Software Design Helix

Basis: Automata

- (abstracte) basisklasse FSM
  - DFA
  - NFA
    - eNFA
- sterk gebaseerd op theorie

```
int num_states;          // Q = [0, num_states[
std::set<char> sigma;
std::map<int, std::map<char, int>> d_data;
int q0;
std::set<int> F;
```

HELIX

Basis: Runner / Walker

- Runner (bevat code om Walker te gebruiken)
- Walker
  - Walker<DFA>
  - Walker<NFA>
  - Walker<eNFA>

```
eNFA N(...);
Runner r(N);
bool accepted = r.process(“test”);
```

HELIX

Suffix trees

- Idee:
  - Efficiente manier om patterns te zoeken
  - Pattern search met errors

- Design:
  - Pointerbased tree-structure
  - Opgebouwd via stringstreams
  - Gebruik van efficiente algoritmes

HELIX

Fase 2.0

- GUI gescheiden houden van de rest
- FASTA formaat (of toch een subset)

HELIX

Problemen

- Combinatie suffixtree en reguliere talen
- volledig FASTA formaat ondersteunen
  - probleem:
    - W = A, T of U
    - R = A of G
    - ...

HELIX