Concept for an Open-Source Drilling Simulator With Focus on Education and Research

Patrick Höhn, Insa Nüske, Julian Kunkel

Georg August Universität Göttingen, Gesellschaft für Wissenschaftliche Datenverarbeitung Göttingen (GWDG)

GWDG
Gesellschaft für wissenschaftliche Datenverarbeitung mbH Göttingen

GEORG-AUGUST-UNIVERSITÄT GÖTTINGEN IN PUBLICA COMMODA SEIT 1737

Outline

- Introduction
- Current state of drilling simulation
- So what is missing? / Proposed solution
- Implementation details
- Conclusions and future work

Introduction – Who are we?

Mission

- Support and enable science by applying appropriate technologies, developing innovative solutions, and providing reliable services
- Planning, implementation and operation of IT infrastructures together with and for our customers

Central responsibilities

- Modern and secure IT infrastructure
- IT support for excellent research
- In-house research for innovative IT-Services

Supra-regional tasks

- National High Performance Computing Center
- National HPC Center of the DLR
- AI service center for sensitive and critical infrastructures
- Data center in four NFDI consortia
- Host for DARIAH-EU, German National Library, GFBio, NUM CODEX, MWS, WirLernenOnline, etc.
- Cloud operator, including AcademicCloud for universities in Lower Saxony

Introduction – Drilling Simulation

Important for:

- Education
  - Deep understanding of drilling process for drilling crews
  - Motivation of students to deeper explore the field
- Industry
  - Optimally planned drilling projects reduce costs and risks
  - Cost effective training of management and rig crews
- Research
  - Detailed simulations provide base for new innovative solutions
  - Complement and enhance understanding of experimental investigations

Existing Drilling Simulators

Commercial:

- 3t Drilling Systems
- NOV Portable Simulators
- PayZone Drilling Simulator

Freemium:

- OpenLab Drilling

Open Source:

- Open Source Drilling Community
- Texas A&M Github Repository

Existing Data Sources

- BGR
  - KTB
  - Genesys
- Equinor
  - Many datasets
- Inhouse data

Existing Models

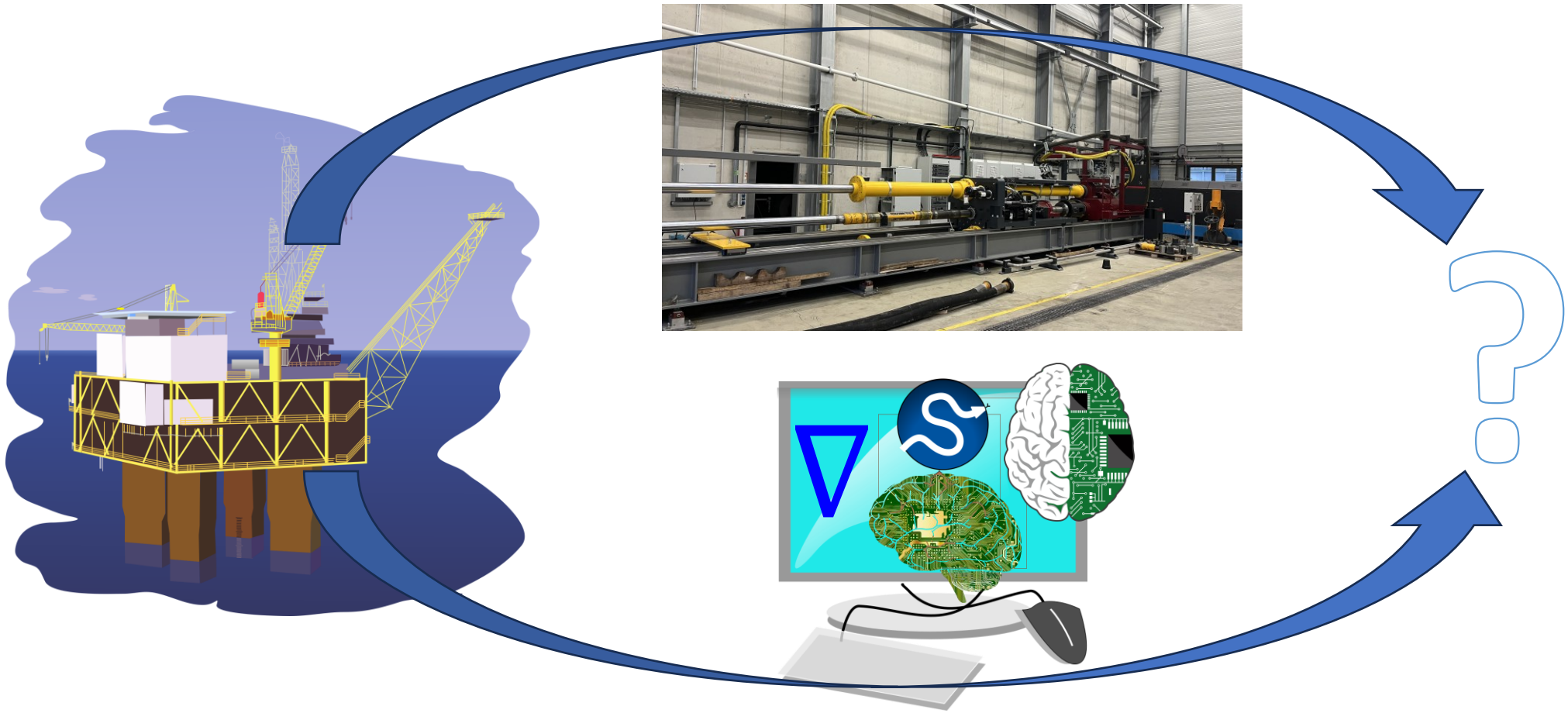
Classical Models

- Temperature models
- ROP literature models
- TOB literature models
- Combined time series TOB and ROP model
- Drift-Flux Cuttings Transport model
- WOB model
- Drillstring dynamics models
- Time series flow models
- Torque & Drag model

Machine Learning Models

- ROP models
  - Symbolic regression
  - Scikit-Learn based ML models
- TOB models
  - Scikit-Learn based ML models
- WOB model

So what is missing?



Education and Research in Drilling Engineering

- Reliant on commercial simulators with no access to underlying source code

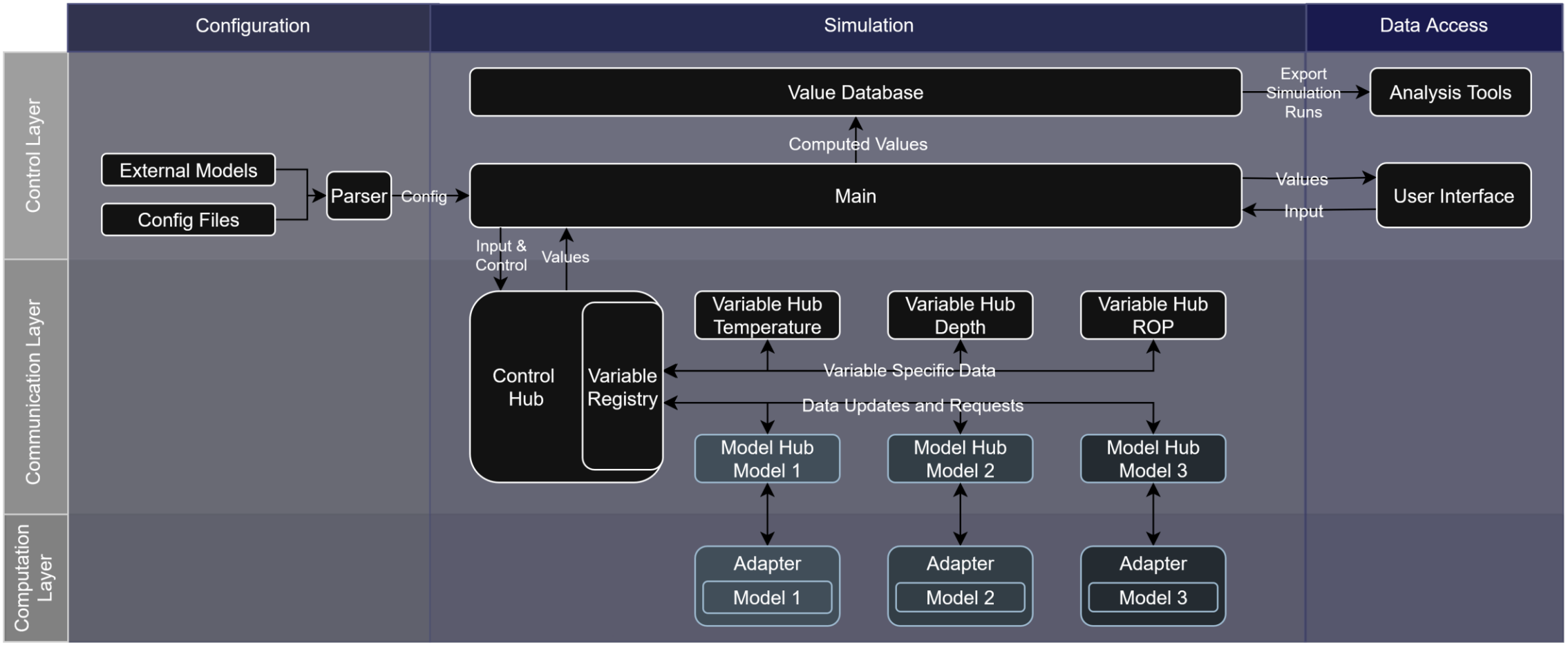
→Limited understanding for both students and researchers of the detailed behaviour of the implementation

- Implementation languages, e.g. C++, C# not easy understandable by drilling people
- Limited possibility of modifications restricted by API access to commercial simulators

Proposed Solution

- Implementation as an Open-Source Simulator accessible to everyone
- Published under AGPL for maximum openness
- Potential for collaborative development
- Possibility to validate used algorithms, replace models and adjust simulation processes
- Implementation using Python as easy entry boundary and support for machine learning
- Adapters to other languages and interfaces, e.g. OPC-UA, FMI, grpc

Architecture



Configuration-Example

```yaml
TEMPERATURE_0D:
  name: "TEMPERATURE"
  desc: "Temperature at the deepest point of the borehole."
  interpolator: "COSINE"
  value:
    type: "float"
    unit: "°C"
    initial_vals:
      0: 21
```

```yaml
ROP_MODEL:
  name: "ROP"
  desc: "Oversimplified model for ROP."
  adapter_type: "SIMPLE_PY"
  definition:
    type: "SIMULATION.CLASSMETHOD"
    location: "models.models.OversimplifiedROPModel"
    target: "get"
    static: true
    input:
      - variable: "ROCK_TYPE"
        datatype: "str"
    output:
      - variable: "ROP"
        unit: "m/h"
        datatype: "float"
```

Highlighted Features

- Models from supported sources connectable using .yaml interface definitions with no additional coding needed
- Simulation customization possible using configuration files in yaml format
  - Including definition of additional variables and interpolation methods
  - In the future configuration files
    - Connection to supported databases
    - Customization of interfaces
- Further customization possible by modular code design beyond limits of yaml configuration, each component with defined interface and replacable independently if necessary

Implementation Status

Current Implementation Status:

- Simulation run startable using CLI tool, with automatic connection to simple externally defined python models using configuration files
- Each model executed asynchronously in an independent process, with their computed values available to other models through a central variable registry
- Simulation results accessible through an output file with timeseries data
- Working prototype of graphical user interface with live data

Upcoming Features:

- Integration of first real open-source models
- Extensive user and developer documentation

Conclusion and Future Work

- Free and open-source framework for research and teaching
- First working prototype implemented and successfully executes simulations using oversimplified models
- Already including concurrent execution to prevent performance bottlenecks
- Next steps: Integration of existing models and export of simulation data

Spread the word for collaboration!