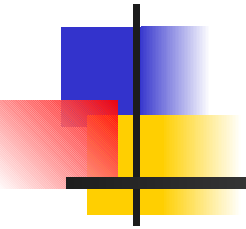
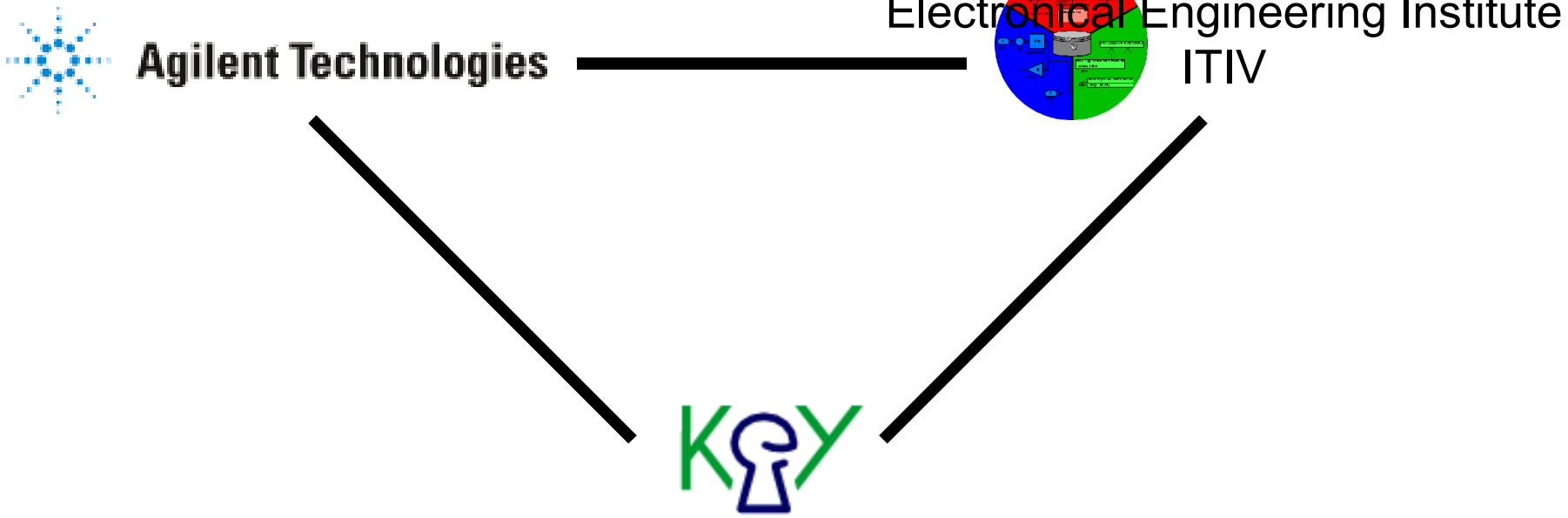


Agilent – GeneralStore a case study

Ignaz Rutter

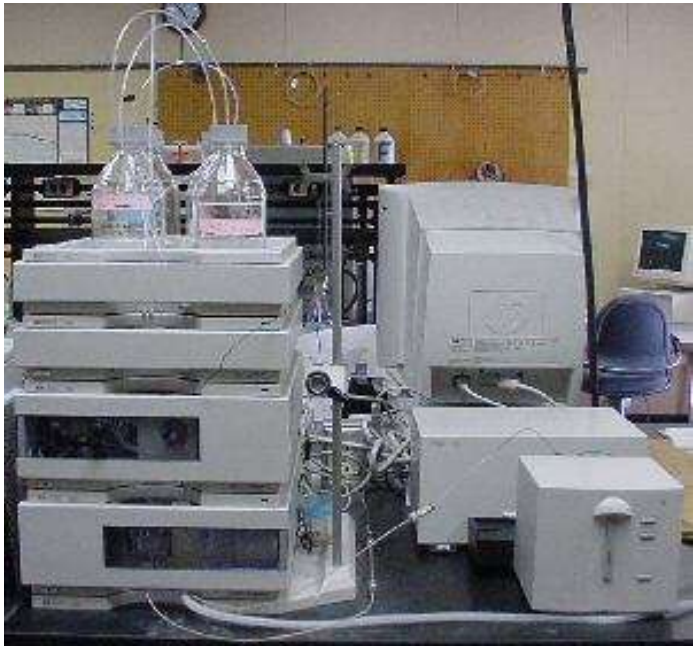
KeY workshop 2004

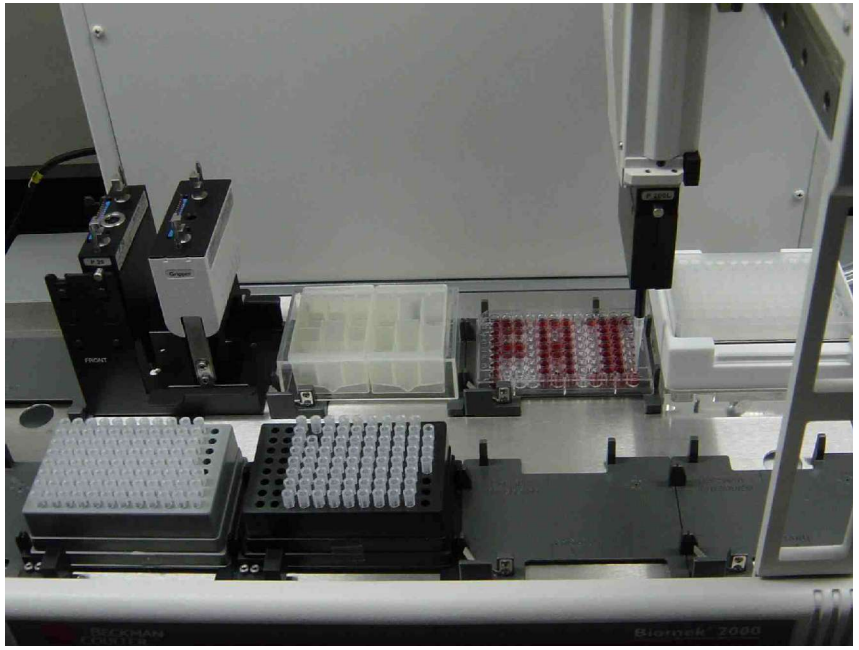
Who is involved



Case study

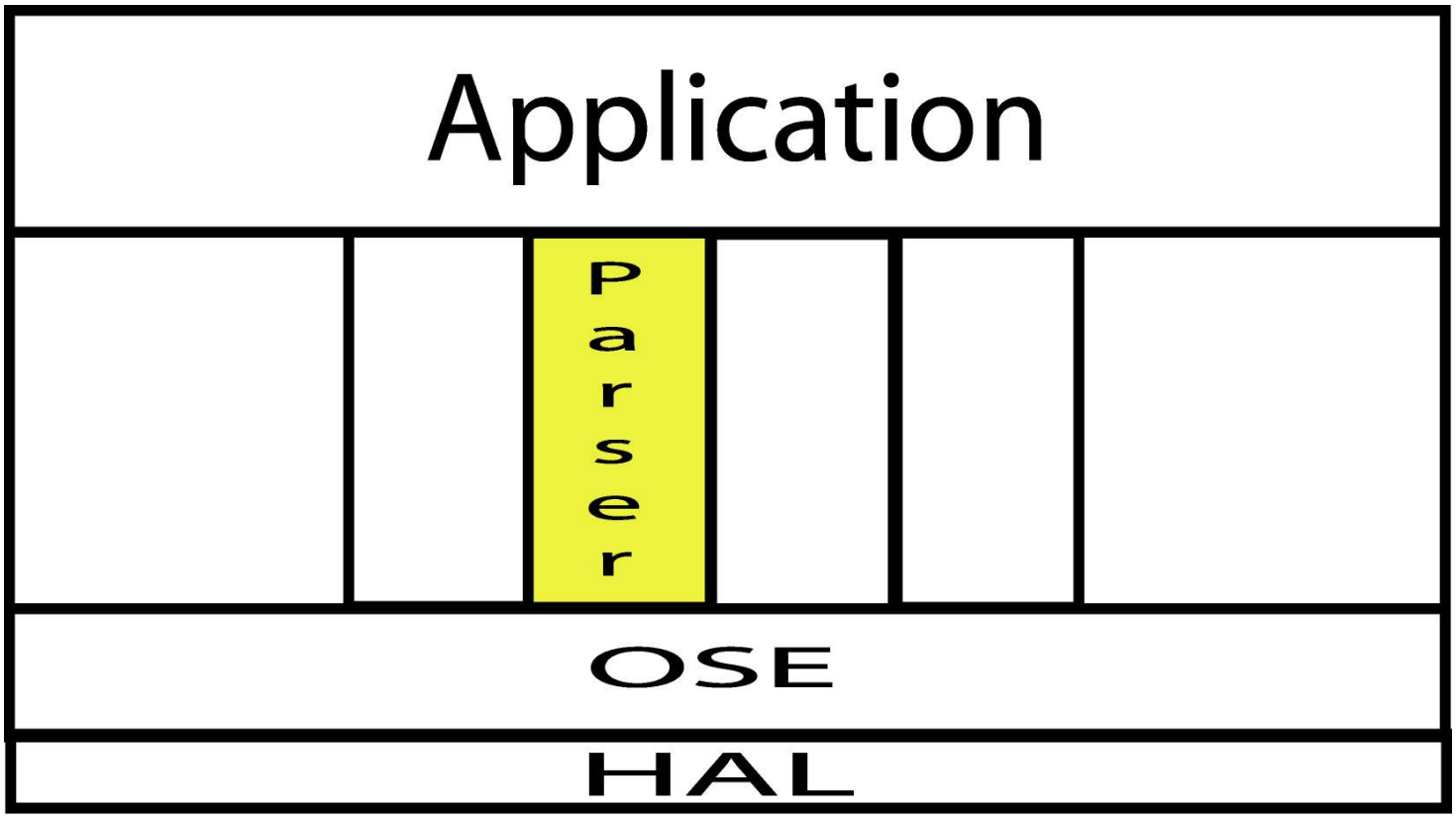
- use KeY in a real world scenario:
  - Software from Agilent Technologies
  - Used in their chemical analysis modules

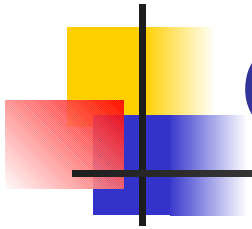




Case study

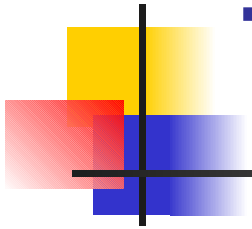
- Parser is a system library





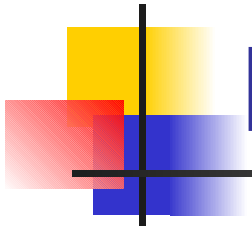
Overview

- Some facts on the system
- Making the system ready for verification with KeY
- Problems expected/encountered during verification
- Our next steps



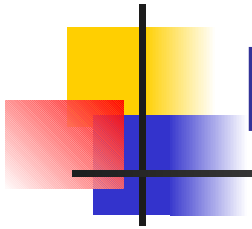
The System

- Parser (better: command interpreter)
  - Generates a parser from a description
  - Can add new instructions and data types at run-time
- Written in C++
- ~12 classes



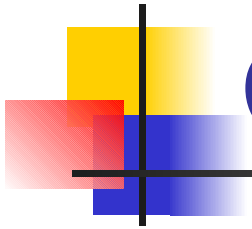
Parser

- Contains the two basic tables:
  - Ranges (RangeTable)
  - Commands (CommandTable)
- Receives incoming InstructionSignal
- Checks incoming instruction using CommandTable
- Replies with a TotabSignal if a matching is found.



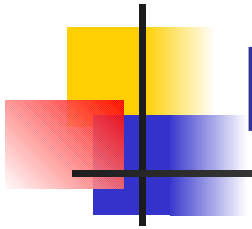
Ranges

- A Range defines a datatype:
  - Basetype: INT, FLOAT, STR, INSTR
  - Example: „myIntRange“, BTYP_INT, „1..10;15;50..100“
- Can be used to describe the syntax of instructions
- Are implemented as lists of RangeElem
- RangeTable holds a list of all Ranges
- May have subranges



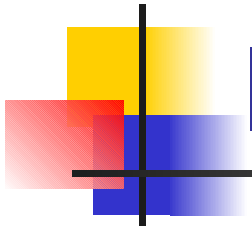
Commands

- It holds all the information for a single command:
  - Parameter types
  - Optional?
- For each Command there exists a CmdTabEntry
- Commands can be looked up by id



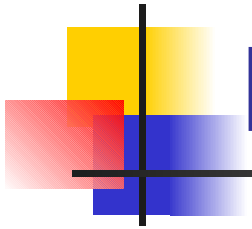
ParElem

- Is the base class of all parameters
- One derived class for each basetype:
  - ParInt, ParFloat, ParStr, ParInstr
- A parameterlist for an instruction is a list of ParElems
- ParElems may have sublists for variable length parameters



Interactions

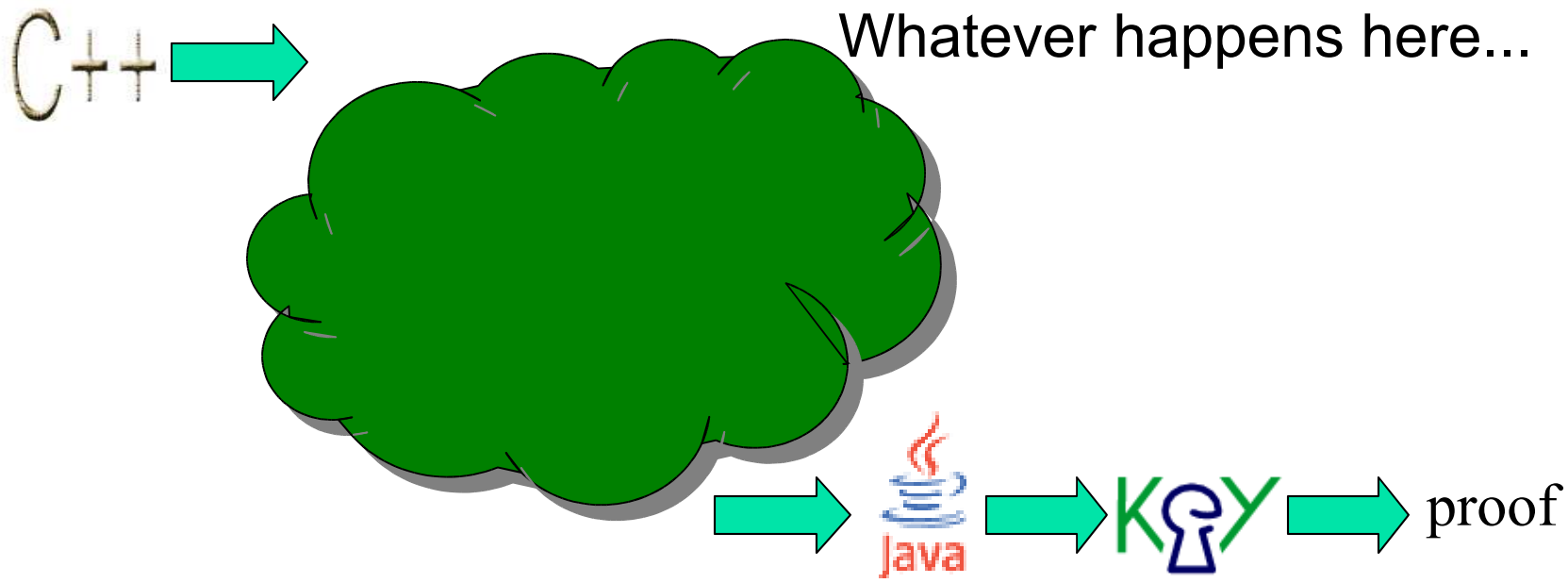
- Set up a new type by sending a RangeDefSignal describing it
- Issue Commands by sending an InstructionSignal
  - The parser checks the syntax
  - Dispatches the command to the execution units
  - A Totabsignal is sent containing the result of the parsing operation



Problems

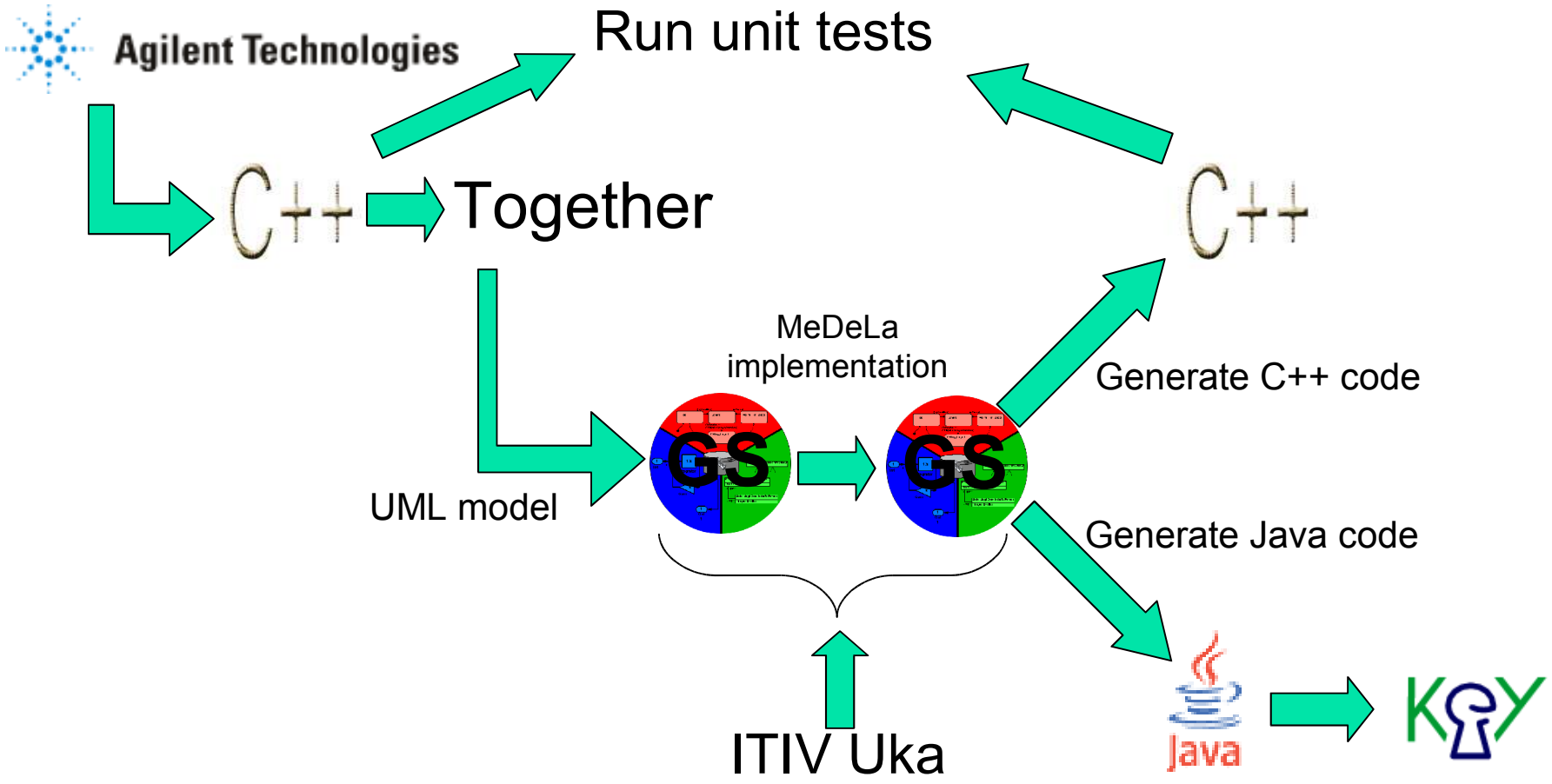
- Parser is written in C++, KeY needs Java.
- How do we get the same program in Java?

Main Problem

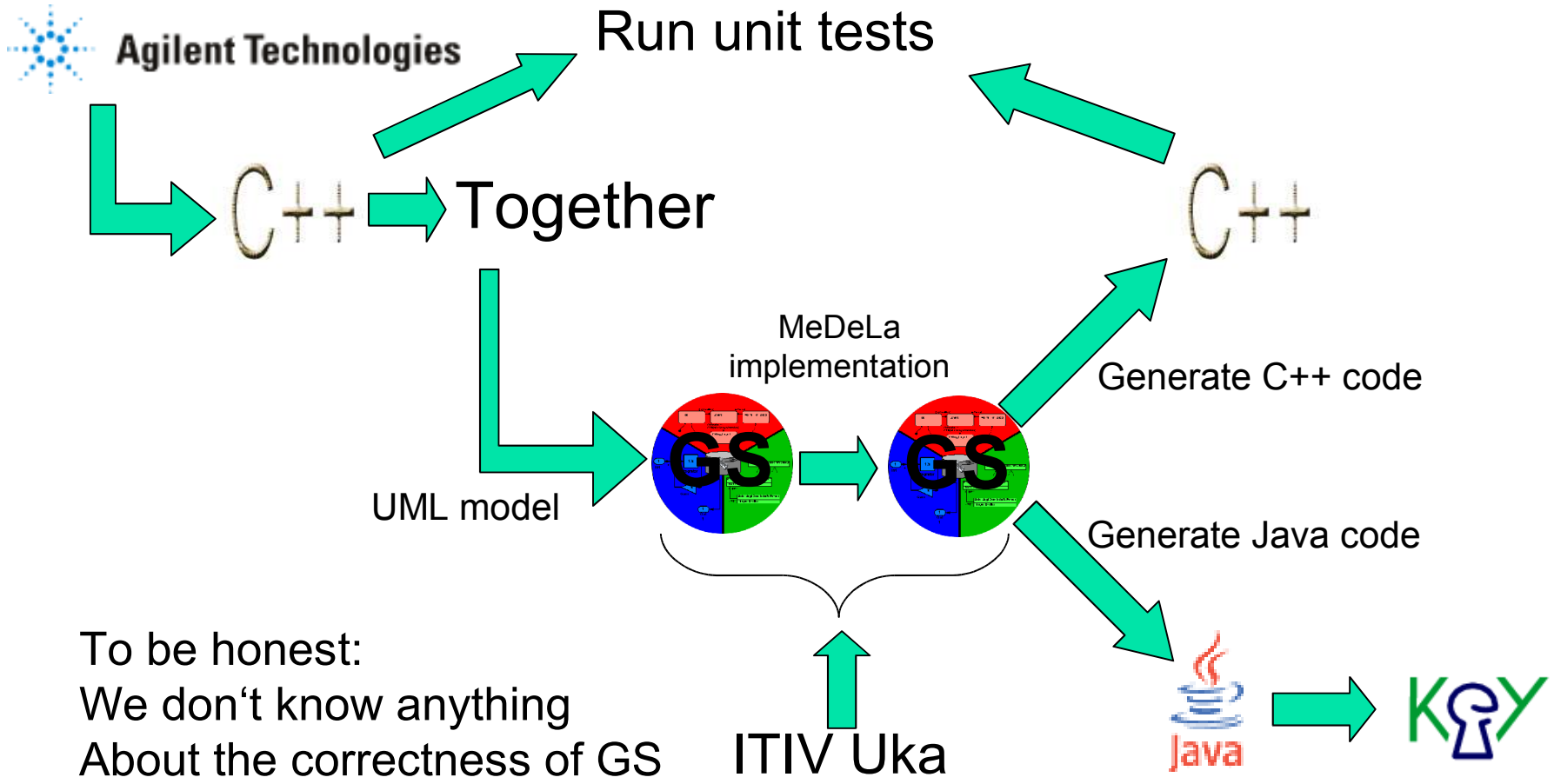


... we should at least have some indication, that the proof says something about the correctness of the C++ program

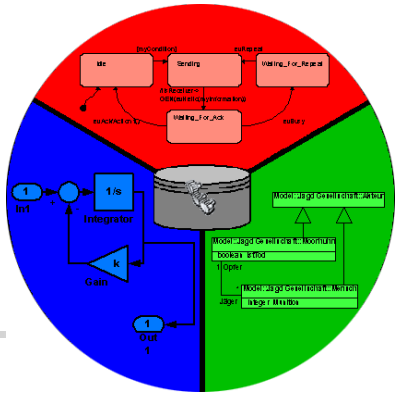
Our Solution



Our Solution



General Store



- Is a UML Tool, developed at ITIV at Uka
- Allows implementation of Methods in a language called MeDeLa (Method Definition language)
- Can generate Java and C++ code

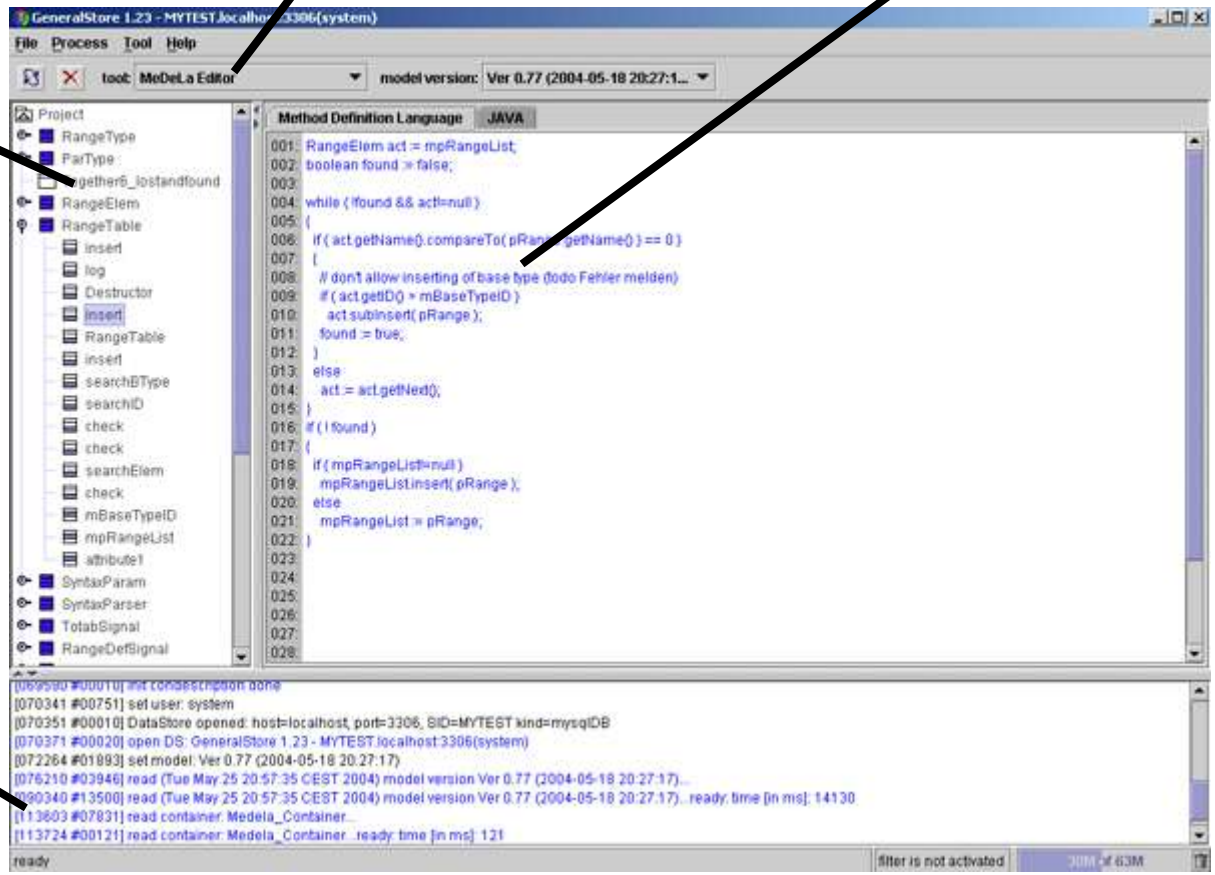
General Store

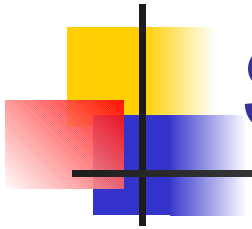
List of tools

MeDeLa editor window

List of all classes and their members

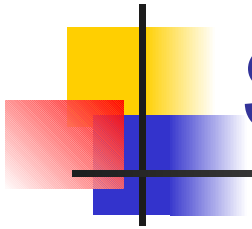
Status window





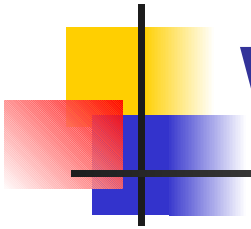
Steps

- Import the C++ code into Together and generate a UML model from it
- Export this model to GeneralStore and (re-)implement the methods in MeDeLa
- Generate Java and C++ code
- Run the original C++ unit tests on the generated C++ code
- Add OCL-constraints to the java version
- Prove the correctness of the generated Java code



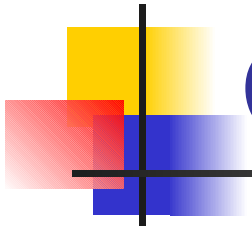
Sounds easy, but...

- GS is still under development
  - Had to type implementations into the properties-window in Together in <medela></medela> tags in the beginning
  - Various problems with code generation:
    - No static and abstract specifier
    - Return type of constructor: void
    - No switch in medela
    - And some more
- Need to emulate some C++ constructs in Java



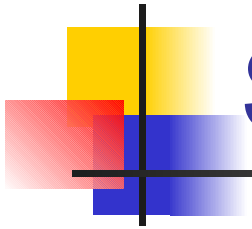
What are we going to prove?

- Correctness of type system:
  - RangeIds are unique
  - inserting and checking ranges is correct
- ParElems should only have a rangeID of their own base type
- Recursive list manipulations are correct
- Relation: input, resulting TotabSignal



Challenges

- Mostly recursive searching and inserting
  - We need some automatic variable renaming in KeY
  - structural induction
- String handling:
  - The parsers output heavily depends on the input: „null vs. not null“ clearly isn‘t enough
  - ParStr and ParInstr are based on strings



So far

- We have translated about ¾ of the system
- I‘ve got familiar with KeY by working through the proof-examples
- We did some minor proofs:
  - Assigning id/name to a range works correctly
  - Assigned id is unique