Implementierung von Operatoren der MovingObjectsLibrary (stlib) in der MMDB

In dieser Anleitung wird demonstriert, wie Operatoren der MovingObjectsLibrary für die Implementierung neuer Operatoren in der MMDB verwendet werden können. Die Anleitung stützt sich dabei im Wesentlichen auf den „Programmer's Guide for New MMDB Operators“ der MMDB und führt die zusätzlich durchzuführenden Schritte auf, die für die Implementierung der Operatoren der Bibliothek erforderlich sind.

Beispielhaft wird die Implementierung anhand des Operators *trajectory* veranschaulicht.

Voraussetzungen

Grundlegende Voraussetzung für die Nutzung eines Operators der MovingObjectsLibrary ist, dass die Objekte, auf die der jeweilige Operator angewendet werden soll, das jeweils erforderliche Interface implementieren. Dies stellt sicher, dass dem jeweiligen Operator alle erforderlichen Methoden zur Realisierung der Operation zur Verfügung stehen. Die Bibliothek stellt für die einfachere Implementierung eines Interfaces konkrete Klassen bereit, die das jeweilige Interface und die erforderlichen Methoden bereits implementieren.

Zur Implementierung des Operators *trajectory*, welcher auf MovingPoint-Objekte angewendet wird, muss das Interface *MovingPointIF* durch die MMDB-Klasse *AttributMpoint* implementiert werden. Um nicht alle Methoden des Interfaces implementieren zu müssen, wird ein neues Attribut vom Typ *MovingPointIF* der Bibliothek aufgenommen, welches Objekte der Bibliotheks-Klasse *MovingPoint* aufnehmen kann. Zur Implementierung der Interface-Methoden werden die Aufrufe dann an die entsprechenden Methoden des neuen Attributs weitergeleitet. Nachfolgend wird ein Ausschnitt der Anpassung der Klasse *AttributMpoint* dargestellt:

```
public class AttributeMpoint extends MovingAttribute<AttributeUpoint> implements
MovingPointIF {

   private MovingPointIF mpoint;

   public AttributeMpoint() {
      mpoint = new MovingPoint(0);
   }
...
   @Override
   public PointIF getValue(TimeInstantIF instant) {
      return mpoint.getValue(instant);
   }
...
```

Implementierung des Operators in der MMDB

Zur Implementierung des Operators in der MMDB wird nun gemäß des „Programmer‘s Guide for New MMDB Operators“ der MMDB verfahren. Als erstes muss entschieden werden, ob der Operator einen Ergebnis-Strom oder ein einzelnes Ergebnis liefern soll. Dementsprechend muss der neue Operator entweder das Interface *StreamOperator*, für einen Ergebnis-Strom, oder das Interface *ObjectNode*, für ein einzelnes Ergebnis, implementieren.

Der Operator *trajectory* liefert ein *Line*-Objekt als einzelnes Ergebnis zurück. Entsprechend implementiert der Operator das Interface *ObjectNode*.

```
public class Trajectory implements ObjectNode {
```

Als nächstes sind der Konstruktor, die *typeCheck()* und die *getOutputType()* Methode zu implementieren. Hier sind keine speziellen Implementierungen vorzunehmen und es kann daher gemäß dem Programmer‘s Guide verfahren werden.

Beim Operator *trajectory* sind das folgende Implementierungen:

```
public Trajectory(final Node input) {
   this.input = input;

}

public void typeCheck() throws TypeException {
   this.input.typeCheck();

   // Is input1 an ObjectNode?
   TypecheckTools.checkNodeType(this.input, ObjectNode.class,
                                this.getClass(), 1);
   this.objectInput = (ObjectNode) this.input;

   if (this.objectInput.getOutputType().getClass() == AttributeMpoint.class){
      this.outputType = new AttributeLine();

   } else {
      throw new TypeException(
            "%s's %ss input needs to provide %s, but provides %s.",
            this.getClass().getSimpleName(),
            Nodes.NodeType.ObjectNode,
            AttributeMpoint.class.getSimpleName(),
            objectInput.getClass().getSimpleName());
   }

}

public MemoryObject getOutputType() {
   return this.outputType;
}
```

Als nächstes ist die jeweilige Auswertungs-Methode, *getNext()* bei Strom-Ergebnissen bzw. *getResult()* bei Einzelergebnissen, zu implementieren. Hier erfolgt die eigentliche Ausführung des Operators bzw. der Aufruf des entsprechenden Operators der Bibliothek. Hier muss nun auch dafür gesorgt werden, dass das Ergebnis-Objekt der Bibliotheks-Operation in ein entsprechendes Objekt einer MMDB-Klasse umgewandelt wird, bevor dieses als Endergebnis zurückgegeben wird.

Beim Operator *trajectory* wird die Methode *getResult()* implementiert. Das Ergebnis des Operators der Bibliothek muss in einen entsprechenden Typ der MMDB umgewandelt werden. Dazu wird ein neues Objekt der MMDB-Klasse *AttributeLine* erzeugt. Nachfolgend ist die Implementierung der *getResult()* Methode dargestellt, wobei der Aufruf des Operators der Bibliothek gesondert hervorgehoben ist:

```
public MemoryObject getResult() throws MemoryException {

   AttributeMpoint mpoint = (AttributeMpoint) this.objectInput.getResult();

   LineIF line = stlib.operations.projection.Trajectory.execute(mpoint);

   AttributeLine lineMMDB = new AttributeLine();

   for (HalfsegmentIF hs : line.getHalfsegments()) {
```

```
      if (hs.isLeftDominating()) {
         Segment seg = new Segment();

         seg.setxValue1((float) hs.getLeftPoint().getXValue());
         seg.setyValue1((float) hs.getLeftPoint().getYValue());
         seg.setxValue2((float) hs.getRightPoint().getXValue());
         seg.setyValue2((float) hs.getRightPoint().getYValue());

         lineMMDB.addSegment(seg);

      }
   }

   return lineMMDB;
}
```

Weitere Schritte

Weitere Schritte sind gemäß dem Programmer‘s Guide durchzuführen. Dazu zählen die erforderlichen Implementierungen, durch die das Parsen des neuen Operators durch den Query-Prozessor der MMDB ermöglicht wird und durch die der neue Operator in der GUI zur Verfügung gestellt wird. Ggf. sind zusätzlich Implementierungen zur Speicherüberwachung vorzunehmen, wenn der zu implementierende Operator große Objekte bzw. eine große Menge an Objekten erzeugt.