*Solution Approach*

# **Project Design Introduction**

In the ever-evolving landscape of electronic reliability, the menace of tin whiskers remains a formidable challenge. These microscopic, conductive filaments have the potential to disrupt the functionality of electronic devices, leading to catastrophic failures and significant financial losses. To combat this threat, our project endeavors to introduce a novel software tool that revolutionizes the assessment and mitigation of tin whisker-induced failures.

Tin whiskers pose a significant risk to electronic components, potentially causing short circuits and other electrical malfunctions. Traditional methods of assessing this risk lack sophistication and accuracy, leaving electronic systems vulnerable to unexpected failures. Our project seeks to address this gap by developing an advanced software tool capable of simulating and analyzing the impact of tin whiskers on printed circuit boards (PCBs).

Project Objectives:

Develop a user-friendly software interface for capturing 3D models of PCBs.

Implement algorithms to identify exposed conductors on the PCB surface.

Simulate the deposition and movement of detached metal whiskers on the PCB.

Enable users to define whisker characteristics and simulation parameters.

Utilize Monte Carlo simulations to assess the probability of whisker-induced failures.

Enhance the tool with time-dependent forces affecting whisker motion (e.g., airflow, vibrations, current).

Project Scope:

The project will focus on the development of a software tool using the Unity 3D Engine, leveraging its capabilities for 3D modeling and physics simulations. While the tool will primarily target tin whisker-induced failures, its modular design will allow for future expansion to address other reliability challenges in the electronics industry.

1. **System Overview**

The proposed software tool aims to revolutionize the assessment and mitigation of tin whisker-induced failures in electronic components. Built upon the Unity 3D Engine, the system combines advanced 3D modeling capabilities with physics-based simulations to provide users with a comprehensive understanding of tin whisker behavior on printed circuit boards (PCBs).

At its core, the system will facilitate the creation of virtual PCB models and the identification of exposed conductors. Users will be able to define the characteristics of detached metal whiskers, simulate their deposition and movement on the PCB surface, and analyze the probability of whisker-induced failures using Monte Carlo simulations.

Key design considerations include user-friendliness, modularity, and scalability. The software will feature an intuitive user interface, allowing users to easily navigate through the modeling and simulation processes. Its modular design will enable future enhancements and extensions to address emerging challenges in electronic reliability. Additionally, the system will be scalable to accommodate varying levels of complexity in PCB designs and whisker behaviors.

Through this system, electronic engineers and reliability professionals will gain valuable insights into the risk posed by tin whiskers and will be empowered to proactively mitigate these risks, thereby enhancing the reliability and longevity of electronic systems across diverse industries.

The subsequent sections will delve deeper into the specific functionalities, design methodologies, and implementation details of the software tool, providing a comprehensive understanding of its capabilities and potential impact.

1. **Architecture Design**
2. **Overview**

This section should describe the overall architecture of your software. The architecture provides the top-level design view of a system and provides a basis for more detailed design work. This will be the initial draft of your software architecture. Next semester you will revise this draft and finalize your design.

* Provide a bird’s-eye view of your software architecture. Mention the architectural pattern you adopted in your software and briefly discuss the rationale for using the proposed architecture (i.e., why that pattern fits well for your system).
* Please refer to CptS 322, CptS 487, CptS 321, and CptS 422 materials to refresh your knowledge on system decomposition and software architectural patterns.
* Briefly describe each layer/component in the architecture and explain its responsibilities.
* Provide a block diagram (e.g., UML component diagram) that illustrates the proposed architecture. The block diagram should show all major subsystems and identify the layers/components in the architecture.

1. **System Decomposition**

This section explains how you decomposed your system into subsystems. A subsystem typically corresponds to the amount of work that a single developer can tackle. You will show your system decomposition, identify the major subsystems, describe the assignment of functionality to each subsystem, and define the interfaces between them. When you decompose your system into subsystems, you need to consider the dependencies within and between the subsystems, i.e., cohesion and coupling measures.

* Briefly explain how you decomposed your system into subsystems.
* Discuss the rationale for the proposed decomposition in terms of cohesion and coupling.
* Redraw your architecture diagram (in section III.1) and show all the services each subsystem provides and requires (for example, a UML component diagram that uses ball-and-socket notation to depict provided and required interfaces).
* For each subsystem in your architecture, include a sub-section.
* To improve clarity, you may provide multiple figures that show different parts of the architecture (illustrating services) and place each figure right before the corresponding subsection.
  1. **[Subsystem Name]**

Include the following sub-sections for each subsystem.

1. ***Description***

Describe the subsystem and identify its responsibilities.

1. ***Concepts and Algorithms Generated***

Discuss the concepts, algorithms, or solutions generated and considered for this subsystem. Report the selected solution and explain the solution selection process. Include any special considerations and/or trade-offs considered for the solution approach you have chosen.

1. ***Interface Description***

Provide a description of the subsystem interface. Explain the provided services in detail and give the names of the required services.

Services Provided:

1. Service name:

Service provided to: [list the receiving subsystems here]

Description: [Describe what the service is and what it does. Provide its input and output values. Briefly describe the major functions that the service provides.]

Services Required:

Names of the required services and the subsystems that provide them.

**[Include sections III.2, III.3, etc., for other subsystems]**

1. **Data Design**

[You may skip this section if your project doesn’t require any data manipulation or storage]

Data Structures:

1. Printed Circuit Board (PCB) Model:
   * The PCB model represents the physical layout of the printed circuit board, including its components, traces, and conductive paths.
   * Data structure: 3D mesh or graph representation, storing node and edge information to define the PCB topology.
2. Conductor Identification Data:
   * Data structure storing information about the location and properties of exposed conductors on the PCB surface.
   * Data structure: Array, list, or dictionary storing coordinates, conductivity, and other relevant attributes of exposed conductors.
3. Whisker Characteristics:
   * Data structure to define the characteristics of detached metal whiskers, such as material composition, length, thickness, and distribution.
   * Data structure: Object or dictionary storing whisker properties as key-value pairs.
4. Simulation Parameters:
   * Data structure to store user-defined simulation parameters, including the number of whiskers, simulation duration, and environmental conditions.
   * Data structure: Object or dictionary storing simulation settings as key-value pairs.
5. Simulation Results:
   * Data structure to store the results of whisker simulation, including information about bridged conductor pairs and probability of failures.
   * Data structure: Array, list, or database table storing simulation output for analysis and visualization.

Databases:

1. Simulation Data Repository:
   * Database to store simulation input and output data for future reference and analysis.
   * Tables:
     1. PCB models: Store information about PCB geometries and components.
     2. Conductor data: Store details of exposed conductors identified in PCB models.
     3. Whisker characteristics: Store parameters defining whisker properties.
     4. Simulation settings: Store user-defined simulation parameters.
     5. Simulation results: Store output data from whisker simulation, including bridged conductor pairs and failure probabilities.
2. User Settings Database:
   * Database to store user preferences and settings for customizing the software tool's behavior.
   * Tables:
     1. User profiles: Store user information and preferences.
     2. Default settings: Store default values for simulation parameters and tool configurations.
     3. These data structures and databases will serve as the foundation for storing and managing the information required for modeling, simulating, and analyzing tin whisker-induced failures in electronic components. Properly understanding and implementing these data structures and databases are crucial for ensuring the effectiveness and scalability of the software application.
3. **User Interface Design**

The user interface (UI) for the tin whisker simulation software will be designed with a focus on intuitiveness, ease of use, and visual clarity. It will consist of several key components to facilitate user interaction with the software tool. Below is a detailed description of each component, accompanied by mock-up images:

1. Main Dashboard:
   * The main dashboard serves as the entry point for users and provides access to various functionalities of the software tool.
   * Use cases: Launching simulations, accessing PCB modeling tools, viewing simulation results.
   * Mock-up:
2. PCB Modeling Tool:
   * This tool allows users to create and modify 3D models of printed circuit boards (PCBs) by adding components, traces, and conductive paths.
   * Use cases: Creating new PCB models, editing existing PCB layouts.
   * Mock-up:
3. Conductor Identification Interface:
   * Users can use this interface to identify and mark exposed conductors on the PCB surface, either manually or through automated detection algorithms.
   * Use cases: Identifying exposed conductors for whisker simulation, adjusting conductor properties.
   * Mock-up:
4. Whisker Simulation Configuration:
   * This interface allows users to specify simulation parameters such as whisker material, density, length distribution, and environmental conditions.
   * Use cases: Setting up whisker simulation scenarios, defining simulation parameters.
   * Mock-up:
5. Simulation Results Viewer:
   * Users can visualize and analyze the results of whisker simulations, including bridged conductor pairs and probability of failures.
   * Use cases: Analyzing simulation outcomes, identifying potential failure points.
   * Mock-up:
6. Settings and Preferences:
   * Users can customize the software tool's behavior and appearance through settings and preferences.
   * Use cases: Adjusting default simulation parameters, changing UI themes.
   * Mock-up:
7. Help and Documentation:
   * This section provides users with access to user manuals, tutorials, and technical documentation to assist them in using the software effectively.
   * Use cases: Accessing help resources, seeking assistance with software functionalities.
   * Mock-up:

These interface components will enable users to interact with the software tool seamlessly, from creating PCB models to simulating whisker events and analyzing simulation results. The design prioritizes clarity, simplicity, and functionality to ensure a positive user experience and effective utilization of the software's capabilities.