4M21 Software Engineering and Design 2025/2026

Solutions

Version: EP3/JD3

Question 1

(a)

We need the following operators for each method. Names, interpretations and estimates may differ provided they are reasonable and motivated.

Pen operators: orient device (OD) to take photo, long press (LP) on the pens button, short press (SP) on the pens button, wait for audio beep (WB), speak query (SQ), wait for system response (WSR), wait for response readout (WRR).
OD: 3.0s (approximate time to orient device to take photo of target object)
LP: 1.0s (long press duration)
SP: 0.5s (short press duration)
WB: 0.2s (assuming a very brief distinct beep)
SQ: 1.0s (assuming a short query such as "What is this?")
WSR: 1.2s (assuming combined latencies in processing, networking and the chatbot itself)
WRR: 5.0s (assuming a response from the chatbot of two to three sentences)

Procedure: OD + LP + WAB + SP + WAB + SQ + WSR + WRR = 3.0 + 1.0 + 0.2 + 0.5 + 0.2 + 1.0 + 1.2 + 5.0 = 12.1s.

App operators: find icon among n icons FI(n), select icon (SI), select add image button (SAI), select new photo button (SNP), orient device (OD), select camera shutter (SS), select new voice query button (SNQ), speak query (SQ), wait for system response (WSR), wait for response readout (WRR).
FI(n): 1.2s (FI(n) = In based on a linear function of the number of preceding icons on the home screen. Let us assume there are 4 preceding icons and I is 0.3s.)
SI: 1.0s (select icon on home screen)
SAI: 1.0s (touch on the 'add image to chat' button)
SNP: 1.0s (touch on the 'new photo' button)
OD: 3.0s (assume that it takes approximately the same amount of time to orient the smartphone as it does the pen)
SS: 1.0s (touch on the camera shutter button to take photo)
SNQ: 1.0s (touch on the 'new voice query' button)
SQ: 1.0s (assume same as for the pen)
WSR: 1.2s (assume same as for the pen)
WRR: 5.0s (assume same as for the pen)

Procedure: FI(n) + SI + SAI + SNP + OD + SS + SNQ + SQ + WSR + WRR = 1.2 + 1.0 + 1.0 + 1.0 + 3.0 + 1.0 + 1.0 + 1.0 + 1.2 + 5.0 = 16.4s

[40%]

(b)

The limitations of this analysis are that it: (1) assumes error-free behaviour from both the user and the system (for example, any failure to correctly process the users speech is ignored); (2) assumes optimal user behaviour; and (3) operator estimations are just heuristics and do not accurately reflect the precise time incurred, nor do they have a notion of uncertainty (variance).

[10%]

(c)

Between-subjects design participants are only exposed to one condition in the experiment

Within-subjects design participants are exposed to all conditions in the experiment

[10%]

(d)

A controlled user study could be conducted where an assigned query task is performed by the participants. The independent variable is the method (pen or app) and the dependent variable is the task completion time. For a within-subjects design, it would be appropriate to balance the order in which participants experience the two different methods, e.g. half will do pen first, half with do app first. Depending on how the task is constructed, it may also be necessary to also vary the object assigned so that the answer is not already known to the participant. For a between-subjects design, participants will only use one of the methods but since statistical power is lower, more participants may be required to determine whether there is a statistically significant difference between the methods. In both designs, it would be sensible to have the participant perform the task multiple times with different objects in order to accurately estimate typical performance and account for inconsistency in task execution.

[20%]

(e)

Some form of field research may be used to capture how users will utilise the pen within their own settings. This would ensure that the influence of the users own environment (including the physical and social context) on how and when they might use the pen is considered. The researcher may sit in the same room as the participant to observe and take notes. Other research methods are of course possible. As long as the motivation for the proposed method is sound and accurate, the answer will be acceptable.

[20%]

Question 2

(a)

Mental workload The mental workload can be assessed by asking the user to rate their perceived workload, by measuring the time taken to respond during a drill (an intentionally created anomaly incident), or by taking regular measurements of blood pressure, etc.

Situation awareness Good situation awareness is important so that the user can act swiftly and correctly when an incident is identified. Situation awareness can be studied by examining gaze patterns provided by the VR headset.

Complacency Complacency can be studied by tracking missed incidents over time or by analysing the frequency of active user interactions with the interface.

Skill degradation Like complacency, skill degradation has to be studied over time. Skill degradation may manifest as over reliance on the anomaly detection function. It can be assessed by intervention studies that test users' ability to identify incidents without the automated assistance.

Automation reliability The ability of the anomaly detection system to correctly identify incidents can be evaluated in studies with representative security feed data or by carrying out drill experiments in the deployment context and tracking the performance of the system.

Costs of decision and action outcomes The risk of incorrect user actions, whether leading to delays in incident response or the overuse of alarms, can be studied by engaging in risk analysis.

[30%]

(b)

Mistakes Mistakes are due to the formation of an incorrect intention. In this particular context, an untrained user might be actively monitoring a different camera feed and experience the pop-up dialog as a nuisance that disrupts their focus. The user may then make a mistake and press the dismiss button without carefully considering the dialogs purpose.

Slips Having formed an intention, the user may experience a *slip* when they fail to carry out the action correctly that executes this intention. In this particular context, a slip might occur when the user reaches to press the Dismiss button but accidentally presses the Raise Alarm button. To help mitigate the negative consequences of such slips, the Raise Alarm button would likely require subsequent confirmation.

[10%]

(c)

Fitts Law can model the relationship between movement time and index of difficulty in mid-air selection. Index of difficulty is defined by the movement amplitude or distance, and the size of the target (in this case, the video feed the user is trying to select).

[15%]

(d)

In designing the visual interface it is important to consider:

Field of view Influences how large the video feed grid can be and how far away it should be placed.

Contrast Influences what colours might be used to either draw attention, or avoid drawing attention. Choice of contrast levels may also be important in and between the feeds themselves to ensure details can be perceived. Contrast between font and background colour is also important for text legibility.

Foveated vision Influences how feeds are placed or cycled through slots in the grid given that the user is less likely to perceive changes or details in the periphery.

[15%]

(e)

Eye tracking could be used to log the trace of the users gaze. This log could be analysed to produce a map of visual saliency to understand which features attract visual attention. The gaze trace could also be examined to understand what strategy the user follows when monitoring the feeds in the grid, e.g. does the user sequentially look at each video feed or does the gaze wander randomly over the grid.

[10%]

(f)

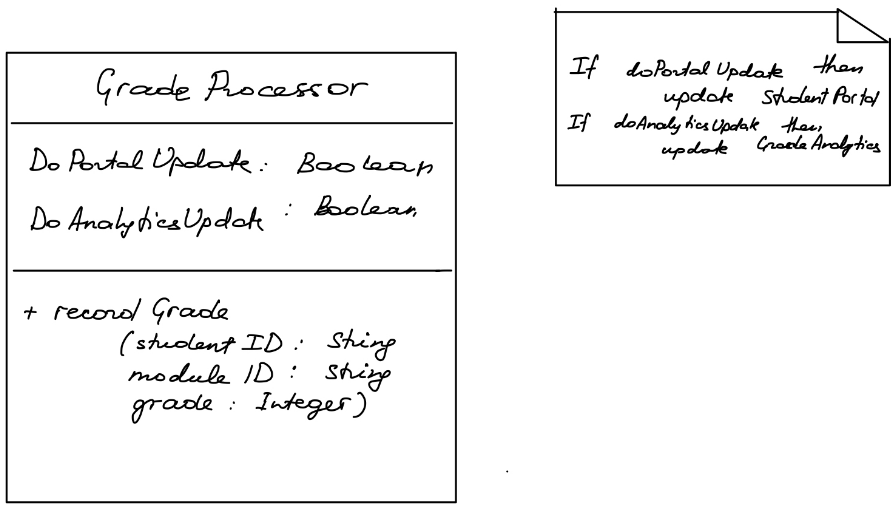
A key risk here is that the human makes an error as a result of their use of the anomaly detection function and fails in some way. This failure could take various forms including a failure to detect an actual anomaly due to an overreliance on the automation (false negative) or the overuse of alarms (false positive) due to overconfidence in the automation. Failure Mode and Effects Analysis (FMEA) is a suitable choice as it can be used to analyse human error at both the individual and the team level. Other risk assessment methods are of course possible. As long as the motivation for the risk assessment method is accurate the answer will be acceptable.

[20%]

Question 3

(a)

One of the possible solutions is presented below.



[10%]

b)

Advantages:

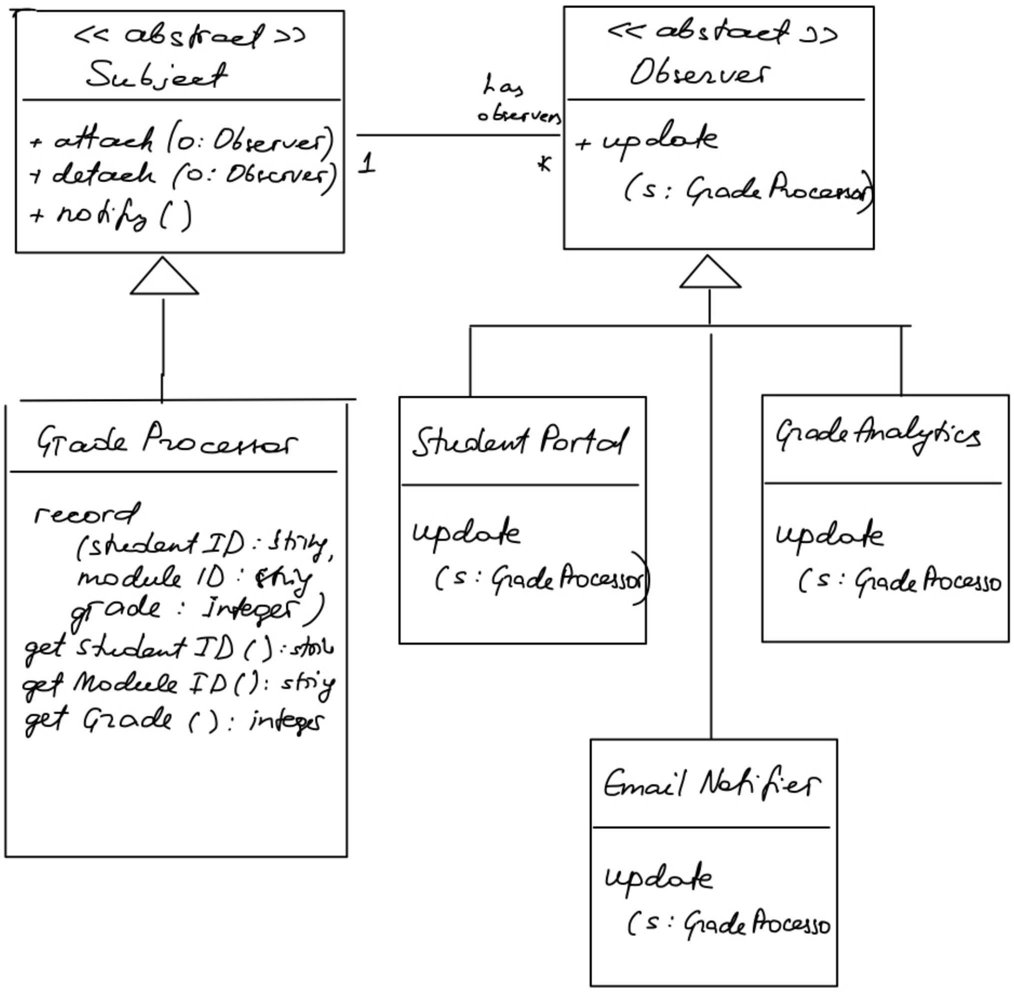
- **simple and easy to understand** - all logic and actions are in one place;
- **quick to implement**.

Disadvanatages:

- **tight coupling**: *GradeProcessor* depends on *StudentPortal* and *GradeAnalytics*, any changes to those classes may impact *GradeProcessor*;
- **mixed responsibilities**: *GradeProcessor* is responsible not only for recording grades but also for coordinating multiple other actions (UI updates, analytics);
- **poor scalability**: adding new actions requires modifying *GradeProcessor*; as more actions are added more and more flags/branches are added - harder to maintain;
- **harder testing**: unit testing would involve multiple external components;
- **limited functionality**: boolean flags do not support richer functionality (conditional rules for example).

[20%]

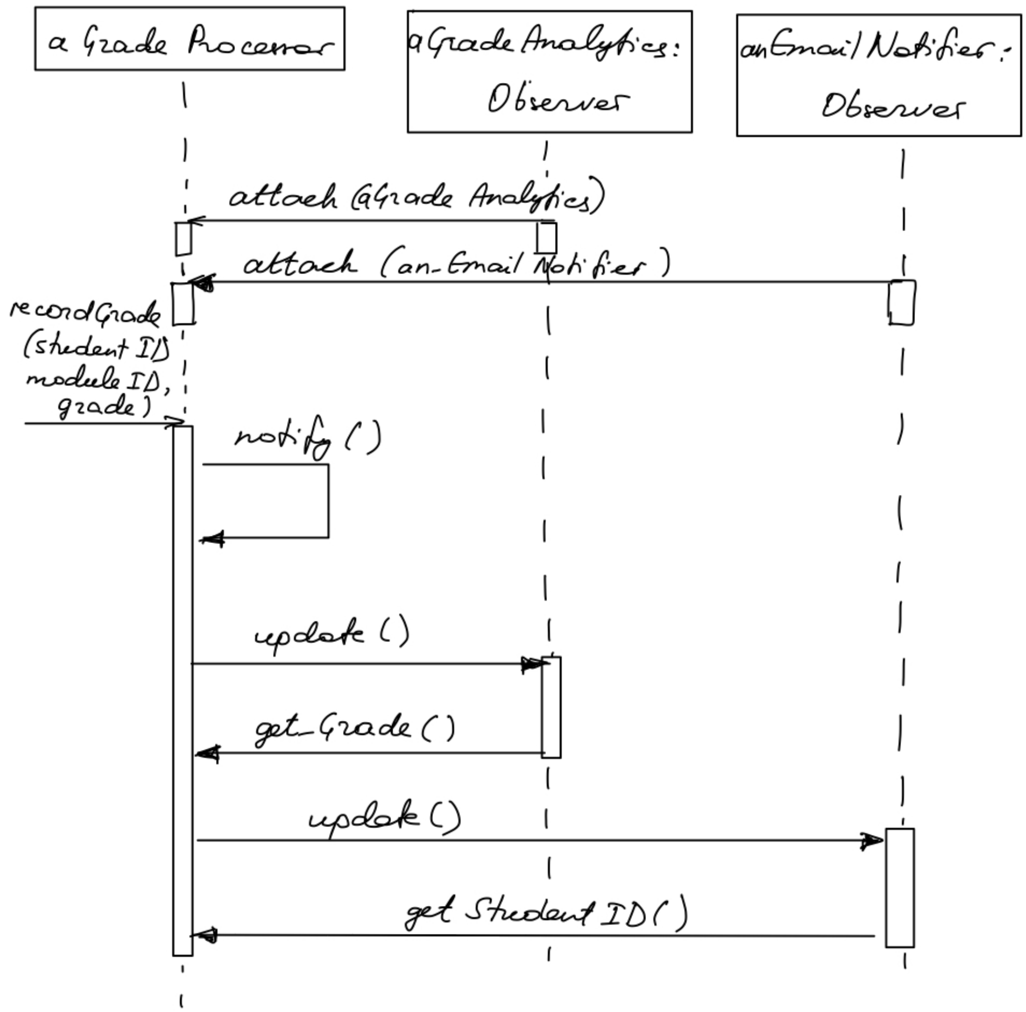
c) (i)



<< abstract >>
Subject
+ attach (o: Observer)
+ detach (o: Observer)
+ notify ( )
has
observers
1
*
<< abstract >>
Observer
+ update
(s : GradeProcessor)
Grade Processor
record
(student ID : string,
module ID : string
grade : Integer)
get Student ID ( ) : string
get Module ID ( ) : string
get Grade ( ) : integer
Student Portal
update
(s : GradeProcessor)
GradeAnalytics
update
(s : GradeProcesso
Email Notifier
update
(s : GradeProcesso

[40%]

c) (ii)



[20%]

c) (iii)

- **reduced tight coupling**: *GradeProcessor* no longer depends directly on *StudentPortal*, *GradeAnalytics* and *EmailNotifier*;
- **separates responsibilities**: *GradeProcessor* is for recording grades; each other action is handled by a corresponding Observer;
- **scalability**: new actions can be added by creating new Observer classes and registering them without modifying *GradeProcessor*;
- **easier testing**: each Observer can be tested independently;
- **clean design**: no long chains of 'if' statements.

[10%]

Question 4

One of the possible solutions is presented below.

(a)

Functional Requirements:

- **Real-time shipment tracking**: the system should allow users to view the current location and delivery status of shipment in real time using tracking data from delivery vehicles.
- **Logistics Performance analysis**: the system should provide reports and summaries so that delivery performance can be analysed, such as delivery times, delays, route efficiency.
- **Multiple user roles support**: the system should enable different features for different roles access.

Non-functional Requirements:

- **Scalability**: the system should handle growing number of users, shipments, delivery vehicles.
- **Reliability**: the system should provide continuous access in real time.
- **Security**: the system should protect user data through authentication, encrypted communication.

[20%]

(b)(i)

Product owner: represents voice of the customers and of the product stakeholder (to ensure that user stories deliver maximum business value).

Development team: a group responsible for designing, implementing, testing, and delivering increments of the shipment tracking system.

Scrum master: facilitates the development team to deliver sprints according to Scrum principles in sprints on budget and on time.

[15%]

(b)(ii)

Scrum is one of the agile frameworks that could be applied to a software development project.

- Requirements can be refined more and more as customers, drivers and managers provide continuous feedback, they do not have to be fixed at the start - **supports evolving requirements**.
- The approach supports **frequent** user involvement and stakeholder **feedback** through short development iterations, regular reviews ensure real needs are met and business value is delivered.
- Incremental delivery **reduce risk** as issues can be identified early. Core features can be **delivered** and validated **early**.

[15%]

(c)

User stories are derived by breaking higher level functional requirements into small, user focused features. Each user story is written from perspective of a specific user wanting to have a specific feature to deliver certain business value. For example,

- As a customer I want to view the real-time location of my parcel so that I can plan.
- As a delivery driver I want to update the delivery status of the parcel when it is delivered.
- As a manager I want to view analytics report so that I can identify bottlenecks and improve the efficiency.

[15%]

(d)

User stories can help to improve testing process. Among others,

- Each user story defines a specific piece of functionality - helps to identify what needs to be tested (and what is not).
- Testing is performed from perspective of real users, i.e. customers, drivers and managers, ensuring meaningful testing.
- Helpful for acceptance testing.
- Well defined user stories can be converted into automated tests; thus enabling continuous integration and delivery.

[15%]

(e)

Functional requirements tend to concentrate on what the system does, whereas non-functional requirements concentrate on how the system performs so the challenges are

- could be difficult to define something measurable for scalability, reliability, security.
- dependance on real-works conditions that are not currently present for scalability and reliability (peak delivery periods or high user volume)
- dependance on interactions with other components (for example, GPS service for tracking) - difficult to isolate
- security vulnerability testing may require specialised expertise and tools

To address them one could

- specify clear performance metric, for example, response time;
- simulate peak delivery periods or high user volumes
- perform performance testing, stress testing and testing under load;
- test in real-life conditions to obtain realistic results;
- testing under failure conditions to support reliability;
- use security experts and specialised tools for security vulnerability analyses to verify authentication protocols, scan for vulnerabilities, address privacy concerns (knowing whereabouts of employees).

[20%]