DIE PROJECT

Reliability evaluation of distributed embedded systems

GROUP-1

Masa
Cirkovic

Bashir
Alam

Kaf
Shahrier

Mete Harun
Akcay

AGENDA

01 Introduction to DES

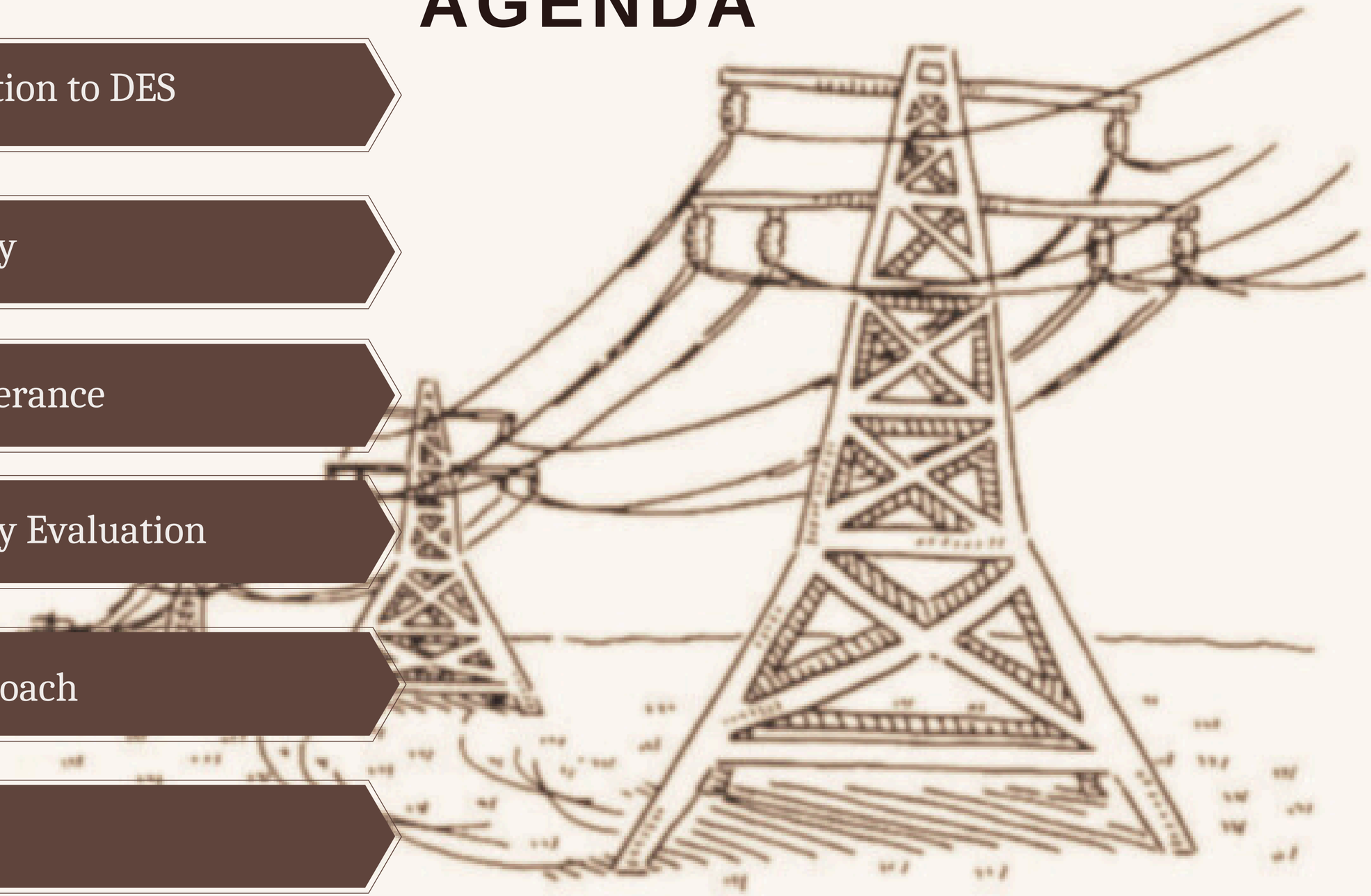
02 Reliability

03 Fault Tolerance

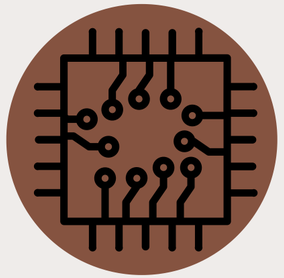
04 Reliability Evaluation

05 Our Approach

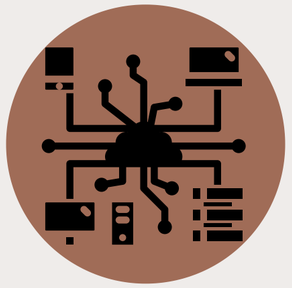
06 Our Aim



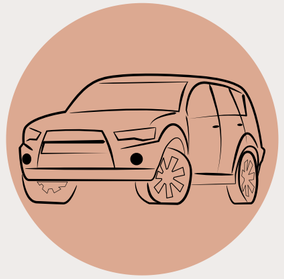
INTRODUCTION



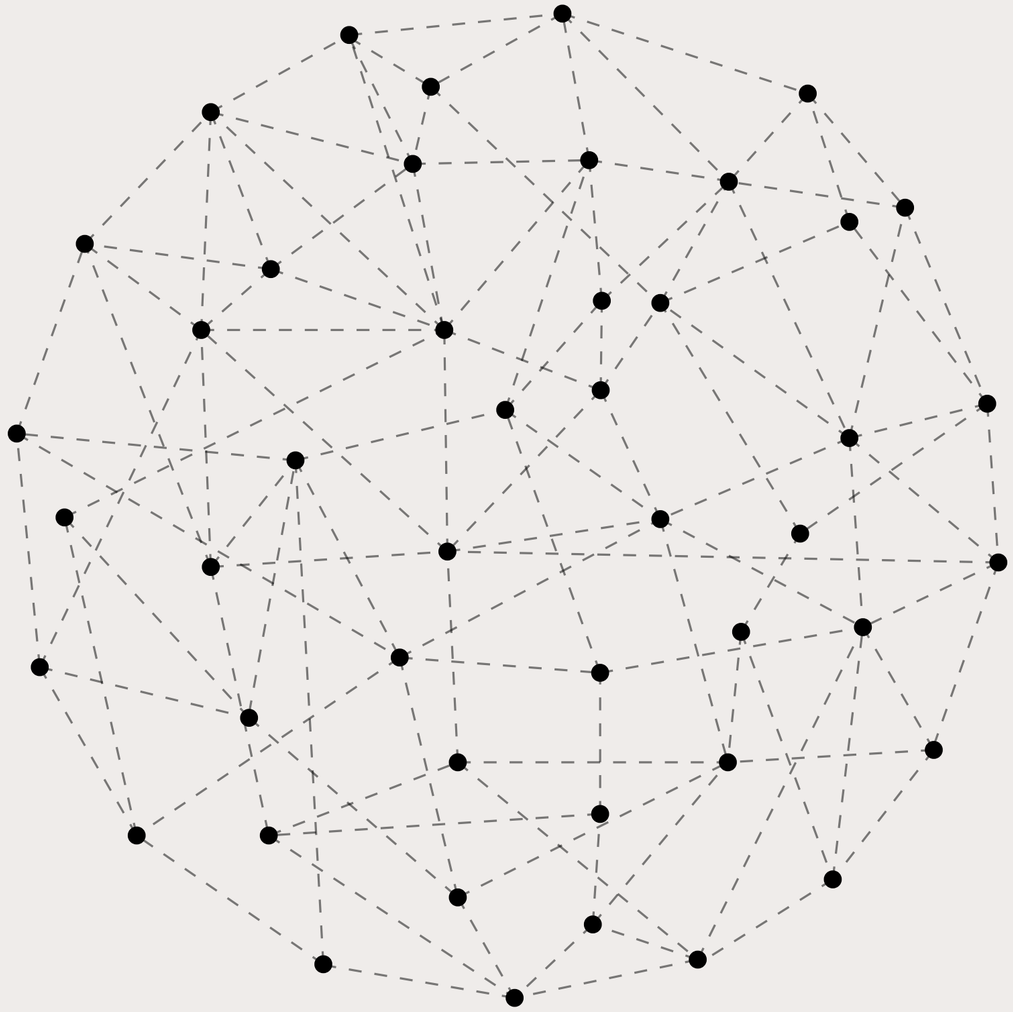
Embedded systems



Distributed systems



Distributed embedded systems



EMBEDDED SYSTEMS

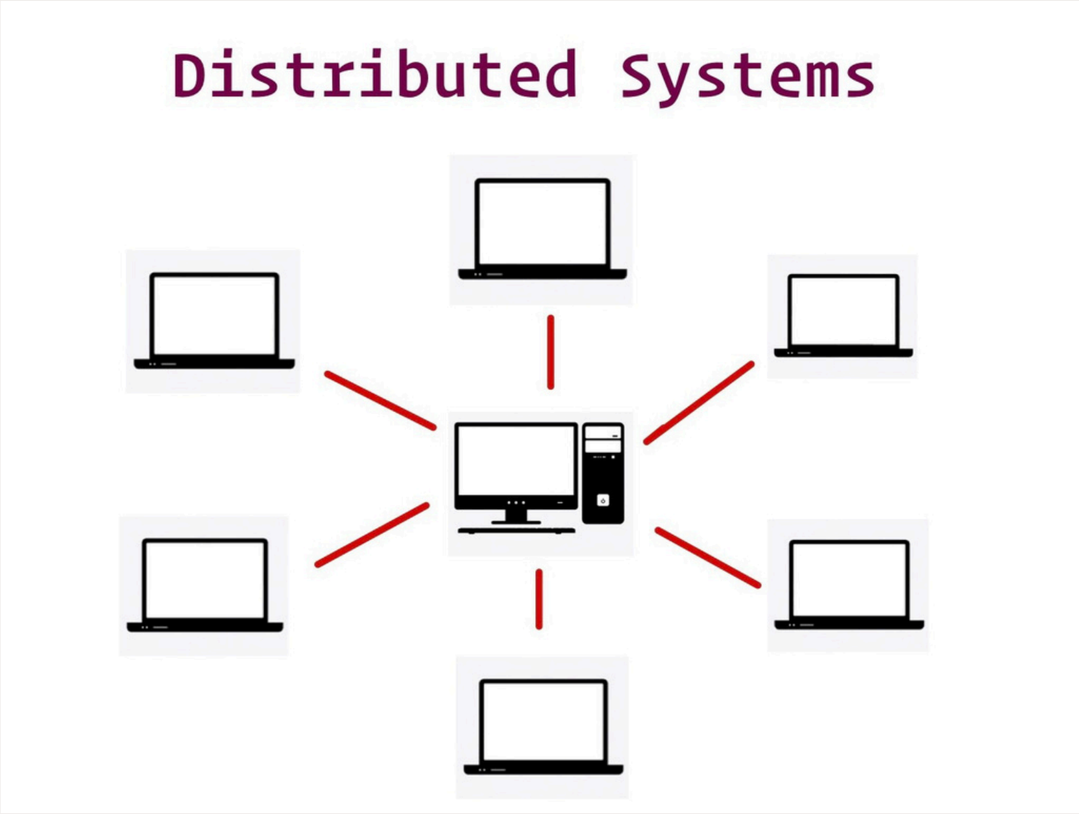
Embedded System

- Specialized computing system
- Part of a larger mechanical or electrical system
- Embedded into devices
- Highly optimized for performance, power, and size
- Real-time constraints
- Microcontrollers in home appliances, medical devices, consumer electronic, automotive control systems

DISTRIBUTED SYSTEMS

Distributed System

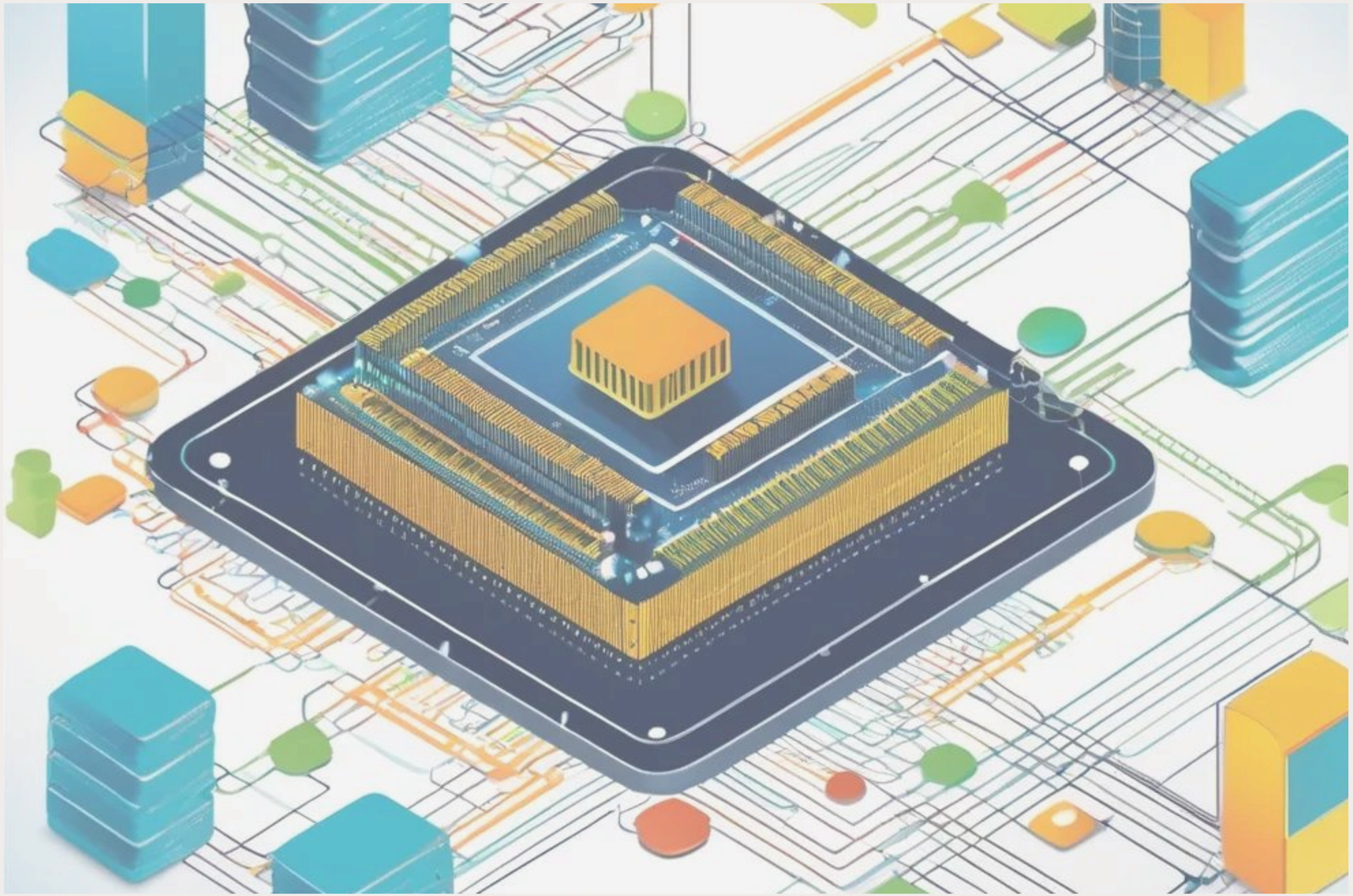
- Collection of components which appear as one to the end user
- Communicate by passing messages
- Work together to achive a specific objective
- Share tasks and responsibilities
- Scalability, fault tolerance and parallel processing
- Problems with synchronization, fault management and latency



DISTRIBUTED EMBEDDED SYSTEMS

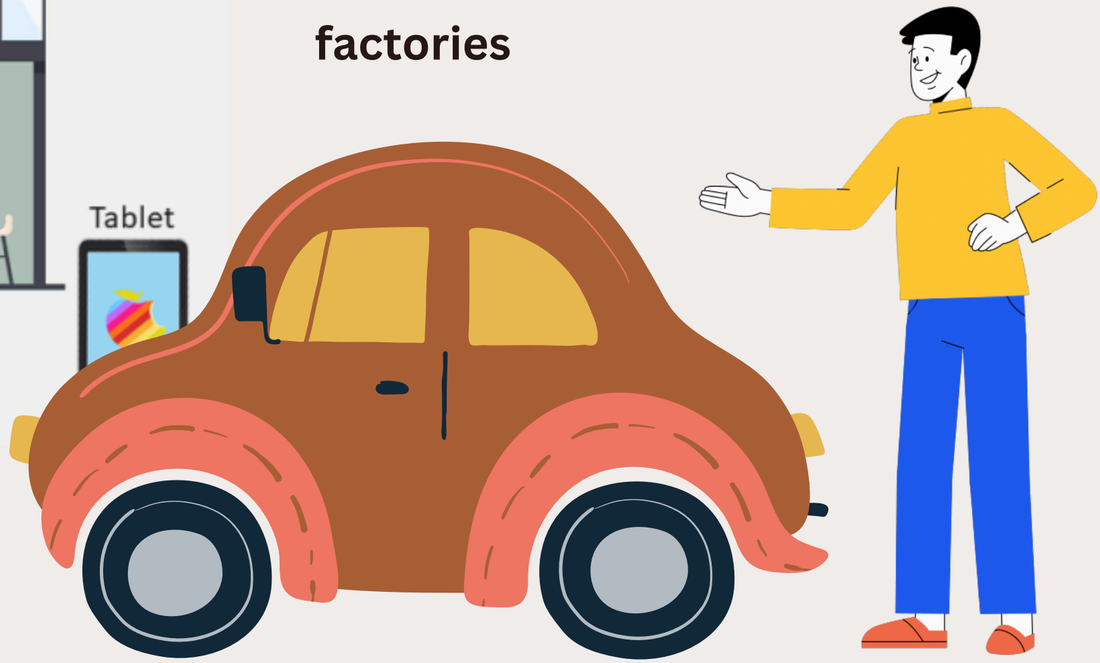
Key Features

- Multiple embedded untis (PE) - can be DSP, CPU, microcontroller, sensor, actuator
- Data exchange – wired and wireless protocols
- Real-time operations
- Coordination and synchronization in time and function



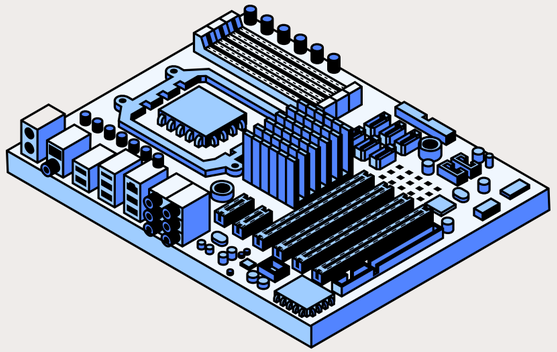
APPLICATIONS OF DES

- Electronic control units - modern cars have 70-100 ECUs - tasks like engine control, brake control, airbag deployment
- Anti-lock Braking System
- Home automation - smart homes
- Medical devices for monitoring patients
- Process control and robotics in factories



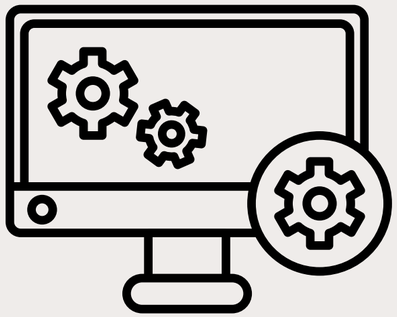
RELIABILITY

- ***Ability of a system to function correctly over time, consistently producing correct outputs***
- ***Failure can have serious consequences since these systems often manage real-time processes***



Hardware Reliability

- Embedded systems deployed in harsh environments, so the hardware components need to be robust
- Components can degrate over time - design with that in mind

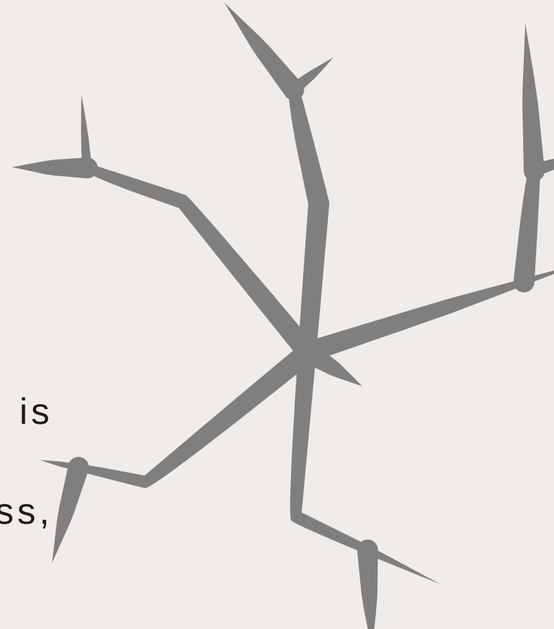


Software Reliability

- Bugs or design flaws can be a problem
- Software must be thoroughly tested

Network Reliability

- Nodes rely on communication to function collectively - network is crucial
- System must account for network congestion, packet loss, downtime, etc.



RELIABILITY

ROBUST TESTING

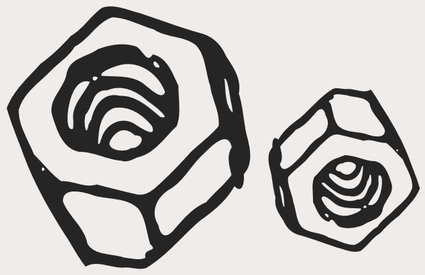
- Stress testing
- Fault injection

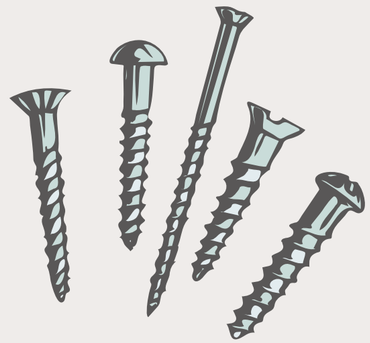
ERROR DETECTION AND CORRECTION

- Checksums
- Parity bits
- Other ECC

REDUNDANCY

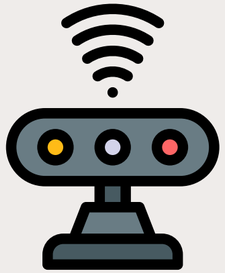
- Backup systems
- Backup components





FAULT TOLERANCE

- ***Ability of a system to continue functioning even when parts of it fail***
- ***Redundancy as the most common technique to achieve fault tolerance and better reliability of a system***



Hardware Redundancy

- Multiple sensors, processors, or actuators performing the same task in parallel
- If one fails, others can take over

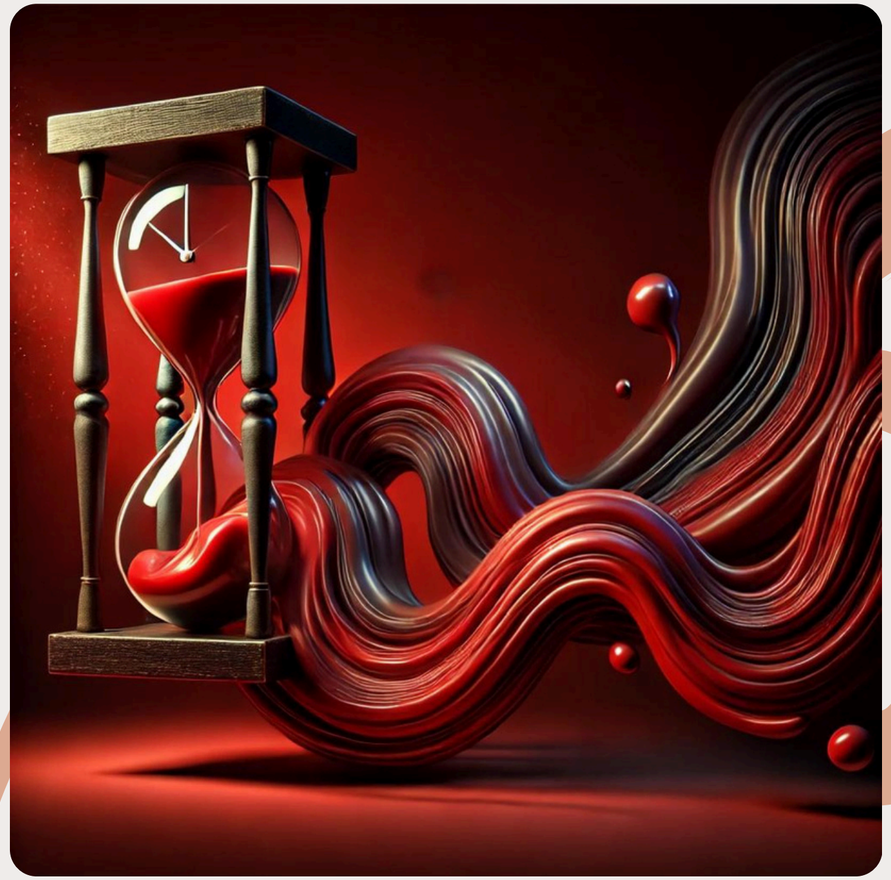
Software Redundancy

- Duplicate software processes or modules running independently
- If one module fails, another can continue the task

Data redundancy

- Data is replicated across nodes
- If one node fails, data is preserved elsewhere

Designing highly reliable systems is not that easy...



Costly in time

Dependent on the designer

RELIABILITY EVALUATION

The process of determining whether an existing system has achieved a specified level of operational reliability.

KEY FACTORS

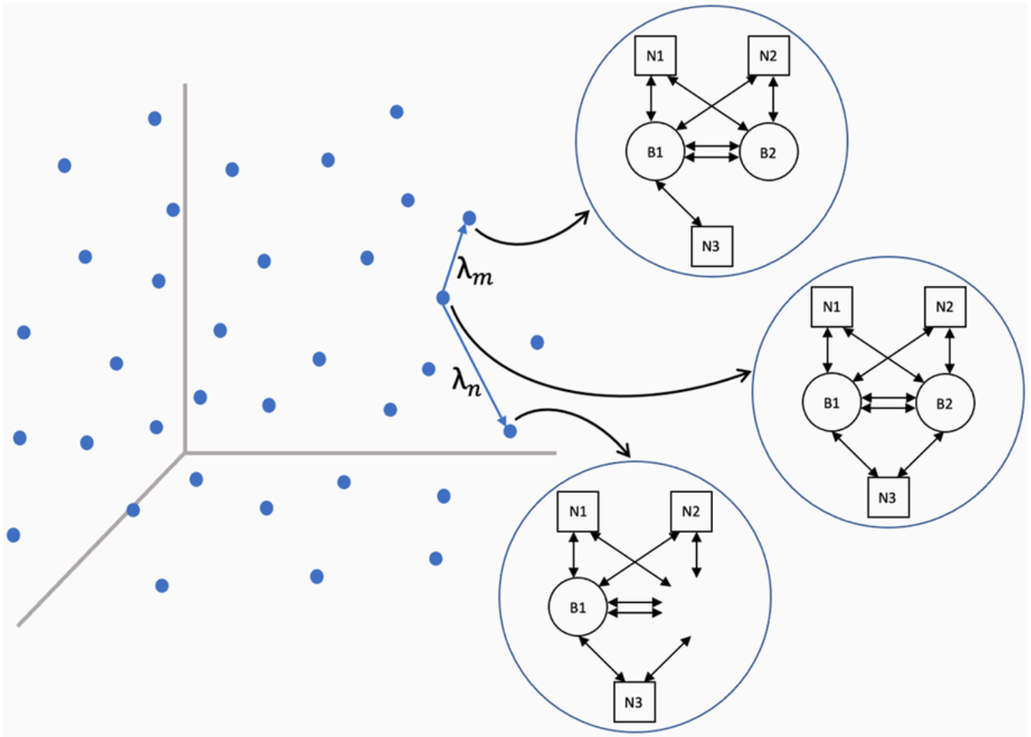
- **Ensures Dependability**
- **Minimizes Risk**
- **Enhances System Design**

Component failures

System redundancy

Environmental conditions

CURRENT PROGRESS



Design space of DES

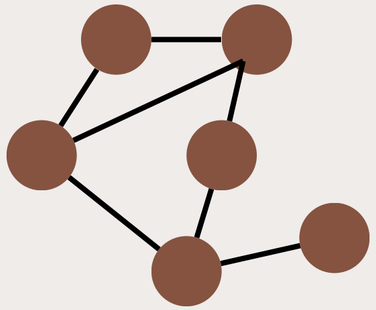
i) Define the Max Architecture

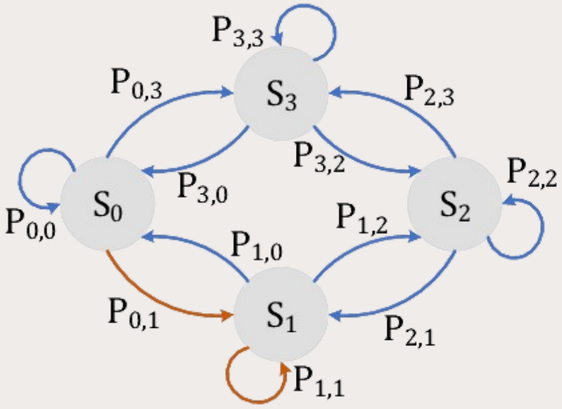
ii) Generate Subsets of Architectures

iii) Represent Architectures in a Graph

iv) Assign Failure Probabilities

CURRENT PROGRESS





PRISM

System Description

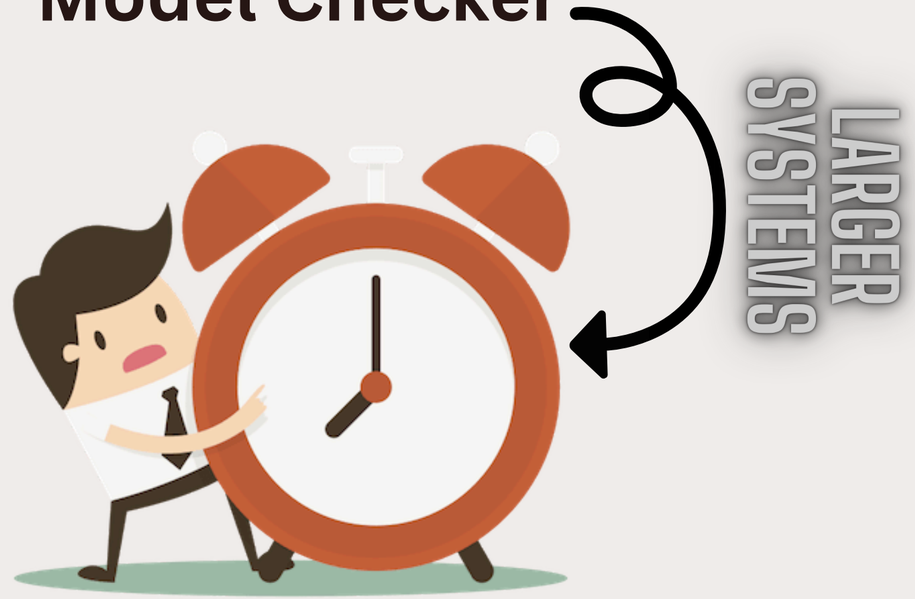
- Types of components
- How they are connected
- How they fail

Markov Chain

Model Checker

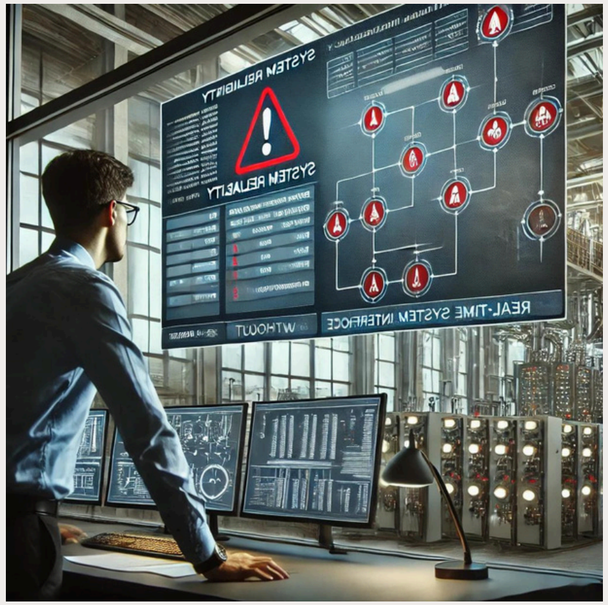
Architecture Restrictions

- Max number of components it can contain
- Min number of components required



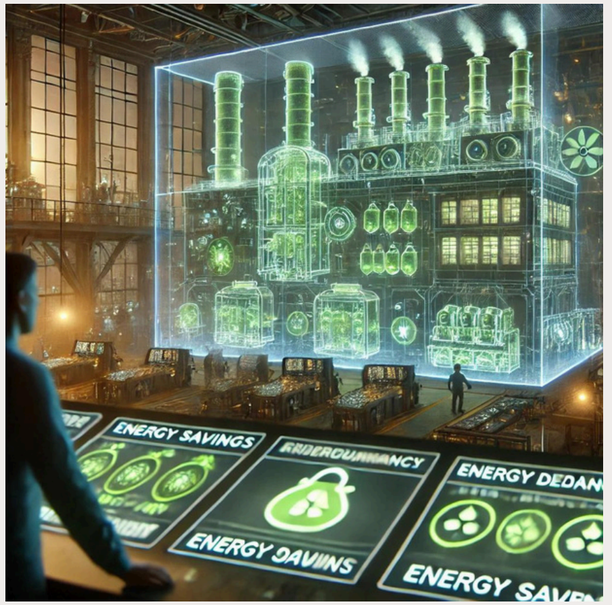
OUR APPROACH

Use machine learning techniques to *automatize* and *speed-up* the *reliability evaluation* after the failure of some components or after changes in the system's requirements.



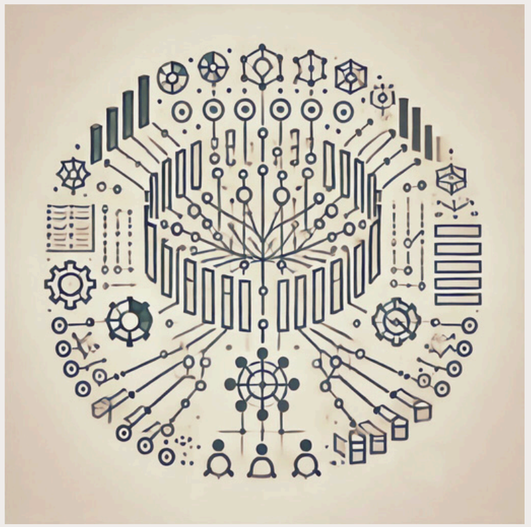
Alert

Re-evaluate



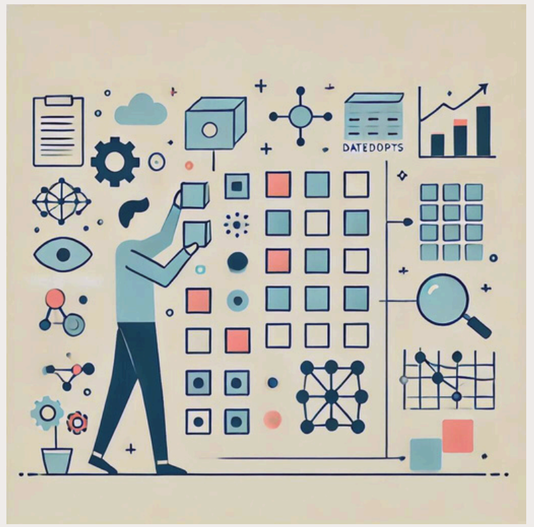
Adapt

PROJECT TIMELINE



Produce a dataset

Select the ML technique



Build, train and validate

OUR AIM

Automate the design of systems by leveraging the benefits of exploring the design space

QUESTIONS