UNIVERSITY EXAMINATIONS

UNISA university of south africa

September/October/November 2021

COS3711

Advanced Programming

80 Marks

Duration 2 Hours

EXAMINERS:
FIRST: DR CL PILKINGTON
SECOND: MR K HALLAND
EXTERNAL: DR L MARSHALL (UNIVERSITY OF PRETORIA)

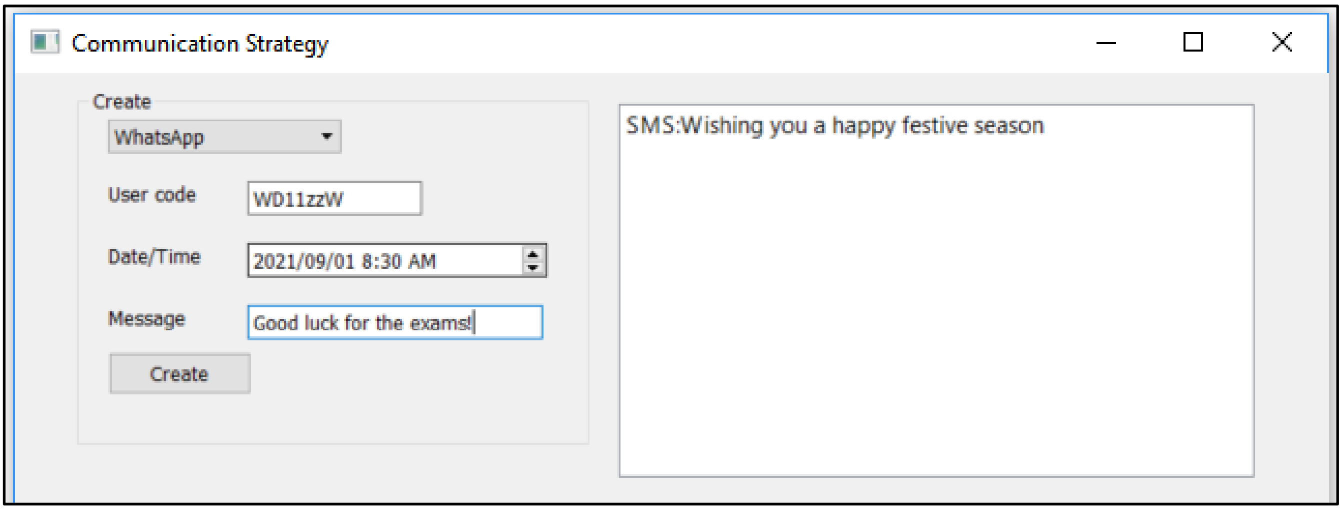
This paper consists of 5 pages.

Instructions

1. You may type your answers in a word processor (and then print to PDF for submission) or handwrite your answers (and then scan to PDF).
2. Answer all questions. Please answer questions in order of appearance.
3. The mark for each question is given in brackets next to each question.
4. Note that no pre-processor directives are required unless specifically asked for.
5. Students must upload their answer scripts in a single PDF file (answer scripts must not be password protected or uploaded as "read only" files)
6. NO emailed scripts will be accepted.
7. Students are advised to preview submissions (answer scripts) to ensure legibility and that the correct answer script file has been uploaded.
8. Students are permitted to resubmit their answer scripts should their initial submission be unsatisfactory. However, only three submissions are allowed.
9. Incorrect file format and uncollated answer scripts will not be considered.
10. Incorrect answer scripts and/or submissions made on unofficial examinations platforms will not be marked and no opportunity will be granted for resubmission.
11. Mark awarded for an incomplete or illegible scanned submissions will be the student's final mark. No opportunity for resubmission will be granted.
12. Submissions will only be accepted from registered student accounts.
13. Students who have not utilised invigilation or proctoring tools (IRIS) will be subjected to disciplinary processes.
14. Students suspected of dishonest conduct during the examinations will be subjected to disciplinary processes. UNISA has a zero tolerance for plagiarism and/or any other forms of academic dishonesty.
15. Students are provided one hour to submit their answer scripts after the official examination time. Submissions made after the official examination time will be rejected by the examination regulations and will not be marked.
16. Students experiencing network or load shedding problems are advised to apply, together with supporting evidence, for an aegrotat within 3 days of the examination session.
17. Students experiencing technical problems should contact the SCSC on 080 000 1870 or email Examenquiries@unisa.ac.za or refer to Get-Help for the list of additional contact numbers. Communication received from your *my*Life account will be considered.

Remember to complete the Honesty Declaration when submitting your answers. By submitting your answers you are confirming that your submission is your own, unaided work.

All businesses need to manage their communication, and the simplified scenario used in this examination is based on communication strategies. The idea is that a user can select a type of communication channel and have a message sent out on that channel at some specific point in time. The application itself allows the user to set up the required options and just the strategy and message will be displayed for the purposes of a brief history of messages sent – See Figure 1.


Communication Strategy

Create

WhatsApp

User code: WD11zzW

Date/Time: 2021/09/01 8:30 AM

Message: Good luck for the exams!

SMS:Wishing you a happy festive season

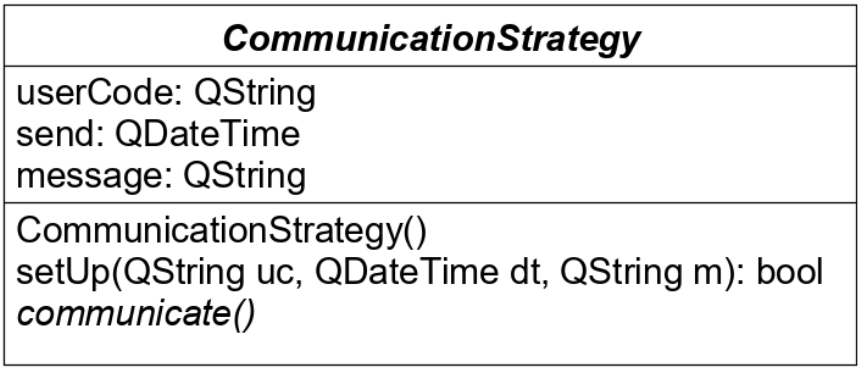
Figure 1. App window.

Question 1 **[15 marks]**

Apply the strategy design pattern to the scenario above that would also adhere to the following requirements.

1. There are currently 3 basic strategies/channels for communication: Signal, SMS, and WhatsApp. Your design should include a strategy for each channel.
2. All that is needed to create one of these is a type name.
3. Once an appropriate type has been created, a `setUp()` function would be used to populate object instance data members, and the `communicate()` function used to process the required communication.

A `CommunicationStrategy` UML class diagram has already been drawn up – see Figure 2. Note that you may abbreviate this class name to `ComStrat` in your answers.


CommunicationStrategy

userCode: QString
send: QDateTime
message: QString

CommunicationStrategy()
setUp(QString uc, QDateTime dt, QString m): bool
communicate()

Figure 2. UML class diagram

1.1 Considering the scenario given and design decisions listed above, draw a UML class diagram for the strategy pattern. You should include the necessary classes, class attributes, class constructors and operations, and class relationships to make it clear you understand how data will be managed and passed between classes. Label all relationships indicated; however, you do not have to include the Client/GUI class nor indicate access specifiers.
[You may use a software tool to create the UML class diagram. If you are hand drawing this answer, you can use an underline to indicate italics in the UML class diagram.] (11)

1.2 The provided UML class diagram included some of the text in italics. What does this indicate? (1)

1.3 Describe what the function of the `communicate()` function would be in this design pattern? (2)

1.4 The decision has been taken to initially host this application using cloud computing. Would you recommend a public or private cloud model? (1)

Question 2 **[33 marks]**

This question relates to the code for creating and setting up of the required strategies.

The `setUp()` function in the `CommunicationStrategy` class (see Figure 2) is used to populate the data members in an instance object. This function does the following.

- It first checks that the user code meets a particular pattern.
- If the user code does not meet requirements, the function returns `false`. If the user code does, the function returns `true`.
- Additionally, if the user code meets requirements,
  - all data member values are populated, and
  - a signal (`created()`) is emitted that sends the communication strategy type and the message (separate by a colon) as a `QString`. See the viewing window in Figure 1.

2.1 Write the code for the class definition for the `CommunicationStrategy` class (that is, what would be in the header file). Ensure that all functionality shown in Figure 2 and that given above is included. (7)

2.2 Write the regular expression that can be used to check whether a string meets the following requirements:

- The first character should be a G, K, or W (for Gauteng, KwaZulu-Natal and Western Cape).
- The second character could be any uppercase alphabetic character.
- This should be followed by one or more lowercase alphabetic characters or digits.
- The final character should be the same as the first character. (6)

2.3 Write the code for `bool CommunicationStrategy::setUp(QString uc, QDateTime dt, QString m)`. Use the information you have so far in this question to guide you, where the regular expression in 2.2 will be used to check the user code.

Note the structure of the signal sent out: “Strategy:message”. Without using a data member within the class, the type of strategy or class (Signal, SMS, or WhatsApp) should be indicated before the colon. (9)

2.4 The signal that is emitted by `setUp()` is picked up by the GUI/client code and used to populate the viewing window in Figure 1. What various widgets could be used to hold this information, explaining what would need to be in place to make such use possible? (6)

2.5 The signal that is emitted by `setUp()` could be seen as an implementation of which design pattern? Explain why you say so. (3)

2.6 Based on the `CommunicationStrategy` class set up in 2.1, would the following code be legal? Explain why or why not. Note that marks are allocated only for the reason given.

```
// assume cs is a pointer to some valid CommunicationStrategy
object
cs->setProperty("priority","high");
```

(2)

Question 3 **[32 marks]**

This question relates to the writing of the data in the viewing window to an XML file.

Assume that the data in the viewing window can be stored in `QStringList list` for use in the writing-to-XML process, where each line consists of the string of characters as seen in the viewing window in Figure 1. This `list` must be passed to an object instance of a class that should be run as a thread.

The XML file should take the following format.

```
<communications>
 <communication channel="Signal">
  <message>Remember to get your COVID19 vaccination.</message>
 </communication>
<communication channel="SMS">
  <message>Wishing you a happy festive season.</message>
 </communication>
<communication channel="WhatsApp">
  <message>Good luck for the exams!</message>
 </communication>
</communications>
```

3.1 The following code has been provided for the class header file. However, this is not considered good practice. Change the code so that it meets what is considered good practice and will support being run as a thread.

```
class WriteToXML: public QThread
{
public:
    WriteToXML(QStringList s);
    void doWrite();  // does the write to file in XML
private:
    QStringList strlist;
}
```

(3)

3.2 Assuming that the string list passed to `WriteToXML` has been saved to `strList` in the class constructor, write the code for the `doWrite()` function so that each line in the string list is written to file in the format above, using DOM. A portion of the code is given below and you have to add the missing parts in the appropriate places.

```
void WriteToXML::doWrite()
{
    QDomDocument doc;

    foreach(QString s, strlist)
    {

    }

    QFile file("data.xml");
    file.open(QIODevice::WriteOnly);
    file.close();
}
```

(16)

3.3 Write the code that would be included in the GUI/client code that would run an instance of this class as a thread. There is no output that needs to be handled as the output is written to file. Assume that no object instances have been created yet. (6)

3.4 When running the code that had been set up by a novice, the IDE gave the message "*QThread: Destroyed while thread is still running*" and no data was written to file.

Explain what the problem could be and how to solve it. (5)

3.5 Suppose that this XML file were to be read in to another application. One developer noted that as DOM was used to write the file, DOM will have to be used to parse it again. Comment on this statement. (2)