UNIVERSITY OF ESWATINI

CSC411 – MINI PROJET

MDUDUZI COMFORT – 202004888

MSIMISI MATSE – 202003895

Using socket programming to implement the Producer-Consumer Problem involves creating a server (producer) and client (consumer) communication model. The server will generate random student information, write it to XML files, and send the file numbers to the client over a socket. The client will receive the file numbers, read the corresponding XML files, process the student information, and calculate the average marks and pass/fail status.

Implementing the server (producer) and client (consumer) using Python's socket module:

Server (Producer):

```
import socket

import random

import xml.etree.ElementTree as ET

import os

import time


# Function to generate random student information

def generate_student_info():

    # Code to generate random student information...

    # (Same as previous implementation)


# Function to write student information to an XML file

def write_to_xml(student, file_path):

    # Code to write student information to an XML file...

    # (Same as previous implementation)


if __name__ == "__main__":

    server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

    host = "127.0.0.1"

    port = 12345

    server_socket.bind((host, port))


    server_socket.listen(1)

    print("Server is listening...")
    connection, address = server_socket.accept()
    print(f"Connected to {address}")

    for i in range(1, 11):
        student_info = generate_student_info()
        file_path = f"student{i}.xml"
        write_to_xml(student_info, file_path)
        print(f"Produced: {file_path}")

        connection.send(str(i).encode())  # Send the file number as a string

        time.sleep(random.randint(1, 3))

    connection.close()
    server_socket.close()
```

Client (Consumer):

```
import socket
import xml.etree.ElementTree as ET
import os

# Function to read student information from an XML file
def read_from_xml(file_path):
    # Code to read student information from an XML file...
    # (Same as previous implementation)

# Function to calculate average mark and pass/fail status
def calculate_average(student):
    # Code to calculate average mark and pass/fail status...
    # (Same as previous implementation)
```

```
if __name__ == "__main__":
    client_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    host = "127.0.0.1"
    port = 12345

    client_socket.connect((host, port))

    while True:
        file_num_str = client_socket.recv(1024).decode()

        if not file_num_str:
            break

        file_num = int(file_num_str)
        file_path = f"student{file_num}.xml"

        student_info = read_from_xml(file_path)
        os.remove(file_path)
        print(f"Consumed: {file_path}")

        average, pass_fail = calculate_average(student_info)
        print(f"Student Name: {student_info.name}")
        print(f"Student ID: {student_info.student_id}")
        print(f"Programme: {student_info.programme}")
        print("Courses and Marks:")
        for course, mark in zip(student_info.courses, student_info.marks):
            print(f"  {course}: {mark}")
        print(f"Average Mark: {average:.2f}")
        print(f"Pass/Fail: {pass_fail}")
        print("\n")

    client_socket.close()
```

In this implementation, the server (producer) and client (consumer) communicate over the local network using TCP sockets. The server listens on IP address "127.0.0.1" (localhost) and port 12345. The client connects to the server's IP address and port. When the client connects, the server starts producing random student information and writing it to XML files. It sends the file numbers (as strings) to the client using the socket connection. The client receives the file numbers, reads the corresponding XML files, processes the student information, calculates the average marks, and determines pass/fail status.

Please ensure that you run the server (producer) first before running the client (consumer) to establish the socket connection successfully.