**USE OF INBUILT SENSORS IN SMART PHONE TO DEVELOP FETAL DOPTONE**

**SOFTWARE DEVELOPMENT PROCESS**

**TEAM:**

* **Project Directors :**

Prof. Abhishek Srivastava

Dr. Sachin Gothi

* **Team of Students :**

Srikrishna Mamillapalli | 180001029

Kandula Chandra Kanth | 180001024

Pulivarthi Karthik Chakravarthi | 180001040

Kumpatla Vijay Babu | 180001027

* **Course Instructor :**

Dr. Puneet Gupta

CSE 258 SOFTWARE ENGINEERING LAB

Spring 2020

* **INTRODUCTION**

Software Engineering is defined as a branch of computer science which deals with the systematic application of [engineering](https://en.wikipedia.org/wiki/Engineering) approaches to the development of software (Wikipedia).

It is essentially a problem solving activity which involves understanding the nature of the problem by breaking it into smaller components and solving the problem by essentially addressing all of its requirements.

Software engineering techniques are needed because software systems unlike software programs are large and cannot be completely understood by a single person. They are usually developed by a team of developers or multiple teams instead of a single person.

As the work is usually divided into smaller components, the software development process requires teamwork and co-ordination between the working teams to ensure that the parts of the system work properly when put together.

Software process models provide us with well-defined techniques and methodologies which function as formal procedures for tackling software development in a thorough and rigorous manner. Following any of their approaches or a combination of their principles will result in the development of a high quality software system.

The main purpose of software process models is to simplify the complexity of developing large software systems by providing systematic approaches to deal with the development process.

* **SOFTWARE PROCESSES**

As Software systems tend to widely vary from one another, the appropriate software engineering process depends on the type of software system as there is no concept of one-size-fits-all universally applicable software process. It mainly depends on the software type and environment, customer requirements and the team attributes.

Though many different software processes exist, all of them involve the four essential design principles of specification, design and implementation, validation and evolution. These activities are complex activities in themselves, and they include sub-activities such as requirements validation, architectural design, and unit testing.

The software process model chosen for this app project is an amalgamation of several design principles from various design approaches. Herein we detail the aspects of different process models which have been considered for adoption to our project including how their principles resonate or contradict with the design philosophies of this project:

* Plan-driven processes are processes where all of the process activities are planned in advance and progress is measured against this plan.
* In agile processes, planning is incremental and it is easier to change the process to reflect changing customer requirements.

Waterfall:

* Principle: You plan and schedule all of the process activities before starting software development
* Result of each phase is one or more approved documents
* The following phase should not start until the previous phase has finished
* The waterfall model is mostly used for large systems engineering projects where a system is developed at several sites.
* The waterfall model is a plan-driven model which involves inflexible partitioning of the project into distinct stages. This in itself makes it difficult to respond to changing client needs which is essential for this app development project, as they client himself is required to be a part of the development team. The big-bang approach followed by the waterfall model is especially incompatible with this project as the app has to be flexible and accommodate change as it is being developed. Due to these reasons the waterfall model is incompatible with this app development project.

Software Prototyping Development:

* May involve leaving out functionality
* Prototype should focus on areas of the product that are not well understood
* Error checking and recovery may not be included in the prototype
* Focus on functional rather than non-functional requirements such as reliability and security
* This project fully inculcates the software prototyping principles to evaluate the feasibility of the software design and architecture decisions along with the client requirements. This is to ensure proper change anticipation and reduce development effort in the long run.

Incremental development:

* Incremental development involves developing a system incrementally and exposing it to customers for comment, without necessarily delivering it and deploying it in the customer’s environment.
* Once the development of an increment is started, the requirements are frozen though requirements for later increments can continue to evolve
* Managers need regular deliverables to measure progress. It is not cost-effective to produce documents that reflect every version of the system
* Lower risk of overall project failure due to the iterative development approach, change avoidance and chance tolerance
* The incremental development model is a highly compatible process model for this type of software project which embraces the evolution of customer needs and indirectly makes them participate in the development process. This is achieved by encouraging them to evaluate every increment for validation of its functionality. As newer increments are completed, they are integrated with existing increments so that system functionality improves with each delivered increment.
* We have chosen incremental development over delivery because of certain drawbacks of in the latter’s approach. The main reason however is because that incremental delivery involves delivering the individual increments to the customer for comment and experimentation. This means the smaller components must be installed in the customer’s working environment, which is not possible in this project unless the entirety of the app with most of its functionality is completely implemented.
* Though it provides a more realistic evaluation about the practical use of software, iterative delivery is also problematic when the new system is intended to replace an existing system. The system structure tends to degrade as new increments are delivered and older systems are discarded in favour of newly developed increments.
* The one downside to using an incremental development approach is that the process is not visible. Regular measurement of progress requires deliverables and it becomes extremely tedious to create documents that reflect the changes made or functionality implemented in every version of the system. But due to the many constructive influences incremental approach has on this app development project, we have decided to integrate the majority of its design principles.

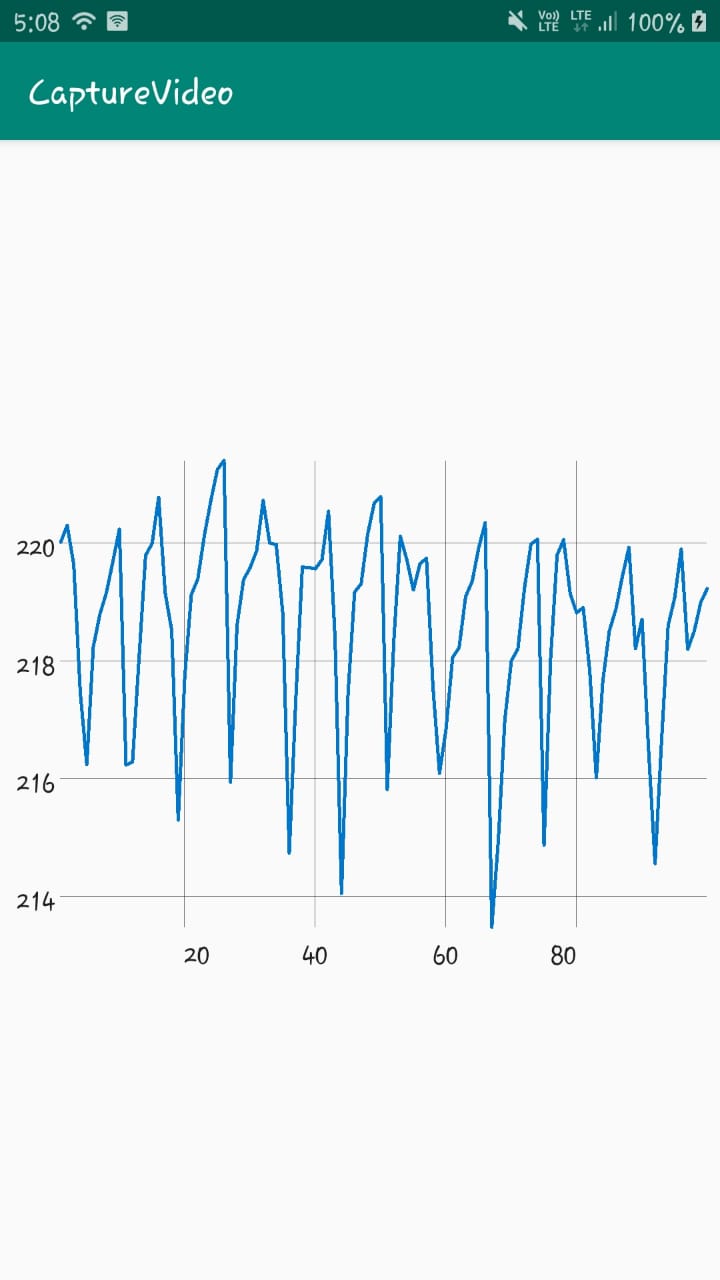
Extreme programming:

* New versions may be built several times per day
* Increments are delivered to customers every 2 weeks
* All tests must be run for every build and the build is only accepted if tests run successfully

Pair programming:

* Pair programming involves programmers working in pairs, developing code together.
* This helps develop common ownership of code and spreads knowledge across the team.
* It serves as an informal review process as each line of code is looked at by more than 1 person.
* It encourages refactoring as the whole team can benefit from improving the system code.
* The sharing of knowledge that happens during pair programming is very important as it reduces the overall risks to a project when team members leave.
* Pair programming is not necessarily inefficient and there is some evidence that suggests that a pair working together is more efficient than 2 programmers working separately.
* We have decided to take the incremental development approach a step further and also adapted the principles of extreme programming and pair programming into our software development process methodology. This is because of our team working in two-man units to simultaneously develop code and design the software architecture. Pair-programming appropriately complements this tactic of pair-teams where the work is being judged and reviewed by 2 sets of eyes at once, which eliminates errors and personal bias to a large extent.

* Extreme and Pair programming techniques also seemingly increase efficiency of the app development as the sharing of knowledge reduces overall risk and improves the understanding of the overall project for everyone involved. Extreme programming also introduces the idea of refactoring which is very useful in keeping the code simple and maintainable. This is further reinforced by the introduction of Pair programming techniques into the mix where refactoring becomes even more viable as the entire team gets to know and understand the code better.
* **SOFTWARE IMPLEMENTATION**
* We have designed a prototype of the app which successfully implements part of the functionality of the app. This requires the app to take a video of the pregnant mother’s tip of index finger using the camera and flash of the smart phone to calculate the mother’s heartrate to distinguish it from the fetal heartrate.
* This video is sampled in an appropriate resolution to convert it into individual frames. The frame-count is inversely proportional to the chosen resolution to output it in the same amount of time. At 1920x1080 resolution, the frame-count of the recorded video had to be drastically reduced to 1 fps. Even this was not feasible for practical implementation as the loop inside the algorithm has to run for 2,073,600(1920\*1080\*1) ~ 2 million times in 1 second, which essentially equates the total number of pixels in a single frame.
* This is to analyse the red frame intensity vs. frame count data to plot a graph using Red frame intensity on the Y-axis and No. of frames on the X-axis, to determine the beats present in the graph which consequently results in the determination of the heart-rate or BPM of a person. The beats are calculated by determining the local extrema in the graph.
* To simplify the above process and make it practically feasible, we have chosen a 144p (256X144) resolution with the frame-count as 15 fps, which involves 36,864(256\*144) pixels comparatively and delivers results as fast as 90secs by contrast. The following image represents the successful and reasonably accurate representation of the red frame intensity vs. frame count graph -



* **CONCLUSION**
* This project aims to develop an initial implementation, getting feedback from the client, and evolving the software through several versions until the required system has been developed. This has been achieved through the software prototyping process as stated earlier. Through the rigorous analysis of several software process models, we have taken the best and most suitable software design techniques which are compatible with our project and designed a robust model to follow throughout our software development phase. To sum up, here are the various principles that we have inculcated into our software methodology -

Principles:

* Software Prototyping
* Incremental planning and development
* Small releases
* Simple design
* Test-first development
* Refactoring
* Pair programming
* Collective ownership
* Continuous integration
* On-site customer
* User stories for specifications: The customer chooses the stories for inclusion in the next release based on their priorities and the schedule estimates
* Writing tests before code clarifies the requirements to be implemented