UNIVERSITA' DEGLI STUDI DI NAPOLI FEDERICO II
Scuola Politecnica e delle Scienze di Base
Corso di Laurea Magistrale in Ingegneria Informatica

Instrumentation & Measurement for Particle Accelerator Lab
IMPALAB

CERN

Modern approach for monitoring cryogenic systems for particle accelerators

Author
Domenico Francesco De Angelis

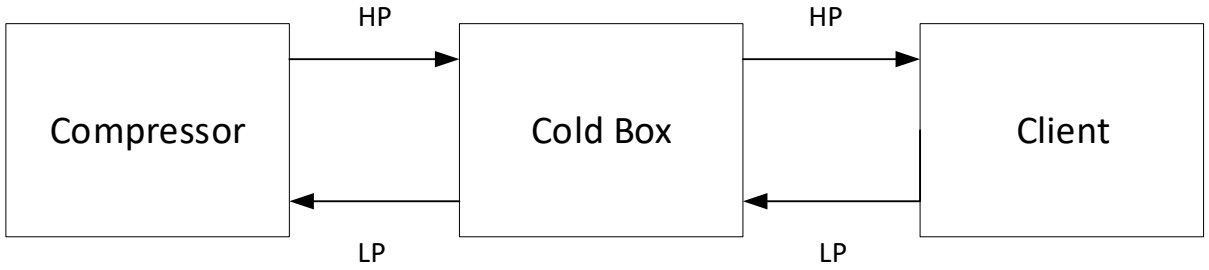
Relatore
Pasquale Arpaia

Co-relatori
Marco Pezzetti
Porfirio Tramontana

Cryogenics at CERN (TE-CRG-CE)

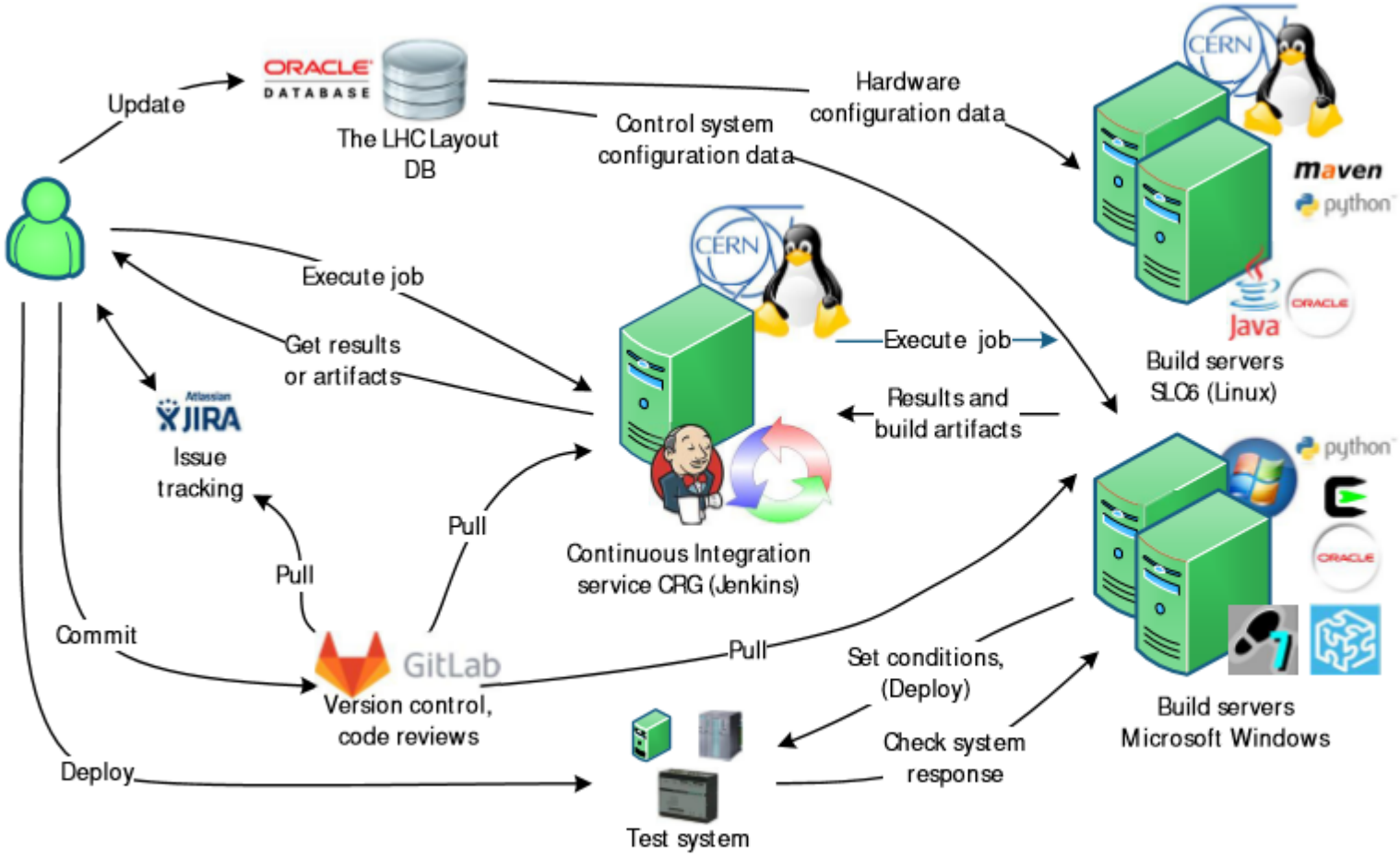
CRG group is responsible of the design, construction, operation & maintenance of cryogenic systems for accelerators and detectors.

The group has to ensure the temperature required in all CERN's complex, and especially extremely low temperatures.



cryoplant

Continuous Integration in CRG



- Generate hardware conf.
- Check data consistency
- Generate source code

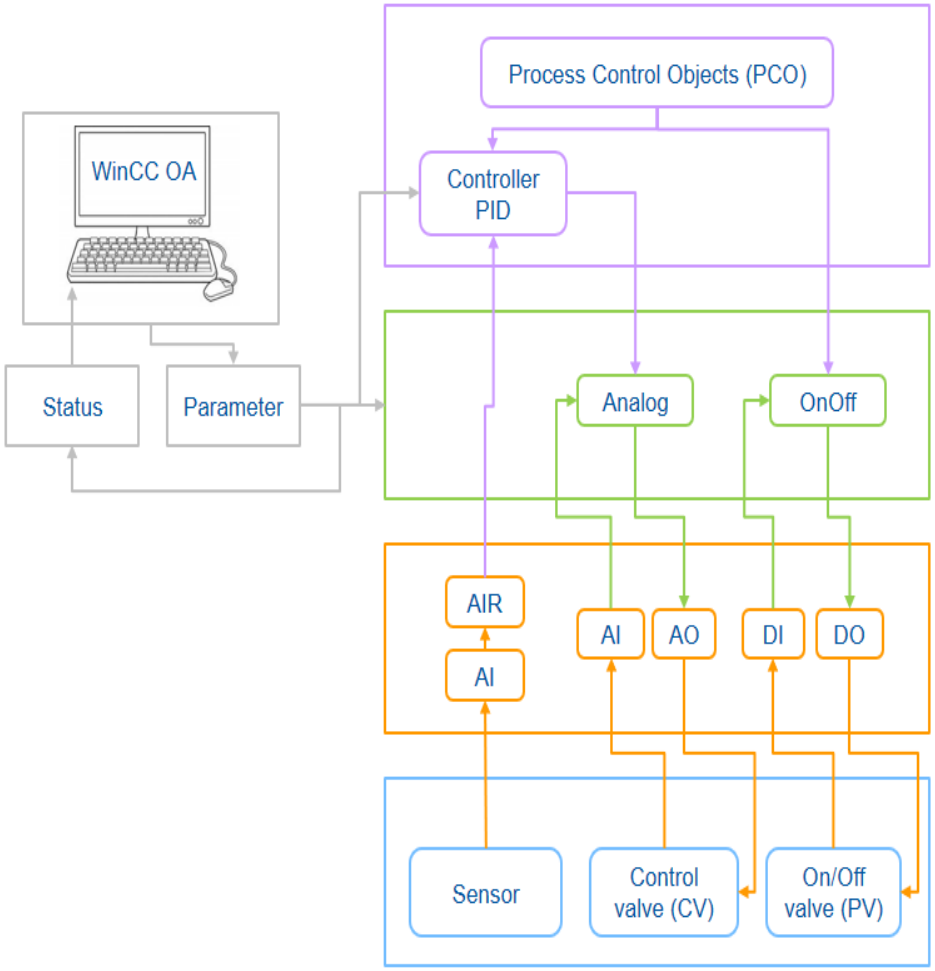
- UNICOS CPC6 UAB 1.8.0
- Apache Maven
- Java 1.8, Python 2.7
- Oracle Instant Client 11
- 8 virtual machines with 2 or 4 job slots (24 in total)

- Generate object specification
- Build PLC project
- (Deploy)
- Test

- Siemens SIMATIC Step7 V5.5
- Schneider Unity 11
- s7cli, ucli (cmd-line for SIMATIC and Unity)
- Oracle Instant client 11
- UNICOS simulation and testing framework
- Cygwin, Python 2.6
- 10 virtual machines with 1 job slot (SIMATIC limit.)

The updated architecture of the CI system.

Management of the CERN Cryoplant



Control Objects :
They are performing control actions in the process logic by driving Field Objects

Field Objects :
It is the collection of objects acting on « interfaces objects » (valves, motors)

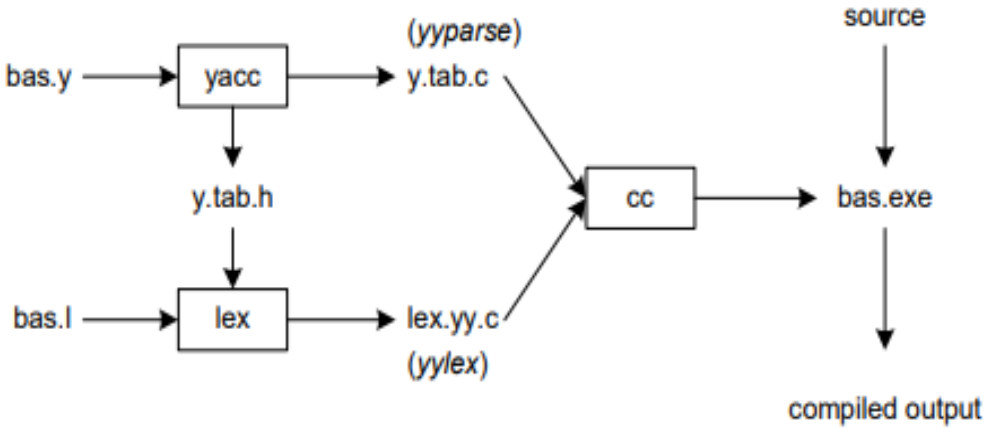
I/O Objects :
It is the collection of objects that interfaces signals informations to the UNICOS architecture

Devices :
Controlled devices

The monitoring and management of the *cryoplant* exploit PLCs complex system.

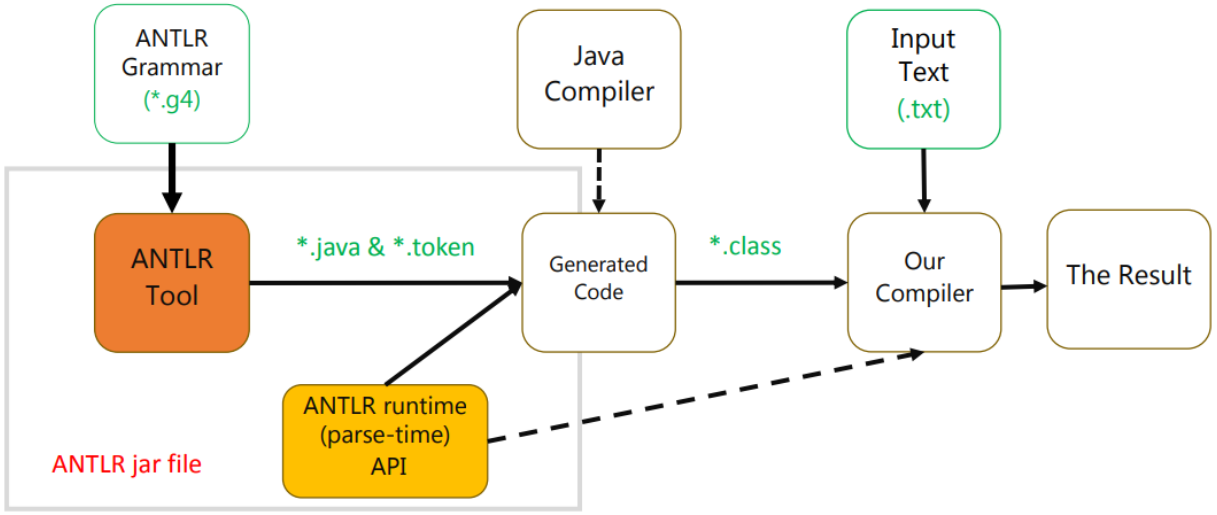
The most part of the legacy code is based on FBD language.

Existing solutions



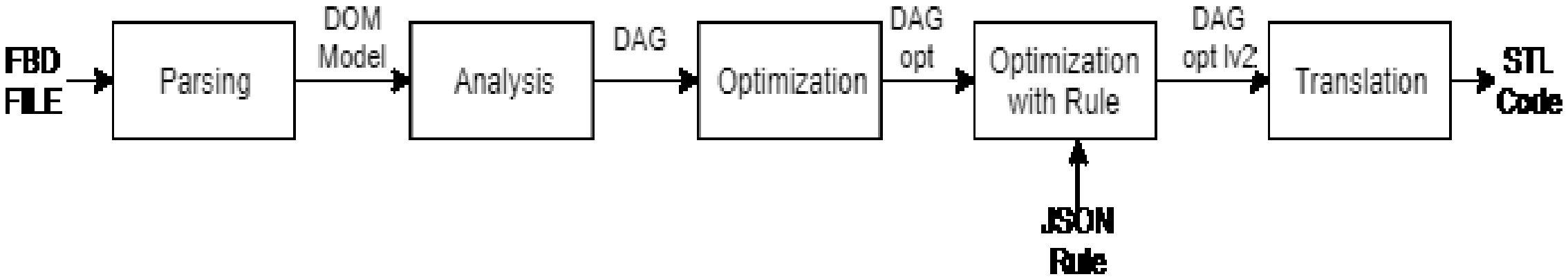
Yacc/Lex

Bison/Flex



ANTLR

Proposal



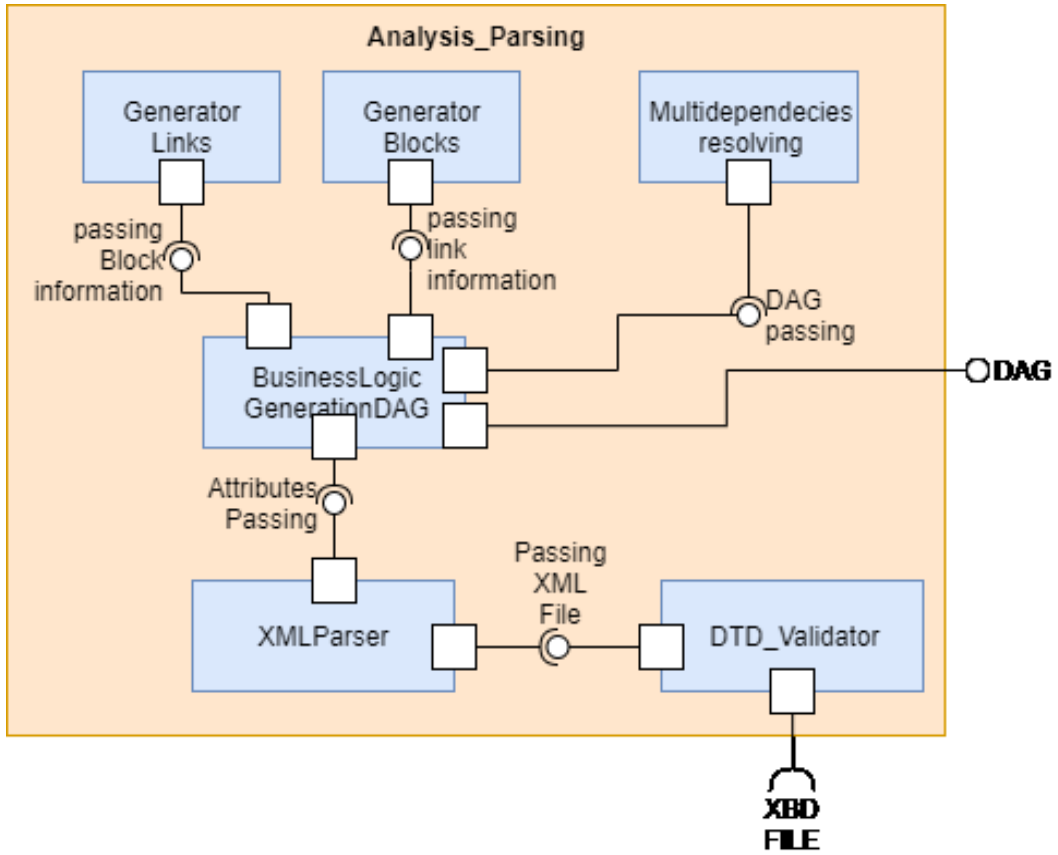
Architecture Pipe-&-filter

Initial input xml file represented the FBD source code

Each block reads an abstraction of source

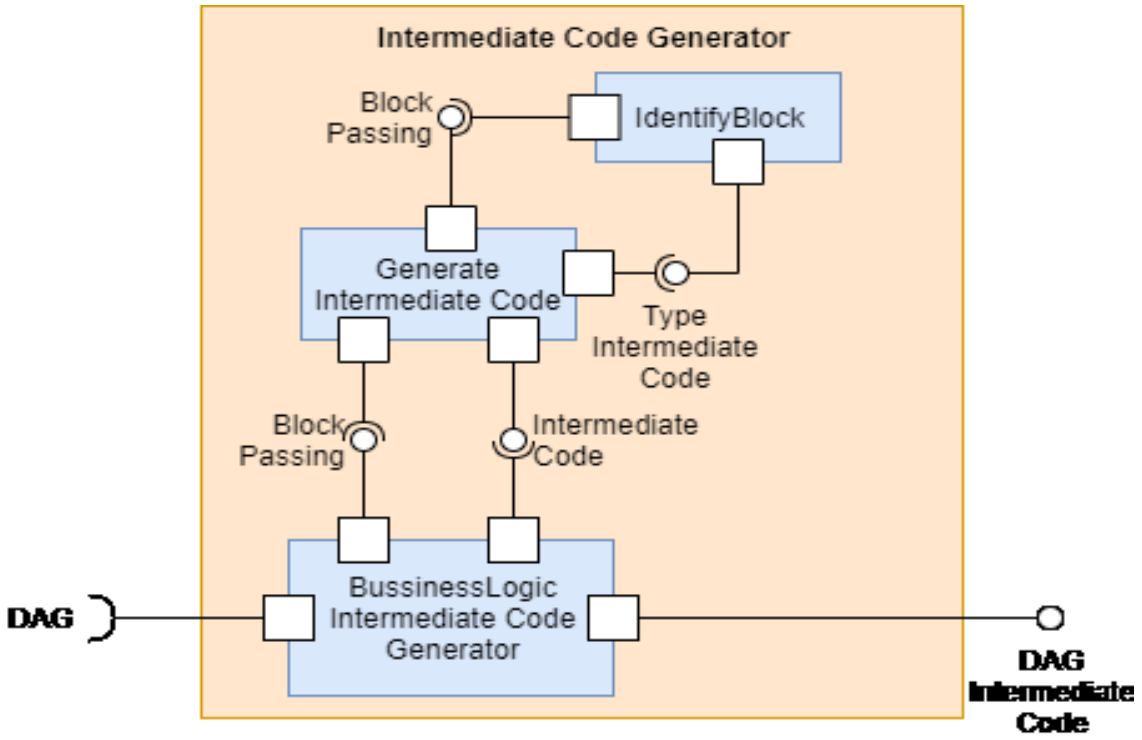
At the end, the output represents the ST source code

Parsing & Analysis



- Validation of .XBD file
- Parsing the XML file
- DAG generation of source code
- Optimization of block multidependency

Intermediate Code Generator

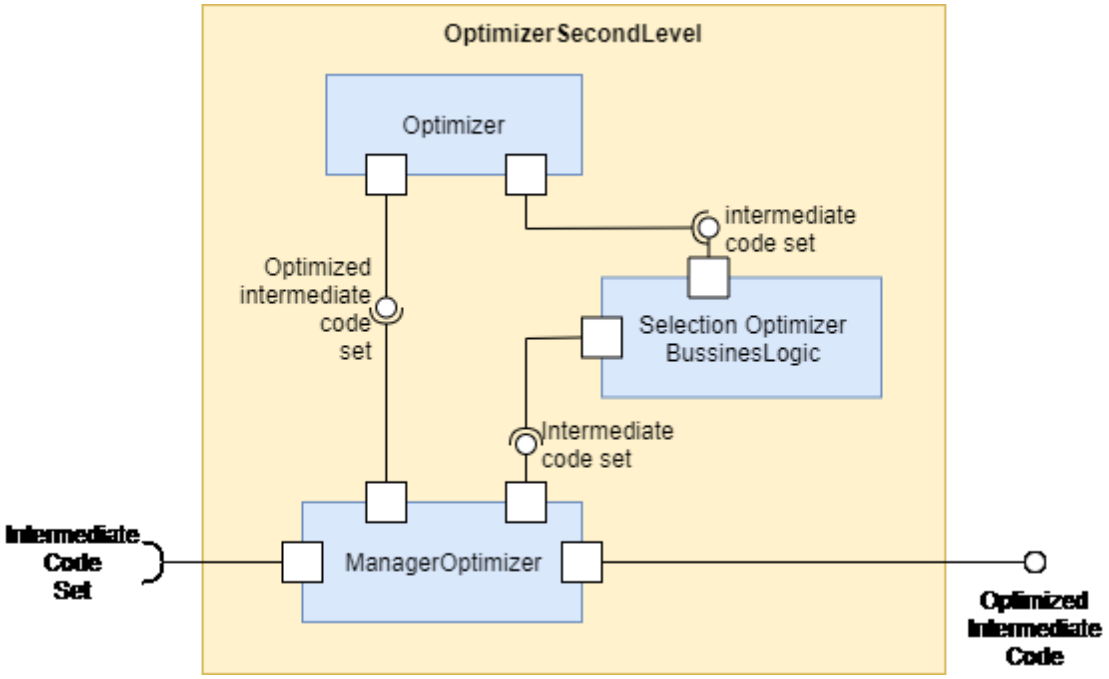


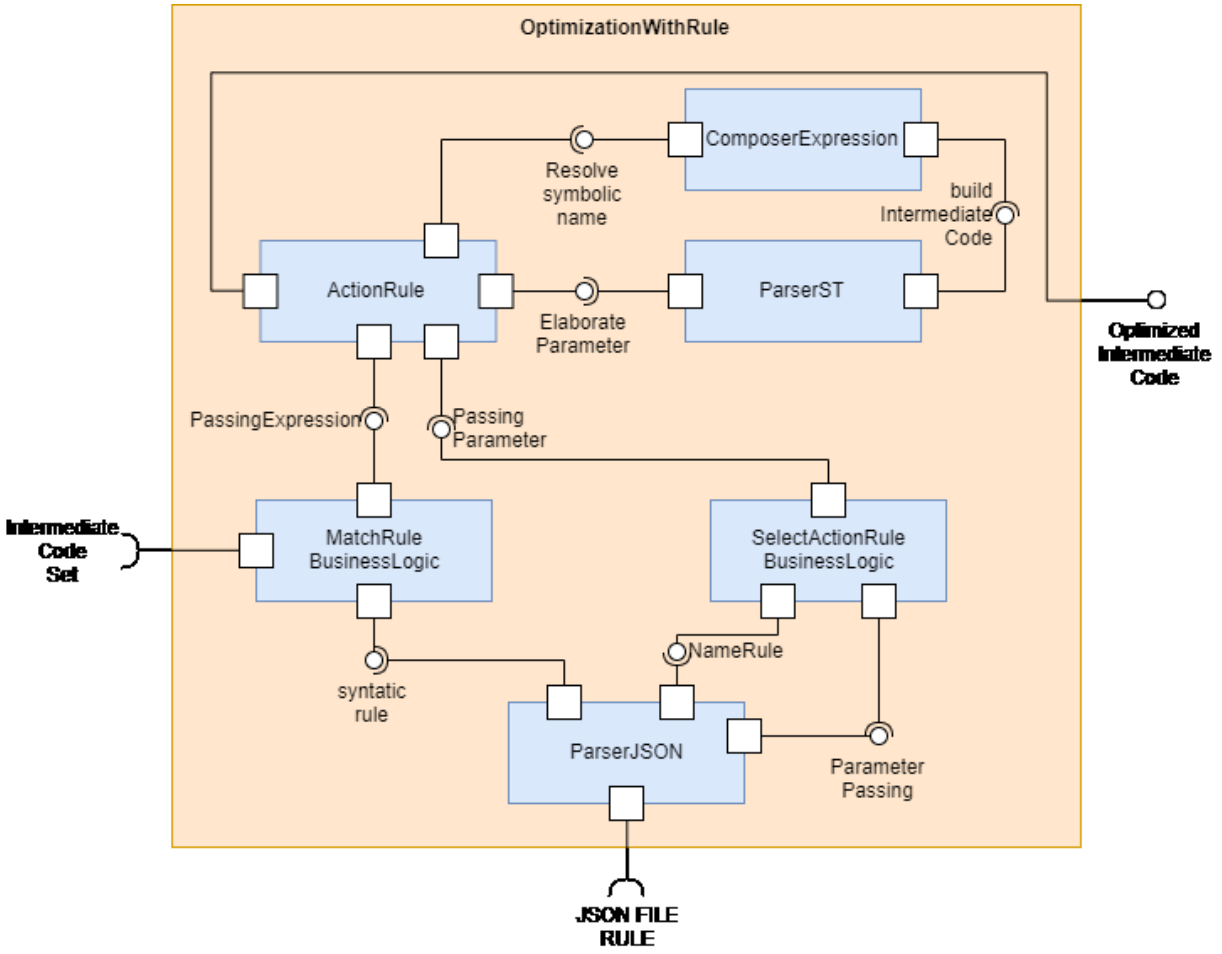
- Blocks identification
- Generation of a intermediate code set

The intermediate code is based on quadruples:

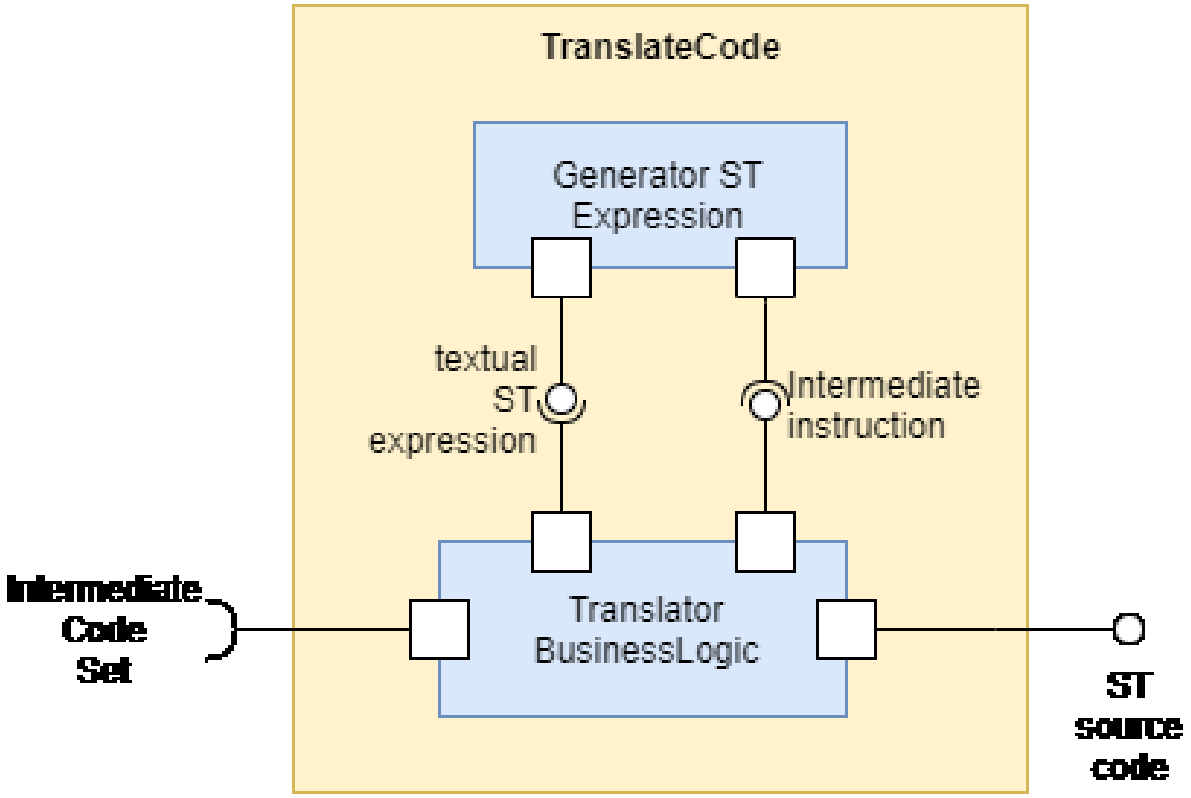
<result, operator, op1, op2>

Optimization Second & third levels





Translation Code



- Generation of textual ST source code
- Any further processing is needed

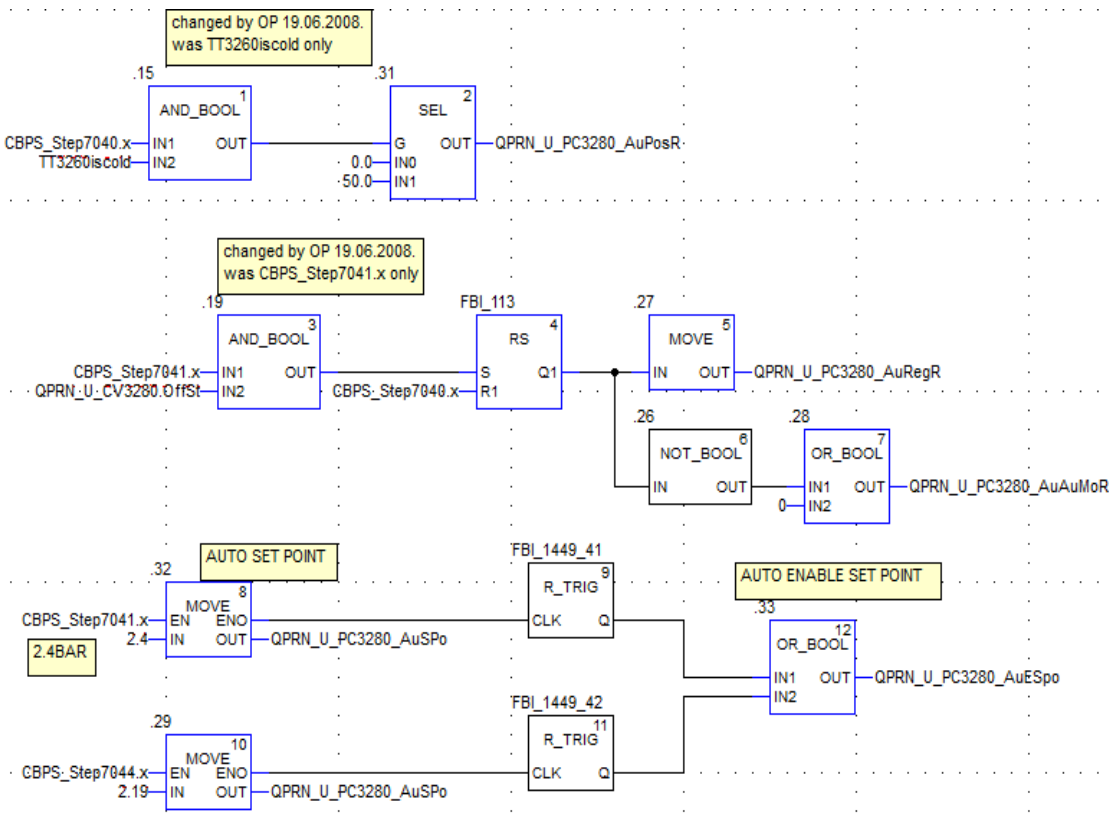
Testing

Unit Test on the following components:

- ParserST – OptimizationWithRule subcomponent
- ComposerExpression – OptimizationWithRule subcomponent
- Parsing_Analysis component
- Intermediate Code generator

Integration Test on whole FBD2ST system

Examples of use



Input FBD file

Non-optimized ST source

```
IF CBPS_Step7044.x THEN
    QPRN_U_PC3280_AuSPo := 2.19;
END_IF;

LV_QPRN_U_CBPS_PC3280_R_TRIG(CLK := CBPS_Step7044.x);

IF CBPS_Step7041.x THEN
    QPRN_U_PC3280_AuSPo := 2.4;
END_IF;

LV_QPRN_U_CBPS_PC3280_R_TRIG1(CLK := CBPS_Step7041.x);

QPRN_U_PC3280_AuESpo := LV_QPRN_U_CBPS_PC3280_R_TRIG1.Q
    OR LV_QPRN_U_CBPS_PC3280_R_TRIG.Q;

QPRN_U_PC3280_AuPosR := SEL(G := (CBPS_Step7040.x AND
TT3260iscold),
    IN0 := 0.0, IN1 := 50.0);

LV_QPRN_U_CBPS_PC3280_RS(S := (CBPS_Step7041.x AND
    QPRN_U_CV3280.OffSt),
    R1 := CBPS_Step7040.x);
QPRN_U_PC3280_AuRegR := LV_QPRN_U_CBPS_PC3280_RS.Q1;
QPRN_U_PC3280_AuAuMoR := NOT(LV_QPRN_U_CBPS_PC3280_RS.Q1)
OR 0;
```

Conclusion & Future work

Conclusions:

- Translation from FBD to ST language.
- Optimization stages.
- Ability to expand the optimizer with a textual rule file, described in JSON.

Future work:

- Behaviours testing of whole logic converted by FBD2ST.
- Import the system inside the CI system of TE-CRG.
- Expand the optimization applicable at intermediate code set.