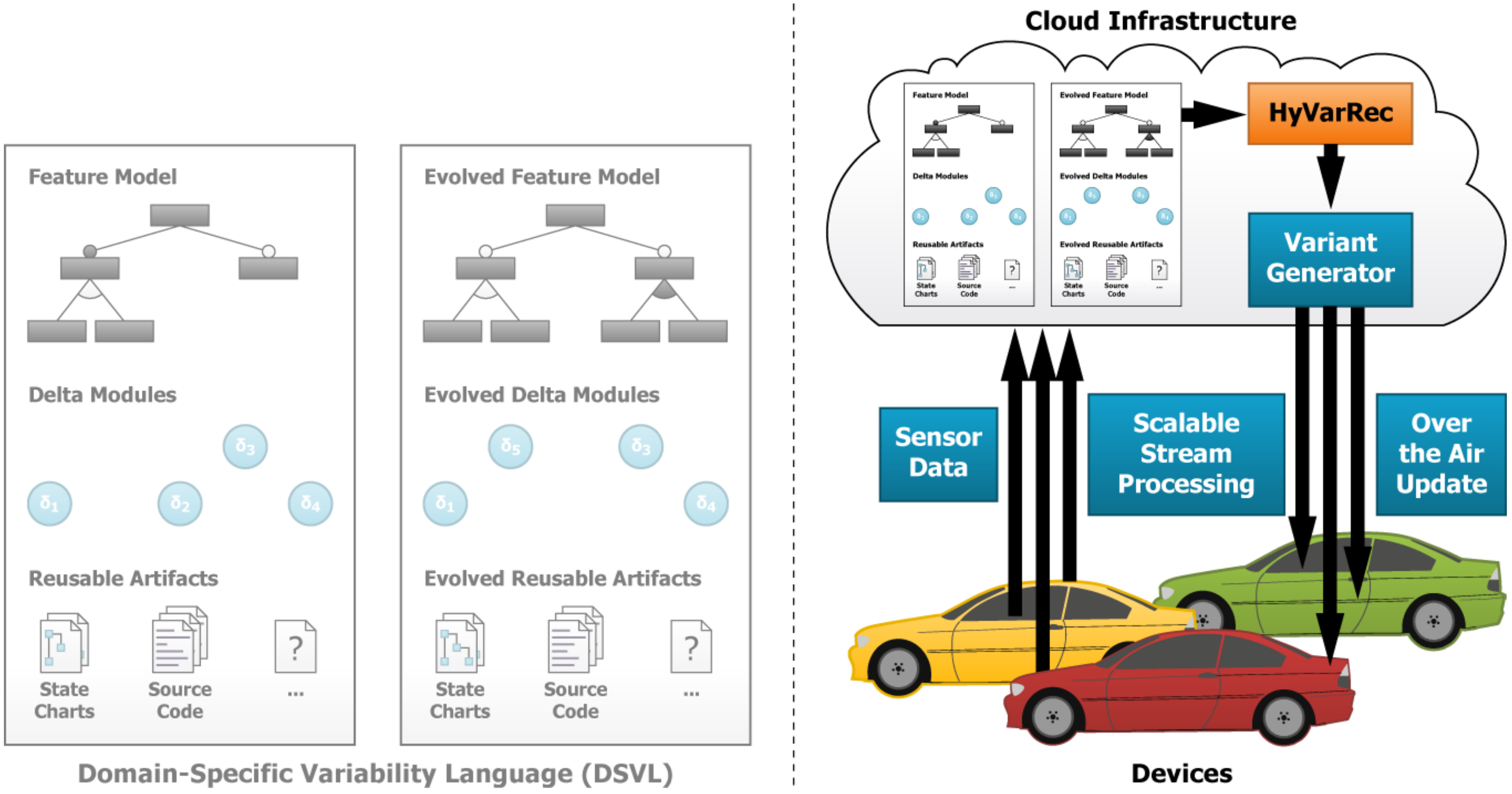
Context Aware Reconfiguration in Software Product Lines

Jacopo Mauro

University of Oslo

NordConsNet, May 2017

HyVarRec: A Hybrid Variability Reconfigurator



Hybrid Feature Model

- Devices have to adapt based on their surroundings
- A valid configuration may depend on some contextual information
  - *E.g., Car in Russia uses GLONASS, not GPS*

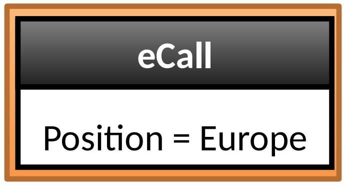
Feature Model

- Contextual change may invalidate the current configuration
- Necessary to compute a new valid configuration
- Minimal changes to the configuration are better
  - Less intrusive for user
  - Less operation to perform

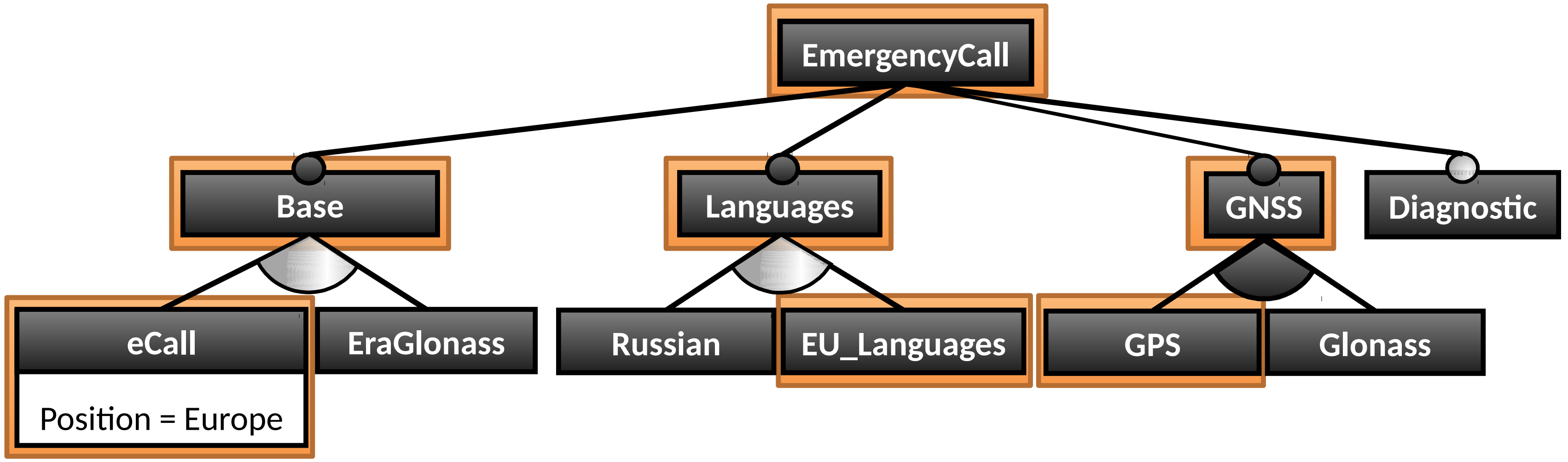
HFM & Demostrator

Extend FM with

- *contextual information* (e.g., Position)
- *validity formulas*



Demonstrator: Initial Configuration



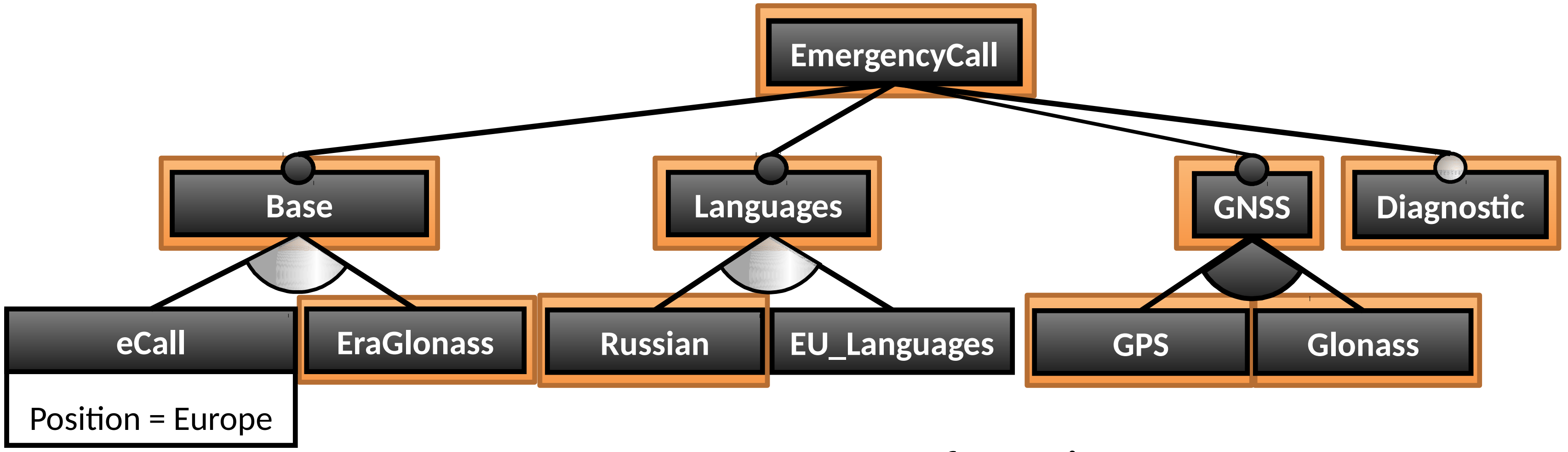
Constraints:

EraGlonass ↔ Russian
EraGlonass → Diagnostic
eCall → EU_Languages
eCall → GPS
EraGlonass → Glonass

Context Information

- Position: Enum[Russia, Europe]

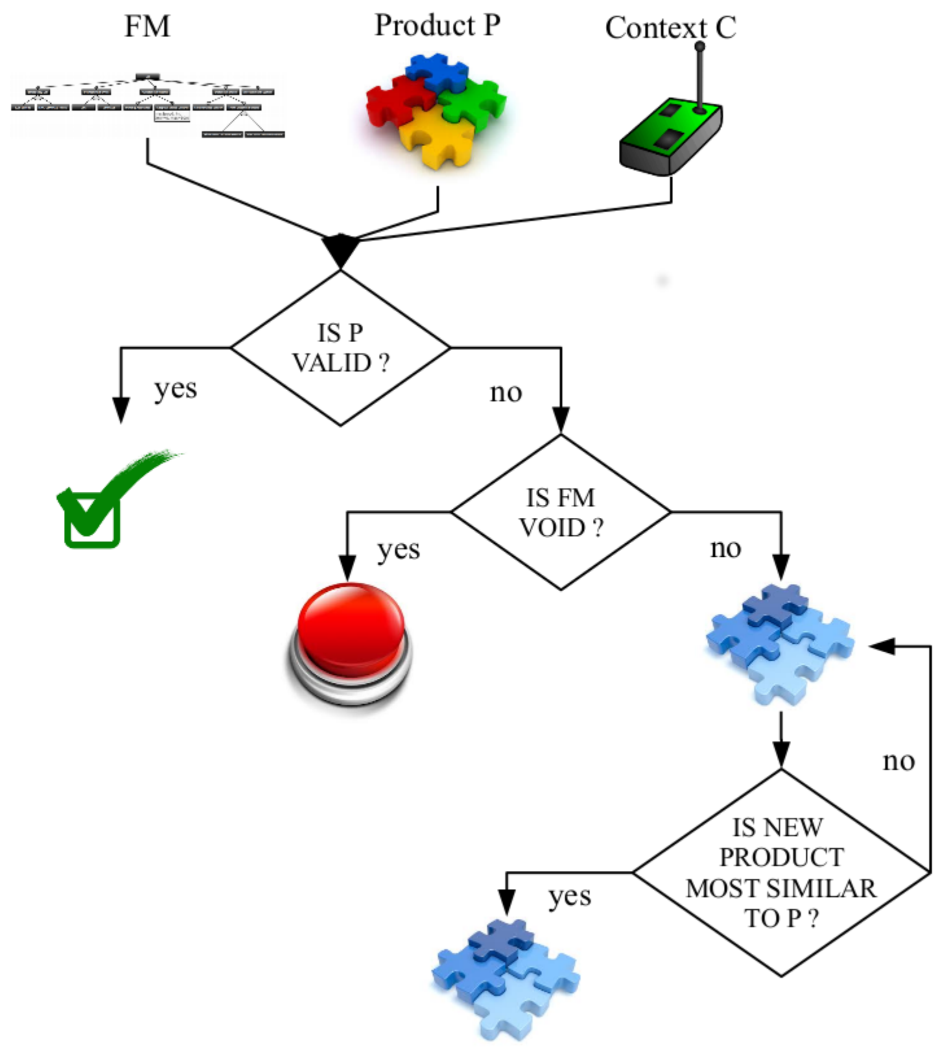
Demonstrator: New Configuration



Constraints:
EraGlonass ↔ Russian
EraGlonass → Diagnostic
eCall → EU_Languages
eCall → GPS
EraGlonass → Glonass

Context Information
- Position: Enum[Russia, Europe]

HyVarRec


FM
Product P
Context C
IS P VALID ?
yes
no
IS FM VOID ?
yes
no
no
IS NEW PRODUCT MOST SIMILAR TO P ?
yes

HyVarRec: Under the hood

- Constraint programming to check validity and compute a new valid product
- Anytime solver
- Uses MiniZinc solver + MiniSearch
- Open source: https://github.com/HyVar/hyvar-rec

HyVarRec

- Features and attributes → integer variables
- Feature dependencies, attribute dependencies, and validity formulas → constraints

- First checks configuration validity
- If not valid -> Triggers reconfiguration
- Finds the most similar valid configuration:
  - Deselect minimal amount of features
  - Change minimal amount of attribute values

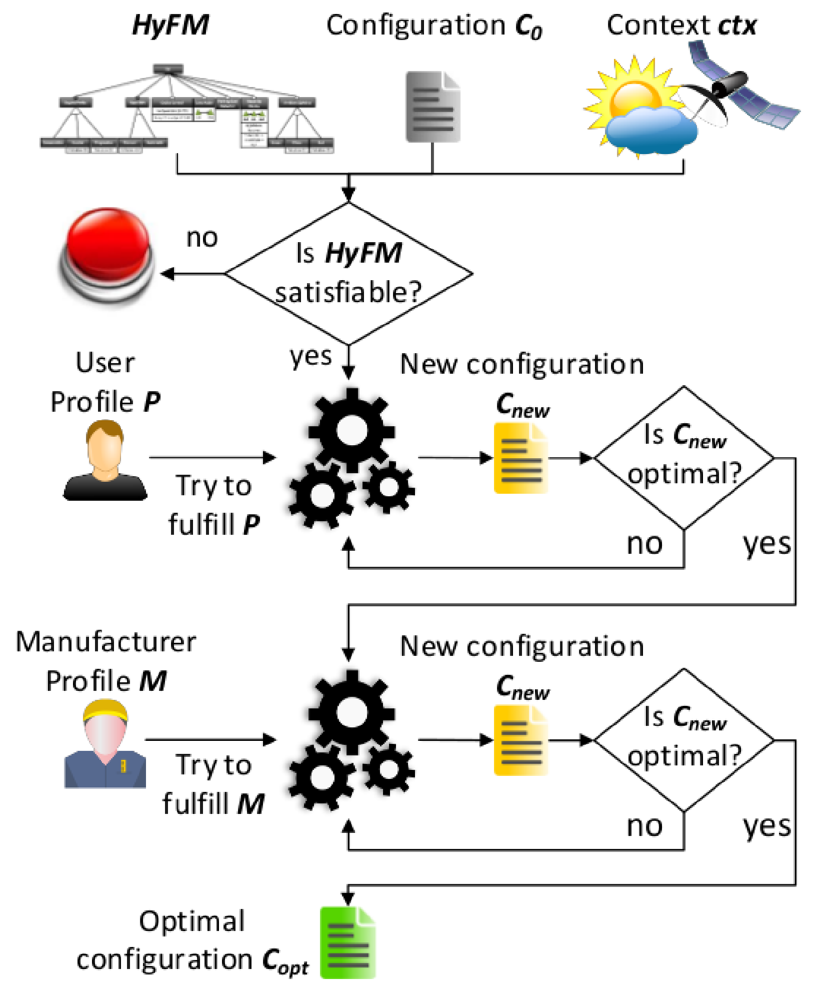
MiniSearch

- Language for specifying meta-search in MiniZinc
- Solver-independent
- Better if solvers use API to avoid restarts

```
include "minisearch.mzn";
var int: obj; % other variables and constraints not shown
solve search min_bab(obj);
output ["Objective: "++show(obj)];


function ann: min_bab(var int: obj) =
  repeat (
          if next() then
             commit() /\ print() /\ post(obj < sol(obj) )
          else break endif
  );
```

HyVarRec & Preferences

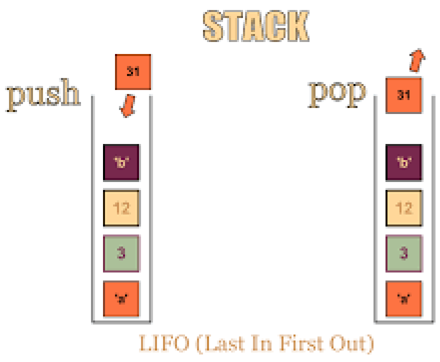


What changes

- Preferences = Soft Constraints added
  - If problem Sat -> OK
  - If problem Unsat -> retract constraint
- Problem 1: with current MiniZinc solvers adding constraints = restart solver !!!
  - Waste of time
  - No reuse of no-goods if LCG
- Problem 2: MiniSearch not supported + bugs

SMTs

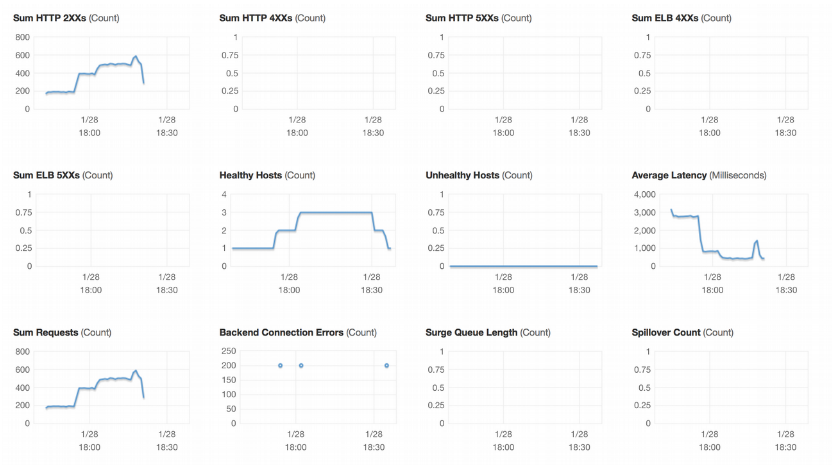
- New HyVarRec version uses SMTs (Z3)
- Push and pop for preferences
- New optimization feature of Z3 (not SMT-lib standard)
- Bonus: almost free FM analysis
  - Support of universal quantification



HyVarRec: Virtualization

- Portable, deployable using Docker
- Scaling → per car
- Preliminary load testing:

  AWS + default Auto Scaling strategy



Future Works

- Use HyVarRec to study standard FM properties (e.g., dead features, …)
  - Check if every context admits a valid configuration
- Improve the performance
  - Local search or heuristics?
- Benchmarking (possible with real FM)

Thank you for your attention!

Questions?