

Projektabschluss

Depiktionsverarbeitung für den Geometrischen Agenten

Nils Andresen

NilsAndresen@web.de

25. Januar 2004

1 Problemstellung

Das Generelle Problem liegt in der „Bildverarbeitung“ für den Geometrischen Agenten. Da dies aber ein breit gefächertes Feld darstellt, werde ich die Bildverarbeitung kurz unterteilen.

- Eingabe von Bilddaten :
 - Aus einem Toolkit,
 - aus einer gescannten Skizze,
 - oder auch aus einer Textuellen Wegbeschreibung.
- Beschreibungssprache der Bilddaten
(DML - Depiction Modelling Language)
 - Formelle Definition (Syntax, Semantik, evtl Timing)
 - Ausarbeitung eines “Checkers”...
- Verarbeitung der Bilddaten
 - Ausgabe als Grafik (jpg, png, svg...),
 - als CRIL-Netz, zur verarbeitung im GA,
 - oder als Textuelle Wegbeschreibung...

Wir werden unser Augenmerk für die Momentane Bearbeitung auf die DML (die ja auf jeden Fall benötigt wird) legen und ausserdem das Toolkit als Eingabe und eine Ausgabe in CRIL-Netzen realisieren.

1.1 Aufgaben

1. Toolkit
 - (a) Oberfläche (gerne in Java...)

(b) mindestens folgende Elemente müssen Repräsentiert werden :

- Wege
- Polygone als Gebäude oder Landmarken
- Pfeile
- Startsymbol
- Zielsymbol
- Beschriftungen (evtl. in einem “Eigenschaften”-Dialog)

(c) Speicherformat innerhalb des Toolkits egal

(d) Kommunikation mit RRS/RRL (Routen Repräsentations Schicht)

2. RRS & DML

(a) RRS : Kommunikation mit anderen Elementen (Server funktionalität)

(b) RRS : Basiselemente Abbilden (siehe oben..)

(c) RRS : checker :

- wohlgeformtheit
- lokalität
- Routenexistenz

(d) RRS : Speichern in DML

(e) Formale Definition der DML

3. Ausgabe in CRIL

(a) Direkt vom RRS-Server

(b) aus DML-Datei

(c) CRIL-Implementiern...

2 Beispiel

2.1 UI (Toolkit)

Der Benutzer wählt aus einem Inventar des Toolkits Objekte aus und ordnet sie auf einer Ebene (Bildschirm, PDA, etc. an). Die Elemente des Toolkits sind nicht nur graphischer Natur, sondern jedem Element ist auch eine festgelegte Interpretation als Routenelement zugeordnet.. Im Beispiel (Carolas Zeichnung) wurde folgendes Inventar an Routenelementen festgelegt:

- Wege (repräsentiert durch Linien)
- Landmarken (Polygone)
- Richtungen (Pfeile)

- Labels (Beschriftung, die das Toolkit einem Routenelement eindeutig zuordnen muss)
- Startmarkierung
- Zielmarkierung
- Ein- und Ausgänge (Doppelstriche, die die Begrenzungen der Landmarken (Polygone) schneiden)

Diese Objekte werden als Liste an die Routenrepräsentationsschicht (RRL) weitergereicht. Dabei enthalten die Objekte Angaben über ihren Typ, ihre Position und räumliche Ausdehnung sowie über die zeitliche Abfolge des Entstehens der Objekte. Letztere Angabe könnte entweder implizit durch eine geordnete Liste erfolgen (das erste Objekt wurde zuerst angelegt, etc.) oder durch explizite Indizierung der Objekte (Objekt mit ID 1 wurde als erstes angelegt, etc.).

2.2 Routenrepräsentationsschicht (RRS/RRL) und DML

Die RRL hat zwei Aufgaben. Erstens muss sie die Eingabe aus dem Toolkit überprüfen und eventuelle Fehler oder Ambiguitäten an das Toolkit zurückmelden, damit der Nutzer seine Skizze anpassen kann. Zweitens wird in der RRL eine Repräsentation der Skizze aufgebaut, die von der Ausgabeschicht genutzt werden kann, um daraus Output irgendeiner Form (Text, Grafik, CRIL) zu generieren. Dies muss nicht unbedingt sequentiell erfolgen. Die Repräsentation erfolgt in einer noch zu definierenden Depiction Modelling Language (DML).

Als erstes wird die Liste des Toolkits vom Lokalitätschecker analysiert. Dieser überprüft, welche Objekte nahe aneinander liegen. Darüberhinaus kann er bei diesen Objekten auch überprüfen, ob das eine das andere umschließt, ob sich die Objekte überlappen, oder wie sie zueinander angeordnet sind (parallel, aufeinander zulaufend, etc.). Sollten z.B. zwei Weg-Elemente nahe beieinander enden, könnte der Lokalitätschecker eine Anfrage an das Toolkit initiieren, ob die beiden Weg-Elemente miteinander verknüpft werden sollen. Überschneiden sich zwei Wegelemente, so kann eine Anfrage an das Toolkit gerichtet werden, ob diese Weg-Elemente eine Kreuzung bilden sollen. Der Lokalitätschecker sollte natürlich effizient genug sein, dass er nicht jedes Element mit jedem vergleicht, sondern sich zu jedem Element nur die heraussucht, die sich in unmittelbarer Nähe (was auch immer das heißen mag) befinden, da ansonsten der Aufwand bei n Elementen mit $(n-1)!$ -Vergleichen recht hoch wäre.

Beispiel für die Analyse des Lokalitätscheckers: die von Carola erstellte Skizze (ich habe das Wegelement in 6 Strecken 6.1 bis 6.6 beginnend bei der Mensa und endend beim Abaton zerlegt)

- : nähe(Mensa, Startpunkt)
- : nähe(Mensa, Pfeil 1)
- : nähe(Mensa, Weg 6.1)

- : $\text{n\ddot{a}he}(\text{Mensa}, \text{Weg } 6.2)$
- : $\text{n\ddot{a}he}(\text{Mensa}, \text{Audimax})$
- : $\text{n\ddot{a}he}(\text{Mensa}, \text{Ein-/Ausgang})$
- : [weitere n\ddot{a}he-Relationen]

Dadurch, dass die Mensa in n\ddot{a}he-Relation zu den o.g. Objekten steht, wird \u00fcberpr\u00fcft, welche r\u00e4umlichen Relationen diese Objekte noch zueinander haben:

- $\text{umschliesst}(\text{Mensa}, \text{Startpunkt}) \longrightarrow \text{istStartpunkt}(\text{Mensa})$
- $\text{aufRand}(\text{Mensa}, \text{Ein-/Ausgang})$
- $\text{aufRand}(\text{Mensa}, \text{Pfeil } 1)$
- $\text{abgewandt}(\text{Pfeil } 1, \text{Mensa}) \longrightarrow \text{Info, Landmarke Mensa zu verlassen}$
- $\text{durchquert}(\text{Pfeil } 1, \text{Ein-/Ausgang})$
- $\text{!parallel}(\text{Pfeil } 1, \text{Weg } 6.1) \longrightarrow \text{verbunden}(\text{Pfeil } 1, \text{Weg } 6.1) \longrightarrow \text{aus diesen beiden Relationen ergibt sich eine Richtungs\u00e4nderung}$
- $\text{parallel}(\text{Mensa}, \text{Weg } 6.1)$
- $\text{parallel}(\text{Mensa}, \text{Weg } 6.2) \longrightarrow \text{!parallel}(\text{Weg } 6.1, \text{Weg } 6.2) \longrightarrow \text{aus diesen beiden Relationen ergibt sich eine Richtungs\u00e4nderung}$
- $\text{parallel}(\text{Mensa}, \text{Audimax})$
- Weitere Relationen sind denkbar.

Der Wohlgeformtheitschecker \u00fcberpr\u00fcft, ob die Anordnung der Objekte akzeptiert werden kann. Er \u00fcberpr\u00fcft, ob folgende Constraints eingehalten werden (z.B. dass es einen Start- und einen Zielmarker geben muss, dass Landmarken sich nicht \u00fcberschneiden (aber eventuell umschlie\u00dfen?) d\u00fcrfen, etc.). Desweiteren k\u00f6nnen manche Konstellationen R\u00fcckfragen an das Toolkit erfordern (wenn z.B. Landmarken sich auf Wegen befinden).

Auf Grund dieser Daten kann der Routenchecker \u00fcberpr\u00fcfen, ob es in dieser Skizze \u00fcberhaupt eine beschriebene Route gibt. Als erstes sucht er sich den Startpunkt (Mensa). Von dieser Landmarke f\u00fchrt eine Route weg (Pfeil 1). Diese ist mit einem Weg (Weg 6) verbunden, dessen Ende mit einer Richtung (Pfeil 2) verkn\u00fcft ist, die die Umgrenzung der Landmarke Abaton schneidet. Diese Landmarke ist mit einer Zielmarkierung ausgezeichnet. Also gibt es einen Weg vom Start zum Ziel.

2.3 Die Ausgabeschicht

In der Ausgabeschicht, wird die in der RRL erzeugte Repräsentation in die gewünschte Ausgabe transformiert (Text, Grafik, CRIL, etc.). Es wird mit dem als Startpunkt ausgezeichneten Objekt (eventuell ist das nur ein Punkt auf einer Fläche, die keine Landmarke darstellt) begonnen, aus der Repräsentation eine räumliche Beschreibung und ggfs. einen Aktionsplan zu erzeugen.

3 Stand des Projektabschnittes

DML Spezifikation am Beispiel (siehe Beispiel.swi)

4 Literatur

?

5 Arbeitsplan und Teilaufgaben

Erstes Ziel sollte die Definition und Implementation der RRS und der DML und ausserdem die Implementation des Toolkits sein. Nach dessen Fertigstellung kann man sich der Ausgabe in CRIL und anderen dingen widmen.

Daraus ergibt sich: Eine Gruppe die das Toolkit erstellt und eine Gruppe die die DML und RRS erstellt.

Bis die "Toolkit-Gruppe" mit den Implementations-Aspekten und den "Speicherungsfragen" fertig ist, sollte auch die "RRS-DML"-Gruppe einen Interface zur Ansteuerung der RRS bereitgestellt haben, auf dem dann die weiteren Überlegungen der "Toolkit-Gruppe" aufbauen.