Dependability Modelling using AADL and the AADL Error Model Annex

Ana Rugina
{aerugina@laas.fr}
Context

- Dependability evaluation for embedded real-time systems

⇒ IST FP6 European Integrated Project ASSERT

**Means:**
Analytical Modelling

- Petri nets,
- Markov chains

**Objectives:**
Dependability Measures

**Industrial practice:**
- UML, AADL models
Outline

• General approach

• Case study

• Conclusion
Approach

AADL Dependability Model

AADL System Architecture Model + AADL System Error Model

Model transformation

GSPN Dependability Model

Model processing

Dependability Measures
Dependability Measures

- **Reliability**
  - measure of continuous delivery of correct service
  - probability of occurrence of a failure before a given instant of time
  - MTTF (mean time to failure)

- **Availability**
  - measure of the readiness for correct service
  - the proportion of correct service deliverance time over a time interval

- **Maintainability**
  - ability to undergo modifications and repairs
  - probability of service restoration before a given instant of time
  - MTTR (mean time to restoration)
Duplex system architecture

**duplex_system**

- **System1**: sub_system.basic_primary
  - **SW1**: software.basic_primary
    - Primary
    - Backup
  - **HW**: computer.basic
- **System2**: sub_system.basic_backup
  - **SW2**: software.basic_backup
    - Primary
    - Backup
  - **HW**: computer.basic

**LAN**
AADL Dependability Model

- **Error model**
  - SW.HWSW_SWSWdep
    - SW-SW dependency
    - HW-SW dependency
  - HW.HWSW_HHWHdep
    - HW-SW dependency
    - HW-HW dependency

- **Primary Backup**
  - SW1: software.basic_primary
  - SW2: software.basic_backup
  - HW: computer.basic

- **Error model**
  - SW.HWSW_SWSWdep
    - SW-SW dependency
    - HW-SW dependency
  - HW.HWSW_HHWHdep
    - HW-SW dependency
    - HW-HW dependency

- **Repairman**
  - Repairman: repairman.basic
  - Repairman.Simple
    - HW-HW dependency
Error Model Type

error model SW
features

SW_Error_Free: initial error state;
SW_Activation_Fault, SW_End_of_Error_Detection_Action,
SW_Error_Non_Detected, SW_Error_Detected,
SW_End_of_Exception_Handling, SW_In_Restart: error state;

SW_Fault, SW_Detection_Action, SW_Detected,
SW_Non_Detected, SW_Non_Detected_Disappear,
SW_Non_Detected_Perceived, SW_Error_Detected_Handling,
SW_Error_Temp, SW_Error_Perm, SW_Restart, Tempo: error event;
end SW;
error model SW
features
  SW_Error_Free: initial error state;
  SW_Activation_Fault,
  SW_End_of_Error_Detection_Action,
  SW_Error_Non_Detected, SW_Error_Detected,
  SW_End_of_Exception_Handling,
  SW_In_Restart: error state;
  SW_Fault, SW_Detection_Action, SW_Detected,
  SW_Non_Detected, SW_Non_Detected_Disappear,
  SW_Non_Detected_Perceived,
  SW_Error_Detected_Handling, SW_Error_Temp,
  SW_Error_Perm, SW_Restart: error event;
end SW;
error model implementation SW.Isolated

transitions
   SW_Error_Free-[SW_Fault] -> SW_Activation_Fault;
   [...]  
   SW_In_Restart-[SW_Restart] -> SW_Error_Free;

properties
   -- a fault occurs following a poisson distribution
   Occurrence => poisson 0,05 applies to SW_Fault;
   [...]  
   -- The restart takes some time
   Occurrence => poisson 60 applies to SW_Restart;

end SW.Isolated;
error model implementation SW.Isolated

transitions

SW_Error_Free-[SW_Fault] -> SW_Activation_Fault;
[...]
SW_In_Restart-[SW_Restart] -> SW_Error_Free;

properties

-- a fault occurs following a poisson distribution
Occurrence => poisson 0.05 applies to SW_Fault;
[...]
-- The restart takes some time
Occurrence => poisson 60 applies to SW_Restart;

end SW.Isolated;
error model implementation SW.Isolated

transitions

- SW_Error_Free-[SW_Fault] -> SW_Activation_Fault;
- SW_Activation_Fault-[SW_Detection_Action] -> SW_End_of_Error_Detection_Action;
- SW_End_of_Error_Detection_Action-[SW_Detected] -> SW_Error_Detected;
- SW_End_of_Error_Detection_Action-[SW_Non_Detected] -> SW_Error_Non_Detected;
- SW_Error_Detected-[SW_Error_Detected_Handling] -> SW_End_of_Exception_Handling;
- SW_Error_Non_Detected-[SW_Non_Detected_Disappear] -> SW_Error_Free;
- SW_Error_Non_Detected-[SW_Non_Detected_Perceived] -> SW_In_ReStart;
- SW_End_of_Exception_Handling-[SW_Error_Temp] -> SW_Error_Free;
- SW_End_of_Exception_Handling-[SW_Error_Perm] -> SW_In_ReStart;
- SW_In_ReStart-[SW_ReStart] -> SW_Error_Free;

properties

- Occurrence => poisson 0.05 applies to SW_Fault;
- Occurrence => poisson 10e+2 applies to SW_Detection_Action;
- Occurrence => fixed 0.7 applies to SW_Detected;
- Occurrence => fixed 0.3 applies to SW_Non_Detected;
- Occurrence => poisson 10e+10 applies to SW_Non_Detected_Disappear;
- Occurrence => poisson 10e+6 applies to SW_Non_Detected_Perceived;
- Occurrence => poisson 10e+2 applies to SW_Error_Detected_Handling;
- Occurrence => fixed 0.98 applies to SW_Error_Temp;
- Occurrence => fixed 0.02 applies to SW_Error_Perm;
- Occurrence => poisson 60 applies to SW_ReStart;
- Occurrence => poisson 10000 applies to Tempo;

end SW.Isolated;
error model SW
features

[...]
SW_KO: in out error propagation;
Both_SW_Dead: in error propagation;
end SW;

error model implementation SW.SWSWdep
features

SW_Needs_Restart, SW_Now_Restart, SW_Both_Dead: error state refines SW_In_Restart;

transitions

[...]
SW_Needs_Restart-[out SW_KO] -> SW_Needs_Restart;
SW_Needs_Restart-[Tempo] -> SW_Now_Restart;
SW_Needs_Restart-[in Both_SW_Dead] -> SW_Both_Dead;

properties

[...]
Occurrence => fixed 1 applies to SW_KO;
end SW.SWSWdep;
Component implementation

system implementation software.primary
modes
  primary: initial mode;
  backup: mode;
  primary-[inp] -> backup;
  backup-[notification] -> primary;

annex Error_Model {**
  Model => Mymodels::SW. SWSWdep;
  Vote_In => SW_Both_Dead when inp[SW_KO] and notification[SW_KO]
             applies to inp, notification;
  Vote_Transition =>
    inp[SW_KO]
    applies to inp;
  Vote_Transition =>
    notification[SW_KO]
    applies to notification; **};
end software.primary;
Error Model Annex Evolution

- Occurrence properties
  - parametric

- Link between the mode model and the error model
  - mode-dependent behaviour in presence of faults

- Vote_In and Vote_Out properties
  - evaluate Boolean error expressions when needed

- Inheritance and refinements
  - similarly to the core standard mechanisms
Summary

• AADL system error model
  • Stepwise construction
    ▪ Building error models as if components were isolated
    ▪ Adding dependencies progressively

• Error Model Annex assessment

• Model transformation: manual ⇒ automatic
Dependability Modelling using AADL and the AADL Error Model Annex

Ana Rugina
{aerugina@laas.fr}