MISSION READY SOFTWARE
SOFTREL LLC

sales@softrel.com
321-514-4659
http://missionreadysoftware.com

Software Root Cause Analysis Services

The software defect Root Cause Analysis is **highly** recommended prerequisite for the software FMEA and any process related improvements. The RCA identifies the flavors of defects that are most common which improves the effectiveness of the other qualitative analyses and the quantitative assessments. The purpose of the defect root cause analysis is to identify:

- The development activity/artifact in which the defects are introduced
- The corresponding failure modes of each defect
- The most common root causes for the failure modes for each defect

More than 400 failure mode/root cause pairs are relevant for any type of software system.

- However, for any given software product a few dozen are typically relevant.
- The most common root causes in the problem reports are an indicator for the weaknesses in the specifications, design, and test procedures.
- The RCA provides supporting evidence of the weaknesses identified in the software reliability assessment.
- It also provides supporting evidence for the priority of the improvements to be made as a result of the assessment.

Example: The software reliability assessment indicates weaknesses in all areas of the design documentation including state diagrams, timing diagrams, data flow diagrams, etc. The software RCA indicates that there are significant problems due to design but the most common defect is from faulty state management. This provides quantitative evidence that the state design should be improved first before other design related improvements.

The artifacts that defect can originate in include but aren't limited to:

- Requirements – the requirement itself is incorrect and needs to be changed or there is a missing requirement
- Interface design – the interface protocol or message had to be changed or there is a missing interface design
- Detailed design – the states, timing, logic, sequence, logic, algorithm, etc. or lack of definition thereof were the cause
- Installation package – the install or update script is the cause
- Maintenance change -defect introduced while changing the code
- Usability – the default involves the use case, faulty assumptions about the end user, overly complicated processes, etc.
- Externally developed/acquired software – a change was made because of a defect in other code externally developed

The below is an **example** of a Pareto analysis of the artifacts which contribute to the defects. The Pareto can and will be different for every application. **It is guaranteed that the profile for your software will be different than what is shown below.** Your organization will receive a detailed Pareto as part of this task.

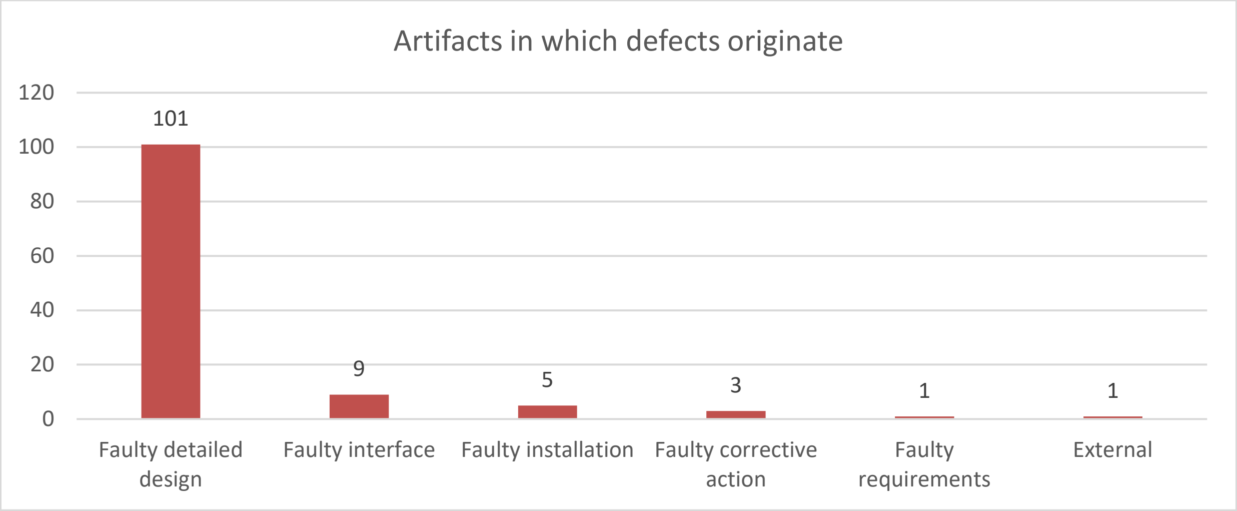

Figure 1 **Example** of Pareto chart of defects by artifact

The software related failure modes include but aren't limited to:

- Faulty functionality – the code doesn't do as required or performs a function that isn't required
- Faulty sequences – commands are executed in the wrong order or wrong state
- Faulty timing – the commands execute in the correct order but too early or too late
- Faulty data – the data is defined in the wrong format, size, unit of measure, etc.
- Faulty error handling – the software fails to detect errors, detects errors that it shouldn't, detects errors but fails to recover from them, detects errors and executes the wrong recovery
- Faulty logic – The logic is incomplete, reversed or missing
- Faulty algorithms – the algorithms are wrong, too accurate, not accurate enough, etc.
- Faulty database, COTS or OS interface – the interface to external software is faulty
- General coding – the software engineer understands the requirements and design but makes a mistake coding it. Examples are compilable typos, displays don't look good, wrong text on screens, etc.

Figure 2 below is an **example** of the defects organized by failure mode. Figure 3 is an example of the defects organized by root cause. **It is guaranteed that the profile for your software will be different than what is shown below.** Your organization will receive a detailed failure mode Pareto as part of this task. The assessment recommendations will address improvements for the most common failure modes.

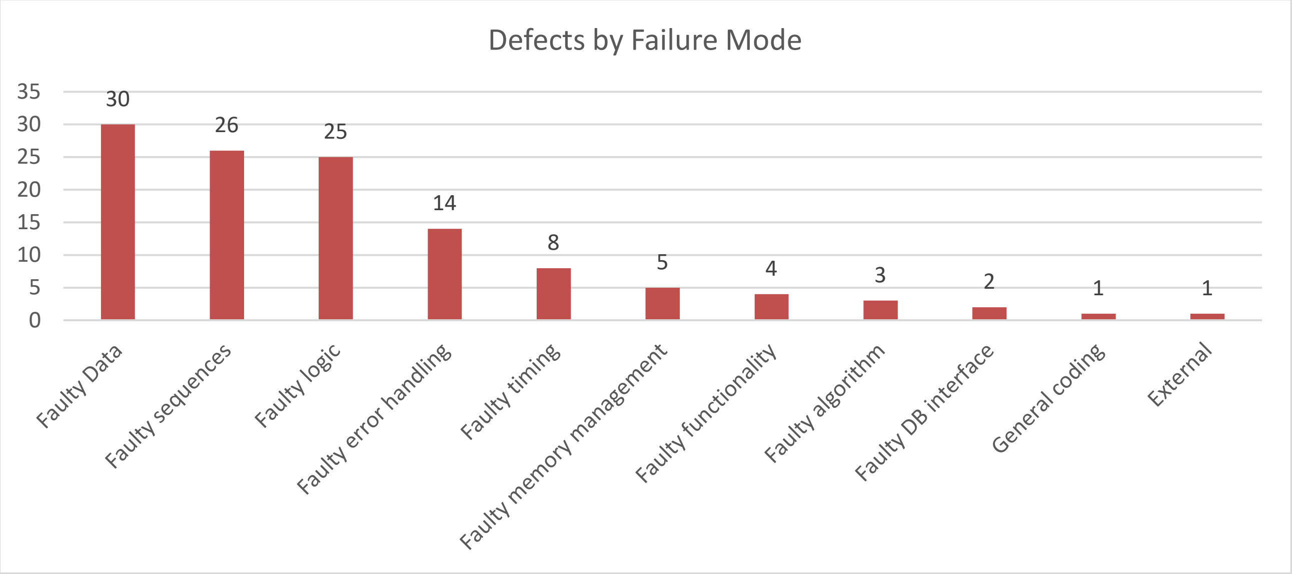

Figure 2 – Example of Pareto of defects by failure mode

The below is an **example** of the same defects organized by root cause. Your organization will receive a detailed Pareto of defects by root causes. **It is guaranteed that the profile for your software will be different than what is shown below.** The most common root causes are incorporated into the short term improvement recommendations.

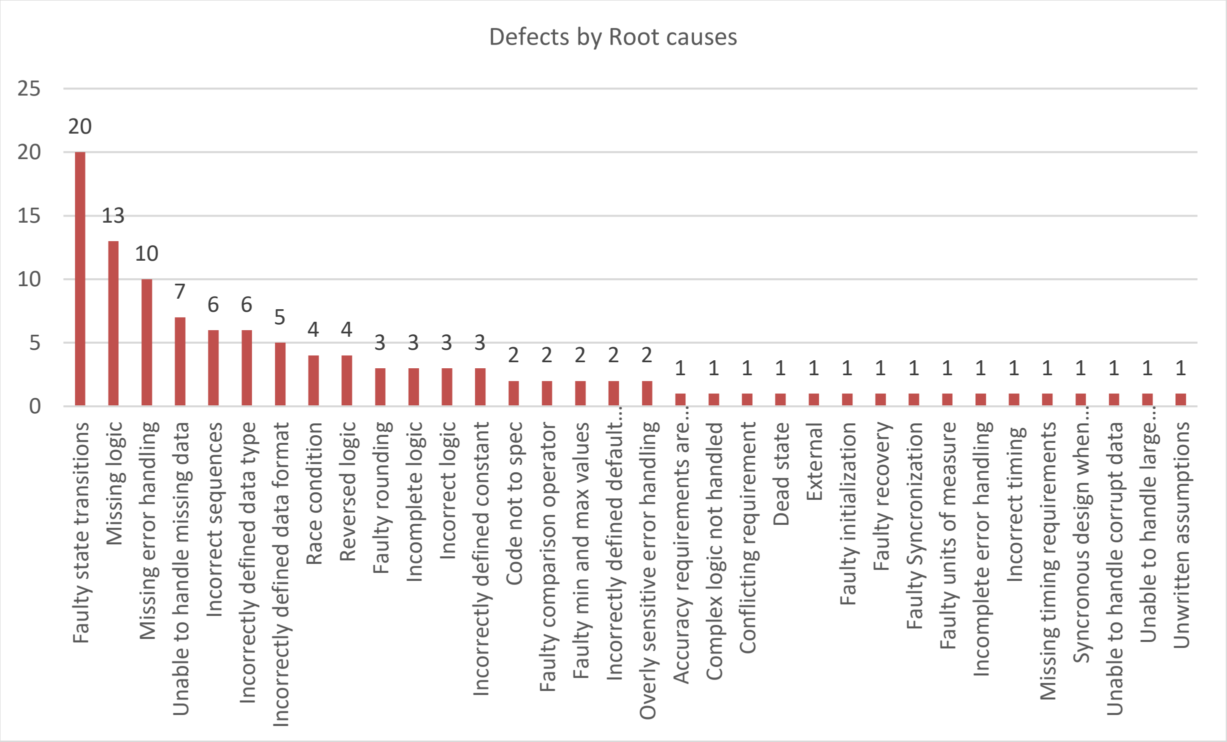

Figure 3 – Example of Pareto of defects by root cause

The most common failure modes identified by the RCA are often directly linked to specific weaknesses in the development practices identified by the reliability assessment. The below table summarizes the relationships between key development weaknesses and failure modes:

Failure modes/Root causes	Associated development practice weakness
Faulty state management – prohibited states are allowed, inadvertent state transition, dead states, orphan states, missing safe state	Failure to employ state diagrams or state tables as part of the design activity
Software cannot handle faulty data – corrupt data; missing, null or blank data; data in wrong unit of measure; improperly formatted data; improperly scaled data	Failure to design and define data types and sizes, unit of measure, format, scale prior to coding.
Faulty error handling – software doesn't detect or recover from known faults	Failure to specify what the software should do in the event of hardware failure, computational failure, software failure, interface failure
Faulty timing – timeouts are too long or too short, events happen too early or too late	Failure to develop timing diagrams in design phase
Faulty sequence – events happen in the wrong order, inadvertently or not at all when commanded	Failure to develop sequence diagrams in the design phase
Faulty functionality – software performs the wrong function or the right function incorrectly	Failure to involve end user domain experts in the requirements activity, failure to prototype prior to development, failure to develop use cases
Faulty corrective actions – software changes introduce new defects	Failure to regression test all changes during testing
Faulty memory management	Failure to apply coding standards and tools that can identify memory issues
Faulty logic	Failure to design truth tables for complex logic and test every path during unit testing

Table 1 Relationship between weaknesses in development practices and failure modes

Statement of work

Ann Marie Neufelder will examine 50 to 200 recent software problem reports which must include a description of the software problem. Ideally the reports also have comments on whether the problem is/was fixed and any findings by the software engineers analyzing the problems. The report will be similar to the illustrations shown above.

This service is conducted in accordance with recognized industry standards as shown in the below table.

Service	Corresponding industry standards
Software root cause analysis	IEEE 1633 Recommended Practices for Software Reliability clauses 5.2.1

Table 2 Relationship between services and industry standards

Related products and training

Related products	Related services	Related training
Software FMEA toolkit	Software FMEA	Software FMEA training

Pricing

Call 321-514-4659 for quotation.

MISSION READY SOFTWARE
SOFTREL LLC

sales@softrel.com
321-514-4659
http://missionreadysoftware.com