Reliability and availability requirements engineering within the Unified Process using a Dependability Analysis and Modeling profile

Simona Bernardi, Università di Torino
José Merseguer, Universidad de Zaragoza
Robyn R. Lutz, Iowa State University
Outline

- Improve elicitation, documentation and analysis of R&A sw requirements within the Unified Process (UP)
- Extension of the requirement workflow to handle R&AR
  - Step-by-step incremental process
  - Use of a UML profile (DAM) to 1) specify R&AR and 2) characterize system faults/failures
- Application to an intrusion-tolerant, distributed firewall for critical information infrastructures (CRUTIAL IST project)
Motivation

- Toward the definition of a methodology for the synergetic use of dependability techniques within the UP

- Why the Unified Process (UP) ?
  - Incremental & iterative: manages risks and handles changes in sw projects better than waterfall models
  - Uses UML as its specification language
  - Can be customized for different kind of sw systems/application domains

- UP pays little attention to non-functional reqs

- Several UML profiles exist that help to gather NFPs
  - MARTE OMG standard profile
  - DAM profile for dependability NFPs
## Unified Process & req. workflow

<table>
<thead>
<tr>
<th>Workflows</th>
<th>Inception</th>
<th>Elaboration</th>
<th>Construction</th>
<th>Transition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Requirements</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Analysis</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Design</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Implementation</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Test</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Phases

- **Inception**
  - It. #1
  - It. #2
  - It. #i
  - It. #i+1
  - It. #n
  - It. #n+1
  - It. #n+2
  - It. #n+3
  - It. #m
  - It. #m+1

### Roles

- **System Analyst**
  - Find actors & UCs

- **Architect**
  - Prioritize UCs
  - Structure UC model

- **UC Specifier**
  - Detail UCs

- **UI Designer**
  - Prototype UI
The set of dependability reqs specification techniques

- (Mis)Use cases
- IEEE Std. 830-1998
  - IEEE Recommended practise for sw requirements specification
- DAM profile
- Fault Trees
(Mis)Use Cases

- Use Cases are textual specifications
- Use of templates, like the Cockburn's one
IEEE 830-1998

- Recommends approaches for SW req specification and describes contents and qualities of a good SRS
- UP Supplementary Spec document inspired by IEEE 830-1998
DAM profile

- DAM Profile has been devised to annotate the design, in this work we use it to specify R&AR
- It is a specialization of the MARTE profile
- MARTE NFP types enable to describe relevant dependability aspect using “properties”:
  - Value: value/parameter name
  - Expr: VSL expression
  - Source: origin of the NFP (req, est, msr, assm)
  - StatQ: statistical qualifier (mean, min, max, ..)
Fault Trees

- FTs are used to
  - Gather information about the potential contributing causes to threats
  - Trace the combination of faults/failures to misuse and use cases
A running example from CRUTIAL project
Step-by-step process: $i^{th}$ iteration in the requirement workflow (I)

- **Input:** $DM_{i-1}, UCD_{i-1}, SS_{i-1}$
- **Output:** $DM_i, UCD_i, SS_i$

1. Discover new UCs, MUCs and actors:
   $$UCDi \leftarrow UCDi-1 \cup UC_{\text{new}} \cup MUC_{\text{new}} \cup AC_{\text{new}}$$

2. Select UCs to be specified: $selUC_i \subseteq UDCi$

3. For all $uc \in selUC_i$
   1. Specify($uc$)
UC specify activity

- Textual description of the UC using Cockburn template
- R&AR from the Special Requirement section
  - Application of DAM profile for rewriting them in a standard and disciplined form
UCD_{i-1}

- **Destination**
- **Sender**
- **Attacker**

**Diagram:**
- **CIS PS**
- **Payload corruption**
- **Generation of illegal traffic**
- **PRRW Service**

- <<include>>
- <<threatens>>
- <<mitigates>>
# CIS PS use case description

<table>
<thead>
<tr>
<th>UC Name</th>
<th>CIS Protection Service</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scope</td>
<td>SCADA</td>
</tr>
<tr>
<td>Main Actors</td>
<td>Sender (computer from the WAN), Receiver (computer of the protected LAN)</td>
</tr>
</tbody>
</table>
| Success guarantee | The correct message is eventually delivered  
The illegal message is not delivered |
| Main scenario | A message is sent by Sender to Receiver  
• It arrives to the CIS Firewall  
• Each CIS Firewall checks if it satisfies the security policy and votes  
• The CIS firewalls agree upon a final judgement (*majority voting*)  
• The message is correct and the CIS Firewall leader forwards it  
• to the Receiver |
| Alternate scenarios | 4.a The message is illegal, then it is not delivered |
| Special Reqs | A1. The CIS PS should be available 99.99% of the time  
R1. The MTBF shall be at least 6 months |
| Relationships | CIS includes PRRW Service, Payload Corruption threatens CIS PS,  
CIS PS mitigates Generation of illegal traffic |
DAM annotation to CIS PS use case

Sender >> DAM annotation

<<tupleType>> DaFailure

- MTBF: NFP_Duration[*]
- ssAvail:NFP_Percent[*]
- failure: DaFailure[*]

<<stereotype>> DaService

ssAvail:(value=99.9%, statQ=min, source=red);
failure = (MTBF = (value=(6, month), statQ=min, source=red));
ssAvail=(value=99.9%, statQ=min, source=red);
Step-by-step process: $i^{th}$ iteration in the requirement workflow (II)

4. Select MUCs related to $selUC_i$: $selMUC_i \subseteq UDC_i$
5. For all $muc \in selMUC_i$ do
   1. $Specify(muc)$
MUC specify activity

- Textual description of the MUC using Cockburn template
- Threats information from Success guarantee, Main/Alternate scenario and Other Req's sections
  - Application of the DAM profile to characterize from both a qualitative/quantitative viewpoints faults/failures
- Faults Trees are used to formally specify UCD relationships
  - Among Negative Actor actions and Misuse Case success
  - Among Misuse Cases and related Use Case
UCD₀

- Generation of illegal traffic
- Payload corruption

<<include>>
<<threatens>>
<<mitigates>>
## Payload Corruption MUC description

<table>
<thead>
<tr>
<th>MUC Name</th>
<th>Payload Corruption</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scope</td>
<td>CIS PS</td>
</tr>
<tr>
<td>Main Actors</td>
<td>Attacker: Outside and Inside Threats</td>
</tr>
<tr>
<td>Success guarantee</td>
<td>The Payload evaluates as “correct” an illegal message or it evaluate as “illegal” a correct message (FM1), or it is subject to a temporary omission (FM2)</td>
</tr>
</tbody>
</table>
| Main Scenario (Outside Threat) | The Attacker identifies the WAN traffic replicator as potential target  
  ● The Attacker sniffs the network traffic  
  ● The Attacker gets an unauthorized access to an host in the LAN  
  ● The Attacker install a malicious logics in the accessed host  
  ● The hosted Payload behaves in an unpredicted manner. |
| Special Reqs   | F1. At most f Payloads can be concurrently corrupted  
  F2. f should be set according to the expected rate of fault occurrence |
| Relationships  | Payload Corruption threatens CIS PS                    |
payload corruption

numberOfFaults=(value=$f,statQ=max,source=est/msr);
fault = (type = (value=malicious-logic);
occurrenceRate = (value=$fr1,statQ=mean,source=est/msr);
effect = (domain = (value=invalid,omission))

tupleType>>
DaFailure

tupleType>>
DaFault

tupleType>>
DaFaultGenerator

<<stereotype>>
DaFaultGenerator
	numerOfFaults:NFP_Integer[*]
fault:DaFault

<<stereotype>>
DaFailure

domain:Domain[*]
...

<<tupleType>>
DaFault

type:FaultType[*]
occurrenceRate:NFP_Frequency[*]
effect: DaFailure[*]
Use of FT to formalize MUC-UC relationships

- CIS PS failure
  - Quorum not reached or wrong judgement
  - The leader is corrupted (fails to fwd the approved message to Destination)

- The leader is corrupted (fails to fwd the approved message to Destination)

- [n/2]+1:n corrupted

- P omission (FM2)

- P1 corrupted

- P1 invalid (FM1)

- P1 omission (FM2)

- <<DaService>>
  - CIS PS

- <<threatens>>

- <<DaFaultGenerator>>
  - Payload corruption
Step-by-step process: $i^{th}$ iteration in the requirement workflow (III)

6. Discover new NFRs: $SS_i \leftarrow SS_{i-1} \cup NFR_{new}$
7. Select a subset of requirements: $selNFR_i \subseteq SS_i$
8. Forall $nfr \in selNFR_i$ do
   1. Elaborate($nfr$)
9. Restructure UCD$_i$ and DM$_i$ if necessary
NFR elaboration activity

- Rewriting of further NFR from the SS, related to dependability/fault-tolerance with the DAM profile
  - Annotation in the Domain Model/Use Case Diagrams
IEEE 830-1998

- Recommends approaches for sw req specification and describes contents and qualities of a good SRS
- UP Supplementary Spec document inspired by IEEE 830-1998

Template of “Specific requirements"

3. Specific requirements
   3.1 External interface requirements
      3.1.1 User interfaces
      3.1.2 Hardware interfaces
      3.1.3 Software interfaces
      3.1.4 Communications interfaces
   3.2 Functional requirements
   3.3 Performance requirements
   3.4 Design constraints
   3.5 Software system attributes
      3.5.1 Reliability requirements
      3.5.2 Availability requirements
      3.5.3 Security requirements
      3.5.4 Maintainability requirements
      3.5.5 Portability requirements
   3.6 Other requirements

3.6 Other requirements:
(Fault Tolerance) There shall be at least 2f+1 CIS Firewalls to tolerate f concurrent faults
DAM annotation to Domain Model

Faults:
2f+1 CIS Firewalls to tolerate f concurrent (Fault Tolerance). There shall be at least

3.6 Other Requirements:

There shall be at least 2f+1 CIS Firewalls to tolerate f concurrent (Fault Tolerance).
Conclusions

- The DAM annotated UML artifacts (UCD, DM) provide input for the other UP workflows (design, test, ..) as well as for V&V activities

- Next steps:
  - Study of the DAM applicability in the other UP workflows
  - V&V activities driven by DAM annotated M(UC)s
Thank you!
DAM Threats model

System::Core::Component

System::Core::Connector

System::Redundancy::RedundantStructure

ErrorPropagation Relation

Impairment

ErrorPropagation

Fault

Error

Fault Generator

SystemCore::Core::Step

Failure

domain
MTTF
....

Hazard

severity
risk
....

ErrorStep

FailureStep

HazardStep
DAM profile overview

- <<profile>> MARTE::GQAM
- <<import>> DAM
- <<modelLibrary>> DAM_Library
- DAM_UML_Extensions
- MARTE::MARTE_Library::BasicNFP_Types
- DAM::DAM_Library
- <<import>> MARTE::NFPs
- Basic_DA_Types
- Complex_DA_Types
- <<profile>> MARTE::VSL::DataType
- <<apply>>