**Adaptive Applications**

**Project Proposal**

*Team Glimmer*

# Table of Contents

[Introduction 2](#_Toc506398543)

[Functional Architecture 3](#_Toc506398544)

[Mood Inference Engine(MIE) 3](#_Toc506398545)

[Song Matching Engine(SME) 4](#_Toc506398546)

[Assignment of Roles 4](#_Toc506398547)

[Time Plan 5](#_Toc506398548)

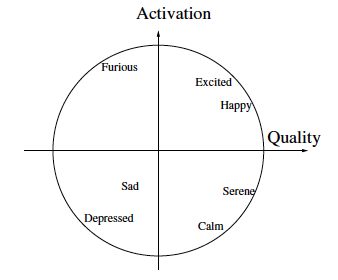
[References 5](#_Toc506398549)

# 

# Introduction

In this assignment, we propose to build a web based Music Player App that adapts to the user’s mood to play songs.

The user’s mood can depend on number of factors such as time of the day, day of the week, location, etc. This can influence his/her choice of song being played and can decide to continue playing the song, skip it, like or dislike the song. These choices made by the user can again be used to infer his/her mood.



*Fig 1. Thayers model of mood [1]*

Any mood can be modelled by assigning values between -1 and +1 using the above representation. It can be represented as a two-dimensional vector, where .

Songs have features such as genre, year of release, singer, average bpm, etc. that can be used to tag them. This information can be combined with user’s preference of the songs to give more meaningful adaptations.

The final goal of this project is to build a user model that assists to adapt to the user’s mood and play songs accordingly.

# Functional Architecture



*Fig 1: Functional architecture of “Go With The Times Music Player”*

The above diagram explains the flow of data from the user interface to the SME. User Interface is responsible for accepting the user input. This input data is sent to the MIE. MIE is responsible for modelling the user actions and predicting the mood score. The mood score is then passed on to the SME which selects a song from a large dataset considering the user preferences.

# Mood Inference Engine(MIE)

Mood Inference Engine plays a major role for the user modelling. The emotions of the user are modelled by this engine. MIE infers the current mood of the person. This is done by collecting the feedback from the user. The client side of the application gets the feedback from the actions (skip, like, dislike, no action) that user performs. This feedback is sent to the MIE. MIE calculates the score for each of the emotions (happy, sad, angry etc) based on the mood of the song played and the actions of the user. Based on the score, the mood of the user is inferred. This is sent to the Song Matching Engine(SME).

Along with the user mood score calculation, it also calculates a weight (α1) based on the type of the music, say, Jazz, RnR, RnB, Pop etc. The type of the user music that the user likes can be obtained from the initial preferences that user makes. These initial preferences are attained from the set of questions asked for the first time the application is used by the user. α1 is also sent to the SME along with mood score to select which type of songs to be played.

A computational model can be used to calculate the mood [2]. The mood is calculated as a function of time. The time refers to the duration for which the user listened to the music before skipping it.

If t is the duration, then each emotion can be calculated as,

Ei = F(m,t) , 0<i<=n (number of emotions)

Where m is recognized mood of the song which is played.

The mood is decided as

M = max(Ei) , 0<i<=n (number of emotions)

Output of the MIE is O = M + G(αj), 0<j<=TM(number of types of the music)

# Song Matching Engine(SME)

This is the most salient component of the architecture for song modelling. It has two subdivisions - song search engine and song database. The song search engine is responsible for picking up the song based on user’s mood. The mood will be conjectured by the Mood Inference Engine and will be given as an input to this engine which will in turn pick up the next relevant song. The songs/tracks will be tagged to various moods in clusters and will be stored in the database. There is a slight chance that if a user likes a particular song there could be a possibility that the system suggests that songs more often as compared to other songs. To nullify this error, the system introduces a noise parameter.

There could be two types of users. One that is moody whose mood changes frequently and likes to listen to songs in the same genre with different moods. And the other user could be experimentative and prefers to listen to new songs irrespective of the genre. To model these user preferences, the SME will follow reinforcement learning and will choose between exploration or exploitation.

# Assignment of Roles

|  |  |
| --- | --- |
| **Team member** | **Role assigned** |
| Meenu Vincent | User modelling |
| Siddhitha Sundari Bhoopathy | User modelling |
| Prashant Aggarwal | Implementation of Mood Inference Engine |
| Amit Prasad | Implementation of Song Matching Engine |
| Jianan Wei | Adaptation technique design |

# Time Plan

|  |  |
| --- | --- |
| **Weeks** | **Tasks** |
| Week 5 | Project Proposal |
| Week 6 | Setting up the infrastructure and basic framework. |
| Week 7 | Formalize User and Data Modelling |
| Week 8 | Implementation of Mood Inference Engine Component and bug fixes |
| Week 9 | Designing Algorithm for Song Matching Engine |
| Week 10 | Implementation of Song Matching Engine Component and bug fixes |
| Week 11 | Integration, Testing and minor improvements |
| Week 12 | Report writing |

# References

[1] M. Tolos, R. Tato, and T. Kemp, “Mood-based navigation through large collections of musical data,” *Second {IEEE} {Consumer} {Communications} {Networking} {Conference}, 2005. {CCNC}. 2005*, pp. 71–75, 2005.

[2] Katsimerou, C., Redi, J. A., & Heynderickx, I. (2014). A Computational Model for Mood Recognition. *International Conference on User Modeling, Adaptation, and Personalization*.