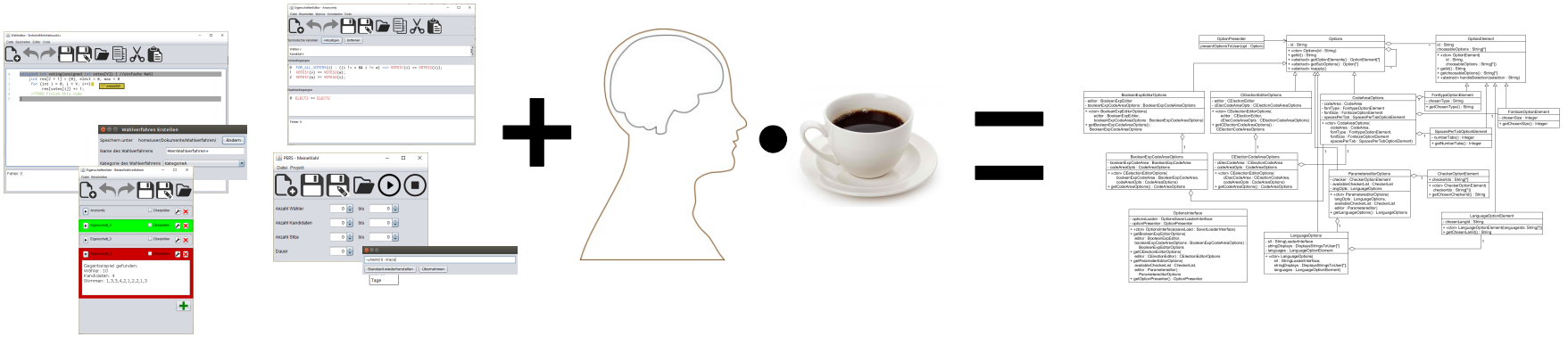
Karlsruher Institut für Technologie

Entwurf eines Softwaresystems zur Analyse formaler Eigenschaften von Wahlverfahren

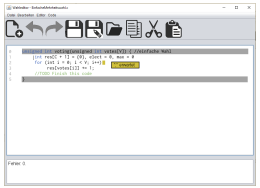
Hanselmann, Hecht, Klein, Schnell, Stapelbroek, Wohnig

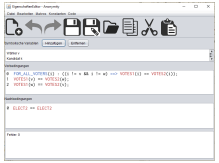
INSTITUT FÜR THEORETISCHE INFORMATIK – ANWENDUNGSORIENTIERTE FORMALE VERIFIKATION

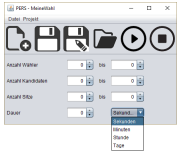


Anforderungen

- **Texteditor** für Beschreibung von **Wahlverfahren** in **C**
- **Texteditor** für Beschreibung **boolscher** Ausdrücke
- Editor für **Parameter**
- **Überprüfung** via **CBMC**
- **Aufbereiten** und **Darstellen** des **Ergebnisses**
- Zielgruppe: Nicht-Informatiker





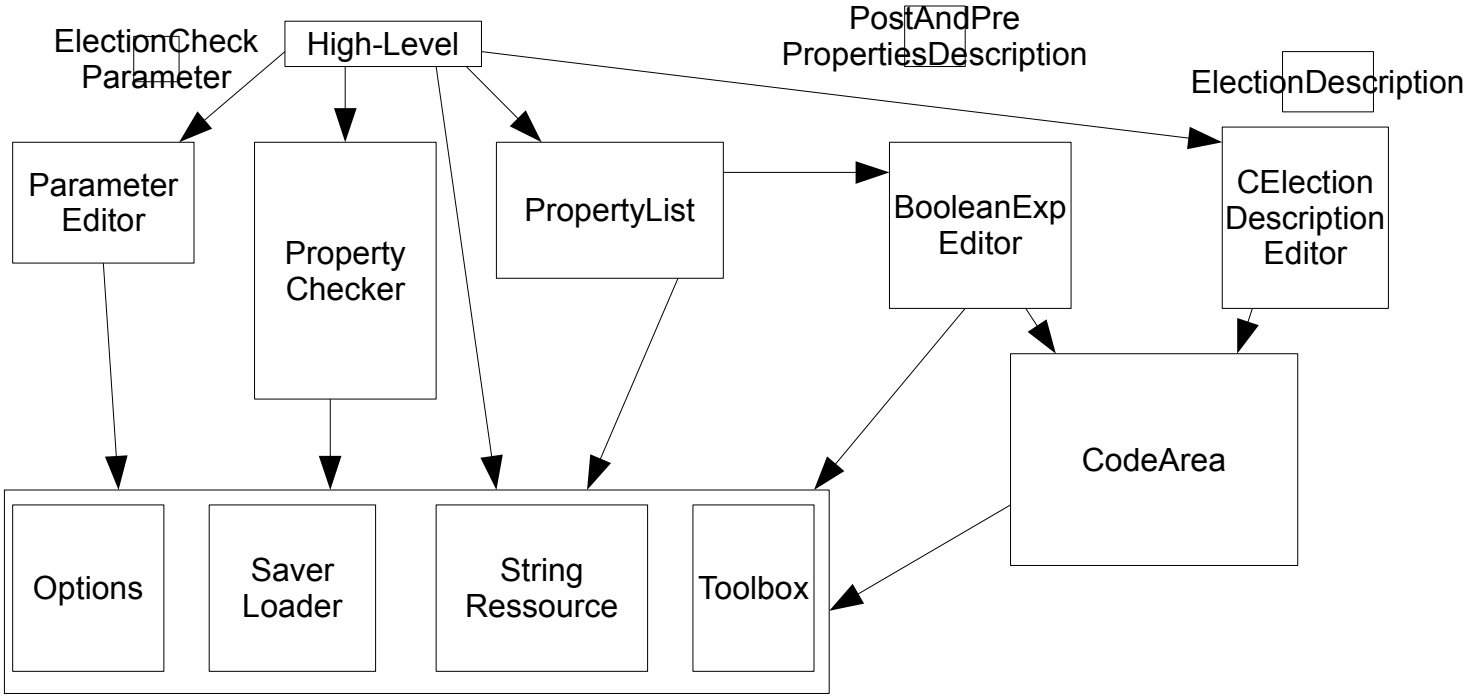


Muss- und Soll-Anforderungen – alle Module

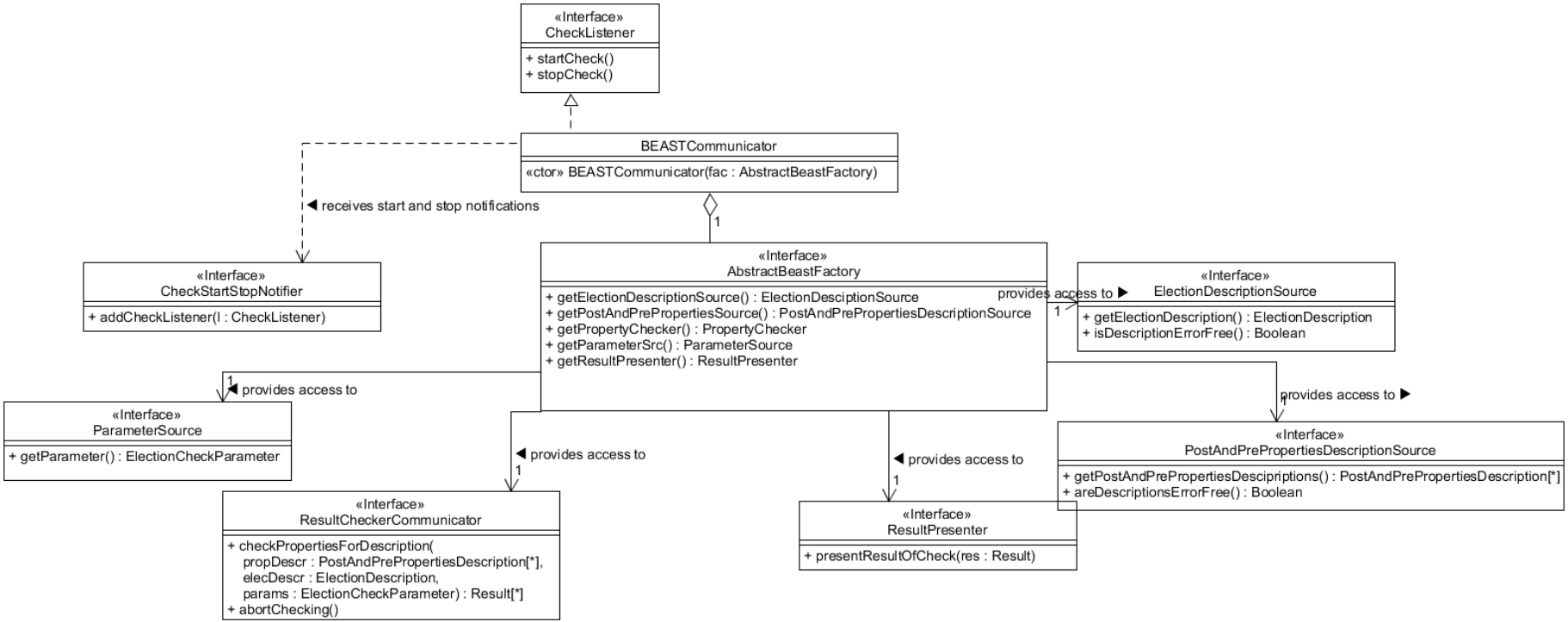
- Aufbau der Objekte bei Systemstart
- Menü, Toolbar (& Shortcuts): Eingabe → Aktion
- Speichern/Laden der Datentypen (Persistenz)
- Optionen
- Anzeigen Strings: verschiedene Sprachen
- Hängen zusammen!

Übersicht

- Fassaden-Muster

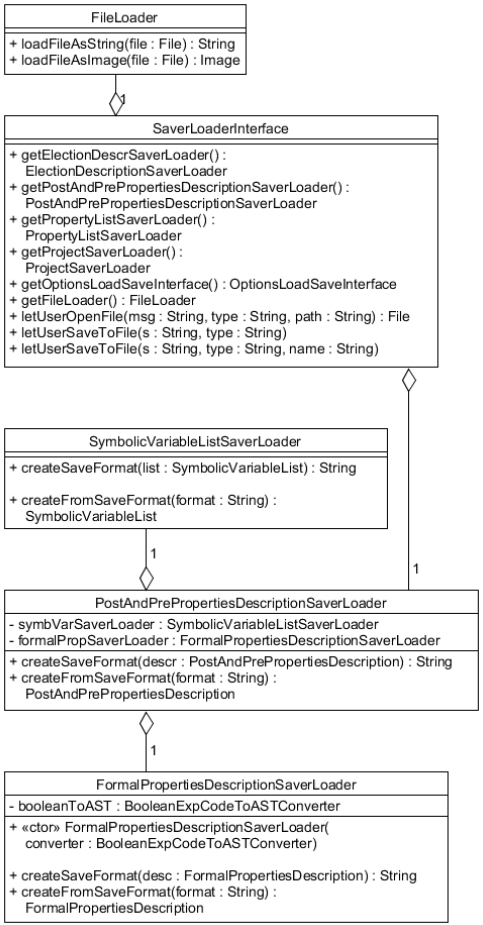


High-Level


«Interface»
CheckListener
+ startCheck()
+ stopCheck()
BEASTCommunicator
«ctor» BEASTCommunicator(fac : AbstractBeastFactory)
receives start and stop notifications
1
«Interface»
CheckStartStopNotifier
+ addCheckListener(l : CheckListener)
«Interface»
AbstractBeastFactory
+ getElectionDescriptionSource() : ElectionDesciptionSource
+ getPostAndPrePropertiesSource() : PostAndPrePropertiesDescriptionSource
+ getPropertyChecker() : PropertyChecker
+ getParameterSrc() : ParameterSource
+ getResultPresenter() : ResultPresenter
provides access to
1
«Interface»
ElectionDescriptionSource
+ getElectionDescription() : ElectionDescription
+ isDescriptionErrorFree() : Boolean
1
provides access to
«Interface»
ParameterSource
+ getParameter() : ElectionCheckParameter
provides access to
1
«Interface»
ResultCheckerCommunicator
+ checkPropertiesForDescription(
propDescr : PostAndPrePropertiesDescription[*],
elecDescr : ElectionDescription,
params : ElectionCheckParameter) : Result[*]
+ abortChecking()
provides access to
1
«Interface»
ResultPresenter
+ presentResultOfCheck(res : Result)
provides access to
«Interface»
PostAndPrePropertiesDescriptionSource
+ getPostAndPrePropertiesDescipriptions() : PostAndPrePropertiesDescription[*]
+ areDescriptionsErrorFree() : Boolean

Persistenz - SaverLoader

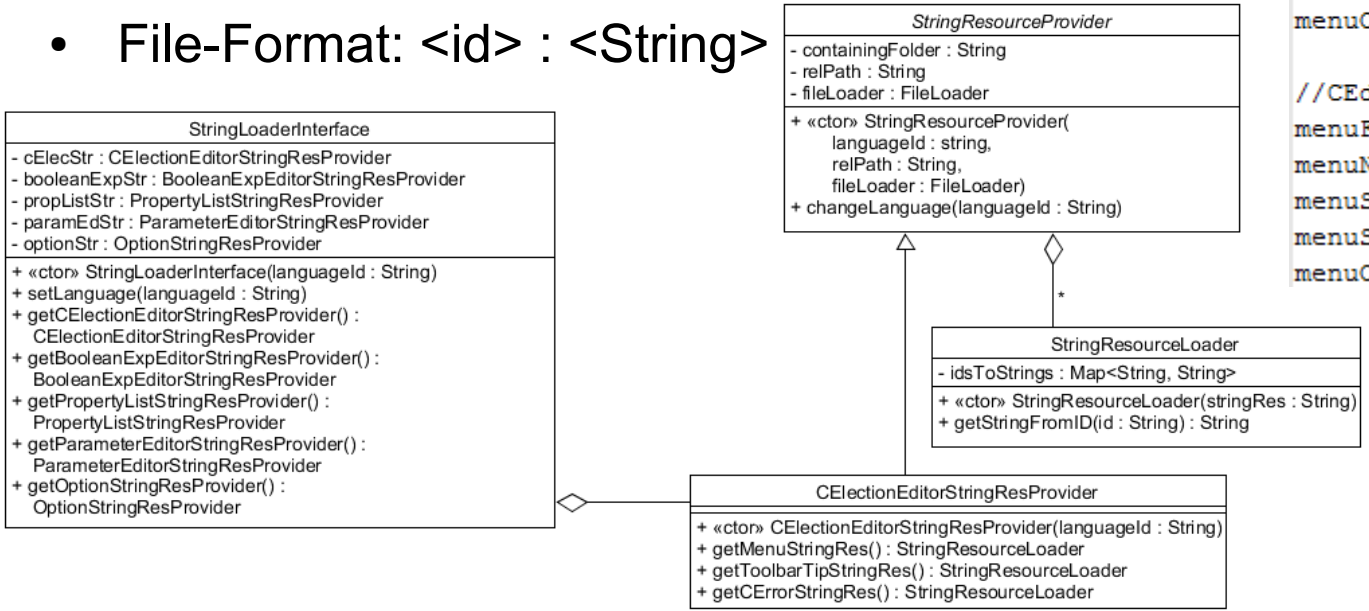
- Datentypen alle POD
- Modifiziertes Memento-Muster
- Pro Datentyp eine SaverLoader Klasse
  - Hierarchie der Datentypen nachvollzogen
- Zustand vom Typ String
- Alternative: Vererbung
- Vorteil: Trennung der Verantwortung



Verschiedene Sprachen – StringRessource

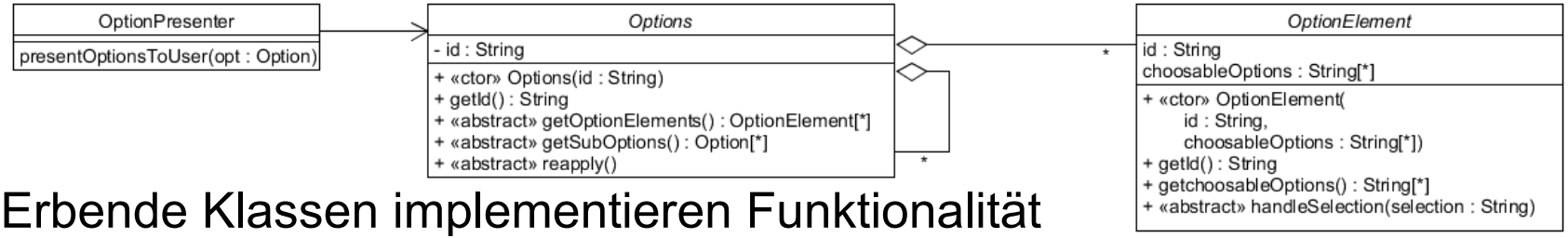
- Alle angezeigten Strings von Speicher geladen
- Pro Sprache ein File
- File-Format: <id> : <String>

```
//CEditorStr_ger
menuFile: Datei
menuNew : Neu
menuSave : Speichern
menuSaveAs : Speichern unter
menuOpen : Öffnen

//CEditorStr_en
menuFile: File
menuNew : New
menuSave : Save
menuSaveAs : Save as
menuOpen : Öffnen
```

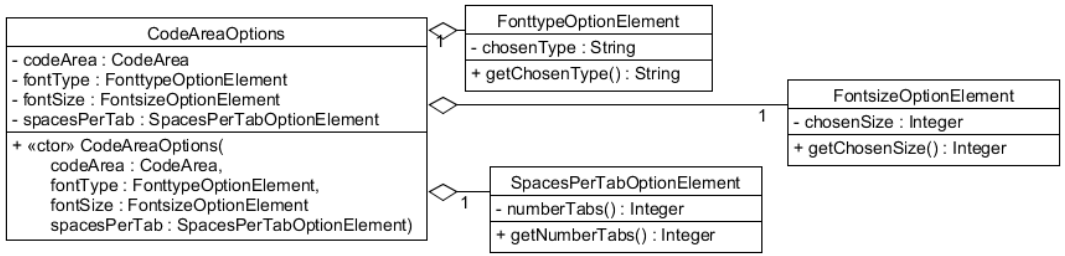


Optionen

- Pro modifizierbarer Klasse eine Options-Klasse
- Präsentieren/Verändern der Optionen verallgemeinert
- Kompositums-Muster

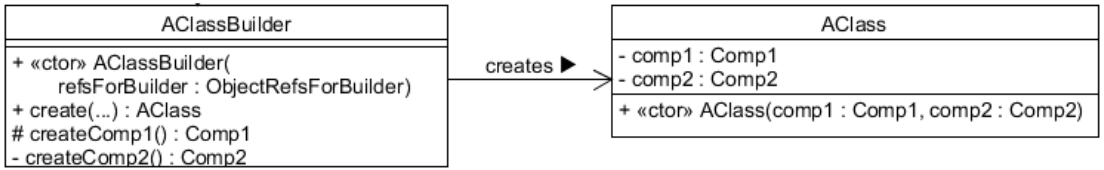


- Erbende Klassen implementieren Funktionalität



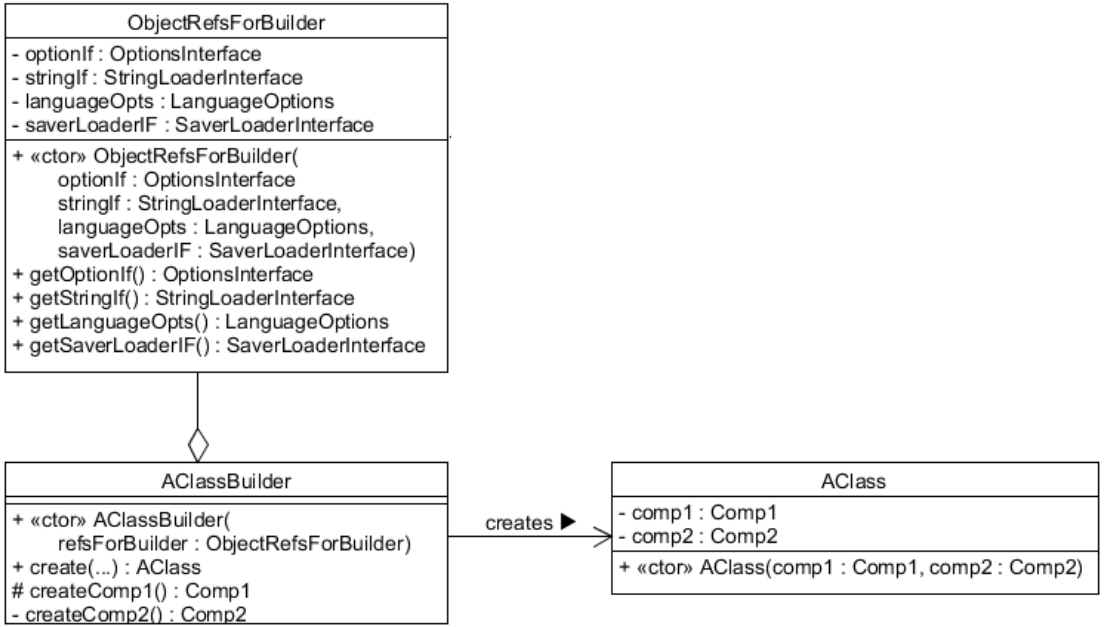
Aufbau von Objekten & Eingabe → Funktion & Sprachoptionen

- Erbauer-Muster



Aufbau von Objekten & Eingabe → Funktion & Sprachoptionen

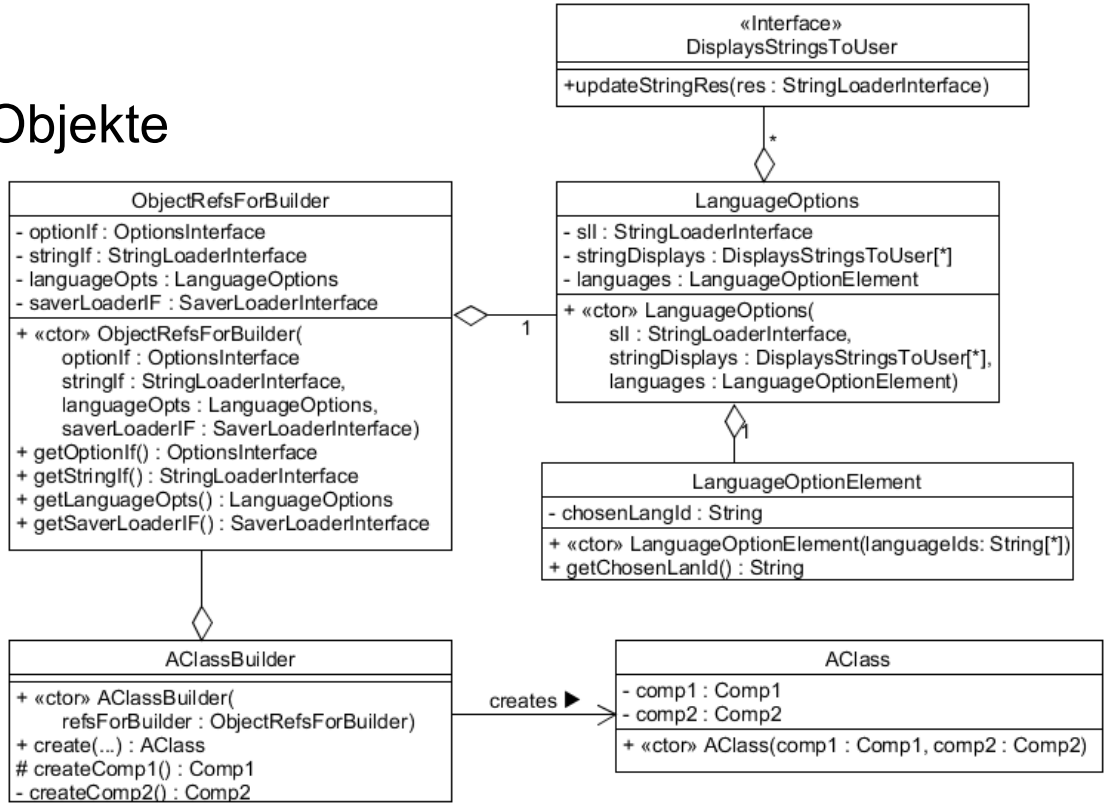
- Erbauer-Muster
- Bekommt notwendige Objekte



Aufbau von Objekten & Eingabe → Funktion & Sprachoptionen

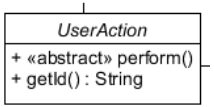
- Erbauer-Muster
- Bekommt notwendige Objekte
- Verlinkt

  DisplaysStringsToUser



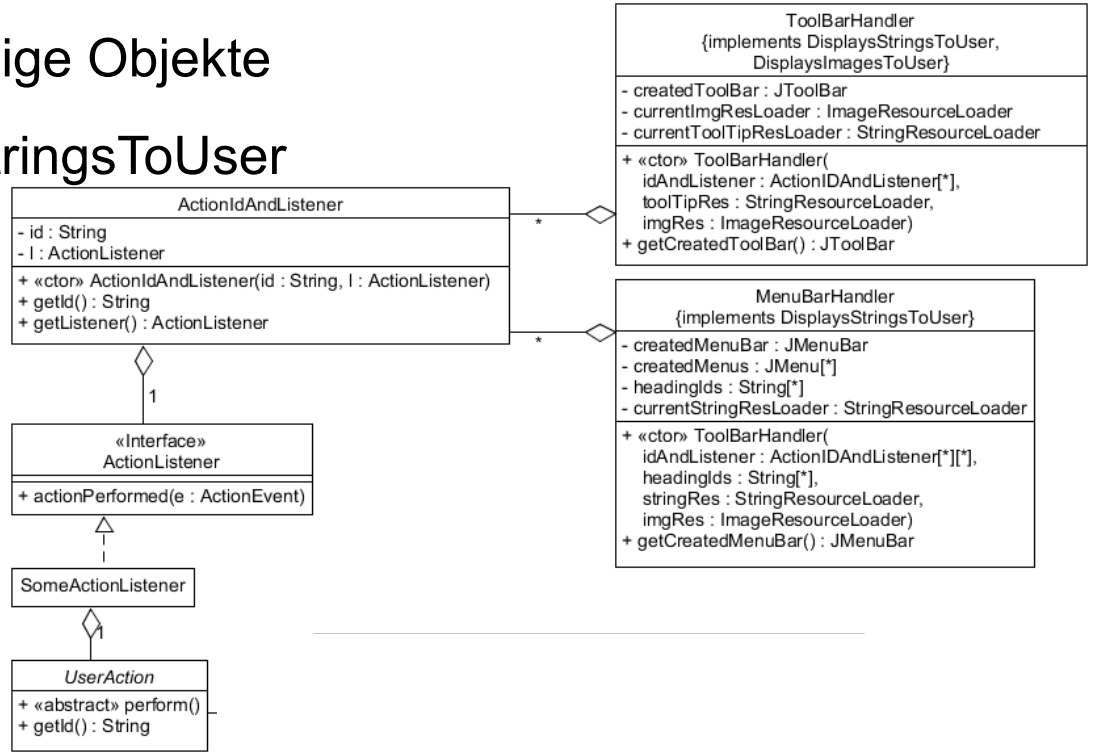
Aufbau von Objekten & Eingabe → Funktion & Sprachoptionen

- Erbauer-Muster
- Bekommt notwendige Objekte
- Verlinkt DisplaysStringsToUser
- Erzeugt Aktionen



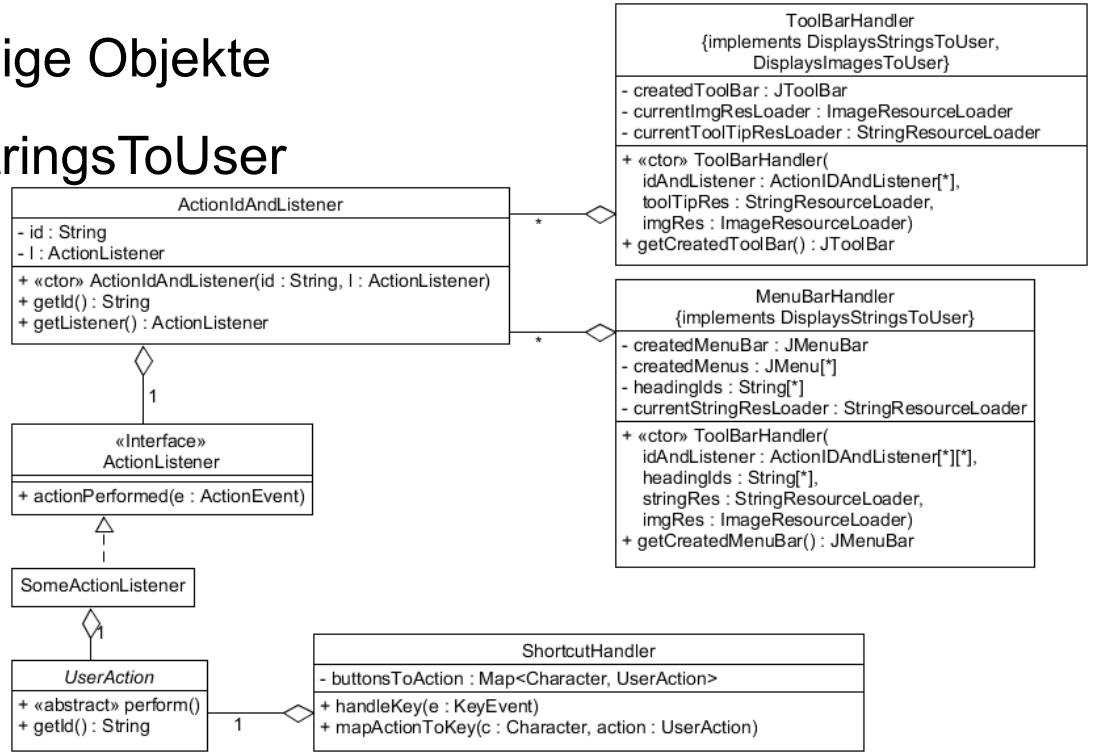
Aufbau von Objekten & Eingabe → Funktion & Sprachoptionen

- Erbauer-Muster
- Bekommt notwendige Objekte
- Verlinkt DisplaysStringsToUser
- Erzeugt Aktionen
- Verlinkt Aktionen



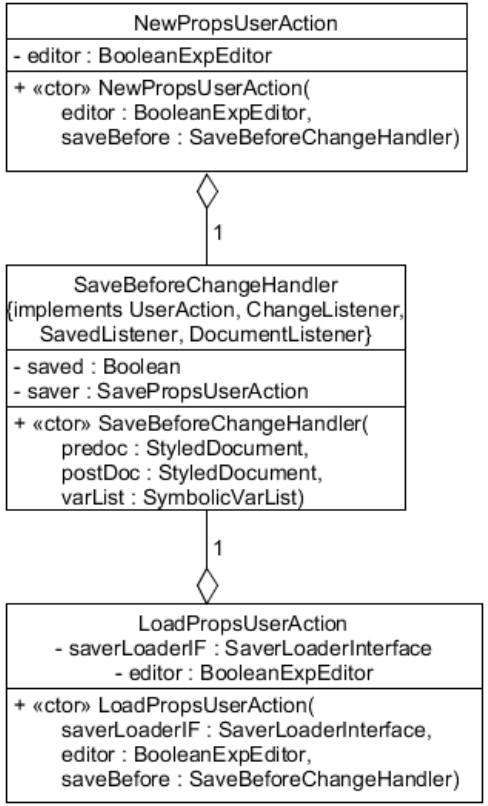
Aufbau von Objekten & Eingabe → Funktion & Sprachoptionen

- Erbauer-Muster
- Bekommt notwendige Objekte
- Verlinkt DisplaysStringsToUser
- Erzeugt Aktionen
- Verlinkt Aktionen



Aufbau von Objekten & Eingabe → Funktion & Sprachoptionen

- Erbauer-Muster
- Bekommt notwendige Objekte
- Verlinkt DisplaysStringsToUser
- Erzeugt Aktionen
- Verlinkt Aktionen
- Pro Aktion eine Klasse

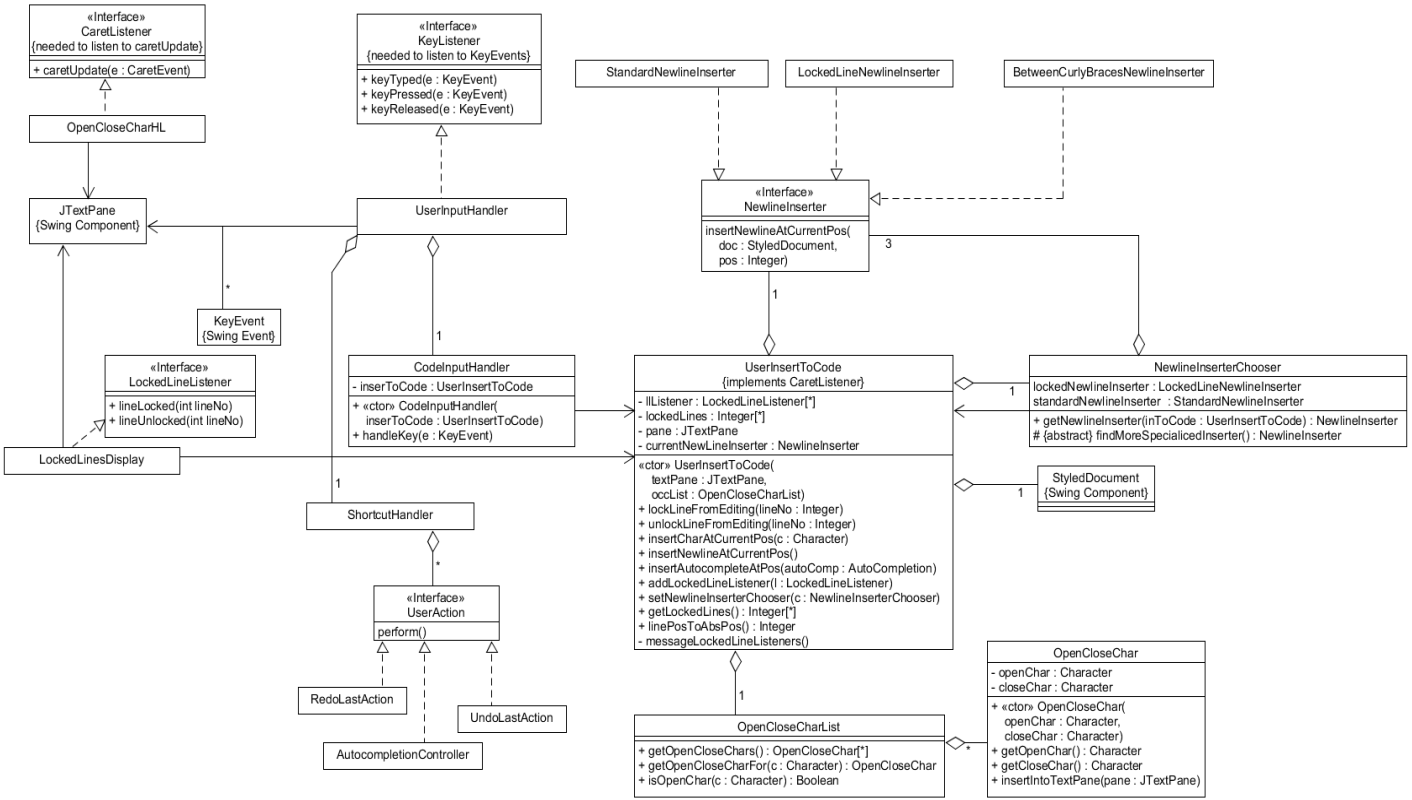


CodeArea

- Ein- & Ausgabe von Code
  - Schließen von Klammern etc.
- Finden & Anzeigen von Fehlern
- Rückgängig machen & Wiederholen
- Syntax Highlighting
- Erweiterbar und Spezialisierbar für Texteditoren

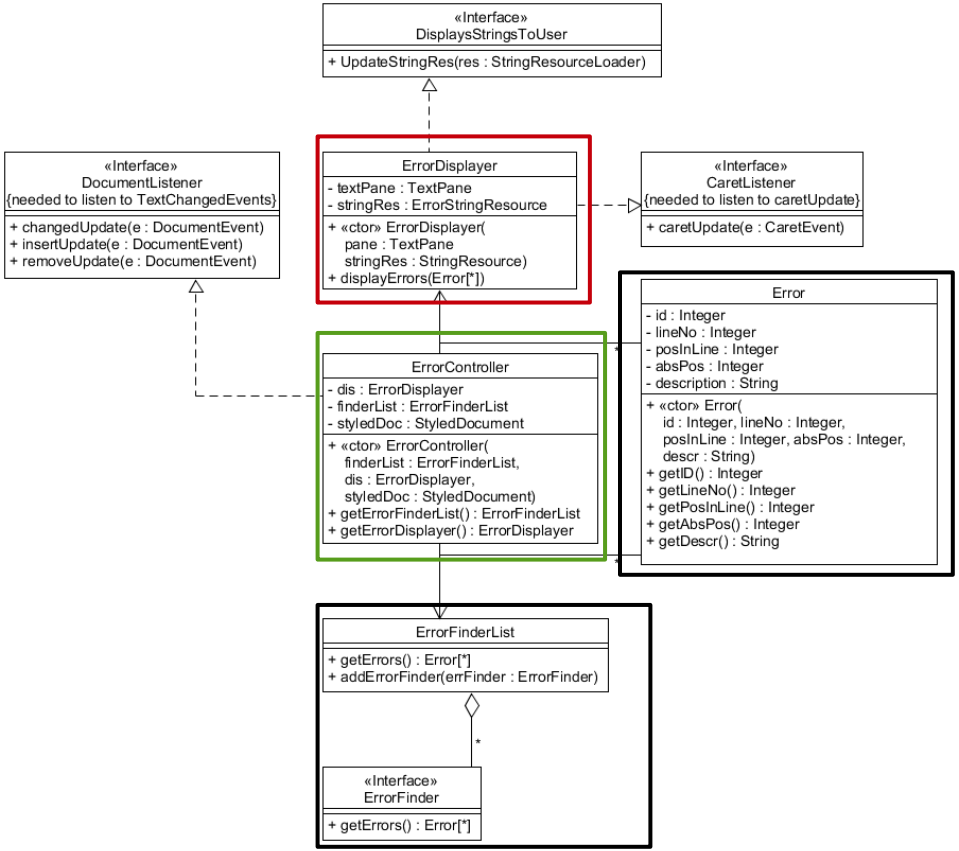
CodeArea: Ein- & Ausgabe von Code

- Modifiziertes MVC



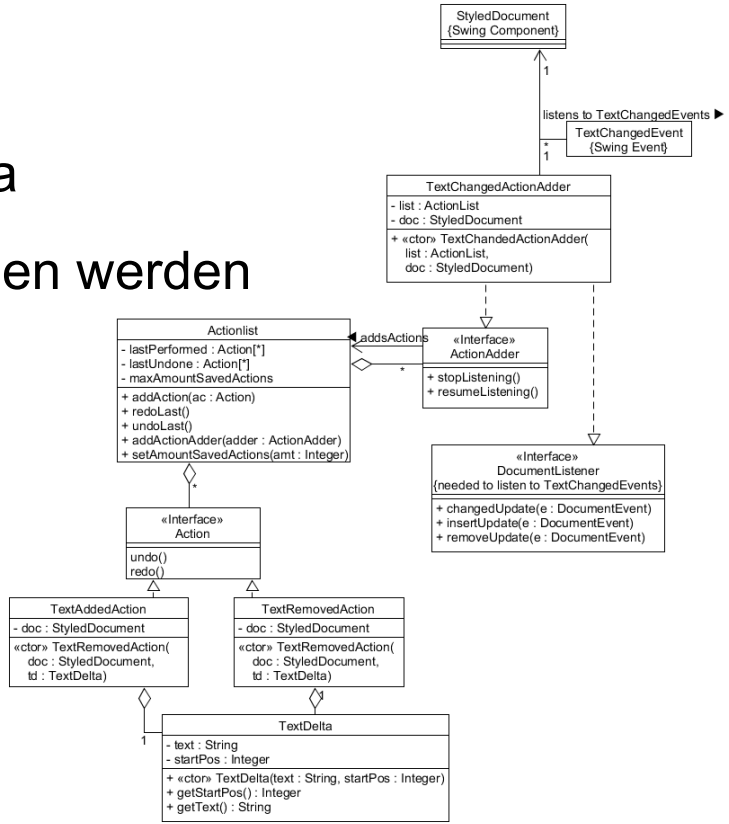
CodeArea: Finden & Anzeigen von Fehlern

- MVC



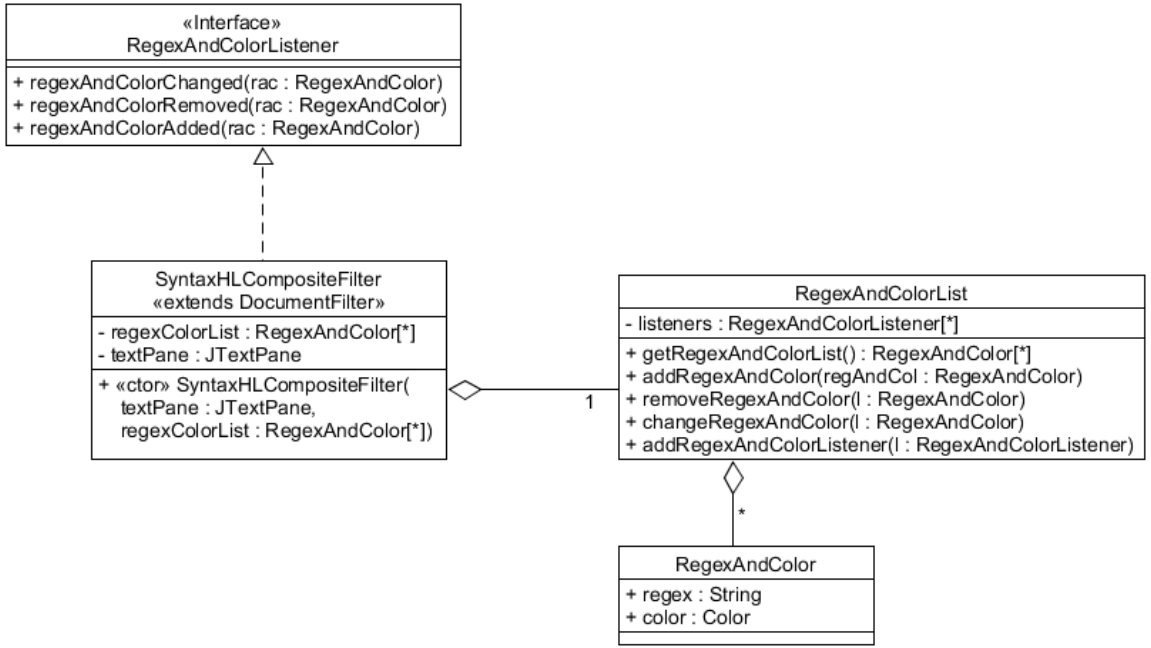
CodeArea: Rückgängig machen & Wiederholen

- Pro Aktion eine Klasse
- Textänderung durch TextDelta
- ActionAdder kann unterbrochen werden



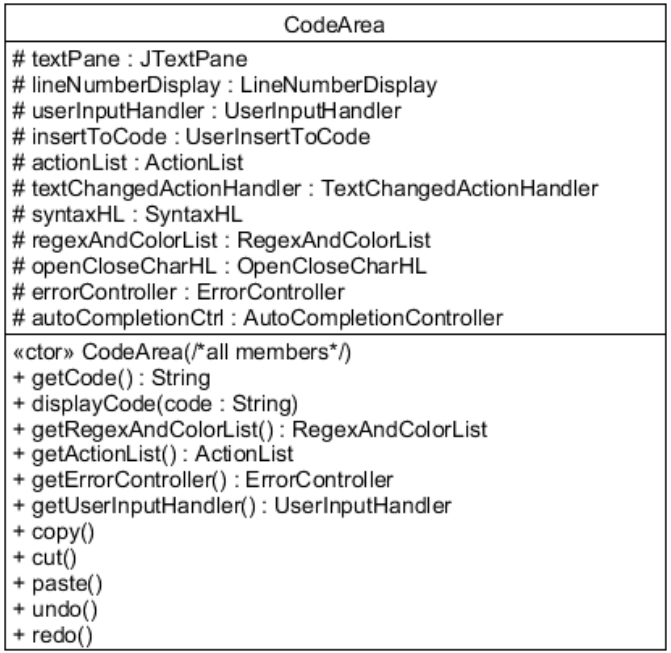
CodeArea: Syntax Highlighting

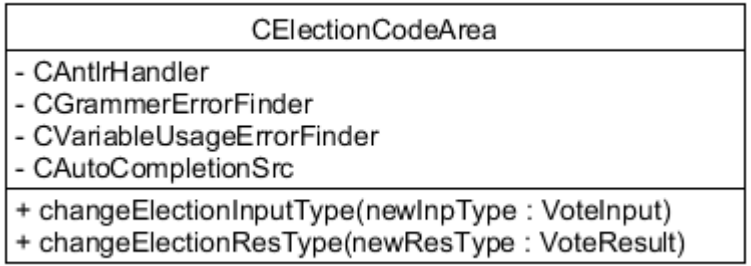
- DocumentFilter
- Regex und Farbe

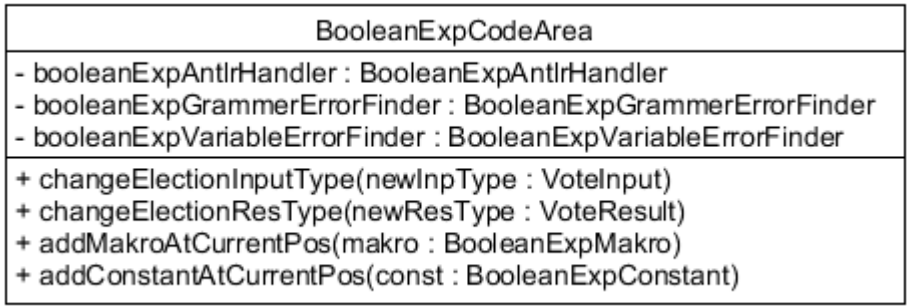


CodeArea: Erweiterbar und Spezialisierbar für Texteditoren

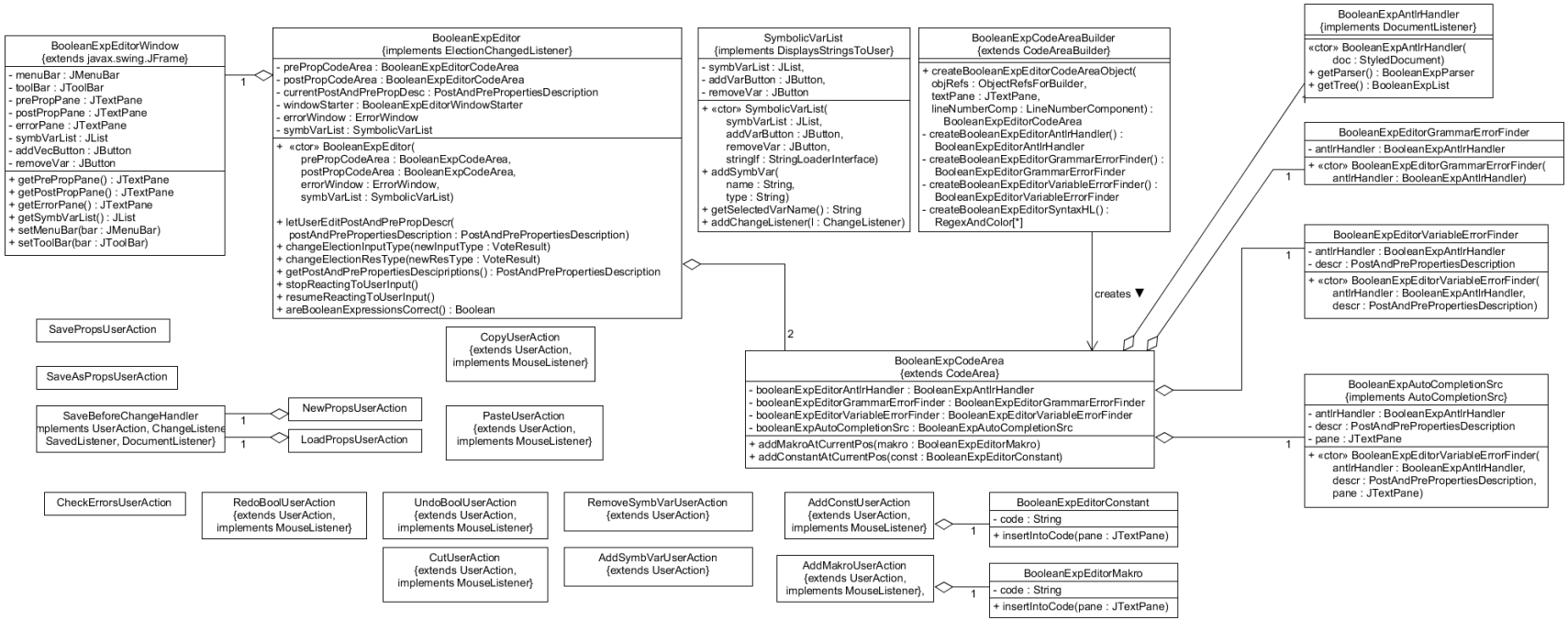
- Beerben von CodeArea und Implementieren von Aktionen







CodeArea: Erweiterbar und Spezialisierbar für Texteditoren



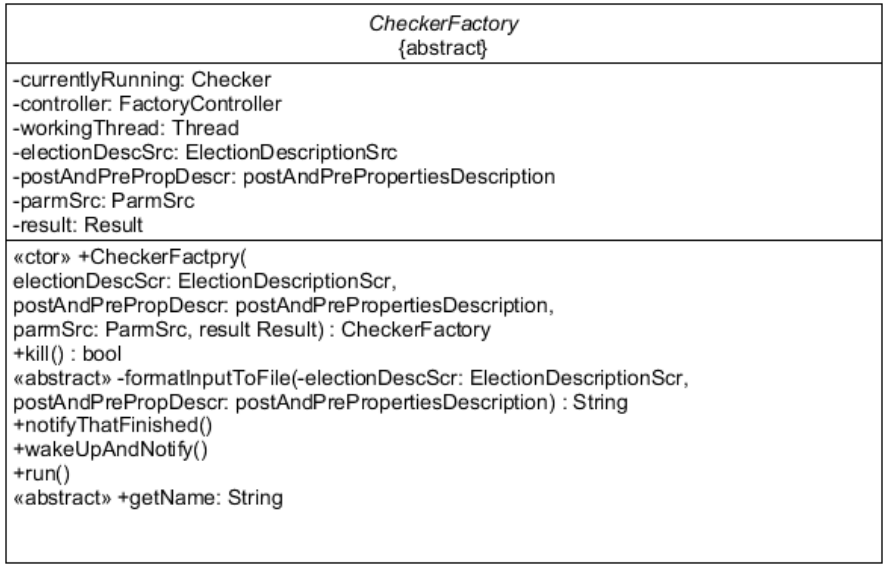
Parametereditor und Eigenschaftenliste

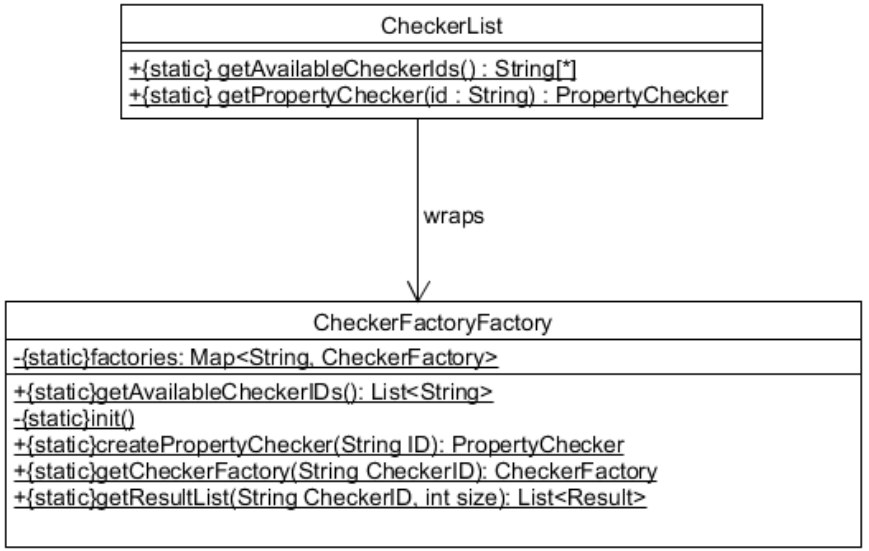
- Selbes Prinzip!
- Andere Interfaces & Aktionen

PropertyChecker

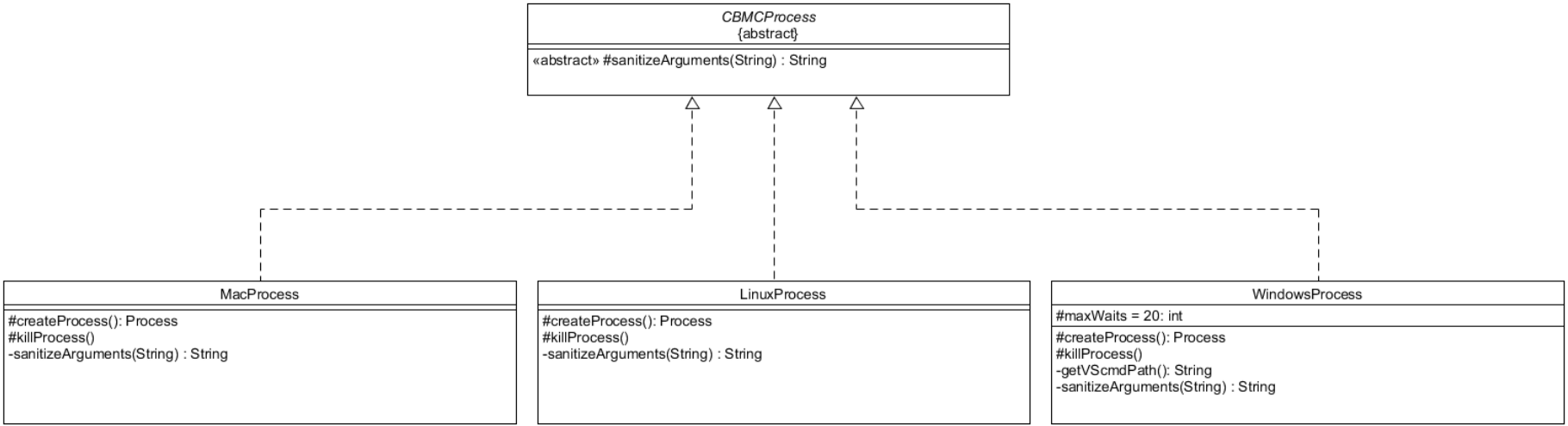
- Verschiedene Checker
- Verschiedene Betriebssysteme
- Mehrere Prozesse zur gleichen Zeit (?)

PropertyChecker: Verschiedene Checker



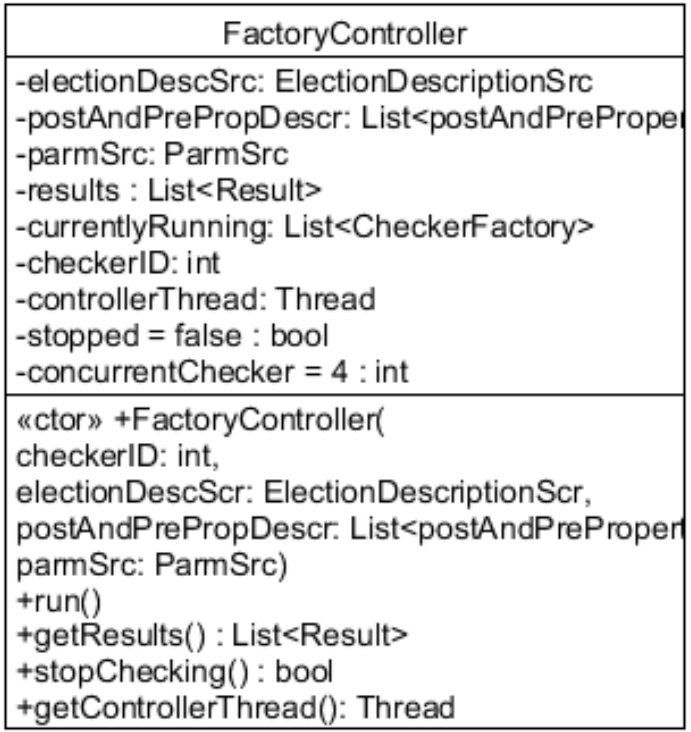


PropertyChecker: Verschiedene Betriebssysteme

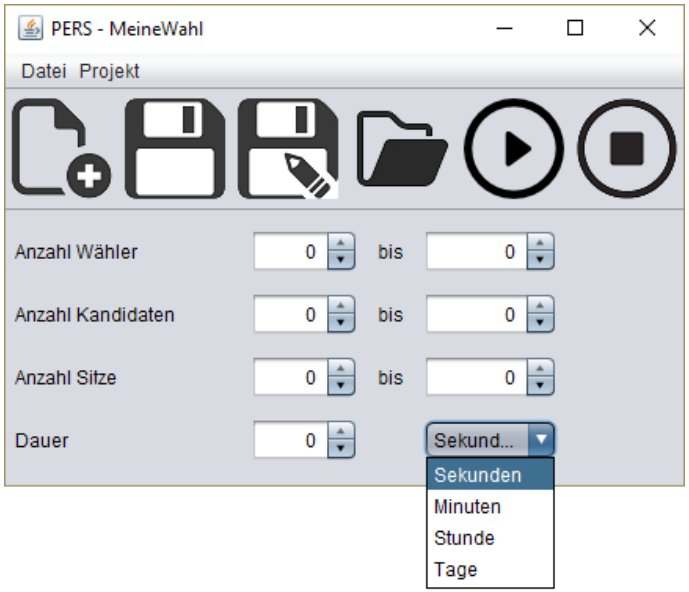


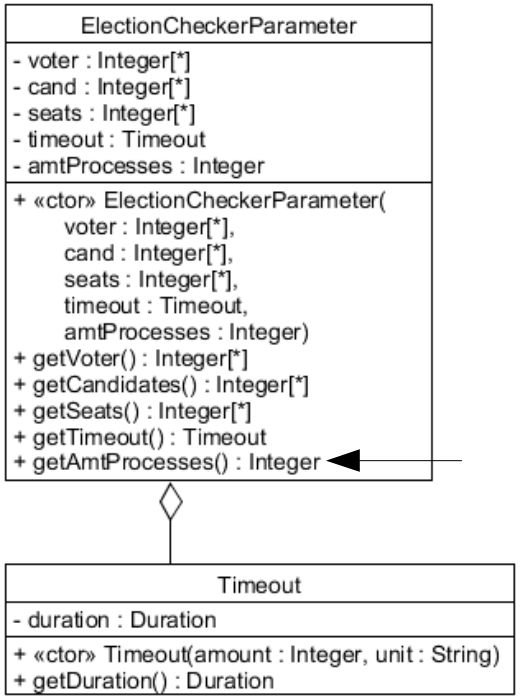
PropertyChecker: Mehrere Prozesse zur gleichen Zeit

- Kontrolliert CheckerFactory
- Diese gibt mehrere Checker
- Diese starten einen Prozess



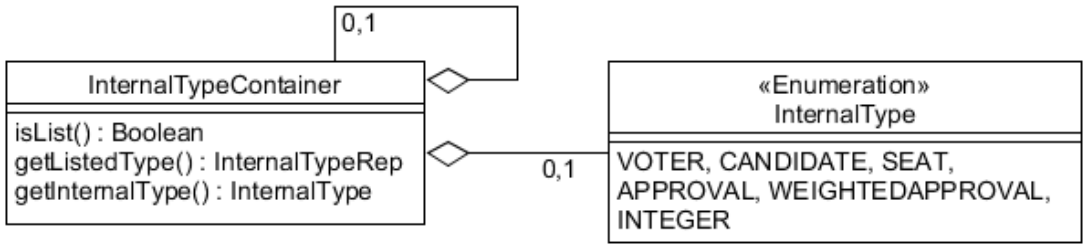
Datentypen: Parameter für CBMC

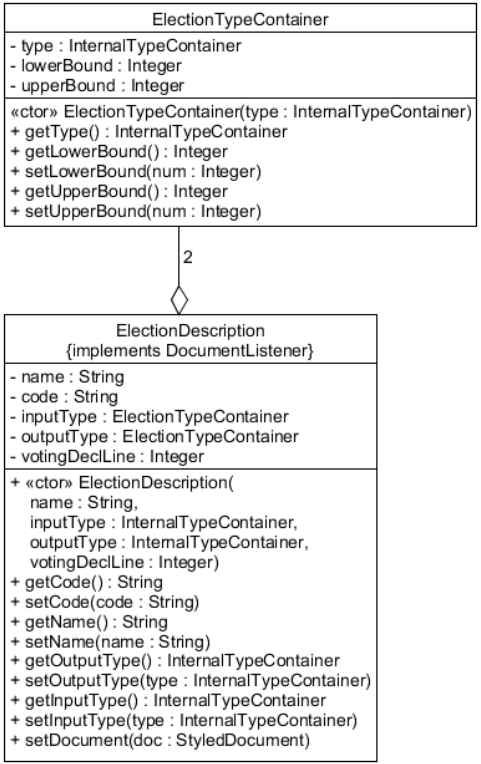




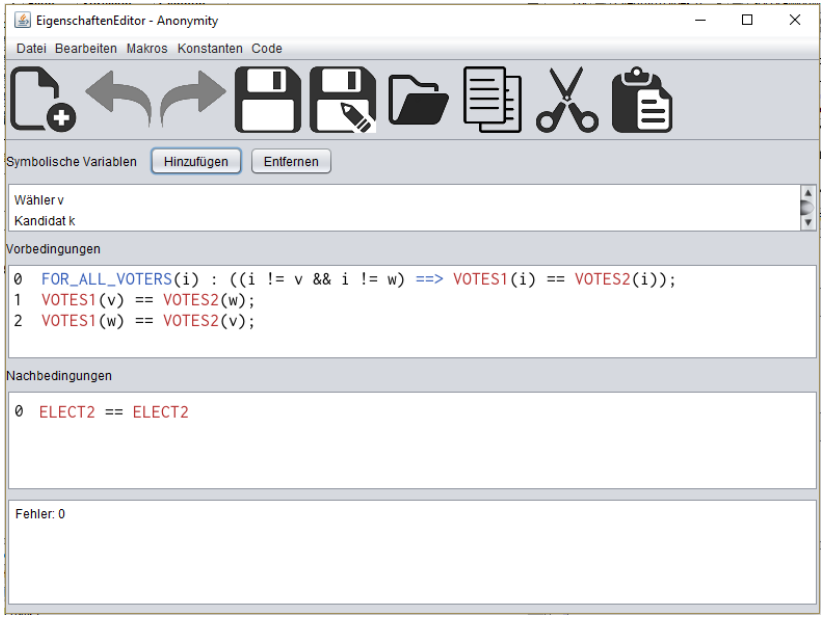
Datentypen: Beschreibung der Wahlverfahren

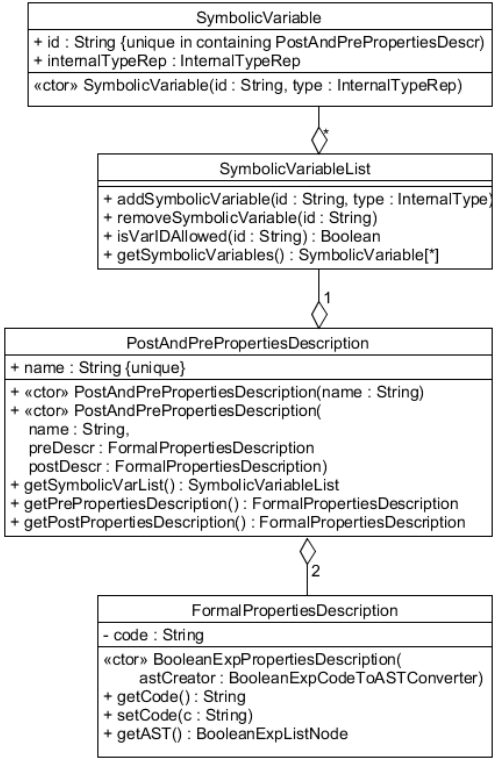
```c
/*
* This function describes the voting procedure
* input: Single-Choice: Every voter picks one candidate
* result: One candidate or a tie
*/
unsigned int voting(unsigned int votes[V]) {
    ...
}

/*
* This function describes the voting procedure
* input: weighted-Choice: Every voter rates every Candidate with an Integer
* result: amount of Seats allocated per party
*/
unsigned int * voting(unsigned int votes[V][C]) {
    ...
}
```



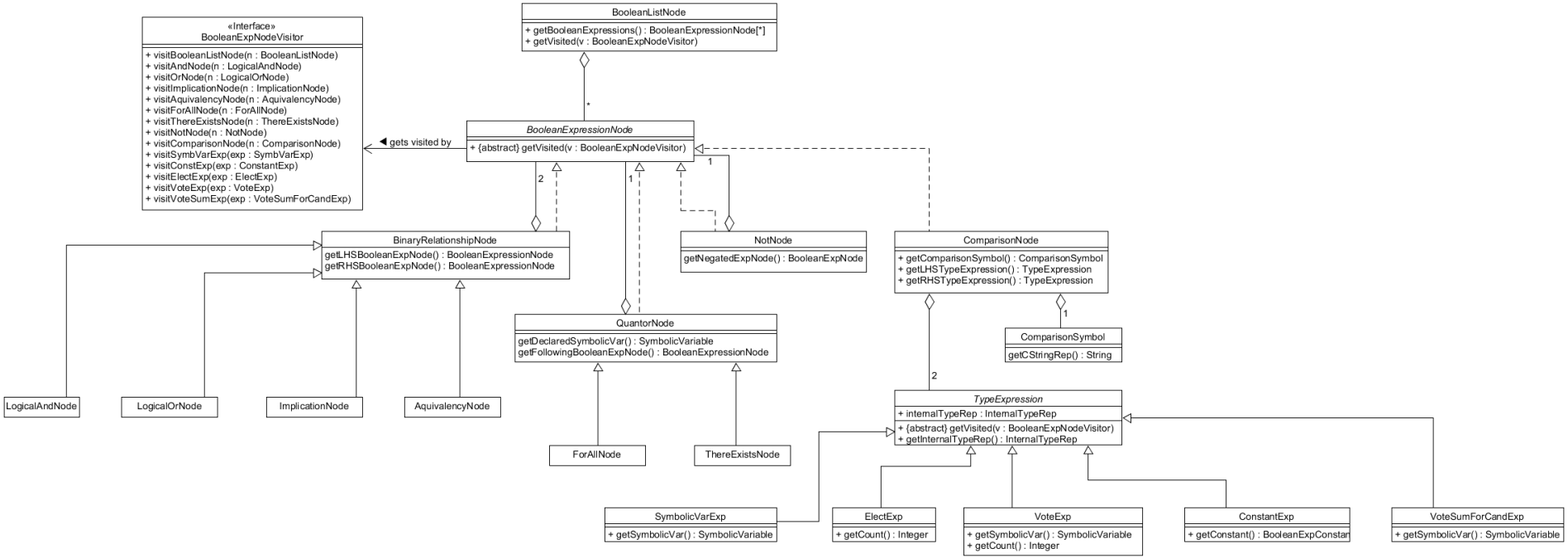


Datentypen: Beschreibung der formalen Eigenschaften





Datentypen: AST-Darstellung boolscher Ausdrücke



Erstellung des Codes für CBMC

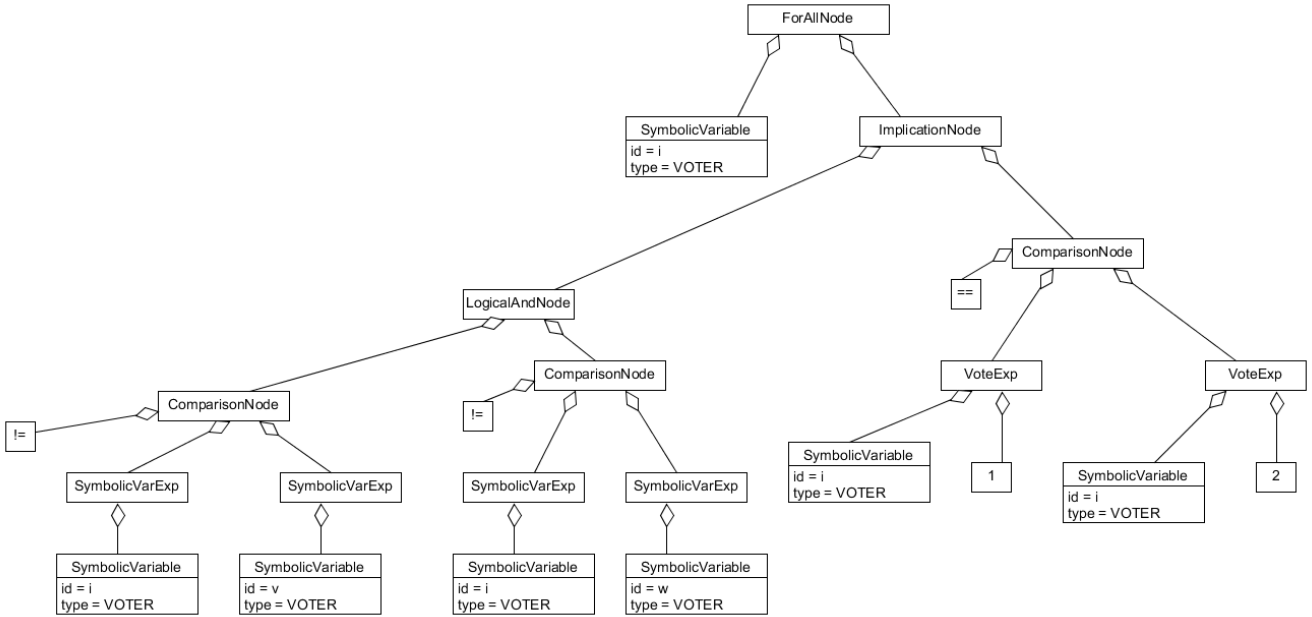
- Eingabe: 2 * BooleanListNode + CElectionDescription
- Ergebnis: C Code für CBMC
- Implementiert BooleanExpNodeVisitor
- Vor- und Nachbedingungen analog
  - assume oder assert
- Pro BooleanExpNode eine boolsche Variable in Kellerspeicher

Erstellung des Codes für CBMC: Beispiel

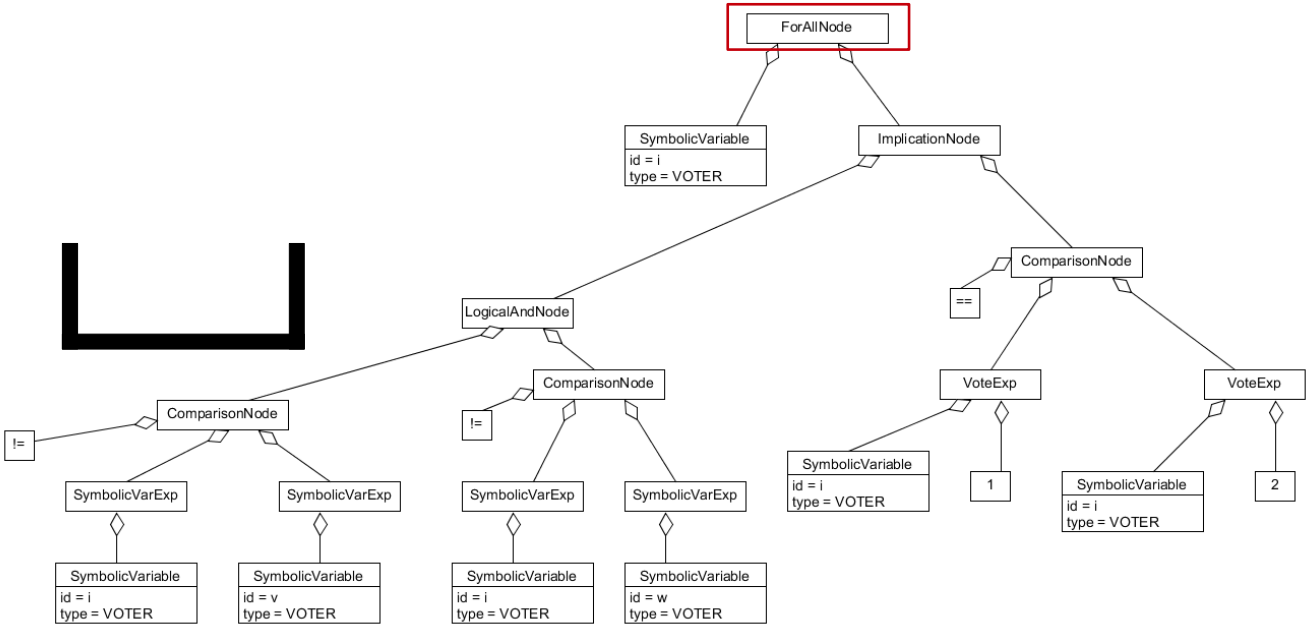
```
/*
* This function describes the voting procedure
* input: Single-Choice: Every voter picks one candidate
* result: One candidate or a tie
*/
unsigned int voting(unsigned int votes[V]) {
    ...
}
```

Vorbedingungen

```
0  FOR_ALL_VOTERS(i) : ((i != v && i != w) ==> VOTES1(i) == VOTES2(i));
1  VOTES1(v) == VOTES2(w);
2  VOTES1(w) == VOTES2(v);
```

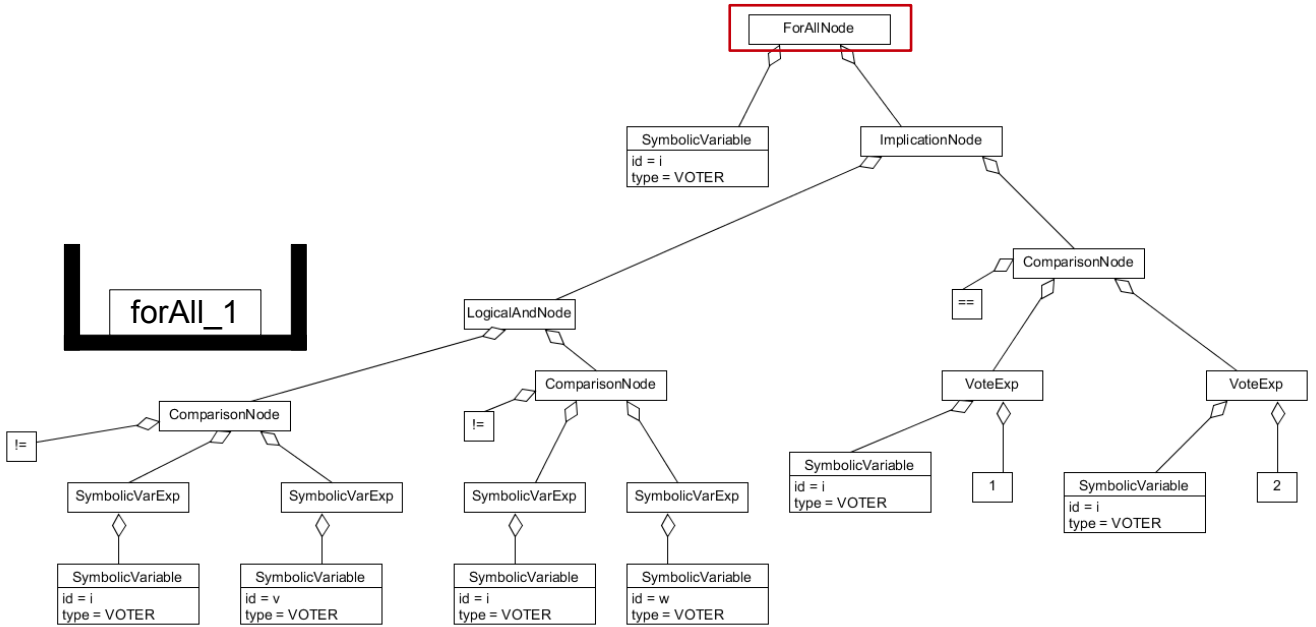


Generierung des Codes für CBMC



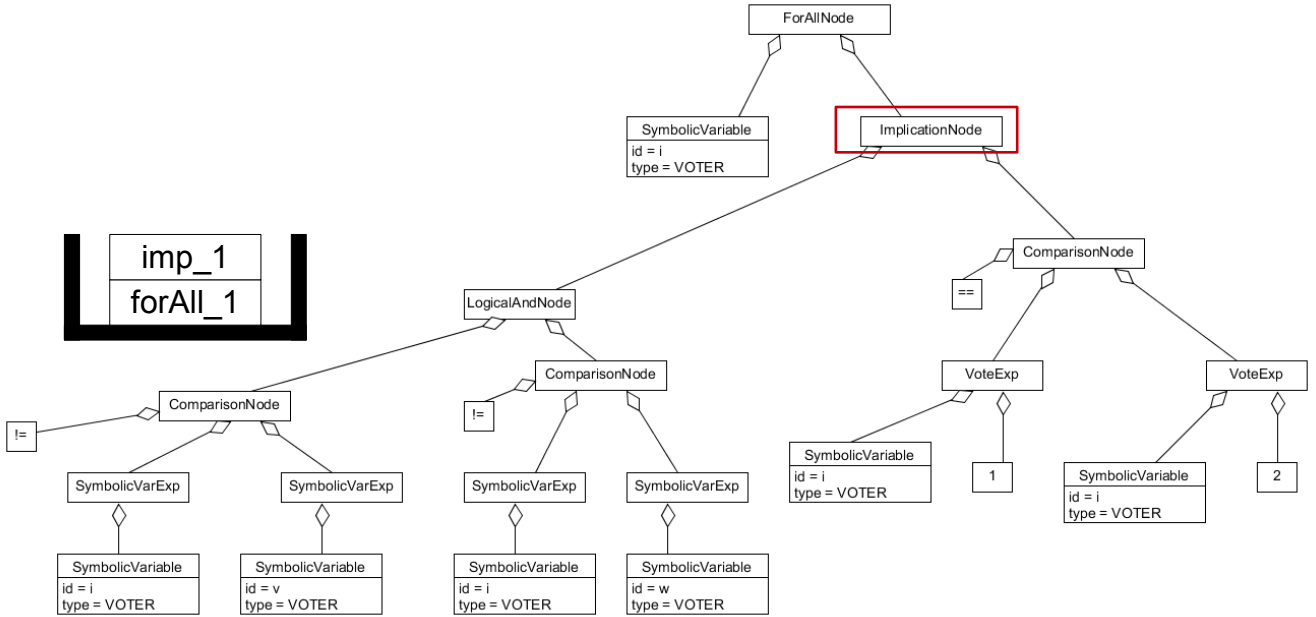
Generierung des Codes für CBMC

```
unsigned int i;
unsigned int forAll_i = 1;

for(i = 0; i < V && forAll_i; i++) {
```



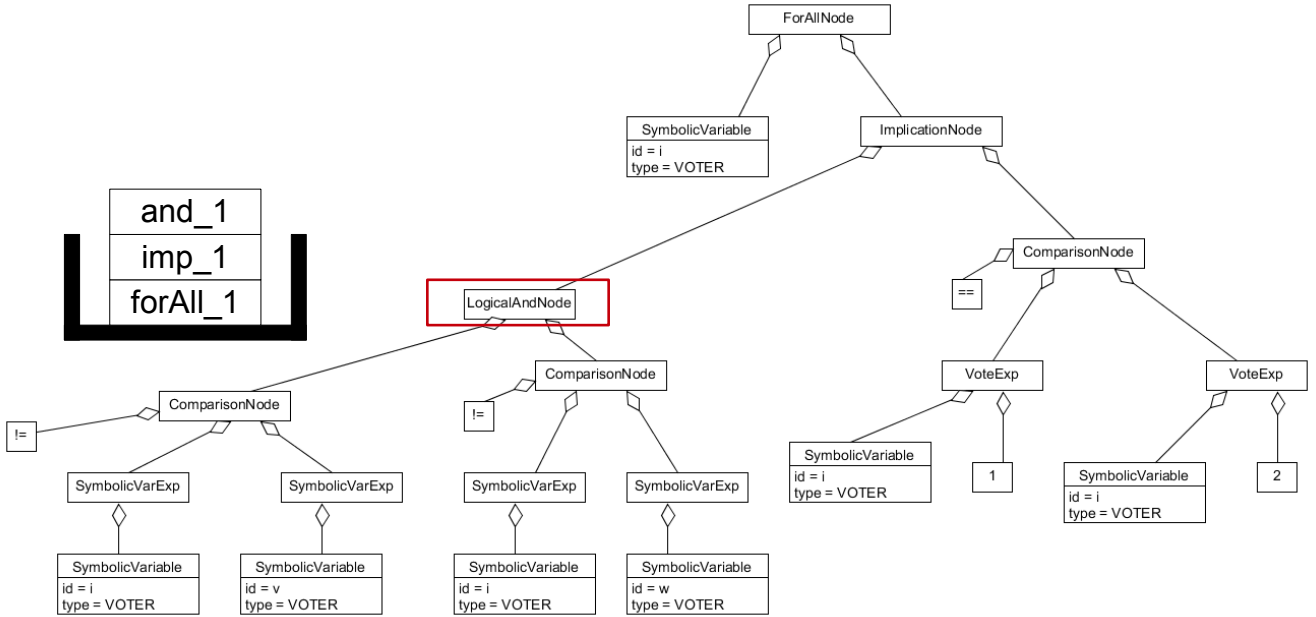
Generierung des Codes für CBMC

```
unsigned int i;
unsigned int forAll_i = 1;

for(i = 0; i < V && forAll_i; i++) {
    unsigned int impl_1;
```



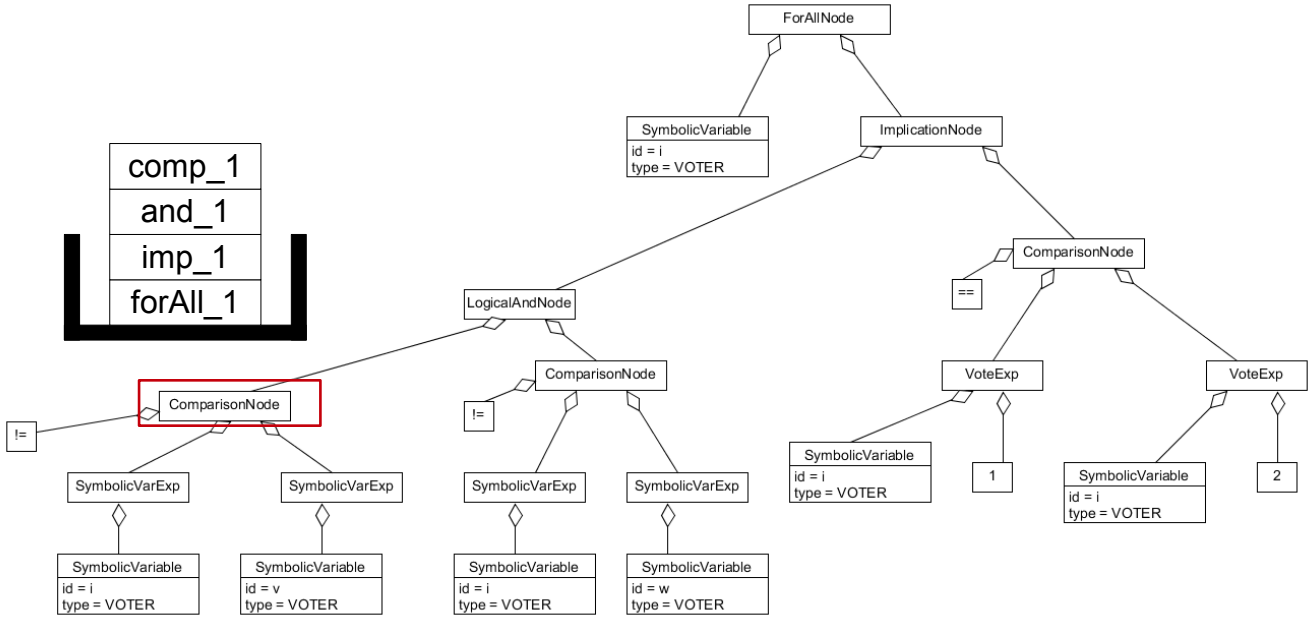
Generierung des Codes für CBMC

```
unsigned int i;
unsigned int forAll_i = 1;

for(i = 0; i < V && forAll_i; i++) {
    unsigned int impl_1;

    unsigned int and_1;
```



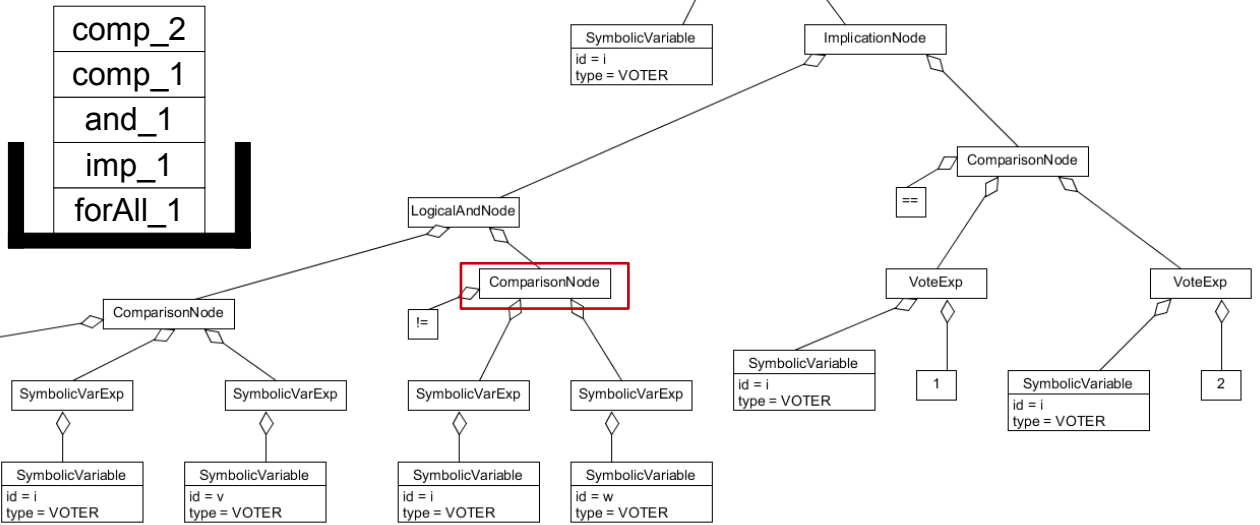
Generierung des Codes für CBMC

```
unsigned int i;
unsigned int forAll_i = 1;

for(i = 0; i < V && forAll_i; i++) {
    unsigned int impl_1;

    unsigned int and_1;

    unsigned int comp_1 = i != v;
```



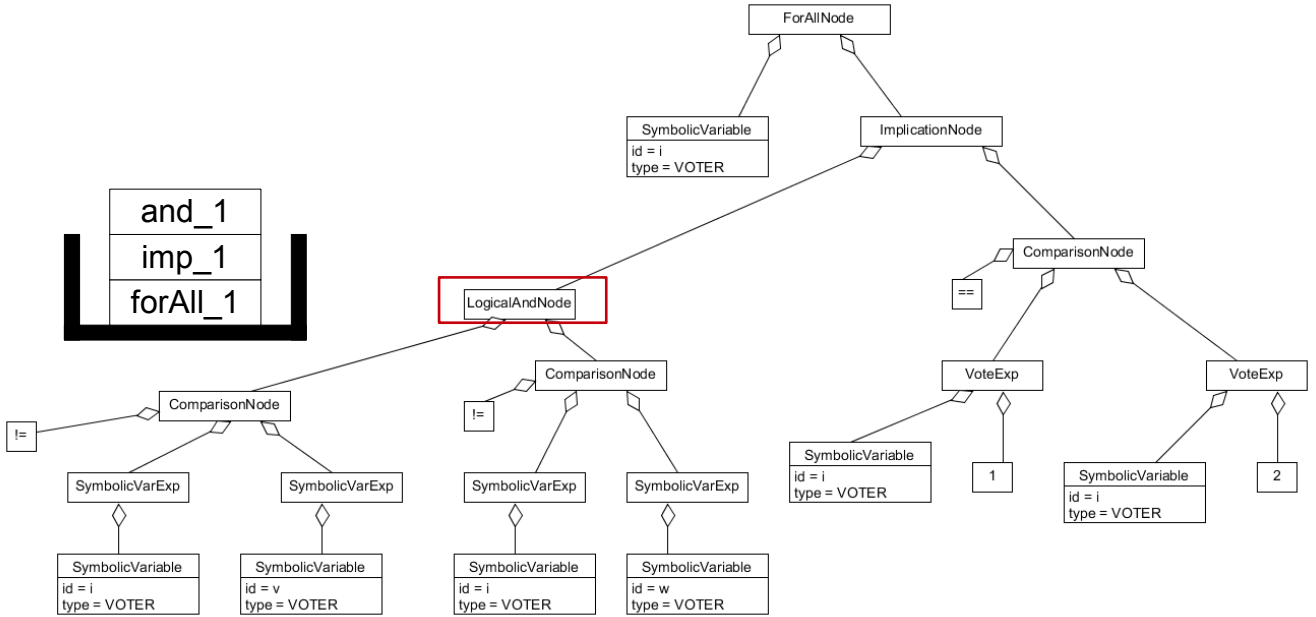
Generierung des Codes für CBMC

```
unsigned int i;
unsigned int forAll_i = 1;

for(i = 0; i < V && forAll_i; i++) {
    unsigned int impl_1;

    unsigned int and_1;

    unsigned int comp_1 = i != v;

    unsigned int comp_2 = i != w;
```



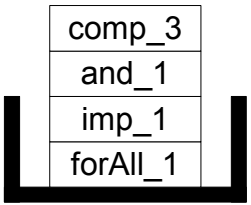
Generierung des Codes für CBMC

```
unsigned int i;
unsigned int forAll_i = 1;

for(i = 0; i < V && forAll_i; i++) {
    unsigned int impl_1;

    unsigned int and_1;

    unsigned int comp_1 = i != v;

    unsigned int comp_2 = i != w;

    and_1 = comp_1 && comp_2;
```



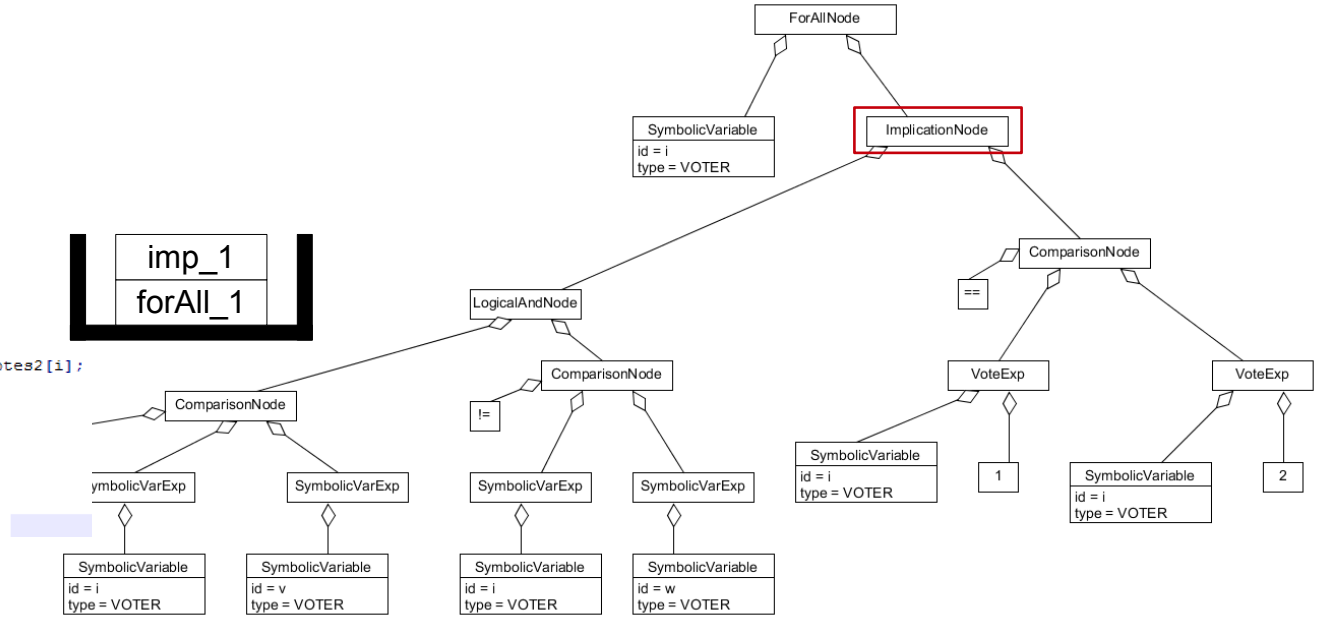
Generierung des Codes für CBMC

```
unsigned int i;
unsigned int forAll_i = 1;

for(i = 0; i < V && forAll_i; i++) {
    unsigned int impl_1;

    unsigned int and_1;

    unsigned int comp_1 = i != v;

    unsigned int comp_2 = i != w;

    and_1 = comp_1 && comp_2;

    unsigned int comp_3 = votes1[i] == votes2[i];
```



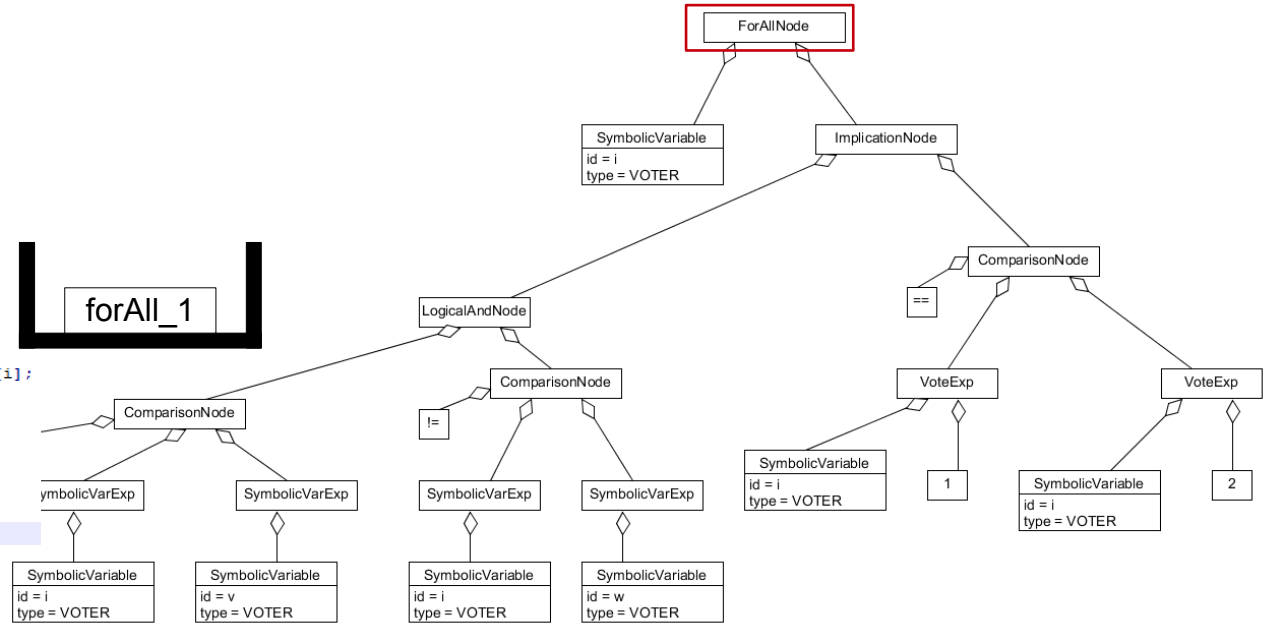
Generierung des Codes für CBMC

```
unsigned int i;
unsigned int forAll_i = 1;

for(i = 0; i < V && forAll_i; i++) {
    unsigned int impl_1;

    unsigned int and_1;

    unsigned int comp_1 = i != v;

    unsigned int comp_2 = i != w;

    and_1 = comp_1 && comp_2;

    unsigned int comp_3 = votes1[i] == votes2[i];

    impl_1 = !and_1 || comp_3;
```



Generierung des Codes für CBMC

```
unsigned int i;
unsigned int forAll_i = 1;

for(i = 0; i < V && forAll_i; i++) {
    unsigned int impl_1;

    unsigned int and_1;

    unsigned int comp_1 = i != v;

    unsigned int comp_2 = i != w;

    and_1 = comp_1 && comp_2;

    unsigned int comp_3 = votes1[i] == votes2[i];

    impl_1 = !and_1 || comp_3;

    forAll_i = impl_1;
}

assume(forAll_i);
```



Generierung des Codes für CBMC

- Problem: Unteres nicht allgemein
- Was, falls Quantor && anderer Ausdruck?
- Für unser Schema kein Problem
- Demo!

```
unsigned int i;
unsigned int forAll_i = 1;

for(i = 0; i < V && forAll_i; i++) {
    unsigned int impl_1;

    unsigned int and_1;

    unsigned int comp_1 = i != v;

    unsigned int comp_2 = i != w;

    and_1 = comp_1 && comp_2;

    unsigned int comp_3 = votes1[i] == votes2[i];

    impl_1 = !and_1 || comp_3;

    forAll_i = impl_1;
}

assume(forAll_i);

unsigned int i = 0;

for (i = 0; i < V; i++) {
    if (i != v && i != w) {
        assume (votes1[i] == votes2[i]);
    }
}
```

Abweichung vom Pflichtenheft

- So gut wie keine
- + Option für Checker Auswahl
- + Option für maximal nebenläufige Checker
- + Option für Verneinung boolscher Ausdruck

Implementierung