KITP Santa Barbara
Collider Physics Workshop
12 February 2004

LUND UNIVERSITY

Whither PYTHIA?

Torbjörn Sjöstrand

Department of Theoretical Physics
Lund University

History
Status
Future

Origin

Q: Why rewrite?

A: Need to clean up!

Q: Why C++?

A: Only game in town!

My original idea:

- simple and robust structure
- throw out-of-date alternatives
- keep current physics $\sim$unchanged
- many minor improvements

Lifestyle

HERWIG camp monolithic (?)

Lund camp pluralistic

Many programs:

- JETSET, PYTHIA
- Ariadne, LDCMC
- Fritiof, Luciae
- Lepto (Aroma, Lucifer, PomPyt, . . .)
- (Rapgap, Cascade)
- ((HIJING, . . .))

Many physics models:

	aspect	number
•	final-state showers	$2 \to 3$
•	initial-state showers	$3 \to 4$
•	string fragmentation	3
•	baryon production	3
•	Bose-Einstein	2
•	colour reconnection	3 + not
•	multiple interactions	$1 \to 2$

Evolution

Leif's idea:

before you worry about physics,

create a generic platform for event generation

= "a language within the language"

HERWIG++ accepted and joined

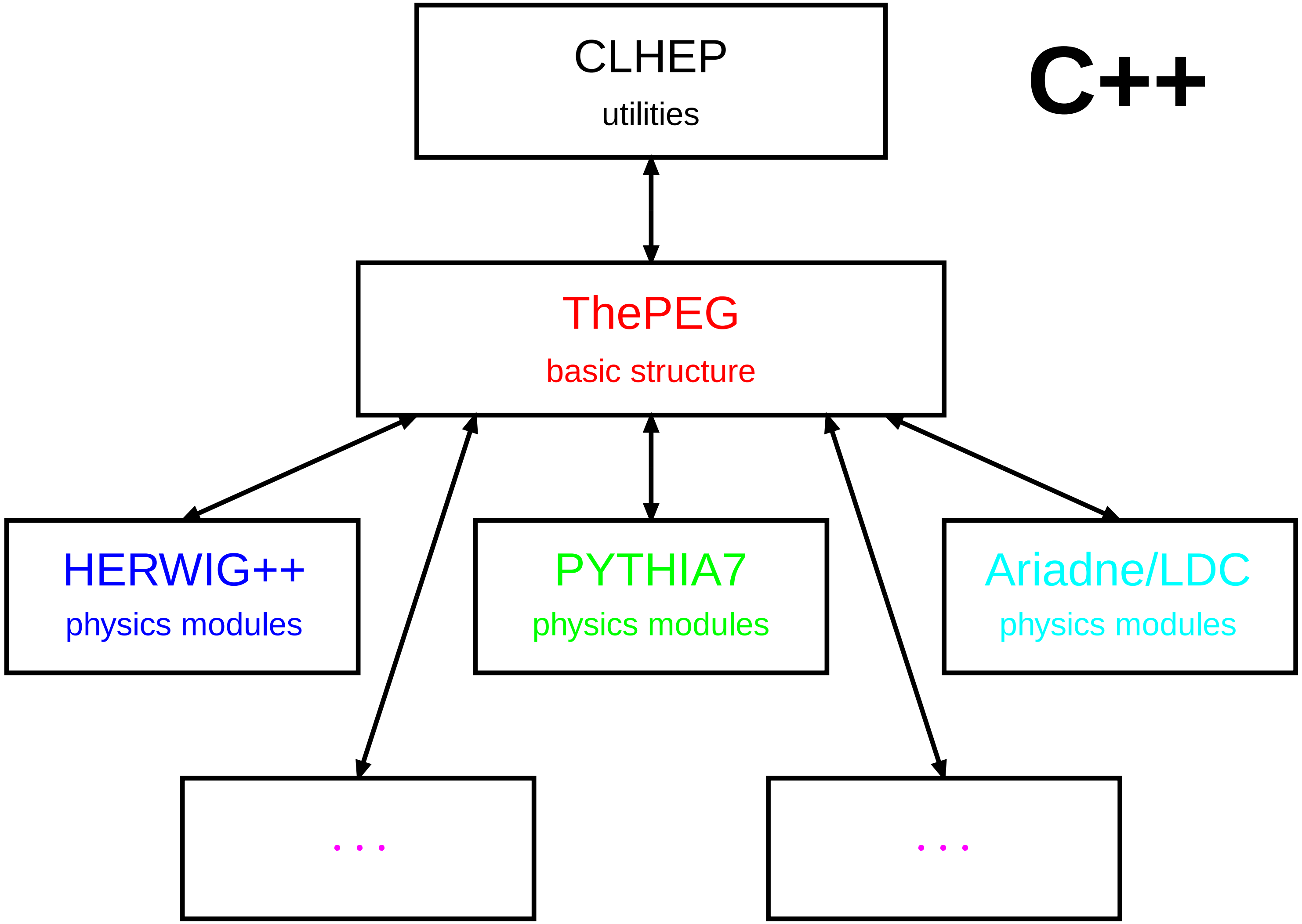
⇒ basic structure must be physics-neutral

PYTHIA7 → ThePEG: administration

PYTHIA7 → TheRest: physics

What is ThePEG?

Toolkit for High Energy Physics Event Generation



What is in TheRest?

- Processes: QCD $2 \to 2$, $e^+e^- \to q\overline{q}$ (LL)
- PDF: GRV 94 series(LL)
- Showers: initial- and final-state (old PYTHIA) (TS)
- Multiple interactions: none
- Beam remnants: Ariadne (LL)
- Fragmentation: simple string (Marc Bertini)
- + simple low-mass corrections (LL)
- Decays: most implemented (LL)

NOT useful for physics studies

Mainly simple pieces done $\Rightarrow$ almost all the hard work remains

- Conversion effort: everything takes longer and costs more
- The physics hurdle is as steep as the C++ learning curve
  $\Rightarrow$ not convenient to use postdocs
- Need continuity $\Rightarrow$ as above
- No work for graduate students

A fresh start

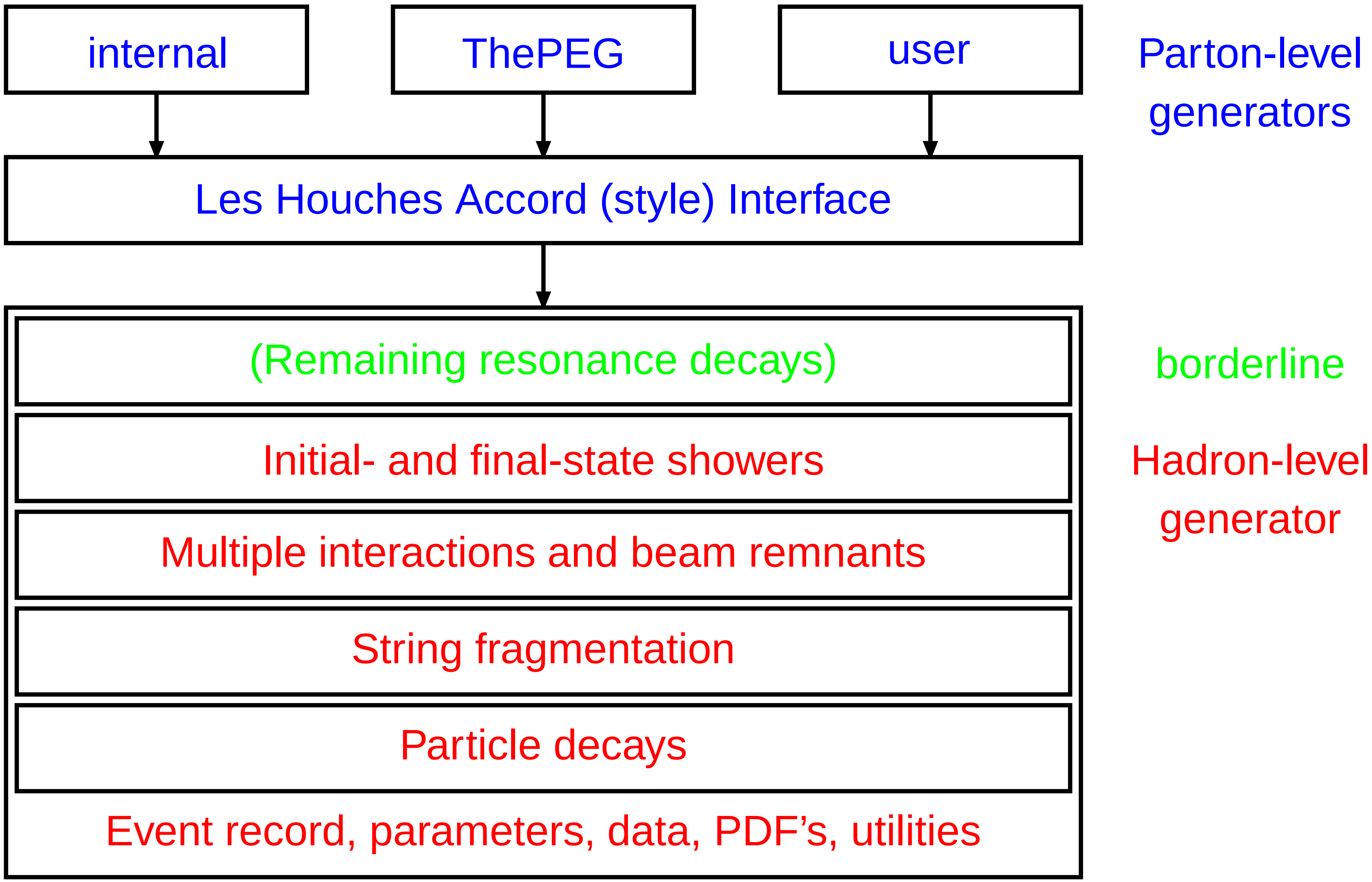
- Lower priority than teaching, administration, master's and graduate students, answering PYTHIA questions, maintaining Fortran code, . . .
- Leif's interests are ThePEG, Ariadne, LDC, . . .
- bad finances in Lund

Solution?: take a sabbatical and work "full-time"!
Sept. 2004 – 2007?: SFT/EP group
(SoFTware development for experiments)

Objectives:

- concentrate on physics, not administration
- pure standard C++, no fancy programming tricks
- independent of ThePEG (or anything else), but
- written to be modular, i.e. easy to interface
- interface to ThePEG later written by Leif(?)

New structure



Technical Notes (1)

Remaining resonance decays (Z, W, H, SUSY, . . .):

- internal or SUSY Les Houches Accord decay tables
- primitive angular correlations

Initial- and final-state showers:

- implement the $p_{\perp}$-ordered algorithms
- can use existing matrix-element matching code
- (introduce L-CKKW-style mixing, $p_{\perp}$-ordered)

Multiple interactions and beam remnants:

- based on new scheme under development

String fragmentation:

- reimplement baseline model, minor physics improvements
- low-mass strings
- junction topologies

Technical Notes (2)

Particle decays:

- update decay tables
- (Bose-Einstein; overlaps with fragmentation)

Event record, parameters, data, PDF's, utilities:

- PYTHIA-style event record with LHA colour tags
- integrated manual/parameters/data in XML?
- LHAPDF parton densities?
- simple event analysis (for debug)

Outside scope:

- γp/γ^*p/$\gamma\gamma$/$\gamma^*\gamma$/$\gamma^*\gamma^*$ physics
- colour reconnection (WW/ZZ)
- SUSY evolution (use SLHA!)
- old e^+e^- annihilation machinery ($\mathcal{O}(\alpha_s^2)$ ME's)
- independent fragmentation
- many out-of-use options

Summary

- Complexity of problem underestimated (C++ **&** physics)
- “Slave labour” not successful strategy (for me)

$\Longrightarrow$ PYTHIA7 (TheRest) nowhere near useful

Tentative schedule:

year	time	hadron-level	parton-level
0 =	fall 2004	begin new assault	
1 =	fall 2005	incomplete draft	LHA-style input
2 =	fall 2006	complete, buggy(?)	a few processes
3 =	fall 2007	stable, debugged	more processes

. . . but don’t forget Murphy’s law

Handlers

ThePEG defines a set of abstract `Handler` classes for
hard partonic sub-processes, parton densities, QCD cascades, hadronization, . . .

These handler classes interacts with the underlying structure using a special `Event Record` and a pre-defined set of `virtual` function definitions.

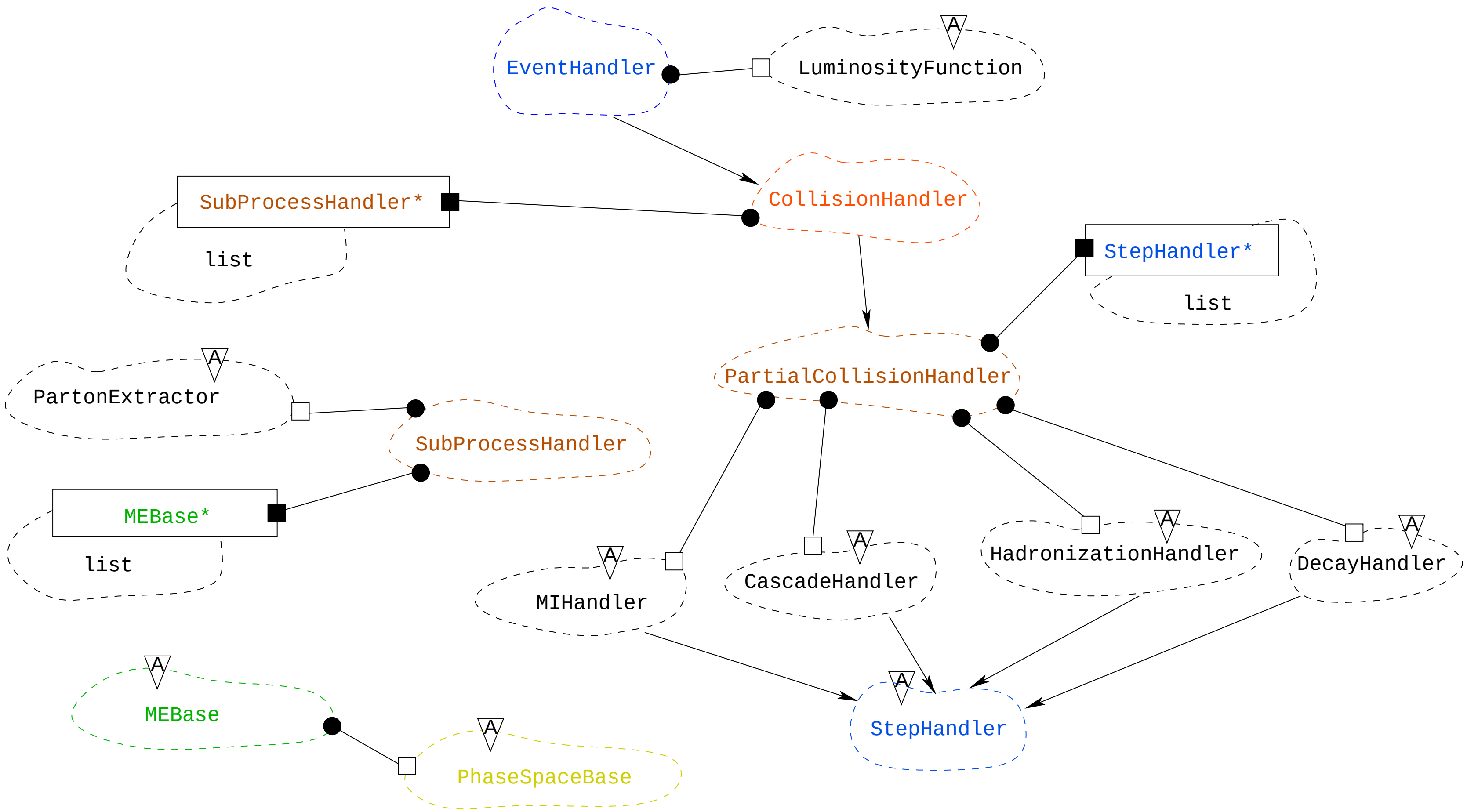
The procedure to implement e.g. a new hadronization model, is to write a new (C++) class *inheriting* from the abstract `HadronizationHandler` base class, implementing the relevant virtual functions.

The structure of the generation process is extremely dynamic:

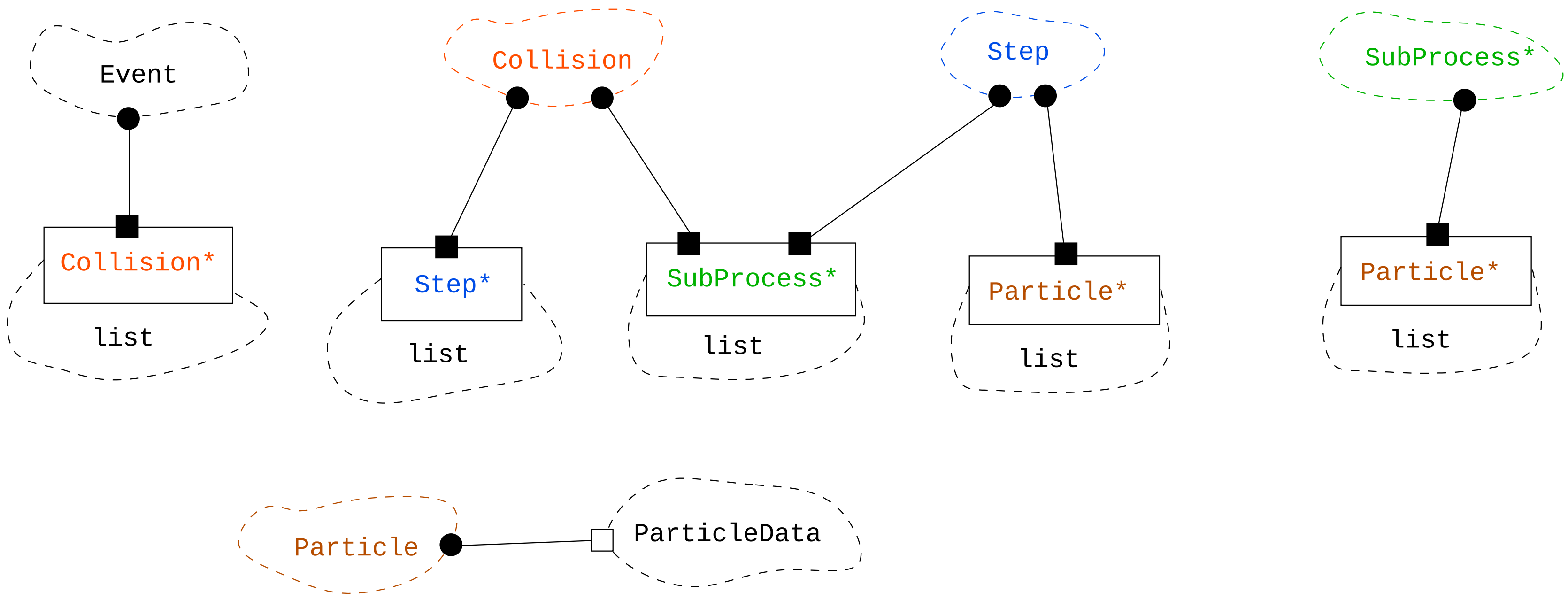
Besides the standard Handler classes, there is also a general `StepHandler` class which can do anything and can be inserted anywhere in the generation chain.

In addition, each handler can add steps in the generation chain or redo previous steps depending on the history of each event.

Class Structure of Handlers



Class Structure of an Event



The `Particle` class provides access to a lot of information. But it only has a pointer to a `ParticleData`, a `Lorentz5Momentum` and a pointer to another object carrying the rest of the information (colour, spin etc.) if needed.

Some of this information can be user-defined by creating classes inheriting from e.g. the `SpinBase` or the completely general `EventInfoBase` classes. This information can then be accessed through `dynamic_casting`.

Running ThePEG

The end-user will use a setup program to be able to pick objects corresponding to different physics models to build up an `EventGenerator` which then can be run interactively or off-line, or as a special slave program e.g. for Geant4.

The setup program is used to choose between a multitude of pre-defined generators, to modify parameters and options of the selected models and, optionally, to specify the analysis to be done on the generated events.

The `Repository` is the central part of the setup phase. It handles a structured list of all available objects and allows the user to manipulate them.

A flashy Graphical User Interface should be built on top of this `Repository`. Currently there is only a rudimentary command-line interpreter.

In the end of the run you will get a number of files with statistics and messages. And a LaTeX-file with references suitable for inclusion in an appendix of a paper.