Butterfly

Project Document

**Team Number: 3**

**Team Leader: Jake Wheeler**

**Team Members**

**Nate Christiansen**

**Nick Kapty**

# **Contents**

## **Project Drivers**

1. The Purpose of the Project

2. The Client, the Customer, and Other Stakeholders

3. Users of the Product

## **Project Constraints**

4. Mandated Constraints

5. Naming Conventions and Definitions

6. Relevant Facts and Assumptions

## **Functional Requirements**

7. The Scope of the Work

8. The Scope of the Product

9. Functional and Data Requirements

## **Nonfunctional Requirements**

10. Look and Feel Requirements

11. Usability and Humanity Requirements

12. Performance Requirements

13. Operational and Environmental Requirements

14. Maintainability and Support Requirements

15. Security Requirements

16. Cultural and Political Requirements

17. Legal Requirements

## **Project Issues**

18. Open Issues

19. Off-the-Shelf Solutions

20. New Problems

21. Tasks

22. Migration to the New Product

23. Risks

24. Costs

25. User Documentation and Training

26. Waiting Room

27. Ideas for Solutions

# **1. The Purpose of the Project**

## **1a. The User Business or Background of the Project Effort**

### The purpose of the project is to create a music player application to play local music, as well as YouTube playlists. Users will be able to sort by album, artist, song title, genre, as well as various other tags. Users will be able to log into social media to share what they are currently listening to.

## **1b. Goals of the Project**

### To create a quality music player that will allow users to listen to online and local music, with customizable features and social media integration.

# **2. The Client, the Customer, and Other Stakeholders**

## **2a. The Client**

### We are our own client, so we are using our own specifications.

## **2b. The Customer**

### The customer of our product would be anyone who wishes to use our music player.

## **2c. Other Stakeholders**

Our only stakeholder is Professor Fan, who is teaching us as we develop the project.

# **3. Users of the Product**

## **3a. The Hands-On Users of the Product**

### Any user who has the knowledge to use a computer and any passing interest in music can use our product.

## **3b. Priorities Assigned to Users**

● Key users: They will use all of the additional features of the product, such as YouTube playlists, social media features, etc.

● Secondary users: They will use the product for its local music playing ability only.

## **3c. User Participation**

### We will act as users for this project, testing all of the features added. We will also consider outside input from unbiased third parties.

## **3d. Maintenance Users and Service Technicians**

### We will maintain and service our product after its initial release and all additional releases.

# **4. Mandated Constraints**

## **4a. Solution Constraints**

### We will be using Java, because Java has a rich framework, and well developed and documented APIs. Java also runs on any operating system, as it is not platform dependent. The program will also meet all of our functional and non-functional requirements.

## **4b. Implementation Environment of the Current System**

### The program will run on any environment with Java installed, and for additional features such as YouTube playlists and social media integration, internet connectivity will be required.

## **4c. Partner or Collaborative Applications**

### The Java Runtime Environment, Twitter API, and YouTube API will be used.

## **4d. Off-the-Shelf Software**

### We will use NetBeans IDE, Git version control, Paint.net, and the Java Development Kit.

## **4e. Anticipated Workplace Environment**

### The environment targeted will be Windows machines with Java Runtime Environment installed, as well as internet connectivity.

## **4f. Schedule Constraints**

### The project must be completed by December 11, 2015, in order to receive full credit for the class.

## **4g. Budget Constraints**

### Our project will have no budget, however we will have free computer usage at Penn State Behrend.

# **5. Naming Conventions and Definitions**

## **5a. Definitions of All Terms, Including Acronyms, Used in the Project**

### Butterfly – The name of our music player application

### Song – Individual music file that can be played

### Playlist – User selected list of songs

### Artist – Creator of a song

### Album – Collection of specific songs chosen by the artist

### Tweet – A short message that is sent to Twitter that is written by the user

# **6. Relevant Facts and Assumptions**

## **6a. Facts**

### Our project will be developed in Java, and will be completed in one semester. The user will need a YouTube account and a Twitter account to take advantage of all of the features that our application will offer.

## **6b. Assumptions**

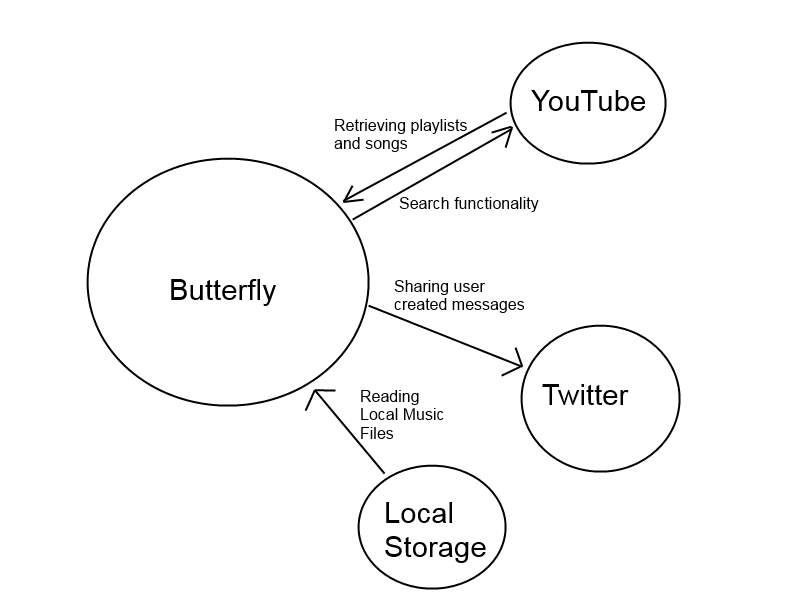
### We assume that we will be able to develop the program using NetBeans and appropriate frameworks. We also assume YouTube and Twitter integration will be available.

# **7. The Scope of the Work**

## **7a. The Current Situation**

### Currently there are no popular music players that play local music with the ability to stream audio from YouTube as well.

## **7b. The Context of the Work**



# **8. The Scope of the Product**

## **8a. Product Boundary**

### A use case diagram identifies the boundaries between the users (actors) and the product. You arrive at the product boundary by inspecting each business use case and determining, in conjunction with the appropriate stakeholders, which part of the business use case should be automated (or satisfied by some sort of product) and what part should be done by the user. This task must take into account the abilities of the actors (section 3), the constraints (section 4), the goals of the project (section 1), and your knowledge of both the work and the technology that can make the best contribution to the work.

### The use case diagram shows the actors outside the product boundary (the rectangle). The product use cases are the ellipses inside the boundary. The lines denote usage. Note that actors can be either automated or human.

## **8b. Product Use Case List**

### The use case diagram is a graphical way of summarizing the product use cases relevant to the product. If you have a large number of product use cases (we find 15–20 is a good limit), then it is better to make a list of the product use cases and model or describe each one individually.

## **8c. Individual Product Use Cases**

### This is where you keep details about the individual product use cases on your list. You can include a scenario for each product use case on your list.

# **9. Functional and Data Requirements**

## **9a. Functional Requirements**

### A specification for each functional requirement. As with all types of requirements, use the requirements shell*.* A full explanation is included in this template’s introductory material.

## **9b. Data Requirements**

### A specification of the essential subject matter, business objects, entities, and classes that are germane to the product. It might take the form of a first-cut class model, an object model, or a domain model. Alternatively, these requirements might be described by defining the terms in the dictionary described in section 5.

# **10. Look and Feel Requirements**

## **10a. Appearance Requirements**

### The section contains requirements relating to the spirit of the product. Your client may have made particular demands for the product, such as corporate branding, colors to be used, and so on. This section captures the requirements for the appearance. Do not attempt to design it until the appearance requirements are known.

## **10b. Style Requirements**

### Requirements that specify the mood, style, or feeling of the product, which influences the way a potential customer will see the product. Also, the stakeholders’ intentions for the amount of interaction the user is to have with the product.

### In this section, you would also describe the appearance of the package if this is to be a manufactured product. The package may have some requirements as to its size, style, and consistency with other packages put out by your organization. Keep in mind the European laws on packaging, which require that the package not be significantly larger than the product it encloses.

### The style requirements that you record here will guide the designers to create a product as envisioned by your client.

# **11. Usability and Humanity Requirements**

## **11a. Ease of Use Requirements**

### This section describes your client’s aspirations for how easy it is for the intended users of the product to operate it. The product’s usability is derived from the abilities of the expected users of the product and the complexity of its functionality.

### The usability requirements should cover properties such as these:

● Efficiency of use: How quickly or accurately the user can use the product.

● Ease of remembering: How much the casual user is expected to remember about using the product.

● Error rates: For some products it is crucial that the user commits very few, or no, errors.

● Overall satisfaction in using the product: This is especially important for commercial, interactive products that face a lot of competition. Web sites are a good example.

● Feedback: How much feedback the user needs to feel confident that the product is actually accurately doing what the user expects. The necessary degree of feedback will be higher for some products (e.g., safety-critical products) than for others.

## **11b. Personalization and Internationalization Requirements**

### This section describes the way in which the product can be altered or configured to take into account the user’s personal preferences or choice of language.

### The personalization requirements should cover issues such as the following:

● Languages, spelling preferences, and language idioms

● Currencies, including the symbols and decimal conventions

● Personal configuration options

## **11c. Learning Requirements**

### Requirements specifying how easy it should be to learn to use the product. This learning curve ranges from zero time for products intended for placement in the public domain (e.g., a parking meter or a web site) to a considerable amount of time for complex, highly technical products. (We know of one product where it was necessary for graduate engineers to spend 18 months in a training program before being qualified to use the product.)

## **11d. Understandability and Politeness Requirements**

### This specifies the requirement for the product to be understood by its users. While “usability” refers to ease of use, efficiency, and similar characteristics, “understandability” determines whether the users instinctively know what the product will do for them and how it fits into their view of the world. You can think of understandability as the product being polite to its users and not expecting them to know or learn things that have nothing to do with their business problem.

## **11e. Accessibility Requirements**

### The requirements for how easy it should be for people with common disabilities to access the product. These disabilities might be related to physical disability or visual, hearing, cognitive, or other abilities.

# **12. Performance Requirements**

## **12a. Speed and Latency Requirements**

### Specifies the amount of time available to complete specified tasks. These requirements often refer to response times. They can also refer to the product’s ability to operate at a speed suitable for the intended environment.

## **12b. Safety-Critical Requirements**

### Quantification of the perceived risk of damage to people, property, and environment. Different countries have different standards, so the fit criteria must specify precisely which standards the product must meet.

## **12c. Precision or Accuracy Requirements**

### Quantification of the desired accuracy of the results produced by the product.

## **12d. Reliability and Availability Requirements**

### This section quantifies the necessary reliability of the product. The reliability is usually expressed as the allowable time between failures, or the total allowable failure rate.

### This section also quantifies the expected availability of the product.

## **12e. Robustness or Fault-Tolerance Requirements**

### Robustness specifies the ability of the product to continue to function under abnormal circumstances.

## **12f. Capacity Requirements**

### This section specifies the volumes that the product must be able to deal with and the amount of data stored by the product.

## **12g. Scalability or Extensibility Requirements**

### This specifies the expected increases in size that the product must be able to handle. As a business grows (or is expected to grow), our software products must increase their capacities to cope with the new volumes.

## **12h. Longevity Requirements**

### This specifies the expected lifetime of the product.

# **13. Operational and Environmental Requirements**

## **13a. Expected Physical Environment**

### This section specifies the physical environment in which the product will operate.

## **13b. Requirements for Interfacing with Adjacent Systems**

### This section describes the requirements to interface with partner applications and/or devices that the product needs to successfully operate.

## **13c. Productization Requirements**

### Any requirements that are necessary to make the product into a distributable or salable item. It is also appropriate to describe here the operations needed to install a software product successfully.

## **13d. Release Requirements**

### Specification of the intended release cycle for the product and the form that the release shall take.

# **14. Maintainability and Support Requirements**

## **14a. Maintenance Requirements**

### A quantification of the time necessary to make specified changes to the product.

## **14b. Supportability Requirements**

### This specifies the level of support that the product requires. Support is often provided via a help desk. If people will provide support for the product, that service is considered part of the product: Are there any requirements for that support? You might also build support into the product itself, in which case this section is the place to write those requirements.

## **14c. Adaptability Requirements**

### Description of other platforms or environments to which the product must be ported.

# **15. Security Requirements**

## **15a. Access Requirements**

### Specification of who has authorized access to the product (both functionality and data), under what circumstances that access is granted, and to which parts of the product access is allowed.

## **15b. Integrity Requirements**

### Specification of the required integrity of databases and other files, and of the product itself.

## **15c. Privacy Requirements**

### Specification of what the product has to do to ensure the privacy of individuals about whom it stores information. The product must also ensure that all laws related to privacy of an individual’s data are observed.

## **15d. Audit Requirements**

### Specification of what the product has to do (usually retain records) to permit the required audit checks.

## **15e. Immunity Requirements**

### The requirements for what the product has to do to protect itself from infection by unauthorized or undesirable software programs, such as viruses, worms, and Trojan horses, among others.

# **16. Cultural and Political Requirements**

## **16a. Cultural Requirements**

### This section contains requirements that are specific to the sociological factors that affect the acceptability of the product. If you are developing a product for foreign markets, then these requirements are particularly relevant.

## **16b. Political Requirements**

### This section contains requirements that are specific to the political factors that affect the acceptability of the product.

# **17. Legal Requirements**

## **17a. Compliance Requirements**

### A statement specifying the legal requirements for this system.

## **17b. Standards Requirements**

### A statement specifying applicable standards and referencing detailed standards descriptions. This does not refer to the law of the land—think of it as an internal law imposed by your company.

# **18. Open Issues**

### Issues that have been raised and do not yet have a conclusion.

# **19. Off-the-Shelf Solutions**

## **19a. Ready-Made Products**

### List of existing products that should be investigated as potential solutions. Reference any surveys that have been done on these products.

## **19b. Reusable Components**

### Description of the candidate components, either bought from outside or built by your company, that could be used by this project. List libraries that could be a source of components.

## **19c. Products That Can Be Copied**

### List of other similar products or parts of products that you can legally copy or easily modify.

### 

# **20. New Problems**

## **20a. Effects on the Current Environment**

### A description of how the new product will affect the current implementation environment. This section should also cover things that the new product should *not* do.

## **20b. Effects on the Installed Systems**

### Specification of the interfaces between new and existing systems.

## **20c. Potential User Problems**

### Details of any adverse reaction that might be suffered by existing users.

## **20d. Limitations in the Anticipated Implementation Environment That May Inhibit the New Product**

### Statement of any potential problems with the new automated technology or new ways of structuring the organization.

## **20e. Follow-Up Problems**

### Identification of situations that we might not be able to cope with.

# **21. Tasks**

## **21a. Project Planning**

### Details of the life cycle and approach that will be used to deliver the product. A high-level process diagram showing the tasks and the interfaces between them is a good way to communicate this information.

## **21b. Planning of the Development Phases**

### Specification of each phase of development and the components in the operating environment.

# **22. Migration to the New Product**

## **22a. Requirements for Migration to the New Product**

### A list of the conversion activities. Timetable for implementation.

## **22b. Data That Has to Be Modified or Translated for the New System**

### List of data translation tasks.

# **23. Risks**

### All projects involve risk—namely, the risk that something will go wrong. Risk is not necessarily a bad thing, as no progress is made without taking some risk. However, there is a difference between unmanaged risk—say, shooting dice at a craps table—and managed risk, where the probabilities are well understood and contingency plans are made. Risk is only a bad thing if the risks are ignored and they become problems. Risk management entails assessing which risks are most likely to apply to the project, deciding a course of action if they become problems, and monitoring projects to give early warnings of risks becoming problems.

### This section of your specification should contain a list of the most likely risks and the most serious risks for your project. For each risk, include the probability of that risk becoming a problem. Capers Jones’s *Assessment and Control of Software Risks* (Prentice-Hall, Englewood Cliffs, N.J., 1994) gives comprehensive lists of risks and their probabilities; you can use these lists as a starting point. For example, Jones cites the following risks as being the most serious:

• Inaccurate metrics

• Inadequate measurement

• Excessive schedule pressure

• Management malpractice

• Inaccurate cost estimating

• Silver bullet syndrome

• Creeping user requirements

• Low quality

• Low productivity

• Cancelled projects

### Use your knowledge of the requirements as input to discover which risks are most relevant to your project.

### It is also useful input to project management if you include the impact on the schedule, or the cost, if the risk does become a problem.

# **24. Costs**

### For details on how to estimate requirements effort and costs, refer to Appendix C Function Point Counting: A Simplified Introduction

### The other cost of requirements is the amount of money or effort that you have to spend building them into a product. Once the requirements specification is complete, you can use one of the estimating methods to assess the cost, expressing the result as a monetary amount or time to build.

### There is no best method to use when estimating. Keep in mind, however, that your estimates should be based on some tangible, countable artifact. If you are using this template, then, as a result of doing the work of requirements specification, you are producing many measurable deliverables. For example:

● Number of input and output flows on the work context

● Number of business events

● Number of product use cases

● Number of functional requirements

● Number of nonfunctional requirements

● Number of requirements constraints

● Number of function points

### The more detailed the work you do on your requirements, the more accurate your deliverables will be. Your cost estimate is the amount of resources you estimate each type of deliverable will take to produce within your environment. You can create some very early cost estimates based on the work context. At that stage, your knowledge of the work will be general, and you should reflect this vagueness by making the cost estimate a range rather than a single figure.

### As you increase your knowledge of the requirements, we suggest you try using function point counting—not because it is an inherently superior method, but because it is so widely accepted. So much is known about function point counting that it is possible to make easy comparisons with other products and other installations’ productivity.

### It is important that your client be told at this stage what the product is likely to cost. You usually express this amount as the total cost to complete the product, but you may also find it advantageous to point out the cost of the requirements effort, or the costs of individual requirements.

### Whatever you do, do not leave the costs in the lap of hysterical optimism. Make sure that this section includes meaningful numbers based on tangible deliverables.

# **25. User Documentation and Training**

## **25a. User Documentation Requirements**

### List of the user documentation to be supplied as part of the product.

### Technical specifications to accompany the product.

### User manuals.

### Service manuals (if not covered by the technical specification).

### Emergency procedure manuals (e.g., the card found in airplanes).

### Installation manuals.

## **25b. Training Requirements**

### A description of the training needed by users of the product.

# **26. Waiting Room**

### Requirements that will not be part of the next release. These requirements might be included in future releases of the product.

### 

# **27. Ideas for Solutions**

### When you gather requirements, you focus on finding out what the real requirements are and try to avoid coming up with solutions. However, when creative people start to think about a problem, they always generate ideas about potential solutions. This section of the template is a place to put those ideas so that you do not forget them and so that you can separate them from the real business requirements.