



Verziókezelők és folytonos integráció

Rendszertervezés laboratórium 1

Jegyzőkönyv

2021. tavasz

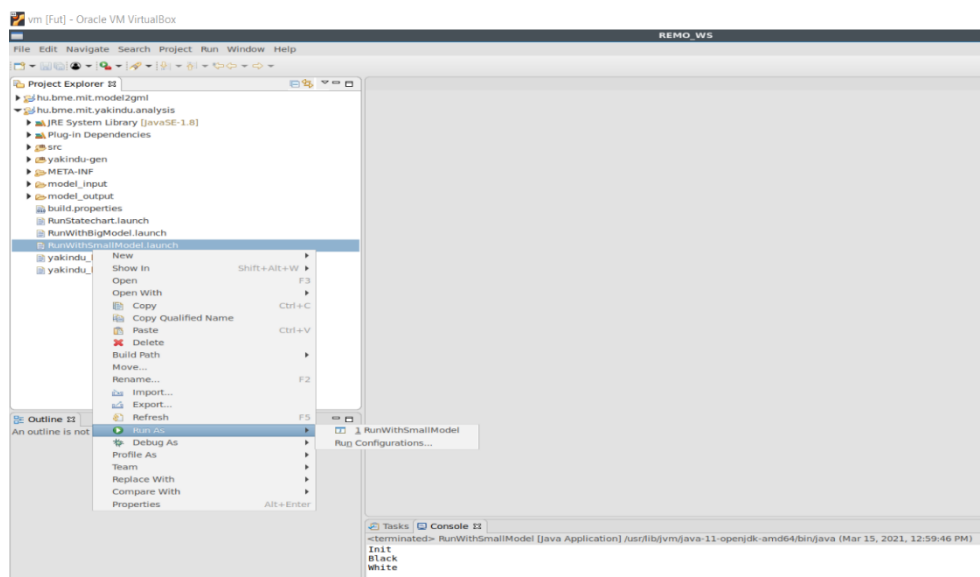
Készítette	Pap Mátyás (A8KADM)
Dátum	2021. március 15.
GitHub hivatkozás	https://github.com/Matyi98/ReLab2
iMSC feladat	elkészült / nem készült el

1 feladatcsoport

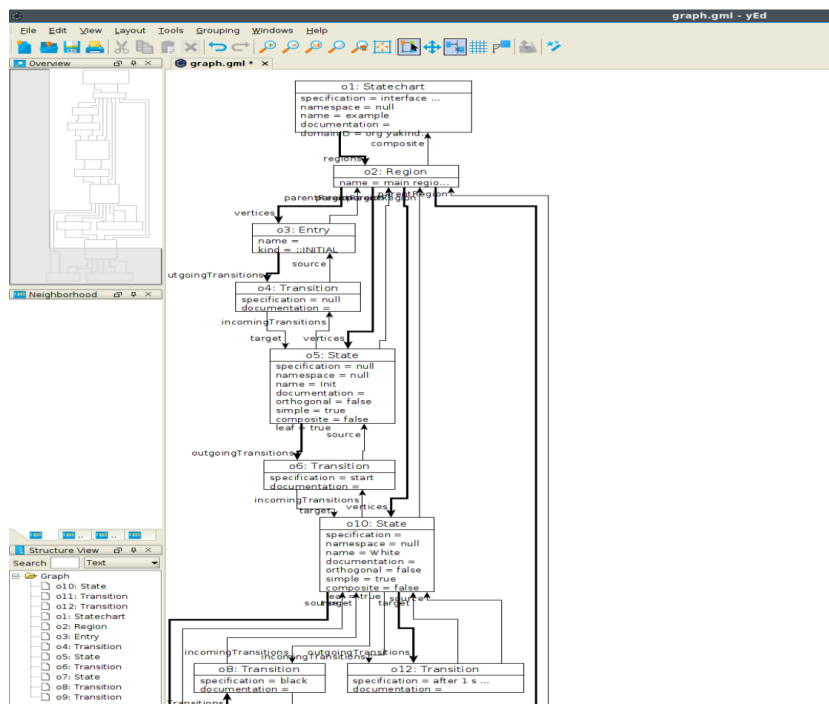
Kezdeti lépésként átolvastam a *Modellek programozott feldolgozása* Laboratórium ismeretőt. Ezután inicializáltam a környezetet. VirtualBoxot használlok. Illetve elvégeztem a leírásban lévő előkészületi lépéseket.

2 feladatcsoport

2.1 *Indítsa el a hu.bme.mit.yakindu.analysis/RunWithSmallModel.launch konfigurációval az alkalmazást (jobb klikk > Run as... > Run), tekintse meg a konzol kimenetet!*



2.2 *Nyissa meg a generált gráfvizualizálást a model_output mappában! A gráf a yEd nevű szerkesztőprogramban nyílik meg, amelyben Layout > Hierarchical > OK paranccsal elrendezheti a gráfot.*



2.3 Egészítse ki az alkalmazást, hogy az állapotok mellett a tranzíciókat is kiírja

```
example.sgen example.sct Main.java
package hu.bme.mit.yakindu.analysis.workhere;

import org.eclipse.emf.common.util.TreeIterator;

public class Main {
    @Test
    public void test() {
        main(new String[0]);
    }

    public static void main(String[] args) {
        ModelManager manager = new ModelManager();
        Model2GML model2gml = new Model2GML();

        // Loading model
       EObject root = manager.loadModel("model_input/example.sct");

        // Reading model
        Statechart s = (Statechart) root;
        TreeIterator<EObject> iterator = s.eAllContents();
        while (iterator.hasNext()) {
            EObject content = iterator.next();
            if (content instanceof State) {
                State state = (State) content;
                System.out.println(state.getName());
            }
            else if (content instanceof org.yakindu.sct.model.sgraph.Transition) {
                Transition trans = (Transition) content;
                System.out.println(trans.getSource().getName() + " -> " + trans.getTarget().getName());
            }
        }

        // Transforming the model into a graph representation
        String content = model2gml.transform(root);
        // and saving it
        manager.saveFile("model_output/graph.gml", content);
    }
}
```

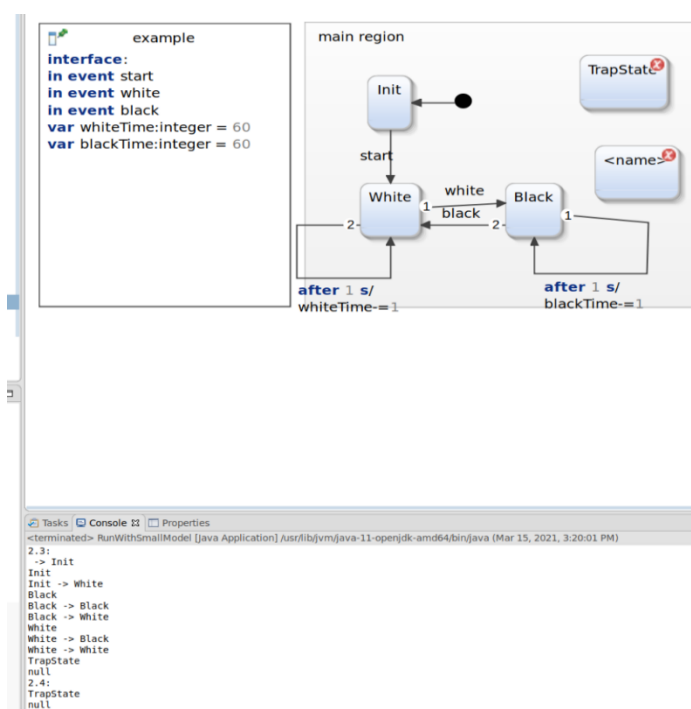
```
Tasks Console Properties
<terminated> RunWithSmallModel [Java Application] /usr/lib/jvm/java-11-openjdk-amd64/bin/java (Mar 15, 2021, 2:55:44 PM)
-> Init
Init
Init -> White
Black
Black -> Black
Black -> White
White
White -> Black
White -> White
|
```

2.4 Egészítse ki az alkalmazást, hogy az keresse meg azokat a csapda állapotokat, amiből nem vezet ki él, és írassa ki a nevüket! A program működését demonstrálja egy példával (rajzolja át a modellt).

```
//Find states with no target state
System.out.println("2.4: ");
TreeIterator<EObject> iterator1 = s.eAllContents();
while (iterator1.hasNext()) {
    EObject content = iterator1.next();
    if (content instanceof State) {
        boolean isTrapState = true;
        TreeIterator<EObject> iterator2 = s.eAllContents();
        while (iterator2.hasNext()) {
            EObject content1 = iterator2.next();
            if (content1 instanceof Transition) {
                Transition trans = (Transition) content1;
                if (trans.getTarget() == content)
                    isTrapState = false;
            }
        }
        if (isTrapState)
            System.out.println(((State) content).getName());
    }
}

// Transforming the model into a graph representation
String content = model2gml.transform(root);
// and saving it
manager.saveFile("model_output/graph.gml", content);
}
```

```
Tasks Console Properties
<terminated> RunWithSmallModel [Java Application] /usr/lib/jvm/java-11-openjdk-amd64
2.3:
-> Init
Init
Init -> White
Black
Black -> Black
Black -> White
White
White -> Black
White -> White
TrapState
null
2.4:
TrapState
null
```



2.5 Egészítse ki az alkalmazást, hogy keresse meg azokat az állapotokat, amelyeknek nincs neve, és javasoljon nekik egy alkalmas nevet. A javasolt nevet írjaki a konzolra.

```
//Find noName states
TreeIterator<EObject> iterator3 = s.eAllContents();
System.out.println("2.5: ");
int i = 0;
int j = 0;
while (iterator3.hasNext()) {
    EObject content = iterator3.next();
    if (content instanceof State) {
        i++;
        State state = (State) content;

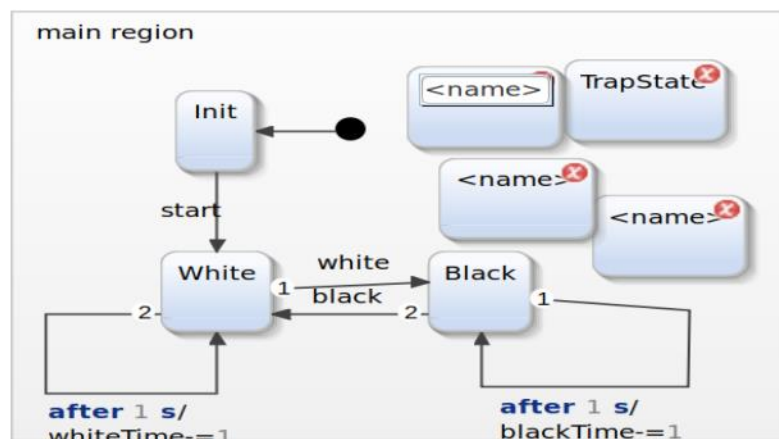
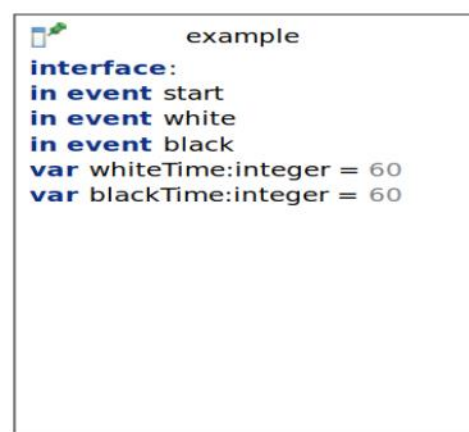
        if (state.getName() == null)
        {
            j++;
            System.out.println(i+". state doesn't have a name, a recomandation: state"+j);
        }
    }
}

// Transforming the model into a graph representation
String content = model2gml.transform(root);
// and saving it
manager.saveFile("model_output/graph.gml", content);
}
```

Tasks Console Properties

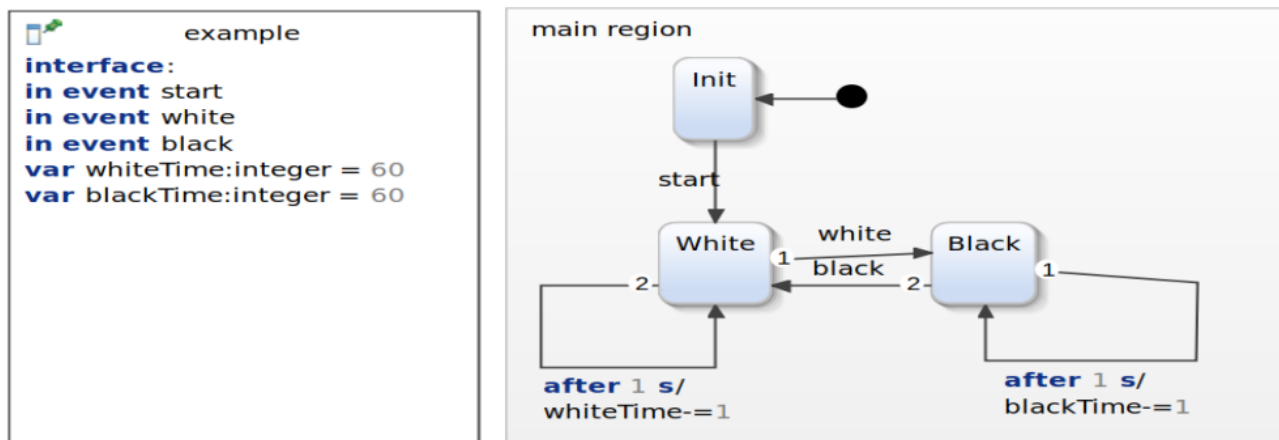
<terminated> RunWithSmallModel [Java Application] /usr/lib/jvm/java-11-openjdk-amd64/bin/java (Mar 15, 2021, 3:37:28 PM)

Init -> White
 Black
 Black -> Black
 Black -> White
 White
 White -> Black
 White -> White
 TrapState
 null
 null
 null
 2.4:
 TrapState
 null
 null
 null
 2.5:
 5. state doesn't have a name, a recomandation: state1
 6. state doesn't have a name, a recomandation: state2
 7. state doesn't have a name, a recomandation: state3

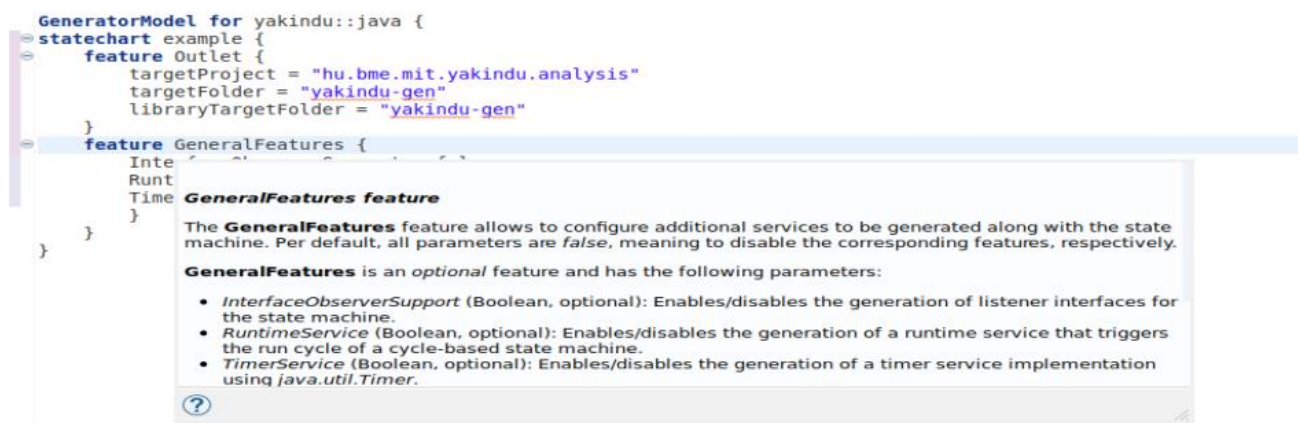


3 feladatcsoport

3.1 Amennyiben módosította az example.stc állományt (például hibákat injekt, állítsa vissza



3.2 ...Írja le hogy ezeknek a paramétereknek a bevezetésével milyen változást idézett elő!



A GeneralFeature lehetővé teszi további szolgáltatások generálását az állapotgépek mellett. Három paramétert említ a dokumentáció:

- **InterfaceObserverSupport**: listener interfészek generálását engedélyezi.
- **RuntimeServices**: futásidejű szolgáltatások generálását engedélyezi, amelyek az állapotgép triggerként szolgálhatnak.
- Engedélyezi időzítók generálását.

3.3 *hu.bme.mit.yakindu.analysis/src/hu/bme/mit/yakindu/analysis/workhere/RunStatechart.java* áttekintése

```
package hu.bme.mit.yakindu.analysis.workhere;

import java.io.IOException;

import hu.bme.mit.yakindu.analysis.RuntimeService;
import hu.bme.mit.yakindu.analysis.TimerService;
import hu.bme.mit.yakindu.analysis.example.ExampleStatemachine;
import hu.bme.mit.yakindu.analysis.example.IExampleStatemachine;

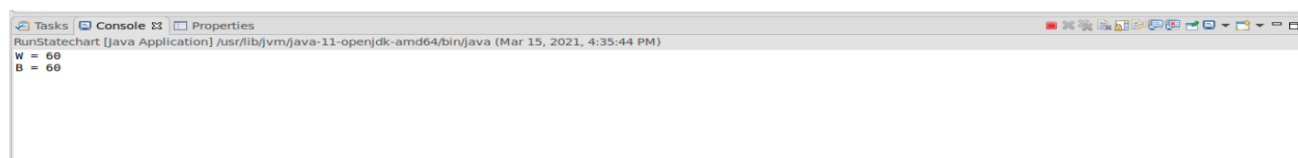
public class RunStatechart {

    public static void main(String[] args) throws IOException {
        ExampleStatemachine s = new ExampleStatemachine();
        s.setTimer(new TimerService());
        RuntimeService.getInstance().registerStatemachine(s, 200);
        s.init();
        s.enter();
        s.runCycle();
        print(s);
        s.raiseStart();
        s.runCycle();
        System.in.read();
        s.raiseWhite();
        s.runCycle();
        print(s);
        System.exit(0);
    }

    public static void print(IExampleStatemachine s) {
        System.out.println("W = " + s.getSCInterface().getWhiteTime());
        System.out.println("B = " + s.getSCInterface().getBlackTime());
    }
}
```

(kivettem a kommenteket és átnéztem a kódot és a leírást ebben a feladatban)

3.4 *Futtassa az alkalmazást az alábbi konfigurációval: hu.bme.mit.yakindu.analysis/RunStatechart.Launch*



3.5 *A kódrészlet alapján készítsen olyan alkalmazást, amely:*

- A konzolról beolvas sorokat
 - Amennyiben a beolvasott szöveg megegyezik egy esemény nevével (start, white vagy black), meghívja az adott eseményt az állapotgépen,
 - Amennyiben a beolvasott szöveg az „exit”, az alkalmazás leáll.
- Minden beolvasott sor után írjuk ki az összes változó (WhiteTime és BlackTime) értékét!

```

public class RunStatechart {

    public static void main(String[] args) throws IOException {
        Scanner scanner = new Scanner(System.in);
        ExampleStatemachine s = new ExampleStatemachine();
        s.setTimer(new TimerService());
        RuntimeService.getInstance().registerStatemachine(s, 200);
        s.init();
        s.enter();
        s.runCycle();
        while (scanner.hasNextLine()) {
            String cmd;
            Scanner lineScanner = new Scanner(scanner.nextLine());
            if (lineScanner.hasNext())
            {
                cmd = lineScanner.next();
                lineScanner.close();
                System.out.println(cmd);
                switch (cmd)
                {
                    case "Start":
                        s.raiseStart();
                        break;
                    case "White":
                        s.raiseWhite();
                        break;
                    case "Black":
                        s.raiseBlack();
                        break;
                    case "Exit":
                        s.exit();
                        System.exit(0);
                    default:
                        System.out.println("Wrong command!: "+cmd);
                }
                s.runCycle();
                print(s);
                if (s.getBlackTime() < 0 || s.getWhiteTime() < 0)
                    System.exit(0);
            }
            else
            {
                lineScanner.close();
                break;
            }
        }
    }
}

```

Tasks Console Properties

<terminated> RunStatechart [Java Application] /usr/lib/jvm/java-11-openjdk-amd64/bin/java (Mar 15, 2021, 5:28:32 PM)

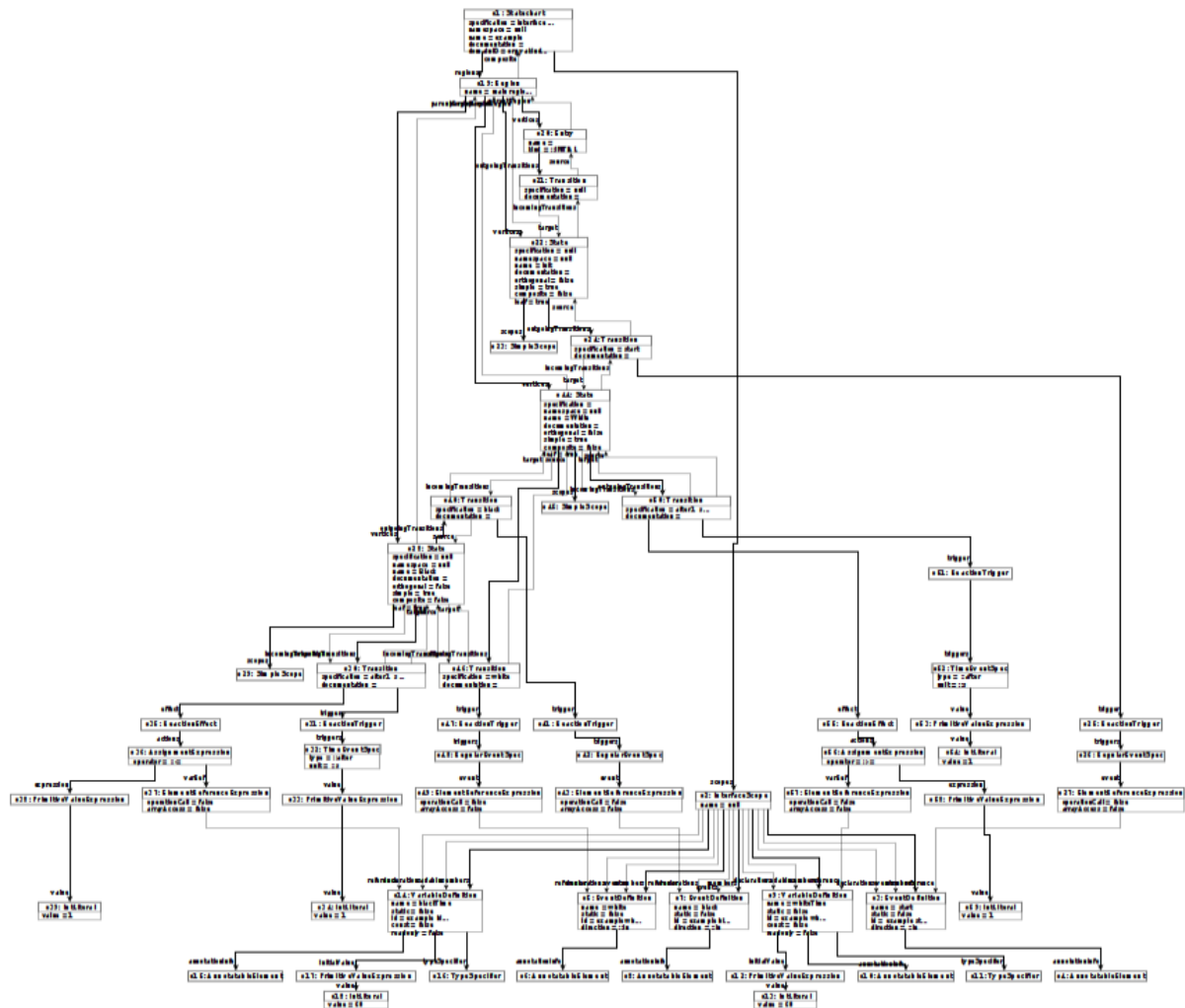
```

Start
Start
W = 60
B = 60
White
White
W = 58
B = 60
Black
Black
W = 58
B = 55
White
White
W = 53
B = 55
Exit
Exit

```

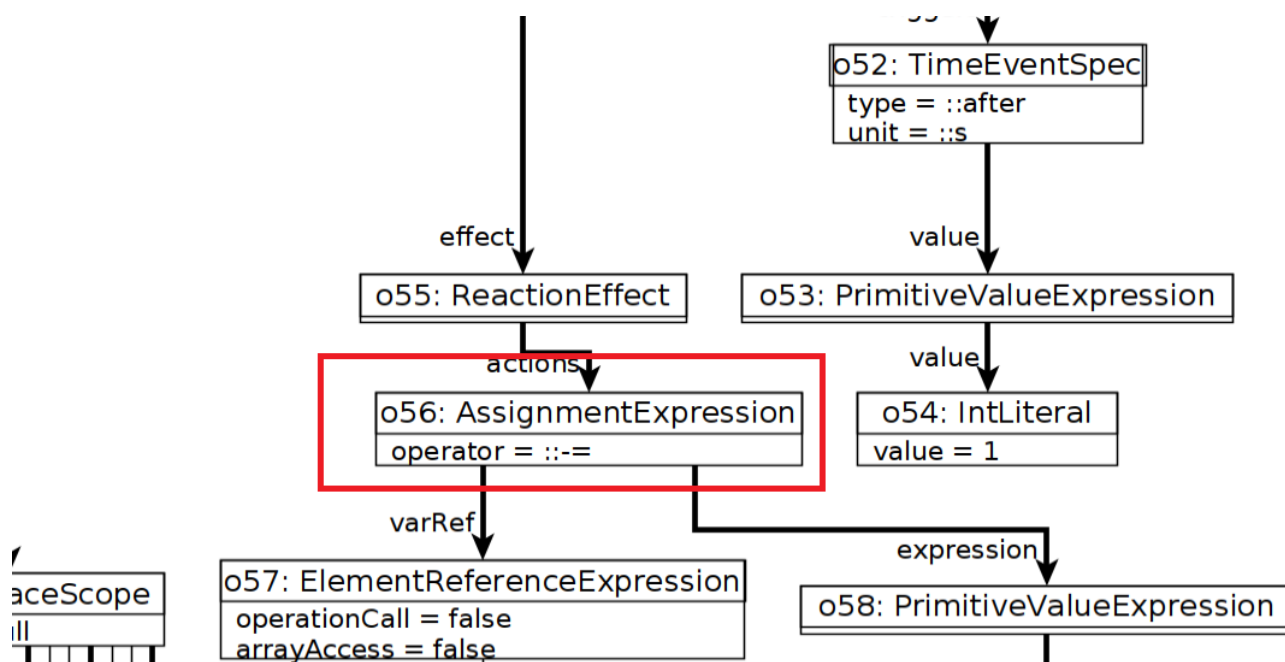

4 Feladatcsoport

4.1 Futtassa a `hu.bme.mit.yakindu.analysis.RunWithBigModel.Launch` konfigurációval az alkalmazást, és tekintse át a kirajzolt gráfot! Milyen változást tapasztal?



Belekerült a sakk időszámláló modellje is (hatalmas lett a gráf).

4.2 Mutassa meg a kirajzolt gráfban, hogy hogy néz ki a `whiteTime -= 1` kifejezés absztrakt szintaxis fája!



4.3 A 2. ponthoz hasonlóan járja be a modellt, és írja ki az összes belső változó és bemenő esemény nevét! Az eseményekhez definíciójának megtalálásához sokat segíthet a generált gráfnézetet!

Változók:

- o14: blackTime
- o9: whiteTime

Események/kifejezések:

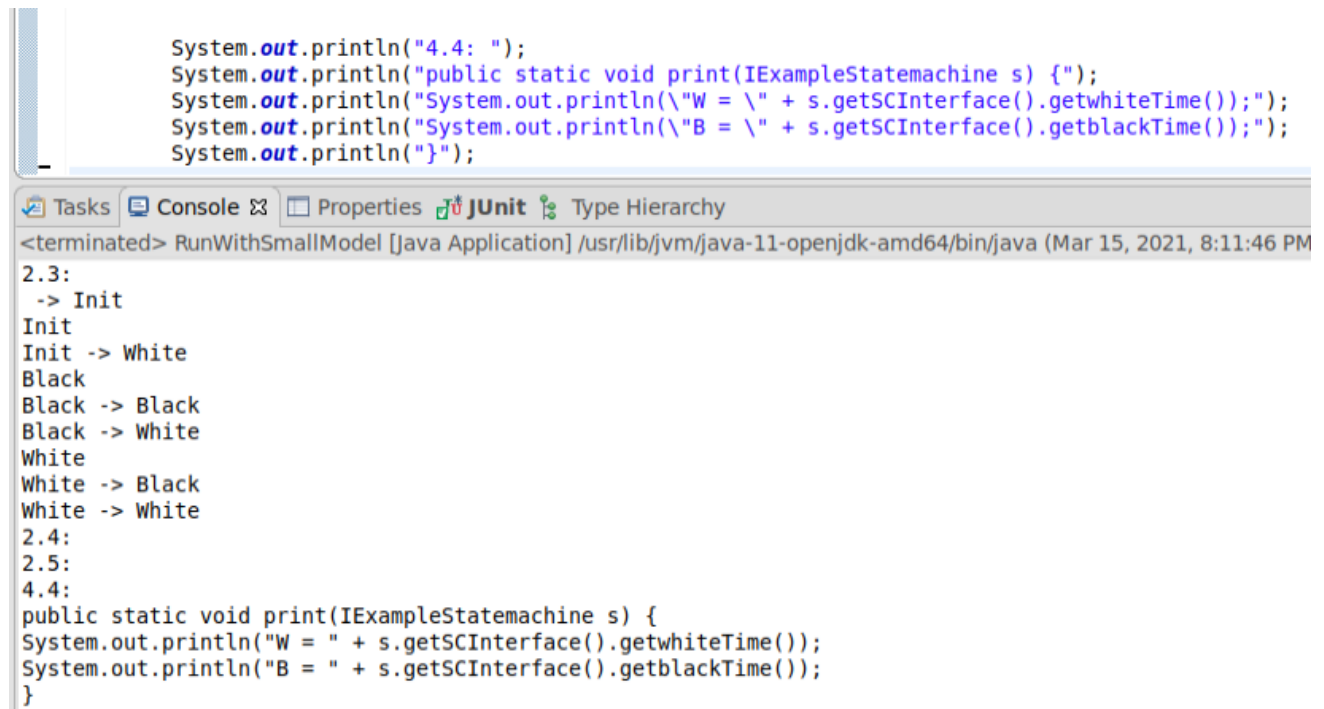
- o3: start
- o5: white
- o7: black

Graph

- o10: AnnotableElement
- o11: TypeSpecifier
- o12: PrimitiveValueExpression
- o13: IntLiteral
- o14: VariableDefinition
- o15: AnnotableElement
- o16: TypeSpecifier
- o17: PrimitiveValueExpression
- o18: IntLiteral
- o19: Region
- o1: Statechart
- o20: Entry
- o21: Transition
- o22: State
- o23: SimpleScope
- o24: Transition
- o25: ReactionTrigger
- o26: RegularEventSpec
- o27: ElementReferenceExpression
- o28: State
- o29: SimpleScope
- o2: InterfaceScope
- o30: Transition
- o31: ReactionTrigger
- o32: TimeEventSpec
- o33: PrimitiveValueExpression
- o34: IntLiteral
- o35: ReactionEffect
- o36: AssignmentExpression
- o37: ElementReferenceExpression
- o38: PrimitiveValueExpression
- o39: IntLiteral
- o3: EventDefinition
- o40: Transition
- o41: ReactionTrigger
- o42: RegularEventSpec
- o43: ElementReferenceExpression
- o44: State
- o45: SimpleScope
- o46: Transition
- o47: ReactionTrigger
- o48: RegularEventSpec
- o49: ElementReferenceExpression
- o4: AnnotableElement
- o50: Transition
- o51: ReactionTrigger
- o52: TimeEventSpec
- o53: PrimitiveValueExpression
- o54: IntLiteral
- o55: ReactionEffect
- o56: AssignmentExpression
- o57: ElementReferenceExpression
- o58: PrimitiveValueExpression
- o59: IntLiteral
- o5: EventDefinition
- o6: AnnotableElement
- o7: EventDefinition
- o8: AnnotableElement
- o9: VariableDefinition

4.4 Írjon egy olyan alkalmazást, ami az események és a változók nevét az alábbi formában írja ki:

```
public static void print(IExampleStatemachine s) {
    System.out.println("W = " + s.getSCInterface().get<Első változó neve>());
    ...
    System.out.println("B = " + s.getSCInterface().get<Utolsó változó neve>());
}
```



The screenshot shows an IDE with a code editor and a console window. The code editor contains the following Java code:

```
System.out.println("4.4: ");
System.out.println("public static void print(IExampleStatemachine s) {");
System.out.println("System.out.println(\"W = \" + s.getSCInterface().getwhiteTime());");
System.out.println("System.out.println(\"B = \" + s.getSCInterface().getblackTime());");
System.out.println("}");
```

The console window shows the output of the program:

```
<terminated> RunWithSmallModel [Java Application] /usr/lib/jvm/java-11-openjdk-amd64/bin/java (Mar 15, 2021, 8:11:46 PM)
2.3:
-> Init
Init
Init -> White
Black
Black -> Black
Black -> White
White
White -> Black
White -> White
2.4:
2.5:
4.4:
public static void print(IExampleStatemachine s) {
System.out.println("W = " + s.getSCInterface().getwhiteTime());
System.out.println("B = " + s.getSCInterface().getblackTime());
}
```

4.5. ...

2 óra próbálkozás után feladtam az utolsó 3 feladat megoldását...