Failure Alert System for a Nuclear Power Plant

By: Alex Bepple, Ben Lea, Sanyam Gupta

Introduction

Introduction

- Almost 15% of the world’s energy is generated from nuclear power plants, but how do we ensure they’re safe to use?
- Disasters at plants like Chernobyl and Fukushima were caused by a lack of proper safety measures.
- With the use of the Ada programming language a safety critical software can be created. Like one that can sound early warning and alarms so techs and engineers can be made aware of anything that can go wrong.

Design Problem

Design Problem

- The safety of a nuclear power plant is critical to public wellbeing. To help aid in ensuring a plant operates safely a failure alert system using Ada has been made.
- The system itself should also be fault tolerant, if a sensor fails, gives bad data or a controller goes down, the whole system should not fail.

Functions

- User Interface: Clearly display important information with a reasonable learning curve
- Alarms: Display several severity levels
- Data generation: Reasonable and random changes to best show variety of outputs

Objectives

- Fault tolerant: No single point of failure should exist within the program
- Scalable: The program allows for easy addition or removal of sensors
- Maintainable: Well designed code that’s easy to understand

Objectives

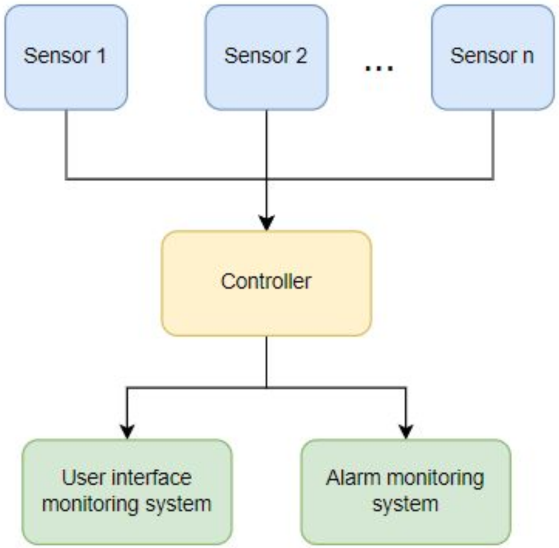
- Economic factors: The system must be designed to be cost effective throughout its life span.
- Environmental Sustainability:Detect any abnormalities that may lead to accelerated wear on the system which may lead to a reactor melt down. Improve efficiency and minimize nuclear waste.

Constraints

- The prototype was to be finished by March 31 2023
- The prototype could only be written in Ada
- No hardware implementation is currently possible so data generation must be done via the software

Solutions

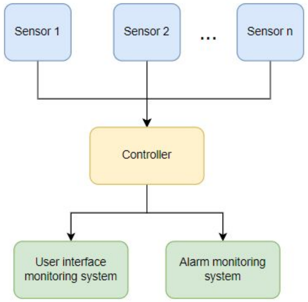
Solution 1 System Architecture



Solution 1

Pros:

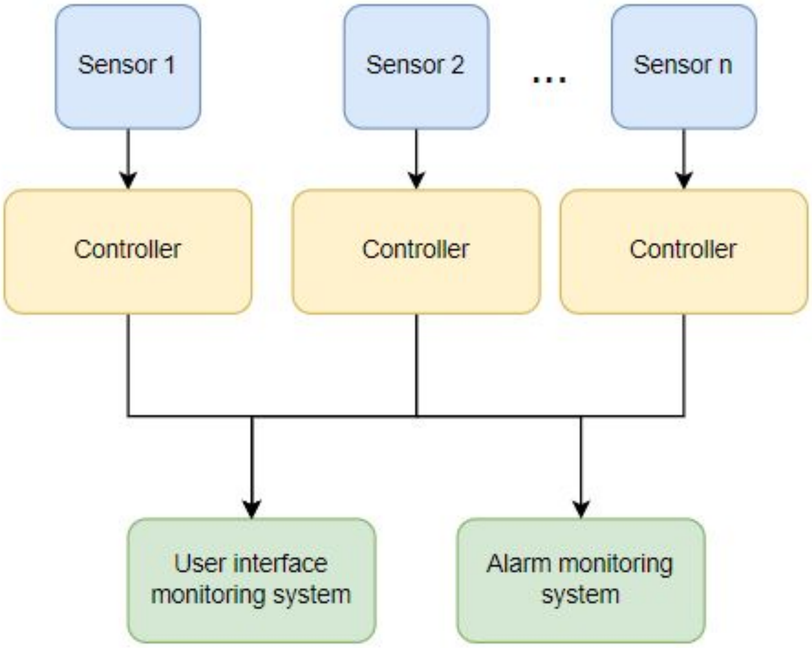
- Easy to implement
  - Shared memory
- Minimal computation power needed
  - Single controller decides if alarm should go off or not
- Low latency
  - When dealing with a minor number of components



Cons:

- Single point of failure
  - 1 Controller
- Not easily scalable
  - Shared memory has issues when dealing with multiple components
- Not practical in physical application
  - Components that are far away from each other physically need to send data using another communication method

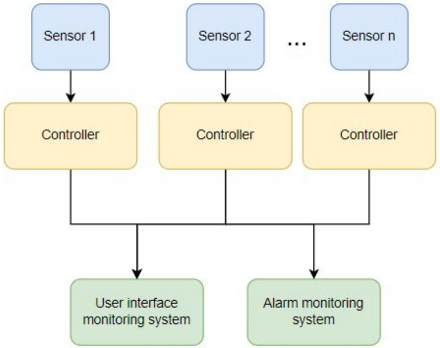
Solution 2 System Architecture



Solution 2

Pros:

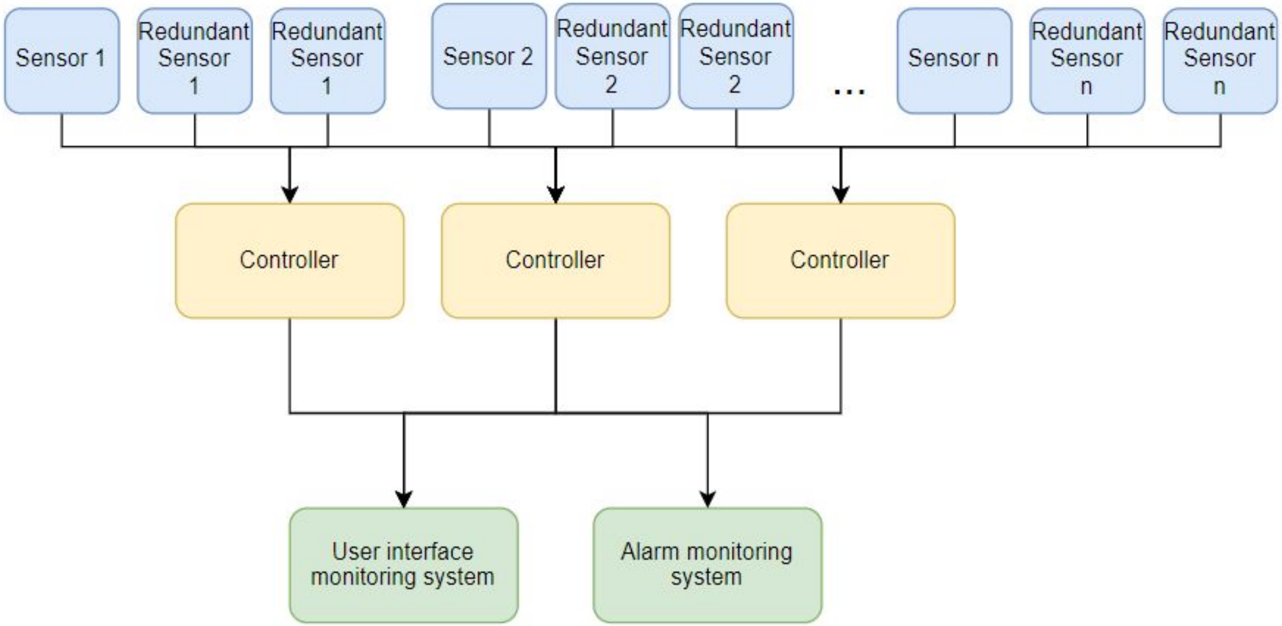
- Relieves single fail point from S1
    - >1 CONTROLLER
- Uses TCP communication
    - Dependable
    - Congestion free
    - Hi reliability



Cons:

- Series layout
    - Single failure in path problem
    - Low error traceability
- TCP downsides
    - Higher overhead
    - Larger latency

Final solution System Architecture



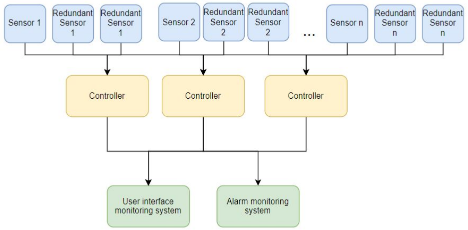
Final Solution

Pros:

- Redundant sensors & controllers
  - High error traceability
  - High reliability
- TCP + UDP communication
  - Low latency data sending
  - High reliability connection

Cons:

- Increased complexity of system
  - Higher cost
- More concurrent tasks necessary
  - Setups
  - Data transmission
  - Logical operations



Conclusion & Future Work

Conclusion

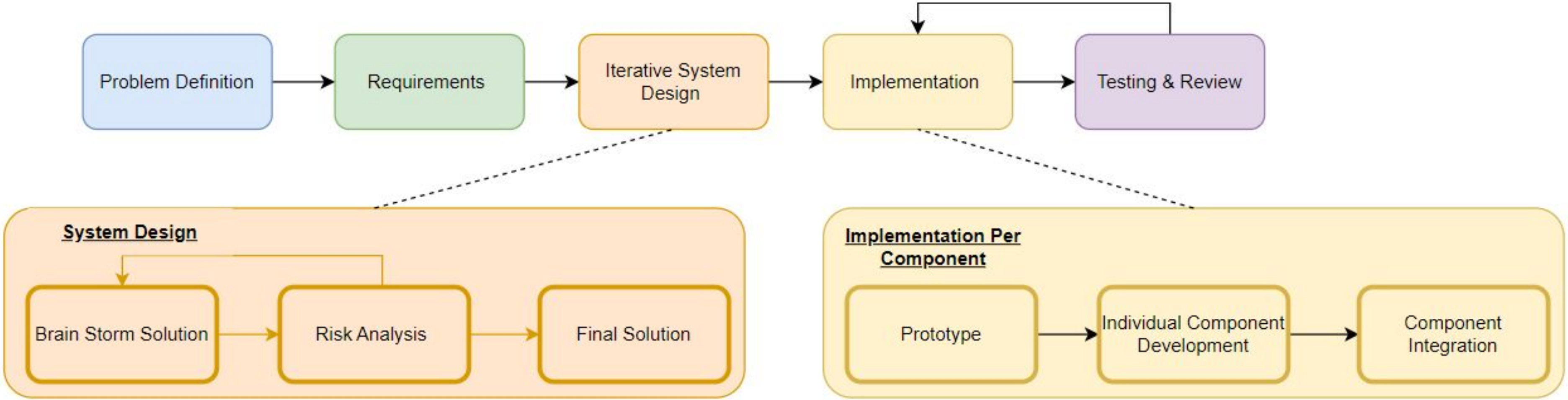
- Made in Ada to be safety critical and allow for
  - Fault tolerance
  - Scalability
  - Maintainability
- Communication over TCP and UDP
  - Creates reliable, secure TCP connections for setup and alarm messages
  - Uses much faster UDP sending for data transfer
    - TCP connections should align with the UDP data addresses
      - This gives us reliability checking for UDP at receiver end

Future Work

- Physical implementation of sensor
  - Eliminates need for code to generate random theoretical values
- Advanced alarm UI and implementation
- Running the software in a nuclear power plant simulator
- Controllable systems in a real reactor environment
  - Add system control functionality in the case of warnings or error

TeamWork & Project Management

- Collaboration Tools:
  - Git
  - Google Docs & Slides
- Project Life Cycle
  - Peer review was completed at the end of each phase to ensure team was ready to move onto following phase.
  - Collaborative peer review for each component at testing & review phase.



References

1. “Ada programming/libraries/gnat.sockets,” *Wikibooks, open books for an open world*. [Online]. Available: https://en.wikibooks.org/wiki/Ada_Programming/Libraries/GNAT.Sockets. [Accessed: 28-Mar-2023].

2. “Pressurized water reactor,” *Wikipedia*, 09-Feb-2023. [Online]. Available: https://en.wikipedia.org/wiki/Pressurized_water_reactor. [Accessed: 14-Mar-2023].

3. AdaCore, “‘learn.adacore.com,’” *learn.adacore.com*. [Online]. Available: https://learn.adacore.com/. [Accessed: 28-Mar-2023].