

Presentation on TIGEN

Abhay Shanker Pathak
(Information Technology - 4th year, 1828413002)
Guided by - Mr. Maan Singh Sir

Information Technology
United Institute of Technology

April 15, 2022



Table Of Contents

- 1 Abstract
- 2 Problem Definition
 - Existing Solutions
 - Our Solution
- 3 SDLC Models
- 4 Project SDLC Model
 - Agile Model
- 5 Diagrams
- 6 Development
- 7 ScreenShots
- 8 Conclusion



Abstract

TIGEN

TIGEN is the schedule (time-table) generator which is based on the *Genetic Algorithm* (Evolutionary Algorithm) which is used to get solutions for problem inspired by nature.



Problem Definition

TIGEN Time-table Generation with Genetic Algorithm

The idea for creation of **TIGEN** came from the *Soft Computing* subject introduced in the curriculum. Our college department also had to update the schedules almost daily. This was the perfect opportunity to use the idea and automate this real-world task.

Thus, the idea of creating **TIGEN** was born.



Existing Solutions

Some of the solutions (on Github) are:

- nuhu-ibrahim/time-table-scheduling
- akazuko/timetable_scheduler
- Baksonator/evolutionary-timetable-scheduling

The closest best solution which is found on Github is created by user *pranavkhurana*.

- pranavkhurana/Time-table-scheduler



Our Solution

TIGEN can be divided into following components:

- Core
- Data Handling
- User Interaction

Whole project is going to be guided and controlled via a *VCS (Version Control System)* named **git**.



Core (of TIGEN)

Core of the TIGEN is the algorithm (based on Genetic Algorithm) which is to be made as shared library which can be used independently with other components.

Advantages:

- solitary updation facility
- portability
- cross-platform

Techonologies: C, C++, CMake, Unix-Makefiles



Data Handling

Data Handling is the part where the data from user interaction and stored data will be stored, extracted and processed into a meaningful information for the project.

Advantages:

- update via GUI/TUI
- manual updation facility
- will be cross-platform

Technologies: MySQL, mysql-connector for C/C++



User Interaction

User input and choices to create schedule and output to user comes under this component.

Advantages:

- TUI (Terminal User Interface) for terminal lovers
- GUI (Graphical User Interface) for gui lovers
- independent of above two components so that users can also make/modify the the interface to their own liking

Technologies(TUI): C/C++, ncurses/pdcurses-lib

Technologies(GUI): QT Framework



Software Development Life Cycle

SDLC is a continuous process, which starts from the moment, when it's made a decision to launch the project, and it ends at the moment of its full remove from the exploitation. There are several SDLC models divided according to there features and weakness. The most used, popular and important SDLC models are:

- Waterfall Model
- Iterative Model
- Spiral Model
- Agile Model



Waterfall SDLC Model

Following are the reasons for not using Waterfall SDLC Model:

Dis-Advantages:

- High amounts of risk and uncertainty
- Not good model model for complex and object-oriented projects
- It is difficult to measure progress withing stages
- Cannot accomodate with changing requirements
- Adjusting scope during life-cycle can end the project



Iterative SDLC Model

Following are the reasons for not using Iterative SDLC Model:

Dis-Advantages:

- More resources are required
- More management attention is required
- Management complexity is more
- End of project may not be known
- Highly skilled resources are required for risk analysis



Spiral SDLC Model

Following are the reasons for not using Spiral SDLC Model:

Dis-Advantages:

- Management is more complex
- Process is complex
- Spiral may go indefinitely
- End of project may not be known early
- Large number of intermediate stages require excessive documentation



SDLC Model for TIGEN

For the development of TIGEN, **Agile** SDLC model is used.

Agile SDLC Model

Agile model believes that every project needs to be handled differently and the existing methods need to be tailored to best suit the project requirements. In Agile, the tasks are divided to time boxes (small time frames) to deliver specific feature for a release.



Agile SDLC Model

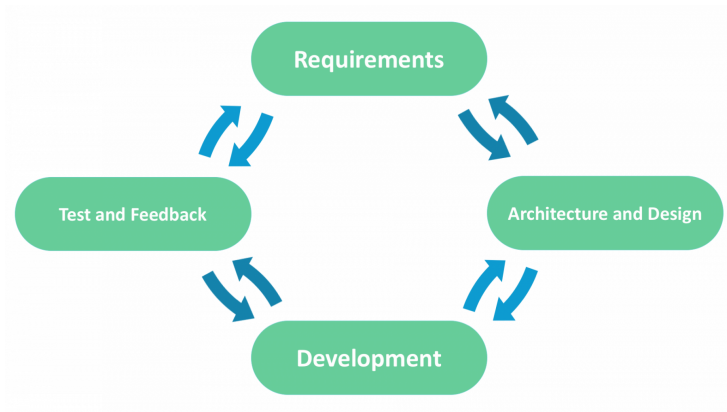


Figure: Different stages of Agile SDLC Model



Agile SDLC Model

Agile Manifesto principles:

- Individuals and interactions
- Working software
- Customer collaboration
- Responding to change



Advantages and Disadvantages of Agile

Advantages	Disadvantages
Realistic approach	Not suitable for handling complex dependencies
Teamwork and cross training	Risk of sustainability
Minimum resource requirements	Risk of maintainability
Fix and changing requirements	Risk of extensibility
Easy to manage	Depends heavily on customer interaction

Table: Advantages and Disadvantages of Agile Model



Class Diagram

Class Diagram: Genetic Algorithm

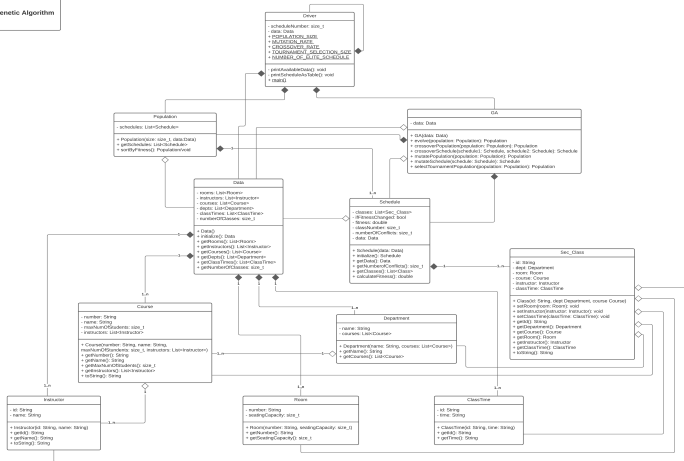


Figure: Class Diagram for TIGEN



Use-Case Diagram

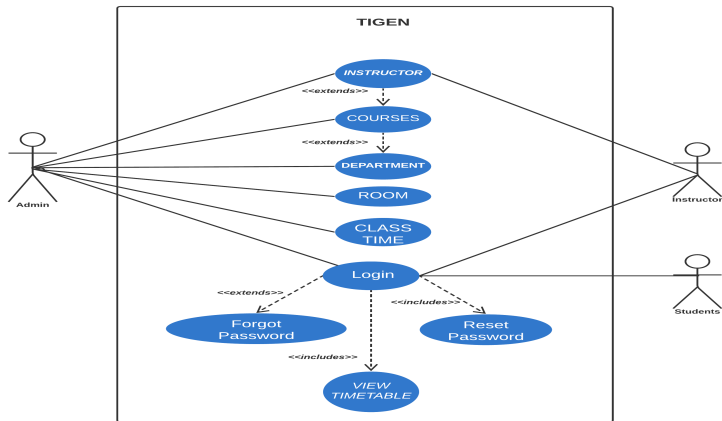


Figure: Use-Case Diagram for TIGEN



Process Flow Chart

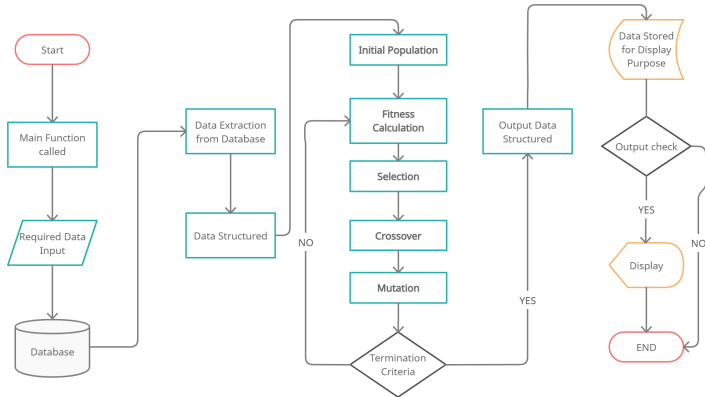


Figure: Process Flow Chart for TIGEN



ER Diagram

ER DIAGRAM

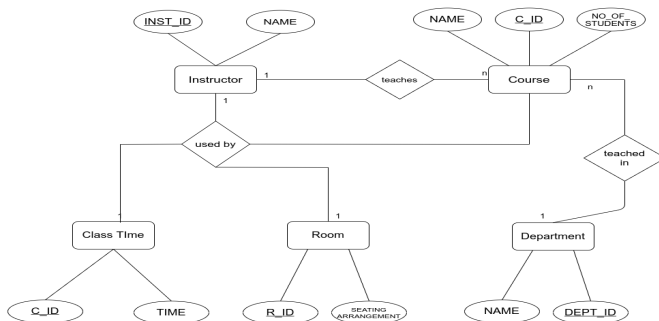


Figure: Entity-Relationship Diagram for TIGEN



DFD (Level ZERO)

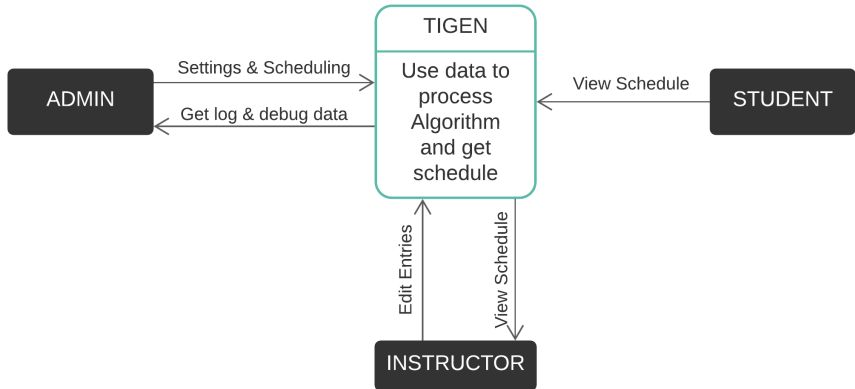


Figure: Data Flow Diagram (Level Zero)



DFD (Level ONE)

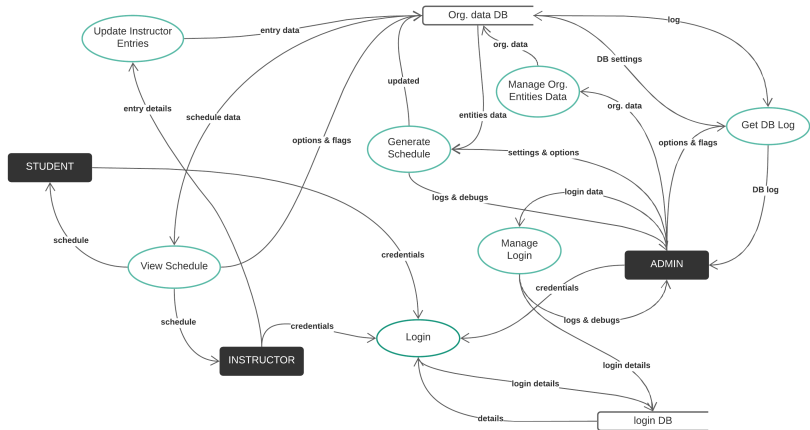


Figure: Data Flow Diagram (Level One)



Designing Library

Library Library is collection of non-volatile resources which may include configuration data, help data, templates, pre-written code, subroutines etc.

Why use Library

As a programmer, you may find yourself using the same function or functions repeatedly. In this case, it is best to put this function or functions in a library to speed up the compilation of the program.

tigen tries to mimic the approach of **MVC** architecture to try to simply and keep the logic separated for different purposes



Types of Libraries in C/C++

C & C++ support namely two types of libraries:

Static Intended to be linked during the creation of an executable or another object file, known as **Static Linking** or **Early Binding**. Linking is usually done by the *linker*.

Dynamic Also known as **Shared Library** or **Shared Object**, it's a file intended to be shared by executable files and further shared object files. Modules used from individual shared objects are loaded into memory at *runtime*.

NOTE:

- In *UNIX/GNU-Linux* based OS, static libraries have **.a**(archive) extension, while dynamic libraries have **.so**(shared object) extension.
- In *Windows* OS, static libraries typically have **.lib** extension, while dynamic libraries have **.dll** extension.



Libraries in Tigen

Tigen basically consists of *two* shared library for development:

tigen_lib Consists of all the logic for working of algorithm with knowledge of used entities and demo data in it.

extras_lib Consists of the features used in *tigen_lib* and view-level interaction to reduce the redundancy during development.

NOTE: *QT* framework and libraries used isn't involved within the two libraries mentioned above of this slide



Working of Algorithm I

The algorithm works as follows:

- After creating the hierarchy of entities used, algorithm creates initial population for calculation.
- This population consists of *chromosome* which is *Schedule* in the case of *tigen*.
- Each chromosome consists of *gene* which is *Class(sec_class)* in the case of *tigen*.
- A population is created which consists of multiple such chromosome.
- The algorithm for *tigen* starts working on the steps of *GA* by first calculating *fitness value* for each chromosome.
- The procedure of selection of chromosome happens from within the population of current generation. *tigen* uses *Tournament Selection* for this purpose.



Working of Algorithm II

- After selection of fittest chromosomes, the algorithm starts to work on crossover operation to create new-offspring of chromosomes for next generation. *tigen* uses *Multi-Point Crossover* for this purpose.
- Next step is mutation of chromosome, *tigen* uses *Scramble-Mutation* for this purpose.
- With it the population for next generation is created and if the *termination criteria* matches the algorithm stops, else the calculation for next generation starts again.
- When the termination criteria satisfies, algorithm puts up the result from the current/fittest population in appropriate data structure to be used by *View Model*.



Helper Utilities

There are 3 different helper utilities for *tigen* in *extras_lib* library:

curses util Using curses library provided in linux/Unix environment to show provide interaction with user on CLI.

exeception util To perform task of exception handling within *tigen*.

extra util This currently contains the logic for random number generation for working of GA. The random number generation is currently done by **mt19937** which is a *Mersenne Twister*.



Facilitating Data Storage

Model

In *MVC* approach, M stands for **Model**. This means to facilitate data from and through the Controller logic.

Data handling is done by two ways in *tigen*:

CLI/TUI Within this, connection via *MYSQL_C++_Connector* is to be provided to connect the data handling logic to MySQL

GUI For data handling in GUI, *SQL* lib provided by *QT* framework is used



View

View

In *MVC* approach, *V* stands for **View**. This layer interacts with user to show and take input/output and to pass to Controller for further processing of any way.

- CLI** *tigen* can dump output to STDOUT. For any sort of interaction with program using this method, user will have to use the provided flags by *tigen*.
- TUI** *tigen* allows better user interaction in terminal via *Terminal User Interface*, which allows user to select different input and processing options.
- GUI** *tigen* also allows user interaction via full-fledged GUI build using *QT* framework. User will get all sorts of input and output option within this approach.



TUI I

```

raytracer@server: ~
d:0.0 B/s u:0.0 B/s | v:40% | t:100°C | c:1% | r:26% | b:99% | Tue Jun 01 10:23:42 PM

Time-Table generation using Genetic Algorithm

Resultant Data Window

Available Departments ==>
name: Dept. of Mechanical Engineering, courses: [
  Automobile,
  Aeronautics,
  Metallurgy
]
name: Dept. of NueroScience, courses: [
  Biological,
  Chemical
]
name: Dept. of Information Technology, courses: [
  Embedded Programming,
  Web Development,
  Networking Security
]

[Data]—[Generations]—[Result]

Press <tab> and <\> key for tab cycle, and arrow keys to scroll(vertical and horizontal)[press 'q' to exit]

```

Figure: show available data for operation



TUI II

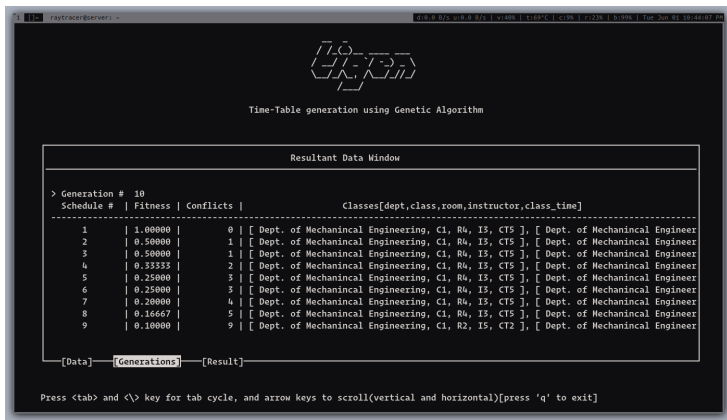


Figure: show population for each generation



TUI III

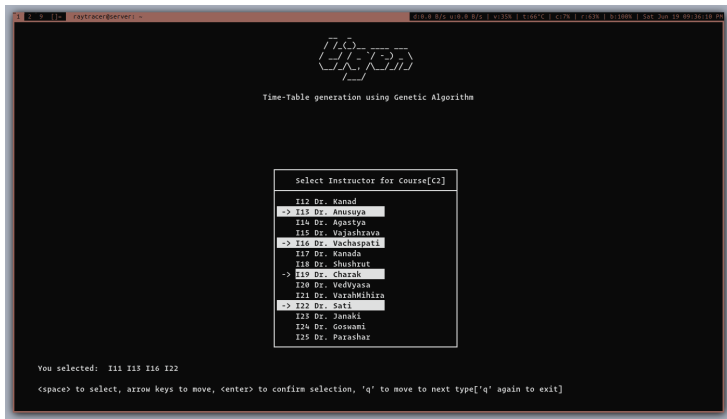
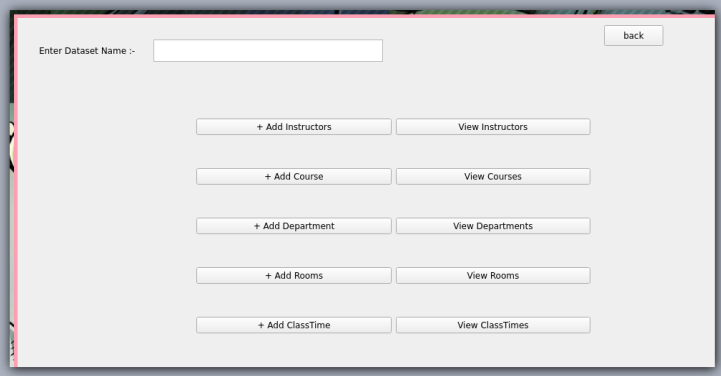


Figure: a selection window demo



GUI I



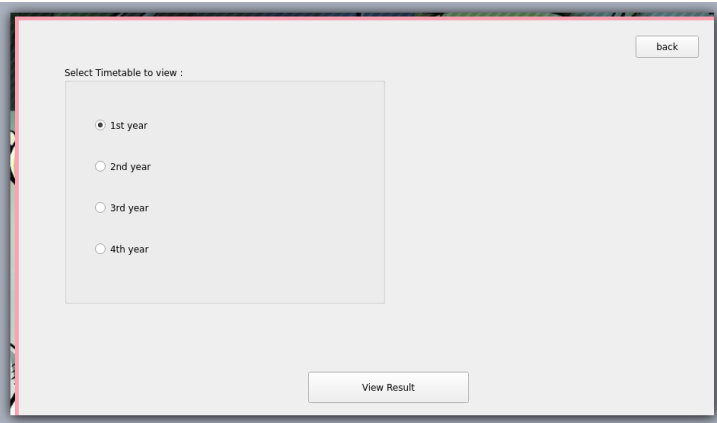
The screenshot shows a web-based GUI for adding dataset information. At the top, there is a label "Enter Dataset Name :-" followed by a text input field. To the right of the input field is a "back" button. Below the input field, there are five rows of buttons, each with an "Add" button on the left and a "View" button on the right. The rows are: "+ Add Instructors" and "View Instructors", "+ Add Course" and "View Courses", "+ Add Department" and "View Departments", "+ Add Rooms" and "View Rooms", and "+ Add ClassTime" and "View ClassTimes".

Dataset Name	Instructors	Courses	Departments	Rooms	ClassTime

Figure: data to add



GUI II



A screenshot of a web application window. The window has a light gray background. In the top right corner, there is a button labeled "back". Below the "back" button, the text "Select Timetable to view :" is displayed. Underneath this text is a white rectangular box containing four radio button options: "1st year", "2nd year", "3rd year", and "4th year". The "1st year" option is selected, indicated by a filled black circle. At the bottom center of the window, there is a button labeled "View Result".

Figure: show timetable created



GUI III

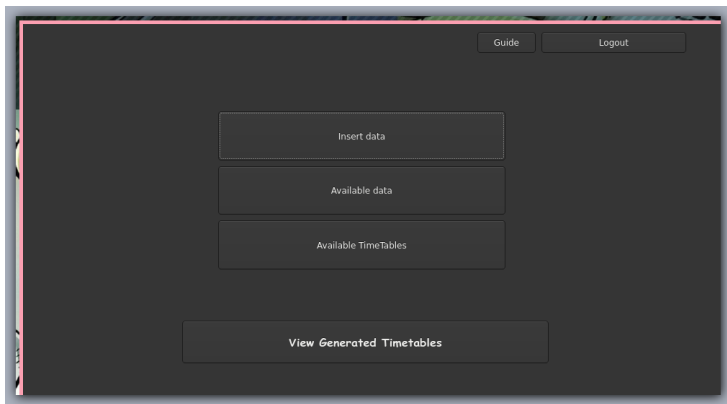


Figure: dark mode gui



Future Scope

tigen has capability to be extended in several ways as a tool:

- Test the research of GA *operators* within the algorithm
- Make modules (libraries) for different types of time-table extending/inspired by the algorithm used
- Hardware support
- Better TUI, with support for *colors* and *vi/emacs* keybindings support
- More features for GUI (depending on usage)



THANK YOU !!!

