Software Safety Assignment 02

Assignment 2-01

Name and shortly describe the two redundancy tactics mentioned in the paper.

**Diverse Redundancy** is a technique used to enhance system reliability by introducing redundancy. It can be applied at either the specification or implementation level. When applied at the implementation level, redundant components use different implementations developed independently from the same specification. At the specification level, diverse redundancy goes a step further, requiring that even the requirement specifications for the redundant components are set up by individual teams. In simpler terms, it’s about having backup plans to handle failures and ensure system robustness.

**Replication redundancy** is a strategy used to boost system reliability by introducing duplicate systems that can detect or conceal random hardware failures. These redundant systems mirror the primary system's functionality, utilizing the same hardware and software setup. For instance, RAID1 storage technology duplicates data across disks, ensuring continuity even if one disk fails. Essentially, replication redundancy safeguards against hardware failures by having backup systems ready to seamlessly take over, ensuring uninterrupted operations.

Assigment 2-02

Which errors are solved by Heterogeneous Duplex Pattern, which cannot be addressed by Homogeneous Duplex Pattern and which are solved by both?

**Solved by both:**  
Random faults and transient faults are addressed by both the Heterogeneous Duplex Pattern and the Homogeneous Duplex Pattern. Both patterns use redundancy, consisting of a primary (active) channel and a secondary (backup) channel, along with fault detection and switching mechanisms to maintain system operation even when a fault occurs in one channel.

**Solved only by Heterogeneous:**  
Systematic faults, software bugs, and vulnerabilities are specifically addressed by the Heterogeneous Duplex Pattern. This pattern uses diverse hardware and software implementations for the primary and backup channels, which reduces the risk of common-mode failures that could affect both channels in a homogeneous setup. By using different designs and different manufacturers, it reduces the likelihood that a design flaw or inherent issue in the hardware or software will impact both channels simultaneously.

Assigment 2-03

**System with a Fail-Safe State**

**Chosen Pattern: Protected Single Channel Pattern**

* **Reason for Choice:** The Protected Single Channel Pattern is designed to handle transient faults and provide a cost-effective solution for light safety-critical applications. It includes mechanisms for input data validation, actuation monitoring, and the ability to enter a fail-safe state if a primary channel fault is detected. This is suitable for systems where a fail-safe state can be reliably entered upon fault detection.
* **Reason Against M-out-of-N Pattern:** The M-out-of-N Pattern involves multiple redundant channels and a voting mechanism to determine the correct output. This pattern is more complex and expensive to implement. For a system with a fail-safe state, the added complexity and cost may not be justified, as entering a fail-safe state upon fault detection can already ensure safety and reliability.

**System without a Fail-Safe State**

**Chosen Pattern: M-out-of-N Pattern**

* **Reason for Choice:** The M-out-of-N Pattern provides high fault tolerance by using multiple identical or diverse channels operating in parallel, with a voter deciding the correct output based on the majority. This pattern is well-suited for systems without a fail-safe state as it ensures continuous operation even if one or more channels fail, maintaining system availability and reliability.
* **Reason Against Protected Single Channel Pattern:** The Protected Single Channel Pattern relies on entering a fail-safe state upon fault detection, which is not feasible for systems that cannot afford to shut down. This pattern does not provide the necessary redundancy and fault tolerance needed to keep the system operational in the absence of a fail-safe state.

**System with a fail-safe state:**

I would select the Protected Single Channel Pattern for this system because it's designed to handle transient faults and offers a cost-effective solution for light safety-critical applications. It includes mechanisms for input data validation, actuation monitoring, and the ability to enter a fail-safe state upon detecting a primary channel fault, ensuring safety and reliability.

I would not select the M-out-of-N Pattern because it's more complex and expensive to implement. For a system with a fail-safe state, the added complexity and cost may not be necessary because entering a fail-safe state upon fault detection can already ensure safety and reliability.

**System without a fail-safe state:**

I would select the M-out-of-N Pattern for this system because it provides high fault tolerance by using multiple redundant channels operating in parallel. A voter decides the correct output based on the majority, ensuring continuous operation even if one or more channels fail, maintaining system availability and reliability.

I would not select the Protected Single Channel Pattern because it relies on entering a fail-safe state upon fault detection, which is not feasible for systems that cannot afford to shut down. This pattern lacks the necessary redundancy and fault tolerance needed to keep the system operational in the absence of a fail-safe state**.**