‘we don’t know what we want’ is coming true. Making use of looking at the larger – larger picture within the problem.

Interviewer Are you aware of any examples of agile projects that have led to innovations within that business or more widely?

James Robertson That have led to innovations?

Interviewer Yes, an agile project that’s produced an innovative outcome. It’s such an obvious question I felt I should ask it.

James Robertson I think the salvos coming from this side of the table, Suzanne –

Interviewer Well, I think we would invite listeners to email or – or social media their comments in on that one.

James Robertson By all means, and I know that –

Suzanne Robertson Yeah, in the small you can see it but I can’t isolate any big thing. Now in the small where someone says ‘oh, could you really do that?’ – I mean, it’s not considered to be an innovation, but it is an innovation because it wouldn’t have happened otherwise. Something even more convenient. Something that makes it possible for somebody to do more work. I think the problem with innovation is we always expect it to be a bolt from the blue, or from heaven, or wherever bolts come from – Usain Bolt –

Interviewer Jamaica – OK. So let’s move on. Looking more broadly, there’s been a lot of association with ‘big A’ Agile around use of some tools and techniques – KANBAN, user story cards and so on. But if you look more widely in software engineering, you’re seeing focus on governance rather than management – for example, softer people skills that are important. A focus on politics as well, nowadays, understanding the role of politics in software engineering. Do you think these are important in agile projects?

James Robertson I think they’re important in all projects. I’m not quite sure what you’re meaning by politics. Do you mean government politics or internal office politics?

Interviewer I was thinking more internal office politics – the role of politics in the target domain. If you’re seeking to change the business, there is inevitably some power relationship which will be enacted as a political action.

James Robertson Yes. I think this is leading business analysts in particular to be better politicians, because having something accepted is the most difficult thing. It’s not hard to come up with innovations. It’s selling the innovation within the organisation and having people accept that is by far the most difficult part of it.

Suzanne Robertson Yeah, but when – I’m just thinking about some of my clients where they have said ‘OK, we want to be agile’ and they’re really trying to be ‘small a’ agile. They’re really, really making an effort. Problem is that sometimes the organisations are structured in silos, but the business analysts have got to work across those silos and in order to do that they've got to be able to negotiate, because the different people in different silos have got very, very different purposes. So once again business analysts have really got to get better and better at the psychology and the politics, and all these things that naive business analysts don't realise because they haven't been trained to do that. They think that, you know, writing requirements down and building models and being – being 'big A' Agile is what you should be doing. And it's a very different – different skill.

James Robertson Well, the most important business analyst skills are not taught, unfortunately. Politics are very important. The case comes to mind of the People First Section of the Royal Borough of Kensington and Chelsea – a very innovative approach to presenting what is available to people and it's worthwhile looking at this. Unfortunately the link to People First is right at the bottom of the Royal Borough's site and it's hard to see without scrolling, but if you go there you see just what a fabulous piece of work it is. Very innovative, very non-traditional local government.

Suzanne Robertson But they had to fight. The guys who made it happen had to really – why did they fight? They had to convince people across different parts of the organisation that it was OK to be as different as they have been in their interface.

James Robertson And you call that politics. You can call it selling. You can call it diplomacy, whatever you want to call it. That was – that was what was required.

Suzanne Robertson And it's very necessary –

Interviewer Do you think the reliance of agile on communication and collaboration more than writing things down has increased the importance of those softer skills that the analyst needs?

Suzanne Robertson I think it's increased people's awareness that they're necessary. I think they've always been very, very important but people haven't really been, well, been prepared to admit it or been aware of it. I think this does make people much more aware of the need for it.

James Robertson Well, it brings it out in the open, which is important, because before the business analysts were able to hide in a cubicle and write things down and if they weren't exact then he or she could always point at other people. Well, now, with communication – with verbal communication being so important, it's very hard to hide if you're having an open discussion. And I think that's a really good thing to have come out of – one of the pretty good things to have come out of agile methods is its emphasis on –

Suzanne Robertson Well, it means that education programmes for business analysts are more and more aware that we've got to have – not just have the technical side. And you said this years ago, the sociotechnical view and that we have to educate business analysts in the socio as well as the technical.

Interviewer Has this been your change in your training of analysts to give them more ‘thinking on their feet’ skills, because it seems to me there is more immediate on-your-feet thinking –

James Robertson In my business analysis course I actually teach presentation skills, because part of the politics, diplomacy, selling, whatever you want to call it is presenting ideas to people. And people are an audience much more likely to buy in to an idea if it has been competently presented. If your presenter is standing there shuffling, mumbling and doing an appalling job then the idea is thrown out along with the – with the presenter. So all these skills of listening, of talking, of presenting – I even try and convince people that drawing is an important skill. That if you can draw a little bit better, it’s going to help you to get your ideas across. This is a very, very hard sell, I hasten to add –

Suzanne Robertson No, no, because Dan Roam, for example – on his website he focuses very much on drawing, and he’s really convinced a lot of people that you can actually learn to draw quite well even though you may not have an innate talent for it. I've done some of his lessons are really good.

James Robertson Anybody can draw. It’s just a matter of them believing they can draw in order to do it. And it doesn’t have to be great art, what I’m talking about, being a great – being another David Hockney or anything. It’s just getting an idea –

Suzanne Robertson Giving people the confidence, I think, that they can – they can do things. Its like that, what’s that wonderful Japanese Pecha Kucha where you get somebody to make a presentation that’s really constrained by time –

James Robertson Twenty slides, each slide lasting 20 seconds, and they’re automatically advanced so you’ve got three minutes and 40 seconds to get whatever it is across. So you can’t be mumbling about bullet points and so forth. It’s got to be a short, sharp presentation.

Suzanne Robertson And it really is something that people can do and they don’t think they can.

James Robertson It’s a wonderful –

Suzanne Robertson It’s a truly agile, an –

Interviewer Ultimate time box.

James Robertson Going back to drawing or sketching or if I can lead into individualising, because if you can present a diagram to somebody that visualises your idea, it’s much easier for the other person to understand it and they’re much more likely to accept it.

Interviewer So we’re moving from user stories to user storyboards?

James Robertson If you like, yes. Any visual graphic presentation is, I think is important.

Suzanne Robertson I think user stories work very well also if they're – if they're intelligently used. It's like everything, you know. It's a really good idea providing you can connect it to everything else that's going on.

James Robertson But the storyboard itself – if we go back to systemic thinking, you get a much better, broader view of what's happening if you've got a storyboard as opposed to the story card. You're much more likely to see the ramifications of anything you're doing.

Suzanne Robertson It's more dimensional, isn't it?

James Robertson It's more dimensioned and a bit of visualisation on it.

Interviewer So looking to the future, what do you see as the big challenges for agile and requirements over the next few years? Are there new trends that are gonna pose challenges? Are we not there yet and still need to improve on what we're doing in 'big A' or 'little a' agile?

Suzanne Robertson I think my biggest bugbear – and you've probably picked this up already – is business analysts have to become better at systems thinking. So that they are stronger and able to take new ideas and not be driven by them. And systems thinking has been around for a hell of a long time. It's just that we haven't really joined these things together. And I think we are starting to do it now. I look – for example, Emma Langman gave a talk at the BA Conference last year. She gave one of the keynotes and she's focused on systems thinking and it turned a lot of people's lights on. And John what's-his-name –

James Robertson Seddon –

Suzanne Robertson John Seddon gave another talk about it. And it's – they came from that discipline but the business analysts said, 'Ah yes, this is definitely a lot to do with what we do and it would really help us if we could do it better.'

Interviewer Does it reside in sprint 0 or sprint –1, or does the systemic thinking happen through the sprint cycles?

Suzanne Robertson All the way through. All the way through, yeah.

James Robertson Although I'm gonna say I think the most valuable systemic thinking happens at sprint –1 or –2 or somewhere back there, because the most valuable systemic thinking – the most valuable innovation – it all comes before you start to build software.

Interviewer We need a marathon before all the sprints –

James Robertson Absolutely.

Interviewer Post Olympic theme here during this interview.

James Robertson Something I see happening that I think is a very, very healthy thing is that the religious aspect of agile is – is fading. And people wanting to be agile but they're adopting more and more of traditional techniques. Systemic thinking is beginning to come, infiltrate into agile. Innovation is beginning to come into it, and so forth. And so I feel this is being very healthy, because there are lots of good things in agile but there are also a lot of good things that are not traditionally part of agile – or at least not part of 'big A' Agile, let's say. So the wholesale adoption of 'big A' Agile techniques is, I see it as breaking down and people are adopting a much more pragmatic approach to developing software and –

Interviewer So the future – is that how you see the future of agile development in the next five to ten years?

Suzanne Robertson Well, I hope so. I mean, I say I hope so because you never know what's going to happen and we've always had yet another paradigm, you know – Fred Brook's silver bullet syndrome. I'm hoping – and I'm an optimist, so I'm hoping that people just start building on what we already know and don't look for another miracle which stops us getting better at what – what we're doing.

James Robertson I've seen a slightly less emphasis on doing things quickly. Now this is never gonna go away. Biggest problem is that we can measure time. We have calendars, we have clocks and so that's what we measure. What is harder to measure is effectiveness and how valuable something actually is. How valuable a change is. And it comes back to something the – I think we opened our new edition of our Requirements book with. If you're gonna build software, it has to be optimally valuable to its owner. And it's this idea of value for the owning organisation which I think is slowly but inevitably becoming part of software development. Are we really producing something which is not just a quick fix for a current problem, but are we producing something which is genuinely valuable and genuinely gonna make a difference within the business? In order to be valuable it probably has to be innovative. It's very hard to think of anything being valuable if it is not an innovation, if it is not some significant change to what's here at the moment. So that's happening. I've seen that happening more and more. I think seeing more and more – more emphasis placed not on software but on business analysis itself. And I'm just using that as a word to describe not the role of the business analyst but the role of actually analysing the business and finding what the real problem is and what are we gonna do to make that business more valuable.

Suzanne Robertson And how to actually make sure that the thread from what we need to do to make it more valuable is coherent to all the people in the chain, using whatever form is appropriate to do it in that type of environment.

James Robertson	So if we take the engineering part of software engineering, it’s engineering the business, engineering the work being done, regardless of what the work is. It can be commercial work, it can be scientific, it can be academic, it can be anything at all. It’s engineering the work and the end result is usually a piece of software. But we are thinking more of the software as almost an accidental byproduct of the re-engineering of the – or the engineering of the piece of work. It’s not re-engineering. It’s engineering.
Interviewer	Exciting times ahead. Suzanne Robertson, James Robertson, thank you very much for your time.
Suzanne Robertson	Thank you, Neil. That was very interesting.
James Robertson	Thank you for having us.

Back