From Safety Analysis to Formal System Specification and Verification with OTS/CafeOBJ

J. Xiang†, K. Ogata†‡, W. Kong†, K. Futatsugi†

† Japan Advanced Institute of Science and Technology
‡ NEC Software Hokuriku, Ltd. Japan
Agenda

- Motivations
- Background of Fault Tree Analysis
- Formal Fault Tree Analysis of Observational Transition System (FTA/OTS)
- Formal Specification from Fault Trees with OTS/CafeOBJ
- Summary & Future Works
Motivations

- Safety analysis techniques ➔ generally informal
  - fault tree analysis (FTA),
  - failure mode and effects analysis (FEMA),
  - hazard and operability analysis (HZAOP), …

- Requirements engineering: the results of safety analysis must be formalized
  - mismatch because of different system aspects
  - transformation effort is non-trivial

- Common Framework
  - Formal FTA
  - Formal specification and verification with OTS/CafeOBJ
Background of Fault Tree Analysis

- First developed in the 1960s (the Minuteman missile system)
- Widely used for reliability and safety analysis for years
  - Aerospace
  - Electronics
  - Nuclear industries
- Top-down and graphic method
  - Use logic gates to decompose the top hazard event
  - **Minimal cut sets**: the *smallest* combinations of basic fault events which cause the top event.

An Example of Fault Tree

**Minimal cut sets**
- \{B1\}
- \{B2, B3\}

**Input Valve**
**Sensor**: determines when the tub is full
**Timer**: prevent from filling indefinitely; avoid a flood in case a leak in the tub

**Tub**

**Washing machine overflows**

- **B1**: Input Valve stuck open (broken)
- **B2**: Timeout Control failed
- **B3**: Full Sensor failed

- **Fill Mode too long**
## Fault Tree Symbols

<table>
<thead>
<tr>
<th>SYMBOL</th>
<th>EVENT</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image" alt="Circle" /></td>
<td>Basic Event</td>
</tr>
<tr>
<td><img src="image" alt="Rectangle" /></td>
<td>Intermediate Event</td>
</tr>
<tr>
<td><img src="image" alt="Ellipse" /></td>
<td>Conditioning Event (<em>normal</em> event)</td>
</tr>
<tr>
<td><img src="image" alt="AND-gate" /></td>
<td>AND-gate</td>
</tr>
<tr>
<td><img src="image" alt="OR-gate" /></td>
<td>OR-gate</td>
</tr>
<tr>
<td><img src="image" alt="INHIBIT-gate" /></td>
<td>INHIBIT-gate</td>
</tr>
</tbody>
</table>
Problems of Traditional Fault Tree Analysis

- Informal method
  - A stand-alone verification is needed wrt correctness and completeness

- Difficult to deal with composite fault events
  - A combination of multiple normal events rather than individual component failures

- Current formal FTAs
  - Focused on providing more precise semantics (TL) for fault tree constructors
  - Neglect the decomposition problem of composite fault events
Example: Radio-based Crossing Control System

- ‘secure’: train → level crossing
- ‘release’: train ← level crossing
- ‘passed’: train → level crossing
Decomposition Problem of Composite Fault Event

\[ \text{Hazard} \overset{\text{def}}{=} \text{OnCrossing}(tr) \land \neg \text{Closed}(ba) \]

- In case the top event consists of a combination of multiple *normal* events, we cannot decompose it directly by its logic structure.
  - Each node in fault tree must be a *fault* event.
  - If we decompose and define \( \text{OnCrossing}(tr) \) and \( \neg \text{Closed}(ba) \) as sub fault events, then to ensure \( \neg \text{Hazard} \), we may get an *unpractical* solution such as:
    \[ \neg \text{OnCrossing}(tr) \lor \text{Closed}(ba) \]
Aim of Our Work

- A formal fault tree analysis based on OTS
  - Solve the decomposition problem
  - Guarantee the correctness by the construction rather than standalone verification process

- A common framework for FTA and requirements formulation and verification with OTS/CafeOBJ
  - A common semantic model of OTS
FTA/OTS

- **Observational Transition System (OTS)**
  - An universal *system state* space $\mathcal{Y} = \{s_0, s_1, \ldots\}$
  - The value of $s$: a set of observations at a *specific* time point (*composite state*)
  - $\mathcal{O} = \{o_i, o_j, o_k, \ldots\}$, a set of observers
  - $\mathcal{T} = \{\tau_1, \tau_2, \tau_3, \ldots\}$, a set of conditional transition rules

- **FTA**
  - Event $\rightarrow$ One or several observations based on a *specific* system state
    - Single fault event, e.g., $o_i(s_1) = p$
    - *Composite* fault event, e.g., $o_i(s_1) = p \land o_j(s_1) = q$
  - Transition rules $\rightarrow$ Event decomposition

![Diagram of OTS](attachment:image.png)
Definition of Events

- **Event**: An occurrence of a specific object state, whose occurrence can be observed by an observation (predicate) $o_i$ over a set of typed parameters in a form of

$$o_i(X_{i_1}, \ldots, X_{i_m}) = c$$

- Event with time duration can also be observed

- **Classification of Event**
  - *Fault event*: an undesired event or fault
  - *Normal event*: an event that is normally expected to occur (CONDITIONING-event)
  - The decision is usually made by domain experts

- **Composite Fault Event**: A conjunction of several normal events, e.g., $o_i = p \land o_j = q$
  - Based on the same system state (observational time point)
Definition of Transition Rule

- **Transition Rule**
  - Given an event $o_i = c$, a transition rule states *all* the effective conditions that will change (rewrite) the value of the observation $o_i$ to $c$ in the successor *system* state.

  - **Standard Form**
    $$A_1 \lor A_2 \lor \ldots \lor A_n \implies o_i = c$$
    - $A_h (h = 1, 2, \ldots, n)$ is an *effective condition* that consists of a conjunction of events.
    - “$\implies$” denotes a *one-step transition* or rewrite relation in a sense of rewriting logic.
    - Only concerns with *one* event.
Example

\begin{align*}
\text{pos}(T1) &= \text{crossing} \land \text{bar} = \text{open} \\
\text{pos}(T1) &= \text{sect1} \\
\text{err-brake}(T1) \land \text{signal-bypass}(T1) \land \text{level}(T1) &= \text{state2} \\
\implies \text{pos}(T1) &= \text{crossing}
\end{align*}

\begin{align*}
\text{pos}(T1) &= \text{sect1} \\
\text{err-brake}(T1) \land \text{signal-bypass}(T1) \land \text{level}(T1) &= \text{state2} \\
\implies \text{pos}(T1) &= \text{crossing} \\
\text{pos}(T1) &= \text{sect1} \land \text{bar} = \text{open} \\
\text{err-brake}(T1)
\end{align*}
Benefits of Formal FTA

- Formally decompose a complex hazard into more manageable problems (basic events)
  - Correctness and completeness are guaranteed by construction rather than a standalone verification
  - Classification of basic events
    - Uncontrollable failures → Safety Assumptions
    - Controllable failures → Safety Commitments

- System modeling knowledge are also provided as for requirements formulation with OTS/CafeOBJ
  - Events → Signature (observations)
  - Transition rules → Axioms (transitions)
Overview of CafeOBJ

- A wide spectrum specification language based on multiple logic foundations – a modern successor of OBJ
  - Ordered-sorted Logic
  - Rewriting Logic
  - Hidden Sorts

- Important properties of CafeOBJ
  - Equational Specification and Programming
  - Behavioural Specification
  - Rewriting Logic Specification …
Formal Specification with OTS/CafeOBJ

Signature \(\leftrightarrow\) Events

\*\[\text{Sys}\]*  -- hidden sort denoting the universal system state

\text{bop pos}  :  \text{Sys Train} \rightarrow \text{TState}

\text{bop level}  :  \text{Sys Train} \rightarrow \text{LState}

\text{bop bar}  :  \text{Sys} \rightarrow \text{BState}

\text{bop signal-bypass}  :  \text{Sys Train} \rightarrow \text{Bool}

\text{bop err-brake}  :  \text{Sys Train} \rightarrow \text{Bool}

Axioms \(\leftrightarrow\) Transition rules

-- axioms for action (transition rule) \text{a1}

\text{ceq pos(a1(S, T1), T2)} = (\text{if (T1 = T2) then crossing else pos(S, T2) fi})

\text{if ((pos(S, T1) = sect1 and signal-bypass(S, T1) = true)}

\text{or (pos(S, T1 = sect1) and err-brake(T1) = true)}

\text{or (pos(S, T1) = sect1 and level(S, T1) = state2) = true} .

\text{eq bar(a1(S, T1)) = bar(S) .}

\text{eq signal-bypass(a1(S, T1), T2) = signal-bypass(S, T2) .}

\text{eq err-brake(a1(S, T1), T2) = err-brake(S, T2) .}
Interactions between FTA and OTS/CafeOBJ

Fault Trees (FTA/OTS)
- Events
- Transition Rules
- Safety & Reliability Analysis

Formal Specification (OTS/CafeOBJ)
- Signature
- Axioms

Proof Scores (OTS/CafeOBJ)

Revising: more reliable FTs

Common Framework: OTS
Summary & Future Works

- Conclusions
  - Problems of traditional FTAs
  - Formal FTA of OTS
  - Common framework for safety analysis and requirements formulation and verification with OTS/CafeOBJ

- Future works
  - Large practical case study
  - Development of semi-automatic environment
Questions?

Thanks