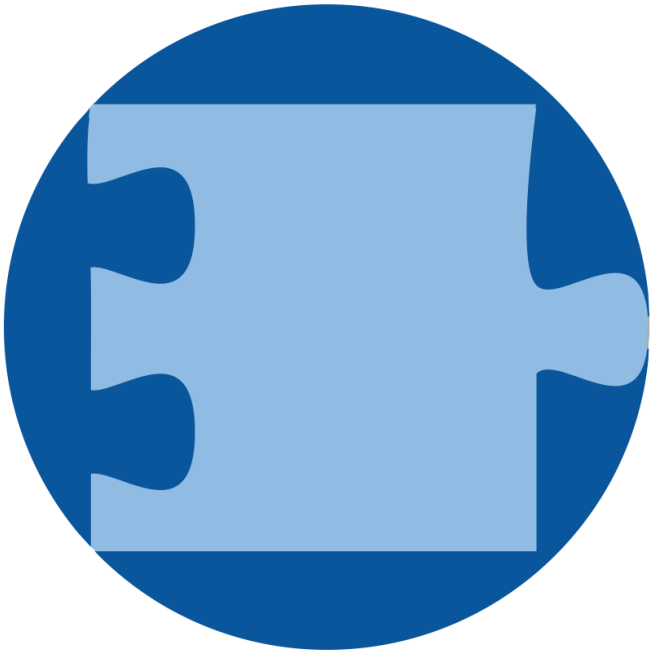
High-Level Overview of Entire EmbeddedMontiArc Project

EmbeddedMontiArcStudio
Sprachen
Generatoren
Simulatoren

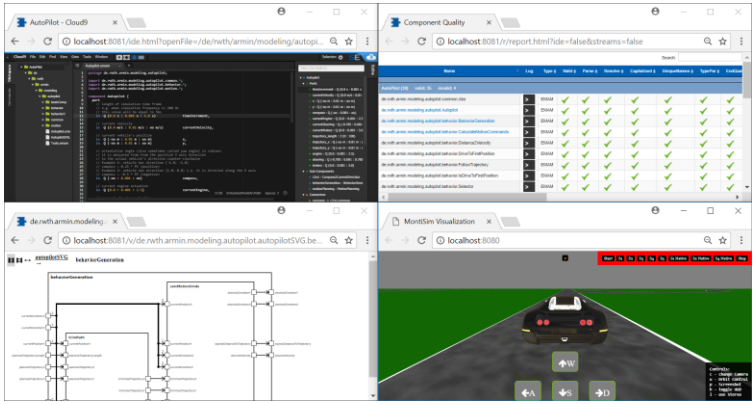


Michael von Wenckstern,
Evgeny Kusmenko

Software Engineering
RWTH Aachen
http://www.se-rwth.de/

EmbeddedMontiArc for SLE (teaching)

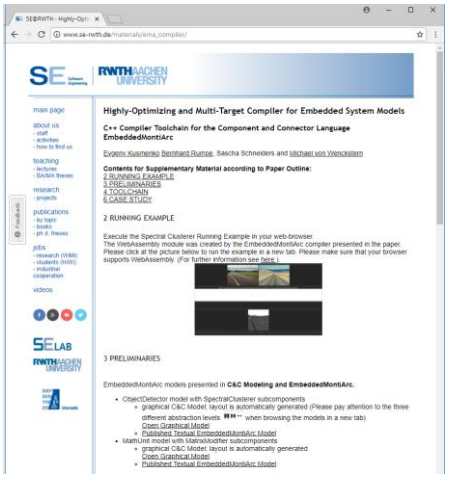
- **Developing Large Language Families with MontiCore is easy**
  - Our Language Family contains about 25 MontiCore grammars
- **Developing powerful Modeling Tools with MontiCore is possible**
  - EmbeddedMontiArcStudio is based on MontiCore infrastructure
- **Multiple Teams can develop language tools together**
  - EmbeddedMontiArc is developed by ca. 15 students in parallel
- **Developing with MontiCore is Fun ;)**
  (Can execute the models and
  see the car drifting)



EmbeddedMontiArc for Publications

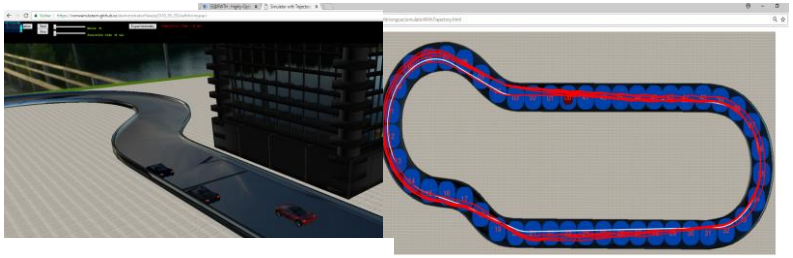
- Reviewers can
  - Inspect Models (textual and visual representation) in Browser
  - Execute generated Code in Browser

⇨ Gain confidence about tools or make larger models online available

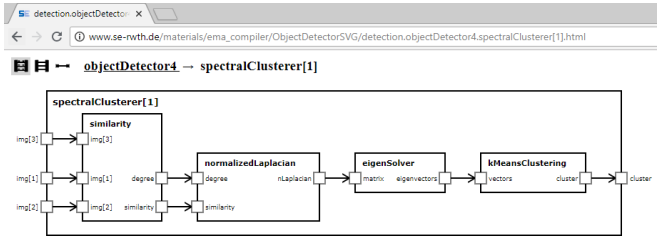


Can be directly uploaded to SE homepage

http://www.se-rwth.de/materials/ema_compiler/

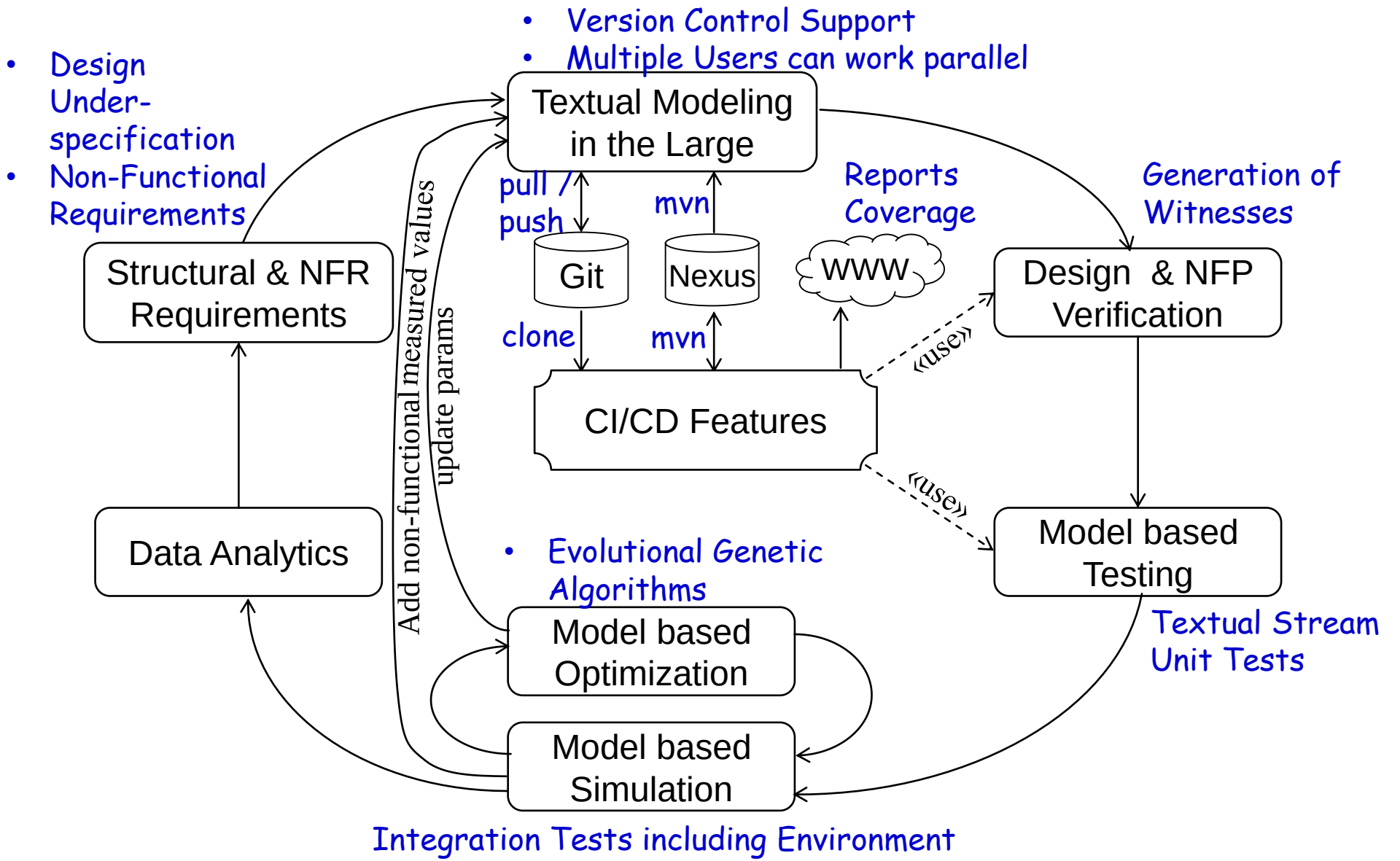


Execute Models in Browser



Inspect Models in Browser

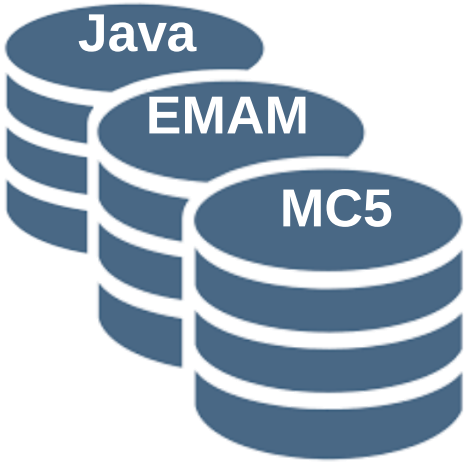
Agile Development with EmbeddedMontiArc



EmbeddedMontiArc as DataSource

- EmbeddedMontArc language family MontiCore grammars with language aggregation, and language embedding
- Have over 1‘500 models for EmbeddedMontiArc grammars
  → large model repository
- Over 10 (RoutePlaning, Parking, ADAS, PacMan, SuperMario, Wheather Balloon, Image Clustering, LapRacing, PumpStation, Turbine Controller) complete presentable examples
- EmbeddedMontiArc incl. Simulator over 70 gitlab repos

- Tested new features in MontiCore 5 (detected some bugs)
- Artifacts are analyzed with SH’s tool (detected some bugs)
- Repos can be used to teach DevOps (Git, CI, CD & more)



EmbeddedMontiArc Integrates SE Methods

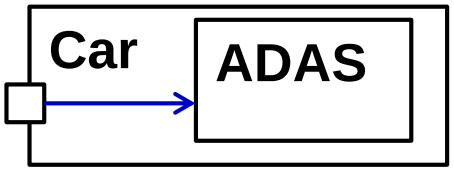
- EmbeddedMontiArcStudio (available for Windows 64bit and Linux*) integrates many SE methods
  - Stream Testing (based on AH's methods)
  - View Verification (based on JOR's methods)
  - Tagging (based on ML methods)
  - Language Aggregation via Symbol Table (based on PN methods)
  - OCL/P and CD4A** (using languages and concepts; BR UML/P)
  - Generator Composition (using MC's template concepts)
  - Simulation and Co-Simulation (inspired by CB)
  - Deployment and Library Concept** (uses AH's SE Infrastructure)
  - Reporting Features (inspired by MontiCore)

* Virtual Machine will be uploaded in next two days

** Full Integration will be finished in 2 months

EmbeddedMontiArc Main Features

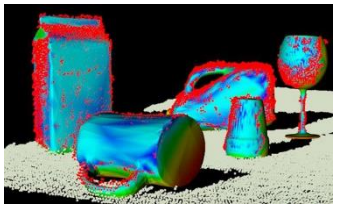
Modeling



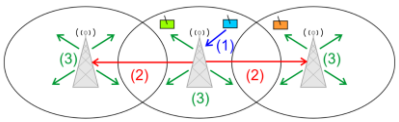
Views & Verification

$$2\sqrt{\ }3\sqrt{\ }4$$

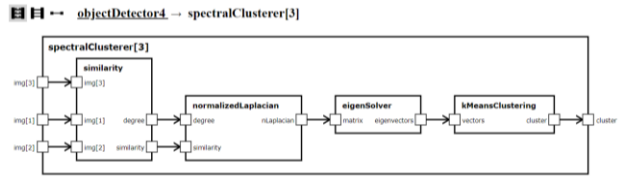
Unit Testing



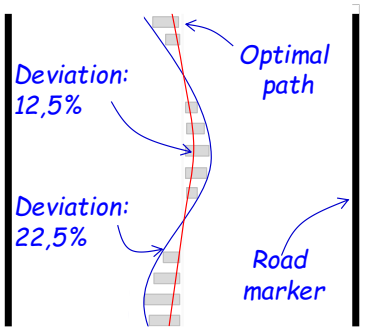
AI & Image Recognition



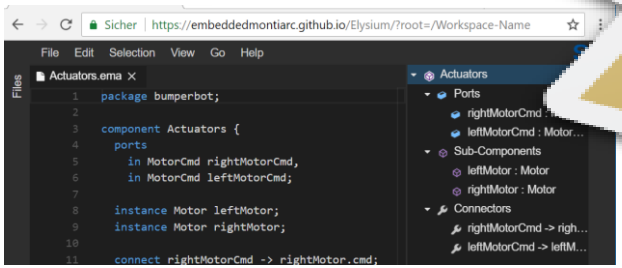
Simulation



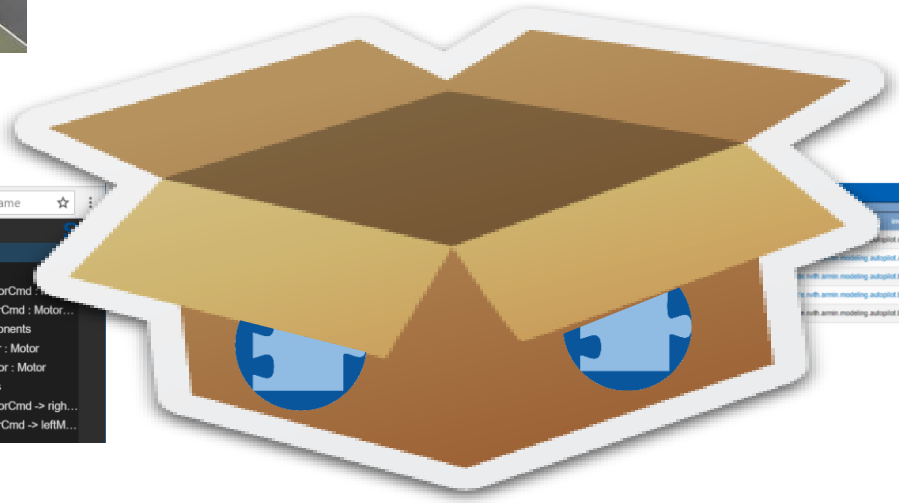
Automatically Layouting

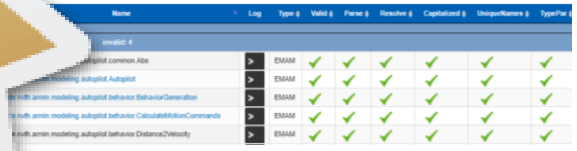


Acceptance Testing



IDE

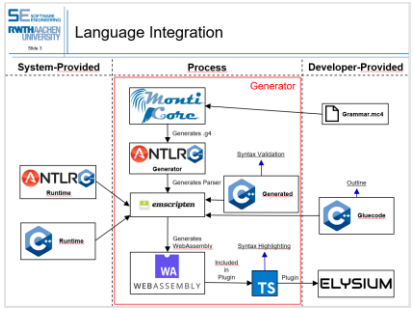




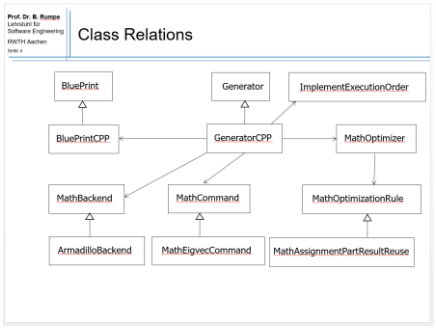
Reporting

EmbeddedMontiArc Repo Quality

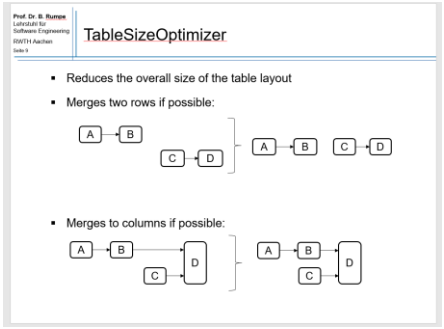
- Research Prototype: Can be shown at conferences
- Own Documentation Repository: https://git.rwth-aachen.de/monticore/EmbeddedMontiArc/Documentation
- Main Repositories are documented by itself:
  14 Compact PPTX-Presentation about Design & Algorithms
  https://git.rwth-aachen.de/monticore/EmbeddedMontiArc/Documentation/tree/master/reposlides



Architecture



Class Relation



Algorithm

- Many Unit- and Integration Tests (Test Coverage about 75%)
- Activated Test Pipeline in GitLab

Files					Complexity	Coverage
src/main/java/de/monticore/lang/monticar	1,028	912	19	97	86.45%	88.71%
Project Totals (62 files)	1,028	912	19	97	86.45%	88.71%

Basic Architecture Design Decisions

- All steps are Self-Contained Services (communicate via CLI or via REST)
  - Each SCS can use a different MontiCore version
  - Can be developed and replaced independently
  - Batch files chain SCS together to useful activities
    - E.g. C++ Generator → CLANG → Simulator/3d Visualisation
- Bundled as portable application in an archive EXE (only Windows 64-bit as requirement)