SPATIO-TEMPORAL DATABASE CONSTRAINTS FOR SPATIAL DYNAMIC SIMULATION

Bianca Maria Pedrosa, Luiz Camolesi Jr.,
Gilberto Câmara, Marina Teresa Pires Vieira

METHODIST UNIVERSITY OF PIRACICABA - UNIMEP
Rod. do Açúcar Km 156, Piracicaba, SP, Brazil
bpedrosa@unimep.br lcamoles@unimep.br mtvieira@unimep.br

NATIONAL INSTITUTE FOR SPACE RESEARCH - INPE
PBox 515, 12201 São José dos Campos, SP, Brazil
gilberto@dpi.inpe.br

Abstract: Spatial Dynamic Simulation Systems have three main components: the space dimension, the time dimension and the dynamic process. Dynamic processes have been modeled by transition rules, which are subject to constraints and which can avoid or force the occurrence of a transition. In this paper we propose a framework to represent the variability of simulation process based on the temporal constraints present in the Database Theory and using the Hybrid Cellular Automata represented by TerraML (cell-based modeling language). *Variability* is a feature to establish the possibility and change limits of objects at the moment or in the space. The variability, associated to simulation process, allows represent behavioral characteristics of elements in real world, particularly in Spatial Dynamic Systems.

Key words: Spatio-temporal modeling, dynamic simulation, constraint, time, XML

1. INTRODUCTION

Spatial Dynamic Modeling simulates spatio-temporal processes in which the state of a location, on the Earth's surface, changes over time, due to some

physical phenomena. In environmental simulation systems, space has been represented as cellular models, mostly in connection with cellular automata (CA) (White, 1997; Clarke and Gaydos, 1998; Couclelis, 1997). In such systems, time evolves in discrete steps and the change process converges in discrete transitions.

Transition rules encapsulate the mechanism upon which a discrete dynamic model evolves. In such systems a transition has a source and a target state, is triggered by an event, and can be associated to actions (Cerri and Fraternali, 1997). Events can be expressed by means of relational conditions, neighborhood configurations, or mathematical operations. Actions are performed in order to manipulate cell attributes or to invoke an operation or any other application-dependent action (Van Gurp and Bosch, 1999). Transitions are subjected to constraints, which are preconditions to limit, avoid, or force the occurrence of a transition in its spatial, temporal, or both contexts.

This paper introduces some spatio-temporal database constraints to a computational environment for dynamic simulation called TerraML, which is a cell based modeling language which supports systems with both discrete and continuous behavior, besides of some level of action-at-distance support (Pedrosa et al., 2002). The remainder of this paper is organized as follows. In section 2 we introduce TerraML, a dynamic modeling language. In section 3 the feature variability is presented to compose the simulation process and the semantic time model is used to represent the interval in simulation. In section 4 the TerraLib time data model is presented using parts of semantic time model to constraint modeling. In section 5 a modeling case is provided.

2. AN ENVIRONMENT FOR SIMULATION

TerraML is a cell-based modeling language to be used in environmental applications. TerraML supports both discrete and continuous change processes, supports different data formats and is quite integrated with general-use databases.

In TerraML, a cell-space is defined as a generalized raster data structure, where each cell holds more than one attribute value. If required, cells can be handled as individual geographic objects, and operations designed for objects (such as 9-intersection predicates) can be applied to them. The attributes can be presented to the user in the same way as vector geographic objects, and familiar visualization operations can be applied to these data sets (Câmara et al., 2000).

In order to capture the discrete and continuous components of the dynamic system, TerraML implements the dynamics of a Hybrid

Automaton. In Figure 1, a control graph illustrates the mechanism by which the Cellular Hybrid Automata (CHA) evolves. In this graph, the nodes represent control modes, and the conditions labeling the edges are known as jump conditions. The nodes of the graph contain flow conditions, which change the variable values. Flow conditions are executed until a jump condition is met (Henzinger, 1996). The CHA has two control modes, GLOBAL and LOCAL. The CHA switches between the GLOBAL and LOCAL control modes when a jump condition is reached. The automaton has an initial condition (`totaldemand > 0`). This trajectory is processed recursively for all simulation steps. At the GLOBAL mode the CHA processes some computations in order to calibrate the system. At the LOCAL mode the CHA processes some computations cell by cell.

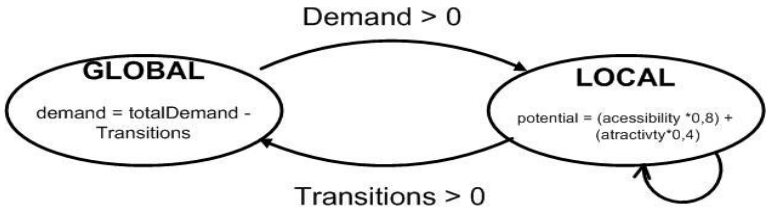

Figure 1. The hybrid Cellular Automata Dynamics

TerraML stands for TerraLib Modeling Language. TerraLib is an open-source general-purpose GIS library under development at the Brazilian National Institute for Space Research (INPE). TerraLib provides functionality for handling the different types of geographic data and facilities for data conversion, graphical output, and spatial database (Pedrosa et al., 2002).

A TerraML document is mapped to a TeCellAutomata object (Figure 2), provided by TerraLib, which has the following components:

- A neighbhood implemented as a (TeProxMatrix) flexible proximity matrix, which allows the user to define his own proximity matrix, according to application
- A cell space (TeCellSet)
- A set of transition rules (TransitionSet)
- A set of Control modes (ModeSet)
- An associated time Sequence

The Figure 3 presents the original TerraML Schema (Pedrosa et al., 2002), according to W3C (2003). In this schema TerraML is organized in sections. The **`cellprocessor`** is the main section, and it is divided into **`input`** and **`control`** sections. In the **`input`** section all data to be

retrieved must be specified. In the **`control`** section, the control modes and transitions are described. The control modes include equations such as Local mean, fuzzy Logic, product, etc.

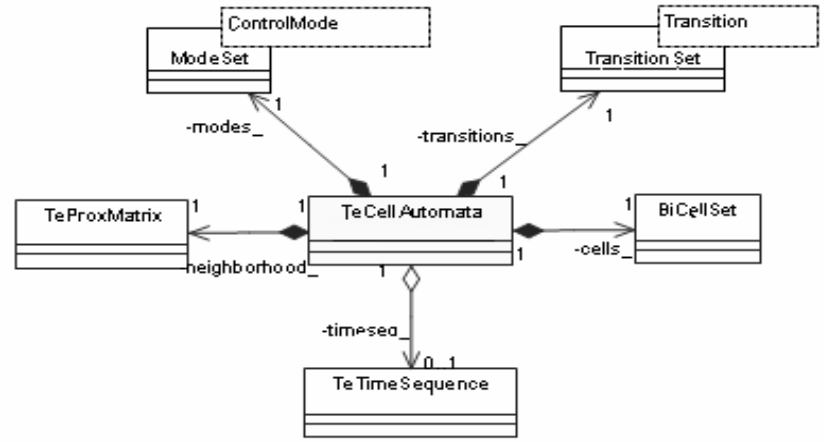

Figure 2. The TeCellAutomata Class Diagram

3. VARIABILITY IN SIMULATION PROCESS

In the simulations, transitions are processes representing evolution and therefore, subject to temporal variation. Therefore, it is important to associate to each spatio-temporal process a *variability constraint*. Variability refers to the conditions that limit, avoid, or force the occurrence of a change in a spatio-temporal process. For example, the evolution of a deforestation process can be limited to 10% over 20 years. On the other hand, variability can force the increase of a reforestation process to 50% of its current area. Invariant variability can also be used to express that no transitions may be applied to a state. For instance, we can specify the state "forest reserve" as a permanent state. This way it is not possible to change the state of a "forest reserve" cell, even though its neighborhood configuration or attributes satisfy a transition rule.

Variability is a feature to establish the possibility and change limits of objects at the moment or in the space (Camolesi, 2004). The variability is associated to object attributes or process attributes to model the structural, functional and behavioral characteristics of elements in real world. The *variability* degree has two enumeration values:

- *Invariant*: - defined to attributes that cannot be changed. Beside of infrequent, invariants are used to represent immutable or stable characteristics;
- *Variant*: - defined to attributes whose alterations are highly provable and therefore, demanding much cares on management and updating to support the natural evolution, involution or revolution of objects or processes.

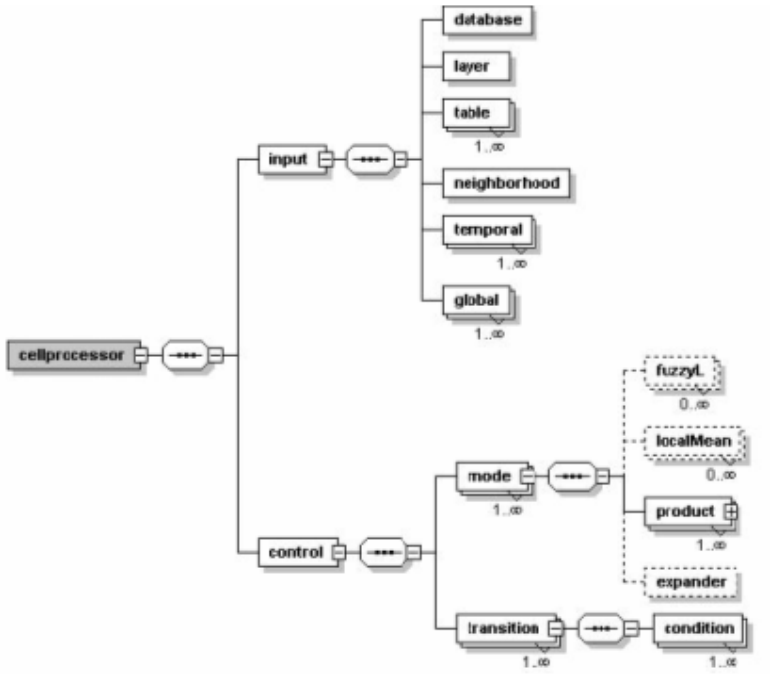

Figure 3. The TerraML Schema

The variability of dynamic objects is established by conditional expression of attributes to determine which objects or processes must change or must be evaluated. Variant's expression can be represented by set of predicates of type P_k (conditional sentence), composing the *Applicability Predicate Set (APS),* $P_1 \wedge P_2 \wedge \ldots \wedge P_k$, evaluating the moment (time) and the space (place) of changes.

3.1 Dimensioning the Limits: Time and Interval

The reliability of simulations based on temporal limits depends on the non-ambiguous definition of time. In an evaluation of the systems using the representation of time, the database designer can find a lot of variations and

ambiguous representation (Mok et al., 2002), what can degenerate the simulation.

Based on many researches about time representation and utilization were already done (Betini et al., 2000), leading to the following characteristics for homogeneous time definition:

- *Moment*: - a time instant value;
- *Granularity*: - precision domain of time instant. It can be based in the ISO pattern (2000) (e.g. PnYnMnDTnHnMnS) or any other pattern established by the application;
- *Orientation*: - reference system for temporal representation, for instance, Gregorian calendar (UTC or Coordinated Universal Time), Chinese calendar or other;
- *Direction*: - all orientation has an origin moment (0) and a time might be the moment before or after this origin moment, for instance, a cave picture of two thousand year old is before UTC origin moment. To represent these directions, a sign can be used to represent the after or before origin moment;
- *Application*: - specification of the use of the temporal representation, allowing the semantics recognition of the type, independently of the context in which is inserted.

This semantic representation for the datatype *time* allows the cost analysis reduction and the simplification of the specification, and the precise definition of time operators used in predicates of the variants.

Based on time, interval is a fundamental datatype to establish the temporal reference to transition. In this datatype (*struct interval*), the type *time* is used with the same semantics, however it is necessary aggregation of two time instants (*StartMoment* and *finalmoment*) intending the limitation to characterize the interval and, the definition of an attribute denominated *Composition_discret*, used to represents the *continuous* or *discrete* time.

```
Struct Interval {
        String              StartMoment;
        String              FinalMoment;
        Enumeration         Granularity;
        String              Orientation;
        Enumeration         Application;
        String              Direction;
        Enumeration         Composition_discret};
```

The limits of variant variability can be defined using the datatype *interval* with *application Duration,* i.e., specifying a moment size of duration for an action (either a past situation or a future one). The specification of *StartMoment* and/or *FinalMoment* defines a *Close Interval* or *Open Interval* of transition on simulation.

Considering the traditional time operator defined in several researches (Betini et al., 2000), in simulation the operators *in*, *before* and *after* are enough to comparison of relative position among a time moment and an (open or close) interval (Figure 4).

In the case of *continuous* interval, the attribute *Composition_discret* should not be used, but in *discrete* interval the granularity of the time moments should be indicated in the interval, allowing recognize the sub-interval contained to be used in sub-transitions. The granularity of the *Composition_discret* should be smaller than the interval granularity, for example, in a time interval of months, the discrete time in this interval should be, days, hours or other smaller granularity.

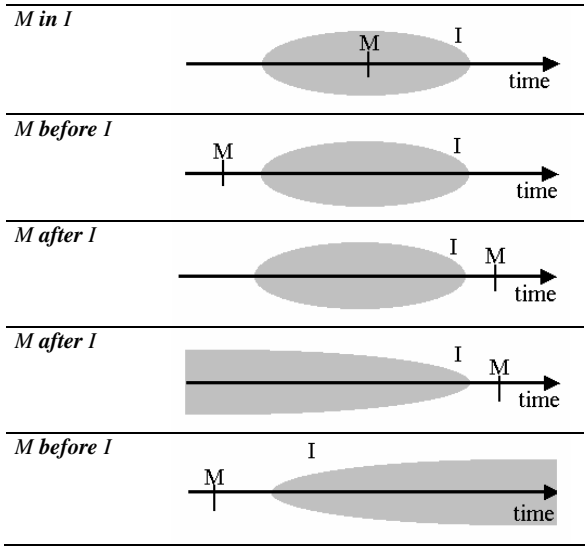

Figure 4. Operations time (M) with interval (I)

4. EXPRESSING VARIABILITY CONDITIONS

The TerraLib data model includes temporal concepts such as granularity (TeChronon*)*, instant and period of time, Figure 5. In TerraLib an interval (TeTimeSpan) is a period of time between two moments (TeTime). Any instant (TeTime) has a chronon, which is the time granularity. There is also a way to represent a sequence of timeSpans (TeTimeSequence).

The constraint DTD shown following specifies that a constraint can be *inv* or *var,* meaning invariant or variant, respectively. A *var* element has an *attribute*, *value*, *begintime* and *finaltime*, all of these attributes are strings (`CDATA`) and they have to be provided (`required`). An *inv* element has only *attribute* and *value*, but no time definition.

```
<!ELEMENT constraint (inv, var)>
<!ELEMENT var EMPTY>
<!ATTLIST var
   attribute CDATA #REQUIRED
   value     CDATA #REQUIRED
   begintime CDATA #REQUIRED
   finaltime CDATA #REQUIRED >
<!ELEMENT inv EMPTY>
<!ATTLIST inv
   attribute CDATA #REQUIRED
   value     CDATA #REQUIRED >
```

The orientation, direction and application characteristics, mentioned on section 3, are not part of the TerraLib temporal data model at the moment, but can be part of a further implementation. The variability feature is implemented as database constraints to be applied to TerraML.

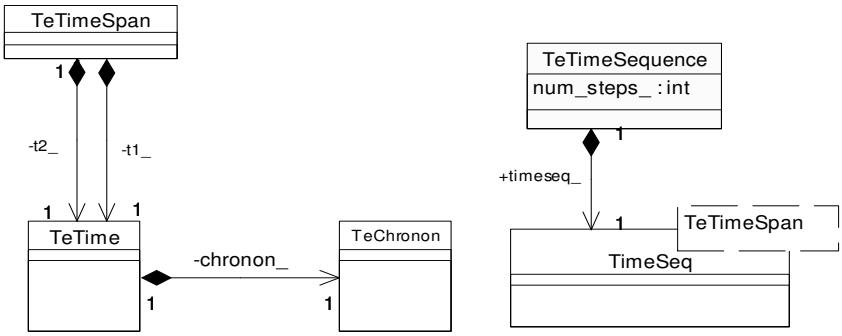

Figure 5. The TerraLib Time Model

5. A LAND-USE CHANGE APPLICATION

In order to illustrate the TerraML usage we present an example in land use change process. In this example, Figure 6, we specify in the `input` section that a relational database named `rondonia.mdb` and a table and layer named `cells450` must be opened.

In the `control` section it is defined that the simulation will take 16 `years` from 1985. At the `GLOBAL` section it is defined the temporal data demand with a different rate at different periods of time. At the `LOCAL` mode two constraints are declared. The first constraint (line 18) states that cells located at a forest reserve area are *invariant*, which means they never change. The second constraint (line 19) states that cells `owned` by the State (Federal) should not change between 1986 and 1990, which can be understand as a *variant* constraint. Another important detail about this simulation concerns to the demand. At the GLOBAL control mode we specify that the demand is *temporal,* which means it has a temporal constraint, which is delimited by its initial and final time (lines 12-14). It is important to note that there is a hidden constraint in that point; therefore every temporal data is an *invariant* constraint.

```
1  <cellprocessor author="bianca" date="11/06/04" model="Rondonia" >
2   <input>
3     <database  host="localhost"  path="c:/tese_dados/"    name="rondonia.mdb"  user=""
          pass=""/>
4     <layer name="celulas450"    layerid="46"/>
5     <table name="celulas450_dinamica"   columns="35"   lines="70"/>
6     <neighborhood name="c:/tese_dados/vizinho1.txt"   zones="1"/>
7     <global name="total_demand"       value="700"/>
8     <global name="demand"   value="0"/>
9    </input>
10  <control initime="1985" intervals="16" step="1" timeUnit="year">
11    <mode name="GLOBAL">
12       <temporal attribute="demand" value="70" inittime="1986" finaltime="1990" />
13      <temporal attribute="demand" value="50" inittime="1991"  finaltime="1995" />
14      <temporal attribute="demand" value="25" inittime = "1996" />
15    </mode>
16    <mode name="LOCAL">
17     <constraint
18      <inv attribute = "florest reserve" value = "1" />
19      <var attribute = "owner" value="Federal"  inittime="1986" finaltime="1990" />
20     </constraint >
21     <fuzzyL attribute="accessibility" column="road_distance"  alpha="0.001" beta="500"
          />
22     <localMean attribute="attractivity" column="land_cover"/>
23      <product attribute="potential">
24       <pair attribute="accessibility"  weight="0.8"/>
25       <pair attribute="attractivity"   weight="0.2"/>
26      </product>
27     <expander attribute="land_cover" column="potential"    demand="demand"/>
28    </mode>
29    <transition from="GLOBAL" to="LOCAL">
30      <condition attribute="demand" op="GT" value="0"/>
```

```
31      </transition>
32              </CONTROL>
33          </CELLPROCESSOR>
```

Figure 6. A TerraML Document

Figure 7 shows the impact of introducing constraints to the simulation. At first, Figure 7 presents the original land use of a small region in Rondonia, in 1985. Them, we can see the deforestation process (light gray) expanding over cells close to the road (doted line). In 1991, all cells around the road are deforested, except to the ones in the forest reserve area (black square), due to the invariant constraint. In this case, we can see that the forested cells (dark gray) are preserved even being so close to the roads and having many deforested neighbors. Another constraint introduced to the model is the one, which preserves the cells owned by the federal government. Note that the forest cells located on the left-down corner have their state changed only after 1990, when the *variant* constraint was over.

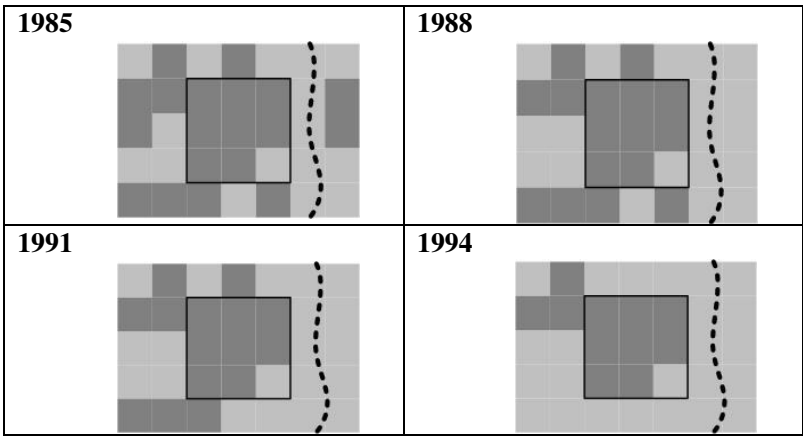

Figure 7. Detailed area of Rondonia simulation

6. CONCLUSION

In this paper we proposed to extend TerraML in order to support constraints over transitions rules. The constraints model proposed is based on semantic representation of variability to transitions in simulations. The model proposed support both variant and invariant conditions and seems to cover the most frequent situations in environment systems.

The development of TerraML and the open source GIS software library is part of an ongoing work. Future efforts will focus on extending constraints to support the orientation and direction aspects of time representation presented in semantic of section 3.1.

REFERENCES

Bettini, C., Jajodia, S. and Wang, X. S., 2000, *Time Granularities in Database, Data Mining and Temporal Reasoning*, Springer-Verlag.

Câmara, G., Souza, R.C.M., Pedrosa, B. M., Vinhas, L., Monteiro, A.M.V., Paiva, J.A., Carvalho, M.T. and Gatass, M., 2000, TerraLib: Technology in Support of GIS Innovation. II Brazilian Workshop of GeoInformatics, June, São Paulo(ed).

Camolesi Jr, L., 2004, Survivability and Applicability in Database Constraints: Temporal Boundary to Data Integrity Scenarios, 5th IEEE International Conference on Information Technology: Coding and Computing, 1: 518-522.

Cerri, S. and Fraternali, P., 1997, *Designing Database Applications with Objects and Rules. The IDEA Methodology*. Harlow, England: Addison Wesley Longman, 579 p.

Clarke, K. C. and Gaydos. L., 1998. Loose-Coupling a Cellular Automaton and GIS: Long Term urban growth prediction for San Francisco and Washington/Baltimore. *International Journal of Geographical Information Science* 12 (7): 699-714.

Couclelis, H., 1997. From Cellular Automata to Urban Models: New Principles for Model Development and Implementation. *Environment and Planning B: Planning and Design,* 24:165-174.

Henzinger, T. A., 1996, The Theory of Hybrid Automata, 11th Symposium on Logic in Computer Science (LICS), pp. 278-292.

Mok, A. K., Lee, C., and Woo, H., 2002, The Monitoring of Timing Constraints on Time Intervals, Proc. IEEE Real-Time Systems Symposium, p. 1-10.

Pedrosa, B. M., Câmara, G., Fonseca, F., Carneiro, T. and Souza, R. C. M., 2002, TerraML: a Language to support Spatial Dynamic Modeling, GeoInfo 2002 - IV Brazilian Symposium on GeoInformatics, Caxambu, December.

Van Gurp, J. and Bosch, J., 1999, On the Implementation of Finite State Machines, 3rd IASTED Intern. Conf. Software Engineering and Applications, Scottsdale, Arizona, USA.

White, R. and Engelen, G., 1997, Cellular Automata as the Basis of Integrated Dynamic Regional Modelling, *Environment and Planning B: Planning and Design*, vol. 24.

World Wide Web Consortium- W3C. http://www.w3c.org. (May 2003).

International Organization for Standardization- ISO, 8601: Data Elements and Interchange formats - Information Interchange - Representation of Dates and Times. Technical Committee ISO/TC 154, 2000.