Detalhamento do Compilador LingPon para Framework Java

Linguagens e Compiladores

Leonardo Trevisan Silio

Alterações iniciais

```
target                          : FRAMEWORK_CPP_2_0 {
                                    $$ = graph->createTarget(Target::FRAMEWORK_CPP_2_0_TARGET);
                                  }
                                  | NAMESPACES_CPP_2_0 {
                                    $$ = graph->createTarget(Target::NAMESPACES_CPP_2_0_TARGET);
                                  }
                                  | FRAMEWORK_JAVA_1_0 {
                                    $$ = graph->createTarget(Target::FRAMEWORK_JAVA_1_0_TARGET);
                                  }
                                  ;
```

```
static const int FRAMEWORK_CPP_2_0_TARGET = 1;
static const int NAMESPACES_CPP_2_0_TARGET = 2;
static const int FRAMEWORK_JAVA_1_0_TARGET = 4;
```

Processo em macroanálise

```
void FrameworkJAVA10Compiler::generateCode()
{
    std::cout << "\nGenerating code..." << std::endl;

    Instance* tempinst;
    for (std::map<std::string, Fbe*>::iterator it = graph->getFbes()->begin(); it != graph->getFbes()->end(); it++)
    {
        generateCodeFbe(it->second);
    }
}
```

Arquivo Objetivo

```
public class SimpleApplication extends NOPApplication{

    public Gate gate;
    public RemoteControl remoteControl;

    public SimpleApplication(int schedulerStrategy) {
        super(schedulerStrategy);

        startApplication();
    }

    @Override
    public void initRules() {

        Condition cond1 = new Condition(Condition.CONJUNCTION);
        cond1.addPremise(new Premise(remoteControl.bIsPressed,NBoolean.TRUE_NOP,Premise.EQUAL,false));
        cond1.addPremise(new Premise(gate.GateStatus,true,Premise.EQUAL,false));
        Action action1 = new Action();

        Rule rule1 = new Rule("Open gate", scheduler, cond1, action1, false);
        action1.addInstigation(new Instigation(remoteControl.bIsPressed,false));
        action1.addInstigation(new Instigation(gate.mtOpenGate));

        Condition cond2 = new Condition(Condition.CONJUNCTION);
        cond2.addPremise(new Premise(remoteControl.bIsPressed,NBoolean.TRUE_NOP,Premise.EQUAL,false));
        cond2.addPremise(new Premise(gate.GateStatus,false,Premise.EQUAL,false));
        Action action2 = new Action();

        Rule rule2 = new Rule("Close gate", scheduler, cond2, action2, false);
        action2.addInstigation(new Instigation(remoteControl.bIsPressed,false));
        action2.addInstigation(new Instigation(gate.mtCloseGate));
    }

    @Override
    public void initFactBase() {
        gate = new Gate();
        remoteControl = new RemoteControl();
    }
```

} Propriedades retiradas da Fbe

} Inicializando estratégia

} Inicialização de uma condição

} Inicialização da Regra e das Instigações

} Inicialização das propriedades

Exemplo de geração

```
for (std::map<std::string, Rule*>::iterator it = rules->begin(); it != rules->end(); ++it, count++)
{
    rule = it->second;
    c = std::to_string(count);
    newline(2);
    int conj;
    if (rule->getCondition()->getConjunction() == 0)
        conj = 1;
    else
        conj = rule->getCondition()->getConjunction()->getConjunctionId();
    //Escreve condição
    filestream << "Condition cond" << c << " = new Condition(Condition." << (conj == 1 ? "CONJUNCTION" : "DISJUNCTION") << ");";
```

```
//Itera todas as premissas
for (std::map<std::string, Premise*>::iterator itpremise = rule->getCondition()->getPremises()->begin();
    itpremise != rule->getCondition()->getPremises()->end(); itpremise++)
    {
        Premise* premise = itpremise->second;
        exp = premise->getExpression();
        newline(2);
        filestream << "cond" << c << ".addPremise(new Premise(" << exp->getLeftFactor()->getStringValue() << ", ";
        writefactorvalue(exp->getRightFactor()->getFactorId(), exp->getRightFactor()->getStringValue());
        write(", premise.");
```

Conclusões

```
void FrameworkJAVA10Compiler::writefactorvalue(int attribute_id, std::string factor_string)
{
    if (attribute_id != 2)
    {
        write("new ");
        writefactor(attribute_id);
        filestream << "(" << factor_string << ")";
    }
    else
    {
        if (factor_string == "1")
            write("new NBoolean(true)");
        else write("new NBoolean(false)");
    }
}
```

```
switch (exp->getSymbol()->getSymbolId())
{
    case 1:
        filestream << "EQUAL";
        break;
    case 2:
        filestream << "DIFFERENT";
        break;
    case 3:
        filestream << "SMALLERTHAN";
        break;
    case 4:
            filestream << "GREATERTHAN";
        break;
    case 5:
        filestream << "SMALLEROREQUAL";
        break;
    case 6:
        filestream << "GREATEROREQUAL";
        break;
}
```