



# Software Engineering: TMA 01

MATTHEW MASON  
C6122243

## Contents

Question 1.....	2
(a).....	2
(b).....	2
(c).....	3
Question 2.....	4
(a).....	4
(b).....	4
(c).....	4
(d).....	4
(e).....	4
(f).....	4
(g).....	4
(h).....	4
Question 3.....	6
(a).....	6
(b).....	7
(c).....	9
Question 4.....	10
(a).....	10
(b).....	10
(c).....	10
(d).....	10
(e).....	10
Question 5.....	12
(a).....	12
(b).....	12

## Question 1

(a)

The interview with Suzanne and James Robertson focuses on understanding the motivation and rationale for using agile methods and how it can add value to the business through requirements gathering. Early adopters of agile disregarded requirements because they thought working closely with people meant they were unnecessary and they took up a large amount of time in the project to develop, however this can lead to systems being built that don't satisfy the users needs leading to a useless product. Little 'a' agile promotes understanding and investigating the problem as quickly as possible so decisions can be made about what to do to give the business the most value. They discuss how agile helps people to become system thinkers, allowing them to start thinking about requirements from the very beginning of a project, identifying a very simple model at the highest level and prioritising requirements so that they are developing for only the important things and not just producing software for the sake of it. This allow requirements to be made measurable and it helps elicit non-functional requirements that connect to the relevant bits of functionality. Finally, they establish the need for documentation in a less formal way providing there is a trail of decisions made and a shared mental model of the content. It doesn't have to be a large document, it can be sketches, videos, flip charts, simulations, etc. as long as the team has a shared understanding of what is required.

(b)

The agile method I have chosen to discuss regarding the manifesto for agile software development<sup>1</sup> is Scrum.

- Individuals and interactions over processes and tools:

By having small self-organising development teams whose members have all the general and specialised skills necessary to create a product increment, they can organise their own work which optimises communication, efficiency and effectiveness between members. The product owner is responsible for maximising the value of the development teams work and manages the clarity of product backlog items to ensure a universal understanding of the level needed for each item. Finally, the Scrum master works with the product owner, development team and those outside the organisation such as stakeholders to ensure best practices and understanding of many different aspects such as the projects scope, domain, goals, techniques, clarity, and planning scrum implementations.

- Working software over comprehensive documentation

Scrum is "A framework within which people can address complex adaptive problems, while productively and creatively delivering products of the highest possible value".<sup>2</sup> It focuses on completion of product backlog items which is a dynamic single source of requirements for the project that constantly evolves throughout the products life-cycle to capture any changes that need to happen. Items are prioritised with clearer and more defined items being taken into the sprint phase of development where the products functionality is incremented. This increment includes several other product backlog items that have been completed meaning the work done is inspectable, in a useable condition and has been thoroughly tested to ensure that all increments work together.

---

<sup>1</sup> (Manifesto for Agile Software Development, 2001)

<sup>2</sup> (Scrum Guide, 2018)

- Customer collaboration over contract negotiation

The responsibility of liaising with stakeholders is the designated to the scrum master. As requirements are likely to change throughout the development process and require reviewing, involving the customers makes it easier to adapt quickly to any changes or miscommunications due to incorrectly documented requirements during their elicitation and help determine the amount of documentation needed. This can be achieved several ways such as through user stories which describe some functionality of the system and help trigger conversations which can add more details to the requirement. This allows acceptance tests to be written and to ensure that the correct product is being developed for the customer that provides them with value to the business.

- Responding to change over following a plan

Scrum is divided up into events with the heart of scrum being the sprint, which is a defined period, normally no longer than a month, covering planning, daily scrums, development work, the sprint review and retrospective. The planning phase defines what can be produced within the upcoming sprint and how the increment will be achieved. Daily 15-minute scrum meetings allow collaboration and to inspect progress on the product backlog. It's mainly for the development team to understand and decide on the work to be carried out. Towards the end, a review is held where the stakeholders are invited to participate along with the entire team to discuss what has been completed, what problems occurred and how they were solved, what the most valuable thing to tackle next is based on the product's potential use and review the timeline and budget. These short periods allow improvements to be made and adapt to any changes before the next sprint occurs.

(500)

(c)

The Scrum method I've chosen for Question 1 is closer to 'big A' agile. James Robertson refers to Scrum as being a big 'A' agile method during the interview "The problem with agile – and I'm talking here about 'big A' agile – following a method, whether it's XP or Scrum"<sup>3</sup>. This is because there is a defined set of steps to follow throughout the method. It has a framework that should be followed, and it becomes less pragmatic and flexible in its ability to cope with changes on a day to day basis. The length of each sprint can't be altered so any response to change has to be dealt with during the next time-boxed iteration which can lead to the development of a system which doesn't end up meeting its user's needs; Robertson states that "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"<sup>4</sup>.

(189)

---

<sup>3</sup> (Episode 188: Requirements in Agile Projects, 2012)

<sup>4</sup> (Episode 188: Requirements in Agile Projects, 2012)

## Question 2

(a)

The purpose of the BillyDoo Software System is to allow users to browse and acquire tickets for free and chargeable events. Advertisers should have the ability to create events and sell or give away tickets for the events that they are organising.

(b)

The stakeholders: The customer is BillyDoo who has commissioned the new system. The main stakeholders are the clients BillDoo, users who are looking to acquire tickets for events, the advertisers who are wanting to promote their events.

(c)

The potential users of the system are the people who are interested in attending different types of events or those who are acquiring tickets for an event on behalf of someone else. The other users might be an individual looking to advertise an event they are organising or potentially a company or brand looking to sell tickets for a larger event.

(d)

The only definition which I feel would need to be included is when the description refers specially to using Paypal as a method of payment as not everyone may be aware of exactly what this is – essentially an online bank account that allows the user to pay for goods and services. All the other terms are self-explanatory and require no further defining.

(e)

The scope for the system is relatively small as it is only required to deal with a small number of interactions and external payment systems depending on which payment method the advertiser decides to use. The main interactions are users selecting an event based on four filters – event type, date, location and price – then provide a name and email to register for a free event and a valid credit/debit card to pay for chargeable events. Advertising requires users to register, fill out a form and provide a payment method. Users can also cancel an event and the system will provide registered advertisers with a history of previous events they have promoted along with some additional details.

(f)

The product should allow users to register as an advertiser

The product should allow user to filter events by any combination of event type, date, location and price

The proud should allow registered advertisers to cancel an event

(g)

The ability to edit an event should be accessible only to the user that created the event.

Category: Security Requirement

Fit Criterion: Only the events advertiser can cancel their event

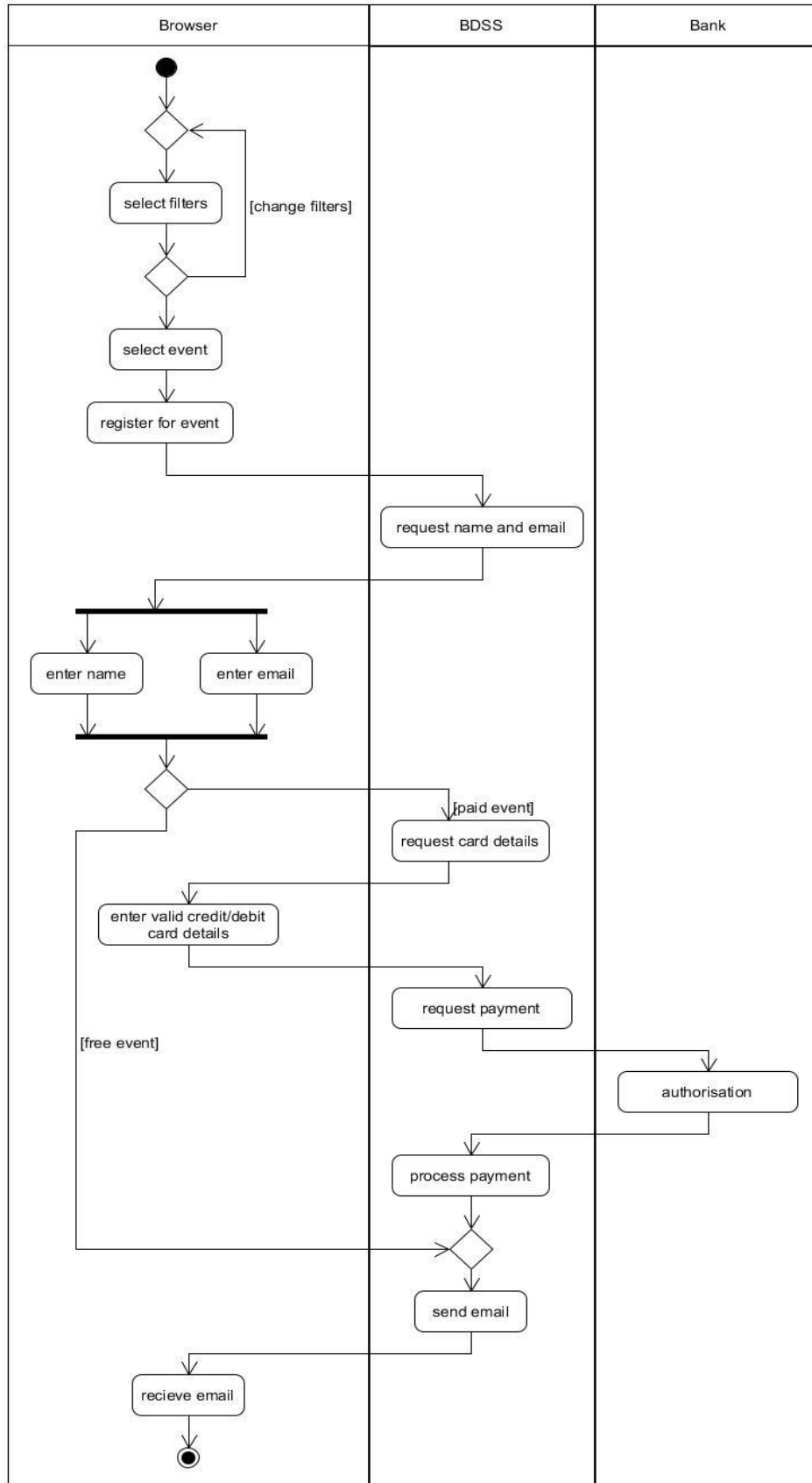
(h)

One project issue in relation to the BillyDoo system could be how cancellations from international purchases are handled regarding currency exchange. If someone visiting from another country purchased a ticket for an event which is then cancelled would they be refunded at the current

exchange rate or the one at the time the purchase was made? As refunds are handled automatically this could cause a potentially problem.


Question 3

(a)



(b)

My comments to [REDACTED]




**Matt Mason**  
25/11/19, 12:23

[REDACTED]

Good first attempt here but there are a few things here that can be improved on. Firstly, disregard everything to do with the advertising user as the question only asks us to model the acquisition of tickets for the browsing user. It would help by separating the activities into swim-lanes so it's easier to determine which actor carries out each task and add start and stop nodes to make it clear where the process begins and ends. To help with the flow and understanding, any activity that has two transitions coming from it can be replaced with merge nodes. Lastly, it doesn't make it clear from the description that the user needs to register to get tickets for an event so I'd rework this section and add a bit more detail to the paid event path to clarify what happens, like providing valid credit/debit card details.

Good luck!

Matt

[Reply](#)  [Like comment](#) [Delete comment](#)



My comments to [REDACTED]



**Matt Mason**  
25/11/19, 12:47

Hi [REDACTED]

Really good first attempt you've made here. It's clear who is carrying out each activity and it's easy to follow. I'd only make a few suggestions for the revised model which is to expand on some details without making it overly complex. When selecting filter options, you could use a fork to separate the activities and use a merge node that loops back round giving the browser the option to change filters if they can't find a suitable event. I don't think the two transitions heading to "create ticket reservation" is valid so another merge node might be needed here. Lastly, from the description it appears a browser is sent an email regardless of whether payment was successful or not so you could rework this to make it a little bit simpler. Hope the feedback helped and good luck with your revised model.

Matt

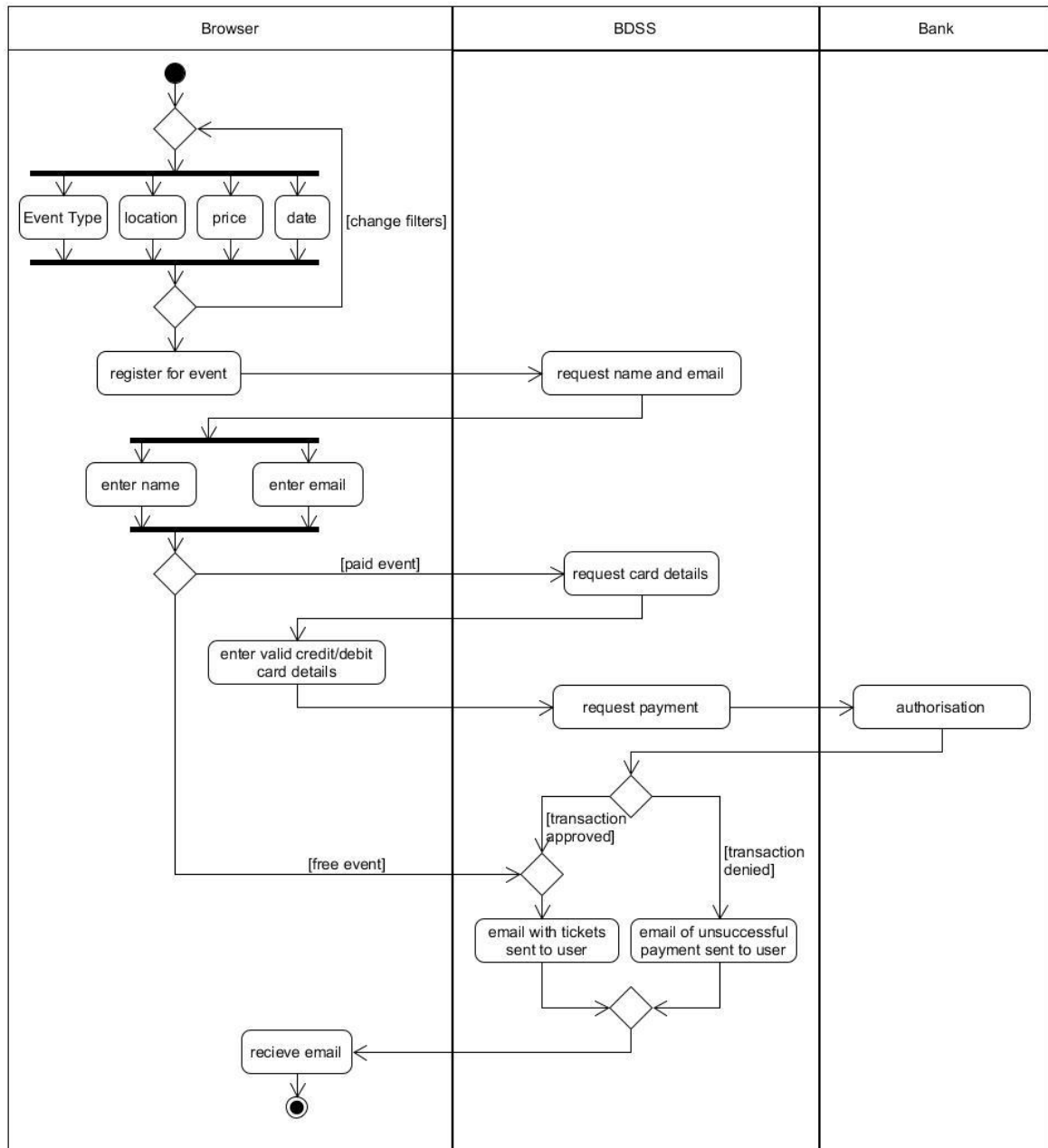
Reply



<sup>1</sup> Like comment

Delete comment

(c)



27/11/19, 15:18

Maximize

### My updated model

I improved my initial model by adding concurrent activities for selecting filters which implies that the browser doesn't have to change every filter option should they not be able to find a specific event. They can simply go back and change only one of the filters if need be before registering for the event. The other improvement I made was by showing the different transitions that can occur when the authorisation of payment is successful or if the payment is unsuccessful as both outcomes are specifically mentioned in the description. This also helps to add some clarity about what can happen over my initial model which simply showed an email being sent to a user regardless of the outcome.

## Question 4

(a)

Here are three different actors that could be potential roles for the system.

Browser

Advertiser

ExternalPaymentSystem

(b)

Here are a number of use case names that could be a part of the system.

Browse events

register for event

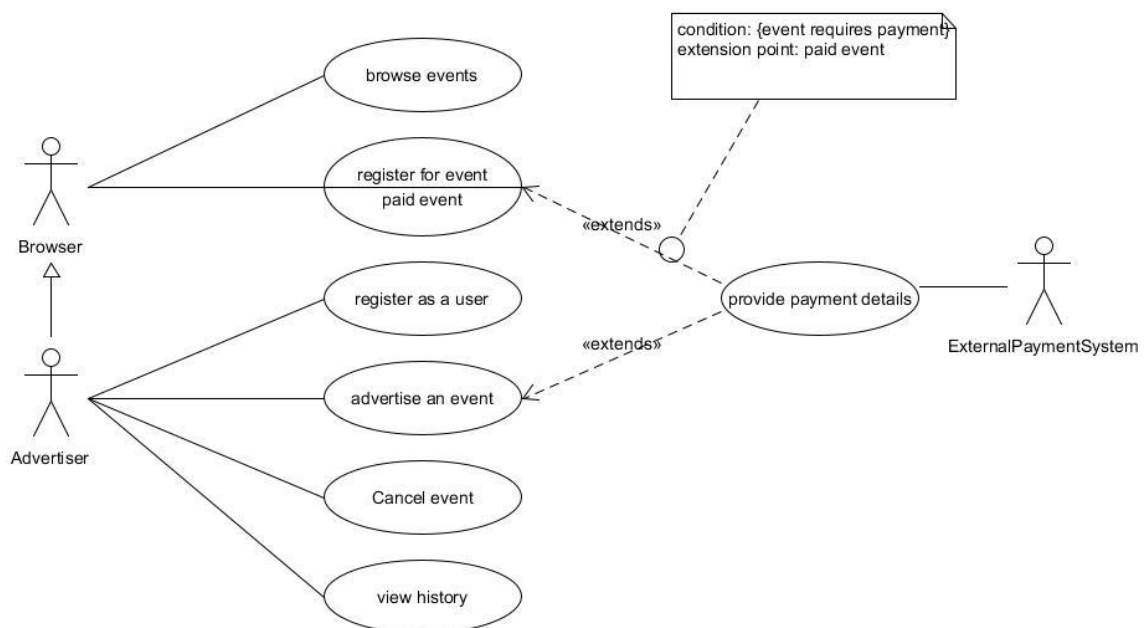
register as user

advertise an event

cancel event

view history

(c)



(d)

In my opinion the functioning core should be the ability to create an event. Without being able to do this, many of the other tasks such as browsing and registering for events can't be completed. The next chunk should be the ability to browse events using the filter options to display different events and after that the ability to register for a free event as the code for this could be potentially reused when developing the activity to register for a paid event reducing the amount of work needed to be done and allowing a quick turn around time.

(e)

The first user story that I would consider to be of high priority would be the following:

“As an advertiser, I want to be able to promote a range of different paid events across the country” . The reason I would consider this high priority is because without having any events created, other functionality can't be tested such as registering to attend an event. It also opens up the conversation regarding what needs to be included when creating an event such as being able to specify a venue, date or other field that needs to be included. The second user story that could be considered high priority could be as follows:

“As a user I want to be able to find an event that is within 30 minutes travel of where I live” . The reason this could be a high priority is to test that the core function of searching for an event works by returning the correct results based on the filters chosen. This again open up the conversation about what filters to include as options when searching such as location, event type, location, price and even potentially other ways to search for events.

## Question 5

(a)

The BDSS can be developed in an agile way due to several reasons. It's a simple and fast method that can adapt to changes through iterative and incremental development which allows for continuous assessment reducing risks. It emphasises a collaboration between the team and client to help understand the requirements of the system and helps to produce software that is valuable to the user. This helps reduce problems of communication and due to the size of the project it allows various techniques to be used that can adapt to change rather than following a strict method. This way of developing can produce artefacts such as use cases, user stories, scenarios, models, and activity diagrams, providing each one is gathered with a purpose. These artefacts help with traceability which in turn allows the system to be maintained throughout its life even by those who were not part of the original development of the system. The directors would need to allow the opportunity for regular discussion of ideas and to be able to free up resources for feedback on the progress through iterations of the development.<sup>5</sup>

(b)

Here are three functional requirements that can be derived from the use case:

SFR1, UC9, Step 2

Description: The system shall accept a valid email address

Fit Criterion: A valid email address shall be accepted

SFR1, UC9, Step 2

Description: A valid password should be used

Fit Criterion: The system shall accept a valid password that must contain a capital letter, a number and be between 6 – 12 characters long

SFR1, UC9, Step 2

Description: The system shall accept a valid name

Fit Criterion: a valid name that consists only of letters shall be accepted

---

<sup>5</sup> (Principles behind the agile manifesto, 2001)