Extensible Software Simulation System for Imprecise Geospatial Process

Đorđe Obradović, Zora Konjović, Milan Segedinac

{obrad, ftn_zora, milansegedinac}@uns.ac.rs

Fakultet tehničkih nauka, Novi Sad

Abstract-in this paper we propose software architecture of the imprecise geospatial process management system. Position imprecision is modeled as fuzzy point in $\mathbf{R^2}$ *linear fuzzy space. Imprecise geospatial process is modeled by using object oriented paradigm. Proposed architecture is extensible framework for imprecise geospatial process simulation and management with support to imprecise data.*

1.Introduction

As the Internet and network infrastructure grown Spatial data information has become more accessible to a wider audience. The diversity of user needs has led to various applications and systems for analysis and use of such data. *Spatial Data Infrastructure* (SDI) [1] is a framework of spatial data, metadata, users and tools that are interactively connected in order to use spatial data in efficient way. Basic assumption underlying spatial data infrastructure is that the spatial data are completely accurate. When modelling spatial objects, these real objects are represented by approximate models which, in addition, include imprecise data. Neglecting this imprecision sometimes leads to incorrect estimations of spatial relations. For example, because of imprecise data, three collinear points can be mistakenly considered to be non-collinear, or two points that coincide in reality are modelled to be non-coincident, *etc*. Models of the geospatial processes which deal with such inadequate imprecise data represent the real process approximately. Because of these differences, the simulation process leads to accumulation of errors and the problem becomes more noticeable.

In general, three basic approaches to spatial data uncertainty/imprecision are recognized : exact models [2-4], probabilistic models [5],[6] and fuzzy models [7-11].

The paper [7] deals with the problem of spatial objects which do not have homogeneous interiors and sharply defined boundaries. For a spatial vagueness called fuzziness this paper provides a conceptual model of fuzzy spatial objects that also incorporates fuzzy geometric union, intersection, and difference operations as well as fuzzy topological predicates. In particular, this model is not based on Euclidean space and on an infinite-precision arithmetic but it rests on a finite, discrete geometric domain called grid partition which takes into account finite-precision number systems available in computers. In the paper [7] the abstract conceptual model of fuzzy spatial data types, such as fuzzy points, fuzzy lines and fuzzy regions, is proposed. The paper focuses on defining their structure and semantics. Theory of fuzzy sets and fuzzy topologies is a formal framework for the conceptual model. Two definitions of fuzzy geometrical point (FGT) were presented. The paper [8] deals with fuzzy plane geometry. The fuzzy point is defined as fuzzy set with membership function that is convex, upper semi-continuous, and with single point core.

PostGIS [12] ads support for geographic objects to the PostgreSQL object-rational database. It is released under the GNU General Public License. PostGIS covers a lot of spatial functions, such as basic topology support, data validation, coordinate transformation, programming APIs and user interface tools.

The paper consists of five sections. Following this introductory section the model of the imprecise point and definition of $\mathbf{R^2}$ linear fuzzy space are set out in Section 2. In the Section 3. model of geospatial process is defined as object model. Section 4. contains software architecture of the imprecise geospatial process management. The final section contains concluding remarks and future research directions.

2. $\mathbf{R^2}$ Linear Fuzzy Space

This paper relies on the concepts of imprecise point and linear fuzzy space defined in [12]. Hereafter, the definitions of the imprecise point and R^2 linear fuzzy space are given.

Definition 1. Fuzzy point at $\mathbf{P \in R^2}$, denoted by $\tilde{\mathbf{P}}$ is defined by its membership function $\mu_{\tilde{P}} \in \mathcal{F}^2$. Set $\mathcal{F}^2$ contains all membership functions $\mathbf{u{:}R^2 \to [0,1]}$ satisfying following conditions:

(i) $(\forall u \in \mathcal{F}^2)(\exists_1 P \in R^2)\ u(P) = 1$

(ii) $(\forall X_1, X_2 \in R^2)(\lambda \in [0,1])\ u(\lambda X_1 + (1-\lambda)X_2) \geq \min(\ u(X$

function $\mathbf{u}$ is upper semi continuous

$[[u]]^{\alpha} = \{X | X \in R^2, u(X) \geq \alpha\}$ α -cut of function $\mathbf{u}$ is convex.

The point from $\mathbf{R^2}$ with membership function $\mu_{\tilde{P}}(P) = 1$ will be denoted by $\mathbf{P}$ ($\mathbf{P}$ is the core of the fuzzy point $\tilde{\mathbf{P}}$) and the membership function of the point $\tilde{\mathbf{P}}$ will be denoted by $\mu_{\tilde{P}}$. By $[P]^{\alpha}$ we denote α -cut of fuzzy point (set from $\mathbf{R^2}$).

Definition 2. $\mathbf{R^2}$ Linear fuzzy space is the set $\mathcal{H}^2 \subset \mathcal{F}^2$ of all functions which, in addition to the properties given in Definition 1, are:

Symmetric with the respect to the core $\mathbf{S \in R^2}$ ($\mu(\mathbf{S}) = \mathbf{1}$),

$\mu(\mathbf{V}) = \mu(\mathbf{M}) \wedge \mu(\mathbf{M}) \neq \mathbf{0} \Rightarrow \mathbf{d(S,V) = d(S,M)}$

where $\mathbf{d(S,M)}$ is the distance in $\mathbf{R^2}$.

Inverse-linear decreasing w.r.t. points' distance from the core according to:

If $\mathbf{r \neq 0}$

$$\mu(\mathbf{V}) = \max\left(0, 1 - \frac{\mathbf{d(S,V)}}{\mathbf{r}}\right)$$

if $\mathbf{r = 0}$

$$\mu(\mathbf{V}) = \begin{cases} \mathbf{1} & \text{if } \mathbf{S = V} \\ \mathbf{0} & \text{if } \mathbf{S \neq V} \end{cases}$$

where $\mathbf{d(S,V)}$ is the distance between the point $\mathbf{V}$ and the core $\mathbf{S}$ ($\mathbf{V,S \in R^n}$) and $\mathbf{r \in R^+}$ is constant.

Elements of that space are represented as ordered pairs $\tilde{\mathbf{S}} = \mathbf{(S, r_S)}$ where $\mathbf{S \in R^2}$ is the core of $\tilde{\mathbf{S}}$, and $\mathbf{r_S \in R^+ \cup \{0\}}$ is the distance from the core for which the function value becomes 0 in the sequel parameter $\mathbf{r_S}$ will be denoted as fuzzy support radius.

3. Process model

System can be represented as set of interconnected objects. Term process denotes a series of successive state transitions. In case that some of objects are represented with geospatial properties, process will be called geospatial process.

In this work, we use finite automaton, declarative and procedural representation of geospatial process. We also propose object oriented model for fuzzy finite automaton and procedural process representation.

For the purpose of the representation of the geospatial process as **finite automaton** we will use fuzzy extension of usual Finite Automaton (FA) definition in Definition 3. and fuzzy extension of Cellular Automaton (CA) in Definition 4. The main feature of this representation is that single object at the same time can be in more than one state with several membership degrees. In this case, process will be represented as series of membership degrees.

Definition 3. A fuzzy finite state automaton FFA is 7-tuple $(S, Q, I, X, \mu, \sigma, \eta)$ where:

(i) S is a non empty set (*state universal set*),

(ii) $Q = \{q_1, q_2, \ldots, q_n\}$ is a non empty finite set of fuzzy sets (*fuzzy states*) defined over set S ($\forall q \in Q) q: S \to [0,1]$),

(iii) I is a non empty set (*input universal set),*

(iv) $X = \{x_1, x_2, \ldots, x_m\}$ is a non empty finite set of fuzzy sets (*input alphabet*), ($\forall x \in X)\ x: I \to [0,1]$)

(v) $\mu: Q \times Q \times X \to [0,1]$ is a fuzzy transition function,

(vi) $\sigma: Q \to [0,1]$ is fuzzy set on Q (*initial state*)

(vii) $\eta: Q \to [0,1]$ is fuzzy set on Q (*final state*).

For $s \in S$ the value $q(s) \in [0,1]$ represents the degree to which automaton is in state q . The membership degrees $qs = Q(s) = \{q_1(s), q_2(s), \ldots q_n(s)\}$ represents fuzzy state of the object s .

For $q, p \in Q$ and $x \in X$ the value $\mu(q,p,x) \in [0,1]$ represents the degree to which the automaton in state q and with the input symbol x may enter to state p . For $q \in Q$, $\sigma(q)$ indicates the degree to which q is initial state and $\eta(q)$ indicates the degree to which q is final state.

Definition 4. Let $\hat{P}$ is a finite set of fuzzy finite state automaton (FFA) called fuzzy cells, $\lambda: \hat{P} \times \hat{P} \to [0,1]$ is a fuzzy relation between two FFA , $*: [0,1] \times [0,1]^n \to [0,1]^n$ and $P = (S_p, Q_p, X_p, \mu_p, \sigma_p, \eta_p) \in \hat{P}$.

Then pair $(\hat{P}, \lambda)$ is a fuzzy cellular automaton (FCA) if

$$\forall P \in \hat{P} \to X_p = \bigcup_{T \in \hat{P}, T \neq P} (\lambda(P,T) * Q_T)$$

For $a \in [0,1]$ and $(b_1, b_2, \ldots, b_n) \in [0,1]^n$ operation is defined as

$a * (b_1, b_2, \ldots, b_n) =_{DF} (T(a, b_1), T(a, b_2), \ldots, T(a, b_n))$

where T is *t-norm.*

The input set X_P of cell P is defined as union of related cell's fuzzy state multiplied by influence to cell P .

Figures 1. and 2. illustrate two state diagrams of a single FCA consist of two FFA A and B respectively. Transitions between states are described by fuzzy rules. Those fuzzy rules consist of logical compositions of current fuzzy state and fuzzy states of all other related objects.

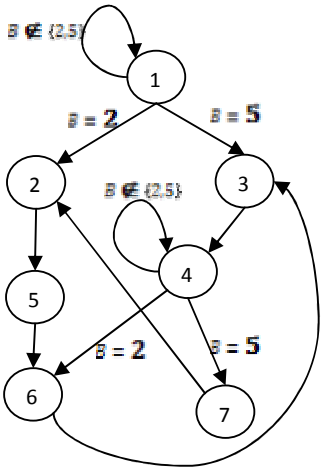


Figure 1. State diagram of Fuzzy Finite Automata A

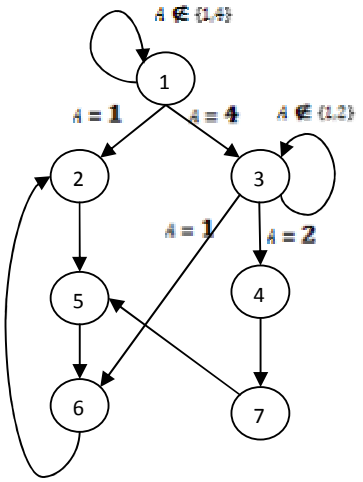


Figure 2. State diagram of Fuzzy Finite Automata B

The table 1 contains fuzzy rules which determine behavior of fuzzy cells $\mathbf{A}$ and $\mathbf{B}$.

	IF	*THEN*
k1	$A_i = 1 \ \& \ B_i = 2$	$A_{i+1} = 2$
k2	$A_i = 1 \ \& \ B_i = 5$	$A_{i+1} = 3$
k3	$A_i = 1 \& B_i \notin \{2,5\}$	$A_{i+1} = 1$
k4	$A_i = 2$	$A_{i+1} = 5$
k5	$A_i = 3$	$A_{i+1} = 4$
k6	$A_i = 4 \ \& \ B_i = 2$	$A_{i+1} = 6$
k7	$A_i = 4 \ \& \ B_i = 5$	$A_{i+1} = 7$
k8	$A_i = 4 \& B_i \notin \{2,5\}$	$A_{i+1} = 4$
k9	$A_i = 5$	$A_{i+1} = 6$
k10	$A_i = 6$	$A_{i+1} = 3$
k11	$A_i = 7$	$A_{i+1} = 2$
k12	$B_i = 1 \ \& \ A_i = 1$	$B_{i+1} = 2$
k13	$B_i = 1 \ \& \ A_i = 4$	$B_{i+1} = 3$
k14	$B_i = 1 \& A_i \notin \{1,4\}$	$B_{i+1} = 1$
k15	$B_i = 2$	$B_{i+1} = 5$
k16	$B_i = 3 \ \& \ A_i = 1$	$B_{i+1} = 6$
k17	$B_i = 3 \ \& \ A_i = 2$	$B_{i+1} = 4$
k18	$B_i = 3 \& A_i \notin \{1,2\}$	$B_{i+1} = 3$
k19	$B_i = 4$	$B_{i+1} = 7$
k20	$B_i = 5$	$B_{i+1} = 6$
k21	$B_i = 6$	$B_{i+1} = 2$
k22	$B_i = 7$	$B_{i+1} = 5$

Table 1. Set of fuzzy rules which determine behavior of fuzzy cells $\mathbf{A}$ and $\mathbf{B}$.

Series of state transitions for the initial fuzzy state $A_1 =$ "1" $= (1,0,0,0,0,0,0)$ and $B_1 =$ "5" $= (0,0,0,0,1,0,0)$ is shown in Table 2. The first column indicates the ordinal number of transition. The second and third column consist of seven sub columns each for every fuzzy state A and B respectively. In this example Fuzzy cellular automata acts like simple cellular automata.

	A							B						
i	1	2	3	4	5	6	7	1	2	3	4	5	6	7
1	1											1		
2		1												1
3			1						1					
4				1							1			
5		1												1
6			1						1					
7				1							1			
8		1												1
9			1						1					
10				1							1			

Table 2. Series of crisp state transitions of $\mathbf{A}$ and $\mathbf{B}$

However, a situation in which the initial states of objects are fuzzy instead of crisp can arise. One such example would be if the object A was in state 1 with degree 0.8, and in state 2 with degree 0.2, while the object B was in states 4, 5 and 6 with degrees 0.1, 0.8 and 0.1 respectively. Fuzzy state transitions for this example are shown in the Table 3.

	A							B						
i	1	2	3	4	5	6	7	1	2	3	4	5	6	7
1	**0.8**	0.2									0.1	**0.8**	0.1	
2	0.17		**0.67**		0.17				0.1				**0.8**	0.1
3	0.15	0.08		**0.62**		0.15			**0.89**			0.11		
4	0.11	0.11	0.11	0.11	0.06	**0.44**	0.06					**0.89**	0.11	
5	0.11	0.06	**0.44**	0.11	0.11	0.06	0.11		0.11				0.89	
6	0.13	0.13	0.06	**0.5**	0.06	0.13			**0.89**			0.11		
7	0.11	0.11	0.11	0.11	0.11	**0.42**	0.05					**0.89**	0.11	
8	0.11	0.05	**0.42**	0.11	0.11	0.11	0.11		0.11				**0.89**	
9	0.12	0.12	0.12	**0.47**	0.06	0.12			**0.89**			0.11		
10	0.11	0.11	0.11	0.11	0.11	**0.42**	0.05					**0.89**	0.11	

Table 3. Series of fuzzy state transitions of $\mathbf{A}$ and $\mathbf{B}$

The system behavior can be described by the **declarative** model as the set of fuzzy predicates. Unlike the previous case, using the declarative model, it is possible to describe state transitions with fuzzy predicates (complex fuzzy rules). Extension of the open source XProlog interpreter is described in [13]. That extension with certain modifications could be used for declarative process modelling. System behavior illustrated by the Figure 1 and Figure 2 then, could be described by the following program.

```
state(s1, [1, 0, 0, 0, 0, 0, 0]).
state(s2, [0, 1, 0, 0, 0, 0, 0]).
state(s3, [0, 0, 1, 0, 0, 0, 0]).
state(s4, [0, 0, 0, 1, 0, 0, 0]).
state(s5, [0, 0, 0, 0, 1, 0, 0]).
state(s6, [0, 0, 0, 0, 0, 1, 0]).
state(s7, [0, 0, 0, 0, 0, 0, 1]).

moveA(A, Anext, B):- state(s1, X1),
  state(s2, X2),
  A=X1, B=X2, Anext is X2.
moveA(A, Anext, B):-state(s1, X1),
  state(s3, X3),
  stanje(s5, X5)
  A=X1, B=X5, Anext is X3.
moveA(A, Anext, B):-stanje(s1, X1),
  state(s2, X2),
  state(s5, X5)
  A=X1, B!=X5, B!=X2, Anext is X1.
```

```
....
simulation(PA,PB,0,Rez,Rez).
simulation(PA,PB,T,Z,Rez):-
   moveA(PA,NA,PB),
   moveB(PB,NB,PA),
   T1 is T-1,
   Z1 is [(T,NA,NB)|Z],
   simulation(NA,NB,T1,Z1,Rez).

simulation(A,B,T,Rez):-
  simulation(A,B,T,[(T,A,B)],Rez).
```

In this example, statement "`A=X1`" represent degree of similarity between two fuzzy sets `A` and `X1`. Similarity is implemented as fuzzy relation. Its value corresponds to height of intersection of the fuzzy sets `A` and `X1`. Also the statement "`Anext is X2`" is true with degree of truth that corresponds to degree of truth of hypothesis (presumption fuzzy rules)

Simulation of the system described by **procedural representation** is simpler than simulation in previous cases, because it could be performed directly on the virtual machine. When modeling the process by using a procedural method, transition function could be built by fuzzy geometry functions such as translation, rotation, sharpening, blurring etc.

Object model of the process

Object approach and the UML language are selected as a framework for modeling geospatial process. Therefore, it is possible to use any of object oriented programming languages for implementation. Proposed object model integrate fuzzy finite automaton, fuzzy cellular automaton and procedural way of representation.

Class diagram of the object model is given by Figure 3.

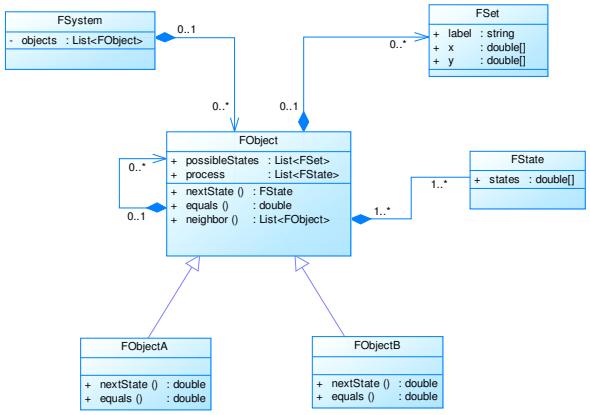


Figure 3. Class diagram of the object model for fuzzy process representation

Main classes used for process modeling are *FSystem*, *FObject* and *FState*. Class *FSystem* represents a model of the system. It contains a set of instances *FObject*. Class *FObject* contains definitions of the possible states, current and all previous state transitions. One of the main operators in the class *FObject* is similarity operator (==) between two FObjects. Transition rules are implemented in the method *nextState*. Classes *FObjectA* and *FObjectB* inherit *FObject* class and redefine method *nextState* by implementing concrete behavior. Simulation is then achieved by sequential execution of the *nextState* method, redefined in appropriate class. Simple example of simulation function is shown in following code:

```
int T = 10;
FState A0 = new FState(new double[]
                {0.8, 0.2, 0, 0, 0, 0, 0 });
FState B0 = new FState(new double[]
                {0, 0, 0, 0.1, 0.8, 0.1, 0});
FObject A = new FObjectA();
FObject B = new FObjectB();
A.neighbor.Add(B);
B.neighbor.Add(A);

A.process.Add(A0);
B.process.Add(B0);
for (int i = 0; i < T; i++)
{
  A.nextState();
  B.nextState();
 }
```

In the following section, we present software architecture of the extensible simulation system which is able to support previously described object model for process representations.

4. Software architecture of the extensible simulation system

Software architecture of the extensible simulation system is modeled by UML language. System consist of three subsystems: fuzzy space (data store), process simulation and subsystem for presentation. Figure 4. illustrate conceptual model of the proposed architecture.

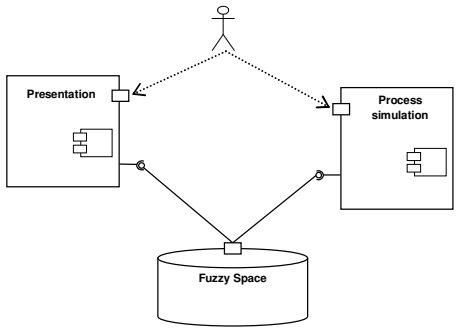
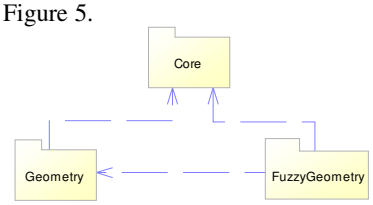


Figure 4. Conceptual model of the proposed software architecture

Fuzzy Space Data Store

Fuzzy Space Data Store subsystem consists of several independent geospatial data management systems. Those systems should store imprecise geospatial data on efficient way. Also, it should deliver those data to process

simulation framework and presentation framework through standard interface (JDBC, ODBC, WFS, GML etc.). Some of the main properties of those systems are scalability and the ability to extend the systems with additional data types and spatial functions. In this work, we choose PostgreSQL object-relational database management system because of its open source. It is developed as scalable and extensible system, with simple mechanism for adding new data types. PostGIS is spatial extension of PostgreSQL. It consist of basic spatial data types with more than 700 spatial functions. Extension of PostGis with support to imprecise point data types is described in [14]. PostgreSQL has two standard protocols, namely JDBC and ODBC, which are interfaces for data access. Using these standard protocols for data access all three subsystems could be implemented as independent systems. Also, all three subsystems could be deployed on different computer platforms and distributed on several computer systems. Support for imprecise data types is implemented as PostGis extension (see [14]), along with appropriate C# and Java class libraries. Those libraries could be used in procedural process representation for both C# and Java parts of simulation framework. Package diagram connections of the C# and Java class libraries are shown on Figure 5.



Slika 5. Package diagram connections

C# and Java class libraries are developed as .dll and .jar files respectively according to UML model given in previous section. Package *Core* contains classes *SpatialReferenceSystem* and *Matrix*. First class contains several geospatial referent system transformation functions. Class Matrix encapsulate matrix data type and all standard matrix operations. Class diagram of package *Geometry* is given in [15]. Package *FuzzyGeometry* contains the implementations of the main imprecise geometry data types.

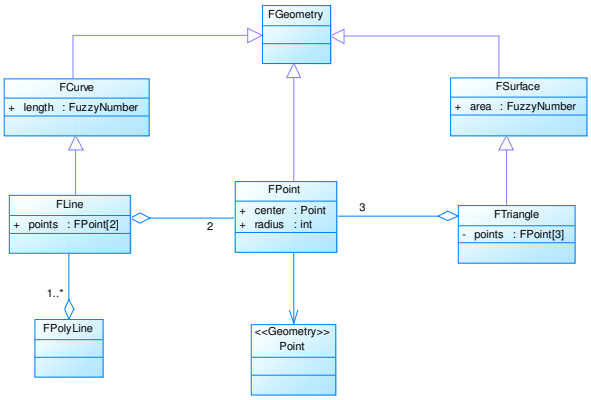


Figure 6. Class diagram of package *FuzzyGeometry*

Class diagram of the package *FuzzyGeometry* is shown on Figure 6. Main class is *FPoint* which is software implementation of mathematical model given in Definition 1.

Presentation framework

Presentation of geospatial data is very complex problem because it should integrate large amount of data from several data sources. Depending on the different context, data should be presented on different way. Real object usually have 3D shape. Presentation framework has to provide efficient user interface with several different visual styles (2D, 3D, more or less details). Also, it should be able to present series of state transitions.
In this work, we analyzed several software packages: UDIG, Google Earth, Web Open Layer and AutoCAD.

Description of the AutoCAD extension which is a part of the presentation framework follows. AutoCad is one of the world's leading 2D and 3D design applications known by almost every designer engineer around the world. That is reason why we choose AutoCAD as main part of presentation framework. It has system solution for extending in form of applications which could be added as separate .dll library written in several programming languages (Lisp, C++, C#, Visual Basic). Because of these properties AutoCAD grow in several directions AutoCAD Civil, AutoCAD Architecture, AutoCad Inventor etc. In this work, we chose C# because of its pure object oriented nature, and sophisticated development environment (Visual Studio 2009.+)
Using this approach it is possible to implement the functions or processes to perform simulation which requires intensive interaction between the process and operators. AutoCAD platform in this way become a part of the presentation subsystem (to present spatial data), simulation subsystem (it could execute simulations) and as part of a data store subsystem (it is possible to store a large amount of spatial data).

In the implementation of .dll AutoCad extension in C# we have used AutoCAD.Net API and ObjectDBX.Net. Figure 7. describes the hierarchy of main objects and relationships among the objects in the AutoCAD.Net API library.

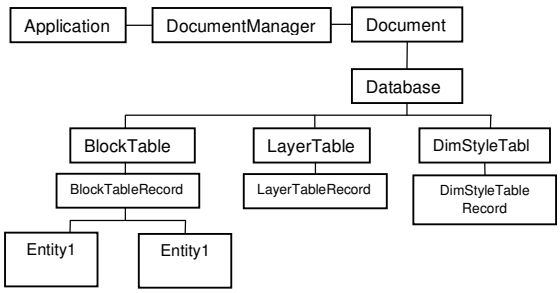


Figure 7. Connection model of the main objects in the AutoCAD.Net API library

AutoCAD.Net API and ObjectDBX.Net are distributed with core packages in the form of two dynamic link libraries acmgd.dll and acdbmgd.dll. Extension of

AutoCAD is implemented as dynamic link library in Visual Studio. It contains at least methods installComponent and uninstallComponent. Implemented library is loaded into AutoCAD by netload command from command window. Then installation starts by executing installComponent method which modify registry to allow AutoCAD to automatically load .dll at the next startup. This extension implementation can access data store directly by the ODBC standard protocol or indirectly by using geoserver data interface in form of WFS and other OpenGIS standards. Software architecture for transforming data from GML into standard object model is described in [15].

Process simulation framework

Subsystem for simulation and executing geospatial process consist of three parts: core, communication part and processing environment. The core contains implementation of object model for process representation described in previous section. The communication part should provide interface to data store and processing environment where all simulations are being executed. Framework that partially implements almost all functionalities of the communication part and processing part is implemented and described in [15],[16].

Simulation framework is implemented as extensible system. Extension could be done in two ways: adding new type of geospatial process and adding new model for process representation. Process simulation framework component is shown on Figure 8.

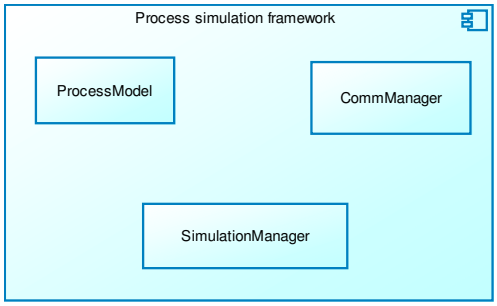


Figure 8. Process simulation framework component

Web based spatial data and application framework (see [16]) is Java J2EE implementation of the process simulation framework. CommManager is implemented using application described in [15].

5.Conclusion

In this work we have proposed software architecture of the imprecise geospatial process management. Also, we have proposed object model for geospatial process representation. By using the implemented object process model it is possible to represent process as fuzzy finite automaton (see Definition 3.), fuzzy cellular automaton (see Definition 4.) and procedural representation of geospatial process. We have used UML language for modeling and object oriented languages (Java and C#) for implementation of the simulation framework. AutoCAD extension has functionalities for presentation, simulation and data store. All three subsystems (presentation, simulation and data store) could be technologically independent.

Proposed software architecture could be extended in several ways. It should be extended by supporting process representation with the system of partial differential equations, extending fuzzy prolog to support process representation as set of fuzzy predicates in declarative way. Finally, it should be provided by utilities for distributed process executing in form of intelligent agents.

References

[1] P. Rigaux, M.O. Scholl, and A. Voisard, *Introduction to spatial databases: with application to GIS*, Morgan Kaufmann, 2002.

[2] L. Guibas, D. Salesin, and J. Stolfi, “Epsilon geometry: building robust algorithms from imprecise computations,” *Proceedings of the fifth annual symposium on Computational geometry*, New York, NY, USA: ACM, 1989, pp. 208–217.

[3] M. Löffler and M. Kreveld, “Largest and Smallest Convex Hulls for Imprecise Points,” *Algorithmica*, vol. 56, 2008, pp. 235-269.

[4] M. Löffler and J. Snoeyink, “Delaunay triangulation of imprecise points in linear time after preprocessing,” *Computational Geometry*, vol. 43, Apr. 2010, pp. 234-242.

[5] M. Burl, M. Weber, and P. Perona, “A probabilistic approach to object recognition using local photometry and global geometry,” *Lecture Notes in Computer Science*, vol. 1407, 1998, pp. 628-641.

[6] C.K. Cheung, W. Shi, and X. Zhou, “A Probability-based Uncertainty Model for Point-in-Polygon Analysis in GIS,” *GeoInformatica*, vol. 8, 2004, pp. 71-98.

[7] M. Schneider, “Design and implementation of finite resolution crisp and fuzzy spatial objects,” *Data & Knowledge Engineering*, vol. 44, Jan. 2003, pp. 81–108.

[8] J.J. Buckley and E. Eslami, “Fuzzy plane geometry I: Points and lines,” *Fuzzy Sets and Systems*, vol. 86, Mar. 1997, pp. 179-187.

[9] P. Diamond and P. Kloeden, “Metric spaces of fuzzy sets,” *Fuzzy Sets and Systems*, vol. 35, Apr. 1990, pp. 241-249.

[10] J. Gasós and A. Rosetti, “Uncertainty representation for mobile robots: perception, modeling and navigation in unknown environments,” *Fuzzy Sets and Systems*, vol. 107, Oct. 1999, pp. 1–24.

[11] Đ. Obradović, Z. Konjović, E. Pap, and N.M. Ralević, “The maximal distance between imprecise point objects,” *Fuzzy Sets and Systems*, vol. In Press, Corrected Proof.

[12] R. Obe and L. Hsu, *PostGIS in Action*, Manning Publications, 2011.

[13] M. Segedinac, Đ. Obradović, and Z. Konjović, "Proširenje interpretera XProlog fazi unifikacijom," *YuInfo 2009*, Kopaonik: 2009.
[14] D. Obradovic, Z. Konjovic, and E. Pap, "Extending PostGIS by imprecise point objects," *IEEE 8th International Symposium on Intelligent Systems and Informatics*, Subotica, Serbia: 2010, pp. 23-28.
[15] Đ. Obradović and M. Segedinac, "Softverska arhitektura za transformaciju geoprostornih podataka," *YuInfo 2009*, Kopaonik: 2009.
[16] Đ. Obradović, D. Jovanović, and M. Govedarica, "Web bazirano okruženje za rad sa prostornim podacima i aplikacijama," *YuINFO 2006*, Kopaonik: 2006.

Acknowledgments

Results presented in this paper are part of the research conducted within the Grant No. III-43007, Ministry of Science and Technological Development of the Republic of Serbia.