h_da

HOCHSCHULE DARMSTADT
UNIVERSITY OF APPLIED SCIENCES

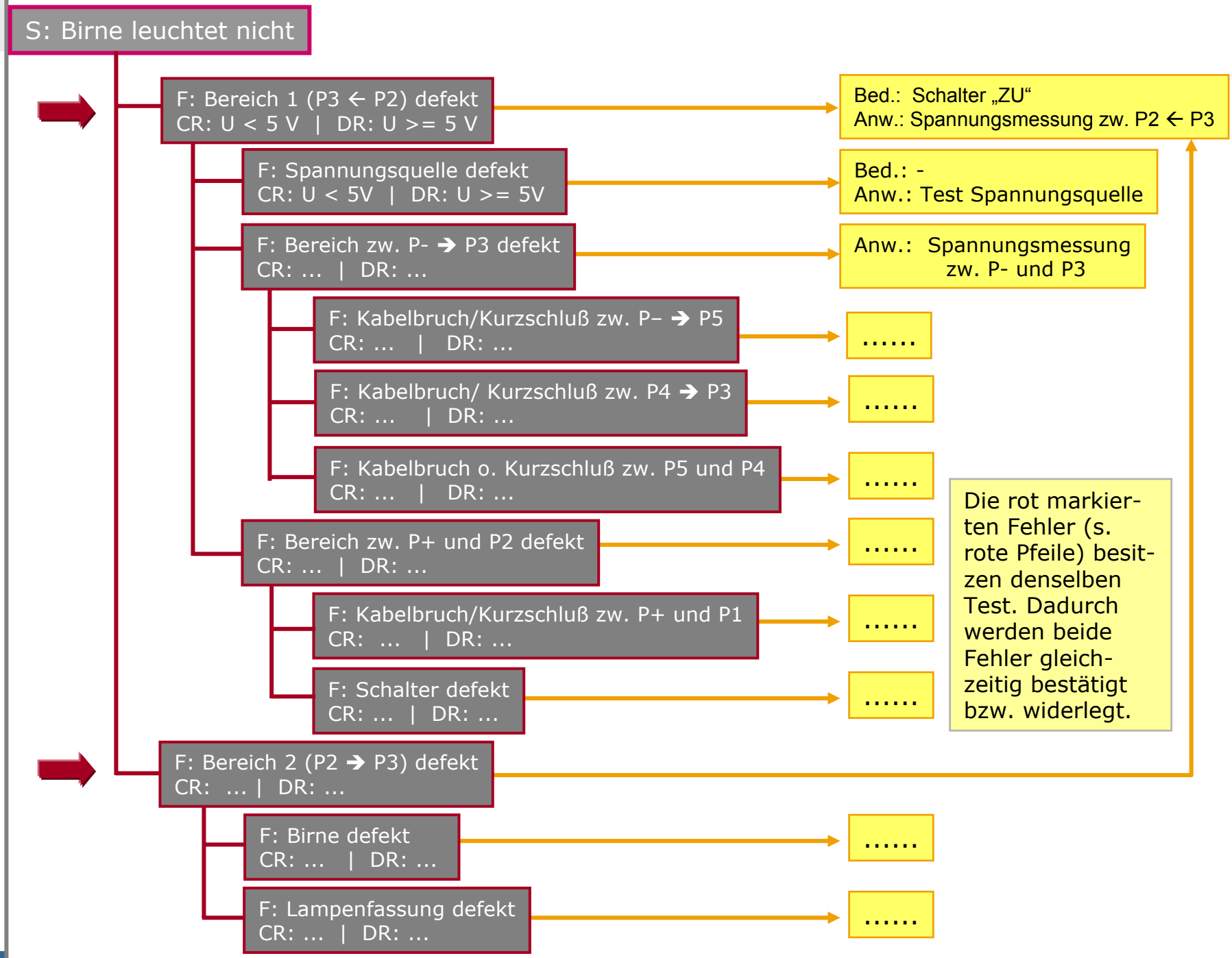
Wissensbasierte Diagnostik

Kap. 6: Fehlerbäume Teil 2

Dr. Norbert Waleschkowski
Fachbereich Informatik

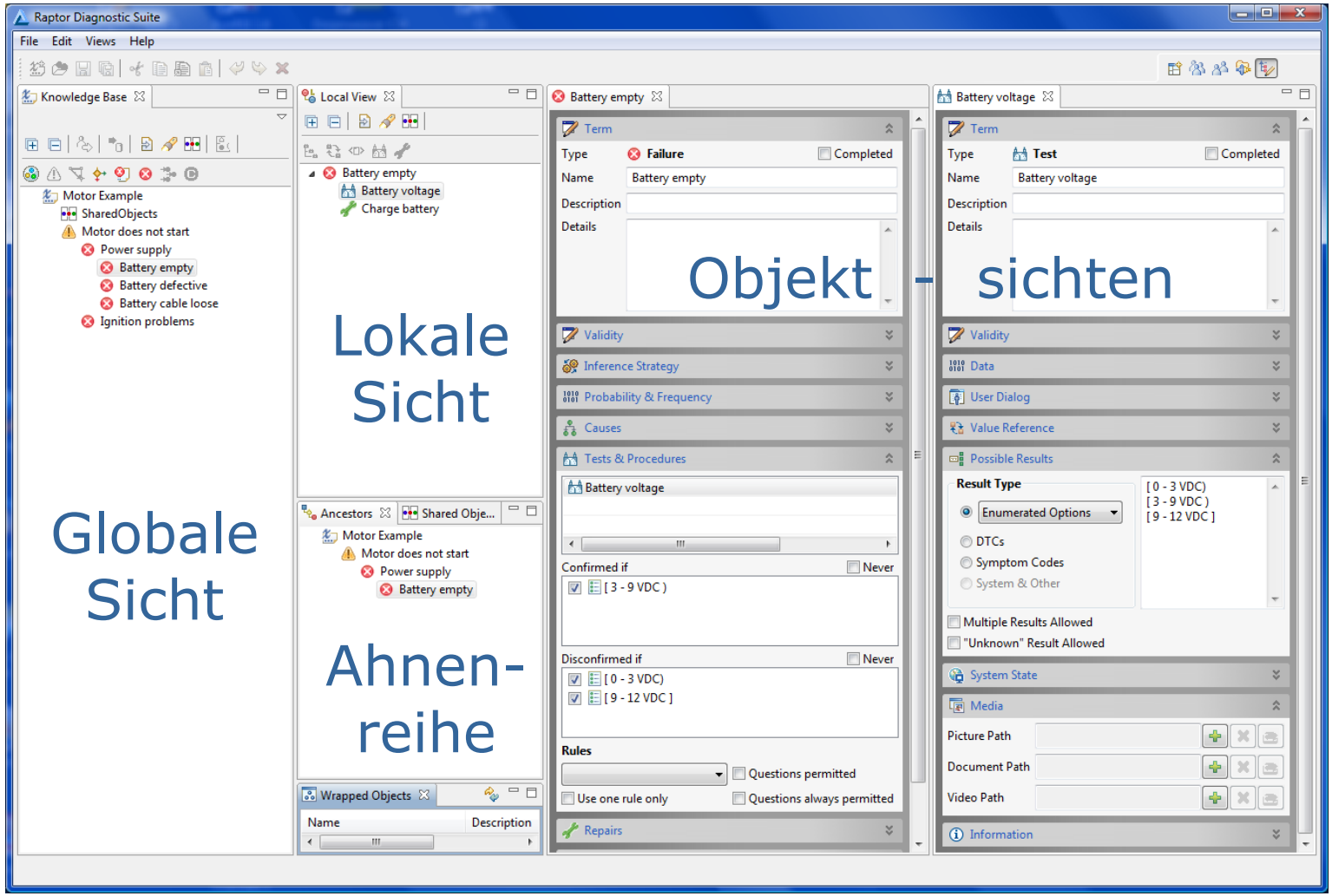
Vorlesung Master-Studiengang
Wintersemester 2009/10

M.C. Escher: Treppauf - Treppab


S: Birne leuchtet nicht
F: Bereich 1 (P3 ← P2) defekt
CR: U < 5 V | DR: U >= 5 V
Bed.: Schalter „ZU“
Anw.: Spannungsmessung zw. P2 ← P3
F: Spannungsquelle defekt
CR: U < 5V | DR: U >= 5V
Bed.: -
Anw.: Test Spannungsquelle
F: Bereich zw. P- → P3 defekt
CR: ... | DR: ...
Anw.: Spannungsmessung zw. P- und P3
F: Kabelbruch/Kurzschluß zw. P– → P5
CR: ... | DR: ...
.....
F: Kabelbruch/ Kurzschluß zw. P4 → P3
CR: ... | DR: ...
.....
F: Kabelbruch o. Kurzschluß zw. P5 und P4
CR: ... | DR: ...
.....
F: Bereich zw. P+ und P2 defekt
CR: ... | DR: ...
.....
F: Kabelbruch/Kurzschluß zw. P+ und P1
CR: ... | DR: ...
.....
F: Schalter defekt
CR: ... | DR: ...
.....
Die rot markierten Fehler (s. rote Pfeile) besitzen denselben Test. Dadurch werden beide Fehler gleichzeitig bestätigt bzw. widerlegt.
F: Bereich 2 (P2 → P3) defekt
CR: ... | DR: ...
F: Birne defekt
CR: ... | DR: ...
.....
F: Lampenfassung defekt
CR: ... | DR: ...
.....

Ein Fehlermodell-Editor

Eine grafische Diagnoseumgebung: Hier ein Fehlermodell-Editor



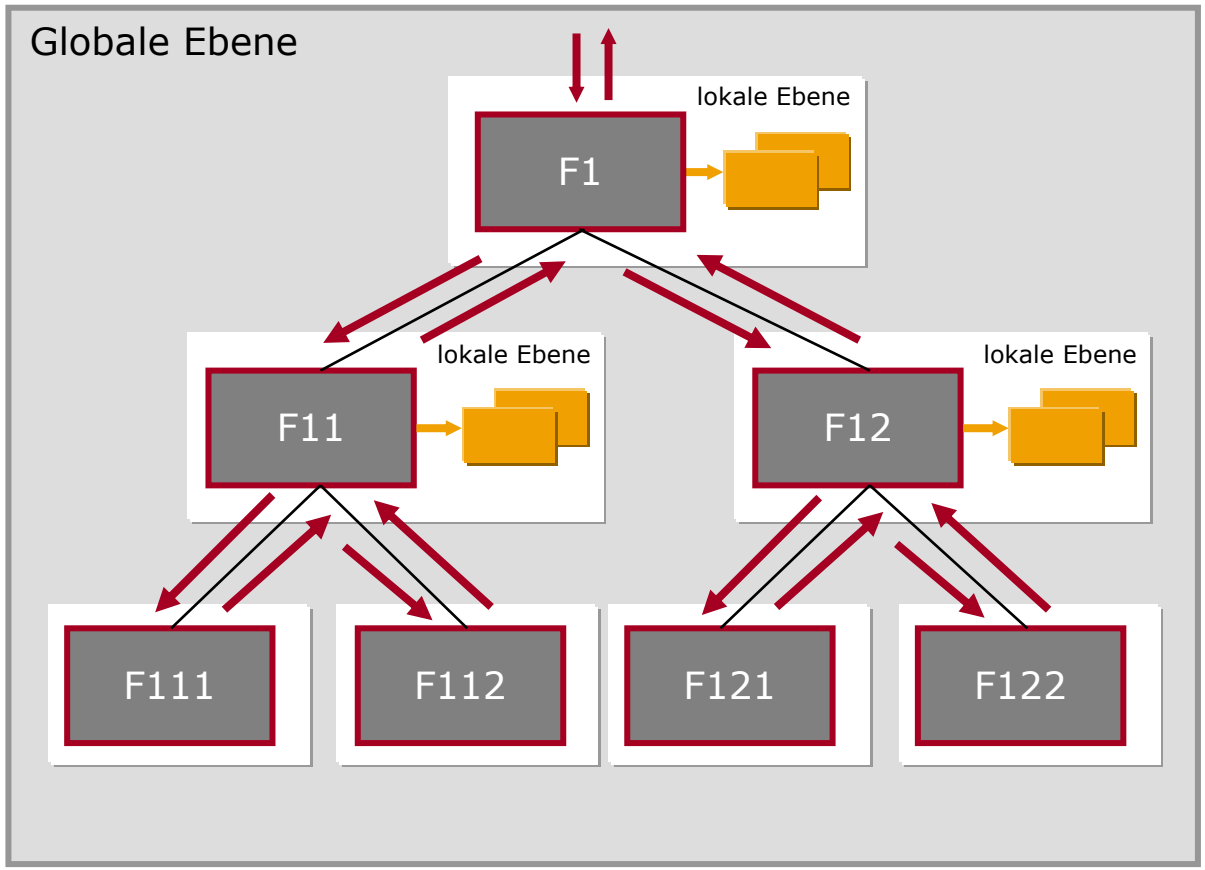
Fehlerbäume – Die Standard-Inferenzstrategie"

Eine einfache Inferenzstratetgie:

Die Inferenz navigiert in der Fehlerstruktur (vertikal) „von oben nach unten" und „von links nach rechts."

Pro besuchtem Knoten wird die lokale Umgebung (horizontal) evaluiert.

Wird der Fehler bestätigt, wird der Unterbaum inspiziert. Wird der Fehler widerlegt, wird der nächste Kindknoten des Vaterknotens inspiziert.



Default-Inferenzstrategie: von oben nach unten
von links nach rechts

Hier wird das Keyword ***nie*** verwendet. D.h. dieser Fehler kann in keinem Falle durch den Test1 "*Lichtkontrolle"* bestätigt werden.

Ein Anwendungsfall

- Es soll der Fehler „Batterie defekt" untersucht werden.
- Wie kann der Fehler effizient gesucht werden ?

Fehler-X: Batterie defekt

...
hat Test: Lichtkontrolle:
CR: *nie*
DR: stark
hat Test: Spannungsmessung:
CR: < 9 [V]
DR: >= 9 [V]
hat-Reparatur: Rep1

Test 1: Lichtkontrolle

...
Anweisung:
Schalten Sie das Licht ein.
Wie brennt das Licht?
Mögliche Ergebnisse:
nicht, schwach, stark

Test 2: Spannungsmessung

...
Anweisung:
Messen Sie die Spannung.
Mögliche Ergebnisse:
0 – 13 [V]

Rep1

Diagnosefall 1:
Inferenz DPS inspiziert Fehler-X.
Erster Test wird gefunden und evaluiert.
Annahme: Licht brennt stark.
Fehler-X wird widerlegt (disconfirmed).
Inferenz verläßt Fehler-X und setzt die Diagnose beim nächsten Fehler fort.

Diagnosefall 2:
Inferenz DPS inspiziert Fehler-X.
Der erster Test wird gefunden und evaluiert.
Annahme: Licht brennt nicht.
Fehler-X wird weder widerlegt (disconfirmed) noch bestätigt (confirmed).
Inferenz evaluiert Test2. Annahme: Test 2 liefert 6 [V].
Fehler X wird bestätigt.
Die Reparatur Rep1 wird ausgeführt.

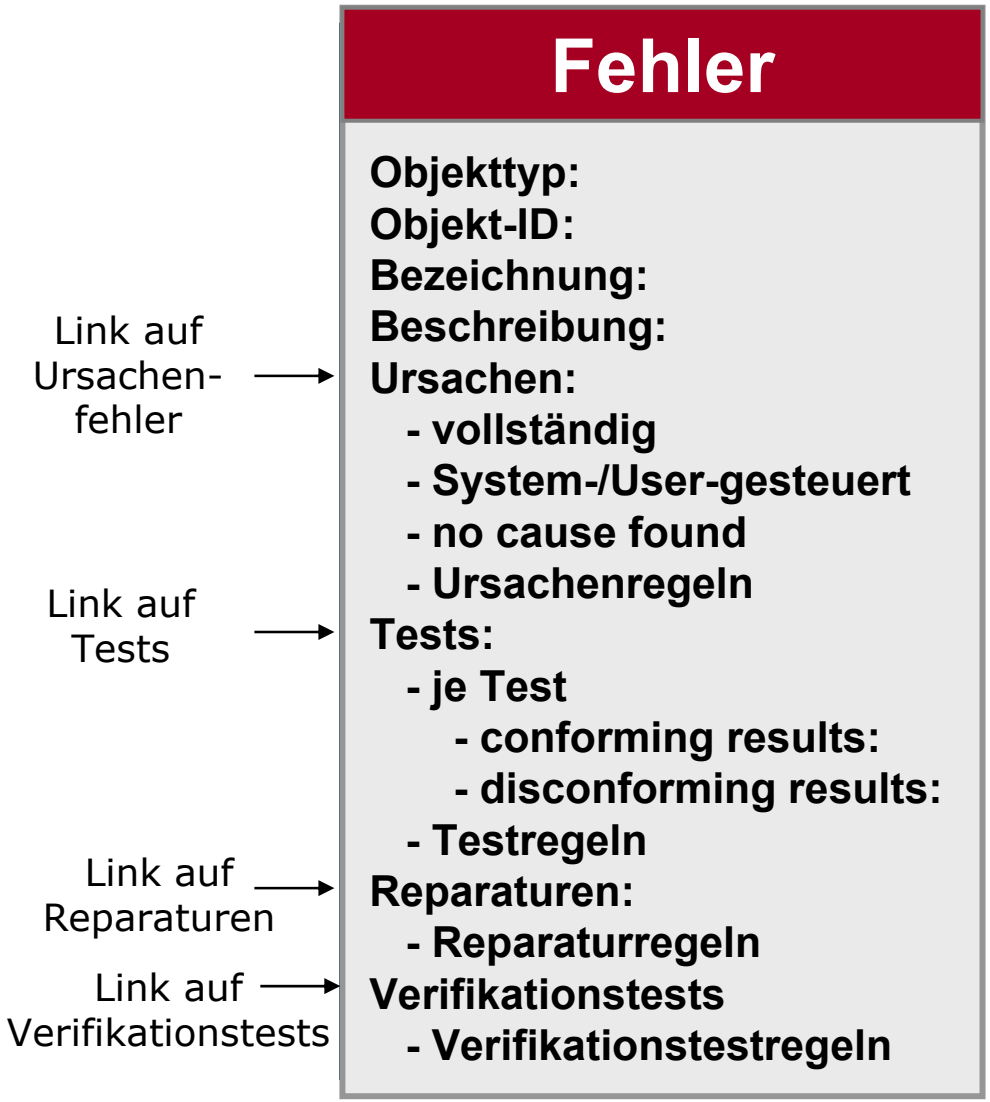
Weitere Objektklassen

- Symptome sind beobachtbare, wahrnehmbare oder meßbare Phänomene
- Testprozeduren beschreiben Testsequenzen, von denen jeder Test ausgeführt werden muss, bevor die Bedeutung der gesamten Testprozedur für den Diagnoseablauf beurteilt werden kann. Die Ergebnisse der einzelnen Test können verknüpft werden: „wenn A und B und nicht C"
- In Testfamilien werden individuelle Tests zusammengefasst, um z.B. einen speziellen Zustand des Systems unter Test auszunutzen. Z.B bietet es sich an, bestimmte Test zusätzlich durchzuführen, wenn das Getriebe ohnehin auseinandergebaut werden muss.
- Regeln bieten die Möglichkeit, die Ablaufsteuerung der Diagnose zu beeinflussen. Mittels Regeln können die Objektstrukturen – z.B. der Fehlerbaum selbst – zur Laufzeit dynamisch verändert werden. Anwendungsbeispiel: Abarbeitung der Fehler nach Wahrscheinlichkeiten („*most probable failure first*") oder nach den damit verbundenen Testkosten („*cheapest test first*").
- Weitere Objektklassen: Hypothesentests, Fehlerklassen, ...

Regeln

- Ersetzungsregeln (Replace Rules) dienen dazu, zur Laufzeit aktuelle Werte in einer WB durch andere zu ersetzen.
  - Anwendungsbeispiel: In einer WB werden verschiedene Modelltypen eines Systems behandelt. In Abhängigkeit vom jeweiligen Modell unter Test sind diese gegebenenfalls zu modifizieren.
- Umordnungsregeln (Reorder Rules) werden verwendet, um zur Laufzeit die Anordnung von Objekten – etwa von Fehlern – zu verändern.
  - Anwendungsbeispiel: Sommer-Winter-Problem, Anordnung nach Wahrscheinlichkeiten, ...
- Mit Löschregeln (Remove Rules) können zur Laufzeit Objekte aus der WB entfernt werden. Z.B. sind bei einem bestimmten Modell gewisse Komponenten nicht vorhanden. Die entsprechenden Fehler werden dann gelöscht.
  - Anwendungsbeispiel: Ein Fahrzeug verfügt nicht über ein Navigations-System. Alle entsprechenden Objekte (Fehler, Tests, Reparaturen) werden dann gelöscht.

Anatomie eines Fehlerobjekts



- Jedem Fehler können beliebig viele Ursachenfehler zugeordnet werden.
- Die Ursachen können durch Ursachenregeln modifiziert werden.
- Jedem Fehler können beliebig viele Tests zugeordnet sein.
- Die Tests können durch Testregeln modifiziert werden.
- Einem Fehler können beliebig viele Reparaturen zugeordnet sein.
- Die Reparaturen können durch Verifikationsregeln modifiziert werden.

Anatomie eines Testobjekts

Test

Objekttyp:
Objekt-ID:
Bezeichnung:
Beschreibung:
Testumfang:
Vorbedingungen:
Warnung:
Nachbedingungen:
Anweisung:
Mögliche Ergebnisse:
Art des Wertebezugs:
Wie?
Warum?
(Link auf) Bildinformation:
(Link auf) techn. Dokumente:
....
....

- Jedem Fehler können beliebig viele Ursachenfehler zugeordnet werden.
- Die Ursachen können durch Ursachenregeln modifiziert werden.
- Jedem Fehler können beliebig viele Tests zugeordnet sein.
- Die Tests können durch Testregeln modifiziert werden.
- Einem Fehler können beliebig viele Reparaturen zugeordnet sein.
- Die Reparaturen können durch Verifikationsregeln modifiziert werden.

Phasen einer Diagnosesitzung

Eine Diagnosesitzung gliedert sich typischerweise in 3 Phasen:

- Symptom-/Fehleranalyse
  - d.h., möglichst genaue Feststellung des Symptom-/Fehlerbildes)
- Diagnosephase
  - Durchführung der Diagnose
- Reparaturphase
  - Durchführung von Fehlerbehebungsmaßnahmen und Prüfung, ob die Maßnahmen die Symptome/Fehler beseitigt haben.

Die Diagnosephase (1)

Das Inferenzverfahren (Der *Diagnostic Problem Solver* DPS)

- In Fehlermodellen sind i.d.R. mehrere Ebenen von Fehlern vorhanden.
- Typische Fehlermodelle besitzen eine Tiefe von 4 – 12 Fehlerebenen. (Konzeptionell können sie beliebig tief sein.)
- Durch die lokale Zuordnung von Objekten wie Test- und Reparaturobjekte um die Fehler entsteht ein übersichtliche und leicht pflegbare Struktur. Diese bildet das Rückgrat einer Diagnoseapplikation.
- Auf solchen Fehlermodellen arbeitet der DPS (Inferenzmaschine).
- DPS führt primär eine Tiefensuche durch, ausgehend vom beobachteten Symptom.
- DSP sucht nach dem Grund des Symptoms durch sequentielle Betrachtung der möglichen Ursachenobjekte von links nach rechts, wie sie durch die „*caused-by*"-Kanten spezifiziert sind.

Die Diagnosephase (2)

- Mögliche Fehler können 3 Zustände einnehmen:
  - bestätigt (confirmed)
  - ausgeschlossen (disconfirmed)
  - unbekannt (unknown)
- Wenn ein möglicher Fehler durch einen Test ausgeschlossen wurde, fokussiert DPS auf den nächstmöglichen rechten Fehler (der noch nicht *disconfirmed* ist).
- Wenn ein möglicher Fehler bestätigt worden ist, legt DPS den Fokus auf dieses Objekt und versucht, die Ursachen für diesen Fehler durch die Untersuchung der Kinder dieses Fehlers festzustellen.
- Wenn der Status eines Fehlers *unbekannt* ist, behandelt DPS diese Unsicherheit durch Fokussierung auf die Kinder dieses Objekts. Wird ein Unterfehler bestätigt, dann wird auch der Vater bestätigt.
- Wird während der Diagnose ein Fehler bestätigt, der nicht mit der aktuellen Diagnose verbunden ist, merkt sich DPS diese Information. Am Ende der Diagnose- und Reparaturphase werden diese zufällig erkannten Fehler ebenfalls diagnostiziert, sofern sie noch bestätigt sind.

Die Diagnosephase (3)

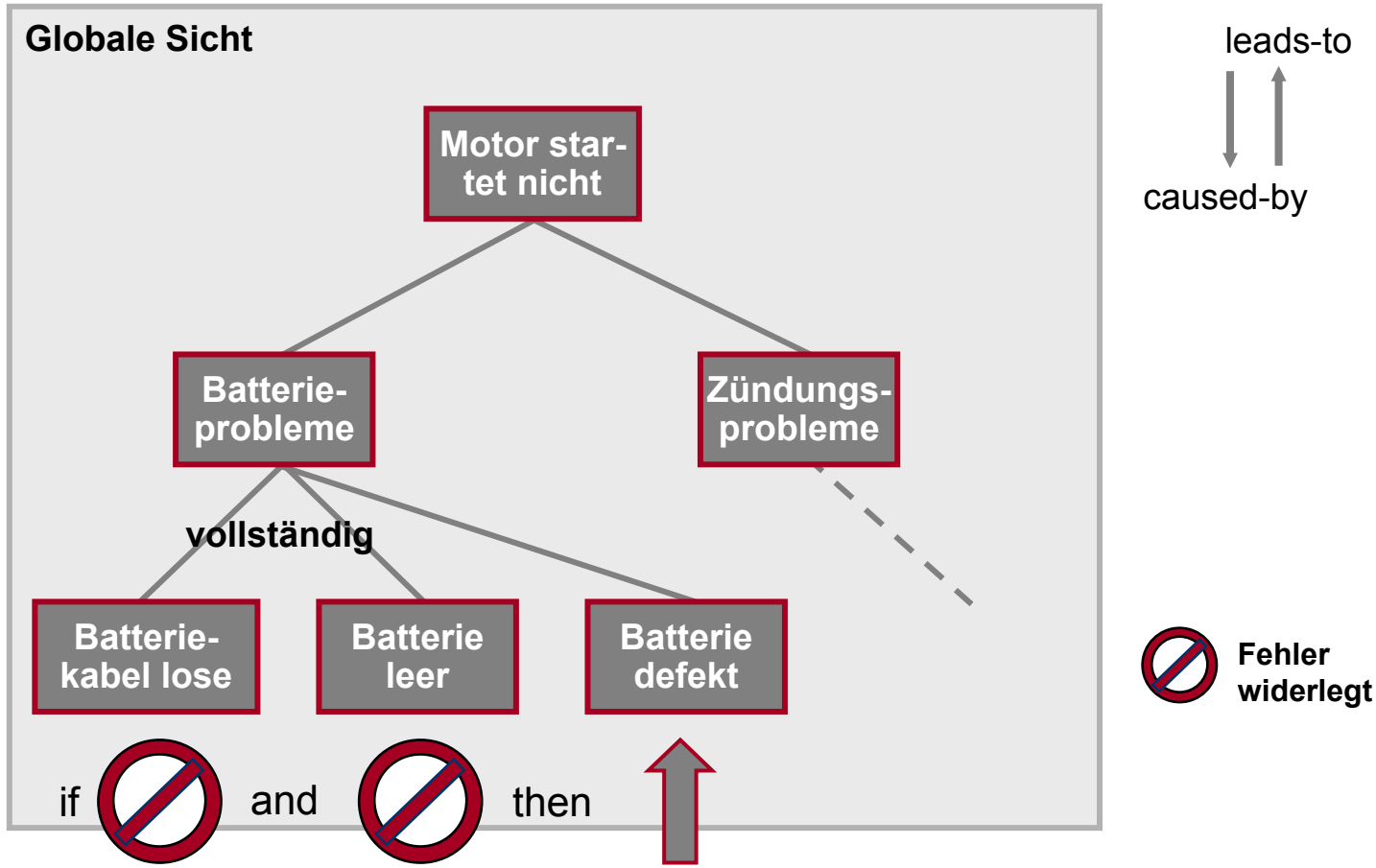
- DPS verfügt über verschiedene Strategien, um den Status eines Fehlers zu bestimmen.
- Standardstrategie:
  - Normalerweise wird hierzu ein direkt mit dem Fehler assoziierter Test verwendet.
- Rule-out-Strategie:
  - Wenn der Status eines Fehlers unbekannt ist und die Kinder den Bereich der möglichen Ursachen vollständig abdecken und alle ausgeschlossen sind, dann wird auch der gegenwärtige Fehler ausgeschlossen.
- Confirmation-by-Elimination:
  - Der gegenwärtige Fehler ist *confirmed* und die Kinder decken die möglichen Ursachen vollständig ab. Sind nun alle Kinder bis auf einen *disconfirmed*, dann wird dieses eine automatisch ohne weitere Untersuchung *confirmed*.
- Es gelten folgende Prioritäten:
  - direkter Test < Rule-out < Confirmation-by-Elimination
    (< bedeutet dabei *„hat Priorität vor"*)

Inferenz: „Confirmation by Elimination“

Wenn die Ursachen eines Fehlers vollständig (erschöpfend, engl. exhaustive) aufgezählt sind, kann der letzte Fehler automatisch bestätigt (confirmed) werden, ohne dass ein Test ausgeführt werden muss, sofern die anderen Ursachen widerlegt (disconfirmed) wurden.

Im Baum wird dies durch **„vollständig“** markiert.

Vorsicht
bei der Anwendung dieses Features ! Man muss sich ganz sicher sein, dass die Ursachenangabe auch wirklich vollständig ist !



- Die Fehler *„Batteriekabel lose“* und *„Batterie leer“* wurden widerlegt (*disconfirmed*). Der Fehler *„Batterie defekt“* kann nun automatisch bestätigt (*confirmed*) werden, wenn die Ursachen vollständig angegeben sind.

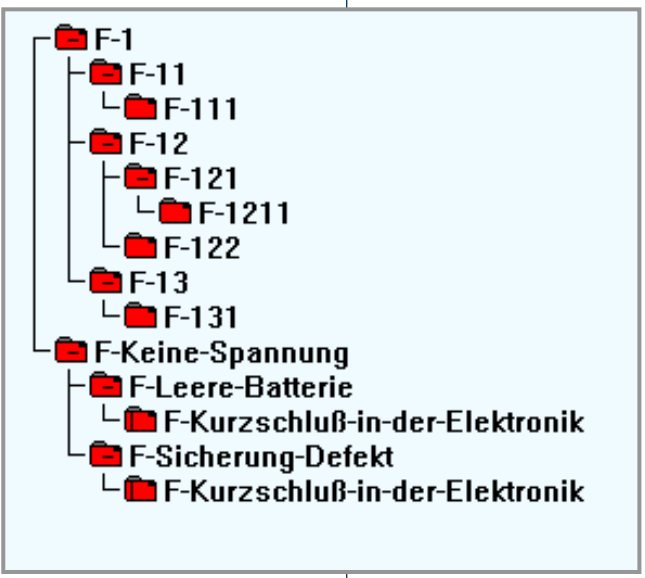
Die Diagnosephase (4)

- Standardmäßig werden die Fehler von links nach rechts abgearbeitet.
- Es gibt aber zahlreiche Alternativen:
- Most probable failure first:
  - Die Fehler werden nach der Wahrscheinlichkeit ihres Auftretens inspiziert. In diesem Falle müssen die Fehler ein Attribut *Wahrscheinlichkeit* bzw. *Häufigkeit* besitzen, um diese Inferenzstrategie zu unterstützen.
- Cheapest test first:
  - Zunächst werden die Fehler inspiziert, die die einfachsten und kostengünstigsten Tests besitzen. Dies impliziert, dass die Tests ein Attribut „Arbeitswert" bzw. „Kosten" besitzen.
- Andere Inferenzstrategien:
  - Minimum average costs first
  - .... sonstige
- Die Reihenfolge kann auch das Ergebnis einer komplexen Strategie sein, bei der Kosten, Wahrscheinlichkeiten, das Verfügbarkeit sinnvoller Reparaturen etc. gleichzeitig einfließen.

Die Diagnosephase (5)

- Weiterhin kann der Suchraum durch Regeln in Abhängigkeit von der während der Diagnose gewonnenen Informationen dynamisch, also zur Laufzeit geändert werden.
- Des weiteren können verschiedene Modi der Benutzerführung bereitgestellt werden:
  - Standardmodus (Geführte Diagnose bzw. „system driven diagnosis"): Der Benutzer wird durch die komplette Diagnose- und Reparaturphase geführt.
  - Expertenmodus („Führung durch den Benutzer" bzw. „user driven strategy"): Der Benutzer kann die Reihenfolge, in welcher Fehlerkandidaten untersucht werden, selbst festlegen.

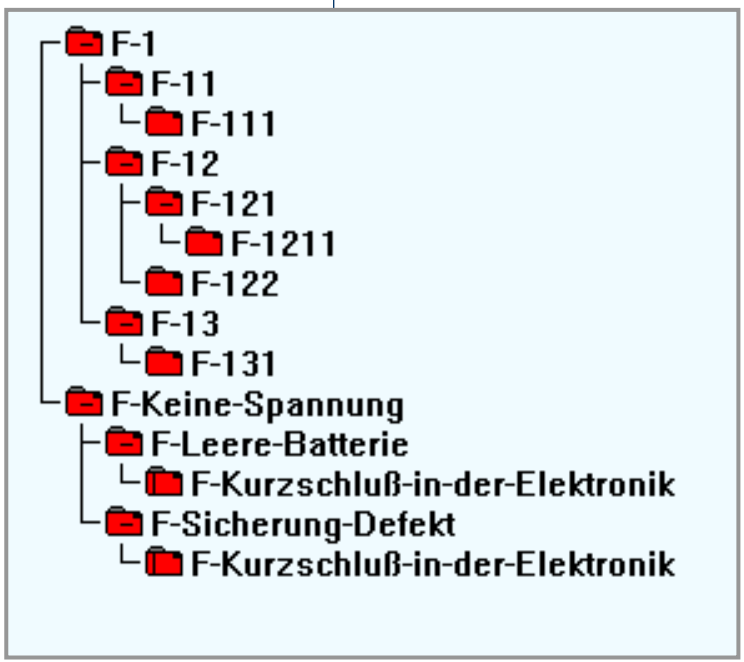
Die Fehlerhierarchiediagnose im Detail (1)



- DPS verwendet die Fehlerhierarchiediagnose zur Bearbeitung einer Wissensbasis. Fehlerhierarchiediagnose besagt, dass eine Fehlerhierarchie den Ablauf der Diagnose bestimmt.
- Fehlerhierarchie besagt, dass die Fehler hierarchisch angeordnet sind. Ein Fehler *F-1* ist einem anderen Fehler *F-11* untergeordnet, wenn er davon kausal abhängig ist, d.h. Fehler *F-1* durch *F-11* verursacht wird.
- Bsp.: Fehler *F-Leere-Batterie* wird verursacht durch Fehler *F-Kurzschluß-in-der-Elektronik*.
- Im Diagnosemodell des Editors sind die Fehlerhierarchien von links nach rechts (*F-links* verursacht durch *F-rechts*) angeordnet. Die Begriffe *Fehlerhierarchie* und *Fehlerbaum* werden gleichwertig benutzt

Die Fehlerhierarchiediagnose im Detail (2)

- Ein Fehler wird im Diagnosemodell durch ein Fehlerobjekt repräsentiert. Ein Fehler kann entweder ein Fehlverhalten oder ein Bauteildefekt sein. Der Fehler *F-Keine-Spannung* ist ein Fehlverhalten, der Fehler *F-Leere-Batterie* dagegen ist ein Defekt. Für Defekte sind in der Regel Reparaturen notwendig.
- Im folgenden ist das Standardverhalten des Diagnoseablaufs (Beschränkung auf Fehler, Prüftests und Reparaturen) dargestellt:



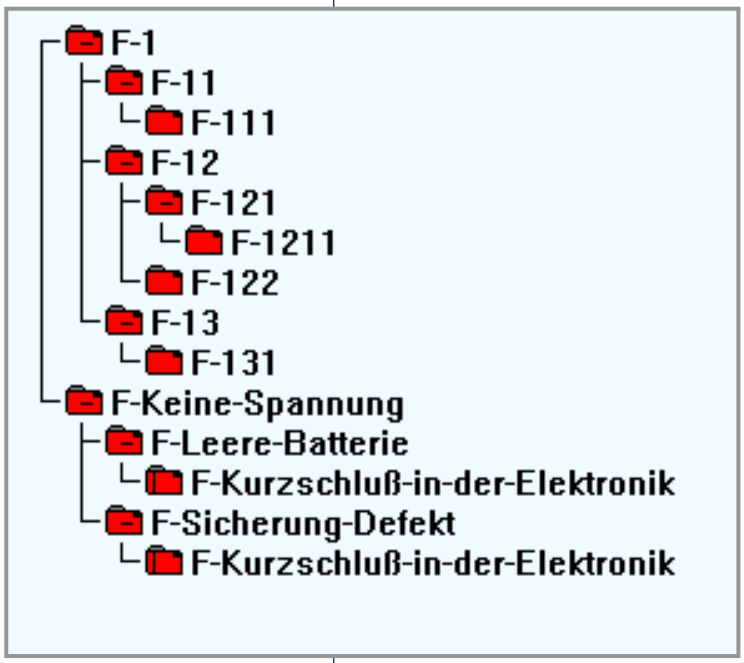
Ein Fehlerbaum wird entsprechend seiner Darstellung im Diagnosemodell von oben nach unten abgearbeitet (fokussiert). Einen Fehler des Fehlerbaums fokussieren bedeutet, den Status eines Fehlers zu bestimmen:

- ein Fehler liegt nicht vor (Widerlegung),
- ein Fehler liegt vor (Bestätigung),
- das Vorliegen eines Fehlers ist unbestimmt (weder widerlegt noch bestätigt).

Um den Status eines Fehlers zu bestimmen, werden Prüftests des Fehlers erfragt .

Der Fehlerbaum wird Zeile für Zeile von oben nach unten abgearbeitet, solange kein Fehler widerlegt oder bestätigt wird.
Bsp.: F-1, F-11, F-111, F-12, F-121 usw.

Die Fehlerhierarchiediagnose im Detail (3)



- Wird ein Fehler widerlegt, dann wird mit dem Fehler fortgefahren (von oben nach unten), der den gleichen oder einen höheren Rang in der Hierarchie aufweist. Dies bedeutet, dass durch den Ausschluss eines Fehlers alle untergeordneten Fehler bedeutungslos sind.

  Bsp.: F-12 wird widerlegt, F-13 erhält den neuen Fokus.
  Bsp.: F-13 wird widerlegt, F-Keine-Spannung erhält den neuen Fokus.

- Sobald in einem Fehlerbaum ein Fehler bestätigt wird, begrenzt sich die weitere Fokussierung auf kausal abhängige Fehler (d.h. den Unterbaum) des zuletzt bestätigten Fehlers. Wird ein Fehler ohne Unterfehler bestätigt, erfolgt die Reparatur des Fehlers. Ansonsten endet die Bearbeitung des Fehlerbaums, sobald der Unterbaum des zuletzt bestätigten Fehlers abgearbeitet wurde. Eine Meldung wird ausgegeben, dass keine tiefere Ursache für den zuletzt bestätigten Fehler gefunden wurde. Danach erfolgt die Rückkehr zur Diagnosephase *Problemauswahl*.

- Bsp.: F-1, F-11, F-111, F-12 (bestätigt), F-121 (bestät.), F-1211 (widerlegt) → "Keine Tiefere Ursache für F-121 gefunden"
  Bsp.: F-1, F-11, F-111, F-12 (bestätigt), F-121 (bestät.), F-1211 (bestätigt) → Reparatur für F-1211

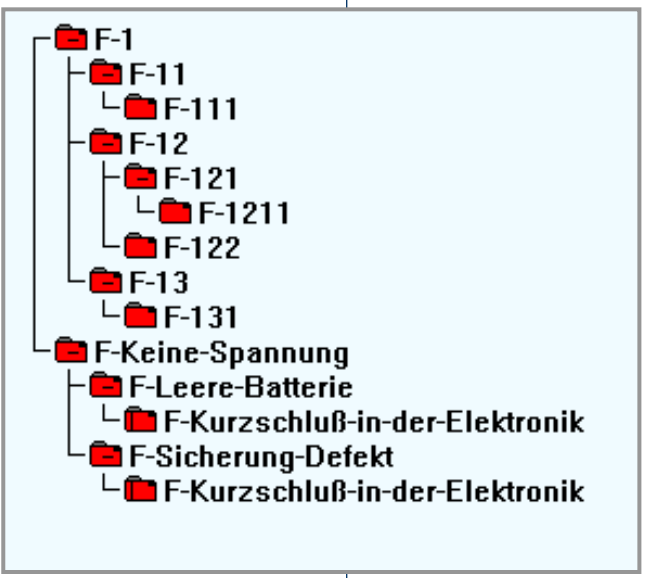
Die Fehlerhierarchiediagnose im Detail (4)

Die Vorwärtsverkettung von Ergebnissen

- Der Status eines Fehlers wird bestimmt, indem Prüftests erfragt werden und deren Ergebnisse mit den Fehlerattributen *Bestätigung*/ *Widerlegung* verglichen werden. Somit wird der Status eines Fehlers ermittelt, indem die Ergebnisse von Prüftests auf Fehler angewandt werden. Wird nun ein Test T-X erfragt, kann T-X direkt auf alle Fehler angewandt werden, für die T-X als Prüftest modelliert wurde.
- Die unmittelbare Anwendung von Ergebnissen auf andere Objekte nennt man *Vorwärtsverkettung von Ergebnissen*. Im Gegensatz hierzu bezeichnet man die Bestimmung eines Objektattributwerts durch Betrachtung bereits vorliegender Ergebnisse als *Rückwärtsverkettung von Ergebnissen*.
- Testergebnisse werden von der Inferenz vorwärtsverkettend propagiert. Ein fokussierter Fehler dient als Anlass für die Erfragung von Testergebnissen. Bei der Anwendung des Testergebnisses auf die Fehler wiederum ist der fokussierte Fehler (allgemein formuliert der Anlass der Ergebniserfragung) nicht von Bedeutung.
- Durch die Vorwärtsverkettung wird der Diagnoseablauf gegenüber der vorher diskutierten Fehlerbaumdiagnosestrategie zur Abarbeitung eines Fehlerbaums optimiert.

Die Fehlerhierarchiediagnose im Detail (5)

Die Vorwärtsverkettung von Ergebnissen (2)



Ist die Ursache eines fokussierten Fehlers bereits durch Vorwärtsverkettung bekannt (d.h. ist eine der Ursachen bereits bestätigt), so erhält die bekannte Ursache den neuen Fokus.

Bsp.: Ein fokussierter und nicht widerlegter Fehler *F-1* hat die Ursachen *F-11* und *F-12*. Ist *F-12* bereits durch Vorwärtsverkettung bestätigt, so wird nach *F-1* nicht der Fehler *F-11*, sondern *F-12* fokussiert.

Ist der Status eines Fehlers bereits durch Vorwärts-verkettung bestimmt, werden für diesen Fehler keine weiteren Prüftests erfragt.

Bsp.: Fehler *F-1* hat den Prüftest *T-1* und Fehler *F-2* die Prüftests *T-2* und *T-1*. Bei der Fokussierung von *F-1* wird der Prüftest *T-1* durchgeführt und anschließend *T-1* auf *F-1* und *F-2* angewandt. Soll in der Folge Fehler *F-2* fokussiert werden und konnte der Status von *F-2* bereits durch *T-1* bestimmt werden, wird *T-2* nicht durchgeführt.

Die Reparaturphase (1)

- Zusätzlich zur Bestimmung des/der Fehler, die das aufgetretene Symptom-Fehlerbild hervorgerufen haben, werden Strategien zur Fehlerbehebung benötigt. Hierzu können verschiedene Strategien bereitgestellt werden.
- Die Reparaturphase muss ihrerseits in zwei Teilphasen zerlegt werden:
  - Die Durchführung der Reparaturen
  - Die Verifikationsstrategie, um festzustellen, ob die Maßnahmen die Fehler behoben haben.
- Hier können einer Reparatur sog. Verifikationstests zugeordnet werden. Diese werden also nicht verwendet, um festzustellen ob ein Fehler vorliegt, sondern um die Reparatur(en) zu verifizieren.

Die Reparaturphase (2)

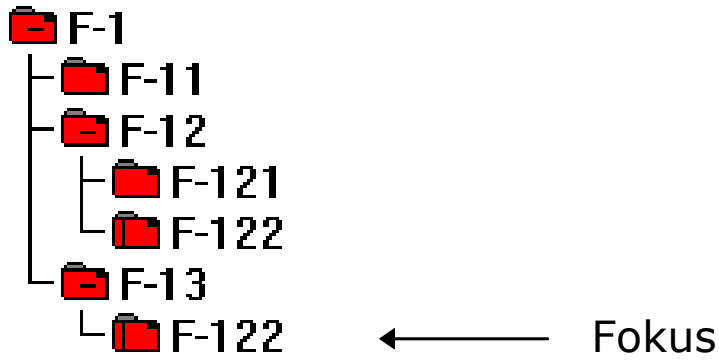
- Wird während der Diagnose ein Blattfehler (d.h. ein Fehler ohne tieferliegende Ursache) bestätigt, beginnt die Reparatur.
- Jeder Blattfehler muss ein Reparaturobjekt besitzen, da schließlich der am tiefsten liegende Fehler behoben werden soll.
- In manchen Situationen ist es jedoch einfacher, einen Test durch Austauschen eines verdächtigen Bauteils durchzuführen. In diesem Fall ist der Test bereits die Reparatur. Im Fehlerobjekt ist dann das Attribut *Test-Ist-Reparatur* zu setzen, so daß hier kein zusätzliches Reparaturobjekt benötigt wird.

Reparaturverifikation entlang des kausalen Pfades (1)

- Ist die Reparatur durchgeführt, so kann diese bei entsprechender Modellierung verifiziert werden. Die Verifikation erfolgt entlang des kausalen Pfads.
- Unter dem kausalen Pfad versteht man den
  - von der Inferenz eingeschlagenen
  - kürzesten Weg
  - von der Wurzel des Fehlerbaums zum fokussierten Fehlerobjekt.
- Der von der Inferenz eingeschlagene Weg ist nicht unbedingt der einzige. Tatsächlich sind die Fehlerhierarchien keine Bäume, sondern sogenannte gerichtete, zyklenfreie, zusammenhängende Graphen. (Bei einem Baum besitzt jedes Objekt außer dem Wurzelobjekt genau ein Vaterobjekt.)
- Als Regel gilt:
  - Bei (echten) Fehlerbäumen wäre jedes Objekt nur einmal vorhanden.
  - Bei Fehlerbäumen, die in Wirklichkeit gerichtete, zyklenfreie, zusammenhängende Graphen sind, darf jedes Objekt mehrfach im Fehlerbaum vorhanden sein.

Reparaturverifikation entlang des kausalen Pfades (2)

- Kürzester Weg bedeutet, dass jede Hierarchieebene des Fehlerbaums zwischen Wurzel und Fokus höchstens einmal vertreten ist.



- Die möglichen kürzesten Wege von Wurzel F-1 zum Fokus F-122 sind
  - F-1, F-12, F-122 sowie
  - F-1, F-13, F-122 !
- Ein möglicher nicht kürzester Weg wäre F-1, F-12, F-121, F-122.
- Der kausale Pfad ist F-1, F-13, F-122.

Reparaturverifikation entlang des kausalen Pfades (3)

- Die Verifikation erfolgt in folgenden Phasen:
- Reparaturverifikation:
  - Überprüft, ob eine Reparatur korrekt durchgeführt wurde.
- Lokale Verifikation:
  - Überprüft, ob der Blattfehler jetzt nicht mehr vorliegt.
- Fehlerverifikation:
  - Jeder Fehler entlang des kausalen Pfades wird überprüft, ob er tatsächlich behoben ist. Voraussetzung ist, dass der Fehler tatsächlich vorlag, also bestätigt war. Nicht bestätigte Fehler werden nicht überprüft. Die Fehlerverifikation erfolgt in Richtung Blatt zur Wurzel des Fehlerbaums.

Reparaturverifikation

- Damit die Reparaturverifikation durchgeführt wird, müssen für ein Reparaturobjekt Verifikationstests modelliert werden, welche die Korrektheit der durchgeführten Reparatur überprüfen. Sind keine Verifikationstests angegeben, erfolgt keine Reparatur-verifikation.
- Mit den Attributen *Erfolg* und *Misserfolg* wird festgelegt, durch welche Ergebnisse eines Verifikationstests die Reparatur als korrekt bzw. nicht korrekt durchgeführt gilt.