5th UCAAT User Conference on Advanced Automated Testing

ETSI

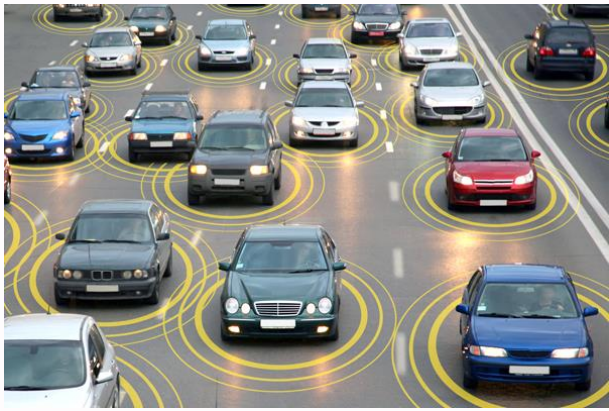
TESTING NON-FUNCTIONAL QUALITY CHARACTERISTICS OF CYBER-PHYSICAL SYSTEMS

Presented by Martin A. Schneider (Fraunhofer FOKUS)

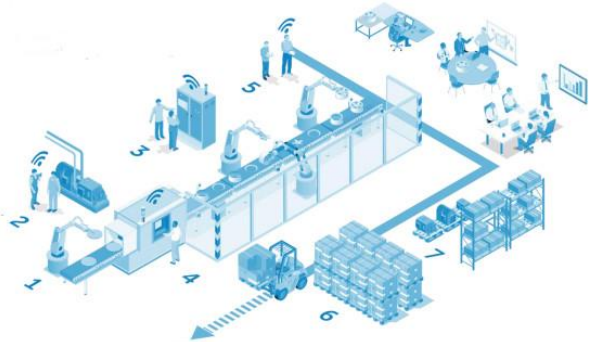
Fraunhofer FOKUS

Cyber-Physical Systems and Non-Functional Quality Characteristics

- CPS interact with their physical environment
  - through sensors, actuators and a communication infrastructure
  - examples of CPS
    - sensor systems
    - connected cars and automated driving
    - factory of the future
    - medical devices
- **reliability, safety and security are critical quality characteristics of CPS**



Jackie Roth: Connected car market gains traction. Canadian Auto Dealer (2016). Available at: https://canadianautodealer.ca/2016/02/connected-car-market-gains-traction/

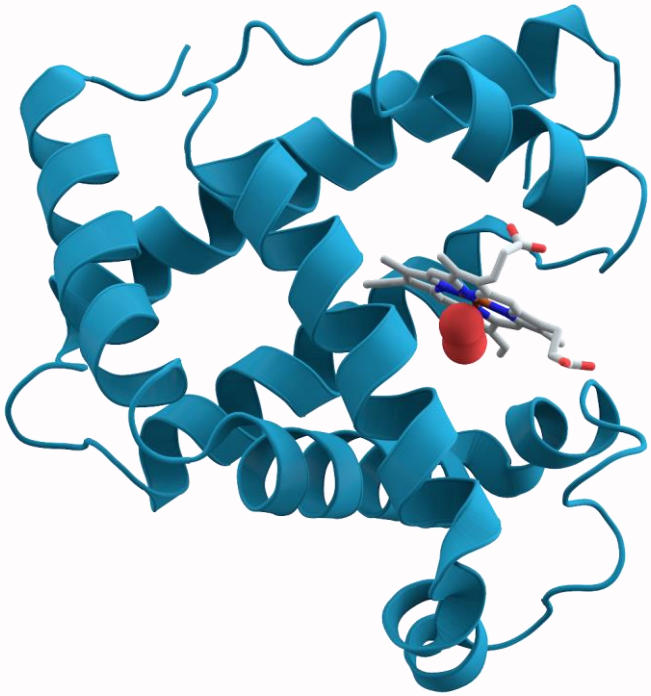


Alun Williams: Comment: Five steps to enabling a data-driven, smart factory. ElectronicsWeekly.com (2017). Available at: https://www.electronicsweekly.com/news/comment-five-steps-enabling-data-driven-smart-factory-2017-02/

CPS may be unreliable

- **undesired behaviour** of a CPS is observed at runtime
  - due to uncertainty in the *digital x physical* environment

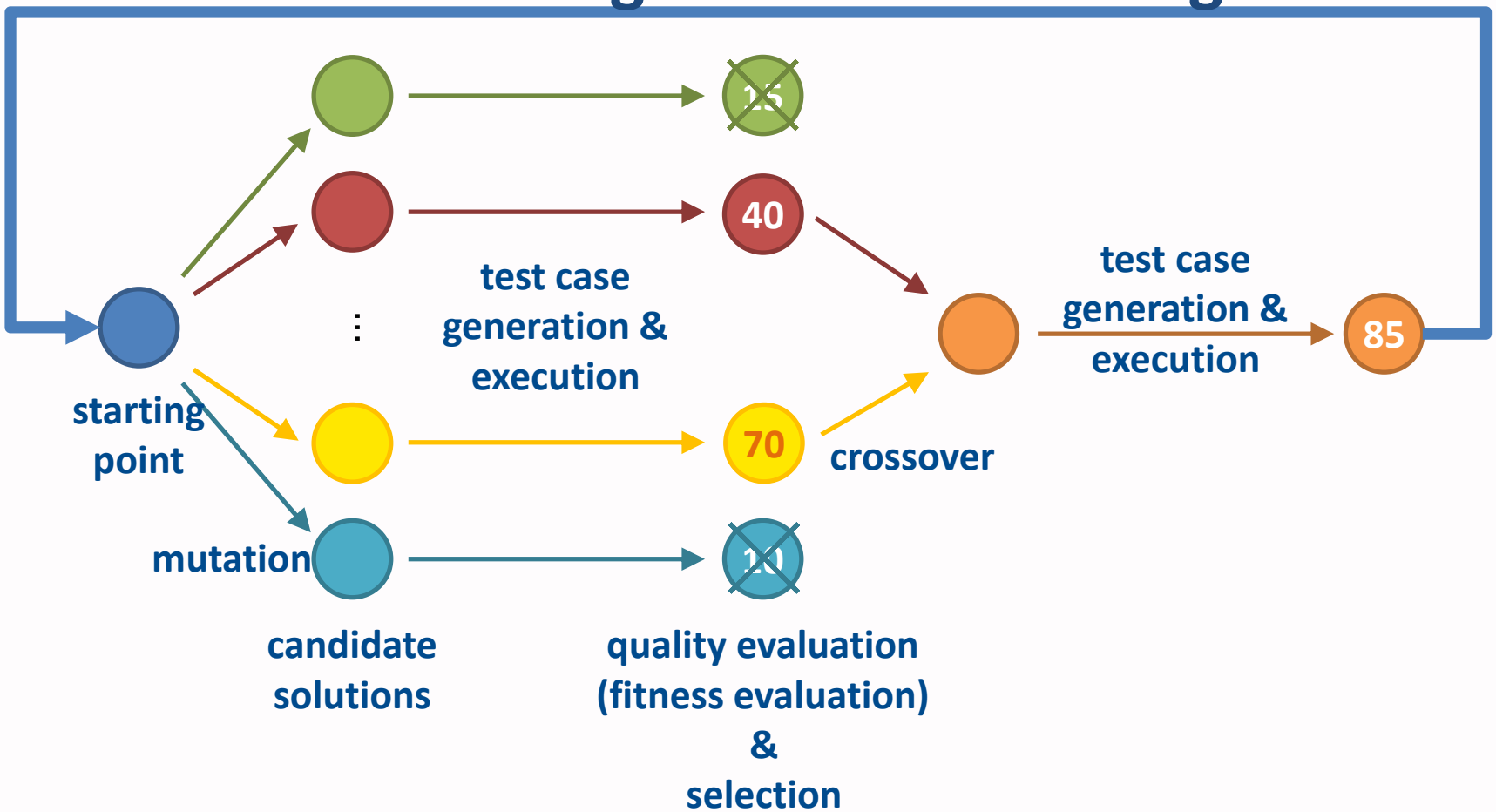
- **Challenge**
  - How to find find such scenarios **efficiently** in the **infinite and complex space of the scenarios**?
- **Solution**
  - Search algorithms



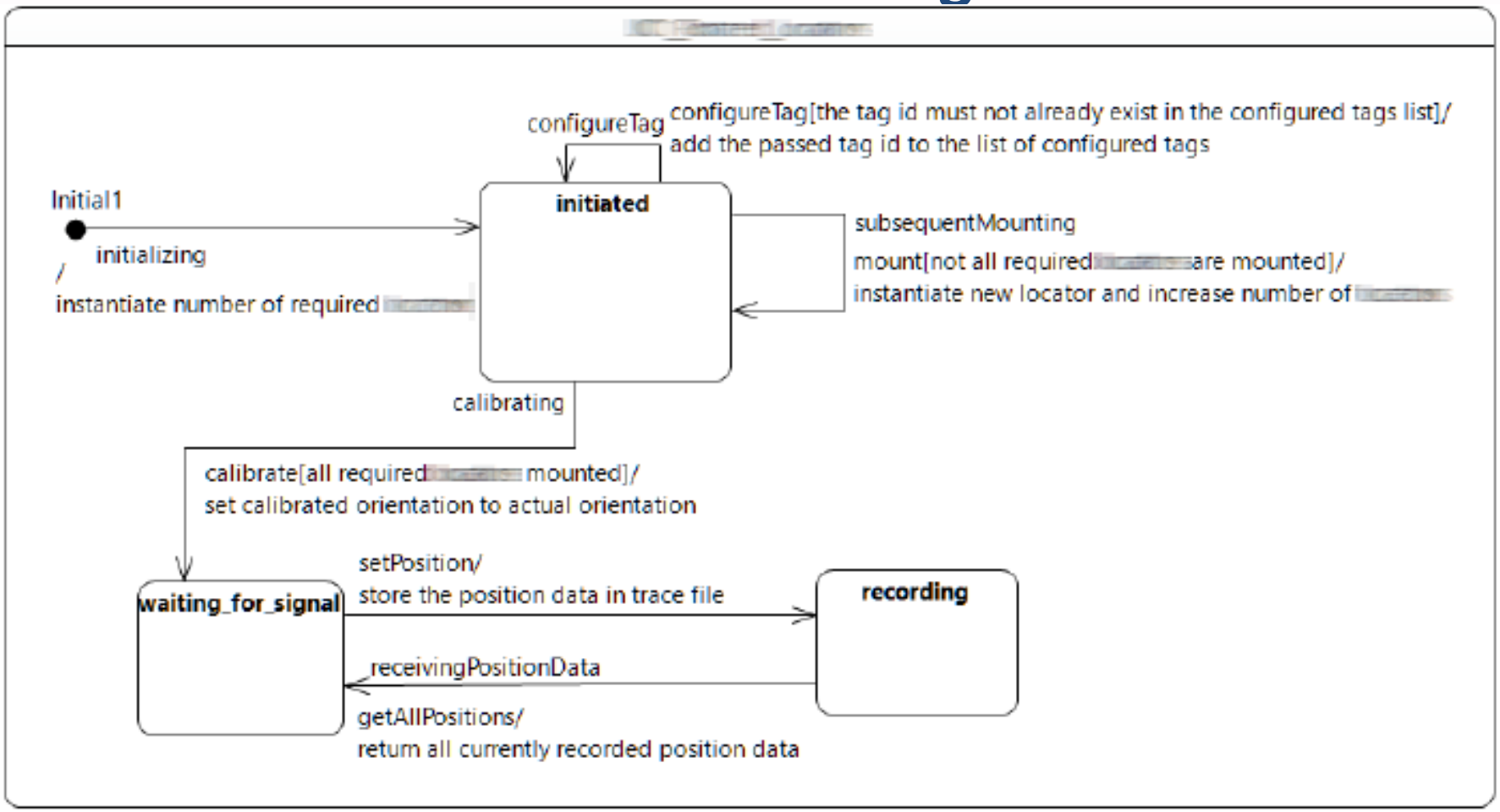
Search Algorithms

- search algorithms do not exactly how to create a solution to a problem
  - derive candidate solutions
  - assess them
- plenty of algorithms
  - genetic algorithm: mutation, crossover
- need to be tailored to the concrete problem
- search-based testing

Search-based Testing with a Genetic Algorithm



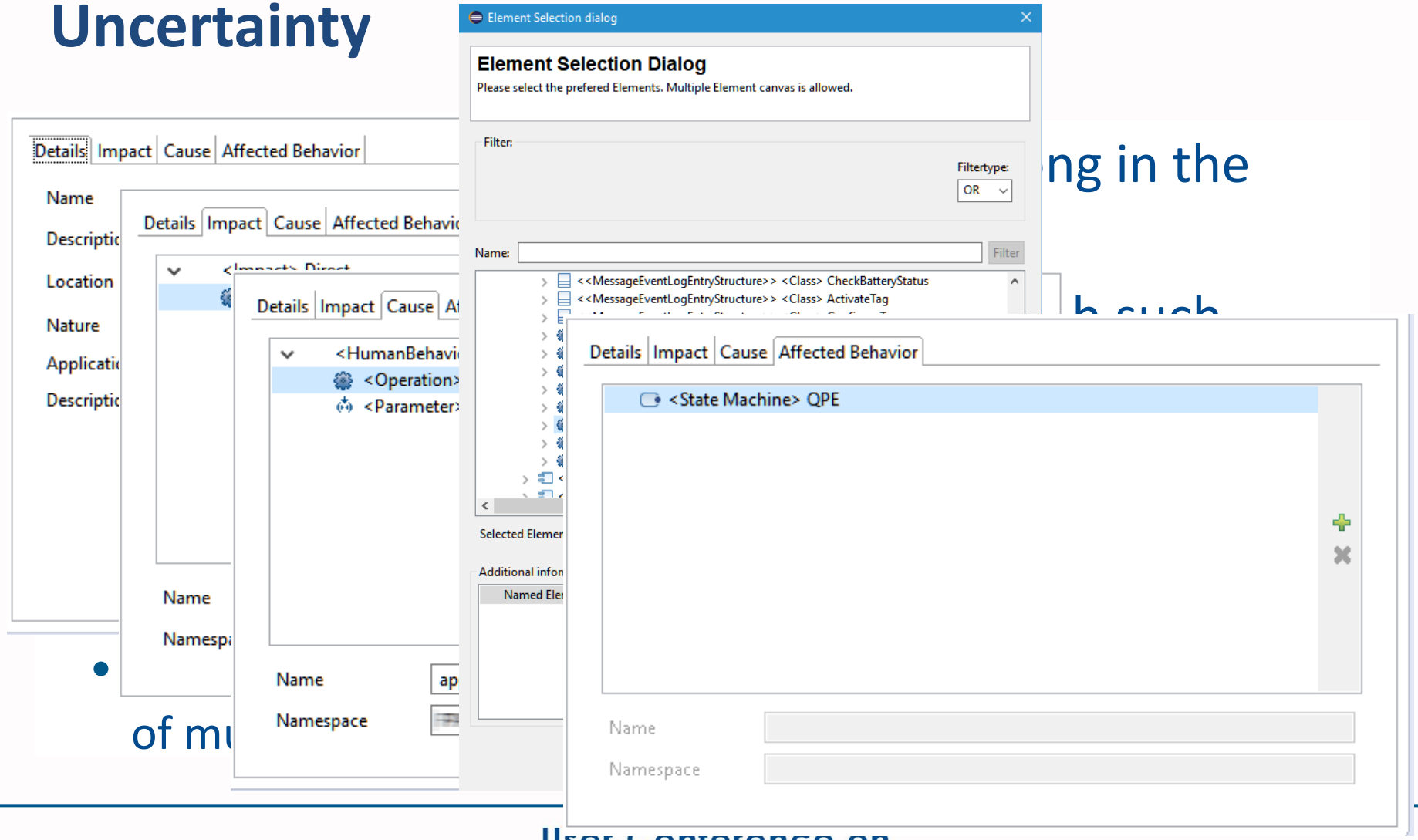
Functional Models as a Starting Point



Mutation Operator	Description	Constraints/Comments
Add Transition	Adds a new transition by duplicating an existing one and setting a new source and target state.	
Remove Transition	Completely removes the transition.	Transitions having an initial state as source or a final node as target must not be removed. Equivalent to 'Change Guard: replace expression with false'.
Remove Transition (with State Merge)	Completely removes the transition. Merges the source and target state if the removed transition is the only one connecting them (optional: with the same direction). This avoid mutilated state machines which inhibit generating test cases.	Transitions having an initial state as source or a final node as target must not be removed. Equivalent to 'Change Guard: replace expression with false'.
Reverse Transition	Swaps source and target of the transition.	Transitions having an initial state as source or a final node as target must not be reversed. Optional: Transitions being the only one that connect source and target state must not be removed (optional: with the same direction). This avoid mutilated state machines which inhibit generating test cases.
Change Source/Target	Move the source or the target of the transition to any other state.	In case the target state of the transition is changed, the target must not be the initial state. In case the source state of the transition is changed, the source must not be the final node.

Mutation Operator	Description	Constraints/Comments
Remove Trigger	Transforms the transition to a completion transition.	
Remove Guard	Removes the guard of a transition completely.	Equivalent to 'Change Guard: replace expression with true'
Remove Effect	Removes the effect of a transition completely.	
Change Trigger Operation	Changes the operation to another one of the same interface of the original operation.	
Change Guard/ Change Effect	- replace expression with true/false - negate expression - replace subexpression with true/false - negate subexpression - change logical operator - change relational operator - change arithmetic operator - change set operator - change quantifier - replace operand guard/effect mutation operators - remove statement - move statement - fix parameter/property of a called method or sent signal - change called method or sent signal - change operator - fix operand (replace it with a literal) - change operand (replace with variable, call parameter or signal property of the same type) - replace result: replace right-hand-side (RHS) expression with default value of left-hand-side (LHS)	Guards and effects are written in C#.

Uncertainty

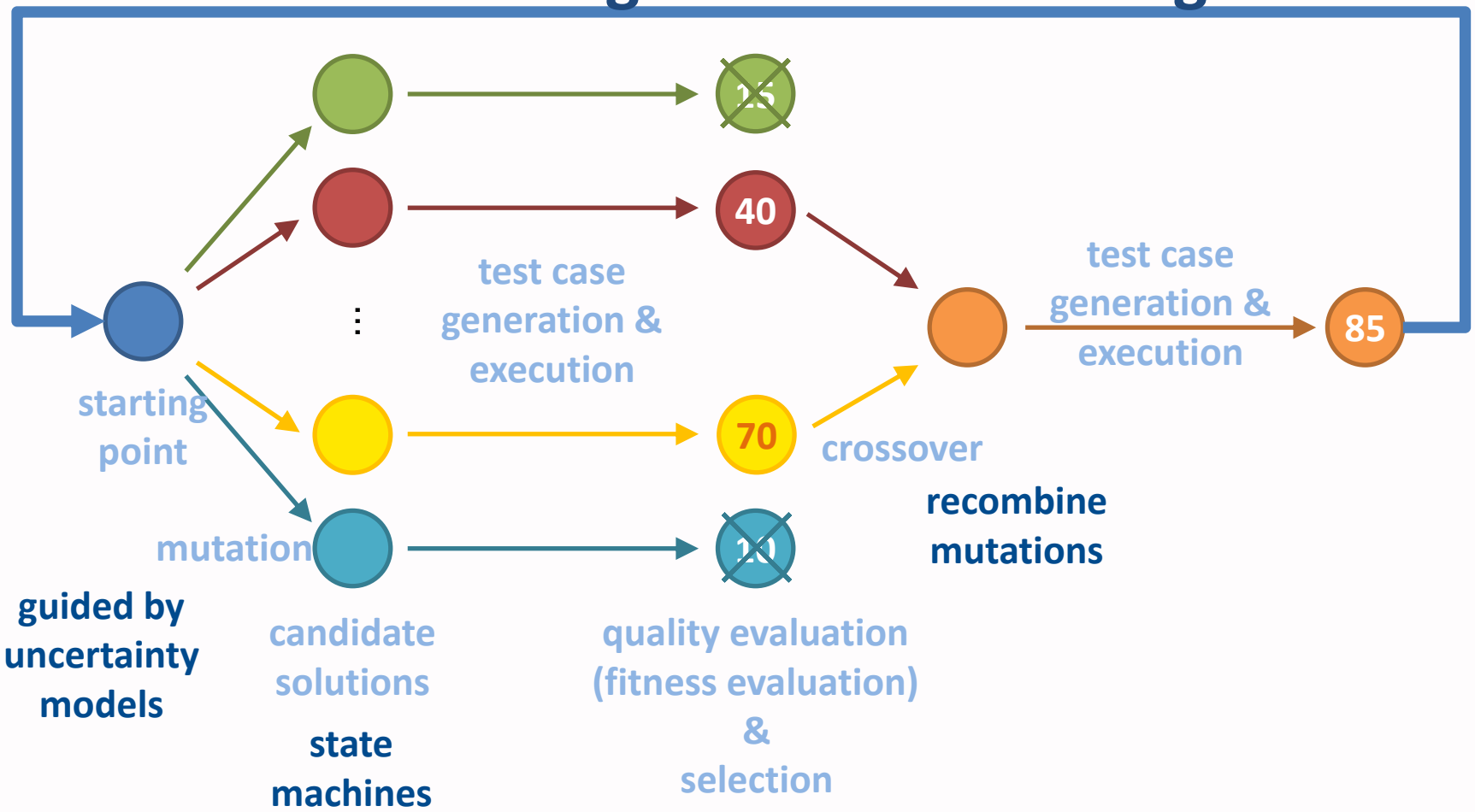


ng in the

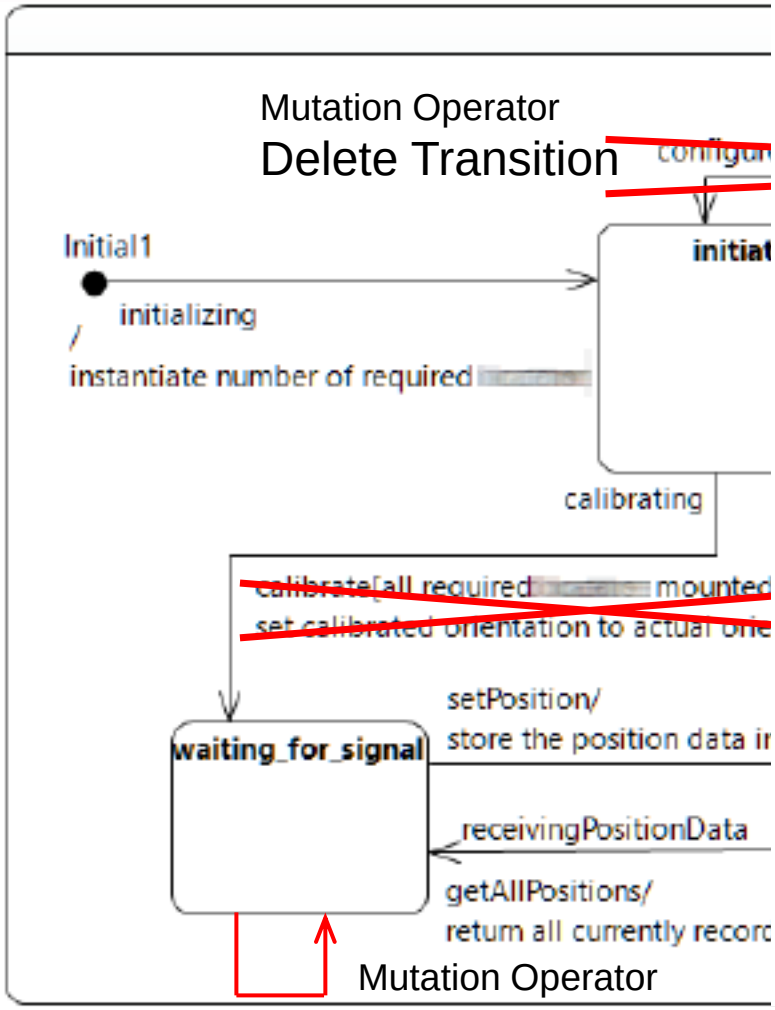
h such

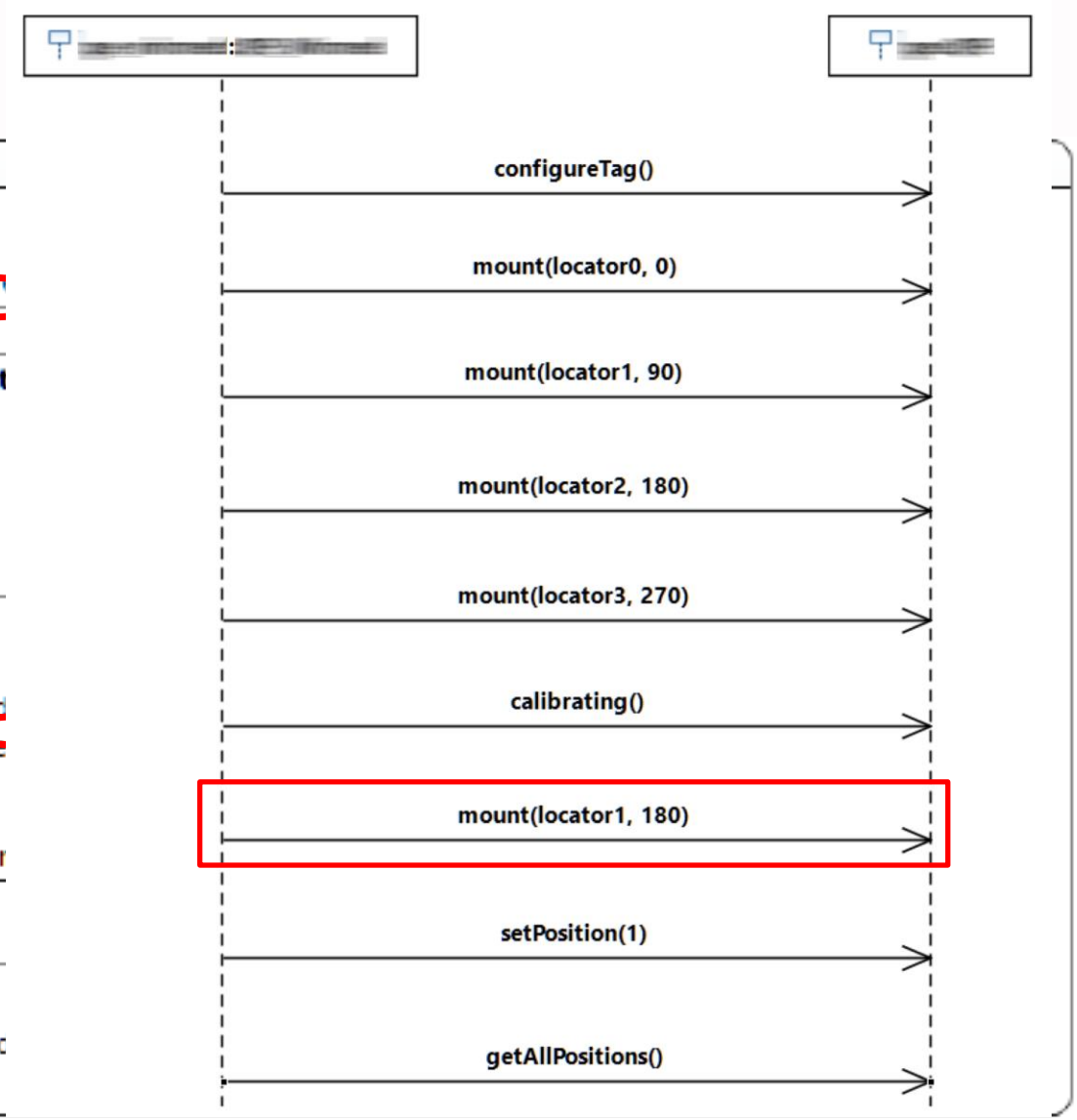
- of mu

Search-based Testing with a Genetic Algorithm

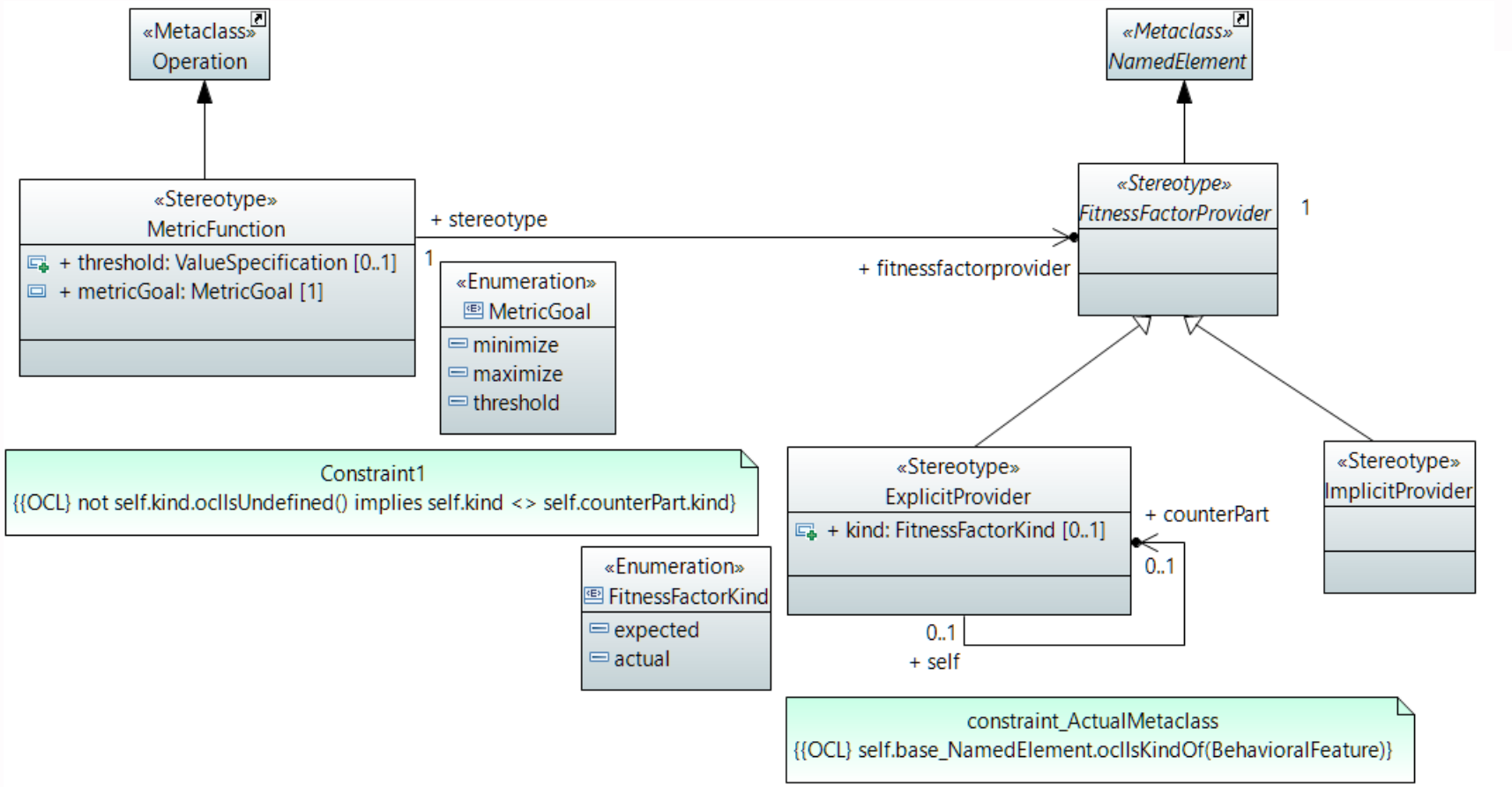


Example





Quality Evaluation: Fitness Factors



Conclusions

- small effort for testers
  - start from functional models (state machines)
  - add uncertainty models
- reduction of search space
  - search is guided by uncertainty models
- configurable
  - by uncertainty models
  - and model-based fitness factors

Conclusions

Thanks for your attention!

Questions!?

Martin Schneider
Senior Scientist
Fraunhofer FOKUS
martin.schneider@fokus.fraunhofer.de