EXAMPLE: IWIR v1.1

An additional compound task is introduced with IWIR v1.1: `blockScope`. This allows grouping of tasks in one scope.

Figure 1 illustrates a `blockScope` workflow:

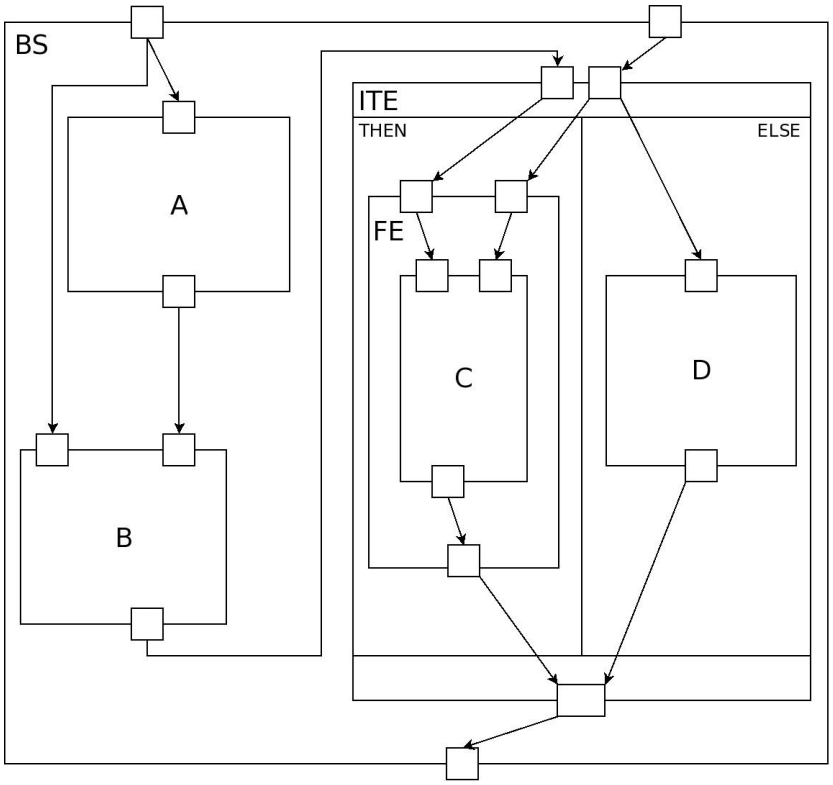

Figure 1: A dummy block scope example.

`BS` defines the block scope, `A, B, C` and `D` are atomic tasks, `ITE` is a if task, and `FE` is a parrallel foreach task.

Among with IWIR v1.1 comes iwirTool-1.1.jar. Beside others, the tool introduces `ConditionExpression`, and checking for cyclic dependencies in the workflow and cross-references in an `ifTask`. The `ConditionExpression` is used to define a condition for the `ifTask` and `whileTask`. For example, `new ConditionExpression("in = 1")`.

Figure 2 depicts a cyclic-dependency of tasks `A` and `B` and a cross-reference in the `ifTask ITE`:

All validation methods are invoked implicitly when constructing the workflow by hand or reading the workflow from a `xml` file.

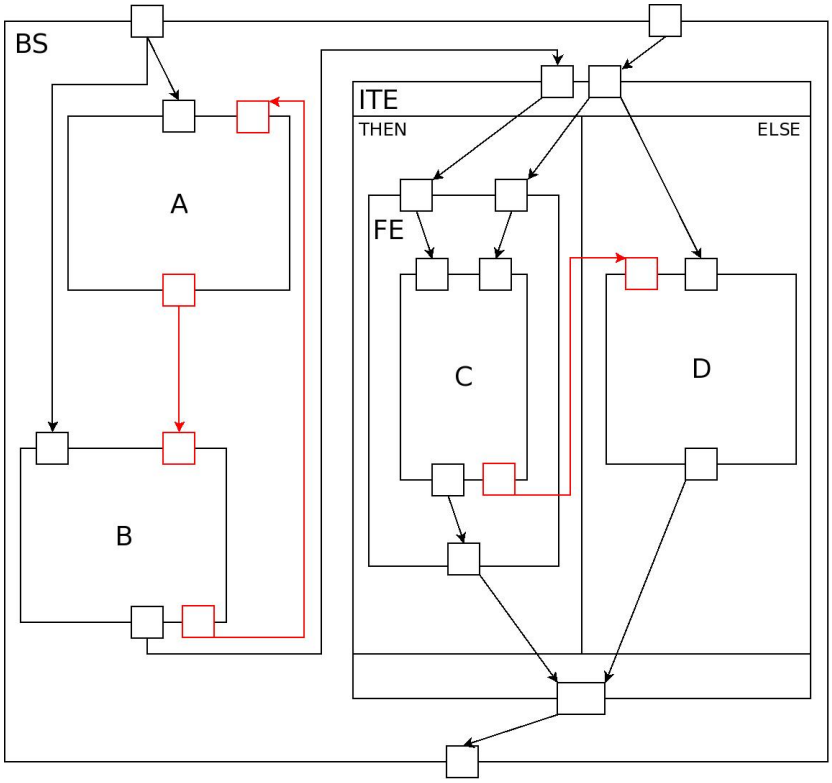

Figure 2: An invalid Workflow.

The corresponding source for the example illustrated in Figure 1 is given in Listing 1:

```
package org.shiwa.fgi.iwir.examples;

import java.io.File;

import org.shiwa.fgi.iwir.BlockScope;
import org.shiwa.fgi.iwir.CollectionType;
import org.shiwa.fgi.iwir.ConditionExpression;
import org.shiwa.fgi.iwir.IWIR;
import org.shiwa.fgi.iwir.IfTask;
import org.shiwa.fgi.iwir.InputPort;
import org.shiwa.fgi.iwir.LoopElement;
import org.shiwa.fgi.iwir.OutputPort;
import org.shiwa.fgi.iwir.ParallelForEachTask;
import org.shiwa.fgi.iwir.SimpleType;
import org.shiwa.fgi.iwir.Task;

public class BlockScopeMock {

  public static void main(String[] args) {
    IWIR i = new BlockScopeMock().build();

    System.out.println(i.asXMLString());
  }

  public IWIR build() {

    BlockScope bs = new BlockScope("topLevel");
    bs.addInputPort(new InputPort("in1", SimpleType.INTEGER));
    bs.addInputPort(new InputPort("in2", new CollectionType(
        SimpleType.INTEGER)));
    // ip s

    Task a = new Task("A", "calc1");
    a.addInputPort(new InputPort("in1", SimpleType.INTEGER));
    a.addOutputPort(new OutputPort("out1", SimpleType.INTEGER));

    Task b = new Task("B", "calc2");
    b.addInputPort(new InputPort("in1", SimpleType.INTEGER));
    b.addInputPort(new InputPort("in2", SimpleType.INTEGER));
    b.addOutputPort(new OutputPort("out1", SimpleType.INTEGER));

    IfTask ite = new IfTask("ITE");
    ite.addInputPort(new InputPort("in1", SimpleType.INTEGER));
    ite.addInputPort(new InputPort("in2", new CollectionType(
        SimpleType.INTEGER)));
    ite.setCondition(new ConditionExpression("in1 = 1"));

    // then
    ParallelForEachTask foreach1 = new ParallelForEachTask("
        foreach1");
    foreach1.addInputPort(new InputPort("in1", SimpleType.INTEGER))
        ;
    foreach1.addLoopElement(new LoopElement("lp1", new
        CollectionType(
        SimpleType.INTEGER)));

    Task c = new Task("C", "consumer");
    c.addInputPort(new InputPort("in1", SimpleType.INTEGER));
    c.addInputPort(new InputPort("in2", SimpleType.INTEGER));
    c.addOutputPort(new OutputPort("out1", SimpleType.INTEGER));
```

```
    foreach1.addTask(c);
    foreach1.addOutputPort(new OutputPort("out1", new
        CollectionType(
        SimpleType.INTEGER)));
    foreach1.addLink(foreach1.getPort("in1"), c.getPort("in1"));
    foreach1.addLink(foreach1.getPort("lp1"), c.getPort("in2"));
    foreach1.addLink(c.getPort("out1"), foreach1.getPort("out1"));

    ite.addTaskToThenBlock(foreach1);

    // else
    Task d = new Task("D", "consumer");
    d.addInputPort(new InputPort("in1", SimpleType.INTEGER));
    d.addOutputPort(new OutputPort("out1", SimpleType.INTEGER));

    ite.addTaskToElseBlock(d);
    ite.addOutputPort(new OutputPort("out1", new CollectionType(
        SimpleType.INTEGER)));
    ite.addLink(ite.getPort("in1"), foreach1.getPort("in1"));
    ite.addLink(ite.getPort("in2"), foreach1.getPort("lp1"));
    ite.addLink(foreach1.getPort("out1"), ite.getPort("out1"));
    ite.addLink(ite.getPort("in1"), d.getPort("in1"));
    ite.addLink(d.getPort("out1"), ite.getPort("out1"));

    bs.addTask(a);
    bs.addTask(b);
    bs.addLink(bs.getPort("in1"), a.getPort("in1"));
    bs.addLink(bs.getPort("in1"), b.getPort("in1"));
    bs.addLink(a.getPort("out1"), b.getPort("in2"));

    bs.addTask(ite);
    bs.addLink(b.getPort("out1"), ite.getPort("in1"));
    bs.addLink(bs.getPort("in2"), ite.getPort("in2"));

    bs.addOutputPort(new OutputPort("out1", new CollectionType(
        SimpleType.INTEGER)));
    bs.addLink(ite.getPort("out1"), bs.getPort("out1"));

    IWIR dummy = new IWIR("blockScope");

    dummy.setTask(bs);
    return dummy;

  }

}
```

Listing 1: BlockScopeMock.java

The workflow in XML representation is given in Listing 2:

```
<IWIR version="1.1" wfname="blockScope" xmlns="http://shiwa-
    workflow.eu/IWIR">
  <blockScope name="topLevel">
    <inputPorts>
      <inputPort name="in1" type="integer"/>
      <inputPort name="in2" type="collection/integer"/>
    </inputPorts>
    <body>
      <task name="A" tasktype="calc1">
        <inputPorts>
          <inputPort name="in1" type="integer"/>
```

```
    </inputPorts>
    <outputPorts>
      <outputPort name="out1" type="integer"/>
    </outputPorts>
  </task>
  <task name="B" tasktype="calc2">
    <inputPorts>
      <inputPort name="in1" type="integer"/>
      <inputPort name="in2" type="integer"/>
    </inputPorts>
    <outputPorts>
      <outputPort name="out1" type="integer"/>
    </outputPorts>
  </task>
  <if name="ITE">
    <inputPorts>
      <inputPort name="in1" type="integer"/>
      <inputPort name="in2" type="collection/integer"/>
    </inputPorts>
    <condition>in1 = 1</condition>
    <then>
      <parallelForEach name="foreach1">
        <inputPorts>
          <inputPort name="in1" type="integer"/>
          <loopElements>
            <loopElement name="lp1" type="collection/integer"/>
          </loopElements>
        </inputPorts>
        <body>
          <task name="C" tasktype="consumer">
            <inputPorts>
              <inputPort name="in1" type="integer"/>
              <inputPort name="in2" type="integer"/>
            </inputPorts>
            <outputPorts>
              <outputPort name="out1" type="integer"/>
            </outputPorts>
          </task>
        </body>
        <outputPorts>
          <outputPort name="out1" type="collection/integer"/>
        </outputPorts>
        <links>
          <link from="foreach1/in1" to="C/in1"/>
          <link from="foreach1/lp1" to="C/in2"/>
          <link from="C/out1" to="foreach1/out1"/>
        </links>
      </parallelForEach>
    </then>
    <else>
      <task name="D" tasktype="consumer">
        <inputPorts>
          <inputPort name="in1" type="integer"/>
        </inputPorts>
        <outputPorts>
          <outputPort name="out1" type="integer"/>
        </outputPorts>
      </task>
    </else>
    <outputPorts>
      <outputPort name="out1" type="collection/integer"/>
    </outputPorts>
```

```
        <links>
          <link from="ITE/in1" to="foreach1/in1"/>
          <link from="ITE/in2" to="foreach1/lp1"/>
          <link from="foreach1/out1" to="ITE/out1"/>
          <link from="ITE/in1" to="D/in1"/>
          <link from="D/out1" to="ITE/out1"/>
        </links>
      </if>
    </body>
    <outputPorts>
      <outputPort name="out1" type="collection/integer"/>
    </outputPorts>
    <links>
      <link from="topLevel/in1" to="A/in1"/>
      <link from="topLevel/in1" to="B/in1"/>
      <link from="A/out1" to="B/in2"/>
      <link from="B/out1" to="ITE/in1"/>
      <link from="topLevel/in2" to="ITE/in2"/>
      <link from="ITE/out1" to="topLevel/out1"/>
    </links>
  </blockScope>
</IWIR>
```

Listing 2: BlockScopeMock.xml