JOHANNES KEPLER
UNIVERSITÄT LINZ

Validation-Driven Development

Sebastian Stock, Atif Mashkoor

JOHANNES KEPLER UNIVERSITY LINZ
Altenberger Straße 69
4040 Linz, Austria
jku.at

Typical modeling experience

- Formal specification part of the RE

- Modeling via experience
  - Structure of specification
  - When to validate something

- Verification
  - It the specification consistent?
  - Proving or MC
  - Main focus

- Validation?

Validation

- The “unloved” child
  - Late
  - Experience decides

- Consistency, completeness, tracing & maintenance of requirements

- Informale components?

- Involving all stakeholders
  - As a good RE component should
  - Showing them they get what they want
  - Late involvement produces late feedback

Global challenges

- Rely less on experience of individuals
  - Reproducible decisions
  - Without reading documentation (that does often not exist in the first place)

- Specification structure
  - What to validate together?
  - What breaks what part?
  - What gives greatest benefit?

Challenges - Specification level

- Consistency & Completeness
  - Measure this?
  - Hard when we validate a finished model

- Tracing & Maintenance
  - Point to the right part
  - Notice if something breaks

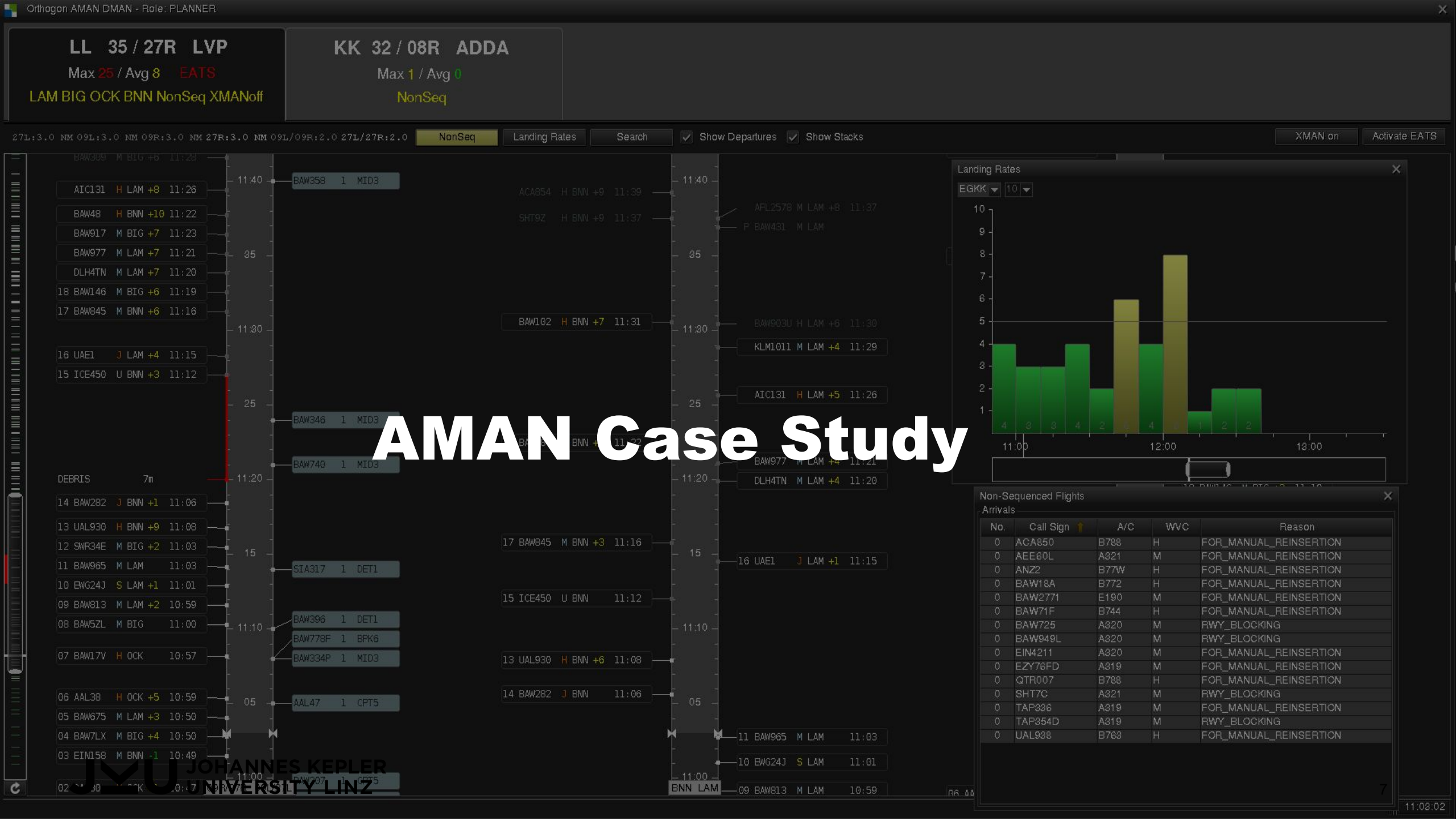
Solution = VDD

- Provides heuristic for specification structure
  - When to refine
  - How to refine
  - Calculating the advantages for a refinement step

- Uses VOs for
  - Consistency & Completeness
  - Tracing & Maintenance

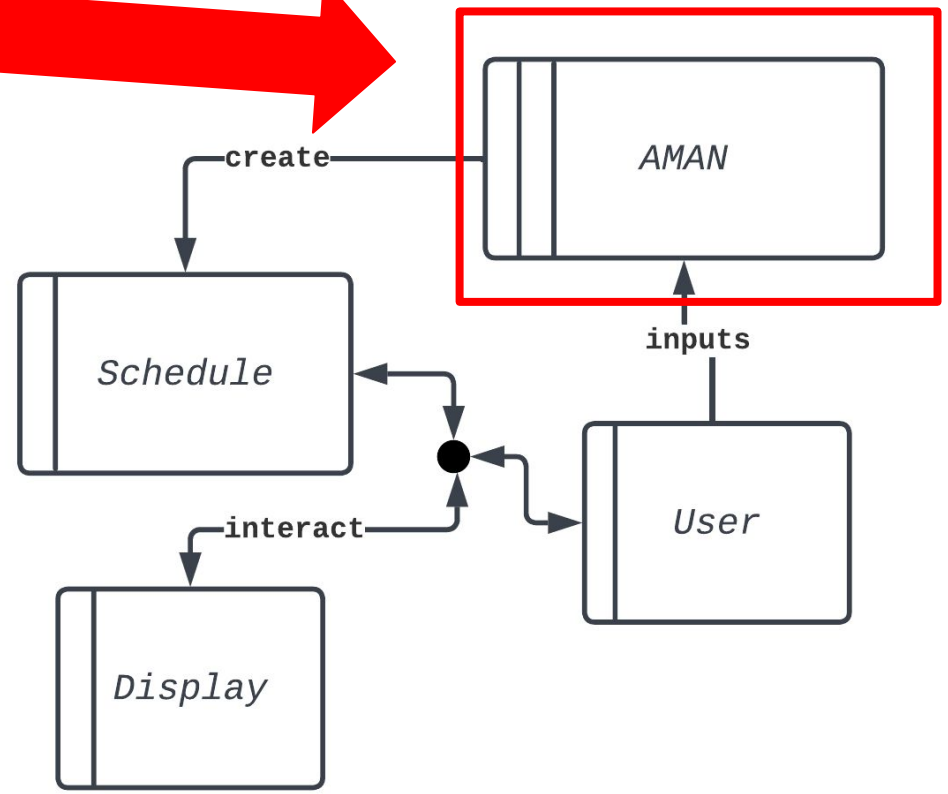
- Bonus: Less experience needed

AMAN Case Study



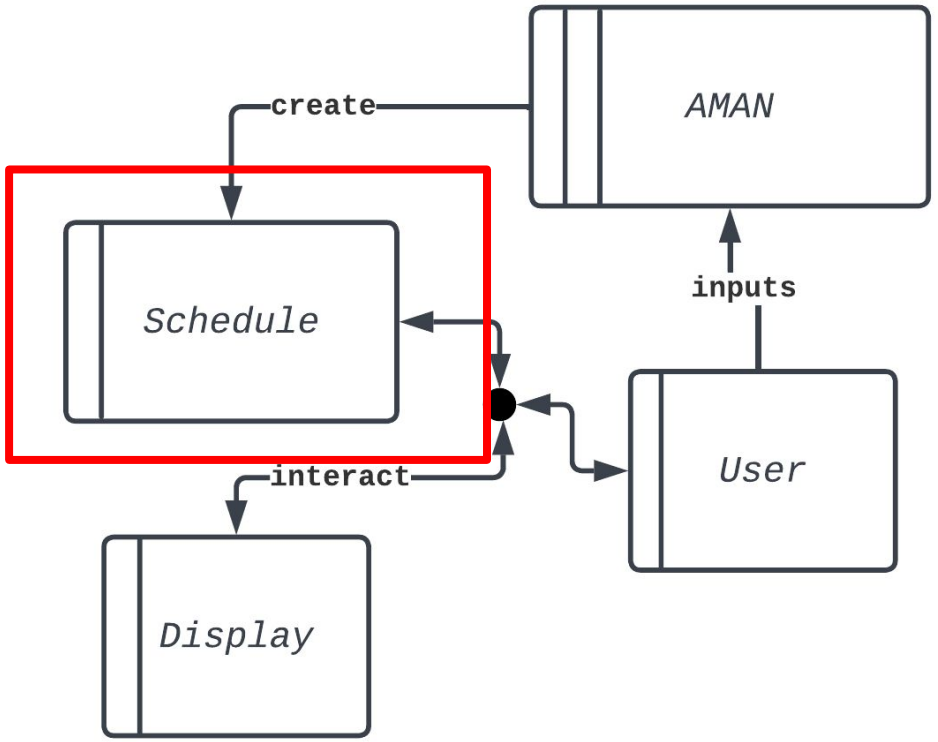
Specification structure - Problem frames

- AMAN is the goal



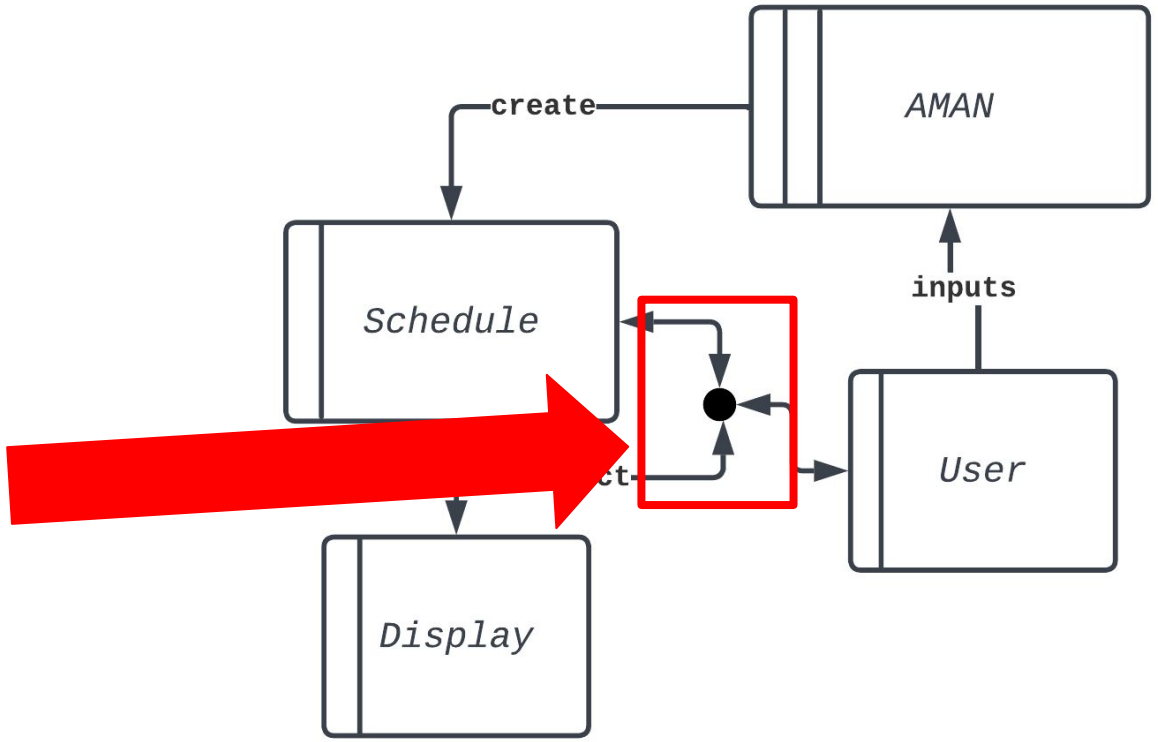
Specification structure - Problem frames

- AMAN is the goal
- Three main areas of concern
  - Schedule (Computer)
  - User interaction (Human)
  - Display (Optional)



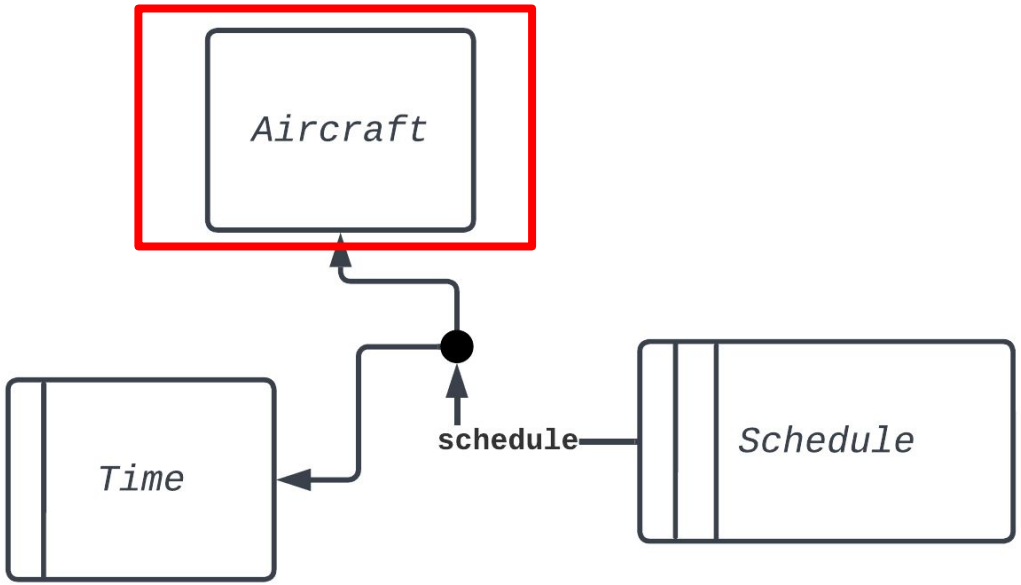
Specification structure - Problem frames

- AMAN is the goal
- Three main areas of concern
  - Schedule (Computer)
  - User interaction (Human)
  - Display (Optional)
- Interfaces
  - Indicate data read
  - User and Schedule and Display
  - AMAN creating schedules
  - User interrupts AMAN



Specification structure - Sub Problems

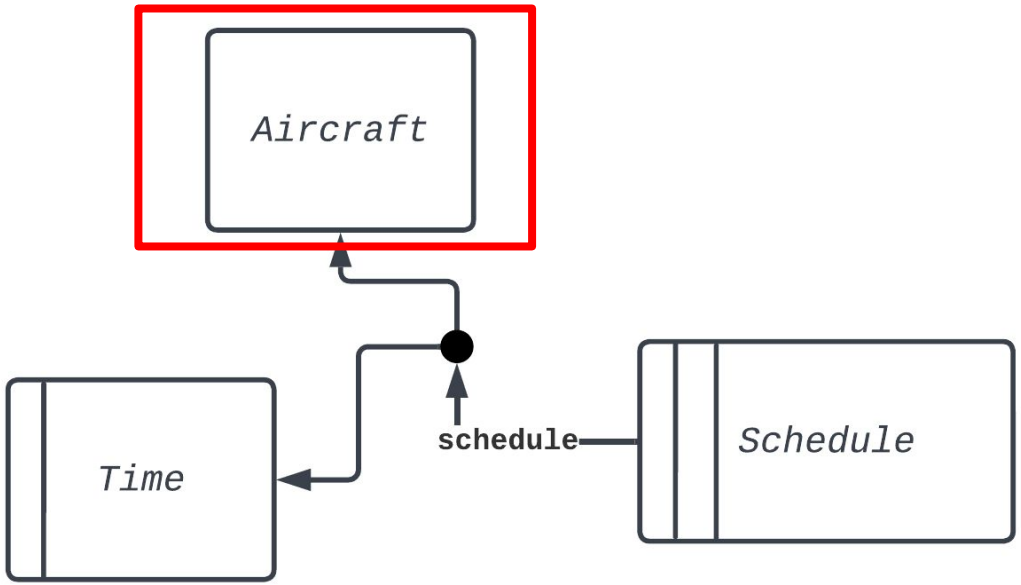
- Missing details
  - Create a sub problem frame
  - Aircraft kept abstract
  - Time is complex



Specification structure - Sub Problems

- Missing details
  - Create a sub problem frame
  - Aircraft kept abstract
  - Time is complex

- Interaction between them
  - Schedule manipulates Time and Aircraft



Deriving a refinement structure

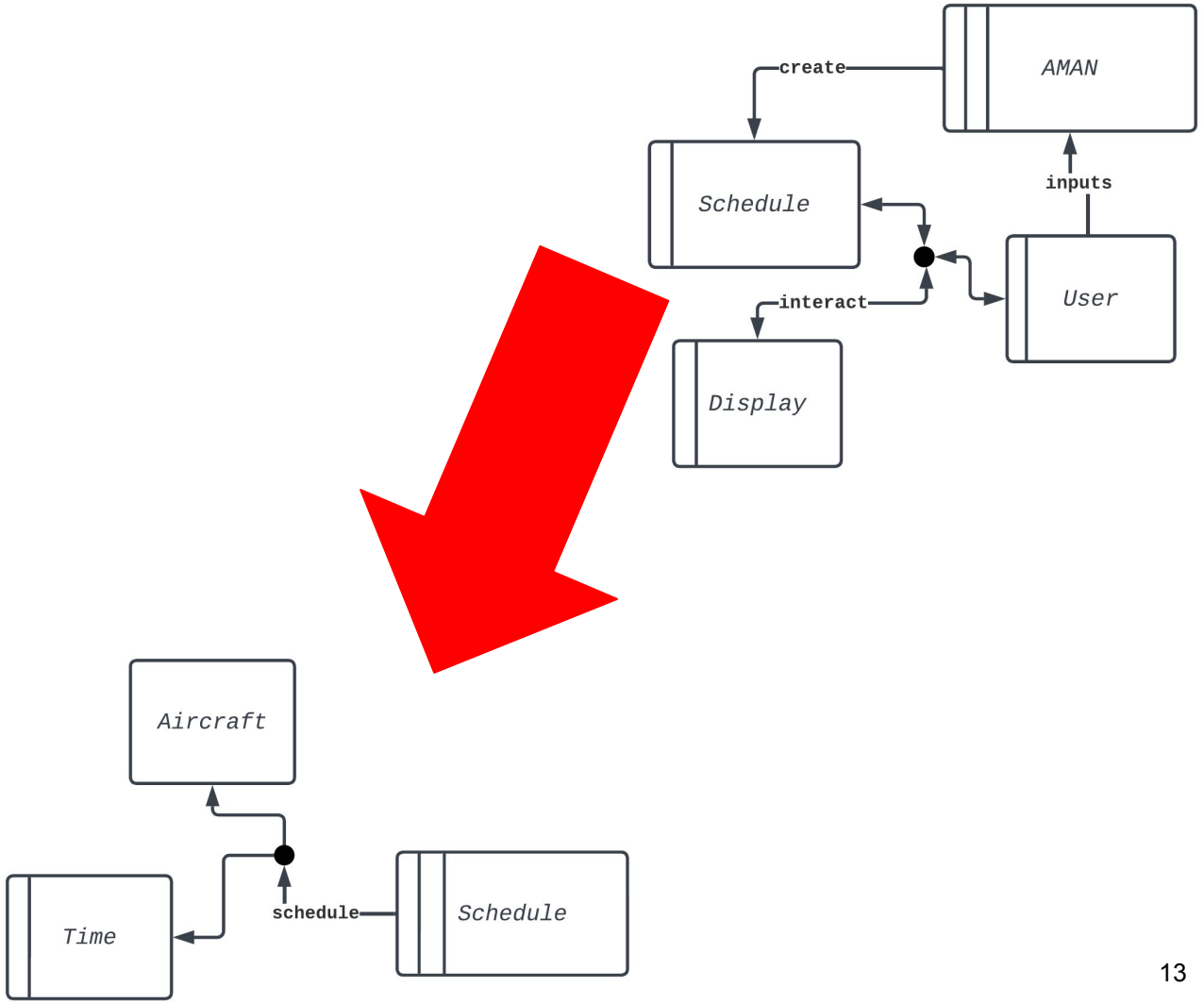
1. Domains sharing interfaces will refine each other eventually

2. The first domain implemented might be the one with most incoming interfaces

3. Global validation vs. local validation

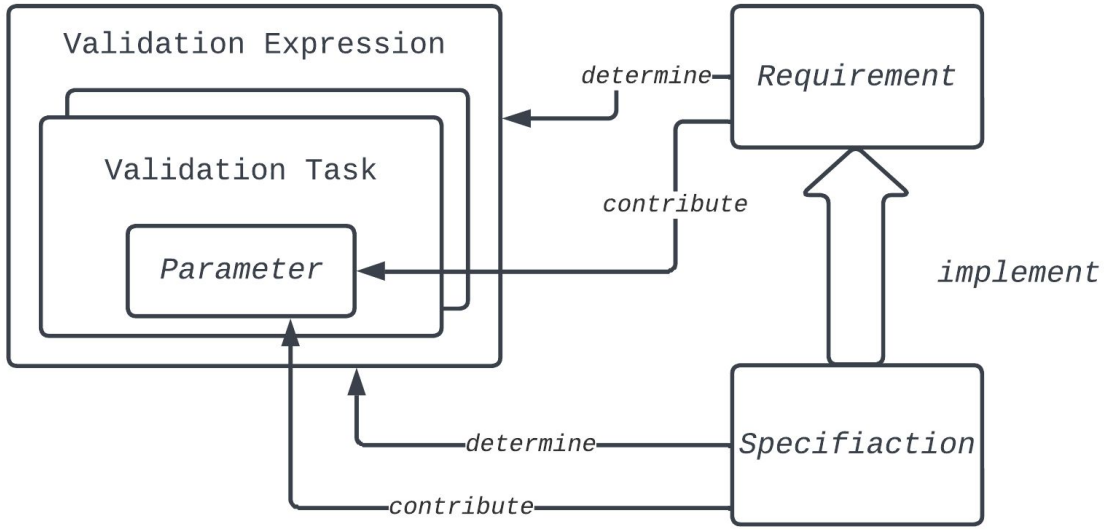
4. Multiple domains sharing interfaces will refine each other vertically

5. Domains not connected to the main goal ar secondary



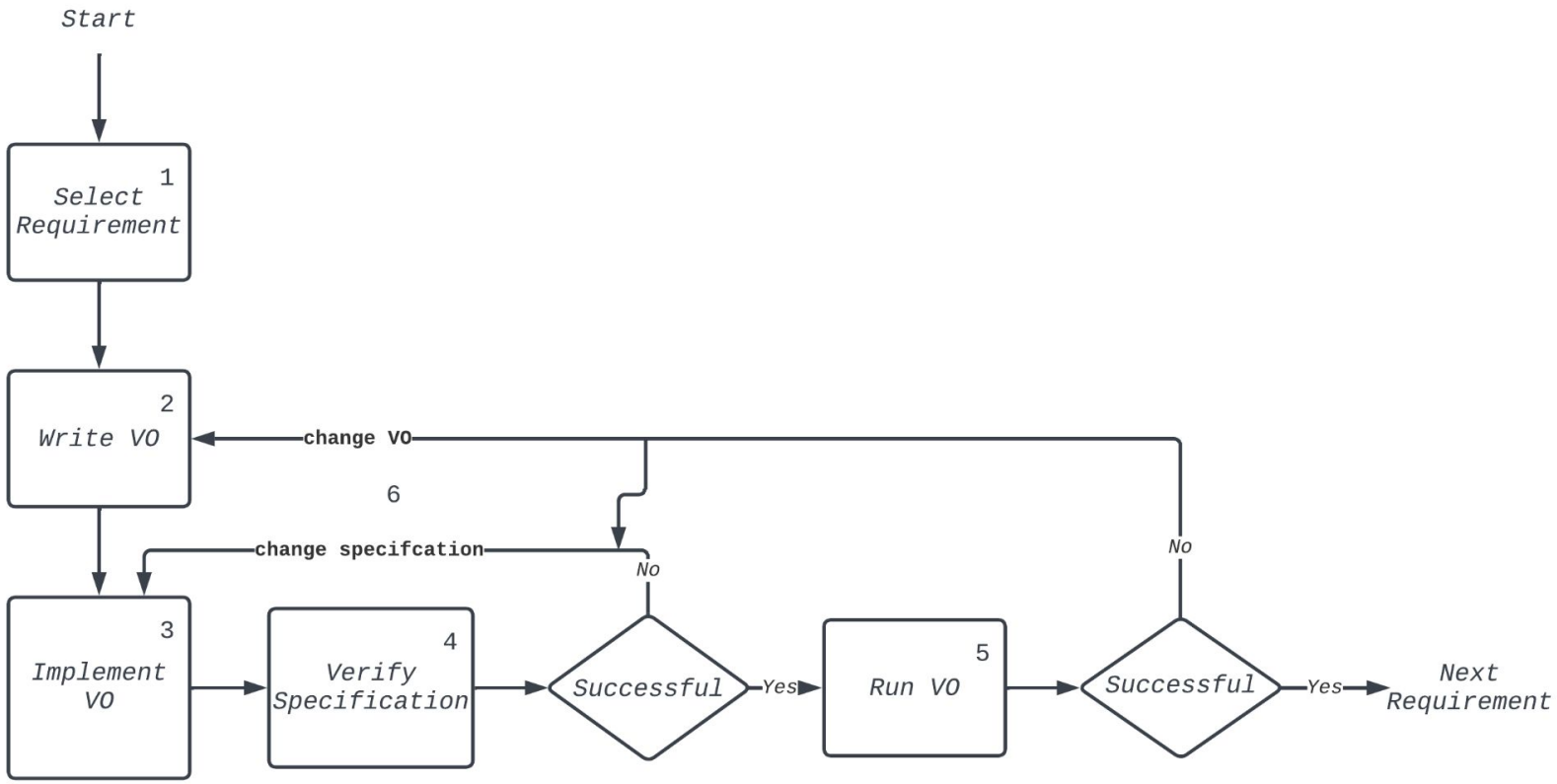
Implementing the specification

- Using VOs
  - One requirement
  - Tasks that show the requirement
  - Applied to one specification



- Every requirement one VO
  - Traces requirement to specification
  - Measurement of completeness
  - Maintenance requirement

VDD implementation cycle



Example

- `REQ1`: Planes can be added to the flight sequence, e.g., planes arriving in close range of the airport.

Example

- `REQ1` : Planes can be added to the flight sequence, e.g., planes arriving in close range of the airport.

$$\texttt{REQ1/M0} : \texttt{GF}(\texttt{BA}(\texttt{scheduledAirplanes} \neq \texttt{scheduledAirplanes\$0})) \implies \texttt{GF}(\texttt{BA}(\{\exists x.(x \in \texttt{scheduledAirplanes} \wedge x \notin \texttt{scheduledAirplanes\$0})\}))$$

Example

- `REQ1` : Planes can be added to the flight sequence, e.g., planes arriving in close range of the airport.

$$\mathtt{REQ1/M0} : \mathtt{GF}(\mathtt{BA}(\mathtt{scheduledAirplanes} \neq \mathtt{scheduledAirplanes\$0})) \implies \mathtt{GF}(\mathtt{BA}(\{\exists x.(x \in \mathtt{scheduledAirplanes} \wedge x \notin \mathtt{scheduledAirplanes\$0})\}))$$

```
machine M0_AMAN_Update sees M0_AMAN_Update_Ctx

variables scheduledAirplanes

  invariants
    @inv0,1 scheduledAirplanes ⊆ AIRPLANES

events
  event INITIALISATION
    then
      @act0,1 scheduledAirplanes ≔ ∅
  end

  event AMAN_Update
    any newScheduledAirplanes
    where
      @grd0,1 newScheduledAirplanes ⊆ AIRPLANES
    then
      @act0,1 scheduledAirplanes ≔ newScheduledAirplanes
  end
```

Example - Refinement

- REQ5: The space between two aircraft is always =< 3, with 3 being the time in minutes.

Example - Refinement

- REQ5: The space between two aircraft is always =< 3, with 3 being the time in minutes.

$$
\begin{aligned}
\mathtt{REQ5/M1} : & \forall \mathtt{a1}, \mathtt{a2} \cdot \mathtt{a1} \in \mathtt{dom}(\mathtt{landing\_sequence}) \wedge \\
& \mathtt{a2} \in \mathtt{dom}(\mathtt{landing\_sequence}) \wedge \mathtt{a1} \neq \mathtt{a2} \implies \\
& (\mathtt{DIST}(\mathtt{landing\_sequence}(\mathtt{a1}) \mapsto \mathtt{landing\_sequence}(\mathtt{a2})) \\
& \geq \mathtt{AIRCRAFT\_SEPARATION\_MIN})
\end{aligned}
$$

Example - Refinement

- REQ5: The space between two aircraft is always =< 3, with 3 being the time in minutes.

$$
\begin{aligned}
\text{REQ5/M1} : & \forall a1, a2 \cdot a1 \in \mathrm{dom}(\mathrm{landing\_sequence}) \wedge \\
& a2 \in \mathrm{dom}(\mathrm{landing\_sequence}) \wedge a1 \neq a2 \implies \\
& (\mathrm{DIST}(\mathrm{landing\_sequence}(a1) \mapsto \mathrm{landing\_sequence}(a2)) \\
& \geq \mathrm{AIRCRAFT\_SEPARATION\_MIN})
\end{aligned}
$$

```
1 machine M1_Landing_Sequence refines M0_AMAN_Update sees M1_Landing_Sequence_Ctx
2
3 variables landing_sequence
4
5
6 invariants
7   @inv1,1 landing_sequence ∈ AIRPLANES ⇸ PLANNING_INTERVAL
8   @inv13,2 ∀a1,a2 · a1 ∈ dom(landing_sequence) ∧
9              a2 ∈ dom(landing_sequence) ∧ a1 ≠ a2 ⇒
10            (DIST(landing_sequence(a1) ↦ landing_sequence(a2))
11            ≥ AIRCRAFT_SEPARATION_MIN)
12  @glue1,1 scheduledAirplanes = dom(landing_sequence)
13
14 events
15   event INITIALISATION
16     then
17       @act1,1 landing_sequence ≔ ∅
18   end
19
20   event AMAN_Update refines AMAN_Update
21     any new_landing_sequence
22     where
23       @grd1,1 new_landing_sequence ∈ AIRPLANES ⇸ PLANNING_INTERVAL
24       @inv1,2 ∀a1,a2 · a1 ∈ dom(new_landing_sequence) ∧
25                a2 ∈ dom(new_landing_sequence) ∧ a1 ≠ a2 ⇒
26               (DIST(new_landing_sequence(a1) ↦ new_landing_sequence(a2))
27               ≥ AIRCRAFT_SEPARATION_MIN)
28     with
29       @wit1,1 newScheduledAirplanes = dom(new_landing_sequence)
30     then
31       @act1,1 landing_sequence ≔ new_landing_sequence
32   end
33 end
```

Example - Refinement

- REQ5: The space between two aircraft is always =< 3, with 3 being the time in minutes.

$$\begin{aligned}\text{REQ5/M1} :& \forall a1, a2 \cdot a1 \in \mathrm{dom}(landing\_sequence) \wedge \\ & a2 \in \mathrm{dom}(landing\_sequence) \wedge a1 \neq a2 \implies \\ & (\mathrm{DIST}(landing\_sequence(a1) \mapsto landing\_sequence(a2)) \\ & \geq AIRCRAFT\_SEPARATION\_MIN)\end{aligned}$$

```
1 machine M1_Landing_Sequence refines M0_AMAN_Update sees M1_Landing_Sequence_Ctx
2
3 variables landing_sequence
4
5
6 inv
7         landing_sequence ∈ AIRPLANES ⇸ PLANNING_INTERVAL
8       @inv3,2 ∀a1,a2 · a1 ∈ dom(landing_sequence) ∧
9                a2 ∈ dom(landing_sequence) ∧ a1 ≠ a2 ⇒
10              (DIST(landing_sequence(a1) ↦ landing_sequence(a2))
11              ≥ AIRCRAFT_SEPARATION_MIN)
12   @glue1,1 scheduledAirplanes = dom(landing_sequence)
13
14 events
15   event INITIALISATION
16     then
       @act1,1 landing_sequence ≔ ∅

20   event AMAN_Update refines AMAN_Update
21     any new_landing_sequence
22     where
23       @grd1,1 new_landing_sequence ∈ AIRPLANES ⇸ PLANNING_INTERVAL
24       @inv1,2 ∀a1,a2 · a1 ∈ dom(new_landing_sequence) ∧
25                a2 ∈ dom(new_landing_sequence) ∧ a1 ≠ a2 ⇒
26               (DIST(new_landing_sequence(a1) ↦ new_landing_sequence(a2))
27               ≥ AIRCRAFT_SEPARATION_MIN)
28     with
29       @wit1,1 newScheduledAirplanes = dom(new_landing_sequence)
30     then
31       @act1,1 landing_sequence ≔ new_landing_sequence
32   end
33 end
```

Conclusion

- VDD
  - Structuring the creation of validatable specifications
  - Structured specification
  - Structured development
  - Focus on validatable specification

- Problem frames
  - Structure refinement
  - Communicate specification structure

- VOs
  - Coverage, Maintenance, Traceability