



# AGENDA

1. Software Reliability -- Introduction
2. Software Reliability in Healthcare
3. Our Problem
4. Approaches and Experience
5. Current Work

# Introduction to Software Reliability

- Reliability : Ability of a system to function as per specifications under stated condition for a specified period of time
- Software reliability engineering – harder to implement:
  - Software engineering is new
  - Increased dynamics of software engineering projects
  - Increased complexity of software products

# Software Reliability in Healthcare

- Healthcare modalities: X-ray, MRI, CT, Ultrasound etc.
- Life critical applications
- Software – increasing both in size and importance
- In-process solutions are well-established
  - Interventional
  - Often not geared towards reliability
- Software reliability estimation and prediction not well-established
- Why is it important:
  - Resource allocation
  - Release timing

# Problem

*Input:* Defect log collected with no explicit stress to reliability specific data

*Problem:* Can we use the above data to estimate and predict the reliability growth of the project?

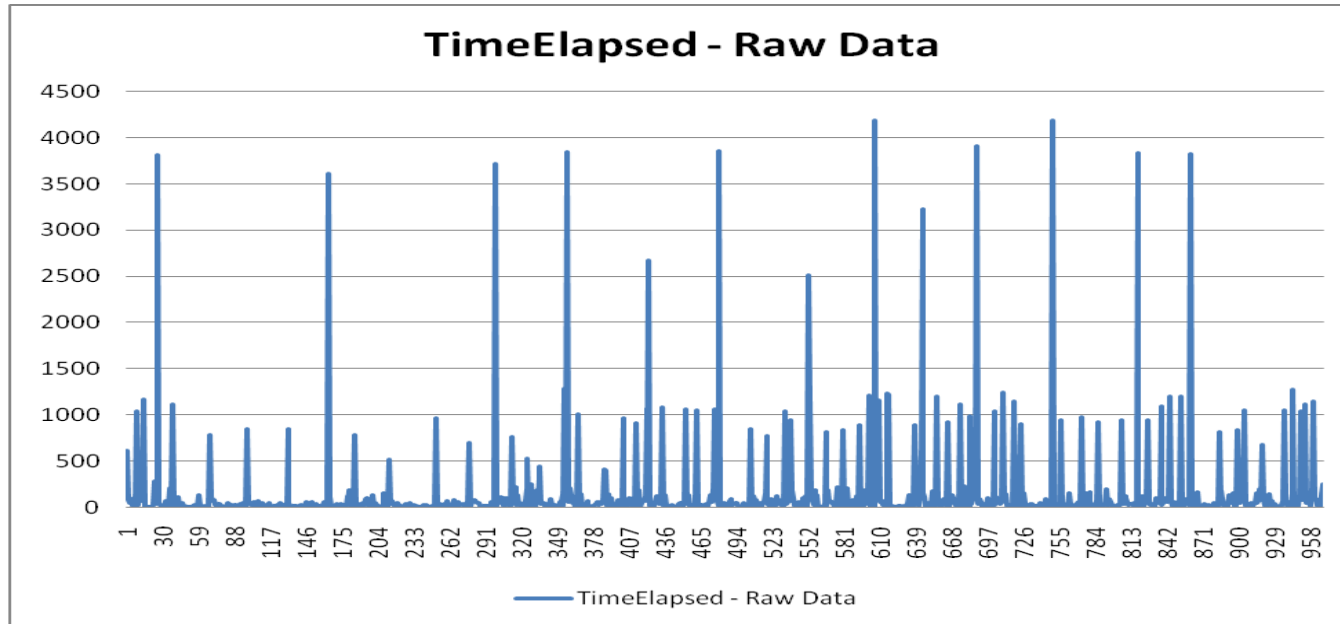
# Approach – Black Box Statistical Prediction

## Steps

- Evaluate and select models
- Fit defect data into the models
- Carry out reliability prediction
  
- Large number of models available
- Selected models: Musa Logarithmic, Goel Okumoto (NHPP), Jelinski Moranda, Weibull
- No *one-model-fits-all* promise
- No model fit our data!

# Sample Defect Data

- Time between failures curve of any one of the projects without



# Observations

- All models expect a generally decreasing trend in the failure intensity (increasing trend in time between failures)
- No such trend was observed in the data coming from our projects
- Assumptions of models
  - System maturity
  - Process stability
  - Testing effort
  - Operational profile

# Possible Reasons for Non-fit of Data

- Significant modifications during testing
- Stable bug injection rate not assured
- Dynamism in testing team
- Use case driven testing

# Possible Reasons for Non-fit of Data

## Significant modifications during testing

- Added modules are added sources of failures
- Modifications are added sources of failures
- Change in testing pattern

## Stable bug injection rate not assured

- Bug injection rate changes with change in maintenance conditions, e.g. competence, effort etc.

# Possible Reasons for Non-fit of Data

## Dynamism in testing team

- Variation in test effort, efficiency, competence

## Use case driven testing

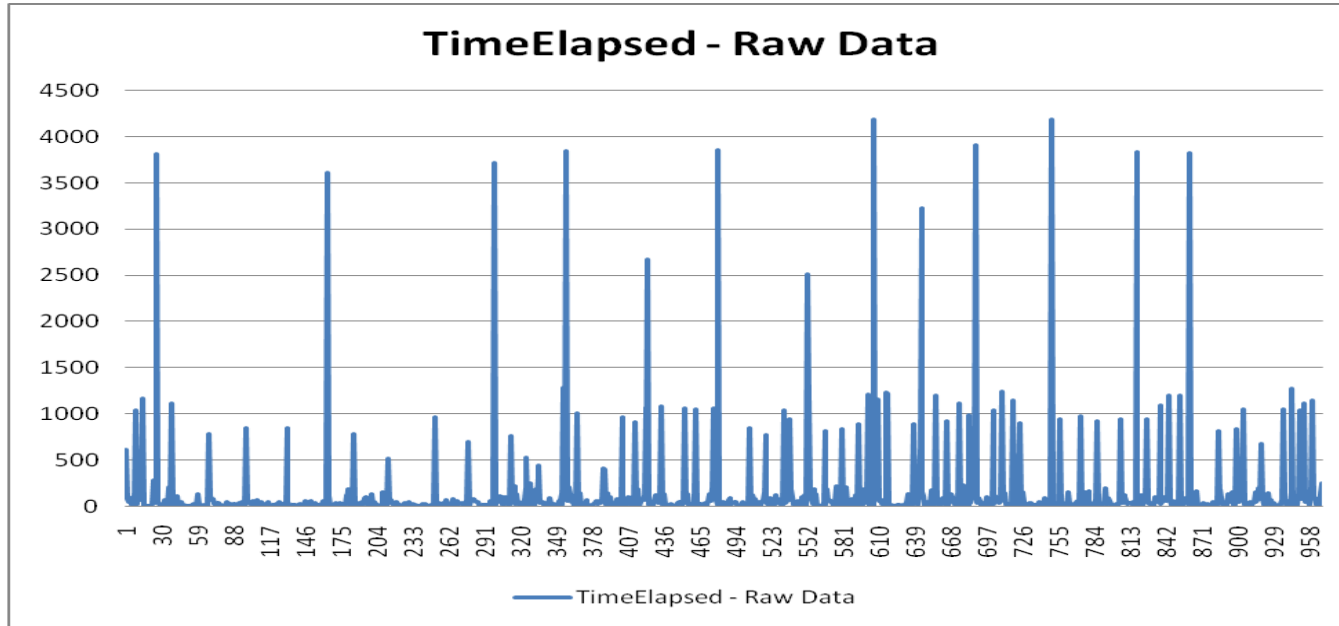
- Sequential coverage of features, components and subsystems
- Coverage criteria not considering operational profile
- Low priority bugs don't get fixed promptly; result in many failures

# Approach – Data Normalisation

- To account for the effect of project and organisation specific factors
- Factors
  - Calendar time
  - Subsystems
  - Test cycles
  - Priority

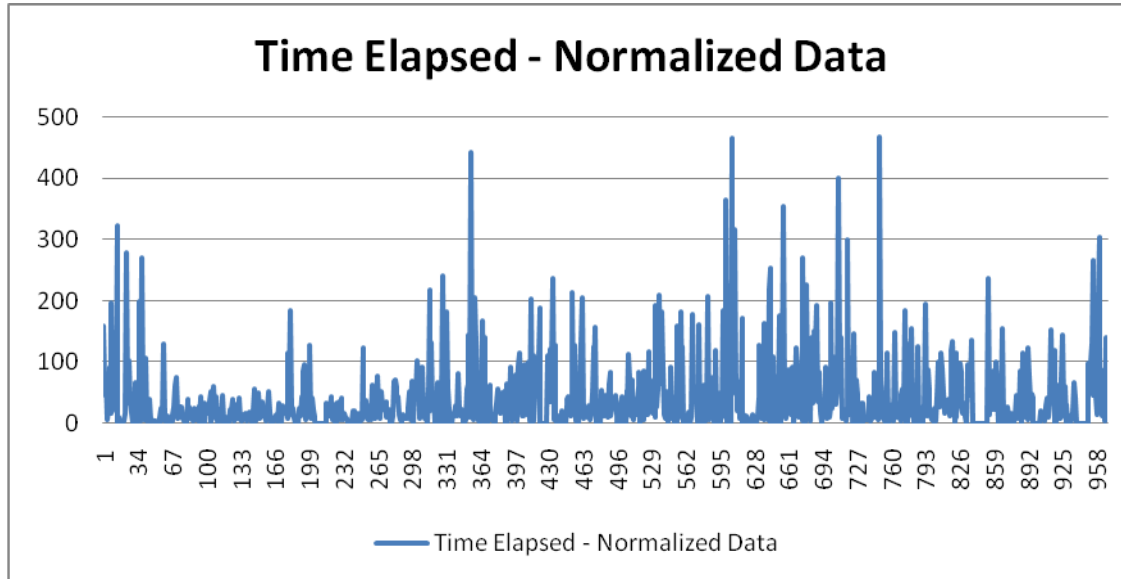
# Approach – Data Normalisation

## Calendar Time – Raw Data



# Approach – Data Normalisation

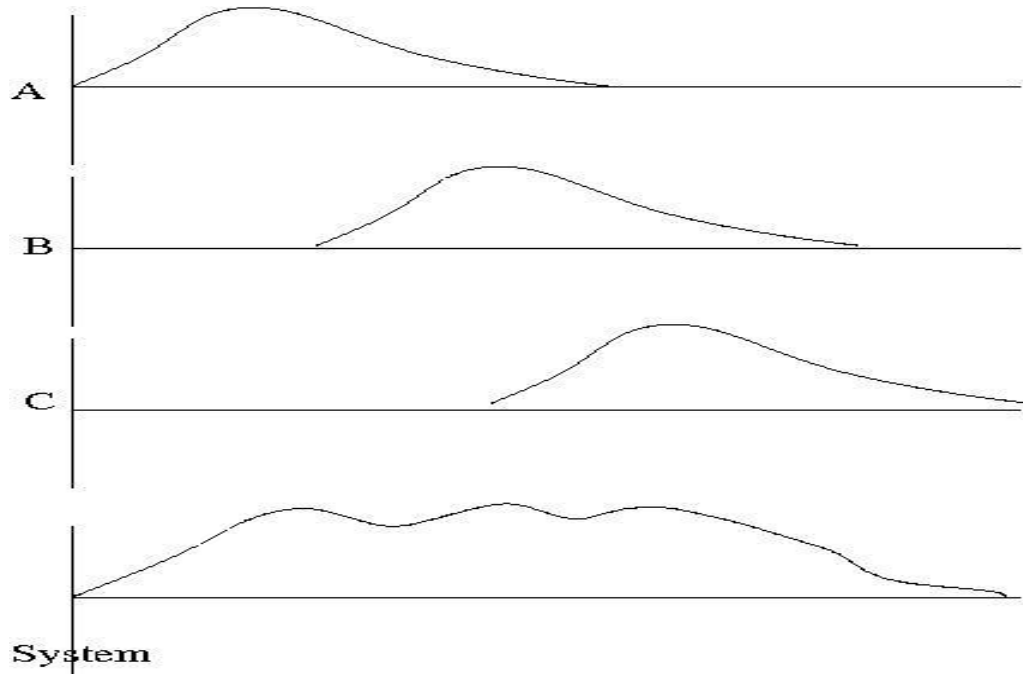
## Calendar Time – Normalised Data



Spikes smoothed after removing holidays and non-working hours

# Data Normalisation

## Subsystem

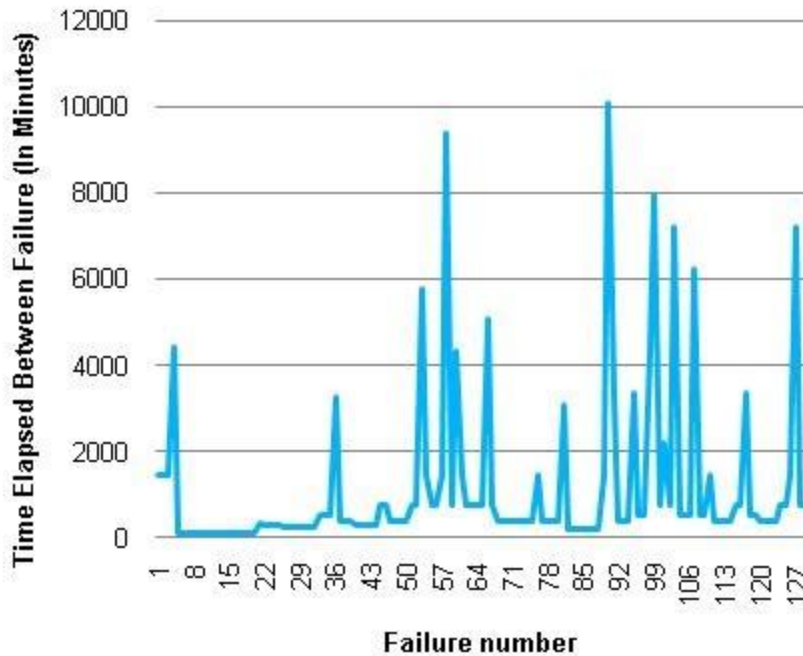


Subsystem specific testing may fit, but the overall system data mayn't!

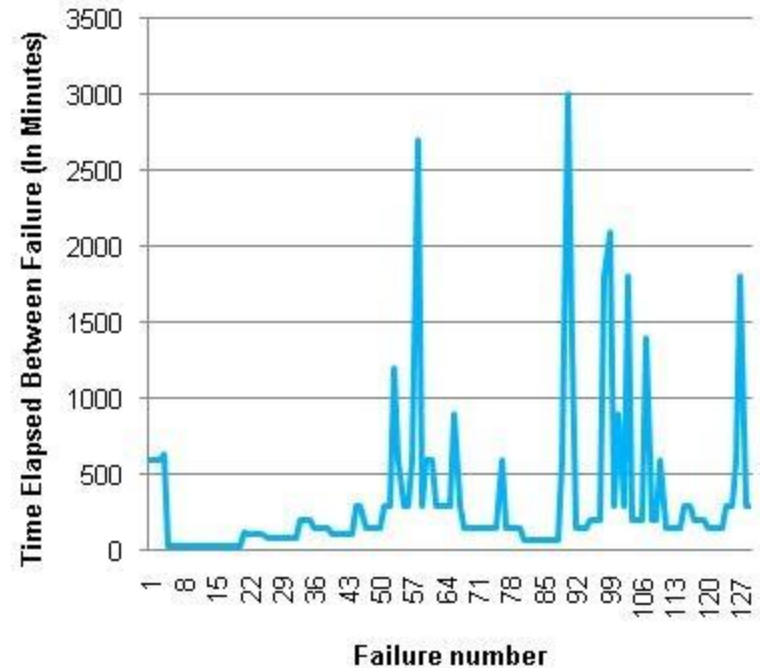
# Data Normalisation

## Subsystem

### SubComponent A-Raw



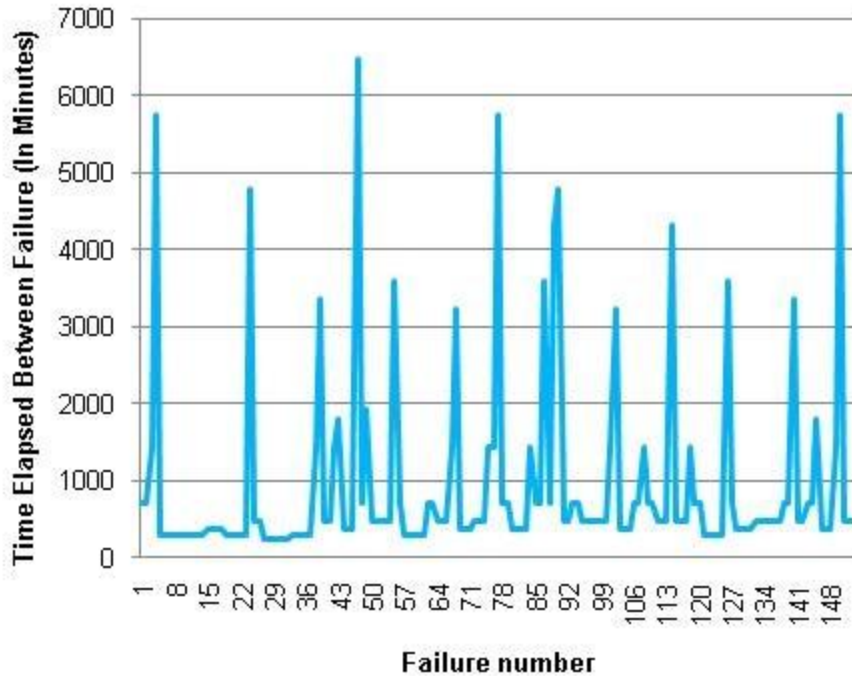
### SubComponent A-Normalized



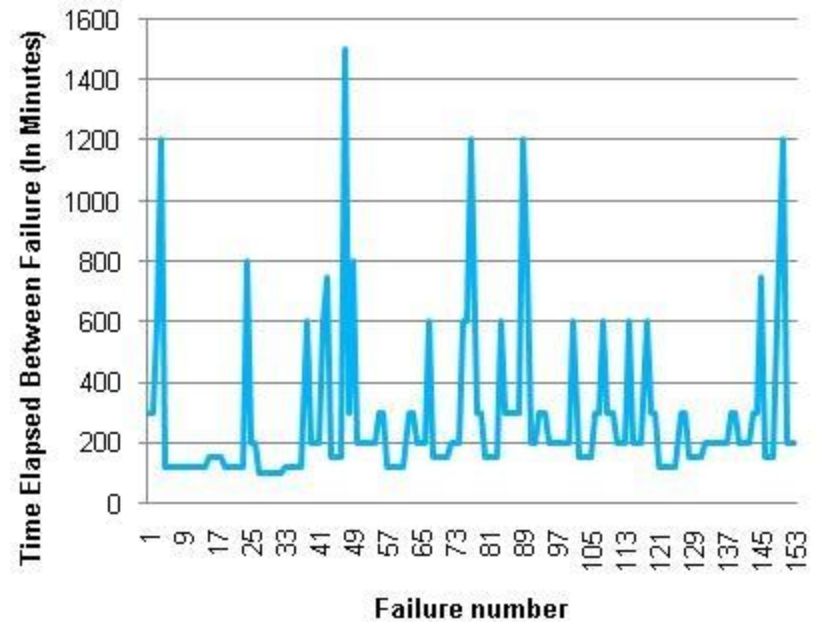
## Data Normalisation

### Subsystem

#### SubComponent B - RAW



#### SubComponent B - Normalized

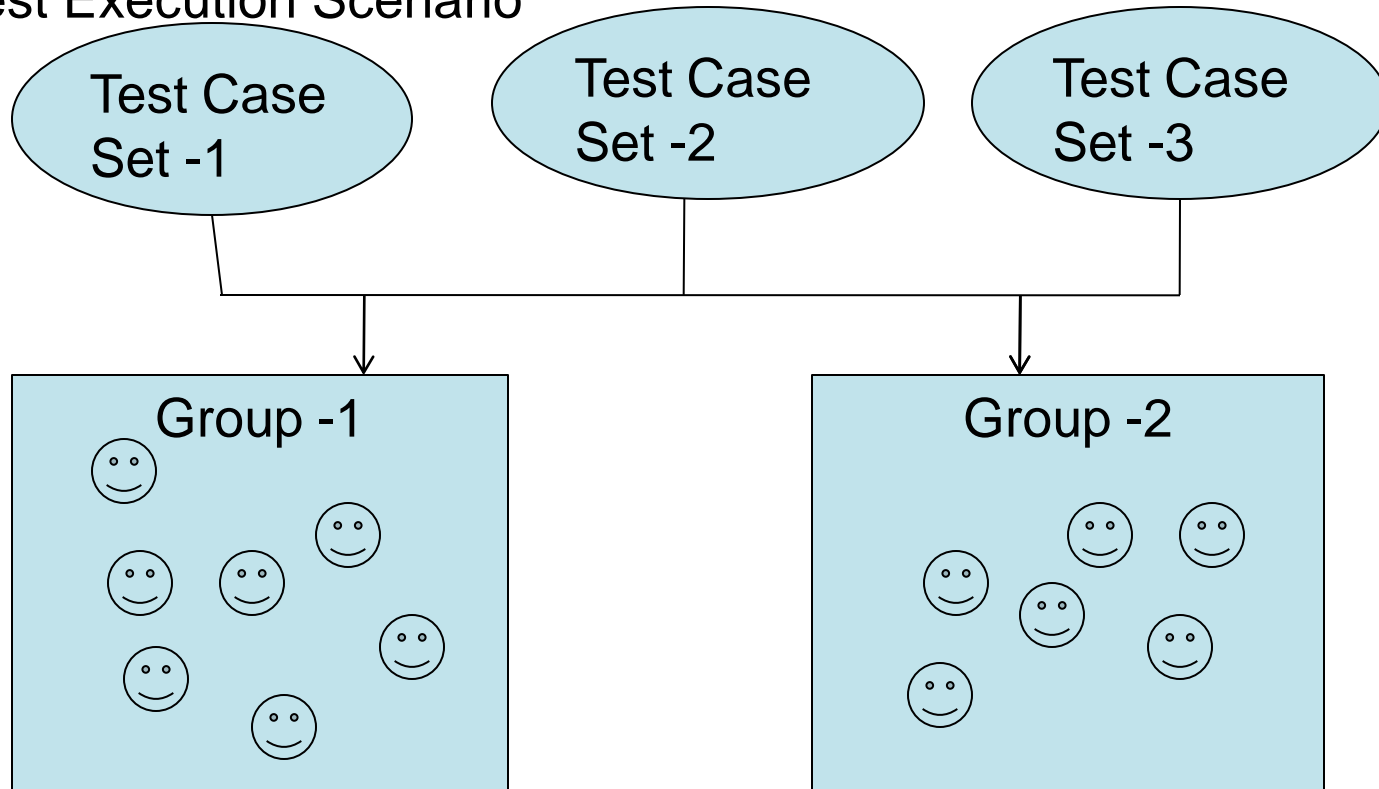


# Approach – Data Reordering

Reordering of defect data resulted in partly successful reliability prediction.

# Reordering of Data

A Test Execution Scenario



# Reordering of Data

- Group1 is asked for defect trends analysis and failure intensity estimation through specified models.
- Group1 executes the set of test cases in following order

- Test Case Set 1 - Day1
- Test Case Set 2 - Day2
- Test Case Set 3 - Day 3

DAY	Number Of Failures
1	3
2	12
3	31

- Verdicts of Test case execution shows
  - The defect intensity trends as increasing
  - Data does not fit into any of the specified models.
  - Models can not calculate the failure intensity estimation.

# Reordering of Data

- Group2 is asked for defect trends analysis and failure intensity estimation through specified models.
- Group2 executes the set of test cases in following order
  - Test Case Set 3 - Day1
  - Test Case Set 2 - Day2
  - Test Case Set 1 - Day 3
- Verdicts of Test case execution shows
  - The defect trends as decreasing.
  - Data does fit into specified models.
  - Models can calculate the failure intensity estimation.

Day	Number of Failures
1	31
2	12
3	3

# Reordering of Data

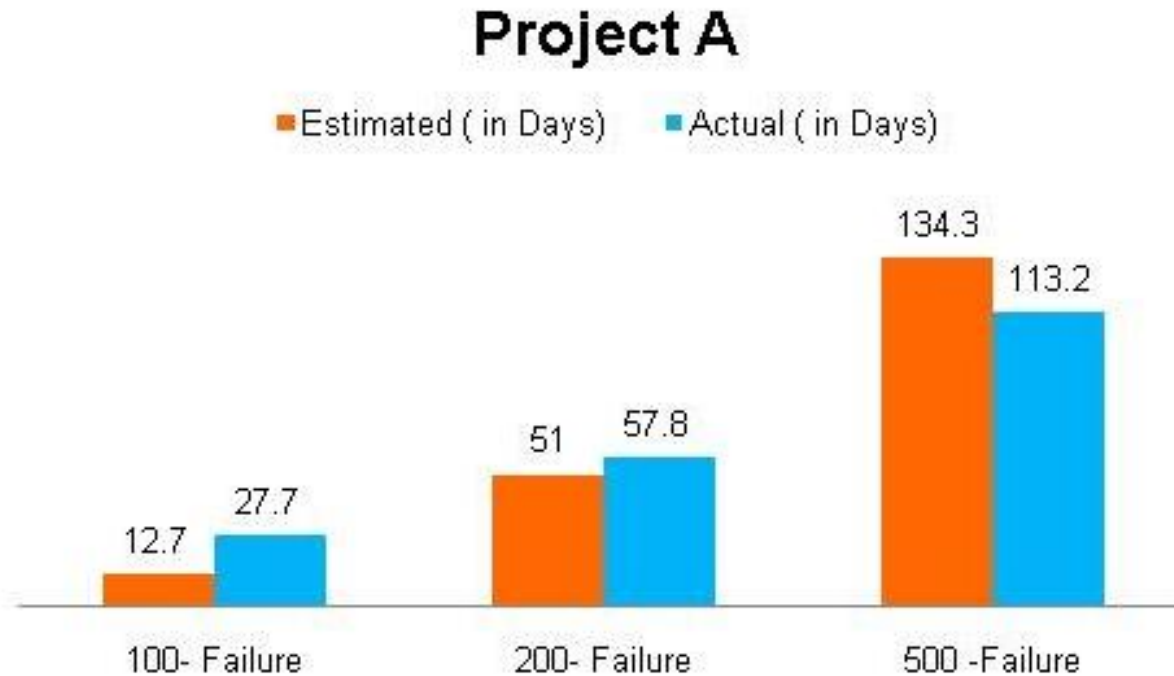
## Method

- Plot the time between failures curve
- Sort values in increasing order
- Fit model
- Predict reliability

## Invariants

- Total number of failures
- Average MTTF

# Reordering of Data



# Reordering of Data

## Observations

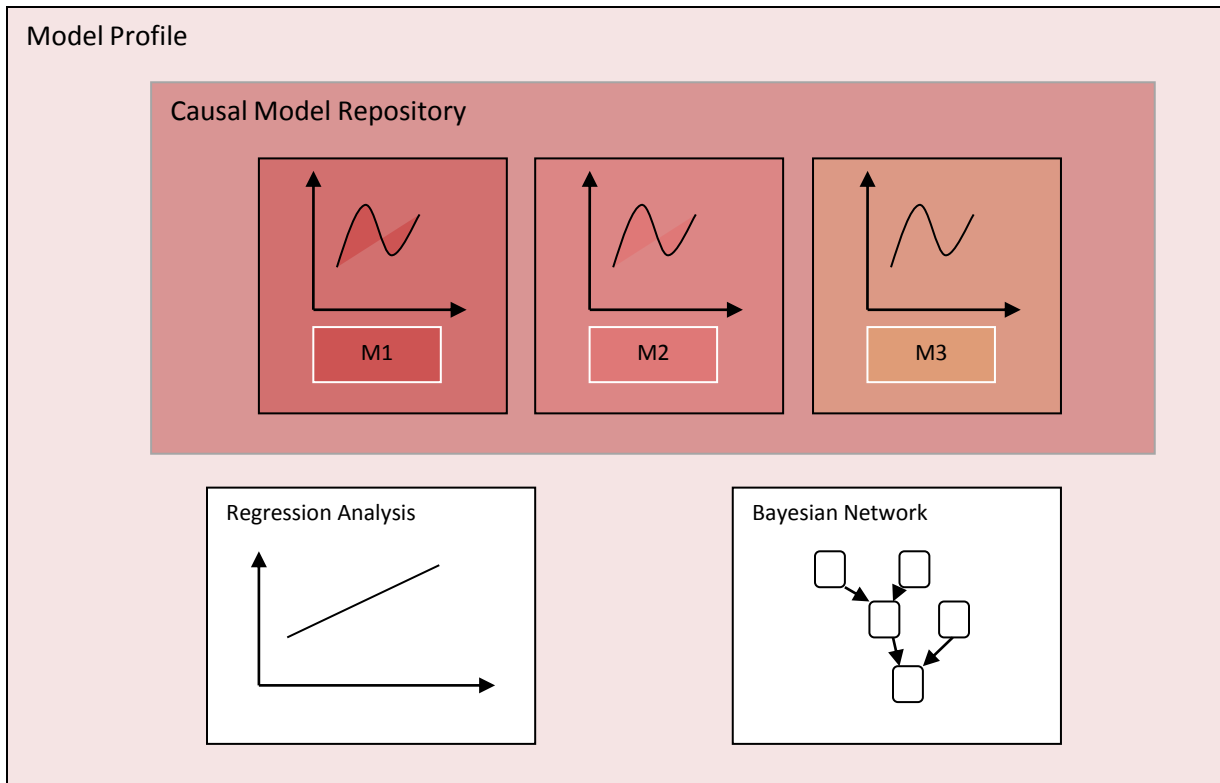
- Reordering resulted in data fitting models in some cases
- The reliability prediction from this fit generally better than constant MTTF prediction
- Issue: Theoretical soundness

# Current Work – Defect Data Normalisation Framework



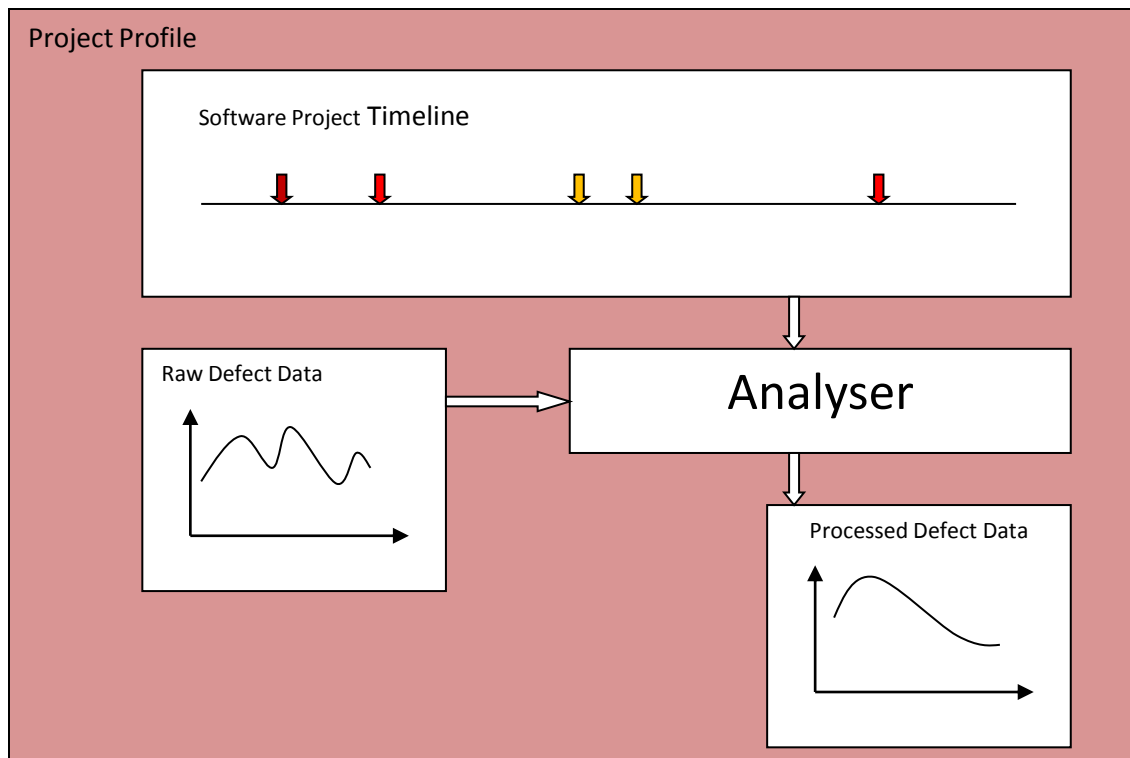
# Current Work – Defect Data Normalisation Framework

## Model Profile



# Current Work – Defect Data Normalisation Framework

## Project Profile



# Current Work – Defect Data Normalisation Framework

- Canonical relations to model effect of project and organisation specific factors on fault removal effectiveness and defect injection rate.
- Bayesian networks for modelling relations between project and organisation specific factors.
- Project profile is created from project data.
- Model profile can be used across similar projects.

