Requirements Driven Framework for Root Cause Analysis in SOA Environments

UNIVERSITY OF WATERLOO

uwaterloo.ca

Hamzeh Zawawy

John Mylopoulos

Serge Mankovskii

Overview

- Root Cause Analysis Definitions
- Root Cause Analysis Tools / Techniques
- Requirement Goal Model Based Root Cause Analysis Framework
- Future Work

Fault, Failure & Incident (ITIL)

- A fault is a design flaw or malfunction that causes a failure of one or more IT Services.
- A failure is the loss of ability to operate to specification, or to deliver the required output. A failure may cascade to cause more failures. A failure may eventually cause an incident.
- An incident is not part of the standard operation of a service and may cause an interruption to, or a reduction in, the quality of that service. An incident is externally observable and is usually recorded in an incident report.

Root Cause Analysis

- Root Cause Analysis (RCA) aims to discover a fault's first or true cause.
- Every software failure is caused by number of reasons. There is a progression of actions that lead to a failure.
- RCA investigation traces the cause and effect links from the end failure back to the root cause.

Root Cause Analysis Tools

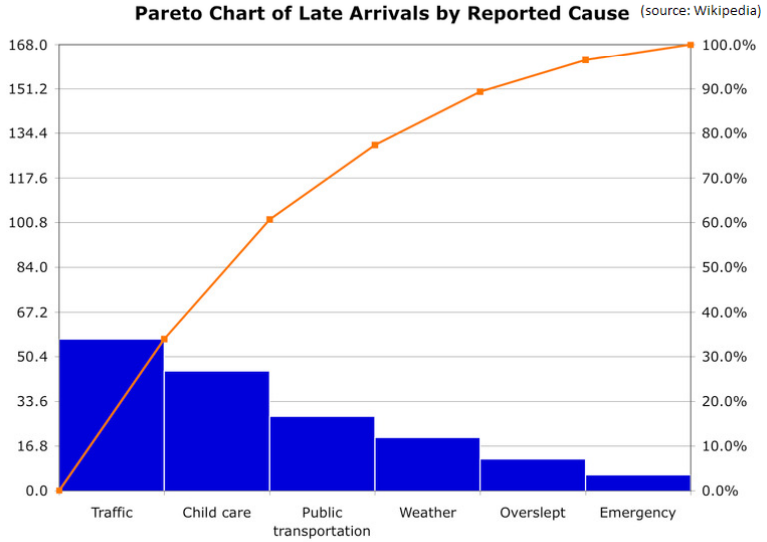
UNIVERSITY OF
WATERLOO

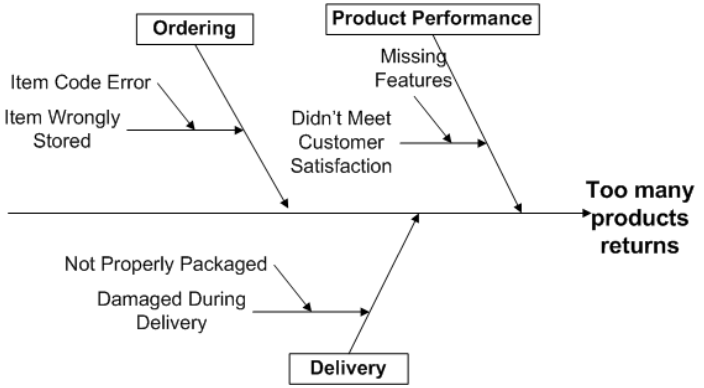
uwaterloo.ca

Root Cause Analysis Graphical Tools

Pareto diagram: a statistical technique in decision making used for the selection of a limited number of tasks that produce significant overall effect.

Fishbone (Ishikawa) diagram originally proposed by Kaoru Ishikawa in Kawasaki shipyards and used for quality management and control.

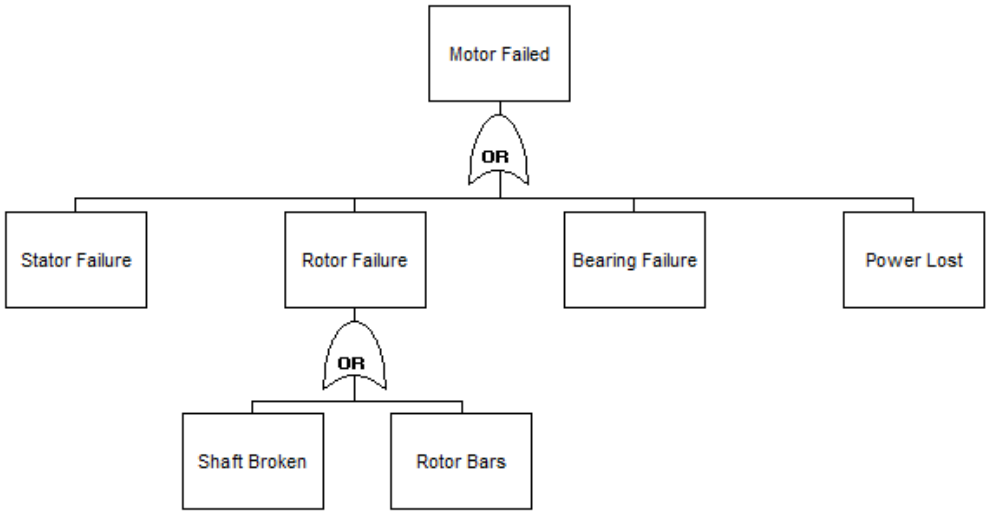




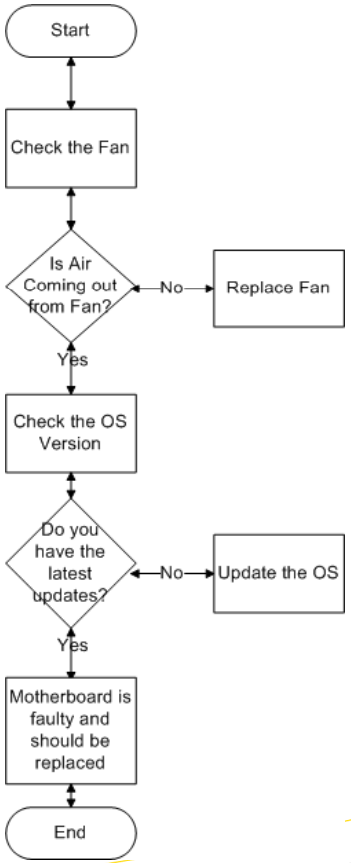
UNIVERSITY OF WATERLOO

More RCA Graphical Tools

Fault tree analysis where an undesired state of a system is analyzed using Boolean logic to combine a series of lower-level events (safety engineering).



Flowchart for RCA



UNIVERSITY OF WATERLOO

RCA Methodologies

- Rule Based
- Case Based
- Probabilistic

Rule Based RCA

- Rules represent general knowledge regarding a domain.
- A set of rules that map symptoms to root causes is built through interviews or using machine learning methods based on existing empirical data.
- To find the root cause of an incident, rules are searched for matching symptoms.
- The basic form of a rule is:
  - if <conditions> then <conclusion> where <conditions>
- When sufficient conditions of a rule are satisfied, the conclusion is derived and the rule is triggered.

Case Based RCA

- Case Based representations store a large set of previous cases with their solutions in the *case base* (case library) and uses them when a similar case has to be dealt with.
- CB cycle : (i) retrieve, (ii) reuse, (iii) revise and (iv) retain.
- in (i), the most relevant stored case(s) to the new case is retrieved based on similarity metrics
- in (ii), a solution for the new case is created based on the retrieved most relevant case(s).
- in (iii), the correctness of the proposed solution is validated with the intervention of the user.
- (iv) decides whether the knowledge learned from the solution of the new case is important enough to be incorporated into the system.

Rule Based vs. Case Based

- Rules usually represent general knowledge, whereas cases encompass knowledge accumulated from specific (specialized) situations.
- Rule Based: The downside for these approaches is that it is difficult to have a comprehensive rules list.
- Case Based: An incident is manually resolved the first time it is encountered, and then storing it to the set of available cases for future incidents. Manually resolving each case might be costly

Probabilistic RCA

- Traditional RCA techniques assume that the system components inter-dependencies are known with certainty or that the information about the monitored system state is accurate and complete, which is not always correct.
- Probabilistic approaches use fault propagation models based on causality graph that records the cause and effect relations among a system components, and model the strength of the relationships through probabilities.
- The graph consists of nodes (for the components), and edges (for the relations), where an edge from x to y shows that a problem in x could cause a problem in y.

Requirement Goal Model Based RCA

UNIVERSITY OF WATERLOO

uwaterloo.ca

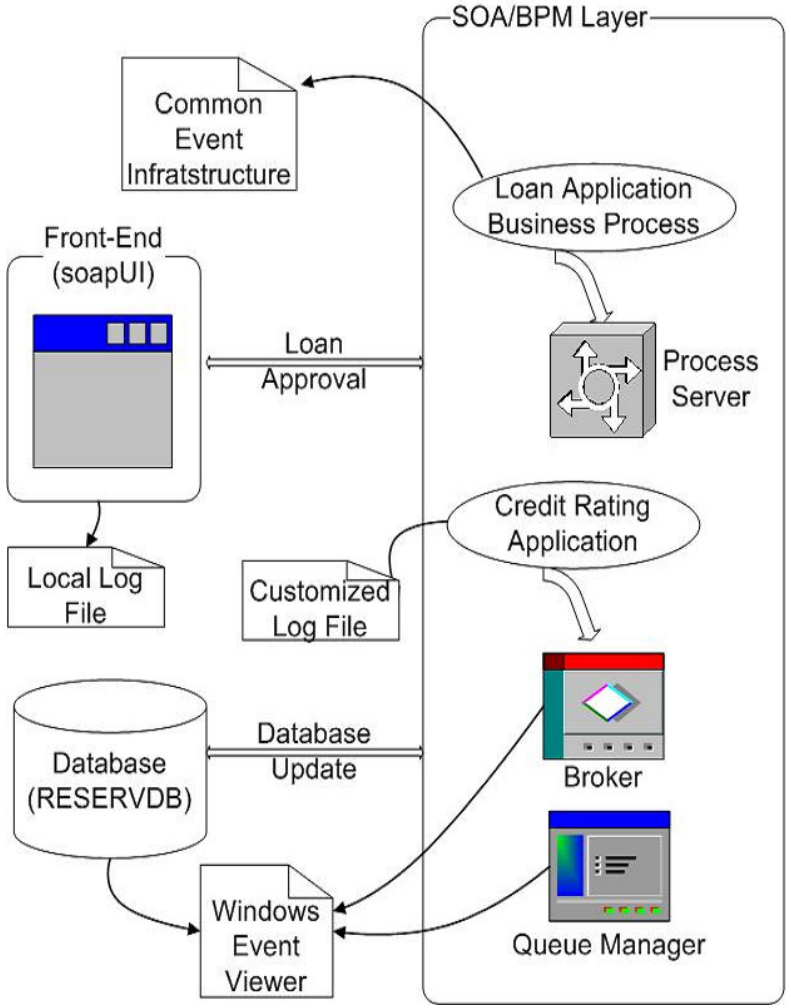
Goal Models For RCA

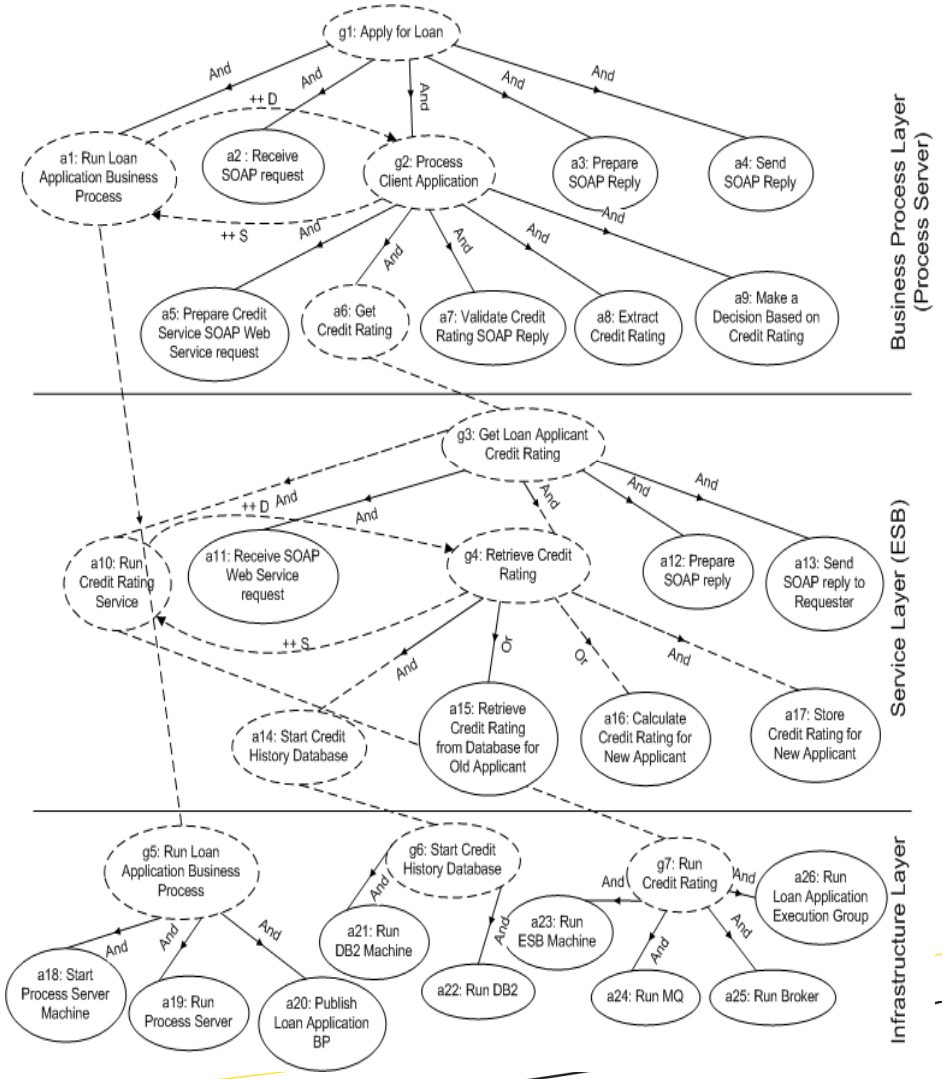
- The early phase of Requirements Engineering is concerned with the organizational context in which a software will be used. It focuses on understanding the “whys” that underlies system requirements rather than on the “what” the system should do.
- The late phase of Requirements Engineering focuses on representing the expected functionality of the software system. Goal models are bridges between "early" and "late" requirements.
- A goal model consists of one or more root goals, representing stakeholder objectives. Each goal is AND-OR decomposed into sub-goals.

Goal Models For RCA

- In the context of root cause analysis, a system failure is the lack of delivery of the intended functional or non-functional system requirement the corresponding goal models. Thus a system failure corresponds to a top goal being unsatisfied (denied).
- A top goal is considered to be denied if one or more of its sub-goals has been denied and thus the AND-OR composition these sub-goals is evaluated to false.
- A task (leaf node in a goal model) is denied if it didn't occur or if it has occurred but either its pre-conditions or post-conditions have been denied.

Goal Model in RCA





UNIVERSITY OF WATERLOO

Proposed Framework

UNIVERSITY OF
WATERLOO

uwaterloo.ca

Proposed RCA Framework

- We propose a root cause analysis framework based on annotated requirement goal model in IT systems.
- The annotated goal trees model the constraints and the conditions by which the functionality of a particular system is being delivered.
- The annotations represent the pre-condition, occurrence and post-condition and they are based on OCL:

Example of Occurrence: (CONTEXT LIKE 'Machine1; 'ApplyForLoan';%) AND DESCRIPTION LIKE '%Apply For Loan%' AND (EVENT NATURE = 'REPLY)

Query Generation / Transformation

- In this process, we extend the OCL constraints (i.e. annotations) with the user's points of interests.
- Next, a transformation process maps the merged constraints and conditions to form a collection of queries that can be applied to a relational database that stores the logged data.

T.TIMESTAMP > T1 AND T.TIMESTAMP < T2 AND (T.CONTEXT = Server1) AND (T.DESCRIPTION LIKE "John Smith")AND (T.PROCESS = P1 OR T.PROCESS = P2)

Diagnostics

- The results of such queries provide a subset of the logged data that is compliant with the goal tree and can be used by a diagnostic SAT-solver based algorithm towards identifying the root cause of the problem being observed.
- The interpreted log data generated for the goal model Apply for Loan (shown earlier):

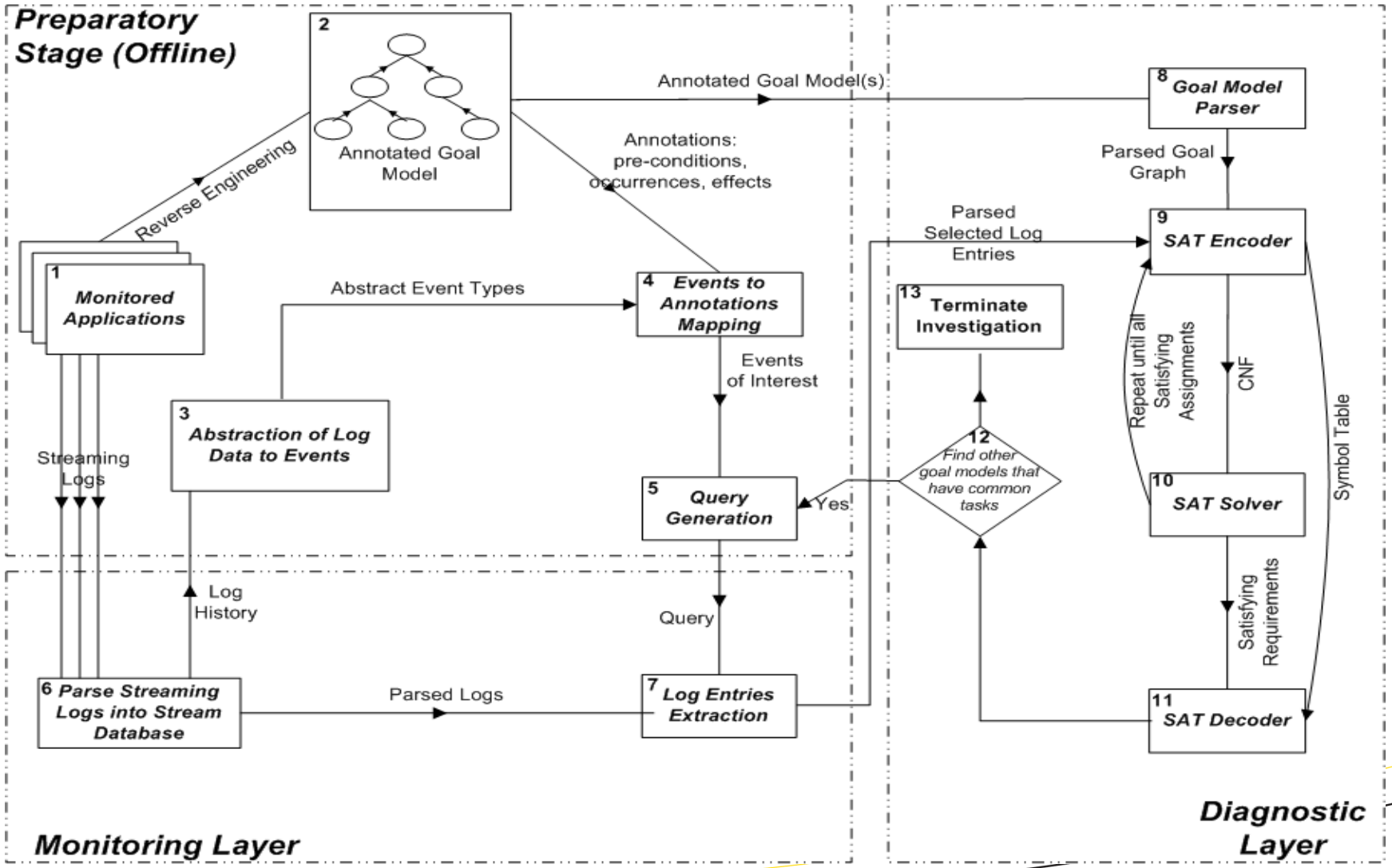
WS1 Sub; BP Idle; occ(3); WSReq Sub; occ(4); BP St; BPLoan St; ClntInf Vld; occ(6); Prpre CR Rq; occ(7); ! Rcvd CR Rpl; !Valid CR; ! CR Avail; ! Decision Done; occ(11);! Reply Gen; occ(12); ! Reply Sub;

RCA Diagnosis

- The diagnosis produced by the SAT solver shows if the top goal has been satisfied or not.
- In case the top goal is not satisfied, the diagnosis will include the different combinations for the tasks that may have failed and led to the failure of the top goal:

Diagnosis 1: [fd(7), fd(11), fd(12)]

Block Diagram



RCA Framework in SOA

- The proposed framework is geared towards service-oriented systems (not device-oriented level).
- The monitored systems are modeled using layered goal models representing business, service and infrastructure layers.
- Heterogeneous native log data incoming from different sources is and normalized into a unified format.

Future Work/Open Issues

- Lack of widely adopted logging standards in industry. Currently, there are standards for different subject areas such as IDEMF for intrusion detection or by vendors such as CBE from IBM, etc.. However, there are no widely accepted standard SOA logging schema, taxonomy, transport, API, etc..
- Large log data size. Current filtering techniques return large false positives.
- Logging components for some of the industrial software failed when we injected errors as part of our testing, this led to lost alarms and thus degradation in the quality of collected evidence.