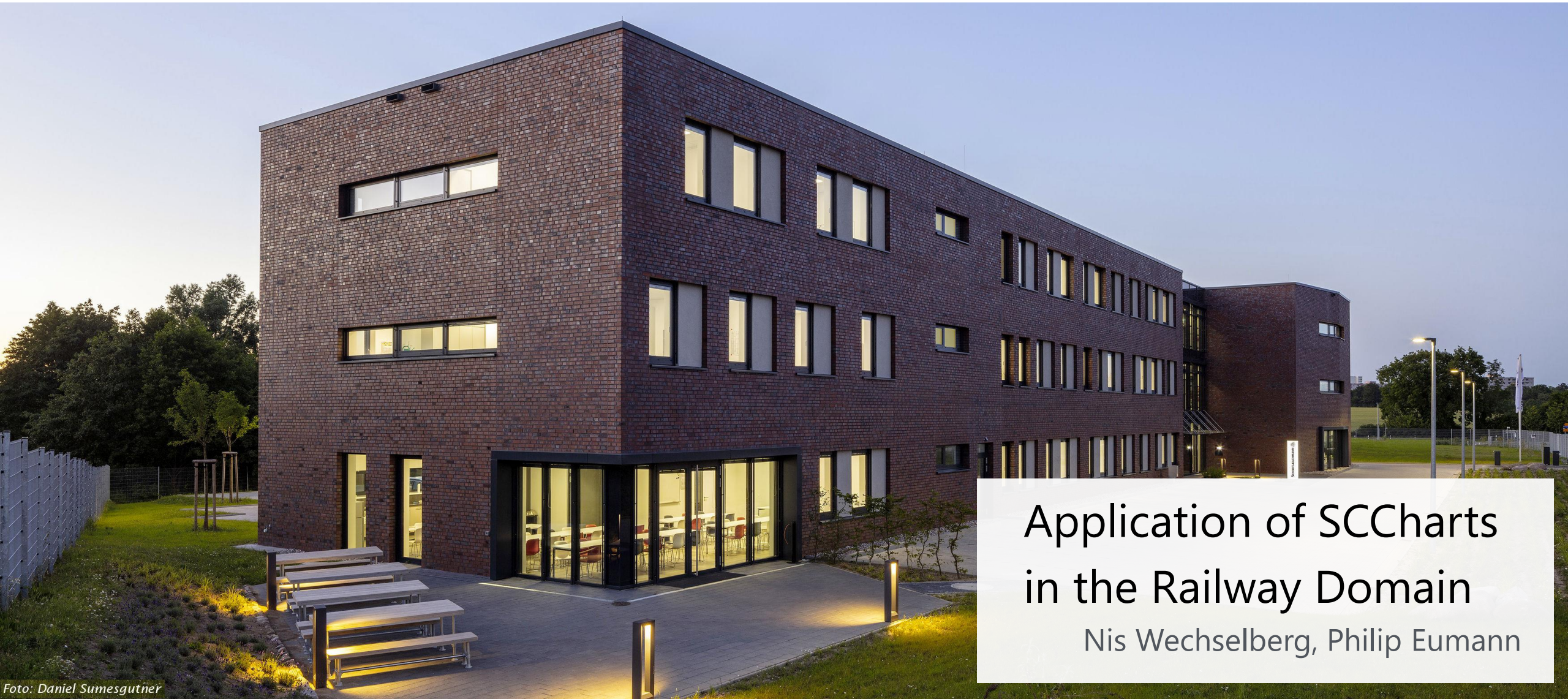


Welcome to

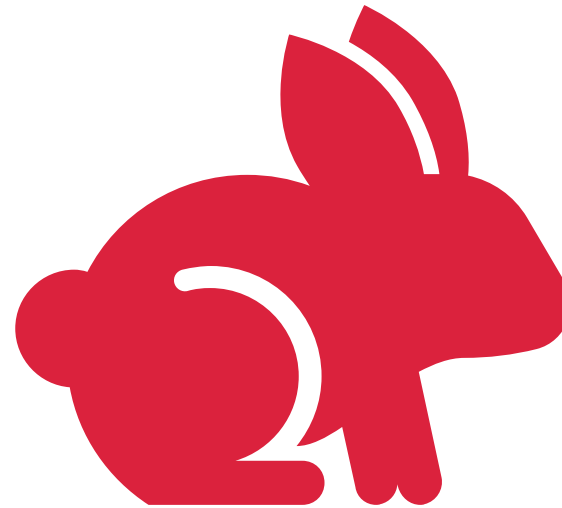
**SCHEIDT&BACHMANN**



## Application of SCCharts in the Railway Domain

Nis Wechselberg, Philip Eumann

# / Two Main Challenges in the Railway Sector



# / Safety?



by Nxr-at – own work, CC BY-SA 4.0,  
<https://commons.wikimedia.org/w/index.php?curid=74649486>



by Alupus – own work, CC BY-SA 3.0,  
<https://commons.wikimedia.org/w/index.php?curid=24722748>

# / Safety?



by I, Sese Ingolstadt, CC BY-SA 2.5, <https://commons.wikimedia.org/w/index.php?curid=1877619>

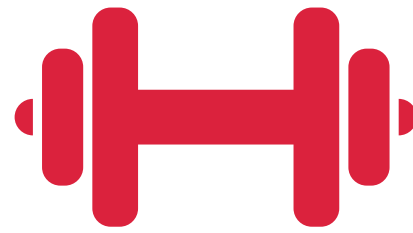
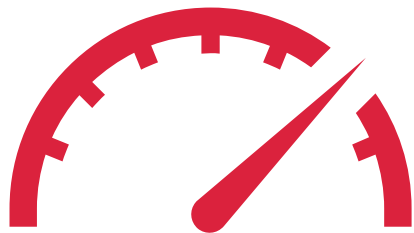
# / Safety



# / Safety



# / Trains cannot be driven on sight



# / Two Core Mechanisms

- Signals can only turn green when the track beyond them is safe
  - Signal Dependency



by Olga Ernst & Hp.Baumeler – own work, CC BY-SA 4.0,  
<https://commons.wikimedia.org/w/index.php?curid=67973231>

# / Two Core Mechanisms

- Trains can only pass green signals



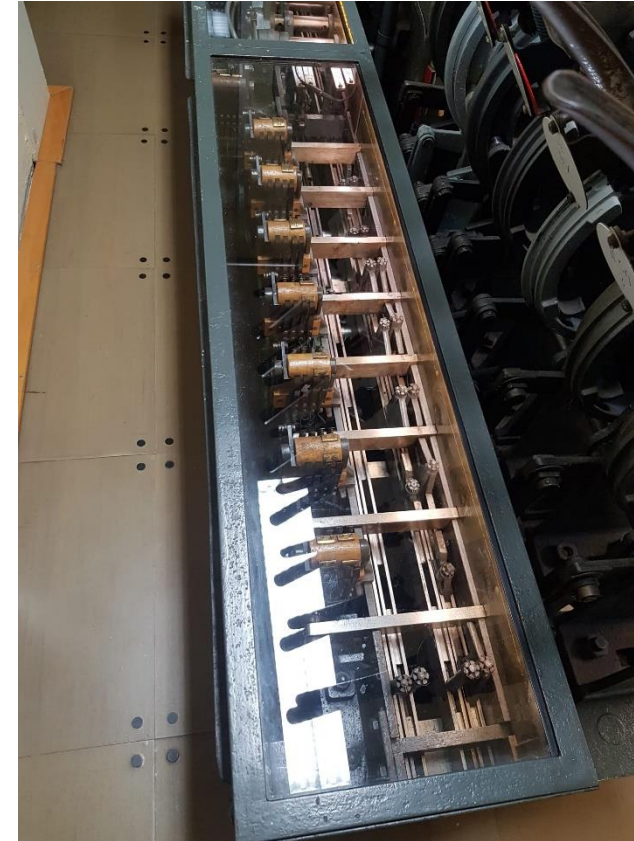
by Christian Liebscher – own work, CC BY-SA 4.0, <https://commons.wikimedia.org/w/index.php?curid=39224581>



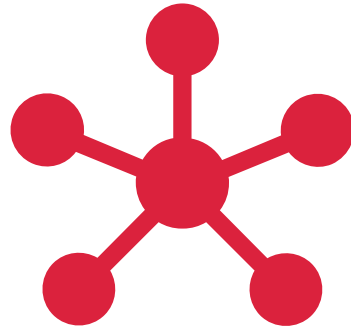
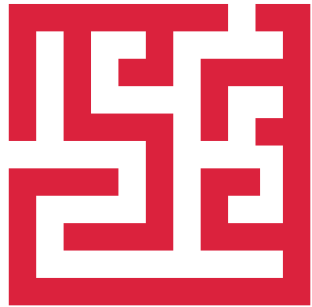
CC BY-SA 3.0, <https://commons.wikimedia.org/w/index.php?curid=519699>

# / Safety Requirements

- Supervision by Federal Railway Agency (Eisenbahnbundesamt, EBA)
- Mechanical components can easily be verified
- Local changes do not affect the entire system
- Changes and updates can be carried out by third-party technicians



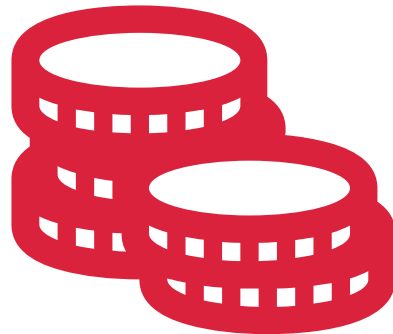
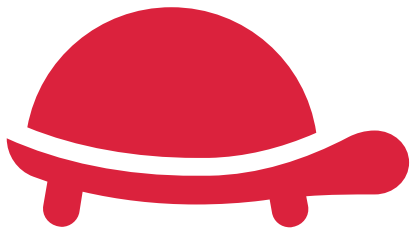
# / The Issue with Software



# / Standard Development Process



- Highest **Safety and Integrity Level 4** for electronic interlockings



# / Efficiency



by 2013-06-08\_Highflyer\_HP\_L4729.JPG: Alchemist-hp (talk)derivative work: Hic et nunc – This file was derived from the following work: 2013-06-08 Highflyer HP L4729.JPG.; CC BY-SA 3.0, <https://commons.wikimedia.org/w/index.php?curid=27038423>



by Hbf878, OpenStreetMap contributors – own work, using OpenStreetMap data, CC BY-SA 2.0, <https://commons.wikimedia.org/w/index.php?curid=89862775>

# / Efficiency



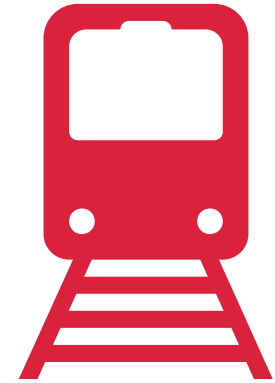
[https://www.swr.de/swraktuell/baden-wuerttemberg/1690538908470%2Cbahnbaustelle-symbolbild-100~\\_v-16x7@2dL\\_-594eb175bf96444e7f86c89c3d9f78feed295e4a.jpg](https://www.swr.de/swraktuell/baden-wuerttemberg/1690538908470%2Cbahnbaustelle-symbolbild-100~_v-16x7@2dL_-594eb175bf96444e7f86c89c3d9f78feed295e4a.jpg)

# / Command and Control Systems

- Ensure efficient traffic flow
- High levels of automation
- Network-wide coordination



# / Command and Control Systems

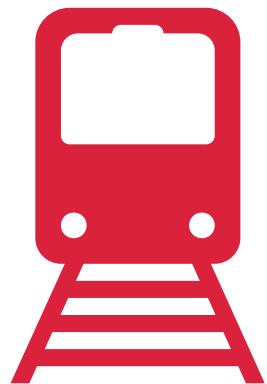
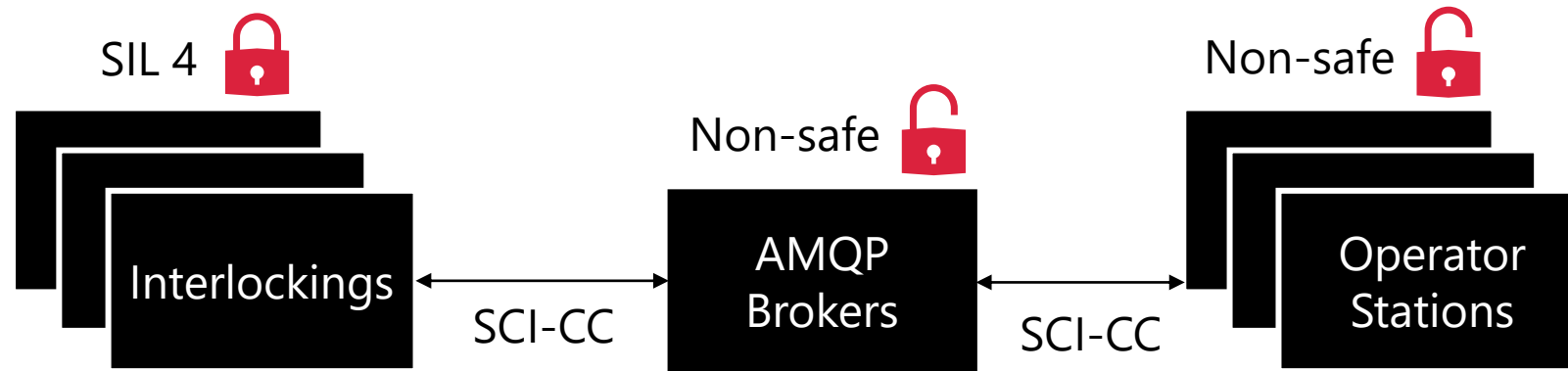


- Complex distributed systems
- Operators can override safety mechanisms

# / Command and Control Systems today

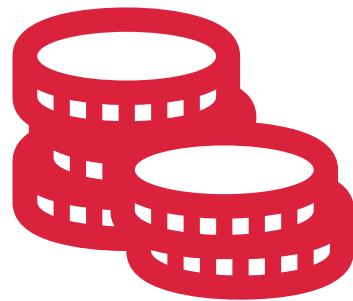
- Operator station is safety-critical system (SIL 2)
- to keep complexity low, other applications run on separate machines
- multiple sign-ins
- multiple input devices
- proprietary communication interfaces
- linked safety evaluation

# / Integrated Command and Control System



# / Integrated Command and Control System

- Standard Communication Interface – Command and Control (SCI-CC) via AMQP
- All operator-side equipment is COTS
- Nationwide network of AMQP brokers
- Cross-compatibility between manufacturers
- All systems integrated on one operator station with single sign-on
- Safety of inputs and outputs is guaranteed by procedures instead



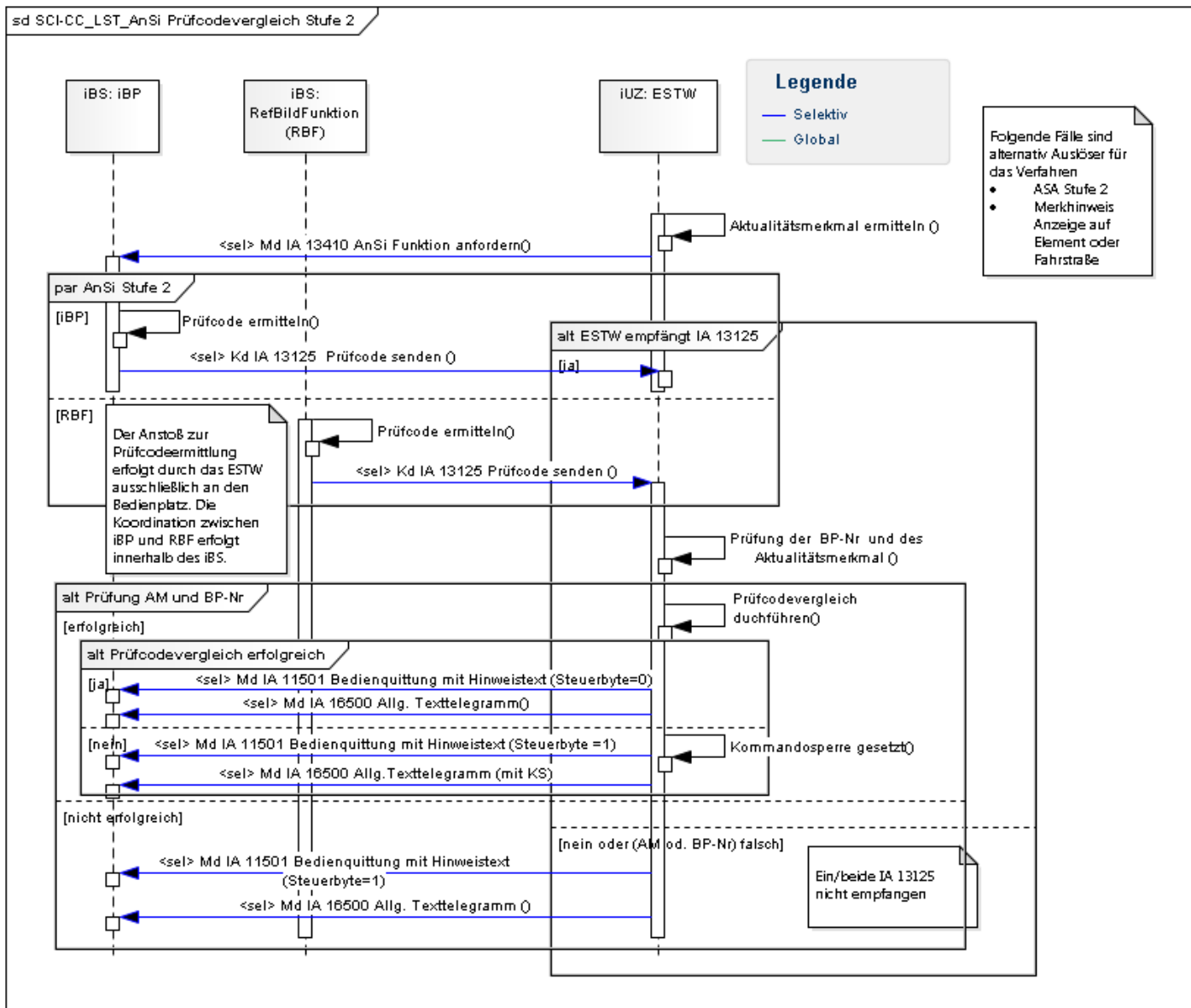
# / Development of complex systems for railways

```
1 private StateMachine<UserInteractionStateDbag, UserInteractionEventDbag> createStateMachine() {
2     return new StateMachineBuilder<UserInteractionStateDbag, UserInteractionEventDbag>(
3         UserInteractionStateDbag.GRUNDZUSTAND).onEnter(UserInteractionStateDbag.GRUNDZUSTAND, () -> {})
4         // Self
5         .addTransition(UserInteractionStateDbag.GRUNDZUSTAND, UserInteractionEventDbag.MTL_D,
6             this::isFreeAreaEvent, UserInteractionStateDbag.GRUNDZUSTAND, () -> {
7                 LOG.info("Ignoring left mouse click at free area."); //NON-NLS-1$
8             }) //
9         .addTransition(UserInteractionStateDbag.GRUNDZUSTAND, UserInteractionEventDbag.MTL_D, () -> {
10             return !isShuntingRouteStart() && !isTrainrouteStart() && isUiElementEvent();
11         }, UserInteractionStateDbag.GRUNDZUSTAND, () -> {
12             LOG.info("Ignoring left mouse click at uiElement {}.", eventUiElement.getId()); //NON-NLS-1$
13         }) //
14         .addTransition(UserInteractionStateDbag.GRUNDZUSTAND, UserInteractionEventDbag.MTR_D, //
15             this::isUiElementEventInsensitive, UserInteractionStateDbag.GRUNDZUSTAND, () -> {
16                 LOG.info("Ignoring right mouse click at uiElement {}.", eventUiElement.getId()); //NON-NLS-1$
17             }) //
18         // DiB_01
19         .addTransition(UserInteractionStateDbag.GRUNDZUSTAND, UserInteractionEventDbag.MTL_D,
20             this::isTrainrouteStart, UserInteractionStateDbag.FS_START_SELEKTIERT, () -> {
21                 startRouteSelection(FahrstrassenTyp.ZUG, VQ_TRAINROUTE_START_SELECTED);
22             }) //
23         // DiB_02
```

# / What are our requirements from the customer?

415.906 4-87	Muss	Telegrammtabelle IA 13000							iUZ:ESTW iUZ:ETCS- Z iBS:iBP	-	CR2292		
		Byte-Nr	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2				Bit 1	Bit 0
		-	Informationsart 13000 ( 32 C8H )										
		H	Bedienplatznummer										
		H+8	Aktualitätsmerkmal										
		H+14	ErgBedMit (Ergebnis Bedienermitwirkung)										
		H+15	PrüfCodeBP										
		H+19	Transaktionsnummer ( Bedienung )										
		H+20	Reserve ( nicht definiert )										
H+25	Sicherungsanhang												

# / What are our requirements from the customer?



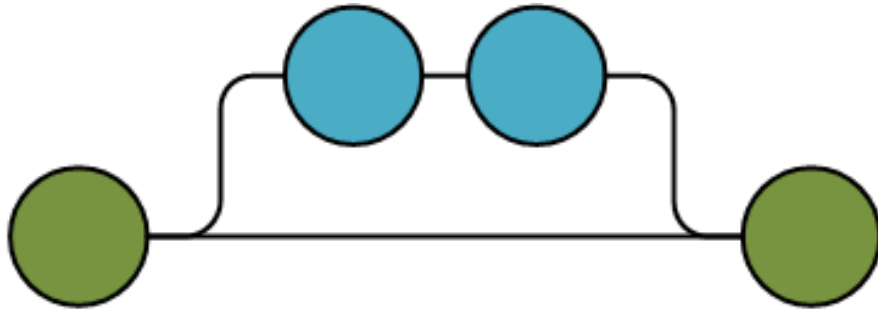
# / How do we adapt the requirements?

- Use generative tools to create common libraries for communication data structures
- Generating code from sequence diagrams not really a thing
  - (As far as I know)
- One sequence diagram usually contains more than one system that is part of the communication
  - Not all of these systems are developed by us
  - Not all of the systems we develop are developed by the same dev team
- But: The sequence diagrams „hide“ one or more state diagrams that we care about



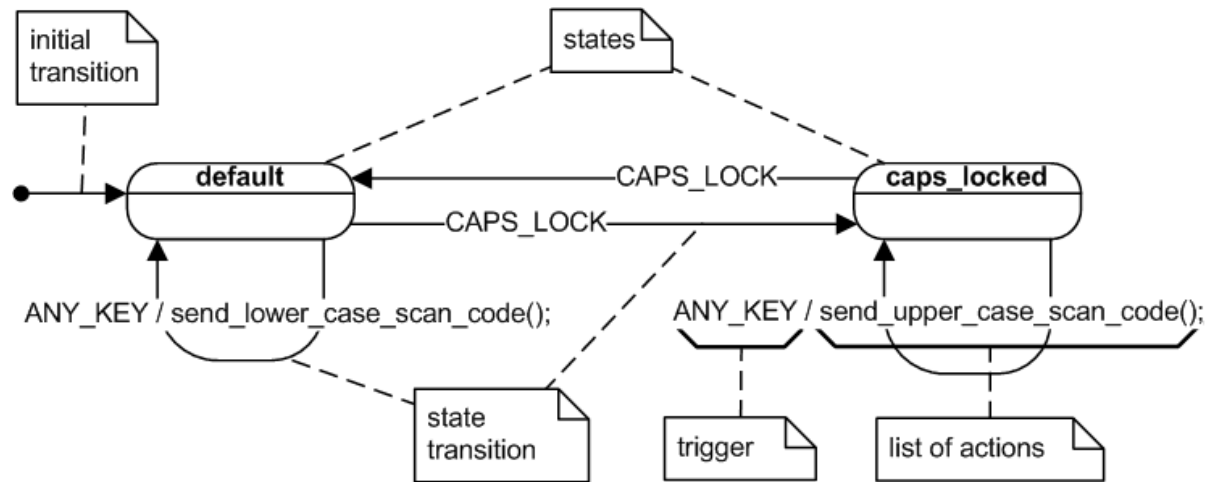
Find a state modelling tool suitable  
for our usage in railway applications

# / Suitable?



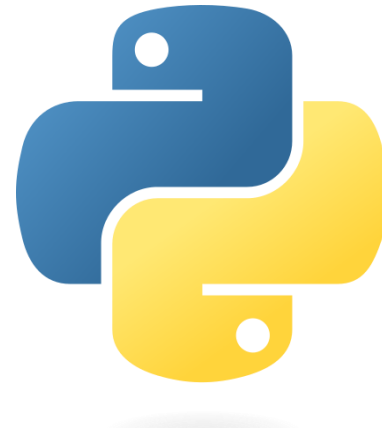
- Work together with commonly used version control systems
  - Text based file format
  - Limited complexity in storage format

# / Suitable?



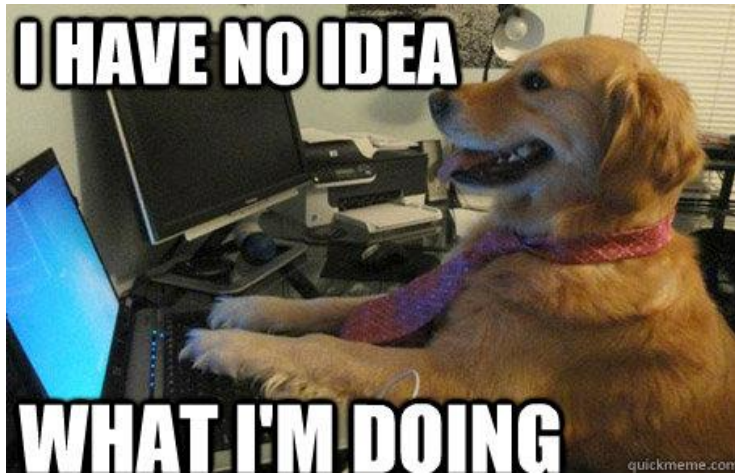
- Some form of visualization for the state machine
  - Bonus points if the visualization actually has some meaning for non-developers (like domain experts)

# / Suitable?



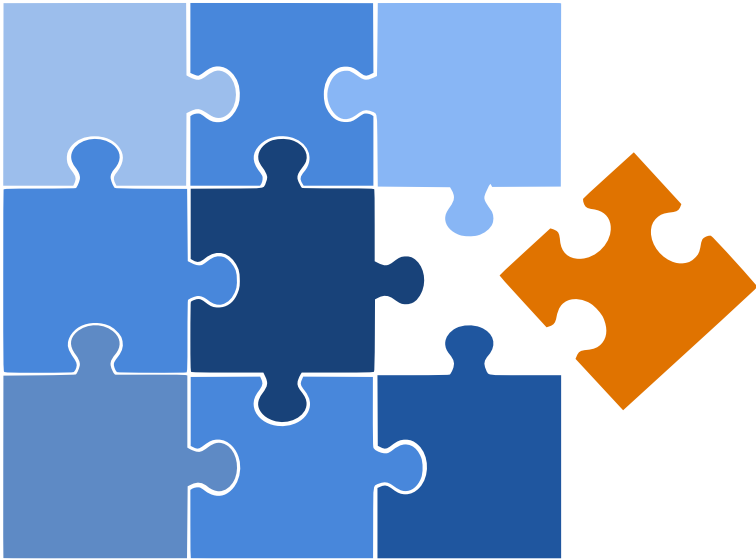
- Support for multiple target languages
  - Java
  - C++
  - Python (bonus)

# / Suitable?



- Usable for software developers used to classic, imperative programming
  - Synchronous languages can be a bit intimidating at times
  - I don't want to teach a 3-week course on special languages for every new dev

# / Suitable?



- Adaptable/Integratable in existing systems
  - Support for various existing systems and toolchains

# / Suitable?



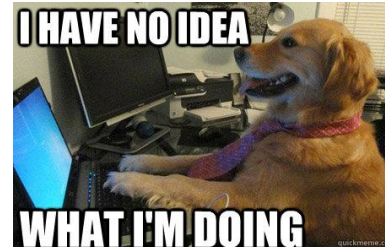
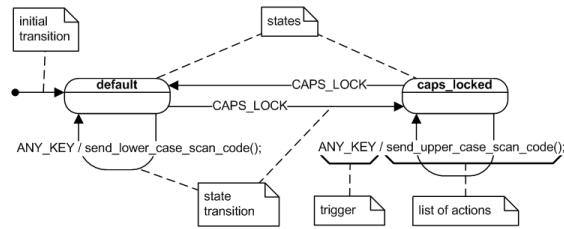
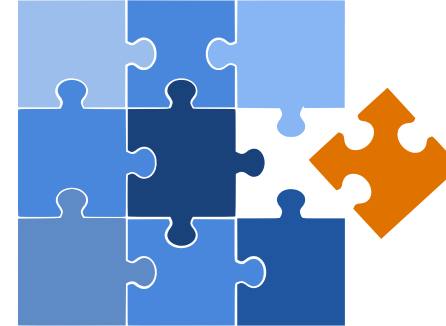
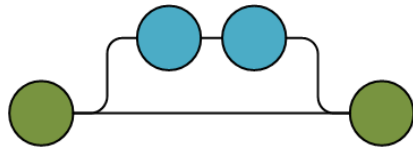
- Usable in certified safety-critical systems
  - Either complete certified language/semantics/compilation
  - Or „clean“ generated code, that can be certified itself

# / Suitable?



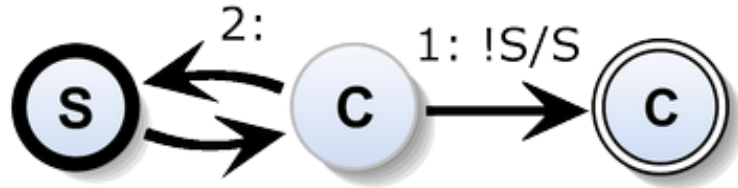
- Long-term availability and usability
  - Our systems need to be supported for multiple decades
  - Common support time is 20-30 years, on occasions up to 40 years

# / Suitable?



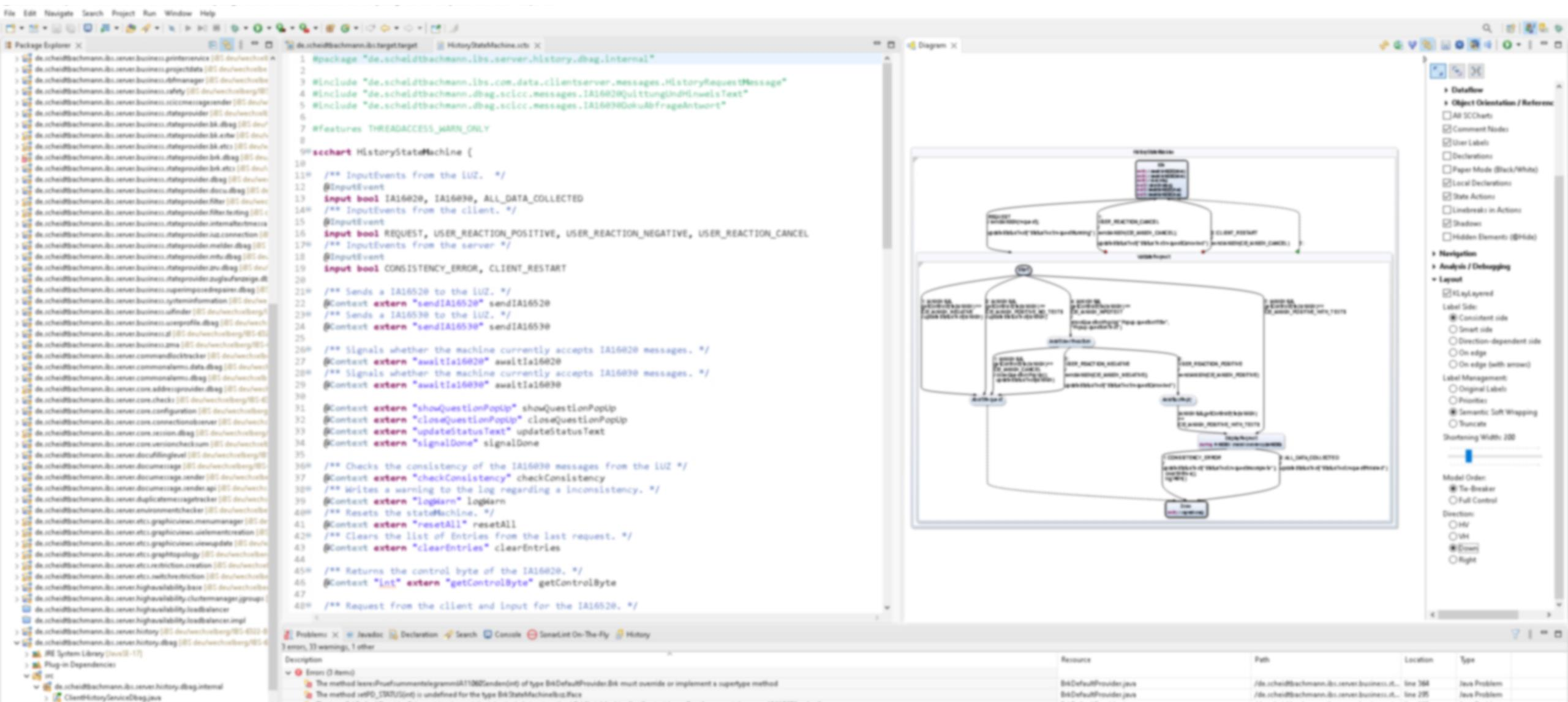
- Given all these constraints there are two options
  - Exceptionally well adaptable, commercial solution
  - Free and open source tool that we can adapt (and/or keep running) ourselves

# / (Mostly) Suitable!



- Sequential constructiveness reduces needed training for new devs
- Existing code generation for Java and C
  - Is the code up-to-spec to be used in safety-critical systems?
- Flexible execution model
- Plain text DSL easy to read
  - Caveat: Getting all the details of the grammar right needs some time
- Flexible transient visualizations

# / Embedding SCCharts in our development



The image shows a screenshot of an IDE (Eclipse) with two main windows. The left window displays the source code for an SCChart named `HistoryStateMachine`. The code includes package declarations, imports, and a list of input events. The right window displays the generated state machine diagram, which is a hierarchical state transition graph. The diagram shows a root state `HistoryStateMachine` with several transitions leading to sub-states like `waitIA16520`, `waitIA16530`, and `waitIA16520IA16530`. The diagram also shows transitions for events like `sendIA16520` and `sendIA16530`.

```

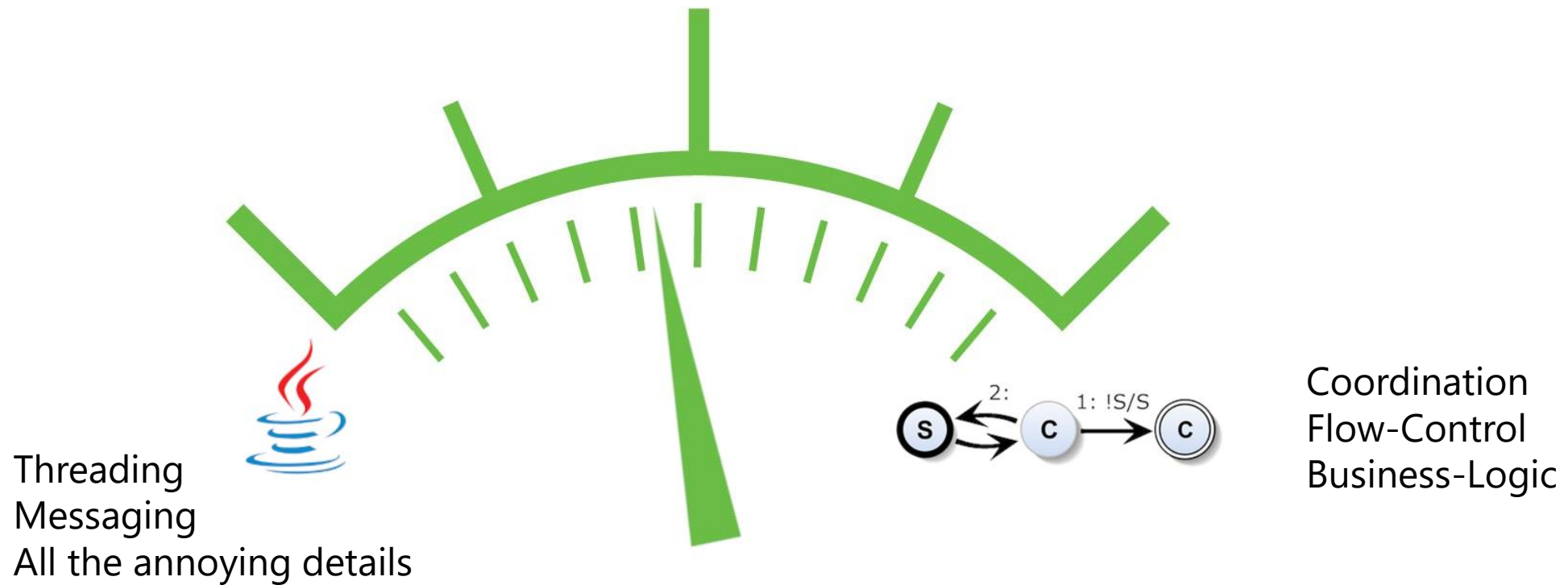
1 #package "de.scheidtbachmann.libs.server.history.dbag.internal"
2
3 #include "de.scheidtbachmann.libs.com.data.clientserver.messages.HistoryRequestMessage"
4 #include "de.scheidtbachmann.dbag.scirc.messages.IA16520QuittungUndHinweisText"
5 #include "de.scheidtbachmann.dbag.scirc.messages.IA16530DokuAbfrageAntwort"
6
7 #features THREADACCESS_WARN_ONLY
8
9 scchart HistoryStateMachine {
10
11  /** InputEvents from the IUZ. */
12  @InputEvent
13  input bool IA16520, IA16530, ALL_DATA_COLLECTED
14  /** InputEvents from the client. */
15  @InputEvent
16  input bool REQUEST, USER_REACTION_POSITIVE, USER_REACTION_NEGATIVE, USER_REACTION_CANCEL
17  /** InputEvents from the server */
18  @InputEvent
19  input bool CONSISTENCY_ERROR, CLIENT_RESTART
20
21  /** Sends a IA16520 to the IUZ. */
22  @Context extern "sendIA16520" sendIA16520
23  /** Sends a IA16530 to the IUZ. */
24  @Context extern "sendIA16530" sendIA16530
25
26  /** Signals whether the machine currently accepts IA16520 messages. */
27  @Context extern "swaitIA16520" swaitIA16520
28  /** Signals whether the machine currently accepts IA16530 messages. */
29  @Context extern "swaitIA16530" swaitIA16530
30
31  @Context extern "showQuestionPopUp" showQuestionPopUp
32  @Context extern "closeQuestionPopUp" closeQuestionPopUp
33  @Context extern "updateStatusText" updateStatusText
34  @Context extern "signalDone" signalDone
35
36  /** Checks the consistency of the IA16530 messages from the IUZ */
37  @Context extern "checkConsistency" checkConsistency
38  /** Writes a warning to the log regarding a inconsistency. */
39  @Context extern "logWarn" logWarn
40  /** Resets the state machine. */
41  @Context extern "resetAll" resetAll
42  /** Clears the list of Entries from the last request. */
43  @Context extern "clearEntries" clearEntries
44
45  /** Returns the control byte of the IA16520. */
46  @Context "int" extern "getControlByte" getControlByte
47
48  /** Request from the client and input for the IA16520. */
  
```

The diagram on the right shows a state machine with a root state `HistoryStateMachine`. It has several transitions leading to sub-states like `waitIA16520`, `waitIA16530`, and `waitIA16520IA16530`. The diagram also shows transitions for events like `sendIA16520` and `sendIA16530`.

Resource	Path	Location	Type
B4DefaultProvider.java	/de.scheidtbachmann.libs.server.business.d...	line 184	New Problem
B4DefaultProvider.java	/de.scheidtbachmann.libs.server.business.d...	line 235	New Problem

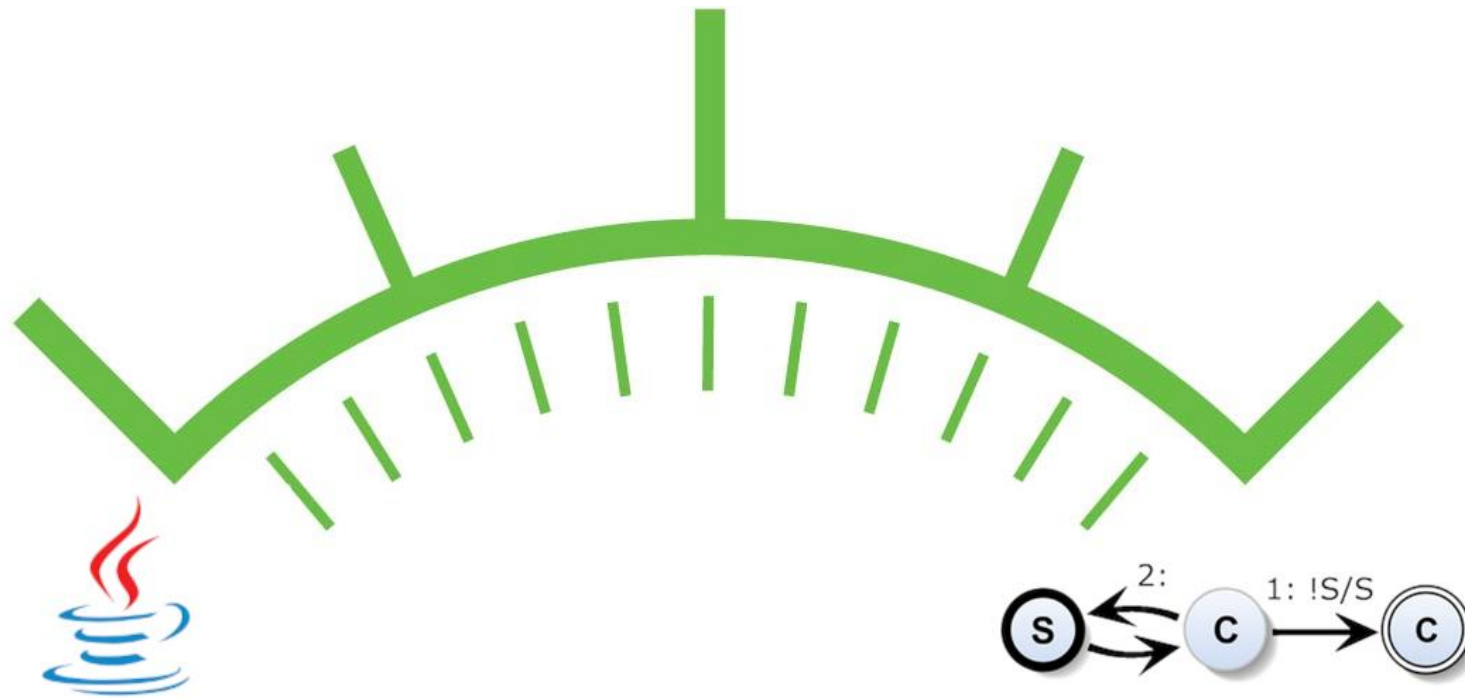
# / Task 1: Deciding on a separation of concerns

- What parts should be handled by SCCharts, what parts by the host language?



# / Subtask 1: Main Event-Loop

- Who should be responsible for the main application control?



## / Task 2: Scheduling the execution of SCCharts

- Free-running mode
  - Run the SCCharts tick loop as fast as it wants
  
- „Fixed“ time mode
  - Have a (semi-)fixed scheduling for the tick loop
  
- Event-based mode
  - Only run the tick loop when there is some actual work to do

## / Subtask 2: Handling concurrency surrounding SCCharts

- The modelled code is thread-safe, regarding concurrency/concurrent events
- However, the generated code is not entirely thread-safe
  - Running the tick() function from multiple concurrent threads causes bad things
- Our application is highly concurrent, so we need to address that
- Solution: Every SCCharts objects carries a dedicated execution thread that is triggered with the current events
- Caveat: Every access to inputs or outputs of the SCChart needs to be performed in this thread

## / Task 3: Choosing a compilation chain

- SCCharts provides multiple ways to compile the model to code
  
- Netlist-based
  - Thoroughly tested
  - Full support for all advanced features
  - But: Code not human-readable / Unusable for certification
  
- Priority-based
  - Not always stable
  - Unclear feature support (though I guess it should be feature-complete)
  - No Java target (as far as I know)

## / Task 3: Choosing a compilation chain

- We might accept a reduction in the feature set of the language if the generated code is „clean“ and human-readable
- „New“ compilation chain: „Lean“ State-based
  - Reduced feature set (especially in regards to dependent concurrent regions)
  - Mostly direct mapping of model to generated code possible
  - Generated code is WAY more verbose, but mostly human readable

# / Future Work / Options





# / Future Work – Model Checking

- Can we check the SCCharts against certain safety and/or liveness criteria?
- Currently ongoing research @RTSYS Group

[www.scheidt-bachmann-st.de](http://www.scheidt-bachmann-st.de)

**SCHEIDT&BACHMANN** 



**Thank you  
for your attention!**