A Report on

Delayed Call Telephone Simulation

Submitted for the Course Requirement of

Simulation And Modelling

Bachelor of Engineering in Software Engineering

under Pokhara University

Submitted by:

Pradip Dhungana, 201751

Date :

28 January 2024

Department of Software Engineering

NEPAL COLLEGE OF INFORMATION TECHNOLOGY

Balkumari, Lalitpur, Nepal

TABLE OF CONTENTS

1 Introduction

Telephone call simulation is a technique that allows users to practice and test various scenarios of voice communication over the internet. It can be used for training purposes, such as improving customer service skills, or for entertainment purposes, such as creating realistic voice effects. One of the challenges of telephone call simulation is to model the behavior of a telephone network, which consists of lines, incoming calls, and call connections. In this project, we aim to simulate a telephone network for delayed call systems.

1.1 Delayed Call System

A delayed call system is a feature that allows users to call a number or address that is not answered immediately by an agent or device. Instead, the calls which couldn't be connected is sent to a queue or waiting list, where it wait for lines to get free, and then connect once a line is free. This feature can improve customer satisfaction by reducing wait time and frustration, as well as increase agent productivity by reducing downtime and workload.

1.2 Overview

This project aims to simulate a telephone network with a delayed call system. It involves modeling the behavior of a telephone network including lines, incoming calls and connections which is expected to delay calls. The project will look into how unanswered calls can be placed on queue until there is an available line for them to go through. The goal with this undertaking is to acquire knowledge on the effectiveness and benefits of implementing delayed call systems in the telephone network.

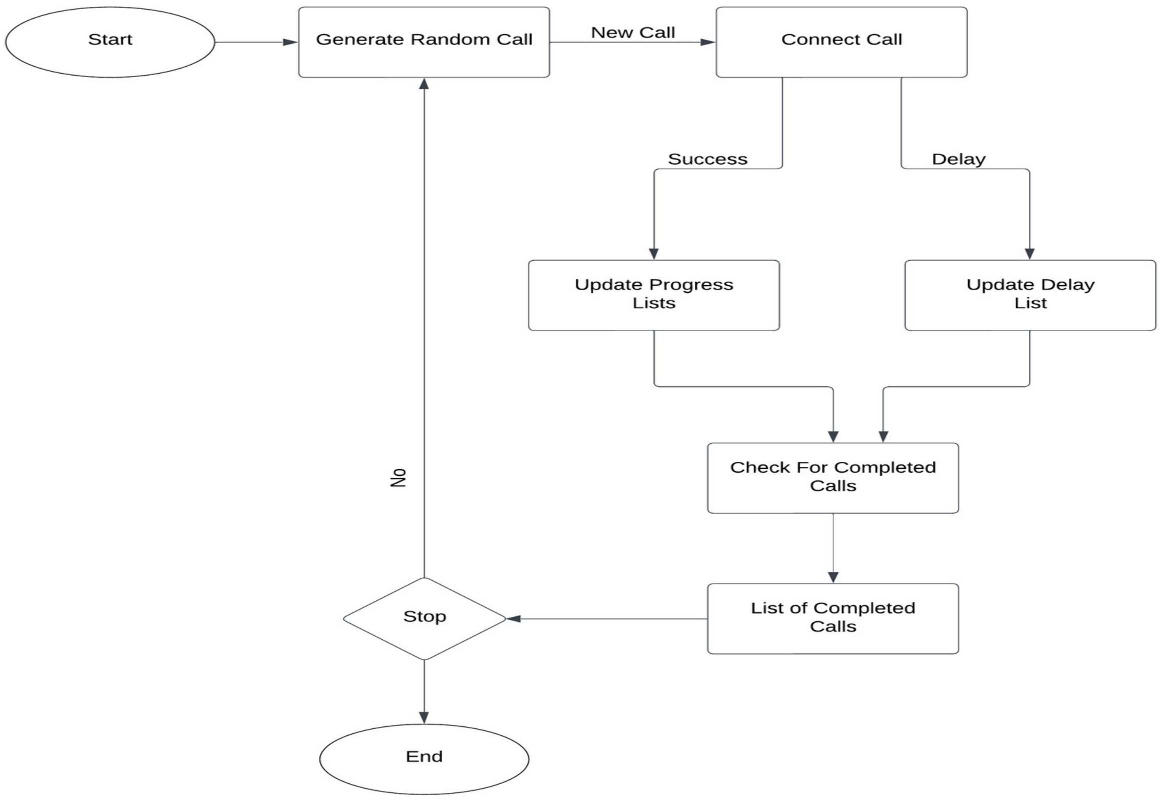
2 Objective

The main objectives of this project are :

1. To simulate the flow of calls through a telephone network, from generation to termination.
2. To model the interactions between lines and calls, including connection handling.
3. To efficiently utilize resources by managing calls in progress and delayed calls.
4. To implement a real-time dynamic simulation of telephone network activities.

3 Methodology

The main focus of the simulation approach is on essential tasks related to creating and managing telephone conversations. Each telephone line is represented by the Line class, which is initialized with a unique identifier and a state of being 'idle'. The Call class is responsible for randomly generating calls, including selecting a line, determining arrival times, and duration. Calls are connected based on the availability of lines, and the simulation keeps track of ongoing calls using specialized lists. Timers are used to coordinate the dynamic generation of random calls, providing valuable insights into the start, connection, and end of calls in a telecommunications network. The methodology places particular emphasis on encapsulation, timers, and specialized lists to efficiently simulate the dynamic behavior of a telephone network.



4 Coding and Implementation

The project involves coding and implementing a telephone network simulation, which includes creating classes and methods. The code is designed to mimic important elements of a telephone network, such as individual lines, ongoing calls, and lists for managing delayed calls. The implementation is done in C# and includes logic for generating random calls, establishing connections, and monitoring their progress. To accurately simulate a real-world telephone system, key features like line statuses and timed events are integrated into the code.

4.1 Brief Overview of Code

4.1.1 Line.cs :

The Line.cs file contains the Line class which is used to represent a single telephone line. It comes with a constructor that sets a unique ID and an initial state of 'idle'. This class also has functions for accessing and modifying the state, retrieving the ID, and displaying a string representation of the line.

Functions :

- `Line()` function :

  - Initializes a line with a unique ID and sets its state to "idle."
- `GetState()` function :

  - Returns the current state of the line (idle or busy).
- `GetID()` function :

  - Returns the unique ID of the line.
- `SetState(string state)` function :

  - Sets the state of the line to the specified value (idle or busy).
- `ToString()` function :

  - Provides a string representation of the line, including its ID and state.

4.1.2 Call.cs :

The Call.cs file presents the Call class which simulates a telephone call. It contains a constructor that generates random calls, functions to establish connections between calls and ongoing lines or to postpone them, and a toString method that formats the call's representation.

Functions :

- `Call()` function :
  - Generates a random call, selects random 'from' and 'to' lines, and sets arrival time and duration.
- `Connect()` function :
  - Connects a call to lines in progress or places it in the delayed calls list based on line states and the number of ongoing calls.

4.1.3 CallsOnProgressList.cs :

The CallsOnProgressList.cs file utilizes the List class to generate a distinct list specifically for ongoing calls. It integrates the CheckEmptyListTask as a background task to regularly monitor and eliminate finished calls from the list, while simultaneously updating the line statuses.

Functions :

- `CallsOnProgressList()` Constructor :
  - Takes a CallsOnProgressList as a parameter.
- `CheckEmptyListTask.Run()` function :
  - Checks for completed calls in the list and removes them, updating line states accordingly.

4.1.4 TelephoneCallSimulation.cs :

The primary controller of the simulation is the TelephoneCallSimulation.cs file, responsible for establishing the initial setup of the environment. This includes creating idle lines, initializing lists for ongoing and delayed calls, and implementing a timer-based system for generating random calls at specific intervals. Additionally, the file houses the GenerateRandomCall class, which oversees the generation of random calls, and the CheckEmptyListTask class, which conducts regular checks on ongoing calls.

Functions :

- `GenerateRandomCall()` Constructor :
  - Initializes a timer for generating random calls.
- `GenerateCall()` function :
  - Stops the timer, creates a random call, connects it, and schedules the next call.

- `Run()` function :
  - Generates a random delay between 5 and 20 seconds and starts the timer for call generation.
- `PrintLists()` function :
  - Displays the current lists of calls in progress and delayed calls.
  - Initiates the scheduling of the next random call after processing the current one.
- `Main()` Method :
  - Initializes lines, creates lists for calls in progress and delayed calls.
  - Initiates the generation of random calls using the GenerateRandomCall class.

5 Future Scope

1. Implement tracking and analysis of call quality metrics, such as call drops or signal strength.
2. Introduce additional line states, like "ringing" or "on hold," to simulate more realistic phone network behavior.
3. Allow dynamic addition or removal of lines during the simulation to mimic changes in network capacity.
4. Implement intelligent call routing strategies based on factors like line load or historical data.

6 Conclusion

The C# Telephone Call Simulation effectively mimics a telephone network and prioritizes handling delayed calls. It efficiently handles incoming calls, monitors their advancement, and addresses delays by transferring calls to a designated list. This feature offers valuable insights into how an actual telecommunication system may handle different call volumes. Future improvements may concentrate on enhancing the management of delayed calls, such as investigating methods for prioritization or dynamic scheduling, to enhance the simulation's authenticity and usefulness.