Making Modeling Assumptions an Explicit Part of Real-Time Systems Models

Pierre de Saqui-Sannes - ISAE-SUPAERO
Ludovic Apvrille - Telecom ParisTech

ERTS 2016, Toulouse, France
January 27, 2016
Outline

Introduction
  Models

Contribution
  Language, method, tool

Demonstration
  UAV

Conclusion
  Objective, contributions, future work and resources
Rationale

- A model abstracts a system
  - Mastering the complexity of real-time systems
- A model is valid for a precise set of assumptions about the system and the system’s environment
- A model should always be associated with the assumptions the designer has made to build up it
  - When the model evolves, the assumptions should evolve accordingly
  - And reciprocally

Our contribution
Making the modeling assumptions a full part of the model
Our Contribution

Objectives

- Solution not restricted to one modeling language
- Supports incremental modeling methods and versioning
- Language, method, tool

Our solution

- A new diagram: Modeling Assumption Diagram (MAD) added to SysML-Sec
- Assumptions attributes
- Relations between modeling assumptions
- Supported by a free and open-source tool (TTool)
- Case study: a UAV
MAD: Assumption Stereotype

- **Stereotype** = system, environment
- **Durability** = temporary, permanent
- **Status** = applied, alleviated
- **Source** = end-user, stakeholder, creator of the model
- **Scope** = language, tool, modeling activity, verification

<<System Assumption>>
CrossingSigns

Text="The drone is informed about the crossing identified with a sign"
Durability="Permanent"
Source="Model creator"
Status="Applied"
Scope="Modeling activity"
MAD: Relations

- **Containment**: a high-level assumption split up into elementary ones
- **Impact**: model elements impacted by a given assumption

![Diagram](image)
MAD: Relations (Cont.)

- Modification of assumptions → Modification of the system model
- Tracing (i) the versions of assumptions, and (ii) how the versions impact the system model
- **Versioning** relation: \( a \) applies until version \( x \) and becomes \( b \) at version \( y \)
Method

Introduction

Contribution

Demonstration

Conclusion

Method

SW/HW Partitioning

Requirements

Architectural view

Functional view

Mapping view

Simulation

Formal analysis

Attacks

Simulation

Formal analysis

Structural view

Behavioral view

Deployment view

System Design

Simulation

Formal analysis

Test

Institut Mines-Telecom
Method

- Assumptions (MAD)
- Requirements
- Attacks
- SW/HW Partitioning
  - Functional view
  - Architectural view
  - Mapping view
- System Design
  - Structural view
  - Behavioral view
  - Deployment view
- Test
An Autonomous UAV

Autonomous drone navigation in harsh conditions, in particular inside buildings

3 scenarios (with different assumptions)

1. **Understanding marks**, e.g., a red line located on the floor
2. **Analyzing the environment** (obstacles, etc.) with image-based processing techniques (3D reconstruction).
3. **Going through obstacles with human assistance**
   - Entering in a room when the entrance door is closed.
**Introduction**

**Contribution**

**Demonstration**

**Conclusion**

---

**TTool: MAD Scenario #1**

---

**Modeling Assumptions**
TTool: Editing an assumption
TT Tool: Design #1

**Introduction**

**Contribution**

**Demonstration**

**Conclusion**

Command values:

0: negative value
1: nothing
2: positive value

Command mode:

0: must take into account pitch, roll and yaw
1: takeOff
2: land

**Modeling Assumptions**

- pitch : int;
- roll : int;
- yaw : int;
- mode : int;

**Datatype**

- sizeH = 0 : int;
- sizeV = 0 : int;
- data = 0 : int;
- bottom : bool;

**Block**

- in compute(Picture p)
  - out result(bool found)

**Datatype**

- Picture
  - sizeH = 0 : int;
  - sizeV = 0 : int;
  - data = 0 : int;
  - bottom : bool;

**Block**

- in placeReached()
- in missionAborted(int status)
- out reach(int place)
**TTool: MAD Scenario #1 → #2**

- **SignsToNavigate**
  - Text: "Assumption description: Double-click to edit"
  - Durability: "Permanent"
  - Source: "Stakeholder"
  - Status: "Applied"
  - Scope: "Undefined"

- **CrossingSigns**
  - Text: "The drone is informed about the crossing identified with a sign"
  - Durability: "Permanent"
  - Source: "Model creator"
  - Status: "Applied"
  - Scope: "Modeling activity"

- **RedLine**
  - Text: "A reline is assumed to be painted/sticked on the floor in order to guide the drone."
  - Durability: "Temporary"
  - Source: "Model creator"
  - Status: "Applied"
  - Scope: "Modeling activity"

- **ThreeDRecognition**
  - Text: "the drone can follow corridors without any sign. It recognizes the path to take by understanding its 3D environment."
  - Durability: "Permanent"
  - Source: "Model creator"
  - Status: "Applied"
  - Scope: "Modeling activity"
TTool: Design #2

Introduction

Contribution

Demonstration

Conclusion

Modeling Assumptions

Command values:
---------------------
0: negative value
1: nothing
2: positive value

Command mode:
-------------------
0: must take into account pitch, roll and yaw
1: takeOff
2: land
**TTool: MAD Scenario #2 → #3**

**System Assumption - NoDoor**
- Text: "The drone cannot go through doors."
- Durability: Temporary
- Source: Model creator
- Status: Applied
- Scope: Modeling activity

**System Assumption - FollowingPersons**
- Text: "The drone can follow a person when it needs to go through an obstacle, e.g., a door."
- Durability: Permanent
- Source: Model creator
- Status: Applied

**Diagram ref.**
- DroneDesign3
- Block_RemoteUSer
- Block_MainController

**Element ref.**
- Block_RemoteUSer
- Block_MainController
- DroneDesign3
TTool: Design #3

<<block>>
CrossingRecognition
- foundI : int;
- found = false : bool;
- p : Picture;
- minComputationTime = 10 : int;
- maxComputationTime = 10 : int;
~ in compute(Picture p)
~ out result(bool found)

<<block>>
MainController
- nbOfCrossings : int;
- aliveTimer = 100 : int;
- mode : int;
- currentPicture : Command;
- movementDelay = 100 : int;
- found : bool;
- position : int;
- hasCurrentPicture : bool;
- oldMode : int;
~ out sendCommand(Command c)
~ in getPosition(Command p)
~ in readNextPicture(Picture p)
~ out startCamera()
~ out stopCamera()
~ missionAborted(int status)
~ placeReached()...

<<block>>
ThreeDRecognitionAlgorithm
- found = false : bool;
- foundI : int;
- position : int;
- minComputationTime = 30 : int;
- maxComputationTime = 70 : int;
- p : Picture;
~ in compute(Picture p)

<<block>>
Camera
- p : Picture;
- x : int;
- interframeDelay = 20 : int;
- bottom = false : bool;
~ out sendPicture(Picture p)
~ in stopMe()
~ in startMe()

<<block>>
EngineController
- lastCommand : Command;
- currentPosition : Command;
~ in getCommand(Command c)
~ out sendPosition(Command c)

<<block>>
RemoteUser
- place : int;
- status : int;
- doorTime = 100 : int;
~ in placeReached()
~ in missionAborted(int status)
~ in doorDetected()
~ out doorHandled()

<<datatype>>
Command
- pitch : int;
- roll : int;
- yaw : int;
- mode : int;

<<datatype>>
Picture
- sizeH = 0 : int;
- sizeV = 0 : int;
- data = 0 : int;
- bottom : bool;

Command values:
----------------
0: negative value
1: nothing
2: positive value

Command mode:
---------------
0: must take into account pitch, roll and yaw
1: takeOff
2: land

---
Conclusion

Objective
Making modeling assumptions a part of every real-time system model

Contributions
▶ Model Assumption Diagram (MAD), attributes, relations
▶ Included into SysML-Sec, supported by TTool

Future work
▶ Optimizing simulation and verification with regards to versioning
To Go Further . . .

- **TTool**: http://ttool.telecom-paristech.fr
- **SysML-Sec**: http://sysml-sec.telecom-paristech.fr

- **Drone4u project**: http://drone4u.telecom-paristech.fr
  - Several videos of the UAV in action!